THE POWER TO KNOW.®

SAS/STAT® 9.2 User's Guide
The CALIS Procedure
(Book Excerpt)

SAS® Documentation

Chapter 25

The CALIS Procedure

Contents

Overview: CALIS Procedure

Structural equation modeling is an important statistical tool in economics and behavioral sciences. Structural equations express relationships among several variables that can be either directly observed variables (manifest variables) or unobserved hypothetical variables (latent variables). For an introduction to latent variable models, refer to Loehlin (1987), Bollen (1989b), Everitt (1984), or Long (1983); and for manifest variables, refer to Fuller (1987).

In structural models, as opposed to functional models, all variables are taken to be random rather than having fixed levels. For maximum likelihood (default) and generalized least squares estimation in PROC CALIS, the random variables are assumed to have an approximately multivariate normal distribution. Nonnormality, especially high kurtosis, can produce poor estimates and grossly incorrect standard errors and hypothesis tests, even in large samples. Consequently, the assumption of normality is much more important than in models with nonstochastic exogenous variables. You should remove outliers and consider transformations of nonnormal variables before using PROC CALIS with maximum likelihood (default) or generalized least squares estimation. If the number of observations is sufficiently large, Browne's asymptotically distribution-free (ADF) estimation method can be used.

You can use the CALIS procedure to estimate parameters and test hypotheses for constrained and unconstrained problems in the following:

- multiple and multivariate linear regression
- linear measurement-error models
- path analysis and causal modeling
- simultaneous equation models with reciprocal causation
- exploratory and confirmatory factor analysis of any order
- canonical correlation
- a wide variety of other (non)linear latent variable models

The parameters are estimated using the following criteria:

- unweighted least squares (ULS)
- generalized least squares (GLS, with optional weight matrix input)
- maximum likelihood (ML, for multivariate normal data)
- weighted least squares (WLS, ADF, with optional weight matrix input)
- diagonally weighted least squares (DWLS, with optional weight matrix input)

The default weight matrix for generalized least squares estimation is the sample covariance or correlation matrix. The default weight matrix for weighted least squares estimation is an estimate of the asymptotic covariance matrix of the sample covariance or correlation matrix. In this case, weighted least squares estimation is equivalent to Browne's (1982, 1984) asymptotic distribution-free estimation. The default weight matrix for diagonally weighted least squares estimation is an estimate of the asymptotic variances of the input sample covariance or correlation matrix. You can also use an input data set to specify the weight matrix in GLS, WLS, and DWLS estimation.

Estimation methods implemented in PROC CALIS do not exhaust all alternatives in the field. For example, partial least squares (PLS) is not implemented. See the section "Estimation Criteria" on page 923 for details about estimation criteria used in PROC CALIS. Note that there is a SAS/STAT procedure called PROC PLS, which employs the partial least squares technique but for a class of models different from those of PROC CALIS. For general path analysis or structural equation model with latent variables you should consider using PROC CALIS.

You can specify the model in several ways:

- If you have a set of structural equations to describe the model, you can use an equation-type LINEQS statement similar to that originally developed by Bentler (1985).
- You can specify simple path models by using an easily formulated list-type RAM statement similar to that originally developed by McArdle (McArdle and McDonald 1984).

- You can do a constrained (confirmatory) first-order factor analysis or component analysis by using the FACTOR statement.
- You can analyze a broad family of matrix models by using COSAN and MATRIX statements that are similar to the COSAN program of McDonald and Fraser (McDonald 1978, 1980). It enables you to specify complex matrix models including nonlinear equation models and higher-order factor models.

You can specify linear and nonlinear equality and inequality constraints on the parameters with several different statements, depending on the type of input. Lagrange multiplier test indices are computed for simple constant and equality parameter constraints and for active boundary constraints. General equality and inequality constraints can be formulated using programming statements. For more information, see the section "SAS Programming Statements" on page 903.

PROC CALIS offers a variety of methods for the automatic generation of initial values for the optimization process:

- two-stage least squares estimation
- instrumental variable factor analysis
- approximate factor analysis
- ordinary least squares estimation
- McDonald's (McDonald and Hartmann 1992) method

In many common applications, these initial values prevent computational problems and save computer time.

Because numerical problems can occur in the (non)linearly constrained optimization process, the CALIS procedure offers several optimization algorithms:

- Levenberg-Marquardt algorithm (Moré, 1978)
- trust-region algorithm (Gay 1983)
- Newton-Raphson algorithm with line search
- ridge-stabilized Newton-Raphson algorithm
- various quasi-Newton and dual quasi-Newton algorithms: Broyden-Fletcher-Goldfarb-Shanno and Davidon-Fletcher-Powell, including a sequential quadratic programming algorithm for processing nonlinear equality and inequality constraints
- various conjugate gradient algorithms: automatic restart algorithm of Powell (1977), Fletcher-Reeves, Polak-Ribiere, and conjugate descent algorithm of Fletcher (1980)

The quasi-Newton and conjugate gradient algorithms can be modified by several line-search methods. All of the optimization techniques can impose simple boundary and general linear constraints on the parameters. Only the dual quasi-Newton algorithm is able to impose general nonlinear equality and inequality constraints.

The procedure creates an OUTRAM= output data set that completely describes the model (except for program statements) and also contains parameter estimates. This data set can be used as input for another execution of PROC CALIS. Small model changes can be made by editing this data set, so you can exploit the old parameter estimates as starting values in a subsequent analysis. An OUTEST= data set contains information about the optimal parameter estimates (parameter estimates, gradient, Hessian, projected Hessian and Hessian of Lagrange function for constrained optimization, the information matrix, and standard errors). The OUTEST= data set can be used as an INEST= data set to provide starting values and boundary and linear constraints for the parameters. An OUTSTAT= data set contains residuals and, for exploratory factor analysis, the rotated and unrotated factor loadings.

Automatic variable selection (using only those variables from the input data set that are used in the model specification) is performed in connection with the RAM and LINEQS input statements or when these models are recognized in an input model file. Also in these cases, the covariances of the exogenous manifest variables are recognized as given constants. With the PREDET option, you can display the predetermined pattern of constant and variable elements in the predicted model matrix before the minimization process starts. For more information, see the section "Automatic Variable Selection" on page 942 and the section "Exogenous Manifest Variables" on page 943.

PROC CALIS offers an analysis of linear dependencies in the information matrix (approximate Hessian matrix) that might be helpful in detecting unidentified models. You also can save the information matrix and the approximate covariance matrix of the parameter estimates (inverse of the information matrix), together with parameter estimates, gradient, and approximate standard errors, in an output data set for further analysis.

PROC CALIS does not provide the analysis of multiple samples with different sample size or a generalized algorithm for missing values in the data. However, the analysis of multiple samples with equal sample size can be performed by the analysis of a moment supermatrix containing the individual moment matrices as block diagonal submatrices.

The new experimental procedure TCALIS is now available. Except for the COSAN model specification, PROC TCALIS supports almost all model specification methods that are available in the CALIS procedure. In addition, there are many new features in PROC TCALIS, including a PATH statement that enables you to specify models by using a path-like syntax, an MSTRUCT statement that enables you to specify patterned covariance structures directly, multiple-group analysis, enhanced mean and covariance structure analysis, a priori parametric function testing, effect analysis with standard error estimates, and more. For more information, see Chapter 88, "The TCALIS Procedure (Experimental)."

The CALIS procedure uses ODS Graphics to create graphs as part of its output. High-quality residual histograms are available in PROC CALIS. See Chapter 21, "Statistical Graphics Using ODS," for general information about ODS Graphics. See the section "ODS Graphics" on page 970 and the PLOTS= option on page 864 for specific information about the statistical graphics available with the CALIS procedure.

Structural Equation Models

The Generalized COSAN Model

PROC CALIS can analyze matrix models of the form

$$\mathbf{C} = \mathbf{F}_1\mathbf{P}_1\mathbf{F}_1' + \cdots + \mathbf{F}_m\mathbf{P}_m\mathbf{F}_m'$$

where $\mathbf{C}$ is a symmetric correlation or covariance matrix, each matrix $\mathbf{F}_k$, $k = 1, \ldots, m$, is the product of $n(k)$ matrices $\mathbf{F}_{k_1}, \ldots, \mathbf{F}_{k_{n(k)}}$, and each matrix $\mathbf{P}_k$ is symmetric; that is,

$$\mathbf{F}_k = \mathbf{F}_{k_1} \cdots \mathbf{F}_{k_{n(k)}} \qquad \text{and} \quad \mathbf{P}_k = \mathbf{P}_k', \quad k = 1, \ldots, m$$

The matrices $\mathbf{F}_{k_j}$ and $\mathbf{P}_k$ in the model are parameterized by the matrices $\mathbf{G}_{k_j}$ and $\mathbf{Q}_k$

$$\mathbf{F}_{k_j} = \begin{cases} \mathbf{G}_{k_j} \\ \mathbf{G}_{k_j}^{-1} \\ (\mathbf{I} - \mathbf{G}_{k_j})^{-1} \end{cases} \quad j = 1, \ldots, n(k) \qquad \text{and} \qquad \mathbf{P}_k = \begin{cases} \mathbf{Q}_k \\ \mathbf{Q}_k^{-1} \end{cases}$$

where you can specify the type of matrix desired.

The matrices $\mathbf{G}_{k_j}$ and $\mathbf{Q}_k$ can contain the following:

- constant values
- parameters to be estimated
- values computed from parameters via programming statements

The parameters can be summarized in a parameter vector $\mathbf{X} = (x_1, \ldots, x_t)$. For a given covariance or correlation matrix $\mathbf{C}$, PROC CALIS computes the unweighted least squares (ULS), generalized least squares (GLS), maximum likelihood (ML), weighted least squares (WLS), or diagonally weighted least squares (DWLS) estimates of the vector $\mathbf{X}$.

Some Special Cases of the Generalized COSAN Model

Original COSAN (Covariance Structure Analysis) Model (McDonald 1978, 1980)

Covariance structure:

$$\mathbf{C} = \mathbf{F}_1 \cdots \mathbf{F}_n\mathbf{P}\mathbf{F}_n' \cdots \mathbf{F}_1'$$

Reticular Action Model—RAM (McArdle 1980; McArdle and McDonald 1984)

Structural equation model:

$$\mathbf{v} = \mathbf{A}\mathbf{v} + \mathbf{u}$$

where $\mathbf{A}$ is a matrix of coefficients, and $\mathbf{v}$ and $\mathbf{u}$ are vectors of random variables. The variables in $\mathbf{v}$ and $\mathbf{u}$ can be manifest or latent variables. The endogenous variables corresponding to the components in $\mathbf{v}$ are expressed as a linear combination of the remaining variables and a residual component in $\mathbf{u}$ with covariance matrix $\mathbf{P}$.

Covariance structure:

$$\mathbf{C} = \mathbf{J}(\mathbf{I} - \mathbf{A})^{-1}\mathbf{P}((\mathbf{I} - \mathbf{A})^{-1})'\mathbf{J}'$$

with selection matrix $\mathbf{J}$ and

$$\mathbf{C} = \mathcal{E}\{\mathbf{J}\mathbf{v}\mathbf{v}'\mathbf{J}'\} \qquad \text{and} \qquad \mathbf{P} = \mathcal{E}\{\mathbf{u}\mathbf{u}'\}$$

LINEQS (Linear Equations) Model (Bentler and Weeks 1980)

Structural equation model:

$$\eta = \beta\eta + \gamma\xi$$

where β and γ are coefficient matrices, and η and ξ are vectors of random variables. The components of η correspond to the endogenous variables; the components of ξ correspond to the exogenous variables and to error variables. The variables in η and ξ can be manifest or latent variables. The endogenous variables in η are expressed as a linear combination of the remaining endogenous variables, the exogenous variables in ξ, and a residual component in ξ. The coefficient matrix β describes the relationships among the endogenous variables of η, and $I - \beta$ should be nonsingular. The coefficient matrix γ describes the relationships between the endogenous variables of η and the exogenous and error variables of ξ.

Covariance structure:

$$\mathbf{C} = \mathbf{J}(\mathbf{I} - \mathbf{B})^{-1}\mathbf{\Gamma}\mathbf{\Phi}\mathbf{\Gamma}'((\mathbf{I} - \mathbf{B})^{-1})'\mathbf{J}'$$

with selection matrix $\mathbf{J}$, $\mathbf{\Phi} = \mathcal{E}\{\xi\xi'\}$, and

$$\mathbf{B} = \begin{pmatrix} \beta & 0 \\ 0 & 0 \end{pmatrix} \qquad \text{and} \qquad \mathbf{\Gamma} = \begin{pmatrix} \gamma \\ \mathbf{I} \end{pmatrix}$$

Keesling-Wiley-Jöreskog LISREL (Linear Structural Relationship) Model (Keesling 1972; Wiley 1973; Jöreskog 1973)

Structural equation model and measurement models:

$$\eta = \mathbf{B}\eta + \mathbf{\Gamma}\xi + \zeta \quad , \qquad \mathbf{y} = \mathbf{\Lambda}_y\eta + \varepsilon \quad , \qquad \mathbf{x} = \mathbf{\Lambda}_x\xi + \delta$$

where η and ξ are vectors of latent variables (factors), and $\mathbf{x}$ and $\mathbf{y}$ are vectors of manifest variables. The components of η correspond to endogenous latent variables; the components of ξ correspond to exogenous latent variables. The endogenous and exogenous latent variables are connected by a system of linear equations (the structural model) with coefficient matrices $\mathbf{B}$ and $\mathbf{\Gamma}$ and an error vector ζ. It is assumed that matrix $\mathbf{I} - \mathbf{B}$ is nonsingular. The random vectors $\mathbf{y}$ and $\mathbf{x}$ correspond to manifest variables that are related to the latent variables η and ξ by two systems of linear equations (the measurement model) with coefficients $\mathbf{\Lambda}_y$ and $\mathbf{\Lambda}_x$ and with measurement errors ε and δ.

Covariance structure:

$$\begin{aligned} \mathbf{C} &= \mathbf{J}(\mathbf{I}-\mathbf{A})^{-1}\mathbf{P}((\mathbf{I}-\mathbf{A})^{-1})'\mathbf{J}' \\ A &= \begin{pmatrix} 0 & 0 & \mathbf{\Lambda}_y & 0 \\ 0 & 0 & 0 & \mathbf{\Lambda}_x \\ 0 & 0 & \mathbf{B} & \mathbf{\Gamma} \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad \text{and} \quad P = \begin{pmatrix} \mathbf{\Theta}_\varepsilon & & & \\ & \mathbf{\Theta}_\delta & & \\ & & \mathbf{\Psi} & \\ & & & \mathbf{\Phi} \end{pmatrix} \end{aligned}$$

with selection matrix $\mathbf{J}$, $\mathbf{\Phi} = \mathcal{E}\{\boldsymbol{\xi}\boldsymbol{\xi}'\}$, $\mathbf{\Psi} = \mathcal{E}\{\boldsymbol{\zeta}\boldsymbol{\zeta}'\}$, $\mathbf{\Theta}_\delta = \mathcal{E}\{\boldsymbol{\delta}\boldsymbol{\delta}'\}$, and $\mathbf{\Theta}_\varepsilon = \mathcal{E}\{\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}'\}$.

Higher-Order Factor Analysis Models

First-order model:

$$\mathbf{C} = \mathbf{F}_1\mathbf{P}_1\mathbf{F}_1' + \mathbf{U}_1^2$$

Second-order model:

$$\mathbf{C} = \mathbf{F}_1\mathbf{F}_2\mathbf{P}_2\mathbf{F}_2'\mathbf{F}_1' + \mathbf{F}_1\mathbf{U}_2^2\mathbf{F}_1' + \mathbf{U}_1^2$$

First-Order Autoregressive Longitudinal Factor Model

Example of McDonald (1980): $k = 3$: Occasions of Measurement; $n = 3$: Variables (Tests); $m = 2$: Common Factors

$$\begin{aligned} \mathbf{C} &= \mathbf{F}_1\mathbf{F}_2\mathbf{F}_3\mathbf{L}\mathbf{F}_3^{-1}\mathbf{F}_2^{-1}\mathbf{P}(\mathbf{F}_2^{-1})'(\mathbf{F}_3^{-1})'\mathbf{L}'\mathbf{F}_3'\mathbf{F}_2'\mathbf{F}_1' + \mathbf{U}^2 \\ F_1 &= \begin{pmatrix} B_1 & & \\ & B_2 & \\ & & B_3 \end{pmatrix}, \quad F_2 = \begin{pmatrix} I_2 & & \\ & D_2 & \\ & & D_2 \end{pmatrix}, \quad F_3 = \begin{pmatrix} I_2 & & \\ & I_2 & \\ & & D_3 \end{pmatrix} \\ L &= \begin{pmatrix} I_2 & o & o \\ I_2 & I_2 & o \\ I_2 & I_2 & I_2 \end{pmatrix}, \quad P = \begin{pmatrix} I_2 & & \\ & S_2 & \\ & & S_3 \end{pmatrix}, \quad U = \begin{pmatrix} U_{11} & U_{12} & U_{13} \\ U_{21} & U_{22} & U_{23} \\ U_{31} & U_{32} & U_{33} \end{pmatrix} \\ S_2 &= I_2 - D_2^2 \quad , \qquad S_3 = I_2 - D_3^2 \end{aligned}$$

For more information about this model, see Example 25.6.

A Structural Equation Example

This example from Wheaton et al. (1977) illustrates the relationships among the RAM, LINEQS, and LISREL models. Different structural models for these data are in Jöreskog and Sörbom (1985) and in Bentler (1985, p. 28). The data set contains covariances among six (manifest) variables collected from 932 people in rural regions of Illinois:

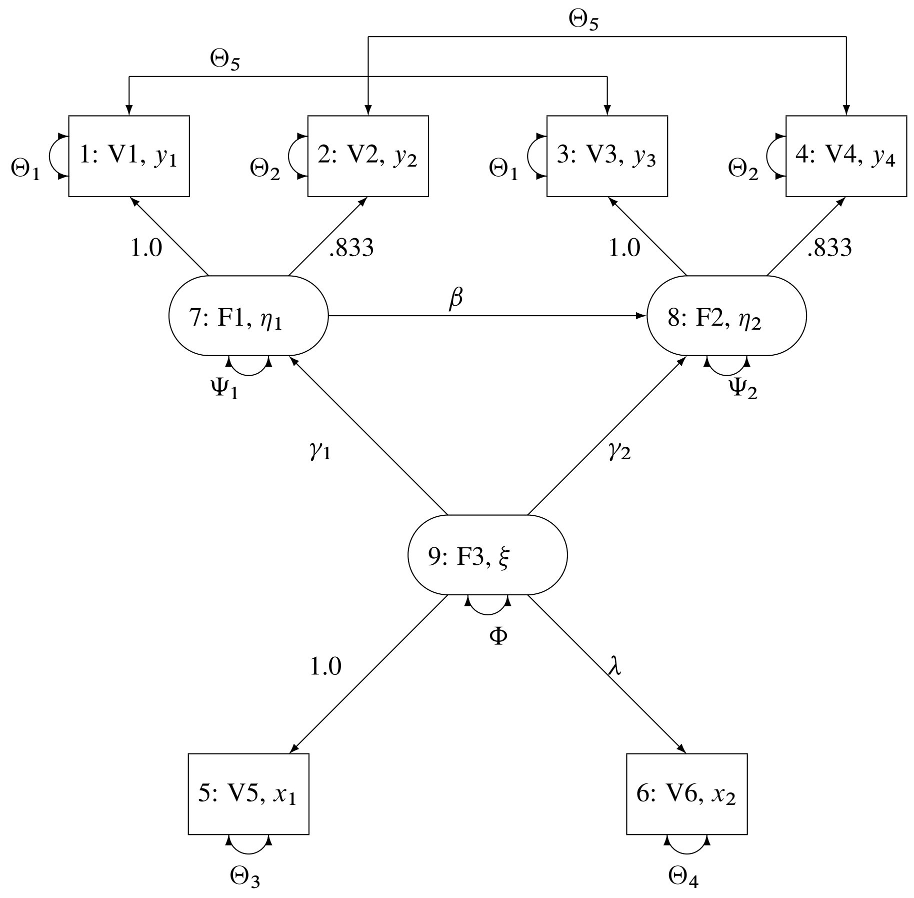
Variable 1:	V1, y_1 : Anomie 1967
Variable 2:	V2, y_2 : Powerlessness 1967
Variable 3:	V3, y_3 : Anomie 1971
Variable 4:	V4, y_4 : Powerlessness 1971
Variable 5:	V5, x_1 : Education (years of schooling)
Variable 6:	V6, x_2 : Duncan's Socioeconomic Index (SEI)

It is assumed that anomie and powerlessness are indicators of an alienation factor and that education and SEI are indicators for a socioeconomic status (SES) factor. Hence, the analysis contains three latent variables:

Variable 7:	F1, η_1 : Alienation 1967
Variable 8:	F2, η_2 : Alienation 1971
Variable 9:	F3, ξ_1 : Socioeconomic Status (SES)

The following path diagram shows the structural model used in Bentler (1985, p. 29) and slightly modified in Jöreskog and Sörbom (1985, p. 56). In this notation for the path diagram, regression coefficients between the variables are indicated as one-headed arrows. Variances and covariances among the variables are indicated as two-headed arrows. Indicating error variances and covariances as two-headed arrows with the same source and destination (McArdle 1988; McDonald 1985) is helpful in transforming the path diagram to RAM model list input for the CALIS procedure.

Figure 25.1 Path Diagram of Stability and Alienation Example



Variables in Figure 25.1 are as follows:

Variable 1:	V1, y_1 : Anomie 1967
Variable 2:	V2, y_2 : Powerlessness 1967
Variable 3:	V3, y_3 : Anomie 1971
Variable 4:	V4, y_4 : Powerlessness 1971
Variable 5:	V5, x_1 : Education (years of schooling)
Variable 6:	V6, x_2 : Duncan's Socioeconomic Index (SEI)
Variable 7:	F1, η_1 : Alienation 1967
Variable 8:	F2, η_2 : Alienation 1971
Variable 9:	F3, ξ_1 : Socioeconomic Status (SES)

LINEQS Model

The vector $\boldsymbol{\eta}$ contains the six endogenous manifest variables V1, …, V6 and the two endogenous latent variables F1 and F2. The vector $\boldsymbol{\xi}$ contains the exogenous error variables E1, …, E6, D1, and D2 and the exogenous latent variable F3. The path diagram corresponds to the following set of structural equations of the LINEQS model:

$$\begin{aligned}
V1 &= 1.000F1 + E1 \\
V2 &= 0.833F1 + E2 \\
V3 &= 1.000F2 + E3 \\
V4 &= 0.833F2 + E4 \\
V5 &= 1.000F3 + E5 \\
V6 &= \lambda F3 + E6 \\
F1 &= \gamma_1 F3 + D1 \\
F2 &= \beta F1 + \gamma_2 F3 + D2
\end{aligned}$$

This gives the matrices $\boldsymbol{\beta}$, $\boldsymbol{\gamma}$, and $\boldsymbol{\Phi}$ in the LINEQS model:

$$\boldsymbol{\beta} = \begin{pmatrix}
o & o & o & o & o & o & 1.000 & o \\
o & o & o & o & o & o & 0.833 & o \\
o & o & o & o & o & o & o & 1.000 \\
o & o & o & o & o & o & o & 0.833 \\
o & o & o & o & o & o & o & o \\
o & o & o & o & o & o & o & o \\
o & o & o & o & o & o & o & o \\
o & o & o & o & o & o & \beta & o
\end{pmatrix}, \quad
\boldsymbol{\gamma} = \begin{pmatrix}
1 & o & o & o & o & o & o & o & o \\
o & 1 & o & o & o & o & o & o & o \\
o & o & 1 & o & o & o & o & o & o \\
o & o & o & 1 & o & o & o & o & o \\
o & o & o & o & 1 & o & o & o & 1.000 \\
o & o & o & o & o & 1 & o & o & \lambda \\
o & o & o & o & o & o & 1 & o & \gamma_1 \\
o & o & o & o & o & o & o & 1 & \gamma_2
\end{pmatrix}$$

$$\boldsymbol{\Phi} = \begin{pmatrix}
\theta_1 & o & \theta_5 & o & o & o & o & o & o \\
o & \theta_2 & o & \theta_5 & o & o & o & o & o \\
\theta_5 & o & \theta_1 & o & o & o & o & o & o \\
o & \theta_5 & o & \theta_2 & o & o & o & o & o \\
o & o & o & o & \theta_3 & o & o & o & o \\
o & o & o & o & o & \theta_4 & o & o & o \\
o & o & o & o & o & o & \psi_1 & o & o \\
o & o & o & o & o & o & o & \psi_2 & o \\
o & o & o & o & o & o & o & o & \phi
\end{pmatrix}$$

The LINEQS model input specification of this example for the CALIS procedure is given in the section “LINEQS Model Specification” on page 841.

RAM Model

The vector **v** contains the six manifest variables v_1 =V1, …, v_6 =V6 and the three latent variables v_7 =F1, v_8 =F2, v_9 =F3. The vector **u** contains the corresponding error variables u_1 =E1, …, u_6 =E6 and u_7 =D1, u_8 =D2, u_9 =D3. The path diagram corresponds to the following set of structural equations of the RAM model:

$$\begin{aligned}
v_1 &= 1.000v_7 + u_1 \\
v_2 &= 0.833v_7 + u_2 \\
v_3 &= 1.000v_8 + u_3 \\
v_4 &= 0.833v_8 + u_4 \\
v_5 &= 1.000v_9 + u_5 \\
v_6 &= \lambda v_9 + u_6 \\
v_7 &= \gamma_1 v_9 + u_7 \\
v_8 &= \beta v_7 + \gamma_2 v_9 + u_8 \\
v_9 &= u_9
\end{aligned}$$

This gives the matrices **A** and **P** in the RAM model:

$$\mathbf{A} = \begin{pmatrix}
o & o & o & o & o & o & 1.000 & o & o \\
o & o & o & o & o & o & 0.833 & o & o \\
o & o & o & o & o & o & o & 1.000 & o \\
o & o & o & o & o & o & o & 0.833 & o \\
o & o & o & o & o & o & o & o & 1.000 \\
o & o & o & o & o & o & o & o & \lambda \\
o & o & o & o & o & o & o & o & \gamma_1 \\
o & o & o & o & o & o & \beta & o & \gamma_2 \\
o & o & o & o & o & o & o & o & o
\end{pmatrix}$$

$$\mathbf{P} = \begin{pmatrix}
\theta_1 & o & \theta_5 & o & o & o & o & o & o \\
o & \theta_2 & o & \theta_5 & o & o & o & o & o \\
\theta_5 & o & \theta_1 & o & o & o & o & o & o \\
o & \theta_5 & o & \theta_2 & o & o & o & o & o \\
o & o & o & o & \theta_3 & o & o & o & o \\
o & o & o & o & o & \theta_4 & o & o & o \\
o & o & o & o & o & o & \psi_1 & o & o \\
o & o & o & o & o & o & o & \psi_2 & o \\
o & o & o & o & o & o & o & o & \phi
\end{pmatrix}$$

The RAM model input specification of this example for the CALIS procedure is given in the section "RAM Model Specification" on page 842.

LISREL Model

The vector **y** contains the four endogenous manifest variables y_1 =V1, …, y_4 =V4, and the vector **x** contains the exogenous manifest variables x_1 =V5 and x_2 =V6. The vector $\boldsymbol{\varepsilon}$ contains the error variables ε_1 =E1, …, ε_4 =E4 corresponding to **y**, and the vector $\boldsymbol{\delta}$ contains the error variables δ_1 =E5 and δ_2 =E6 corresponding to **x**. The vector $\boldsymbol{\eta}$ contains the endogenous latent variables (factors) η_1 =F1 and η_2 =F2, while the vector $\boldsymbol{\xi}$ contains the exogenous latent variable (factor) ξ_1 =F3. The vector $\boldsymbol{\zeta}$ contains the errors ζ_1 =D1 and ζ_2 =D2 in the equations (disturbance terms) corresponding to $\boldsymbol{\eta}$. The path diagram corresponds to the following set of structural equations of the LISREL model:

$$\begin{aligned}
y_1 &= 1.000\eta_1 + \epsilon_1 \\
y_2 &= 0.833\eta_1 + \epsilon_2 \\
y_3 &= 1.000\eta_2 + \epsilon_3 \\
y_4 &= 0.833\eta_2 + \epsilon_4 \\
x_1 &= 1.000\xi_1 + \delta_1 \\
x_2 &= \lambda\xi_1 + \delta_2 \\
\eta_1 &= \gamma_1\xi_1 + \zeta_1 \\
\eta_2 &= \beta\eta_1 + \gamma_2\xi_1 + \zeta_2
\end{aligned}$$

This gives the matrices $\boldsymbol{\Lambda}_y$, $\boldsymbol{\Lambda}_x$, $\mathbf{B}$, $\boldsymbol{\Gamma}$, and $\boldsymbol{\Phi}$ in the LISREL model:

$$\boldsymbol{\Lambda}_y = \begin{pmatrix} 1.000 & o \\ 0.833 & o \\ o & 1.000 \\ o & 0.833 \end{pmatrix}, \boldsymbol{\Lambda}_x = \begin{pmatrix} 1. \\ \lambda \end{pmatrix}, \mathbf{B} = \begin{pmatrix} o & o \\ \beta & o \end{pmatrix}, \boldsymbol{\Gamma} = \begin{pmatrix} \gamma_1 \\ \gamma_2 \end{pmatrix}$$

$$\boldsymbol{\Theta}_\varepsilon^2 = \begin{pmatrix} \theta_1 & o & \theta_5 & o \\ o & \theta_2 & o & \theta_5 \\ \theta_5 & o & \theta_1 & o \\ o & \theta_5 & o & \theta_2 \end{pmatrix}, \boldsymbol{\Theta}_\delta^2 = \begin{pmatrix} \theta_3 & o \\ \theta_4 & o \end{pmatrix}, \boldsymbol{\Psi} = \begin{pmatrix} \psi_1 & o \\ o & \psi_2 \end{pmatrix}, \boldsymbol{\Phi} = (\phi)$$

The CALIS procedure does not provide a LISREL model input specification. However, any model that can be specified by the LISREL model can also be specified by using the LINEQS, RAM, or COSAN model specifications in PROC CALIS.

Getting Started: CALIS Procedure

Statements in the CALIS procedure are used for various purposes:

- PROC CALIS statement invokes the procedure.
- Main model specification statements: LINEQS, RAM, FACTOR, and COSAN. These statements define the model type and specify the main model.
- Subsidiary model specification statements: STD, COV, VARNAMES, MATRIX, and PARAMETERS. These statements define additional parameters and model features that supplement to the main model specification. The usage of these statements depends on the model type specified in the main model statement. See the section "The Model Type and Related Statements" on page 840 for more details.

- Statements for constraining parameters: BOUNDS, LINCON, and NLINCON. The SAS programming statements can also be used to constrain parameters. See the section “Constrained Estimation” on page 845 for more details.
- General data processing statements: BY, VAR, PARTIAL, FREQ, and WEIGHT. The usages of these statements are common to other SAS procedures.
- Other statements: The STRUCTEQ statement is used for defining “structural equations” in the model. The NLOPTIONS statement is used for fine-tuning the optimization techniques for the analysis. The SAS programming statements are used for computing dependent parametric functions that are useful for model specification or setting up constraints in the model.

The Model Type and Related Statements

There are four sets of statements available in the CALIS procedure to specify a model. Each set represents a model type supported by the CALIS procedure. For the LINEQS model with linear equations input, you can specify the following statements:

LINEQS *analysis model in equations notation*;
  STD *variance pattern*;
  COV *covariance pattern*;
  PARAMETERS *parameter names from programming statements*;

For the RAM model, you can specify the following statements:

RAM *analysis model in list notation*;
  VARNAMES *names of latent and error variables*;
  PARAMETERS *parameter names from programming statements*;

For the (confirmatory) factor model, you can specify the following statements:

FACTOR *options*;
  MATRIX *definition of matrix elements*;
  VARNAMES *names of latent and residual variables*;
  PARAMETERS *parameter names from programming statements*;

For the COSAN model, you can specify the following statements:

COSAN *analysis model in matrix notation*;
  MATRIX *definition of matrix elements* ;
  VARNAMES *names of additional variables* ;
  PARAMETERS *parameter names from programming statements*;

You can also specify a model by using an INRAM= data set, which is usually a version of an OUTRAM= data set produced by a previous PROC CALIS analysis (and possibly modified). If no INRAM= data set is specified, you must use one of the four main model specification statements that defines the analysis model: LINEQS, RAM, FACTOR, or COSAN.

LISREL model input is not directly supported in PROC CALIS. However, with careful constructions you can use either the LINEQS, RAM, or COSAN model input to specify a LISREL model equivalently.

In the following sections, each model type supported by the CALIS procedure is discussed in more details.

LINEQS Model Specification

By using notation similar to that originally developed by Bentler for his EQS program, you can also describe the model by a set of linear equations combined with variance and covariance specifications. The displayed output can be in either equation form or matrix form.

The following statements define the structural model of the alienation example as a LINEQS model:

```
lineqs
     V1 =         F1                   + E1,
     V2 =    .833 F1                   + E2,
     V3 =         F2                   + E3,
     V4 =    .833 F2                   + E4,
     V5 =         F3                   + E5,
     V6 = Lamb (.5) F3                 + E6,
     F1 = Gam1(-.5) F3                 + D1,
     F2 = Beta (.5) F1 + Gam2(-.5) F3 + D2;
std
     E1-E6 = The1-The2 The1-The4 (6 * 3.),
     D1-D2 = Psi1-Psi2 (2 * 4.),
     F3    = Phi (6.) ;
cov
     E1 E3 = The5 (.2),
     E4 E2 = The5 (.2);
```

The LINEQS statement shows the equations in the section "LINEQS Model" on page 837, except that in this case the coefficients to be estimated can be followed (optionally) by the initial value to use in the optimization process. If you do not specify initial values for the parameters in a LINEQS statement, PROC CALIS tries to assign these values automatically. The endogenous variables used on the left side can be manifest variables (with names that must be defined by the input data set) or latent variables (which must have names starting with F). The variables used on the right side can be manifest variables, latent variables (with names that must start with an F), or error variables (which must have names starting with an E or D). Commas separate the equations. The coefficients to be estimated are indicated by names. If no name is used, the coefficient is constant, either equal to a specified number or, if no number is used, equal to 1. The VAR statement in Bentler's notation is replaced here by the STD statement, because the VAR statement in PROC CALIS defines the subset of manifest variables in the data set to be analyzed. The variable names used in the STD or COV statement must be exogenous (that is, they should not occur on the left side of any equation). The STD and COV statements define the diagonal and off-diagonal elements in the $\mathbf{\Phi}$ matrix. The parameter specifications in the STD and COV statements are separated by commas. Using k variable names on the left of an equal sign in a COV statement means that the parameter list on the right side refers to all $k(k-1)/2$ distinct variable pairs in the $\mathbf{\Phi}$ matrix. Identical coefficient names indicate parameters constrained to be equal. You can also use prefix names to specify those parameters for which you do not need a precise name in any parameter constraint.

See the section "LINEQS Model Statement" on page 881 for more information about the precise syntax rules for a LINEQS statement.

RAM Model Specification

The RAM model enables a path diagram to be transcribed into a RAM statement in list form. The displayed output from the RAM statement is in matrix or list form.

The following statement defines the structural model of the alienation example as a RAM model:

```
ram
     1   1  7  1.       ,
     1   2  7  .833     ,
     1   3  8  1.       ,
     1   4  8  .833     ,
     1   5  9  1.       ,
     1   6  9  .5    Lamb ,
     1   7  9  -.5   Gam1 ,
     1   8  7  .5    Beta ,
     1   8  9  -.5   Gam2 ,
     2   1  1  3.    The1 ,
     2   2  2  3.    The2 ,
     2   3  3  3.    The1 ,
     2   4  4  3.    The2 ,
     2   5  5  3.    The3 ,
     2   6  6  3.    The4 ,
     2   1  3  .2    The5 ,
     2   2  4  .2    The5 ,
     2   7  7  4.    Psi1 ,
     2   8  8  4.    Psi2 ,
     2   9  9  6.    Phi ;
```

You must assign numbers to the nodes in the path diagram. In the path diagram of Figure 25.1, the boxes corresponding to the six manifest variables V1, . . . , V6 are assigned the number of the variable in the covariance matrix (1, . . . , 6); the circles corresponding to the three latent variables F1, F2, and F3 are given the numbers 7, 8, and 9. The path diagram contains 20 paths between the nine nodes; nine of the paths are one-headed arrows and eleven are two-headed arrows.

The RAM statement contains a list of items separated by commas. Each item corresponds to an arrow in the path diagram. The first entry in each item is the number of arrow heads (matrix number), the second entry shows where the arrow points to (row number), the third entry shows where the arrow comes from (column number), the fourth entry gives the (initial) value of the coefficient, and the fifth entry assigns a name if the path represents a parameter rather than a constant. If you specify the fifth entry as a parameter name, then the fourth list entry can be omitted, since PROC CALIS tries to assign an initial value to this parameter.

See the section "RAM Model Statement" on page 899 for more information about the RAM statement.

COSAN Model Specification

You specify the model for a generalized COSAN analysis with a COSAN statement and one or more MATRIX statements. The COSAN statement determines the name, dimension, and type (identity, diagonal, symmetric, upper, lower, general, inverse, and so forth) of each matrix in the model. You can specify the values of the constant elements in each matrix and give names and initial values to the elements that are to be estimated as parameters or functions of parameters by using MATRIX statements. The resulting displayed output is in matrix form.

The following statements define the structural model of the alienation example as a COSAN model:

```
cosan J(9, Ide) * A(9, Gen, Imi) * P(9, Sym);
matrix A
          [ ,7] = 1. .833  5 * 0. Beta (.5) ,
          [ ,8] = 2 * 0.  1.  .833 ,
          [ ,9] = 4 * 0.  1.  Lamb Gam1-Gam2 (.5 2 * -.5);
matrix P
          [1,1] = The1-The2 The1-The4 (6 * 3.) ,
          [7,7] = Psi1-Psi2 Phi (2 * 4. 6.) ,
          [3,1] = The5 (.2) ,
          [4,2] = The5 (.2) ;
```

The matrix model specified in the COSAN statement is the RAM model

$$\mathbf{C} = \mathbf{J}(\mathbf{I} - \mathbf{A})^{-1}\mathbf{P}((\mathbf{I} - \mathbf{A})^{-1})'\mathbf{J}'$$

with selection matrix $\mathbf{J}$ and

$$\mathbf{C} = \mathcal{E}\{\mathbf{J}\mathbf{v}\mathbf{v}'\mathbf{J}'\}, \qquad \mathbf{P} = \mathcal{E}\{\mathbf{u}\mathbf{u}'\}$$

The COSAN statement must contain only the matrices up to the central matrix $\mathbf{P}$ because of the symmetry of each matrix term in a COSAN model. Each matrix name is followed by one to three arguments in parentheses. The first argument is the number of columns. The second and third arguments are optional, and they specify the form of the matrix. The selection matrix $\mathbf{J}$ in the RAM model is specified by the 6×9 identity (IDE) (sub)matrix $\mathbf{J}$ because the first six variables in vector $\mathbf{v}$ correspond to the six manifest variables in the data set. The 9×9 parameter matrix $\mathbf{A}$ has a general (GEN) form and is used as $(\mathbf{I} - \mathbf{A})^{-1}$ in the analysis, as indicated by the identity-minus-inverse (IMI) argument. The central 9×9 matrix P is specified as a symmetric (SYM) matrix.

The MATRIX statement for matrix $\mathbf{A}$ specifies the values in columns 7, 8, and 9, which correspond to the three latent variables F1, F2, and F3, in accordance with the RAM model. The other columns of $\mathbf{A}$ are assumed to be zero. The initial values for the parameter elements in $\mathbf{A}$ are chosen as in the path diagram to be

$$\lambda = \beta = 0.5, \qquad \gamma_1 = \gamma_2 = -0.5$$

In accordance with matrix $\mathbf{P}$ of the RAM model and the path model, the nine diagonal elements of matrix $\mathbf{P}$ are parameters with initial values

$$\theta_1 = \theta_2 = \theta_3 = \theta_4 = 3, \quad \psi_1 = \psi_2 = 4, \quad \phi = 6$$

There are also two off-diagonal elements in each triangle of $\mathbf{P}$ that are constrained to be equal, and they have an initial value of 0.2.

See the section "COSAN Model Statement" on page 873 for more information about the COSAN statement.

FACTOR Model Specification

You can specify the FACTOR statement to compute factor loadings **F** and unique variances **U** of an exploratory or confirmatory first-order factor (or component) analysis. By default, the factor correlation matrix **P** is an identity matrix.

$$\mathbf{C} = \mathbf{F}\mathbf{F}' + \mathbf{U}, \quad \mathbf{U} = diag$$

For a first-order confirmatory factor analysis, you can use MATRIX statements to define elements in the matrices **F**, **P**, and **U** of the more general model

$$\mathbf{C} = \mathbf{F}\mathbf{P}\mathbf{F}' + \mathbf{U}, \quad \mathbf{P} = \mathbf{P}', \quad \mathbf{U} = diag$$

To perform a component analysis, specify the COMPONENT option to constrain the matrix **U** to a zero matrix; that is, the model is replaced by

$$\mathbf{C} = \mathbf{F}\mathbf{F}'$$

Note that the rank of $\mathbf{F}\mathbf{F}'$ is equal to the number m of components in **F**, and if m is smaller than the number of variables in the moment matrix **C**, the matrix of predicted model values is singular and maximum likelihood estimates for **F** cannot be computed. You should compute ULS estimates in this case.

The HEYWOOD option constrains the diagonal elements of **U** to be nonnegative; that is, the model is replaced by

$$\mathbf{C} = \mathbf{F}\mathbf{F}' + \mathbf{U}^2, \quad \mathbf{U} = diag$$

If the factor loadings are unconstrained, they can be orthogonally rotated by one of the following methods:

- principal axes rotation
- quartimax
- varimax
- equamax
- parsimax

The most common approach to factor analysis consists of two steps:

1. Obtain estimates for factor loadings and unique variances.
2. Apply an orthogonal or oblique rotation method.

PROC CALIS enables you to specify general linear and nonlinear equality and inequality constraints by using the LINCON and NLINCON statements. You can specify the NLINCON statement to estimate orthogonal or oblique rotated factor loadings; refer to Browne and Du Toit (1992). Unlike PROC FACTOR, PROC CALIS currently does not provide standard errors for the rotated factor loadings.

PROC CALIS also computes the factor score regression coefficients for the final factor solution.

For default (exploratory) factor analysis, PROC CALIS computes initial estimates. If you use a MATRIX statement together with a FACTOR model specification, initial values are generally computed by McDonald's (McDonald and Hartmann 1992) method or are set by the START= option. See the section "FACTOR Model Statement" on page 877 and Example 25.3 for more information about the FACTOR statement.

Constrained Estimation

- Simple equality constraints, $x_i = c_i,\ c_i = const,$ and $x_i = x_j,$ can be defined in each model by specifying constants or using the same name for parameters constrained to be equal.

- BOUNDS statement: You can specify boundary constraints, $l_i \le x_i \le u_i,\ l_i,\ u_i = const,$ with the BOUNDS statement for the COSAN, LINEQS, and RAM models and in connection with an INRAM= data set. There might be serious convergence problems if negative values appear in the diagonal locations (variances) of the central model matrices during the minimization process. You can use the BOUNDS statement to constrain these parameters to have nonnegative values.

- LINCON statement: You can specify general linear equality and inequality constraints of the parameter estimates with the LINCON statement or by using an INEST= data set. The variables listed in the LINCON statements must be (a subset of) the model parameters. All optimization methods can be used with linear constraints.

- NLINCON statement: You can specify general nonlinear equality and inequality constraints of the parameter estimates with the NLINCON statement. The syntax of the NLINCON statement is almost the same as that for the BOUNDS statement with the exception that the BOUNDS statement can contain only names of the model parameters. However, the variables listed in the NLINCON statement can be defined by programming statements. Only the quasi-Newton optimization method can be used when there are nonlinear constraints.

- Reparameterizing the model: Complex linear equality and inequality constraints can be defined by means of programming statements similar to those used in the DATA step. In this case, some of the parameters x_i are not elements of the matrices $\mathbf{G}_{kj}$ and $\mathbf{Q}_k$ but are instead defined in a PARAMETERS statement. Elements of the model matrices can then be computed by programming statements as functions of parameters in the PARAMETERS statement. This approach is similar to the classical COSAN program of McDonald, implemented by Fraser (McDonald 1978, 1980). One advantage of the CALIS procedure is that you need not supply code for the derivatives of the specified functions. The analytic derivatives of the user-written functions are computed automatically by PROC CALIS. The specified functions must be continuous and have continuous first-order partial derivatives. See the section "SAS Programming Statements" on page 903 and the section "Constrained Estimation by

Using Program Code" on page 955 for more information about imposing linear and nonlinear restrictions on parameters by using programming statements.

Although much effort has been made to implement reliable and numerically stable optimization methods, no practical algorithm exists that can always find the global optimum of a nonlinear function, especially when there are nonlinear constraints.

Syntax: CALIS Procedure

The following statements are available in PROC CALIS:

PROC CALIS < *options* > **;**
  LINEQS *model equations* **;**
  STD *variance pattern* **;**
  COV *covariance pattern* **;**
  RAM *model list* **;**
  VARNAMES *variables* **;**
  FACTOR < *options* > **;**
  COSAN *matrix model* **;**
  MATRIX *matrix elements* **;**
  PARAMETERS *parameters* **;**
  STRUCTEQ *variables* **;**
  BOUNDS *boundary constraints* **;**
  LINCON *linear constraints* **;**
  NLINCON *nonlinear constraints* **;**
  NLOPTIONS *optimization options* **;**
  BY *variables* **;**
  VAR *variables* **;**
  PARTIAL *variables* **;**
  FREQ *variable* **;**
  WEIGHT *variable* **;**
  SAS programming statements ;

- If no INRAM= data set is specified, one of the four statements that define the input form of the analysis model, LINEQS, RAM, FACTOR, or COSAN, must be used.
- The MATRIX statement can be used multiple times for the same or different matrices along with a COSAN or FACTOR statement. If the MATRIX statement is used multiple times for the same matrix, later definitions override earlier ones.
- The STD and COV statements can be used only with the LINEQS model statement.
- You can formulate a generalized COSAN model by using a COSAN statement. MATRIX statements can be used to define the elements of a matrix used in the COSAN statement. The input notation resembles the COSAN program of McDonald and Fraser (McDonald 1978, 1980).

- The RAM statement uses a simple list input that is especially suitable for describing McArdle's RAM analysis model (McArdle 1980; McArdle and McDonald 1984) for causal and path analysis problems.

- The LINEQS statement formulates the analysis model by means of a system of linear equations similar to Bentler's (1989) EQS program notation. The STD and COV statements can be used to define the variances and covariances corresponding to elements of matrix $\mathbf{\Phi}$ in the LINEQS model.

- A FACTOR statement can be used to compute a first-order exploratory or confirmatory factor (or component) analysis. The analysis of a simple exploratory factor analysis model performed by PROC CALIS is not as efficient as one performed by the FACTOR procedure. The CALIS procedure is designed for more general structural problems, and it needs significantly more computation time for a simple unrestricted factor or component analysis than PROC FACTOR does.

- You can add SAS programming statements to impose linear or nonlinear constraints on the parameters if you specify the model by means of a COSAN, LINEQS, or RAM statement. The PARAMETERS statement defines additional parameters that are needed as independent variables in your program code and that belong to the set of parameters to be estimated. Variable names used in the program code should differ from the preceding statement names. The code should respect the syntax rules of SAS statements usually used in the DATA step. See the section "SAS Programming Statements" on page 903 for more information.

- The BOUNDS statement can be used to specify simple lower and upper boundary constraints for the parameters.

- You can specify general linear equality and inequality constraints with the LINCON statement (or via an INEST= data set). The NLINCON statement can be used to specify general nonlinear equality and inequality constraints by referring to nonlinear functions defined by programming statements.

- The VAR, PARTIAL, WEIGHT, FREQ, and BY statements can be used in the same way as in other procedures, such as the FACTOR or PRINCOMP procedure. You can select a subset of the input variables to analyze with the VAR statement. The PARTIAL statement defines a set of input variables that are chosen as partial variables for the analysis of a matrix of partial correlations or covariances. The BY statement specifies groups in which separate covariance structure analyses are performed.

The following sections describe the PROC CALIS statement and then describe the other statements in alphabetical order.

PROC CALIS Statement

PROC CALIS < *options* > **;**

The options available with the PROC CALIS statement are listed in the following table and then are described in alphabetical order.

Table 25.1 Options Available in the PROC CALIS Statement

Option	Description
Data Set Options	
DATA=	specifies the input data set
INEST=	inputs the initial values and constraints
INRAM=	inputs the model specifications
INWGT=	specifies the input weight matrix
OUTEST=	outputs the covariance matrix of estimates
OUTJAC	outputs the Jacobian into the OUTEST= data set
OUTRAM=	outputs the model specifications
OUTSTAT=	outputs the statistical results
OUTWGT=	outputs the weight matrix
Data Processing	
AUGMENT	analyzes augmented moment matrix
COVARIANCE	analyzes covariance matrix
EDF=	defines the number of observations by the error degrees of freedom
NOBS=	defines the number of observations
NOINT	analyzes uncorrected moments
RDF=	specifies regression degrees of freedom for modifying the number of observations
RIDGE	specifies ridge factor for the covariance matrix
UCORR	analyzes uncorrected correlation matrix
UCOV	analyzes uncorrected covariance matrix
VARDEF=	specifies the method for computing the variance divisor
Estimation Methods	
ASYCOV=	specifies the formula for computing asymptotic covariances
DFREDUCE=	reduces the degrees of freedom for model fit chi-square test
G4=	specifies the algorithm for computing standard errors
METHOD=	specifies the estimation method
NODIAG	excludes the diagonal elements of the covariance matrix from model fitting
WPENALTY=	specifies the penalty weight to fit correlations
WRIDGE=	specifies the ridge factor for weight matrix
Statistical Analysis	
ALPHAECV=	specifies the α level for computing the confidence interval of ECV (Browne and Cudeck 1993)

Table 25.1 *continued*

Option	Description
ALPHARMS=	specifies the α level for computing the confidence interval of RMSEA (Steiger and Lind 1980)
BIASKUR	computes the skewness and kurtosis without bias corrections
KURTOSIS	computes and displays kurtosis
MODIFICATION	computes modification indices
NOMOD	suppresses modification indices
NOSTDERR	suppresses standard error computations
PCOVES	displays the covariance matrix of estimates
PDETERM	computes the determination coefficients
PLATCOV	computes the latent variable covariances and score coefficients
PREDET	displays predetermined moment matrix
RESIDUAL=	specifies the type of residuals being computed
SIMPLE	prints univariate statistics
SLMW=	specifies the probability limit for Wald tests
STDERR	computes the standard errors
TOTEFF	displays total and indirect effects
ODS Graphics	
PLOTS=	specifies ODS Graphics selection
Control Display Output	
NOPRINT	suppresses the displayed output
PALL	displays all output (ALL)
PCORR	displays analyzed and estimated moment matrix
PESTIM	prints parameter estimates
PINITIAL	prints initial pattern and values
PJACPAT	displays structure of variable and constant elements of the Jacobian matrix
PRIMAT	displays output in matrix form
PRINT	adds default displayed output
PRIVEC	displays output in vector form
PSHORT	reduces default output (SHORT)
PSUMMARY	displays fit summary only (SUMMARY)
PWEIGHT	displays the weight matrix
Optimization Techniques	
FCONV=	specifies the objective function convergence criterion
GCONV=	specifies the gradient convergence criterion
INSTEP=	specifies the initial step length (RADIUS=, SALPHA=)
LINESEARCH=	specifies the line-search method
LSPRECISION=	specifies the line-search precision (SPRECISION=)
MAXFUNC=	specifies the maximum number of function calls
MAXITER=	specifies the maximum number of iterations
TECHNIQUE=	specifies the minimization method
UPDATE=	specifies the update technique

Table 25.1 *continued*

Option	Description
Numerical Properties	
ASINGULAR=	specifies the absolute singularity information matrix
COVSING=	specifies the singularity tolerance of information matrix
MSINGULAR=	specifies the relative M singularity of information matrix
SINGULAR=	specifies the singularity criterion
VSINGULAR=	specifies the relative V singularity of information matrix
Miscellaneous	
DEMPHAS=	emphasizes the diagonal entries
FDCODE	uses numeric derivatives for programming code
HESSALG=	specifies the algorithm for computing the Hessian
NOADJDF	requests no degrees of freedom adjustment be made for active constraints
RANDOM=	specifies the seed for randomly generated initial values
START=	specifies the constant for the initial values

Options for General Output Display

Some output display control options enable or disable more than a single display. These are general output display control options, which include the PALL, PRINT, PSHORT, PSUMMARY, and NOPRINT options. If the NOPRINT option is not specified, a default set of output is displayed. The PRINT and PALL options add more output to the default set of output, while the PSHORT and PSUMMARY options reduce from the set of default output.

The relationships of the general output display control options (excluding the NOPRINT option) with other specific options or displays are summarized in the following table.

Output Options	**PALL**	**PRINT**	**default**	**PSHORT**	**PSUMMARY**
fit indices	*	*	*	*	*
linear dependencies	*	*	*	*	*
iteration history	*	*	*	*	
model matrices	*	*	*	*	
PESTIM	*	*	*	*	
PREDET	*	(*)	(*)	(*)	
PINITIAL	*	*	*		
SIMPLE	*	*	*		
STDERR	*	*	*		
RESIDUAL	*	*			
KURTOSIS	*	*			
PLATCOV	*	*			
TOTEFF	*	*			
PCORR	*				
MODIFICATION	*				
PWEIGHT	*				
PCOVES					
PDETERM					
PJACPAT					
PRIMAT					
PRIVEC					

Each "*" in the table represents a specific display or a set of displays enabled by the general output display control option. For example, if you specify the PSUMMARY option, the fit indices and the linear dependencies of parameter estimates (if present) will be shown in the printed output. With the PSHORT option, iteration history and model matrices will also be printed. In addition, the PESTIM option and the PREDET option are also enabled by the PSHORT option. Entries with "(*)" represent specific conditions that enable the PREDET option. See the PREDET option for more details.

Alphabetical Listing of Options with the PROC CALIS Statement

ALPHAECV=α

specifies the significance level for a $1-\alpha$ confidence interval, $0 \leq \alpha \leq 1$, for the Browne and Cudeck (1993) expected cross validation index (ECVI). The default value is $\alpha = 0.1$, which corresponds to a 90% confidence interval for the ECVI.

ALPHARMS=*α*

specifies the significance level for a $1 - \alpha$ confidence interval, $0 \leq \alpha \leq 1$, for the Steiger and Lind (1980) root mean squared error of approximation (RMSEA) coefficient (refer to Browne and Du Toit 1992). The default value is $\alpha = 0.1$, which corresponds to a 90% confidence interval for the RMSEA.

ASINGULAR | ASING=*r*

specifies an absolute singularity criterion r, $r > 0$, for the inversion of the information matrix, which is needed to compute the covariance matrix. The following singularity criterion is used:

$$|d_{j,j}| \leq \max(ASING, VSING * |H_{j,j}|, MSING * \max(|H_{1,1}|, \ldots, |H_{n,n}|))$$

In the preceding criterion, $d_{j,j}$ is the diagonal pivot of the matrix, and *VSING* and *MSING* are the specified values of the VSINGULAR= and MSINGULAR= options. The default value for *ASING* is the square root of the smallest positive double-precision value. Note that, in many cases, a normalized matrix $\mathbf{D}^{-1}\mathbf{H}\mathbf{D}^{-1}$ is decomposed, and the singularity criteria are modified correspondingly.

ASYCOV | ASC=*name*

specifies the formula for asymptotic covariances used in the weight matrix $\mathbf{W}$ for WLS and DWLS estimation. The ASYCOV option is effective only if METHOD=WLS or METHOD=DWLS and no INWGT= input data set is specified. The following formulas are implemented:

BIASED	Browne's (1984) formula (3.4) biased asymptotic covariance estimates; the resulting weight matrix is at least positive semidefinite. This is the default for analyzing a covariance matrix.
UNBIASED	Browne's (1984) formula (3.8) asymptotic covariance estimates corrected for bias; the resulting weight matrix can be indefinite (that is, can have negative eigenvalues), especially for small N.
CORR	Browne and Shapiro's (1986) formula (3.2), which is identical to DeLeeuw's (1983) formulas (2, 3, 4) the asymptotic variances of the diagonal elements are set to the reciprocal of the value r specified by the WPENALTY= option (default: $r = 100$). This formula is the default for analyzing a correlation matrix.

CAUTION: Using the WLS and DWLS methods with the ASYCOV=CORR option means that you are fitting a correlation (rather than a covariance) structure. Since the fixed diagonal of a correlation matrix for some models does not contribute to the model's degrees of freedom, you can specify the DFREDUCE=*i* option to reduce the degrees of freedom by the number of manifest variables used in the model. See the section "Counting the Degrees of Freedom" on page 956 for more information.

AUGMENT | AUG

analyzes the augmented correlation or covariance matrix. Using the AUG option is equivalent to specifying UCORR (NOINT but not COV) or UCOV (NOINT and COV) for a data set that is augmented by an intercept variable INTERCEPT that has constant values equal to 1. The variable INTERCEP can be used instead of the default INTERCEPT only if you specify the SAS option OPTIONS VALIDVARNAME=V6. The dimension of an augmented matrix is one higher than that of the corresponding correlation or covariance matrix. The AUGMENT option is effective only if the data set does not contain a variable called INTERCEPT and if you specify the UCOV, UCORR, or NOINT option.

CAUTION: The INTERCEPT variable is included in the moment matrix as the variable with number $n + 1$. Using the RAM model statement assumes that the first n variable numbers correspond to the n manifest variables in the input data set. Therefore, specifying the AUGMENT option assumes that the numbers of the latent variables used in the RAM or path model have to start with number $n + 2$.

BIASKUR

computes univariate skewness and kurtosis by formulas uncorrected for bias. See the section "Measures of Multivariate Kurtosis" on page 938 for more information.

COVARIANCE | COV

analyzes the covariance matrix instead of the correlation matrix. By default, PROC CALIS (like the FACTOR procedure) analyzes a correlation matrix. If the DATA= input data set is a valid TYPE=CORR data set (containing a correlation matrix and standard deviations), using the COV option means that the covariance matrix is computed and analyzed.

COVSING=*r*

specifies a nonnegative threshold r, which determines whether the eigenvalues of the information matrix are considered to be zero. If the inverse of the information matrix is found to be singular (depending on the VSINGULAR=, MSINGULAR=, ASINGULAR=, or SINGULAR= option), a generalized inverse is computed using the eigenvalue decomposition of the singular matrix. Those eigenvalues smaller than r are considered to be zero. If a generalized inverse is computed and you do not specify the NOPRINT option, the distribution of eigenvalues is displayed.

DATA=*SAS-data-set*

specifies an input data set that can be an ordinary SAS data set or a specially structured TYPE=CORR, TYPE=COV, TYPE=UCORR, TYPE=UCOV, TYPE=SSCP, or TYPE=FACTOR SAS data set, as described in the section "Input Data Sets" on page 909. If the DATA= option is omitted, the most recently created SAS data set is used.

DEMPHAS | DE=*r*

changes the initial values of all parameters that are located on the diagonals of the central model matrices by the relationship

$$diag_{new} = r(|diag_{old}| + 1)$$

The initial values of the diagonal elements of the central matrices should always be nonnegative to generate positive-definite predicted model matrices in the first iteration. By using values of $r > 1$, such as $r = 2$, $r = 10$, ..., you can increase these initial values to produce predicted model matrices with high positive eigenvalues in the first iteration. The DEMPHAS= option is effective independently of the way the initial values are set; that is, it changes the initial values set in the model specification as well as those set by an INRAM= data set and those automatically generated for RAM, LINEQS, or FACTOR model statements. It also affects the initial values set by the START= option, which uses, by default, DEMPHAS=100 if a covariance matrix is analyzed and DEMPHAS=10 for a correlation matrix.

DFREDUCE | DFRED=*i*

reduces the degrees of freedom of the χ^2 test by i. In general, the number of degrees of freedom is the number of elements of the lower triangle of the predicted model matrix $\mathbf{C}$, $n(n+1)/2$, minus the number of parameters, t. If the NODIAG option is used, the number of degrees of freedom is additionally reduced by n. Because negative values of i are allowed, you can also increase the number of degrees of freedom by using this option. If the DFREDUCE= or NODIAG option is used in a correlation structure analysis, PROC CALIS does not additionally reduce the degrees of freedom by the number of constant elements in the diagonal of the predicted model matrix, which is otherwise done automatically. See the section "Counting the Degrees of Freedom" on page 956 for more information.

EDF | DFE=*n*

makes the effective number of observations $n + i$, where i is 0 if the NOINT, UCORR, or UCOV option is specified without the AUGMENT option or where i is 1 otherwise. You can also use the NOBS= option to specify the number of observations.

FCONV | FTOL=*r*

specifies the relative function convergence criterion. The optimization process is terminated when the relative difference of the function values of two consecutive iterations is smaller than the specified value of r; that is,

$$\frac{|f(x^{(k)}) - f(x^{(k-1)})|}{\max(|f(x^{(k-1)})|, FSIZE)} \leq r$$

where $FSIZE$ can be defined by the FSIZE= option in the NLOPTIONS statement.

The default value is $r = 10^{-FDIGITS}$, where $FDIGITS$ either can be specified in the NLOPTIONS statement or is set by default to $-\log_{10}(\epsilon)$, where ϵ is the machine precision.

FDCODE

replaces the analytic derivatives of the programming statements by numeric derivatives (finite-difference approximations). In general, this option is needed only when you have programming statements that are too difficult for the built-in function compiler to differentiate analytically. For example, if the program code for the nonlinear constraints contains many arrays and many DO loops with array processing, the built-in function compiler can require too much time and memory to compute derivatives of the constraints with respect to the parameters. In this case, the Jacobian matrix of constraints is computed numerically by using finite-difference approximations. The FDCODE option does not modify the kind of derivatives specified with the HESSALG= option.

G4=*i*

specifies the algorithm to compute the approximate covariance matrix of parameter estimates used for computing the approximate standard errors and modification indices when the information matrix is singular. If the number of parameters t used in the model you analyze is smaller than the value of i, the time-expensive Moore-Penrose (G4) inverse of the singular information matrix is computed by eigenvalue decomposition. Otherwise, an inexpensive pseudo (G1) inverse is computed by sweeping. By default, $i = 60$. For more details, see the section "Estimation Criteria" on page 923.

GCONV | GTOL=*r*

specifies the relative gradient convergence criterion (see the ABSGCONV= option on page 892 for the absolute gradient convergence criterion).

Termination of all techniques (except the CONGRA technique) requires the normalized predicted function reduction to be small,

$$\frac{[g(x^{(k)})]'[\mathbf{G}^{(k)}]^{-1}g(x^{(k)})}{\max(|f(x^{(k)})|, FSIZE)} \leq r$$

where $FSIZE$ can be defined by the FSIZE= option in the NLOPTIONS statement. For the CONGRA technique (where a reliable Hessian estimate $\mathbf{G}$ is not available),

$$\frac{\parallel g(x^{(k)}) \parallel_2^2 \quad \parallel s(x^{(k)}) \parallel_2}{\parallel g(x^{(k)}) - g(x^{(k-1)}) \parallel_2 \max(|f(x^{(k)})|, FSIZE)} \leq r$$

is used. The default value is $r = 10^{-8}$.

Note that prior to SAS 6.11, the GCONV= option specified the absolute gradient convergence criterion.

HESSALG | HA = 1 | 2 | 3 | 4 | 5 | 6 | 11

specifies the algorithm used to compute the (approximate) Hessian matrix when TECHNIQUE=LEVMAR and NEWRAP, to compute approximate standard errors of the parameter estimates, and to compute Lagrange multipliers. There are different groups of algorithms available:

- analytic formulas: HA=*1,2,3,4,11*
- finite-difference approximation: HA=*5,6*
- dense storage: HA=*1,2,3,4,5,6*
- sparse storage: HA=*11*

If the Jacobian is more than 25% dense, the dense analytic algorithm, HA= 1, is used by default. The HA= 1 algorithm is faster than the other dense algorithms, but it needs considerably more memory for large problems than HA= 2,3,4. If the Jacobian is more than 75% sparse, the sparse analytic algorithm, HA= 11, is used by default. The dense analytic algorithm HA= 4 corresponds to the original COSAN algorithm; you are advised not to specify HA= 4 due to its very slow performance. If there is not enough memory available for the dense analytic algorithm HA= 1 and you must specify HA= 2 or HA= 3, it might be more efficient to use one of the quasi-Newton or conjugate-gradient optimization techniques since Levenberg-Marquardt and Newton-Raphson optimization techniques need to compute the Hessian matrix in each iteration. For approximate standard errors and modification indices, the Hessian matrix has to be computed at least once, regardless of the optimization technique.

The algorithms HA= 5 and HA= 6 compute approximate derivatives by using forward-difference formulas. The HA= 5 algorithm corresponds to the analytic HA= 1: it is faster than HA= 6, but it needs much more memory. The HA= 6 algorithm corresponds to the analytic HA= 2: it is slower than HA= 5, but it needs much less memory.

Test computations of large sparse problems show that the sparse algorithm HA= 11 can be up to 10 times faster than HA= 1 (and needs much less memory).

INEST | INVAR | ESTDATA=*SAS-data-set*

specifies an input data set that contains initial estimates for the parameters used in the optimization process and can also contain boundary and general linear constraints on the parameters. If the model did not change too much, you can specify an OUTEST= data set from a previous PROC CALIS analysis. The initial estimates are taken from the values of the PARMS observation.

INRAM=*SAS-data-set*

specifies an input data set that contains in RAM list form all information needed to specify an analysis model. The INRAM= data set is described in the section "Input Data Sets" on page 909. Typically, this input data set is an OUTRAM= data set (possibly modified) from a previous PROC CALIS analysis. If you use an INRAM= data set to specify the analysis model, you cannot use the model specification statement COSAN, MATRIX, RAM, LINEQS, STD, COV, FACTOR, or VARNAMES, but you can use the BOUNDS and PARAMETERS statements and programming statements. If the INRAM= option is omitted, you must define the analysis model with a COSAN, RAM, LINEQS, or FACTOR statement.

INSTEP=*r*

For highly nonlinear objective functions, such as the EXP function, the default initial radius of the trust-region algorithms TRUREG, DBLDOG, and LEVMAR or the default step length of the line-search algorithms can produce arithmetic overflows. If this occurs, specify decreasing values of $0 < r < 1$ such as INSTEP=1E−1, INSTEP=1E−2, INSTEP=1E−4, ..., until the iteration starts successfully.

- For trust-region algorithms (TRUREG, DBLDOG, and LEVMAR), the INSTEP option specifies a positive factor for the initial radius of the trust-region. The default initial trust-region radius is the length of the scaled gradient, and it corresponds to the default radius factor of $r = 1$.
- For line-search algorithms (NEWRAP, CONGRA, and QUANEW), INSTEP specifies an upper bound for the initial step length for the line search during the first five iterations. The default initial step length is $r = 1$.

For releases prior to SAS 6.11, specify the SALPHA= and RADIUS= options. For more details, see the section "Computational Problems" on page 958.

INWGT=*SAS-data-set*

specifies an input data set that contains the weight matrix **W** used in generalized least squares (GLS), weighted least squares (WLS, ADF), or diagonally weighted least squares (DWLS) estimation. If the weight matrix **W** defined by an INWGT= data set is not positive-definite, it can be ridged by using the WRIDGE= option. See the section "Estimation Criteria" on page 923 for more information. If no INWGT= data set is specified, default settings for the weight matrices are used in the estimation process. The INWGT= data set is described in the section "Input Data Sets" on page 909. Typically, this input data set is an OUTWGT= data set from a previous PROC CALIS analysis.

KURTOSIS | KU

computes and displays univariate kurtosis and skewness, various coefficients of multivariate kurtosis, and the numbers of observations that contribute most to the normalized multivariate kurtosis. See the section "Measures of Multivariate Kurtosis" on page 938 for more information. Using the KURTOSIS option implies the SIMPLE display option. This information is computed only if the DATA= data set is a raw data set, and it is displayed by default if the PRINT option is specified. The multivariate LS kappa and the multivariate mean kappa are displayed only if you specify METHOD=WLS and the weight matrix is computed from an input raw data set. All measures of skewness and kurtosis are corrected for the mean. If an intercept variable is included in the analysis, the measures of multivariate kurtosis do not include the intercept variable in the corrected covariance matrix, as indicated by a displayed message. Using the BIASKUR option displays the biased values of univariate skewness and kurtosis.

LINESEARCH | LIS | SMETHOD | SM=*i*

specifies the line-search method for the CONGRA, QUANEW, and NEWRAP optimization techniques. Refer to Fletcher (1980) for an introduction to line-search techniques. The value of i can be $1, \ldots, 8$; the default is $i = 2$.

LIS=1	specifies a line-search method that needs the same number of function and gradient calls for cubic interpolation and cubic extrapolation; this method is similar to one used by the Harwell subroutine library.
LIS=2	specifies a line-search method that needs more function calls than gradient calls for quadratic and cubic interpolation and cubic extrapolation; this method is implemented as shown in Fletcher (1987) and can be modified to an exact line search by using the LSPRECISION= option.
LIS=3	specifies a line-search method that needs the same number of function and gradient calls for cubic interpolation and cubic extrapolation; this method is implemented as shown in Fletcher (1987) and can be modified to an exact line search by using the LSPRECISION= option.
LIS=4	specifies a line-search method that needs the same number of function and gradient calls for stepwise extrapolation and cubic interpolation.
LIS=5	specifies a line-search method that is a modified version of LIS=4.
LIS=6	specifies golden section line search (Polak 1971), which uses only function values for linear approximation.
LIS=7	specifies bisection line search (Polak 1971), which uses only function values for linear approximation.
LIS=8	specifies Armijo line-search technique (Polak 1971), which uses only function values for linear approximation.

LSPRECISION | LSP=*r*

SPRECISION | SP=*r*

specifies the degree of accuracy that should be obtained by the line-search algorithms LIS=2 and LIS=3. Usually an imprecise line search is inexpensive and successful. For more difficult optimization problems, a more precise and more expensive line search might be necessary

(Fletcher 1980, p. 22). The second (default for NEWRAP, QUANEW, and CONGRA) and third line-search methods approach exact line search for small LSPRECISION= values. If you have numerical problems, you should decrease the LSPRECISION= value to obtain a more precise line search. The default LSPRECISION= values are displayed in the following table.

TECH=	**UPDATE=**	**LSP Default**
QUANEW	DBFGS, BFGS	r = 0.4
QUANEW	DDFP, DFP	r = 0.06
CONGRA	all	r = 0.1
NEWRAP	no update	r = 0.9

For more details, refer to Fletcher (1980, pp. 25–29).

MAXFUNC | MAXFU=*i*

specifies the maximum number i of function calls in the optimization process. The default values are displayed in the following table.

TECH=	**MAXFUNC Default**
LEVMAR, NEWRAP, NRRIDG, TRUREG	i=125
DBLDOG, QUANEW	i=500
CONGRA	i=1000

The default is used if you specify MAXFUNC=0. The optimization can be terminated only after completing a full iteration. Therefore, the number of function calls that are actually performed can exceed the number that is specified by the MAXFUNC= option.

MAXITER | MAXIT=*i* < *n* >

specifies the maximum number i of iterations in the optimization process. The default values are displayed in the following table.

TECH=	**MAXITER Default**
LEVMAR, NEWRAP, NRRIDG, TRUREG	i=50
DBLDOG, QUANEW	i=200
CONGRA	i=400

The default is used if you specify MAXITER=0 or if you omit the MAXITER option.

The optional second value n is valid only for TECH=QUANEW with nonlinear constraints. It specifies an upper bound n for the number of iterations of an algorithm and reduces the violation of nonlinear constraints at a starting point. The default is n=20. For example, specifying

```
maxiter= . 0
```

means that you do not want to exceed the default number of iterations during the main optimization process and that you want to suppress the feasible-point algorithm for nonlinear constraints.

METHOD | MET=*name*

specifies the method of parameter estimation. Valid values for *name* are as follows:

ML | M | MAX	performs normal-theory maximum likelihood parameter estimation. The ML method requires a nonsingular covariance or correlation matrix. This is the default method.
GLS | G	performs generalized least squares parameter estimation. If no INWGT= data set is specified, the GLS method uses the inverse sample covariance or correlation matrix as weight matrix **W**. Therefore, METHOD=GLS requires a nonsingular covariance or correlation matrix.
WLS | W | ADF	performs weighted least squares parameter estimation. If no INWGT= data set is specified, the WLS method uses the inverse matrix of estimated asymptotic covariances of the sample covariance or correlation matrix as the weight matrix **W**. In this case, the WLS estimation method is equivalent to Browne's (1982, 1984) asymptotically distribution-free estimation. The WLS method requires a nonsingular weight matrix.
DWLS | D	performs diagonally weighted least squares parameter estimation. If no INWGT= data set is specified, the DWLS method uses the inverse diagonal matrix of asymptotic variances of the input sample covariance or correlation matrix as the weight matrix **W**. The DWLS method requires a nonsingular diagonal weight matrix.
ULS | LS | U	performs unweighted least squares parameter estimation.
LSML | LSM | LSMAX	performs unweighted least squares followed by normal-theory maximum likelihood parameter estimation.
LSGLS | LSG	performs unweighted least squares followed by generalized least squares parameter estimation.
LSWLS | LSW | LSADF	performs unweighted least squares followed by weighted least squares parameter estimation.
LSDWLS | LSD	performs unweighted least squares followed by diagonally weighted least squares parameter estimation.
NONE | NO	uses no estimation method. This option is suitable for checking the validity of the input information and for displaying the model matrices and initial values.

The default estimation method is maximum likelihood (METHOD=ML), assuming a multivariate normal distribution of the observed variables. The two-stage estimation methods METHOD=LSML, METHOD=LSGLS, METHOD=LSWLS, and METHOD=LSDWLS first compute unweighted least squares estimates of the model parameters and their residuals. Afterward, these estimates are used as initial values for the optimization process to compute maximum likelihood, generalized least squares, weighted least squares, or diagonally weighted least squares parameter estimates. You can do the same thing by using an OUTRAM= data set with least squares estimates as an INRAM= data set for a further analysis to obtain the second set of parameter estimates. This strategy is also discussed in the section "Use of Optimization Techniques" on page 945. For more details, see the section "Estimation Criteria" on page 923.

MODIFICATION | MOD

computes and displays Lagrange multiplier test indices for constant parameter constraints, equality parameter constraints, and active boundary constraints, as well as univariate and multivariate Wald test indices. The modification indices are not computed in the case of unweighted or diagonally weighted least squares estimation.

The Lagrange multiplier test (Bentler 1986; Lee 1985; Buse 1982) provides an estimate of the χ^2 reduction that results from dropping the constraint. For constant parameter constraints and active boundary constraints, the approximate change of the parameter value is displayed also. You can use this value to obtain an initial value if the parameter is allowed to vary in a modified model. For more information, see the section "Modification Indices" on page 953.

MSINGULAR | MSING=*r*

specifies a relative singularity criterion r, $r > 0$, for the inversion of the information matrix, which is needed to compute the covariance matrix. The following singularity criterion is used:

$$|d_{j,j}| \le \max(ASING, VSING * |H_{j,j}|, MSING * \max(|H_{1,1}|, \ldots, |H_{n,n}|))$$

where $d_{j,j}$ is the diagonal pivot of the matrix, and *ASING* and *VSING* are the specified values of the ASINGULAR= and VSINGULAR= options, respectively. If you do not specify the SINGULAR= option, the default value for *MSING* is 1E−12; otherwise, the default value is 1E−4 * SINGULAR. Note that, in many cases, a normalized matrix $\mathbf{D}^{-1}\mathbf{H}\mathbf{D}^{-1}$ is decomposed, and the singularity criteria are modified correspondingly.

NOADJDF

turns off the automatic adjustment of degrees of freedom when there are active constraints in the analysis. When the adjustment is in effect, most fit statistics and the associated probability levels will be affected. This option should be used when the researcher believes that the active constraints observed in the current sample will have little chance to occur in repeated sampling.

NOBS=*nobs*

specifies the number of observations. If the DATA= input data set is a raw data set, *nobs* is defined by default to be the number of observations in the raw data set. The NOBS= and EDF= options override this default definition. You can use the RDF= option to modify the *nobs* specification. If the DATA= input data set contains a covariance, correlation, or scalar product matrix, you can specify the number of observations either by using the NOBS=, EDF=, and RDF= options in the PROC CALIS statement or by including a _TYPE_='N' observation in the DATA= input data set.

NODIAG | NODI

omits the diagonal elements of the analyzed correlation or covariance matrix from the fit function. This option is useful only for special models with constant error variables. The NODIAG option does not allow fitting of those parameters that contribute to the diagonal of the estimated moment matrix. The degrees of freedom are automatically reduced by n. A simple example for the usefulness of the NODIAG option is the fit of the first-order factor model, $\mathbf{S} = \mathbf{F}\mathbf{F}' + \mathbf{U}^2$. In this case, you do not have to estimate the diagonal matrix of unique variances $\mathbf{U}^2$ that are fully determined by $diag(\mathbf{S} - \mathbf{F}\mathbf{F}')$.

NOINT

specifies that no intercept be used in computing covariances and correlations; that is, covariances or correlations are not corrected for the mean. You can specify this option (or UCOV or UCORR) to analyze mean structures in an uncorrected moment matrix—that is, to compute intercepts in systems of structured linear equations (see Example 25.2). The term NOINT is misleading in this case because an uncorrected covariance or correlation matrix is analyzed containing a constant (intercept) variable that is used in the analysis model. The degrees of freedom used in the variance divisor (specified by the VARDEF= option) and some of the assessment of the fit function (see the section "Assessment of Fit" on page 928) depend on whether an intercept variable is included in the model (the intercept is used in computing the corrected covariance or correlation matrix or is used as a variable in the uncorrected covariance or correlation matrix to estimate mean structures) or not included (an uncorrected covariance or correlation matrix is used that does not contain a constant variable).

NOMOD

does not compute modification indices. The NOMOD option is useful in connection with the PALL option because it saves computing time.

NOPRINT | NOP

suppresses the displayed output. Note that this option temporarily disables the Output Delivery System (ODS). For more information, see Chapter 20, "Using the Output Delivery System."

NOSTDERR | NOSE

specifies that standard errors should not be computed. Standard errors are not computed for unweighted least squares (ULS) or diagonally weighted least squares (DWLS) estimation. In general, standard errors are computed even if the STDERR display option is not used (for file output).

OUTEST | OUTVAR=*SAS-data-set*

creates an output data set containing the parameter estimates, their gradient, Hessian matrix, and boundary and linear constraints. For METHOD=ML, METHOD=GLS, and METHOD=WLS, the OUTEST= data set also contains the information matrix, the approximate covariance matrix of the parameter estimates ((generalized) inverse of information matrix), and approximate standard errors. If linear or nonlinear equality or active inequality constraints are present, the Lagrange multiplier estimates of the active constraints, the projected Hessian, and the Hessian of the Lagrange function are written to the data set. The OUTEST= data set also contains the Jacobian if the OUTJAC option is used.

The OUTEST= data set is described in the section "OUTEST= SAS-data-set" on page 913. If you want to create a permanent SAS data set, you must specify a two-level name. Refer to the chapter titled "SAS Data Files" in *SAS Language Reference: Concepts* for more information about permanent data sets.

OUTJAC

writes the Jacobian matrix, if it has been computed, to the OUTEST= data set. This is useful when the information and Jacobian matrices need to be computed for other analyses.

OUTRAM=*SAS-data-set*

creates an output data set containing the model information for the analysis, the parameter estimates, and their standard errors. An OUTRAM= data set can be used as an input INRAM= data set in a subsequent analysis by PROC CALIS. The OUTRAM= data set also contains a set of fit indices; it is described in more detail in the section "OUTRAM= SAS-data-set" on page 917. If you want to create a permanent SAS data set, you must specify a two-level name. Refer to the chapter titled "SAS Data Files" in *SAS Language Reference: Concepts* for more information about permanent data sets.

OUTSTAT=*SAS-data-set*

creates an output data set containing the BY-group variables, the analyzed covariance or correlation matrices, and the predicted and residual covariance or correlation matrices of the analysis. You can specify the correlation or covariance matrix in an OUTSTAT= data set as an input DATA= data set in a subsequent analysis by PROC CALIS. The OUTSTAT= data set is described in the section "OUTSTAT= SAS-data-set" on page 920. If the model contains latent variables, this data set also contains the predicted covariances between latent and manifest variables and the latent variable score regression coefficients (see the PLATCOV option on page 864). If the FACTOR statement is used, the OUTSTAT= data set also contains the rotated and unrotated factor loadings, the unique variances, the matrix of factor correlations, the transformation matrix of the rotation, and the matrix of standardized factor loadings.

You can use the latent variable score regression coefficients with PROC SCORE to compute latent variable or factor scores. For details, see the section "Latent Variable Scores" on page 937.

If you want to create a permanent SAS data set, you must specify a two-level name. Refer to the chapter titled "SAS Data Files" in *SAS Language Reference: Concepts* for more information about permanent data sets.

OUTWGT=*SAS-data-set*

creates an output data set containing the weight matrix **W** used in the estimation process. You cannot create an OUTWGT= data set with an unweighted least squares or maximum likelihood estimation. The fit function in GLS, WLS (ADF), and DWLS estimation contains the inverse of the (Cholesky factor of the) weight matrix **W** written in the OUTWGT= data set. The OUTWGT= data set contains the weight matrix to which the WRIDGE= and the WPENALTY= options are applied. An OUTWGT= data set can be used as an input INWGT= data set in a subsequent analysis by PROC CALIS. The OUTWGT= data set is described in the section "OUTWGT= SAS-data-set" on page 922. If you want to create a permanent SAS data set, you must specify a two-level name. Refer to the chapter titled "SAS Data Files" in *SAS Language Reference: Concepts* for more information about permanent data sets.

PALL | ALL

displays all optional output except the output generated by the PCOVES, PDETERM, PJACPAT, and PRIVEC options.

CAUTION: The PALL option includes the very expensive computation of the modification indices. If you do not really need modification indices, you can save computing time by specifying the NOMOD option in addition to the PALL option.

PCORR | CORR

displays the (corrected or uncorrected) covariance or correlation matrix that is analyzed and the predicted model covariance or correlation matrix.

PCOVES | PCE

displays the following:

- the information matrix (crossproduct Jacobian)
- the approximate covariance matrix of the parameter estimates (generalized inverse of the information matrix)
- the approximate correlation matrix of the parameter estimates

The covariance matrix of the parameter estimates is not computed for the ULS and DWLS estimation methods. This displayed output is not included in the output generated by the PALL option.

PDETERM | PDE

displays three coefficients of determination: the determination of all equations (DETAE), the determination of the structural equations (DETSE), and the determination of the manifest variable equations (DETMV). These determination coefficients are intended to be global means of the squared multiple correlations for different subsets of model equations and variables. The coefficients are displayed only when you specify a RAM or LINEQS model, but they are displayed for all five estimation methods: ULS, GLS, ML, WLS, and DWLS.

You can use the STRUCTEQ statement to define which equations are structural equations. If you do not use the STRUCTEQ statement, PROC CALIS uses its own default definition to identify structural equations.

The term "structural equation" is not defined in a unique way. The LISREL program defines the structural equations by the user-defined BETA matrix. In PROC CALIS, the default definition of a structural equation is an equation that has a dependent left-side variable that appears at least once on the right side of another equation, or an equation that has at least one right-side variable that is the left-side variable of another equation. Therefore, PROC CALIS sometimes identifies more equations as structural equations than the LISREL program does.

If the model contains structural equations, PROC CALIS also displays the "Stability Coefficient of Reciprocal Causation"—that is, the largest eigenvalue of the $\mathbf{BB}'$ matrix, where $\mathbf{B}$ is the causal coefficient matrix of the structural equations. These coefficients are computed as in the LISREL VI program of Jöreskog and Sörbom (1985). This displayed output is not included in the output generated by the PALL option.

PESTIM | PES

displays the parameter estimates. In some cases, this includes displaying the standard errors and t values.

PINITIAL | PIN

displays the input model matrices and the vector of initial values.

PJACPAT | PJP

displays the structure of variable and constant elements of the Jacobian matrix. This displayed output is not included in the output generated by the PALL option.

PLATCOV | PLC

displays the following:

- the estimates of the covariances among the latent variables
- the estimates of the covariances between latent and manifest variables
- the latent variable score regression coefficients

The estimated covariances between latent and manifest variables and the latent variable score regression coefficients are written to the OUTSTAT= data set. You can use the score coefficients with PROC SCORE to compute factor scores. For details, see the section "Latent Variable Scores" on page 937.

PLOTS < (*global-plot-options*) > = *plot-request* < (*options*) >

PLOTS < (*global-plot-options*) > = (*plot-request* < (*options*) > < ... *plot-request* < (*options*) > >)

specifies the ODS graphical plots. Currently, the only available ODS graphical plots in PROC CALIS are for residual distributions. Also, when the residual histograms are requested, the bar charts of residual tallies are suppressed. To display these bar charts with the residual histograms, you must use the RESIDUAL(TALLY) option.

When you specify only one *plot-request*, you can omit the parentheses around the *plot-request*. For example:

```
plots=all
plots=residuals
```

You must enable ODS Graphics before requesting plots; for example:

```
ods graphics on;

proc calis plots=residuals;
run;

ods graphics off;
```

For more information about the ODS GRAPHICS statement, see Chapter 21, "Statistical Graphics Using ODS."

The following table shows the available *plot-requests*:

Plot-request	Plot Description
ALL	all available plots
NONE	no ODS graphical plots
RESIDUALS	distribution of residuals

PREDET | PRE

displays the pattern of variable and constant elements of the predicted moment matrix that is predetermined by the analysis model. It is especially helpful in finding manifest variables that are not used or that are used as exogenous variables in a complex model specified in the COSAN statement. Those entries of the predicted moment matrix for which the model generates variable (rather than constant) elements are displayed as missing values. This output is displayed even without specifying the PREDET option if the model generates constant elements in the predicted model matrix different from those in the analysis moment matrix and if you specify at least the PSHORT amount of displayed output.

If the analyzed matrix is a correlation matrix (containing constant elements of 1s in the diagonal) and the model generates a predicted model matrix with q constant (rather than variable) elements in the diagonal, the degrees of freedom are automatically reduced by q. The output generated by the PREDET option displays those constant diagonal positions. If you specify the DFREDUCE= or NODIAG option, this automatic reduction of the degrees of freedom is suppressed. See the section "Counting the Degrees of Freedom" on page 956 for more information.

PRIMAT | PMAT

displays parameter estimates, approximate standard errors, and *t* values in matrix form if you specify the analysis model in the RAM or LINEQS statement. When a COSAN statement is used, this occurs by default.

PRINT | PRI

adds the options KURTOSIS, RESIDUAL, PLATCOV, and TOTEFF to the default output.

PRIVEC | PVEC

displays parameter estimates, approximate standard errors, the gradient, and *t* values in vector form. The values are displayed with more decimal places. This displayed output is not included in the output generated by the PALL option.

PSHORT | SHORT | PSH

excludes the output produced by the PINITIAL, SIMPLE, and STDERR options from the default output.

PSUMMARY | SUMMARY | PSUM

displays the fit assessment table and the ERROR, WARNING, and NOTE messages.

PWEIGHT | PW

displays the weight matrix $\mathbf{W}$ used in the estimation. The weight matrix is displayed after the WRIDGE= and WPENALTY= options are applied to it.

RADIUS=*r*

is an alias for the INSTEP= option for Levenberg-Marquardt minimization.

RANDOM =*i*

specifies a positive integer as a seed value for the pseudo-random number generator to generate initial values for the parameter estimates for which no other initial value assignments in the model definitions are made. Except for the parameters in the diagonal locations of the central matrices in the model, the initial values are set to random numbers in the range

$0 \leq r \leq 1$. The values for parameters in the diagonals of the central matrices are random numbers multiplied by 10 or 100. For more information, see the section "Initial Estimates" on page 941.

RDF | DFR=*n*

makes the effective number of observations the actual number of observations minus the RDF= value. The degree of freedom for the intercept should not be included in the RDF= option. If you use PROC CALIS to compute a regression model, you can specify RDF= *number-of-regressor-variables* to get approximate standard errors equal to those computed by PROC REG.

RESIDUAL | RES < (TALLY | TALLIES) > < = NORM | VARSTAND | ASYSTAND >

displays the raw and normalized residual covariance matrices, the rank order of the largest residuals, and the bar charts of residual tallies. This information is displayed by default when you specify the PRINT option.

Three types of normalized or standardized residual matrices can be chosen with the RESIDUAL= specification:

RESIDUAL= NORM	normalized residuals
RESIDUAL= VARSTAND | VARSTD	variance standardized residuals
RESIDUAL= ASYSTAND | ASYMSTD	asymptotically standardized residuals

When ODS graphical plots of residuals are also requested, the bar charts of residual tallies are suppressed. They are replaced with high-quality graphical histograms showing residual distributions. If you still want to display the bar charts in this situation, use the RESIDUAL(TALLY) or RESIDUAL(TALLY)= option.

For more details, see the section "Assessment of Fit" on page 928.

RIDGE< =*r* >

defines a ridge factor r for the diagonal of the moment matrix **S** that is analyzed. The matrix **S** is transformed to

$$\mathbf{S} \longrightarrow \tilde{\mathbf{S}} = \mathbf{S} + r(diag(\mathbf{S}))$$

If you do not specify r in the RIDGE option, PROC CALIS tries to ridge the moment matrix **S** so that the smallest eigenvalue is about 10^{-3}.

CAUTION: The moment matrix in the OUTSTAT= output data set does not contain the ridged diagonal.

SALPHA=*r*

is an alias for the INSTEP= option for line-search algorithms.

SIMPLE | S

displays means, standard deviations, skewness, and univariate kurtosis if available. This information is displayed when you specify the PRINT option. If you specify the UCOV, UCORR, or NOINT option, the standard deviations are not corrected for the mean. If the KURTOSIS option is specified, the SIMPLE option is set by default.

SINGULAR | SING=*r*

specifies the singularity criterion r, $0 < r < 1$, used, for example, for matrix inversion. The default value is the square root of the relative machine precision or, equivalently, the square root of the largest double-precision value that, when added to 1, results in 1.

SLMW=*r*

specifies the probability limit used for computing the stepwise multivariate Wald test. The process stops when the univariate probability is smaller than r. The default value is $r = 0.05$.

SPRECISION | SP=*r*

is an alias for the LSPRECISION= option.

START=*r*

In general, this option is needed only in connection with the COSAN model statement, and it specifies a constant r as an initial value for all the parameter estimates for which no other initial value assignments in the pattern definitions are made. Start values in the diagonal locations of the central matrices are set to $100|r|$ if a COV or UCOV matrix is analyzed and $10|r|$ if a CORR or UCORR matrix is analyzed. The default value is $r = 0.5$. Unspecified initial values in a FACTOR, RAM, or LINEQS model are usually computed by PROC CALIS. If none of the initialization methods are able to compute all starting values for a model specified by a FACTOR, RAM, or LINEQS statement, then the start values of parameters that could not be computed are set to r, $10|r|$, or $100|r|$. If the DEMPHAS= option is used, the initial values of the diagonal elements of the central model matrices are multiplied by the value specified in the DEMPHAS= option. For more information, see the section "Initial Estimates" on page 941.

STDERR | SE

displays approximate standard errors if estimation methods other than unweighted least squares (ULS) or diagonally weighted least squares (DWLS) are used (and the NOSTDERR option is not specified). If you specify neither the STDERR nor the NOSTDERR option, the standard errors are computed for the OUTRAM= data set. This information is displayed by default when you specify the PRINT option.

TECHNIQUE | TECH=*name*

OMETHOD | OM=*name*

specifies the optimization technique. Valid values for *name* are as follows:

CONGRA | CG	chooses one of four different conjugate-gradient optimization algorithms, which can be more precisely defined with the UPDATE= option and modified with the LINESEARCH= option. The conjugate-gradient techniques need only $O(t)$ memory compared to the $O(t^2)$ memory for the other three techniques, where t is the number of parameters. On the other hand, the conjugate-gradient techniques are significantly slower than other optimization techniques and should be used only when memory is insufficient for more efficient techniques. When you choose this option, UPDATE=PB by default. This is the default optimization technique if there are more than 400 parameters to estimate.

DBLDOG | DD performs a version of double-dogleg optimization, which uses the gradient to update an approximation of the Cholesky factor of the Hessian. This technique is, in many ways, very similar to the dual quasi-Newton method, but it does not use line search. The implementation is based on Dennis and Mei (1979) and Gay (1983).

LEVMAR | LM | MARQUARDT performs a highly stable but, for large problems, memory- and time-consuming Levenberg-Marquardt optimization technique, a slightly improved variant of the Moré (1978) implementation. This is the default optimization technique if there are fewer than 40 parameters to estimate.

NEWRAP | NRA performs a usually stable but, for large problems, memory- and time-consuming Newton-Raphson optimization technique. The algorithm combines a line-search algorithm with ridging, and it can be modified with the LINESEARCH= option. Prior to SAS 6.11, this option invokes the NRRIDG option.

NRRIDG | NRR | NR | NEWTON performs a usually stable but, for large problems, memory- and time-consuming Newton-Raphson optimization technique. This algorithm does not perform a line search. Since TECH=NRRIDG uses an orthogonal decomposition of the approximate Hessian, each iteration of TECH=NRRIDG can be slower than that of TECH=NEWRAP, which works with Cholesky decomposition. However, usually TECH=NRRIDG needs fewer iterations than TECH=NEWRAP.

QUANEW | QN chooses one of four different quasi-Newton optimization algorithms that can be more precisely defined with the UPDATE= option and modified with the LINESEARCH= option. If boundary constraints are used, these techniques sometimes converge slowly. When you choose this option, UPDATE=DBFGS by default. If nonlinear constraints are specified in the NLINCON statement, a modification of Powell's (1982a, 1982b) VMCWD algorithm is used, which is a sequential quadratic programming (SQP) method. This algorithm can be modified by specifying VERSION=1, which replaces the update of the Lagrange multiplier estimate vector μ to the original update of Powell (1978a, 1978b) that is used in the VF02AD algorithm. This can be helpful for applications with linearly dependent active constraints. The QUANEW technique is the default optimization technique if there are nonlinear constraints specified or if there are more than 40 and fewer than 400 parameters to estimate. The QUANEW algorithm uses only first-order derivatives of the objective function and, if available, of the nonlinear constraint functions.

TRUREG | TR performs a usually very stable but, for large problems, memory- and time-consuming trust-region optimization technique. The algorithm is implemented similar to Gay (1983) and Moré and Sorensen (1983).

NONE | NO does not perform any optimization. This option is similar to METHOD=NONE, but TECH=NONE also computes and displays residuals and goodness of fit statistics. If you specify METHOD=ML, METHOD=LSML, METHOD=GLS, METHOD=LSGLS, METHOD=WLS, or METHOD=LSWLS, this option enables computing and displaying (if the display options are specified) of the standard error estimates and modification indices corresponding to the input parameter estimates.

Since there is no single nonlinear optimization algorithm available that is clearly superior (in terms of stability, speed, and memory) for all applications, different types of optimization techniques are provided in the CALIS procedure. Each technique can be modified in various ways. The default optimization technique for fewer than 40 parameters ($t < 40$) is TECHNIQUE=LEVMAR. For $40 \leq t < 400$, TECHNIQUE=QUANEW is the default method, and for $t \geq 400$, TECHNIQUE=CONGRA is the default method. For more details, see the section "Use of Optimization Techniques" on page 945. You can specify the following set of options in the PROC CALIS statement or in the NLOPTIONS statement.

TOTEFF | TE

computes and displays total effects and indirect effects.

UCORR

analyzes the uncorrected correlation matrix instead of the correlation matrix corrected for the mean. Using the UCORR option is equivalent to specifying the NOINT option but not the COV option.

UCOV

analyzes the uncorrected covariance matrix instead of the covariance matrix corrected for the mean. Using the UCOV option is equivalent to specifying both the COV and NOINT options. You can specify this option to analyze mean structures in an uncorrected covariance matrix—that is, to compute intercepts in systems of linear structural equations (see Example 25.2).

UPDATE | UPD=*name*

specifies the update method for the quasi-Newton or conjugate-gradient optimization technique.

For TECHNIQUE=CONGRA, the following updates can be used:

PB	performs the automatic restart update method of Powell (1977) and Beale (1972). This is the default.
FR	performs the Fletcher-Reeves update (Fletcher 1980, p. 63).
PR	performs the Polak-Ribiere update (Fletcher 1980, p. 66).
CD	performs a conjugate-descent update of Fletcher (1987).

For TECHNIQUE=DBLDOG, the following updates (Fletcher 1987) can be used:

DBFGS	performs the dual Broyden-Fletcher-Goldfarb-Shanno (BFGS) update of the Cholesky factor of the Hessian matrix. This is the default.
DDFP	performs the dual Davidon-Fletcher-Powell (DFP) update of the Cholesky factor of the Hessian matrix.

For TECHNIQUE=QUANEW, the following updates (Fletcher 1987) can be used:

BFGS	performs original BFGS update of the inverse Hessian matrix. This is the default for earlier releases.
DFP	performs the original DFP update of the inverse Hessian matrix.
DBFGS	performs the dual BFGS update of the Cholesky factor of the Hessian matrix. This is the default.
DDFP	performs the dual DFP update of the Cholesky factor of the Hessian matrix.

VARDEF= DF | N | WDF | WEIGHT | WGT

specifies the divisor used in the calculation of covariances and standard deviations. The default value is VARDEF=DF. The values and associated divisors are displayed in the following table, where $i = 0$ if the NOINT option is used and $i = 1$ otherwise and where k is the number of partial variables specified in the PARTIAL statement. Using an intercept variable in a mean structure analysis, by specifying the AUGMENT option, includes the intercept variable in the analysis. In this case, $i = 1$. When a WEIGHT statement is used, w_j is the value of the WEIGHT variable in the jth observation, and the summation is performed only over observations with positive weight.

Value	Description	Divisor
DF	degrees of freedom	$N - k - i$
N	number of observations	N
WDF	sum of weights DF	$\sum_j^N w_j - k - i$
WEIGHT | WGT	sum of weights	$\sum_j^N w_j$

VSINGULAR | VSING=*r*

specifies a relative singularity criterion r, $r > 0$, for the inversion of the information matrix, which is needed to compute the covariance matrix. The following singularity criterion is used:

$$|d_{j,j}| \le \max(ASING, VSING * |H_{j,j}|, MSING * \max(|H_{1,1}|, \ldots, |H_{n,n}|))$$

where $d_{j,j}$ is the diagonal pivot of the matrix, and *ASING* and *MSING* are the specified values of the ASINGULAR= and MSINGULAR= options. If you do not specify the SINGULAR= option, the default value for *VSING* is 1E−8; otherwise, the default value is SINGULAR. Note that in many cases a normalized matrix $\mathbf{D}^{-1}\mathbf{H}\mathbf{D}^{-1}$ is decomposed, and the singularity criteria are modified correspondingly.

WPENALTY | WPEN=*r*

specifies the penalty weight $r \ge 0$ for the WLS and DWLS fit of the diagonal elements of a correlation matrix (constant 1s). The criterion for weighted least squares estimation of a correlation structure is

$$\mathbf{F}_{WLS} = \sum_{i=2}^{n} \sum_{j=1}^{i-1} \sum_{k=2}^{n} \sum_{l=1}^{k-1} w^{ij,kl}(s_{ij} - c_{ij})(s_{kl} - c_{kl}) + r \sum_{i}^{n} (s_{ii} - c_{ii})^2$$

where r is the penalty weight specified by the WPENALTY=r option and the $w^{ij,kl}$ are the elements of the inverse of the reduced $(n(n-1)/2) \times (n(n-1)/2)$ weight matrix that contains only the nonzero rows and columns of the full weight matrix **W**. The second term is a penalty term to fit the diagonal elements of the correlation matrix. The default value is 100. The reciprocal of this value replaces the asymptotic variance corresponding to the diagonal elements of a correlation matrix in the weight matrix **W**, and it is effective only with the ASYCOV=CORR option. The often used value $r = 1$ seems to be too small in many cases to fit the diagonal elements of a correlation matrix properly. The default WPENALTY= value emphasizes the importance of the fit of the diagonal elements in the correlation matrix. You can decrease or increase the value of r if you want to decrease or increase the importance of the diagonal elements fit. This option is effective only with the WLS or DWLS estimation method and the analysis of a correlation matrix. See the section "Estimation Criteria" on page 923 for more details.

WRIDGE=*r*

defines a ridge factor r for the diagonal of the weight matrix **W** used in GLS, WLS, or DWLS estimation. The weight matrix **W** is transformed to

$$\mathbf{W} \longrightarrow \tilde{\mathbf{W}} = \mathbf{W} + r(diag(\mathbf{W}))$$

The WRIDGE= option is applied to the weight matrix as follows:

- before the WPENALTY= option is applied to it
- before the weight matrix is written to the OUTWGT= data set
- before the weight matrix is displayed

BOUNDS Statement

BOUNDS *constraint < , constraint . . . >* **;**

where *constraint* represents
< *number operator* > *parameter-list* < *operator number* >

You can use the BOUNDS statement to define boundary constraints for any parameter that has its name specified in a MATRIX, LINEQS, STD, COV, or RAM statement or that is used in the model of an INRAM= data set. Valid operators are <=, <, >=, >, and = or, equivalently, LE, LT, GE, GT, and EQ. The following is an example of the BOUNDS statement:

```
   bounds         0.   <= a1-a9 x    <= 1. ,
                 -1.   <= c2-c5            ,
                          b1-b10 y   >= 0. ;
```

You must separate boundary constraints with a comma, and you can specify more than one BOUNDS statement. The feasible region for a parameter is the intersection of all boundary constraints specified for that parameter; if a parameter has a maximum lower boundary constraint larger than its minimum upper bound, the parameter is set equal to the minimum of the upper bounds.

If you need to compute the values of the upper or lower bounds, create a TYPE=EST data set containing _TYPE_='UPPERBD' or _TYPE_='LOWERBD' observations and use it as an INEST= or INVAR= input data set in a later PROC CALIS run.

The BOUNDS statement can contain only parameter names and numerical constants. You cannot use the names of variables created in SAS programming statements.

The active set strategies made available in PROC CALIS cannot realize the strict inequality constraints < or >. For example, you cannot specify **`BOUNDS x > 0;`** to prevent infinite values for $y = log(x)$. Use **`BOUNDS x > 1E-8;`** instead.

If the CALIS procedure encounters negative diagonal elements in the central model matrices during the minimization process, serious convergence problems can occur. You can use the BOUNDS statement to constrain these parameters to nonnegative values. Using negative values in these locations can lead to a smaller χ^2 value but uninterpretable estimates.

BY Statement

BY *variables* **;**

You can specify a BY statement with PROC CALIS 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 by using the SORT procedure with a similar BY statement.
- Specify the BY statement option NOTSORTED or DESCENDING in the BY statement for the CALIS 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 by using the DATASETS procedure.

For more information about the BY statement, see *SAS Language Reference: Concepts*. For more information about the DATASETS procedure, see the *Base SAS Procedures Guide*.

COSAN Model Statement

COSAN *matrix_term* < + *matrix_term*. . . > **;**

where *matrix_term* represents
matrix_definition < ∗ *matrix_definition* . . . >
and *matrix_definition* represents
matrix_name (*column_number* < ,*general_form* < ,*transformation* >>)

The COSAN statement constructs the symmetric matrix model for the covariance analysis mentioned earlier (see the section "The Generalized COSAN Model" on page 832):

$$\mathbf{C} = \mathbf{F}_1\mathbf{P}_1\mathbf{F}_1' + \cdots + \mathbf{F}_m\mathbf{P}_m\mathbf{F}_m',$$

$$\mathbf{F}_k = \mathbf{F}_{k_1} \cdots \mathbf{F}_{k_{n(k)}}, \qquad \text{and} \quad \mathbf{P}_k = \mathbf{P}_k', \quad k = 1, \ldots, m$$

$$\mathbf{F}_{k_j} = \begin{cases} \mathbf{G}_{k_j} \\ \mathbf{G}_{k_j}^{-1} \\ (\mathbf{I} - \mathbf{G}_{k_j})^{-1} \end{cases} \quad j = 1, \ldots, n(k), \qquad \text{and} \quad \mathbf{P}_k = \begin{cases} \mathbf{Q}_k \\ \mathbf{Q}_k^{-1} \end{cases}$$

You can specify only one COSAN statement with each PROC CALIS statement. The COSAN statement contains m *matrix_terms* corresponding to the generalized COSAN formula. The *matrix_terms* are separated by plus signs (+) according to the addition of the terms within the model.

Each *matrix_term* of the COSAN statement contains the definitions of the first $n(k) + 1$ matrices, $\mathbf{F}_{k_j}$ and $\mathbf{P}_k$, separated by asterisks (*) according to the multiplication of the matrices within the term. The matrices $\mathbf{F}_k'$ of the right-hand-side product are redundant and are not specified within the COSAN statement.

Each *matrix_definition* consists of the name of the matrix (*matrix_name*), followed in parentheses by the number of columns of the matrix (*column_number*) and, optionally, one or two matrix properties, separated by commas, describing the form of the matrix.

The number of rows of the first matrix in each term is defined by the input correlation or covariance matrix. You can reorder and reduce the variables in the input moment matrix by using the VAR statement. The number of rows of the other matrices within the term is defined by the number of columns of the preceding matrix.

The first matrix property is indicated by *general_form*, which describes the general form of the matrix in the model. You can choose one of the following specifications of the first matrix property. The default first matrix property is GEN.

IDE	specifies an identity matrix; if the matrix is not square, this specification describes an identity submatrix followed by a rectangular zero submatrix.
ZID	specifies an identity matrix; if the matrix is not square, this specification describes a rectangular zero submatrix followed by an identity submatrix.
DIA	specifies a diagonal matrix; if the matrix is not square, this specification describes a diagonal submatrix followed by a rectangular zero submatrix.

ZDI	specifies a diagonal matrix; if the matrix is not square, this specification describes a rectangular zero submatrix followed by a diagonal submatrix.
LOW	specifies a lower triangular matrix; the matrix can be rectangular.
UPP	specifies an upper triangular matrix; the matrix can be rectangular.
SYM	specifies a symmetric matrix; the matrix cannot be rectangular.
GEN	specifies a general rectangular matrix (default).

The second matrix property is indicated by *transformation*, which describes the kind of inverse matrix transformation. If the second matrix property is omitted, no transformation is applied to the matrix.

INV	uses the inverse of the matrix.
IMI	uses the inverse of the difference between the identity and the matrix.

You cannot specify a nonsquare parameter matrix as an INV or IMI model matrix. Specifying a matrix of type DIA, ZDI, UPP, LOW, or GEN is not necessary if you do not use the *unspecified location* list in the corresponding MATRIX statements. After PROC CALIS processes the corresponding MATRIX statements, the matrix type DIA, ZDI, UPP, LOW, or GEN is recognized from the pattern of possibly nonzero elements. If you do not specify the first matrix property and you use the *unspecified location* list in a corresponding MATRIX statement, the matrix is recognized as a GEN matrix. You can also generate an IDE or ZID matrix by specifying a DIA, ZDI, or IMI matrix and by using MATRIX statements that define the pattern structure. However, PROC CALIS would be unable to take advantage of the fast algorithms that are available for IDE and ZID matrices in this case.

For example, to specify a second-order factor analysis model

$$\mathbf{S} = \mathbf{F}_1\mathbf{F}_2\mathbf{P}_2\mathbf{F}_2'\mathbf{F}_1' + \mathbf{F}_1\mathbf{U}_2^2\mathbf{F}_1' + \mathbf{U}_1^2$$

with $m_1 = 3$ first-order factors and $m_2 = 2$ second-order factors and with $n = 9$ variables, you can use the following COSAN statement:

```
cosan F1(3) * F2(2) * P2(2,SYM)+F1(3) * U2(3,DIA) * I1(3,IDE)
      +U1(9,DIA) * I2(9,IDE)
```

COV Statement

COV *assignment* < , *assignment* . . . > **;**

where *assignment* represents   *variables* < $*$ *variables2* >> = *pattern-definition*

The COV statement tells which covariances are parameters to estimate and which are fixed. The COV statement can be used only with the LINEQS statement. The COV statement differs from the STD statement only in the meaning of the left-hand-side *variables* list. You can specify only one COV statement with each LINEQS statement. The COV statement defines the off-diagonal elements of the central model matrix $\mathbf{\Phi}$. These elements correspond to the covariances of the exogenous variables and to the error covariances of the endogenous variables. Elements that are not defined are assumed to be zero. The *assignment*s in the COV statement must be separated by commas.

The *variables* list on the left-hand side of the equal sign should contain only names of variables that do not appear on the left-hand side of an equation in the LINEQS statement—that is, exogenous, error, and disturbance variables.

The *pattern-definition* on the right-hand side is similar to that used in the MATRIX statement. Each list element on the right-hand side defines the covariance of a pair of variables in the list on the left-hand side. A name on the right-hand side can be followed by a number inside parentheses that gives the initial value. A number on the right-hand side means that the corresponding covariance of the variable on the left-hand side is fixed. If the right-hand-side list is longer than the left-hand-side variable list, the right-hand-side list is shortened to the length of the variable list. If the right-hand-side list is shorter than the variable list, the right-hand-side list is filled with repetitions of the last item in the list.

You can use one of two alternatives to refer to parts of $\mathbf{\Phi}$. The first alternative uses only one variable list and refers to all distinct pairs of variables within the list. The second alternative uses two variable lists separated by an asterisk and refers to all pairs of variables among the two lists.

Within-List Covariances

Using k variable names in the *variables* list on the left-hand side of an equal sign in a COV statement means that the parameter list (*pattern-definition*) on the right-hand side refers to all $k(k-1)/2$ distinct variable pairs in the below-diagonal part of the $\mathbf{\Phi}$ matrix. Order is very important. The order relation between the left-hand-side variable pairs and the right-hand-side parameter list is illustrated by the following example:

```
COV E1-E4 = PHI1-PHI6 ;
```

This is equivalent to the following specification:

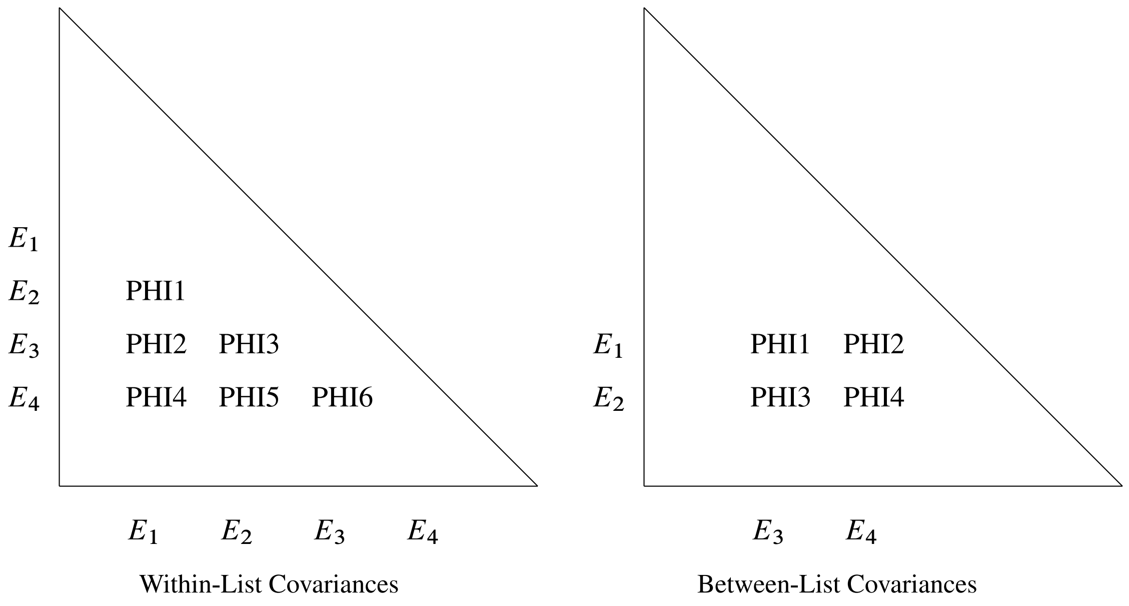
```
COV E2 E1 = PHI1,
    E3 E1 = PHI2, E3 E2 = PHI3,
    E4 E1 = PHI4, E4 E2 = PHI5, E4 E3 = PHI6;
```

The symmetric elements are generated automatically. When you use prefix names on the right-hand sides, you do not have to count the exact number of parameters. For example,

```
COV E1-E4 = PHI: ;
```

generates the same list of parameter names if the prefix PHI is not used in a previous statement. This is illustrated in the left panel of Figure 25.2. Integers starting from 1 are appended to the prefix PHI to form the parameter names for each of the within-list covariances.

Figure 25.2 Within-List and Between-List Covariances



Between-List Covariances

Using k_1 and k_2 variable names in the two lists (separated by an asterisk) on the left-hand side of an equal sign in a COV statement means that the parameter list on the right-hand side refers to all $k_1 \times k_2$ distinct variable pairs in the $\mathbf{\Phi}$ matrix. Order is very important. The order relation between the left-hand-side variable pairs and the right-hand-side parameter list is illustrated by the following example:

```
COV E1 E2 * E3 E4 = PHI1-PHI4 ;
```

This is equivalent to the following specification:

```
COV  E1 E3 = PHI1, E1 E4 = PHI2,
     E2 E3 = PHI3, E2 E4 = PHI4;
```

The symmetric elements are generated automatically. Using prefix names on the right-hand sides lets you achieve the same purpose without counting the number of parameters. That is,

```
COV  E1 E2 * E3 E4 = PHI: ;
```

This is illustrated in the right panel of Figure 25.2. Integers starting from 1 are appended to the prefix PHI to form the parameter names for each of the between-list covariances.

FACTOR Model Statement

FACTOR < *options* > **;**

You can use the FACTOR statement to specify an exploratory or confirmatory first-order factor analysis of the given covariance or correlation matrix **C**,

$$\mathbf{C} = \mathbf{F}\mathbf{F}' + \mathbf{U}, \quad \mathbf{U} = diag$$

or

$$\mathbf{C} = \mathbf{F}\mathbf{P}\mathbf{F}' + \mathbf{U}, \quad \mathbf{P} = \mathbf{P}'$$

where **U** is a diagonal matrix and **P** is symmetric. Within this section, n denotes the number of manifest variables corresponding to the rows and columns of matrix **C**, and m denotes the number of latent variables (factors or components) corresponding to the columns of the loading matrix **F**.

You can specify only one FACTOR statement with each PROC CALIS statement. You can specify higher-order factor analysis problems by using a COSAN model specification. PROC CALIS requires more computing time and memory than PROC FACTOR because it is designed for more general structural estimation problems and is unable to exploit the special properties of the unconstrained factor analysis model.

For default (exploratory) factor analysis, PROC CALIS computes initial estimates for factor loadings and unique variances by an algebraic method of approximate factor analysis. If you use a MATRIX statement together with a FACTOR model specification, initial values are computed by McDonald's (McDonald and Hartmann 1992) method (if possible). For details, see "Using the FACTOR and MATRIX Statements" on page 879. If neither of the two methods is appropriate, the initial values are set by the START= option.

The unrestricted factor analysis model is not identified because any orthogonal rotated factor loading matrix $\tilde{\mathbf{F}} = \mathbf{F}\mathbf{\Theta}$ is equivalent to the result **F**,

$$\mathbf{C} = \tilde{\mathbf{F}}\tilde{\mathbf{F}}' + \mathbf{U}, \qquad \tilde{\mathbf{F}} = \mathbf{F}\mathbf{\Theta}, \quad \text{where} \quad \mathbf{\Theta}'\mathbf{\Theta} = \mathbf{\Theta}\mathbf{\Theta}' = \mathbf{I}$$

To obtain an identified factor solution, the FACTOR statement imposes zero constraints on the $m(m-1)/2$ elements in the upper triangle of **F** by default.

The following options are available in the FACTOR statement.

COMPONENT | COMP

computes a component analysis instead of a factor analysis (the diagonal matrix **U** in the model is set to 0). Note that the rank of $\mathbf{F}\mathbf{F}'$ is equal to the number m of components in **F**. If m is smaller than the number of variables in the moment matrix **C**, the matrix of predicted model values is singular and maximum likelihood estimates for **F** cannot be computed. You should compute ULS estimates in this case.

HEYWOOD | HEY

constrains the diagonal elements of **U** to be nonnegative; in other words, the model is replaced by

$$\mathbf{C} = \mathbf{F}\mathbf{F}' + \mathbf{U}^2, \quad \mathbf{U} = diag$$

N=*m*

specifies the number of first-order factors or components. The number m of factors should not exceed the number n of variables in the covariance or correlation matrix analyzed. For the saturated model, $m = n$, the COMP option should generally be specified for $\mathbf{U} = 0$; otherwise, $df < 0$. For $m = 0$ no factor loadings are estimated, and the model is $\mathbf{C} = \mathbf{U}$, with $\mathbf{U} = diag$. By default, $m = 1$.

NORM < = KAISER | NONE >

Kaiser-normalizes the rows of the factor pattern for rotation. NORM=KAISER, which is the default, has exactly the same effect as NORM. You can turn off the normalization by NORM=NONE.

RCONVERGE=*p*

RCONV=*p*

specifies the convergence criterion for rotation cycles. The option is applicable to rotation by using either the QUARTIMAX, VARIMAX, EQUAMAX, or PARSIMAX method in the ROTATE= option. Rotation stops when the scaled change of the simplicity function value is less than the RCONVERGE= value. The default convergence criterion is

$$|f_{new} - f_{old}|/K < \epsilon$$

where f_{new} and f_{old} are simplicity function values of the current cycle and the previous cycle, respectively, $K = max(1, |f_{old}|)$ is a scaling factor, and ϵ is 1E–9 by default and is modified by the RCONVERGE= value.

RITER=*n*

specifies the maximum number of cycles n for factor rotation by using either the QUARTIMAX, VARIMAX, EQUAMAX, or PARSIMAX method in the ROTATE= option. The default n is the maximum between 10 times the number of variables and 100.

ROTATE | R=*name*

specifies an orthogonal rotation. By default, ROTATE=NONE. The possible values for *name* are as follows:

PRINCIPAL | PC	specifies a principal axis rotation. If ROTATE=PRINCIPAL is used with a factor rather than a component model, the following rotation is performed: $$\mathbf{F}_{new} = \mathbf{F}_{old}\mathbf{T}, \quad \text{with} \quad \mathbf{F}'_{old}\mathbf{F}_{old} = \mathbf{T}\mathbf{\Lambda}\mathbf{T}'$$ where the columns of matrix $\mathbf{T}$ contain the eigenvectors of $\mathbf{F}'_{old}\mathbf{F}_{old}$.
QUARTIMAX | Q	specifies quartimax rotation.
VARIMAX | V	specifies varimax rotation.
EQUAMAX | E	specifies equamax rotation.
PARSIMAX | P	specifies parsimax rotation.
NONE	performs no rotation (default).

Using the FACTOR and MATRIX Statements

You can specify the MATRIX statement and the FACTOR statement to compute a confirmatory first-order factor or component analysis. You can define the elements of the matrices **F**, **P**, and **U** of the oblique model,

$$\mathbf{C} = \mathbf{F}\mathbf{P}\mathbf{F}' + \mathbf{U}^2, \quad \mathbf{P} = \mathbf{P}', \quad \mathbf{U} = diag$$

To specify the structure for matrix **F**, **P**, or **U**, you have to refer to the matrix _F_ , _P_ , or _U_ in the MATRIX statement. Matrix names automatically set by PROC CALIS always start with an underscore. As you name your own matrices or variables, you should avoid leading underscores.

The default matrix forms are as follows:

_F_	lower triangular matrix (0 upper triangle for problem identification, removing rotational invariance)
_P_	identity matrix (constant)
_U_	diagonal matrix

For details about specifying the elements in matrices, see the section "MATRIX Statement" on page 883. If you are using at least one MATRIX statement in connection with a FACTOR model statement, you can also use the BOUNDS or PARAMETERS statement and programming statements to constrain the parameters named in the MATRIX statement. Initial estimates are computed by McDonald's (McDonald and Hartmann 1992) method. McDonald's method of computing initial values works better if you scale the factors by setting the factor variances to 1 rather than by setting the loadings of the reference variables equal to 1.

FREQ Statement

FREQ *variable* **;**

If one variable in your data set represents the frequency of occurrence for the other values in the observation, specify the variable's name in a FREQ statement. PROC CALIS then treats the data set as if each observation appears n_i times, where n_i is the value of the FREQ variable for observation i. Only the integer portion of the value is used. If the value of the FREQ variable is less than 1 or is missing, that observation is not included in the analysis. The total number of observations is considered to be the sum of the FREQ values when the procedure computes significance probabilities. You can use only one FREQ statement with each PROC CALIS statement.

LINCON Statement

LINCON *constraint* < , *constraint* . . . > **;**

where *constraint* represents one of the following:

- *number operator linear-term*
- *linear-term operator number*

and *linear-term* is

< + | − >< *coefficient* ∗ > *parameter* << + | − >< *coefficient* ∗ > *parameter* . . . >

The LINCON statement specifies a set of linear equality or inequality constraints of the form

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i , \quad i = 1, \ldots, m$$

The constraints must be separated by commas. Each linear constraint i in the statement consists of a linear combination $\sum_j a_{ij} x_j$ of a subset of the n parameters x_j, $j = 1, \ldots, n$, and a constant value b_i separated by a comparison operator. Valid operators are <=, <, >=, >, and = or, equivalently, LE, LT, GE, GT, and EQ. PROC CALIS cannot enforce the strict inequalities < or >. Note that the coefficients a_{ij} in the linear combination must be constant numbers and must be followed by an asterisk and the name of a parameter (for example, listed in the PARMS, STD, or COV statement). The following is an example of the LINCON statement that sets a linear constraint on parameters x1 and x2:

```
lincon      x1 + 3 * x2 <= 1;
```

Although you can easily express boundary constraints in LINCON statements, for many applications it is much more convenient to specify both the BOUNDS and LINCON statements in the same PROC CALIS call.

The LINCON statement can contain only parameter names, operators, and numerical constants. If you need to compute the values of the coefficients a_{ij} or right-hand sides b_i, you can run a preliminary DATA step and create a TYPE=EST data set containing _TYPE_='LE', _TYPE_='GE', or _TYPE_='EQ' observations, and then specify this data set as an INEST= or INVAR= data set in a subsequent PROC CALIS run.

LINEQS Model Statement

LINEQS *equation* < , *equation* . . . > **;**

where *equation* represents
dependent = *term* < + *term* . . . >
and each *term* represents one of the following:

- *coefficient-name* < (*number*) > *variable-name*
- *prefix-name* < (*number*) > *variable-name*
- < *number* > *variable-name*

The LINEQS statement defines the following LINEQS model:

$$
\begin{aligned}
\boldsymbol{\eta} &= \boldsymbol{\beta}\boldsymbol{\eta} + \boldsymbol{\gamma}\boldsymbol{\xi} \\
\mathbf{C} &= \mathbf{J}(\mathbf{I}-\mathbf{B})^{-1}\boldsymbol{\Gamma}\boldsymbol{\Phi}\boldsymbol{\Gamma}'((\mathbf{I}-\mathbf{B})^{-1})'\mathbf{J}'
\end{aligned}
$$

You can specify only one LINEQS statement with each PROC CALIS statement. There are some differences from Bentler's notation in choosing the variable names. The length of each variable name is restricted to eight characters. The names of the manifest variables are defined in the DATA= input data set. The VAR statement can be used to select a subset of manifest variables in the DATA= input data set to analyze. You do not need to use a V prefix for manifest variables in the LINEQS statement, nor do you need to use a numerical suffix in any variable name. The names of the latent variables must start with the prefix letter F (for Factor); the names of the residuals must start with the prefix letters E (for Error) or D (for Disturbance). The trailing part of the variable name can contain letters or digits. The prefix letter E is used for the errors of the manifest variables, and the prefix letter D is used for the disturbances of the latent variables. The names of the manifest variables in the DATA= input data set can start with F, E, or D, but these names should not coincide with the names of latent or error variables used in the model. The left-hand side (that is, endogenous *dependent* variable) of each equation should be either a manifest variable of the data set or a latent variable with prefix letter F. The left-hand-side variable should not appear on the right-hand side of the same equation; this means that matrix $\boldsymbol{\beta}$ should not have a nonzero diagonal element. Each equation should contain, at most, one E or D variable.

The equations must be separated by a comma. The order of the equations is arbitrary. The displayed output generally contains equations and terms in an order different from that of the input.

Coefficients to estimate are indicated in the equations by a name preceding the independent variable's name. The coefficient's name can be followed by a number inside parentheses indicating the initial value for this coefficient. A number preceding the independent variable's name indicates a constant coefficient. If neither a coefficient name nor a number precedes the independent variable's name, a constant coefficient of 1 is assumed.

If the initial value of a parameter is not specified in the equation, the initial value is chosen in one of the following ways:

- If you specify the RANDOM= option in the PROC CALIS statement, the variable obtains a randomly generated initial value r, such that $0 \leq r \leq 1$. The uninitialized parameters in the diagonals of the central model matrices are given the nonnegative random values *r* multiplied by 10, 100, or the value specified in the DEMPHAS= option.

- If the RANDOM= option is not used, PROC CALIS tries to estimate the initial values.
- If the initial values cannot be estimated, the value of the START= option is used as an initial value.

In Bentler's notation, estimated coefficients are indicated by asterisks. Referring to a parameter in Bentler's notation requires the specification of two variable names that correspond to the row and column of the position of the parameter in the matrix. Specifying the estimated coefficients by parameter names makes it easier to impose additional constraints with code. You do not need any additional statements to express equality constraints. Simply specify the same name for parameters that should have equal values.

If your model contains many unconstrained parameters and it is too cumbersome to find different parameter names, you can specify all those parameters by the same prefix name. A prefix is a short name followed by a colon. The CALIS procedure then generates a parameter name by appending an integer suffix to this prefix name. The prefix name should have no more than five or six characters so that the generated parameter name is not longer than eight characters. To avoid unintentional equality constraints, the prefix names should not coincide with explicitly defined parameter names.

For example, consider the following model described in the section "Specifying a Second-Order Factor-Analysis Model" on page 885:

$$\mathbf{S} = \mathbf{F}_1\mathbf{F}_2\mathbf{P}_2\mathbf{F}_2'\mathbf{F}_1' + \mathbf{F}_1\mathbf{U}_2^2\mathbf{F}_1' + \mathbf{U}_1^2$$

You can fit this model by using the LINEQS and STD statements:

```
lineqs
       V1 = X1 F1 + E1,
       V2 = X2 F1 + E2,
       V3 = X3 F1 + E3,
       V4 = X4 F2 + E4,
       V5 = X5 F2 + E5,
       V6 = X6 F2 + E6,
       V7 = X7 F3 + E7,
       V8 = X8 F3 + E8,
       V9 = X9 F3 + E9,
       F1 = Y1 F4 + D1,
       F2 = Y1 F4 + Y2 F5 + D2,
       F3 = Y2 F5 + D3;

std
       E1-E9 = 9 * U:,
       D1-D3 = 3 * V:,
       F4 F5 = 2 * P;
run;
```

MATRIX Statement

MATRIX *matrix-name < location > = list < , location = list . . . >* **;**

You can specify one or more MATRIX statements with a COSAN or FACTOR statement. A MATRIX statement specifies which elements of the matrix are constant and which are parameters. You can also assign values to the constant elements and initial values for the parameters. The input notation resembles that used in the COSAN program of McDonald and Fraser (personal communication), except that in PROC CALIS, parameters are distinguished from constants by giving parameters names instead of by using positive and negative integers.

A MATRIX statement cannot be used for an IDE or ZID matrix. For all other types of matrices, each element is assumed to be a constant of 0 unless a MATRIX statement specifies otherwise. Hence, there must be at least one MATRIX statement for each matrix mentioned in the COSAN statement except for IDE and ZID matrices. There can be more than one MATRIX statement for a given matrix. If the same matrix element is given different definitions, later definitions override earlier definitions.

At the start, all elements of each model matrix, except IDE or ZID matrices, are set equal to 0.

Description of *location*:

There are several ways to specify the starting *location* and continuation direction of a *list* with $n+1$, $n \geq 0$, elements within the parameter matrix.

[*i* , *j*] The *list* elements correspond to the diagonally continued matrix elements [*i*,*j*], [*i*+1,*j*+1], . . . , [*i*+*n*,*j*+*n*]. The number of elements is defined by the length of the list and eventually terminated by the matrix boundaries. If the list contains just one element (constant or variable), then it is assigned to the matrix element [*i*,*j*].

[*i* ,] The *list* elements correspond to the horizontally continued matrix elements [*i*,*j*], [*i*,*j*+1], . . . , [*i*,*j*+*n*], where the starting column *j* is the diagonal position for a DIA, ZDI, or UPP matrix and is the first column for all other matrix types. For a SYM matrix, the list elements refer only to the matrix elements in the lower triangle. For a DIA or ZDI matrix, only one list element is accepted.

[, *j*] The *list* elements correspond to the vertically continued matrix elements [*i*,*j*], [*i*+1, *j*], . . . , [*i*+*n*,*j*], where the starting row *i* is equal to the diagonal position for a DIA, ZDI, SYM, or LOW matrix and is the first row for each other matrix type. For a SYM matrix, the list elements refer only to the matrix elements in the lower triangle. For a DIA or ZDI matrix, only one list element is accepted.

[,] unspecified location: The *list* is allocated to all valid matrix positions (except for a ZDI matrix) starting at the element [1,1] and continuing rowwise. The only valid matrix positions for a DIA or ZDI matrix are the diagonal elements; for an UPP or LOW matrix, the valid positions are the elements above or below the diagonal; and for a symmetric matrix, the valid positions are the elements in the lower triangle since the other triangle receives the symmetric allocation automatically. This *location* definition differs from the definitions with specified pattern locations in one important respect: if the number

of elements in the *list* is smaller than the number of valid matrix elements, the list is repeated in the allocation process until all valid matrix elements are filled.

Omitting the left-hand-side term is equivalent to using [,] for an *unspecified location*.

Description of *list*:

The *list* contains numeric values or parameter names, or both, that are assigned to a list of matrix elements starting at a specified position and proceeding in a specified direction. A real number r in the list defines the corresponding matrix element as a constant element with this value. The notation $n * r$ generates n values of r in the list. A name in the list defines the corresponding matrix element as a parameter to be estimated. You can use numbered name lists (X1-X10) or the asterisk notation (5 *X means five occurrences of the parameter X). If a sublist of n_1 names inside a *list* is followed by a list of $n_2 \leq n_1$ real values inside parentheses, the last n_2 parameters in the name sublist are given the initial values mentioned inside the parentheses. For example, the *list*

```
0. 1. A2-A5 (1.4 1.9 2.5) 5.
```

specifies that the first two matrix elements (specified by the *location* to the left of the equal sign) are constants with values 0 and 1. The next element is parameter A2 with no specified initial value. The next three matrix elements are the variable parameters A3, A4, and A5 with initial values 1.4, 1.9, and 2.5, respectively. The next matrix element is specified by the seventh list element to be the constant 5.

If your model contains many unconstrained parameters and it is too cumbersome to find different parameter names, you can specify all those parameters by the same prefix name. A prefix is a short name followed by a colon. The CALIS procedure generates a parameter name by appending an integer suffix to this prefix name. The prefix name should have no more than five or six characters so that the generated parameter name is not longer than eight characters. For example, if the prefix A (the parameter A1) is already used once in a *list*, the previous example would be identical to

```
0. 1. 4 * A: (1.4 1.9 2.5) 5.
```

To avoid unintentional equality constraints, the prefix names should not coincide with explicitly defined parameter names.

If you do not assign initial values to the parameters (listed in parentheses following a name sublist within the pattern list), PROC CALIS assigns initial values as follows:

- If the PROC CALIS statement contains a START=*r* option, each uninitialized parameter is given the initial value r. The uninitialized parameters in the diagonals of the central model matrices are given the initial value $10|r|$, $100|r|$, or $|r|$ multiplied by the value specified in the DEMPHAS= option.
- If the PROC CALIS statement contains a RANDOM=*i* option, each uninitialized parameter is given a random initial value $0 \leq r \leq 1$. The uninitialized parameters in the diagonals of the central model matrices are given the random values multiplied by 10, 100, or the value specified in the DEMPHAS= option.
- Otherwise, the initial value is set corresponding to START=0.5.

Specifying a Second-Order Factor-Analysis Model

For example, to specify a confirmatory second-order factor analysis model

$$\mathbf{S} = \mathbf{F}_1\mathbf{F}_2\mathbf{P}_2\mathbf{F}_2'\mathbf{F}_1' + \mathbf{F}_1\mathbf{U}_2^2\mathbf{F}_1' + \mathbf{U}_1^2$$

with $m_1 = 3$ first-order factors, $m_2 = 2$ second-order factors, and $n = 9$ variables and the following matrix pattern,

$$\mathbf{F}_1 = \begin{pmatrix} X_1 & 0 & 0 \\ X_2 & 0 & 0 \\ X_3 & 0 & 0 \\ 0 & X_4 & 0 \\ 0 & X_5 & 0 \\ 0 & X_6 & 0 \\ 0 & 0 & X_7 \\ 0 & 0 & X_8 \\ 0 & 0 & X_9 \end{pmatrix}, \quad \mathbf{U}_1 = \begin{pmatrix} U_1 & & & & & & & & \\ & U_2 & & & & & & & \\ & & U_3 & & & & & & \\ & & & U_4 & & & & & \\ & & & & U_5 & & & & \\ & & & & & U_6 & & & \\ & & & & & & U_7 & & \\ & & & & & & & U_8 & \\ & & & & & & & & U_9 \end{pmatrix}$$

$$\mathbf{F}_2 = \begin{pmatrix} Y_1 & 0 \\ Y_1 & Y_2 \\ 0 & Y_2 \end{pmatrix}, \quad \mathbf{P}_2 = \begin{pmatrix} P & 0 \\ 0 & P \end{pmatrix}, \quad \mathbf{U}_2 = \begin{pmatrix} V_1 & & \\ & V_2 & \\ & & V_3 \end{pmatrix}$$

you can specify the following COSAN and MATRIX statements:

```
cosan f1(3) * f2(2) * p2(2,dia) + f1(3) * u2(3,dia) * i1(3,ide)
      + u1(9,dia) * i2(9,ide);
matrix f1
         [ ,1]= x1-x3,
         [ ,2]= 3 * 0 x4-x6,
         [ ,3]= 6 * 0 x7-x9;
matrix u1
         [1,1]=u1-u9;
matrix f2
         [ ,1]= 2 * y1,
         [ ,2]= 0. 2 * y2;
matrix u2 = 3 * v:;
matrix p2 = 2 * p;
run;
```

The matrix pattern includes several equality constraints. Two loadings in the first and second factor of $\mathbf{F}_2$ (parameter names Y1 and Y2) and the two factor correlations in the diagonal of matrix $\mathbf{P}_2$ (parameter name P) are constrained to be equal. There are many other ways to specify the same model. See Figure 25.3 for the path diagram of this model.

The MATRIX statement can also be used with the FACTOR model statement. See "Using the FACTOR and MATRIX Statements" on page 879 for the usage.

NLINCON Statement

NLINCON | NLC *constraint* < , *constraint* . . . > **;**

where *constraint* represents one of the following:

- *number operator variable-list number operator*
- *variable-list operator number*
- *number operator variable-list*

You can specify nonlinear equality and inequality constraints with the NLINCON or NLC statement. The QUANEW optimization subroutine is used when you specify nonlinear constraints by using the NLINCON statement.

The syntax of the NLINCON statement is similar to that of the BOUNDS statement, except that the NLINCON statement must contain the names of variables that are defined in the programming statements and are defined as continuous functions of parameters in the model. They must not be confused with the variables in the data set.

As with the BOUNDS statement, one- or two-sided constraints are allowed in the NLINCON statement; equality constraints must be one-sided. Valid operators are $<=$, $<$, $>=$, $>$, and $=$ or, equivalently, LE, LT, GE, GT, and EQ.

PROC CALIS cannot enforce the strict inequalities < or > but instead treats them as $<=$ and $>=$, respectively. The listed nonlinear constraints must be separated by commas. The following is an example of the NLINCON statement that constrains the nonlinear parametric function $x_1 * x_1 + u_1$, which is defined in a programming statement, to a fixed value of 1:

```
nlincon    xx = 1;
xx = x1 * x1 + u1;
```

Note that x1 and u1 are parameters defined in the model. The following three NLINCON statements, which require xx1, xx2, and xx3 to be between 0 and 10, are equivalent:

```
nlincon  0. <= xx1-xx3,
               xx1-xx3 <= 10;
nlincon 0. <= xx1-xx3 <= 10.;
nlincon 10. >= xx1-xx3 >= 0.;
```

NLOPTIONS Statement

NLOPTIONS *option(s)* **;**

Many options that are available in PROC NLP can now be specified for the optimization subroutines in PROC CALIS by using the NLOPTIONS statement. The NLOPTIONS statement provides more displayed and file output on the results of the optimization process, and it permits the same set of termination criteria as in PROC NLP. These are more technical options that you might not need to specify in most cases. The available options are summarized in Table 25.2 through Table 25.4, and the options are described in detail in the following three sections.

Table 25.2 Options Documented in the PROC CALIS Statement

Option	Description
Estimation Methods	
G4=*i*	specifies the algorithm for computing standard errors
Optimization Techniques	
FCONV=*r*	specifies the relative change function convergence criterion
GCONV=*r*	specifies the relative gradient convergence criterion
INSTEP=*r*	specifies the initial step length (SALPHA=, RADIUS=)
LINESEARCH=*i*	specifies the line-search method
LSPRECISION=*r*	specifies the line-search precision
MAXFUNC=*i*	specifies the maximum number of function calls
MAXITER=*i* <*n*>	specifies the maximum number of iterations
TECHNIQUE=*name*	specifies the minimization method
UPDATE=*name*	specifies the update technique
Miscellaneous Options	
ASINGULAR=*r*	specifies the absolute singularity criterion for inversion of the information matrix
COVSING=*r*	specifies the singularity tolerance of the information matrix
MSINGULAR=*r*	specifies the relative M singularity criterion for inversion of the information matrix
SINGULAR=*r*	specifies the singularity criterion for inversion of the Hessian
VSINGULAR=*r*	specifies the relative V singularity criterion for inversion of the information matrix

Table 25.3 Termination Criteria Options

Option	Description
Options Used by All Techniques	
ABSCONV=*r*	specifies the absolute function convergence criterion
MAXFUNC=*i*	specifies the maximum number of function calls
MAXITER=*i* <*n*>	specifies the maximum number of iterations
MAXTIME=*r*	specifies the maximum CPU time
MINITER=*i*	specifies the minimum number of iterations
Options for Unconstrained and Linearly Constrained Techniques	
ABSFCONV=*r* <*n*>	specifies the absolute change function convergence criterion
ABSGCONV=*r* <*n*>	specifies the absolute gradient convergence criterion
ABSXCONV=*r* <*n*>	specifies the absolute change parameter convergence criterion
FCONV=*r* <*n*>	specifies the relative change function convergence criterion
FCONV2=*r* <*n*>	specifies the function convergence criterion

Table 25.3 *continued*

Option	**Description**
FDIGITS=*r*	specifies the precision in computation of the objective function
FSIZE=*r*	specifies the parameter for FCONV= and GCONV=
GCONV=*r* <*n*>	specifies the relative gradient convergence criterion
GCONV2=*r* <*n*>	specifies the relative gradient convergence criterion
XCONV=*r* <*n*>	specifies the relative change parameter convergence criterion
XSIZE=*r*	specifies the parameter for XCONV=
Options for Nonlinearly Constrained Techniques	
ABSGCONV=*r* <*n*>	specifies the maximum absolute gradient of Lagrange function criterion
FCONV2=*r* <*n*>	specifies the predicted objective function reduction criterion
GCONV=*r* <*n*>	specifies the normalized predicted objective function reduction criterion

Table 25.4 Miscellaneous Options

Option	**Description**
Options for the Approximate Covariance Matrix of Parameter Estimates	
CFACTOR=*r*	specifies the scalar factor for standard errors
NOHLF	uses the Hessian of the objective function for standard errors
Options for Additional Displayed Output	
PALL	displays the initial and final optimization values
PCRPJAC	displays the approximate Hessian matrix
PHESSIAN	displays the Hessian matrix
PHISTORY	displays the optimization history
PINIT	displays the initial values and derivatives
PNLCJAC	displays the Jacobian matrix of nonlinear constraints
PRINT	displays the results of the optimization process
Additional Options for Optimization Techniques	
DAMPSTEP< =*r* >	controls the initial line-search step size
HESCAL=*n*	specifies the scaling version of Hessian or Jacobian
LCDEACT=*r*	specifies the Lagrange multiplier threshold of constraint
LCEPSILON=*r*	specifies the range for boundary and linear constraints
LCSINGULAR=*r*	specifies the QR decomposition linear dependence criterion
NOEIGNUM	suppresses the computation of matrices
RESTART=*i*	specifies the restart algorithm with a steepest descent direction
VERSION=1 | 2	specifies the quasi-Newton optimization technique version

Options Documented in the PROC CALIS Statement

The following options are the same as in the PROC CALIS statement and are documented in the section "PROC CALIS Statement" on page 848.

Estimation Method Option

G4=*i*

specifies the method for computing the generalized (G2 or G4) inverse of a singular matrix needed for the approximate covariance matrix of parameter estimates. This option is valid only for applications where the approximate covariance matrix of parameter estimates is found to be singular.

Optimization Technique Options

FCONV | FTOL=*r*

specifies the relative function convergence criterion. For more details, see the section "Termination Criteria Options" on page 890.

GCONV | GTOL=*r*

specifies the relative gradient convergence criterion. For more details, see the section "Termination Criteria Options" on page 890.

INSTEP | SALPHA | RADIUS=*r*

restricts the step length of an optimization algorithm during the first iterations.

LINESEARCH | LIS=*i*

specifies the line-search method for the CONGRA, QUANEW, and NEWRAP optimization techniques.

LSPRECISION | LSP=*r*

specifies the degree of accuracy that should be obtained by the line-search algorithms LIS=2 and LIS=3.

MAXFUNC | MAXFU=*i*

specifies the maximum number i of function calls in the optimization process. For more details, see the section "Termination Criteria Options" on page 890.

MAXITER | MAXIT=*i < n >*

specifies the maximum number i of iterations in the optimization process. For more details, see the section "Termination Criteria Options" on page 890.

TECHNIQUE | TECH=*name*

OMETHOD | OM=*name*

specifies the optimization technique.

UPDATE | UPD=*name*

specifies the update method for the quasi-Newton or conjugate-gradient optimization technique.

Miscellaneous Options

ASINGULAR | ASING=*r*

specifies an absolute singularity criterion r, $r > 0$, for the inversion of the information matrix, which is needed to compute the approximate covariance matrix of parameter estimates.

COVSING=*r*

specifies a nonnegative threshold r, $r > 0$, that decides whether the eigenvalues of the information matrix are considered to be zero. This option is valid only for applications where the approximate covariance matrix of parameter estimates is found to be singular.

MSINGULAR | MSING=*r*

specifies a relative singularity criterion r, $r > 0$, for the inversion of the information matrix, which is needed to compute the approximate covariance matrix of parameter estimates.

SINGULAR | SING =*r*

specifies the singularity criterion r, $0 \leq r \leq 1$, that is used for the inversion of the Hessian matrix. The default value is 1E−8.

VSINGULAR | VSING=*r*

specifies a relative singularity criterion r, $r > 0$, for the inversion of the information matrix, which is needed to compute the approximate covariance matrix of parameter estimates.

Termination Criteria Options

Let x^* be the point at which the objective function $f(\cdot)$ is optimized, and let $x^{(k)}$ be the parameter values attained at the kth iteration. All optimization techniques stop at the kth iteration if at least one of a set of termination criteria is satisfied. The specified termination criteria should enable termination in an area of sufficient size around x^*. You can avoid termination respective to any of the following function, gradient, or parameter criteria by setting the corresponding option to zero. There is a default set of termination criteria for each optimization technique; most of these default settings make the criteria ineffective for termination. PROC CALIS might have problems due to rounding errors (especially in derivative evaluations) that prevent an optimizer from satisfying strong termination criteria.

Note that PROC CALIS also terminates if the point $x^{(k)}$ is fully constrained by linearly independent active linear or boundary constraints, and all Lagrange multiplier estimates of active inequality constraints are greater than a small negative tolerance.

The following options are available only in the NLOPTIONS statement (except for FCONV, GCONV, MAXFUNC, and MAXITER), and they affect the termination criteria.

Options Used by All Optimization Techniques

ABSCONV | ABSTOL=*r*

specifies an absolute function convergence criterion.

- For minimization, termination requires

$$f^{(k)} = f(x^{(k)}) \leq ABSCONV$$

- For maximization, termination requires

$$f^{(k)} = f(x^{(k)}) \geq ABSCONV$$

The default value of ABSCONV is as follows:

- for minimization, the negative square root of the largest double-precision value
- for maximization, the positive square root of the largest double-precision value

MAXFUNC | MAXFU=*i*

requires the number of function calls to be no larger than i. The default values are listed in the following table.

TECH=	MAXFUNC Default
LEVMAR, NEWRAP, NRRIDG, TRUREG	i=125
DBLDOG, QUANEW	i=500
CONGRA	i=1000

The default is used if you specify MAXFUNC=0. The optimization can be terminated only after completing a full iteration. Therefore, the number of function calls that are actually performed can exceed the number that is specified by the MAXFUNC= option.

MAXITER | MAXIT= *i* < *n* >

requires the number of iterations to be no larger than i. The default values are listed in the following table.

TECH=	MAXITER Default
LEVMAR, NEWRAP, NRRIDG, TRUREG	i=50
DBLDOG, QUANEW	i=200
CONGRA	i=400

The default is used if you specify MAXITER=0 or you omit the MAXITER option.

The optional second value n is valid only for TECH=QUANEW with nonlinear constraints. It specifies an upper bound n for the number of iterations of an algorithm and reduces the violation of nonlinear constraints at a starting point. The default value is n=20. For example, specifying `MAXITER= .  0` means that you do not want to exceed the default number of iterations during the main optimization process and that you want to suppress the feasible-point algorithm for nonlinear constraints.

MAXTIME=*r*

requires the CPU time to be no larger than r. The default value of the MAXTIME= option is the largest double floating-point number on your computer.

MINITER | MINIT=*i*

specifies the minimum number of iterations. The default value is $i = 0$.

The ABSCONV=, MAXITER=, MAXFUNC=, and MAXTIME= options are useful for dividing a time-consuming optimization problem into a series of smaller problems by using the OUTEST= and INEST= data sets.

Options for Unconstrained and Linearly Constrained Optimization Techniques

ABSFCONV | ABSFTOL=*r < n>*

specifies the absolute function convergence criterion. Termination requires a small change of the function value in successive iterations,

$$|f(x^{(k-1)}) - f(x^{(k)})| \leq r$$

The default value is $r = 0$. The optional integer value n determines the number of successive iterations for which the criterion must be satisfied before the process can be terminated.

ABSGCONV | ABSGTOL=*r < n>*

specifies the absolute gradient convergence criterion. Termination requires the maximum absolute gradient element to be small,

$$\max_j |g_j^{(k)}| \leq r$$

The default value is r=1E−5. The optional integer value n determines the number of successive iterations for which the criterion must be satisfied before the process can be terminated.

NOTE: In some applications, the small default value of the ABSGCONV= criterion is too difficult to satisfy for some of the optimization techniques.

ABSXCONV | ABSXTOL=*r < n>*

specifies the absolute parameter convergence criterion. Termination requires a small Euclidean distance between successive parameter vectors,

$$\parallel x^{(k)} - x^{(k-1)} \parallel_2 \leq r$$

The default value is $r = 0$. The optional integer value n determines the number of successive iterations for which the criterion must be satisfied before the process can be terminated.

FCONV | FTOL=*r < n>*

specifies the relative function convergence criterion. Termination requires a small relative change of the function value in successive iterations,

$$\frac{|f(x^{(k)}) - f(x^{(k-1)})|}{\max(|f(x^{(k-1)})|, FSIZE)} \leq r$$

where $FSIZE$ is defined by the FSIZE= option. The default value is $r = 10^{-FDIGITS}$, where $FDIGITS$ either is specified or is set by default to $-log_{10}(\epsilon)$, where ϵ is the machine precision. The optional integer value n determines the number of successive iterations for which the criterion must be satisfied before the process can be terminated.

FCONV2 | FTOL2=*r < n>*

specifies another function convergence criterion. For least squares problems, termination requires a small predicted reduction

$$df^{(k)} \approx f(x^{(k)}) - f(x^{(k)} + s^{(k)})$$

of the objective function.

The predicted reduction

$$\begin{aligned} df^{(k)} &= -g^{(k)\prime}s^{(k)} - \frac{1}{2}s^{(k)\prime}G^{(k)}s^{(k)} \\ &= -\frac{1}{2}s^{(k)\prime}g^{(k)} \\ &\le r \end{aligned}$$

is computed by approximating the objective function f by the first two terms of the Taylor series and substituting the Newton step

$$s^{(k)} = -G^{(k)-1}g^{(k)}$$

The FCONV2 criterion is the unscaled version of the GCONV criterion. The default value is $r = 0$. The optional integer value n determines the number of successive iterations for which the criterion must be satisfied before the process can be terminated.

FDIGITS=*r*

specifies the number of accurate digits in evaluations of the objective function. Fractional values such as FDIGITS=4.7 are allowed. The default value is $r = -log_{10}\epsilon$, where ϵ is the machine precision. The value of r is used for the specification of the default value of the FCONV= option.

FSIZE=*r*

specifies the $FSIZE$ parameter of the relative function and relative gradient termination criteria. The default value is $r = 0$. See the FCONV= and GCONV= options for details.

GCONV | GTOL=*r* <*n*>

specifies the relative gradient convergence criterion. For all techniques except the CONGRA technique, termination requires that the normalized predicted function reduction be small,

$$\frac{[g^{(k)}]'[G^{(k)}]^{-1}g^{(k)}}{\max(|f(x^{(k)})|, FSIZE)} \le r$$

where $FSIZE$ is defined by the FSIZE= option. For the CONGRA technique (where a reliable Hessian estimate **G** is not available),

$$\frac{\| g^{(k)} \|_2^2 \quad \| s^{(k)} \|_2}{\| g^{(k)} - g^{(k-1)} \|_2 \max(|f(x^{(k)})|, FSIZE)} \le r$$

is used. The default value is r=1E−8. The optional integer value n determines the number of successive iterations for which the criterion must be satisfied before the process can be terminated.

NOTE: The default setting for the GCONV= option sometimes leads to early termination far from the location of the optimum. This is especially true for the special form of this criterion used in the CONGRA optimization.

GCONV2 | GTOL2=*r* <*n*>

specifies another relative gradient convergence criterion. For least squares problems and the

TRUREG, LEVMAR, NRRIDG, and NEWRAP techniques, the criterion of Browne (1982) is used,

$$\max_j \frac{|g_j^{(k)}|}{\sqrt{f(x^{(k)})G_{j,j}^{(k)}}} \leq r$$

This criterion is not used by the other techniques. The default value is $r = 0$. The optional integer value n determines the number of successive iterations for which the criterion must be satisfied before the process can be terminated.

XCONV | XTOL=*r < n>*

specifies the relative parameter convergence criterion. Termination requires a small relative parameter change in subsequent iterations,

$$\frac{\max_j |x_j^{(k)} - x_j^{(k-1)}|}{\max(|x_j^{(k)}|, |x_j^{(k-1)}|, XSIZE)} \leq r$$

The default value is $r = 0$. The optional integer value n determines the number of successive iterations for which the criterion must be satisfied before the process can be terminated.

XSIZE=*r*

specifies the $XSIZE$ parameter of the relative function and relative gradient termination criteria. The default value is $r = 0$. See the XCONV= option for details.

Options for Nonlinearly Constrained Techniques

The non-NMSIMP algorithms available for nonlinearly constrained optimization (currently only TECH=QUANEW) do not monotonically reduce either the value of the objective function or some kind of merit function that combines objective and constraint functions. Furthermore, the algorithm uses the watchdog technique with backtracking (Chamberlain et al., 1982). Therefore, no termination criteria are implemented that are based on the values (x or f) of successive iterations. In addition to the criteria used by all optimization techniques, only three more termination criteria are currently available, and they are based on the Lagrange function:

$$L(x, \lambda) = f(x) - \sum_{i=1}^{m} \lambda_i c_i(x)$$

and its gradient

$$\nabla_x L(x, \lambda) = g(x) - \sum_{i=1}^{m} \lambda_i \nabla_x c_i(x)$$

Here, m denotes the total number of constraints, $g = g(x)$ denotes the gradient of the objective function, and λ denotes the m vector of Lagrange multipliers. The Kuhn-Tucker conditions require that the gradient of the Lagrange function is zero at the optimal point (x^*, λ^*):

$$\nabla_x L(x^*, \lambda^*) = 0$$

The termination criteria available for nonlinearly constrained optimization follow.

ABSGCONV | ABSGTOL=*r < n>*

specifies that termination requires the maximum absolute gradient element of the Lagrange function to be small,

$$\max_j |\{\nabla_x L(x^{(k)}, \lambda^{(k)})\}_j| \le r$$

The default value is r=1E−5. The optional integer value n determines the number of successive iterations for which the criterion must be satisfied before the process can be terminated.

FCONV2 | FTOL2=*r < n>*

specifies that termination requires the predicted objective function reduction to be small:

$$|g(x^{(k)})s(x^{(k)})| + \sum_{i=1}^{m} |\lambda_i c_i| \le r$$

The default value is r=1E−6. This is the criterion used by the programs VMCWD and VF02AD (Powell 1982b). The optional integer value n determines the number of successive iterations for which the criterion must be satisfied before the process can be terminated.

GCONV | GTOL=*r < n>*

specifies that termination requires the normalized predicted objective function reduction to be small:

$$\frac{|g(x^{(k)})s(x^{(k)})| + \sum_{i=1}^{m} |\lambda_i c_i(x^{(k)})|}{\max(|f(x^{(k)})|, FSIZE)} \le r$$

where $FSIZE$ is defined by the FSIZE= option. The default value is r=1E−8. The optional integer value n determines the number of successive iterations for which the criterion must be satisfied before the process can be terminated.

Miscellaneous Options

Options for the Approximate Covariance Matrix of Parameter Estimates

CFACTOR=*r*

specifies the scalar factor for the covariance matrix of parameter estimates. The scalar $r \ge 0$ replaces the default value c/NM. For more details, see the section "Approximate Standard Errors" on page 928.

NOHLF

specifies that the Hessian matrix of the objective function (rather than the Hessian matrix of the Lagrange function) be used for computing the approximate covariance matrix of parameter estimates and, therefore, the approximate standard errors.

It is theoretically not correct to use the NOHLF option. However, since most implementations use the Hessian matrix of the objective function and not the Hessian matrix of the Lagrange function for computing approximate standard errors, the NOHLF option can be used to compare the results.

Options for Additional Displayed Output

PALL | ALL
displays information about the starting values and final values of the optimization process.

PCRPJAC | PJTJ
displays the approximate Hessian matrix. If general linear or nonlinear constraints are active at the solution, the projected approximate Hessian matrix is also displayed.

PHESSIAN | PHES
displays the Hessian matrix. If general linear or nonlinear constraints are active at the solution, the projected Hessian matrix is also displayed.

PHISTORY | PHIS
displays the optimization history. The PHISTORY option is set automatically if the PALL or PRINT option is set.

PINIT | PIN
displays the initial values and derivatives (if available). The PINIT option is set automatically if the PALL option is set.

PNLCJAC
displays the Jacobian matrix of nonlinear constraints specified by the NLINCON statement. The PNLCJAC option is set automatically if the PALL option is set.

PRINT | PRI
displays the results of the optimization process, such as parameter estimates and constraints.

Options for Fine-Tuning Optimization Techniques

DAMPSTEP | DS *< =r >*
specifies that the initial step-size value $\alpha^{(0)}$ for each line search (used by the QUANEW, CONGRA, or NEWRAP technique) cannot be larger than r times the step-size value used in the former iteration. If the factor r is not specified, the default value is $r = 2$. The DAMPSTEP option can prevent the line-search algorithm from repeatedly stepping into regions where some objective functions are difficult to compute or where they can lead to floating-point overflows during the computation of objective functions and their derivatives. The DAMPSTEP<=*r*> option can prevent time-costly function calls during line searches with very small step sizes α of objective functions. For more information about setting the start values of each line search, see the section "Restricting the Step Length" on page 952.

HESCAL | HS = 0 | 1 | 2 | 3
specifies the scaling version of the Hessian or crossproduct Jacobian matrix used in NRRIDG, TRUREG, LEVMAR, NEWRAP, or DBLDOG optimization. If HS is not equal to zero, the first iteration and each restart iteration set the diagonal scaling matrix $\mathbf{D}^{(0)} = diag(d_i^{(0)})$:

$$d_i^{(0)} = \sqrt{\max(|\mathbf{G}_{i,i}^{(0)}|, \epsilon)}$$

where $\mathbf{G}_{i,i}^{(0)}$ are the diagonal elements of the Hessian or crossproduct Jacobian matrix. In every other iteration, the diagonal scaling matrix $\mathbf{D}^{(0)} = diag(d_i^{(0)})$ is updated depending on the HS option:

HS=0 specifies that no scaling is done.

HS=1 specifies the Moré (1978) scaling update:

$$d_i^{(k+1)} = \max(d_i^{(k)}, \sqrt{\max(|\mathbf{G}_{i,i}^{(k)}|, \epsilon)})$$

HS=2 specifies the Dennis, Gay, and Welsch (1981) scaling update:

$$d_i^{(k+1)} = \max(0.6 * d_i^{(k)}, \sqrt{\max(|\mathbf{G}_{i,i}^{(k)}|, \epsilon)})$$

HS=3 specifies that d_i is reset in each iteration:

$$d_i^{(k+1)} = \sqrt{\max(|\mathbf{G}_{i,i}^{(k)}|, \epsilon)}$$

In the preceding equations, ϵ is the relative machine precision. The default is HS=1 for LEVMAR minimization and HS=0 otherwise. Scaling of the Hessian or crossproduct Jacobian can be time-consuming in the case where general linear constraints are active.

LCDEACT | LCD=*r*

specifies a threshold r for the Lagrange multiplier that decides whether an active inequality constraint remains active or can be deactivated. For maximization, r must be greater than zero; for minimization, r must be smaller than zero. The default is

$$r = \pm \min(0.01, \max(0.1 * ABSGCONV, 0.001 * gmax^{(k)}))$$

where "+" stands for maximization, "−" stands for minimization, $ABSGCONV$ is the value of the absolute gradient criterion, and $gmax^{(k)}$ is the maximum absolute element of the (projected) gradient $g^{(k)}$ or $Z'g^{(k)}$.

LCEPSILON | LCEPS | LCE=*r*

specifies the range r, $r \geq 0$, for active and violated boundary and linear constraints. If the point $x^{(k)}$ satisfies the condition

$$|\sum_{j=1}^{n} a_{ij} x_j^{(k)} - b_i| \leq r * (|b_i| + 1)$$

the constraint i is recognized as an active constraint. Otherwise, the constraint i is either an inactive inequality or a violated inequality or equality constraint. The default value is r=1E−8. During the optimization process, the introduction of rounding errors can force PROC NLP to increase the value of r by factors of 10. If this happens, it is indicated by a message displayed in the log.

LCSINGULAR | LCSING | LCS=*r*

specifies a criterion r, $r \geq 0$, used in the update of the QR decomposition that decides whether an active constraint is linearly dependent on a set of other active constraints. The default is r=1E−8. The larger r becomes, the more the active constraints are recognized as being linearly dependent.

NOEIGNUM

suppresses the computation and displayed output of the determinant and the inertia of the Hessian, crossproduct Jacobian, and covariance matrices. The inertia of a symmetric matrix are the numbers of negative, positive, and zero eigenvalues. For large applications, the NOEIGNUM option can save computer time.

RESTART | REST=*i*

specifies that the QUANEW or CONGRA algorithm is restarted with a steepest descent/ascent search direction after at most i iterations, $i > 0$. Default values are as follows:

- CONGRA: UPDATE=PB: restart is done automatically so specification of i is not used.
- CONGRA: UPDATE≠PB: $i = \min(10n, 80)$, where n is the number of parameters.
- QUANEW: i is the largest integer available.

VERSION | VS = 1 | 2

specifies the version of the quasi-Newton optimization technique with nonlinear constraints.

VS=1	specifies the update of the μ vector as in Powell (1978a, 1978b) (update like VF02AD).
VS=2	specifies the update of the μ vector as in Powell (1982a, 1982b) (update like VMCWD).

The default is VS=2.

PARAMETERS Statement

PARAMETERS | PARMS *parameter(s)* << = > *number(s)* >
<< , > *parameter(s)* << = > *number(s)* > . . . > **;**

The PARAMETERS statement defines additional parameters that are not elements of a model matrix to use in your own programming statements. You can specify more than one PARAMETERS statement with each PROC CALIS statement. The *parameters* can be followed by an equal sign and a number list. The values of the *numbers* list are assigned as initial values to the preceding parameters in the *parameters* list. For example, each of the following statements assigns the initial values ALPHA=.5 and BETA=-.5 for the parameters used in programming statements:

```
parameters alfa beta=.5 -.5;
parameters alfa beta (.5 -.5);
parameters alfa beta .5 -.5;
parameters alfa=.5 beta (-.5);
```

The number of parameters and the number of values does not have to match. When there are fewer values than parameter names, either the RANDOM= or START= option is used. When there are more values than parameter names, the extra values are dropped. Parameters listed in the PARAMETERS statement can be assigned initial values by programming statements or by the START= or RANDOM= option in the PROC CALIS statement.

CAUTION: The OUTRAM= and INRAM= data sets do not contain any information about the PARAMETERS statement or additional programming statements.

PARTIAL Statement

PARTIAL *variables* **;**

If you want the analysis to be based on a partial correlation or covariance matrix, use the PARTIAL statement to list the variables used to partial out the variables in the analysis. You can specify only one PARTIAL statement with each PROC CALIS statement.

RAM Model Statement

RAM *list-entry* < , *list-entry* . . . > **;**

where *list-entry* represents
matrix-number row-number column-number < *value* > < *parameter-name* >

The RAM statement defines the elements of the symmetric RAM matrix model

$$\mathbf{v} = \mathbf{A}\mathbf{v} + \mathbf{u}$$

in the form of a list type input (McArdle and McDonald 1984).

The covariance structure is given by

$$\mathbf{C} = \mathbf{J}(\mathbf{I} - \mathbf{A})^{-1}\mathbf{P}((\mathbf{I} - \mathbf{A})^{-1})'\mathbf{J}'$$

with selection matrix **J** and

$$\mathbf{C} = \mathcal{E}\{\mathbf{J}\mathbf{v}\mathbf{v}'\mathbf{J}'\}, \qquad \mathbf{P} = \mathcal{E}\{\mathbf{u}\mathbf{u}'\}$$

You can specify only one RAM statement with each PROC CALIS statement. Using the RAM statement requires that the first n variable numbers in the path diagram and in the vector v correspond to the numbers of the n manifest variables of the given covariance or correlation matrix. If you are not sure what the order of the manifest variables in the DATA= data set is, use a VAR statement to specify the order of these observed variables. Using the AUGMENT option includes the INTERCEPT variable as a manifest variable with number $n + 1$ in the RAM model. In this case, latent variables have to start with $n + 2$. The box of each manifest variable in the path diagram is assigned the number of the variable in the covariance or correlation matrix.

The selection matrix **J** is always a rectangular identity (IDE) matrix, and it does not have to be specified in the RAM statement.

A constant matrix element is defined in a RAM statement by a *list-entry* with four numbers. You define a parameter element by three or four numbers followed by a name for the parameter. Separate the list entries with a comma. Each *list-entry* in the RAM statement corresponds to a path in the diagram, as follows:

- The first number in each list entry (*matrix-number*) is the number of arrowheads of the path, which is the same as the number of the matrix in the RAM model (1 := **A** , 2 := **P**).
- The second number in each list entry (*row-number*) is the number of the node in the diagram to which the path points, which is the same as the row number of the matrix element.
- The third number in each list entry (*column-number*) is the number of the node in the diagram from which the path originates, which is the same as the column number of the matrix element.
- The fourth number (*value*) gives the (initial) value of the path coefficient. If you do not specify a fifth *list-entry*, this number specifies a constant coefficient; otherwise, this number specifies the initial value of this parameter. It is not necessary to specify the fourth item. If you specify neither the fourth nor the fifth item, the constant is set to 1 by default. If the fourth item (*value*) is not specified for a parameter, PROC CALIS tries to compute an initial value for this parameter.
- If the path coefficient is a parameter rather than a constant, then a fifth item in the list entry (*parameter-name*) is required to assign a name to the parameter. Using the same name for different paths constrains the corresponding coefficients to be equal.

If the initial value of a parameter is not specified in the list, the initial value is chosen in one of the following ways:

- If the PROC CALIS statement contains a RANDOM=*i* option, then the parameter obtains a randomly generated initial value r, such that $0 \le r \le 1$. The uninitialized parameters in the diagonals of the central model matrices are given the random values r multiplied by 10, 100, or the value specified in the DEMPHAS= option.
- If the RANDOM= option is not used, PROC CALIS tries to estimate the initial values.
- If the initial values cannot be estimated, the value of the START= option is used as an initial value.

If your model contains many unconstrained parameters and it is too cumbersome to find different parameter names, you can specify all those parameters by the same prefix name. A prefix is a short name followed by a colon. The CALIS procedure then generates a parameter name by appending an integer suffix to this prefix name. The prefix name should have no more than five or six characters so that the generated parameter name is not longer than eight characters. To avoid unintentional equality constraints, the prefix names should not coincide with explicitly defined parameter names.

For example, consider the following model described in the section “Specifying a Second-Order Factor-Analysis Model” on page 885:

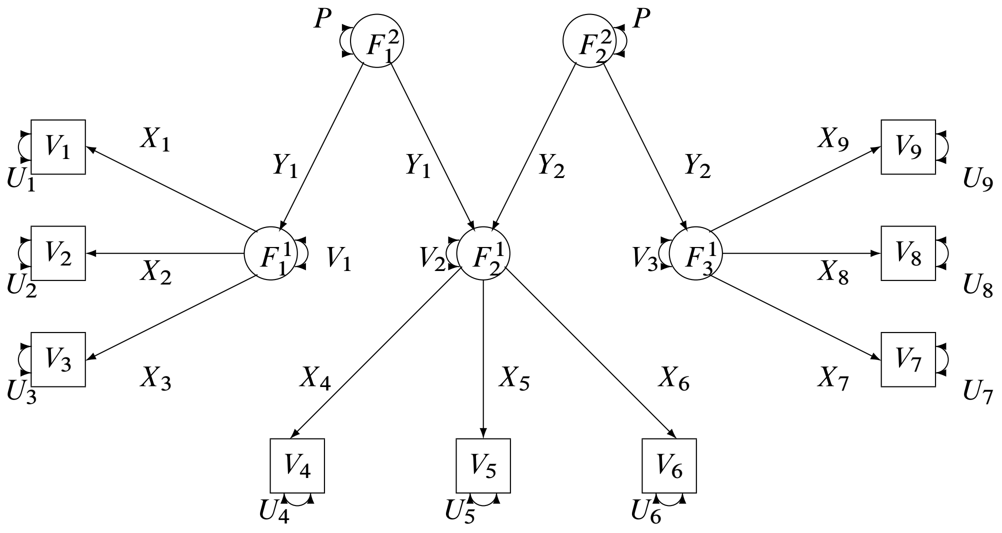
$$\mathbf{S} = \mathbf{F}_1\mathbf{F}_2\mathbf{P}_2\mathbf{F}_2'\mathbf{F}_1' + \mathbf{F}_1\mathbf{U}_2^2\mathbf{F}_1' + \mathbf{U}_1^2$$

You can fit this model by using the following RAM model statement:

```
ram
    1  1 10    x1,
    1  2 10    x2,
    1  3 10    x3,
    1  4 11    x4,
    1  5 11    x5,
    1  6 11    x6,
    1  7 12    x7,
    1  8 12    x8,
    1  9 12    x9,
    1 10 13    y1,
    1 11 13    y1,
    1 11 14    y2,
    1 12 14    y2,
    2  1  1    u:,
    2  2  2    u:,
    2  3  3    u:,
    2  4  4    u:,
    2  5  5    u:,
    2  6  6    u:,
    2  7  7    u:,
    2  8  8    u:,
    2  9  9    u:,
    2 10 10    v:,
    2 11 11    v:,
    2 12 12    v:,
    2 13 13    p ,
    2 14 14    p ;
run;
```

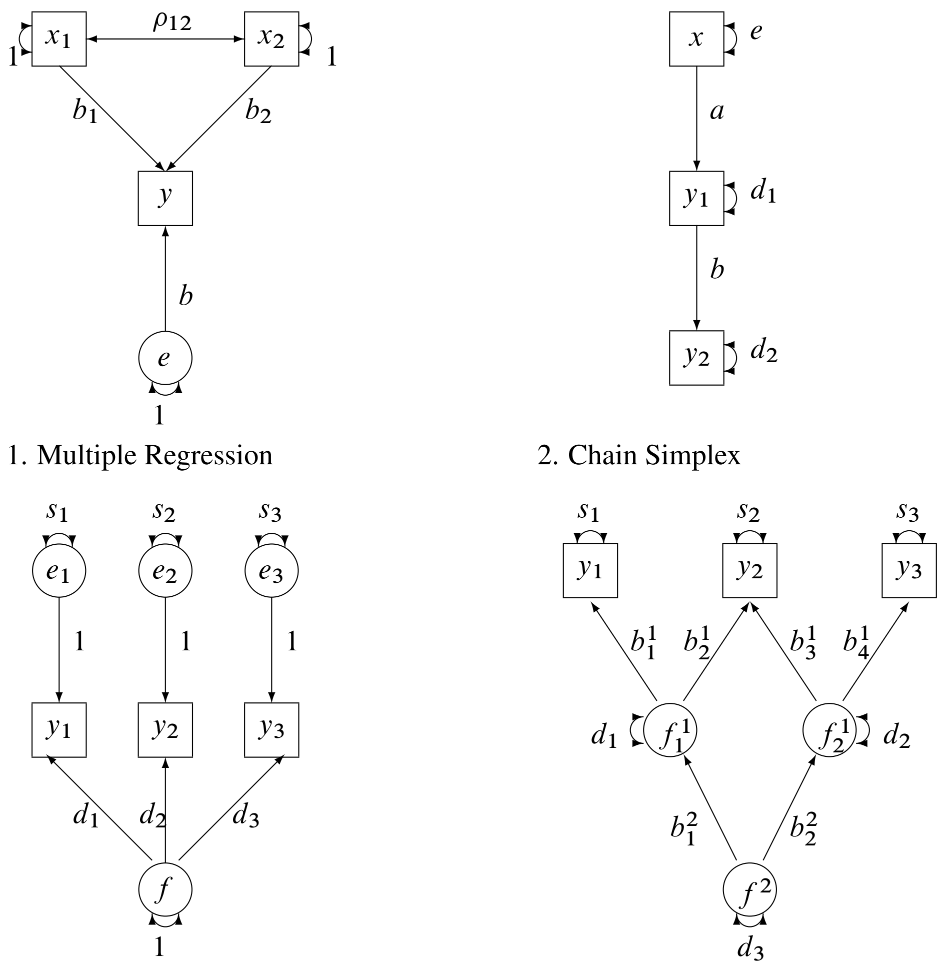
The confirmatory second-order factor analysis model corresponds to the path diagram displayed in Figure 25.3.

Figure 25.3 Path Diagram of Second-Order Factor Analysis Model



There is a very close relationship between the RAM model algebra and the specification of structural linear models by path diagrams. See Figure 25.4 for an example.

Figure 25.4 Examples of RAM Nomography



Refer to McArdle (1980) for the interpretation of the models displayed in Figure 25.4.

SAS Programming Statements

This section lists the programming statements used to express the linear and nonlinear constraints on the parameters and documents the differences between programming statements in PROC CALIS and program statements in the DATA step. The very different use of the ARRAY statement by PROC CALIS is also discussed. Most of the programming statements that can be used in the SAS DATA step also can be used in PROC CALIS. Refer to *SAS Language Reference: Dictionary* for a description of the SAS programming statements. You can specify the following SAS programming statements to compute parameter constraints with the CALIS procedure:

ABORT;
CALL *name* < **(** *expression* <, *expression* . . . > **)** >**;**
DELETE;
DO < *variable* **=** *expression*
  < **TO** *expression* > < **BY** *expression* >
  < , *expression* < **TO** *expression* > < **BY** *expression* > . . . >
  >
  < **WHILE** *expression* > < **UNTIL** *expression* >**;**
END;
GOTO *statement-label***;**
IF *expression***;**
IF *expression* **THEN** *program-statement***;**
    ELSE *program-statement***;**
variable **=** *expression***;**
variable **+** *expression***;**
LINK *statement-label***;**
PUT < *variable* > < **=** > < ... >**;**
RETURN;
SELECT < **(** *expression* **)** >**;**
STOP;
SUBSTR(*variable, index, length* **)=** *expression***;**
WHEN (*expression* **)** *program-statement***;**
  OTHERWISE *program-statement***;**

For the most part, the SAS programming statements work the same as they do in the SAS DATA step as documented in *SAS Language Reference: Concepts*. However, there are several differences that should be noted.

- The ABORT statement does not allow any arguments.
- The DO statement does not allow a character index variable. Thus,

  ```
  do I=1,2,3;
  ```

  is supported; however,

  ```
  do I='A','B','C';
  ```

  is not valid in PROC CALIS, although it is supported in the DATA step.
- The PUT statement, used mostly for program debugging in PROC CALIS, supports only some of the features of the DATA step PUT statement, and it has some new features that the DATA step PUT statement does not have:
  - The CALIS procedure PUT statement does not support line pointers, factored lists, iteration factors, overprinting, _INFILE_, the colon (:) format modifier, or $.
  - The CALIS procedure PUT statement does support expressions enclosed in parentheses. For example, the following statement displays the square root of x:

```
        put (sqrt(x));
```

- – The CALIS procedure PUT statement supports the print item _PDV_ to display a formatted listing of all variables in the program. For example, the following statement displays a much more readable listing of the variables than the _ALL_ print item:

```
        put _pdv_ ;
```

- The WHEN and OTHERWISE statements allow more than one target statement. That is, DO/END groups are not necessary for multiple WHEN statements. For example, the following syntax is valid:

```
select;
   when ( expression1 ) statement1;
                        statement2;
   when ( expression2 ) statement3;
                        statement4;
end;
```

You can specify one or more PARMS statements to define parameters used in the programming statements that are not defined in the model matrices (MATRIX, RAM, LINEQS, STD, or COV statement).

Parameters that are used only on the right-hand side of your programming statements are called independent, and parameters that are used at least once on the left-hand side of an equation in the program code are called dependent parameters. The dependent parameters are used only indirectly in the minimization process. They should be fully defined as functions of the independent parameters. The independent parameters are included in the set **X** of parameters used in the minimization. Be sure that all independent parameters used in your programming statements are somehow connected to elements of the model matrices. Otherwise the minimization function does not depend on those independent parameters, and the parameters vary without control (since the corresponding derivative is the constant 0). You also can specify the PARMS statement to set the initial values of all independent parameters used in the programming statements that are not defined as elements of model matrices.

ARRAY Statement

ARRAY *arrayname* < *dimensions* >< $ >< *variables and constants* > **;**

The ARRAY statement is similar to, but not the same as, the ARRAY statement in the DATA step. The ARRAY statement is used to associate a name with a list of variables and constants. The array name can then be used with subscripts in the program to refer to the items in the list.

The ARRAY statement supported by PROC CALIS does not support all the features of the DATA step ARRAY statement. With PROC CALIS, the ARRAY statement cannot be used to give initial values to array elements. Implicit indexing variables cannot be used; all array references must have explicit subscript expressions. Only exact array dimensions are allowed; lower-bound specifications are not supported. A maximum of six dimensions is allowed.

On the other hand, the ARRAY statement supported by PROC CALIS does allow both variables and constants to be used as array elements. Constant array elements cannot be changed. Both the dimension specification and the list of elements are optional, but at least one must be given. When the list of elements is not given or fewer elements than the size of the array are listed, array variables are created by suffixing element numbers to the array name to complete the element list.

STD Statement

STD *assignment < , assignment . . . >* **;**

where *assignment* represents
variables = pattern-definition

The STD statement tells which variances are parameters to estimate and which are fixed. The STD statement can be used only with the LINEQS statement. You can specify only one STD statement with each LINEQS model statement. The STD statement defines the diagonal elements of the central model matrix $\mathbf{\Phi}$. These elements correspond to the variances of the exogenous variables and to the error variances of the endogenous variables. Elements that are not defined are assumed to be zero.

Each *assignment* consists of a variable list (*variables*) on the left-hand side and a pattern list (*pattern-definition*) on the right-hand side of an equal sign. The *assignments* in the STD statement must be separated by commas. The *variables* list on the left-hand side of the equal sign should contain only names of variables that do not appear on the left-hand side of an equation in the LINEQS statement—that is, exogenous, error, and disturbance variables.

The *pattern-definition* on the right-hand side is similar to that used in the MATRIX statement. Each list element on the right-hand side defines the variance of the variable on the left-hand side in the same list position. A name on the right-hand side means that the corresponding variance is a parameter to estimate. A name on the right-hand side can be followed by a number inside parentheses that gives the initial value. A number on the right-hand side means that the corresponding variance of the variable on the left-hand side is fixed. If the right-hand-side list is longer than the left-hand-side variable list, the right-hand-side list is shortened to the length of the variable list. If the right-hand-side list is shorter than the variable list, the right-hand-side list is filled with repetitions of the last item in the list.

The right-hand side can also contain prefixes. A prefix is a short name followed by a colon. The CALIS procedure then generates a parameter name by appending an integer suffix to this prefix name. The prefix name should have no more than five or six characters so that the generated parameter name is not longer than eight characters. To avoid unintentional equality constraints, the prefix names should not coincide with explicitly defined parameter names. For example, if the prefix A is not used in any previous statement, the STD statement

```
std E1-E6=6 * A: (6 * 3.) ;
```

defines the six error variances as free parameters A1, . . . , A6, all with starting values of 3.

STRUCTEQ Statement

STRUCTEQ *variable* < *variable* . . . > **;**

The STRUCTEQ statement is used to list the dependent variables of the structural equations. This statement is ignored if you omit the PDETERM option. This statement is useful because the term *structural equation* is not defined in a unique way, and PROC CALIS has difficulty identifying the structural equations.

If LINEQS statements are used, the names of the left-hand-side (dependent) variables of those equations to be treated as structural equations should be listed in the STRUCTEQ statement.

If the RAM statement is used, variable names in the STRUCTEQ statements depend on the VARNAMES statement:

- If the VARNAMES statement is used, variable names must correspond to those in the VARNAMES statement.
- If the VARNAMES statement is not used, variable names must correspond to the names of manifest variables or latent (F) variables.

The STRUCTEQ statement also defines the names of variables used in the causal coefficient matrix of the structural equations, $\mathbf{B}$, for computing the *Stability Coefficient of Reciprocal Causation* (the largest eigenvalue of the $\mathbf{BB}'$ matrix). If the PROC CALIS option PDETERM is used without the STRUCTEQ statement, the structural equations are defined as described in the PDETERM option. See the PROC CALIS option PDETERM on page 863 for more details.

VAR Statement

VAR *variables* **;**

The VAR statement lists the numeric variables to be analyzed. If the VAR statement is omitted, all numeric variables not mentioned in other statements are analyzed. You can use the VAR statement to ensure that the manifest variables appear in correct order for use in the RAM statement. Only one VAR statement can be used with each PROC CALIS statement. If you do not use all manifest variables when you specify the model with a RAM or LINEQS statement, PROC CALIS does automatic variable selection. For more information, see the section "Automatic Variable Selection" on page 942.

VARNAMES Statement

VARNAMES | VNAMES *assignment* < , *assignment* . . . > **;**

where *assignment* represents
matrix-id variable-names or *matrix-name = matrix-name*

Use the VARNAMES statement in connection with the RAM, COSAN, or FACTOR model statement to allocate names to latent variables including error and disturbance terms. This statement is not needed if you are using the LINEQS statement.

In connection with the RAM model statement, the *matrix-id* must be specified by the integer number as it is used in the RAM list input (1 for matrix **A**, 2 for matrix **P**). Because the first variables of matrix **A** correspond to the manifest variables in the input data set, you can specify names only for the latent variables following the manifest variables in the rows of **A**. For example, in the RAM notation of the alienation example, you can specify the latent variables by names F1, F2, F3 and the error variables by names E1, . . . , E6, D1, D2, D3 with the following statement:

```
vnames 1  F1-F3,
       2  E1-E6 D1-D3;
```

If the RAM model statement is not accompanied by a VNAMES statement, default variable names are assigned using the prefixes “F”, “E”, and “D” with numerical suffixes: latent variables are F1, F2, . . . , and error variables are E1, E2, ...

The *matrix-id* must be specified by its name when used with the COSAN or FACTOR statement. The *variable-names* following the matrix name correspond to the columns of this matrix. The variable names corresponding to the rows of this matrix are set automatically by the following rules:

- If it is the first matrix of any term, the row variable names are the names of the manifest variables.
- If it is the central symmetric matrix of any term, the row variable names are the same as the column variable names of the same matrix.
- For any other matrices, the row variable names are the same as the column variable names of the preceding matrix.

You also can use the second kind of name assignment in connection with a COSAN statement. Two matrix names separated by an equal sign allocate the column names of one matrix to the column names of the other matrix. This assignment assumes that the column names of at least one of the two matrices are already allocated. For example, in the COSAN notation of the alienation example, you can specify the variable names by using the following statements to allocate names to the columns of **J**, **A**, and **P**:

```
vnames  J  V1-V6 F1-F3 ,
        A =J ,
        P  E1-E6 D1-D3 ;
```

WEIGHT Statement

WEIGHT *variable* **;**

To compute weighted covariances or correlations, specify the name of the weighting variable in a WEIGHT statement. This is often done when the error variance associated with each observation is different and the values of the weight variable are proportional to the reciprocals of the variances. You can use only one WEIGHT statement with each PROC CALIS statement. The WEIGHT and FREQ statements have a similar effect, except the WEIGHT statement does not alter the number of observations unless VARDEF=WGT or VARDEF=WDF. An observation is used in the analysis only if the WEIGHT variable is greater than 0 and is not missing.

Details: CALIS Procedure

Input Data Sets

You can use four different kinds of input data sets in the CALIS procedure, and you can use them simultaneously. The DATA= data set contains the data to be analyzed, and it can be an ordinary SAS data set containing raw data or a special TYPE=COV, TYPE=UCOV, TYPE=CORR, TYPE=UCORR, TYPE=SYMATRIX, TYPE=SSCP, or TYPE=FACTOR data set containing previously computed statistics. The INEST= data set specifies an input data set that contains initial estimates for the parameters used in the optimization process, and it can also contain boundary and general linear constraints on the parameters. If the model does not change too much, you can use an OUTEST= data set from a previous PROC CALIS analysis; the initial estimates are taken from the values of the PARMS observation. The INRAM= data set names a third input data set that contains all information needed to specify the analysis model in RAM list form (except for user-written program statements). Often the INRAM= data set can be the OUTRAM= data set from a previous PROC CALIS analysis. See the section "OUTRAM= SAS-data-set" on page 917 for the structure of both OUTRAM= and INRAM= data sets. Using the INWGT= data set enables you to read in the weight matrix **W** that can be used in generalized least squares, weighted least squares, or diagonally weighted least squares estimation.

DATA= SAS-data-set

A TYPE=COV, TYPE=UCOV, TYPE=CORR, or TYPE=UCORR data set can be created by the CORR procedure or various other procedures. It contains means, standard deviations, the sample size, the covariance or correlation matrix, and possibly other statistics depending on which procedure is used.

If your data set has many observations and you plan to run PROC CALIS several times, you can save computer time by first creating a TYPE=COV, TYPE=UCOV, TYPE=CORR, or TYPE=UCORR data set and using it as input to PROC CALIS. For example, assuming that PROC CALIS is first run with an OUTRAM=MOD option, you can run the following statements:

```
* create TYPE=COV data set;
proc corr cov nocorr data=raw outp=cov(type=cov);
run;
* analysis using correlations;
proc calis data=cov inram=mod;
run;
* analysis using covariances;
proc calis cov data=cov inram=mod;
run;
```

Most procedures automatically set the TYPE= option of an output data set appropriately. However, the CORR procedure sets TYPE=CORR unless an explicit TYPE= option is used. Thus, (TYPE=COV) is needed in the preceding PROC CORR request, since the output data set is a covariance matrix. If you use a DATA step with a SET statement to modify this data set, you must declare the TYPE=COV, TYPE=UCOV, TYPE=CORR, or TYPE=UCORR attribute in the new data set.

You can use a VAR statement with PROC CALIS when reading a TYPE=COV, TYPE=UCOV, TYPE=CORR, TYPE=UCORR, or TYPE=SSCP data set to select a subset of the variables or change the order of the variables.

CAUTION: Problems can arise from using the CORR procedure when there are missing data. By default, PROC CORR computes each covariance or correlation from all observations that have values present for the pair of variables involved ("pairwise deletion"). The resulting covariance or correlation matrix can have negative eigenvalues. A correlation or covariance matrix with negative eigenvalues is recognized as a singular matrix in PROC CALIS, and you cannot compute (default) generalized least squares or maximum likelihood estimates. You can specify the RIDGE option to ridge the diagonal of such a matrix to obtain a positive-definite data matrix. If the NOMISS option is used with the CORR procedure, observations with any missing values are completely omitted from the calculations ("listwise deletion"), and there is no possibility of negative eigenvalues (but still a chance for a singular matrix).

PROC CALIS can also create a TYPE=COV, TYPE=UCOV, TYPE=CORR, or TYPE=UCORR data set that includes all the information needed for repeated analyses. If the data set DATA=RAW does not contain missing values, the following statements should give the same PROC CALIS results as the previous example:

```
* using correlations;
proc calis data=raw outstat=cov inram=mod;
run;
* using covariances;
proc calis cov data=cov inram=mod;
run;
```

You can create a TYPE=COV, TYPE=UCOV, TYPE=CORR, TYPE=UCORR, or TYPE=SSCP data set in a DATA step. Be sure to specify the TYPE= option in parentheses after the data set name in the DATA statement, and include the _TYPE_ and _NAME_ variables. If you want to analyze the covariance matrix but your DATA= data set is a TYPE=CORR or TYPE=UCORR data set, you should include an observation with _TYPE_=STD giving the standard deviation of each variable. If you specify the COV option, PROC CALIS analyzes the recomputed covariance matrix:

```
data correl(type=corr);
   input _type_ $ _name_ $ X1-X3;
   datalines;
std    .    4.   2.   8.
corr   X1   1.0   .    .
corr   X2    .7 1.0   .
corr   X3    .5   .4 1.0
;
proc calis cov inram=model;
run;
```

If you want to analyze the UCOV or UCORR matrix but your DATA= data set is a TYPE=COV or TYPE=CORR data set, you should include observations with _TYPE_=STD and _TYPE_=MEAN giving the standard deviation and mean of each variable.

INEST= SAS-data-set

You can use the INEST= (or INVAR= or ESTDATA=) input data set to specify the initial values of the parameters used in the optimization and to specify boundary constraints and the more general linear constraints that can be imposed on these parameters.

The variables of the INEST= data set must correspond to the following variables:

- a character variable _TYPE_ that indicates the type of the observation
- *n* numeric variables with the parameter names used in the specified PROC CALIS model
- the BY variables that are used in a DATA= input data set
- a numeric variable _RHS_ (right-hand side) (needed only if linear constraints are used)
- additional variables with names corresponding to constants used in the programming statements

The content of the _TYPE_ variable defines the meaning of the observation of the INEST= data set. PROC CALIS recognizes observations with the following _TYPE_ specifications.

PARMS	specifies initial values for parameters that are defined in the model statements of PROC CALIS. The _RHS_ variable is not used. Additional variables can contain the values of constants that are referred to in programming statements. At the beginning of each run of PROC CALIS, the values of the constants are read from the PARMS observation initializing the constants in the program statements.
UPPERBD | UB	specifies upper bounds with nonmissing values. The use of a missing value indicates that no upper bound is specified for the parameter. The _RHS_ variable is not used.

LOWERBD | LB	specifies lower bounds with nonmissing values. The use of a missing value indicates that no lower bound is specified for the parameter. The _RHS_ variable is not used.
LE | <= | <	specifies the linear constraint $\sum_j a_{ij} x_j \leq b_i$. The n parameter values contain the coefficients a_{ij}, and the _RHS_ variable contains the right-hand-side b_i. The use of a missing value indicates a zero coefficient a_{ij}.
GE | >= | >	specifies the linear constraint $\sum_j a_{ij} x_j \geq b_i$. The n parameter values contain the coefficients a_{ij}, and the _RHS_ variable contains the right-hand-side b_i. The use of a missing value indicates a zero coefficient a_{ij}.
EQ | =	specifies the linear constraint $\sum_j a_{ij} x_j = b_i$. The n parameter values contain the coefficients a_{ij}, and the _RHS_ variable contains the right-hand-side b_i. The use of a missing value indicates a zero coefficient a_{ij}.

The constraints specified in the INEST=, INVAR=, or ESTDATA= data set are added to the constraints specified in BOUNDS and LINCON statements.

You can use an OUTEST= data set from a PROC CALIS run as an INEST= data set in a new run. However, be aware that the OUTEST= data set also contains the boundary and general linear constraints specified in the previous run of PROC CALIS. When you are using this OUTEST= data set without changes as an INEST= data set, PROC CALIS adds the constraints from the data set to the constraints specified by a BOUNDS and LINCON statement. Although PROC CALIS automatically eliminates multiple identical constraints, you should avoid specifying the same constraint a second time.

INRAM= SAS-data-set

This data set is usually created in a previous run of PROC CALIS. It is useful if you want to reanalyze a problem in a different way, such as by using a different estimation method. You can alter an existing OUTRAM= data set, either in the DATA step or using the FSEDIT procedure, to create the INRAM= data set describing a modified model. For more details on the INRAM= data set, see the section "OUTRAM= SAS-data-set" on page 917.

In the case of a RAM or LINEQS analysis of linear structural equations, the OUTRAM= data set always contains the variable names of the model specified. These variable names and the model specified in the INRAM= data set are the basis of the automatic variable selection algorithm performed after reading the INRAM= data set.

INWGT= SAS-data-set

This data set enables you to specify a weight matrix other than the default matrix for the generalized, weighted, and diagonally weighted least squares estimation methods. The specification of any INWGT= data set for unweighted least squares or maximum likelihood estimation is ignored. For generalized and diagonally weighted least squares estimation, the INWGT= data set must contain a _TYPE_ and a _NAME_ variable as well as the manifest variables used in the analysis. The value of the _NAME_ variable indicates the row index i of the weight w_{ij}. For weighted least squares, the INWGT= data set must contain _TYPE_, _NAME_, _NAM2_, and _NAM3_ variables as well as the manifest variables used in the analysis. The values of the _NAME_, _NAM2_, and _NAM3_ variables indicate the three indices i, j, k of the weight $w_{ij,kl}$. You can store information other than the weight matrix in the INWGT= data set, but only observations with _TYPE_=WEIGHT are used to specify the weight matrix $\mathbf{W}$. This property enables you to store more than one weight matrix in the INWGT= data set. You can then run PROC CALIS with each of the weight matrices by changing only the _TYPE_ observation in the INWGT= data set with an intermediate DATA step.

For more details on the INWGT= data set, see the section "OUTWGT= SAS-data-set" on page 922.

Output Data Sets

OUTEST= SAS-data-set

The OUTEST= (or OUTVAR=) data set is of TYPE=EST and contains the final parameter estimates, the gradient, the Hessian, and boundary and linear constraints. For METHOD=ML, METHOD=GLS, and METHOD=WLS, the OUTEST= data set also contains the approximate standard errors, the information matrix (crossproduct Jacobian), and the approximate covariance matrix of the parameter estimates ((generalized) inverse of the information matrix). If there are linear or nonlinear equality or active inequality constraints at the solution, the OUTEST= data set also contains Lagrange multipliers, the projected Hessian matrix, and the Hessian matrix of the Lagrange function.

The OUTEST= data set can be used to save the results of an optimization by PROC CALIS for another analysis with either PROC CALIS or another SAS procedure. Saving results to an OUTEST= data set is advised for expensive applications that cannot be repeated without considerable effort.

The OUTEST= data set contains the BY variables, two character variables _TYPE_ and _NAME_, t numeric variables corresponding to the parameters used in the model, a numeric variable _RHS_ (right-hand side) that is used for the right-hand-side value b_i of a linear constraint or for the value $f = f(x)$ of the objective function at the final point x^* of the parameter space, and a numeric variable _ITER_ that is set to zero for initial values, set to the iteration number for the OUTITER output, and set to missing for the result output.

The _TYPE_ observations in Table 25.5 are available in the OUTEST= data set, depending on the request.

Table 25.5 _TYPE_ Observations in the OUTEST= Data Set

TYPE	**Description**
ACTBC	If there are active boundary constraints at the solution x^*, three observations indicate which of the parameters are actively constrained, as follows.

NAME	**Description**
GE	indicates the active lower bounds
LE	indicates the active upper bounds
EQ	indicates the active masks

Table 25.5 *continued*

TYPE	Description
COV	contains the approximate covariance matrix of the parameter estimates; used in computing the approximate standard errors.
COVRANK	contains the rank of the covariance matrix of the parameter estimates.
CRPJ_LF	contains the Hessian matrix of the Lagrange function (based on CRPJAC).
CRPJAC	contains the approximate Hessian matrix used in the optimization process. This is the inverse of the information matrix.
EQ	If linear constraints are used, this observation contains the ith linear constraint $\sum_j a_{ij} x_j = b_i$. The parameter variables contain the coefficients a_{ij}, $j = 1, \ldots, n$, the _RHS_ variable contains b_i, and _NAME_=ACTLC or _NAME_=LDACTLC.
GE	If linear constraints are used, this observation contains the ith linear constraint $\sum_j a_{ij} x_j \geq b_i$. The parameter variables contain the coefficients a_{ij}, $j = 1, \ldots, n$, and the _RHS_ variable contains b_i. If the constraint i is active at the solution x^*, then _NAME_=ACTLC or _NAME_=LDACTLC.
GRAD	contains the gradient of the estimates.
GRAD_LF	contains the gradient of the Lagrange function. The _RHS_ variable contains the value of the Lagrange function.
HESSIAN	contains the Hessian matrix.
HESS_LF	contains the Hessian matrix of the Lagrange function (based on HESSIAN).
INFORMAT	contains the information matrix of the parameter estimates (only for METHOD=ML, METHOD=GLS, or METHOD=WLS).
INITIAL	contains the starting values of the parameter estimates.
JACNLC	contains the Jacobian of the nonlinear constraints evaluated at the final estimates.
JACOBIAN	contains the Jacobian matrix (only if the OUTJAC option is used).
LAGM BC	contains Lagrange multipliers for masks and active boundary constraints.

NAME	Description
GE	indicates the active lower bounds
LE	indicates the active upper bounds
EQ	indicates the active masks

Table 25.5 *continued*

TYPE	Description
LAGM LC	contains Lagrange multipliers for linear equality and active inequality constraints in pairs of observations containing the constraint number and the value of the Lagrange multiplier.

NAME	Description
LEC_NUM	number of the linear equality constraint
LEC_VAL	corresponding Lagrange multiplier value
LIC_NUM	number of the linear inequality constraint
LIC_VAL	corresponding Lagrange multiplier value

TYPE	Description
LAGM NLC	contains Lagrange multipliers for nonlinear equality and active inequality constraints in pairs of observations containing the constraint number and the value of the Lagrange multiplier.

NAME	Description
NLEC_NUM	number of the nonlinear equality constraint
NLEC_VAL	corresponding Lagrange multiplier value
NLIC_NUM	number of the linear inequality constraint
NLIC_VAL	corresponding Lagrange multiplier value

TYPE	Description
LE	If linear constraints are used, this observation contains the ith linear constraint $\sum_j a_{ij} x_j \leq b_i$. The parameter variables contain the coefficients a_{ij}, $j = 1, \ldots, n$, and the _RHS_ variable contains b_i. If the constraint i is active at the solution x^*, then _NAME_=ACTLC or _NAME_=LDACTLC.
LOWERBD | LB	If boundary constraints are used, this observation contains the lower bounds. Those parameters not subjected to lower bounds contain missing values. The _RHS_ variable contains a missing value, and the _NAME_ variable is blank.
NACTBC	All parameter variables contain the number n_{abc} of active boundary constraints at the solution x^*. The _RHS_ variable contains a missing value, and the _NAME_ variable is blank.
NACTLC	All parameter variables contain the number n_{alc} of active linear constraints at the solution x^* that are recognized as linearly independent. The _RHS_ variable contains a missing value, and the _NAME_ variable is blank.

Table 25.5 *continued*

TYPE	Description
NLC_EQ NLC_GE NLC_LE	contains values and residuals of nonlinear constraints. The _NAME_ variable is described as follows. **_NAME_** — **Description** NLC — inactive nonlinear constraint NLCACT — linear independent active nonlinear constr. NLCACTLD — linear dependent active nonlinear constr.
NLDACTBC	contains the number of active boundary constraints at the solution x^* that are recognized as linearly dependent. The _RHS_ variable contains a missing value, and the _NAME_ variable is blank.
NLDACTLC	contains the number of active linear constraints at the solution x^* that are recognized as linearly dependent. The _RHS_ variable contains a missing value, and the _NAME_ variable is blank.
NOBS	contains the number of observations.
PARMS	contains the final parameter estimates. The _RHS_ variable contains the value of the objective function.
PCRPJ_LF	contains the projected Hessian matrix of the Lagrange function (based on CRPJAC).
PHESS_LF	contains the projected Hessian matrix of the Lagrange function (based on HESSIAN).
PROJCRPJ	contains the projected Hessian matrix (based on CRPJAC).
PROJGRAD	If linear constraints are used in the estimation, this observation contains the $n - n_{act}$ values of the projected gradient $g_Z = Z'g$ in the variables corresponding to the first $n - n_{act}$ parameters. The _RHS_ variable contains a missing value, and the _NAME_ variable is blank.
PROJHESS	contains the projected Hessian matrix (based on HESSIAN).
SIGSQ	contains the scalar factor of the covariance matrix of the parameter estimates.
STDERR	contains approximate standard errors (only for METHOD=ML, METHOD=GLS, or METHOD=WLS).
TERMINAT	The _NAME_ variable contains the name of the termination criterion.

Table 25.5 *continued*

TYPE	Description
UPPERBD | UB	If boundary constraints are used, this observation contains the upper bounds. Those parameters not subjected to upper bounds contain missing values. The _RHS_ variable contains a missing value, and the _NAME_ variable is blank.

If the technique specified by the TECH= option cannot be performed (for example, no feasible initial values can be computed, or the function value or derivatives cannot be evaluated at the starting point), the OUTEST= data set might contain only some of the observations (usually only the PARMS and GRAD observations).

OUTRAM= SAS-data-set

The OUTRAM= data set is of TYPE=RAM and contains the model specification and the computed parameter estimates. This data set is intended to be reused as an INRAM= data set to specify good initial values in a subsequent analysis by PROC CALIS.

The OUTRAM= data set contains the following variables:

- the BY variables, if any
- the character variable _TYPE_, which takes the values MODEL, ESTIM, VARNAME, METHOD, and STAT
- six additional variables whose meaning depends on the _TYPE_ of the observation

Each observation with _TYPE_=MODEL defines one matrix in the generalized COSAN model. The additional variables are as follows.

Table 25.6 Additional Variables When _TYPE_=MODEL

Variable	Contents
NAME	name of the matrix (character)
MATNR	number for the term and matrix in the model (numeric)
ROW	matrix row number (numeric)
COL	matrix column number (numeric)
ESTIM	first matrix type (numeric)
STDERR	second matrix type (numeric)

If the generalized COSAN model has only one matrix term, the _MATNR_ variable contains only the number of the matrix in the term. If there is more than one term, then it is the term number multiplied by 10,000 plus the matrix number (assuming that there are no more than 9,999 matrices specified in the COSAN model statement).

Each observation with _TYPE_=ESTIM defines one element of a matrix in the generalized COSAN model. The variables are used as follows.

Table 25.7 Additional Variables When _TYPE_=ESTIM

Variable	Contents
NAME	name of the parameter (character)
MATNR	term and matrix location of parameter (numeric)
ROW	row location of parameter (numeric)
COL	column location of parameter (numeric)
ESTIM	parameter estimate or constant value (numeric)
STDERR	standard error of estimate (numeric)

For constants rather than estimates, the _STDERR_ variable is 0. The _STDERR_ variable is missing for ULS and DWLS estimates if NOSTDERR is specified or if the approximate standard errors are not computed.

Each observation with _TYPE_=VARNAME defines a column variable name of a matrix in the generalized COSAN model.

The observations with _TYPE_=METHOD and _TYPE_=STAT are not used to build the model. The _TYPE_=METHOD observation contains the name of the estimation method used to compute the parameter estimates in the _NAME_ variable. If METHOD=NONE is not specified, the _ESTIM_ variable of the _TYPE_=STAT observations contains the information summarized in Table 25.8 (described in the section "Assessment of Fit" on page 928).

Table 25.8 _ESTIM_ Contents for _TYPE_=STAT

NAME	_ESTIM_
N	sample size
NPARM	number of parameters used in the model
DF	degrees of freedom
N_ACT	number of active boundary constraints for ML, GLS, and WLS estimation
FIT	fit function
GFI	goodness of fit index (GFI)
AGFI	adjusted GFI for degrees of freedom
RMR	root mean square residual
SRMR	standardized root mean square residual
PGFI	parsimonious GFI of Mulaik et al. (1989)
CHISQUAR	overall χ^2
P_CHISQ	probability $> \chi^2$
CHISQNUL	null (baseline) model χ^2
RMSEAEST	Steiger and Lind's (1980) RMSEA index estimate
RMSEALOB	lower range of RMSEA confidence interval
RMSEAUPB	upper range of RMSEA confidence interval
P_CLOSFT	Browne and Cudeck's (1993) probability of close fit
ECVI_EST	Browne and Cudeck's (1993) ECV index estimate

Table 25.8 *continued*

NAME	_ESTIM_
ECVI_LOB	lower range of ECVI confidence interval
ECVI_UPB	upper range of ECVI confidence interval
COMPFITI	Bentler's (1989) comparative fit index
ADJCHISQ	adjusted χ^2 for elliptic distribution
P_ACHISQ	probability corresponding adjusted χ^2
RLSCHISQ	reweighted least squares χ^2 (only ML estimation)
AIC	Akaike's information criterion
CAIC	Bozdogan's consistent information criterion
SBC	Schwarz's Bayesian criterion
CENTRALI	McDonald's centrality criterion
PARSIMON	Parsimonious index of James, Mulaik, and Brett
ZTESTWH	z test of Wilson and Hilferty
BB_NONOR	Bentler-Bonett (1980) nonnormed index ρ
BB_NORMD	Bentler-Bonett (1980) normed index Δ
BOL_RHO1	Bollen's (1986) normed index ρ_1
BOL_DEL2	Bollen's (1989a) nonnormed index Δ_2
CNHOELT	Hoelter's critical N index

You can edit the OUTRAM= data set to use its contents for initial estimates in a subsequent analysis by PROC CALIS, perhaps with a slightly changed model. But you should be especially careful for _TYPE_=MODEL when changing matrix types. The codes for the two matrix types are listed in Table 25.9.

Table 25.9 Matrix Type Codes

Code	First Matrix Type	Description
1:	IDE	identity matrix
2:	ZID	zero matrix concatenated by an identity matrix
3:	DIA	diagonal matrix
4:	ZDI	zero matrix concatenated by a diagonal matrix
5:	LOW	lower triangular matrix
6:	UPP	upper triangular matrix
7:		temporarily not used
8:	SYM	symmetric matrix
9:	GEN	general-type matrix
10:	BET	identity minus general-type matrix
11:	PER	selection matrix
12:		first matrix ($\mathbf{J}$) in LINEQS model statement
13:		second matrix ($\boldsymbol{\beta}$) in LINEQS model statement
14:		third matrix ($\boldsymbol{\gamma}$) in LINEQS model statement
Code	**Second Matrix Type**	**Description**
0:		noninverse model matrix
1:	INV	inverse model matrix
2:	IMI	"identity minus inverse" model matrix

OUTSTAT= SAS-data-set

The OUTSTAT= data set is similar to the TYPE=COV, TYPE=UCOV, TYPE=CORR, or TYPE=UCORR data set produced by the CORR procedure. The OUTSTAT= data set contains the following variables:

- the BY variables, if any
- two character variables, _TYPE_ and _NAME_
- the variables analyzed—that is, those in the VAR statement, or if there is no VAR statement, all numeric variables not listed in any other statement but used in the analysis. (**CAUTION:** Using the LINEQS or RAM model statements selects variables automatically.)

The OUTSTAT= data set contains the following information (when available):

- the mean and standard deviation
- the skewness and kurtosis (if the DATA= data set is a raw data set and the KURTOSIS option is specified)
- the number of observations
- if the WEIGHT statement is used, sum of the weights
- the correlation or covariance matrix to be analyzed
- the predicted correlation or covariance matrix
- the standardized or normalized residual correlation or covariance matrix
- if the model contains latent variables, the predicted covariances between latent and manifest variables, and the latent variable (or factor) score regression coefficients (see the PLATCOV display option on page 864)

In addition, if the FACTOR model statement is used, the OUTSTAT= data set contains the following:

- the unrotated factor loadings, the unique variances, and the matrix of factor correlations
- the rotated factor loadings and the transformation matrix of the rotation
- the matrix of standardized factor loadings

Each observation in the OUTSTAT= data set contains some type of statistic as indicated by the _TYPE_ variable. The values of the _TYPE_ variable are given in Table 25.10.

Table 25.10 _TYPE_ Observations in the OUTSTAT= Data Set

TYPE	Contents
MEAN	means
STD	standard deviations
USTD	uncorrected standard deviations
SKEWNESS	univariate skewness
KURTOSIS	univariate kurtosis
N	sample size
SUMWGT	sum of weights (if WEIGHT statement is used)
COV	covariances analyzed
CORR	correlations analyzed
UCOV	uncorrected covariances analyzed
UCORR	uncorrected correlations analyzed
ULSPRED	ULS predicted model values
GLSPRED	GLS predicted model values
MAXPRED	ML predicted model values
WLSPRED	WLS predicted model values
DWLSPRED	DWLS predicted model values
ULSNRES	ULS normalized residuals
GLSNRES	GLS normalized residuals
MAXNRES	ML normalized residuals
WLSNRES	WLS normalized residuals
DWLSNRES	DWLS normalized residuals
ULSSRES	ULS variance standardized residuals
GLSSRES	GLS variance standardized residuals
MAXSRES	ML variance standardized residuals
WLSSRES	WLS variance standardized residuals
DWLSSRES	DWLS variance standardized residuals
ULSASRES	ULS asymptotically standardized residuals
GLSASRES	GLS asymptotically standardized residuals
MAXASRES	ML asymptotically standardized residuals
WLSASRES	WLS asymptotically standardized residuals
DWLSASRS	DWLS asymptotically standardized residuals
UNROTATE	unrotated factor loadings
FCORR	matrix of factor correlations
UNIQUE_V	unique variances
TRANSFOR	transformation matrix of rotation
LOADINGS	rotated factor loadings
STD_LOAD	standardized factor loadings
LSSCORE	latent variable (or factor) score regression coefficients for ULS method
SCORE	latent variable (or factor) score regression coefficients other than ULS method

The _NAME_ variable contains the name of the manifest variable corresponding to each row for the covariance, correlation, predicted, and residual matrices and contains the name of the latent variable in case of factor regression scores. For other observations, _NAME_ is blank.

The unique variances and rotated loadings can be used as starting values in more difficult and constrained analyses.

If the model contains latent variables, the OUTSTAT= data set also contains the latent variable score regression coefficients and the predicted covariances between latent and manifest variables. You can use the latent variable score regression coefficients with PROC SCORE to compute latent variable or factor scores. For details, see the section "Latent Variable Scores" on page 937.

If the analyzed matrix is a (corrected or uncorrected) covariance rather than a correlation matrix, the _TYPE_=STD or _TYPE_=USTD observation is not included in the OUTSTAT= data set. In this case, the standard deviations can be obtained from the diagonal elements of the covariance matrix. Dropping the _TYPE_=STD or _TYPE_=USTD observation prevents PROC SCORE from standardizing the observations before computing the factor scores.

OUTWGT= SAS-data-set

You can create an OUTWGT= data set that is of TYPE=WEIGHT and contains the weight matrix used in generalized, weighted, or diagonally weighted least squares estimation. The *inverse* of the weight matrix is used in the corresponding fit function. The OUTWGT= data set contains the weight matrix to which the WRIDGE= and the WPENALTY= options are applied. For unweighted least squares or maximum likelihood estimation, no OUTWGT= data set can be written. The last weight matrix used in maximum likelihood estimation is the predicted model matrix (observations with _TYPE_=MAXPRED) that is included in the OUTSTAT= data set.

For generalized and diagonally weighted least squares estimation, the weight matrices **W** of the OUTWGT= data set contain all elements w_{ij}, where the indices i and j correspond to all manifest variables used in the analysis. Let $varnam_i$ be the name of the ith variable in the analysis. In this case, the OUTWGT= data set contains n observations with variables as displayed in the following table.

Table 25.11 Contents of OUTWGT= Data Set for GLS and DWLS Estimation

Variable	Contents
TYPE	WEIGHT (character)
NAME	name of variable $varnam_i$ (character)
$varnam_1$	weight w_{i1} for variable $varnam_1$ (numeric)
$\vdots$	$\vdots$
$varnam_n$	weight w_{in} for variable $varnam_n$ (numeric)

For weighted least squares estimation, the weight matrix **W** of the OUTWGT= data set contains only the nonredundant elements $w_{ij,kl}$. In this case, the OUTWGT= data set contains $n(n+1)(2n+1)/6$ observations with variables as in the following table.

Table 25.12 Contents of OUTWGT= Data Set for WLS Estimation

Variable	Contents
TYPE	WEIGHT (character)
NAME	name of variable $varnam_i$ (character)
NAM2	name of variable $varnam_j$ (character)
NAM3	name of variable $varnam_k$ (character)
$varnam_1$	weight $w_{ij,k1}$ for variable $varnam_1$ (numeric)
$\vdots$	$\vdots$
$varnam_n$	weight $w_{ij,kn}$ for variable $varnam_n$ (numeric)

Symmetric redundant elements are set to missing values.

Missing Values

If the DATA= data set contains raw data (rather than a covariance or correlation matrix), observations with missing values for any variables in the analysis are omitted from the computations. If a covariance or correlation matrix is read, missing values are allowed as long as every pair of variables has at least one nonmissing value.

Estimation Criteria

The following five estimation methods are available in PROC CALIS:

- unweighted least squares (ULS)
- generalized least squares (GLS)
- normal-theory maximum likelihood (ML)
- weighted least squares (WLS, ADF)
- diagonally weighted least squares (DWLS)

An INWGT= data set can be used to specify other than the default weight matrices **W** for GLS, WLS, and DWLS estimation.

PROC CALIS do not exhaust all estimation methods in the field. As mentioned in the section "Overview: CALIS Procedure" on page 828, partial least squares (PLS) is not implemented. The PLS method is developed under less restrictive statistical assumptions. It circumvents some computational and theoretical problems encountered by the existing estimation methods in PROC CALIS. However, PLS estimates are less efficient in general. When the statistical assumptions of PROC

CALIS are tenable (for example, large sample size, correct distributional assumptions, etc.), ML, GLS, or WLS methods yield better estimates than the PLS method. Note that there is a SAS/STAT procedure called PROC PLS, which employs the partial least squares technique but for a class of models different from those of PROC CALIS. For example, in a PROC CALIS model each latent variable is typically associated with only a subset of manifest variables (predictor or outcome variables). However, in PROC PLS latent variables are not prescribed with subsets of manifest variables. Rather, they are extracted from linear combinations of all manifest predictor variables. Therefore, for general path analysis with latent variables you should consider using PROC CALIS.

In each estimation method, the parameter vector is estimated iteratively by a nonlinear optimization algorithm that optimizes a goodness of fit function F. When n denotes the number of manifest variables, $\mathbf{S}$ denotes the given sample covariance or correlation matrix for a sample with size N, and $\mathbf{C}$ denotes the predicted moment matrix, then the fit function for unweighted least squares estimation is

$$F_{ULS} = 0.5Tr[(\mathbf{S} - \mathbf{C})^2]$$

For normal-theory generalized least squares estimation, the function is

$$F_{GLS} = 0.5Tr[(\mathbf{S}^{-1}(\mathbf{S} - \mathbf{C}))^2]$$

For normal-theory maximum likelihood estimation, the function is

$$F_{ML} = Tr(\mathbf{S}\mathbf{C}^{-1}) - n + ln(det(\mathbf{C})) - ln(det(\mathbf{S}))$$

The first three functions can be expressed by the generalized weighted least squares criterion (Browne 1982):

$$F_{GWLS} = 0.5Tr[(\mathbf{W}^{-1}(\mathbf{S} - \mathbf{C}))^2]$$

For unweighted least squares, the weight matrix $\mathbf{W}$ is chosen as the identity matrix $\mathbf{I}$; for generalized least squares, the default weight matrix $\mathbf{W}$ is the sample covariance matrix $\mathbf{S}$; and for normal-theory maximum likelihood, $\mathbf{W}$ is the iteratively updated predicted moment matrix $\mathbf{C}$. The values of the normal-theory maximum likelihood function F_{ML} and the generally weighted least squares criterion F_{GWLS} with $\mathbf{W} = \mathbf{C}$ are asymptotically equivalent.

The goodness of fit function that is minimized in weighted least squares estimation is

$$F_{WLS} = Vec(s_{ij} - c_{ij})'\mathbf{W}^{-1}Vec(s_{ij} - c_{ij})$$

where $Vec(s_{ij} - c_{ij})$ denotes the vector of the $n(n + 1)/2$ elements of the lower triangle of the symmetric matrix $\mathbf{S} - \mathbf{C}$, and $\mathbf{W} = (w_{ij,kl})$ is a positive-definite symmetric matrix with $n(n + 1)/2$ rows and columns.

If the moment matrix **S** is considered as a covariance rather than a correlation matrix, the default setting of $\mathbf{W} = (w_{ij,kl})$ is the consistent but biased estimators of the asymptotic covariances $\sigma_{ij,kl}$ of the sample covariance s_{ij} with the sample covariance s_{kl}, as defined in the following:

$$w_{ij,kl} = s_{ij,kl} - s_{ij} s_{kl}$$

where

$$s_{ij,kl} = \frac{1}{N} \sum_{r=1}^{N} (z_{ri} - \bar{z}_i)(z_{rj} - \bar{z}_j)(z_{rk} - \bar{z}_k)(z_{rl} - \bar{z}_l)$$

The formula of the asymptotic covariances of uncorrected covariances (using the UCOV or NOINT option) is a straightforward generalization of this expression.

The resulting weight matrix **W** is at least positive semidefinite (except for rounding errors). Using the ASYCOV option, you can use Browne's (1984, formula (3.8)) unbiased estimators

$$\begin{aligned} w_{ij,kl} &= \frac{N(N-1)}{(N-2)(N-3)}(s_{ij,kl} - s_{ij} s_{kl}) \\ &\quad - \frac{N}{(N-2)(N-3)}(s_{ik} s_{jl} + s_{il} s_{jk} - \frac{2}{N-1} s_{ij} s_{kl}) \end{aligned}$$

There is no guarantee that this weight matrix is positive semidefinite. However, the second part is of order $O(N^{-1})$ and does not destroy the positive semidefinite first part for sufficiently large N. For a large number of independent observations, default settings of the weight matrix **W** result in asymptotically distribution-free parameter estimates with unbiased standard errors and a correct χ^2 test statistic (Browne 1982, 1984).

If the moment matrix **S** is a correlation (rather than a covariance) matrix, the default setting of $\mathbf{W} = (w_{ij,kl})$ is the estimators of the asymptotic covariances $\sigma_{ij,kl}$ of the correlations $\mathbf{S} = (s_{ij})$ (Browne and Shapiro 1986; DeLeeuw 1983)

$$\begin{aligned} w_{ij,kl} &= r_{ij,kl} - \frac{1}{2} r_{ij}(r_{ii,kl} + r_{jj,kl}) - \frac{1}{2} r_{kl}(r_{kk,ij} + r_{ll,ij}) \\ &\quad + \frac{1}{4} r_{ij} r_{kl}(r_{ii,kk} + r_{ii,ll} + r_{jj,kk} + r_{jj,ll}) \end{aligned}$$

where

$$r_{ij,kl} = \frac{s_{ij,kl}}{\sqrt{s_{ii} s_{jj} s_{kk} s_{ll}}}$$

The asymptotic variances of the diagonal elements of a correlation matrix are 0. Therefore, the weight matrix computed by Browne and Shapiro's formula is always singular. In this case the goodness of fit function for weighted least squares estimation is modified to

$$F_{WLS} = \sum_{i=2}^{n} \sum_{j=1}^{i-1} \sum_{k=2}^{n} \sum_{l=1}^{k-1} w^{ij,kl}(s_{ij} - c_{ij})(s_{kl} - c_{kl}) + r \sum_{i}^{n} (s_{ii} - c_{ii})^2$$

where r is the penalty weight specified by the WPENALTY=r option and the $w^{ij,kl}$ are the elements of the inverse of the reduced $(n(n-1)/2) \times (n(n-1)/2)$ weight matrix that contains only the nonzero rows and columns of the full weight matrix $\mathbf{W}$. The second term is a penalty term to fit the diagonal elements of the moment matrix $\mathbf{S}$. The default value of $r = 100$ can be decreased or increased by the WPENALTY= option. The often used value of $r = 1$ seems to be too small in many cases to fit the diagonal elements of a correlation matrix properly. If your model does not fit the diagonal of the moment matrix $\mathbf{S}$, you can specify the NODIAG option to exclude the diagonal elements from the fit function.

Storing and inverting the huge weight matrix $\mathbf{W}$ in WLS estimation needs considerable computer resources. A compromise is found by implementing the DWLS method that uses only the diagonal of the weight matrix $\mathbf{W}$ from the WLS estimation in the minimization function

$$F_{DWLS} = Vec(s_{ij} - c_{ij})' diag(\mathbf{W})^{-1} Vec(s_{ij} - c_{ij})$$

The statistical properties of DWLS estimates are still not known.

In generalized, weighted, or diagonally weighted least squares estimation, you can change from the default settings of weight matrices $\mathbf{W}$ by using an INWGT= data set. Because the diagonal elements $w_{ii,kk}$ of the weight matrix $\mathbf{W}$ are interpreted as asymptotic variances of the sample covariances or correlations, they cannot be negative. The CALIS procedure requires a positive-definite weight matrix that has positive diagonal elements.

Relationships among Estimation Criteria

The five estimation functions, F_{ULS}, F_{GLS}, F_{ML}, F_{WLS}, and F_{DWLS}, belong to the following two groups:

- The functions F_{ULS}, F_{GLS}, and F_{ML} take into account all n^2 elements of the symmetric residual matrix $\mathbf{S} - \mathbf{C}$. This means that the off-diagonal residuals contribute twice to F, as lower and as upper triangular elements.
- The functions F_{WLS} and F_{DWLS} take into account only the $n(n+1)/2$ lower triangular elements of the symmetric residual matrix $\mathbf{S} - \mathbf{C}$. This means that the off-diagonal residuals contribute to F only once.

The F_{DWLS} function used in PROC CALIS differs from that used by the LISREL 7 program. Formula (1.25) of the LISREL 7 manual (Jöreskog and Sörbom 1988, p. 23) shows that LISREL groups the F_{DWLS} function in the first group by taking into account all n^2 elements of the symmetric residual matrix $\mathbf{S} - \mathbf{C}$.

- Relationship between DWLS and WLS:
  PROC CALIS: The F_{DWLS} and F_{WLS} estimation functions deliver the same results for the special case that the weight matrix $\mathbf{W}$ used by WLS estimation is a diagonal matrix.
  LISREL 7: This is not the case.

- Relationship between DWLS and ULS:
  LISREL 7: The F_{DWLS} and F_{ULS} estimation functions deliver the same results for the special case that the diagonal weight matrix $\mathbf{W}$ used by DWLS estimation is an identity matrix (contains only 1s).
  PROC CALIS: To obtain the same results with F_{DWLS} and F_{ULS} estimation, set the diagonal weight matrix $\mathbf{W}$ used in DWLS estimation to

  $$w_{ik,ik} = \begin{cases} 1. & \text{if } i = k \\ 0.5 & \text{otherwise} \end{cases}$$

  Because the reciprocal elements of the weight matrix are used in the goodness of fit function, the off-diagonal residuals are weighted by a factor of 2.

Testing Rank Deficiency in the Approximate Covariance Matrix

The inverse of the information matrix (or approximate Hessian matrix) is used for the covariance matrix of the parameter estimates, which is needed for the computation of approximate standard errors and modification indices. The numerical condition of the information matrix (computed as the crossproduct $\mathbf{J}'\mathbf{J}$ of the Jacobian matrix $\mathbf{J}$) can be very poor in many practical applications, especially for the analysis of unscaled covariance data. The following four-step strategy is used for the inversion of the information matrix.

1. The inversion (usually of a normalized matrix $\mathbf{D}^{-1}\mathbf{H}\mathbf{D}^{-1}$) is tried using a modified form of the Bunch and Kaufman (1977) algorithm, which allows the specification of a different singularity criterion for each pivot. The following three criteria for the detection of rank loss in the information matrix are used to specify thresholds:

   - *ASING* specifies absolute singularity.
   - *MSING* specifies relative singularity depending on the whole matrix norm.
   - *VSING* specifies relative singularity depending on the column matrix norm.

   If no rank loss is detected, the inverse of the information matrix is used for the covariance matrix of parameter estimates, and the next two steps are skipped.

2. The linear dependencies among the parameter subsets are displayed based on the singularity criteria.

3. If the number of parameters t is smaller than the value specified by the G4= option (the default value is 60), the Moore-Penrose inverse is computed based on the eigenvalue decomposition of the information matrix. If you do not specify the NOPRINT option, the distribution of eigenvalues is displayed, and those eigenvalues that are set to zero in the Moore-Penrose inverse are indicated. You should inspect this eigenvalue distribution carefully.

4. If PROC CALIS did not set the right subset of eigenvalues to zero, you can specify the COVSING= option to set a larger or smaller subset of eigenvalues to zero in a further run of PROC CALIS.

Approximate Standard Errors

Except for unweighted and diagonally weighted least squares estimation, approximate standard errors can be computed as the diagonal elements of the matrix

$$\frac{c}{NM}\mathbf{H}^{-1}, \quad \text{where}$$

$$NM = \begin{cases} (N-1) & \text{if the CORR or COV matrix is analyzed} \\ & \text{or the intercept variable is not used in the model} \\ N & \text{if the UCORR or UCOV matrix is analyzed} \\ & \text{and the intercept variable is not used in the model} \end{cases}$$

The matrix $\mathbf{H}$ is the approximate Hessian matrix of F evaluated at the final estimates, $c = 1$ for the WLS estimation method, $c = 2$ for the GLS and ML method, and N is the sample size. If a given correlation or covariance matrix is singular, PROC CALIS offers two ways to compute a generalized inverse of the information matrix and, therefore, two ways to compute approximate standard errors of implicitly constrained parameter estimates, t values, and modification indices. Depending on the G4= specification, either a Moore-Penrose inverse or a G2 inverse is computed. The expensive Moore-Penrose inverse computes an estimate of the null space by using an eigenvalue decomposition. The cheaper G2 inverse is produced by sweeping the linearly independent rows and columns and zeroing out the dependent ones. The information matrix, the approximate covariance matrix of the parameter estimates, and the approximate standard errors are not computed in the cases of unweighted or diagonally weighted least squares estimation.

Assessment of Fit

This section contains a collection of formulas used in computing indices to assess the goodness of fit by PROC CALIS. The following notation is used:

- N for the sample size
- n for the number of manifest variables
- t for the number of parameters to estimate
- $NM = \begin{cases} (N-1) & \text{if the CORR or COV matrix is analyzed} \\ & \text{or the intercept variable is not used in the model} \\ N & \text{if the UCORR or UCOV matrix is analyzed} \\ & \text{and the intercept variable is not used in the model} \end{cases}$
- df for the degrees of freedom
- $\boldsymbol{\gamma} = \mathbf{X}$ for the t vector of optimal parameter estimates
- $\mathbf{S} = (s_{ij})$ for the $n \times n$ input COV, CORR, UCOV, or UCORR matrix
- $\mathbf{C} = (c_{ij}) = \hat{\boldsymbol{\Sigma}} = \boldsymbol{\Sigma}(\hat{\boldsymbol{\gamma}})$ for the predicted model matrix

- $\mathbf{W}$ for the weight matrix ($\mathbf{W} = \mathbf{I}$ for ULS, $\mathbf{W} = \mathbf{S}$ for default GLS, and $\mathbf{W} = \mathbf{C}$ for ML estimates)
- $\mathbf{U}$ for the $n^2 \times n^2$ asymptotic covariance matrix of sample covariances
- $\Phi(x|\lambda, df)$ for the cumulative distribution function of the noncentral chi-squared distribution with noncentrality parameter λ

The following notation is for indices that support testing nested models by a χ^2 difference test:

- f_0 for the function value of the independence model
- df_0 for the degrees of freedom of the independence model
- $f_{min} = F$ for the function value of the fitted model
- $df_{min} = df$ for the degrees of freedom of the fitted model

The degrees of freedom df_{min} and the number of parameters t are adjusted automatically when there are active constraints in the analysis. The computation of many fit statistics and indices are affected. You can turn off the automatic adjustment by using the NOADJDF option. See the section "Counting the Degrees of Freedom" on page 956 for more information.

Residuals

PROC CALIS computes four types of residuals and writes them to the OUTSTAT= data set.

- **Raw Residuals**

  $$Res = \mathbf{S} - \mathbf{C}, Res_{ij} = s_{ij} - c_{ij}$$

  The raw residuals are displayed whenever the PALL, PRINT, or RESIDUAL option is specified.

- **Variance Standardized Residuals**

  $$VSRes_{ij} = \frac{s_{ij} - c_{ij}}{\sqrt{s_{ii}s_{jj}}}$$

  The variance standardized residuals are displayed when you specify the following:

  - the PALL, PRINT, or RESIDUAL option and METHOD=NONE, METHOD=ULS, or METHOD=DWLS
  - RESIDUAL=VARSTAND

  The variance standardized residuals are equal to those computed by the EQS 3 program (Bentler 1989).

- **Asymptotically Standardized Residuals**

  $$ASRes_{ij} = \frac{s_{ij} - c_{ij}}{\sqrt{v_{ij,ij}}}, \quad \text{where}$$

  $$v_{ij,ij} = diag(\mathbf{U} - \mathbf{J}Cov(\boldsymbol{\gamma})\mathbf{J}')_{ij}$$

  The matrix **J** is the $n^2 \times t$ Jacobian matrix $d\boldsymbol{\Sigma}/d\boldsymbol{\gamma}$, and $Cov(\boldsymbol{\gamma})$ is the $t \times t$ asymptotic covariance matrix of parameter estimates (the inverse of the information matrix). Asymptotically standardized residuals are displayed when one of the following conditions is met:

  - The PALL, PRINT, or RESIDUAL option is specified, and METHOD=ML, METHOD=GLS, or METHOD=WLS, and the expensive information and Jacobian matrices are computed for some other reason.
  - RESIDUAL= ASYSTAND is specified.

  The asymptotically standardized residuals are equal to those computed by the LISREL 7 program (Jöreskog and Sörbom 1988) except for the denominator NM in the definition of matrix **U**.

- **Normalized Residuals**

  $$NRes_{ij} = \frac{s_{ij} - c_{ij}}{\sqrt{u_{ij,ij}}}$$

  where the diagonal elements $u_{ij,ij}$ of the $n^2 \times n^2$ asymptotic covariance matrix **U** of sample covariances are defined for the following methods.

  - **GLS** as $u_{ij,ij} = \frac{1}{NM}(s_{ii}s_{jj} + s_{ij}^2)$
  - **ML** as $u_{ij,ij} = \frac{1}{NM}(c_{ii}c_{jj} + c_{ij}^2)$
  - **WLS** as $u_{ij,ij} = \frac{1}{NM}w_{ij,ij}$

  Normalized residuals are displayed when one of the following conditions is met:

  - The PALL, PRINT, or RESIDUAL option is specified, and METHOD=ML, METHOD=GLS, or METHOD=WLS, and the expensive information and Jacobian matrices are **not** computed for some other reason.
  - RESIDUAL=NORM is specified.

  The normalized residuals are equal to those computed by the LISREL VI program (Jöreskog and Sörbom 1985) except for the definition of the denominator NM in matrix **U**.

For estimation methods that are not BGLS estimation methods (Browne 1982, 1984), such as METHOD=NONE, METHOD=ULS, or METHOD=DWLS, the assumption of an asymptotic covariance matrix **U** of sample covariances does not seem to be appropriate. In this case, the normalized residuals should be replaced by the more relaxed variance standardized residuals. Computation of asymptotically standardized residuals requires computing the Jacobian and information matrices. This is computationally very expensive and is done only if the Jacobian matrix has to be computed for some other reason—that is, if at least one of the following items is true:

- The default, PRINT, or PALL displayed output is requested, and neither the NOMOD nor NOSTDERR option is specified.
- Either the MODIFICATION (included in PALL), PCOVES, or STDERR (included in default, PRINT, and PALL output) option is requested or RESIDUAL=ASYSTAND is specified.
- The LEVMAR or NEWRAP optimization technique is used.
- An OUTRAM= data set is specified without using the NOSTDERR option.
- An OUTEST= data set is specified without using the NOSTDERR option.

Since normalized residuals use an overestimate of the asymptotic covariance matrix of residuals (the diagonal of $\mathbf{U}$), the normalized residuals cannot be larger than the asymptotically standardized residuals (which use the diagonal of $\mathbf{U} - \mathbf{J}Cov(\boldsymbol{\gamma})\mathbf{J}'$).

Together with the residual matrices, the values of the average residual, the average off-diagonal residual, and the rank order of the largest values are displayed. The distribution of the normalized and standardized residuals is displayed also.

Goodness of Fit Indices Based on Residuals

The following items are computed for all five kinds of estimation: ULS, GLS, ML, WLS, and DWLS. All these indices are written to the OUTRAM= data set. The goodness of fit (GFI), adjusted goodness of fit (AGFI), and root mean square residual (RMR) are computed as in the LISREL VI program of Jöreskog and Sörbom (1985).

- **Goodness of Fit Index**
  The goodness of fit index for the ULS, GLS, and ML estimation methods is

  $$GFI = 1 - \frac{Tr((\mathbf{W}^{-1}(\mathbf{S} - \mathbf{C}))^2)}{Tr((\mathbf{W}^{-1}\mathbf{S})^2)}$$

  but for WLS and DWLS estimation, it is

  $$GFI = 1 - \frac{Vec(s_{ij} - c_{ij})'\mathbf{W}^{-1}Vec(s_{ij} - c_{ij})}{Vec(s_{ij})'\mathbf{W}^{-1}Vec(s_{ij})}$$

  where $\mathbf{W} = diag$ for DWLS estimation, and $Vec(s_{ij} - c_{ij})$ denotes the vector of the $n(n+1)/2$ elements of the lower triangle of the symmetric matrix $\mathbf{S} - \mathbf{C}$. For a constant weight matrix $\mathbf{W}$, the goodness of fit index is 1 minus the ratio of the minimum function value and the function value before any model has been fitted. The GFI should be between 0 and 1. The data probably do not fit the model if the GFI is negative or much larger than 1.

- **Adjusted Goodness of Fit Index**
  The AGFI is the GFI adjusted for the degrees of freedom of the model

  $$AGFI = 1 - \frac{n(n+1)}{2df}(1 - GFI)$$

The AGFI corresponds to the GFI in replacing the total sum of squares by the mean sum of squares.

CAUTION:

- Large n and small df can result in a negative AGFI. For example, GFI= 0.90, $n = 19$, and $df = 2$ result in an AGFI of -8.5.
- AGFI is not defined for a saturated model, due to division by $df = 0$.
- AGFI is not sensitive to losses in df.

The AGFI should be between 0 and 1. The data probably do not fit the model if the AGFI is negative or much larger than 1. For more information, refer to Mulaik et al. (1989).

- **Root Mean Square Residual**
  The RMR is the root of the mean of the squared residuals:

$$RMR = \sqrt{\frac{2}{n(n+1)} \sum_i^n \sum_j^i (s_{ij} - c_{ij})^2}$$

- **Standardized Root Mean Square Residual**
  The SRMR is the root of the mean of the squared standardized residuals:

$$SRMR = \sqrt{\frac{2}{n(n+1)} \sum_i^n \sum_j^i \frac{(s_{ij} - c_{ij})^2}{s_{ii}s_{jj}}}$$

- **Parsimonious Goodness of Fit Index**
  The PGFI (Mulaik et al. 1989) is a modification of the GFI that takes the parsimony of the model into account:

$$PGFI = \frac{df_{min}}{df_0} GFI$$

The PGFI uses the same parsimonious factor as the parsimonious normed Bentler-Bonett index (James, Mulaik, and Brett 1982).

Goodness of Fit Indices Based on the χ^2

The following items are transformations of the overall χ^2 value and in general depend on the sample size N. These indices are not computed for ULS or DWLS estimates.

- **Uncorrected χ^2**
  The overall χ^2 measure is the optimum function value F multiplied by $N - 1$ if a CORR or COV matrix is analyzed, or multiplied by N if a UCORR or UCOV matrix is analyzed. This gives the likelihood ratio test statistic for the null hypothesis that the predicted matrix **C** has the specified model structure against the alternative that **C** is unconstrained. The χ^2 test is valid only if the observations are independent and identically distributed, the analysis is based

on the nonstandardized sample covariance matrix **S**, and the sample size N is sufficiently large (Browne 1982; Bollen 1989b; Jöreskog and Sörbom 1985). For ML and GLS estimates, the variables must also have an approximately multivariate normal distribution. The notation Prob>Chi**2 means "the probability under the null hypothesis of obtaining a greater χ^2 statistic than that observed."

$$\chi^2 = NM * F$$

where F is the function value at the minimum.

- **χ_0^2 Value of the Independence Model**
  The χ_0^2 value of the independence model

  $$\chi_0^2 = NM * f_0$$

  and the corresponding degrees of freedom df_0 can be used (in large samples) to evaluate the gain of explanation by fitting the specific model (Bentler 1989).

- **RMSEA Index (Steiger and Lind 1980)**
  The Steiger and Lind (1980) root mean squared error approximation (RMSEA) coefficient is

  $$\epsilon_\alpha = \sqrt{\max(\frac{F}{df} - \frac{1}{NM}, 0)}$$

  The lower and upper limits of the confidence interval are computed using the cumulative distribution function of the noncentral chi-squared distribution $\Phi(x|\lambda, df) = \alpha$, with $x = NM * F$, λ_L satisfying $\Phi(x|\lambda_L, df) = 1 - \frac{\alpha}{2}$, and λ_U satisfying $\Phi(x|\lambda_U, df) = \frac{\alpha}{2}$:

  $$(\epsilon_{\alpha_L}; \epsilon_{\alpha_U}) = (\sqrt{\frac{\lambda_L}{NM * df}}; \sqrt{\frac{\lambda_U}{NM * df}})$$

  Refer to Browne and Du Toit (1992) for more details. The size of the confidence interval is defined by the option ALPHARMS=α, $0 \le \alpha \le 1$. The default is $\alpha = 0.1$, which corresponds to the 90% confidence interval for the RMSEA.

- **Probability for Test of Close Fit (Browne and Cudeck 1993)**
  The traditional exact χ^2 test hypothesis H_0: $\epsilon_\alpha = 0$ is replaced by the null hypothesis of close fit H_0: $\epsilon_\alpha \le 0.05$ and the exceedance probability P is computed as

  $$P = 1 - \Phi(x|\lambda^*, df)$$

  where $x = NM * F$ and $\lambda^* = 0.05^2 * NM * df$. The null hypothesis of close fit is rejected if P is smaller than a prespecified level (for example, $P < 0.05$).

- **Expected Cross Validation Index (Browne and Cudeck 1993)**
  For GLS and WLS, the estimator c of the ECVI is linearly related to AIC:

  $$c = F(\mathbf{S}, \mathbf{C}) + \frac{2t}{NM}$$

  For ML estimation, c_{ML} is used.

  $$c_{ML} = F_{ML}(\mathbf{S}, \mathbf{C}) + \frac{2t}{NM - n - 1}$$

The confidence interval $(c_L; c_U)$ for c is computed using the cumulative distribution function $\Phi(x|\lambda, df)$ of the noncentral chi-squared distribution,

$$(c_L; c_U) = (\frac{\lambda_L + nnt}{NM}; \frac{\lambda_U + nnt}{NM})$$

with $nnt = n(n+1)/2 + t$, $x = NM * F$, $\Phi(x|\lambda_U, df) = 1 - \frac{\alpha}{2}$, and $\Phi(x|\lambda_L, df) = \frac{\alpha}{2}$. The confidence interval $(c_L^*; c_U^*)$ for c_{ML} is

$$(c_L^*; c_U^*) = (\frac{\lambda_L^* + nnt}{NM - n - 1}); \frac{\lambda_U^* + nnt}{NM - n - 1})$$

where $nnt = n(n+1)/2+t$, $x = (NM-n-1)*F$, $\Phi(x|\lambda_U^*, df) = \frac{\alpha}{2}$ and $\Phi(x|\lambda_L^*, df) = 1 - \frac{\alpha}{2}$. Refer to Browne and Cudeck (1993) for details. The size of the confidence interval is defined by the option ALPHAECV=α, $0 \le \alpha \le 1$. The default is $\alpha = 0.1$, which corresponds to the 90% confidence interval for the ECVI.

- **Comparative Fit Index (Bentler 1989)**

$$CFI = 1 - \frac{\max(NM * f_{min} - df_{min}, 0)}{\max(NM * f_0 - df_0, 0)}$$

- **Adjusted χ^2 Value (Browne 1982)**
  If the variables are n-variate elliptic rather than normal and have significant amounts of multivariate kurtosis (leptokurtic or platykurtic), the χ^2 value can be adjusted to

$$\chi^2_{ell} = \frac{\chi^2}{\eta_2}$$

  where η_2 is the multivariate relative kurtosis coefficient.

- **Normal Theory Reweighted LS χ^2 Value**
  This index is displayed only if METHOD=ML. Instead of the function value F_{ML}, the reweighted goodness of fit function F_{GWLS} is used,

$$\chi^2_{GWLS} = NM * F_{GWLS}$$

  where F_{GWLS} is the value of the function at the minimum.

- **Akaike's Information Criterion (AIC) (Akaike 1974; Akaike 1987)**
  This is a criterion for selecting the best model among a number of candidate models. The model that yields the smallest value of AIC is considered the best.

$$AIC = \chi^2 - 2df$$

- **Consistent Akaike's Information Criterion (CAIC) (Bozdogan 1987)**
  This is another criterion, similar to AIC, for selecting the best model among alternatives. The model that yields the smallest value of CAIC is considered the best. CAIC is preferred by some people to AIC or the χ^2 test.

$$CAIC = \chi^2 - (ln(N) + 1)df$$

- **Schwarz's Bayesian Criterion (SBC) (Schwarz 1978; Sclove 1987)**
  This is another criterion, similar to AIC, for selecting the best model. The model that yields the smallest value of SBC is considered the best. SBC is preferred by some people to AIC or the χ^2 test.

  $$SBC = \chi^2 - ln(N)df$$

- **McDonald's Measure of Centrality (McDonald 1989)**

  $$CENT = exp(-\frac{(\chi^2 - df)}{2N})$$

- **Parsimonious Normed Fit Index (James, Mulaik, and Brett 1982)**
  The PNFI is a modification of Bentler-Bonett's normed fit index that takes parsimony of the model into account,

  $$PNFI = \frac{df_{min}}{df_0} \frac{(f_0 - f_{min})}{f_0}$$

  The PNFI uses the same parsimonious factor as the parsimonious GFI of Mulaik et al. (1989).

- **Z-Test (Wilson and Hilferty 1931)**
  The Z-test of Wilson and Hilferty assumes an n-variate normal distribution:

  $$Z = \frac{\sqrt[3]{\frac{\chi^2}{df}} - (1 - \frac{2}{9df})}{\sqrt{\frac{2}{9df}}}$$

  Refer to McArdle (1988) and Bishop, Fienberg, and Holland (1977, p. 527) for an application of the Z-test.

- **Nonnormed Coefficient (Bentler and Bonett 1980)**

  $$\rho = \frac{f_0/df_0 - f_{min}/df_{min}}{f_0/df_0 - 1/NM}$$

  Refer to Tucker and Lewis (1973) for details.

- **Normed Coefficient (Bentler and Bonett 1980)**

  $$\Delta = \frac{f_0 - f_{min}}{f_0}$$

  Mulaik et al. (1989) recommend the parsimonious weighted form PNFI.

- **Normed Index ρ_1 (Bollen 1986)**

  $$\rho_1 = \frac{f_0/df_0 - f_{min}/df_{min}}{f_0/df_0}$$

  ρ_1 is always less than or equal to 1; $\rho_1 < 0$ is unlikely in practice. Refer to the discussion in Bollen (1989a) for details.

- **Nonnormed Index Δ_2 (Bollen 1989a)**

$$\Delta_2 = \frac{f_0 - f_{min}}{f_0 - \frac{df}{NM}}$$

is a modification of Bentler and Bonett's Δ that uses df and "lessens the dependence" on N. Refer to the discussion in Bollen (1989b). Δ_2 is identical to Mulaik et al.'s (1989) IFI2 index.

- **Critical N Index (Hoelter 1983)**

$$CN = \frac{\chi^2_{crit}}{F} + 1$$

where χ^2_{crit} is the critical chi-square value for the given df degrees of freedom and probability $\alpha = 0.05$, and F is the value of the estimation criterion (minimization function). Refer to Bollen (1989b, p. 277) for details. Hoelter (1983) suggests that CN should be at least 200; however, Bollen (1989b) notes that the CN value might lead to an overly pessimistic assessment of fit for small samples.

Squared Multiple Correlation

The following are measures of the squared multiple correlation for manifest and endogenous variables and are computed for all five estimation methods: ULS, GLS, ML, WLS, and DWLS. These coefficients are computed as in the LISREL VI program of Jöreskog and Sörbom (1985). The DETAE, DETSE, and DETMV determination coefficients are intended to be global means of the squared multiple correlations for different subsets of model equations and variables. These coefficients are displayed only when you specify the PDETERM option with a RAM or LINEQS model.

- **R^2 Values Corresponding to Endogenous Variables**

$$R_i^2 = 1 - \frac{\widehat{var(\zeta_i)}}{\widehat{var(\eta_i)}}$$

- **Total Determination of All Equations**

$$DETAE = 1 - \frac{det(\hat{\mathbf{\Theta}}, \hat{\mathbf{\Psi}})}{det(\widehat{Cov(y, x, \eta)})}$$

- **Total Determination of the Structural Equations**

$$DETSE = 1 - \frac{det(\hat{\mathbf{\Psi}})}{det(\widehat{Cov(\eta)})}$$

- **Total Determination of the Manifest Variables**

$$DETMV = 1 - \frac{det(\hat{\mathbf{\Theta}})}{det(\mathbf{S})}$$

CAUTION: In the LISREL program, the structural equations are defined by specifying the BETA matrix. In PROC CALIS, a structural equation has a dependent left-hand-side variable that appears at least once on the right-hand side of another equation, or the equation has at least one right-hand-side variable that is the left-hand-side variable of another equation. Therefore, PROC CALIS sometimes identifies more equations as structural equations than the LISREL program does.

Latent Variable Scores

Although latent variable or factor scores are unobserved, they can be estimated after you fit your model using PROC CALIS. Latent variable or factor scores are estimated as linear combinations of observed variables, weighted by the latent variable (factor) score regression coefficients.

To display the latent variable (factor) score regression coefficients in the PROC CALIS output, you can use the PLATCOV option. You can also save these coefficients in an output data set by using the OUTSTAT= option. These coefficients can then be used by the SCORE procedure to compute the latent variable (factor) scores.

To summarize, do the following if you need to compute latent variable (factor) scores for each observation by using the SCORE procedure:

- Create an output data set with the OUTSTAT= option in the PROC CALIS statement.
- Use the SCORE procedure with both the raw data and the OUTSTAT= data set.

For example, you can use the following statements to compute the latent variable (factor) scores, which are stored in the OUT= data set named “**scores**”:

```
proc calis data=raw outstat=ostat;
   lineqs
      v1 = a1 f1 + e1,
      v2 = a2 f1 + e2,
      v3 = a3 f1 + e3;
   std
      f1 = 1.,
      e1-e3 = evar1-evar3;
run;

proc score data=raw score=ostat out=scores;
run;
```

If you analyze a correlation or covariance matrix in PROC CALIS, the original raw data must be used in the PROC SCORE procedure, as shown in the following statements:

```
proc corr data=raw out=correl;
run;

proc calis data=correl outstat=ostat;
   lineqs
      v1 = a1 f1 + e1,
      v2 = a2 f1 + e2,
      v3 = a3 f1 + e3;
   std
      f1 = 1.,
      e1-e3 = evar1-evar3;
run;

proc score data=raw score=ostat out=scores;
run;
```

For a more detailed example, see Example 76.1 in Chapter 76, “The SCORE Procedure.” Although the FACTOR procedure is used in that example, it is also applicable to the CALIS procedure. Essentially, an OUTSTAT= data set containing the latent variable or factor score regression coefficients, either from PROC FACTOR or PROC CALIS, is used with the raw data in the SCORE procedure to create the latent variable or factor scores.

Measures of Multivariate Kurtosis

In many applications, the manifest variables are not even approximately multivariate normal. If this happens to be the case with your data set, the default generalized least squares and maximum likelihood estimation methods are not appropriate, and you should compute the parameter estimates and their standard errors by an asymptotically distribution-free method, such as the WLS estimation method. If your manifest variables are multivariate normal, then they have a zero relative multivariate kurtosis, and all marginal distributions have zero kurtosis (Browne 1982). If your DATA= data set contains raw data, PROC CALIS computes univariate skewness and kurtosis and a set of multivariate kurtosis values. By default, the values of univariate skewness and kurtosis are corrected for bias (as in PROC UNIVARIATE), but using the BIASKUR option enables you to compute the uncorrected values also. The values are displayed when you specify the PROC CALIS statement option KURTOSIS.

- **Corrected Variance for Variable** z_j

$$\sigma_j^2 = \frac{1}{N-1}\sum_i^N (z_{ij} - \bar{z}_j)^2$$

- **Corrected Univariate Skewness for Variable** z_j

$$\gamma_{1(j)} = \frac{N}{(N-1)(N-2)} \frac{\sum_i^N (z_{ij} - \bar{z}_j)^3}{\sigma_j^3}$$

- **Uncorrected Univariate Skewness for Variable** z_j

$$\gamma_{1(j)} = \frac{N \sum_i^N (z_{ij} - \bar{z}_j)^3}{\sqrt{N[\sum_i^N (z_{ij} - \bar{z}_j)^2]^3}}$$

- **Corrected Univariate Kurtosis for Variable** z_j

$$\gamma_{2(j)} = \frac{N(N+1)}{(N-1)(N-2)(N-3)} \frac{\sum_i^N (z_{ij} - \bar{z}_j)^4}{\sigma_j^4} - \frac{3(N-1)^2}{(N-2)(N-3)}$$

- **Uncorrected Univariate Kurtosis for Variable** z_j

$$\gamma_{2(j)} = \frac{N \sum_i^N (z_{ij} - \bar{z}_j)^4}{[\sum_i^N (z_{ij} - \bar{z}_j)^2]^2} - 3$$

- **Mardia's Multivariate Kurtosis**

$$\gamma_2 = \frac{1}{N}\sum_i^N [(z_i - \bar{z})' S^{-1} (z_i - \bar{z})]^2 - n(n+2)$$

- **Relative Multivariate Kurtosis**

$$\eta_2 = \frac{\gamma_2 + n(n+2)}{n(n+2)}$$

- **Normalized Multivariate Kurtosis**

$$\kappa_0 = \frac{\gamma_2}{\sqrt{8n(n+2)/N}}$$

- **Mardia-Based Kappa**

$$\kappa_1 = \frac{\gamma_2}{n(n+2)}$$

- **Mean Scaled Univariate Kurtosis**

$$\kappa_2 = \frac{1}{3n}\sum_j^n \gamma_{2(j)}$$

- **Adjusted Mean Scaled Univariate Kurtosis**

  $$\kappa_3 = \frac{1}{3n} \sum_{j}^{n} \gamma^*_{2(j)}$$

  with

  $$\gamma^*_{2(j)} = \begin{cases} \gamma_{2(j)} \quad , & if \quad \gamma_{2(j)} > \frac{-6}{n+2} \\ \frac{-6}{n+2} \quad , & \text{otherwise} \end{cases}$$

If variable Z_j is normally distributed, the uncorrected univariate kurtosis $\gamma_{2(j)}$ is equal to 0. If Z has an n-variate normal distribution, Mardia's multivariate kurtosis γ_2 is equal to 0. A variable Z_j is called *leptokurtic* if it has a positive value of $\gamma_{2(j)}$ and is called *platykurtic* if it has a negative value of $\gamma_{2(j)}$. The values of κ_1, κ_2, and κ_3 should not be smaller than a lower bound (Bentler 1985):

$$\hat{\kappa} \geq \frac{-2}{n+2}$$

PROC CALIS displays a message if this happens.

If weighted least squares estimates (METHOD=WLS or METHOD=ADF) are specified and the weight matrix is computed from an input raw data set, the CALIS procedure computes two more measures of multivariate kurtosis.

- **Multivariate Mean Kappa**

  $$\kappa_4 = \frac{1}{m} \sum_{i}^{n} \sum_{j}^{i} \sum_{k}^{j} \sum_{l}^{k} \hat{\kappa}_{ij,kl} - 1$$

  where

  $$\hat{\kappa}_{ij,kl} = \frac{s_{ij,kl}}{s_{ij}s_{kl} + s_{ik}s_{jl} + s_{il}s_{jk}}$$

  and $m = n(n+1)(n+2)(n+3)/24$ is the number of elements in the vector $s_{ij,kl}$ (Bentler 1985).

- **Multivariate Least Squares Kappa**

  $$\kappa_5 = \frac{s_4' s_2}{s_2' s_2} - 1$$

  where

  $$s_{ij,kl} = \frac{1}{N} \sum_{r=1}^{N} (z_{ri} - \bar{z}_i)(z_{rj} - \bar{z}_j)(z_{rk} - \bar{z}_k)(z_{rl} - \bar{z}_l)$$

  s_4 is the vector of the $s_{ij,kl}$, and s_2 is the vector of the elements in the denominator of $\hat{\kappa}$ (Bentler 1985).

The occurrence of significant nonzero values of Mardia's multivariate kurtosis γ_2 and significant amounts of some of the univariate kurtosis values $\gamma_{2(j)}$ indicate that your variables are not multivariate normal distributed. Violating the multivariate normality assumption in (default) generalized least squares and maximum likelihood estimation usually leads to the wrong approximate standard errors and incorrect fit statistics based on the χ^2 value. In general, the parameter estimates are more stable against violation of the normal distribution assumption. For more details, refer to Browne (1974, 1982, 1984).

Initial Estimates

Each optimization technique requires a set of initial values for the parameters. To avoid local optima, the initial values should be as close as possible to the globally optimal solution. You can check for local optima by running the analysis with several different sets of initial values; the RANDOM= option in the PROC CALIS statement is useful in this regard.

- RAM and LINEQS: There are several default estimation methods available in PROC CALIS for initial values of parameters in a linear structural equation model specified by a RAM or LINEQS model statement, depending on the form of the specified model.
  - two-stage least squares estimation
  - instrumental variable method (Hägglund 1982; Jennrich 1987)
  - approximative factor analysis method
  - ordinary least squares estimation
  - estimation method of McDonald (McDonald and Hartmann 1992)
- FACTOR: For default (exploratory) factor analysis, PROC CALIS computes initial estimates for factor loadings and unique variances by an algebraic method of approximate factor analysis. If you use a MATRIX statement together with a FACTOR model specification, initial values are computed by McDonald's (McDonald and Hartmann 1992) method if possible. McDonald's method of computing initial values works better if you scale the factors by setting the factor variances to 1 rather than setting the loadings of the reference variables equal to 1. If neither of the two methods seems to be appropriate, the initial values are set by the START= option.
- COSAN: For the more general COSAN model, there is no default estimation method for the initial values. In this case, the START= or RANDOM= option can be used to set otherwise unassigned initial values.

Poor initial values can cause convergence problems, especially with maximum likelihood estimation. You should not specify a constant initial value for all parameters since this would produce a singular predicted model matrix in the first iteration. Sufficiently large positive diagonal elements in the central matrices of each model matrix term provide a nonnegative definite initial predicted model matrix. If maximum likelihood estimation fails to converge, it might help to use METHOD=LSML, which uses the final estimates from an unweighted least squares analysis as initial estimates for maximum likelihood. Or you can fit a slightly different but better-behaved model and produce an OUTRAM= data set, which can then be modified in accordance with the original model and used as an INRAM= data set to provide initial values for another analysis.

If you are analyzing a covariance or scalar product matrix, be sure to take into account the scales of the variables. The default initial values might be inappropriate when some variables have extremely large or small variances.

Automatic Variable Selection

You can use the VAR statement to reorder the variables in the model and to delete the variables not used. Using the VAR statement saves memory and computation time. If a linear structural equation model that uses the RAM or LINEQS statement (or an INRAM= data set specifying a RAM or LINEQS model) does not use all the manifest variables given in the input DATA= data set, PROC CALIS automatically deletes those manifest variables not used in the model.

In some special circumstances, the automatic variable selection performed for the RAM and LINEQS statements might be inappropriate—for example, if you are interested in modification indices connected to some of the variables that are not used in the model. You can include such manifest variables as exogenous variables in the analysis by specifying constant zero coefficients.

For example, the first three steps in a stepwise regression analysis of the Werner blood chemistry data (Jöreskog and Sörbom 1988, p. 111) can be performed as follows:

```
proc calis data=dixon method=gls nobs=180 print mod;
   lineqs y=0 x1+0 x2+0 x3+0 x4+0 x5+0 x6+0 x7+e;
   std    e=var;
run;
proc calis data=dixon method=gls nobs=180 print mod;
   lineqs y=g1 x1+0 x2+0 x3+0 x4+0 x5+0 x6+0 x7+e;
   std    e=var;
run;
proc calis data=dixon method=gls nobs=180 print mod;
   lineqs y=g1 x1+0 x2+0 x3+0 x4+0 x5+g6 x6+0 x7+e;
   std    e=var;
run;
```

Using the COSAN statement does not automatically delete those variables from the analysis that are not used in the model. You can use the output of the predetermined values in the predicted model matrix (PREDET option) to detect unused variables. Variables that are not used in the model are indicated by 0 in the rows and columns of the predetermined predicted model matrix.

Exogenous Manifest Variables

If there are exogenous manifest variables in the linear structural equation model, then there is a one-to-one relationship between the given covariances and corresponding estimates in the central model matrix (**P** or **Φ**). In general, using exogenous manifest variables reduces the degrees of freedom since the corresponding sample correlations or covariances are not part of the exogenous information provided for the parameter estimation. See the section "Counting the Degrees of Freedom" on page 956 for more information.

If you specify a RAM or LINEQS model statement, or if such a model is recognized in an INRAM= data set, those elements in the central model matrices that correspond to the exogenous manifest variables are reset to the sample values after computing covariances or correlations within the current BY group.

The COSAN statement does not automatically set the covariances in the central model matrices that correspond to manifest exogenous variables.

You can use the output of the predetermined values in the predicted model matrix (PREDET option) that correspond to manifest exogenous variables to see which of the manifest variables are exogenous variables and to help you set the corresponding locations of the central model matrices with their covariances.

The following two examples show how different the results of PROC CALIS can be if manifest variables are considered as either endogenous or exogenous variables. (See Figure 25.5.) In both examples, a correlation matrix **S** is tested against an identity model matrix **C**; that is, no parameter is estimated. The following three runs of the first example (specified by the COSAN, LINEQS, and RAM statements) consider the two variables y and x as endogenous variables:

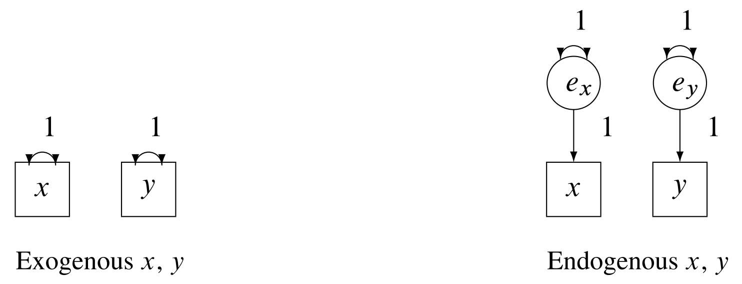
```
title2 'Data: FULLER (1987, p.18)';
data corn;
   input y x;
   datalines;
 86  70
115  97
 90  53
 86  64
110  95
 91  64
 99  50
 96  70
 99  94
104  69
 96  51
;
```

```
title3 'Endogenous Y and X';
proc calis data=corn;
   cosan corr(2,ide);
run;
proc calis data=corn;
   lineqs
          y=ey,
          x=ex;
   std    ey ex=2 * 1;
run;
proc calis data=corn;
   ram
       1  1  3  1.,
       1  2  4  1.,
       2  3  3  1.,
       2  4  4  1.;
run;
```

The following two runs of the second example (specified by the LINEQS and RAM statements) consider y and x as exogenous variables:

```
title3 'Exogenous Y and X';
proc calis data=corn;
   std y x=2 * 1;
run;
proc calis data=corn;
   ram
       2  1  1  1.,
       2  2  2  1.;
run;
```

Figure 25.5 Exogenous and Endogenous Variables



The LINEQS and the RAM model statements set the covariances (correlations) of exogenous manifest variables in the estimated model matrix and automatically reduce the degrees of freedom.

Use of Optimization Techniques

No algorithm for optimizing general nonlinear functions exists that will always find the global optimum for a general nonlinear minimization problem in a reasonable amount of time. Since no single optimization technique is invariably superior to others, PROC CALIS provides a variety of optimization techniques that work well in various circumstances. However, you can devise problems for which none of the techniques in PROC CALIS will find the correct solution. All optimization techniques in PROC CALIS use $O(n^2)$ memory except the conjugate gradient methods, which use only $O(n)$ of memory and are designed to optimize problems with many parameters.

The PROC CALIS statement NLOPTIONS can be especially helpful for tuning applications with nonlinear equality and inequality constraints on the parameter estimates. Some of the options available in NLOPTIONS might also be invoked as PROC CALIS options. The NLOPTIONS statement can specify almost the same options as the SAS/OR NLP procedure.

Nonlinear optimization requires the repeated computation of the following:

- the function value (optimization criterion)
- the gradient vector (first-order partial derivatives)
- for some techniques, the (approximate) Hessian matrix (second-order partial derivatives)
- values of linear and nonlinear constraints
- the first-order partial derivatives (Jacobian) of nonlinear constraints

For the criteria used by PROC CALIS, computing the gradient takes more computer time than computing the function value, and computing the Hessian takes *much* more computer time and memory than computing the gradient, especially when there are many parameters to estimate. Unfortunately, optimization techniques that do not use the Hessian usually require many more iterations than techniques that do use the (approximate) Hessian, and so they are often slower. Techniques that do not use the Hessian also tend to be less reliable (for example, they might terminate at local rather than global optima).

The available optimization techniques are displayed in Table 25.13 and can be chosen by the TECH=*name* option.

Table 25.13 Optimization Techniques

TECH=	**Optimization Technique**
LEVMAR	Levenberg-Marquardt method
TRUREG	Trust-region method
NEWRAP	Newton-Raphson method with line search
NRRIDG	Newton-Raphson method with ridging
QUANEW	Quasi-Newton methods (DBFGS, DDFP, BFGS, DFP)
DBLDOG	Double-dogleg method (DBFGS, DDFP)
CONGRA	Conjugate gradient methods (PB, FR, PR, CD)

Table 25.14 shows, for each optimization technique, which derivatives are needed (first-order or second-order) and what kind of constraints (boundary, linear, or nonlinear) can be imposed on the parameters.

Table 25.14 Derivatives Needed and Constraints Allowed

	Derivatives		**Constraints**		
TECH=	**First Order**	**Second Order**	**Boundary**	**Linear**	**Nonlinear**
LEVMAR	x	x	x	x	-
TRUREG	x	x	x	x	-
NEWRAP	x	x	x	x	-
NRRIDG	x	x	x	x	-
QUANEW	x	-	x	x	x
DBLDOG	x	-	x	x	-
CONGRA	x	-	x	x	-

The Levenberg-Marquardt, trust-region, and Newton-Raphson techniques are usually the most reliable, work well with boundary and general linear constraints, and generally converge after a few iterations to a precise solution. However, these techniques need to compute a Hessian matrix in each iteration. For HESSALG=1, this means that you need about $4(n(n+1)/2)t$ bytes of work memory (n = the number of manifest variables, t = the number of parameters to estimate) to store the Jacobian and its crossproduct. With HESSALG=2 or HESSALG=3, you do not need this work memory, but the use of a utility file increases execution time. Computing the approximate Hessian in each iteration can be very time- and memory-consuming, especially for large problems (more than 60 or 100 parameters, depending on the computer used). For large problems, a quasi-Newton technique, especially with the BFGS update, can be far more efficient.

For a poor choice of initial values, the Levenberg-Marquardt method seems to be more reliable.

If memory problems occur, you can use one of the conjugate gradient techniques, but they are generally slower and less reliable than the methods that use second-order information.

There are several options to control the optimization process. First of all, you can specify various termination criteria. You can specify the GCONV= option to specify a relative gradient termination criterion. If there are active boundary constraints, only those gradient components that correspond to inactive constraints contribute to the criterion. When you want very precise parameter estimates, the GCONV= option is useful. Other criteria that use relative changes in function values or parameter estimates in consecutive iterations can lead to early termination when active constraints cause small steps to occur. The small default value for the FCONV= option helps prevent early termination. Using the MAXITER= and MAXFUNC= options enables you to specify the maximum number of iterations and function calls in the optimization process. These limits are especially useful in combination with the INRAM= and OUTRAM= options; you can run a few iterations at a time, inspect the results, and decide whether to continue iterating.

Nonlinearly Constrained QN Optimization

The algorithm used for nonlinearly constrained quasi-Newton optimization is an efficient modification of Powell's (1978a, 1978b, 1982a, 1982b) *Variable Metric Constrained WatchDog* (VMCWD) algorithm. A similar but older algorithm (VF02AD) is part of the Harwell library. Both VMCWD and VF02AD use Fletcher's VE02AD algorithm (also part of the Harwell library) for positive-definite quadratic programming. The PROC CALIS QUANEW implementation uses a quadratic programming subroutine that updates and downdates the approximation of the Cholesky factor when the active set changes. The nonlinear QUANEW algorithm is not a feasible-point algorithm, and the value of the objective function need not decrease (minimization) or increase (maximization) monotonically. Instead, the algorithm tries to reduce a linear combination of the objective function and constraint violations, called the *merit function*.

The following are similarities and differences between this algorithm and VMCWD:

- A modification of this algorithm can be performed by specifying VERSION=1, which replaces the update of the Lagrange vector μ with the original update of Powell (1978a, 1978b), which is used in VF02AD. This can be helpful for some applications with linearly dependent active constraints.
- If the VERSION= option is not specified or VERSION=2 is specified, the evaluation of the Lagrange vector μ is performed in the same way as Powell (1982a, 1982b) describes.
- Instead of updating an approximate Hessian matrix, this algorithm uses the dual BFGS (or DFP) update that updates the Cholesky factor of an approximate Hessian. If the condition of the updated matrix gets too bad, a restart is done with a positive diagonal matrix. At the end of the first iteration after each restart, the Cholesky factor is scaled.
- The Cholesky factor is loaded into the quadratic programming subroutine, automatically ensuring positive-definiteness of the problem. During the quadratic programming step, the Cholesky factor of the projected Hessian matrix $\mathbf{Z}_k'\mathbf{G}\mathbf{Z}_k$ and the QT decomposition are updated simultaneously when the active set changes. Refer to Gill et al. (1984) for more information.
- The line-search strategy is very similar to that of Powell (1982a, 1982b). However, this algorithm does not call for derivatives during the line search; hence, it generally needs fewer derivative calls than function calls. The VMCWD algorithm always requires the same number of derivative and function calls. It was also found in several applications of VMCWD that Powell's line-search method sometimes uses steps that are too long during the first iterations. In those cases, you can use the INSTEP= option specification to restrict the step length α of the first iterations.
- Also the watchdog strategy is similar to that of Powell (1982a, 1982b). However, this algorithm does not return automatically after a fixed number of iterations to a former better point. A return here is further delayed if the observed function reduction is close to the expected function reduction of the quadratic model.
- Although Powell's termination criterion still is used (as FCONV2), the QUANEW implementation uses two additional termination criteria (GCONV and ABSGCONV).

This algorithm is automatically invoked when you specify the NLINCON statement. The nonlinear QUANEW algorithm needs the Jacobian matrix of the first-order derivatives (constraints normals) of the constraints

$$(\nabla c_i) = (\frac{\partial c_i}{\partial x_j}), \quad i = 1, \ldots, nc, j = 1, \ldots, n$$

where nc is the number of nonlinear constraints for a given point x.

You can specify two update formulas with the UPDATE= option:

- UPDATE=DBFGS performs the dual BFGS update of the Cholesky factor of the Hessian matrix. This is the default.
- UPDATE=DDFP performs the dual DFP update of the Cholesky factor of the Hessian matrix.

This algorithm uses its own line-search technique. All options and parameters (except the INSTEP= option) controlling the line search in the other algorithms do not apply here. In several applications, large steps in the first iterations are troublesome. You can specify the INSTEP= option to impose an upper bound for the step size α during the first five iterations. The values of the LCSINGULAR=, LCEPSILON=, and LCDEACT= options, which control the processing of linear and boundary constraints, are valid only for the quadratic programming subroutine used in each iteration of the nonlinear constraints QUANEW algorithm.

Optimization and Iteration History

The optimization and iteration histories are displayed by default because it is important to check for possible convergence problems.

The optimization history includes the following summary of information about the initial state of the optimization:

- the number of constraints that are active at the starting point, or more precisely, the number of constraints that are currently members of the working set. If this number is followed by a plus sign, there are more active constraints, of which at least one is temporarily released from the working set due to negative Lagrange multipliers.
- the value of the objective function at the starting point
- if the (projected) gradient is available, the value of the largest absolute (projected) gradient element
- for the TRUREG and LEVMAR subroutines, the initial radius of the trust-region around the starting point

The optimization history ends with some information concerning the optimization result:

- the number of constraints that are active at the final point, or more precisely, the number of constraints that are currently members of the working set. If this number is followed by a plus sign, there are more active constraints, of which at least one is temporarily released from the working set due to negative Lagrange multipliers.

- the value of the objective function at the final point
- if the (projected) gradient is available, the value of the largest absolute (projected) gradient element
- other information specific to the optimization technique

The iteration history generally consists of one line of displayed output containing the most important information for each iteration. The _LIST_ variable (see the section "SAS Programming Statements" on page 903) also enables you to display the parameter estimates and the gradient in some or all iterations.

The iteration history always includes the following (the words in parentheses are the column header output):

- the iteration number (Iter)
- the number of iteration restarts (rest)
- the number of function calls (nfun)
- the number of active constraints (act)
- the value of the optimization criterion (optcrit)
- the difference between adjacent function values (difcrit)
- the maximum of the absolute gradient components corresponding to inactive boundary constraints (maxgrad)

An apostrophe trailing the number of active constraints indicates that at least one of the active constraints is released from the active set due to a significant Lagrange multiplier.

For the Levenberg-Marquardt technique (LEVMAR), the iteration history also includes the following information:

- An asterisk trailing the iteration number means that the computed Hessian approximation is singular and consequently ridged with a positive lambda value. If all or the last several iterations show a singular Hessian approximation, the problem is not sufficiently identified. Thus, there are other locally optimal solutions that lead to the same optimum function value for different parameter values. This implies that standard errors for the parameter estimates are not computable without the addition of further constraints.
- the value of the Lagrange multiplier (lambda); this is 0 if the optimum of the quadratic function approximation is inside the trust-region (a trust-region-scaled Newton step can be performed) and is greater than 0 when the optimum of the quadratic function approximation is located at the boundary of the trust-region (the scaled Newton step is too long to fit in the trust-region and a quadratic constraint optimization is performed). Large values indicate optimization difficulties. For a nonsingular Hessian matrix, the value of lambda should go to 0 during the last iterations, indicating that the objective function can be well approximated by a quadratic function in a small neighborhood of the optimum point. An increasing lambda value often indicates problems in the optimization process.

- the value of the ratio ρ (rho) between the actually achieved difference in function values and the predicted difference in the function values on the basis of the quadratic function approximation. Values much less than 1 indicate optimization difficulties. The value of the ratio ρ indicates the goodness of the quadratic function approximation; in other words, $\rho << 1$ means that the radius of the trust-region has to be reduced. A fairly large value of ρ means that the radius of the trust region need not be changed. And a value close to or larger than 1 means that the radius can be increased, indicating a good quadratic function approximation.

For the Newton-Raphson technique (NRRIDG), the iteration history also includes the following information:

- the value of the ridge parameter. This is 0 when a Newton step can be performed, and it is greater than 0 when either the Hessian approximation is singular or a Newton step fails to reduce the optimization criterion. Large values indicate optimization difficulties.
- the value of the ratio ρ (rho) between the actually achieved difference in function values and the predicted difference in the function values on the basis of the quadratic function approximation. Values much less than 1.0 indicate optimization difficulties.

For the Newton-Raphson with line-search technique (NEWRAP), the iteration history also includes the following information:

- the step size α (alpha) computed with one of the line-search algorithms
- the slope of the search direction at the current parameter iterate. For minimization, this value should be significantly negative. Otherwise, the line-search algorithm has difficulty reducing the function value sufficiently.

For the trust-region technique (TRUREG), the iteration history also includes the following information:

- An asterisk after the iteration number means that the computed Hessian approximation is singular and consequently ridged with a positive lambda value.
- the value of the Lagrange multiplier (lambda). This value is zero when the optimum of the quadratic function approximation is inside the trust-region (a trust-region-scaled Newton step can be performed) and is greater than zero when the optimum of the quadratic function approximation is located at the boundary of the trust-region (the scaled Newton step is too long to fit in the trust-region and a quadratically constrained optimization is performed). Large values indicate optimization difficulties. As in Gay (1983), a negative lambda value indicates the special case of an indefinite Hessian matrix (the smallest eigenvalue is negative in minimization).
- the value of the radius Δ of the trust-region. Small trust-region radius values combined with large lambda values in subsequent iterations indicate optimization problems.

For the quasi-Newton (QUANEW) and conjugate gradient (CONGRA) techniques, the iteration history also includes the following information:

- the step size (alpha) computed with one of the line-search algorithms
- the descent of the search direction at the current parameter iterate. This value should be significantly smaller than 0. Otherwise, the line-search algorithm has difficulty reducing the function value sufficiently.

Frequent update restarts (rest) of a quasi-Newton algorithm often indicate numerical problems related to required properties of the approximate Hessian update, and they decrease the speed of convergence. This can happen particularly if the ABSGCONV= termination criterion is too small—that is, when the requested precision cannot be obtained by quasi-Newton optimization. Generally, the number of automatic restarts used by conjugate gradient methods is much higher.

For the nonlinearly constrained quasi-Newton technique, the iteration history also includes the following information:

- the maximum value of all constraint violations,

$$\text{conmax} = \max(|c_i(x)| : c_i(x) < 0)$$

- the value of the predicted function reduction used with the GCONV and FCONV2 termination criteria,

$$\text{pred} = |g(x^{(k)})s(x^{(k)})| + \sum_{i=1}^{m} |\lambda_i c_i(x^{(k)})|$$

- the step size α of the quasi-Newton step. Note that this algorithm works with a special line-search algorithm.
- the maximum element of the gradient of the Lagrange function,

$$\begin{aligned} \text{lfgmax} &= \nabla_x L(x^{(k)}, \lambda^{(k)}) \\ &= \nabla_x f(x^{(k)}) - \sum_{i=1}^{m} \lambda_i^{(k)} \nabla_x c_i(x^{(k)}) \end{aligned}$$

For the double-dogleg technique, the iteration history also includes the following information:

- the parameter λ of the double-dogleg step. A value $\lambda = 0$ corresponds to the full (quasi) Newton step.
- the slope of the search direction at the current parameter iterate. For minimization, this value should be significantly negative.

Line-Search Methods

In each iteration k, the (dual) quasi-Newton, hybrid quasi-Newton, conjugate gradient, and Newton-Raphson minimization techniques use iterative line-search algorithms that try to optimize a linear, quadratic, or cubic approximation of the nonlinear objective function f of n parameters x along a feasible descent search direction $s^{(k)}$

$$f(x^{(k+1)}) = f(x^{(k)} + \alpha^{(k)} s^{(k)})$$

by computing an approximately optimal scalar $\alpha^{(k)} > 0$. Since the outside iteration process is based only on the approximation of the objective function, the inside iteration of the line-search algorithm does not have to be perfect. Usually, it is satisfactory that the choice of α significantly reduces (in a minimization) the objective function. Criteria often used for termination of line-search algorithms are the Goldstein conditions (Fletcher 1987).

Various line-search algorithms can be selected by using the LIS= option. The line-search methods LIS=1, LIS=2, and LIS=3 satisfy the left-hand-side and right-hand-side Goldstein conditions (refer to Fletcher 1987). When derivatives are available, the line-search methods LIS=6, LIS=7, and LIS=8 try to satisfy the right-hand-side Goldstein condition; if derivatives are not available, these line-search algorithms use only function calls.

The line-search method LIS=2 seems to be superior when function evaluation consumes significantly less computation time than gradient evaluation. Therefore, LIS=2 is the default value for Newton-Raphson, (dual) quasi-Newton, and conjugate gradient optimizations.

Restricting the Step Length

Almost all line-search algorithms use iterative extrapolation techniques that can easily lead to feasible points where the objective function f is no longer defined (resulting in indefinite matrices for ML estimation) or is difficult to compute (resulting in floating-point overflows). Therefore, PROC CALIS provides options that restrict the step length or trust-region radius, especially during the first main iterations.

The inner product $g's$ of the gradient g and the search direction s is the slope of $f(\alpha) = f(x + \alpha s)$ along the search direction s with step length α. The default starting value $\alpha^{(0)} = \alpha^{(k,0)}$ in each line-search algorithm ($\min_{\alpha>0} f(x + \alpha s)$) during the main iteration k is computed in three steps.

1. Use either the difference $df = |f^{(k)} - f^{(k-1)}|$ of the function values during the last two consecutive iterations or the final step-size value α_- of the previous iteration $k-1$ to compute a first value $\alpha_1^{(0)}$.

   - Using the DAMPSTEP<=*r*> option:

     $$\alpha_1^{(0)} = \min(1, r\alpha_-)$$

     The initial value for the new step length can be no larger than r times the final step length α_- of the previous iteration. The default is $r = 2$.

- Not using the DAMPSTEP option:

$$\alpha_1^{(0)} = \begin{cases} step & \text{if } 0.1 \le step \le 10 \\ 10 & \text{if } step > 10 \\ 0.1 & \text{if } step < 0.1 \end{cases}$$

  with

$$step = \begin{cases} df/|g's| & \text{if } |g's| \ge \epsilon \max(100df, 1) \\ 1 & \text{otherwise} \end{cases}$$

  This value of $\alpha_1^{(0)}$ can be too large and can lead to a difficult or impossible function evaluation, especially for highly nonlinear functions such as the EXP function.

2. During the first five iterations, the second step enables you to reduce $\alpha_1^{(0)}$ to a smaller starting value $\alpha_2^{(0)}$ using the INSTEP=*r* option:

$$\alpha_2^{(0)} = \min(\alpha_1^{(0)}, r)$$

   After more than five iterations, $\alpha_2^{(0)}$ is set to $\alpha_1^{(0)}$.

3. The third step can further reduce the step length by

$$\alpha_3^{(0)} = \min(\alpha_2^{(0)}, \min(10, u))$$

   where u is the maximum length of a step inside the feasible region.

The INSTEP=*r* option lets you specify a smaller or larger radius of the trust region used in the first iteration by the trust-region, double-dogleg, and Levenberg-Marquardt algorithms. The default initial trust-region radius is the length of the scaled gradient (Moré 1978). This step corresponds to the default radius factor of $r = 1$. This choice is successful in most practical applications of the TRUREG, DBLDOG, and LEVMAR algorithms. However, for bad initial values used in the analysis of a covariance matrix with high variances, or for highly nonlinear constraints (such as using the EXP function) in your programming code, the default start radius can result in arithmetic overflows. If this happens, you can try decreasing values of INSTEP=*r*, $0 < r < 1$, until the iteration starts successfully. A small factor r also affects the trust-region radius of the next steps because the radius is changed in each iteration by a factor $0 < c \le 4$ depending on the ρ ratio. Reducing the radius corresponds to increasing the ridge parameter λ that produces smaller steps directed closer toward the gradient direction.

Modification Indices

While fitting structural models, you might want to modify the specified model in order to do the following:

- reduce the χ^2 value significantly
- reduce the number of parameters to estimate without increasing the χ^2 value too much

If you specify the MODIFICATION or MOD option, PROC CALIS computes and displays a default set of modification indices:

- **Univariate Lagrange multiplier test indices** for most elements in the model matrices that are constrained to *equal constants*. These are second-order approximations of the decrease in the χ^2 value that would result from allowing the constant matrix element to vary. Besides the value of the Lagrange multiplier, the corresponding probability ($df = 1$) and the approximate change of the parameter value (should the constant be changed to a parameter) are displayed. If allowing the constant to be a free estimated parameter would result in a singular information matrix, the string 'sing' is displayed instead of the Lagrange multiplier index. Not all elements in the model matrices should be allowed to vary; the diagonal elements of the inverse matrices in the RAM or LINEQS model must be constant ones. The univariate Lagrange multipliers are displayed at the constant locations of the model matrices.

- **Univariate Wald test indices** for those matrix elements that correspond to *parameter estimates* in the model. These are second-order approximations of the increase in the χ^2 value that would result from constraining the parameter to a 0 constant. The univariate Wald test indices are the same as the squared t values that are displayed together with the parameter estimates and standard errors. The univariate Wald test indices are displayed at the parameter locations of the model matrices.

- **Univariate Lagrange multiplier test indices** that are second-order approximations of the decrease in the χ^2 value that would result from the release of *equality constraints*. Multiple equality constraints containing $n > 2$ parameters are tested successively in n steps, each assuming the release of one of the equality-constrained parameters. The expected change of the parameter values of the separated parameter and the remaining parameter cluster are displayed, too.

- **Univariate Lagrange multiplier test indices** for releasing *active boundary constraints* specified by the BOUNDS statement

- **Stepwise multivariate Wald test indices** for constraining estimated parameters to 0 are computed and displayed. In each step, the parameter that would lead to the smallest increase in the multivariate χ^2 value is set to 0. Besides the multivariate χ^2 value and its probability, the univariate increments are also displayed. The process stops when the univariate probability is smaller than the specified value in the SLMW= option.

All of the preceding tests are approximations. You can often get more accurate tests by actually fitting different models and computing likelihood ratio tests. For more details about the Wald and the Lagrange multiplier test, refer to MacCallum (1986), Buse (1982), Bentler (1986), or Lee (1985).

Note that, for large model matrices, the computation time for the default modification indices can considerably exceed the time needed for the minimization process.

The modification indices are not computed for unweighted least squares or diagonally weighted least squares estimation.

CAUTION: Modification indices are not computed if the model matrix is an identity matrix (IDE or ZID), a selection matrix (PER), or the first matrix **J** in the LINEQS model. If you want to display the modification indices for such a matrix, you should specify the matrix as another type; for example, specify an identity matrix used in the COSAN statement as a diagonal matrix with constant diagonal elements of 1.

Constrained Estimation by Using Program Code

The CALIS procedure offers a very flexible way to constrain parameter estimates. You can use your own programming statements to express special properties of the parameter estimates. This tool is also present in McDonald's COSAN implementation but is considerably easier to use in the CALIS procedure. PROC CALIS is able to compute analytic first- and second-order derivatives that you would have to specify using the COSAN program. There are also three PROC CALIS statements you can use:

- the BOUNDS statement, to specify simple bounds on the parameters used in the optimization process
- the LINCON statement, to specify general linear equality and inequality constraints on the parameters used in the optimization process
- the NLINCON statement, to specify general nonlinear equality and inequality constraints on the parameters used in the optimization process. The variables listed in the NLINCON statement must be specified in the program code.

There are some traditional ways to enforce parameter constraints by using parameter transformations (McDonald 1980).

- **One-sided boundary constraints:** For example, the parameter q_k should be at least as large (or at most as small) as a given constant value a (or b),

  $$q_k \geq a \qquad \text{or} \qquad q_k \leq b$$

  This inequality constraint can be expressed as an equality constraint

  $$q_k = a + x_j^2 \qquad \text{or} \qquad q_k = b - x_j^2$$

  in which the fundamental parameter x_j is unconstrained.

- **Two-sided boundary constraints:** For example, the parameter q_k should be located between two given constant values a and b, $a < b$,

  $$a \leq q_k \leq b$$

  This inequality constraint can be expressed as an equality constraint

  $$q_k = a + (b - a)\frac{exp(x_j)}{1 + exp(x_j)}$$

  in which the fundamental parameter x_j is unconstrained.

- **One-sided order constraints:** For example, the parameters $q_1, \ldots, q_k$ should be ordered in the form

  $$q_1 \leq q_2, \quad q_1 \leq q_3, \quad \ldots, \quad q_1 \leq q_k$$

  These inequality constraints can be expressed as a set of equality constraints

  $$q_1 = x_1, \quad q_2 = x_1 + x_2^2, \quad \ldots, \quad q_k = x_1 + x_k^2$$

  in which the fundamental parameters $x_1, \ldots, x_k$ are unconstrained.

- **Two-sided order constraints:** For example, the parameters $q_1, \ldots, q_k$ should be ordered in the form

  $$q_1 \leq q_2 \leq q_3 \leq \ldots \leq q_k$$

  These inequality constraints can be expressed as a set of equality constraints

  $$q_1 = x_1, \quad q_2 = q_1 + x_2^2, \quad \ldots, \quad q_k = q_{k-1} + x_k^2$$

  in which the fundamental parameters $x_1, \ldots, x_k$ are unconstrained.

- **Linear equation constraints:** For example, the parameters q_1, q_2, q_3 should be linearly constrained in the form

  $$q_1 + q_2 + q_3 = a$$

  which can be expressed in the form of three explicit equations in which the fundamental parameters x_1 and x_2 are unconstrained:

  $$q_1 = x_1, \quad q_2 = x_2, \quad q_3 = a - x_1 - x_2$$

Refer to McDonald (1980) and Browne (1982) for further notes on reparameterizing techniques. If the optimization problem is not too large to apply the Levenberg-Marquardt or Newton-Raphson algorithm, boundary constraints should be requested by the BOUNDS statement rather than by reparameterizing code. If the problem is so large that you must use a quasi-Newton or conjugate gradient algorithm, reparameterizing techniques might be more efficient than the BOUNDS statement.

Counting the Degrees of Freedom

In a regression problem, the number of degrees of freedom for the error estimate is the number of observations in the data set minus the number of parameters. The NOBS=, DFR= (RDF=), and DFE= (EDF=) options refer to degrees of freedom in this sense. However, these values are not related to the degrees of freedom of a test statistic used in a covariance or correlation structure analysis. The NOBS=, DFR=, and DFE= options should be used in PROC CALIS to specify only the effective number of observations in the input DATA= data set.

In general, the number of degrees of freedom in a covariance or correlation structure analysis is defined as the difference between the number of nonredundant values q in the observed $n \times n$

correlation or covariance matrix **S** and the number t of free parameters **X** used in the fit of the specified model, $df = q - t$. Both values, q and t, are counted differently in different situations by PROC CALIS.

The number of nonredundant values q is generally equal to the number of lower triangular elements in the $n \times n$ moment matrix **S** including all diagonal elements, minus a constant c dependent upon special circumstances,

$$q = n(n+1)/2 - c$$

The number c is evaluated by adding the following quantities:

- If you specify a linear structural equation model containing exogenous manifest variables by using the RAM or LINEQS statement, PROC CALIS adds to c the number of variances and covariances among these manifest exogenous variables, which are automatically set in the corresponding locations of the central model matrices (see the section "Exogenous Manifest Variables" on page 943).
- If you specify the DFREDUCE=i option, PROC CALIS adds the specified number i to c. The number i can be a negative integer.
- If you specify the NODIAG option to exclude the fit of the diagonal elements of the data matrix **S**, PROC CALIS adds the number n of diagonal elements to c.
- If all the following conditions hold, then PROC CALIS adds to c the number of the diagonal locations:
  - NODIAG and DFREDUC= options are not specified.
  - A correlation structure is being fitted.
  - The predicted correlation matrix contains constants on the diagonal.

In some complicated models, especially those use programming statements, PROC CALIS might not be able to detect all the constant predicted values. In such cases, you must specify the DFREDUCE= option to get the correct degrees of freedom.

The number t is the number of different parameter names used in constructing the model if you do not use programming statements to impose constraints on the parameters. Using programming statements in general introduces two kinds of parameters:

- independent parameters, which are used only on the right-hand side of the expressions
- dependent parameters, which are used at least once on the left-hand side of the expressions

The independent parameters belong to the parameters involved in the estimation process, whereas the dependent parameters are fully defined by the programming statements and can be computed from the independent parameters. In this case, the number t is the number of different parameter names used in the model specification, but not used in the programming statements, plus the number of independent parameters. The independent parameters and their initial values can be defined in a model specification statement or in a PARMS statement.

The degrees of freedom are automatically increased by the number of active constraints in the solution. Similarly, the number of parameters are decreased by the number of active constraints. This affects the computation of many fit statistics and indices. Refer to Dijkstra (1992) for a discussion of the validity of statistical inferences with active boundary constraints. If the researcher believes that the active constraints will have a small chance of occurrence in repeated sampling, it might be more suitable to turn off the automatic adjustment by using the NOADJDF option.

Computational Problems

First Iteration Overflows

Analyzing a covariance matrix including high variances in the diagonal and using bad initial estimates for the parameters can easily lead to arithmetic overflows in the first iterations of the minimization algorithm. The line-search algorithms that work with cubic extrapolation are especially sensitive to arithmetic overflows. If this occurs with quasi-Newton or conjugate gradient minimization, you can specify the INSTEP= option to reduce the length of the first step. If an arithmetic overflow occurs in the first iteration of the Levenberg-Marquardt algorithm, you can specify the INSTEP= option to reduce the trust-region radius of the first iteration. You also can change the minimization technique or the line-search method. If none of these help, you should consider the following:

- scaling the covariance matrix
- providing better initial values
- changing the model

No Convergence of Minimization Process

If convergence does not occur during the minimization process, perform the following tasks.

- If there are *negative variance estimates* in the diagonal locations of the central model matrices, you can do the following:
  - Specify the BOUNDS statement to obtain nonnegative variance estimates.
  - Specify the HEYWOOD option, if the FACTOR model statement is specified.
- Change the estimation method to obtain a better set of initial estimates. For example, if you use METHOD=ML, you can do the following:
  - Change to METHOD=LSML.
  - Run some iterations with METHOD=DWLS or METHOD=GLS, write the results in an OUTRAM= data set, and use the results as initial values specified by an INRAM= data set in a second run with METHOD=ML.

- Change the optimization technique. For example, if you use the default TECH=LEVMAR, you can do the following:
  - Change to TECH=QUANEW or TECH=NEWRAP.
  - Run some iterations with TECH=CONGRA, write the results in an OUTRAM= data set, and use the results as initial values specified by an INRAM= data set in a second run with a different TECH= technique.
- Change or modify the update technique or the line-search algorithm, or both, when using TECH=QUANEW or TECH=CONGRA. For example, if you use the default update formula and the default line-search algorithm, you can do the following:
  - Change the update formula with the UPDATE= option.
  - Change the line-search algorithm with the LIS= option.
  - Specify a more precise line search with the LSPRECISION= option, if you use LIS=2 or LIS=3.
- Add more iterations and function calls by using the MAXIT= and MAXFU= options.
- Change the initial values. For many categories of model specifications done by the LINEQS, RAM, or FACTOR model, PROC CALIS computes an appropriate set of initial values automatically. However, for some of the model specifications (for example, structural equations with latent variables on the left-hand side and manifest variables on the right-hand side), PROC CALIS can generate very obscure initial values. In these cases, you have to set the initial values yourself.
  - Increase the initial values of the parameters located at the diagonal of central matrices in one of the following ways:
    - manually, by setting the values in the model specification
    - automatically, by using the DEMPHAS= option
  - Use a slightly different, but more stable, model to obtain preliminary estimates.
  - Use additional information to specify initial values, such as by using other SAS software like the FACTOR, REG, SYSLIN, and MODEL (SYSNLIN) procedures for the modified, unrestricted model case.

Unidentified Model

The parameter vector $\mathbf{x}$ in the covariance structure model

$$\mathbf{C} = \mathbf{C}(\mathbf{x})$$

is said to be identified in a parameter space G, if

$$\mathbf{C}(\mathbf{x}) = \mathbf{C}(\tilde{\mathbf{x}}), \quad \tilde{\mathbf{x}} \in G$$

implies $\mathbf{x} = \tilde{\mathbf{x}}$. The parameter estimates that result from an unidentified model can be very far from the parameter estimates of a very similar but identified model. They are usually machine dependent. Do not use parameter estimates of an unidentified model as initial values for another run of PROC CALIS.

Singular Predicted Model Matrix

You can easily specify models with singular predicted model matrices, such as by fixing diagonal elements of central matrices to 0. In such cases, you cannot compute maximum likelihood estimates (the ML function value F is not defined). Since singular predicted model matrices can also occur temporarily in the minimization process, PROC CALIS tries in such cases to change the parameter estimates so that the predicted model matrix becomes positive-definite. In such cases, the following message is displayed:

```
NOTE: Parameter set changed.
```

This process does not always work well, especially if there are fixed instead of variable diagonal elements in the central model matrices. A famous example where you cannot compute ML estimates is a component analysis with fewer components than given manifest variables. See the section "FACTOR Model Statement" on page 877 for more details. If you continue to get a singular predicted model matrix after changing initial values and optimization techniques, then your model is perhaps specified so that ML estimates cannot be computed.

Saving Computing Time

For large models, the most computing time is needed to compute the modification indices. If you do not really need the Lagrange multipliers or multiple Wald test indices (the univariate Wald test indices are the same as the t values), using the NOMOD option can save a considerable amount of computing time.

Central Matrices with Negative Eigenvalues

A covariance matrix cannot have negative eigenvalues, since a negative eigenvalue means that some linear combination of the variables has negative variance. PROC CALIS displays a warning if a central model matrix has negative eigenvalues but does not actually compute the eigenvalues. Sometimes this warning can be triggered by 0 or very small positive eigenvalues that appear negative because of numerical error. If you want to be sure that the central model matrix you are fitting can be considered to be a variance-covariance matrix, you can use the SAS/IML command *VAL=EIGVAL(U)* to compute the vector *VAL* of eigenvalues of matrix **U**.

Negative R^2 Values

The estimated squared multiple correlations R^2 of the endogenous variables are computed using the estimated error variances

$$R_i^2 = 1 - \frac{\widehat{var(\zeta_i)}}{\widehat{var(\eta_i)}}$$

If the model is a poor fit, it is possible that $\widehat{var(\zeta_i)} > \widehat{var(\eta_i)}$, which results in $R_i^2 < 0$.

Displayed Output

The output displayed by PROC CALIS depends on the statement used to specify the model. Since an analysis requested by the LINEQS or RAM statement implies the analysis of a structural equation model, more statistics can be computed and displayed than for a covariance structure analysis following the generalized COSAN model requested by the COSAN statement. The displayed output resulting from use of the FACTOR statement includes all the COSAN displayed output as well as more statistics displayed only when you specify the FACTOR statement. Since the displayed output by using the RAM statement differs only in its form from that generated by the LINEQS statement, in this section distinctions are made between COSAN and LINEQS output only.

The unweighted least squares and diagonally weighted least squares estimation methods do not provide a sufficient statistical basis to provide the following output (neither displayed nor written to an OUTEST= data set):

- most of the fit indices
- approximate standard errors
- normalized or asymptotically standardized residuals
- modification indices
- information matrix
- covariance matrix of parameter estimates

The notation $\mathbf{S} = (s_{ij})$ is used for the analyzed covariance or correlation matrix, $\mathbf{C} = (c_{ij})$ for the predicted model matrix, $\mathbf{W}$ for the weight matrix (for example, $\mathbf{W} = \mathbf{I}$ for ULS, $\mathbf{W} = \mathbf{S}$ for GLS, $\mathbf{W} = \mathbf{C}$ for ML estimates), $\mathbf{X}$ for the vector of optimal parameter estimates, n for the number of manifest variables, t for the number of parameter estimates, and N for the sample size.

The output of PROC CALIS includes the following:

- COSAN and LINEQS: List of the matrices and their properties specified by the generalized COSAN model if you specify at least the PSHORT option.
- LINEQS: List of manifest variables that are not used in the specified model and that are automatically omitted from the analysis. Note that there is no automatic variable reduction with the COSAN or FACTOR statement. If necessary, you should use the VAR statement in these cases.
- LINEQS: List of the endogenous and exogenous variables specified by the LINEQS, STD, and COV statements if you specify at least the PSHORT option.
- COSAN: Initial values of the parameter matrices indicating positions of constants and parameters. The output, or at least the default output, is displayed if you specify the PINITIAL option.

- LINEQS: The set of structural equations containing the initial values and indicating constants and parameters, and output of the initial error variances and covariances. The output, or at least the default output, is displayed if you specify the PINITIAL option.
- COSAN and LINEQS: The weight matrix **W** is displayed if GLS, WLS, or DWLS estimation is used and you specify the PWEIGHT or PALL option.
- COSAN and LINEQS: General information about the estimation problem: number of observations (N), number of manifest variables (n), amount of independent information in the data matrix (information, $n(n + 1)/2$), number of terms and matrices in the specified generalized COSAN model, and number of parameters to be estimated (parameters, t). If there are no exogenous manifest variables, the difference between the amount of independent information ($n(n + 1)/2$) and the number of requested estimates (t) is equal to the degrees of freedom (df). A necessary condition for a model to be identified is that the degrees of freedom are nonnegative. The output, or at least the default output, is displayed if you specify the SIMPLE option.
- COSAN and LINEQS: Mean and Std Dev (standard deviation) of each variable if you specify the SIMPLE option, as well as skewness and kurtosis if the DATA= data set is a raw data set and you specify the KURTOSIS option.
- COSAN and LINEQS: Various coefficients of multivariate kurtosis and the numbers of observations that contribute most to the normalized multivariate kurtosis if the DATA= data set is a raw data set and the KURTOSIS option, or you specify at least the PRINT option. See the section "Measures of Multivariate Kurtosis" on page 938 for more information.
- COSAN and LINEQS: Covariance or correlation matrix to be analyzed and the value of its determinant if you specify the output option PCORR or PALL. A 0 determinant indicates a singular data matrix. In this case, the generalized least squares estimates with default weight matrix **S** and maximum likelihood estimates cannot be computed.
- LINEQS: If exogenous manifest variables in the linear structural equation model are specified, then there is a one-to-one relationship between the given covariances and corresponding estimates in the central model matrix Φ or P. The output indicates which manifest variables are recognized as exogenous—that is, for which variables the entries in the central model matrix are set to fixed parameters. The output, or at least the default output, is displayed if you specify the PINITIAL option.
- COSAN and LINEQS: Vector of parameter names, initial values, and corresponding matrix locations, also indicating dependent parameter names used in your programming statements that are not allocated to matrix locations and have no influence on the fit function. The output, or at least the default output, is displayed if you specify the PINITIAL option.
- COSAN and LINEQS: The pattern of variable and constant elements of the predicted moment matrix that is predetermined by the analysis model is displayed if there are significant differences between constant elements in the predicted model matrix and the data matrix and you specify at least the PSHORT option. It is also displayed if you specify the PREDET option. The output indicates the differences between constant values in the predicted model matrix and the data matrix that is analyzed.
- COSAN and LINEQS: Special features of the optimization technique chosen if you specify at least the PSHORT option.

- COSAN and LINEQS: Optimization history if at least the PSHORT option is specified. For more details, see the section "Use of Optimization Techniques" on page 945.

- COSAN and LINEQS: Specific output requested by options in the NLOPTIONS statement; for example, parameter estimates, gradient, gradient of Lagrange function, constraints, Lagrange multipliers, projected gradient, Hessian, projected Hessian, Hessian of Lagrange function, Jacobian of nonlinear constraints.

- COSAN and LINEQS: The predicted model matrix and its determinant, if you specify the output option PCORR or PALL.

- COSAN and LINEQS: Residual and normalized residual matrix if you specify the RESIDUAL, or at least the PRINT option. The variance standardized or asymptotically standardized residual matrix can be displayed also. The average residual and the average off-diagonal residual are also displayed. See the section "Assessment of Fit" on page 928 for more details.

- COSAN and LINEQS: Rank order of the largest normalized residuals if you specify the RESIDUAL, or at least the PRINT option.

- COSAN and LINEQS: Bar chart of the normalized residuals if you specify the RESIDUAL, or at least the PRINT option.

- COSAN and LINEQS: Value of the fit function F. See the section "Estimation Criteria" on page 923 for more details. This output can be suppressed only by the NOPRINT option.

- COSAN and LINEQS: Goodness of fit index (GFI), adjusted goodness of fit index (AGFI), and root mean square residual (RMR) (Jöreskog and Sörbom 1985). See the section "Assessment of Fit" on page 928 for more details. This output can be suppressed only by the NOPRINT option.

- COSAN and LINEQS: Parsimonious goodness of fit index (PGFI) of Mulaik et al. (1989). See the section "Assessment of Fit" on page 928 for more detail. This output can be suppressed only by the NOPRINT option.

- COSAN and LINEQS: Overall χ^2, df, and Prob>Chi**2 if the METHOD= option is not ULS or DWLS. The χ^2 measure is the optimum function value F multiplied by $(N-1)$ if a CORR or COV matrix is analyzed or multiplied by N if a UCORR or UCOV matrix is analyzed; χ^2 measures the likelihood ratio test statistic for the null hypothesis that the predicted matrix $\mathbf{C}$ has the specified model structure against the alternative that $\mathbf{C}$ is unconstrained. The notation Prob>Chi**2 means "the probability under the null hypothesis of obtaining a greater χ^2 statistic than that observed." This output can be suppressed only by the NOPRINT option.

- COSAN and LINEQS: If METHOD= is not ULS or DWLS, the χ_0^2 value of the independence model and the corresponding degrees of freedom can be used (in large samples) to evaluate the gain of explanation by fitting the specific model (Bentler 1989). See the section "Assessment of Fit" on page 928 for more detail. This output can be suppressed only by the NOPRINT option.

- COSAN and LINEQS: If METHOD= is not ULS or DWLS, the value of the Steiger and Lind (1980) root mean squared error of approximation (RMSEA) coefficient and the lower and upper limits of the confidence interval. The size of the confidence interval is defined by the option ALPHARMS=α, $0 \leq \alpha \leq 1$. The default is $\alpha = 0.1$, which corresponds to a 90% confidence interval. See the section "Assessment of Fit" on page 928 for more detail. This output can be suppressed only by the NOPRINT option.

- COSAN and LINEQS: If the value of the METHOD= option is not ULS or DWLS, the value of the *probability of close fit* (Browne and Cudeck 1993). See the section "Assessment of Fit" on page 928 for more detail. This output can be suppressed only by the NOPRINT option.

- COSAN and LINEQS: If the value of the METHOD= option is not ULS or DWLS, the value of the Browne and Cudeck (1993) expected cross validation (ECVI) index and the lower and upper limits of the confidence interval. The size of the confidence interval is defined by the option ALPHAECV=α, $0 \leq \alpha \leq 1$. The default is $\alpha = 0.1$, which corresponds to a 90% confidence interval. See the section "Assessment of Fit" on page 928 for more detail. This output can be suppressed only by the NOPRINT option.

- COSAN and LINEQS: If the value of the METHOD= option is not ULS or DWLS, Bentler's (1989) comparative fit index. See the section "Assessment of Fit" on page 928 for more detail. This output can be suppressed only by the NOPRINT option.

- COSAN and LINEQS: If you specify METHOD=ML or METHOD=GLS, the χ^2 value and corresponding probability adjusted by the relative kurtosis coefficient η_2, which should be a close approximation of the χ^2 value for elliptically distributed data (Browne 1982). See the section "Assessment of Fit" on page 928 for more detail. This output can be suppressed only by the NOPRINT option.

- COSAN and LINEQS: The normal theory reweighted LS χ^2 value is displayed if METHOD= ML. Instead of the function value F_{ML}, the reweighted goodness of fit function F_{GWLS} is used. See the section "Assessment of Fit" on page 928 for more detail.

- COSAN and LINEQS: Akaike's information criterion if the value of the METHOD= option is not ULS or DWLS. See the section "Assessment of Fit" on page 928 for more detail. This output can be suppressed only by the NOPRINT option.

- COSAN and LINEQS: Bozdogan's (1987) consistent information criterion, CAIC. See the section "Assessment of Fit" on page 928 for more detail. This output can be suppressed only by the NOPRINT option.

- COSAN and LINEQS: Schwarz's Bayesian criterion (SBC) if the value of the METHOD= option is not ULS or DWLS (Schwarz 1978). See the section "Assessment of Fit" on page 928 for more detail. This output can be suppressed only by the NOPRINT option.

- COSAN and LINEQS: If the value of the METHOD= option is not ULS or DWLS, the following fit indices based on the overall χ^2 value are displayed:

  - McDonald's (McDonald 1989) measure of centrality
  - Parsimonious index of James, Mulaik, and Brett (1982)
  - Z-test of Wilson and Hilferty (1931)
  - Bentler and Bonett's (1980) nonnormed coefficient
  - Bentler and Bonett's (1980) normed coefficient
  - Bollen's (1986) normed index ρ_1
  - Bollen's (1989a) nonnormed index Δ_2

See the section "Assessment of Fit" on page 928 for more detail. This output can be suppressed only by the NOPRINT option.

- COSAN and LINEQS: Hoelter's (1983) critical N index is displayed (Bollen 1989b, p. 277). See the section "Assessment of Fit" on page 928 for more detail. This output can be suppressed only by the NOPRINT option.
- COSAN and LINEQS: Equations of linear dependencies among the parameters used in the model specification if the information matrix is recognized as singular at the final solution.
- COSAN: Model matrices containing the parameter estimates. Except for ULS or DWLS estimates, the approximate standard errors and t values are also displayed. This output is displayed if you specify the PESTIM option or at least the PSHORT option.
- LINEQS: Linear equations containing the parameter estimates. Except for ULS and DWLS estimates, the approximate standard errors and t values are also displayed. This output is displayed if you specify the PESTIM option, or at least the PSHORT option.
- LINEQS: Variances and covariances of the exogenous variables. This output is displayed if you specify the PESTIM option, or at least the PSHORT option.
- LINEQS: Linear equations containing the standardized parameter estimates. This output is displayed if you specify the PESTIM option, or at least the PSHORT option.
- LINEQS: Table of correlations among the exogenous variables. This output is displayed if you specify the PESTIM option, or at least the PSHORT option.
- LINEQS: Correlations among the exogenous variables. This output is displayed if you specify the PESTIM option, or at least the PSHORT option.
- LINEQS: Squared multiple correlations table, which displays the error variances of the endogenous variables. These are the diagonal elements of the predicted model matrix. Also displayed is the Total Variance and the R^2 values corresponding to all endogenous variables. See the section "Assessment of Fit" on page 928 for more detail. This output is displayed if you specify the PESTIM option, or at least the PSHORT option.
- LINEQS: If you specify the PDETERM or the PALL option, the total determination of all equations (DETAE), the total determination of the structural equations (DETSE), and the total determination of the manifest variables (DETMV) are displayed. See the section "Assessment of Fit" on page 928 for more detail. If one of the determinants in the formulas is 0, the corresponding coefficient is displayed as a missing value. If there are structural equations, PROC CALIS also displays the stability coefficient of reciprocal causation—that is, the largest eigenvalue of the $\mathbf{BB}'$ matrix, where $\mathbf{B}$ is the causal coefficient matrix of the structural equations.
- LINEQS: The matrix of estimated covariances among the latent variables if you specify the PLATCOV option, or at least the PRINT option.
- LINEQS: The matrix of estimated covariances between latent and manifest variables used in the model if you specify the PLATCOV option, or at least the PRINT option.

- LINEQS and FACTOR: The matrix **FSR** of latent variable scores regression coefficients if you specify the PLATCOV option, or at least the PRINT option. The **FSR** matrix is a generalization of Lawley and Maxwell's (1971, p. 109) factor scores regression matrix,

  $$\mathbf{FSR} = \mathbf{C}_{yx}\mathbf{C}_{xx}^{-1}$$

  where $\mathbf{C}_{xx}$ is the $n \times n$ predicted model matrix. (predicted covariances among manifest variables) and $\mathbf{C}_{yx}$ is the $n_{lat} \times n$ matrix of the predicted covariances between latent and manifest variables. You can multiply the manifest observations by this matrix to estimate the scores of the latent variables used in your model.

- LINEQS: The matrix **TEF** of total effects if you specify the TOTEFF option, or at least the PRINT option. For the LINEQS model, the matrix of total effects is

  $$\mathbf{TEF} = (\mathbf{I} - \boldsymbol{\beta})^{-1}\boldsymbol{\gamma} - (\mathbf{O} : \mathbf{I})$$

  (For the LISREL model, refer to Jöreskog and Sörbom 1985.) The matrix of indirect effects is displayed also.

- FACTOR: The matrix of rotated factor loadings and the orthogonal transformation matrix if you specify the ROTATE= and PESTIM options, or at least the PSHORT option.

- FACTOR: Standardized (rotated) factor loadings, variance estimates of endogenous variables, R^2 values, correlations among factors, and factor scores regression matrix, if you specify the PESTIM option, or at least the PSHORT option. The determination of manifest variables is displayed only if you specify the PDETERM option.

- COSAN and LINEQS: Univariate Lagrange multiplier and Wald test indices are displayed in matrix form if you specify the MODIFICATION (or MOD) or PALL option. Those matrix locations that correspond to constants in the model in general contain three values: the value of the Lagrange multiplier, the corresponding probability ($df = 1$), and the estimated change of the parameter value should the constant be changed to a parameter. If allowing the constant to be an estimated parameter would result in a singular information matrix, the string 'sing' is displayed instead of the Lagrange multiplier index. Those matrix locations that correspond to parameter estimates in the model contain the Wald test index and the name of the parameter in the model. See the section "Modification Indices" on page 953 for more detail.

- COSAN and LINEQS: Univariate Lagrange multiplier test indices for releasing equality constraints if you specify the MODIFICATION (or MOD) or PALL option. See the section "Modification Indices" on page 953 for more detail.

- COSAN and LINEQS: Univariate Lagrange multiplier test indices for releasing active boundary constraints specified by the BOUNDS statement if you specify the MODIFICATION (or MOD) or PALL option. See the section "Modification Indices" on page 953 for more detail.

- COSAN and LINEQS: If the MODIFICATION (or MOD) or PALL option is specified, the stepwise multivariate Wald test for constraining estimated parameters to zero constants is performed as long as the univariate probability is larger than the value specified in the PMW= option (default PMW=0.05). See the section "Modification Indices" on page 953 for more detail.

ODS Table Names

PROC CALIS 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 about ODS, see Chapter 20, "Using the Output Delivery System."

Table 25.15 ODS Tables Produced by PROC CALIS

ODS Table Name	Model[1]	Description	Option[2]
AddParms	C, F, L, R	Additional parameters in the PARAMETERS statement	PINITIAL or default
AsymStdRes	C, F, L, R	Asymptotically standardized residual matrix	RESIDUAL= or PRINT
AveAsymStdRes	C, F, L, R	Average absolute asymptotically standardized residuals	RESIDUAL= or PRINT
AveNormRes	C, F, L, R	Average absolute normalized residuals	RESIDUAL= or PRINT
AveRawRes	C, F, L, R	Average absolute raw residuals	RESIDUAL= or PRINT
AveVarStdRes	C, F, L, R	Average absolute variance standardized residuals	RESIDUAL= or PRINT
ContKurtosis	C, F, L, R	Contributions to kurtosis	KURTOSIS or PRINT
ConvergenceStatus	C, F, L, R	Convergence status	PSHORT
CorrExog	L	Correlations among exogenous variables	PESTIM or PSHORT
CorrParm	C, F, L, R	Correlations among parameter estimates	PCOVES and default
CovMat	C, F, L, R	Assorted covariance matrices	PCOVES and default
DependParms	C, F, L, R	Dependent parameters (if specified by programming statements)	PRIVEC and default
Determination	L, F, R	Coefficients of determination	PDETERM and default
DistAsymStdRes	C, F, L, R	Distribution of asymptotically standardized residuals	RESIDUAL= or PRINT
DistNormRes	C, F, L, R	Distribution of normalized residuals	RESIDUAL= or PRINT
DistVarStdRes	C, F, L, R	Distribution of variance standardized residuals	RESIDUAL= or PRINT
Eigenvalues	C, F, L, R	Eigenvalues of the approximate Hessian matrix	PCOVES[3] and default
EndogenousVar	L	Endogenous variables	PESTIM or PSHORT
EstCovExog	L	Estimated covariances among exogenous variables	PESTIM or PSHORT
Estimates	C, F, L, R	Vector of estimates	PRIVEC
EstLatentEq	L	Estimated latent variable equations	PESTIM or PSHORT
EstManifestEq	L	Estimated manifest variable equations	PESTIM or PSHORT
EstParms	C, F	Estimated parameter matrix	PESTIM or PSHORT
EstVarExog	L	Estimated variances of exogenous variables	PESTIM or PSHORT
ExogenousVar	L	List of exogenous variables	PESTIM or PSHORT
FactCorrExog	F	Correlations among factors	PESTIM or PSHORT

Table 25.15 *continued*

ODS Table Name	Model[1]	Description	Option[2]
FactScoreCoef	F	Factor score regression coefficients	PESTIM or PSHORT
Fit	C, F, L, R	Fit statistics	PSUMMARY
GenModInfo	C, F, L, R	General modeling information	PSIMPLE or default
Gradient	C, F, L, R	First partial derivatives (Gradient)	PRIVEC and default
InCorr	C, F, L, R	Input correlation matrix	PCORR or PALL
InCorrDet	C, F, L, R	Determinant of the input correlation matrix	PCORR or PALL
InCov	C, F, L, R	Input covariance matrix	PCORR or PALL
InCovDet	C, F, L, R	Determinant of the input covariance matrix	PCORR or PALL
InCovExog	L	Input covariances among exogenous variables	PESTIM or PSHORT
IndirectEffects	L, R	Indirect effects	TOTEFF or PRINT
Information	C, F, L, R	Information matrix	PCOVES and default
InitEstimates	C, F, L, R	Initial vector of parameter estimates	PINITIAL or default
InitParms	C, F	Initial matrix of parameter estimates	PINITIAL or default
InitParms	L, R	Initial matrix of parameter estimates	PRIMAT and default
InitRAMEstimates	R	Initial RAM estimates	PESTIM or PSHORT
InLatentEq	L	Input latent variable equations	PESTIM or PSHORT
InManifestEq	L	Input manifest variable equations	PESTIM or PSHORT
InSymmetric	C, F, L, R	Input symmetric matrix (SYMATRIX data type)	PCORR or PALL
InVarExog	L	Input variances of exogenous variables	PESTIM or PSHORT
IterHist	C, F, L, R	Iteration history	PSHORT
IterStart	C, F, L, R	Iteration start	PSHORT
IterStop	C, F, L, R	Iteration stop	PSHORT
Jacobian	C, F, L, R	Jacobi column pattern	PJACPAT
Kurtosis	C, F, L, R	Kurtosis, with raw data input	KURTOSIS or PRINT
LagrangeBoundary	C, F, L, R	Lagrange, releasing active boundary constraints	MODIFICATION[4] or PALL
LagrangeEquality	C, F, L, R	Lagrange, releasing equality constraints	MODIFICATION or PALL
LatentScoreCoef	L, R	Latent variable regression score coefficients	PLATCOV or PRINT
ModelStatement	C, F, L, R	Model summary	PSHORT
ModIndices	C, F, L, R	Lagrange multiplier and Wald test statistics	MODIFICATION or PALL
NormRes	C, F, L, R	Normalized residual matrix	RESIDUAL= or PRINT
PredetElements	C, F, L, R	Predetermined elements	PREDET or PALL
PredModel	C, F, L, R	Predicted model matrix	PCORR or PALL
PredModelDet	C, F, L, R	Predicted model determinant	PCORR or PALL
PredMomentLatent	L, R	Predicted latent variable moments	PLATCOV or PRINT
PredMomentManLat	L, R	Predicted manifest and latent variable moments	PLATCOV or PRINT
ProblemDescription	C, F, L, R	Problem description	PSHORT
RAMCorrExog	R	Correlations among exogenous variables	PESTIM or PSHORT

Table 25.15 *continued*

ODS Table Name	Model[1]	Description	Option[2]
RAMEstimates	R	RAM final estimates	PESTIM, or PSHORT
RAMStdEstimates	R	Standardized estimates	PESTIM, or PSHORT
RankAsymStdRes	C, F, L, R	Ranking of the largest asymptotically standardized residuals	RESIDUAL= or PRINT
RankLagrange	C, F, L, R	Ranking of the largest Lagrange indices	RESIDUAL= or PRINT
RankNormRes	C, F, L, R	Ranking of the largest normalized residuals	RESIDUAL= or PRINT
RankRawRes	C, F, L, R	Ranking of the largest raw residuals	RESIDUAL= or PRINT
RankVarStdRes	C, F, L, R	Ranking of the largest variance standardized residuals	RESIDUAL= or PRINT
RawRes	C, F, L, R	Raw residual matrix	RESIDUAL= or PRINT
RotatedLoadings	F	Rotated loadings, with ROTATE= option in FACTOR statement	PESTIM or PSHORT
Rotation	F	Rotation matrix, with ROTATE= option in FACTOR statement	PESTIM or PSHORT
SetCovExog	L, R	Set covariance parameters for manifest exogenous variables	PINITIAL or default
SimpleStatistics	C, F, L, R	Simple statistics, with raw data input	SIMPLE or default
SqMultCorr	F, L, R	Squared multiple correlations	PESTIM or PSHORT
Stability	L, R	Stability of reciprocal causation	PDETERM and default
StdErrs	C, F, L, R	Vector of standard errors	PRIVEC and default
StdLatentEq	L	Standardized latent variable equations	PESTIM or PSHORT
StdLoadings	F	Standardized factor loadings	PESTIM or PSHORT
StdManifestEq	L	Standardized manifest variable equations	PESTIM or PSHORT
StructEq	L, R	Variables in the structural equations	PDETERM and default
SumSqDif	C, F, L, R	Sum of squared differences of pre-determined elements	PREDET or PALL
TotalEffects	L, R	Total effects	TOTEFF or PRINT
tValues	C, F, L, R	Vector of *t* values	PRIVEC and default
VarSelection	L, R	Manifest variables, if not all are used, selected for modeling	default
VarStdRes	C, F, L, R	Variance standardized residual matrix	RESIDUAL= or PRINT
WaldTest	C, F, L, R	Wald test	MODIFICATION or PALL
Weights	C, F, L, R	Weight matrix	PWEIGHT[5] or PALL
WeightsDet	C, F, L, R	Determinant of the weight matrix	PWEIGHT[5] or PALL

1. Most CALIS output tables are specific to the model statement used. Keys: C: COSAN model, F: FACTOR model, L: LINEQS model, R: RAM model.
2. The printing options PALL, PRINT, "default," PSHORT, and PSUMM form hierarchical levels of output control, with PALL including all the output enabled by the options at the lower levels, and so on. The "default" option means that NOPRINT is not specified. Therefore, in the table, for example, if PSHORT is the printing option for an output, PALL, PRINT, or "default" will also enable the same output printing.
3. The eigenvalues are printed only when there are negative eigenvalues in the approximate Hessian matrix.
4. The printing of LagrangeBoundary is effective only if you have set some boundary constraints for parameters.
5. The printing of Weights or WeightsDet is effective only if your estimation method uses the weight matrix (for example, WLS or LSWLS).

ODS Graphics

To request graphics with PROC CALIS, you must first enable ODS Graphics by specifying the **ods graphics on** statement. See Chapter 21, "Statistical Graphics Using ODS," for more information. The names of the graphs that PROC CALIS generates are listed in Table 25.16, along with the required statements and options.

Table 25.16 ODS Graphics Produced by PROC CALIS

ODS Graph Name	Plot Description	PLOTS= Option
AsymStdResidualHistogram	Asymptotically standardized residuals	PLOTS=RESIDUALS and RESIDUAL=ASYMSTD; METHOD= is not ULS or DWLS
NormResidualHistogram	Normalized residuals	PLOTS=RESIDUALS and RESIDUAL=NORM
RawResidualHistogram	Raw residuals	PLOTS=RESIDUALS
VarStdResidualHistogram	Variance standardized residuals	PLOTS=RESIDUALS and RESIDUAL=VARSTD

Examples: CALIS Procedure

Example 25.1: Path Analysis: Stability of Alienation

The following covariance matrix from Wheaton et al. (1977) has served to illustrate the performance of several implementations for the analysis of structural equation models. Two different models have been analyzed by an early implementation of LISREL and are mentioned in Jöreskog (1978). You can also find a more detailed discussion of these models in the LISREL VI manual (Jöreskog and Sörbom 1985).

A slightly modified model for this covariance matrix is included in the EQS 2.0 manual (Bentler 1985, p. 28). The path diagram of this model is displayed in Figure 25.1. The same model is reanalyzed here by PROC CALIS. However, for the analysis with the EQS implementation, the last variable (V6) is rescaled by a factor of 0.1 to make the matrix less ill-conditioned. Since the Levenberg-Marquardt or Newton-Raphson optimization technique is used with PROC CALIS, rescaling the data matrix is not necessary and, therefore, is not done here. The results reported here reflect the estimates based on the original covariance matrix.

The DATA step and the CALIS model specification are shown as follows:

```
data Wheaton(TYPE=COV);
title "Stability of Alienation";
title2 "Data Matrix of WHEATON, MUTHEN, ALWIN & SUMMERS (1977)";
   _type_ = 'cov'; input _name_ $ v1-v6;
   label v1='Anomie (1967)' v2='Anomie (1971)' v3='Education'
         v4='Powerlessness (1967)' v5='Powerlessness (1971)'
         v6='Occupational Status Index';
   datalines;
v1   11.834      .         .         .        .        .
v2    6.947    9.364       .         .        .        .
v3    6.819    5.091    12.532       .        .        .
v4    4.783    5.028     7.495     9.986      .        .
v5   -3.839   -3.889    -3.841    -3.625   9.610       .
v6  -21.899  -18.831   -21.748   -18.775  35.522  450.288
;


ods graphics on;

proc calis cov data=Wheaton tech=nr edf=931 pall plots=residuals;
   lineqs
      v1 =         f1                  + e1,
      v2 =    .833 f1                  + e2,
      v3 =         f2                  + e3,
      v4 =    .833 f2                  + e4,
      v5 =         f3                  + e5,
      v6 = Lamb (.5) f3                + e6,
      f1 = Gam1(-.5) f3                + d1,
      f2 = Beta (.5) f1 + Gam2(-.5) f3 + d2;
   std
      e1-e6 = The1-The2 The1-The4 (6 * 3.),
      d1-d2 = Psi1-Psi2 (2 * 4.),
      f3    = Phi (6.) ;
   cov
      e1 e3 = The5 (.2),
      e4 e2 = The5 (.2);
run;

ods graphics off;
```

The COV option in the PROC CALIS statement requests the analysis of the covariance matrix. Without the COV option, the correlation matrix would be computed and analyzed. Since no METHOD= option has been used, maximum likelihood estimates are computed by default. The TECH=NR option requests the Newton-Raphson optimization method. The PALL option produces the almost complete set of displayed output, as displayed in Output 25.1.1 through Output 25.1.23. Note that, when you specify the PALL option, you can produce large amounts of output. The PALL option is used in this example to show how you can get a wide spectrum of useful information from PROC CALIS.

PROC CALIS can produce a high-quality residual histogram that is useful for showing the distribution of residuals. To request the residual histogram, you must first enable ODS Graphics by specifying the **ods graphics on** statement, as shown in the preceding code before the PROC CALIS statement. Then, the residual histogram is requested by the **plots=residuals** option in the PROC CALIS statement.

Output 25.1.1 displays the model specification in matrix terms, followed by the lists of endogenous and exogenous variables. Output 25.1.2 displays equations and initial parameter estimates. You can use these output to ensure that the desired model is being analyzed.

Output 25.1.1 Model and Variables

```
                         Stability of Alienation
               Data Matrix of WHEATON, MUTHEN, ALWIN & SUMMERS (1977)

                           The CALIS Procedure
             Covariance Structure Analysis: Pattern and Initial Values

                          LINEQS Model Statement


                    Matrix      Rows    Columns    ------Matrix Type-------

Term 1        1     _SEL_          6         17    SELECTION
              2     _BETA_        17         17    EQSBETA        IMINUSINV
              3     _GAMMA_       17          9    EQSGAMMA
              4     _PHI_          9          9    SYMMETRIC

                         The 8 Endogenous Variables

    Manifest        v1  v2  v3  v4  v5  v6
    Latent          f1  f2

                         The 9 Exogenous Variables

    Manifest
    Latent          f3
    Error           e1  e2  e3  e4  e5  e6  d1  d2
```

Output 25.1.2 Initial Model Specifications

```
                Manifest Variable Equations with Initial Estimates


                    v1      =   1.0000 f1   +  1.0000 e1
                    v2      =   0.8330 f1   +  1.0000 e2
                    v3      =   1.0000 f2   +  1.0000 e3
                    v4      =   0.8330 f2   +  1.0000 e4
                    v5      =   1.0000 f3   +  1.0000 e5
                    v6      =   0.5000*f3   +  1.0000 e6
                                       Lamb
```

Output 25.1.2 *continued*

```
                 Latent Variable Equations with Initial Estimates


              f1      =  -0.5000*f3    +  1.0000 d1
                                 Gam1
              f2      =   0.5000*f1    + -0.5000*f3    +  1.0000 d2
                                 Beta           Gam2

                          Variances of Exogenous Variables

                          Variable Parameter      Estimate

                          f3       Phi             6.00000
                          e1       The1            3.00000
                          e2       The2            3.00000
                          e3       The1            3.00000
                          e4       The2            3.00000
                          e5       The3            3.00000
                          e6       The4            3.00000
                          d1       Psi1            4.00000
                          d2       Psi2            4.00000

                       Covariances Among Exogenous Variables

                         Var1 Var2 Parameter      Estimate

                         e1   e3   The5            0.20000
                         e2   e4   The5            0.20000
```

General modeling information, descriptive statistics, and the input covariance matrix are displayed in Output 25.1.3. Because the input data set contains only the covariance matrix, the means of the manifest variables are assumed to be zero. Note that this has no impact on the estimation, unless a mean structure model is being analyzed.

Output 25.1.3 Modeling Information, Simple Statistics, and Input Covariance Matrix

```
              Observations          932     Model Terms             1
              Variables               6     Model Matrices          4
              Informations           21     Parameters             12

                      Variable                      Mean       Std Dev

           v1    Anomie (1967)                         0       3.44006
           v2    Anomie (1971)                         0       3.06007
           v3    Education                             0       3.54006
           v4    Powerlessness (1967)                  0       3.16006
           v5    Powerlessness (1971)                  0       3.10000
           v6    Occupational Status Index             0      21.21999
```

Output 25.1.3 *continued*

```
                                    Covariances

                                             v1              v2              v3

v1     Anomie (1967)                11.83400000      6.94700000      6.81900000
v2     Anomie (1971)                 6.94700000      9.36400000      5.09100000
v3     Education                     6.81900000      5.09100000     12.53200000
v4     Powerlessness (1967)          4.78300000      5.02800000      7.49500000
v5     Powerlessness (1971)         -3.83900000     -3.88900000     -3.84100000
v6     Occupational Status Index   -21.89900000    -18.83100000    -21.74800000

                                    Covariances

                                             v4              v5              v6

v1     Anomie (1967)                 4.78300000     -3.83900000     -21.8990000
v2     Anomie (1971)                 5.02800000     -3.88900000     -18.8310000
v3     Education                     7.49500000     -3.84100000     -21.7480000
v4     Powerlessness (1967)          9.98600000     -3.62500000     -18.7750000
v5     Powerlessness (1971)         -3.62500000      9.61000000      35.5220000
v6     Occupational Status Index   -18.77500000     35.52200000     450.2880000

                  Determinant        6080570     Ln      15.620609
```

The 12 parameter estimates in the model and their respective locations in the parameter matrices are displayed in Output 25.1.4. Each of the parameters, The1, The2, and The5, is specified for two elements in the parameter matrix _PHI_.

Output 25.1.4 Initial Estimates

```
                             Vector of Initial Estimates

              Parameter       Estimate     Type

         1    Beta             0.50000     Matrix Entry: _BETA_[8:7]
         2    Lamb             0.50000     Matrix Entry: _GAMMA_[6:1]
         3    Gam1            -0.50000     Matrix Entry: _GAMMA_[7:1]
         4    Gam2            -0.50000     Matrix Entry: _GAMMA_[8:1]
         5    Phi              6.00000     Matrix Entry: _PHI_[1:1]
         6    The1             3.00000     Matrix Entry: _PHI_[2:2]  _PHI_[4:4]
         7    The2             3.00000     Matrix Entry: _PHI_[3:3]  _PHI_[5:5]
         8    The5             0.20000     Matrix Entry: _PHI_[4:2]  _PHI_[5:3]
         9    The3             3.00000     Matrix Entry: _PHI_[6:6]
        10    The4             3.00000     Matrix Entry: _PHI_[7:7]
        11    Psi1             4.00000     Matrix Entry: _PHI_[8:8]
        12    Psi2             4.00000     Matrix Entry: _PHI_[9:9]
```

PROC CALIS examines whether each element in the moment matrix is modeled by the parameters defined in the model. If an element is not structured by the model parameters, it is predetermined by its observed value. This occurs, for example, when there are exogenous manifest variables in the model. If present, the predetermined values of the elements are displayed. In the current example, the '.' displayed for all elements in the predicted moment matrix (Output 25.1.5) indicates that there are no predetermined elements in the model.

Output 25.1.5 Predetermined Elements

```
             Predetermined Elements of the Predicted Moment Matrix

                                              v1            v2            v3

v1    Anomie (1967)                            .             .             .
v2    Anomie (1971)                            .             .             .
v3    Education                                .             .             .
v4    Powerlessness (1967)                     .             .             .
v5    Powerlessness (1971)                     .             .             .
v6    Occupational Status Index                .             .             .

             Predetermined Elements of the Predicted Moment Matrix

                                              v4            v5            v6

v1    Anomie (1967)                            .             .             .
v2    Anomie (1971)                            .             .             .
v3    Education                                .             .             .
v4    Powerlessness (1967)                     .             .             .
v5    Powerlessness (1971)                     .             .             .
v6    Occupational Status Index                .             .             .


                   Sum of Squared Differences             0
```

Output 25.1.6 displays the optimization information. You can check this table to determine whether the convergence criterion is satisfied. PROC CALIS displays an error message when problematic solutions are encountered.

Output 25.1.6 Optimization

```
                        Newton-Raphson Ridge Optimization

                           Without Parameter Scaling

                   Parameter Estimates                     12
                   Functions (Observations)                21


                              Optimization Start

Active Constraints                      0  Objective Function        119.33282242
Max Abs Gradient Element   74.016932345
```

Output 25.1.6 *continued*

```
                                                                      Actual
                                                         Max Abs        Over
          Rest     Func        Act     Objective  Obj Fun Gradient        Pred
 Iter     arts    Calls        Con      Function   Change  Element  Ridge   Change

    1        0        2          0       0.82689    118.5   1.3507      0   0.0154
    2        0        3          0       0.09859   0.7283   0.2330      0    0.716
    3        0        4          0       0.01581   0.0828  0.00684      0    1.285
    4        0        5          0       0.01449  0.00132 0.000286      0    1.042
    5        0        6          0       0.01448 9.936E-7 0.000045      0    1.053
    6        0        7          0       0.01448 4.227E-9 1.685E-6      0    1.056

                              Optimization Results

Iterations                            6  Function Calls                      8
Jacobian Calls                        7  Active Constraints                  0
Objective Function         0.0144844811  Max Abs Gradient Element   1.6847829E-6
Ridge                                 0  Actual Over Pred Change    1.0563187228

ABSGCONV convergence criterion satisfied.
```

Next, the predicted model matrix is displayed in the Output 25.1.7, followed by a list of model test statistics or fit indices (Output 25.1.8). Depending on your modeling philosophy, some indices might be preferred to others. In this example, all indices and test statistics point to a good fit of the model.

Output 25.1.7 Predicted Model Matrix

```
                             Predicted Model Matrix

                                         v1              v2              v3

v1     Anomie (1967)            11.90390632      6.91059048      6.83016211
v2     Anomie (1971)             6.91059048      9.35145064      4.93499582
v3     Education                 6.83016211      4.93499582     12.61574998
v4     Powerlessness (1967)      4.93499582      5.01664889      7.50355625
v5     Powerlessness (1971)     -4.16791157     -3.47187034     -4.06565606
v6     Occupational Status Index -22.37688158   -18.63994236    -21.82788734

                             Predicted Model Matrix

                                         v4              v5              v6

v1     Anomie (1967)             4.93499582     -4.16791157     -22.3768816
v2     Anomie (1971)             5.01664889     -3.47187034     -18.6399424
v3     Education                 7.50355625     -4.06565606     -21.8278873
v4     Powerlessness (1967)      9.84539112     -3.38669150     -18.1826302
v5     Powerlessness (1971)     -3.38669150      9.61000000      35.5219999
v6     Occupational Status Index -18.18263015    35.52199986     450.2879993

                  Determinant       6169285     Ln      15.635094
```

Output 25.1.8 Fit Statistics

```
Fit Function                                         0.0145
Goodness of Fit Index (GFI)                          0.9953
GFI Adjusted for Degrees of Freedom (AGFI)           0.9890
Root Mean Square Residual (RMR)                      0.2281
Standardized Root Mean Square Residual (SRMR)        0.0150
Parsimonious GFI (Mulaik, 1989)                      0.5972
Chi-Square                                          13.4851
Chi-Square DF                                             9
Pr > Chi-Square                                      0.1419
Independence Model Chi-Square                        2131.4
Independence Model Chi-Square DF                         15
RMSEA Estimate                                       0.0231
RMSEA 90% Lower Confidence Limit                          .
RMSEA 90% Upper Confidence Limit                     0.0470
ECVI Estimate                                        0.0405
ECVI 90% Lower Confidence Limit                           .
ECVI 90% Upper Confidence Limit                      0.0556
Probability of Close Fit                             0.9705
Bentler's Comparative Fit Index                      0.9979
Normal Theory Reweighted LS Chi-Square              13.2804
Akaike's Information Criterion                      -4.5149
Bozdogan's (1987) CAIC                             -57.0509
Schwarz's Bayesian Criterion                       -48.0509
McDonald's (1989) Centrality                         0.9976
Bentler & Bonett's (1980) Non-normed Index           0.9965
Bentler & Bonett's (1980) NFI                        0.9937
James, Mulaik, & Brett (1982) Parsimonious NFI       0.5962
Z-Test of Wilson & Hilferty (1931)                   1.0754
Bollen (1986) Normed Index Rho1                      0.9895
Bollen (1988) Non-normed Index Delta2                0.9979
Hoelter's (1983) Critical N                            1170
```

PROC CALIS can perform a detailed residual analysis. Large residuals might indicate misspecification of the model. In Output 25.1.9, raw residuals are reported and ranked. Because of the differential scaling of the variables, it is usually more useful to examine the standardized residuals instead. In Output 25.1.10, for example, the table for the 10 largest asymptotically standardized residuals is displayed. The model performs the poorest concerning the variable v5 and its covariance with v2, v1, and v3. This might suggest a misspecification of the model equation for v5. However, because the model fit is quite good, such a possible misspecification is not a serious concern in the analysis.

Output 25.1.9 Raw Residuals and Ranking

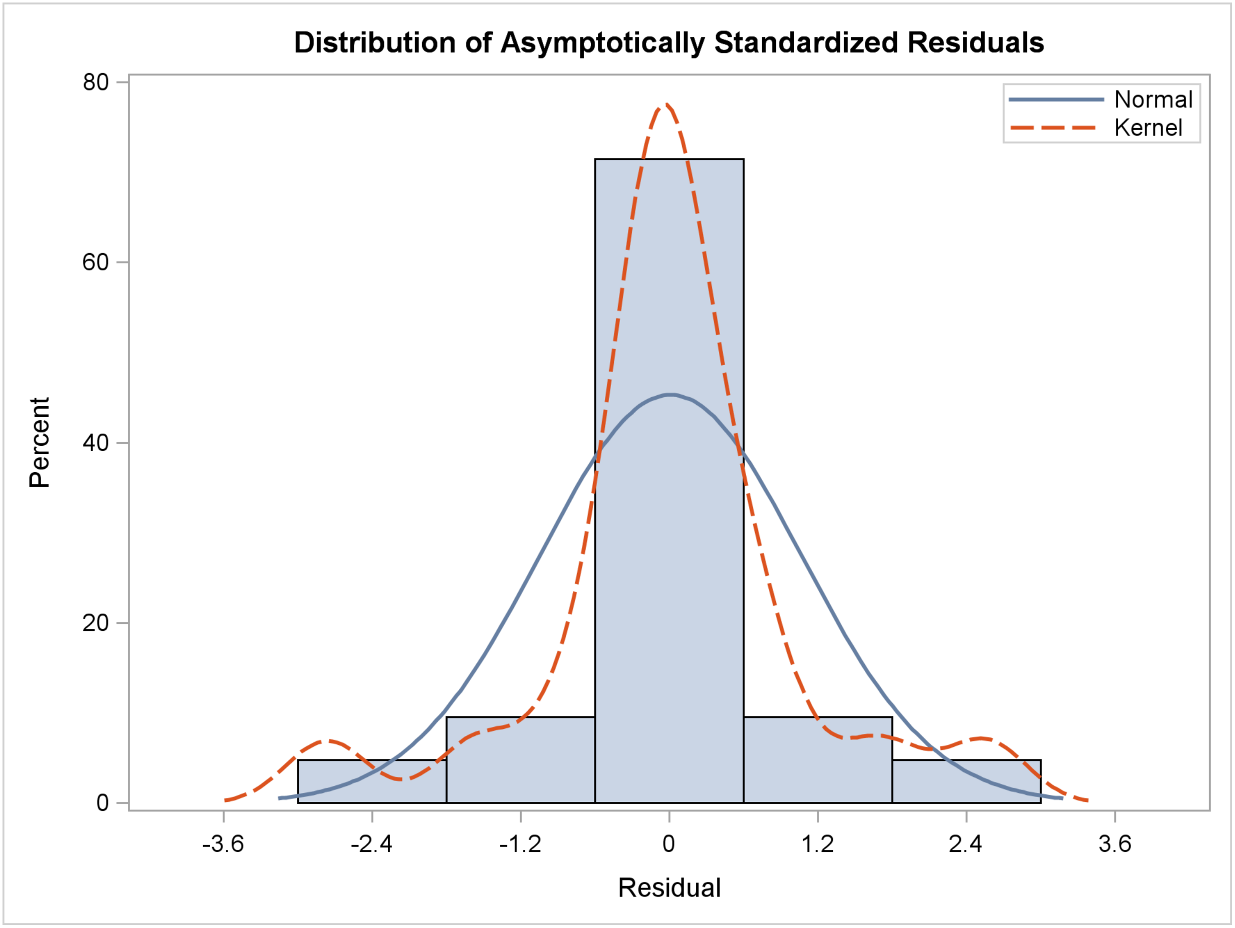
```
                             Raw Residual Matrix

                                           v1             v2             v3

 v1   Anomie (1967)              -.0699063150   0.0364095216   -.0111621061
 v2   Anomie (1971)              0.0364095216   0.0125493646   0.1560041795
 v3   Education                  -.0111621061   0.1560041795   -.0837499788
 v4   Powerlessness (1967)       -.1519958205   0.0113511059   -.0085562504
 v5   Powerlessness (1971)       0.3289115712   -.4171296612   0.2246560598
 v6   Occupational Status Index  0.4778815840   -.1910576405   0.0798873380

                             Raw Residual Matrix

                                           v4             v5             v6

 v1   Anomie (1967)              -.1519958205   0.3289115712   0.4778815840
 v2   Anomie (1971)              0.0113511059   -.4171296612   -.1910576405
 v3   Education                  -.0085562504   0.2246560598   0.0798873380
 v4   Powerlessness (1967)       0.1406088766   -.2383085022   -.5923698474
 v5   Powerlessness (1971)       -.2383085022   0.0000000000   0.0000000000
 v6   Occupational Status Index  -.5923698474   0.0000000000   0.0000000000

        Average Absolute Residual                        0.153928
        Average Off-diagonal Absolute Residual           0.195045

                   Rank Order of the 10 Largest Raw Residuals

                       Row          Column         Residual

                       v6           v4             -0.59237
                       v6           v1              0.47788
                       v5           v2             -0.41713
                       v5           v1              0.32891
                       v5           v4             -0.23831
                       v5           v3              0.22466
                       v6           v2             -0.19106
                       v3           v2              0.15600
                       v4           v1             -0.15200
                       v4           v4              0.14061
```

Output 25.1.10 Asymptotically Standardized Residuals and Ranking

```
                 Asymptotically Standardized Residual Matrix

                                         v1            v2            v3

v1  Anomie (1967)                -0.308548787   0.526654452  -0.056188826
v2  Anomie (1971)                 0.526654452   0.054363484   0.876120855
v3  Education                    -0.056188826   0.876120855  -0.354347092
v4  Powerlessness (1967)         -0.865070455   0.057354415  -0.121874301
v5  Powerlessness (1971)          2.553366366  -2.763708659   1.697931678
v6  Occupational Status Index     0.464866661  -0.170127806   0.070202664

                 Asymptotically Standardized Residual Matrix

                                         v4            v5            v6

v1  Anomie (1967)                -0.865070455   2.553366366   0.464866661
v2  Anomie (1971)                 0.057354415  -2.763708659  -0.170127806
v3  Education                    -0.121874301   1.697931678   0.070202664
v4  Powerlessness (1967)          0.584930625  -1.557412695  -0.495982427
v5  Powerlessness (1971)         -1.557412695   0.000000000   0.000000000
v6  Occupational Status Index    -0.495982427   0.000000000   0.000000000

        Average Standardized Residual                    0.646622
        Average Off-diagonal Standardized Residual       0.818457

    Rank Order of the 10 Largest Asymptotically Standardized Residuals

                    Row         Column        Residual

                    v5          v2            -2.76371
                    v5          v1             2.55337
                    v5          v3             1.69793
                    v5          v4            -1.55741
                    v3          v2             0.87612
                    v4          v1            -0.86507
                    v4          v4             0.58493
                    v2          v1             0.52665
                    v6          v4            -0.49598
                    v6          v1             0.46487
```

The histogram of the asymptotically standardized residuals is displayed in Output 25.1.11, which also shows the normal and kernel approximations. The residual distribution looks quite symmetrical. It shows a small to medium departure from the normal distribution, as evidenced by the discrepancies between the kernel and the normal distribution curves.

Output 25.1.11 Distribution of Asymptotically Standardized Residuals



Output 25.1.12 displays the equations and parameter estimates. Each parameter estimate is displayed with its standard error and the corresponding *t* ratio. As a general rule, a *t* ratio larger than 2 represents a statistically significant departure from 0. From these results, it is observed that both f1 (Alienation 1967) and f2 (Alienation 1971) are regressed negatively on f3 (Socioeconomic Status), and f1 has a positive effect on f2. The estimates and significance tests for the variance and covariance of the exogenous variables are also displayed.

Output 25.1.12 Equations and Parameter Estimates

```
                Manifest Variable Equations with Estimates


         v1      =   1.0000 f1       +  1.0000 e1
         v2      =   0.8330 f1       +  1.0000 e2
         v3      =   1.0000 f2       +  1.0000 e3
         v4      =   0.8330 f2       +  1.0000 e4
         v5      =   1.0000 f3       +  1.0000 e5
         v6      =   5.3688*f3       +  1.0000 e6
         Std Err     0.4337 Lamb
         t Value    12.3788
```

Output 25.1.12 *continued*

```
                    Latent Variable Equations with Estimates


        f1      =  -0.6299*f3       +  1.0000 d1
        Std Err     0.0563 Gam1
        t Value   -11.1809
        f2      =   0.5931*f1       + -0.2409*f3       +  1.0000 d2
        Std Err     0.0468 Beta        0.0549 Gam2
        t Value    12.6788            -4.3885

                       Variances of Exogenous Variables

                                                  Standard
            Variable Parameter      Estimate         Error    t Value

            f3       Phi             6.61632       0.63914      10.35
            e1       The1            3.60788       0.20092      17.96
            e2       The2            3.59493       0.16448      21.86
            e3       The1            3.60788       0.20092      17.96
            e4       The2            3.59493       0.16448      21.86
            e5       The3            2.99368       0.49861       6.00
            e6       The4          259.57580      18.31150      14.18
            d1       Psi1            5.67047       0.42301      13.41
            d2       Psi2            4.51480       0.33532      13.46

                     Covariances Among Exogenous Variables

                                                   Standard
            Var1 Var2 Parameter      Estimate         Error    t Value

            e1   e3   The5            0.90580       0.12167       7.44
            e2   e4   The5            0.90580       0.12167       7.44
```

The measurement scale of variables is often arbitrary. Therefore, it can be useful to look at the standardized equations produced by PROC CALIS. Output 25.1.13 displays the standardized equations and predicted moments. From the standardized structural equations for f1 and f2, you can conclude that SES (f3) has a larger impact on earlier Alienation (f1) than on later Alienation (f3). The squared multiple correlation for each equation is also shown in Output 25.1.13. These correlations indicate the proportion of systematic variance in the equations. Finally, correlations among the exogenous variables are shown.

Output 25.1.13 Standardized Solutions

```
              Manifest Variable Equations with Standardized Estimates


                      v1      =   0.8348 f1   +  0.5505 e1
                      v2      =   0.7846 f1   +  0.6200 e2
                      v3      =   0.8450 f2   +  0.5348 e3
                      v4      =   0.7968 f2   +  0.6043 e4
                      v5      =   0.8297 f3   +  0.5581 e5
                      v6      =   0.6508*f3   +  0.7593 e6
                                         Lamb

               Latent Variable Equations with Standardized Estimates


               f1      =  -0.5626*f3   +  0.8268 d1
                                 Gam1
               f2      =   0.5692*f1   + -0.2064*f3    +  0.7080 d2
                                 Beta           Gam2

                         Squared Multiple Correlations

                                    Error         Total
                    Variable      Variance      Variance    R-Square

                1    v1            3.60788      11.90391      0.6969
                2    v2            3.59493       9.35145      0.6156
                3    v3            3.60788      12.61575      0.7140
                4    v4            3.59493       9.84539      0.6349
                5    v5            2.99368       9.61000      0.6885
                6    v6          259.57580     450.28800      0.4235
                7    f1            5.67047       8.29603      0.3165
                8    f2            4.51480       9.00787      0.4988

                      Correlations Among Exogenous Variables

                        Var1 Var2 Parameter      Estimate

                        e1   e3   The5            0.25106
                        e2   e4   The5            0.25197
```

The predicted covariances among the latent variables and between the observed and the latent variables are displayed in Output 25.1.14.

Output 25.1.14 Predicted Moments

```
                      Predicted Moments of Latent Variables

                              f1                f2                f3

              f1       8.296026985       5.924364730      -4.167911571
              f2       5.924364730       9.007870649      -4.065656060
              f3      -4.167911571      -4.065656060       6.616317547
```

Output 25.1.14 *continued*

```
              Predicted Moments between Manifest and Latent Variables

                                f1                f2                f3

              v1        8.29602698        5.92436473       -4.16791157
              v2        6.91059048        4.93499582       -3.47187034
              v3        5.92436473        9.00787065       -4.06565606
              v4        4.93499582        7.50355625       -3.38669150
              v5       -4.16791157       -4.06565606        6.61631755
              v6      -22.37688158      -21.82788734       35.52199986
```

For interpreting the model, these predicted moments are not as useful as the main results shown previously. However, these predicted moments can be useful for further analysis. For example, they can be useful in constructing bootstrap "populations" for resampling. Another use of these moments is to compute the latent variable score regression coefficients. PROC CALIS computes these coefficients automatically, as shown in Output 25.1.15.

Output 25.1.15 Latent Variable Score Regression Coefficients

```
                   Latent Variable Score Regression Coefficients

                                          f1              f2              f3

v1    Anomie (1967)             0.4131113567    0.0482681051    -.0521264408
v2    Anomie (1971)             0.3454029627    0.0400143300    -.0435560637
v3    Education                 0.0526632293    0.4306175653    -.0399927539
v4    Powerlessness (1967)      0.0437036855    0.3600452776    -.0334000265
v5    Powerlessness (1971)      -.0749215200    -.0639697183    0.5057060770
v6    Occupational Status Index -.0046390513    -.0039609288    0.0313127184
```

In Output 25.1.15, each latent variable is expressed as a linear combination of the observed variables. By computing these linear combinations for each individual, you can estimate the latent variable scores. See Chapter 76, "The SCORE Procedure," for more information about the creation of latent variable scores.

The total effects and indirect effects of the exogenous variables are displayed in Output 25.1.16. These results supplement to those shown in the linear equations in Output 25.1.12, which shows only the direct effects of predictor variables on outcome variables. Total, direct, and indirect effects have the following simple relationship:

$$\text{Total Effect} = \text{Direct Effect} + \text{Indirect Effect}$$

To illustrate, consider the relationships between latent factor f3 and variables v1–v4. In the linear equations shown in Output 25.1.12, latent factor f3 does not have direct effects on variables v1–v4. This does not mean that f3 has no effects on these variables at all. As shown in the first table of Output 25.1.16, latent factor f3 indeed has nonzero total effects on all variables, including variables v1–v4. In the next table that shows indirect effects, latent factor f3, again, has nonzero indirect effects on variables v1–v4, and these effects are identical to the total effects. Because the sum of direct and indirect effects is the total effect, this means that the effects of f3 on v1–v4 are all *indirect*. Similar decomposition of effects can be made for other relationships. For example, while f1 has a total effect of 1.0 on v1, it has no indirect effect on v1. This means that all the effect of f1 on v1 is *direct*, which is also shown in an equation in Output 25.1.12. Finally, consider the effects of f3 on f2. In Output 25.1.16, latent factor f3 has nonzero total effect (−0.6145) and indirect effect (−0.3736) on f2, and these two effects are not identical. The difference of these two effects is the direct effect −0.2409, as shown in an equation in Output 25.1.12.

Output 25.1.16 Total and Indirect Effects

```
                          Total Effects

                      f3                f1                f2

          v1     -0.629944307       1.000000000       0.000000000
          v2     -0.524743608       0.833000000       0.000000000
          v3     -0.614489258       0.593112208       1.000000000
          v4     -0.511869552       0.494062469       0.833000000
          v5      1.000000000       0.000000000       0.000000000
          v6      5.368847492       0.000000000       0.000000000
          f1     -0.629944307       0.000000000       0.000000000
          f2     -0.614489258       0.593112208       0.000000000

                         Indirect Effects

                      f3                f1                f2

          v1     -.6299443069      0.0000000000                 0
          v2     -.5247436076      0.0000000000                 0
          v3     -.6144892580      0.5931122083                 0
          v4     -.5118695519      0.4940624695                 0
          v5     0.0000000000      0.0000000000                 0
          v6     0.0000000000      0.0000000000                 0
          f1     0.0000000000      0.0000000000                 0
          f2     -.3736276589      0.0000000000                 0
```

PROC CALIS can display Lagrange multiplier and Wald statistics for model modifications. Modification indices are displayed for each parameter matrix, as shown in Output 25.1.17 through Output 25.1.22. Only the Lagrange multiplier statistics have significance levels and approximate changes of values displayed. The significance level of the Wald statistic for a given parameter is the same as that shown in the equation output. An insignificant *p*-value for a Wald statistic means that the corresponding parameter can be dropped from the model without significantly worsening the fit of the model.

A significant *p*-value for a Lagrange multiplier test indicates that the model would achieve a better fit if the corresponding parameter were free. To aid in determining significant results, PROC CALIS displays the rank order of the 10 largest Lagrange multiplier statistics. For example, [E5:E2] in the _PHI_ matrix is associated with the largest Lagrange multiplier statistic; the associated *p*-value is 0.0067. This means that adding a parameter for the covariance between E5 and E2 will lead to a significantly better fit of the model. However, adding parameters indiscriminately can result in specification errors. An overfitted model might not perform well with future samples. As always, the decision to add parameters should be accompanied by consideration and knowledge of the application area.

Output 25.1.17 Lagrange Multiplier and Wald Tests for _PHI_

```
            Lagrange Multiplier and Wald Test Indices _PHI_   [9:9]
                              Symmetric Matrix
                   Univariate Tests for Constant Constraints
         Lagrange Multiplier or Wald Index / Probability / Approx Change of Value

                   f3            e1            e2            e3            e4

 f3          107.1619        3.3903        3.3901        0.5752        0.5753
                    .        0.0656        0.0656        0.4482        0.4482
                    .        0.5079       -0.4231        0.2090       -0.1741
                [Phi]

 e1            3.3903      322.4501        0.1529       55.4237        1.2037
               0.0656             .        0.6958             .        0.2726
               0.5079             .        0.0900             .       -0.3262
                             [The1]                      [The5]

 e2            3.3901        0.1529      477.6768        0.5946       55.4237
               0.0656        0.6958             .        0.4406             .
              -0.4231        0.0900             .        0.2328             .
                                           [The2]                    [The5]

 e3            0.5752       55.4237        0.5946      322.4501        0.1528
               0.4482             .        0.4406             .        0.6958
               0.2090             .        0.2328             .       -0.0900
                             [The5]                      [The1]

 e4            0.5753        1.2037       55.4237        0.1528      477.6768
               0.4482        0.2726             .        0.6958             .
              -0.1741       -0.3262             .       -0.0900             .
                                           [The5]                    [The2]

 e5                 .        5.8025        7.3649        1.5982        1.2044
                    .        0.0160        0.0067        0.2062        0.2724
                    .        0.5193       -0.5060        0.2709       -0.2037
                 Sing

 e6                 .        0.7398        1.4168        0.0991        0.0029
                    .        0.3897        0.2339        0.7529        0.9568
                    .       -1.2587        1.5431       -0.4579        0.0700
                 Sing

 d1                 .        0.4840        0.4840        1.1825        1.1825
                    .        0.4866        0.4866        0.2768        0.2768
                    .        0.2276       -0.1896        0.2984       -0.2486
                 Sing

 d2                 .        0.0000        0.0000        0.5942        0.5942
                    .        0.9961        0.9961        0.4408        0.4408
                    .        0.0014       -0.0011       -0.2806        0.2338
                 Sing
```

Output 25.1.17 *continued*

```
                 Lagrange Multiplier and Wald Test Indices _PHI_   [9:9]
                                  Symmetric Matrix
                         Univariate Tests for Constant Constraints
         Lagrange Multiplier or Wald Index / Probability / Approx Change of Value

                            e5             e6             d1             d2

          f3                 .              .              .              .
                             .              .              .              .
                             .              .              .              .
                          Sing           Sing           Sing           Sing

          e1            5.8025         0.7398         0.4840         0.0000
                        0.0160         0.3897         0.4866         0.9961
                        0.5193        -1.2587         0.2276         0.0014


          e2            7.3649         1.4168         0.4840         0.0000
                        0.0067         0.2339         0.4866         0.9961
                       -0.5060         1.5431        -0.1896        -0.0011


          e3            1.5982         0.0991         1.1825         0.5942
                        0.2062         0.7529         0.2768         0.4408
                        0.2709        -0.4579         0.2984        -0.2806


          e4            1.2044         0.0029         1.1825         0.5942
                        0.2724         0.9568         0.2768         0.4408
                       -0.2037         0.0700        -0.2486         0.2338


          e5           36.0486              .         0.1033         0.1035
                             .              .         0.7479         0.7477
                             .              .        -0.2776         0.1062
                        [The3]           Sing

          e6                 .       200.9466         0.1034         0.1035
                             .              .         0.7478         0.7477
                             .              .         1.4906        -0.5700
                          Sing         [The4]

          d1            0.1033         0.1034       179.6950              .
                        0.7479         0.7478              .              .
                       -0.2776         1.4906              .              .
                                                      [Psi1]           Sing

          d2            0.1035         0.1035              .       181.2787
                        0.7477         0.7477              .              .
                        0.1062        -0.5700              .              .
                                                        Sing         [Psi2]
```

Output 25.1.18 Ranking of Lagrange Multipliers in _PHI_

```
          Rank Order of the 10 Largest Lagrange Multipliers in _PHI_

                Row         Column       Chi-Square    Pr > ChiSq

                e5          e2              7.36486        0.0067
                e5          e1              5.80246        0.0160
                e1          f3              3.39030        0.0656
                e2          f3              3.39013        0.0656
                e5          e3              1.59820        0.2062
                e6          e2              1.41677        0.2339
                e5          e4              1.20437        0.2724
                e4          e1              1.20367        0.2726
                d1          e3              1.18251        0.2768
                d1          e4              1.18249        0.2768
```

Output 25.1.19 Lagrange Multiplier and Wald Tests for _GAMMA_

```
             Lagrange Multiplier and Wald Test Indices _GAMMA_ [8:1]
                                General Matrix
                      Univariate Tests for Constant Constraints
       Lagrange Multiplier or Wald Index / Probability / Approx Change of Value

                                    f3

                        v1          3.3903
                                    0.0656
                                    0.0768


                        v2          3.3901
                                    0.0656
                                   -0.0639


                        v3          0.5752
                                    0.4482
                                    0.0316


                        v4          0.5753
                                    0.4482
                                   -0.0263


                        v5            .
                                      .
                                      .
                                      Sing

                        v6        153.2354
                                      .
                                      .
                                    [Lamb]

                        f1        125.0132
                                      .
                                      .
                                    [Gam1]

                        f2         19.2585
                                      .
                                      .
                                    [Gam2]
```

Output 25.1.20 Ranking of Lagrange Multipliers in _GAMMA_

```
           Rank Order of the 4 Largest Lagrange Multipliers in _GAMMA_

                 Row         Column      Chi-Square    Pr > ChiSq

                 v1          f3             3.39030        0.0656
                 v2          f3             3.39013        0.0656
                 v4          f3             0.57526        0.4482
                 v3          f3             0.57523        0.4482
```

Output 25.1.21 Lagrange Multiplier and Wald Tests for _BETA_

```
              Lagrange Multiplier and Wald Test Indices _BETA_  [8:8]
                                  General Matrix
                          Identity-Minus-Inverse Model Matrix
                       Univariate Tests for Constant Constraints
     Lagrange Multiplier or Wald Index / Probability / Approx Change of Value

                         v1             v2             v3             v4

          v1              .         0.1647         0.0511         0.8029
                          .         0.6849         0.8212         0.3702
                          .        -0.0159        -0.0063        -0.0284
                       Sing

          v2         0.5957              .         0.6406         0.0135
                     0.4402              .         0.4235         0.9076
                     0.0218              .         0.0185         0.0032
                                      Sing

          v3         0.3839         0.3027              .         0.1446
                     0.5355         0.5822              .         0.7038
                     0.0178         0.0180              .        -0.0145
                                                     Sing

          v4         0.4487         0.2519         0.0002              .
                     0.5030         0.6157         0.9877              .
                    -0.0160        -0.0144        -0.0004              .
                                                                    Sing

          v5         5.4085         8.6455         2.7123         2.1457
                     0.0200         0.0033         0.0996         0.1430
                     0.1242        -0.1454         0.0785        -0.0674


          v6         0.4209         1.4387         0.3044         0.0213
                     0.5165         0.2304         0.5811         0.8841
                    -0.2189         0.3924        -0.1602         0.0431


          f1         1.0998         1.1021         1.6114         1.6128
                     0.2943         0.2938         0.2043         0.2041
                     0.0977        -0.0817         0.0993        -0.0831


          f2         0.0193         0.0194         0.4765         0.4760
                     0.8896         0.8892         0.4900         0.4902
                    -0.0104         0.0087        -0.0625         0.0522
```

Output 25.1.21 *continued*

```
            Lagrange Multiplier and Wald Test Indices _BETA_  [8:8]
                               General Matrix
                       Identity-Minus-Inverse Model Matrix
                     Univariate Tests for Constant Constraints
      Lagrange Multiplier or Wald Index / Probability / Approx Change of Value

                         v5             v6             f1             f2

        v1           5.4083         0.1233         0.4047         0.4750
                     0.0200         0.7255         0.5247         0.4907
                     0.0697         0.0015        -0.0257        -0.0239


        v2           5.8858         0.0274         0.4047         0.4750
                     0.0153         0.8686         0.5247         0.4907
                    -0.0609        -0.0006         0.0214         0.0199


        v3           1.1537         0.0296         0.1588         0.0817
                     0.2828         0.8634         0.6902         0.7750
                     0.0322         0.0007         0.0144        -0.0110


        v4           0.9867         0.1442         0.1588         0.0817
                     0.3206         0.7041         0.6903         0.7750
                    -0.0249        -0.0014        -0.0120         0.0092


        v5                .              .         0.1033         0.1035
                          .              .         0.7479         0.7476
                          .              .        -0.0490         0.0329
                       Sing           Sing

        v6                .              .         0.1034         0.1035
                          .              .         0.7478         0.7477
                          .              .         0.2629        -0.1765
                       Sing           Sing

        f1           0.1032         0.1035              .              .
                     0.7480         0.7477              .              .
                    -0.0927         0.0057              .              .
                                                     Sing           Sing

        f2           0.1034         0.1035       160.7520              .
                     0.7477         0.7477              .              .
                     0.0355        -0.0022              .              .
                                                   [Beta]           Sing
```

Output 25.1.22 Ranking of Lagrange Multipliers in _BETA_

```
          Rank Order of the 10 Largest Lagrange Multipliers in _BETA_

                Row          Column       Chi-Square    Pr > ChiSq

                v5           v2              8.64546        0.0033
                v2           v5              5.88576        0.0153
                v5           v1              5.40848        0.0200
                v1           v5              5.40832        0.0200
                v5           v3              2.71233        0.0996
                v5           v4              2.14572        0.1430
                f1           v4              1.61279        0.2041
                f1           v3              1.61137        0.2043
                v6           v2              1.43867        0.2304
                v3           v5              1.15372        0.2828
```

When you specify equality constraints, PROC CALIS displays Lagrange multiplier tests for releasing the constraints, as shown in Output 25.1.23. In the current example, none of the three constraints achieve a *p*-value smaller than 0.05. This means that releasing the constraints might not lead to a significantly better fit of the model. Therefore, all constraints are retained in the model.

Output 25.1.23 Tests for Equality Constraints

```
     Univariate Lagrange Multiplier Test for Releasing Equality Constraints

      Equality Constraint     -----Changes-----    Chi-Square    Pr > ChiSq

      [e1:e1] = [e3:e3]        0.0293   -0.0308       0.02106        0.8846
      [e2:e2] = [e4:e4]       -0.1342    0.1388       0.69488        0.4045
      [e3:e1] = [e4:e2]        0.2468   -0.1710       1.29124        0.2558
```

The current model is specified using the LINEQS, STD, and COV statements. As discussed in the section "Getting Started: CALIS Procedure" on page 839, you can also specify the same model by using other specification methods. In the following statements, equivalent COSAN and RAM specifications of the current model are shown. These two specifications would give essentially the same estimation results for the model specified using the LINEQS model statements.

```
proc calis cov data=Wheaton tech=nr edf=931;
   Cosan J(9, Ide) * A(9, Gen, Imi) * P(9, Sym);
   Matrix A
           [ ,7] = 1. .833  5 * 0. Beta (.5) ,
           [ ,8] = 2 * 0.  1.  .833 ,
           [ ,9] = 4 * 0.  1.  Lamb Gam1-Gam2 (.5 2 * -.5);
   Matrix P
           [1,1] = The1-The2 The1-The4 (6 * 3.) ,
           [7,7] = Psi1-Psi2 Phi (2 * 4. 6.) ,
           [3,1] = The5 (.2) ,
           [4,2] = The5 (.2) ;
   Vnames J V1-V6 F1-F3 ,
          A = J ,
          P E1-E6 D1-D3 ;
run;
```

```
proc calis cov data=Wheaton tech=nr edf=931;
   Ram
      1   1  7  1.       ,
      1   2  7  .833     ,
      1   3  8  1.       ,
      1   4  8  .833     ,
      1   5  9  1.       ,
      1   6  9  .5    Lamb ,
      1   7  9  -.5   Gam1 ,
      1   8  7  .5    Beta ,
      1   8  9  -.5   Gam2 ,
      2   1  1  3.    The1 ,
      2   2  2  3.    The2 ,
      2   3  3  3.    The1 ,
      2   4  4  3.    The2 ,
      2   5  5  3.    The3 ,
      2   6  6  3.    The4 ,
      2   1  3  .2    The5 ,
      2   2  4  .2    The5 ,
      2   7  7  4.    Psi1 ,
      2   8  8  4.    Psi2 ,
      2   9  9  6.    Phi  ;
   Vnames 1 F1-F3,
          2 E1-E6 D1-D3;
run;
```

Example 25.2: Simultaneous Equations with Intercept

The demand-and-supply food example of Kmenta (1971, pp. 565, 582) is used to illustrate the use of PROC CALIS for the estimation of intercepts and coefficients of simultaneous equations. The model is specified by two simultaneous equations containing two endogenous variables Q and P and three exogenous variables D, F, and Y,

$$Q_t(demand) = \alpha_1 + \beta_1 P_t + \gamma_1 D_t$$

$$Q_t(supply) = \alpha_2 + \beta_2 P_t + \gamma_2 F_t + \gamma_3 Y_t$$

for $t = 1, \ldots, 20$.

The LINEQS statement requires that each endogenous variable appear on the left-hand side of exactly one equation. Instead of analyzing the system

$$\mathbf{B}^* \eta = \mathbf{\Gamma} \xi + \zeta$$

PROC CALIS analyzes the equivalent system

$$\eta = \mathbf{B}\eta + \mathbf{\Gamma} \xi + \zeta$$

with $\mathbf{B}^* = \mathbf{I} - \mathbf{B}$. This requires that one of the preceding equations be solved for P_t. Solving the second equation for P_t yields

$$P_t = \frac{1}{\beta_2} Q_t - \frac{\alpha_2}{\beta_2} - \frac{\gamma_2}{\beta_2} F_t - \frac{\gamma_3}{\beta_2} Y_t$$

You can estimate the intercepts of a system of simultaneous equations by applying PROC CALIS to the uncorrected covariance (UCOV) matrix of the data set that is augmented by an additional constant variable with the value 1. In the following statements, the uncorrected covariance matrix is augmented by an additional variable INTERCEPT by using the AUGMENT option. The PROC CALIS statement contains the options UCOV and AUG to compute and analyze an augmented UCOV matrix from the input data set FOOD.

```
data food;
title 'Food example of KMENTA(1971, p.565 & 582)';
  input Q P D F Y;
  label  Q='Food Consumption per Head'
         P='Ratio of Food Prices to General Price'
         D='Disposable Income in Constant Prices'
         F='Ratio of Preceding Years Prices'
         Y='Time in Years 1922-1941';
datalines;
  98.485  100.323   87.4   98.0   1
  99.187  104.264   97.6   99.1   2
 102.163  103.435   96.7   99.1   3
 101.504  104.506   98.2   98.1   4
 104.240   98.001   99.8  110.8   5
 103.243   99.456  100.5  108.2   6
 103.993  101.066  103.2  105.6   7
  99.900  104.763  107.8  109.8   8
 100.350   96.446   96.6  108.7   9
 102.820   91.228   88.9  100.6  10
  95.435   93.085   75.1   81.0  11
  92.424   98.801   76.9   68.6  12
  94.535  102.908   84.6   70.9  13
  98.757   98.756   90.6   81.4  14
 105.797   95.119  103.1  102.3  15
 100.225   98.451  105.1  105.0  16
 103.522   86.498   96.4  110.5  17
  99.929  104.016  104.4   92.5  18
 105.223  105.769  110.7   89.3  19
 106.232  113.490  127.1   93.0  20
;

proc calis ucov aug data=food pshort;
   lineqs
      Q = alf1 Intercept + alf2 P + alf3 D + E1,
      P = gam1 Intercept + gam2 Q + gam3 F + gam4 Y + E2;
   std
      E1-E2 = eps1-eps2;
   cov
      E1-E2 = eps3;
   bounds
      eps1-eps2 >= 0. ;
run;
```

The following statements, an essentially equivalent model specification, use program code to reparameterize the model in terms of the original equations; the output is displayed in Output 25.2.1.

```
proc calis data=food ucov aug pshort;
   lineqs
      Q = alpha1 Intercept + beta1 P + gamma1 D + E1,
      P = alpha2_b Intercept + gamma2_b F + gamma3_b Y + _b Q + E2;
   std
      E1-E2 = eps1-eps2;
   cov
      E1-E2 = eps3;
   parameters alpha2 (50.) beta2 gamma2 gamma3 (3*.25);
      alpha2_b = -alpha2 / beta2;
      gamma2_b = -gamma2 / beta2;
      gamma3_b = -gamma3 / beta2;
      _b       = 1 / beta2;
   bounds
      eps1-eps2 >= 0. ;
run;
```

Output 25.2.1 Food Example of Kmenta (1971)

```
                                 LINEQS Model Statement


                        Matrix      Rows    Columns    ------Matrix Type-------

 Term 1            1    _SEL_          6          8    SELECTION
                   2    _BETA_         8          8    EQSBETA          IMINUSINV
                   3    _GAMMA_        8          6    EQSGAMMA
                   4    _PHI_          6          6    SYMMETRIC

                               The 2 Endogenous Variables

     Manifest        Q          P
     Latent


                               The 6 Exogenous Variables

     Manifest        D          F          Y          Intercept
     Latent
     Error           E1         E2

                           Levenberg-Marquardt Optimization

                            Scaling Update of More (1978)

                       Parameter Estimates                     10
                       Functions (Observations)                21
                       Lower Bounds                             2
                       Upper Bounds                             0


                                  Optimization Start

Active Constraints                     0  Objective Function         2.3500065042
Max Abs Gradient Element     203.9741437  Radius                     62167.829174
```

Output 25.2.1 *continued*

```
                                                                   Actual
                                                     Max Abs         Over
        Rest      Func       Act     Objective  Obj Fun Gradient       Pred
 Iter    arts     Calls       Con      Function   Change  Element  Lambda  Change

    1       0         2         0       1.19094   1.1591   3.9410       0   0.688
    2       0         5         0       0.32678   0.8642   9.9864 0.00127   2.356
    3       0         7         0       0.19108   0.1357   5.5100 0.00006   0.685
    4       0        10         0       0.16682   0.0243   2.0513 0.00005   0.867
    5       0        12         0       0.16288  0.00393   1.0570 0.00014   0.828
    6       0        13         0       0.16132  0.00156   0.3643 0.00004   0.864
    7       0        15         0       0.16077 0.000557   0.2176 0.00006   0.984
    8       0        16         0       0.16052 0.000250   0.1819 0.00001   0.618
    9       0        17         0       0.16032 0.000201   0.0662       0   0.971
   10       0        18         0       0.16030 0.000011   0.0195       0   1.108
   11       0        19         0       0.16030 6.116E-7  0.00763       0   1.389
   12       0        20         0       0.16030 9.454E-8  0.00301       0   1.389
   13       0        21         0       0.16030 1.461E-8  0.00118       0   1.388
   14       0        22         0       0.16030 2.269E-9 0.000465       0   1.395
   15       0        23         0       0.16030 3.59E-10 0.000182       0   1.427

                          Optimization Results

Iterations                         15  Function Calls                      24
Jacobian Calls                     16  Active Constraints                   0
Objective Function       0.1603035477  Max Abs Gradient Element  0.0001820805
Lambda                              0  Actual Over Pred Change   1.4266532872
Radius                   0.0010322573

GCONV convergence criterion satisfied.
```

Output 25.2.1 *continued*

```
          Fit Function                                          0.1603
          Goodness of Fit Index (GFI)                           0.9530
          GFI Adjusted for Degrees of Freedom (AGFI)            0.0120
          Root Mean Square Residual (RMR)                       2.0653
          Standardized Root Mean Square Residual (SRMR)         0.0009
          Parsimonious GFI (Mulaik, 1989)                       0.0635
          Chi-Square                                            3.0458
          Chi-Square DF                                              1
          Pr > Chi-Square                                       0.0809
          Independence Model Chi-Square                         534.27
          Independence Model Chi-Square DF                          15
          RMSEA Estimate                                        0.3281
          RMSEA 90% Lower Confidence Limit                           .
          RMSEA 90% Upper Confidence Limit                      0.7777
          ECVI Estimate                                         1.8270
          ECVI 90% Lower Confidence Limit                            .
          ECVI 90% Upper Confidence Limit                       3.3493
          Probability of Close Fit                              0.0882
          Bentler's Comparative Fit Index                       0.9961
          Normal Theory Reweighted LS Chi-Square                2.8142
          Akaike's Information Criterion                        1.0458
          Bozdogan's (1987) CAIC                               -0.9500
          Schwarz's Bayesian Criterion                          0.0500
          McDonald's (1989) Centrality                          0.9501
          Bentler & Bonett's (1980) Non-normed Index            0.9409
          Bentler & Bonett's (1980) NFI                         0.9943
          James, Mulaik, & Brett (1982) Parsimonious NFI        0.0663
          Z-Test of Wilson & Hilferty (1931)                    1.4250
          Bollen (1986) Normed Index Rho1                       0.9145
          Bollen (1988) Non-normed Index Delta2                 0.9962
          Hoelter's (1983) Critical N                               25

                   Manifest Variable Equations with Estimates


     Q         =  -0.2295*P        +  0.3100*D        + 93.6193*Intercept
                         beta1               gamma1              alpha1

                + 1.0000 E1


     P         =   4.2140*Q        + -0.9305*F        + -1.5579*Y
                         _b                  gamma2_b            gamma3_b

                + -218.9*Intercept +  1.0000 E2
                         alpha2_b
```

Output 25.2.1 *continued*

```
                        Variances of Exogenous Variables

                        Variable  Parameter      Estimate

                        D                           10154
                        F                            9989
                        Y                       151.05263
                        Intercept                 1.05263
                        E1        eps1            3.51274
                        E2        eps2          105.06746

                      Covariances Among Exogenous Variables

                   Var1      Var2      Parameter      Estimate

                   D         F                            9994
                   D         Y                            1101
                   F         Y                            1046
                   D         Intercept               102.66842
                   F         Intercept               101.71053
                   Y         Intercept                11.05263
                   E1        E2        eps3          -18.87270

             Manifest Variable Equations with Standardized Estimates


      Q         =  -0.2278*P         +  0.3016*D         +  0.9272*Intercept
                           beta1                gamma1               alpha1

                     +  0.0181 E1


      P         =   4.2467*Q         + -0.9048*F         + -0.1863*Y
                           _b                   gamma2_b             gamma3_b

                     + -2.1849*Intercept +  0.0997 E2
                               alpha2_b

                           Squared Multiple Correlations

                                     Error         Total
                     Variable     Variance      Variance     R-Square

                1    Q             3.51274         10730       0.9997
                2    P           105.06746         10565       0.9901
```

Output 25.2.1 *continued*

```
                   Correlations Among Exogenous Variables

                 Var1      Var2      Parameter      Estimate

                 D         F                         0.99237
                 D         Y                         0.88903
                 F         Y                         0.85184
                 D         Intercept                 0.99308
                 F         Intercept                 0.99188
                 Y         Intercept                 0.87652
                 E1        E2        eps3           -0.98237

                  Additional PARMS and Dependent Parameters

                   The Number of Dependent Parameters is 4

                                       Standard
               Parameter      Estimate        Error    t Value

               alpha2         51.94453            .          .
               beta2           0.23731            .          .
               gamma2          0.22082            .          .
               gamma3          0.36971            .          .
               _b              4.21397            .          .
               gamma2_b       -0.93053            .          .
               gamma3_b       -1.55794            .          .
               alpha2_b     -218.89288            .          .
```

You can obtain almost equivalent results by applying the SAS/ETS procedure SYSLIN to this problem.

Example 25.3: Second-Order Confirmatory Factor Analysis

A second-order confirmatory factor analysis model is applied to a correlation matrix of Thurstone reported by McDonald (1985). Using the LINEQS statement, the three-term second-order factor analysis model is specified in equations notation. The first-order loadings for the three factors, f1, f2, and f3, each refer to three variables, X1-X3, X4-X6, and X7-X9. One second-order factor, f4, reflects the correlations among the three first-order factors. The second-order factor correlation matrix P is defined as a 1×1 identity matrix. Choosing the second-order uniqueness matrix U2 as a diagonal matrix with parameters U21-U23 gives an unidentified model. To compute identified maximum likelihood estimates, the matrix U2 is defined as a 3×3 identity matrix. The following statements generate results that are partially displayed in Output 25.3.1 through Output 25.3.4:

```
data Thurst(TYPE=CORR);
title "Example of THURSTONE resp. McDONALD (1985, p.57, p.105)";
   _TYPE_ = 'CORR'; Input _NAME_ $ Obs1-Obs9;
   label Obs1='Sentences' Obs2='Vocabulary' Obs3='Sentence Completion'
         Obs4='First Letters' Obs5='Four-letter Words' Obs6='Suffices'
         Obs7='Letter series' Obs8='Pedigrees' Obs9='Letter Grouping';
   datalines;
Obs1  1.       .       .       .       .       .       .       .       .
Obs2   .828   1.       .       .       .       .       .       .       .
Obs3   .776   .779    1.       .       .       .       .       .       .
Obs4   .439   .493    .460   1.        .       .       .       .       .
Obs5   .432   .464    .425    .674   1.        .       .       .       .
Obs6   .447   .489    .443    .590    .541   1.        .       .       .
Obs7   .447   .432    .401    .381    .402    .288   1.        .       .
Obs8   .541   .537    .534    .350    .367    .320    .555   1.        .
Obs9   .380   .358    .359    .424    .446    .325    .598    .452   1.
;

proc calis data=Thurst method=max edf=212 pestim se;
lineqs
   Obs1 = X1 F1 + E1,
   Obs2 = X2 F1 + E2,
   Obs3 = X3 F1 + E3,
   Obs4 = X4 F2 + E4,
   Obs5 = X5 F2 + E5,
   Obs6 = X6 F2 + E6,
   Obs7 = X7 F3 + E7,
   Obs8 = X8 F3 + E8,
   Obs9 = X9 F3 + E9,
   F1   = X10 F4 + E10,
   F2   = X11 F4 + E11,
   F3   = X12 F4 + E12;
std
   F4      = 1.,
   E1-E9   = U11-U19,
   E10-E12 = 3 * 1.;
bounds
   0. <= U11-U19;
run;
```

Output 25.3.1 Optimization

```
                  Parameter Estimates                   21
                  Functions (Observations)              45
                  Lower Bounds                           9
                  Upper Bounds                           0


                             Optimization Start

Active Constraints                    0  Objective Function        0.7151823452
Max Abs Gradient Element   0.4067179803  Radius                    2.2578762496
```

Output 25.3.1 *continued*

```
                                                                   Actual
                                                        Max Abs      Over
         Rest     Func       Act    Objective  Obj Fun Gradient      Pred
 Iter    arts    Calls       Con     Function   Change  Element  Lambda    Change

    1       0        2         0      0.23113   0.4840   0.1299       0     1.363
    2       0        3         0      0.18322   0.0479   0.0721       0     1.078
    3       0        4         0      0.18051  0.00271   0.0200       0     1.006
    4       0        5         0      0.18022 0.000289  0.00834       0     1.093
    5       0        6         0      0.18018 0.000041  0.00251       0     1.201
    6       0        7         0      0.18017 6.523E-6  0.00114       0     1.289
    7       0        8         0      0.18017 1.085E-6 0.000388       0     1.347
    8       0        9         0      0.18017 1.853E-7 0.000173       0     1.380
    9       0       10         0      0.18017 3.208E-8 0.000063       0     1.399
   10       0       11         0      0.18017 5.593E-9 0.000028       0     1.408
   11       0       12         0      0.18017 9.79E-10 0.000011       0     1.414

                              Optimization Results

Iterations                              11  Function Calls                      13
Jacobian Calls                          12  Active Constraints                   0
Objective Function            0.1801712147  Max Abs Gradient Element    0.0000105805
Lambda                                   0  Actual Over Pred Change     1.4135882439
Radius                        0.0002026368

GCONV convergence criterion satisfied.
```

Output 25.3.2 Fit Statistics

```
              Example of THURSTONE resp. McDONALD (1985, p.57, p.105)
                Identified Second Order Confirmatory Factor Analysis
          C = F1 * F2 * P * F2' * F1' + F1 * U2 * F1' + U1, With P=U2=Ide

                                The CALIS Procedure
            Covariance Structure Analysis: Maximum Likelihood Estimation

            Fit Function                                          0.1802
            Goodness of Fit Index (GFI)                           0.9596
            GFI Adjusted for Degrees of Freedom (AGFI)            0.9242
            Root Mean Square Residual (RMR)                       0.0436
            Standardized Root Mean Square Residual (SRMR)         0.0436
            Parsimonious GFI (Mulaik, 1989)                       0.6397
            Chi-Square                                           38.1963
            Chi-Square DF                                             24
            Pr > Chi-Square                                       0.0331
            Independence Model Chi-Square                         1101.9
            Independence Model Chi-Square DF                          36
            RMSEA Estimate                                        0.0528
            RMSEA 90% Lower Confidence Limit                      0.0153
            RMSEA 90% Upper Confidence Limit                      0.0831
            ECVI Estimate                                         0.3881
            ECVI 90% Lower Confidence Limit                            .
            ECVI 90% Upper Confidence Limit                       0.4888
            Probability of Close Fit                              0.4088
            Bentler's Comparative Fit Index                       0.9867
            Normal Theory Reweighted LS Chi-Square               40.1947
            Akaike's Information Criterion                       -9.8037
            Bozdogan's (1987) CAIC                             -114.4747
            Schwarz's Bayesian Criterion                        -90.4747
            McDonald's (1989) Centrality                          0.9672
            Bentler & Bonett's (1980) Non-normed Index            0.9800
            Bentler & Bonett's (1980) NFI                         0.9653
            James, Mulaik, & Brett (1982) Parsimonious NFI        0.6436
            Z-Test of Wilson & Hilferty (1931)                    1.8373
            Bollen (1986) Normed Index Rho1                       0.9480
            Bollen (1988) Non-normed Index Delta2                 0.9868
            Hoelter's (1983) Critical N                              204
```

Output 25.3.3 Estimation Results

```
                    Manifest Variable Equations with Estimates


                  Obs1     =    0.5151*F1       +  1.0000 E1
                  Std Err       0.0629 X1
                  t Value       8.1868
                  Obs2     =    0.5203*F1       +  1.0000 E2
                  Std Err       0.0634 X2
                  t Value       8.2090
                  Obs3     =    0.4874*F1       +  1.0000 E3
                  Std Err       0.0608 X3
                  t Value       8.0151
                  Obs4     =    0.5211*F2       +  1.0000 E4
                  Std Err       0.0611 X4
                  t Value       8.5342
                  Obs5     =    0.4971*F2       +  1.0000 E5
                  Std Err       0.0590 X5
                  t Value       8.4213
                  Obs6     =    0.4381*F2       +  1.0000 E6
                  Std Err       0.0560 X6
                  t Value       7.8283
                  Obs7     =    0.4524*F3       +  1.0000 E7
                  Std Err       0.0660 X7
                  t Value       6.8584
                  Obs8     =    0.4173*F3       +  1.0000 E8
                  Std Err       0.0622 X8
                  t Value       6.7135
                  Obs9     =    0.4076*F3       +  1.0000 E9
                  Std Err       0.0613 X9
                  t Value       6.6484

                     Latent Variable Equations with Estimates


                  F1       =    1.4438*F4       +  1.0000 E10
                  Std Err       0.2565 X10
                  t Value       5.6282
                  F2       =    1.2538*F4       +  1.0000 E11
                  Std Err       0.2114 X11
                  t Value       5.9320
                  F3       =    1.4065*F4       +  1.0000 E12
                  Std Err       0.2689 X12
                  t Value       5.2307
```

Output 25.3.3 *continued*

```
                        Variances of Exogenous Variables

                                                     Standard
           Variable Parameter      Estimate             Error     t Value

           F4                       1.00000
           E1       U11             0.18150           0.02848        6.37
           E2       U12             0.16493           0.02777        5.94
           E3       U13             0.26713           0.03336        8.01
           E4       U14             0.30150           0.05102        5.91
           E5       U15             0.36450           0.05264        6.93
           E6       U16             0.50642           0.05963        8.49
           E7       U17             0.39032           0.05934        6.58
           E8       U18             0.48138           0.06225        7.73
           E9       U19             0.50509           0.06333        7.98
           E10                      1.00000
           E11                      1.00000
           E12                      1.00000
```

Output 25.3.4 Standardized and Supplementary Results

```
            Manifest Variable Equations with Standardized Estimates


                    Obs1     =    0.9047*F1    +  0.4260 E1
                                         X1
                    Obs2     =    0.9138*F1    +  0.4061 E2
                                         X2
                    Obs3     =    0.8561*F1    +  0.5168 E3
                                         X3
                    Obs4     =    0.8358*F2    +  0.5491 E4
                                         X4
                    Obs5     =    0.7972*F2    +  0.6037 E5
                                         X5
                    Obs6     =    0.7026*F2    +  0.7116 E6
                                         X6
                    Obs7     =    0.7808*F3    +  0.6248 E7
                                         X7
                    Obs8     =    0.7202*F3    +  0.6938 E8
                                         X8
                    Obs9     =    0.7035*F3    +  0.7107 E9
                                         X9

             Latent Variable Equations with Standardized Estimates


                    F1       =    0.8221*F4    +  0.5694 E10
                                         X10
                    F2       =    0.7818*F4    +  0.6235 E11
                                         X11
                    F3       =    0.8150*F4    +  0.5794 E12
                                         X12
```

Output 25.3.4 *continued*

```
                     Squared Multiple Correlations

                                     Error          Total
                 Variable          Variance       Variance      R-Square

             1   Obs1               0.18150        1.00000        0.8185
             2   Obs2               0.16493        1.00000        0.8351
             3   Obs3               0.26713        1.00000        0.7329
             4   Obs4               0.30150        1.00000        0.6985
             5   Obs5               0.36450        1.00000        0.6355
             6   Obs6               0.50642        1.00000        0.4936
             7   Obs7               0.39032        1.00000        0.6097
             8   Obs8               0.48138        1.00000        0.5186
             9   Obs9               0.50509        1.00000        0.4949
            10   F1                 1.00000        3.08452        0.6758
            11   F2                 1.00000        2.57213        0.6112
            12   F3                 1.00000        2.97832        0.6642
```

To compute McDonald's unidentified model, you would have to change the STD and BOUNDS statements to include three more parameters:

```
std
   F4       = 1.,
   E1-E9    = U11-U19,
   E10-E12 = U21-U23;
bounds
   0. <= U11-U19,
   0. <= U21-U23;
```

The unidentified model is indicated in the output by an analysis of the linear dependencies in the approximate Hessian matrix (not shown). Because the information matrix is singular, standard errors are computed based on a Moore-Penrose inverse. The results computed by PROC CALIS differ from those reported by McDonald (1985). In the case of an unidentified model, the parameter estimates are not unique.

To specify the identified model by using the COSAN model statement, you can use the following statements:

```
proc calis data=Thurst method=max edf=212 pestim se;
   cosan F1(3) * F2(1) * P(1,Ide) + F1(3) * U2(3,Ide) + U1(9,Dia);
   matrix F1
          [ ,1] = X1-X3,
          [ ,2] = 3 * 0. X4-X6,
          [ ,3] = 6 * 0. X7-X9;
   matrix F2
          [ ,1] = X10-X12;
   matrix U1
          [1,1] = U11-U19;
   bounds
          0. <= U11-U19;
run;
```

Because PROC CALIS cannot compute initial estimates for a model specified by the general COSAN statement, this analysis might require more iterations than one that uses the LINEQS statement, depending on the precision of the processor.

Example 25.4: Linear Relations among Factor Loadings

In this example, a confirmatory factor-analysis model with linear constraints on loadings is specified by the COSAN statement. SAS programming statements are used to set the constraints. In the context of the current example, the differences between fitting covariance structures and correlation structures will also be discussed.

The correlation matrix of six variables from Kinzer and Kinzer (N=326) is used by Guttman (1957) as an example that yields an approximate simplex. McDonald (1980) uses this data set as an example of factor analysis where he supposes that the loadings of the second factor are linear functions of the loadings on the first factor. Let **B** be the factor loading matrix containing the two factors and six variables so that

$$\mathbf{B} = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \\ b_{41} & b_{42} \\ b_{51} & b_{52} \\ b_{61} & b_{62} \end{pmatrix}$$

and

$$b_{j2} = \alpha + \beta b_{j1}, \quad j = 1, \ldots, 6$$

The correlation structures are represented by

$$\mathbf{P} = \mathbf{B}\mathbf{B}' + \Psi$$

where $\Psi = \mathrm{diag}(\psi_{11}, \psi_{22}, \psi_{33}, \psi_{44}, \psi_{55}, \psi_{66})$ represents the diagonal matrix of unique variances for the variables.

With parameters α and β being unconstrained, McDonald (1980) has fitted an underidentified model with seven degrees of freedom. Browne (1982) imposes the following identification condition:

$$\beta = -1$$

In this example, Browne's identification condition is imposed. The following statements specify the confirmatory factor model:

```
Data Kinzer(TYPE=CORR);
Title "Data Matrix of Kinzer & Kinzer, see GUTTMAN (1957)";
   _TYPE_ = 'CORR'; INPUT _NAME_ $ var1-var6;
   Datalines;
var1  1.00   .     .     .     .     .
var2   .51  1.00   .     .     .     .
var3   .46   .51  1.00   .     .     .
var4   .46   .47   .54  1.00   .     .
var5   .40   .39   .49   .57  1.00   .
var6   .33   .39   .47   .45   .56  1.00
  ;

proc calis data=Kinzer method=max nobs=326 nose;
   cosan B(2,Gen) * I(2,Ide) + U(6,Ide) * PSI(6,DIA);
   matrix B
          [ ,1]= b11 b21 b31 b41 b51 b61,
          [ ,2]= b12 b22 b32 b42 b52 b62;
   matrix Psi
          [1,1]= psi1-psi6;
   parameters alpha = .5;
   /* SAS Programming Statements */
   b12  = alpha - b11;
   b22  = alpha - b21;
   b32  = alpha - b31;
   b42  = alpha - b41;
   b52  = alpha - b51;
   b62  = alpha - b61;
   vnames B  Fact1 Fact2,
          I  Fact1 Fact2,
          U  var1-var6,
          PSI var1-var6;
run;
```

In the COSAN statement, you specify the model matrices for the correlation structures. Matrix **B** is the 6 $\times$ 2 factor loading matrix. Matrix **I** is a fixed 2 $\times$ 2 identity matrix, representing the factor variances and covariances. In the next term, matrix **U** is a 6 $\times$ 6 identity matrix, appearing there only because it makes the specification conform to the format of the generalized COSAN model. The **PSI** matrix is the matrix for variances and covariances for errors. The matrix specification in the COSAN statement realizes the following matrix model:

$$\mathbf{P} = \mathbf{BIB}' + \mathbf{U}\Psi\mathbf{U}' = \mathbf{BB}' + \Psi$$

which is the intended correlation structures.

In the MATRIX statements, parameters are defined. The loading parameters are all specified with prefix “b” and are located in the **B** matrix. Error variances are all specified with prefix “psi” and are located in the **PSI** matrix. In this model, all manifest variables have nonzero loadings on the two factors. These loading parameters are specified after the equal signs and are named with the prefix “b”.

An additional parameter alpha is specified in the PARAMETERS statement with an initial value of 0.5. Then, six SAS programming statements are used to define the loadings on the second factor as functions of the loadings on the first factor.

Finally, the VNAMES statement is used to name the columns of the model matrices.

In the specification, there are 12 loadings in matrix **B** and six unique variances in matrix **PSI**. Adding the parameter alpha in the list, there are 19 parameters in total. However, the loading parameters are not all independent of each other. As defined in the SAS programming statements, six loadings are dependent. This reduces the number of free parameters to 13. Hence the degrees of freedom for the model is $8 = 21 - 13$.

In Output 25.4.1, a fit summary table is shown. The chi-square test statistic of model fit is 10.337 with df=8 (p = 0.242). This indicates a good fit.

Output 25.4.1 Fit of the Correlation Structures

```
                 Data Matrix of Kinzer & Kinzer, see GUTTMAN (1957)
               Fitting Correlations As Covariance Structures Directly

                               The CALIS Procedure
            Covariance Structure Analysis: Maximum Likelihood Estimation

            Fit Function                                         0.0318
            Goodness of Fit Index (GFI)                          0.9897
            GFI Adjusted for Degrees of Freedom (AGFI)           0.9730
            Root Mean Square Residual (RMR)                      0.0409
            Standardized Root Mean Square Residual (SRMR)        0.0409
            Parsimonious GFI (Mulaik, 1989)                      0.5278
            Chi-Square                                          10.3374
            Chi-Square DF                                             8
            Pr > Chi-Square                                      0.2421
            Independence Model Chi-Square                        682.87
            Independence Model Chi-Square DF                         15
            RMSEA Estimate                                       0.0300
            RMSEA 90% Lower Confidence Limit                          .
            RMSEA 90% Upper Confidence Limit                     0.0756
            ECVI Estimate                                        0.1136
            ECVI 90% Lower Confidence Limit                           .
            ECVI 90% Upper Confidence Limit                      0.1525
            Probability of Close Fit                             0.7137
            Bentler's Comparative Fit Index                      0.9965
            Normal Theory Reweighted LS Chi-Square              10.1441
            Akaike's Information Criterion                      -5.6626
            Bozdogan's (1987) CAIC                             -43.9578
            Schwarz's Bayesian Criterion                       -35.9578
            McDonald's (1989) Centrality                         0.9964
            Bentler & Bonett's (1980) Non-normed Index           0.9934
            Bentler & Bonett's (1980) NFI                        0.9849
            James, Mulaik, & Brett (1982) Parsimonious NFI       0.5253
            Z-Test of Wilson & Hilferty (1931)                   0.7019
            Bollen (1986) Normed Index Rho1                      0.9716
            Bollen (1988) Non-normed Index Delta2                0.9965
            Hoelter's (1983) Critical N                             489
```

The estimated factor loading matrix and error covariance matrix are presented in Output 25.4.2. The estimates for alpha and other dependent parameters are presented in Output 25.4.3.

Output 25.4.2 Estimates

```
                  Data Matrix of Kinzer & Kinzer, see GUTTMAN (1957)
                Fitting Correlations As Covariance Structures Directly

                               The CALIS Procedure
          Covariance Structure Analysis: Maximum Likelihood Estimation

                        Estimated Parameter Matrix B[6:2]
                                 General Matrix

                                Fact1          Fact2

                     var1      0.3609         0.6174
                                [b11]          <b12>

                     var2      0.3212         0.6571
                                [b21]          <b22>

                     var3      0.4859         0.4923
                                [b31]          <b32>

                     var4      0.5745         0.4038
                                [b41]          <b42>

                     var5      0.7985         0.1797
                                [b51]          <b52>

                     var6      0.6736         0.3046
                                [b61]          <b62>
```

Output 25.4.2 *continued*

```
                  Data Matrix of Kinzer & Kinzer, see GUTTMAN (1957)
                Fitting Correlations As Covariance Structures Directly

                                The CALIS Procedure
             Covariance Structure Analysis: Maximum Likelihood Estimation

                        Estimated Parameter Matrix PSI[6:6]
                                  Diagonal Matrix

                  var1         var2         var3         var4         var5         var6

var1            0.5304            0            0            0            0            0
                [psi1]

var2                 0       0.4499            0            0            0            0
                             [psi2]

var3                 0            0       0.4876            0            0            0
                                          [psi3]

var4                 0            0            0       0.4728            0            0
                                                       [psi4]

var5                 0            0            0            0       0.3112            0
                                                                    [psi5]

var6                 0            0            0            0            0       0.5382
                                                                                 [psi6]

                         Estimated Parameter Matrix U[6:6]
                                  Identity Matrix
                               Constant Model Matrix
```

Output 25.4.3 Additional Parameter

```
                     Additional PARMS and Dependent Parameters

                      The Number of Dependent Parameters is 6

                             Parameter      Estimate

                             alpha           0.97825
                             b12             0.61736
                             b22             0.65709
                             b32             0.49234
                             b42             0.40378
                             b52             0.17973
                             b62             0.30462
```

All these estimates are essentially the same as reported in Browne (1982). Notice that there are no standard error estimates in the output, as requested by the NOSE option in the PROC CALIS statement. Standard error estimates are not of interest in the current example.

Fitting the Correct Correlation Structures

In specifying the previous factor model by using COSAN, covariance structures rather than the correlation structures have been incorrectly assumed. That is, when fitting the correlation structures, the diagonal elements of **P** must always be fixed ones. But these fixed ones have not been handled properly in the previous code, resulting in a specification of covariance structures rather than correlation structures. To fix the problem, additional model constraints on the diagonal elements of the correlation structures represented by $(\mathbf{BB}' + \Psi)$ are required. For example, it can be done by constraining the error variances

$$\Psi_{jj} = 1.0 - b_{j1}^2 - b_{j2}^2, \quad j = 1, \ldots, 6$$

Other constraints could also serve the same purpose, but these are the most convenient and therefore are used subsequently. Without setting up these constraints, the previous PROC CALIS run is analyzing covariance structures. As pointed out in Browne (1982), fitting such covariance structures directly is not entirely appropriate for analyzing correlations. For the current data example, according to Browne (1982), the estimates obtained from fitting the wrong covariance structure model are still consistent (as if they were estimating the population parameters in the correlation structures). However, the chi-square test statistic as reported in the preceding section is not correct.

In addition, due to the fact that discrepancy functions used in PROC CALIS are derived for covariance matrices rather than correlation matrices, PROC CALIS is essentially set up for analyzing covariance structures (with or without mean structures), but not correlation structures. Hence, the statistical theory behind PROC CALIS applies to covariance structure analysis, but might not generalize to correlation structure analysis. Despite that, can PROC CALIS somehow fit the correct correlation structures to the current data without compromising the statistical theory? The answer is yes.

First, recall that the correlation structures are represented by:

$$\mathbf{P} = \mathbf{BB}' + \Psi$$

As before, in the **B** matrix, there are six linear constraints on the factor loadings. In addition, the diagonal elements of $(\mathbf{BB}' + \Psi)$ must be ones. To analyze the correlation structures by using PROC CALIS, a covariance structure model with such correlation structures embedded is now specified. That is, the covariance structure to be fitted by PROC CALIS is:

$$\mathbf{\Sigma} = \mathbf{DPD}' = \mathbf{DBB}'\mathbf{D}' + \mathbf{D}\Psi\mathbf{D}'$$

where **D** is a 6×6 diagonal matrix containing the population standard deviations for the manifest variables. This form of covariance structures implies the specification of the following COSAN model:

```
proc calis data=Kinzer method=max nobs=326 nose;
   cosan D(6,DIA) * B(2,Gen) * I(2,Ide) + D(6,DIA) * PSI(6,DIA);
   matrix B
          [ ,1]= b11 b21 b31 b41 b51 b61,
          [ ,2]= b12 b22 b32 b42 b52 b62;
   matrix Psi
          [1,1]= psi1-psi6;
   matrix D
          [1,1]= d1-d6 (6 * 1.);
   parameters alpha (1.);
   /* SAS Programming Statements */
   /* 12 Constraints on Correlation structures */
   b12  = alpha - b11;
   b22  = alpha - b21;
   b32  = alpha - b31;
   b42  = alpha - b41;
   b52  = alpha - b51;
   b62  = alpha - b61;
   psi1 = 1. - b11 * b11 - b12 * b12;
   psi2 = 1. - b21 * b21 - b22 * b22;
   psi3 = 1. - b31 * b31 - b32 * b32;
   psi4 = 1. - b41 * b41 - b42 * b42;
   psi5 = 1. - b51 * b51 - b52 * b52;
   psi6 = 1. - b61 * b61 - b62 * b62;
   vnames D  var1-var6,
          B  Fact1 Fact2,
          I  Fact1 Fact2,
          Psi var1-var6;
run;
```

In the COSAN statement, the covariance structures are described in the form of the generalized COSAN model. The most significant change is the inclusion of matrix **D** for the population standard deviations. Accompanying this change is the addition of a MATRIX statement for defining the error variance parameters in matrix **D**. Also, in this new specification, Psi1-Psi6 are defined as functions of the loading parameters:

$$\Psi_{jj} = 1. - b_{j1}^2 - b_{j2}^2, \quad j = 1, \ldots, 6$$

This ensures that the correlation structures will have fixed ones for the diagonal elements.

Other constraints remain the same way as defined in the previous run. For example, the loadings on the second factor are functions of the loadings on the first factor:

$$b_{j2} = \alpha - b_{j1}, \quad j = 1, \ldots, 6$$

The fit summary is presented in Output 25.4.4. The chi-square test statistic is 14.63 with df=8 (p = 0.067). This shows that the previous chi-square test based on fitting a wrong covariance structure model is indeed questionable.

Output 25.4.4 Model Fit of the Correlation Structures

```
     Data Matrix of Kinzer & Kinzer, see GUTTMAN (1957)
Fitting Correlations By Using Correct Covariance Structures

                    The CALIS Procedure
Covariance Structure Analysis: Maximum Likelihood Estimation

Fit Function                                         0.0450
Goodness of Fit Index (GFI)                          0.9849
GFI Adjusted for Degrees of Freedom (AGFI)           0.9604
Root Mean Square Residual (RMR)                      0.0378
Standardized Root Mean Square Residual (SRMR)        0.0378
Parsimonious GFI (Mulaik, 1989)                      0.5253
Chi-Square                                          14.6269
Chi-Square DF                                             8
Pr > Chi-Square                                      0.0668
Independence Model Chi-Square                        682.87
Independence Model Chi-Square DF                         15
RMSEA Estimate                                       0.0505
RMSEA 90% Lower Confidence Limit                          .
RMSEA 90% Upper Confidence Limit                     0.0908
ECVI Estimate                                        0.1268
ECVI 90% Lower Confidence Limit                           .
ECVI 90% Upper Confidence Limit                      0.1729
Probability of Close Fit                             0.4382
Bentler's Comparative Fit Index                      0.9901
Normal Theory Reweighted LS Chi-Square              14.9199
Akaike's Information Criterion                      -1.3731
Bozdogan's (1987) CAIC                             -39.6683
Schwarz's Bayesian Criterion                       -31.6683
McDonald's (1989) Centrality                         0.9899
Bentler & Bonett's (1980) Non-normed Index           0.9814
Bentler & Bonett's (1980) NFI                        0.9786
James, Mulaik, & Brett (1982) Parsimonious NFI       0.5219
Z-Test of Wilson & Hilferty (1931)                   1.5034
Bollen (1986) Normed Index Rho1                      0.9598
Bollen (1988) Non-normed Index Delta2                0.9902
Hoelter's (1983) Critical N                             346
```

Estimates of the loadings, population standard deviations, error variances, and the alpha parameter are presented in Output 25.4.5 and Output 25.4.6.

Output 25.4.5 Estimates of Loadings and Standard Deviations

```
                         Estimated Parameter Matrix B[6:2]
                                   General Matrix

                                      Fact1          Fact2

                         var1        0.3422         0.6318
                                      [b11]          <b12>

                         var2        0.3210         0.6531
                                      [b21]          <b22>

                         var3        0.4918         0.4822
                                      [b31]          <b32>

                         var4        0.5755         0.3985
                                      [b41]          <b42>

                         var5        0.7769         0.1971
                                      [b51]          <b52>

                         var6        0.6666         0.3074
                                      [b61]          <b62>

                         Estimated Parameter Matrix D[6:6]
                                  Diagonal Matrix

               var1        var2        var3        var4        var5        var6

   var1      1.0077           0           0           0           0           0
               [d1]

   var2           0      0.9971           0           0           0           0
                           [d2]

   var3           0           0      0.9908           0           0           0
                                       [d3]

   var4           0           0           0      0.9909           0           0
                                                   [d4]

   var5           0           0           0           0      0.9964           0
                                                               [d5]

   var6           0           0           0           0           0      1.0169
                                                                           [d6]
```

Output 25.4.6 Estimates of Error Variances and Alpha

```
                         Estimated Parameter Matrix PSI[6:6]
                                  Diagonal Matrix

                var1         var2         var3         var4         var5         var6

   var1       0.4837            0            0            0            0            0
             <psi1>

   var2            0       0.4705            0            0            0            0
                          <psi2>

   var3            0            0       0.5256            0            0            0
                                       <psi3>

   var4            0            0            0       0.5100            0            0
                                                    <psi4>

   var5            0            0            0            0       0.3576            0
                                                                 <psi5>

   var6            0            0            0            0            0       0.4612
                                                                              <psi6>

                      Additional PARMS and Dependent Parameters

                       The Number of Dependent Parameters is 12

                               Parameter      Estimate

                               alpha           0.97400
                               b12             0.63184
                               b22             0.65305
                               b32             0.48222
                               b42             0.39849
                               b52             0.19715
                               b62             0.30741
                               psi1            0.48370
                               psi2            0.47051
                               psi3            0.52561
                               psi4            0.50999
                               psi5            0.35763
                               psi6            0.46115
```

Except for the population standard deviation parameter d's, all other parameters estimated in the current model can be compared with those from the previous fitting of an incorrect covariance structure model. Although estimates in the current model do not differ very much from those in the previous analysis, it is assuring that they are obtained from fitting a correctly specified covariance structure model with the intended correlation structures embedded.

Example 25.5: Ordinal Relations among Factor Loadings

The same data set as in Example 25.4 is used in McDonald (1980) for analysis with ordinally constrained factor loadings. In Example 25.4, the results of the linearly constrained factor analysis show that the loadings of the two factors are ordered as 2, 1, 3, 4, 6, 5. McDonald (1980) then tests the hypothesis that the factor loadings are all nonnegative and can be ordered in the following manner:

$$b_{11} \le b_{21} \le b_{31} \le b_{41} \le b_{51} \le b_{61}$$

$$b_{12} \ge b_{22} \ge b_{32} \ge b_{42} \ge b_{52} \ge b_{62}$$

In this example, these ordinal relationships are implemented using the LINCON statement in the following COSAN model specification:

```
proc calis data=Kinzer method=max nobs=326 nose;
   cosan D(6,DIA) * B(2,Gen) * I(2,Ide) + D(6,DIA) * PSI(6,DIA);
   matrix B
          [ ,1]= 0.  b21 b31 b41 b51 b61,
          [ ,2]= b12 b22 b32 b42 b52 b62;
   matrix Psi
          [1,1]= psi1-psi6;
   matrix D
          [1,1]= d1-d6 (6 * 1.);
   lincon  b21  <= b31,
           b31  <= b41,
           b41  <= b51,
           b51  <= b61,
           b62  <= b52,
           b52  <= b42,
           b42  <= b32,
           b32  <= b22,
           b22  <= b12;
   bounds b21 >= 0;
   /* SAS Programming Statements */
   /* 6 Constraints on Correlation structures */
   psi1 = 1. - b12 * b12;
   psi2 = 1. - b21 * b21 - b22 * b22;
   psi3 = 1. - b31 * b31 - b32 * b32;
   psi4 = 1. - b41 * b41 - b42 * b42;
   psi5 = 1. - b51 * b51 - b52 * b52;
   psi6 = 1. - b61 * b61 - b62 * b62;
   vnames D  var1-var6,
          B  Fact1 Fact2,
          I  Fact1 Fact2,
          Psi var1-var6;
run;
```

Again, as in Example 25.4, correlation structures are analyzed in the current example so that the error variance parameters psi1-psi6 are defined as functions of the loadings in the SAS programming statements. However, the loading parameters are not constrained the same way as in Example 25.4.

First, b11 is fixed at 0 for identification purposes. Then, the LINCON statement is used to specify the ordinal relations of the factor loadings. Parameter b21 is set to be nonnegative in the BOUNDS statement.

As shown in Output 25.5.1, the solution converges in 10 iterations. In the fit summary table, the chi-square test statistic is 8.48 ($df = 6$, $p = 0.20$). This indicates a good fit. However, in the model there are 11 loading parameters (the b's) and 6 population standard deviation parameters (the d's). The degrees of freedom should have been $4 = 21 - 11 - 6$, but why is this number 6 in the fit summary table?

Output 25.5.1 Final Iteration Status and Fit

```
                              Optimization Results

Iterations                             10  Function Calls                     13
Jacobian Calls                         11  Active Constraints                  2
Objective Function           0.0260990149  Max Abs Gradient Element   6.2794755E-6
Lambda                                  0  Actual Over Pred Change    0.9491590286
Radius                       0.0001243346

ABSGCONV convergence criterion satisfied.

WARNING: There are 2 active constraints at the solution.  The standard errors
         and Chi-Square test statistic assume the solution is located in the
         interior of the parameter space and hence do not apply if it is likely
         that some different set of inequality constraints could be active.


NOTE: The degrees of freedom are increased by the number of active constraints
      (see Dijkstra, 1992). The number of parameters in calculating fit indices
      is decreased by the number of active constraints. To turn off the
      adjustment, use the NOADJDF option.
```

Output 25.5.1 *continued*

```
            Fit Function                                            0.0261
            Goodness of Fit Index (GFI)                             0.9914
            GFI Adjusted for Degrees of Freedom (AGFI)              0.9699
            Root Mean Square Residual (RMR)                         0.0169
            Standardized Root Mean Square Residual (SRMR)           0.0169
            Parsimonious GFI (Mulaik, 1989)                         0.3966
            Chi-Square                                              8.4822
            Chi-Square DF                                                6
            Pr > Chi-Square                                         0.2049
            Independence Model Chi-Square                           682.87
            Independence Model Chi-Square DF                            15
            RMSEA Estimate                                          0.0357
            RMSEA 90% Lower Confidence Limit                             .
            RMSEA 90% Upper Confidence Limit                        0.0860
            ECVI Estimate                                           0.1204
            ECVI 90% Lower Confidence Limit                              .
            ECVI 90% Upper Confidence Limit                         0.1576
            Probability of Close Fit                                0.6154
            Bentler's Comparative Fit Index                         0.9963
            Normal Theory Reweighted LS Chi-Square                  8.4575
            Akaike's Information Criterion                         -3.5178
            Bozdogan's (1987) CAIC                                -32.2392
            Schwarz's Bayesian Criterion                          -26.2392
            McDonald's (1989) Centrality                            0.9962
            Bentler & Bonett's (1980) Non-normed Index              0.9907
            Bentler & Bonett's (1980) NFI                           0.9876
            James, Mulaik, & Brett (1982) Parsimonious NFI          0.3950
            Z-Test of Wilson & Hilferty (1931)                      0.8281
            Bollen (1986) Normed Index Rho1                         0.9689
            Bollen (1988) Non-normed Index Delta2                   0.9963
            Hoelter's (1983) Critical N                                484
```

The reason is that there are two active constraints in the solution, making it two free parameters fewer in the final solution than originally specified. Active constraints are those inequality constraints that are fulfilled on the boundary equalities. As shown in the "Optimization Results" table, the number of active constraints for the current fitting is two. The default treatment in PROC CALIS is to treat these active constraints as if they were going to happen for all possible repeated sampling. This might as well be seen as fitting the active equality constraints on every possible repeated samples. This results in an increase of the degrees of freedom for model fit, as adjusted in the current fit summary table in Output 25.5.1. To warn you about the degrees-of-freedom adjustment, the following messages are also printed with the output:

```
WARNING: There are 2 active constraints at the solution.  The
         standard errors and Chi-Square test statistic assume
         solution is located in the interior of the parameter
         space and hence do not apply if it is likely that some
         different set of inequality constraints could be
         active.
```

```
NOTE: The degrees of freedom are increased by the number of
      active constraints (see Dijkstra, 1992). The number of
      parameters in calculating fit indices is decreased by the
      number of active constraints. To turn off the adjustment,
      use the NOADJDF option.
```

When active constraints are encountered, you need to be cautious about two implications. First, the estimates fall on the boundary of the parameter space originally specified. As shown in Output 25.5.2, estimates for b51 and b61 are the same, and so are the pair of estimates for b12 and b22. These pairs of parameters were originally constrained by inequalities in the model. For example, b61 was constrained to be at least as big as b51. The fact that this constraint is honored only on the bound means that a better model fit might exist with b61 being smaller than b51. Similarly, a better model fit might result without requiring b12 to be at least as big as b22. Solutions with active constraints thus might imply that the original strict inequality constraints are not appropriate for the data.

Output 25.5.2 Estimates

```
                    Estimated Parameter Matrix B[6:2]
                              General Matrix

                                  Fact1           Fact2

                    var1              0          0.7100
                                                  [b12]

                    var2         0.0393          0.7100
                                  [b21]           [b22]

                    var3         0.2463          0.6799
                                  [b31]           [b32]

                    var4         0.3295          0.6561
                                  [b41]           [b42]

                    var5         0.5432          0.5541
                                  [b51]           [b52]

                    var6         0.5432          0.4733
                                  [b61]           [b62]
```

Output 25.5.2 *continued*

```
                    Estimated Parameter Matrix D[6:6]
                             Diagonal Matrix

              var1        var2        var3        var4        var5        var6

  var1      1.0022           0           0           0           0           0
              [d1]

  var2           0      0.9985           0           0           0           0
                          [d2]

  var3           0           0      1.0004           0           0           0
                                      [d3]

  var4           0           0           0      1.0004           0           0
                                                  [d4]

  var5           0           0           0           0      0.9990           0
                                                              [d5]

  var6           0           0           0           0           0      1.0021
                                                                          [d6]

                   Estimated Parameter Matrix PSI[6:6]
                             Diagonal Matrix

              var1        var2        var3        var4        var5        var6

  var1      0.4959           0           0           0           0           0
            <psi1>

  var2           0      0.4944           0           0           0           0
                        <psi2>

  var3           0           0      0.4771           0           0           0
                                    <psi3>

  var4           0           0           0      0.4610           0           0
                                                <psi4>

  var5           0           0           0           0      0.3979           0
                                                            <psi5>

  var6           0           0           0           0           0      0.4809
                                                                        <psi6>
```

The second implication for the presence of active constraints is that the chi-square test statistic and the standard error estimates are computed as if repeated samples were fitted by the model with the presence of the active equality constraints. The degrees-of-freedom adjustment by PROC CALIS is based on this assumption. However, if the presence of particular active constraints in fitting reflects only a rare sampling event that is not likely to happen in repeated sampling, degrees-of-freedom adjustment or even the computation of chi-square statistic and standard error estimates might not be justified. Unfortunately, whether the active constraints are reflecting the truth of the model or pure sampling fluctuation is usually difficult to determine.

Example 25.6: Longitudinal Factor Analysis

The following example (McDonald 1980) illustrates both the ability of PROC CALIS to formulate complex covariance structure analysis problems by the generalized COSAN matrix model and the use of programming statements to impose nonlinear constraints on the parameters. The example is a longitudinal factor analysis that uses the Swaminathan (1974) model. For $m = 3$ tests, $k = 3$ occasions, and $r = 2$ factors the matrix model is formulated in the section "First-Order Autoregressive Longitudinal Factor Model" on page 834 as follows:

$$\mathbf{C} = \mathbf{F}_1\mathbf{F}_2\mathbf{F}_3\mathbf{L}\mathbf{F}_3^{-1}\mathbf{F}_2^{-1}\mathbf{P}(\mathbf{F}_2^{-1})'(\mathbf{F}_3^{-1})'\mathbf{L}'\mathbf{F}_3'\mathbf{F}_2'\mathbf{F}_1' + \mathbf{U}^2$$

$$F_1 = \begin{pmatrix} B_1 & & \\ & B_2 & \\ & & B_3 \end{pmatrix}, \quad F_2 = \begin{pmatrix} I_2 & & \\ & D_2 & \\ & & D_2 \end{pmatrix}, \quad F_3 = \begin{pmatrix} I_2 & & \\ & I_2 & \\ & & D_3 \end{pmatrix}$$

$$L = \begin{pmatrix} I_2 & o & o \\ I_2 & I_2 & o \\ I_2 & I_2 & I_2 \end{pmatrix}, \quad P = \begin{pmatrix} I_2 & & \\ & S_2 & \\ & & S_3 \end{pmatrix}, \quad U = \begin{pmatrix} U_{11} & U_{12} & U_{13} \\ U_{21} & U_{22} & U_{23} \\ U_{31} & U_{32} & U_{33} \end{pmatrix}$$

$$S_2 = I_2 - D_2^2, \qquad S_3 = I_2 - D_3^2$$

The Swaminathan longitudinal factor model assumes that the factor scores for each (m) common factor change from occasion to occasion (k) according to a first-order autoregressive scheme. The matrix $\mathbf{F}_1$ contains the k factor loading matrices $\mathbf{B}_1, \mathbf{B}_2, \mathbf{B}_3$ (each is $n \times m$). The matrices $\mathbf{D}_2, \mathbf{D}_3, \mathbf{S}_2, \mathbf{S}_3$ and $\mathbf{U}_{ij}, i, j = 1, \ldots, k$, are diagonal, and the matrices $\mathbf{D}_i$ and $\mathbf{S}_i, i = 2, \ldots, k$, are subjected to the constraint

$$\mathbf{S}_i + \mathbf{D}_i^2 = \mathbf{I}$$

Since the constructed correlation matrix given by McDonald (1980) is singular, only unweighted least squares estimates can be computed. The following statements specify the COSAN model for the correlation structures.

```
data Mcdon(TYPE=CORR);
Title "Swaminathan's Longitudinal Factor Model, Data: McDONALD(1980)";
Title2 "Constructed Singular Correlation Matrix, GLS & ML not possible";
   _TYPE_ = 'CORR'; INPUT _NAME_ $ Obs1-Obs9;
   datalines;
Obs1  1.000    .      .      .      .      .      .      .      .
Obs2   .100  1.000    .      .      .      .      .      .      .
Obs3   .250   .400  1.000    .      .      .      .      .      .
Obs4   .720   .108   .270  1.000    .      .      .      .      .
Obs5   .135   .740   .380   .180  1.000    .      .      .      .
Obs6   .270   .318   .800   .360   .530  1.000    .      .      .
Obs7   .650   .054   .135   .730   .090   .180  1.000    .      .
Obs8   .108   .690   .196   .144   .700   .269   .200  1.000    .
Obs9   .189   .202   .710   .252   .336   .760   .350   .580  1.000
     ;
```

```
proc calis data=Mcdon method=ls tech=nr nobs=100;
cosan B(6,Gen) * D1(6,Dia) * D2(6,Dia) * T(6,Low) * D3(6,Dia,Inv)
               * D4(6,Dia,Inv) * P(6,Dia) + U(9,Sym);
   matrix B
            [ ,1]= X1-X3,
            [ ,2]= 0. X4-X5,
            [ ,3]= 3 * 0. X6-X8,
            [ ,4]= 4 * 0. X9-X10,
            [ ,5]= 6 * 0. X11-X13,
            [ ,6]= 7 * 0. X14-X15;
   matrix D1
            [1,1]= 2 * 1. X16 X17 X16 X17;
   matrix D2
            [1,1]= 4 * 1. X18 X19;
   matrix T
            [1,1]= 6 * 1.,
            [3,1]= 4 * 1.,
            [5,1]= 2 * 1.;
   matrix D3
            [1,1]= 4 * 1. X18 X19;
   matrix D4
            [1,1]= 2 * 1. X16 X17 X16 X17;
   matrix P
            [1,1]= 2 * 1. X20-X23;
   matrix U
            [1,1]= X24-X32,
            [4,1]= X33-X38,
            [7,1]= X39-X41;
   bounds 0. <= X24-X32,
         -1. <= X16-X19 <= 1.;
   X20 = 1. - X16 * X16;
   X21 = 1. - X17 * X17;
   X22 = 1. - X18 * X18;
   X23 = 1. - X19 * X19;
run;
```

Because this formulation of Swaminathan's model in general leads to an unidentified problem, the results given here are different from those reported by McDonald (1980). The displayed output of PROC CALIS also indicates that the fitted central model matrices **P** and **U** are not positive-definite. The BOUNDS statement constrains the diagonals of the matrices **P** and **U** to be nonnegative, but this cannot prevent **U** from having three negative eigenvalues. The fact that many of the published results for more complex models in covariance structure analysis are connected to unidentified problems implies that more theoretical work should be done to study the general features of such models.

References

Akaike, H. (1974), “A New Look at the Statistical Identification Model,” *IEEE Transactions on Automatic Control*, 19, 716–723.

Akaike, H. (1987), “Factor Analysis and AIC,” *Psychometrika*, 52, 317–332.

Beale, E. M. L. (1972), “A Derivation of Conjugate Gradients,” in *Numerical Methods for Nonlinear Optimization*, ed. F. A. Lootsma, London: Academic Press.

Bentler, P. M. (1985), *Theory and Implementation of EQS: A Structural Equations Program*, Manual for Program Version 2.0, Los Angeles: BMDP Statistical Software.

Bentler, P. M. (1986), *Lagrange Multiplier and Wald Tests for EQS and EQS/PC*, Los Angeles: BMDP Statistical Software.

Bentler, P. M. (1989), *EQS, Structural Equations, Program Manual*, Program Version 3.0, Los Angeles: BMDP Statistical Software.

Bentler, P. M. and Bonett, D. G. (1980), “Significance Tests and Goodness of Fit in the Analysis of Covariance Structures,” *Psychological Bulletin*, 88, 588–606.

Bentler, P. M. and Weeks, D. G. (1980), “Linear Structural Equations with Latent Variables,” *Psychometrika*, 45, 289–308.

Bishop, Y. M. M., Fienberg, S. E., and Holland, P. W. (1977), *Discrete Multivariate Analysis: Theory and Practice*, Cambridge and London: MIT Press.

Bollen, K. A. (1986), “Sample Size and Bentler and Bonett’s Nonnormed Fit Index,” *Psychometrika*, 51, 375–377.

Bollen, K. A. (1989a), “A New Incremental Fit Index for General Structural Equation Models,” *Sociological Methods and Research*, 17, 303–316.

Bollen, K. A. (1989b), *Structural Equations with Latent Variables*, New York: John Wiley & Sons.

Bozdogan, H. (1987), “Model Selection and Akaike’s Information Criterion (AIC): The General Theory and its Analytical Extensions,” *Psychometrika*, 52, 345–370.

Browne, M. W. (1974), “Generalized Least Squares Estimators in the Analysis of Covariance Structures,” *South African Statistical Journal*, 8, 1–24.

Browne, M. W. (1982), “Covariance Structures,” in *Topics in Multivariate Analyses*, ed. D. M. Hawkins, New York: Cambridge University Press.

Browne, M. W. (1984), “Asymptotically Distribution-Free Methods for the Analysis of Covariance Structures,” *British Journal of Mathematical and Statistical Psychology*, 37, 62–83.

Browne, M. W. and Cudeck, R. (1993), “Alternative Ways of Assessing Model Fit,” in *Testing Structural Equation Models*, ed. K. A. Bollen and S. Long, Newbury Park, CA: Sage Publications.

Browne, M. W. and Du Toit, S. H. C. (1992), “Automated Fitting of Nonstandard Models,” *Multivariate Behavioral Research*, 27, 269–300.

Browne, M. W. and Shapiro, A. (1986), “The Asymptotic Covariance Matrix of Sample Correlation Coefficients under General Conditions,” *Linear Algebra and Its Applications*, 82, 169–176.

Bunch, J. R. and Kaufman, K. (1977), “Some Stable Methods for Calculating Inertia and Solving Symmetric Linear Systems,” *Mathematics of Computation*, 31, 162–179.

Buse, A. (1982), “The Likelihood Ratio, Wald, and Lagrange Multiplier Tests: An Expository Note,” *The American Statistician*, 36, 153–157.

Chamberlain, R. M., Powell, M. J. D., Lemarechal, C., and Pedersen, H. C. (1982), “The Watchdog Technique for Forcing Convergence in Algorithms for Constrained Optimization,” *Mathematical Programming*, 16, 1–17.

DeLeeuw, J. (1983), “Models and Methods for the Analysis of Correlation Coefficients,” *Journal of Econometrics*, 22, 113–137.

Dennis, J. E., Gay, D. M., and Welsch, R. E. (1981), “An Adaptive Nonlinear Least-Squares Algorithm,” *ACM Trans. Math. Software*, 7, 348–368.

Dennis, J. E. and Mei, H. H. W. (1979), “Two New Unconstrained Optimization Algorithms Which Use Function and Gradient Values,” *J. Optim. Theory Appl.*, 28, 453–482.

Dijkstra, T. K. (1992), “On Statistical Inference with Parameter Estimates on the Boundary of the Parameter Space,” *British Journal of Mathematical and Statistical Psychology*, 45, 289–309.

Everitt, B. S. (1984), *An Introduction to Latent Variable Methods*, London: Chapman & Hall.

Fletcher, R. (1980), *Practical Methods of Optimization*, Vol. 1, Chichester: John Wiley & Sons.

Fletcher, R. (1987), *Practical Methods of Optimization*, Second Edition, Chichester: John Wiley & Sons.

Fuller, W. A. (1987), *Measurement Error Models*, New York: John Wiley & Sons.

Gay, D. M. (1983), “Subroutines for Unconstrained Minimization,” *ACM Trans. Math. Software*, 9, 503–524.

Gill, E. P., Murray, W., Saunders, M. A., and Wright, M. H. (1983), “Computing Forward-Difference Intervals for Numerical Optimization,” *SIAM J. Sci. Stat. Comput.*, 4, 310–321.

Gill, E. P., Murray, W., and Wright, M. H. (1981), *Practical Optimization*, London: Academic Press.

Gill, E. P., Murray, W., Saunders, M. A., and Wright, M. H. (1984), “Procedures for Optimization Problems with a Mixture of Bounds and General Linear Constraints,” *ACM Trans. Math. Software*, 10, 282–298.

Guttman, L. (1957), “Empirical Verification of the Radex Structure of Mental Abilities and Personality Traits,” *Educational and Psychological Measurement*, 17, 391–407.

Hägglund, G. (1982), "Factor Analysis by Instrumental Variable Methods," *Psychometrika*, 47, 209–222.

Hoelter, J. W. (1983), "The Analysis of Covariance Structures: Goodness-of-Fit Indices," *Sociological Methods and Research*, 11, 325–344.

James, L. R., Mulaik, S. A., and Brett, J. M. (1982), *Causal Analysis: Assumptions, Models, and Data*, Beverly Hills, CA: Sage Publications.

Jennrich, R. I. (1987), "Tableau Algorithms for Factor Analysis by Instrumental Variable Methods," *Psychometrika*, 52, 469–476.

Jöreskog, K. G. (1973), "A General Method for Estimating a Linear Structural Equation System," in *Structural Equation Models in the Social Sciences*, ed. A. S. Goldberger and O. D. Duncan, New York: Seminar Press.

Jöreskog, K. G. (1978), "Structural Analysis of Covariance and Correlation Matrices," *Psychometrika*, 43, 443–477.

Jöreskog, K. G. and Sörbom, D. (1985), *LISREL VI; Analysis of Linear Structural Relationships by Maximum Likelihood, Instrumental Variables, and Least Squares*, Uppsala: University of Uppsala.

Jöreskog, K. G. and Sörbom, D. (1988), *LISREL 7: A Guide to the Program and Applications*, Chicago: SPSS Inc.

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

Kmenta, J. (1971), *Elements of Econometrics*, New York: Macmillan.

Lawley, D. N. and Maxwell, A. E. (1971), *Factor Analysis as a Statistical Method*, New York: American Elsevier.

Lee, S. Y. (1985), "On Testing Functional Constraints in Structural Equation Models," *Biometrika*, 72, 125–131.

Loehlin, J. C. (1987), *Latent Variable Models, An Introduction to Factor, Path, and Structural Analysis*, Hillsdale, NJ: Lawrence Erlbaum Associates.

Long, J. S. (1983), *Covariance Structure Models, an Introduction to LISREL*, Beverly Hills, CA: Sage Publications.

MacCallum, R. (1986), "Specification Searches in Covariance Structure Modeling," *Psychological Bulletin*, 100, 107–120.

McArdle, J. J. (1980), "Causal Modeling Applied to Psychonomic Systems Simulation," *Behavior Research Methods & Instrumentation*, 12, 193–209.

McArdle, J. J. (1988), "Dynamic but Structural Equation Modeling of Repeated Measures Data," in *The Handbook of Multivariate Experimental Psychology*, ed. J. R. Nesselroade and R. B. Cattell, New York: Plenum Press.

McArdle, J. J. and McDonald, R. P. (1984), "Some Algebraic Properties of the Reticular Action Model," *British Journal of Mathematical and Statistical Psychology*, 37, 234–251.

McDonald, R. P. (1978), "A Simple Comprehensive Model for the Analysis of Covariance Structures," *British Journal of Mathematical and Statistical Psychology*, 31, 59–72.

McDonald, R. P. (1980), "A Simple Comprehensive Model for the Analysis of Covariance Structures: Some Remarks on Applications," *British Journal of Mathematical and Statistical Psychology*, 33, 161–183.

McDonald, R. P. (1985), *Factor Analysis and Related Methods*, Hillsdale, NJ: Lawrence Erlbaum Associates.

McDonald, R. P. (1989), "An Index of Goodness-of-Fit Based on Noncentrality," *Journal of Classification*, 6, 97–103.

McDonald, R. P. and Hartmann, W. (1992), "A Procedure for Obtaining Initial Values of Parameters in the RAM Model," *Multivariate Behavioral Research*, 27, 57–176.

Moré, J. J. (1978), "The Levenberg-Marquardt Algorithm: Implementation and Theory," in *Numerical Analysis—Dundee 1977*, ed. G. A. Watson, Lecture Notes in Mathematics 630, Berlin: Springer-Verlag.

Moré, J. J. and Sorensen, D. C. (1983), "Computing a Trust-Region Step," *SIAM J. Sci. Stat. Comput.*, 4, 553–572.

Mulaik, S. A., James, L. R., Van Alstine, J., Bennett, N., Lind, S. and Stilwell, C. D. (1989), "Evaluation of Goodness-of-Fit Indices for Structural Equation Models," *Psychological Bulletin*, 105, 430–445.

Polak, E. (1971), *Computational Methods in Optimization*, New York: Academic Press.

Powell, J. M. D. (1977), "Restart Procedures for the Conjugate Gradient Method," *Math. Prog.*, 12, 241–254.

Powell, J. M. D. (1978a), "A Fast Algorithm for Nonlinearly Constraint Optimization Calculations," in *Numerical Analysis, Dundee 1977, Lecture Notes in Mathematics 630*, ed. G. A. Watson, Berlin: Springer-Verlag, 144–175.

Powell, J. M. D. (1978b), "Algorithms for Nonlinear Constraints That Use Lagrangian Functions," *Mathematical Programming*, 14, 224–248.

Powell, M. J. D. (1982a), "Extensions to Subroutine VF02AD," in *Systems Modeling and Optimization, Lecture Notes In Control and Information Sciences 38*, ed. R. F. Drenick and F. Kozin, Berlin: Springer-Verlag, 529–538.

Powell, J. M. D. (1982b), "VMCWD: A Fortran Subroutine for Constrained Optimization," *DAMTP 1982/NA4*, Cambridge, England.

Schwarz, G. (1978), "Estimating the Dimension of a Model," *Annals of Statistics*, 6, 461–464.

Sclove, L. S. (1987), "Application of Model-Selection Criteria to Some Problems in Multivariate Analysis," *Psychometrika*, 52, 333–343.

Steiger, J. H. and Lind, J. C. (1980), "Statistically Based Tests for the Number of Common Factors," paper presented at the annual meeting of the Psychometric Society, Iowa City, IA.

Swaminathan, H. (1974), "A General Factor Model for the Description of Change," Report LR-74-9, Laboratory of Psychometric and Evaluative Research, University of Massachusetts.

Tucker, L. R. and Lewis, C. (1973), "A Reliability Coefficient for Maximum Likelihood Factor Analysis," *Psychometrika*, 38, 1–10.

Wheaton, B., Muthèn, B., Alwin, D. F., and Summers, G. F. (1977), "Assessing Reliability and Stability in Panel Models," in *Sociological Methodology*, ed. D. R. Heise, San Francisco: Jossey Bass.

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

Wilson, E. B. and Hilferty, M. M. (1931), "The Distribution of Chi-Square," *Proceeding of the National Academy of Science*, 17, 694.

Subject Index

Syntax Index

Your Turn

We welcome your feedback.

- If you have comments about this book, please send them to **`yourturn@sas.com`**. Include the full title and page numbers (if applicable).
- If you have comments about the software, please send them to **`suggest@sas.com`**.