



THE
POWER
TO KNOW.

SAS/STAT[®] 14.1 User's Guide

The NLIN Procedure

This document is an individual chapter from *SAS/STAT® 14.1 User's Guide*.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2015. *SAS/STAT® 14.1 User's Guide*. Cary, NC: SAS Institute Inc.

SAS/STAT® 14.1 User's Guide

Copyright © 2015, SAS Institute Inc., Cary, NC, USA

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

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

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

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

U.S. Government License Rights; Restricted Rights: The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication, or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a), and DFAR 227.7202-4, and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government's rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, NC 27513-2414

July 2015

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

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

Chapter 81

The NLIN Procedure

Contents

Overview: NLIN Procedure	6410
Getting Started: NLIN Procedure	6412
Nonlinear or Linear Model	6412
Notation for Nonlinear Regression Models	6413
Estimating the Parameters in the Nonlinear Model	6414
Syntax: NLIN Procedure	6419
PROC NLIN Statement	6420
BOOTSTRAP Statement	6430
BOUNDS Statement	6432
BY Statement	6433
CONTROL Statement	6434
DER Statement	6434
ID Statement	6434
MODEL Statement	6434
OUTPUT Statement	6435
PARAMETERS Statement	6439
PROFILE Statement	6442
RETAIN Statement	6443
Other Programming Statements	6443
Details: NLIN Procedure	6445
Automatic Derivatives	6445
Measures of Nonlinearity, Diagnostics and Inference	6446
Missing Values	6454
Special Variables	6454
Troubleshooting	6456
Computational Methods	6457
Output Data Sets	6462
Confidence Intervals	6463
Covariance Matrix of Parameter Estimates	6464
Convergence Measures	6465
Displayed Output	6466
Incompatibilities with SAS 6.11 and Earlier Versions of PROC NLIN	6466
ODS Table Names	6468
ODS Graphics	6469
Examples: NLIN Procedure	6471
Example 81.1: Segmented Model	6471

Example 81.2: Iteratively Reweighted Least Squares	6476
Example 81.3: Probit Model with Likelihood Function	6479
Example 81.4: Affecting Curvature through Parameterization	6482
Example 81.5: Comparing Nonlinear Trends among Groups	6489
Example 81.6: ODS Graphics and Diagnostics	6500
Example 81.7: Parameter Profiling and Bootstrapping	6505
References	6514

Overview: NLIN Procedure

The NLIN procedure fits nonlinear regression models and estimates the parameters by nonlinear least squares or weighted nonlinear least squares. You specify the model with programming statements. This gives you great flexibility in modeling the relationship between the response variable and independent (regressor) variables. It does, however, require additional coding compared to model specifications in linear modeling procedures such as the REG, GLM, and MIXED procedures.

Estimating parameters in a nonlinear model is an iterative process that commences from starting values. You need to declare the parameters in your model and supply their initial values for the NLIN procedure. You do not need to specify derivatives of the model equation with respect to the parameters. Although facilities for specifying first and second derivatives exist in the NLIN procedure, it is not recommended that you specify derivatives this way. Obtaining derivatives from user-specified expressions predates the high-quality automatic differentiator that is now used by the NLIN procedure.

Nonlinear least-squares estimation involves finding those values in the parameter space that minimize the (weighted) residual sum of squares. In a sense, this is a “distribution-free” estimation criterion since the distribution of the data does not need to be fully specified. Instead, the assumption of homoscedastic and uncorrelated model errors with zero mean is sufficient. You can relax the homoscedasticity assumption by using a weighted residual sum of squares criterion. The assumption of uncorrelated errors (independent observations) cannot be relaxed in the NLIN procedure. In summary, the primary assumptions for analysis with the NLIN procedure are as follows:

- The structure in the response variable can be decomposed additively into a mean function and an error component.
- The model errors are uncorrelated and have zero mean. Unless a weighted analysis is performed, the errors are also assumed to be homoscedastic (have equal variance).
- The mean function consists of known regressor (independent) variables and unknown constants (the parameters).

Fitting nonlinear models can be a difficult undertaking. There is no closed-form solution for the parameter estimates, and the process is iterative. There can be one or more local minima in the residual sum of squares surface, and the process depends on the starting values supplied by the user. You can reduce the dependence on the starting values and reduce the chance to arrive at a local minimum by specifying a grid of starting values. The NLIN procedure then computes the residual sum of squares at each point on the grid and starts the

iterative process from the point that yields the lowest sum of squares. Even in this case, however, convergence does not guarantee that a global minimum has been found.

The numerical behavior of a model and a model–data combination can depend on the way in which you parameterize the model—for example, whether parameters are expressed on the logarithmic scale or not. Parameterization also has bearing on the interpretation of the estimated quantities and the statistical properties of the parameter estimators. Inferential procedures in nonlinear regression models are typically approximate in that they rely on the asymptotic properties of the parameter estimators that are obtained as the sample size grows without bound. Such asymptotic inference can be questionable in small samples, especially if the behavior of the parameter estimators is “far-from-linear.” Reparameterization of the model can yield parameters whose behavior is akin to that of estimators in linear models. These parameters exhibit close-to-linear behavior.

The NLIN procedure solves the nonlinear least squares problem by one of the following four algorithms (methods):

- steepest-descent or gradient method
- Newton method
- modified Gauss-Newton method
- Marquardt method

These methods use derivatives or approximations to derivatives of the SSE with respect to the parameters to guide the search for the parameters producing the smallest SSE. Derivatives computed automatically by the NLIN procedure are analytic, unless the model contains functions for which an analytic derivative is not available.

Using PROC NLIN, you can also do the following:

- confine the estimation procedure to a certain range of values of the parameters by imposing bounds on the estimates
- produce new SAS data sets containing predicted values, parameter estimates, residuals and other model diagnostics, estimates at each iteration, and so forth.

You can use the NLIN procedure for segmented models (see [Example 81.1](#)) or robust regression (see [Example 81.2](#)). You can also use it to compute maximum-likelihood estimates for certain models (see Jennrich and Moore 1975; Charnes, Frome, and Yu 1976). For maximum likelihood estimation in a model with a linear predictor and binomial error distribution, see the LOGISTIC, PROBIT, GENMOD, GLIMMIX, and CATMOD procedures. For a linear model with a Poisson, gamma, or inverse Gaussian error distribution, see the GENMOD and GLIMMIX procedures. For likelihood estimation in a linear model with a normal error distribution, see the MIXED, GENMOD, and GLIMMIX procedures. The PHREG and LIFEREG procedures fit survival models by maximum likelihood. For general maximum likelihood estimation, see the NLP procedure in the *SAS/OR User's Guide: Mathematical Programming* and the NLMIXED procedure. These procedures are recommended over the NLIN procedure for solving maximum likelihood problems.

PROC NLIN uses the Output Delivery System (ODS). ODS enables you to convert any of the output from PROC NLIN into a SAS data set. See the section “[ODS Table Names](#)” on page 6468 for a listing of the ODS tables that are produced by the NLIN procedure.

PROC NLIN exploits all the available cores on a multicore machine during bootstrap estimation and parameter profiling. It does so by performing multiple optimizations in parallel for these tasks. see the [BOOTSTRAP](#) and [PROFILE](#) statements.

In addition, PROC NLIN can produce graphs when ODS Graphics is enabled. For more information, see the [PLOTS](#) option and the section “[ODS Graphics](#)” on page 6469 for a listing of the ODS graphs.

Getting Started: NLIN Procedure

Nonlinear or Linear Model

The NLIN procedure performs univariate nonlinear regression by using the least squares method. Nonlinear regression analysis is indicated when the functional relationship between the response variable and the predictor variables is nonlinear. Nonlinearity in this context refers to a nonlinear relationship in the *parameters*. Many linear regression models exhibit a relationship in the regressor (predictor) variables that is not simply a straight line. This does not make the models nonlinear. A model is nonlinear in the parameters if the derivative of the model with respect to a parameter depends on this or other parameters.

Consider, for example the models

$$E[Y|x] = \beta_0 + \beta_1 x$$

$$E[Y|x] = \beta_0 + \beta_1 x + \beta_2 x^2$$

$$E[Y|x] = \beta + x/\alpha$$

In these expressions, $E[Y|x]$ denotes the expected value of the response variable Y at the fixed value of x . (The conditioning on x simply indicates that the predictor variables are assumed to be non-random in models fit by the NLIN procedure. Conditioning is often omitted for brevity in this chapter.)

Only the third model is a nonlinear model. The first model is a simple linear regression. It is linear in the parameters β_0 and β_1 since the model derivatives do not depend on unknowns:

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

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

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

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

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

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

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

$$\begin{aligned}\frac{\partial}{\partial \beta} (\beta + x/\alpha) &= 1 \\ \frac{\partial}{\partial \alpha} (\beta + x/\alpha) &= -\frac{x}{\alpha^2}\end{aligned}$$

The second of these derivatives depends on a parameter α . A model is nonlinear if it is not linear in at least one parameter.

Notation for Nonlinear Regression Models

This section briefly introduces the basic notation for nonlinear regression models that applies in this chapter. Additional notation is introduced throughout as needed.

The $(n \times 1)$ vector of observed responses is denoted as \mathbf{y} . This vector is the realization of an $(n \times 1)$ random vector \mathbf{Y} . The NLIN procedure assumes that the variance matrix of this random vector is $\sigma^2 \mathbf{I}$. In other words, the observations have equal variance (are homoscedastic) and are uncorrelated. By defining the special variable `_WEIGHT_` in your NLIN programming statements, you can introduce heterogeneous variances. If a `_WEIGHT_` variable is present, then $\text{Var}[\mathbf{Y}] = \sigma^2 \mathbf{W}^{-1}$, where \mathbf{W} is a diagonal matrix containing the values of the `_WEIGHT_` variable.

The mean of the random vector is represented by a nonlinear model that depends on parameters β_1, \dots, β_p and regressor (independent) variables z_1, \dots, z_k :

$$E[Y_i] = f(\beta_1, \beta_2, \dots, \beta_p; z_{i1}, \dots, z_{ik})$$

In contrast to linear models, the number of regressor variables (k) does not necessarily equal the number of parameters (p) in the mean function $f(\cdot)$. For example, the model fitted in the next subsection contains a single regressor and two parameters.

To represent the mean of the vector of observations, boldface notation is used in an obvious extension of the previous equation:

$$E[\mathbf{Y}] = \mathbf{f}(\boldsymbol{\beta}; \mathbf{z}_1, \dots, \mathbf{z}_k)$$

The vector \mathbf{z}_1 , for example, is an $(n \times 1)$ vector of the values for the first regressor variables. The explicit dependence of the mean function on $\boldsymbol{\beta}$ and/or the \mathbf{z} vectors is often omitted for brevity.

In summary, the stochastic structure of models fit with the NLIN procedure is mathematically captured by

$$\begin{aligned}\mathbf{Y} &= \mathbf{f}(\boldsymbol{\beta}; \mathbf{z}_1, \dots, \mathbf{z}_k) + \boldsymbol{\epsilon} \\ E[\boldsymbol{\epsilon}] &= \mathbf{0} \\ \text{Var}[\boldsymbol{\epsilon}] &= \sigma^2 \mathbf{I}\end{aligned}$$

Note that the residual variance σ^2 is typically also unknown. Since it is not estimated in the same fashion as the other p parameters, it is often not counted in the number of parameters of the nonlinear regression. An estimate of σ^2 is obtained after the model fit by the method of moments based on the residual sum of squares.

A matrix that plays an important role in fitting nonlinear regression models is the $(n \times p)$ matrix of the first partial derivatives of the mean function \mathbf{f} with respect to the p model parameters. It is frequently denoted as

$$\mathbf{X} = \frac{\partial \mathbf{f}(\boldsymbol{\beta}; \mathbf{z}_1, \dots, \mathbf{z}_k)}{\partial \boldsymbol{\beta}}$$

The use of the symbol \mathbf{X} —common in linear statistical modeling—is no accident here. The first derivative matrix plays a similar role in nonlinear regression to that of the \mathbf{X} matrix in a linear model. For example, the asymptotic variance of the nonlinear least-squares estimators is proportional to $(\mathbf{X}'\mathbf{X})^{-1}$, and projection-type matrices in nonlinear regressions are based on $\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$. Also, fitting a nonlinear regression model can be cast as an iterative process where a nonlinear model is approximated by a series of linear models in which the derivative matrix is the regressor matrix. An important difference between linear and nonlinear models is that the derivatives in a linear model do not depend on any parameters (see previous subsection). In contrast, the derivative matrix $\partial \mathbf{f}(\boldsymbol{\beta})/\partial \boldsymbol{\beta}$ is a function of at least one element of $\boldsymbol{\beta}$. It is this dependence that lies at the core of the fact that estimating the parameters in a nonlinear model cannot be accomplished in closed form, but it is an iterative process that commences with user-supplied starting values and attempts to continually improve on the parameter estimates.

Estimating the Parameters in the Nonlinear Model

As an example of a nonlinear regression analysis, consider the following theoretical model of enzyme kinetics. The model relates the initial velocity of an enzymatic reaction to the substrate concentration.

$$f(\mathbf{x}, \boldsymbol{\theta}) = \frac{\theta_1 x_i}{\theta_2 + x_i}, \quad \text{for } i = 1, 2, \dots, n$$

where x_i represents the amount of substrate for n trials and $f(\mathbf{x}, \boldsymbol{\theta})$ is the velocity of the reaction. The vector $\boldsymbol{\theta}$ contains the rate parameters. This model is known as the Michaelis-Menten model in biochemistry (Ratkowsky 1990, p. 59). The model exists in many parameterizations. In the form shown here, θ_1 is the maximum velocity of the reaction that is theoretically attainable. The parameter θ_2 is the substrate concentration at which the velocity is 50% of the maximum.

Suppose that you want to study the relationship between concentration and velocity for a particular enzyme/substrate pair. You record the reaction rate (velocity) observed at different substrate concentrations.

A SAS data set is created for this experiment in the following DATA step:

```
data Enzyme;
  input Concentration Velocity @@;
  datalines;
0.26 124.7    0.30 126.9
0.48 135.9    0.50 137.6
0.54 139.6    0.68 141.1
0.82 142.8    1.14 147.6
1.28 149.8    1.38 149.4
1.80 153.9    2.30 152.5
2.44 154.5    2.48 154.7
;
```

The SAS data set Enzyme contains the two variables Concentration (substrate concentration) and Velocity (reaction rate). The following statements fit the Michaelis-Menten model by nonlinear least squares:

```
proc nlin data=Enzyme method=marquardt hougard;
  parms theta1=155
        theta2=0 to 0.07 by 0.01;
  model Velocity = theta1*Concentration / (theta2 + Concentration);
run;
```

The **DATA=** option specifies that the SAS data set Enzyme be used in the analysis. The **METHOD=** option directs PROC NLIN to use the MARQUARDT iterative method. The **HOUGAARD** option requests that a skewness measure be calculated for the parameters.

The **PARMS** statement declares the parameters and specifies their initial values. Suppose that V represents the velocity and C represents the substrate concentration. In this example, the initial estimates listed in the **PARMS** statement for θ_1 and θ_2 are obtained as follows:

θ_1 : Because the model is a monotonic increasing function in C , and because

$$\lim_{C \rightarrow \infty} \left(\frac{\theta_1 C}{\theta_2 + C} \right) = \theta_1$$

you can take the largest observed value of the variable Velocity (154.7) as the initial value for the parameter Theta1. Thus, the **PARMS** statement specifies 155 as the initial value for Theta1, which is approximately equal to the maximum observed velocity.

θ_2 : To obtain an initial value for the parameter θ_2 , first rearrange the model equation to solve for θ_2 :

$$\theta_2 = \frac{\theta_1 C}{V} - C$$

By substituting the initial value of Theta1 for θ_1 and taking each pair of observed values of Concentration and Velocity for C and V , respectively, you obtain a set of possible starting values for Theta2 ranging from about 0.01 to 0.07.

You can choose any value within this range as a starting value for Theta2, or you can direct PROC NLIN to perform a preliminary search for the best initial Theta2 value within that range of values. The **PARMS** statement specifies a range of values for Theta2, resulting in a search over the grid points from 0 to 0.07 in increments of 0.01.

The **MODEL** statement specifies the enzymatic reaction model

$$V = \frac{\theta_1 C}{\theta_2 + C}$$

in terms of the data set variables Velocity and Concentration and in terms of the parameters in the **PARMS** statement.

The results from this PROC NLIN invocation are displayed in the following figures.

PROC NLIN evaluates the model at each point on the specified grid for the Theta2 parameter. [Figure 81.1](#) displays the calculations resulting from the grid search.

Figure 81.1 Nonlinear Least-Squares Grid Search

**The NLIN Procedure
Dependent Variable Velocity**

Grid Search		
theta1	theta2	Sum of Squares
155.0	0	3075.4
155.0	0.0100	2074.1
155.0	0.0200	1310.3
155.0	0.0300	752.0
155.0	0.0400	371.9
155.0	0.0500	147.2
155.0	0.0600	58.1130
155.0	0.0700	87.9662

The parameter Theta1 is held constant at its specified initial value of 155, the grid is traversed, and the residual sum of squares is computed at each point. The “best” starting value is the point that corresponds to the smallest value of the residual sum of squares. The best set of starting values is obtained for $\theta_1 = 155$, $\theta_2 = 0.06$ ([Figure 81.1](#)). PROC NLIN uses this point from which to start the following, iterative phase of nonlinear least-squares estimation.

Figure 81.2 displays the iteration history. Note that the first entry in the “Iterative Phase” table echoes the starting values and the residual sum of squares for the best value combination in Figure 81.1. The subsequent rows of the table show the updates of the parameter estimates and the improvement (decrease) in the residual sum of squares. For this data-and-model combination, the first iteration yielded a large improvement in the sum of squares (from 58.113 to 19.7017). Further steps were necessary to improve the estimates in order to achieve the convergence criterion. The NLIN procedure by default determines convergence by using R , the relative offset measure of Bates and Watts (1981). Convergence is declared when this measure is less than 10^{-5} —in this example, after three iterations.

Figure 81.2 Iteration History and Convergence Status

The NLIN Procedure
Dependent Variable Velocity
Method: Marquardt

Iterative Phase			
Iter	theta1	theta2	Sum of Squares
0	155.0	0.0600	58.1130
1	158.0	0.0736	19.7017
2	158.1	0.0741	19.6606
3	158.1	0.0741	19.6606

NOTE: Convergence criterion met.

Figure 81.3 Estimation Summary

Estimation Summary	
Method	Marquardt
Iterations	3
R	5.861E-6
PPC(theta2)	8.569E-7
RPC(theta2)	0.000078
Object	2.902E-7
Objective	19.66059
Observations Read	14
Observations Used	14
Observations Missing	0

A summary of the estimation including several convergence measures (R , PPC, RPC, and Object) is displayed in Figure 81.3.

The “R” measure in Figure 81.3 is the relative offset convergence measure of Bates and Watts. A “PPC” value of 8.569E–7 indicates that the parameter Theta2 (which has the largest PPC value of the parameters) would change by that relative amount, if PROC NLIN were to take an additional iteration step. The “RPC” value indicates that Theta2 changed by 0.000078, relative to its value in the last iteration. These changes are measured before step length adjustments are made. The “Object” measure indicates that the objective function changed by 2.902E–7 in relative value from the last iteration.

Figure 81.4 displays the analysis of variance table for the model. The table displays the degrees of freedom, sums of squares, and mean squares along with the model F test.

Figure 81.4 Nonlinear Least-Squares Analysis of Variance

Note: An intercept was not specified for this model.

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	2	290116	145058	88537.2	<.0001
Error	12	19.6606	1.6384		
Uncorrected Total	14	290135			

Figure 81.5 Parameter Estimates and Approximate 95% Confidence Intervals

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits		Skewness
theta1	158.1	0.6737	156.6	159.6	0.0152
theta2	0.0741	0.00313	0.0673	0.0809	0.0362

Figure 81.5 displays the estimates for each parameter, the associated asymptotic standard error, and the upper and lower values for the asymptotic 95% confidence interval. PROC NLIN also displays the asymptotic correlations between the estimated parameters (not shown).

The skewness measures of 0.0152 and 0.0362 indicate that the parameter estimators exhibit close-to-linear behavior and that their standard errors and confidence intervals can be safely used for inferences.

Thus, the estimated nonlinear model that relates reaction velocity and substrate concentration can be written as

$$\hat{V} = \frac{158.1C}{0.0741 + C}$$

where \hat{V} represents the predicted velocity or rate of the reaction and C represents the substrate concentration.

Syntax: NLIN Procedure

The following statements are available in the NLIN procedure:

```

PROC NLIN < options > ;
  BOOTSTRAP < / options > ;
  BOUNDS inequality < , ... , inequality > ;
  BY variables ;
  CONTROL variable < =values > < ... variable < =values > > ;
  DER. parameter=expression ;
  DER. parameter.parameter=expression ;
  ID variables ;
  MODEL dependent = expression ;
  OUTPUT OUT=SAS-data-set keyword=names < ... keyword=names > ;
  PARAMETERS < parameter-specification > < , ... , parameter-specification >
    < / PDATA=SAS-data-set > ;
  PROFILE parameter < ... parameter > < / options > ;
  RETAIN variable < =values > < ... variable < =values > > ;
  Programming statements ;

```

The statements in the NLIN procedure, in addition to the **PROC NLIN** statement, are as follows:

BOOTSTRAP	requests bootstrap resampling and estimation
BOUNDS	constrains the parameter estimates within specified bounds
BY	specifies variables to define subgroups for the analysis
DER	specifies the first or second partial derivatives
ID	specifies additional variables to add to the output data set
MODEL	defines the relationship between the dependent and independent variables (the mean function)
OUTPUT	creates an output data set containing observation-wise statistics
PARAMETERS	identifies parameters to be estimated and their starting values
PROFILE	identifies parameters to be profiled
<i>Programming Statements</i>	includes assignment statements, ARRAY statements, DO loops, and other program control statements. These are valid SAS expressions that can appear in the DATA step. PROC NLIN enables you to create new variables within the procedure and use them in the nonlinear analysis. These programming statements can appear anywhere in the PROC NLIN code, but new variables must be created before they appear in other statements. The NLIN procedure automatically creates several variables that are also available for use in the analysis. See the section “ Special Variables ” on page 6454 for more information.

The **PROC NLIN**, **PARAMETERS**, and **MODEL** statements are required.

PROC NLIN Statement

PROC NLIN < options > ;

The PROC NLIN statement invokes the NLIN procedure.

Table 81.1 summarizes the *options* available in the PROC NLIN statement. All *options* are subsequently discussed in alphabetical order.

Table 81.1 Summary of Options in PROC NLIN Statement

Option	Description
Options Related to Data Sets	
DATA=	Specifies the input data set
OUTEST=	Specifies the output data set for parameter estimates, covariance matrix, and so on
SAVE	Requests that final estimates be added to the OUTEST= data set
Optimization Options	
BEST=	Limits display of grid search
METHOD=	Chooses the optimization method
MAXITER=	Specifies the maximum number of iterations
MAXSUBIT=	Specifies the maximum number of step halvings
NOHALVE	Allows the objective function to increase between iterations
RHO=	Controls the step-size search
SMETHOD=	Specifies the step-size search method
TAU=	Controls the step-size search
G4	Uses the Moore-Penrose inverse
UNCORRECTEDDF	Does not expense degrees of freedom when bounds are active
SIGSQ=	Specifies the fixed value for residual variance
Singularity and Convergence Criteria	
CONVERGE=	Tunes the convergence criterion
CONVERGEOBJ=	Uses the change in loss function as the convergence criterion and tunes its value
CONVERGEPARM=	Uses the maximum change in parameter estimates as the convergence criterion and tunes its value
SINGULAR=	Tunes the singularity criterion used in matrix inversions
ODS Graphics Options	
PLOTS=	Produces ODS graphical displays
Displayed Output	
HOUGAARD	Adds Hougaard's skewness measure to the "Parameter Estimates" table
BIAS	Adds Box's bias measure to the "Parameter Estimates" table
NOITPRINT	Suppresses the "Iteration History" table
NOPRINT	Suppresses displayed output

Table 81.1 *continued*

Option	Description
LIST	Displays the model program and variable list
LISTALL	Selects the LIST, LISTDEP, LISTDER, and LISTCODE options
LISTCODE	Displays the model program code
LISTDEP	Displays dependencies of model variables
LISTDER	Displays the derivative table
NLINMEASURES	Displays the global nonlinearity measures table
TOTALSS	Adds the uncorrected or corrected total sum of squares to the analysis of variance table
XREF	Displays the cross-reference of variables
Trace Model Execution	
FLOW	Displays execution messages for program statements
PRINT	Displays results of statements in model program
TRACE	Displays results of operations in model program

ALPHA= α

specifies the level of significance α used in the construction of $100(1 - \alpha)\%$ confidence intervals. The value must be strictly between 0 and 1; the default value of $\alpha = 0.05$ results in 95% intervals. This value is used as the default confidence level for limits computed in the “Parameter Estimates” table and with the LCLM, LCL, UCLM, and UCL options in the OUTPUT statement.

BEST= n

requests that PROC NLIN display the residual sums of squares only for the best n combinations of possible starting values from the grid. If you do not specify the BEST= option, PROC NLIN displays the residual sum of squares for every combination of possible parameter starting values.

BIAS

adds Box’s bias and percentage bias measures to the “Parameter Estimates” table (Box 1971). Box’s bias measure, along with Hougaard’s measure of skewness, is used for assessing a parameter estimator’s close-to-linear behavior (Ratkowsky 1983, 1990). Hence, it is useful for identifying problematic parameters (Seber and Wild 1989, sec. 4.7.1). When you specify the BIAS option, Box’s bias measure (Box 1971) and the percentage bias (the bias expressed as a percentage of the least-squares estimator) are added for each parameter to the “Parameter Estimates” table. Ratkowsky (1983, p. 21) takes a percentage bias in excess of 1% to be a good rule of thumb for indicating nonlinear behavior.

See the section “Box’s Measure of Bias” on page 6447 for further details. Example 81.4 shows how to use this measure, along with Hougaard’s measure of skewness, to evaluate changes in the parameterization of a nonlinear model. Computation of the Box’s bias measure requires first and second derivatives. If you do not provide derivatives with the DER statement—and it is recommended that you do not—the analytic derivatives are computed for you.

CONVERGE= c

specifies the convergence criterion for PROC NLIN. For all iterative methods the relative offset convergence measure of Bates and Watts is used by default to determine convergence. This measure

is labeled “R” in the “Estimation Summary” table. The iterations are said to have converged for `CONVERGE=c` if

$$\sqrt{\frac{\mathbf{r}'\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{r}}{\text{LOSS}^{(i)}}} < c$$

where \mathbf{r} is the residual vector and \mathbf{X} is the $(n \times p)$ matrix of first derivatives with respect to the parameters. The default LOSS function is the sum of squared errors (SSE), and $\text{LOSS}^{(i)}$ denotes the value of the loss function at the i th iteration. By default, `CONVERGE=10-5`. The R convergence measure cannot be computed accurately in the special case of a perfect fit (residuals close to zero). When the SSE is less than the value of the `SINGULAR=` criterion, convergence is assumed.

CONVERGEOBJ=c

uses the change in the LOSS function as the convergence criterion and tunes the criterion. The iterations are said to have converged for `CONVERGEOBJ=c` if

$$\frac{|\text{LOSS}^{(i-1)} - \text{LOSS}^{(i)}|}{|\text{LOSS}^{(i-1)} + 10^{-6}|} < c$$

where $\text{LOSS}^{(i)}$ is the LOSS for the i th iteration. The default LOSS function is the sum of squared errors (SSE), the residual sum of squares. The constant c should be a small positive number. For more details about the LOSS function, see the section “[Special Variable Used to Determine Convergence Criteria](#)” on page 6455. For more details about the computational methods in the NLIN procedure, see the section “[Computational Methods](#)” on page 6457.

Note that in SAS 6 the `CONVERGE=` and `CONVERGEOBJ=` options both requested that convergence be tracked by the relative change in the loss function. If you specify the `CONVERGEOBJ=` option in newer releases, the `CONVERGE=` option is disabled. This enables you to track convergence as in SAS 6.

CONVERGEPARM=c

uses the maximum change among parameter estimates as the convergence criterion and tunes the criterion. The iterations are said to have converged for `CONVERGEPARM=c` if

$$\max_j \left(\frac{|\beta_j^{(i-1)} - \beta_j^{(i)}|}{|\beta_j^{(i-1)}|} \right) < c$$

where $\beta_j^{(i)}$ is the value of the j th parameter at the i th iteration.

The default convergence criterion is `CONVERGE`. If you specify `CONVERGEPARM=c`, the maximum change in parameters is used as the convergence criterion. If you specify both the `CONVERGEOBJ=` and `CONVERGEPARM=` options, PROC NLIN continues to iterate until the decrease in LOSS is sufficiently small (as determined by the `CONVERGEOBJ=` option) and the maximum change among the parameters is sufficiently small (as determined by the `CONVERGEPARM=` option).

DATA=SAS-data-set

specifies the input SAS data set to be analyzed by PROC NLIN. If you omit the `DATA=` option, the most recently created SAS data set is used.

FLOW

displays a message for each statement in the model program as it is executed. This debugging option is rarely needed, and it produces large amounts of output.

G4

uses a Moore-Penrose inverse (g_4 -inverse) in parameter estimation. See Kennedy and Gentle (1980) for details.

HOUGAARD

adds Hougaard's measure of skewness to the "Parameter Estimates" table (Hougaard 1982, 1985). The skewness measure is one method of assessing a parameter estimator's close-to-linear behavior in the sense of Ratkowsky (1983, 1990). The behavior of estimators that are close to linear approaches that of least squares estimators in linear models, which are unbiased and have minimum variance. When you specify the HOUGAARD option, the standardized skewness measure of Hougaard (1985) is added for each parameter to the "Parameter Estimates" table. Because of the linkage between nonlinear behavior of a parameter estimator in nonlinear regression and the nonnormality of the estimator's sampling distribution, Ratkowsky (1990, p. 28) provides the following rules to interpret the (standardized) Hougaard skewness measure:

- Values less than 0.1 in absolute value indicate very close-to-linear behavior.
- Values between 0.1 and 0.25 in absolute value indicate reasonably close-to-linear behavior.
- The nonlinear behavior is apparent for absolute values above 0.25 and is considerable for absolute values above 1.

See the section "[Hougaard's Measure of Skewness](#)" on page 6447 for further details. [Example 81.4](#) shows how to use this measure to evaluate changes in the parameterization of a nonlinear model. Computation of the Hougaard skewness measure requires first and second derivatives. If you do not provide derivatives with the [DER](#) statement—and it is recommended that you do not—the analytic derivatives are computed for you. For weighted least squares, the NLIN procedure ignores the weights for computing the Hougaard skewness measure. This can be a strong assumption as the formulation in Hougaard (1985) assumes homoscedastic errors.

LIST

displays the model program and variable lists, including the statements added by macros. Note that the expressions displayed by the LIST option do not necessarily represent the way the expression is actually calculated—because intermediate results for common subexpressions can be reused—but are shown in expanded form. To see how the expression is actually evaluated, use the [LISTCODE](#) option.

LISTALL

selects the [LIST](#), [LISTDEP](#), [LISTDER](#), and [LISTCODE](#) options.

LISTCODE

displays the derivative tables and the compiled model program code. The LISTCODE option is a debugging feature and is not normally needed.

LISTDEP

produces a report that lists, for each variable in the model program, the variables that depend on it and the variables on which it depends.

LISTDER

displays a table of derivatives. The derivatives table lists each nonzero derivative computed for the problem. The derivative listed can be a constant, a variable in the model program, or a special derivative variable created to hold the result of an expression.

MAXITER=*n*

specifies the maximum number *n* of iterations in the optimization process. The default is *n* = 100.

MAXSUBIT=*n*

places a limit on the number of step halvings. The value of MAXSUBIT must be a positive integer and the default value is *n* = 30.

METHOD=GAUSS | MARQUARDT | NEWTON | GRADIENT

specifies the iterative method employed by the NLIN procedure in solving the nonlinear least squares problem. The GAUSS, MARQUARDT, and NEWTON methods are more robust than the GRADIENT method. If you omit the METHOD= option, METHOD=GAUSS is used. See the section “[Computational Methods](#)” on page 6457 for more information.

NLINMEASURES

displays the global nonlinearity measures table. These measures include the maximum intrinsic and parameter-effects curvatures (Bates and Watts 1980), the root mean square (RMS) intrinsic and parameter-effects curvatures and the critical curvature value (Bates and Watts 1980). In addition, the variances of the ordinary and projected residuals are included. According to Bates and Watts (1980), both intrinsic and parameter-effects curvatures are deemed negligible if they are less than the critical curvature value. This critical value is given by $1/(\sqrt{F})$ where $F = F(p, n - p; \alpha)$. The value $1/\sqrt{F}$ can be considered as the radius of curvature of the 100(1 - α) percent confidence region (Bates and Watts 1980). For weighted least squares, the NLIN procedure ignores the weights for computing the curvature measures. This can be a strong assumption as the original derivation in Bates and Watts (1980) assumes homoscedastic errors.

NOITPRINT

suppresses the display of the “Iteration History” table.

NOHALVE

removes the restriction that the objective value must decrease at every iteration. Step halving is still used to satisfy [BOUNDS](#) and to ensure that the number of observations that can be evaluated does not decrease. The NOHALVE option can be useful in weighted nonlinear least squares problems where the weights depend on the parameters, such as in iteratively reweighted least squares (IRLS) fitting. See [Example 81.2](#) for an application of IRLS fitting.

NOPRINT

suppresses the display of the output. Note that this option temporarily disables the Output Delivery System (ODS). For more information, see Chapter 20, “[Using the Output Delivery System](#).”

OUTEST=SAS-data-set

specifies an output data set that contains the parameter estimates produced at each iteration. See the section “[Output Data Sets](#)” for details. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Language Reference: Concepts*.

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

controls most of the plots that are produced through ODS Graphics (other plots are controlled by the [BOOTSTRAP](#) and [PROFILE](#) statements). When you specify only one *plot-request*, you can omit the parentheses around it. Here are some examples:

```
plots
plots          = none
plots          = diagnostics(unpack)
plots          = fit(stats=none)
plots          = residuals(residualtype=proj unpack smooth)
plots(stats=all) = (diagnostics(stats=(maxincurv maxpecurv)) fit)
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc nlin plots=diagnostics(stats=all);
  model y = alpha - beta*(gamma**x);
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section “[Enabling and Disabling ODS Graphics](#)” on page 609 in Chapter 21, “[Statistical Graphics Using ODS](#).”

If ODS Graphics is enabled and if you specify the PLOTS option without any *global-plot-option* or *plot-requests*, PROC NLIN produces the plots listed in [Table 81.2](#) with the default set of statistics and options. If you do not specify the PLOTS option, PROC NLIN does not produce any of these graphs.

Table 81.2 Graphs Produced When the PLOTS Option Is Specified

Plot	Conditional On
ContourFitPlot	Model with two regressors
FitDiagnosticsPanel	Unconditional
FitPlot	Model with one regressor
LeveragePlot	Unconditional
LocalInfluencePlot	Unconditional
ResidualPanel	Unconditional

You can request additional plots by specifying *plot-requests*. For a listing of all the plots that PROC NLIN produces, see the section “[ODS Graphics](#)” on page 6469. Each *global-plot-option* applies to all plots that are generated by the NLIN procedure except for plots that are controlled by the [BOOTSTRAP](#) and [PROFILE](#) statements. The *global-plot-option* can be overridden by a specific *option* after a *plot-request*.

The following *global-plot-options* are available:

RESIDUALTYPE=RAW | PROJ | BOTH

specifies the residual type to be plotted in the fit diagnostics and residual plots. RESIDUALTYPE=RAW requests that only the ordinary residuals be included in the plots; RESIDUALTYPE=PROJ sets the choice to projected residuals. By default, both residual types are included, which can also be effected by setting RESIDUALTYPE=BOTH. See the section “[Residuals in Nonlinear Regression](#)” on page 6450 for details about the properties of ordinary and projected residuals in nonlinear regression.

STATS=ALL | DEFAULT | NONE | (*plot-statistics*)

requests the statistics to be included in all plots, except the ResidualPlots and the unpacked diagnostics plots. [Table 81.3](#) lists the statistics that you can request. STATS=ALL requests all these statistics, STATS=NONE suppresses all statistics, and STATS=DEFAULT selects the default statistics. You request statistics in addition to the default set by including the keyword DEFAULT in the *plot-statistics* list.

Table 81.3 Statistics Available in Plots

Keyword	Default	Description
DEFAULT		All default statistics
MAXINCURV		Maximum intrinsic curvature
MAXPECURV		Maximum parameter-effects curvature
MSE	x	Mean squared error, estimated or set by the SIGSQ option
NOBS	x	Number of observations used
NPARM	x	Number of parameters in the model
PVAR	x	Estimated variance of the projected residuals
RMSINCURV		Root mean square intrinsic curvature
RMSPECURV		Root mean square parameter-effects curvature
VAR	x	Estimated variance of the ordinary residuals

Along with the maximum intrinsic and parameter-effects curvatures, the critical curvature (CURVCRIT) value, $1/\sqrt{F}$ where $F = F(p, n - p; \alpha)$, is also displayed. You do not need to specify any option for it. See the section “[Relative Curvature Measures of Nonlinearity](#)” on page 6448 for details about curvature measures of nonlinearity.

UNPACK

suppresses paneling.

You can specify the following *plot-requests* in the PLOTS= option:

ALL

produces all appropriate plots.

NONE

suppresses all plots.

DIAGNOSTICS < (*diagnostics-options*) >

produces a summary panel of fit diagnostics, leverage plots, and local-influence plots. The fit diagnostics panel includes:

- histogram of the ordinary residuals
- histogram of the projected residuals
- response variable values versus the predicted values
- expectation or mean of the ordinary residuals versus the predicted values
- ordinary and projected residuals versus the predicted values
- standardized ordinary and projected residuals versus the predicted values
- standardized ordinary and projected residuals versus the tangential leverage
- standardized ordinary and projected residuals versus the Jacobian leverage
- box plot of the ordinary and projected residuals if you specify the STATS=NONE suboption

The leverage and local influence plots are produced separately. The leverage plot is an index plot of the tangential and Jacobian leverages (by observation), and the local-influence plot contains the local influence by observation for a perturbation of the response variable. See the sections “[Leverage in Nonlinear Regression](#)” on page 6449 and “[Local Influence in Nonlinear Regression](#)” on page 6449 for a some details about leverages and local-influence in nonlinear regression.

You can specify the following *diagnostics-options*:

RESIDUALTYPE=RAW | PROJ | BOTH

specifies the residual type to be plotted in the panel. See the RESIDUALTYPE= *global-plot-option* for details. This *diagnostics-option* overrides the PLOTS RESIDUALTYPE *global-plot-option*. Only the plots that overlay both ordinary and projected residuals in the same plot are affected by this option.

LEVERAGETYPE=TAN | JAC | BOTH

specifies the leverage type to be plotted in the leverage plot. LEVERAGETYPE=TAN specifies that only the tangential leverage be included in the leverage plot, and LEVERAGETYPE=JAC specifies that only the Jacobian leverage be included. By default, both are displayed in the leverage plot. The same result can be effected by setting LEVERAGETYPE=BOTH. Only the leverage plot is affected by this option.

LABELOBS

specifies that the leverage and local-influence plots be labeled with the observation number. Only these two plots are affected by this option.

STATS=stats-options

determines which statistics are included in the panel. See the **STATS= global-plot-option** for details. This *diagnostics-option* overrides the **PLOTS STATS global-plot-option**.

UNPACK

produces the plots in the diagnostics panel as individual plots. The statistics panel is not included in the individual plots, even if **STATS= global-plot-option** or **STATS= diagnostics-option** or both are specified.

FITPLOT | FIT <(fit-options)>

produces, depending on the number of regressors, a scatter or contour fit plot. For a single-regressor model, a scatter plot of the data overlaid with the regression curve, confidence, and prediction bands is produced. For two-regressor models, a contour fit plot of the model with overlaid data is produced. If the model contains more than two regressors, no fit plot is produced.

You can specify the following *fit-options*:

NOCLI

suppresses the prediction limits for single-regressor models.

NOCLM

suppresses the confidence limits for single-regressor models.

NOLIMITS

suppresses the confidence and prediction limits for single-regressor models.

OBS=GRADIENT | NONE | OUTLINE | OUTLINEGRADIENT

controls how the observations are displayed. The *suboptions* are as follows:

GRADIENT

specifies that observations be displayed as circles colored by the observed response. The same color gradient is used to display the fitted surface and the observations. Observations for which the predicted response is close to the observed response have similar colors—the greater the contrast between the color of an observation and the surface, the larger the residual is at that point. **OBS=GRADIENT** is the default.

NONE

suppresses the observations.

OUTLINE

specifies that observations be displayed as circles with a border but with a completely transparent fill.

OUTLINEGRADIENT

is the same as **OBS=GRADIENT** except that a border is shown around each observation. This option is useful for identifying the location observations for which the residuals are small, because at these points the color of the observations and the color of the surface are indistinguishable.

CONTLEG

specifies that a continuous legend be included in the contour fit plot of a two-regressor model.

STATS=stats-options

determines which model fit statistics are included in the panel. See the STATS= *global-plot-option* for details. This *fit-option* overrides the PLOTS STATS *global-plot-option*.

RESIDUALS <(residual-options)>

produces panels of the ordinary and projected residuals versus the regressors in the model. Each panel contains at most six plots, and multiple panels are used in the case where there are more than six regressors in the model.

The following *residual-options* are available:

RESIDUALTYPE=RAW | PROJ | BOTH

specifies the residual type to be plotted in the panel. See the RESIDUALTYPE= *global-plot-option* for details. This *residual-option* overrides the PLOTS RESIDUALTYPE *global-plot-option*.

SMOOTH

requests a nonparametric smooth of the residuals for each regressor. Each nonparametric fit is a loess fit that uses local linear polynomials, linear interpolation, and a smoothing parameter selected that yields a local minimum of the corrected Akaike information criterion (AICC). See Chapter 71, “[The LOESS Procedure](#),” for details.

UNPACK

suppresses paneling.

PRINT

displays the result of each statement in the program as it is executed. This option is a debugging feature that produces large amounts of output and is normally not needed.

RHO=value

specifies a value that controls the step-size search. By default RHO=0.1, except when **METHOD=MARQUARDT**. In that case, RHO=10. See the section “[Step-Size Search](#)” on page 6462 for more details.

SAVE

specifies that, when the iteration limit is exceeded, the parameter estimates from the final iteration be output to the **OUTEST=** data set. These parameter estimates are associated with the observation for which **_TYPE_**="FINAL". If you omit the SAVE option, the parameter estimates from the final iteration are not output to the data set unless convergence has been attained.

SIGSQ=value

specifies a value to use as the estimate of the residual variance in lieu of the estimated mean-squared error. This value is used in computing the standard errors of the estimates. Fixing the value of the residual variance can be useful, for example, in maximum likelihood estimation.

SINGULAR=s

specifies the singularity criterion, **s**, which is the absolute magnitude of the smallest pivot value allowed when inverting the Hessian or the approximation to the Hessian. The default value is 1E4 times the machine epsilon; this product is approximately 1E-12 on most computers.

SMETHOD=HALVE | GOLDEN | CUBIC

specifies the step-size search method. The default is SMETHOD=HALVE. See the section “[Step-Size Search](#)” on page 6462 for details.

TAU=value

specifies a value that is used to control the step-size search. The default is TAU=1, except when METHOD=MARQUARDT. In that case the default is TAU=0.01. See the section “[Step-Size Search](#)” on page 6462 for details.

TOTALSS

adds to the analysis of variance table the uncorrected total sum of squares in models that have an (implied) intercept, and adds the corrected total sum of squares in models that do not have an (implied) intercept.

TRACE

displays the result of each operation in each statement in the model program as it is executed, in addition to the information displayed by the [FLOW](#) and [PRINT](#) options. This debugging option is needed very rarely, and it produces even more output than the [FLOW](#) and [PRINT](#) options.

XREF

displays a cross-reference of the variables in the model program showing where each variable is referenced or given a value. The XREF listing does not include derivative variables.

UNCORRECTEDDF

specifies that no degrees of freedom be lost when a bound is active. When the UNCORRECTEDDF option is not specified, an active bound is treated as if a restriction were applied to the set of parameters, so one parameter degree of freedom is deducted.

BOOTSTRAP Statement

BOOTSTRAP *</ options>* ;

A BOOTSTRAP statement requests bootstrap estimation of confidence intervals, the covariance matrix, and the correlation matrix of parameter estimates. To produce the plots that are controlled by the [BOOTSTRAP](#) statement, ODS Graphics must be enabled. If the main data set contains observations that PROC NLIN deems unusable, the procedure issues a message that these observations are excluded from the bootstrap resampling. PROC NLIN ignores the BOOTSTRAP statement for nonconvergent and singular models.

[Table 81.4](#) summarizes the *options* available in the BOOTSTRAP statement.

Table 81.4 Summary of Options in BOOTSTRAP Statement

Option	Description
BOOTCI	Produces bootstrap confidence intervals of the parameters
BOOTCORR	Produces a bootstrap correlation matrix estimate table
BOOTCOV	Produces a bootstrap covariance matrix estimate table
BOOTDATA=	Specifies the bootstrap output data set
BOOTPLOTS	Produces plots of the bootstrap parameter estimates
DGP=	Specifies the bootstrap data generating process (DGP)

Table 81.4 *continued*

Option	Description
NSAMPLES=	Specifies the number of bootstrap sample data sets (replicates)
SEED=	Provides the seed that initializes the random number stream

BOOTCI <(BC | NORMAL | PERC | ALL)>

produces bootstrap-based confidence intervals for the parameters and adds columns that contain these values to the “Parameter Estimates” table. You can specify the following types of bootstrap confidence intervals:

BC	produces bias-corrected confidence intervals.
NORMAL	produces confidence intervals based on the assumption that bootstrap parameter estimates follow a normal distribution.
PERC	produces percentile-based confidence intervals.
ALL	produces all three confidence intervals.

The **ALPHA=** option in the **PROC NLIN** statement sets the level of significance that is used in constructing these bootstrap confidence intervals. By default, without the **BOOTCI** option, **PROC NLIN** produces bias-corrected (BC) confidence intervals and adds a column that contains the standard deviation of the bootstrap parameter estimates to the “Parameter Estimates” table. For more information, see the section “[Bootstrap Resampling and Estimation](#)” on page 6451.

BOOTCORR

produces the “Bootstrap Correlation Matrix” table, which contains a bootstrap estimate of the correlation matrix of the parameter estimates. For more information, see the section “[Bootstrap Resampling and Estimation](#)” on page 6451.

BOOTCOV

produces the “Bootstrap Covariance Matrix” table, which contains a bootstrap estimate of the covariance matrix of the parameter estimates. For more information, see the section “[Bootstrap Resampling and Estimation](#)” on page 6451.

BOOTDATA=SAS-data-set

specifies the SAS data set that contains the bootstrap sample data when you use a **BOOTSTRAP** statement. For more information about this data set, see the section “[Output Data Sets](#)” on page 6462.

BOOTPLOTS <(HIST | SCATTER | ALL)>

produces ODS graphics of bootstrap parameter estimates. You can specify the following types of plots:

HIST	produces histograms of the bootstrap parameter estimates.
SCATTER	produces pairwise scatter plots of the bootstrap parameter estimates.
ALL	produces both plots.

By default, if ODS Graphics is enabled, **PROC NLIN** produces histograms of the bootstrap parameter estimates.

DGP=RESIDUAL <(*scaling-option*)> | **WILD**

specifies the bootstrap data generating process (DGP). DGP=RESIDUAL requests the residual bootstrap, and DGP=WILD requests the wild bootstrap. The *scaling-option* determines the type of residual scaling to be performed for DGP=RESIDUAL.

Table 81.5 Scaling Options Available for DGP=RESIDUAL

Keyword	Description
ADJSSE	Simple uniform scaling
JAC	Scaling based on Jacobian leverage
RAW	No scaling
TAN	Scaling based on tangential leverage

By default, if the **BOOTSTRAP** statement is specified with no **DGP=** option or if no *scaling-option* is specified for DGP=RESIDUAL, PROC NLIN performs a residual bootstrap with simple scaling (ADJSSE) for unweighted least squares and a wild bootstrap (WILD) for weighted least squares. For more information, see the section “[Bootstrap Resampling and Estimation](#)” on page 6451.

NSAMPLES=*n*

specifies the number of bootstrap sample data sets (replicates). By default, NSAMPLES=1000. For more information, see the section “[Bootstrap Resampling and Estimation](#)” on page 6451.

SEED=*n*

provides the seed that initializes the random number stream for generating the bootstrap sample data sets (replicates). If you do not specify the **SEED=** value or if you specify a value less than or equal to 0, the seed is generated from reading the time of day from the computer’s clock. The largest possible value for the seed is $2^{31} - 1$. The **_SEED_** in the data set that is produced by the **BOOTDATA=** option contains the value that is used as the initial seed for a particular replicate.

You can use the **SYSRANDOM** and **SYSRANEND** macro variables after a PROC NLIN run to query the initial and final seed values. However, using the final seed value as the starting seed for a subsequent analysis does not continue the random number stream where the previous analysis left off. The **SYSRANEND** macro variable provides a mechanism to pass on seed values to ensure that the sequence of random numbers is the same every time you run an entire program.

BOUNDS Statement

BOUNDS *inequality* <, ..., *inequality*> ;

The **BOUNDS** statement restricts the parameter estimates so that they lie within specified regions. In each **BOUNDS** statement, you can specify a series of boundary values separated by commas. The series of bounds is applied simultaneously. Each boundary specification consists of a list of parameters, an inequality comparison operator, and a value. In a single-bounded expression, these three elements follow one another in the order described. The following are examples of valid single-bounded expressions:

```

bounds a1-a10 <= 20;
bounds c > 30;
bounds a b c > 0;

```

Multiple-bounded expressions are also permitted. For example:

```

bounds 0 <= B<= 10;
bounds 15 < x1 <= 30;
bounds r <= s <= p < q;

```

If you need to restrict an expression involving several parameters (for example, $\alpha + \beta < 1$), you can reparameterize the model so that the expression becomes a parameter or so that the boundary constraint can be expressed as a simple relationship between two parameters. For example, the boundary constraint $\alpha + \beta < 1$ in the model

```
model y = alpha + beta*x;
```

can be achieved by parameterizing $\theta = 1 - \beta$ as follows:

```

bounds alpha < theta;
model y = alpha + (1-theta)*x;

```

Starting with SAS 7.01, Lagrange multipliers are reported for all bounds that are enforced (active) when the estimation terminates. In the “Parameter Estimates” table, the Lagrange multiplier estimates are identified with names *Bound1*, *Bound2*, and so forth. An active bound is treated as if a restriction were applied to the set of parameters so that one parameter degree of freedom is deducted. You can use the [UNCORRECTEDDF](#) option to prevent the loss of degrees of freedom when bounds are active.

BY Statement

BY variables ;

You can specify a BY statement with PROC NLIN to obtain separate analyses of observations in groups that are 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 you specify more than one BY statement, only the last one specified is used.

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 NOTSORTED or DESCENDING option in the BY statement for the NLIN 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 (in Base SAS software).

For more information about BY-group processing, see the discussion in *SAS Language Reference: Concepts*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CONTROL Statement

CONTROL *variable* <=*values*> <... *variable* <=*values*>> ;

The CONTROL statement declares control variables and specifies their values. A control variable is like a retained variable (see the section “[RETAIN Statement](#)” on page 6443) except that it is retained *across* iterations, and the derivative of the model with respect to a control variable is always zero.

DER Statement

DER. *parameter*=*expression* ;

DER. *parameter.parameter*=*expression* ;

The DER statement specifies first or second partial derivatives. By default, analytical derivatives are automatically computed. However, you can specify the derivatives yourself by using the DER.parm syntax. Use the first form shown to specify first partial derivatives, and use the second form to specify second partial derivatives. Note that the DER.parm syntax is retained for backward compatibility. The automatic analytical derivatives are, in general, a better choice. For additional information about automatic analytical derivatives, see the section “[Automatic Derivatives](#)” on page 6445.

For most of the computational methods, you need only specify the first partial derivative with respect to each parameter to be estimated. For the NEWTON method, specify both the first and the second derivatives. If any needed derivatives are not specified, they are automatically computed.

The expression can be an algebraic representation of the partial derivative of the expression in the [MODEL](#) statement with respect to the parameter or parameters that appear on the left side of the [DER](#) statement. Numerical derivatives can also be used. The expression in the DER statement must conform to the rules for a valid SAS expression, and it can include any quantities that the [MODEL](#) statement expression contains.

ID Statement

ID *variables* ;

The ID statement specifies additional variables to place in the output data set created by the [OUTPUT](#) statement. Any variable on the left side of any assignment statement is eligible. Also, the special variables created by the procedure can be specified. Variables in the input data set do not need to be specified in the ID statement since they are automatically included in the output data set.

MODEL Statement

MODEL *dependent* = *expression* ;

The MODEL statement defines the prediction equation by declaring the dependent variable and defining an expression that evaluates predicted values. The expression can be any valid SAS expression that yields a numeric result. The expression can include parameter names, variables in the data set, and variables created

by programming statements in the NLIN procedure. Any operators or functions that can be used in a DATA step can also be used in the MODEL statement.

A statement such as

```
model y=expression;
```

is translated into the form

```
model.y=expression;
```

using the compound variable name `model.y` to hold the predicted value. You can use this assignment directly as an alternative to the MODEL statement. Either a MODEL statement or an assignment to a compound variable such as `model.y` must appear.

OUTPUT Statement

OUTPUT **OUT**=SAS-data-set keyword=names <... keyword=names> </ options> ;

The OUTPUT statement specifies an output data set to contain statistics calculated for each observation. For each statistic, specify the *keyword*, an equal sign, and a variable name for the statistic in the output data set. All of the names appearing in the OUTPUT statement must be valid SAS names, and none of the new variable names can match a variable already existing in the data set to which PROC NLIN is applied.

If an observation includes a missing value for one of the independent variables, both the predicted value and the residual value are missing for that observation. If the iterations fail to converge, all the values of all the variables named in the OUTPUT statement are missing values.

Table 81.6 summarizes the *options* available in the OUTPUT statement.

Table 81.6 OUTPUT Statement Options

Option	Description
Output data set and OUTPUT statement options	
ALPHA=	Specifies the level of significance α
DER	Saves the first derivatives of the model
OUT=	Specifies the output data set
Keyword options	
H=	Specifies the tangential leverage
J=	Specifies the Jacobian leverage
L95=	Specifies the lower bound of an approximate 95% prediction interval
L95M=	Specifies the lower bound of an approximate 95% confidence interval for the mean
LCL=	Specifies the lower bound of an approximate $100(1-\alpha)\%$ prediction interval
LCLM=	Specifies the lower bound of an approximate $100(1-\alpha)\%$ confidence interval for the mean
LMAX=	Specifies the direction of maximum local influence
PARMS=	Specifies the parameter estimates

Table 81.6 continued

Option	Description
PREDICTED=	Specifies the predicted values
PROJRES=	Specifies the projected residuals
PROJSTUDENT=	Specifies the standardized projected residuals
RESEXPEC=	Specifies the mean of the residuals
RESIDUAL=	Specifies the residuals
SSE=	Specifies the residual sum of squares
STDI=	Specifies the standard error of the individual predicted value
STDP=	Specifies the standard error of the mean predicted value
STDR=	Specifies the standard error of the residual
STUDENT=	Specifies the standardized residuals
U95=	Specifies the upper bound of an approximate 95% prediction interval
U95M=	Specifies the upper bound of an approximate 95% confidence interval for the mean
UCL=	Specifies the upper bound of an approximate $100(1-\alpha)\%$ prediction interval
UCLM=	Specifies the upper bound of an approximate $100(1-\alpha)\%$ confidence interval for the mean
WEIGHT=	Specifies the special variable <code>_WEIGHT_</code>

You can specify the following *options* in the OUTPUT statement. For a description of computational formulas, see Chapter 4, “[Introduction to Regression Procedures](#).”

OUT=SAS-data-set

specifies the SAS data set to be created by PROC NLIN when an OUTPUT statement is included. The new data set includes the variables in the input data set. Also included are any **ID** variables specified in the **ID** statement, plus new variables with names that are specified in the OUTPUT statement.

The following values can be calculated and output to the new data set.

H=name

specifies a variable that contains the tangential leverage. See the section “[Leverage in Nonlinear Regression](#)” on page 6449 for details.

J=name

specifies a variable that contains the Jacobian leverage. See the section “[Leverage in Nonlinear Regression](#)” on page 6449 for details.

L95=name

specifies a variable that contains the lower bound of an approximate 95% confidence interval for an individual prediction. This includes the variance of the error as well as the variance of the parameter estimates. See also the description for the **U95=** option later in this section.

L95M=name

specifies a variable that contains the lower bound of an approximate 95% confidence interval for the expected value (mean). See also the description for the **U95M=** option later in this section.

LCL=*name*

specifies a variable that contains the lower bound of an approximate $100(1 - \alpha)\%$ confidence interval for an individual prediction. The α level is equal to the value of the **ALPHA=** option in the OUTPUT statement or, if this option is not specified, to the value of the **ALPHA=** option in the **PROC NLIN** statement. If neither of these options is specified, then $\alpha = 0.05$ by default, resulting in a lower bound for an approximate 95% confidence interval. For the corresponding upper bound, see the **UCL** keyword.

LCLM=*name*

specifies a variable that contains the lower bound of an approximate $100(1 - \alpha)\%$ confidence interval for the expected value (mean). The α level is equal to the value of the **ALPHA=** option in the OUTPUT statement or, if this option is not specified, to the value of the **ALPHA=** option in the **PROC NLIN** statement. If neither of these options is specified, then $\alpha = 0.05$ by default, resulting in a lower bound for an approximate 95% confidence interval. For the corresponding lower bound, see the **UCLM** keyword.

LMAX=*name*

specifies a variable that contains the direction of maximum local influence of an additive perturbation of the response variable. See the section “[Local Influence in Nonlinear Regression](#)” on page 6449 for details.

PARMS=*names*

specifies variables in the output data set that contains parameter estimates. These can be the same variable names that are listed in the **PARAMETERS** statement; however, you can choose new names for the parameters identified in the sequence from the parameter estimates table. A note in the log indicates which variable in the output data set is associated with which parameter name. Note that, for each of these new variables, the values are the same for every observation in the new data set.

PREDICTED=*name***P=***name*

specifies a variable in the output data set that contains the predicted values of the dependent variable.

PROJRES=*name*

specifies a variable that contains the projected residuals obtained by projecting the residuals (ordinary residuals) into the null space of $(X|H)$. For the ordinary residuals, see the **RESIDUAL=** option later in this section. The section “[Residuals in Nonlinear Regression](#)” on page 6450 describes the statistical properties of projected residuals in nonlinear regression.

PROJSTUDENT=*name*

specifies a variable that contains the standardized projected residuals. See the section “[Residuals in Nonlinear Regression](#)” on page 6450 for details and the **STUDENT=** option later in this section.

RESEXPEC=*name*

specifies a variable that contains the mean of the residuals. In contrast to linear regressions where the mean of the residuals is zero, in nonlinear regression the residuals have a nonzero mean value and show a negative covariance with the mean response. See the section “[Residuals in Nonlinear Regression](#)” on page 6450 for details.

RESIDUAL=*name***R=***name*

specifies a variable in the output data set that contains the residuals. See also the description of **PROJRES=** option stated previously in this section and the section “[Residuals in Nonlinear Regression](#)” on page 6450 for the statistical properties of residuals and projected residuals.

SSE=*name***ESS=***name*

specifies a variable in the output data set that contains the residual sum of squares finally determined by the procedure. The value of the variable is the same for every observation in the new data set.

STDI=*name*

specifies a variable that contains the standard error of the individual predicted value.

STDP=*name*

specifies a variable that contains the standard error of the mean predicted value.

STDR=*name*

specifies a variable that contains the standard error of the residual.

STUDENT=*name*

specifies a variable that contains the standardized residuals. These are residuals divided by their estimated standard deviation. See the **PROJSTUDENT=** option defined previously in this section and the section “[Residuals in Nonlinear Regression](#)” on page 6450 for the statistical properties of residuals and projected residuals.

U95=*name*

specifies a variable that contains the upper bound of an approximate 95% confidence interval for an individual prediction. See also the description for the **L95=** option.

U95M=*name*

specifies a variable that contains the upper bound of an approximate 95% confidence interval for the expected value (mean). See also the description for the **L95M=** option.

UCL=*name*

specifies a variable that contains the upper bound of an approximate $100(1 - \alpha)\%$ confidence interval an individual prediction. The α level is equal to the value of the **ALPHA=** option in the **OUTPUT** statement or, if this option is not specified, to the value of the **ALPHA=** option in the **PROC NLIN** statement. If neither of these options is specified, then $\alpha = 0.05$ by default, resulting in an upper bound for an approximate 95% confidence interval. For the corresponding lower bound, see the **LCL** keyword.

UCLM=*name*

specifies a variable that contains the upper bound of an approximate $100(1 - \alpha)\%$ confidence interval for the expected value (mean). The α level is equal to the value of the **ALPHA=** option in the **OUTPUT** statement or, if this option is not specified, to the value of the **ALPHA=** option in the **PROC NLIN** statement. If neither of these options is specified, then $\alpha = 0.05$ by default, resulting in an upper bound for an approximate 95% confidence interval. For the corresponding lower bound, see the **LCLM** keyword.

WEIGHT=name

specifies a variable in the output data set that contains values of the special variable `_WEIGHT_`.

You can specify the following *options* in the OUTPUT statement after a slash (/) :

ALPHA= α

specifies the level of significance α for $100(1 - \alpha)\%$ confidence intervals. By default, α is equal to the value of the **ALPHA=** option in the **PROC NLIN** statement or 0.05 if that option is not specified. You can supply values that are strictly between 0 and 1.

DER

saves the first derivatives of the model with respect to the parameters to the OUTPUT data set. The derivatives are named `DER_parmname`, where “parmname” is the name of the model parameter in your NLIN statements. You can use the DER option to extract the $\mathbf{X} = \partial f / \partial \boldsymbol{\beta}$ matrix into a SAS data set. For example, the following statements create the data set `nlinX`, which contains the **X** matrix:

```
proc nlin;
  parms theta1=155 theta3=0.04;
  model V = theta1*c / (theta3 + c);
  output out=nlinout / der;
run;

data nlinX;
  set nlinout (keep=DER_);
run;
```

The derivatives are evaluated at the final parameter estimates.

PARAMETERS Statement

PARAMETERS < *parameter-specification* > < , ... , *parameter-specification* >
< / **PDATA=SAS-data-set** > ;

PARMS < *parameter-specification* > < , ... , *parameter-specification* >
< / **PDATA=SAS-data-set** > ;

A **PARAMETERS** (or **PARMS**) statement is required. The purpose of this statement is to provide starting values for the NLIN procedure. You can provide values that define a point in the parameter space or a set of points.

All parameters must be assigned starting values through the **PARAMETERS** statement. The NLIN procedure does not assign default starting values to parameters in your model that do not appear in the **PARAMETERS** statement. However, it is not necessary to supply parameters and starting values explicitly through a *parameter-specification*. Starting values can also be provided through a data set. The names assigned to parameters must be valid SAS names and must not coincide with names of variables in the input data set (see the **DATA=** option in the **PROC NLIN** statement). Parameters that are assigned starting values through the **PARAMETERS** statement can be omitted from the estimation, for example, if the expression in the **MODEL** statement does not depend on them.

Assigning Starting Values with Parameter-Specification

A *parameter-specification* has the general form

name = *value-list*

where *name* identifies the parameter and *value-list* provides the set of starting values for the parameter.

Very often, the *value-list* contains only a single value, but more general and flexible list specifications are possible:

<i>m</i>	a single value
<i>m1, m2, ..., mn</i>	several values
<i>m TO n</i>	a sequence where <i>m</i> equals the starting value, <i>n</i> equals the ending value, and the increment equals 1
<i>m TO n BY i</i>	a sequence where <i>m</i> equals the starting value, <i>n</i> equals the ending value, and the increment is <i>i</i>
<i>m1, m2 TO m3</i>	mixed values and sequences

When you specify more than one value for a parameter, PROC NLIN sorts the values in ascending sequence and removes duplicate values from the parameter list before forming the grid for the parameter search. If you specify several values for each parameter, PROC NLIN evaluates the model at each point on the grid. The iterations then commence from the point on the grid that yields the smallest objective function value.

For example, the following PARMS statement specifies five parameters and sets their possible starting values as shown in the table:

```
parms  b0 = 0
       b1 = 4 to 8
       b2 = 0 to .6 by .2
       b3 = 1, 10, 100
       b4 = 0, .5, 1 to 4;
```

Possible starting values				
B0	B1	B2	B3	B4
0	4	0.0	1	0.0
	5	0.2	10	0.5
	6	0.4	100	1.0
	7	0.6		2.0
	8			3.0
				4.0

Residual sums of squares are calculated for each of the $1 \times 5 \times 4 \times 3 \times 6 = 360$ combinations of possible starting values. (Specifying a large grid of values can take considerable computing time.)

If you specify a starting value with a *parameter-specification*, any starting values provided for this parameter through the PDATA= data set are ignored. The *parameter-specification* overrides the information in the PDATA= data set. When you specify a **BY** statement, the same *parameter-specification* is applied in each

BY group. To vary starting values with BY groups, use the PDATA= option in the PARAMETERS statement as described in the following paragraphs.

Assigning Starting Values from a SAS Data Set

The PDATA= option in the PARAMETERS statement enables you to assign starting values for parameters through a SAS data set. The data set must contain at least two variables that identify the parameter and contain starting values, respectively. The *character* variable identifying the parameters must be named Parameter or Parm. The *numeric* variable containing the starting value must be named Estimate or Est. This enables you, for example, to use the contents of the ParameterEstimates table from one PROC NLIN run to supply starting values for a subsequent run, as in the following example:

```
proc nlin data=FirstData;
  parameters alpha=100 beta=3 gamma=4;
  < other NLIN statements >
  model y = ... ;
  ods output ParameterEstimates=pest;
run;

proc nlin data=SecondData;
  parameters / pdata=pest;
  Switch = 1/(1+gamma*exp(beta*log(dose)));
  model y = alpha*Switch;
run;
```

You can specify multiple values for a parameter in the PDATA= data set, and the parameters can appear in any order. The starting values are collected by parameter and arranged in ascending order, and duplicate values are removed. The parameter names in the PDATA= data set are not case sensitive. For example, the following DATA step defines starting values for three parameters and a starting grid with $1 \times 3 \times 1 = 3$ points:

```
data test;
  input Parameter $ Estimate;
  datalines;
alpha 100
BETA 4
beta 4.1
beta 4.2
beta 4.1
gamma 30
;
```

If starting values for a parameter are provided through the PDATA= data set and through an explicit *parameter-specification*, the latter takes precedence.

When you specify a BY statement, you can control whether the same starting values are applied to each BY group or whether the starting values are varied. If the BY variables are not present in the PDATA= data set, the entire contents of the PDATA= data set are applied in each BY group. If the BY variables are present in the PDATA= data set, then BY-group-specific starting values are assigned.

PROFILE Statement

PROFILE <...parameter> </options> ;

A PROFILE statement selects parameters for profiling to assess their nonlinear characteristics. It can also gauge the influence of each observation on the selected parameters. If none of the parameters is specified in the PROFILE statement, all parameters are selected for profiling. To produce the plots that are controlled by the PROFILE statement, ODS Graphics must be enabled. If there are observations in the main data set that PROC NLIN deems unusable, the procedure issues a message stating that these observations are excluded from the parameter profiling. PROC NLIN ignores PROFILE statements for nonconvergent and singular models.

Table 81.8 summarizes the *options* available in the PROFILE statement.

Table 81.8 Summary of Options in PROFILE Statement

Option	Description
ALL	Produces all profiling related plots and statistics
CONFCURV	Produces confidence curves for the listed parameters
JACKKNIFE	Produces plots of the absolute relative percentage change by observation for the listed parameters
PROFDATA=	Specifies the profile output data set
RANGE=	Specifies a profiling range
TPLOT	Produces profile <i>t</i> plots for the listed parameters

ALL

produces all appropriate plots for the parameters that are listed in the PROFILE statement. This option overrides all other *options*. If the PROFDATA= option is specified, the values of the corresponding statistics are written to the data set.

CONFCURV

produces confidence curves for each parameter listed in the PROFILE statement. If the PROFDATA= option is specified, the values of the corresponding statistics are written to the data set. See the section “Profiling Parameters and Assessing the Influence of Observations on Parameter Estimates” on page 6451 for details.

JACKKNIFE

produces, for each parameter listed in the PROFILE statement, plots of absolute relative percentage change in the value of the parameter that results from a fit to a jackknife-resampled datum versus observation number. The jackknife resampling is performed with a leave-one-out method. If the PROFDATA= option is specified, the values of the corresponding statistics are written to the data set. See the section “Profiling Parameters and Assessing the Influence of Observations on Parameter Estimates” on page 6451 for details.

PROFDATA=SAS-data-set

specifies the SAS data set to be created by PROC NLIN when a PROFILE statement is included. See the section “Output Data Sets” on page 6462 for details about this data set.

RANGE= γ_f to γ_t by γ_b

specifies a profiling range for each parameter listed in the **PROFILE** statement. γ_f , γ_t and γ_b represent the starting, final, and consecutive incremental values of the range in terms of the standard errors of the parameters to be profiled. If you do not specify the **RANGE=** option, PROC NLIN uses -2, 2, and 0.2 as default values for γ_f , γ_t , and γ_b , respectively. For example, if parameter β is in the **PROFILE** statement, then during the profiling of β , PROC NLIN performs approximately $(\gamma_t - \gamma_f)/\gamma_b$ number of constrained optimizations. At the i th constrained optimization, the value of β is constrained to β_i ,

$$\beta_i = \hat{\beta} + \text{stderr}_{\beta} \left(\gamma_f + i \frac{(\gamma_t - \gamma_f)}{\gamma_b} \right)$$

and the rest of the parameters are freely optimized. In the preceding expression, $\hat{\beta}$ and stderr_{β} are the estimated value and standard error, respectively, for β . If parameter bounds are specified with **BOUNDS** statements, PROC NLIN enforces those bounds during the profiling optimizations.

NOTE: The profiling optimizations are performed using quasi-Newton algorithm. During these optimizations, any optimization options specified in the **PROC NLIN** statement and the **_LOSS_** variable are ignored.

TPLOT

produces profile t plots for each parameter listed in the **PROFILE** statement. If the **PROFDATA=** option is specified, the values of the corresponding statistics are written to the data set. See the section “[Profiling Parameters and Assessing the Influence of Observations on Parameter Estimates](#)” on page 6451 for details.

RETAIN Statement

RETAIN *variable* <=*values*> <... *variable* <=*values*>> ;

The **RETAIN** statement declares retained variables and specifies their values. A retained variable is like a control variable (see the section “[CONTROL Statement](#)” on page 6434) except that it is retained only *within* iterations. An iteration involves a single pass through the data set.

Other Programming Statements

PROC NLIN supports many statements that are similar to SAS programming statements used in a **DATA** step. However, there are some differences in capabilities; for additional information, see also the section “[Incompatibilities with SAS 6.11 and Earlier Versions of PROC NLIN](#)” on page 6466.

Several SAS programming statements can be used after the **PROC NLIN** statement. These statements can appear anywhere in the **PROC NLIN** code, but new variables must be created before they appear in other statements. For example, the following statements are valid since they create the variable **temp** before it is used in the **MODEL** statement:

```
proc nlin;
  parms b0=0 to 2 by 0.5 b1=0.01 to 0.09 by 0.01;
  temp = exp(-b1*x);
  model y=b0*(1-temp);
run;
```

The following statements result in missing values for y because the variable temp is undefined before it is used:

```
proc nlin;
  parms b0=0 to 2 by 0.5 b1=0.01 to 0.09 by 0.01;
  model y = b0*(1-temp);
  temp = exp(-b1*x);
run;
```

PROC NLIN can process assignment statements, explicitly or implicitly subscripted ARRAY statements, explicitly or implicitly subscripted array references, IF statements, SAS functions, and program control statements. You can use programming statements to create new SAS variables for the duration of the procedure. These variables are not included in the data set to which PROC NLIN is applied. Program statements can include variables in the [DATA=](#) data set, parameter names, variables created by preceding programming statements within PROC NLIN, and special variables used by PROC NLIN. The following SAS programming statements can be used in PROC NLIN:

```
ABORT;
ARRAY arrayname < [ dimensions ] > < $ > < variables-and-constants >;
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;
```

These statements can use the special variables created by PROC NLIN. See the section “[Special Variables](#)” on page 6454 for more information.

Details: NLIN Procedure

Automatic Derivatives

Depending on the optimization method you select, analytical first- and second-order derivatives are computed automatically. Derivatives can still be supplied using the DER.parm syntax. These DER.parm derivatives are not verified by the differentiator. If any needed derivatives are not supplied, they are computed and added to the programming statements. To view the computed derivatives, use the [LISTDER](#) or [LIST](#) option.

The following model is solved using Newton's method. Analytical first- and second-order derivatives are automatically computed. The compiled program code is shown in [Figure 81.6](#).

```
proc nlin data=Enzyme method=newton list;  
  parms x1=4 x2=2;  
  model Velocity = x1 * exp (x2 * Concentration);  
run;
```

Figure 81.6 Model and Derivative Code Output

The NLIN Procedure

Listing of Compiled Program Code		
Stmt	Line:Col	Statement as Parsed
1	1589:4	MODEL.Velocity = x1 * EXP(x2 * Concentration);
1	1589:4	@MODEL.Velocity/@x1 = EXP(x2 * Concentration);
1	1589:4	@MODEL.Velocity/@x2 = x1 * Concentration * EXP(x2 * Concentration);
1	1589:4	@@MODEL.Velocity/@x1/@x2 = Concentration * EXP(x2 * Concentration);
1	1589:4	@@MODEL.Velocity/@x2/@x1 = Concentration * EXP(x2 * Concentration);
1	1589:4	@@MODEL.Velocity/@x2/@x2 = x1 * Concentration * Concentration * EXP(x2 * Concentration);

Note that all the derivatives require the evaluation of EXP(X2 * Concentration). The actual machine-level code is displayed if you specify the [LISTCODE](#) option, as in the following statements:

```
proc nlin data=Enzyme method=newton listcode;  
  parms x1=4 x2=2;  
  model Velocity = x1 * exp (x2 * Concentration);  
run;
```

Note that, in the generated code, only one exponentiation is performed ([Figure 81.7](#)). The generated code reuses previous operations to be more efficient.

Figure 81.7 LISTCODE Output
The NLIN Procedure

Code Listing		
1 Stmt	line 1596 column 4. (1) arg=MODEL.Velocity	
MODEL	argsave=MODEL.Velocity	
	Source Text:	model Velocity = x1 * exp (x2 * Concentration);
Oper *	at 1596:34 (30,0,2).	* : _temp1 <- x2 Concentration
Oper EXP	at 1596:30 (103,0,1).	EXP : _temp2 <- _temp1
Oper *	at 1596:24 (30,0,2).	* : MODEL.Velocity <- x1 _temp2
Oper eeocf	at 1596:24 (18,0,1).	eeocf : _DER_ <- _DER_
Oper =	at 1596:24 (1,0,1).	= : @MODEL.Velocity/@x1 <- _temp2
Oper *	at 1596:30 (30,0,2).	* : @1dt1_1 <- Concentration _temp2
Oper *	at 1596:24 (30,0,2).	* : @MODEL.Velocity/@x2 <- x1 @1dt1_1
Oper =	at 1596:24 (1,0,1).	= : @@MODEL.Velocity/@x1/@x2 <- @1dt1_1
Oper =	at 1596:24 (1,0,1).	= : @@MODEL.Velocity/@x2/@x1 <- @1dt1_1
Oper *	at 1596:30 (30,0,2).	* : @2dt1_1 <- Concentration @1dt1_1
Oper *	at 1596:24 (30,0,2).	* : @@MODEL.Velocity/@x2/@x2 <- x1 @2dt1_1

Measures of Nonlinearity, Diagnostics and Inference

A “close-to-linear” nonlinear regression model, in the sense of Ratkowsky (1983, 1990), is a model in which parameter estimators have properties similar to those in a linear regression model. That is, the least squares estimators of the parameters are close to being unbiased and normally distributed, and they have minimum variance.

A nonlinear regression model sometimes fails to be close to linear due to the properties of one or several parameters. When this occurs, bias in the parameter estimates can render inferences that use the reported standard errors and confidence limits invalid.

PROC NLIN provides various measures of nonlinearity. To assess the nonlinearity of a model-data combination, you can use both of the following complementary sets of measures:

- Box’s bias (Box 1971) and Hougaard’s skewness (Hougaard 1982, 1985) of the least squares parameter estimates
- curvature measures of nonlinearity (Bates and Watts 1980).

Furthermore, PROC NLIN provides residual, leverage, and local-influence diagnostics (St. Laurent and Cook 1993).

In the following several sections, these nonlinearity measures and diagnostics are discussed. For this material, several basic definitions are required. Let \mathbf{X} be the Jacobian matrix for the model, $\mathbf{X} = \frac{\partial \mathbf{f}}{\partial \boldsymbol{\beta}}$, and let \mathbf{Q} and \mathbf{R} be the components of the QR decomposition of $\mathbf{X} = \mathbf{QR}$ of \mathbf{X} , where \mathbf{Q} is an $(n \times n)$ orthogonal matrix. Finally, let \mathbf{B} be the inverse of the matrix constructed from the first p rows of the $(n \times p)$ dimensional matrix

\mathbf{R} (that is, $\mathbf{B} = \mathbf{R}_p^{-1}$). Next define

$$\begin{aligned} [\mathbf{H}_j]_{kl} &= \frac{\partial^2 \mathbf{f}_j}{\partial \beta_k \partial \beta_l} \\ [\mathbf{U}_j]_{kl} &= \sum_{mn} \mathbf{B}'_{km} [\mathbf{H}_j]_{mn} \mathbf{B}_{nl} \\ [\mathbf{A}_j]_{kl} &= \sqrt{p \times \text{mse}} \sum_m \mathbf{Q}'_{jm} [\mathbf{U}_m]_{kl}, \end{aligned}$$

where \mathbf{H} , \mathbf{U} and the acceleration array \mathbf{A} are three-dimensional $(n \times p \times p)$ matrices. The first p faces of the acceleration array constitute a $(p \times p \times p)$ parameter-effects array and the last $(n - p)$ faces constitute the $(n - p \times p \times p)$ intrinsic curvature array (Bates and Watts 1980). The previous and subsequent quantities are computed at the least squares parameter estimators.

Box's Measure of Bias

The degree to which parameter estimators exhibit close-to-linear behavior can be assessed with Box's bias (Box 1971) and Hougaard's measure of skewness (Hougaard 1982, 1985). The bias and percentage bias measures are available through the **BIAS** option in the **PROC NLIN** statement. Box's bias measure is defined as

$$\hat{\mathbf{E}}[\hat{\beta} - \beta] = -\frac{\sigma^2}{2} (\mathbf{X}'\mathbf{W}\mathbf{X})^{-1} \sum_{i=1}^n w_i \mathbf{x}'_i \text{Tr} \left((\mathbf{X}'\mathbf{W}\mathbf{X})^{-1} [\mathbf{H}_i] \right)$$

where $\sigma^2 = \text{mse}$ if the **SIGSQ** option is not set. Otherwise, σ^2 is the value you set with the **SIGSQ** option. \mathbf{W} is the diagonal weight matrix specified with the **_WEIGHT_** variable (or the identity matrix if **_WEIGHT_** is not defined) and $[\mathbf{H}_i]$ is the $(p \times p)$ Hessian matrix at the i th observation. In the case of unweighted least squares, the bias formula can be expressed in terms of the acceleration array \mathbf{A} ,

$$\hat{\mathbf{E}}[\hat{\beta}_i - \beta_i] = -\frac{\sigma^2}{2p \times \text{mse}} \sum_{j,k=1}^p \mathbf{B}_{ij} [\mathbf{A}_j]_{kk}$$

As the preceding formulas illustrate, the bias depends solely on the parameter-effects array, thereby permitting its reduction through reparameterization. [Example 81.4](#) shows how changing the parameterization of a four-parameter logistic model can reduce the bias. Ratkowsky (1983, p. 21) recommends that you consider reparameterization if the percentage bias exceeds 1%.

Hougaard's Measure of Skewness

In addition to Box's bias, Hougaard's measure of skewness, g_{1i} (Hougaard 1982, 1985), is also provided in **PROC NLIN** to assess the close-to-linear behavior of parameter estimators. This measure is available through the **HOUGAARD** option in the **PROC NLIN** statement. Hougaard's skewness measure for the i th parameter is based on the third central moment, defined as

$$\mathbf{E} \left[\hat{\beta}_i - \mathbf{E}(\hat{\beta}_i) \right]^3 = -(\sigma^2)^2 \sum_{jkl} [\mathbf{L}]_{ij} [\mathbf{L}]_{ik} [\mathbf{L}]_{il} ([\mathbf{V}_j]_{kl} + [\mathbf{V}_k]_{jl} + [\mathbf{V}_l]_{jk})$$

where the sum is a triple sum over the number of parameters and

$$\mathbf{L} = (\mathbf{X}'\mathbf{X})^{-1} = \left(\frac{\partial \mathbf{f}}{\partial \beta'} \frac{\partial \mathbf{f}}{\partial \beta} \right)^{-1}$$

The term $[L]_{ij}$ denotes the value in row i , column j of the matrix L . (Hougaard (1985) uses superscript notation to denote elements in this inverse.) The matrix V is a three-dimensional ($p \times p \times p$) array

$$[V_j]_{kl} = \sum_{m=1}^n \frac{\partial F_m}{\partial \beta_j} \frac{\partial^2 F_m}{\partial \beta_k \partial \beta_l}$$

The third central moment is then normalized using the standard error as

$$G_{1i} = E \left[\hat{\beta}_i - E(\hat{\beta}_i) \right]^3 / (\sigma^2 \times [L]_{ii})^{3/2}$$

The previous expressions depend on the unknown values of the parameters and on the residual variance σ^2 . In order to evaluate the Hougaard measure in a particular data set, the NLIN procedure computes

$$g_{1i} = \hat{E} \left[\hat{\beta}_i - E(\hat{\beta}_i) \right]^3 / (\text{mse} \times [\hat{L}]_{ii})^{3/2}$$

$$\hat{E} \left[\hat{\beta}_i - E(\hat{\beta}_i) \right]^3 = -\text{mse}^2 \sum_{jkl} [\hat{L}]_{ij} [\hat{L}]_{ik} [\hat{L}]_{il} \left([\hat{V}_j]_{kl} + [\hat{V}_k]_{jl} + [\hat{V}_l]_{jk} \right)$$

Following Ratkowsky (1990, p. 28), the parameter β_i is considered to be very close to linear, reasonably close, skewed, or quite nonlinear according to the absolute value of the Hougaard measure $|g_{1i}|$ being less than 0.1, between 0.1 and 0.25, between 0.25 and 1, or greater than 1, respectively.

Relative Curvature Measures of Nonlinearity

Bates and Watts (1980) formulated the maximum parameter-effects and maximum intrinsic curvature measures of nonlinearity to assess the close-to-linear behavior of nonlinear models. Ratkowsky (1990) notes that of the two curvature components in a nonlinear model, the parameter-effects curvature is typically larger. It is this component that you can affect by changing the parameterization of a model. PROC NLIN provides these two measures of curvature both through the **STATS plot-option** and through the **NLINMEASURES** option in the **PROC NLIN** statement.

The maximum parameter-effects and intrinsic curvatures are defined, in a compact form, as

$$\mathbf{T}^\tau = \max_{\beta} \left\| \beta' \mathbf{A}^\tau \beta \right\|$$

$$\mathbf{T}^\eta = \max_{\beta} \left\| \beta' \mathbf{A}^\eta \beta \right\|$$

where \mathbf{T}^τ and \mathbf{T}^η denote the maximum parameter-effects and intrinsic curvatures, while \mathbf{A}^τ and \mathbf{A}^η stand for the parameter-effects and intrinsic curvature arrays. The maximization is carried out over a unit-vector of the parameter values (Bates and Watts 1980). In line with Bates and Watts (1980), PROC NLIN takes 10^{-4} as the convergence tolerance for the maximum intrinsic and parameter-effects curvatures. Note that the preceding matrix products involve contraction of the faces of the three-dimensional acceleration arrays with the normalized parameter vector, β . The corresponding expressions for the RMS (root mean square) parameter-effects and intrinsic curvatures can be found in Bates and Watts (1980).

The statistical significance of \mathbf{T}^τ and \mathbf{T}^η and the corresponding RMS values can be assessed by comparing these values with $1/\sqrt{F}$, where F is the upper $\alpha \times 100\%$ quantile of an F distribution with p and $n - p$ degrees of freedom (Bates and Watts 1980).

One motivation for fitting a nonlinear model in a different parameterization is to obtain a particular interpretation and to give parameter estimators more close-to-linear behavior. [Example 81.4](#) shows how changing the parameterization of a four-parameter logistic model can reduce the parameter-effects curvature and can yield a useful parameter interpretation at the same time. In addition, [Example 81.6](#) shows a nonlinear model with a high intrinsic curvature and the corresponding diagnostics.

Leverage in Nonlinear Regression

In contrast to linear regression, there are several measures of leverage in nonlinear regression. Furthermore, in nonlinear regression, the effect of a change in the i th response on the i th predicted value might depend on both the size of the change and the i th response itself (St. Laurent and Cook 1992). As a result, some observations might show superleverage —namely, leverages in excess of one (St. Laurent and Cook 1992).

PROC NLIN provides two measures of leverages: tangential and Jacobian leverages through the **PLOTS** option in the **PROC NLIN** statement and the **H=** and **J=** options of **OUTPUT** statement. Tangential leverage, H_i , is based on approximating the nonlinear model with a linear model that parameterizes the tangent plane at the least squares parameter estimators. In contrast, Jacobian leverage, J_i , is simply defined as the instantaneous rate of change in the i th predicted value with respect to the i th response (St. Laurent and Cook 1992).

The mathematical formulas for tangential and Jacobian leverages are

$$H_i = w_i \mathbf{x}_i (\mathbf{X}' \mathbf{W} \mathbf{X})^{-1} \mathbf{x}_i'$$

$$J_i = w_i \mathbf{x}_i (\mathbf{X}' \mathbf{W} \mathbf{X} - [\mathbf{W} \mathbf{e}][\mathbf{H}])^{-1} \mathbf{x}_i' ,$$

where \mathbf{e} is the vector of residuals, \mathbf{W} is the diagonal weight matrix if you specify the special variable `_WEIGHT_` and otherwise the identity matrix, and i indexes the corresponding quantities for the i th observation. The brackets `[.]` indicate column multiplication as defined in Bates and Watts (1980). The preceding formula for tangential leverage holds if the gradient, Marquardt, or Gauss methods are used. For the Newton method, the tangential leverage is set equal to the Jacobian leverage.

In a model with a large intrinsic curvature, the Jacobian and tangential leverages can be very different. In fact, the two leverages are identical only if the model provides an exact fit to the data ($\mathbf{e} = 0$) or the model is intrinsically linear (St. Laurent and Cook 1993). This is also illustrated by the leverage plot and nonlinearity measures provided in [Example 81.6](#).

Local Influence in Nonlinear Regression

St. Laurent and Cook (1993) suggest using l_{\max} , the direction that yields the maximum normal curvature, to assess the local influence of an additive perturbation to the response variable on the estimation of the parameters and variance of a nonlinear model. Defining the normal curvature components

$$C_\beta = \max_l \frac{2}{\sigma^2} l' \mathbf{J} l$$

$$C_\sigma = \max_l \frac{4}{\sigma^2} l' \mathbf{P}_e l$$

where \mathbf{J} is the Jacobian leverage matrix and $\mathbf{P}_e = \mathbf{e} \mathbf{e}' / (\mathbf{e}' \mathbf{e})$, you choose the l_{\max} that results in the maximum of the two curvature components (St. Laurent and Cook 1993). PROC NLIN provides l_{\max} through the **PLOTS** option in the **PROC NLIN** statement and the **LMAX=** option in the **OUTPUT** statement. [Example 81.6](#) shows a plot of l_{\max} for a model with high intrinsic curvature.

Residuals in Nonlinear Regression

If a nonlinear model is intrinsically nonlinear, using the residuals $\mathbf{e} = \mathbf{y} - \hat{\mathbf{y}}$ for diagnostics can be misleading (Cook and Tsai 1985). This is due to the fact that in correctly specified intrinsically nonlinear models, the residuals have nonzero means and different variances, even when the original error terms have identical variances. Furthermore, the covariance between the residuals and the predicted values tends to be negative semidefinite, complicating the interpretation of plots based on \mathbf{e} (Cook and Tsai 1985).

Projected residuals are proposed by Cook and Tsai (1985) to overcome these shortcomings of residuals, which are henceforth called raw (ordinary) residuals to differentiate them from their projected counterparts. Projected residuals have zero means and are uncorrelated with the predicted values. In fact, projected residuals are identical to the raw residuals in intrinsically linear models.

PROC NLIN provides raw and projected residuals, along with their standardized forms. In addition, the mean or expectation of the raw residuals is available. These can be accessed with the **PLOTS** option in the **PROC NLIN** statement and the **OUTPUT** statement options **PROJRES=**, **PROJSTUDENT=**, **RESEXPEC=**, **RESIDUAL=** and **STUDENT=**.

Denote the projected residuals by \mathbf{e}_p and the expectation of the raw residuals by $E[\mathbf{e}]$. Then

$$\mathbf{e}_p = (\mathbf{I}_n - \mathbf{P}_{xh}) \mathbf{e}$$

$$E[\mathbf{e}_i] = -\frac{\sigma^2}{2} \sum_{j=1}^n \tilde{\mathbf{P}}_{x,ij} \text{Tr} \left([\mathbf{H}_j] (\mathbf{X}'\mathbf{X})^{-1} \right)$$

where \mathbf{e}_i is the i th observation raw residual, \mathbf{I}_n is an n -dimensional identity matrix, \mathbf{P}_{xh} is the projector onto the column space of $(\mathbf{X}|\mathbf{H})$, and $\tilde{\mathbf{P}}_x = \mathbf{I}_n - \mathbf{P}_x$. The preceding formulas are general with the projectors defined accordingly to take the weighting into consideration. In unweighted least squares, $E[\mathbf{e}]$ reduces to

$$E[\mathbf{e}] = -\frac{1}{2}\sigma^2\tilde{\mathbf{Q}}\mathbf{a}$$

with $\tilde{\mathbf{Q}}$ being the last $n - p$ columns of the \mathbf{Q} matrix in the QR decomposition of \mathbf{X} and the $(n - p)$ dimensional vector \mathbf{a} being defined in terms of the intrinsic acceleration array

$$\mathbf{a}_i = \sum_{j=1}^p [\mathbf{A}_{i+p}]_{jj}$$

Standardization of the projected residuals requires the variance of the projected residuals. This is estimated using the formula (Cook and Tsai 1985)

$$\sigma_p^2 = \frac{\mathbf{e}_p' \mathbf{e}_p}{\text{Tr}(\mathbf{I}_n - \mathbf{P}_{xh})}$$

The standardized raw and projected residuals, denoted by $\tilde{\mathbf{e}}$ and $\tilde{\mathbf{e}}_p$ respectively, are given by

$$\tilde{\mathbf{e}} = \frac{\sqrt{w_i} \mathbf{e}}{\sigma \sqrt{1 - \mathbf{P}_{x,ii}}}$$

$$\tilde{\mathbf{e}}_p = \frac{\sqrt{w_i} \mathbf{e}_p}{\sigma \sqrt{1 - \mathbf{P}_{xh,ii}}}$$

The use of raw and projected residuals for diagnostics in nonlinear regression is illustrated in [Example 81.6](#).

Profiling Parameters and Assessing the Influence of Observations on Parameter Estimates

The global measures of nonlinearity, discussed in the preceding section, are very useful for assessing the overall nonlinearity of the model. However, the impact of global nonlinearity on inference regarding subsets of the parameter set cannot be easily determined (Cook and Tsai 1985). The impact of the nonlinearity on the uncertainty of individual parameters can be efficiently described by profile t plots and confidence curves (Bates and Watts 1988; Cook and Weisberg 1990).

A profile t plot for parameter β is a plot of the likelihood ratio pivotal statistic, $\tau(\beta)$, and the corresponding Wald pivotal statistic, $\sigma(\beta)$ (Bates and Watts 1988). The likelihood ratio pivotal statistic is defined as

$$\tau(\beta) = \text{sign}(\beta - \hat{\beta}) L(\beta)$$

with

$$L(\beta) = \left(\frac{\text{SSE}(\beta, \tilde{\Theta}) - \text{SSE}(\hat{\beta}, \hat{\Theta})}{\text{mse}} \right)^{1/2}$$

where β is the profile parameter and Θ refers to the remaining parameters. $\text{SSE}(\beta, \tilde{\Theta})$ is the sum of square errors where the profile parameter β is constrained at a given value and $\tilde{\Theta}$ is the least squares estimate of Θ conditional on a given value of β . In contrast, $\text{SSE}(\hat{\beta}, \hat{\Theta})$ is the sum of square errors for the full model. For linear models, $\tau(\beta)$ matches $\sigma(\beta)$, which is defined as

$$\sigma(\beta) = \frac{\beta - \hat{\beta}}{\text{stderr}_{\beta}}$$

where β is the constrained value, $\hat{\beta}$ is the estimated value for the parameter, and stderr_{β} is the standard error for the parameter. Usually a profile t plot is overlaid with a reference line that passes through the origin and has a slope of one. PROC NLIN follows this convention.

A confidence curve for a particular parameter is useful for validating Wald-based confidence intervals for the parameter against likelihood-based confidence intervals (Cook and Weisberg 1990). A confidence curve contains a scatter plot of the constrained parameter value versus $L(\beta)$. The Wald-based confidence intervals are overlaid as two straight lines that pass through $(0, \hat{\beta})$ with a slope of $\pm \text{stderr}_{\beta}$. Hence, for different levels of significance, you can easily compare the Wald-based confidence intervals against the corresponding confidence intervals that are based on the likelihood ratio. Cook and Weisberg (1990) recommend that you report a single Wald-based confidence interval only if there is a good agreement between the Wald and likelihood intervals up to at least the 99% confidence level.

Compared to local influence, the leave-one-out method performs a more complete analysis of the influence of observations on the values of parameter estimates. In this method, jackknife resampling removes each observation in turn and fits the model to the remaining data set. Hence, a data set with n observations will have n corresponding data sets with $n-1$ observations. The impact of each observation on a parameter is assessed by the absolute relative percentage change in the value of the parameter compared with the reference value from the full data.

Bootstrap Resampling and Estimation

Bootstrap resampling and estimation methods can be used to produce confidence intervals and covariance matrix estimates that have second-order, $\mathcal{O}(1/n)$ accuracy, where n is the number of observations (DiCiccio and Efron 1996). In contrast, the standard Wald-based confidence interval has first-order, $\mathcal{O}(1/\sqrt{n})$ accuracy.

Bootstrap methods achieve this higher accuracy at the cost of an orders-of-magnitude increase in numerical computation compared to standard asymptotic approximations. However, dramatic increases in performance and decreases in the cost of numerical computation have made bootstrap methods very attractive for routine statistical inference (MacKinnon 2002).

PROC NLIN samples as many bootstrap sample data sets (replicates) as you specify in the **NSAMPLES=** option and performs least squares fit on each replicate. For each least squares fit, PROC NLIN uses the original parameter estimates as starting values for the model parameters. The statistics from the least squares fits that converge are collected and used to produce confidence intervals, covariance and correlation matrices, and histograms and scatter plots of the bootstrap parameter estimates.

Each replicate is obtained by sampling the residuals instead of the input data set. The sampled residuals are used to simulate the response by using a bootstrap data generating process (DGP). PROC NLIN's bootstrap DGP produces replicates that contain the same number of observations as the number of usable observations in the input data set. In fact, the bootstrap DGP for a particular replicate starts by discarding the input data set observations that PROC NLIN deems unusable during the original least squares estimation. The next step of the bootstrap DGP for a replicate can be represented by the formula

$$\tilde{Y}_i = f(\hat{\beta}; \mathbf{z}'_i) + \tilde{\epsilon}_i$$

where \tilde{Y}_i is the i th simulated response, $\hat{\beta}$ is the original least squares parameter estimate, \mathbf{z}'_i is the i th regressor vector, and $\tilde{\epsilon}_i$ is the i th simulated error.

PROC NLIN makes several bootstrap DGP types available in the **DGP=** option of the **BOOTSTRAP** statement. These bootstrap DGP types differ only in how they obtain the i th simulated error, $\tilde{\epsilon}_i$. In general, $\tilde{\epsilon}_i$ can be represented as

$$\tilde{\epsilon}_i = s_r \hat{\epsilon}_r$$

where r is a uniform random integer between 1 and the number of usable observations, s_r is a scale factor that depends on the chosen bootstrap DGP options, and $\hat{\epsilon}_r$ is the r th residual obtained from the r th usable observation of the original least squares fit. The scale factor, s_r , that captures the differences among the various bootstrap DGP types takes one of the following values:

$$s_r = \begin{cases} 1 & \text{if DGP=RESIDUAL(RAW)} \\ \sqrt{\frac{\mathbf{n}}{\mathbf{n}-\mathbf{p}}} & \text{if DGP=RESIDUAL(ADJSSE)} \\ \frac{1}{\sqrt{1-\mathbf{H}_r}} & \text{if DGP=RESIDUAL(TAN)} \\ \frac{1}{\sqrt{1-\mathbf{J}_r}} & \text{if DGP=RESIDUAL(JAC)} \\ \frac{\gamma}{\sqrt{1-\mathbf{H}_r}} & \text{if DGP=WILD} \end{cases}$$

In the preceding formula, \mathbf{n} is the number of usable observations, \mathbf{p} is the number of model parameters, and \mathbf{H}_r and \mathbf{J}_r are the r th tangential and Jacobian leverages, respectively. For the WILD bootstrap DGP (Wu 1986), which is the only bootstrap DGP type exclusively available for weighted least squares, γ is a random number given by

$$\gamma = \begin{cases} -\frac{\sqrt{5}-1}{2} & \text{with probability } \frac{\sqrt{5}+1}{2\sqrt{5}} \\ \frac{\sqrt{5}+1}{2} & \text{with probability } \frac{\sqrt{5}-1}{2\sqrt{5}} \end{cases}$$

PROC NLIN makes three types of bootstrap confidence intervals available in the **BOOTCI** option in the **BOOTSTRAP** statement. These confidence intervals are the percentile, normal, and bias-corrected bootstrap

confidence intervals. The computational details of these confidence intervals for β_i , the i th model parameter, follow. For simplicity of notation, denote β_i as β . Also, let B represent the number of replicates for which the least squares fit converges

The option that computes the percentile bootstrap confidence interval, **BOOTCI(PERC)**, does so by computing the $100(\alpha/2)$ th and $100(1 - \alpha/2)$ th percentiles from the bootstrap parameter estimates. α is from the **ALPHA=** option in the **PROC NLIN** statement. These percentiles are computed as follows. Let $\tilde{\beta}_1, \tilde{\beta}_2, \dots, \tilde{\beta}_B$ represent the ordered values of the bootstrap estimates for β . Let the k th weighted average percentile be y , set $p = \frac{k}{100}$, and let

$$np = j + g$$

where j is the integer part of np and g is the fractional part of np . Then the k th percentile, y , is given by

$$y = \begin{cases} \frac{1}{2}(\tilde{\beta}_j + \tilde{\beta}_{j+1}) & \text{if } g = 0 \\ \tilde{\beta}_{j+1} & \text{if } g > 0 \end{cases}$$

which corresponds to the default percentile definition of the UNIVARIATE procedure.

In contrast, the **BOOTCI(NORMAL)** option in the **BOOTSTRAP** statement computes the normal bootstrap confidence interval by approximating the distribution of the bootstrap parameter estimates as a normal distribution. Consequently, the normal bootstrap confidence interval, for α level, is given by

$$\bar{\tilde{\beta}} \pm \text{stdb}_{\tilde{\beta}} \times z^{(1-\alpha/2)}$$

where $\bar{\tilde{\beta}}$ is the mean of the bootstrap estimates, $\tilde{\beta}$; $\text{stdb}_{\tilde{\beta}}$ is the standard deviation of the bootstrap parameter estimates; and $z^{(1-\alpha/2)}$ is the $100(1 - \alpha/2)$ th percentile of the standard normal distribution.

The **BOOTSTRAP** statement option that computes the bias-corrected bootstrap confidence interval, **BOOTCI(BC)**, does so by making use of the cumulative distribution function (CDF), $G(\tilde{\beta})$, of the bootstrap parameter estimates to correct for the upper and lower endpoints of the α level. The bias-corrected bootstrap confidence interval is given by

$$G^{-1}(\Phi(2z_0 \pm z_\alpha))$$

where Φ is the standard normal CDF, $z_\alpha = \Phi^{-1}(\alpha)$, and z_0 is a bias correction given by

$$z_0 = \Phi^{-1}\left(\frac{N(\tilde{\beta} \leq \hat{\beta})}{B}\right)$$

where $\hat{\beta}$ is the estimate of β from the original least squares fit and $N(\tilde{\beta} \leq \hat{\beta})$ is the number of bootstrap estimates, $\tilde{\beta}$, that are less than or equal to $\hat{\beta}$.

In addition, **PROC NLIN** produces bootstrap estimates of the covariance and correlation matrices. For the i th and j th model parameters, the covariance is estimated by

$$\text{Cov}(\tilde{\beta}_i, \tilde{\beta}_j) = \sum_{k=1}^{k=B} \frac{(\tilde{\beta}_{ik} - \bar{\tilde{\beta}}_i)(\tilde{\beta}_{jk} - \bar{\tilde{\beta}}_j)}{B - 1}$$

where the sum runs over the nonmissing bootstrap parameter estimates. The bootstrap correlation matrix is estimated by scaling the bootstrap covariance matrix

$$\text{Corr}(\tilde{\beta}_i, \tilde{\beta}_j) = \frac{\text{Cov}(\tilde{\beta}_i, \tilde{\beta}_j)}{\text{stdb}_{\tilde{\beta}_i} \text{stdb}_{\tilde{\beta}_j}}$$

where $\text{stdb}_{\tilde{\beta}_i}$ and $\text{stdb}_{\tilde{\beta}_j}$ are the bootstrap standard deviations for the i th and j th parameters.

Missing Values

If the value of any one of the SAS variables involved in the model is missing from an observation, PROC NLIN deems that observation unusable and omits it from the analysis. If only the value of the dependent variable is missing, that observation has a predicted value calculated for it when you use an **OUTPUT** statement and specify the **PREDICTED=** option.

If an observation includes a missing value for one of the independent variables, both the predicted value and the residual value are missing for that observation. If the iterations fail to converge, the values for all variables named in the **OUTPUT** statement are set to missing.

Special Variables

Several special variables are created automatically and can be used in PROC NLIN programming statements.

Special Variables with Values That Are Set by PROC NLIN

The values of the following six special variables are set by PROC NLIN and should not be reset to a different value by programming statements:

ERROR	is set to 1 if a numerical error or invalid argument to a function occurs during the current execution of the program. It is reset to 0 before each new execution.
ITER	represents the current iteration number. The variable _ITER_ is set to -1 during the grid search phase.
MODEL	is set to 1 for passes through the data when only the predicted values are needed, and not the derivatives. It is 0 when both predicted values and derivatives are needed. If your derivative calculations consume a lot of time, you can save resources by using the following statement after your MODEL statement but before your derivative calculations:

```
if _model_ then return;
```

The derivatives generated by PROC NLIN do this automatically.

N	indicates the number of times the PROC NLIN step has been executed. It is never reset for successive passes through the data set.
OBS	indicates the observation number in the data set for the current program execution. It is reset to 1 to start each pass through the data set (unlike the _N_ variable).
SSE	contains the error sum of squares of the last iteration. During the grid search phase, the _SSE_ variable is set to 0. For iteration 0, the _SSE_ variable is set to the SSE associated with the point chosen from the grid search.

PROC NLIN always sets the preceding special variables. In contrast, it sets the following special variables only when the **BOOTSTRAP** or **PROFILE** statements are invoked:

`__COMPUTED_DEP__` is used to define `_P_WEIGHTED_SSE_`.

`_P_WEIGHTED_SSE_` contains the error sum of squares for the optimization tasks that are required for profiling parameters, assessing the influence of each observation on the estimate of each profiled parameter, and bootstrap estimation.

Special Variable Used to Determine Convergence Criteria

The special variable `_LOSS_` can be used to determine the criterion function for convergence and step shortening. PROC NLIN looks for the variable `_LOSS_` in the programming statements and, if it is defined, uses the (weighted) sum of this value instead of the residual sum of squares to determine the criterion function for convergence and step shortening. This feature is useful in certain types of maximum-likelihood estimation.

NOTE: Even if you specify the `_LOSS_` variable in your programming statements, the NLIN procedure continues to solve a least squares problem. The specified `_LOSS_` variable does **not** define or alter the objective function for parameter estimation.

Weighted Regression with the Special Variable `_WEIGHT_`

To obtain weighted nonlinear least squares estimates of parameters, make an assignment to the `_WEIGHT_` variable as in the following statement:

```
__weight__ = expression;
```

When this statement is included, the expression on the right side of the assignment statement is evaluated for each observation in the data set. The values multiplied by $1/\sigma^2$ are then taken as inverse elements of the diagonal variance-covariance matrix of the dependent variable.

When a variable name is given after the equal sign, the values of the variable are taken as the inverse elements of the variance-covariance matrix. The larger the `_WEIGHT_` value, the more importance the observation is given.

The `_WEIGHT_` variable can be a function of the estimated parameters. For estimation purposes, the derivative of the `_WEIGHT_` variable with respect to the parameters is not included in the gradient and the Hessian of the loss function. This is normally the desired approach for iteratively reweighted least squares estimation. When the `_WEIGHT_` variable is a function of the parameters, the gradient and the Hessian used can lead to poor convergence or nonconvergence of the requested estimation. To have the derivative of the `_WEIGHT_` variable with respect to the parameters included in the gradient and the Hessian of the loss function, do not use the `_WEIGHT_` variable. Instead, redefine the model as

$$(y - f(x, \beta)) \times \sqrt{\text{wgt}(\beta)}$$

where y is the original dependent variable, $f(x, \beta)$ is the nonlinear model, and $\text{wgt}(\beta)$ is the weight that is a function of the parameters.

If the `_WEIGHT_ =` statement is not used, the default value of 1 is used, and regular least squares estimates are obtained.

Troubleshooting

This section describes a number of problems that might occur in your analysis with PROC NLIN.

Excessive Computing Time

If you specify a grid of starting values that contains many points, the analysis might take excessive time since the procedure must go through the entire data set for each point on the grid.

The analysis might also take excessive time if your problem takes many iterations to converge, since each iteration requires as much time as a linear regression with predicted values and residuals calculated.

Dependencies

The matrix of partial derivatives can be singular, possibly indicating an overparameterized model. For example, if b0 starts at zero in the following model, the derivatives for b1 are all zero for the first iteration:

```
parms b0=0 b1=.022;
model pop=b0*exp(b1*(year-1790));
der.b0=exp(b1*(year-1790));
der.b1=(year-1790)*b0*exp(b1*(year-1790));
```

The first iteration changes a subset of the parameters; then the procedure can make progress in succeeding iterations. This singularity problem is local. The next example displays a global problem. The term b2 in the exponent is not identifiable since it trades roles with b0.

```
parms b0=3.9 b1=.022 b2=0;
model pop=b0*exp(b1*(year-1790)+b2);
der.b0 = exp(b1*(year-1790)+b2);
der.b1 = (year-1790)*b0*exp(b1*(year-1790)+b2);
der.b2 = b0*exp(b1*(year-1790)+b2);
```

Unable to Improve

The method can lead to steps that do not improve the estimates even after a series of step halvings. If this happens, the procedure issues a message stating that it is unable to make further progress, and it then displays the following warning message:

```
PROC NLIN failed to converge
```

Then it displays the results. This means that the procedure has not converged. If you provided your own derivatives, check them carefully and then examine the residual sum of squares surface. If PROC NLIN has not converged, try a different set of starting values, a different METHOD= specification, the G4 option, or a different model.

Divergence

The iterative process might diverge, resulting in overflows in computations. It is also possible that parameters enter a space where arguments to such functions as LOG and SQRT become invalid. For example, consider the following model:

```
parms b=0;
model y = x / b;
```

Suppose that y contains only zeros, and suppose that the values for variable x are not zero. There is no least squares estimate for b since the SSE declines as b approaches infinity or minus infinity. To avoid the problem, the same model could be parameterized as $y = a \cdot x$.

If you have divergence problems, try reparameterizing the model, selecting different starting values, increasing the maximum allowed number of iterations (the `MAXITER=` option), specifying an alternative `METHOD=` option, or including a `BOUNDS` statement.

Local Minimum

The program might converge to a local rather than a global minimum. For example, consider the following model:

```
parms a=1 b=-1;
model y=(1-a*x)*(1-b*x);
```

Once a solution is found, an equivalent solution with the same SSE can be obtained by swapping the values of a and b .

Discontinuities

The computational methods assume that the model is a continuous and smooth function of the parameters. If this is not true, the method does not work. For example, the following models do not work:

```
model y=a+int(b*x);

model y=a+b*x+4*(z>c);
```

Responding to Trouble

PROC NLIN does not necessarily produce a good solution the first time. Much depends on specifying good initial values for the parameters. You can specify a grid of values in the `PARMS` statement to search for good starting values. While most practical models should give you no trouble, other models can require switching to a different iteration method or a different computational method for matrix inversion. Specifying the option `METHOD=MARQUARDT` sometimes works when the default method (Gauss-Newton) does not work.

Computational Methods

Nonlinear Least Squares

Recall the basic notation for the nonlinear regression model from the section “[Notation for Nonlinear Regression Models](#)” on page 6413. The parameter vector β belongs to Ω , a subset of R^p . Two points of this set are of particular interest: the true value $\widetilde{\beta}$ and the least squares estimate $\widehat{\beta}$. The general nonlinear model fit with the NLIN procedure is represented by the equation

$$Y = f(\widetilde{\beta}_0, \widetilde{\beta}_1, \dots, \widetilde{\beta}_p; z_1, z_2, \dots, z_k) + \epsilon = f(\widetilde{\beta}; z_1, z_2, \dots, z_k) + \epsilon$$

where \mathbf{z}_j denotes the $(n \times 1)$ vector of the j th regressor (independent) variable, $\tilde{\boldsymbol{\beta}}$ is the true value of the parameter vector, and $\boldsymbol{\epsilon}$ is the $(n \times 1)$ vector of homoscedastic and uncorrelated model errors with zero mean.

To write the model for the i th observation, the i th elements of $\mathbf{z}_1, \dots, \mathbf{z}_k$ are collected in the row vector \mathbf{z}'_i , and the model equation becomes

$$Y_i = f(\boldsymbol{\beta}; \mathbf{z}'_i) + \epsilon_i$$

The shorthand $f_i(\boldsymbol{\beta})$ will also be used throughout to denote the mean of the i th observation.

For any given value $\boldsymbol{\beta}$ we can compute the residual sum of squares

$$\begin{aligned} \text{SSE}(\boldsymbol{\beta}) &= \sum_{i=1}^n (y_i - f(\boldsymbol{\beta}; \mathbf{z}'_i))^2 \\ &= \sum_{i=1}^n (y_i - f_i(\boldsymbol{\beta}))^2 = \mathbf{r}(\boldsymbol{\beta})' \mathbf{r}(\boldsymbol{\beta}) \end{aligned}$$

The aim of nonlinear least squares estimation is to find the value $\hat{\boldsymbol{\beta}}$ that minimizes $\text{SSE}(\boldsymbol{\beta})$. Because \mathbf{f} is a nonlinear function of $\boldsymbol{\beta}$, a closed-form solution does not exist for this minimization problem. An iterative process is used instead. The iterative techniques that PROC NLIN uses are similar to a series of linear regressions involving the matrix \mathbf{X} and the residual vector $\mathbf{r} = \mathbf{y} - \mathbf{f}(\boldsymbol{\beta})$, evaluated at the current values of $\boldsymbol{\beta}$.

It is more insightful, however, to describe the algorithms in terms of their approach to minimizing the residual sum of squares and in terms of their updating formulas. If $\hat{\boldsymbol{\beta}}^{(u)}$ denotes the value of the parameter estimates at the u th iteration, and $\hat{\boldsymbol{\beta}}^{(0)}$ are your starting values, then the NLIN procedure attempts to find values k and $\boldsymbol{\Delta}$ such that

$$\hat{\boldsymbol{\beta}}^{(u+1)} = \hat{\boldsymbol{\beta}}^{(u)} + k \boldsymbol{\Delta}$$

and

$$\text{SSE}(\hat{\boldsymbol{\beta}}^{(u+1)}) < \text{SSE}(\hat{\boldsymbol{\beta}}^{(u)})$$

The various methods to fit a nonlinear regression model—which you can select with the **METHOD=** option in the **PROC NLIN** statement—differ in the calculation of the update vector $\boldsymbol{\Delta}$.

The gradient and Hessian of the residual sum of squares with respect to individual parameters and pairs of parameters are, respectively,

$$\begin{aligned} \mathbf{g}(\boldsymbol{\beta})_j &= \frac{\partial \text{SSE}(\boldsymbol{\beta})}{\partial \beta_j} = -2 \sum_{i=1}^n (y_i - f_i(\boldsymbol{\beta})) \frac{\partial f_i(\boldsymbol{\beta})}{\partial \beta_j} \\ [\mathbf{H}(\boldsymbol{\beta})]_{jk} &= \frac{\partial^2 \text{SSE}(\boldsymbol{\beta})}{\partial \beta_j \partial \beta_k} = 2 \sum_{i=1}^n \frac{\partial f_i(\boldsymbol{\beta})}{\partial \beta_j} \frac{\partial f_i(\boldsymbol{\beta})}{\partial \beta_k} - (y_i - f_i(\boldsymbol{\beta})) \frac{\partial^2 f_i(\boldsymbol{\beta})}{\partial \beta_j \partial \beta_k} \end{aligned}$$

Denote as $\mathbf{H}_i^*(\boldsymbol{\beta})$ the Hessian matrix of the mean function,

$$[\mathbf{H}_i^*(\boldsymbol{\beta})]_{jk} = \left[\frac{\partial^2 f_i(\boldsymbol{\beta})}{\partial \beta_j \partial \beta_k} \right]_{jk}$$

Collecting the derivatives across all parameters leads to the expressions

$$\mathbf{g}(\boldsymbol{\beta}) = \frac{\partial \text{SSE}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = -2\mathbf{X}'\mathbf{r}(\boldsymbol{\beta})$$

$$\mathbf{H}(\boldsymbol{\beta}) = \frac{\partial^2 \text{SSE}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}'} = 2 \left(\mathbf{X}'\mathbf{X} - \sum_{i=1}^n r_i(\boldsymbol{\beta}) \mathbf{H}_i^*(\boldsymbol{\beta}) \right)$$

The change in the vector of parameter estimates is computed as follows, depending on the estimation method:

$$\text{Gauss-Newton: } \boldsymbol{\Delta} = (-\mathbf{E}[\mathbf{H}(\boldsymbol{\beta})])^{-1} \mathbf{g}(\boldsymbol{\beta}) = (\mathbf{X}'\mathbf{X})^{-1} \mathbf{X}'\mathbf{r}$$

$$\text{Marquardt: } \boldsymbol{\Delta} = (\mathbf{X}'\mathbf{X} + \lambda \text{diag}(\mathbf{X}'\mathbf{X}))^{-1} \mathbf{X}'\mathbf{r}$$

$$\text{Newton: } \boldsymbol{\Delta} = -\mathbf{H}(\boldsymbol{\beta})^{-1} \mathbf{g}(\boldsymbol{\beta}) = \mathbf{H}(\boldsymbol{\beta})^{-1} \mathbf{X}'\mathbf{r}$$

$$\text{Steepest descent: } \boldsymbol{\Delta} = -\frac{1}{2} \frac{\partial \text{SSE}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = \mathbf{X}'\mathbf{r}$$

The Gauss-Newton and Marquardt iterative methods regress the residuals onto the partial derivatives of the model with respect to the parameters until the estimates converge. You can view the Marquardt algorithm as a Gauss-Newton algorithm with a ridgeing penalty. The Newton iterative method regresses the residuals onto a function of the first and second derivatives of the model with respect to the parameters until the estimates converge. Analytical first- and second-order derivatives are automatically computed as needed.

The default method used to compute $(\mathbf{X}'\mathbf{X})^{-1}$ is the sweep (Goodnight 1979). It produces a reflexive generalized inverse (a g_2 -inverse, Pringle and Rayner 1971). In some cases it might be preferable to use a Moore-Penrose inverse (a g_4 -inverse) instead. If you specify the **G4** option in the **PROC NLIN** statement, a g_4 -inverse is used to calculate $\boldsymbol{\Delta}$ on each iteration.

The four algorithms are now described in greater detail.

Algorithmic Details

Gauss-Newton and Newton Methods

From the preceding set of equations you can see that the Marquardt method is a ridged version of the Gauss-Newton method. If the ridge parameter λ equals zero, the Marquardt step is identical to the Gauss-Newton step. An important difference between the Newton methods and the Gauss-Newton-type algorithms lies in the use of second derivatives. To motivate this distinctive element between Gauss-Newton and the Newton method, focus first on the objective function in nonlinear least squares. To numerically find the minimum of

$$\text{SSE}(\boldsymbol{\beta}) = \mathbf{r}(\boldsymbol{\beta})'\mathbf{r}(\boldsymbol{\beta})$$

you can approach the problem by approximating the sum of squares criterion by a criterion for which you can compute a closed-form solution. Following Seber and Wild (1989, Sect. 2.1.3), we can achieve that by doing the following:

- approximating the model and substituting the approximation into $SSE(\boldsymbol{\beta})$
- approximating $SSE(\boldsymbol{\beta})$ directly

The first method, approximating the nonlinear model with a first-order Taylor series, is the purview of the Gauss-Newton method. Approximating the residual sum of squares directly is the realm of the Newton method.

The first-order Taylor series of the residual $\mathbf{r}(\boldsymbol{\beta})$ at the point $\hat{\boldsymbol{\beta}}$ is

$$\mathbf{r}(\boldsymbol{\beta}) \approx \mathbf{r}(\hat{\boldsymbol{\beta}}) - \hat{\mathbf{X}} (\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})$$

Substitution into $SSE(\boldsymbol{\beta})$ leads to the objective function for the Gauss-Newton step:

$$SSE(\boldsymbol{\beta}) \approx S_G(\boldsymbol{\beta}) = \mathbf{r}(\hat{\boldsymbol{\beta}})' - 2\mathbf{r}'(\hat{\boldsymbol{\beta}})\hat{\mathbf{X}} (\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}) + (\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})' (\hat{\mathbf{X}}'\hat{\mathbf{X}}) (\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})$$

“Hat” notation is used here to indicate that the quantity in question is evaluated at $\hat{\boldsymbol{\beta}}$.

To motivate the Newton method, take a second-order Taylor series of $SSE(\boldsymbol{\beta})$ around the value $\hat{\boldsymbol{\beta}}$:

$$SSE(\boldsymbol{\beta}) \approx S_N(\boldsymbol{\beta}) = SSE(\hat{\boldsymbol{\beta}}) + \mathbf{g}(\hat{\boldsymbol{\beta}})' (\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}) + \frac{1}{2} (\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})' \mathbf{H}(\hat{\boldsymbol{\beta}}) (\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})$$

Both $S_G(\boldsymbol{\beta})$ and $S_N(\boldsymbol{\beta})$ are quadratic functions in $\boldsymbol{\beta}$ and are easily minimized. The minima occur when

$$\text{Gauss-Newton: } \boldsymbol{\beta} - \hat{\boldsymbol{\beta}} = (\hat{\mathbf{X}}'\hat{\mathbf{X}})^{-} \hat{\mathbf{X}}'\mathbf{r}(\hat{\boldsymbol{\beta}})$$

$$\text{Newton: } \boldsymbol{\beta} - \hat{\boldsymbol{\beta}} = -\mathbf{H}(\hat{\boldsymbol{\beta}})^{-} \mathbf{g}(\hat{\boldsymbol{\beta}})$$

and these terms define the preceding $\boldsymbol{\Delta}$ update vectors.

Gauss-Newton Method

Since the Gauss-Newton method is based on an approximation of the model, you can also derive the update vector by first considering the “normal” equations of the nonlinear model

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

and then substituting the Taylor approximation

$$\mathbf{f}(\boldsymbol{\beta}) \approx \mathbf{f}(\hat{\boldsymbol{\beta}}) + \mathbf{X} (\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})$$

for $\mathbf{f}(\boldsymbol{\beta})$. This leads to

$$\mathbf{X}' (\mathbf{f}(\hat{\boldsymbol{\beta}}) + \mathbf{X}(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})) = \mathbf{X}'\mathbf{Y}$$

$$(\mathbf{X}'\mathbf{X})(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}) = \mathbf{X}'\mathbf{Y} - \mathbf{X}'\mathbf{f}(\hat{\boldsymbol{\beta}})$$

$$(\mathbf{X}'\mathbf{X})\boldsymbol{\Delta} = \mathbf{X}'\mathbf{r}(\hat{\boldsymbol{\beta}})$$

and the update vector becomes

$$\boldsymbol{\Delta} = (\mathbf{X}'\mathbf{X})^{-} \mathbf{X}'\mathbf{r}(\hat{\boldsymbol{\beta}})$$

NOTE: If $\mathbf{X}'\mathbf{X}$ is singular or becomes singular, PROC NLIN computes $\boldsymbol{\Delta}$ by using a generalized inverse for the iterations after singularity occurs. If $\mathbf{X}'\mathbf{X}$ is still singular for the last iteration, the solution should be examined.

Newton Method

The Newton method uses the second derivatives and solves the equation

$$\Delta = H^{-1}X'r$$

If the automatic variables `_WEIGHT_`, `_WGTJPJ_`, and `_RESID_` are used, then

$$\Delta = H^{-1}X'W^{SSE}r^*$$

is the direction, where

$$H = X'W^{XPX}X - \sum_{i=1}^n H_i^*(\beta)w_i^{XPX}r_i^*$$

and

W^{SSE} is an $n \times n$ diagonal matrix with elements w_i^{SSE} of weights from the `_WEIGHT_` variable. Each element w_i^{SSE} contains the value of `_WEIGHT_` for the i th observation.

W^{XPX} is an $n \times n$ diagonal matrix with elements w_i^{XPX} of weights from the `_WGTJPJ_` variable. Each element w_i^{XPX} contains the value of `_WGTJPJ_` for the i th observation.

r^* is a vector with elements r_i^* from the `_RESID_` variable. Each element r_i^* contains the value of `_RESID_` evaluated for the i th observation.

Marquardt Method

The updating formula for the Marquardt method is as follows:

$$\Delta = (X'X + \lambda \text{diag}(X'X))^{-1}X'e$$

The Marquardt method is a compromise between the Gauss-Newton and steepest descent methods (Marquardt 1963). As $\lambda \rightarrow 0$, the direction approaches Gauss-Newton. As $\lambda \rightarrow \infty$, the direction approaches steepest descent.

Marquardt's studies indicate that the average angle between Gauss-Newton and steepest descent directions is about 90° . A choice of λ between 0 and infinity produces a compromise direction.

By default, PROC NLIN chooses $\lambda = 10^{-7}$ to start and computes Δ . If $SSE(\beta_0 + \Delta) < SSE(\beta_0)$, then $\lambda = \lambda/10$ for the next iteration. Each time $SSE(\beta_0 + \Delta) > SSE(\beta_0)$, then $\lambda = 10\lambda$.

NOTE: If the SSE decreases on each iteration, then $\lambda \rightarrow 0$, and you are essentially using the Gauss-Newton method. If SSE does not improve, then λ is increased until you are moving in the steepest descent direction.

Marquardt's method is equivalent to performing a series of ridge regressions, and it is useful when the parameter estimates are highly correlated or the objective function is not well approximated by a quadratic.

Steepest Descent (Gradient) Method

The steepest descent method is based directly on the gradient of $0.5r(\beta)'r(\beta)$:

$$\frac{1}{2} \frac{\partial SSE(\beta)}{\partial \beta} = -X'r$$

The quantity $-\mathbf{X}'\mathbf{r}$ is the gradient along which $\epsilon'\epsilon$ increases. Thus $\mathbf{\Delta} = \mathbf{X}'\mathbf{r}$ is the direction of steepest descent.

If the automatic variables `_WEIGHT_` and `_RESID_` are used, then

$$\mathbf{\Delta} = \mathbf{X}'\mathbf{W}^{\text{SSE}}\mathbf{r}^*$$

is the direction, where

\mathbf{W}^{SSE} is an $n \times n$ diagonal matrix with elements w_i^{SSE} of weights from the `_WEIGHT_` variable. Each element w_i^{SSE} contains the value of `_WEIGHT_` for the i th observation.

\mathbf{r}^* is a vector with elements r_i^* from `_RESID_`. Each element r_i^* contains the value of `_RESID_` evaluated for the i th observation.

Using the method of steepest descent, let

$$\boldsymbol{\beta}^{(k+1)} = \boldsymbol{\beta}^{(k)} + \alpha \mathbf{\Delta}$$

where the scalar α is chosen such that

$$\text{SSE}(\boldsymbol{\beta}_i + \alpha \mathbf{\Delta}) < \text{SSE}(\boldsymbol{\beta}_i)$$

NOTE: The steepest-descent method can converge very slowly and is therefore not generally recommended. It is sometimes useful when the initial values are poor.

Step-Size Search

The default method of finding the step size k is step halving by using `SMETHOD=HALVE`. If $\text{SSE}(\boldsymbol{\beta}^{(u)} + \mathbf{\Delta}) > \text{SSE}(\boldsymbol{\beta}^{(u)})$, compute $\text{SSE}(\boldsymbol{\beta}^{(u)} + 0.5\mathbf{\Delta})$, $\text{SSE}(\boldsymbol{\beta}^{(u)} + 0.25\mathbf{\Delta})$, \dots , until a smaller SSE is found.

If you specify `SMETHOD=GOLDEN`, the step size k is determined by a golden section search. The parameter `TAU` determines the length of the initial interval to be searched, with the interval having length `TAU` (or $2 \times \text{TAU}$), depending on $\text{SSE}(\boldsymbol{\beta}^{(u)} + \mathbf{\Delta})$. The `RHO` parameter specifies how fine the search is to be. The SSE at each endpoint of the interval is evaluated, and a new subinterval is chosen. The size of the interval is reduced until its length is less than `RHO`. One pass through the data is required each time the interval is reduced. Hence, if `RHO` is very small relative to `TAU`, a large amount of time can be spent determining a step size. For more information about the golden section search, see Kennedy and Gentle (1980).

If you specify `SMETHOD=CUBIC`, the NLIN procedure performs a cubic interpolation to estimate the step size. If the estimated step size does not result in a decrease in SSE, step halving is used.

Output Data Sets

The data set produced by the `OUTEST=` option in the `PROC NLIN` statement contains the parameter estimates on each iteration, including the grid search.

The variable `_ITER_` contains the iteration number. The variable `_TYPE_` denotes whether the observation contains iteration parameter estimates (“ITER”), final parameter estimates (“FINAL”), or covariance estimates (“COVB”). The variable `_NAME_` contains the parameter name for covariances, and the variable `_SSE_`

contains the objective function value for the parameter estimates. The variable `_STATUS_` indicates whether the estimates have converged.

The data set produced by the `OUTPUT` statement contains statistics calculated for each observation. In addition, the data set contains the variables from the input data set and any `ID` variables that are specified in the `ID` statement.

The data set that is produced by the `PROFDATA=` option in the `PROFILE` statement contains statistics that are calculated during parameter profiling. The variable `_TYPE_` denotes whether the observation is from a profiling task (set as `PROF`) or jackknife resampling task (set as `JACK`). The variable `_PROF_PARM_` refers to the name of the profiled parameter for profiling tasks. The variable `_OBS_INF_MEAS_` is the observation index of the deleted observation for jackknife resampling tasks, the variable `_SSE_` stands for the sum of squared errors, and the variables `_L_STAT_` and `_W_STAT_` contain the likelihood ratio pivotal statistic and the Wald pivotal statistic, respectively. The rest of the variables are the parameter estimates. Missing values for `_SSE_` indicate that the least squares fit for that particular profiling or jackknife task does not converge.

The data set that is produced by the `BOOTDATA=` option in the `BOOTSTRAP` statement contains statistics that are calculated during bootstrap resampling and estimation. The variable `_REPLICATE_` indexes the bootstrap sample data sets. The variable `_SEED_` refers to the initial seed value that is used in obtaining each sample data set, the variable `_SSE_` stands for the sum of squared errors, and the rest of the variables are the parameter estimates for the model that is fit to the bootstrap sample data set. Missing values for `_SSE_` indicate that the least squares fit for that particular bootstrap sample data set does not converge.

Confidence Intervals

Parameter Confidence Intervals

By default, the parameter confidence intervals are computed using the Wald-based formula:

$$\hat{\beta}_i \pm \text{stderr}_i \times t(n - p, 1 - \alpha/2)$$

where $\hat{\beta}_i$ is the i th parameter estimate, stderr_i is its estimated standard error, $t(n - p, 1 - \alpha/2)$ is a t statistic with $n - p$ degrees of freedom, n is the number of usable observations, and p is the number of parameters. You can specify the significance level α used in constructing these confidence limits with the `ALPHA=` option in the `PROC NLIN` statement; the default value is $\alpha = 0.05$.

Because Wald-based confidence intervals are only asymptotically valid, you can use the `BOOTSTRAP` statement to request bootstrap-based confidence intervals. Furthermore, you can augment these confidence intervals with confidence curves for a more complete inference on confidence intervals. For more information, see the section “[Bootstrap Resampling and Estimation](#)” on page 6451 and “[Profiling Parameters and Assessing the Influence of Observations on Parameter Estimates](#)” on page 6451.

Model Confidence Intervals

Model confidence intervals are output when an `OUT=` data set is specified and one or more of the *keywords* `LCLM`, `UCLM`, `LCL`, `UCL`, `L95M=`, `U95M=`, `L95=`, and `U95=` is specified. The expressions for these terms

are as follows:

$$\begin{aligned}
 \text{LCLM} &= f(\boldsymbol{\beta}, \mathbf{z}_i) - \sqrt{\text{MSE} \times h_i / w_i} \times t(n - p, 1 - \alpha/2) \\
 \text{UCLM} &= f(\boldsymbol{\beta}, \mathbf{z}_i) + \sqrt{\text{MSE} \times h_i / w_i} \times t(n - p, 1 - \alpha/2) \\
 \text{LCL} &= f(\boldsymbol{\beta}, \mathbf{z}_i) - \sqrt{\text{MSE}(h_i + 1/w_i)} \times t(n - p, 1 - \alpha/2) \\
 \text{UCL} &= f(\boldsymbol{\beta}, \mathbf{z}_i) + \sqrt{\text{MSE}(h_i + 1/w_i)} \times t(n - p, 1 - \alpha/2) \\
 \text{L95M} &= f(\boldsymbol{\beta}, \mathbf{z}_i) - \sqrt{\text{MSE} \times h_i / w_i} \times t(n - p, 1 - 0.05/2) \\
 \text{U95M} &= f(\boldsymbol{\beta}, \mathbf{z}_i) + \sqrt{\text{MSE} \times h_i / w_i} \times t(n - p, 1 - 0.05/2) \\
 \text{L95} &= f(\boldsymbol{\beta}, \mathbf{z}_i) - \sqrt{\text{MSE}(h_i + 1/w_i)} \times t(n - p, 1 - 0.05/2) \\
 \text{U95} &= f(\boldsymbol{\beta}, \mathbf{z}_i) + \sqrt{\text{MSE}(h_i + 1/w_i)} \times t(n - p, 1 - 0.05/2)
 \end{aligned}$$

where $h_i = w_i \mathbf{x}_i (\mathbf{X}' \mathbf{W} \mathbf{X})^{-1} \mathbf{x}_i'$ is the leverage, $\mathbf{X} = \partial \mathbf{f} / \partial \boldsymbol{\beta}$, and \mathbf{x}_i is the i th row of \mathbf{X} . These results are derived for linear systems. The intervals are approximate for nonlinear models. The value α in the preceding formulas for **LCLM**, **UCLM**, **LCL**, and **UCL** can be set with the **ALPHA=** option in the **PROC NLIN** statement or with the **ALPHA=** option in the **OUTPUT** statement. If both **ALPHA=** options are specified, the option in the **OUTPUT** takes precedence.

Covariance Matrix of Parameter Estimates

By default, PROC NLIN estimates the covariance matrix of parameter estimates based on the following first order asymptotic approximation. For unconstrained estimates (no active bounds), the covariance matrix of the parameter estimates is

$$\text{mse} \times (\mathbf{X}' \mathbf{X})^{-1}$$

for the gradient, Marquardt, and Gauss methods and

$$\text{mse} \times \mathbf{H}^{-1}$$

for the Newton method. Recall that \mathbf{X} is the matrix of the first partial derivatives of the nonlinear model with respect to the parameters. The matrices are evaluated at the final parameter estimates. The mean squared error, the estimate of the residual variance σ^2 , is computed as

$$\text{mse} = \mathbf{r}' \mathbf{r} / (n - p)$$

where n is the number of nonmissing (used) observations and p is the number of estimable parameters. The standard error reported for the parameter estimates is the square root of the corresponding diagonal element of this matrix. If you specify a value for the residual variance with the **SIGSQ=** option, then that value replaces mse in the preceding expressions.

Now suppose that restrictions or bounds are active. Equality restrictions can be written as a vector function, $h(\boldsymbol{\theta}) = \mathbf{0}$. Inequality restrictions are either active or inactive. When an inequality restriction is active, it is treated as an equality restriction.

Assume that the vector $h(\boldsymbol{\theta})$ contains the current active restrictions. The constraint matrix \mathbf{A} is then

$$\mathbf{A}(\hat{\boldsymbol{\theta}}) = \frac{\partial h(\hat{\boldsymbol{\theta}})}{\partial \hat{\boldsymbol{\theta}}}$$

The covariance matrix for the restricted parameter estimates is computed as

$$\mathbf{Z}(\mathbf{Z}'\mathbf{H}\mathbf{Z})^{-1}\mathbf{Z}'$$

where \mathbf{H} is the Hessian (or approximation to the Hessian) and \mathbf{Z} collects the last $(p - n_c)$ columns of \mathbf{Q} from an LQ factorization of the constraint matrix. Further, n_c is the number of active constraints, and p denotes the number of parameters. See Gill, Murray, and Wright (1981) for more details about the LQ factorization. The covariance matrix for the Lagrange multipliers is computed as

$$(\mathbf{A}\mathbf{H}^{-1}\mathbf{A}')^{-1}$$

In addition to the default covariance matrix estimate, the **BOOTSTRAP** statement can be used to produce bootstrap covariance matrix estimate. See the section “**Bootstrap Resampling and Estimation**” on page 6451 for details.

Convergence Measures

The NLIN procedure computes and reports four convergence measures, labeled R, PPC, RPC, and OBJECT.

R is the primary convergence measure for the parameters. It measures the degree to which the residuals are orthogonal to the columns of \mathbf{X} , and it approaches 0 as the gradient of the objective function becomes small. R is defined as

$$\sqrt{\frac{\mathbf{r}'\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{r}}{\text{LOSS}^i}}$$

PPC is the prospective parameter change measure. PPC measures the maximum relative change in the parameters implied by the parameter-change vector computed for the next iteration. At the k th iteration, PPC is the maximum over the parameters

$$\frac{|\widehat{\beta}_i^{(k+1)} - \widehat{\beta}_i^{(k)}|}{|\widehat{\beta}_i^{(k)}| + 1\text{E} - 6}$$

where $\widehat{\beta}_i^{(k)}$ is the current value of the i th parameter and $\widehat{\beta}_i^{(k+1)}$ is the prospective value of this parameter after adding the change vector computed for the next iteration. These changes are measured before step length adjustments are made. The parameter with the maximum prospective relative change is displayed with the value of PPC, unless the PPC is nearly 0.

RPC is the retrospective parameter change measure. RPC measures the maximum relative change in the parameters from the previous iteration. At the k th iteration, RPC is the maximum over i of

$$\frac{|\widehat{\beta}_i^{(k)} - \widehat{\beta}_i^{(k-1)}|}{|\widehat{\beta}_i^{(k-1)}| + 1\text{E} - 6}$$

where $\widehat{\beta}_i^{(k)}$ is the current value of the i th parameter and $\widehat{\beta}_i^{(k-1)}$ is the previous value of this parameter. These changes are measured before step length adjustments are made. The name of the parameter with the maximum retrospective relative change is displayed with the value of RPC, unless the RPC is nearly 0.

OBJECT measures the relative change in the objective function value between iterations:

$$\frac{|O^{(k)} - O^{(k-1)}|}{|O^{(k-1)} + 1\text{E} - 6|}$$

where $O^{(k-1)}$ is the value of the objective function ($O^{(k)}$) from the previous iteration. This is the old `CONVERGEOBJ=` criterion.

Displayed Output

In addition to the output data sets, PROC NLIN also produces the following output objects:

- the residual sums of squares associated with all or some of the combinations of possible starting values of the parameters
- the estimates of the parameters and the residual sums of squares at each iteration
- the estimation summary table, which displays information about the estimation method, the number of observations in the analysis, the objective function, and convergence measures
- the analysis of variance table, including sums of squares for the “Model,” “Residual,” and “Total” sources of variation (“Corrected Total” or “Uncorrected Total”), and the model F test. Note that beginning in SAS[®] 9, only the uncorrected total SS is reported and the respective F test is based on the uncorrected total SS if PROC NLIN determines the model does not include an intercept. If PROC NLIN determines the model does include an intercept, only the corrected total SS is reported and the respective F test is based on the corrected total SS.
- the table of parameter estimates, which contains for each parameter in the model its estimate, the approximate standard error of the estimate, and a 95% confidence interval based on the approximate standard error. The confidence level can be changed with the `ALPHA=` option in the `PROC NLIN` statement. The `HOUGAARD` option in the `PROC NLIN` statement requests that Hougaard’s skewness measure be added for each parameter. The standard errors and confidence limits are labeled approximate because they are valid asymptotically as the number of observations grows. If your model is linear in the parameters, the standard errors and confidence intervals are not approximate.
- the approximate correlation matrix of the parameter estimates. This correlation matrix is labeled approximate because it is computed from the approximate covariance matrix of the parameter estimates. If your model is linear in the parameters, the correlation matrix is not approximate.

Incompatibilities with SAS 6.11 and Earlier Versions of PROC NLIN

The NLIN procedure now uses a compiler that is different from the DATA step compiler. The compiler was changed so that analytical derivatives could be computed automatically. For the most part, the syntax accepted by the old NLIN procedure can be used in the new NLIN procedure. However, there are several differences that should be noted:

- You cannot specify a character index variable in the DO statement, and you cannot specify a character test in the IF statement. Thus `do i=1,2,3;` is supported, but `do i='ONE','TWO','THREE';` is not supported. And `if 'THIS' < 'THAT' then ...;` is supported, but `if 'THIS' THEN ...;` is not supported.
- The PUT statement, which is used mostly for program debugging in PROC NLIN, 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.
 - The PUT statement does not support line pointers, factored lists, iteration factors, overprinting, the `_INFILE_` option, the `‘:’` format modifier, or the symbol `‘$’`.
 - The PUT statement does support expressions inside of parentheses. For example, `put (sqrt(x));` produces the square root of X.
 - The PUT statement also supports the option `_PDV_` to display a formatted listing of all the variables in the program. The statement `put _pdv_;` prints a much more readable listing of the variables than `put _all_;` does.
- You cannot use the `‘*’` subscript, but you can specify an array name in a PUT statement without subscripts. Thus, `array a ...; put a;` is acceptable, but `put a[*];` is not. The statement `put a;` displays all the elements of the array a. The `put a=;` statement displays all the elements of A with each value labeled by the name of the element variable.
- You cannot specify arguments in the ABORT statement.
- You can specify more than one target statement in the WHEN and OTHERWISE statements. That is, DO/END groups are not necessary for multiple WHEN statements, such as `select; when(exp1); stmt1; stmt2; when(exp2); stmt3; stmt4; end;`.
- You can specify only the options LOG, PRINT, and LIST in the FILE statement.
- The RETAIN statement retains only values across one pass through the data set. If you need to retain values across iterations, use the CONTROL statement to make a control variable.

The ARRAY statement in PROC NLIN is similar to, but not the same as, the ARRAY statement in the DATA step. The ARRAY statement is used to associate a name (of no more than 8 characters) 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 NLIN does not support all the features of the DATA step ARRAY statement. You cannot specify implicit indexing variables; all array references must have explicit subscript expressions. You can specify simple array dimensions; lower bound specifications are not supported. A maximum of six dimensions are accepted.

On the other hand, the ARRAY statement supported by PROC NLIN does accept both variables and constants as array elements. In the following statements, b is a constant array and c is a variable array. Note that the constant array elements cannot be changed with assignment statements.

```
proc nlin data=nld;
array b[4] 1 2 3 4;      /* Constant array */
array c[4] ( 1 2 3 4 ); /* Numeric array with initial values */
```

```

b[1] = 2;           /* This is an ERROR, b is a constant array*/
c[2] = 7.5;         /* This is allowed */

```

Both dimension specification and the list of elements are optional, but at least one must be specified. When the list of elements is not specified, 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.

If the array is used as a pure array in the program rather than a list of symbols (the individual symbols of the array are not referenced in the code), the array is converted to a numerical array. A pure array is literally a vector of numbers that are accessed only by index. Using these types of arrays results in faster derivatives and compiled code. The assignment to c1 in the following statements forces the array to be treated as a list of symbols:

```

proc nlin data=nld;
array c[4] ( 1 2 3 4 ); /* Numeric array with initial values */

c[2] = 7.5;             /* This is C used as a pure array */
c1 = -92.5;             /* This forces C to be a list of symbols */

```

ODS Table Names

PROC NLIN assigns a name to each table it creates. You can use these names to refer to the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 81.9. For more information about ODS, see Chapter 20, “Using the Output Delivery System.”

Table 81.9 ODS Tables Produced by PROC NLIN

ODS Table Name	Description	Statement
ANOVA	Analysis of variance	default
BootstrapCorr	Bootstrap estimate of correlation matrix of the parameters	BOOTSTRAP
BootstrapCov	Bootstrap estimate of covariance matrix of the parameters	BOOTSTRAP
CodeDependency	Variable cross reference	LISTDEP
CodeList	Listing of program statements	LISTCODE
ConvergenceStatus	Convergence status	default
CorrB	Correlation of the parameters	default
EstSummary	Summary of the estimation	default
FirstDerivatives	First derivative table	LISTDER
IterHistory	Iteration output	default
MissingValues	Missing values generated by the program	default
NonlinearityMeasures	Global nonlinearity measures	NLINMEASURES
ParameterEstimates	Parameter estimates	default

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 21, “[Statistical Graphics Using ODS](#).”

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section “[Enabling and Disabling ODS Graphics](#)” on page 609 in Chapter 21, “[Statistical Graphics Using ODS](#).”

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section “[A Primer on ODS Statistical Graphics](#)” on page 608 in Chapter 21, “[Statistical Graphics Using ODS](#).”

PROC NLIN assigns a name to each graph it creates using ODS. You can use these names to refer to the graphs when using ODS. The graphs that are controlled by the **PLOTS** option in the **PROC NLIN** statement are listed in [Table 81.10](#), those that are controlled by the options in **PROFILE** statement are in [Table 81.11](#).

Table 81.10 Graphs Controlled by the PLOTS option in the PROC NLIN Statement

ODS Graph Name	Plot Description	PLOTS Option
ContourFitPlot	Contour fit plot for models with two regressors	FIT
FitPlot	Fit plot for models with one regressor	FIT
FitDiagnosticsPanel	Panel of fit diagnostics	DIAGNOSTICS
LeveragePlot	Tangential and Jacobian leverages versus observation number	DIAGNOSTICS
LocalInfluencePlot	Local influence versus observation number	DIAGNOSTICS
ObservedByPredictedPlot	Dependent variable versus predicted values	DIAGNOSTICS(UNPACK)
ProjectedResidualHistogram	A histogram of the projected residuals	DIAGNOSTICS(UNPACK)
RawResidualExpectationPlot	Raw residual expectation versus predicted values	DIAGNOSTICS(UNPACK)
RawResidualHistogram	A histogram of the raw residuals	DIAGNOSTICS(UNPACK)
ResidualBoxPlot	A box plot of the raw and projected residuals	DIAGNOSTICS(UNPACK)
ResidualPanel	A panel of the raw and projected residuals versus the regressors	RESIDUALS
ResidualPlot	A plot of the raw and projected residuals versus the regressors	RESIDUALS(UNPACK)
ResidualByPredictedPlot	Raw and projected residuals versus the predicted values	DIAGNOSTICS(UNPACK)
RStudentByJacLeveragePlot	Standardized raw and projected residuals versus Jacobian leverage	DIAGNOSTICS(UNPACK)
RStudentByPredictedPlot	Standardized raw and projected residuals versus the predicted values	DIAGNOSTICS(UNPACK)
RStudentByTanLeveragePlot	Standardized raw and projected residuals versus tangential leverage	DIAGNOSTICS(UNPACK)

Table 81.11 Graphs Controlled by the PROFILE Statement

ODS Graph Name	Plot Description	PROFILE Option
ConfidenceCurve	Parameter value versus t value	CONFCURV
JackknifePlot	Absolute relative percentage difference versus observation number	JACKKNIFE
ProfiletPlot	Likelihood ratio pivotal statistic versus Wald pivotal statistic	TPLOT

Table 81.12 Graphs Controlled by the BOOTSTRAP Statement

ODS Graph Name	Plot Description	BOOTSTRAP Option
BootstrapHistPlot	A histogram of bootstrap parameter estimates	BOOTPLOTS(HIST)
BootstrapScatterPlot	Pairwise scatter plot of bootstrap parameter estimates	BOOTPLOTS(SCATTER)

Convergence Status Table

The “Convergence Status” table can be used to programmatically check the status of an estimation. This table contains the Status variable that takes on the value 0, 1, 2, or 3. If Status takes on a value less than 3, the convergence criterion was met. Specifically, the values mean the following:

Status=0	indicates that the convergence criterion was met and no warning or error messages were issued during the PROC NLIN run. Also, no notes that could indicate a problem with the model were issued.
Status=1	indicates that the convergence criterion was met and notes were written to the log that might indicate a problem with the model.
Status=2	indicates that the convergence criterion was met and one or more warning messages were produced during the PROC NLIN run.
Status=3	indicates that the convergence criterion was not met.

The following sample program demonstrates how the “Convergence Status” table can be used:

```
ods output ConvergenceStatus=ConvStatus;
proc nlin data=YourData;
  parameters a=1 b=1 c=1;
  model wgt = a + x / (b*y+c*z);
run;

data _null_;
  set ConvStatus;
  if status > 0 then put "A problem occurred";
run;
```


Examples: NLIN Procedure

Example 81.1: Segmented Model

Suppose you are interested in fitting a model that consists of two segments that connect in a smooth fashion. For example, the following model states that for values of x less than x_0 the mean of Y is a quadratic function in x , and for values of x greater than x_0 the mean of Y is constant:

$$E[Y|x] = \begin{cases} \alpha + \beta x + \gamma x^2 & \text{if } x < x_0 \\ c & \text{if } x \geq x_0 \end{cases}$$

In this model equation α , β , and γ are the coefficients of the quadratic segment, and c is the plateau of the mean function. The NLIN procedure can fit such a segmented model even when the join point, x_0 , is unknown.

We also want to impose conditions on the two segments of the model. First, the curve should be continuous—that is, the quadratic and the plateau section need to meet at x_0 . Second, the curve should be smooth—that is, the first derivative of the two segments with respect to x need to coincide at x_0 .

The continuity condition requires that

$$p = E[Y|x_0] = \alpha + \beta x_0 + \gamma x_0^2$$

The smoothness condition requires that

$$\frac{\partial E[Y|x_0]}{\partial x} = \beta + 2\gamma x_0 \equiv 0$$

If you solve for x_0 and substitute into the expression for c , the two conditions jointly imply that

$$\begin{aligned} x_0 &= -\beta/2\gamma \\ c &= \alpha - \beta^2/4\gamma \end{aligned}$$

Although there are apparently four unknowns, the model contains only three parameters. The continuity and smoothness restrictions together completely determine one parameter given the other three.

The following DATA step creates the SAS data set for this example:

```
data a;
  input y x @@;
  datalines;
.46 1 .47 2 .57 3 .61 4 .62 5 .68 6 .69 7
.78 8 .70 9 .74 10 .77 11 .78 12 .74 13 .80 13
.80 15 .78 16
;
```

The following PROC NLIN statements fit this segmented model:

```

title 'Quadratic Model with Plateau';
proc nlin data=a;
  parms alpha=.45 beta=.05 gamma=-.0025;

  x0 = -.5*beta / gamma;

  if (x < x0) then
    mean = alpha + beta*x + gamma*x*x;
  else mean = alpha + beta*x0 + gamma*x0*x0;
  model y = mean;

  if _obs_=1 and _iter_ =. then do;
    plateau =alpha + beta*x0 + gamma*x0*x0;
    put / x0= plateau= ;
  end;
  output out=b predicted=yp;
run;

```

The parameters of the model are α , β , and γ , respectively. They are represented in the PROC NLIN statements by the variables alpha, beta, and gamma, respectively. In order to model the two segments, a conditional statement is used that assigns the appropriate expression to the mean function depending on the value of x_0 . A PUT statement is used to print the constrained parameters every time the program is executed for the first observation. The **OUTPUT** statement computes predicted values for plotting and saves them to data set b.

Note that there are other ways in which you can write the conditional expressions for this model. For example, you could formulate a condition with two model statements, as follows:

```

proc nlin data=a;
  parms alpha=.45 beta=.05 gamma=-.0025;
  x0 = -.5*beta / gamma;
  if (x < x0) then
    model y = alpha+beta*x+gamma*x*x;
  else model y = alpha+beta*x0+gamma*x0*x0;
run;

```

Or you could use a single expression with a conditional evaluation, as in the following statements:

```

proc nlin data=a;
  parms alpha=.45 beta=.05 gamma=-.0025;
  x0 = -.5*beta / gamma;
  model y = (x < x0)*(alpha+beta*x +gamma*x*x) +
            (x >= x0)*(alpha+beta*x0+gamma*x0*x0);
run;

```

The results from fitting this model with PROC NLIN are shown in [Output 81.1.1–Output 81.1.3](#). The iterative optimization converges after six iterations ([Output 81.1.1](#)). [Output 81.1.1](#) indicates that the join point is 12.747 and the plateau value is 0.777.

Output 81.1.1 Nonlinear Least-Squares Iterative Phase**Quadratic Model with Plateau**

The NLIN Procedure
Dependent Variable y
Method: Gauss-Newton

Iterative Phase				
Iter	alpha	beta	gamma	Sum of Squares
0	0.4500	0.0500	-0.00250	0.0562
1	0.3881	0.0616	-0.00234	0.0118
2	0.3930	0.0601	-0.00234	0.0101
3	0.3922	0.0604	-0.00237	0.0101
4	0.3921	0.0605	-0.00237	0.0101
5	0.3921	0.0605	-0.00237	0.0101
6	0.3921	0.0605	-0.00237	0.0101

NOTE: Convergence criterion met.

Output 81.1.2 Results from Put Statement

x0=12.747669162 plateau=0.7774974276

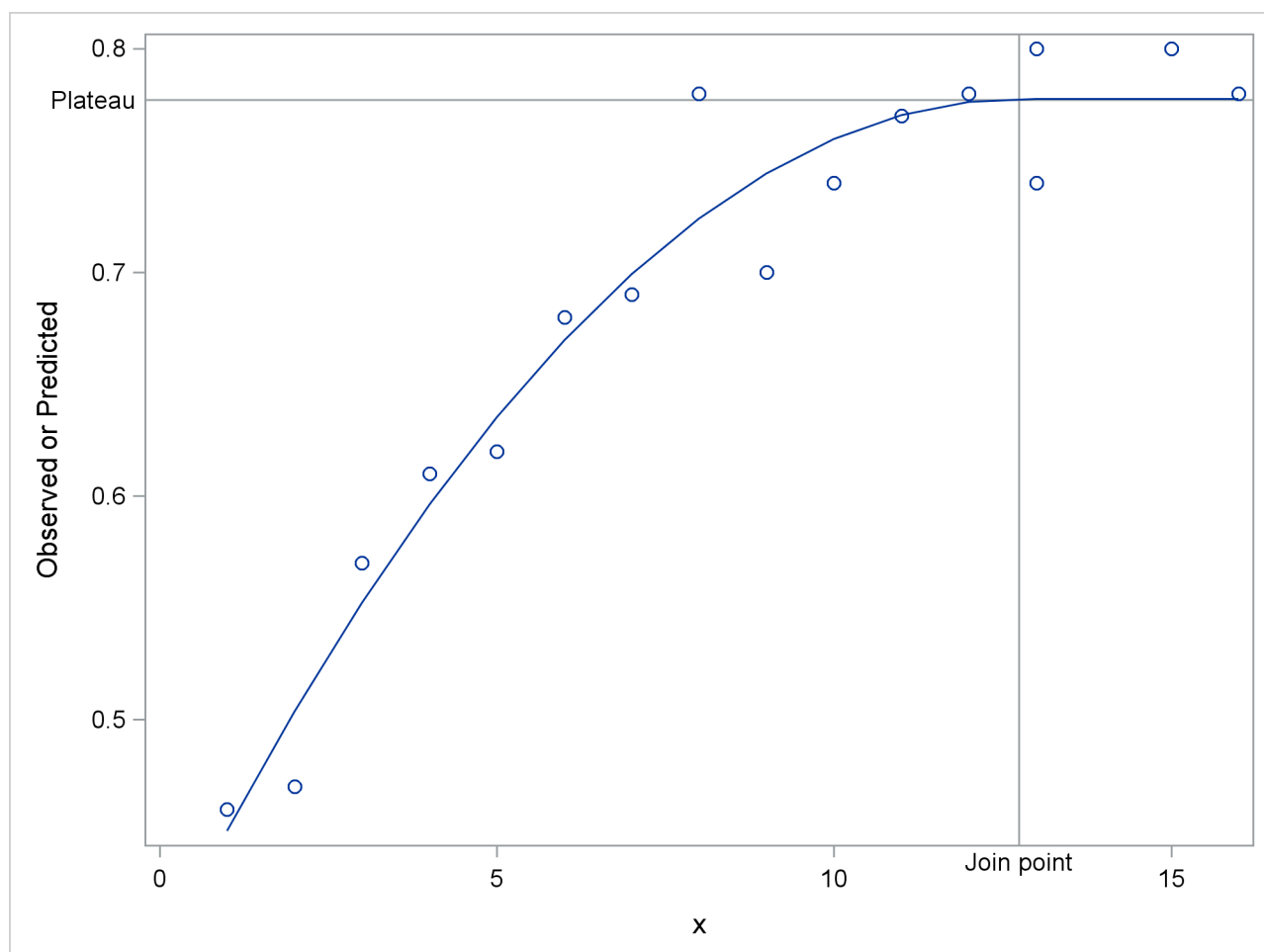
Output 81.1.3 Least-Squares Analysis for the Quadratic Model

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	2	0.1769	0.0884	114.22	<.0001
Error	13	0.0101	0.000774		
Corrected Total	15	0.1869			

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits	
alpha	0.3921	0.0267	0.3345	0.4497
beta	0.0605	0.00842	0.0423	0.0787
gamma	-0.00237	0.000551	-0.00356	-0.00118

The following statements produce a graph of the observed and predicted values with reference lines for the join point and plateau estimates (Output 81.1.4):

```
proc sgplot data=b noautolegend;
  yaxis label='Observed or Predicted';
  refline 0.777 / axis=y label="Plateau"    labelpos=min;
  refline 12.747 / axis=x label="Join point" labelpos=min;
  scatter y=y x=x;
  series y=yp x=x;
run;
```

Output 81.1.4 Observed and Predicted Values for the Quadratic Model

If you want to estimate the join point directly, you can use the relationship between the parameters to change the parameterization of the model in such a way that the mean function depends directly on x_0 . Using the smoothness condition that relates x_0 to γ ,

$$x_0 = -\beta/2\gamma$$

you can express γ as a function of β and x_0 :

$$\gamma = -\beta/(2x_0)$$

Substituting for γ in the model equation

$$E[Y|x] = \begin{cases} \alpha + \beta x + \gamma x^2 & \text{if } x < x_0 \\ \alpha - \beta^2/(4\gamma) & \text{if } x \geq x_0 \end{cases}$$

yields the reparameterized model

$$E[Y|x] = \begin{cases} \alpha + \beta x(1 - x/(2x_0)) & \text{if } x < x_0 \\ \alpha + \beta x_0/2 & \text{if } x \geq x_0 \end{cases}$$

This model is fit with the following PROC NLIN statements:

```
proc nlin data=a;
  parms alpha=.45 beta=.05 x0=10;
  if (x<x0) then
    mean = alpha + beta*x *(1-x/(2*x0));
  else mean = alpha + beta*x0/2;
  model y = mean;
run;
```

Output 81.1.5 Results from Reparameterized Model

The NLIN Procedure Dependent Variable y Method: Gauss-Newton

NOTE: Convergence criterion met.

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	2	0.1769	0.0884	114.22	<.0001
Error	13	0.0101	0.000774		
Corrected Total	15	0.1869			

Parameter	Estimate	Approx Std Error	Approx 95% Confidence Limits
alpha	0.3921	0.0267	0.3345 0.4497
beta	0.0605	0.00842	0.0423 0.0787
x0	12.7477	1.2781	9.9864 15.5089

The analysis of variance table in the reparameterized model is the same as in the earlier analysis (compare [Output 81.1.5](#) and [Output 81.1.3](#)). Changing the parameterization of a model does not affect the fit. The “Parameter Estimates” table now shows x_0 as a parameter in the model. The estimate agrees with the earlier result that uses the PUT statement ([Output 81.1.2](#)). Since x_0 is now a model parameter, the NLIN procedure also reports its asymptotic standard error and its approximate 95% confidence interval.

Example 81.2: Iteratively Reweighted Least Squares

With the NLIN procedure you can perform weighted nonlinear least squares regression in situations where the weights are functions of the parameters. To minimize a weighted sum of squares, you assign an expression to the `_WEIGHT_` variable in your PROC NLIN statements. When the `_WEIGHT_` variable depends on the model parameters, the estimation technique is known as iteratively reweighted least squares (IRLS). In this situation you should employ the `NOHALVE` option in the `PROC NLIN` statement. Because the weights change from iteration to iteration, it is not reasonable to expect the *weighted* residual sum of squares to decrease between iterations. The `NOHALVE` option removes that restriction.

Examples where IRLS estimation is used include robust regression via M-estimation (Huber 1964, 1973), generalized linear models (McCullagh and Nelder 1989), and semivariogram fitting in spatial statistics (Schabenberger and Pierce 2002, Sect. 9.2). There are dedicated SAS/STAT procedures for robust regression (the `ROBUSTREG` procedure) and generalized linear models (the `GENMOD` and `GLIMMIX` procedures). Examples of weighted least squares fitting of a semivariogram function can be found in Chapter 122, “[The VARIOGRAM Procedure](#).”

In this example we show an application of PROC NLIN for M-estimation only to illustrate the connection between robust regression and weighted least squares. The `ROBUSTREG` procedure is the appropriate tool to fit these models with SAS/STAT software.

M-estimation was introduced by Huber (1964, 1973) to estimate location parameters robustly. Beaton and Tukey (1974) applied the idea of M-estimation in regression models and introduced the biweight (or bisquare) weight function. See Holland and Welsch (1977) for this and other robust methods. Consider a linear regression model of the form

$$E[Y_i|x] = \mathbf{x}_i' \boldsymbol{\beta} + \epsilon_i$$

In weighted least squares estimation you seek the parameters $\hat{\boldsymbol{\beta}}$ that minimize

$$\sum_{i=1}^n w_i (y_i - \mathbf{x}_i' \boldsymbol{\beta})^2 = \sum_{i=1}^n w_i e_i^2$$

where w_i is the weight associated with the i th observation. The *normal* equations of this minimization problem can be written as

$$\sum_{i=1}^n w_i e_i \mathbf{x}_i = \mathbf{0}$$

In M-estimation the corresponding equations take on the form

$$\sum_{i=1}^n \psi(e_i) \mathbf{x}_i = \mathbf{0}$$

where $\psi(\cdot)$ is a weighing function. The Beaton-Tukey biweight, for example, can be written as

$$\psi(e_i) = \begin{cases} e_i \left(1 - \left(\frac{e_i}{\sigma k}\right)^2\right)^2 & |e_i/\sigma| \leq k \\ 0 & |e_i/\sigma| > k \end{cases}$$

Substitution into the estimating equation for M-estimation yields weighted least squares equations

$$\sum_{i=1}^n \psi(e_i) \mathbf{x}_i = \sum_{i=1}^n w_i e_i \mathbf{x}_i = \mathbf{0}$$

$$w_i = \begin{cases} \left(1 - \left(\frac{e_i}{\sigma k}\right)^2\right)^2 & |e_i/\sigma| \leq k \\ 0 & |e_i/\sigma| > k \end{cases}$$

The biweight function involves two constants, σ and k . The scale σ can be fixed or estimated from the fit in the previous iteration. If σ is estimated, a robust estimator of scale is typically used. In this example σ is fixed at 2. A common value for the constant k is $k = 4.685$.

The following DATA step creates a SAS data set of the population of the United States (in millions), recorded at 10-year intervals starting in 1790 and ending in 1990. The aim is to fit a quadratic linear model to the population over time.

```

title 'U.S. Population Growth';
data uspop;
    input pop :6.3 @@;
    retain year 1780;
    year    = year+10;
    yearsq = year*year;
    datalines;
3929 5308 7239 9638 12866 17069 23191 31443 39818 50155
62947 75994 91972 105710 122775 131669 151325 179323 203211
226542 248710
;

```

The PROC NLIN code that follows fits this linear model by M-estimation and IRLS. The weight function is set to a zero or nonzero value depending on the value of the scaled residual. The **NOHALVE** option removes the requirement that the (weighted) residual sum of squares must decrease between iterations.

```

title 'Beaton/Tukey Biweight Robust Regression using IRLS';
proc nlin data=uspop nohalve;
    parms b0=20450.43 b1=-22.7806 b2=.0063456;
    model pop=b0+b1*year+b2*year*year;
    resid = pop-model.pop;
    sigma = 2;
    k      = 4.685;
    if abs(resid/sigma)<=k then _weight_=(1-(resid / (sigma*k))**2)**2;
    else _weight_=0;
    output out=c r=rbi;
run;

```

Parameter estimates from this fit are shown in [Output 81.2.1](#), and the computed weights at the final iteration are displayed in [Output 81.2.2](#). The observations for 1940 and 1950 are highly discounted because of their large residuals.

Output 81.2.1 Nonlinear Least-Squares Analysis**Beaton/Tukey Biweight Robust Regression using IRLS****The NLIN Procedure**

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	2	113564	56782.0	49454.5	<.0001
Error	18	20.6670	1.1482		
Corrected Total	20	113585			

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits	
b0	20828.7	259.4	20283.8	21373.6
b1	-23.2004	0.2746	-23.7773	-22.6235
b2	0.00646	0.000073	0.00631	0.00661

Output 81.2.2 Listing of Computed Weights from PROC NLIN

Obs	pop	year	yearsq	rbi	sigma	k	_weight_
1	3.929	1790	3204100	-0.93711	2	4.685	0.98010
2	5.308	1800	3240000	0.46091	2	4.685	0.99517
3	7.239	1810	3276100	1.11853	2	4.685	0.97170
4	9.638	1820	3312400	0.95176	2	4.685	0.97947
5	12.866	1830	3348900	0.32159	2	4.685	0.99765
6	17.069	1840	3385600	-0.62597	2	4.685	0.99109
7	23.191	1850	3422500	-0.94692	2	4.685	0.97968
8	31.443	1860	3459600	-0.43027	2	4.685	0.99579
9	39.818	1870	3496900	-1.08302	2	4.685	0.97346
10	50.155	1880	3534400	-1.06615	2	4.685	0.97427
11	62.947	1890	3572100	0.11332	2	4.685	0.99971
12	75.994	1900	3610000	0.25539	2	4.685	0.99851
13	91.972	1910	3648100	2.03607	2	4.685	0.90779
14	105.710	1920	3686400	0.28436	2	4.685	0.99816
15	122.775	1930	3724900	0.56725	2	4.685	0.99268
16	131.669	1940	3763600	-8.61325	2	4.685	0.02403
17	151.325	1950	3802500	-8.32415	2	4.685	0.04443
18	179.323	1960	3841600	-0.98543	2	4.685	0.97800
19	203.211	1970	3880900	0.95088	2	4.685	0.97951
20	226.542	1980	3920400	1.03780	2	4.685	0.97562
21	248.710	1990	3960100	-1.33067	2	4.685	0.96007

You can obtain this analysis more conveniently with PROC ROBUSTREG. The procedure re-estimates the scale parameter robustly between iterations. To obtain an analysis with a fixed scale parameter as in this example, use the following PROC ROBUSTREG statements:


```
proc robustreg data=uspop method=m(scale=2);
  model pop = year year*year;
  output out=weights weight=w;
run;

proc print data=weights;
run;
```

Note that the computation of standard errors in the ROBUSTREG procedure is different from the calculations in the NLIN procedure.

Example 81.3: Probit Model with Likelihood Function

The data in this example, taken from Lee (1974), consist of patient characteristics and a variable indicating whether cancer remission occurred. This example demonstrates how to use PROC NLIN with a likelihood function. In this case, twice the negative of the log-likelihood function is to be minimized. This is the objective function for the analysis:

$$-2 \log L = -2 \sum_{i=1}^n \log \{ \pi_i(y_i, \mathbf{x}_i) \}$$

In this expression, π_i denotes the success probability of the n Bernoulli trials, and $\log L$ is the log likelihood of n independent binary (Bernoulli) observations. The probability π_i depends on the observations through the linear predictor η_i ,

$$\pi_i(y_i, \mathbf{x}_i) = \begin{cases} 1 - \Phi(\eta_i) & y_i = 1 \\ \Phi(\eta_i) & y_i = 0 \end{cases}$$

The linear predictor takes the form of a regression equation that is linear in the parameters,

$$\eta_i = \beta_0 + \beta_1 z_{1i} + \beta_2 z_{2i} + \cdots + \beta_k z_{ki} = \mathbf{z}_i \boldsymbol{\beta}$$

Despite this linearity of η in the z variables, the probit model is nonlinear, because the linear predictor appears inside the nonlinear probit function.

In order to use the NLIN procedure to minimize the function, the estimation problem must be cast in terms of a nonlinear least squares problem with objective function

$$\sum_{i=1}^n (y_i - f(\boldsymbol{\beta}, \mathbf{z}'_i))^2$$

This can be accomplished by setting $y_i = 0$ and $f(\boldsymbol{\beta}, \mathbf{z}'_i) = \sqrt{-2 \log \{ \pi_i \}}$. Because $0 \leq \pi \leq 1$, the function $-2 \log \{ \pi_i \}$ is strictly positive and the square root can be taken safely.

The following DATA step creates the data for the probit analysis. The variable `like` is created in the DATA step, and it contains the value 0 throughout. This variable serves as the “dummy” response variable in the PROC NLIN step. The variable `remiss` indicates whether cancer remission occurred. It is the binary outcome variable of interest and is used to determine the relevant probability for observation i as the success or failure probability of a Bernoulli experiment.

```

data remiss;
  input remiss cell smear infil li blast temp;
  label remiss = 'complete remission';
  like = 0;
  label like = 'dummy variable for nlin';
  datalines;
1 0.8 .83 .66 1.9 1.10 .996
1 0.9 .36 .32 1.4 0.74 .992
0 0.8 .88 .70 0.8 0.176 .982
0 1 .87 .87 0.7 1.053 .986
1 0.9 .75 .68 1.3 0.519 .980
0 1 .65 .65 0.6 0.519 .982
1 0.95 .97 .92 1 1.23 .992
0 0.95 .87 .83 1.9 1.354 1.020
0 1 .45 .45 0.8 0.322 .999
0 0.95 .36 .34 0.5 0 1.038
0 0.85 .39 .33 0.7 0.279 .988
0 0.7 .76 .53 1.2 0.146 .982
0 0.8 .46 .37 0.4 0.38 1.006
0 0.2 .39 .08 0.8 0.114 .990
0 1 .90 .90 1.1 1.037 .990
1 1 .84 .84 1.9 2.064 1.020
0 0.65 .42 .27 0.5 0.114 1.014
0 1 .75 .75 1 1.322 1.004
0 0.5 .44 .22 0.6 0.114 .990
1 1 .63 .63 1.1 1.072 .986
0 1 .33 .33 0.4 0.176 1.010
0 0.9 .93 .84 0.6 1.591 1.020
1 1 .58 .58 1 0.531 1.002
0 0.95 .32 .30 1.6 0.886 .988
1 1 .60 .60 1.7 0.964 .990
1 1 .69 .69 0.9 0.398 .986
0 1 .73 .73 0.7 0.398 .986
;

```

The following NLIN statements fit the probit model:

```

proc nlin data=remiss method=newton sigsq=1;
  parms int=-10 a = -2 b = -1 c=6;

  linp = int + a*cell + b*li + c*temp;
  p = probnorm(linp);

  if (remiss = 1) then pi = 1-p;
  else pi = p;

  model.like = sqrt(- 2 * log(pi));
  output out=p p=predict;
run;

```

The assignment to the variable `linp` creates the linear predictor of the generalized linear model,

$$\eta = \beta_0 + \beta_1 \text{cell}_i + \beta_2 \text{li}_i + \beta_3 * \text{temp}_i$$

In this example, the variables `cell`, `li`, and `temp` are used as regressors.

By default, the NLIN procedure computes the covariance matrix of the parameter estimates based on the nonlinear least squares assumption. That is, the procedure computes the estimate of the residual variance as the mean squared error and uses that to multiply the inverse crossproduct matrix or the inverse Hessian matrix. (See the section “[Covariance Matrix of Parameter Estimates](#)” on page 6464 for details.) In the probit model, there is no residual variance. In addition, standard errors in maximum likelihood estimation are based on the inverse Hessian matrix. The `METHOD=NEWTON` option in the `PROC NLIN` statement is used to employ the Hessian matrix in computing the covariance matrix of the parameter estimates. The `SIGSQ=1` option replaces the residual variance estimate that PROC NLIN would use by default as a multiplier of the inverse Hessian with the value 1.0.

[Output 81.3.1](#) shows the results of this analysis. The analysis of variance table shows an apparently strange result. The total sum of squares is zero, and the model sum of squares is negative. Recall that the values of the response variable were set to zero and the mean function was constructed as $-2 \log\{\pi_i\}$ in order for the NLIN procedure to minimize a log-likelihood function in terms of a nonlinear least squares problem. The value 21.9002 shown as the “Error” sum of squares is the value of the function $-2 \log L$.

Output 81.3.1 Nonlinear Least-Squares Analysis from PROC NLIN

The NLIN Procedure

Note: An intercept was not specified for this model.

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	4	-21.9002	-5.4750	-5.75	.
Error	23	21.9002	0.9522		
Uncorrected Total	27	0			

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits	
int	-36.7548	32.3607	-103.7	30.1885
a	-5.6298	4.6376	-15.2235	3.9639
b	-2.2513	0.9790	-4.2764	-0.2262
c	45.1815	34.9095	-27.0343	117.4

The problem can be more simply solved using dedicated procedures for generalized linear models:

```
proc glimmix data=remiss;
    model remiss = cell li temp / dist=binary link=probit s;
run;

proc genmod data=remiss;
    model remiss = cell li temp / dist=bin link=probit;
run;

proc logistic data=remiss;
    model remiss = cell li temp / link=probit technique=newton;
run;
```

Example 81.4: Affecting Curvature through Parameterization

The work of Ratkowsky (1983, 1990) has brought into focus the importance of close-to-linear behavior of parameters in nonlinear regression models. The curvature in a nonlinear model consists of two components: the intrinsic curvature and the parameter-effects curvature. See the section “[Relative Curvature Measures of Nonlinearity](#)” on page 6448 for details. Intrinsic curvature expresses the degree to which the nonlinear model bends as values of the *parameters* change. This is not the same as the curviness of the model as a function of the covariates (the *x* variables). Intrinsic curvature is a function of the type of model you are fitting and the data. This curvature component cannot be affected by reparameterization of the model. According to Ratkowsky (1983), the intrinsic curvature component is typically smaller than the parameter-effects curvature, which can be affected by altering the parameterization of the model.

In models with low curvature, the nonlinear least squares parameter estimators behave similarly to least squares estimators in linear regression models, which have a number of desirable properties. If the model is correct, they are best linear unbiased estimators and are normally distributed if the model errors are normal (otherwise they are asymptotically normal). As you lower the curvature of a nonlinear model, you can expect that the parameter estimators approach the behavior of the linear regression model estimators: they behave “close to linear.”

This example uses a simple data set and a commonly applied model for dose-response relationships to examine how the parameter-effects curvature can be reduced. The statistics by which an estimator’s behavior is judged are Box’s bias (Box 1971) and Hougaard’s measure of skewness (Hougaard 1982, 1985).

The log-logistic model

$$E[Y|x] = \delta + \frac{\alpha - \delta}{1 + \gamma \exp\{\beta \ln(x)\}}$$

is a popular model to express the response Y as a function of dose x . The response is bounded between the asymptotes α and δ . The term in the denominator governs the transition between the asymptotes and depends on two parameters, γ and β . The log-logistic model can be viewed as a member of a broader class of dose-response functions, those relying on *switch-on* or *switch-off* mechanisms (see, for example, Schabenberger and Pierce 2002, sec. 5.8.6). A switch function is usually a monotonic function $S(x, \theta)$ that takes values between 0 and 1. A switch-on function increases in x ; a switch-off function decreases in x . In the log-logistic case, the function

$$S(x, [\beta, \gamma]) = \frac{1}{1 + \gamma \exp\{\beta \ln(x)\}}$$

is a switch-off function for $\beta > 0$ and a switch-on function for $\beta < 0$. You can write general dose-response functions with asymptotes simply as

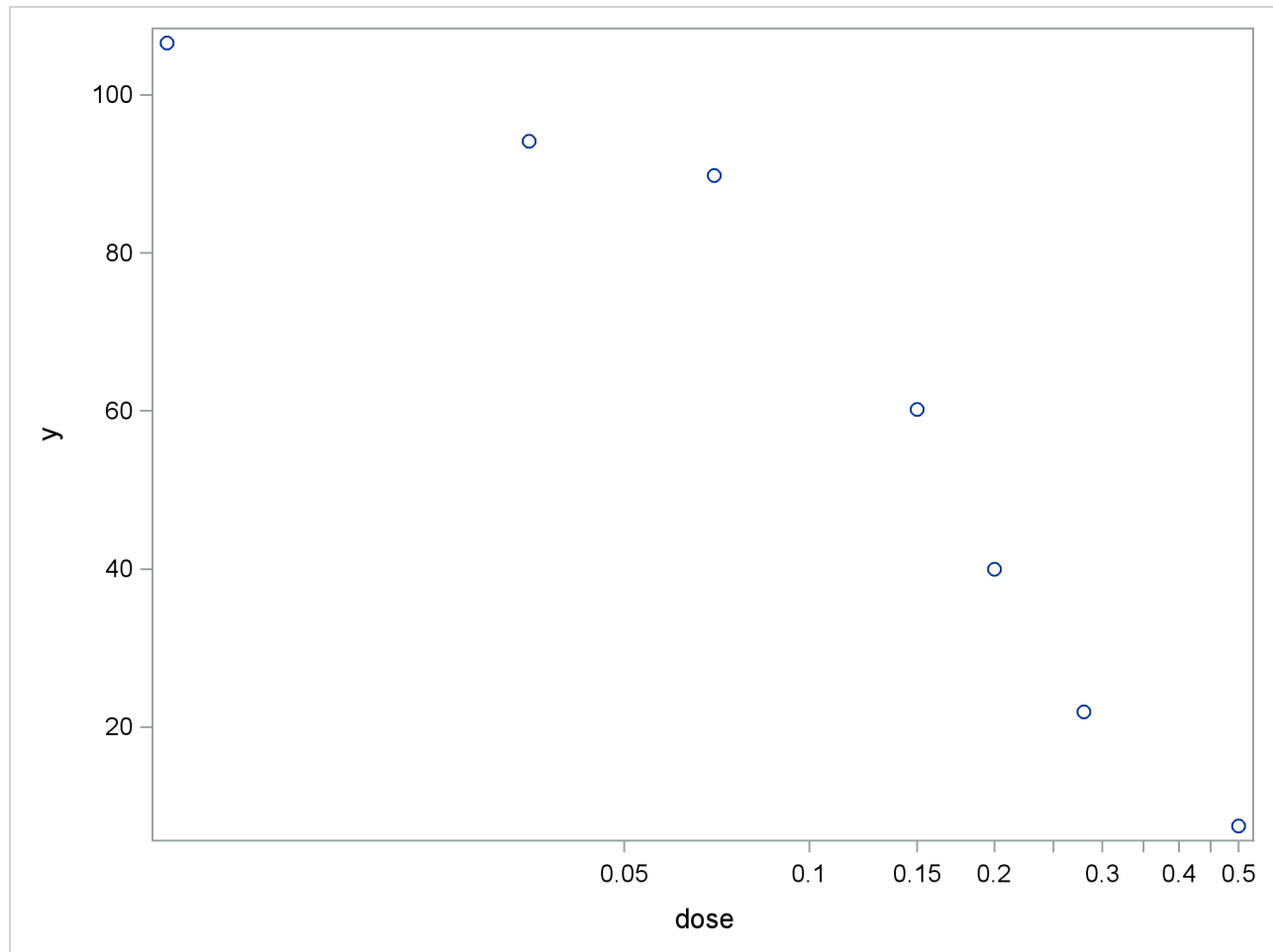
$$E[Y|x] = \mu_{\min} + (\mu_{\max} - \mu_{\min})S(x, \theta)$$

The following DATA step creates a small data set from a dose-response experiment with response y :

```
data logistic;
  input dose y;
  logdose = log(dose);
  datalines;
0.009 106.56
0.035 94.12
0.07 89.76
0.15 60.21
0.20 39.95
0.28 21.88
0.50 7.46
;
```

A graph of these data is produced with the following statements:

```
proc sgplot data=logistic;
  scatter y=y x=dose;
  xaxis type=log logstyle=linear;
run;
```

Output 81.4.1 Observed Data in Dose-Response Experiment

When dose is expressed on the log scale, the sigmoidal shape of the dose-response relationship is clearly visible ([Output 81.4.1](#)). The log-logistic switching model in the preceding parameterization is fit with the following statements in the NLIN procedure:

```
proc nlin data=logistic bias hougaard nlinmeasures;
  parameters alpha=100 beta=3 gamma=300;
  delta = 0;
  Switch = 1/(1+gamma*exp(beta*log(dose)));
  model y = delta + (alpha - delta)*Switch;
run;
```

The lower asymptote δ is assumed to be 0 in this case. Since δ is not listed in the [PARAMETERS](#) statement and is assigned a value in the program, it is assumed to be constant. Note that the term Switch is the switch-off function in the log-logistic model. The [BIAS](#) and [HOUGAARD](#) options in the [PROC NLIN](#) statement request that Box's bias, percentage bias, and Hougaard's skewness measure be added to the table of parameter estimates, and the [NLINMEASURES](#) option requests that the global nonlinearity measures be produced.

The NLIN procedure converges after 10 iterations and achieves a residual mean squared error of 15.1869 (Output 81.4.2). This value is not that important by itself, but it is worth noting since this model fit is compared to the fit with other parameterizations later on.

Output 81.4.2 Iteration History and Analysis of Variance

The NLIN Procedure
Dependent Variable y
Method: Gauss-Newton

Iterative Phase				
Iter	alpha	beta	gamma	Sum of Squares
0	100.0	3.0000	300.0	386.4
1	100.4	2.8011	162.8	129.1
2	100.8	2.6184	101.4	69.2710
3	101.3	2.4266	69.7579	68.2167
4	101.7	2.3790	69.0358	60.8223
5	101.8	2.3621	67.3709	60.7516
6	101.8	2.3582	67.0044	60.7477
7	101.8	2.3573	66.9150	60.7475
8	101.8	2.3571	66.8948	60.7475
9	101.8	2.3570	66.8902	60.7475
10	101.8	2.3570	66.8892	60.7475

NOTE: Convergence criterion met.

Note: An intercept was not specified for this model.

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	3	33965.4	11321.8	745.50	<.0001
Error	4	60.7475	15.1869		
Uncorrected Total	7	34026.1			

The table of parameter estimates displays the estimates of the three model parameters, their approximate standard errors, 95% confidence limits, Hougaard's skewness measure, Box's bias, and percentage bias (Output 81.4.3). Parameters for which the skewness measure is less than 0.1 in absolute value and with percentage bias less than 1% exhibit very close-to-linear behavior, and skewness values less than 0.25 in absolute value indicate reasonably close-to-linear behavior (Ratkowsky 1990). According to these rules, the estimators $\hat{\beta}$ and $\hat{\gamma}$ suffer from substantial curvature. The estimator $\hat{\gamma}$ is especially "far-from-linear." Inferences that involve $\hat{\gamma}$ and rely on the reported standard errors or confidence limits (or both) for this parameter might be questionable.

Output 81.4.3 Parameter Estimates, Hougaard's Skewness, and Box's Bias

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits		Skewness	Percent Bias	
						Bias	Bias
alpha	101.8	3.0034	93.4751	110.2	0.1415	0.1512	0.15
beta	2.3570	0.2928	1.5440	3.1699	0.4987	0.0303	1.29
gamma	66.8892	31.6146	-20.8870	154.7	1.9200	10.9230	16.3

The related global nonlinearity measures output table (Output 81.4.4) shows that both the maximum and RMS parameter-effects curvature are substantially larger than the critical curvature value recommended by Bates and Watts (1980). In contrast, the intrinsic curvatures of the model are less than the critical value. This implies that most of the nonlinearity can be removed by reparameterization.

Output 81.4.4 Global Nonlinearity Measures

Global Nonlinearity Measures	
Max Intrinsic Curvature	0.2397
RMS Intrinsic Curvature	0.1154
Max Parameter-Effects Curvature	4.0842
RMS Parameter-Effects Curvature	1.8198
Curvature Critical Value	0.3895
Raw Residual Variance	15.187
Projected Residual Variance	5.922

One method of reducing the parameter-effects curvature, and thereby reduce the bias and skewness of the parameter estimators, is to replace a parameter with its expected-value parameterization. Schabenberger et al. (1999) and Schabenberger and Pierce (2002, sec. 5.7.2) refer to this method as *reparameterization through defining relationships*. A defining relationship is obtained by equating the mean response at a chosen value of x (say, x^*) to the model:

$$E[Y|x^*] = \delta + \frac{\alpha - \delta}{1 + \gamma \exp\{\beta \ln(x^*)\}}$$

This equation is then solved for a parameter that is subsequently replaced in the original equation. This method is particularly useful if x^* has an interesting interpretation. For example, let λ_K denote the value that reduces the response by $K \times 100\%$,

$$E[Y|\lambda_K] = \delta + \left(\frac{100 - K}{100}\right)(\alpha - \delta)$$

Because γ exhibits large bias and skewness, it is the target in the first round of reparameterization. Setting the expression for the conditional mean at λ_K equal to the mean function when $x = \lambda_K$ yields the following expression:

$$\delta + \left(\frac{100 - K}{100}\right)(\alpha - \delta) = \delta + \frac{\alpha - \delta}{1 + \gamma \exp\{\beta \ln(\lambda_K)\}}$$

This expression is solved for γ , and the result is substituted back into the model equation. This leads to a log-logistic model in which γ is replaced by the parameter λ_K , the dose at which the response was reduced by $K \times 100\%$. The new model equation is

$$E[Y|x] = \delta + \frac{\alpha - \delta}{1 + K/(100 - K) \exp\{\beta \ln(x/\lambda_K)\}}$$

A particularly interesting choice is $K = 50$, since λ_{50} is the dose at which the response is halved. In studies of mortality, this concentration is also known as the LD50. For the special case of λ_{50} the model equation becomes

$$E[Y|x] = \delta + \frac{\alpha - \delta}{1 + \exp\{\beta \ln(x/\lambda_{50})\}}$$

You can fit the model in the LD50 parameterization with the following statements:

```
proc nlin data=logistic bias hougard;
  parameters alpha=100 beta=3 LD50=0.15;
  delta = 0;
  Switch = 1/(1+exp(beta*log(dose/LD50)));
  model y = delta + (alpha - delta)*Switch;
  output out=nlinout pred=p lcl=lcl ucl=ucl;
run;
```

Partial results from this NLIN run are shown in [Output 81.4.5](#). The analysis of variance tables in [Output 81.4.2](#) and [Output 81.4.5](#) are identical. Changing the parameterization of a model does not affect the model fit. It does, however, affect the interpretation of the parameters and the statistical properties (close-to-linear behavior) of the parameter estimators. The skewness and bias measures of the parameter LD50 is considerably reduced compared to those values for the parameter γ in the previous parameterization. Also, γ has been replaced by a parameter with a useful interpretation, the dose that yields a 50% reduction in mean response. Also notice that the bias and skewness measures of α and β are not affected by the $\gamma \rightarrow$ LD50 reparameterization.

Output 81.4.5 ANOVA Table and Parameter Estimates in LD50 Parameterization

The NLIN Procedure Dependent Variable y Method: Gauss-Newton

NOTE: Convergence criterion met.

Note: An intercept was not specified for this model.

Source	Sum of DF	Squares	Mean Square	F Value	Approx Pr > F
Model	3	33965.4	11321.8	745.50	<.0001
Error	4	60.7475	15.1869		
Uncorrected Total	7	34026.1			

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits	Skewness	Bias	Percent Bias
alpha	101.8	3.0034	93.4752 110.2	0.1415	0.1512	0.15
beta	2.3570	0.2928	1.5440 3.1699	0.4987	0.0303	1.29
LD50	0.1681	0.00915	0.1427 0.1935	-0.0605	-0.00013	-0.08

To reduce the parameter-effects curvature of the β parameter, you can use the technique of defining relationships again. This can be done generically, by solving

$$\mu^* = \delta + \frac{\alpha - \delta}{1 + \exp\{\beta \ln(x/\lambda)\}}$$

for β , treating μ^* as the new parameter (in lieu of β), and choosing a value for x^* that leads to low skewness. This results in the expected-value parameterization of β . Solving for β yields

$$\beta = \frac{\log\left(\frac{\alpha - \mu^*}{\mu^* - \delta}\right)}{\log(x^*/\lambda)}$$

The interpretation of the parameter μ^* that replaces β in the model equation is simple: it is the mean dose response when the dose is x^* . Fixing $x^* = 0.3$, the following PROC NLIN statements fit this model:

```
proc nlin data=logistic bias hougard nlinmeasures;
  parameters alpha=100 mustar=20 LD50=0.15;
  delta      = 0;
  xstar      = 0.3;
  beta       = log((alpha - mustar)/(mustar - delta)) / log(xstar/LD50);
  Switch     = 1/(1+exp(beta*log(dose/LD50)));
  model y = delta + (alpha - delta)*Switch;
  output out=nlinout pred=p lcl=lcl ucl=ucl;
run;
```

Note that the switch-off function continues to be written in terms of β and the LD50. The only difference from the previous model is that β is now expressed as a function of the parameter μ^* . Using expected-value parameterizations is a simple mechanism to lower the curvature in a model and to arrive at starting values. The starting value for μ^* can be gleaned from [Output 81.4.1](#) at $x = 0.3$.

[Output 81.4.6](#) shows selected results from this NLIN run. The ANOVA table is again unaffected by the change in parameterization. The skewness for μ^* is significantly reduced in comparison to those of the β parameter in the previous model (compare [Output 81.4.6](#) and [Output 81.4.5](#)), while its bias remains on the same scale from [Output 81.4.5](#) to [Output 81.4.6](#). Also note the substantial reduction in the parameter-effects curvature values. As expected, the intrinsic curvature values remain intact.

Output 81.4.6 ANOVA Table and Parameter Estimates in Expected-Value Parameterization

The NLIN Procedure
Dependent Variable y
Method: Gauss-Newton

NOTE: Convergence criterion met.

Note: An intercept was not specified for this model.

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	3	33965.4	11321.8	745.50	<.0001
Error	4	60.7475	15.1869		
Uncorrected Total	7	34026.1			

Output 81.4.6 *continued*

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits		Skewness	Bias	Percent Bias
alpha	101.8	3.0034	93.4752	110.2	0.1415	0.1512	0.15
mustar	20.7073	2.6430	13.3693	28.0454	-0.0572	-0.0983	-0.47
LD50	0.1681	0.00915	0.1427	0.1935	-0.0605	-0.00013	-0.08

Global Nonlinearity Measures	
Max Intrinsic Curvature	0.2397
RMS Intrinsic Curvature	0.1154
Max Parameter-Effects Curvature	0.2925
RMS Parameter-Effects Curvature	0.1500
Curvature Critical Value	0.3895
Raw Residual Variance	15.187
Projected Residual Variance	5.9219

Example 81.5: Comparing Nonlinear Trends among Groups

When you model nonlinear trends in the presence of group (classification) variables, two questions often arise: whether the trends should be varied by group, and how to decide which parameters should be varied across groups. A large battery of tools is available on linear statistical models to test hypotheses involving the model parameters, especially to test linear hypotheses. To test similar hypotheses in nonlinear models, you can draw on analogous tools. Especially important in this regard are comparisons of nested models by contrasting their residual sums of squares.

In this example, a two-group model from a pharmacokinetic application is fit to data that are in part based on the theophylline data from Pinheiro and Bates (1995) and the first example in the documentation for the NLMIXED procedure. In a pharmacokinetic application you study how a drug is dispersed through a living organism. The following data represent concentrations of the drug theophylline over a 25-hour period following oral administration. The data are derived by collapsing and averaging the subject-specific data from Pinheiro and Bates (1995) in a particular, yet unimportant, way. The purpose of arranging the data in this way is purely to demonstrate the methodology.

```

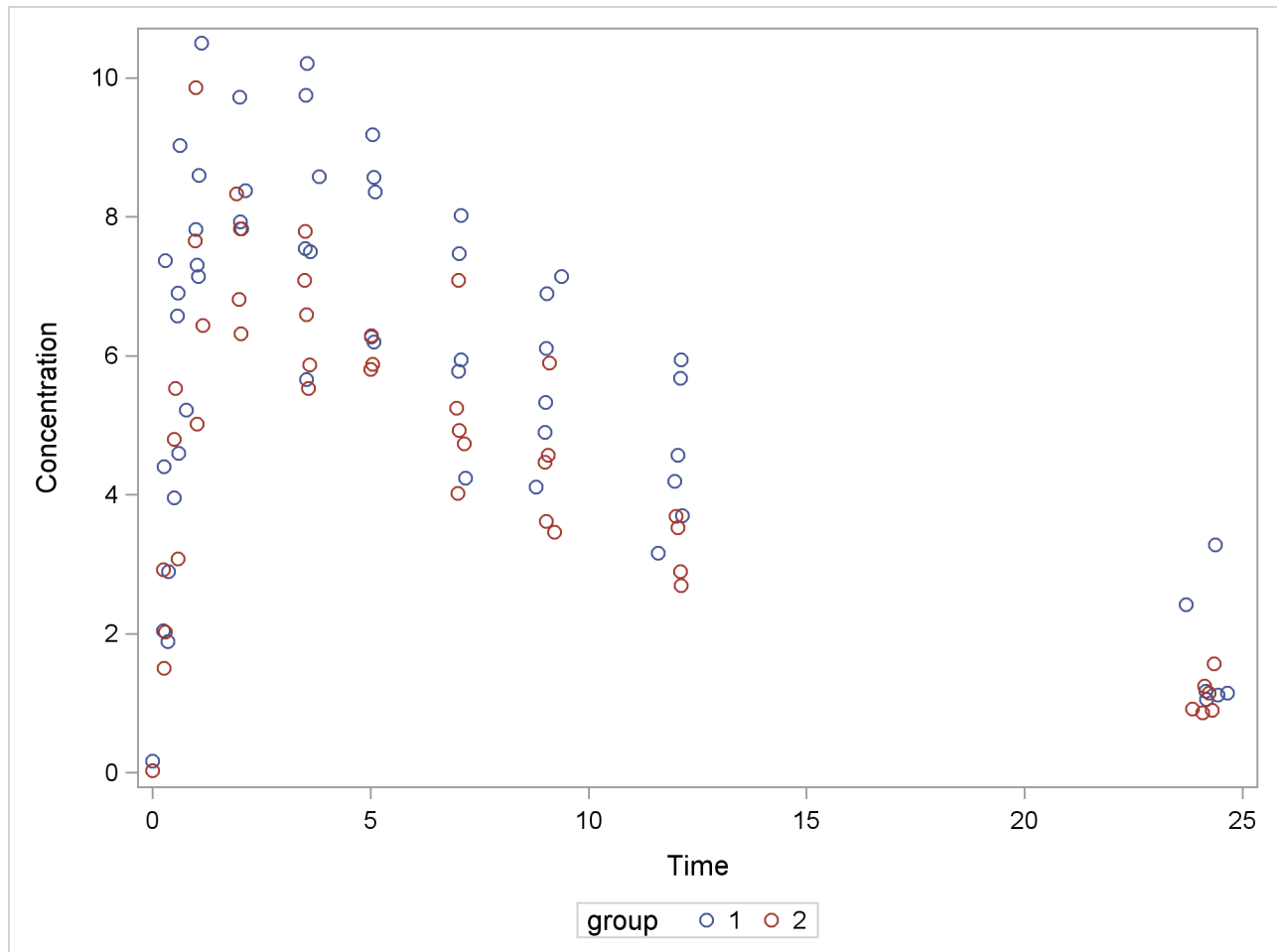
data theop;
  input time dose conc @@;
  if (dose = 4) then group=1; else group=2;
  datalines;
0.00 4 0.1633 0.25 4 2.045
0.27 4 4.4 0.30 4 7.37
0.35 4 1.89 0.37 4 2.89
0.50 4 3.96 0.57 4 6.57
0.58 4 6.9 0.60 4 4.6
0.63 4 9.03 0.77 4 5.22
1.00 4 7.82 1.02 4 7.305
1.05 4 7.14 1.07 4 8.6
1.12 4 10.5 2.00 4 9.72
2.02 4 7.93 2.05 4 7.83
2.13 4 8.38 3.50 4 7.54
3.52 4 9.75 3.53 4 5.66
3.55 4 10.21 3.62 4 7.5
3.82 4 8.58 5.02 4 6.275
5.05 4 9.18 5.07 4 8.57
5.08 4 6.2 5.10 4 8.36
7.02 4 5.78 7.03 4 7.47
7.07 4 5.945 7.08 4 8.02
7.17 4 4.24 8.80 4 4.11
9.00 4 4.9 9.02 4 5.33
9.03 4 6.11 9.05 4 6.89
9.38 4 7.14 11.60 4 3.16
11.98 4 4.19 12.05 4 4.57
12.10 4 5.68 12.12 4 5.94
12.15 4 3.7 23.70 4 2.42
24.15 4 1.17 24.17 4 1.05
24.37 4 3.28 24.43 4 1.12
24.65 4 1.15 0.00 5 0.025
0.25 5 2.92 0.27 5 1.505
0.30 5 2.02 0.50 5 4.795
0.52 5 5.53 0.58 5 3.08
0.98 5 7.655 1.00 5 9.855
1.02 5 5.02 1.15 5 6.44
1.92 5 8.33 1.98 5 6.81
2.02 5 7.8233 2.03 5 6.32
3.48 5 7.09 3.50 5 7.795
3.53 5 6.59 3.57 5 5.53
3.60 5 5.87 5.00 5 5.8
5.02 5 6.2867 5.05 5 5.88
6.98 5 5.25 7.00 5 4.02
7.02 5 7.09 7.03 5 4.925
7.15 5 4.73 9.00 5 4.47
9.03 5 3.62 9.07 5 4.57
9.10 5 5.9 9.22 5 3.46
12.00 5 3.69 12.05 5 3.53
12.10 5 2.89 12.12 5 2.69
23.85 5 0.92 24.08 5 0.86
24.12 5 1.25 24.22 5 1.15
24.30 5 0.9 24.35 5 1.57
;

```

The following code plots the theophylline concentration data over time for the two groups ([Output 81.5.1](#)). In each group the concentration tends to rise sharply right after the drug is administered, followed by a prolonged tapering of the concentration.

```
proc sgplot data=theop;
  scatter x=time y=conc / group=group;
  yaxis label='Concentration';
  xaxis label='Time';
run;
```

Output 81.5.1 Observed Responses in Two Groups



In the context of nonlinear mixed models, Pinheiro and Bates (1995) consider a first-order compartment model for these data. In terms of two fixed treatment groups, the model can be written as

$$C_{it} = \frac{Dk_{e_i}k_{a_i}}{Cl_i(k_{a_i} - k_{e_i})}[\exp(-k_{e_i}t) - \exp(-k_{a_i}t)] + \epsilon_{it}$$

where C_{it} is the observed concentration in group i at time t , D is the dose of theophylline, k_{e_i} is the elimination rate in group i , k_{a_i} is the absorption rate in group i , Cl_i is the clearance in group i , and ϵ_{it} denotes the model error. Because the rates and the clearance must be positive, you can parameterize the model in terms of log rates and the log clearance:

$$\begin{aligned} Cl_i &= \exp\{\beta_{1i}\} \\ k_{a_i} &= \exp\{\beta_{2i}\} \\ k_{e_i} &= \exp\{\beta_{3i}\} \end{aligned}$$

In this parameterization the model contains six parameters, and the rates and clearance vary by group. This produces two separate response profiles, one for each group. On the other extreme, you could model the trends as if there were no differences among the groups:

$$\begin{aligned} Cl_i &= \exp\{\beta_1\} \\ k_{a_i} &= \exp\{\beta_2\} \\ k_{e_i} &= \exp\{\beta_3\} \end{aligned}$$

In between these two extremes lie other models, such as a model where both groups have the same absorption and elimination rate, but different clearances. The question then becomes how to go about building a model in an organized manner.

To test hypotheses about nested nonlinear models, you can apply the idea of a “Sum of Squares Reduction Test.” A reduced model is nested within a full model if you can impose q constraints on the full model to obtain the reduced model. Then, if SSE_r and SSE_f denote the residual sum of squares in the reduced and the full model, respectively, the test statistic is

$$F_R = \frac{(SSE_r - SSE_f)/q}{SSE_f/(n - p)} = \frac{(SSE_r - SSE_f)/q}{MSE_f}$$

where n are the number of observations used and p are the number of parameters in the full model. The numerator of the F_R statistic is the average reduction in residual sum of squares per constraint. The mean squared error of the full model is used to scale this average because it is less likely to be a biased estimator of the residual variance than the variance estimate from a constrained (reduced) model. The F_R statistic is then compared against quantiles from an F distribution with q numerator and $n - p$ denominator degrees of freedom. Schabenberger and Pierce (2002) discuss the justification for this test and compare it to other tests in nonlinear models.

In the present application we might phrase the initial question akin to the overall F test for a factor in a linear model: Should any parameters be varied between the two groups? The corresponding null hypothesis is

$$H: \begin{cases} \beta_{11} = \beta_{12} \\ \beta_{21} = \beta_{22} \\ \beta_{31} = \beta_{32} \end{cases}$$

where the first subscript identifies the type of the parameter and the second subscript identifies the group. Note that this hypothesis implies

$$H: \begin{cases} Cl_1 = Cl_2 \\ k_{a1} = k_{a2} \\ k_{e1} = k_{e2} \end{cases}$$

If you fail to reject this hypothesis, there is no need to further examine individual parameter differences.

The reduced model—the model subject to the null hypothesis—is fit with the following PROC NLIN statements:

```
proc nlin data=theop;
  parms beta1=-3.22 beta2=0.47 beta3=-2.45;
  cl  = exp(beta1);
  ka  = exp(beta2);
  ke  = exp(beta3);
  mean = dose*ke*ka*(exp(-ke*time)-exp(-ka*time))/cl/(ka-ke);
  model conc = mean;
  ods output Anova=aovred(rename=(ss=ssred ms=msred df=dfred));
run;
```

The clearance, the rates, and the mean function are formed independently of the group membership. The analysis of variance table is saved to the data set aovred, and some of its variables are renamed. This is done so that the data set can be merged easily with the analysis of variance table for the full model (see following).

The converged model has a residual sum of square of $SSE_r = 286.4$ and a mean squared error of 3.0142 (Output 81.5.2). The table of parameter estimates gives the values for the estimated log clearance ($\hat{\beta}_1 = -3.2991$), the estimated log absorption rate ($\hat{\beta}_2 = 0.4769$), and the estimated log elimination rate ($\hat{\beta}_3 = -2.5555$).

Output 81.5.2 Fit Results for the Reduced Model

The NLIN Procedure
Dependent Variable conc
Method: Gauss-Newton

NOTE: Convergence criterion met.

Note: An intercept was not specified for this model.

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	3	3100.5	1033.5	342.87	<.0001
Error	95	286.4	3.0142		
Uncorrected Total	98	3386.8			

Output 81.5.2 *continued*

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits	
beta1	-3.2991	0.0956	-3.4888	-3.1094
beta2	0.4769	0.1640	0.1512	0.8025
beta3	-2.5555	0.1410	-2.8354	-2.2755

The full model, in which all three parameters are varied by group, can be fit with the following statements in the NLIN procedure:

```
proc nlin data=theop;
  parms beta1_1=-3.22 beta2_1=0.47 beta3_1=-2.45
        beta1_2=-3.22 beta2_2=0.47 beta3_2=-2.45;
  if (group=1) then do;
    cl  = exp(beta1_1);
    ka  = exp(beta2_1);
    ke  = exp(beta3_1);
  end; else do;
    cl  = exp(beta1_2);
    ka  = exp(beta2_2);
    ke  = exp(beta3_2);
  end;
  mean = dose*ke*ka*(exp(-ke*time)-exp(-ka*time))/cl/(ka-ke);
  model conc = mean;
  ods output Anova=aovfull;
run;
```

Separate parameters for the groups are now specified in the **PARMS** statement, and the value of the model variables *cl*, *ka*, and *ke* is assigned conditional on the group membership of an observation. Notice that the same expression as in the previous run can be used to model the mean function.

The results from this PROC NLIN run are shown in [Output 81.5.3](#). The residual sum of squares in the full model is only $SSE_f = 138.9$, compared to $SSE_r = 286.4$ in the reduced model ([Output 81.5.3](#)).

Output 81.5.3 Fit Results for the Full Model

The NLIN Procedure
Dependent Variable conc
Method: Gauss-Newton

NOTE: Convergence criterion met.

Note: An intercept was not specified for this model.

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	6	3247.9	541.3	358.56	<.0001
Error	92	138.9	1.5097		
Uncorrected Total	98	3386.8			

Output 81.5.3 *continued*

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits	
beta1_1	-3.5671	0.0864	-3.7387	-3.3956
beta2_1	0.4421	0.1349	0.1742	0.7101
beta3_1	-2.6230	0.1265	-2.8742	-2.3718
beta1_2	-3.0111	0.1061	-3.2219	-2.8003
beta2_2	0.3977	0.1987	0.00305	0.7924
beta3_2	-2.4442	0.1618	-2.7655	-2.1229

Whether this reduction in sum of squares is sufficient to declare that the full model provides a significantly better fit than the reduced model depends on the number of constraints imposed on the full model and on the variability in the data. In other words, before drawing any conclusions, you have to take into account how many parameters have been dropped from the model and how much variation around the regression trends the data exhibit. The F_R statistic sets these quantities into relation. The following macro merges the analysis of variance tables from the full and reduced model, and computes F_R and its p -value:

```
%macro SSReductionTest;
  data aov; merge aovred aovfull;
  if (Source='Error') then do;
    Fstat = ((SSred-SS) / (dfred-df)) / ms;
    pvalue = 1-ProbF(Fstat,dfred-df,df);
    output;
  end;
run;
proc print data=aov label noobs;
  label Fstat = 'F Value'
        pValue = 'Prob > F';
  format pvalue pvalue8.;
  var Fstat pValue;
run;
%mend;
%SSReductionTest;
```

Output 81.5.4 F Statistic and P-value for Hypothesis of Equal Trends

F Value	Prob > F
32.5589	<.000001

There is clear evidence that the model with separate trends fits these data significantly better ([Output 81.5.4](#)). To decide whether all parameters should be varied between the groups or only one or two of them, we first refit the model in a slightly different parameterization:

```
proc nlin data=theop;
  parms beta1_1=-3.22 beta2_1=0.47 beta3_1=-2.45
        beta1_diff=0 beta2_diff=0 beta3_diff=0;
  if (group=1) then do;
    cl  = exp(beta1_1);
    ka  = exp(beta2_1);
    ke  = exp(beta3_1);
  end; else do;
    cl  = exp(beta1_1 + beta1_diff);
    ka  = exp(beta2_1 + beta2_diff);
    ke  = exp(beta3_1 + beta3_diff);
  end;
  mean = dose*ke*ka*(exp(-ke*time)-exp(-ka*time))/cl/(ka-ke);
  model conc = mean;
run;
```

In the preceding statements, the parameters in the second group were expressed using offsets from parameters in the first group. For example, the parameter `beta1_diff` measures the change in log clearance between group 2 and group 1.

This simple reparameterization does not affect the model fit. The analysis of variance tables in [Output 81.5.5](#) and [Output 81.5.3](#) are identical. It does, however, affect the interpretation of the estimated quantities. Since the parameter `beta1_diff` measures the change in the log clearance rates between the groups, you can use the approximate 95% confidence limits in [Output 81.5.5](#) to assess whether that quantity in the pharmacokinetic equation varies between groups. Only the confidence interval for the difference in the log clearances excludes 0. The intervals for `beta2_diff` and `beta3_diff` include 0.

Output 81.5.5 Fit Results for the Full Model in Difference Parameterization

The NLIN Procedure
Dependent Variable conc
Method: Gauss-Newton

NOTE: Convergence criterion met.

Note: An intercept was not specified for this model.

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	6	3247.9	541.3	358.56	<.0001
Error	92	138.9	1.5097		
Uncorrected Total	98	3386.8			

Output 81.5.5 *continued*

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits	
beta1_1	-3.5671	0.0864	-3.7387	-3.3956
beta2_1	0.4421	0.1349	0.1742	0.7101
beta3_1	-2.6230	0.1265	-2.8742	-2.3718
beta1_diff	0.5560	0.1368	0.2842	0.8278
beta2_diff	-0.0444	0.2402	-0.5214	0.4326
beta3_diff	0.1788	0.2054	-0.2291	0.5866

This suggests as the final model one where the absorption and elimination rates are the same for both groups and only the clearances are varied. The following statements fit this model and perform the sum of squares reduction test:

```
proc nlin data=theop;
  parms beta1_1=-3.22 beta2_1=0.47 beta3_1=-2.45
        beta1_diff=0;
  ka = exp(beta2_1);
  ke = exp(beta3_1);
  if (group=1) then do;
    cl = exp(beta1_1);
  end; else do;
    cl = exp(beta1_1 + beta1_diff);
  end;
  mean = dose*ke*ka*(exp(-ke*time)-exp(-ka*time))/cl/(ka-ke);
  model conc = mean;
  ods output Anova=aovred(rename=(ss=ssred ms=msred df=dfred));
  output out=predvals predicted=p;
run;

%SSReductionTest;
```

The results for this model with common absorption and elimination rates are shown in [Output 81.5.6](#). The sum-of-squares reduction test comparing this model against the full model with six parameters shows—as expected—that the full model does not fit the data significantly better ($p = 0.6193$, [Output 81.5.7](#)).

Output 81.5.6 Fit Results for Model with Common Rates

The NLIN Procedure
Dependent Variable conc
Method: Gauss-Newton

NOTE: Convergence criterion met.

Note: An intercept was not specified for this model.

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	4	3246.5	811.6	543.60	<.0001
Error	94	140.3	1.4930		
Uncorrected Total	98	3386.8			

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits	
beta1_1	-3.5218	0.0681	-3.6570	-3.3867
beta2_1	0.4226	0.1107	0.2028	0.6424
beta3_1	-2.5571	0.0988	-2.7532	-2.3610
beta1_diff	0.4346	0.0454	0.3444	0.5248

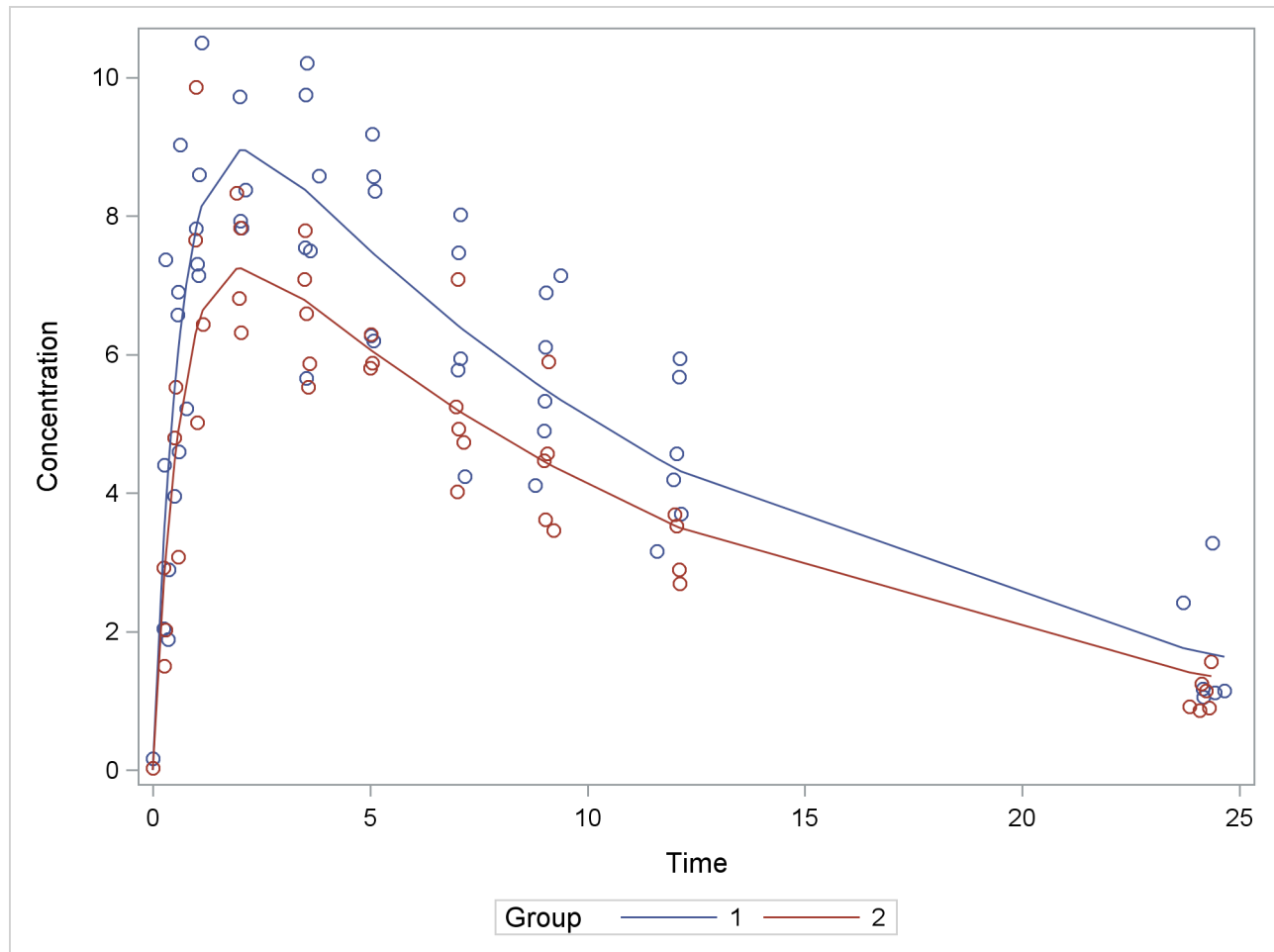
Output 81.5.7 F Statistic and P-value for Hypothesis of Common Rates

F Value	Prob > F
0.48151	0.619398

A plot of the observed and predicted values for this final model is obtained with the following statements:

```
proc sgplot data=predvals;
  scatter x=time y=conc / group=group;
  series x=time y=p / group=group name='fit';
  keylegend 'fit' / across=2 title='Group';
  yaxis label='Concentration';
  xaxis label='Time';
run;
```

The plot is shown in [Output 81.5.8](#).

Output 81.5.8 Observed and Fitted Values for Theophylline Data

The sum-of-squares reduction test is not the only possibility of performing linear-model style hypothesis testing in nonlinear models. You can also perform Wald-type testing of linear hypotheses about the parameter estimates. See [Example 45.17](#) in Chapter 45, “[The GLIMMIX Procedure](#),” for an application of this example that uses the NLIN and GLIMMIX procedures to compare the parameters across groups and adjusts p -values for multiplicity.

Example 81.6: ODS Graphics and Diagnostics

The model in this example, taken from St. Laurent and Cook (1993), shows an unusual behavior in that the intrinsic curvature is substantially larger than the parameter-effects curvature. This example demonstrates how the diagnostics features of PROC NLIN can be used to perform postconvergence diagnostics.

The model takes the form

$$E[Y|x_1, x_2] = \alpha x_1 + \exp\{\gamma x_2\}$$

The following DATA step creates a small data set to be used in this example:

```
data contrived;
  input x1 x2 y;
  datalines;
-4.0  -2.5 -10.0
-3.0  -2.0  -5.0
-2.0  -1.5  -2.0
-1.0  -1.0  -1.0
 0.0   0.0   1.5
 1.0   1.0   4.0
 2.0   1.5   5.0
 3.0   2.0   6.0
 4.0   2.5   7.0
-3.5  -2.2  -7.1
-3.5  -1.7  -5.1
 3.5   0.7   6.1
 2.5   1.2   7.5
;
```

The model is fit with the following statements in the NLIN procedure:

```
ods graphics on;
proc nlin data=contrived bias hougard
  NLINMEASURES plots(stats=all)=(diagnostics);
  parms alpha=2.0
        gamma=0.0;
  model y = alpha*x1 + exp(gamma*x2);
run;
ods graphics off;
```

Output 81.6.1 Bias, Skewness, and Global Nonlinearity Measures**The NLIN Procedure**

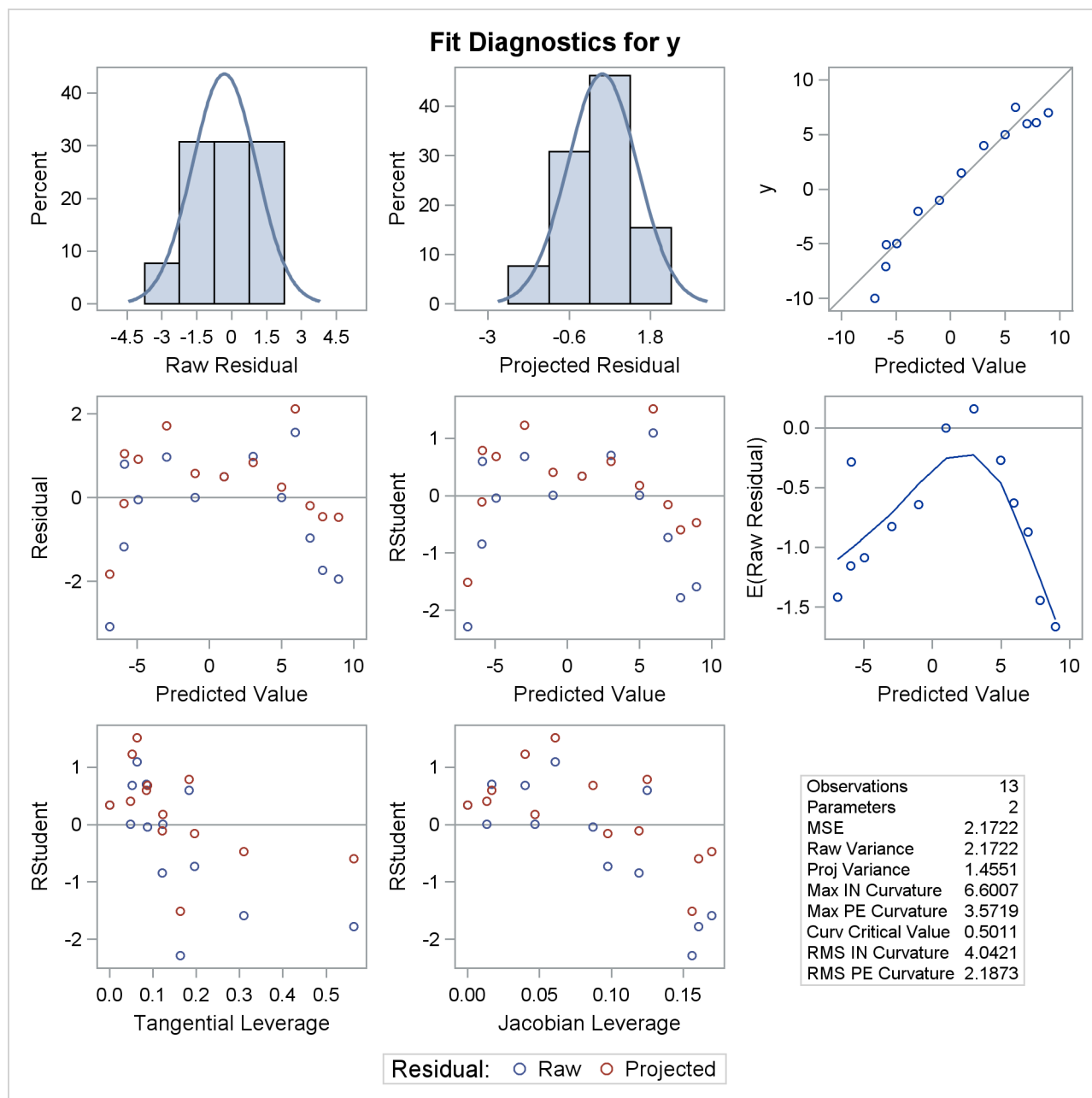
Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits		Skewness	Bias	Percent Bias
alpha	1.9378	0.4704	0.9024	2.9733	6.6491	0.5763	29.7
gamma	0.0718	0.7923	-1.6720	1.8156	-7.5596	-0.9982	-1390

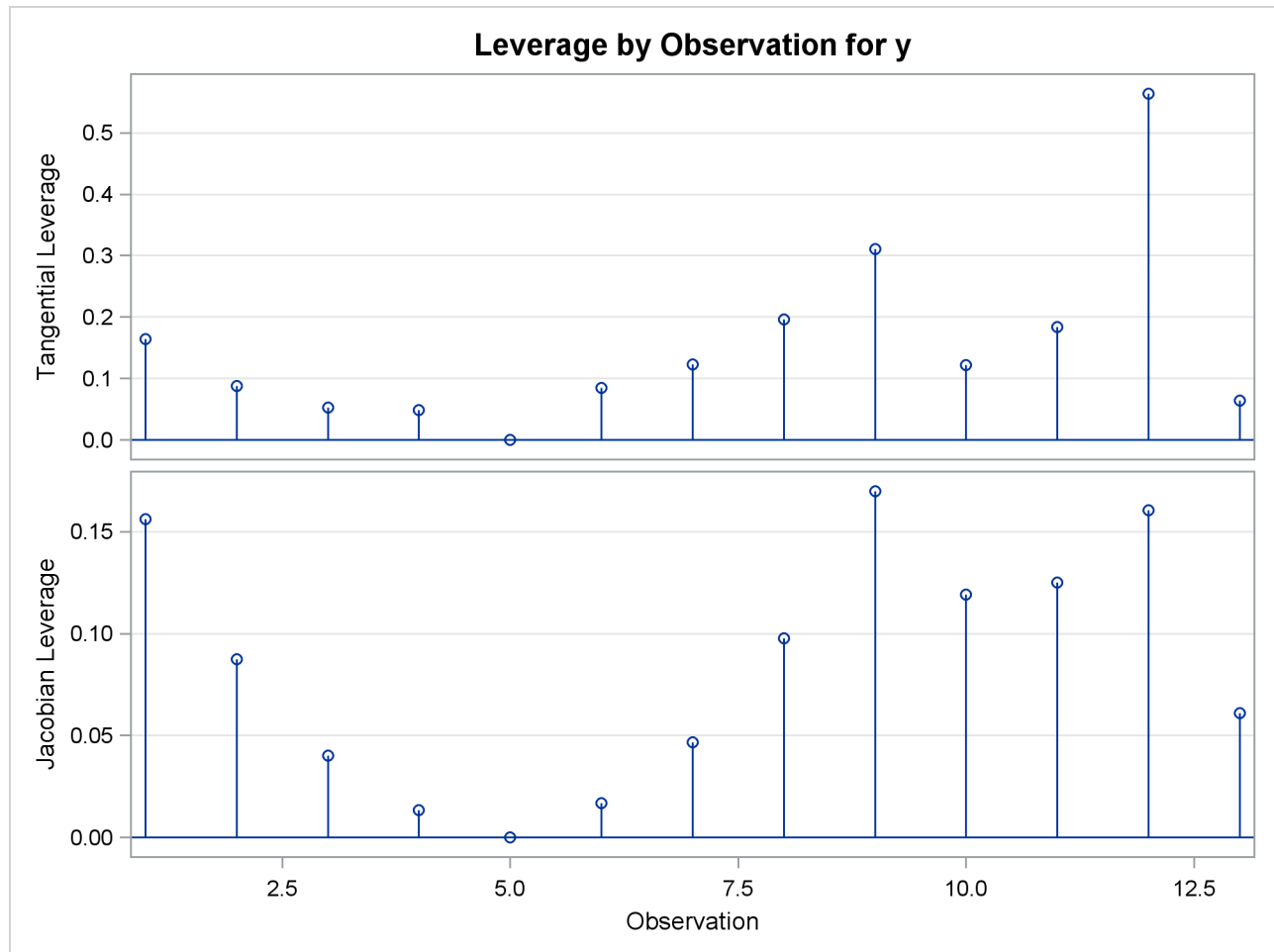
Global Nonlinearity Measures	
Max Intrinsic Curvature	6.6007
RMS Intrinsic Curvature	4.0421
Max Parameter-Effects Curvature	3.5719
RMS Parameter-Effects Curvature	2.1873
Curvature Critical Value	0.5011
Raw Residual Variance	2.1722
Projected Residual Variance	1.4551

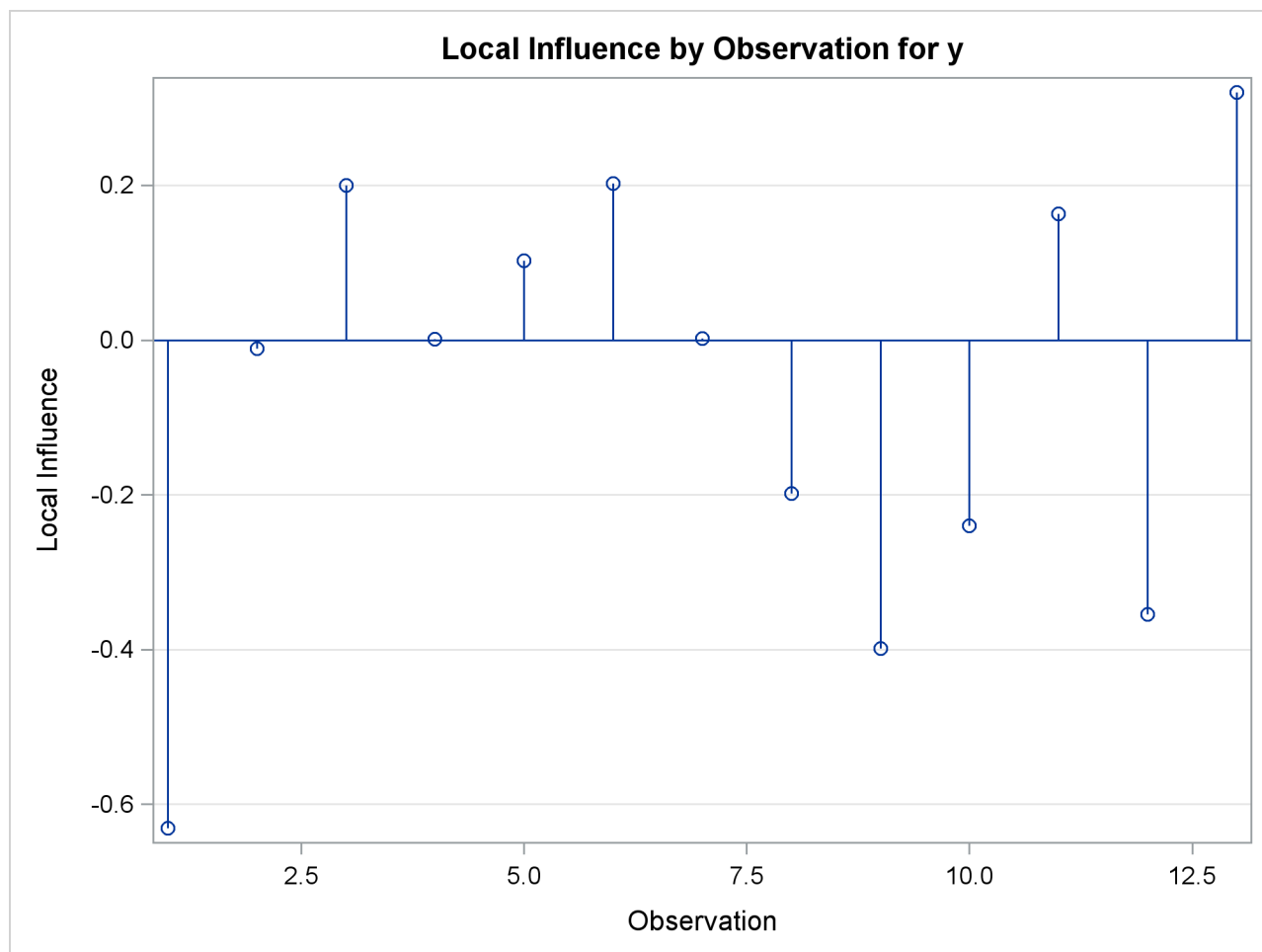
The bias, skewness, and both the maximum and RMS intrinsic curvatures, compared to the critical curvature value, show that the model is highly nonlinear ([Output 81.6.1](#)). As such, performing diagnostics with the raw residuals can be problematic because they might have undesirable statistical properties: a nonzero mean and a negative semidefinite (instead of zero) covariance with the predicted values and different variances. In addition, the use of tangential leverage is questionable in this case.

The partial results from this NLIN run are shown in [Output 81.6.2](#), [Output 81.6.3](#), and [Output 81.6.4](#). The diagnostics plots corroborate the previously mentioned expectations: highly correlated raw residuals (with the predicted values), significant differences between tangential and Jacobian leverages and projected residuals which overcome some of the shortcomings of the raw residuals. Finally, considering the large intrinsic curvature, reparameterization might not make the model close-to-linear, perhaps necessitating the construction of another model.

Output 81.6.2 Diagnostics Panel



Output 81.6.3 Leverage Plots

Output 81.6.4 Local Influence Plot

Example 81.7: Parameter Profiling and Bootstrapping

This example, which uses a model from Clarke (1987), shows why it is difficult to intuitively reason about the characteristics of the parameters that occur in a nonlinear model. As a consequence, the example demonstrates the indispensability of parameter profiling and confidence curves to identify the nonlinear characteristics of model parameters. The parameter profiling is also augmented with a plot that shows the influence of each observation on the parameter estimates. Furthermore, the **BOOTSTRAP** statement in PROC NLIN is invoked in order to study the sampling distribution of parameter estimates and make more accurate statistical inferences.

The model takes the form

$$E[Y|x] = \theta_3 + \theta_2 \exp\{\theta_1 x\}$$

The data set in this example is from Clarke (1987). The following DATA step creates this data set:

```
data clarke1987a;
  input  x y;
  datalines;
1      3.183
2      3.059
3      2.871
4      2.622
5      2.541
6      2.184
7      2.110
8      2.075
9      2.018
10     1.903
11     1.770
12     1.762
13     1.550
;
```

The model is fit by using the following statements in the NLIN procedure:

```
ods graphics on;
proc nlin data=clarke1987a plots(stats=none)=diagnostics;
  parms theta1=-0.15
         theta2=2.0
         theta3=0.80;
  profile theta1 theta3 / range = -6 to 2 by 0.2 all;
  bootstrap / nsamples = 2000 seed=123 bootplots bootci bootcov;
  model y = theta3 + theta2*exp(theta1*x);
run;
ods graphics off;
```

Output 81.7.1 Parameter Estimates and Bootstrap Estimate of the Covariance Matrix**The NLIN Procedure**

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits		Bootstrap Std Dev	Bootstrap Bias-Corrected 95% Confidence Limits	
theta1	-0.1031	0.0255	-0.1599	-0.0462	0.0253	-0.1505	-0.0531
theta2	2.5190	0.2658	1.9268	3.1112	0.4450	2.1716	3.5325
theta3	0.9631	0.3216	0.2466	1.6797	0.4907	-0.2565	1.3697

Bootstrap Covariance Matrix Estimate			
	theta1	theta2	theta3
theta1	0.00063801	0.00834138	-.01006296
theta2	0.00834138	0.19802119	-.21570467
theta3	-.01006296	-.21570467	0.24082392

By default, the **BOOTCI** option in the **BOOTSTRAP** statement adds a column that contains the bias-corrected bootstrap confidence limits to the “Parameter Estimates” table (Output 81.7.1). The **BOOTCOV** option produces the “Bootstrap Covariance Matrix” table (Output 81.7.1). The discrepancy between the Wald-based confidence limits and the bias-corrected bootstrap confidence limits is much greater for θ_3 than for θ_1 . The profile t plot, confidence curve, and histograms of the bootstrap estimates show why this is the case.

The profile t plot for parameter θ_3 in Output 81.7.3 shows a definite deviation from the linear reference line that has a slope of 1 and passes through the origin. Hence, Wald-based inference for θ_3 is not appropriate. In contrast, the profile t plot for parameter θ_1 in Output 81.7.2 shows that Wald-based inference for θ_1 might be sufficient.

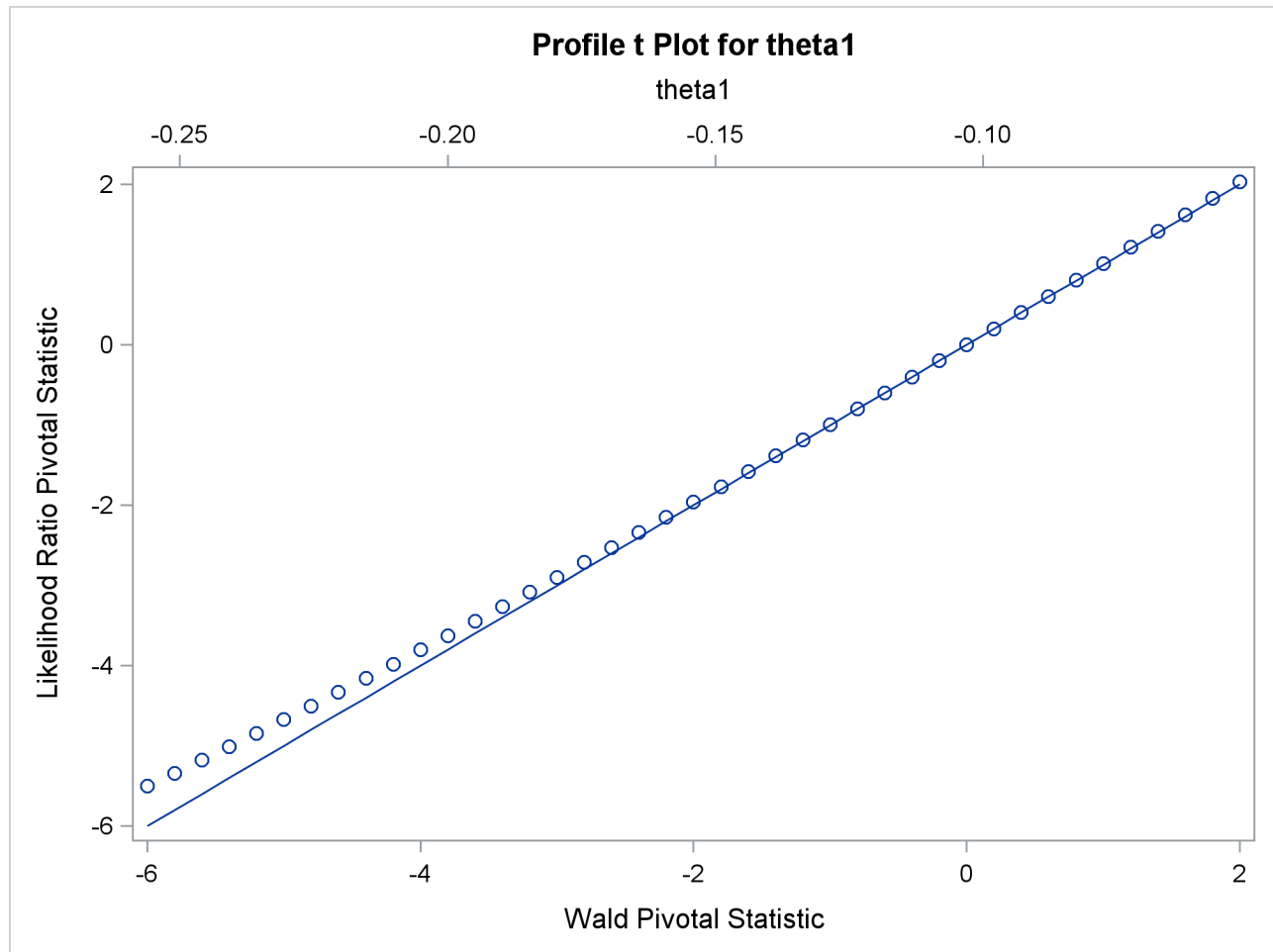
Output 81.7.4 and Output 81.7.5 show the confidence curves for θ_1 and θ_3 , respectively. For θ_3 , you can see a significant difference between the Wald-based confidence interval and the corresponding likelihood-based interval. In such cases, the likelihood-based intervals are preferred because their coverage rate is much closer to the nominal values than the coverage rate of the Wald-based intervals (Donaldson and Schnabel 1987; Cook and Weisberg 1990).

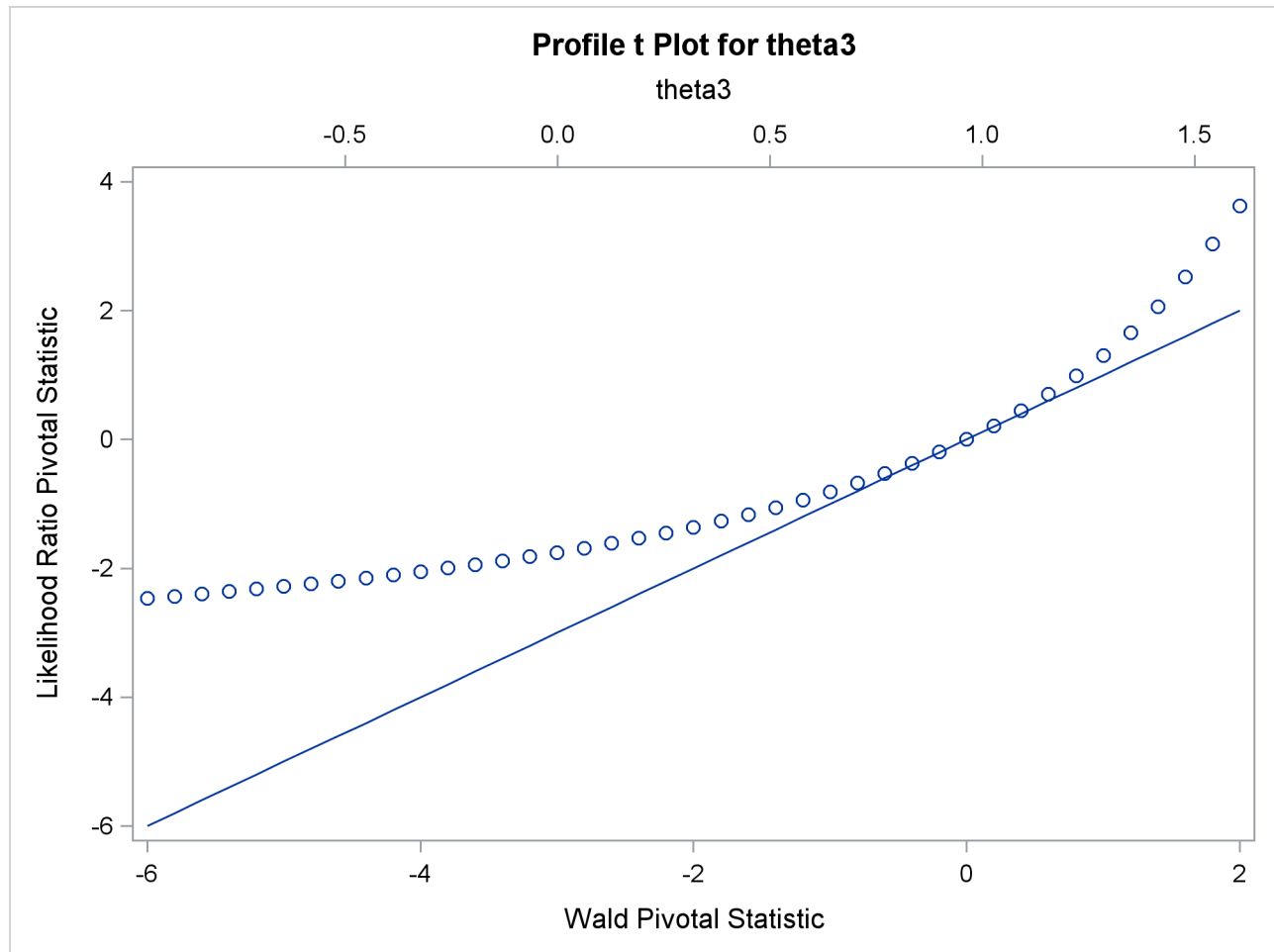
Output 81.7.6 depicts the influence of each observation on the value of θ_3 . Observations 6 and 13 have the most influence on the value of this parameter. The plot is generated using the leave-one-out method and should be contrasted with the local influence plot in Output 81.7.7, which is based on assessing the influence of an additive perturbation of the response variable.

Output 81.7.8 and Output 81.7.9 are histograms that show the distribution of the bootstrap parameter estimates for θ_1 and θ_3 , respectively. These histograms complement the information that is obtained about θ_1 and θ_3 from the profile t plots. Specifically, they show that the bootstrap parameter estimate of θ_1 has a distribution close to normal, whereas that of θ_3 has a distribution that deviates significantly from normal. Again, this leads to the conclusion that inferences based on linear approximations, such as Wald-based confidence intervals, work better for θ_1 than for θ_3 .

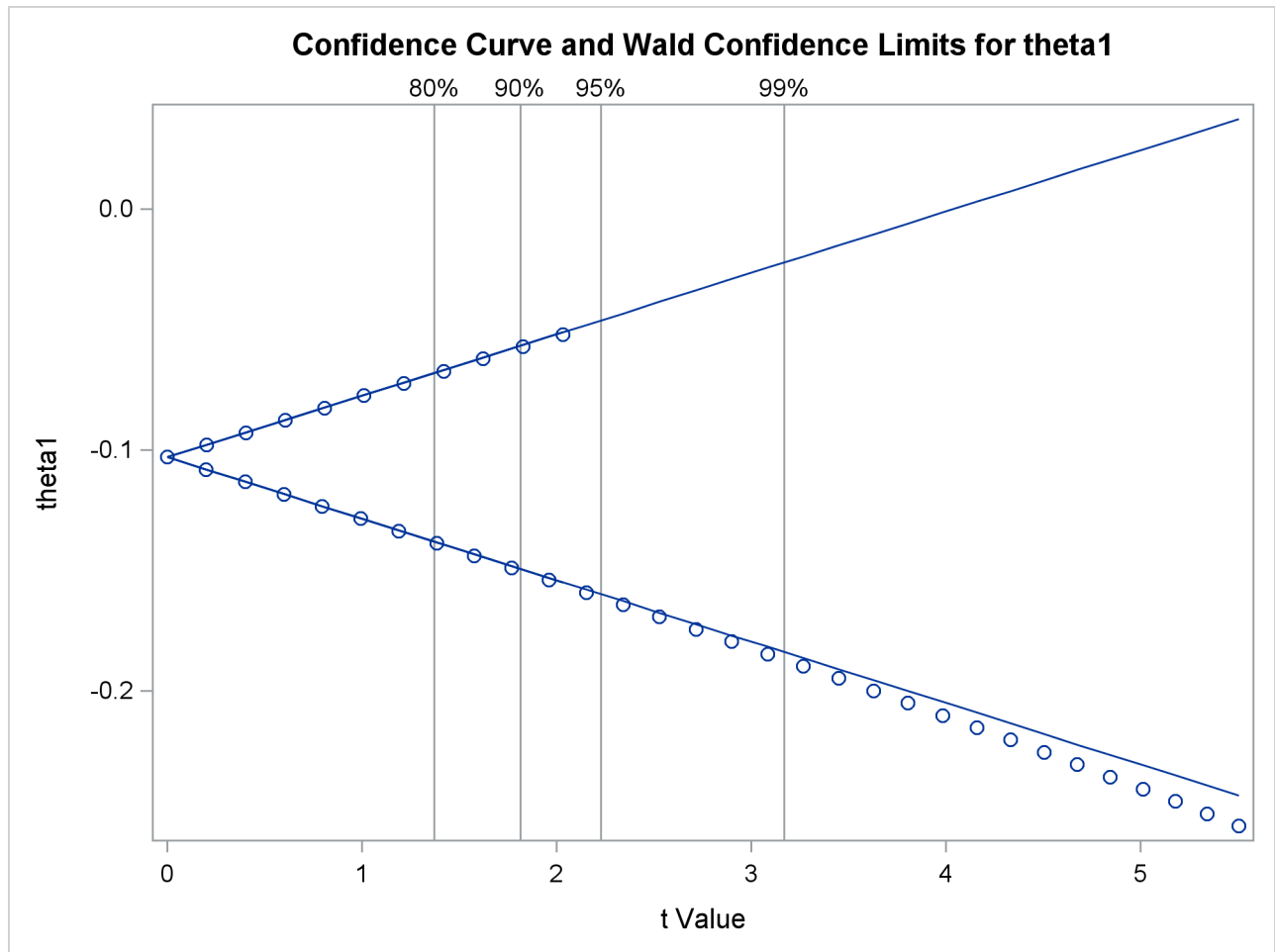
Finally, this example shows that the adequacy of a linear approximation with regard to a certain parameter cannot be inferred directly from the model. If it could, then θ_3 , which enters the model linearly, would have a completely linear behavior, whereas θ_1 would have a highly nonlinear behavior. However, the diagnostics that are based on the profile t plot and confidence curves, and the histograms of the bootstrap parameter estimates, show that the opposite holds. For a detailed discussion about this issue, see Cook and Weisberg (1990).

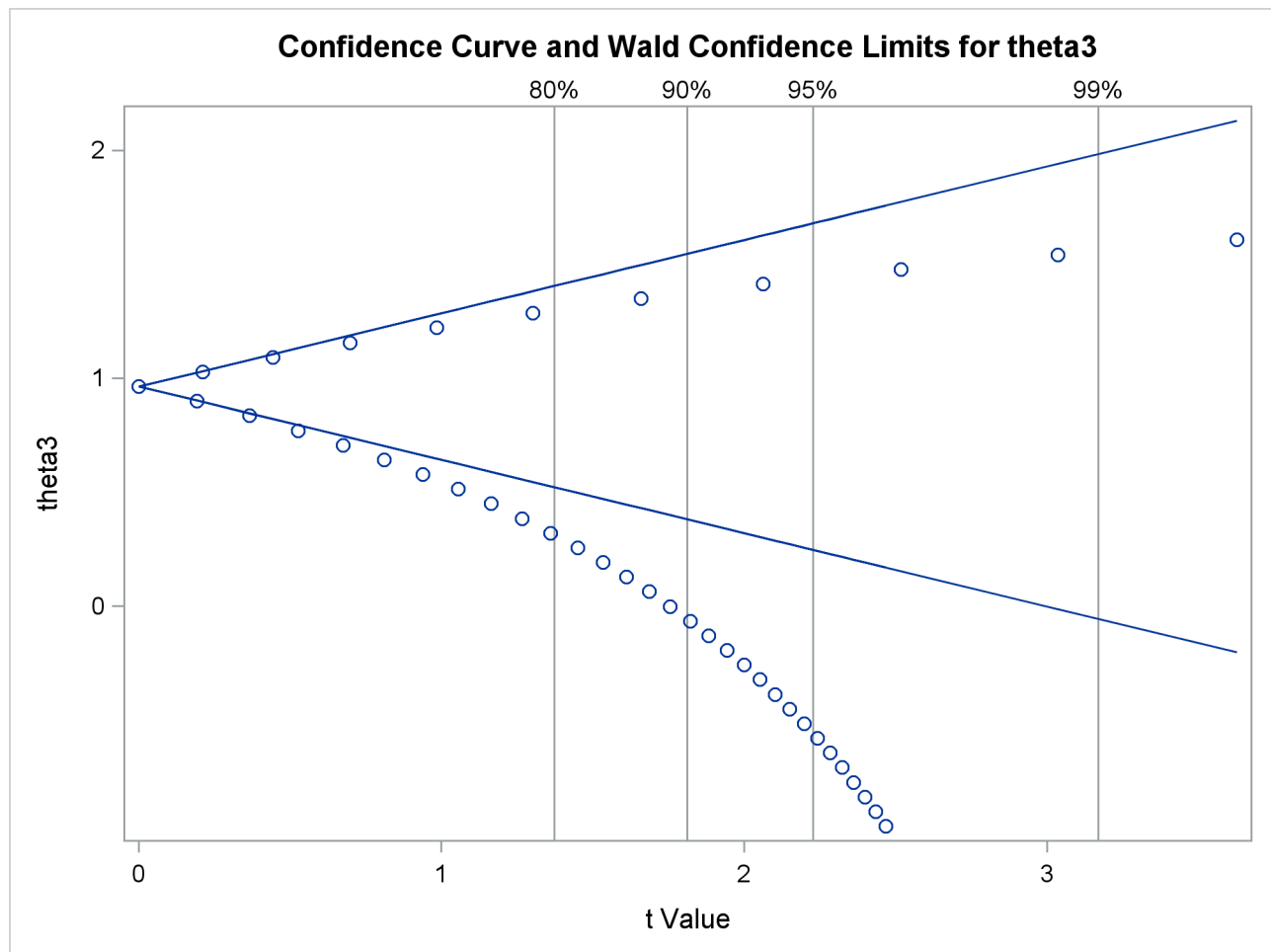
Output 81.7.2 Profile t Plot

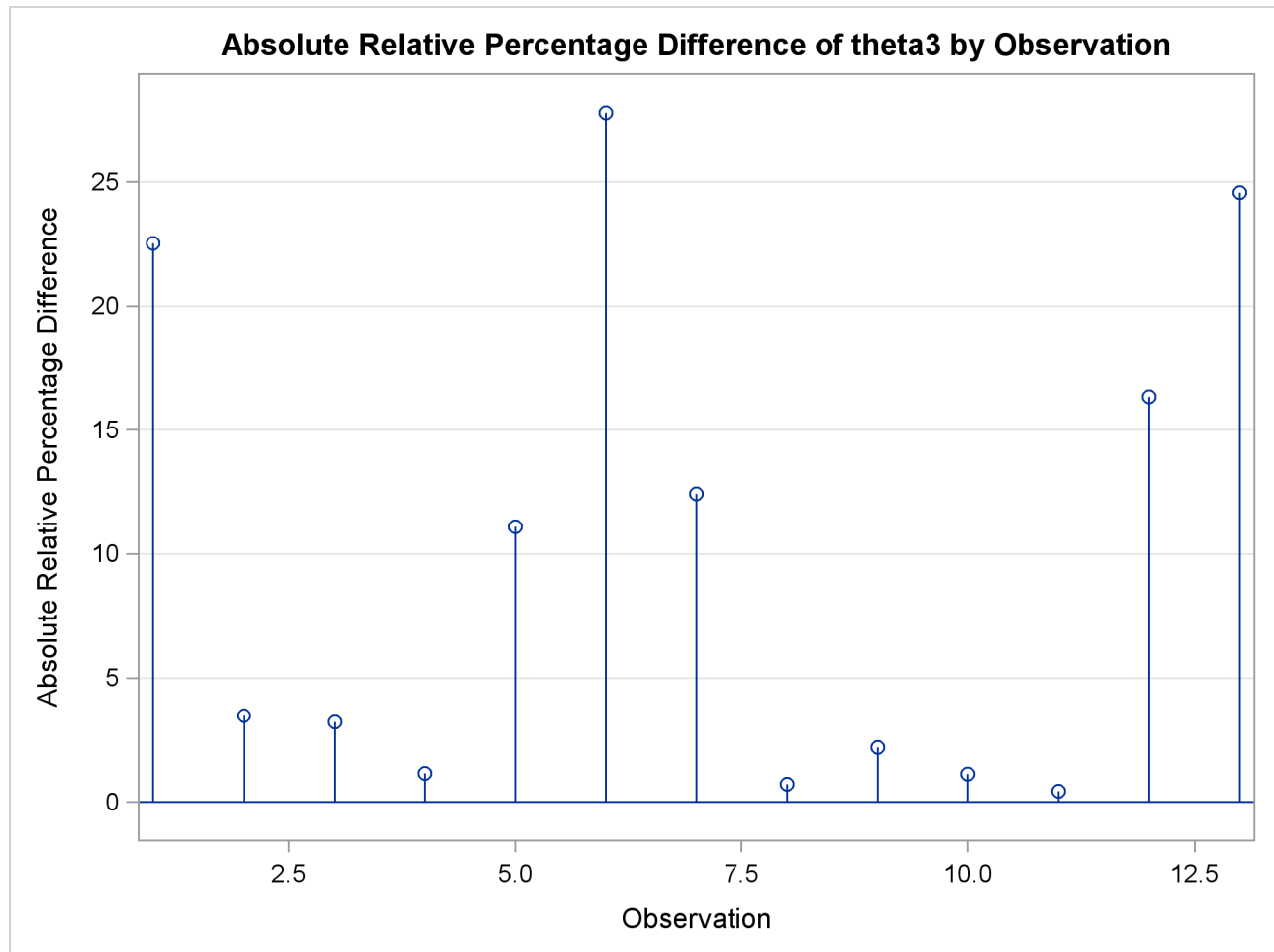


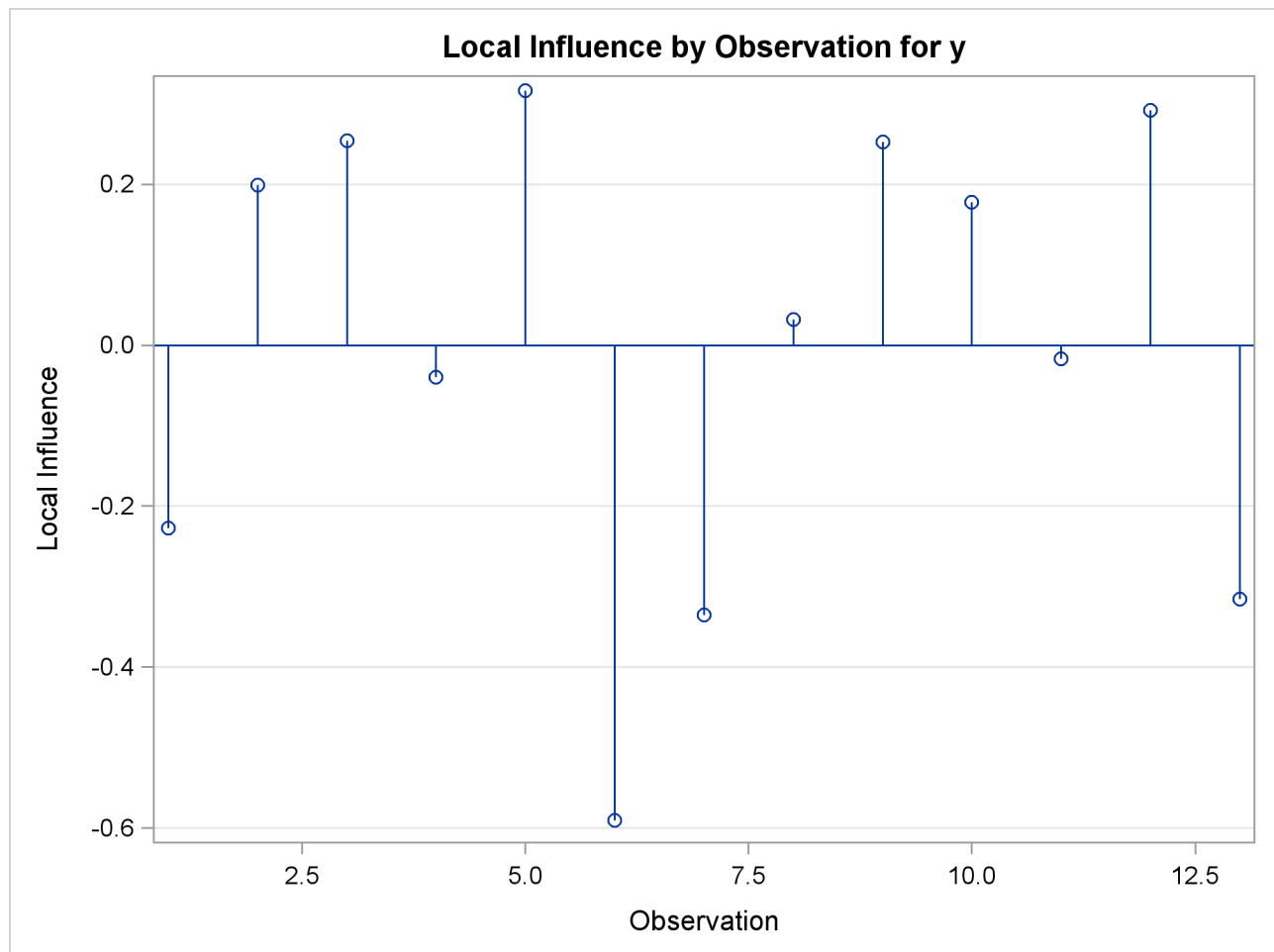
Output 81.7.3 Profile t Plot

Output 81.7.4 Confidence Curve

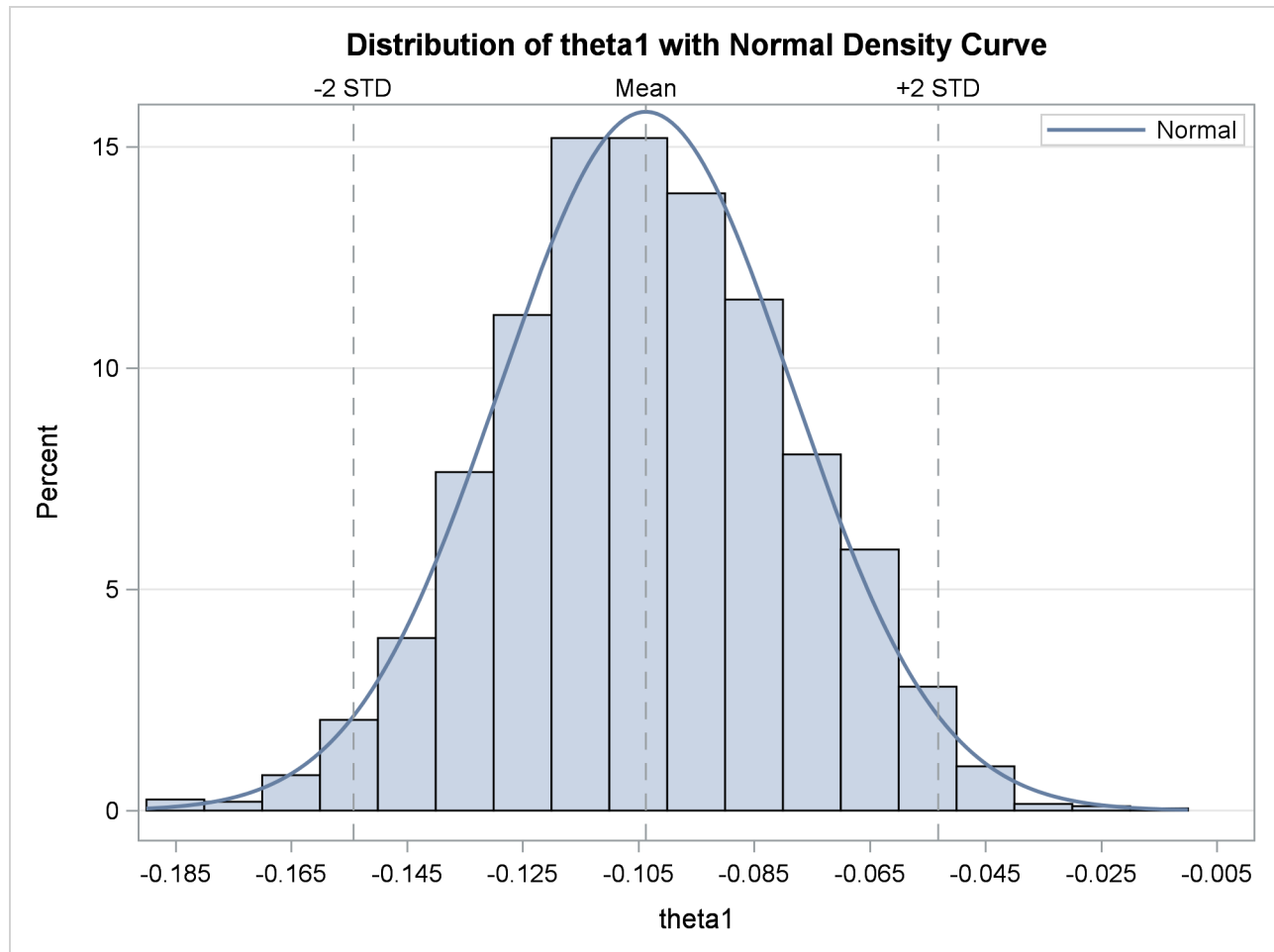


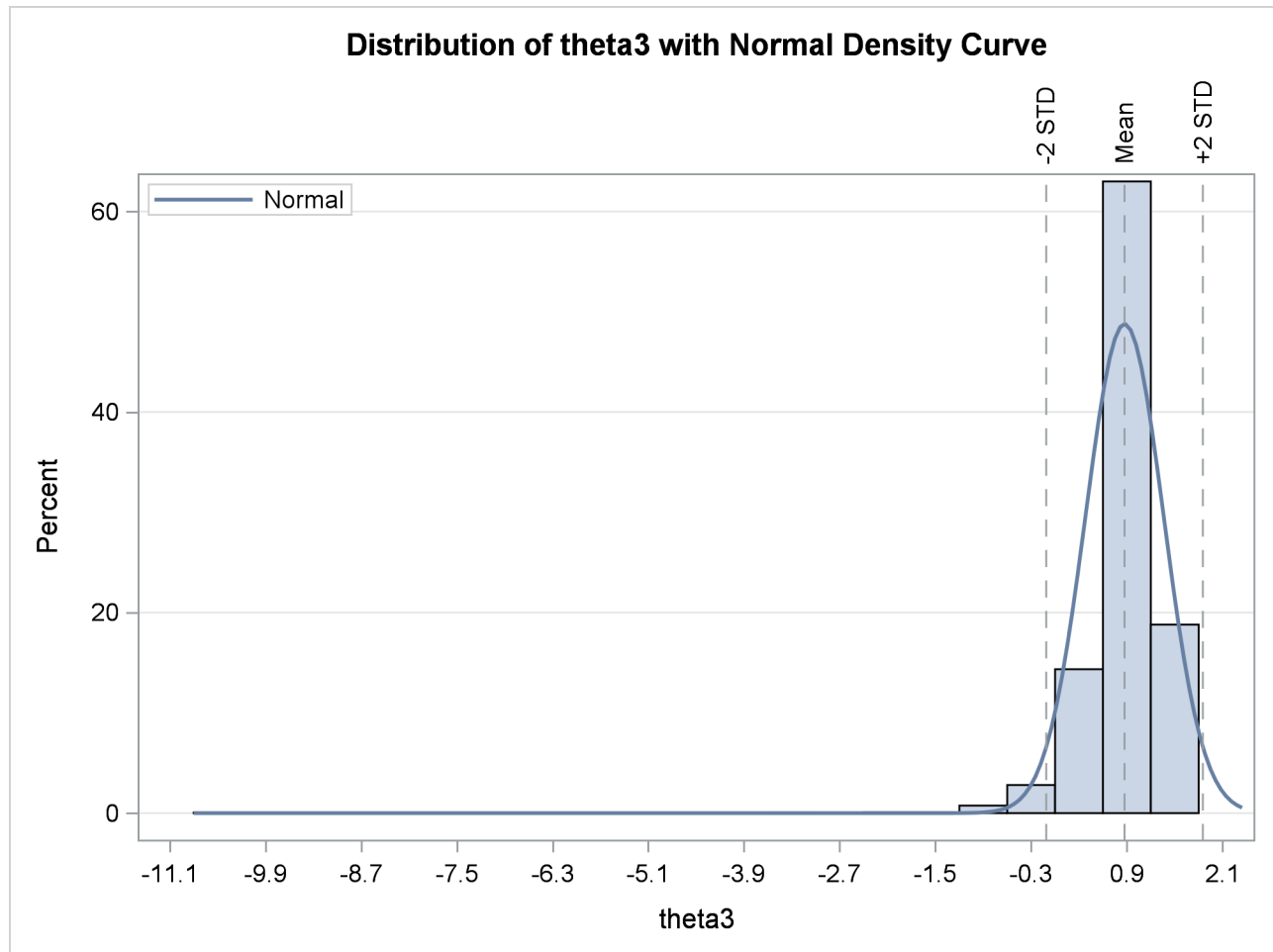
Output 81.7.5 Confidence Curve

Output 81.7.6 Observation Influence on Parameter Estimate

Output 81.7.7 Local Influence Plot

Output 81.7.8 Histogram of Bootstrap Parameter Estimates



Output 81.7.9 Histogram of Bootstrap Parameter Estimates

References

- Bard, Y. (1970). "Comparison of Gradient Methods for the Solution of the Nonlinear Parameter Estimation Problem." *SIAM Journal on Numerical Analysis* 7:157–186.
- Bard, Y. (1974). *Nonlinear Parameter Estimation*. New York: Academic Press.
- Bates, D. M., and Watts, D. G. (1980). "Relative Curvature Measures of Nonlinearity (with Discussion)." *Journal of the Royal Statistical Society, Series B* 42:1–25.
- Bates, D. M., and Watts, D. G. (1981). "A Relative Offset Orthogonality Convergence Criterion for Nonlinear Least Squares." *Technometrics* 23:179–183.
- Bates, D. M., and Watts, D. G. (1988). *Nonlinear Regression Analysis and Its Applications*. New York: John Wiley & Sons.
- Beaton, A. E., and Tukey, J. W. (1974). "The Fitting of Power Series, Meaning Polynomials, Illustrated on Band-Spectroscopic Data." *Technometrics* 16:147–185.

- Box, M. J. (1971). "Bias in Nonlinear Estimation (with Discussion)." *Journal of the Royal Statistical Society, Series B* 33:171–201.
- Charnes, A., Frome, E. L., and Yu, P. L. (1976). "The Equivalence of Generalized Least Squares and Maximum Likelihood Estimation in the Exponential Family." *Journal of the American Statistical Association* 71:169–171.
- Clarke, G. P. Y. (1987). "Approximate Confidence Limits for a Parameter Function in Nonlinear Regression." *Journal of the American Statistical Association* 82:221–230.
- Cook, R. D., and Tsai, C.-L. (1985). "Residuals in Nonlinear Regression." *Biometrika* 72:23–29.
- Cook, R. D., and Weisberg, S. (1990). "Confidence Curves in Nonlinear Regression." *Journal of the American Statistical Association* 85:544–551.
- Cox, D. R. (1970). *Analysis of Binary Data*. London: Methuen.
- DiCiccio, T. J., and Efron, B. (1996). "Bootstrap Confidence Intervals." *Statistical Science* 11:189–212.
- Donaldson, J. R., and Schnabel, R. B. (1987). "Computational Experience with Confidence Regions and Confidence Intervals for Nonlinear Least Squares." *Technometrics* 29:67–82.
- Finney, D. J. (1971). *Probit Analysis*. 3rd ed. Cambridge: Cambridge University Press.
- Gallant, A. R. (1975). "Nonlinear Regression." *American Statistician* 29:73–81.
- Gill, P. E., Murray, W., and Wright, M. H. (1981). *Practical Optimization*. New York: Academic Press.
- Goodnight, J. H. (1979). "A Tutorial on the Sweep Operator." *American Statistician* 33:149–158.
- Hartley, H. O. (1961). "The Modified Gauss-Newton Method for the Fitting of Non-linear Regression Functions by Least Squares." *Technometrics* 3:269–280.
- Holland, P. W., and Welsch, R. E. (1977). "Robust Regression Using Iteratively Reweighted Least-Squares." *Communications in Statistics—Theory and Methods* 6:813–827.
- Hougaard, P. (1982). "Parameterizations of Nonlinear Models." *Journal of the Royal Statistical Society, Series B* 44:244–252.
- Hougaard, P. (1985). "The Appropriateness of the Asymptotic Distribution in a Nonlinear Regression Model in Relation to Curvature." *Journal of the Royal Statistical Society, Series B* 47:103–114.
- Huber, P. J. (1964). "Robust Estimation of a Location Parameter." *Annals of Mathematical Statistics* 35:73–101.
- Huber, P. J. (1973). "Robust Regression: Asymptotics, Conjectures, and Monte Carlo." *Annals of Statistics* 1:799–821.
- Jennrich, R. I. (1969). "Asymptotic Properties of Nonlinear Least Squares Estimators." *Annals of Mathematical Statistics* 40:633–643.
- Jennrich, R. I., and Moore, R. H. (1975). "Maximum Likelihood Estimation by Means of Nonlinear Least Squares." *American Statistical Association, 1975 Proceedings of the Statistical Computing Section* 57–65.

- Jennrich, R. I., and Sampson, P. F. (1968). "Application of Stepwise Regression to Nonlinear Estimation." *Technometrics* 10:63–72.
- Judge, G. G., Griffiths, W. E., Hill, R. C., and Lee, T.-C. (1980). *The Theory and Practice of Econometrics*. New York: John Wiley & Sons.
- Kennedy, W. J., Jr., and Gentle, J. E. (1980). *Statistical Computing*. New York: Marcel Dekker.
- Lee, E. T. (1974). "A Computer Program for Linear Logistic Regression Analysis." *Computer Programs in Biomedicine* 4:80–92.
- MacKinnon, J. G. (2002). "Bootstrap Inference in Econometrics." *Canadian Journal of Economics* 35:615–645.
- Marquardt, D. W. (1963). "An Algorithm for Least-Squares Estimation of Nonlinear Parameters." *Journal of the Society for Industrial and Applied Mathematics* 11:431–441.
- McCullagh, P., and Nelder, J. A. (1989). *Generalized Linear Models*. 2nd ed. London: Chapman & Hall.
- Nelder, J. A., and Wedderburn, R. W. M. (1972). "Generalized Linear Models." *Journal of the Royal Statistical Society, Series A* 135:370–384.
- Pinheiro, J. C., and Bates, D. M. (1995). "Approximations to the Log-Likelihood Function in the Nonlinear Mixed-Effects Model." *Journal of Computational and Graphical Statistics* 4:12–35.
- Pringle, R. M., and Rayner, A. A. (1971). *Generalized Inverse Matrices with Applications to Statistics*. New York: Hafner Publishing.
- Ratkowsky, D. (1983). *Nonlinear Regression Modeling*. New York: Marcel Dekker.
- Ratkowsky, D. (1990). *Handbook of Nonlinear Regression Models*. New York: Marcel Dekker.
- Schabenberger, O., and Pierce, F. J. (2002). *Contemporary Statistical Models for the Plant and Soil Sciences*. Boca Raton, FL: CRC Press.
- Schabenberger, O., Tharp, B. E., Kells, J. J., and Penner, D. (1999). "Statistical Tests for Hormesis and Effective Dosages in Herbicide Dose Response." *Agronomy Journal* 91:713–721.
- Seber, G. A. F., and Wild, C. J. (1989). *Nonlinear Regression*. New York: John Wiley & Sons.
- St. Laurent, R. T., and Cook, R. D. (1992). "Leverages and Superleverages in Nonlinear Regression." *Journal of the American Statistical Association* 87:985–990.
- St. Laurent, R. T., and Cook, R. D. (1993). "Leverages, Local Influence, and Curvature in Nonlinear Regression." *Biometrika* 80:99–106.
- Wu, C. F. J. (1986). "Jackknife, Bootstrap and Other Resampling Methods in Regression Analysis." *Annals of Statistics* 80:1261–1295.

Subject Index

- alpha level
 - NLIN procedure, 6439
- bias
 - NLIN procedure, 6421
- bias-corrected confidence interval
 - NLIN procedure, 6430
- comparing trends
 - NLIN procedure, 6489
- confidence curves
 - NLIN procedure, 6442
- confidence interval
 - NLIN procedure, 6430
- confidence intervals
 - model confidence interval (NLIN), 6463
 - parameter confidence interval (NLIN), 6463
- derivatives
 - NLIN procedure, 6439
- diagnostics for model with a high intrinsic curvature
 - NLIN procedure, 6500
- examples, NLIN
 - array, constant, 6467
 - array, variable, 6467
 - biweight, 6477
 - boundary specifications, 6432
 - Box's bias and Hougaard's skewness measure, 6488
 - Box's bias and Hougaard's skewness measures, 6487
 - cancer remission data, 6480
 - conditional model expression, 6472
 - constant array, 6467
 - contrasts among parameters, 6496
 - convergence status, 6470
 - derivative code, 6445
 - derivatives, output, 6439
 - derivatives, zero, 6456
 - discontinuity, 6457
 - divergence, 6456
 - dose-response data, 6483
 - enzyme data, 6415
 - expected value parameterization, 6488
 - GENMOD procedure, 6482
 - getting started, 6414
 - GLIMMIX procedure, 6482
 - Hougaard's skewness measure, 6415, 6484
 - iteratively reweighted least squares, 6477
 - join point, 6472
 - join point, estimated, 6475
 - LD50 parameterization, 6487
 - local minimum, 6457
 - log-logistic model, 6484
 - LOGISTIC procedure, 6482
 - machine-level code, 6445
 - maximum likelihood estimation, 6480
 - model code, 6445
 - Newton-Raphson algorithm, 6480
 - non-identifiable parameter, 6456
 - ODS Graphics and diagnostics, 6500
 - one-compartment model, 6493
 - output, derivatives, 6439
 - output, predicted, 6488
 - output, predicted values, 6472
 - output, prediction limits, 6488
 - parameter differences, 6496
 - Parameter Profiling, 6505
 - pharmacokinetic model, 6493
 - plateau model, 6472
 - plot, observed and predicted, 6473
 - predicted values, 6472
 - probit model, 6480
 - programming statements, 6443
 - programming statements, efficiency, 6454
 - PUT statement, 6472
 - reparameterization, 6475, 6487, 6488, 6496
 - reparameterizing constraint, 6433
 - robust regression, 6477
 - ROBUSTREG procedure, 6479
 - segmented model, 6472
 - SGPLOT procedure, 6473, 6483, 6484, 6491, 6498
 - SIGSQ= option, 6480
 - simulated data, 6500
 - starting values, data set, 6441
 - starting values, grid, 6440
 - starting values, multiple, 6415
 - sum-of-squares reduction test, 6495, 6497
 - switching function, 6484
 - theophylline data, 6490
 - U.S. population growth data, 6477
 - variable array, 6467
 - varying parameters by groups, 6494
 - weight variable, 6477
 - weighted least squares, 6455

- g2 inverse
 - NLIN procedure, 6459
- g4 inverse
 - NLIN procedure, 6459
- generalized inverse
 - NLIN procedure, 6459
- group comparisons
 - NLIN procedure, 6489
- initial values
 - NLIN procedure, 6439
- Jacobian
 - NLIN procedure, 6422
- Lagrange multiplier
 - covariance matrix, 6465
 - NLIN procedure, 6433
- LMSELECT procedure
 - ODS Graphics, 6425
- log-logistic model
 - NLIN procedure, 6482
- Moore-Penrose inverse
 - NLIN procedure, 6459
- NLIN procedure
 - alpha level, 6439
 - analytic derivatives, 6434, 6445
 - automatic derivatives, 6445
 - bias, 6421
 - bias-corrected confidence interval, 6430
 - Box's bias measure, 6421
 - close-to-linear, 6447
 - comparing trends, 6489
 - confidence curves, 6442
 - confidence interval, 6430, 6436–6438, 6463
 - convergence, 6456
 - convergence criterion, 6421
 - covariance matrix, 6464
 - cross-referencing variables, 6430
 - debugging execution, 6423
 - derivatives, 6434, 6439, 6445
 - diagnostics for model with a high intrinsic curvature, 6500
 - g2 inverse, 6459
 - G4 inverse, 6423
 - g4 inverse, 6459
 - Gauss iterative method, 6424
 - Gauss-Newton iterative method, 6459, 6460
 - Gauss-Newton method, 6459
 - generalized inverse, 6459
 - gradient method, 6459, 6461
 - group comparisons, 6489
 - Hessian, 6465
 - Hougaard's measure, 6423, 6447
 - incompatibilities, 6466, 6467
 - initial values, 6439
 - iteratively reweighted least squares example, 6476
 - Jacobian, 6422
 - Lagrange multiplier, 6433
 - Lagrange multipliers, covariance matrix, 6465
 - log-logistic model, 6482
 - Marquardt iterative method, 6424, 6459, 6461
 - maximum iterations, 6424
 - maximum subiterations, 6424
 - mean square error specification, 6429
 - missing values, 6454
 - model confidence interval, 6463
 - model.variable syntax, 6435
 - Moore-Penrose inverse, 6459
 - Newton iterative method, 6424, 6459, 6461
 - object convergence measure, 6417
 - observation influence on parameters, 6442
 - options summary (PROC NLIN statement), 6420
 - output table names, 6468
 - parameter confidence interval, 6463
 - parameter histograms, 6430
 - parameter profiling and bootstrap for nonlinear models, 6505
 - PPC convergence measure, 6417
 - predicted values, output, 6437
 - prediction interval, 6436–6438
 - profile *t* plots, 6442
 - R convergence measure, 6417
 - residual values, output, 6438
 - retaining variables, 6434, 6443
 - RPC convergence measure, 6417
 - segmented model example, 6471
 - singularity criterion, 6429
 - skewness, 6418, 6423, 6447
 - SMETHOD=GOLDEN step size search, 6462
 - special variables, 6454
 - specifying bounds, 6432
 - standard error, 6438
 - starting values, 6439
 - steepest descent method, 6459, 6461
 - step size search, 6462
 - switching model, 6482
 - trend comparisons, 6489
 - troubleshooting, 6456
 - tuning display of iteration computation, 6424
 - weighted regression, 6455
- observation influence on parameters
 - NLIN procedure, 6442
- ODS Graphics
 - LMSELECT procedure, 6425
- ODS Graphics and diagnostics

- examples, NLIN, [6500](#)
- options summary (PROC NLIN statement)
 - NLIN procedure, [6420](#)
- parameter histograms
 - NLIN procedure, [6430](#)
- Parameter Profiling
 - examples, NLIN, [6505](#)
- parameter profiling and bootstrap for nonlinear models
 - NLIN procedure, [6505](#)
- PPC convergence measure, [6465](#)
- predicted values
 - NLIN procedure, [6437](#)
- profile t plots
 - NLIN procedure, [6442](#)
- R convergence measure, [6465](#)
- residuals
 - NLIN procedure, [6438](#)
- RPC convergence measure, [6465](#)
- S convergence measure, [6465](#)
- simulated data
 - examples, NLIN, [6500](#)
- singularity criterion
 - NLIN procedure, [6429](#)
- skewness
 - NLIN procedure, [6423](#)
- starting values
 - NLIN procedure, [6439](#)
- switching model
 - NLIN procedure, [6482](#)
- theophylline data
 - examples, NLIN, [6490](#)
- trend comparisons
 - NLIN procedure, [6489](#)

Syntax Index

- ALL option
 - PROFILE statement (NLIN), [6442](#)
- ALPHA= option
 - OUTPUT statement (NLIN), [6439](#)
 - PROC NLIN statement, [6421](#)
- BEST= option
 - PROC NLIN statement, [6421](#)
- BIAS option
 - PROC NLIN statement, [6421](#)
- BOOTCI option
 - BOOTSTRAP statement (NLIN), [6431](#)
- BOOTCORR option
 - BOOTSTRAP statement (NLIN), [6431](#)
- BOOTCOV option
 - BOOTSTRAP statement (NLIN), [6431](#)
- BOOTDATA option
 - BOOTSTRAP statement (NLIN), [6431](#)
- BOOTPLOTS option
 - BOOTSTRAP statement (NLIN), [6431](#)
- BOOTSTRAP statement
 - NLIN procedure, [6430](#)
- BOUNDS statement
 - NLIN procedure, [6432](#)
- BY statement
 - NLIN procedure, [6433](#)
- CONFCURV option
 - PROFILE statement (NLIN), [6442](#)
- CONTROL statement
 - NLIN procedure, [6434](#)
- CONVERGE= option
 - PROC NLIN statement, [6421](#)
- CONVERGEOBJ= option
 - PROC NLIN statement, [6422](#)
- CONVERGEPARM= option
 - PROC NLIN statement, [6422](#)
- DATA= option
 - PROC NLIN statement, [6422](#)
- DER option
 - OUTPUT statement (NLIN), [6439](#)
- DER statement
 - NLIN procedure, [6434](#)
- DGP option
 - BOOTSTRAP statement (NLIN), [6432](#)
- FLOW option
 - PROC NLIN statement, [6423](#)
- G4 option
 - PROC NLIN statement, [6423](#)
- H= option
 - OUTPUT statement (NLIN), [6436](#)
- HOUGAARD option
 - PROC NLIN statement, [6423](#)
- ID statement
 - NLIN procedure, [6434](#)
- J= option
 - OUTPUT statement (NLIN), [6436](#)
- JACKKNIFE option
 - PROFILE statement (NLIN), [6442](#)
- L95= option
 - OUTPUT statement (NLIN), [6436](#)
- L95M= option
 - OUTPUT statement (NLIN), [6436](#)
- LCL= option
 - OUTPUT statement (NLIN), [6437](#)
- LCLM= option
 - OUTPUT statement (NLIN), [6437](#)
- LIST option
 - PROC NLIN statement, [6423](#)
- LISTALL option
 - PROC NLIN statement, [6423](#)
- LISTCODE option
 - PROC NLIN statement, [6423](#)
- LISTDEP option
 - PROC NLIN statement, [6423](#)
- LISTDER option
 - PROC NLIN statement, [6424](#)
- LMAX= option
 - OUTPUT statement (NLIN), [6437](#)
- MAXITER= option
 - PROC NLIN statement, [6424](#)
- MAXSUBIT= option
 - PROC NLIN statement, [6424](#)
- METHOD= option
 - PROC NLIN statement, [6424](#)
- MODEL statement
 - NLIN procedure, [6434](#)
- NLIN procedure
 - syntax, [6419](#)
 - NLIN procedure, BOOTSTRAP statement, [6430](#)

- BOOTCI option, 6431
- BOOTCORR option, 6431
- BOOTCOV option, 6431
- BOOTDATA option, 6431
- BOOTPLOTS option, 6431
- DGP option, 6432
- NSAMPLES option, 6432
- SEED option, 6432
- NLIN procedure, BOUNDS statement, 6432
- NLIN procedure, BY statement, 6433
- NLIN procedure, CONTROL statement, 6434
- NLIN procedure, DER statement, 6434
- NLIN procedure, ID statement, 6434
- NLIN procedure, MODEL statement, 6434
- NLIN procedure, OUTPUT statement, 6435
 - ALPHA= option, 6439
 - DER option, 6439
 - H= option, 6436
 - J= option, 6436
 - L95= option, 6436
 - L95M= option, 6436
 - LCL= option, 6437
 - LCLM= option, 6437
 - LMAX= option, 6437
 - OUT= option, 6436
 - PARMS= option, 6437
 - PREDICTED= option, 6437
 - PROJRES= option, 6437
 - PROJSTUDENT= option, 6437
 - RESEXPEC= option, 6437
 - RESIDUAL= option, 6438
 - SSE= option, 6438
 - STDI= option, 6438
 - STDP= option, 6438
 - STDR= option, 6438
 - STUDENT= option, 6438
 - U95= option, 6438
 - U95M= option, 6438
 - UCL= option, 6438
 - UCLM= option, 6438
 - WEIGHT= option, 6439
- NLIN procedure, PARAMETERS statement, 6439
- NLIN procedure, PROC NLIN statement, 6420
 - ALPHA= option, 6421
 - BEST= option, 6421
 - BIAS option, 6421
 - CONVERGE= option, 6421
 - CONVERGEOBJ= option, 6422
 - CONVERGEPARM= option, 6422
 - DATA= option, 6422
 - FLOW option, 6423
 - G4 option, 6423
 - HOUGAARD option, 6423
 - LIST option, 6423
 - LISTALL option, 6423
 - LISTCODE option, 6423
 - LISTDEP option, 6423
 - LISTDER option, 6424
 - MAXITER= option, 6424
 - MAXSUBIT= option, 6424
 - METHOD= option, 6424
 - NLINMEASURES option, 6424
 - NOHALVE option, 6424
 - NOITPRINT option, 6424
 - NOPRINT option, 6424
 - OUTEST= option, 6424
 - PLOT option, 6425
 - PLOTS option, 6425
 - PRINT option, 6429
 - RHO= option, 6429
 - SAVE option, 6429
 - SIGSQ= option, 6429
 - SINGULAR= option, 6429
 - SMETHOD= option, 6430
 - TAU= option, 6430
 - TOTALSS option, 6430
 - TRACE option, 6430
 - UNCORRECTEDDF option, 6430
 - XREF option, 6430
- NLIN procedure, PROFILE statement, 6442
 - ALL option, 6442
 - CONFCURV option, 6442
 - JACKKNIFE option, 6442
 - PROFDATA= option, 6442
 - RANGE= option, 6443
 - TPLOT option, 6443
- NLIN procedure, program statements, 6443
- NLIN procedure, programming statements, 6443
- NLIN procedure, RETAIN statement, 6443
- NLINMEASURES option
 - PROC NLIN statement, 6424
- NOHALVE option
 - PROC NLIN statement, 6424
- NOITPRINT option
 - PROC NLIN statement, 6424
- NOPRINT option
 - PROC NLIN statement, 6424
- NSAMPLES option
 - BOOTSTRAP statement (NLIN), 6432
- OUT= option
 - OUTPUT statement (NLIN), 6436
- OUTEST= option
 - PROC NLIN statement, 6424
- OUTPUT statement
 - NLIN procedure, 6435
- PARAMETERS statement

- NLIN procedure, 6439
- PARMS= option
 - OUTPUT statement (NLIN), 6437
- PLOT option
 - PROC NLIN statement, 6425
- PLOTS option
 - PROC NLIN statement, 6425
- PREDICTED= option
 - OUTPUT statement (NLIN), 6437
- PRINT option
 - PROC NLIN statement, 6429
- PROC NLIN statement, *see* NLIN procedure
- PROFDATA= option
 - PROFILE statement, 6442
- PROFILE statement
 - NLIN procedure, 6442
- PROJRES= option
 - OUTPUT statement (NLIN), 6437
- PROJSTUDENT= option
 - OUTPUT statement (NLIN), 6437
- RANGE= option
 - PROFILE statement (NLIN), 6443
- RESEXPEC= option
 - OUTPUT statement (NLIN), 6437
- RESIDUAL= option
 - OUTPUT statement (NLIN), 6438
- RETAIN statement
 - NLIN procedure, 6443
- RHO= option
 - PROC NLIN statement, 6429
- SAVE option
 - PROC NLIN statement, 6429
- SEED option
 - BOOTSTRAP statement (NLIN), 6432
- SIGSQ= option
 - PROC NLIN statement, 6429
- SINGULAR= option
 - PROC NLIN statement, 6429
- SMETHOD= option
 - PROC NLIN statement, 6430
- SSE= option
 - OUTPUT statement (NLIN), 6438
- STDI= option
 - OUTPUT statement (NLIN), 6438
- STDP= option
 - OUTPUT statement (NLIN), 6438
- STDR= option
 - OUTPUT statement (NLIN), 6438
- STUDENT= option
 - OUTPUT statement (NLIN), 6438
- TAU= option
 - PROC NLIN statement, 6430

- TPLOT option
 - PROFILE statement (NLIN), 6443
- TRACE option
 - PROC NLIN statement, 6430
- TTOTALSS option
 - PROC NLIN statement, 6430
- U95= option
 - OUTPUT statement (NLIN), 6438
- U95M= option
 - OUTPUT statement (NLIN), 6438
- UCL= option
 - OUTPUT statement (NLIN), 6438
- UCLM= option
 - OUTPUT statement (NLIN), 6438
- UNCORRECTEDDF option
 - PROC NLIN statement, 6430
- WEIGHT= option
 - OUTPUT statement (NLIN), 6439
- XREF option
 - PROC NLIN statement, 6430