Chapter 53
The PRINQUAL Procedure

Chapter Table of Contents

Chapter 53
The PRINQUAL Procedure

Overview

The PRINQUAL procedure obtains linear and nonlinear transformations of variables by using the method of alternating least squares to optimize properties of the transformed variables' covariance or correlation matrix. Nonoptimal transformations for logarithm, rank, exponentiation, inverse sine, and logit are also available with PROC PRINQUAL.

The PRINQUAL (principal components of qualitative data) procedure is a data transformation procedure that is based on the work of Kruskal and Shepard (1974); Young, Takane, and de Leeuw (1978); Young (1981); and Winsberg and Ramsay (1983). You can use PROC PRINQUAL to

- generalize ordinary principal component analysis to a method capable of analyzing data that are not quantitative
- perform metric and nonmetric multidimensional preference (MDPREF) analyses (Carroll 1972)
- preprocess data, transforming variables prior to their use in other data analyses
- summarize mixed quantitative and qualitative data and detect nonlinear relationships
- reduce the number of variables for subsequent use in regression analyses, cluster analyses, and other analyses

The PRINQUAL procedure provides three methods of transforming a set of qualitative and quantitative variables to optimize the transformed variables' covariance or correlation matrix. These methods are

- maximum total variance (MTV)
- minimum generalized variance (MGV)
- maximum average correlation (MAC)

All three methods attempt to find transformations that decrease the rank of the covariance matrix computed from the transformed variables. Transforming the variables to maximize the variance accounted for by a few linear combinations (using the MTV method) locates the observations in a space with dimensionality that approximates

the stated number of linear combinations as much as possible, given the transformation constraints. Transforming the variables to minimize their generalized variance or maximize the sum of correlations also reduces the dimensionality. The transformed qualitative (nominal and ordinal) variables can be thought of as quantified by the analysis, with the quantification done in the context set by the algorithm. The data are quantified so that the proportion of variance accounted for by a stated number of principal components is locally maximal, the generalized variance of the variables is locally minimal, or the average of the correlations is locally maximal.

The data can contain variables with nominal, ordinal, interval, and ratio scales of measurement (Siegel 1956). Any mix is allowed with all methods. PROC PRINQUAL can

- transform nominal variables by scoring the categories to optimize the covariance matrix (Fisher 1938)
- transform ordinal variables monotonically by scoring the ordered categories so that order is weakly preserved (adjacent categories can be merged) and the covariance matrix is optimized. You can untie ties optimally or leave them tied (Kruskal 1964). You can also transform ordinal variables to ranks.
- transform interval and ratio scale of measurement variables linearly, or transform them nonlinearly with spline transformations (de Boor 1978; van Rijckevorsel 1982) or monotone spline transformations (Winsberg and Ramsay 1983). In addition, nonoptimal transformations for logarithm, exponential, power, logit, and inverse trigonometric sine are available.
- for all transformations, estimate missing data without constraint, with category constraints (missing values within the same group get the same value), and with order constraints (missing value estimates in adjacent groups can be tied to preserve a specified ordering). Refer to Gifi (1990) and Young (1981).

The PROC PRINQUAL iterations produce a set of transformed variables. Each variable's new scoring satisfies a set of constraints based on the original scoring of the variable and the specified transformation type. First, all variables are required to satisfy transformation standardization constraints; that is, all variables have a fixed mean and variance. The other constraints include linear constraints, weak order constraints, category constraints, and smoothness constraints. The new set of scores is selected from the sets of possible scorings that do not violate the constraints so that the method criterion is locally optimized.

The displayed output from PROC PRINQUAL is a listing of the iteration history. However, the primary output from PROC PRINQUAL is an output data set. By default, the procedure creates an output data set that contains variables with _TYPE_='SCORE'. These observations contain original variables, transformed variables, components, or data approximations. If you specify the CORRELATIONS option in the PROC PRINQUAL statement, the data set also contains observations with _TYPE_='CORR'; these observations contain correlations or component structure information.

The Three Methods of Variable Transformation

The three methods of variable transformation provided by PROC PRINQUAL are discussed in the following sections.

The Maximum Total Variance (MTV) Method

The MTV method (Young, Takane, and de Leeuw 1978) is based on the principal component model, and it attempts to maximize the sum of the first r eigenvalues of the covariance matrix. This method transforms variables to be (in a least-squares sense) as similar to linear combinations of r principal component score variables as possible, where r can be much smaller than the number of variables. This maximizes the total variance of the first r components (the trace of the covariance matrix of the first r principal components). Refer to Kuhfeld, Sarle, and Young (1985).

On each iteration, the MTV algorithm alternates classical principal component analysis (Hotelling 1933) with optimal scaling (Young 1981). When all variables are ordinal preference ratings, this corresponds to Carroll's (1972) MDPREF analysis. You can request the dummy variable initialization method suggested by Tenenhaus and Vachette (1977), who independently proposed the same iterative algorithm for nominal and interval scale-of-measurement variables.

The Minimum Generalized Variance (MGV) Method

The MGV method (Sarle 1984) uses an iterated multiple regression algorithm in an attempt to minimize the determinant of the covariance matrix of the transformed variables. This method transforms each variable to be (in a least-squares sense) as similar to linear combinations of the remaining variables as possible. This locally minimizes the generalized variance of the transformed variables, the determinant of the covariance matrix, the volume of the parallelepiped defined by the transformed variables, and the sphericity (the extent to which a quadratic form in the optimized covariance matrix defines a sphere). Refer to Kuhfeld, Sarle, and Young (1985).

On each iteration for each variable, the MGV algorithm alternates multiple regression with optimal scaling. The multiple regression involves predicting the selected variable from all other variables. You can request a dummy variable initialization using a modification of the Tenenhaus and Vachette (1977) method that is appropriate with a regression algorithm. This method can be viewed as a way of investigating the nature of the linear and nonlinear dependencies in, and the rank of, a data matrix containing variables that can be nonlinearly transformed. This method tries to create a less-than-full rank data matrix. The matrix contains the transformation of each variable that is most similar to what the other transformed variables predict.

The Maximum Average Correlation (MAC) Method

The MAC method (de Leeuw 1985) uses an iterated constrained multiple regression algorithm in an attempt to maximize the average of the elements of the correlation matrix. This method transforms each variable to be (in a least-squares sense) as similar to the average of the remaining variables as possible.

On each iteration for each variable, the MAC algorithm alternates computing an equally weighted average of the other variables with optimal scaling. The MAC method is similar to the MGV method in that each variable is scaled to be as similar

to a linear combination of the other variables as possible, given the constraints on the transformation. However, optimal weights are not computed. You can use the MAC method when all variables are positively correlated or when no monotonicity constraints are placed on any transformations. Do not use this method with negatively correlated variables when some optimal transformations are constrained to be increasing because the signs of the correlations are not taken into account. The MAC method is useful as an initialization method for the MTV and MGV methods.

Getting Started

In the following example, PROC PRINQUAL uses the MTV method. Suppose that the problem is to linearize a curve through three-dimensional space. Let

$$\begin{aligned} \mathrm{X}_1 &= \mathrm{X}^3 \\ \mathrm{X}_2 &= \mathrm{X}_1 - \mathrm{X}^5 \\ \mathrm{X}_3 &= \mathrm{X}_2 - \mathrm{X}^6 \end{aligned}$$

where $\mathrm{X} = -1.00, -0.98, -0.96, \ldots, 1.00$.

These three variables define a curve in three-dimensional space. The GPLOT procedure is used to display two-dimensional views of this curve. These data are completely described by three linear components, but they define a single curve, which could be described as a single nonlinear component.

PROC PRINQUAL is used to attempt to straighten the curve into a one-dimensional line with a continuous transformation of each variable. The N=1 option in the PROC PRINQUAL statement requests one principal component. The TRANSFORM statement requests a cubic spline transformation with nine knots. *Splines* are curves, which are usually required to be continuous and smooth. Splines are usually defined as piecewise polynomials of degree n with function values and first $n - 1$ derivatives that agree at the points where they join. The abscissa values of the join points are called *knots*. The term "spline" is also used for polynomials (splines with no knots) and piecewise polynomials with more than one discontinuous derivative. Splines with no knots are generally smoother than splines with knots, which are generally smoother than splines with multiple discontinuous derivatives. Splines with few knots are generally smoother than splines with many knots; however, increasing the number of knots usually increases the fit of the spline function to the data. Knots give the curve freedom to bend to more closely follow the data. Refer to Smith (1979) for an excellent introduction to splines. For another example of using splines, see Example 65.1 in Chapter 65, "The TRANSREG Procedure."

One component accounts for 71 percent of the variance of the untransformed data, and after 50 iterations, over 98 percent of the variance of the transformed data is accounted for by one component (see Figure 53.2). The algorithm did not converge with 50 iterations, so more iterations may be needed for this problem.

PROC PRINQUAL creates an output data set (which is not displayed) that contains both the original and transformed variables. The original variables have the names X1, X2, and X3. Transformed variables are named TX1, TX2, and TX3. All observations in the output data set have _TYPE_='SCORE', since the CORRELATIONS option is not specified in the PROC PRINQUAL statement. The GPLOT procedure uses this output data set and displays the nonlinear transformations of all three variables and the nearly one-dimensional scatter plot (see Figure 53.3 and Figure 53.4).

PROC PRINQUAL tries to project each variable on the first principal component. Notice that the curve in this example is closer to a circle than to a function from some views (see the plot of X3 vs. X2 in Figure 53.1) and that the first component does not run approximately from one end point of the curve to the other (see Figure 53.4). Since the curve has these characteristics, PROC PRINQUAL linearizes the scatter plot by collapsing the scatter around the principal axis, not by straightening the curve into a single line. PROC PRINQUAL would straighten simpler curves.

The following statements produce Figure 53.1 through Figure 53.4:

```
* Generate a Three-Dimensional Curve;
data X;
   do X = -1 to 1 by 0.02;
      X1 =      X ** 3;
      X2 = X1 - X ** 5;
      X3 = X2 - X ** 6;
      output;
   end;
   drop X;
run;

goptions goutmode=replace nodisplay;
%let opts = haxis=axis2 vaxis=axis1 frame cframe=ligr;
* Depending on your goptions, these plot options may work better:
* %let opts = haxis=axis2 vaxis=axis1 frame;

proc gplot data=X;
   title;
   axis1 minor=none label=(angle=90 rotate=0)
         order=(-1 to 1);
   axis2 minor=none order=(-1 to 1);
   plot X1*X2 / &opts name='prqin1';
   plot X3*X2 / &opts name='prqin2' vreverse;
   plot X1*X3 / &opts name='prqin3';
   symbol1 color=blue;
run; quit;

goptions display;
proc greplay nofs tc=sashelp.templt template=l2r2;
   igout gseg;
   treplay 1:prqin1 2:prqin2 3:prqin3;
run; quit;
```

```
* Try to Straighten the Curve;
proc prinqual data=X n=1 maxiter=50 covariance;
   title 'Iteratively Derive Variable Transformations';
   transform spline(X1-X3 / nknots=9);
run;

* Plot the Transformations;
goptions nodisplay;
proc gplot;
   title;
   axis1 minor=none label=(angle=90 rotate=0);
   axis2 minor=none;
   plot TX1*X1 / &opts name='prqin4';
   plot TX2*X2 / &opts name='prqin5';
   plot TX3*X3 / &opts name='prqin6';
   symbol1 color=blue;
run; quit;

goptions display;
proc greplay nofs tc=sashelp.templt template=l2r2;
   igout gseg;
   treplay 1:prqin4 2:prqin6 3:prqin5;
run; quit;

* Plot the Straightened Scatter Plot;
goptions nodisplay;
proc gplot;
   axis1 minor=none label=(angle=90 rotate=0)
         order=(-1 to 1);
   axis2 minor=none order=(-1 to 1);
   plot TX1*TX2 / &opts name='prqin7';
   plot TX3*TX2 / &opts name='prqin8' vreverse;
   plot TX1*TX3 / &opts name='prqin9';
   symbol1 color=blue;
run; quit;

goptions display;
proc greplay nofs tc=sashelp.templt template=l2r2;
   igout gseg;
   treplay 1:prqin7 2:prqin8 3:prqin9;
run; quit;
```

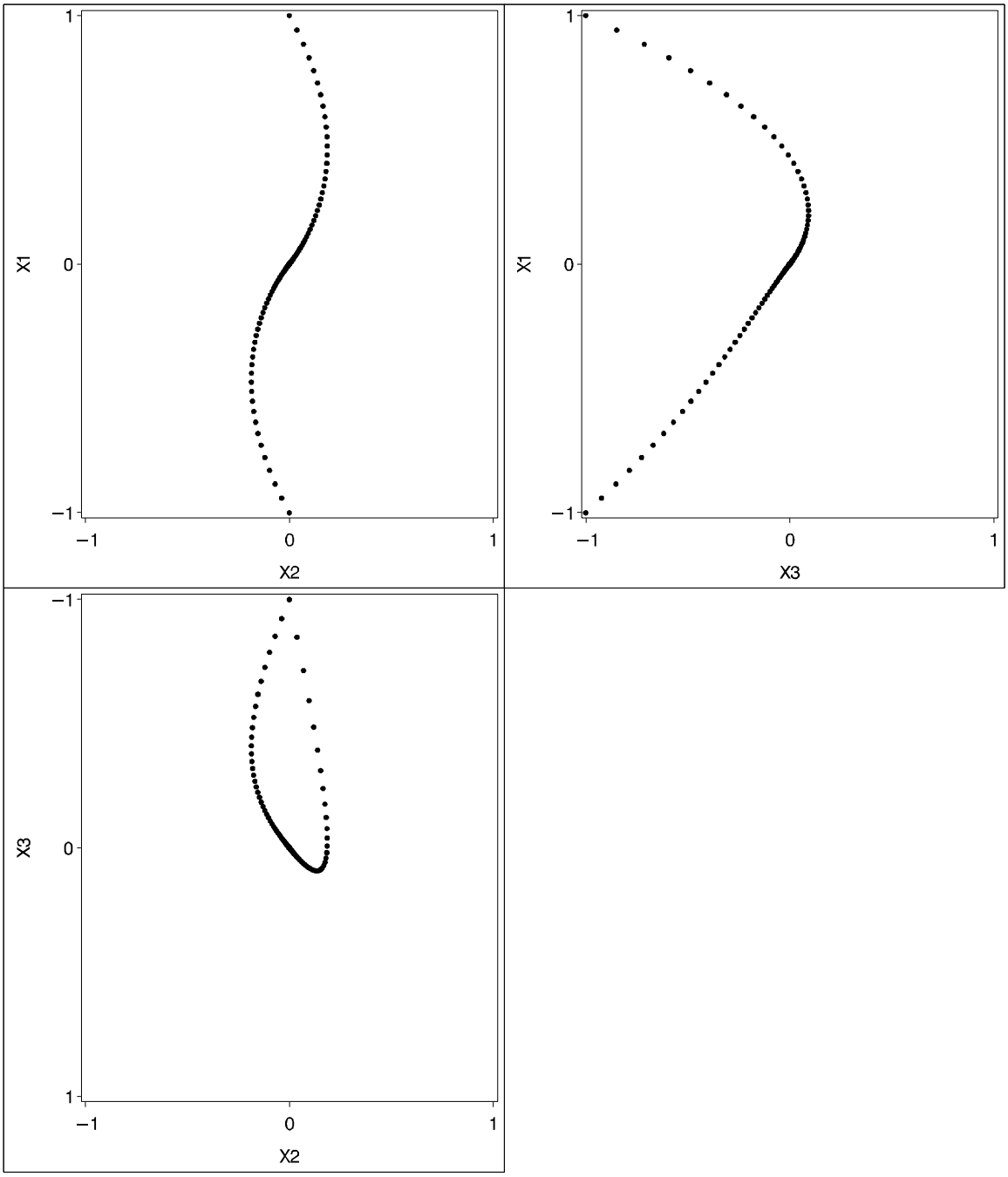


Figure 53.1. Three-Dimensional Curve Example Output

Iteratively Derive Variable Transformations

The PRINQUAL Procedure

PRINQUAL MTV Algorithm Iteration History

Iteration Number	Average Change	Maximum Change	Proportion of Variance	Criterion Change	Note
1	0.16253	1.33045	0.71369		
2	0.07871	0.94549	0.79035	0.07667	
3	0.06518	0.80219	0.86334	0.07299	
4	0.05322	0.57928	0.91379	0.05045	
5	0.04154	0.38404	0.94204	0.02825	
6	0.03181	0.24391	0.95640	0.01436	
7	0.02461	0.15397	0.96349	0.00709	
8	0.01982	0.10205	0.96704	0.00355	
9	0.01662	0.07393	0.96894	0.00189	
10	0.01439	0.06232	0.97005	0.00112	
11	0.01288	0.05436	0.97081	0.00075	
12	0.01189	0.04911	0.97139	0.00058	
13	0.01119	0.04531	0.97188	0.00049	
14	0.01068	0.04276	0.97232	0.00044	
15	0.01027	0.04115	0.97273	0.00041	
16	0.00993	0.04039	0.97313	0.00040	
17	0.00965	0.04249	0.97351	0.00038	
18	0.00940	0.04400	0.97388	0.00037	
19	0.00919	0.04509	0.97423	0.00036	
20	0.00900	0.04587	0.97458	0.00034	
21	0.00883	0.04643	0.97491	0.00033	
22	0.00867	0.04681	0.97523	0.00032	
23	0.00852	0.04705	0.97555	0.00031	
24	0.00839	0.04719	0.97585	0.00031	
25	0.00827	0.04724	0.97615	0.00030	
26	0.00816	0.04722	0.97644	0.00029	
27	0.00805	0.04713	0.97672	0.00028	
28	0.00795	0.04699	0.97700	0.00027	
29	0.00785	0.04680	0.97726	0.00027	
30	0.00776	0.04656	0.97752	0.00026	
31	0.00768	0.04629	0.97777	0.00025	
32	0.00760	0.04598	0.97802	0.00025	
33	0.00752	0.04564	0.97826	0.00024	
34	0.00745	0.04528	0.97849	0.00023	
35	0.00739	0.04489	0.97872	0.00023	
36	0.00733	0.04448	0.97894	0.00022	
37	0.00729	0.04405	0.97915	0.00022	
38	0.00724	0.04361	0.97936	0.00021	
39	0.00720	0.04315	0.97957	0.00021	
40	0.00716	0.04268	0.97977	0.00020	
41	0.00713	0.04219	0.97997	0.00020	
42	0.00709	0.04170	0.98016	0.00019	
43	0.00706	0.04120	0.98035	0.00019	
44	0.00703	0.04070	0.98054	0.00019	
45	0.00699	0.04019	0.98072	0.00018	
46	0.00696	0.03967	0.98090	0.00018	
47	0.00693	0.03916	0.98107	0.00017	
48	0.00690	0.03864	0.98124	0.00017	
49	0.00687	0.03812	0.98141	0.00017	
50	0.00684	0.03760	0.98158	0.00017	Not Converged

ERROR: Failed to converge.

Figure 53.2. PROC PRINQUAL MTV Iteration History

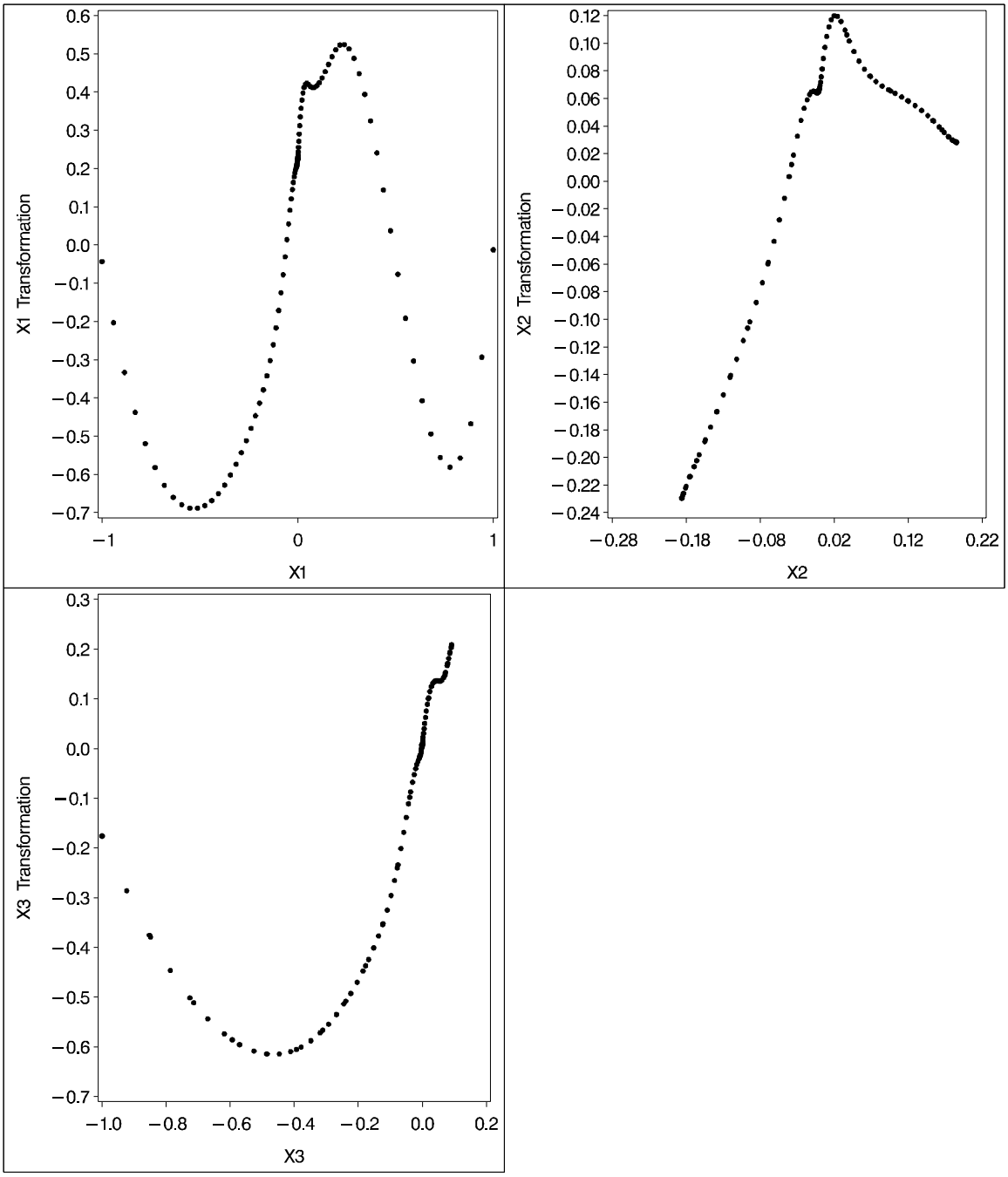


Figure 53.3. Variable Transformation Plots

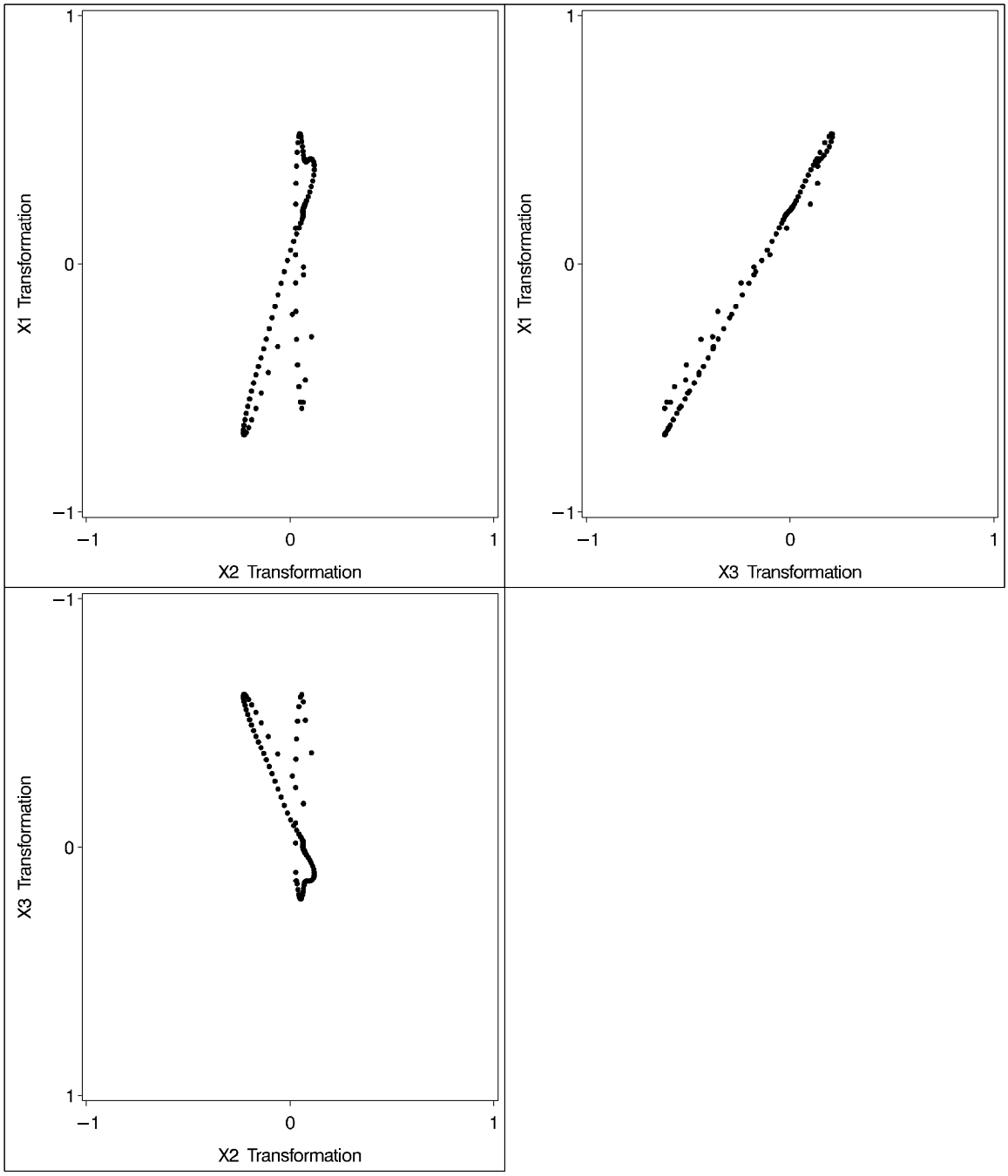


Figure 53.4. Plots of the Nearly One-Dimensional Curve

Syntax

The following statements are available in PROC PRINQUAL.

PROC PRINQUAL < *options* > **;**
TRANSFORM *transform(variables* < */ t-options* >*)*
< . . . *transform(variables* < */ t-options* >*)* > **;**
BY *variables* **;**
FREQ *variable* **;**
ID *variables* **;**
WEIGHT *variable* **;**

To use PROC PRINQUAL, you need the PROC PRINQUAL and TRANSFORM statements. You can abbreviate all *options* and *t-options* to their first three letters. This is a special feature of PROC PRINQUAL and is not generally true of other SAS/STAT procedures.

The rest of this section provides detailed syntax information for each of the preceding statements, beginning with the PROC PRINQUAL statement. The remaining statements are described in alphabetical order.

PROC PRINQUAL Statement

PROC PRINQUAL < *options* > **;**

The PROC PRINQUAL statement starts the PRINQUAL procedure. Optionally, this statement identifies an input data set, creates an output data set, specifies the algorithm and other computational details, and controls displayed output. The following table summarizes options available in the PROC PRINQUAL statement.

Task	Option
Identify input data set	
specifies input SAS data set	DATA=
Specify details for output data set	
outputs approximations to transformed variables	APPROXIMATIONS
specifies prefix for approximation variables	APREFIX=
outputs correlations and component structure matrix	CORRELATIONS
specifies a multidimensional preference analysis	MDPREF
specifies output data set	OUT=
specifies prefix for principal component scores variables	PREFIX=
replaces raw data with transformed data	REPLACE
outputs principal component scores	SCORES
standardizes principal component scores	STANDARD
specifies transformation standardization	TSTANDARD=
specifies prefix for transformed variables	TPREFIX=
Control iterative algorithm	
analyzes covariances	COVARIANCE
initializes using dummy variables	DUMMY
specifies iterative algorithm	METHOD=
specifies number of principal components	N=
suppresses numerical error checking	NOCHECK
specifies number of MGV models before refreshing	REFRESH=
restarts iterations	REITERATE
specifies singularity criterion	SINGULAR=
specifies input observation type	TYPE=
Control the number of iterations	
specifies minimum criterion change	CCONVERGE=
specifies number of first iteration to be displayed	CHANGE=
specifies minimum data change	CONVERGE=
specifies number of MAC initialization iterations	INITITER=
specifies maximum number of iterations	MAXITER=
Specify details for handling missing values	
includes monotone special missing values	MONOTONE=
excludes observations with missing values	NOMISS
unties special missing values	UNTIE=
Suppress displayed output	
suppresses displayed output	NOPRINT

The following list describes these options in alphabetical order.

APREFIX=*name*
APR=*name*
specifies a prefix for naming the approximation variables. By default, APREFIX=A. Specifying the APREFIX= option also implies the APPROXIMATIONS option.

APPROXIMATIONS
APPROX
APP

includes principal component approximations to the transformed variables (Eckart and Young 1936) in the output data set. Variable names are constructed from the value of the APREFIX= option and the input variable names. If you specify the APREFIX= option, then approximations are automatically included. If you specify the APPROXIMATIONS option and not the APREFIX= option, then the APPROXIMATIONS option uses the default, APREFIX=A, to construct the variable names.

CCONVERGE=*n*
CCO=*n*

specifies the minimum change in the criterion being optimized that is required to continue iterating. By default, CCONVERGE=0.0. The CCONVERGE= option is ignored for METHOD=MAC. For the MGV method, specify CCONVERGE=-2 to ensure data convergence.

CHANGE=*n*
CHA=*n*

specifies the number of the first iteration to be displayed in the iteration history table. The default is CHANGE=1. When you specify a larger value for n, the first $n - 1$ iterations are not displayed, thus speeding up the analysis. The CHANGE= option is most useful with the MGV method, which is much slower than the other methods.

CONVERGE=*n*
CON=*n*

specifies the minimum average absolute change in standardized variable scores that is required to continue iterating. By default, CONVERGE=0.00001. Average change is computed over only those variables that can be transformed by the iterations, that is, all LINEAR, OPSCORE, MONOTONE, UNTIE, SPLINE, MSPLINE, and SSPLINE variables and nonoptimal transformation variables with missing values. For more information, see the section "Optimal Transformations" on page 2792.

COVARIANCE
COV

computes the principal components from the covariance matrix. The variables are always centered to mean zero. If you do not specify the COVARIANCE option, the variables are also standardized to variance one, which means the analysis is based on the correlation matrix.

CORRELATIONS
COR

includes correlations and the component structure matrix in the output data set. By default, this information is not included.

DATA=*SAS-data-set*

specifies the SAS data set to be analyzed. The data set must be an ordinary SAS data set; it cannot be a TYPE=CORR or TYPE=COV data set. If you omit the DATA= option, the PRINQUAL procedure uses the most recently created SAS data set.

DUMMY
DUM

expands variables specified for OPSCORE optimal transformations to dummy variables for the initialization (Tenenhaus and Vachette 1977). By default, the initial values of OPSCORE variables are the actual data values. The dummy variable nominal initialization requires considerable time and memory, so it might not be possible to use the DUMMY option with large data sets. No separate report of the initialization is produced. Initialization results are incorporated into the first iteration displayed in the iteration history table. For details, see the section "Optimal Transformations" on page 2792.

INITITER=*n*
INI=*n*

specifies the number of MAC iterations required to initialize the data before starting MTV or MGV iterations. By default, INITITER=0. The INITITER= option is ignored if METHOD=MAC.

MAXITER=*n*
MAX=*n*

specifies the maximum number of iterations. By default, MAXITER=30.

MDPREF
MDP

specifies a multidimensional preference analysis by implying the STANDARD, SCORES, and CORRELATIONS options. This option also suppresses warnings when there are more variables than observations.

METHOD=MAC | MGV | MTV
MET=MAC | MGV | MTV

specifies the optimization method. By default, METHOD=MTV. Values of the METHOD= option are MTV for maximum total variance, MGV for minimum generalized variance, or MAC for maximum average correlation. You can use the MAC method when all variables are positively correlated or when no monotonicity constraints are placed on any transformations. See the section "The Three Methods of Variable Transformation" on page 2773.

MONOTONE=*two-letters*
MON=*two-letters*

specifies the first and last special missing value in the list of those special missing values to be estimated using within-variable order and category constraints. By default, there are no order constraints on missing value estimates. The *two-letters* value must consist of two letters in alphabetical order. For example, MONOTONE=DF means that the estimate of .D must be less than or equal to the estimate of .E, which must be less than or equal to the estimate of .F; no order constraints are placed on estimates of ._, .A through .C, and .G through .Z. For details, see the "Missing Values" section on page 2797, and "Optimal Scaling" in Chapter 65, "The TRANSREG Procedure."

N=*n*

specifies the number of principal components to be computed. By default, N=2.

NOCHECK
NOC

turns off computationally intensive numerical error checking for the MGV method. If you do not specify the NOCHECK option, the procedure computes R^2 from the squared length of the predicted values vector and compares this value to the R^2 computed from the error sum of squares that is a by-product of the sweep algorithm (Goodnight 1978). If the two values of R^2 differ by more than the square root of the value of the SINGULAR= option, a warning is displayed, the value of the REFRESH= option is halved, and the model is refit after refreshing. Specifying the NOCHECK option slightly speeds up the algorithm. Note that other less computationally intensive error checking is always performed.

NOMISS
NOM

excludes all observations with missing values from the analysis, but does not exclude them from the OUT= data set. If you omit the NOMISS option, PROC PRINQUAL simultaneously computes the optimal transformations of the nonmissing values and estimates the missing values that minimize squared error.

Casewise deletion of observations with missing values occurs when you specify the NOMISS option, when there are missing values in IDENTITY variables, when there are weights less than or equal to 0, or when there are frequencies less than 1. Excluded observations are output with a blank value for the _TYPE_ variable, and they have a weight of 0. They do not contribute to the analysis but are scored and transformed as *supplementary* or passive observations. See the "Passive Observations" section on page 2803 and the "Missing Values" section on page 2797 for more information on excluded observations and missing data.

NOPRINT
NOP

suppresses the display of all output. Note that this option temporarily disables the Output Delivery System (ODS). For more information, see Chapter 15, "Using the Output Delivery System."

OUT=*SAS-data-set*

specifies an output SAS data set that contains results of the analysis. If you omit the OUT= option, PROC PRINQUAL still creates an output data set and names it using the DATAn convention. If you want to create a permanent SAS data set, you must specify a two-level name. (Refer to the discussion in *SAS Language Reference: Concepts*.) You can use the REPLACE, APPROXIMATIONS, SCORES, and CORRELATIONS options to control what information is included in the output data set. For details, see the "Output Data Set" section on page 2799.

PREFIX=*name*
PRE=*name*

specifies a prefix for naming the principal components. By default, PREFIX=Prin. As a result, the principal component default names are Prin1, Prin2,. . ., Prin$n$.

REFRESH=*n*
REF=*n*

specifies the number of variables to scale in the MGV method before computing a new inverse. By default, REFRESH=5. PROC PRINQUAL uses the REFRESH= option in the sweep algorithm of the MGV method. Large values for the REFRESH= option make the method run faster but with increased error. Small values make the method run more slowly and with more numerical accuracy.

REITERATE
REI

enables the PRINQUAL procedure to use previous transformations as starting points. The REITERATE option affects only variables that are iteratively transformed (specified as LINEAR, SPLINE, MSPLINE, SSPLINE, UNTIE, OPSCORE, and MONOTONE). For iterative transformations, the REITERATE option requests a search in the input data set for a variable that consists of the value of the TPREFIX= option followed by the original variable name. If such a variable is found, it is used to provide the initial values for the first iteration. The final transformation is a member of the transformation family defined by the original variable, not the transformation family defined by the initialization variable. See the "REITERATE Option Usage" section on page 2802.

REPLACE
REP

replaces the original data with the transformed data in the output data set. The names of the transformed variables in the output data set correspond to the names of the original variables in the input data set. If you do not specify the REPLACE option, both original variables and transformed variables (with names constructed from the TPREFIX= option and the original variable names) are included in the output data set.

SCORES
SCO

includes principal component scores in the output data set. By default, scores are not included.

SINGULAR=*n*
SIN=*n*

specifies the largest value within rounding error of zero. By default, SINGULAR=1E−8. The PRINQUAL procedure uses the value of the SINGULAR= option for checking $(1 - R^2)$ when constructing full rank matrices of predictor variables, checking denominators before dividing, and so on.

STANDARD
STD

standardizes the principal component scores in the output data set to mean zero and variance one instead of the default mean zero and variance equal to the corresponding eigenvalue. See the SCORES option.

TPREFIX=*name*
TPR=*name*

specifies a prefix for naming the transformed variables. By default, TPREFIX=T. The TPREFIX= option is ignored if you specify the REPLACE option.

TSTANDARD=CENTER | NOMISS | ORIGINAL | Z
TST=CEN | NOM | ORI | Z

specifies the standardization of the transformed variables in the OUT= data set. By default, TSTANDARD=ORIGINAL. When the TSTANDARD= option is specified in the PROC statement, it specifies the default standardization for all variables. When you specify TSTANDARD= as a *t-option*, it overrides the default standardization just for selected variables.

CENTER centers the output variables to mean zero, but the variances are the same as the variances of the input variables.

NOMISS sets the means and variances of the transformed variables in the OUT= data set, computed over all output values that correspond to nonmissing values in the input data set, to the means and variances computed from the nonmissing observations of the original variables. The TSTANDARD=NOMISS specification is useful with missing data. When a variable is linearly transformed, the final variable contains the original nonmissing values and the missing value estimates. In other words, the nonmissing values are unchanged. If your data have no missing values, TSTANDARD=NOMISS and TSTANDARD=ORIGINAL produce the same results.

ORIGINAL sets the means and variances of the transformed variables to the means and variances of the original variables. This is the default.

Z standardizes the variables to mean zero, variance one.

For nonoptimal variable transformations, the means and variances of the original variables are actually the means and variances of the nonlinearly transformed variables, unless you specify the ORIGINAL nonoptimal *t-option* in the TRANSFORM statement. For example, if a variable X with no missing values is specified as LOG, then, by default, the final transformation of X is simply LOG(X), not LOG(X) standardized to the mean of X and variance of X.

TYPE='*text***'|***name*
TYP='*text***'|***name*

specifies the valid value for the _TYPE_ variable in the input data set. If PROC PRINQUAL finds an input _TYPE_ variable, it uses only observations with a _TYPE_ value that matches the TYPE= value. This enables a PROC PRINQUAL OUT= data set containing correlations to be used as input to PROC PRINQUAL without requiring a WHERE statement to exclude the correlations. If a _TYPE_ variable is not in the data set, all observations are used. The default is TYPE='SCORE', so if you do not specify the TYPE= option, only observations with _TYPE_ = 'SCORE' are used.

PROC PRINQUAL displays a note when it reads observations with blank values of _TYPE_, but it does not automatically exclude those observations. Data sets created by the TRANSREG and PRINQUAL procedures have blank _TYPE_ values for those observations that were excluded from the analysis due to nonpositive weights, nonpositive frequencies, or missing data. When these observations are read again, they are excluded for the same reason that they were excluded from their original analysis, not because their _TYPE_ value is blank.

UNTIE=*two-letters*
UNT=*two-letters*
specifies the first and last special missing value in the list of those special missing values that are to be estimated with within-variable order constraints but no category constraints. The *two-letters* value must consist of two letters in alphabetical order. By default, there are category constraints but no order constraints on special missing value estimates. For details, see the "Missing Values" section on page 2797. Also, see "Optimal Scaling" in Chapter 65, "The TRANSREG Procedure."

BY Statement

BY *variables* **;**

You can specify a BY statement with PROC PRINQUAL to obtain separate analyses on observations in groups defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data using the SORT procedure with a similar BY statement.
- Specify the BY statement options NOTSORTED or DESCENDING in the BY statement for the PRINQUAL procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables using the DATASETS procedure.

For more information on the BY statement, refer to the discussion in *SAS Language Reference: Concepts*. For more information on the DATASETS procedure, refer to the discussion in the *SAS Procedures Guide.*

FREQ Statement

FREQ *variable* **;**

If one variable in the input data set represents the frequency of occurrence for other values in the observation, list the variable's name in a FREQ statement. PROC PRINQUAL then treats the data set as if each observation appeared n times, where n is the value of the FREQ variable for the observation. Noninteger values of the FREQ variable are truncated to the largest integer less than the FREQ value. The observation is used in the analysis only if the value of the FREQ statement variable is greater than or equal to 1.

ID Statement

ID *variables* **;**

The ID statement includes additional character or numeric variables in the output data set. The variables must be contained in the input data set.

TRANSFORM Statement

TRANSFORM *transform(variables* < */ t-options* >*)*
< . . . *transform(variables* < */ t-options* >*)* > **;**

The TRANSFORM statement lists the variables to be analyzed (*variables*) and specifies the transformation (*transform*) to apply to each variable listed. You must specify a transformation for each variable list in the TRANSFORM statement. The variables are variables in the data set. The *t-options* are transformation options that provide details for the transformation; these depend on the *transform* chosen. The *t-options* are listed after a slash in the parentheses that enclose the variables.

For example, the following statements find a quadratic polynomial transformation of all variables in the data set:

```
proc prinqual;
   transform spline(_all_ / degree=2);
run;
```

Or, if N1 through N10 are nominal variables and M1 through M10 are ordinal variables, you can use the following statements.

```
proc prinqual;
   transform opscore(N1-N10) monotone(M1-M10);
run;
```

The following sections describe the transformations available (specified with *transform*) and the options available for some of the transformations (specified with *t-options*).

Families of Transformations

There are three types of transformation families: nonoptimal, optimal, and other. Each family is summarized as follows.

Nonoptimal transformations — preprocess the specified variables, replacing each one with a single new nonoptimal, nonlinear transformation.

Optimal transformations — replace the specified variables with new, iteratively derived optimal transformation variables that fit the specified model better than the original variable (except for contrived cases where the transformation fits the model exactly as well as the original variable).

Other transformations — are the IDENTITY and SSPLINE transformations. These do not fit into either of the preceding categories.

The following table summarizes the transformations in each family.

Family	Members of Family
Nonoptimal transformations	
inverse trigonometric sine	ARSIN
exponential	EXP
logarithm	LOG
logit	LOGIT
raises variables to specified power	POWER
transforms to ranks	RANK
Optimal transformations	
linear	LINEAR
monotonic, ties preserved	MONOTONE
monotonic B-spline	MSPLINE
optimal scoring	OPSCORE
B-spline	SPLINE
monotonic, ties not preserved	UNTIE
Other transformations	
identity, no transformation	IDENTITY
iterative smoothing spline	SSPLINE

The *transform* is followed by a variable (or list of variables) enclosed in parentheses. Optionally, depending on the *transform*, the parentheses can also contain *t-options*, which follow the variables and a slash. For example,

```
transform log(X Y);
```

computes the LOG transformation of X and Y. A more complex example is

```
transform spline(Y / nknots=2) log(X1 X2 X3);
```

The preceding statement uses the SPLINE transformation of the variable Y and the LOG transformation of the variables X1, X2, and X3. In addition, it uses the NKNOTS= option with the SPLINE transformation and specifies two knots.

The rest of this section provides syntax details for members of the three families of transformations. The *t-options* are discussed in the section "Transformation Options (t-options)" on page 2793.

Nonoptimal Transformations

Nonoptimal transformations are computed before the iterative algorithm begins. Nonoptimal transformations create a single new transformed variable that replaces the original variable. The new variable is not transformed by the subsequent iterative algorithms (except for a possible linear transformation and missing value estimation).

The following list provides syntax and details for nonoptimal variable transformations.

ARSIN
ARS

finds an inverse trigonometric sine transformation. Variables following ARSIN must be numeric, in the interval $(-1.0 \leq \mathsf{X} \leq 1.0)$, and they are typically continuous.

EXP

exponentiates variables (the variable X is transformed to a^X). To specify the value of a, use the PARAMETER= *t-option*. By default, a is the mathematical constant $e = 2.718\ldots$. Variables following EXP must be numeric, and they are typically continuous.

LOG

transforms variables to logarithms (the variable X is transformed to $\log_a(X)$). To specify the base of the logarithm, use the PARAMETER= *t-option*. The default is a natural logarithm with base $e = 2.718\ldots$. Variables following LOG must be numeric and positive, and they are typically continuous.

LOGIT

finds a logit transformation on the variables. The logit of X is $\log(X/(1-X))$. Unlike other transformations, LOGIT does not have a three-letter abbreviation. Variables following LOGIT must be numeric, in the interval $(0.0 < \mathsf{X} < 1.0)$, and they are typically continuous.

POWER
POW

raises variables to a specified power (the variable X is transformed to X^a). You must specify the power parameter a by specifying the PARAMETER= *t-option* following the variables:

```
power(variable / parameter=number)
```

You can use POWER for squaring variables (PARAMETER=2), reciprocal transformations (PARAMETER=-1), square roots (PARAMETER=0.5), and so on. Variables following POWER must be numeric, and they are typically continuous.

RANK
RAN

transforms variables to ranks. Ranks are averaged within ties. The smallest input value is assigned the smallest rank. Variables following RANK must be numeric.

Optimal Transformations

Optimal transformations are iteratively derived. Missing values for these types of variables can be optimally estimated (see the "Missing Values" section on page 2797).

The following list provides syntax and details for optimal transformations.

LINEAR
LIN

finds an optimal linear transformation of each variable. For variables with no missing values, the transformed variable is the same as the original variable. For variables with missing values, the transformed nonmissing values have a different scale and origin than the original values. Variables following LINEAR must be numeric.

MONOTONE
MON

finds a monotonic transformation of each variable, with the restriction that ties are preserved. The Kruskal (1964) secondary least-squares monotonic transformation is used. This transformation weakly preserves order and category membership (ties). Variables following MONOTONE must be numeric, and they are typically discrete.

MSPLINE
MSP

finds a monotonically increasing B-spline transformation with monotonic coefficients (de Boor 1978; de Leeuw 1986) of each variable. You can specify the DEGREE=, KNOTS=, NKNOTS=, and EVENLY *t-options* with MSPLINE. By default, PROC PRINQUAL uses a quadratic spline. Variables following MSPLINE must be numeric, and they are typically continuous.

OPSCORE
OPS

finds an optimal scoring of each variable. The OPSCORE transformation assigns scores to each class (level) of the variable. Fisher's (1938) optimal scoring method is used. Variables following OPSCORE can be either character or numeric; numeric variables should be discrete.

SPLINE
SPL

finds a B-spline transformation (de Boor 1978) of each variable. By default, PROC PRINQUAL uses a cubic polynomial transformation. You can specify the DEGREE=, KNOTS=, NKNOTS=, and EVENLY *t-options* with SPLINE. Variables following SPLINE must be numeric, and they are typically continuous.

UNTIE
UNT

finds a monotonic transformation of each variable without the restriction that ties are preserved. The PRINQUAL procedure uses the Kruskal (1964) primary least-squares monotonic transformation method. This transformation weakly preserves order but not category membership (it may untie some previously tied values). Variables following UNTIE must be numeric, and they are typically discrete.

Other Transformations

IDENTITY
IDE

specifies variables that are not changed by the iterations. The IDENTITY transformation is used for variables when no transformation and no missing data estimation are desired. However, the REFLECT, ADDITIVE, TSTANDARD=Z, and TSTANDARD=CENTER options can linearly transform all variables, including IDENTITY variables, after the iterations. Observations with missing values in IDENTITY variables are excluded from the analysis, and no optimal scores are computed for missing values in IDENTITY variables. Variables following IDENTITY must be numeric.

SSPLINE
SSP

finds an iterative smoothing spline transformation of each variable. The SSPLINE transformation does not generally minimize squared error. You can specify the smoothing parameter with either the SM= *t-option* or the PARAMETER= *t-option*. The default smoothing parameter is SM=0. Variables following SSPLINE must be numeric, and they are typically continuous.

Transformation Options (t-options)

If you use a nonoptimal, optimal or other transformation, you can use *t-options*, which specify additional details of the transformation. The *t-options* are specified within the parentheses that enclose variables and are listed after a slash. For example,

```
proc prinqual;
   transform spline(X Y / nknots=3);
run;
```

The preceding statements find an optimal variable transformation (SPLINE) of the variables X and Y and use a *t-option* to specify the number of knots (NKNOTS=). The following is a more complex example.

```
proc prinqual;
   transform spline(Y / nknots=3) spline(X1 X2 / nknots=6);
run;
```

These statements use the SPLINE transformation for all three variables and use *t-options* as well; the NKNOTS= option specifies the number of knots for the spline.

The following sections discuss the *t-options* available for nonoptimal, optimal, and other transformations.

The following table summarizes the *t-options*.

Table 53.1. t-options Available in the TRANSFORM Statement

Task	Option
Nonoptimal transformation t-options	
uses original mean and variance	ORIGINAL
Parameter t-options	
specifies miscellaneous parameters	PARAMETER=
specifies smoothing parameter	SM=
Spline t-options	
specifies the degree of the spline	DEGREE=
spaces the knots evenly	EVENLY
specifies the interior knots or break points	KNOTS=
creates n knots	NKNOTS=
Other t-options	
renames variables	NAME=
reflects the variable around the mean	REFLECT
specifies transformation standardization	TSTANDARD=

Nonoptimal Transformation t-options

ORIGINAL
ORI

matches the variable's final mean and variance to the mean and variance of the original variable. By default, the mean and variance are based on the transformed values. The ORIGINAL *t-option* is available for all of the nonoptimal transformations.

Parameter t-options

PARAMETER=*number*
PAR=*number*

specifies the transformation parameter. The PARAMETER= *t-option* is available for the EXP, LOG, POWER, SMOOTH, and SSPLINE transformations. For EXP, the parameter is the value to be exponentiated; for LOG, the parameter is the base value; and for POWER, the parameter is the power. For SMOOTH and SSPLINE, the parameter is the raw smoothing parameter. (You can specify a SAS/GRAPH-style smoothing

parameter with the SM= *t-option*.) The default for the PARAMETER= *t-option* for the LOG and EXP transformations is $e = 2.718\ldots$. The default parameter for SSPLINE is computed from SM=0. For the POWER transformation, you must specify the PARAMETER= *t-option*; there is no default.

SM=*n*

specifies a SAS/GRAPH-style smoothing parameter in the range 0 to 100. You can specify the SM= *t-option* only with the SSPLINE transformation. The smoothness of the function increases as the value of the smoothing parameter increases. By default, SM=0.

Spline t-options

The following *t-options* are available with the SPLINE and MSPLINE optimal transformations.

DEGREE=*n*
DEG=*n*

specifies the degree of the B-spline transformation. The degree must be a nonnegative integer. The defaults are DEGREE=3 for SPLINE variables and DEGREE=2 for MSPLINE variables.

The polynomial degree should be a small integer, usually 0, 1, 2, or 3. Larger values are rarely useful. If you have any doubt as to what degree to specify, use the default.

EVENLY
EVE

is used with the NKNOTS= *t-option* to space the knots evenly. The differences between adjacent knots are constant. If you specify NKNOTS=k, k knots are created at

$$\text{minimum} + i((\text{maximum} - \text{minimum})/(k+1))$$

for $i = 1, \ldots, k$. For example, if you specify

```
spline(X / knots=2 evenly)
```

and the variable X has a minimum of 4 and a maximum of 10, then the two interior knots are 6 and 8. Without the EVENLY *t-option*, the NKNOTS= *t-option* places knots at percentiles, so the knots are not evenly spaced.

KNOTS=*number-list* **|** *n* **TO** *m* **BY** *p*
KNO=*number-list* **|** *n* **TO** *m* **BY** *p*

specifies the interior knots or break points. By default, there are no knots. The first time you specify a value in the knot list, it indicates a discontinuity in the nth (from DEGREE=n) derivative of the transformation function at the value of the knot. The second mention of a value indicates a discontinuity in the $(n-1)$th derivative of the transformation function at the value of the knot. Knots can be repeated any number of times for decreasing smoothness at the break points, but the values in the knot list can never decrease.

You cannot use the KNOTS= *t-option* with the NKNOTS= *t-option*. You should keep the number of knots small (see the section "Specifying the Number of Knots" on page 3431 in Chapter 65, "The TRANSREG Procedure").

NKNOTS=*n*
NKN=*n*

creates n knots, the first at the $100/(n+1)$ percentile, the second at the $200/(n+1)$ percentile, and so on. Knots are always placed at data values; there is no interpolation. For example, if NKNOTS=3, knots are placed at the twenty-fifth percentile, the median, and the seventy-fifth percentile. By default, NKNOTS=0. The NKNOTS= *t-option* must be ≥ 0.

You cannot use the NKNOTS= *t-option* with the KNOTS= *t-option*. You should keep the number of knots small (see the section "Specifying the Number of Knots" on page 3431 in Chapter 65, "The TRANSREG Procedure").

Other t-options

The following *t-options* are available for all transformations.

NAME=*(variable-list)*
NAM=*(variable-list)*

renames variables as they are used in the TRANSFORM statement. This option allows a variable to be used more than once. For example, if the variable X is a character variable, then the following step stores both the original character variable X and a numeric variable XC that contains category numbers in the output data set.

```
proc prinqual data=A n=1 out=B;
   transform linear(Y) opscore(X / name=(XC));
   id X;
run;
```

REFLECT
REF

reflects the transformation

$$y = -(y - \bar{y}) + \bar{y}$$

after the iterations are completed and before the final standardization and results calculations.

TSTANDARD=CENTER | NOMISS | ORIGINAL | Z
TST=CEN | NOM | ORI | Z

specifies the standardization of the transformed variables in the OUT= data set. By default, TSTANDARD=ORIGINAL. When the TSTANDARD= option is specified in the PROC PRINQUAL statement, it specifies the default standardization for all variables. When you specify TSTANDARD= as a *t-option*, it overrides the default standardization just for selected variables.

WEIGHT Statement

WEIGHT *variable* **;**

When you use a WEIGHT statement, a weighted residual sum of squares is minimized. The WEIGHT statement has no effect on degrees of freedom or number of observations, but the weights affect most other calculations. The observation is used in the analysis only if the value of the WEIGHT statement variable is greater than 0.

Details

Missing Values

PROC PRINQUAL can estimate missing values, subject to optional constraints, so that the covariance matrix is optimized. The procedure provides several approaches for handling missing data. When you specify the NOMISS option in the PROC PRINQUAL statement, observations with missing values are excluded from the analysis. Otherwise, missing data are estimated, using variable means as initial estimates. Missing values for OPSCORE character variables are treated the same as any other category during the initialization. See the section "Missing Values" on page 3418 in Chapter 65, "The TRANSREG Procedure," for more information on missing data estimation.

Controlling the Number of Iterations

Several options in the PROC PRINQUAL statement control the number of iterations performed. Iteration terminates when any one of the following conditions is satisfied:

- The number of iterations equals the value of the MAXITER= option.
- The average absolute change in variable scores from one iteration to the next is less than the value of the CONVERGE= option.
- The criterion change is less than the value of the CCONVERGE= option.

With the MTV method, the change in the proportion of variance criterion can become negative when the data have converged so that it is numerically impossible, within machine precision, to increase the criterion. Because the MTV algorithm is convergent, a negative criterion change is the result of very small amounts of rounding error. The MGV method displays the average squared multiple correlation (which is not the criterion being optimized), so the criterion change can become negative well before convergence. The MAC method criterion (average correlation) is never computed, so the CCONVERGE= option is ignored for METHOD=MAC. You can specify a negative value for either convergence option if you want to define convergence only in terms of the other convergence option.

With the MGV method, iterations minimize the generalized variance (determinant), but the generalized variance is not reported for two reasons. First, in most data sets,

the generalized variance is almost always near zero (or will be after one or two iterations), which is its minimum. This does not mean that iteration is complete; it simply means that at least one multiple correlation is at or near one. The algorithm continues minimizing the determinant in $(m-1), (m-2)$ dimensions, and so on. Because the generalized variance is almost always near zero, it does not provide a good indication of how the iterations are progressing. The mean R^2 provides a better indication of convergence. The second reason for not reporting the generalized variance is that almost no additional time is required to compute R^2 values for each step. This is because the error sum of squares is a by-product of the algorithm at each step. Computing the determinant at the end of each iteration adds more computations to an already computationally intensive algorithm.

You can increase the number of iterations to ensure convergence by increasing the value of the MAXITER= option and decreasing the value of the CONVERGE= option. Because the average absolute change in standardized variable scores seldom decreases below 1E−11, you typically do not specify a value for the CONVERGE= option less than 1E−8 or 1E−10. Most of the data changes occur during the first few iterations, but the data can still change after 50 or even 100 iterations. You can try different combinations of values for the CONVERGE= and MAXITER= options to ensure convergence without extreme overiteration. If the data do not converge with the default specifications, specify the REITERATE option, or try CONVERGE=1E−8 and MAXITER=50, or CONVERGE=1E−10 and MAXITER=200.

Performing a Principal Component Analysis of Transformed Data

PROC PRINQUAL produces an iteration history table that displays (for each iteration) the iteration number, the maximum and average absolute change in standardized variable scores computed over the iteratively transformed variables, the criterion being optimized, and the criterion change. In order to examine the results of the analysis in more detail, you can analyze the information in the output data set using other SAS procedures.

Specifically, use the PRINCOMP procedure to perform a components analysis on the transformed data. PROC PRINCOMP accepts the raw data from PROC PRINQUAL but issues a warning because the PROC PRINQUAL output data set has _NAME_ and _TYPE_ variables, but it is not a TYPE=CORR data set. You can ignore this warning.

If the output data set contains both scores and correlations, you must subset it for analysis with PROC PRINCOMP. Otherwise, the correlation observations are treated as ordinary observations and the PROC PRINCOMP results are incorrect. For example, consider the following statements:

```
proc prinqual data=a out=b correlations replace;
   transform spline(var1-var50 / nknots=3);
run;

proc princomp data=b;
   where _TYPE_='SCORE';
run;
```

Also note that the proportion of variance accounted for, as reported by PROC PRINCOMP, can exceed the proportion of variance accounted for in the last PROC PRINQUAL iteration. This is because PROC PRINQUAL reports the variance accounted for by the components analysis that generated the current scaling of the data, not a components analysis of the current scaling of the data.

Using the MAC Method

You can use the MAC algorithm alone by specifying METHOD=MAC, or you can use it as an initialization algorithm for METHOD=MTV and METHOD=MGV analyses by specifying the iteration option INITITER=. If any variables are negatively correlated, do not use the MAC algorithm with monotonic transformations (MONOTONE, UNTIE, and MSPLINE) because the signs of the correlations among the variables are not used when computing variable approximations. If an approximation is negatively correlated with the original variable, monotone constraints would make the optimally scaled variable a constant, which is not allowed (see the section "Avoiding Constant Transformations" on page 2802). When used with other transformations, the MAC algorithm can reverse the scoring of the variables. So, for example, if variable X is designated LOG(X) with METHOD=MAC and TSTANDARD=ORIGINAL, the final transformation (for example, TX) may not be LOG(X). If TX is not LOG(X), it has the same mean as LOG(X) and the same variance as LOG(X), and it is perfectly negatively correlated with LOG(X). PROC PRINQUAL displays a note for every variable that is reversed in this manner.

You can use the METHOD=MAC algorithm to reverse the scorings of some rating variables before a factor analysis. The correlations among bipolar ratings such as 'like - dislike', 'hot - cold', and 'fragile - monumental' are typically both positive and negative. If some items are reversed to say 'dislike - like', 'cold - hot', and 'monumental - fragile', some of the negative signs can be eliminated, and the factor pattern matrix would be cleaner. You can use PROC PRINQUAL with METHOD=MAC and LINEAR transformations to reverse some items, maximizing the average of the intercorrelations.

Output Data Set

The PRINQUAL procedure produces an output data set by default. By specifying the OUT=, APPROXIMATIONS, SCORES, REPLACE, and CORRELATIONS options in the PROC PRINQUAL statement, you can name this data set and control, to some extent, the contents of it.

Structure and Content

The output data set can have 16 different forms, depending on the specified combinations of the REPLACE, SCORES, APPROXIMATIONS, and CORRELATIONS options. You can specify any combination of these options. To illustrate, assume that the data matrix consists of N observations and m variables, and n components are computed. Then, define the following:

D the $N \times m$ matrix of original data with variable names that correspond to the names of the variables in the input data set. However, when you use the OP-

SCORE transformation on character variables, those variables are replaced by numeric variables that contain category numbers

T the $N \times m$ matrix of transformed data with variable names constructed from the value of the TPREFIX= option (if you do not specify the REPLACE option) and the names of the variables in the input data set

S the $N \times n$ matrix of component scores with variable names constructed from the value of the PREFIX= option and integers

A the $N \times m$ matrix of data approximations with variable names constructed from the value of the APREFIX= option and the names of the variables in the input data set

$\mathbf{R_{TD}}$ the $m \times m$ matrix of correlations between the transformed variables and the original variables with variable names that correspond to the names of the variables in the input data set. When missing values exist, casewise deletion is used to compute the correlations.

$\mathbf{R_{TT}}$ the $m \times m$ matrix of correlations among the transformed variables with the variable names constructed from the value of the TPREFIX= option (if you do not specify the REPLACE option) and the names of the variables in the input data set

$\mathbf{R_{TS}}$ the $m \times n$ matrix of correlations between the transformed variables and the principal component scores (component structure matrix) with variable names constructed from the value of the PREFIX= option and integers

$\mathbf{R_{TA}}$ the $m \times m$ matrix of correlations between the transformed variables and the variable approximations with variable names constructed from the value of the APREFIX= option and the names of the variables in the input data set

To create a data set WORK.A that contains all information, specify the following options in the PROC PRINQUAL statement

```
proc prinqual scores approximations correlations out=a;
```

and also use a TRANSFORM statement appropriate for your data. Then the WORK.A data set contains

D T S A

$\mathbf{R_{TD}}$ $\mathbf{R_{TT}}$ $\mathbf{R_{TS}}$ $\mathbf{R_{TA}}$

To eliminate the bottom partitions that contain the correlations and component structure, do not specify the CORRELATIONS option. For example, use the following PROC PRINQUAL statement with an appropriate TRANSFORM statement.

```
proc prinqual scores approximations out=a;
```

Then the WORK.A data set contains

D T S A

If you use the following PROC PRINQUAL statement (with an appropriate TRANSFORM statement)

```
proc prinqual out=a;
```

this creates a data set WORK.A of the form

```
D T
```

To output transformed data and component scores only, specify the following options in the PROC PRINQUAL statement:

```
proc prinqual replace scores out=a;
```

Then the WORK.A data set contains

```
T S
```

TYPE *and* _NAME_ *Variables*

In addition to the preceding information, the output data set contains two character variables, the variable _TYPE_ (length 8) and the variable _NAME_ (length 32).

The _TYPE_ variable has the value 'SCORE' if the observation contains variables, transformed variables, components, or data approximations; the _TYPE_ variable has the value 'CORR' if the observation contains correlations or component structure.

By default, the _NAME_ variable has values 'ROW1', 'ROW2', and so on, for the observations with _TYPE_='SCORE'. If you use an ID statement, the variable _NAME_ contains the formatted ID variable for SCORES observations. The values of the variable _NAME_ for observations with _TYPE_='CORR' are the names of the transformed variables.

Certain procedures, such as PROC PRINCOMP, which can use the PROC PRINQUAL output data set, issue a warning that the PROC PRINQUAL data set contains _NAME_ and _TYPE_ variables but is not a TYPE=CORR data set. You can ignore this warning.

Variable Names

The TPREFIX=, APREFIX=, and PREFIX= options specify prefixes for the transformed and approximation variable names and for principal component score variables, respectively. PROC PRINQUAL constructs transformed and approximation variable names from a prefix and the first characters of the original variable name. The number of characters in the prefix plus the number of characters in the original variable name (including the final digits, if any) required to uniquely designate the new variables should not exceed 32. For example, if the APREFIX= parameter that you specify is one character, PROC PRINQUAL adds the first 31 characters of the original variable name; if your prefix is four characters, only the first 28 characters of the original variable name are added.

Effect of the TSTANDARD= and COVARIANCE Options

The values in the output data set are affected by the TSTANDARD= and COVARIANCE options. If you specify TSTANDARD=NOMISS, the NOMISS standardization is performed on the transformed data after the iterations have been completed, but before the output data set is created. The new means and variances are used in creating the output data set. Then, if you do not specify the COVARIANCE option, the data are transformed to mean zero and variance one. The principal component scores and data approximations are computed from the resulting matrix. The data are then linearly transformed to have the mean and variance specified by the TSTANDARD= option. The data approximations are transformed so that the means within each pair of a transformed variable and its approximation are the same. The ratio of the variance of a variable approximation to the variance of the corresponding transformed variable equals the proportion of the variance of the variable that is accounted for by the components model.

If you specify the COVARIANCE option and do not specify TSTANDARD=Z, you can input the transformed data to PROC PRINCOMP, again specifying the COVARIANCE option, to perform a components analysis of the results of PROC PRINQUAL. Similarly, if you do not specify the COVARIANCE option with PROC PRINQUAL and you input the transformed data to PROC PRINCOMP without the COVARIANCE option, you receive the same report. However, some combinations of PROC PRINQUAL options, such as COVARIANCE and TSTANDARD=Z, while valid, produce approximations and scores that cannot be reproduced by PROC PRINCOMP.

The component scores in the output data set are computed from the correlations among the transformed variables, or from the covariances if you specified the COVARIANCE option. The component scores are computed after the TSTANDARD=NOMISS transformation, if specified. The means of the component scores in the output data set are always zero. The variances equal the corresponding eigenvalues, unless you specify the STANDARD option; then the variances are set to one.

Avoiding Constant Transformations

There are times when the optimal scaling produces a constant transformed variable. This can happen with the MONOTONE, UNTIE, and MSPLINE transformations when the target is negatively correlated with the original input variable. It can happen with all transformations when the target is uncorrelated with the original input variable. When this happens, the procedure modifies the target to avoid a constant transformation. This strategy avoids certain nonoptimal solutions.

If the transformation is monotonic and a constant transformed variable results, the procedure multiplies the target by -1 and tries the optimal scaling again. If the transformation is not monotonic or if the multiplication by -1 did not help, the procedure tries using a random target. If the transformation is still constant, the previous nonconstant transformation is retained. When a constant transformation is avoided by any strategy, this message is displayed: "A constant transformation was avoided for *name*."

Constant Variables

Constant and almost constant variables are zeroed and ignored.

REITERATE Option Usage

You can use the REITERATE option to perform additional iterations when PROC PRINQUAL stops before the data have adequately converged. For example, suppose that you execute the following code:

```
proc prinqual data=A cor out=B;
   transform mspline(X1-X5);
run;
```

If the transformations do not converge in the default 30 iterations, you can perform more iterations without repeating the first 30 iterations.

```
proc prinqual data=B reiterate cor out=B;
   transform mspline(X1-X5);
run;
```

Note that a WHERE statement is not necessary to exclude the correlation observations. They are automatically excluded because their _TYPE_ variable value is not 'SCORE'.

You can also use the REITERATE option to specify starting values other than the original values for the transformations. Providing alternate starting points may avoid local optima. Here are two examples.

```
proc prinqual data=A out=B;
   transform rank(X1-X5);
run;

proc prinqual data=B reiterate out=C;
   /* Use ranks as the starting point. */
   transform monotone(X1-X5);
run;

data B;
   set A;
   array TXS[5] TX1-TX5;
   do j = 1 to 5;
      TXS[j] = normal(0);
   end;
run;

proc prinqual data=B reiterate out=C;
   /* Use a random starting point. */
   transform monotone(X1-X5);
run;
```

Note that divergence with the REITERATE option, particularly in the second iteration, is not an error since the initial transformation is not required to be a valid member of the transformation family. When you specify the REITERATE option, the iteration does not terminate when the criterion change is negative during the first ten iterations.

Passive Observations

Observations may be excluded from the analysis for several reasons, including zero weight, zero frequency, missing values in variables designated IDENTITY, or missing values with the NOMISS option specified. These observations are passive in that they do not contribute to determining transformations, R^2, total variance, and so on. However, some information can be computed for them, such as approximations, principal component scores, and transformed values. Passive observations in the output data set have a blank value for the variable _TYPE_.

Missing value estimates for passive observations may converge slowly with METHOD=MTV. In the following example, the missing value estimates should be 2, 5, and 8. Since the nonpassive observations do not change, the procedure converges in one iteration but the missing value estimates do not converge. The extra iterations produced by specifying CONVERGE=-1 and CCONVERGE=-1, as shown in the second PROC PRINQUAL step, generate the expected results.

```
data A;
   input X Y;
   datalines;
1 1
2 .
3 3
4 4
5 .
6 6
7 7
8 .
9 9
;

proc prinqual nomiss data=A nomiss n=1 out=B method=mtv;
   transform lin(X Y);
run;

proc print;
run;

proc prinqual nomiss data=A nomiss n=1 out=B method=mtv
   converge=-1 cconverge=-1;
   transform lin(X Y);
run;

proc print;
run;
```

Computational Resources

This section provides information on the computational resources required to run PROC PRINQUAL.

Let

$$\begin{aligned} N &= \text{number of observations} \\ m &= \text{number of variables} \\ n &= \text{number of principal components} \\ k &= \text{maximum spline degree} \\ p &= \text{maximum number of knots} \end{aligned}$$

- For the MTV algorithm, more than

$$56m + 8Nm + 8\left(6N + (p+k+2)(p+k+11)\right)$$

  bytes of array space are required.

- For the MGV and MAC algorithms, more than $56m$ plus the maximum of the data matrix size and the optimal scaling work space bytes of array space are required. The data matrix size is $8Nm$ bytes. The optimal scaling work space requires less than $8\left(6N + (p+k+2)(p+k+11)\right)$ bytes.

- For the MTV and MGV algorithms, more than $56m + 4m(m+1)$ bytes of array space are required.

- PROC PRINQUAL tries to store the original and transformed data in memory. If there is not enough memory, a utility data set is used, potentially resulting in a large increase in execution time. The amount of memory for the preceding data formulas are underestimates of the amount of memory needed to handle most problems. These formulas give an absolute minimum amount of memory required. If a utility data set is used, and if memory could be used with perfect efficiency, then roughly the amount of memory stated above would be needed. In reality, most problems require at least two or three times the minimum.

- PROC PRINQUAL sorts the data once. The sort time is roughly proportional to $mN^{3/2}$.

- For the MTV algorithm, the time required to compute the variable approximations is roughly proportional to $2Nm^2 + 5m^3 + nm^2$.

- For the MGV algorithm, one regression analysis per iteration is required to compute model parameter estimates. The time required for accumulating the crossproduct matrix is roughly proportional to Nm^2. The time required to compute the regression coefficients is roughly proportional to m^3. For each variable for each iteration, the swept crossproduct matrix is updated with time roughly proportional to m(N+m). The swept crossproduct matrix is updated

for each variable with time roughly proportional to m^2, until computations are refreshed, requiring all sweeps to be performed again.

- The only computationally intensive part of the MAC algorithm is the optimal scaling, since variable approximations are simple averages.
- Each optimal scaling is a multiple regression problem, although some transformations are handled with faster special case algorithms. The number of regressors for the optimal scaling problems depends on the original values of the variable and the type of transformation. For each monotone spline transformation, an unknown number of multiple regressions is required to find a set of coefficients that satisfies the constraints. The B-spline basis is generated twice for each SPLINE and MSPLINE transformation for each iteration. The time required to generate the B-spline basis is roughly proportional to Nk^2.

Displayed Output

The main output from the PRINQUAL procedure is the output data set. However, the procedure does produce displayed output in the form of an iteration history table that includes the following:

- Iteration Number
- the criterion being optimized
- criterion change
- Maximum and Average Absolute Change in standardized variable scores computed over variables that can be iteratively transformed
- notes
- final convergence status

ODS Table Names

PROC PRINQUAL assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in the following table.

For more information on ODS, see Chapter 15, "Using the Output Delivery System."

Table 53.2. ODS Tables Produced in PROC PRINQUAL

ODS Table Name	Description	Statement	Option
MTV	MTV Iteration History	PROC	METHOD=MTV
MGV	MGV Iteration History	PROC	METHOD=MGV
MAC	MAC Iteration History	PROC	METHOD=MAC
ConvergenceStatus	Convergence Status		default

Examples

Example 53.1. Multidimensional Preference Analysis of Cars Data

This example uses PROC PRINQUAL to perform a nonmetric multidimensional preference (MDPREF) analysis (Carroll 1972). MDPREF analysis is a principal component analysis of a data matrix with columns that correspond to people and rows that correspond to objects. The data are ratings or rankings of each person's preference for each object. The data are the transpose of the usual multivariate data matrix. (In other words, the columns are people instead of the more typical matrix where rows represent people.) The final result of an MDPREF analysis is a biplot (Gabriel 1981) of the resulting preference space. A biplot displays the judges and objects in a single plot by projecting them onto the plane in the transformed variable space that accounts for the most variance.

The data are ratings by 25 judges of their preference for each of 17 automobiles. The ratings are made on a 0 to 9 scale, with 0 meaning very weak preference and 9 meaning very strong preference for the automobile. These judgments were made in 1980 about that year's products. There are two additional variables that indicate the manufacturer and model of the automobile.

This example uses PROC PRINQUAL, PROC FACTOR, and the %PLOTIT macro. PROC FACTOR is used before PROC PRINQUAL to perform a principal component analysis of the raw judgments. PROC FACTOR is also used immediately after PROC PRINQUAL since PROC PRINQUAL is a scoring procedure that optimally scores the data but does not report the principal component analysis.

The %PLOTIT macro produces the biplot. For information on the %PLOTIT macro, see Appendix B, "Using the %PLOTIT Macro."

The scree plot, in the standard principal component analysis reported by PROC FACTOR, shows that two principal components should be retained for further use. (See the scree plot in Output 53.1.1—there is a clear separation between the first two components and the remaining components.) There are nine eigenvalues that are precisely zero because there are nine fewer observations than variables in the data matrix. PROC PRINQUAL is then used to monotonically transform the raw judgments to maximize the proportion of variance accounted for by the first two principal components. The following statements create the data set and perform a principal component analysis of the original data. These statements produce Output 53.1.1.

```
title 'Preference Ratings for Automobiles Manufactured in 1980';

data CarPref;
   input Make $ 1-10 Model $ 12-22 @25 (Judge1-Judge25) (1.);
   datalines;
Cadillac   Eldorado     80079904912405089710938 09
Chevrolet  Chevette     00512004234510430035156 98
Chevrolet  Citation     40533058141616435447477 95
Chevrolet  Malibu       60274007231213455456686 58
Ford       Fairmont     20240067150214435306486 55
Ford       Mustang      50071977050211018506575 55
Ford       Pinto        00210003030302015005140 78
Honda      Accord       59568976096999529989750 78
Honda      Civic        48367095074888525677650 75
Lincoln    Continental  70089905922304099620919 09
Plymouth   Gran Fury    70060004341011073334587 08
Plymouth   Horizon      30050056354613024446756 55
Plymouth   Volare       40050036140216027544765 55
Pontiac    Firebird     01078956132012069582659 07
Volkswagen Dasher       48586965088777953778950 00
Volkswagen Rabbit       48585097096957954878850 00
Volvo      DL           99899989099999879899190 00
;

* Principal Component Analysis of the Original Data;
options ls=80 ps=65;
proc factor data=CarPref nfactors=2 scree;
   ods select Eigenvalues ScreePlot;
   var Judge1-Judge25;
   title3 'Principal Components of Original Data';
run;
```

Output 53.1.1. Principal Component Analysis of Original Data

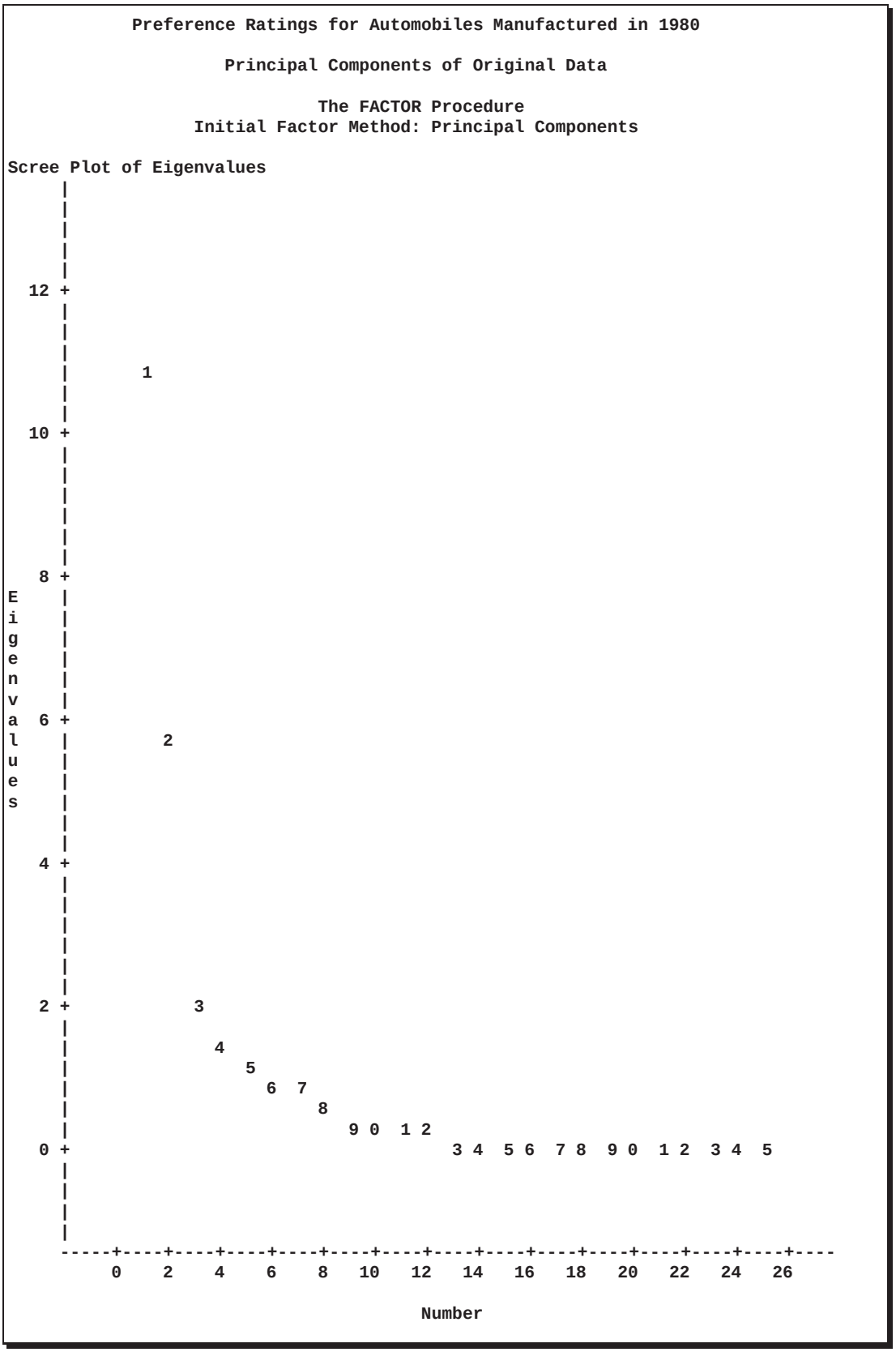
Preference Ratings for Automobiles Manufactured in 1980

Principal Components of Original Data

The FACTOR Procedure
Initial Factor Method: Principal Components

Eigenvalues of the Correlation Matrix: Total = 25 Average = 1

	Eigenvalue	Difference	Proportion	Cumulative
1	10.8857202	5.0349926	0.4354	0.4354
2	5.8507276	3.8077964	0.2340	0.6695
3	2.0429312	0.5207808	0.0817	0.7512
4	1.5221504	0.3078035	0.0609	0.8121
5	1.2143469	0.2564839	0.0486	0.8606
6	0.9578630	0.2197345	0.0383	0.8989
7	0.7381286	0.1497259	0.0295	0.9285
8	0.5884027	0.2117186	0.0235	0.9520
9	0.3766841	0.1091250	0.0151	0.9671
10	0.2675591	0.0773893	0.0107	0.9778
11	0.1901698	0.0463921	0.0076	0.9854
12	0.1437776	0.0349382	0.0058	0.9911
13	0.1088394	0.0607418	0.0044	0.9955
14	0.0480977	0.0056610	0.0019	0.9974
15	0.0424367	0.0202714	0.0017	0.9991
16	0.0221653	0.0221653	0.0009	1.0000
17	0.0000000	0.0000000	0.0000	1.0000
18	0.0000000	0.0000000	0.0000	1.0000
19	0.0000000	0.0000000	0.0000	1.0000
20	0.0000000	0.0000000	0.0000	1.0000
21	0.0000000	0.0000000	0.0000	1.0000
22	0.0000000	0.0000000	0.0000	1.0000
23	0.0000000	0.0000000	0.0000	1.0000
24	0.0000000	0.0000000	0.0000	1.0000
25	0.0000000		0.0000	1.0000


Preference Ratings for Automobiles Manufactured in 1980
Principal Components of Original Data
The FACTOR Procedure
Initial Factor Method: Principal Components
Scree Plot of Eigenvalues
Eigenvalues
12
10
8
6
4
2
0
1
2
3
4
5
6 7
8
9 0 1 2
3 4 5 6 7 8 9 0 1 2 3 4 5
0 2 4 6 8 10 12 14 16 18 20 22 24 26
Number

To fit the nonmetric MDPREF model, you can use the PRINQUAL procedure. The MONOTONE option is specified in the TRANSFORM statement to request a nonmetric MDPREF analysis; alternatively, you can instead specify the IDENTITY option for a metric analysis. Several options are used in the PROC PRINQUAL statement. The option DATA=CarPref specifies the input data set, OUT=Results creates an output data set, and N=2 and the default METHOD=MTV transform the data to better fit a two-component model. The REPLACE option replaces the original data with the monotonically transformed data in the OUT= data set. The MDPREF option standardizes the component scores to variance one so that the geometry of the biplot is correct, and it creates two variables in the OUT= data set named Prin1 and Prin2. These variables contain the standardized principal component scores and structure matrix, which are used to make the biplot. If the variables in data matrix $\mathbf{X}$ are standardized to mean zero and variance one, and n is the number of rows in $\mathbf{X}$, then $\mathbf{X} = \mathbf{V}\mathbf{\Lambda}^{1/2}\mathbf{W}'$ is the principal component model, where $\mathbf{X}'\mathbf{X}/(n-1) = \mathbf{W}\mathbf{\Lambda}\mathbf{W}'$. The $\mathbf{W}$ and $\mathbf{\Lambda}$ contain the eigenvectors and eigenvalues of the correlation matrix of $\mathbf{X}$. The first two columns of $\mathbf{V}$, the standardized component scores, and $\mathbf{W}\mathbf{\Lambda}^{1/2}$, which is the structure matrix, are output. The advantage of creating a biplot based on principal components is that coordinates do not depend on the sample size. The following statements transform the data and produce Output 53.1.2.

```
* Transform the Data to Better Fit a Two Component Model;
proc prinqual data=CarPref out=Results n=2 replace mdpref;
   id model;
   transform monotone(Judge1-Judge25);
   title2 'Multidimensional Preference (MDPREF) Analysis';
   title3 'Optimal Monotonic Transformation of Preference Data';
run;
```

Output 53.1.2. Transformation of Automobile Preference Data

Preference Ratings for Automobiles Manufactured in 1980
Multidimensional Preference (MDPREF) Analysis
Optimal Monotonic Transformation of Preference Data

The PRINQUAL Procedure

PRINQUAL MTV Algorithm Iteration History

Iteration Number	Average Change	Maximum Change	Proportion of Variance	Criterion Change	Note
1	0.24994	1.28017	0.66946		
2	0.07223	0.36958	0.80194	0.13249	
3	0.04522	0.29026	0.81598	0.01404	
4	0.03096	0.25213	0.82178	0.00580	
5	0.02182	0.23045	0.82493	0.00315	
6	0.01602	0.19017	0.82680	0.00187	
7	0.01219	0.14748	0.82793	0.00113	
8	0.00953	0.11031	0.82861	0.00068	
9	0.00737	0.06461	0.82904	0.00043	
10	0.00556	0.04469	0.82930	0.00026	
11	0.00445	0.04087	0.82944	0.00014	
12	0.00381	0.03706	0.82955	0.00011	
13	0.00319	0.03348	0.82965	0.00009	
14	0.00255	0.02999	0.82971	0.00006	
15	0.00213	0.02824	0.82976	0.00005	
16	0.00183	0.02646	0.82980	0.00004	
17	0.00159	0.02472	0.82983	0.00003	
18	0.00139	0.02305	0.82985	0.00003	
19	0.00123	0.02145	0.82988	0.00002	
20	0.00109	0.01993	0.82989	0.00002	
21	0.00096	0.01850	0.82991	0.00001	
22	0.00086	0.01715	0.82992	0.00001	
23	0.00076	0.01588	0.82993	0.00001	
24	0.00067	0.01440	0.82994	0.00001	
25	0.00059	0.00871	0.82994	0.00001	
26	0.00050	0.00720	0.82995	0.00000	
27	0.00043	0.00642	0.82995	0.00000	
28	0.00037	0.00573	0.82995	0.00000	
29	0.00031	0.00510	0.82995	0.00000	
30	0.00027	0.00454	0.82995	0.00000	Not Converged

WARNING: Failed to converge, however criterion change is less than 0.0001.

The iteration history displayed by PROC PRINQUAL indicates that the proportion of variance is increased from an initial 0.66946 to 0.82995. The proportion of variance accounted for by PROC PRINQUAL on the first iteration equals the cumulative proportion of variance shown by PROC FACTOR for the first two principal components. In this example, PROC PRINQUAL's initial iteration performs a standard principal component analysis of the raw data. The columns labeled Average Change, Maximum Change, and Variance Change contain values that always decrease, indicating that PROC PRINQUAL is improving the transformations at a monotonically decreasing rate over the iterations. This does not always happen, and when it does not, it suggests that the analysis may be converging to a degenerate solution. See Example 53.2 on page 2817 for a discussion of a degenerate solution. The algorithm does not converge in 30 iterations. However, the criterion change is small, indicating that more iterations are unlikely to have much effect on the results.

The second PROC FACTOR analysis is performed on the transformed data. The WHERE statement is used to retain only the monotonically transformed judgments. The scree plot shows that the first two eigenvalues are now much larger than the remaining smaller eigenvalues. The second eigenvalue has increased markedly at the expense of the next several eigenvalues. Two principal components seem to be necessary and sufficient to adequately describe these judges' preferences for these automobiles. The cumulative proportion of variance displayed by PROC FACTOR for the first two principal components is 0.83. The following statements perform the analysis and produce Output 53.1.3:

```
* Final Principal Component Analysis;
proc factor data=Results nfactors=2 scree;
   ods select Eigenvalues ScreePlot;
   var Judge1-Judge25;
   where _TYPE_='SCORE';
   title3 'Principal Components of Monotonically Transformed Data';
run;
```

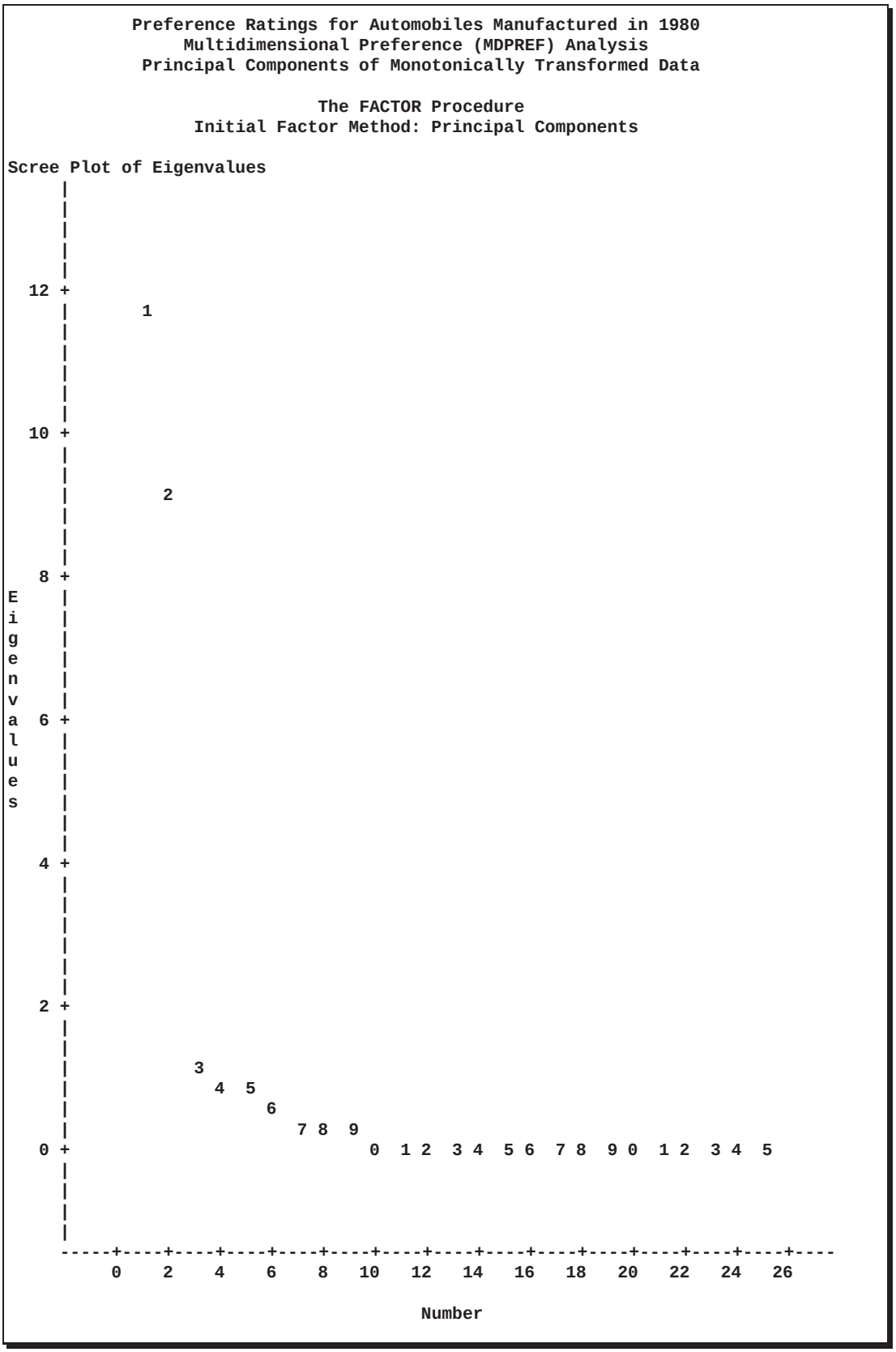
Output 53.1.3. Principal Components of Transformed Data

Preference Ratings for Automobiles Manufactured in 1980
Multidimensional Preference (MDPREF) Analysis
Principal Components of Monotonically Transformed Data

The FACTOR Procedure
Initial Factor Method: Principal Components

Eigenvalues of the Correlation Matrix: Total = 25 Average = 1

	Eigenvalue	Difference	Proportion	Cumulative
1	11.5959045	2.4429455	0.4638	0.4638
2	9.1529589	7.9952554	0.3661	0.8300
3	1.1577036	0.3072013	0.0463	0.8763
4	0.8505023	0.1284323	0.0340	0.9103
5	0.7220700	0.2613540	0.0289	0.9392
6	0.4607160	0.0958339	0.0184	0.9576
7	0.3648821	0.0877851	0.0146	0.9722
8	0.2770970	0.1250945	0.0111	0.9833
9	0.1520025	0.0506622	0.0061	0.9894
10	0.1013403	0.0292763	0.0041	0.9934
11	0.0720640	0.0200979	0.0029	0.9963
12	0.0519661	0.0336675	0.0021	0.9984
13	0.0182987	0.0027059	0.0007	0.9991
14	0.0155927	0.0093669	0.0006	0.9997
15	0.0062258	0.0055503	0.0002	1.0000
16	0.0006755	0.0006755	0.0000	1.0000
17	0.0000000	0.0000000	0.0000	1.0000
18	0.0000000	0.0000000	0.0000	1.0000
19	0.0000000	0.0000000	0.0000	1.0000
20	0.0000000	0.0000000	0.0000	1.0000
21	0.0000000	0.0000000	0.0000	1.0000
22	0.0000000	0.0000000	0.0000	1.0000
23	0.0000000	0.0000000	0.0000	1.0000
24	0.0000000	0.0000000	0.0000	1.0000
25	0.0000000		0.0000	1.0000


Preference Ratings for Automobiles Manufactured in 1980
Multidimensional Preference (MDPREF) Analysis
Principal Components of Monotonically Transformed Data
The FACTOR Procedure
Initial Factor Method: Principal Components
Scree Plot of Eigenvalues
Eigenvalues
Number

The remainder of the example constructs the MDPREF biplot. A biplot is a plot that displays the relation between the row points and the columns of a data matrix. The rows of $\mathbf{V}$, the standardized component scores, and $\mathbf{W}\mathbf{\Lambda}^{1/2}$, which is the structure matrix, contain enough information to reproduce $\mathbf{X}$. The (i, j) element of $\mathbf{X}$ is the product of row i of $\mathbf{V}$ and row j of $\mathbf{W}\mathbf{\Lambda}^{1/2}$. If all but the first two columns of $\mathbf{V}$ and $\mathbf{W}\mathbf{\Lambda}^{1/2}$ are discarded, the (i, j) element of $\mathbf{X}$ is approximated by the product of row i of $\mathbf{V}$ and row j of $\mathbf{W}\mathbf{\Lambda}^{1/2}$.

Since the MDPREF analysis is based on a principal component model, the dimensions of the MDPREF biplot are the first two principal components. The first principal component is the longest dimension through the MDPREF biplot. The first principal component is overall preference, which is the most salient dimension in the preference judgments. One end points in the direction that is on the average preferred most by the judges, and the other end points in the least preferred direction. The second principal component is orthogonal to the first principal component, and it is the orthogonal direction that is the second most salient. The interpretation of the second dimension varies from example to example.

With an MDPREF biplot, it is geometrically appropriate to represent each automobile (object) by a point and each judge by a vector. The automobile points have coordinates that are the scores of the automobile on the first two principal components. The judge vectors emanate from the origin of the space and go through a point with coordinates that are the coefficients of the judge (variable) on the first two principal components.

The absolute length of a vector is arbitrary. However, the relative lengths of the vectors indicate fit, with the squared lengths being proportional to the communalities in the PROC FACTOR output. The direction of the vector indicates the direction that is most preferred by the individual judge, with preference increasing as the vector moves from the origin. Let $\mathbf{v}'$ be row i of $\mathbf{V}$, $\mathbf{u}'$ be row j of $\mathbf{U} = \mathbf{W}\mathbf{\Lambda}^{1/2}$, $\|\mathbf{v}\|$ be the length of $\mathbf{v}$, $\|\mathbf{u}\|$ be the length of $\mathbf{u}$, and θ be the angle between $\mathbf{v}$ and $\mathbf{u}$. The predicted degree of preference that an individual judge has for an automobile is $\mathbf{u}'\mathbf{v} = \|\mathbf{u}\|\ \|\mathbf{v}\| \cos\theta$. Each car point can be orthogonally projected onto the vector. The projection of car i on vector j is $\mathbf{u}((\mathbf{u}'\mathbf{v})/(\mathbf{u}'\mathbf{u}))$ and the length of this projection is $\|\mathbf{v}\| \cos\theta$. The automobile that projects farthest along a vector in the direction it points is that judge's most preferred automobile, since the length of this projection, $\|\mathbf{v}\| \cos\theta$, differs from the predicted preference, $\|\mathbf{u}\|\ \|\mathbf{v}\| \cos\theta$, only by $\|\mathbf{u}\|$, which is constant within each judge.

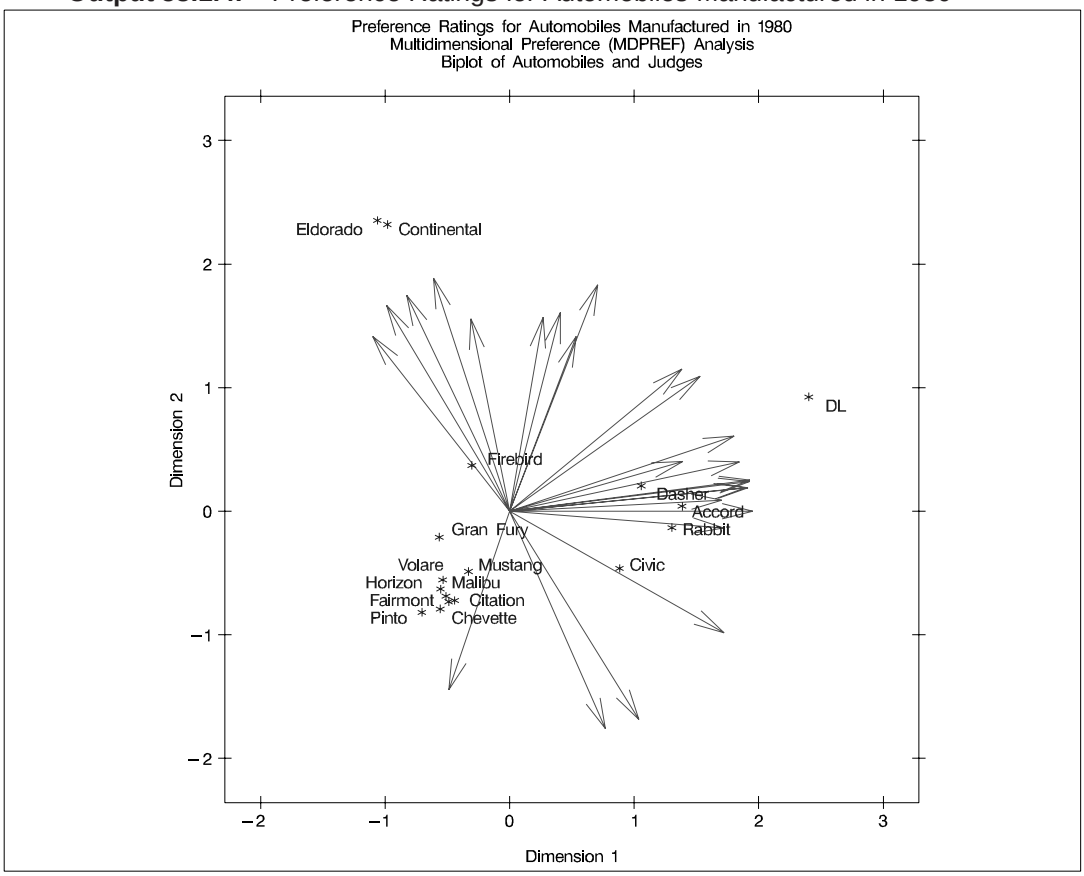
To interpret the biplot, look for directions through the plot that show a continuous change in some attribute of the automobiles, or look for regions in the plot that contain clusters of automobile points and determine what attributes the automobiles have in common. Those points that are tightly clustered in a region of the plot represent automobiles that have the same preference patterns across the judges. Those vectors that point in roughly the same direction represent judges who tend to have similar preference patterns.

The following statement constructs the biplot and produces Output 53.1.4:

```
title3 'Biplot of Automobiles and Judges';
%plotit(data=results, datatype=mdpref 2);
```

The DATATYPE=MDPREF 2 option indicates that the coordinates come from an MDPREF analysis, so the macro represents the scores as points and the structure as vectors, with the vectors stretched by a factor of two to make a better graphical display.

Output 53.1.4. Preference Ratings for Automobiles Manufactured in 1980



In the biplot, American automobiles are located on the left of the space, while European and Japanese automobiles are located on the right. At the top of the space are expensive American automobiles (Cadillac Eldorado, Lincoln Continental) while at the bottom are inexpensive ones (Pinto, Chevette). The first principal component differentiates American from imported automobiles, and the second arranges automobiles by price and other associated characteristics.

The two expensive American automobiles form a cluster, the sporty automobile (Firebird) is by itself, the Volvo DL is by itself, and the remaining imported autos form a cluster, as do the remaining American autos. It seems there are 5 prototypical automobiles in this set of 17, in terms of preference patterns among the 25 judges.

Most of the judges prefer the imported automobiles, especially the Volvo. There is also a fairly large minority that prefer the expensive cars, whether or not they are American (those with vectors that point towards one o'clock), or simply prefer expensive American automobiles (vectors that point towards eleven o'clock). There are two people who prefer anything except expensive American cars (five o'clock vectors), and one who prefers inexpensive American cars (seven o'clock vector).

Several vectors point toward the upper-right corner of the plot, toward a region with no cars. This is the region between the European and Japanese cars on the right and the luxury cars on the top. This suggests that there is a market for luxury Japanese and European cars.

Example 53.2. Principal Components of Basketball Rankings

The data in this example are 1985–1986 preseason rankings of 35 college basketball teams by 10 different news services. The services do not all rank the same teams or the same number of teams, so there are missing values in these data. Each of the 35 teams in the data set is ranked by at least one news service. One way of summarizing these data is with a principal component analysis, since the rankings should all be related to a single underlying variable, the first principal component.

You can use PROC PRINQUAL to estimate the missing ranks and compute scores for all observations. You can formulate a PROC PRINQUAL analysis that assumes that the observed ranks are ordinal variables and replaces the ranks with new numbers that are monotonic with the ranks and better fit the one principal component model. The missing rank estimates need to be constrained since a news service would have positioned the unranked teams below the teams it ranked. PROC PRINQUAL should impose order constraints within the nonmissing values and between the missing and nonmissing values, but not within the missing values. PROC PRINQUAL has sophisticated missing data handling facilities; however, these facilities cannot directly handle this problem. The solution requires reformulating the problem.

By performing some preliminary data manipulations, specifying the N=1 option in the PROC PRINQUAL statement, and specifying the UNTIE transformation in the TRANSFORM statement, you can make the missing value estimates conform to the requirements. The PROC MEANS step finds the largest rank for each variable. The next DATA step replaces missing values with a value that is one larger than the largest observed rank. The N=1 option (in the PRINQUAL procedure) specifies that the variables should be transformed to make them as one-dimensional as possible. The UNTIE transformation in the TRANSFORM statement monotonically transforms the ranks, untying any ties in an optimal way. Because the only ties are for the values that replace the missing values, and because these values are larger than the observed values, the rescoring of the data satisfies the preceding requirements.

The following statements create the data set and perform the transformations discussed previously. These statements produce Output 53.2.1.

```
* Example 2: Basketball Data
*
* Preseason 1985 College Basketball Rankings
* (rankings of 35 teams by 10 news services)
*
* Note: (a) Various news services rank varying numbers of teams.
*       (b) Not all 35 teams are ranked by all news services.
*       (c) Each team is ranked by at least one service.
*       (d) Rank 20 is missing for UPI.;
```

```
title1 '1985 Preseason College Basketball Rankings';

data bballm;
   input School $13. CSN DurhamSun DurhamHerald WashingtonPost
         USA_Today SportMagazine InsideSports UPI AP
         SportsIllustrated;
   label CSN               = 'Community Sports News
                             (Chapel Hill, NC)'
         DurhamSun         = 'Durham Sun'
         DurhamHerald      = 'Durham Morning Herald'
         WashingtonPost    = 'Washington Post'
         USA_Today         = 'USA Today'
         SportMagazine     = 'Sport Magazine'
         InsideSports      = 'Inside Sports'
         UPI               = 'United Press International'
         AP                = 'Associated Press'
         SportsIllustrated = 'Sports Illustrated'
         ;
   format CSN--SportsIllustrated 5.1;
   datalines;
Louisville     1  8  1  9  8  9  6 10  9  9
Georgia Tech   2  2  4  3  1  1  1  2  1  1
Kansas         3  4  5  1  5 11  8  4  5  7
Michigan       4  5  9  4  2  5  3  1  3  2
Duke           5  6  7  5  4 10  4  5  6  5
UNC            6  1  2  2  3  4  2  3  2  3
Syracuse       7 10  6 11  6  6  5  6  4 10
Notre Dame     8 14 15 13 11 20 18 13 12  .
Kentucky       9 15 16 14 14 19 11 12 11 13
LSU           10  9 13  . 13 15 16  9 14  8
DePaul        11  . 21 15 20  . 19  .  . 19
Georgetown    12  7  8  6  9  2  9  8  8  4
Navy          13 20 23 10 18 13 15  . 20  .
Illinois      14  3  3  7  7  3 10  7  7  6
Iowa          15 16  .  . 23  .  . 14  . 20
Arkansas      16  .  .  . 25  .  .  .  . 16
Memphis State 17  . 11  . 16  8 20  . 15 12
Washington    18  .  .  .  .  .  . 17  .  .
UAB           19 13 10  . 12 17  . 16 16 15
UNLV          20 18 18 19 22  . 14 18 18  .
NC State      21 17 14 16 15  . 12 15 17 18
Maryland      22  .  .  . 19  .  .  . 19 14
Pittsburgh    23  .  .  .  .  .  .  .  .  .
Oklahoma      24 19 17 17 17 12 17  . 13 17
Indiana       25 12 20 18 21  .  .  .  .  .
Virginia      26  . 22  .  . 18  .  .  .  .
Old Dominion  27  .  .  .  .  .  .  .  .  .
Auburn        28 11 12  8 10  7  7 11 10 11
St. Johns     29  .  .  .  . 14  .  .  .  .
UCLA          30  .  .  .  .  .  . 19  .  .
St. Joseph's   .  . 19  .  .  .  .  .  .  .
Tennessee      .  . 24  .  . 16  .  .  .  .
```

```
Montana        .  .  . 20  .  .  .  .  .  .
Houston        .  .  .  . 24  .  .  .  .  .
Virginia Tech  .  .  .  .  .  . 13  .  .  .
;

* Find maximum rank for each news service and replace
* each missing value with the next highest rank.;

proc means data=bballm noprint;
   output out=maxrank
      max=mcsn mdurs mdurh mwas musa mspom mins mupi map mspoi;
run;

data bball;
   set bballm;
   if _n_=1 then set maxrank;
   array services[10] CSN--SportsIllustrated;
   array maxranks[10] mcsn--mspoi;
   keep School CSN--SportsIllustrated;
   do i=1 to 10;
      if services[i]=. then services[i]=maxranks[i]+1;
      end;
run;

* Assume that the ranks are ordinal and that unranked teams
* would have been ranked lower than ranked teams.  Monotonically
* transform all ranked teams while estimating the unranked teams.
* Enforce the constraint that the missing ranks are estimated to
* be less than the observed ranks.  Order the unranked teams
* optimally within this constraint.  Do this so as to maximize
* the variance accounted for by one linear combination.  This
* makes the data as nearly rank one as possible, given the
* constraints.
*
* NOTE: The UNTIE transformation should be used with caution.
* If frequently produces degenerate results.;

proc prinqual data=bball out=tbball scores n=1 tstandard=z;
   title2 'Optimal Monotonic Transformation of Ranked Teams';
   title3 'with Constrained Estimation of Unranked Teams';
   transform untie(CSN -- SportsIllustrated);
   id School;
run;
```

Output 53.2.1. Transformation of Basketball Team Rankings

1985 Preseason College Basketball Rankings
Optimal Monotonic Transformation of Ranked Teams
with Constrained Estimation of Unranked Teams

The PRINQUAL Procedure

PRINQUAL MTV Algorithm Iteration History

Iteration Number	Average Change	Maximum Change	Proportion of Variance	Criterion Change	Note
1	0.18563	0.76531	0.85850		
2	0.03225	0.14627	0.94362	0.08512	
3	0.02126	0.10530	0.94669	0.00307	
4	0.01467	0.07526	0.94801	0.00132	
5	0.01067	0.05282	0.94865	0.00064	
6	0.00800	0.03669	0.94899	0.00034	
7	0.00617	0.02862	0.94919	0.00020	
8	0.00486	0.02636	0.94932	0.00013	
9	0.00395	0.02453	0.94941	0.00009	
10	0.00327	0.02300	0.94947	0.00006	
11	0.00275	0.02166	0.94952	0.00005	
12	0.00236	0.02041	0.94956	0.00004	
13	0.00205	0.01927	0.94959	0.00003	
14	0.00181	0.01818	0.94962	0.00003	
15	0.00162	0.01719	0.94964	0.00002	
16	0.00147	0.01629	0.94966	0.00002	
17	0.00136	0.01546	0.94968	0.00002	
18	0.00128	0.01469	0.94970	0.00002	
19	0.00121	0.01398	0.94971	0.00001	
20	0.00115	0.01332	0.94973	0.00001	
21	0.00111	0.01271	0.94974	0.00001	
22	0.00105	0.01213	0.94975	0.00001	
23	0.00099	0.01155	0.94976	0.00001	
24	0.00095	0.01095	0.94977	0.00001	
25	0.00091	0.01038	0.94978	0.00001	
26	0.00088	0.00986	0.94978	0.00001	
27	0.00084	0.00936	0.94979	0.00001	
28	0.00081	0.00889	0.94980	0.00001	
29	0.00077	0.00846	0.94980	0.00000	
30	0.00073	0.00805	0.94980	0.00000	Not Converged

WARNING: Failed to converge, however criterion change is less than 0.0001.

An alternative approach is to use the pairwise deletion option of the CORR procedure to compute the correlation matrix and then use PROC PRINCOMP or PROC FACTOR to perform the principal component analysis. This approach has several disadvantages. The correlation matrix may not be positive semidefinite (psd), an assumption required for principal component analysis. PROC PRINQUAL always produces a psd correlation matrix. Even with pairwise deletion, PROC CORR removes the six observations with only a single nonmissing value from this data set. Finally, it is still not possible to calculate scores on the principal components for those teams that have missing values.

It is possible to compute the composite ranking using PROC PRINCOMP and some preliminary data manipulations, similar to those discussed previously. Chapter 52, "The PRINCOMP Procedure," contains an example where the average of the unused ranks in each poll is substituted for the missing values, and each observation is weighted by the number of nonmissing values. This method has much to recommend

it. It is much faster and simpler than using PROC PRINQUAL. It is also much less prone to degeneracies and capitalization on chance. However, PROC PRINCOMP does not allow the nonmissing ranks to be monotonically transformed and the missing values untied to optimize fit.

PROC PRINQUAL monotonically transforms the observed ranks and estimates the missing ranks (within the constraints given previously) to account for almost 95 percent of the variance of the transformed data by just one dimension. PROC FACTOR is then used to report details of the principal component analysis of the transformed data. As shown by the Factor Pattern values in Output 53.2.2, nine of the ten news services have a correlation of 0.95 or larger with the scores on the first principal component after the data are optimally transformed. The scores are sorted and the composite ranking is displayed following the PROC FACTOR output. More confidence can be placed in the stability of the scores for the teams that are ranked by the majority of the news services than in scores for teams that are seldom ranked.

The monotonic transformations are plotted for each of the ten news services. These plots are the values of the raw ranks (with the missing ranks replaced by the maximum rank plus one) versus the rescored (transformed) ranks. The transformations are the step functions that maximize the fit of the data to the principal component model. Smoother transformations could be found by using MSPLINE transformations, but MSPLINE transformations would not correctly handle the missing data problem.

The following statements perform the final analysis and produce Output 53.2.2 through Output 53.2.3:

```
* Perform the Final Principal Component Analysis;
proc factor nfactors=1;
   var TCSN -- TSportsIllustrated;
   title4 'Principal Component Analysis';
run;

proc sort;
   by Prin1;
run;

* Display Scores on the First Principal Component;
proc print;
   title4 'Teams Ordered by Scores on First Principal Component';
   var School Prin1;
run;

* Plot the Transformations;
goptions goutmode=replace nodisplay;
%let opts = haxis=axis2 vaxis=axis1 frame cframe=ligr;
* Depending on your goptions, these plot options may work better:
* %let opts = haxis=axis2 vaxis=axis1 frame;

proc gplot;
   title;
   axis1 minor=none label=(angle=90 rotate=0)
         order=(-3 to 2 by 1);
```

```
   axis2 minor=none order=(0 to 40 by 10);
   plot TCSN*CSN                              / &opts name='prqex1';
   plot TDurhamSun*DurhamSun                  / &opts name='prqex2';
   plot TDurhamHerald*DurhamHerald            / &opts name='prqex3';
   plot TWashingtonPost*WashingtonPost        / &opts name='prqex4';
   plot TUSA_Today*USA_Today                  / &opts name='prqex5';
   plot TSportMagazine*SportMagazine          / &opts name='prqex6';
   plot TInsideSports*InsideSports            / &opts name='prqex7';
   plot TUPI*UPI                              / &opts name='prqex8';
   plot TAP*AP                                / &opts name='prqex9';
   plot TSportsIllustrated*SportsIllustrated / &opts name='prqex10';
   symbol1 c=blue;
run; quit;

goptions display;
proc greplay nofs tc=sashelp.templt template=l2r2;
   igout gseg;
   treplay 1:prqex1 2:prqex2 3:prqex3 4:prqex4;
   treplay 1:prqex5 2:prqex6 3:prqex7 4:prqex8;
   treplay 1:prqex9 3:prqex10;
run; quit;
```

Output 53.2.2. Alternative Approach for Analyzing Basketball Rankings

1985 Preseason College Basketball Rankings
Optimal Monotonic Transformation of Ranked Teams
with Constrained Estimation of Unranked Teams
Principal Component Analysis

The FACTOR Procedure
Initial Factor Method: Principal Components

Prior Communality Estimates: ONE

Eigenvalues of the Correlation Matrix: Total = 10 Average = 1

	Eigenvalue	Difference	Proportion	Cumulative
1	9.49808040	9.27698055	0.9498	0.9498
2	0.22109985	0.13434105	0.0221	0.9719
3	0.08675881	0.01266762	0.0087	0.9806
4	0.07409119	0.03048596	0.0074	0.9880
5	0.04360523	0.00567364	0.0044	0.9924
6	0.03793160	0.02098385	0.0038	0.9962
7	0.01694775	0.00299099	0.0017	0.9979
8	0.01395675	0.00982630	0.0014	0.9992
9	0.00413045	0.00073249	0.0004	0.9997
10	0.00339797		0.0003	1.0000

1 factor will be retained by the NFACTOR criterion.

Factor Pattern

		Factor1
TCSN	CSN Transformation	0.91136
TDurhamSun	DurhamSun Transformation	0.98887
TDurhamHerald	DurhamHerald Transformation	0.97402
TWashingtonPost	WashingtonPost Transformation	0.97408
TUSA_Today	USA_Today Transformation	0.98867
TSportMagazine	SportMagazine Transformation	0.95331
TInsideSports	InsideSports Transformation	0.98521
TUPI	UPI Transformation	0.98534
TAP	AP Transformation	0.99590
TSportsIllustrated	SportsIllustrated Transformation	0.98615

Variance Explained by Each Factor

Factor1
9.4980804

Final Communality Estimates: Total = 9.498080

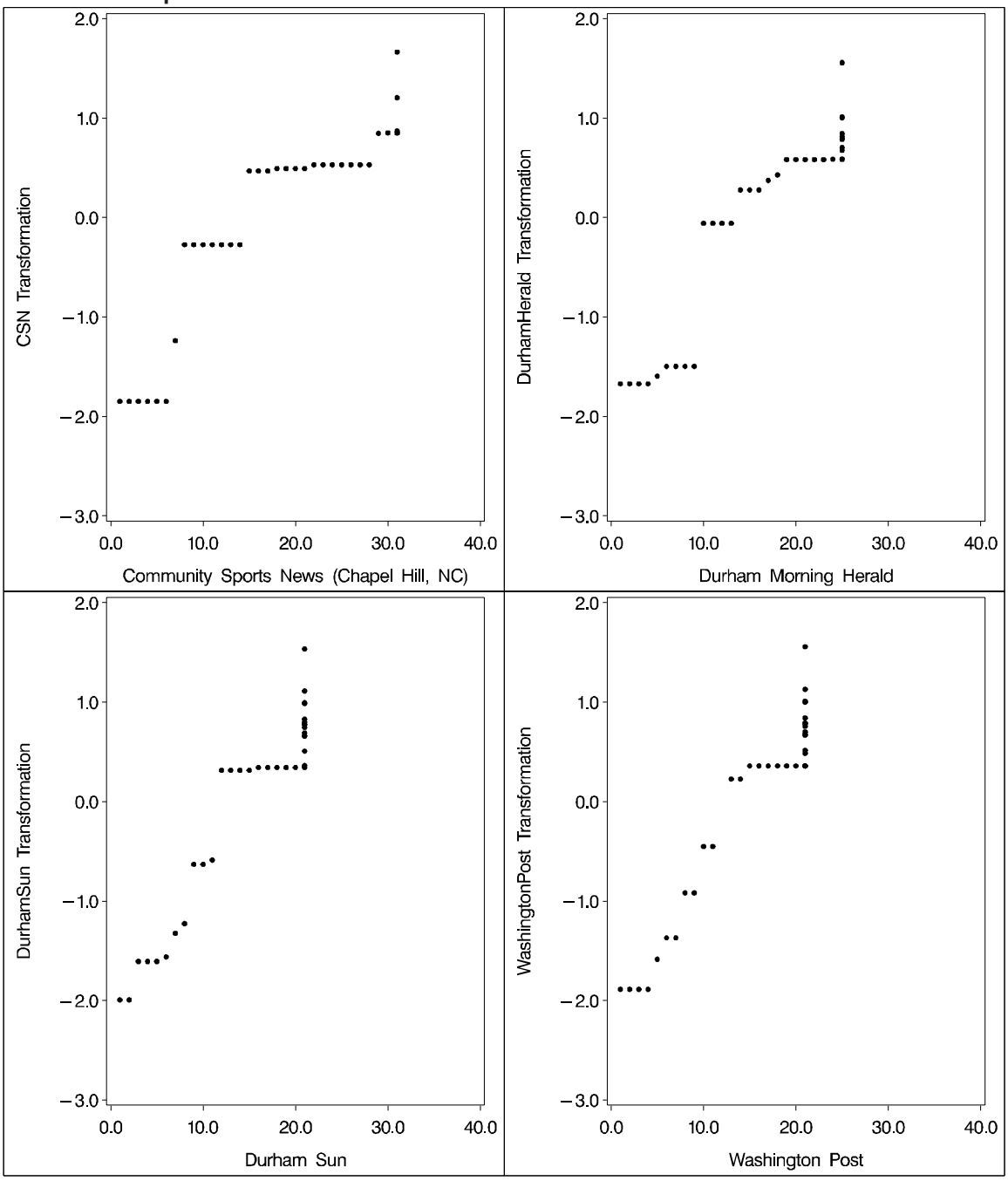
TCSN	TDurhamSun	TDurham Herald	TWashington Post	TUSA_Today
0.83057866	0.97785439	0.94870875	0.94882907	0.97747798

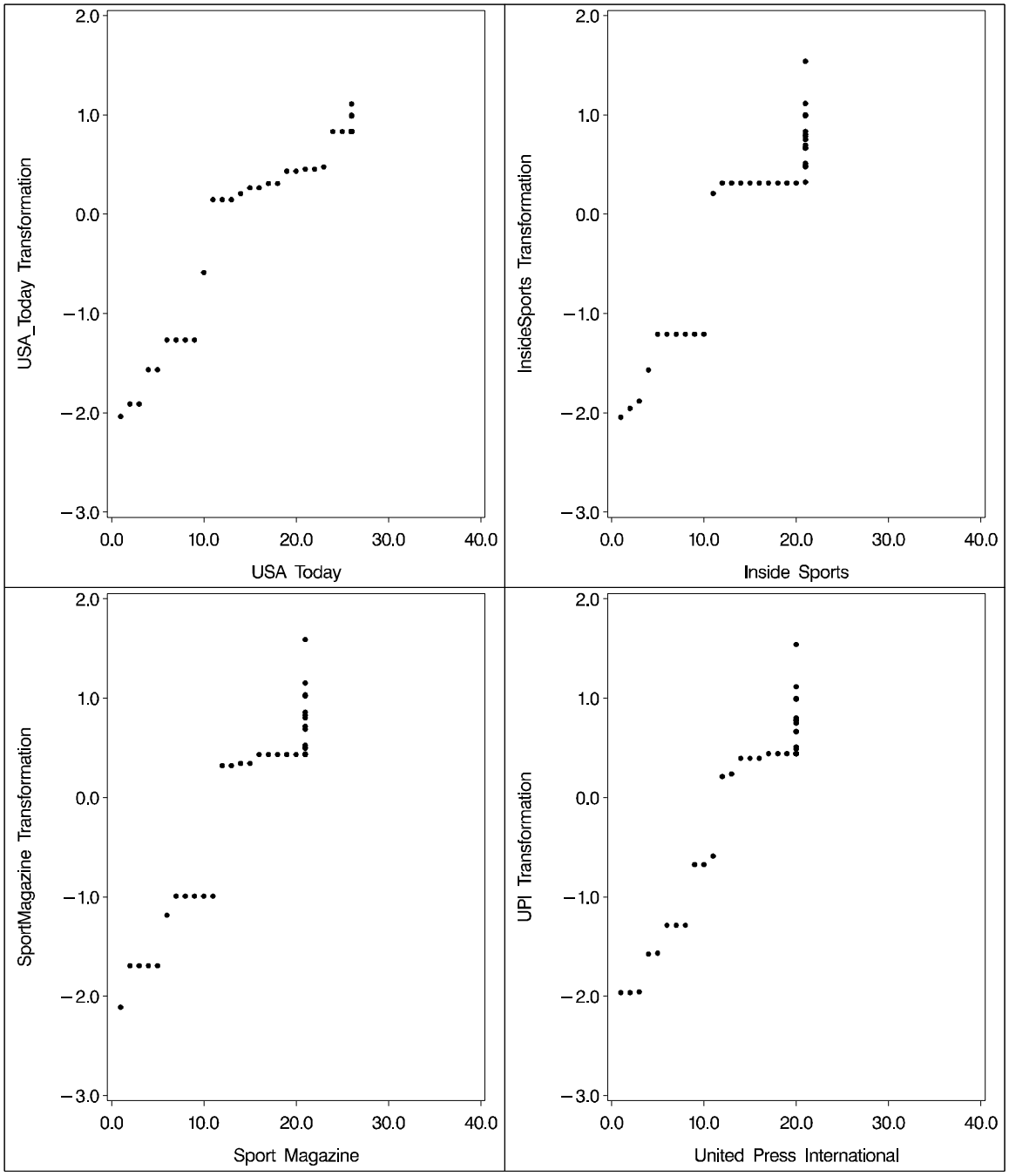
TSport Magazine	TInside Sports	TUPI	TAP	TSports Illustrated
0.90879058	0.97064640	0.97088804	0.99181626	0.97249026

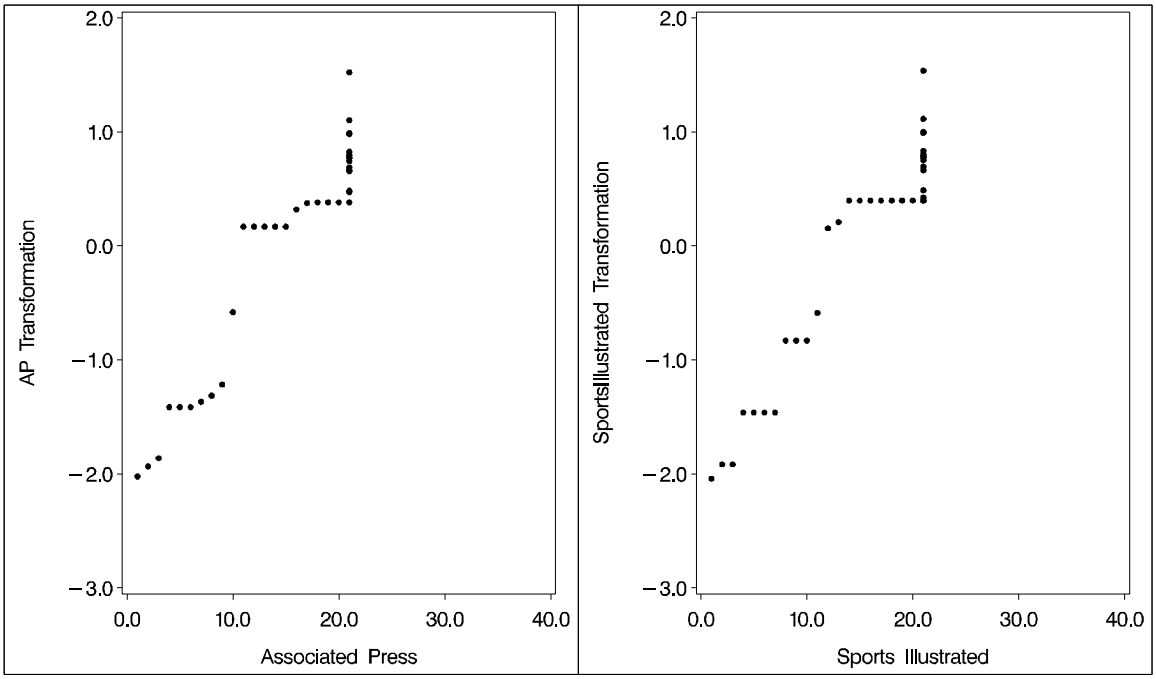
1985 Preseason College Basketball Rankings
Optimal Monotonic Transformation of Ranked Teams
with Constrained Estimation of Unranked Teams
Teams Ordered by Scores on First Principal Component

Obs	School	Prin1
1	Georgia Tech	-6.20315
2	UNC	-5.93314
3	Michigan	-5.71034
4	Kansas	-4.78699
5	Duke	-4.75896
6	Illinois	-4.19220
7	Georgetown	-4.02861
8	Louisville	-3.73087
9	Syracuse	-3.47497
10	Auburn	-1.78429
11	LSU	-0.35928
12	Memphis State	0.46737
13	Kentucky	0.63661
14	Notre Dame	0.71919
15	Navy	0.76187
16	UAB	0.98316
17	DePaul	1.09891
18	Oklahoma	1.12012
19	NC State	1.15144
20	UNLV	1.28766
21	Iowa	1.45260
22	Indiana	1.48123
23	Maryland	1.54935
24	Virginia	2.01385
25	Arkansas	2.02718
26	Washington	2.10878
27	Tennessee	2.27770
28	Virginia Tech	2.36103
29	St. Johns	2.37387
30	Montana	2.43502
31	UCLA	2.52481
32	Pittsburgh	3.00907
33	Old Dominion	3.03324
34	St. Joseph’s	3.39259
35	Houston	4.69614

Output 53.2.3. Monotonic Transformation for Each News Service




USA_Today Transformation
USA Today
InsideSports Transformation
Inside Sports
SportMagazine Transformation
Sport Magazine
UPI Transformation
United Press International
2.0
1.0
0.0
−1.0
−2.0
−3.0
0.0
10.0
20.0
30.0
40.0



The ordinary PROC PRINQUAL missing data handling facilities do not work for these data because they do not constrain the missing data estimates properly. If you code the missing ranks as missing and specify linear transformations, then you can compute least-squares estimates of the missing values without transforming the observed values. The first principal component then accounts for 92 percent of the variance after 20 iterations. However, Virginia Tech is ranked number 11 by its score even though it appeared in only one poll (InsideSports ranked it number 13, anchoring it firmly in the middle). Specifying monotone transformations is also inappropriate since they too allow unranked teams to move in between ranked teams.

With these data, the combination of monotone transformations and the freedom to score the missing ranks without constraint leads to degenerate transformations. PROC PRINQUAL tries to merge the 35 points into two points, producing a perfect fit in one dimension. There is evidence for this after 20 iterations when the Average Change, Maximum Change, and Variance Change values are all increasing, instead of the more stable decreasing change rate seen in the analysis shown. The change rates all stop increasing after 41 iterations, and it is clear by 70 or 80 iterations that one component will account for 100 percent of the transformed variables variance after sufficient iteration. While this may seem desirable (after all, it is a perfect fit), you should, in fact, be on guard when this happens. Whenever convergence is slow, the rates of change increase, or the final data perfectly fit the model, the solution is probably degenerating due to too few constraints on the scorings.

PROC PRINQUAL can account for 100 percent of the variance by scoring Montana and UCLA with one positive value on all variables and scoring all the other teams with one negative value on all variables. This inappropriate analysis suggests that all ranked teams are equally good except for two teams that are less good. Both of these two teams are ranked by only one news service, and their only nonmissing rank is last in the poll. This accounts for the degeneracy.

References

de Boor, C. (1978), *A Practical Guide to Splines*, New York: Springer Verlag.

Carroll, J.D. (1972), "Individual Differences and Multidimensional Scaling," in *Multidimensional Scaling: Theory and Applications in the Behavioral Sciences (Volume 1)*, eds. R.N. Shepard, A.K. Romney, and S.B. Nerlove, New York: Seminar Press.

Eckart, C. and Young, G. (1936), "The Approximation of One Matrix by Another of Lower Rank," *Psychometrika*, 1, 211–218.

Fisher, R. (1938), *Statistical Methods for Research Workers (10th Edition)*, Edinburgh: Oliver and Boyd Press.

Gabriel, K.R. (1981), "Biplot Display of Multivariate Matrices for Inspection of Data and Diagnosis," *Interpreting Multivariate Data*, ed. V. Barnett, London: John Wiley & Sons, Inc.

Gifi, A. (1990), *Nonlinear Multivariate Analysis*, New York: John Wiley & Sons, Inc.

Goodnight, J.H. (1978), SAS Technical Report R-106, *The SWEEP Operator: Its Importance in Statistical Computing*, Cary, NC: SAS Institute Inc.

Harman, H.H. (1976), *Modern Factor Analysis*, Third Edition, Chicago: University of Chicago Press.

Hotelling, H. (1933), "Analysis of a Complex of Statistical Variables into Principal Components," *Journal of Educational Psychology*, 24, 498–520.

Kuhfeld, W.F., Sarle, W.S., and Young, F.W. (1985), "Methods of Generating Model Estimates in the PRINQUAL Macro," *SAS Users Group International Conference Proceedings: SUGI 10*, Cary, NC: SAS Institute Inc., 962–971.

Kruskal, J.B. (1964)," Multidimensional Scaling By Optimizing Goodness of Fit to a Nonmetric Hypothesis," *Psychometrika*, 29, 1–27.

Kruskal, J.B., and Shepard, R.N. (1974), "A Nonmetric Variety of Linear Factor Analysis," *Psychometrika*, 38, 123–157.

de Leeuw, J. (1985), (Personal Conversation).

de Leeuw, J. (1986), "Regression with Optimal Scaling of the Dependent Variable," Department of Data Theory, The University of Leiden, The Netherlands.

van Rijckeveorsel, J. (1982), "Canonical Analysis with B-Splines," in *COMPUSTAT 1982*, Part I, eds. H. Caussinus, P. Ettinger, and R. Tomassone, Vienna: Wein, Physica Verlag.

Sarle, W.S. (1984), (Personal Conversation).

Siegel, S. (1956), *Nonparametric Statistics*, New York: McGraw Hill Book Co.

Smith, P.L. (1979), "Splines as a Useful and Convenient Statistical Tool," *The American Statistician*, 33, 57–62.

Tenenhaus, M. and Vachette, J.L. (1977), "PRINQUAL: Un Programme d'Analyse en Composantes Principales d'un Ensemble de Variables Nominales ou Numeriques," *Les Cahiers de Recherche #68*, CESA, Jouy-en-Josas, France.

Winsberg, S. and Ramsay, J.O. (1983), "Monotone Spline Transformations for Dimension Reduction," *Psychometrika*, 48, 575–595.

Young, F.W. (1981), "Quantitative Analysis of Qualitative Data," *Psychometrika*, 46, 357–388.

Young, F.W., Takane, Y., and de Leeuw, J. (1978), "The Principal Components of Mixed Measurement Level Multivariate Data: An Alternating Least Squares Method with Optimal Scaling Features," *Psychometrika*, 43, 279–281.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS/STAT® User's Guide, Version 8*, Cary, NC: SAS Institute Inc., 1999.

SAS/STAT® User's Guide, Version 8

ISBN 1–58025–494–2

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st printing, October 1999