SAS/STAT® 14.1 User's Guide
The GLMSELECT Procedure

Chapter 49
The GLMSELECT Procedure

Contents

Overview: GLMSELECT Procedure

The GLMSELECT procedure performs effect selection in the framework of general linear models. A variety of model selection methods are available, including the LASSO method of Tibshirani (1996) and the related LAR method of Efron et al. (2004). The procedure offers extensive capabilities for customizing the selection with a wide variety of selection and stopping criteria, from traditional and computationally efficient significance-level-based criteria to more computationally intensive validation-based criteria. The procedure also provides graphical summaries of the selection search.

The GLMSELECT procedure compares most closely to REG and GLM. The REG procedure supports a variety of model-selection methods but does not support a CLASS statement. The GLM procedure supports a CLASS statement but does not include effect selection methods. The GLMSELECT procedure fills this gap. GLMSELECT focuses on the standard independently and identically distributed general linear model for univariate responses and offers great flexibility for and insight into the model selection algorithm. GLMSELECT provides results (displayed tables, output data sets, and macro variables) that make it easy to take the selected model and explore it in more detail in a subsequent procedure such as REG or GLM.

Features

The main features of the GLMSELECT procedure are as follows:

- **Model Specification**
  - supports different parameterizations for classification effects

  - supports any degree of interaction (crossed effects) and nested effects
  - supports hierarchy among effects
  - supports partitioning of data into training, validation, and testing roles
  - supports constructed effects including spline and multimember effects

- **Selection Control**
  - provides multiple effect selection methods
  - enables selection from a very large number of effects (tens of thousands)
  - offers selection of individual levels of classification effects
  - provides effect selection based on a variety of selection criteria
  - provides stopping rules based on a variety of model evaluation criteria
  - provides leave-one-out, *k*-fold cross validation, and *k*-fold external cross validation
  - supports data resampling and model averaging

- **Display and Output**
  - produces graphical representation of selection process
  - produces output data sets containing predicted values and residuals
  - produces an output data set containing the design matrix
  - produces macro variables containing selected models
  - supports parallel processing of BY groups
  - supports multiple SCORE statements

The GLMSELECT procedure supports the following effect selection methods. For more information about these methods, see the section "Model-Selection Methods" on page 3846.

forward selection	starts with no effects in the model and adds effects.
backward elimination	starts with all effects in the model and deletes effects.
stepwise regression	is similar to forward selection except that effects already in the model do not necessarily stay there.
least angle regression (LAR)	is similar to forward selection in that it starts with no effects in the model and adds effects. The parameter estimates at any step are "shrunk" when compared to the corresponding least squares estimates.
LASSO	adds and deletes parameters based on a version of ordinary least squares where the sum of the absolute regression coefficients is constrained.
elastic net	is an extension of LASSO that estimates parameters based on a version of ordinary least squares in which both the sum of the absolute regression coefficients and the sum of the squared regression coefficients are constrained.
group LASSO	is a variant of LASSO that estimates parameters based on a version of ordinary least squares in which the sum of the Euclidean norms of a group of regression coefficients is constrained.

PROC GLMSELECT also supports hybrid versions of the LAR and LASSO methods. They use LAR and LASSO to select the model but then estimate the regression coefficients by ordinary weighted least squares.

The GLMSELECT procedure is intended primarily as a model selection procedure and does not include regression diagnostics or other postselection facilities such as hypothesis testing, testing of contrasts, and LS-means analyses. The intention is that you use PROC GLMSELECT to select a model or a set of candidate models. Further investigation of these models can be done by using these models in existing regression procedures.

Getting Started: GLMSELECT Procedure

The Sashelp.Baseball data set contains salary and performance information for Major League Baseball players who played at least one game in both the 1986 and 1987 seasons, excluding pitchers. The salaries (*Sports Illustrated,* April 20, 1987) are for the 1987 season and the performance measures are from 1986 (Collier Books, *The 1987 Baseball Encyclopedia Update*). The following step displays in Figure 49.1 the variables in the data set:

```
proc contents varnum data=sashelp.baseball;
   ods select position;
run;
```

Suppose you want to investigate whether you can model the players' salaries for the 1987 season based on performance measures for the previous season. The aim is to obtain a parsimonious model that does not overfit this particular data, making it useful for prediction. This example shows how you can use PROC GLMSELECT as a starting point for such an analysis. Since the variation of salaries is much greater for the higher salaries, it is appropriate to apply a log transformation to the salaries before doing the model selection.

The following code selects a model with the default settings:

```
ods graphics on;
proc glmselect data=sashelp.baseball plots=all;
   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                     yrMajor crAtBat crHits crHome crRuns crRbi
                     crBB league division nOuts nAssts nError
                   / details=all stats=all;
run;
ods graphics off;
```

PROC GLMSELECT performs effect selection where effects can contain classification variables that you specify in a CLASS statement. The "Class Level Information" table shown in Figure 49.2 lists the levels of the classification variables Division and League.

Figure 49.1 Sashelp.Baseball Data Set

The CONTENTS Procedure

Variables in Creation Order

#	Variable	Type	Len	Label
1	Name	Char	18	Player's Name
2	Team	Char	14	Team at the End of 1986
3	nAtBat	Num	8	Times at Bat in 1986
4	nHits	Num	8	Hits in 1986
5	nHome	Num	8	Home Runs in 1986
6	nRuns	Num	8	Runs in 1986
7	nRBI	Num	8	RBIs in 1986
8	nBB	Num	8	Walks in 1986
9	YrMajor	Num	8	Years in the Major Leagues
10	CrAtBat	Num	8	Career Times at Bat
11	CrHits	Num	8	Career Hits
12	CrHome	Num	8	Career Home Runs
13	CrRuns	Num	8	Career Runs
14	CrRbi	Num	8	Career RBIs
15	CrBB	Num	8	Career Walks
16	League	Char	8	League at the End of 1986
17	Division	Char	8	Division at the End of 1986
18	Position	Char	8	Position(s) in 1986
19	nOuts	Num	8	Put Outs in 1986
20	nAssts	Num	8	Assists in 1986
21	nError	Num	8	Errors in 1986
22	Salary	Num	8	1987 Salary in $ Thousands
23	Div	Char	16	League and Division
24	logSalary	Num	8	Log Salary

Figure 49.2 Class Level Information

The GLMSELECT Procedure

Class Level Information

Class	Levels	Values
League	2	American National
Division	2	East West

When you specify effects that contain classification variables, the number of parameters is usually larger than the number of effects. The "Dimensions" table in Figure 49.3 shows the number of effects and the number of parameters considered.

Figure 49.3 Dimensions

Dimensions	
Number of Effects	19
Number of Parameters	21

Figure 49.4 Model Information

The GLMSELECT Procedure

Data Set	SASHELP.BASEBALL
Dependent Variable	logSalary
Selection Method	Stepwise
Select Criterion	SBC
Stop Criterion	SBC
Effect Hierarchy Enforced	None

You find details of the default search settings in the "Model Information" table shown in Figure 49.4. The default selection method is a variant of the traditional stepwise selection where the decisions about what effects to add or drop at any step and when to terminate the selection are both based on the Schwarz Bayesian information criterion (SBC). The effect in the current model whose removal yields the maximal decrease in the SBC statistic is dropped provided this lowers the SBC value. Once no decrease in the SBC value can be obtained by dropping an effect in the model, the effect whose addition to the model yields the lowest SBC statistic is added and the whole process is repeated. The method terminates when dropping or adding any effect increases the SBC statistic.

Figure 49.5 Candidates for Entry at Step Two

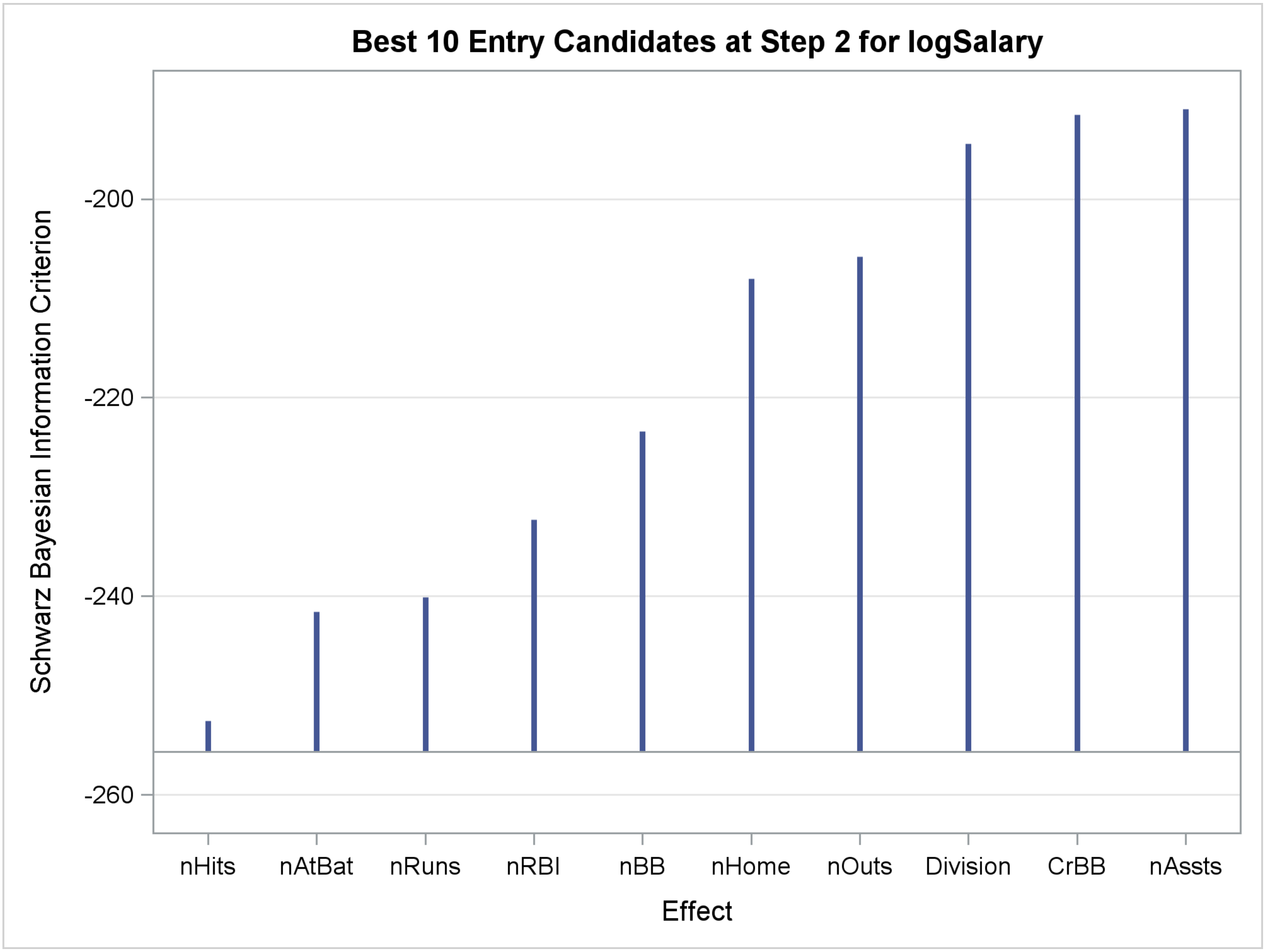
Best 10 Entry Candidates

Rank	Effect	SBC
1	nHits	-252.5794
2	nAtBat	-241.5789
3	nRuns	-240.1010
4	nRBI	-232.2880
5	nBB	-223.3741
6	nHome	-208.0565
7	nOuts	-205.8107
8	Division	-194.4688
9	CrBB	-191.5141
10	nAssts	-190.9425

The DETAILS=ALL option requests details of each step of the selection process. The “Best 10 Entry Candidates” table at each step shows the candidates for inclusion or removal at that step ranked from best to worst in terms of the selection criterion, which in this example is the SBC statistic. By default only the 10 best candidates are shown. Figure 49.5 shows the candidate table at step two.

To help in the interpretation of the selection process, you can use graphics supported by PROC GLMSELECT. ODS Graphics must be enabled before requesting plots. For general information about ODS Graphics, see Chapter 21, “Statistical Graphics Using ODS.” With ODS Graphics enabled, the PLOTS=ALL option together with the DETAILS=STEPS option in the MODEL statement produces a needle plot view of the “Candidates” tables. The plot corresponding to the “Candidates” table at step two is shown in Figure 49.6. You can see that adding the effect `'nHits'` yields the smallest SBC value, and so this effect is added at step two.

Figure 49.6 Needle Plot of Entry Candidates at Step Two



The "Stepwise Selection Summary" table in Figure 49.7 shows the effect that was added or dropped at each step of the selection process together with fit statistics for the model at each step. The STATS=ALL option in the MODEL statement requests that all the available fit statistics are displayed. See the section "Criteria Used in Model Selection Methods" on page 3857 for descriptions and formulas. The criterion panel in Figure 49.8 provides a graphical view of the progression of these fit criteria as the selection process evolves. Note that none of these criteria has a local optimum before step five.

Figure 49.7 Selection Summary Table

The GLMSELECT Procedure

Stepwise Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	Number Parms In	Model R-Square	Adjusted R-Square	AIC	AICC	BIC	CP	SBC
0	Intercept		1	1	0.0000	0.0000	204.2238	204.2699	-60.6397	375.9275	-57.2041
1	CrRuns		2	2	0.4187	0.4165	63.5391	63.6318	-200.7872	111.2315	-194.3166
2	nHits		3	3	0.5440	0.5405	1.7041	1.8592	-261.8807	33.4438	-252.5794
3	YrMajor		4	4	0.5705	0.5655	-12.0208	-11.7873	-275.3333	18.5870	-262.7322
4		CrRuns	3	3	0.5614	0.5581	-8.5517	-8.3967	-271.9095	22.3357	-262.8353
5	nBB		4	4	0.5818	0.5770*	-19.0690*	-18.8356*	-282.1700*	11.3524*	-269.7804*

*** Optimal Value of Criterion**

Stepwise Selection Summary

Step	Effect Entered	Effect Removed	PRESS	ASE	F Value	Pr > F
0	Intercept		208.7381	0.7877	0.00	1.0000
1	CrRuns		123.9195	0.4578	188.01	<.0001
2	nHits		97.6368	0.3592	71.42	<.0001
3	YrMajor		92.2998	0.3383	15.96	<.0001
4		CrRuns	93.1482	0.3454	5.44	0.0204
5	nBB		89.5434*	0.3294	12.62	0.0005

*** Optimal Value of Criterion**

The stop reason and stop details tables in Figure 49.9 gives details of why the selection process terminated. This table shows that at step five the best add candidate, `'Division'`, and the best drop candidate, `'nBB'`, yield models with SBC values of –268.6094 and –262.8353, respectively. Both of these values are larger than the current SBC value of –269.7804, and so the selection process stops at the model at step five.

Figure 49.8 Criterion Panel

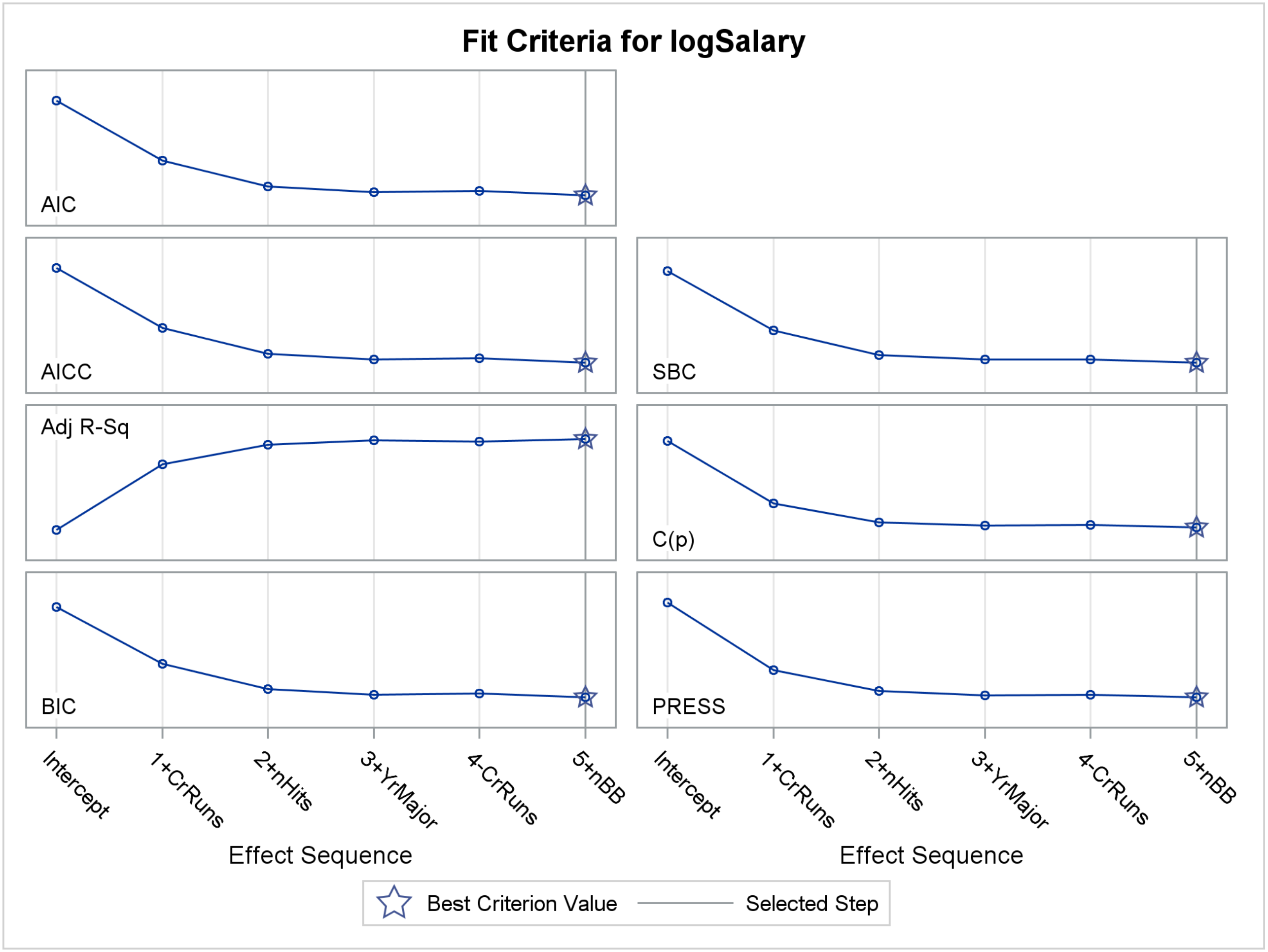


Figure 49.9 Stopping Details

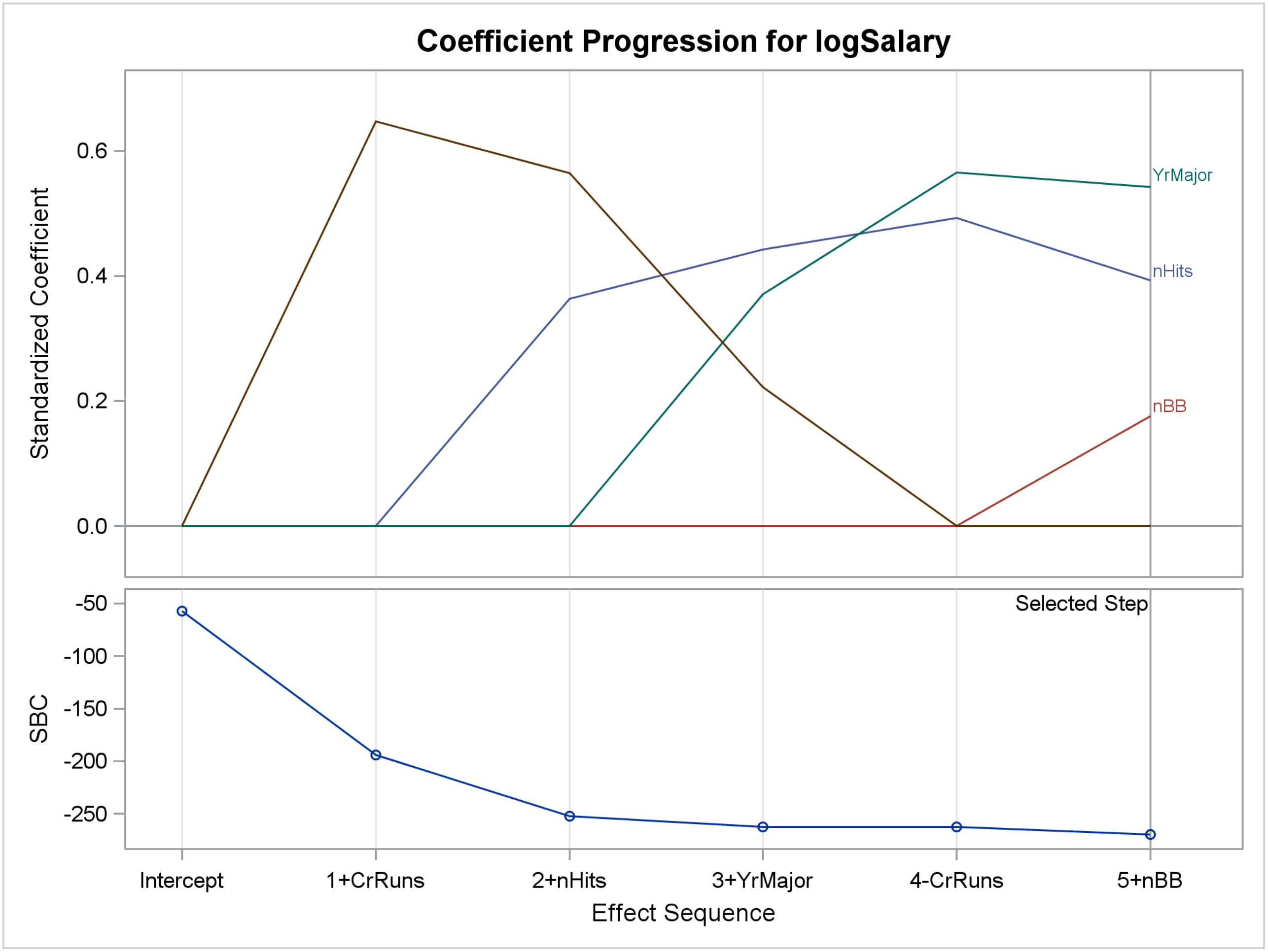
Selection stopped at a local minimum of the SBC criterion.

Stop Details

Candidate For	Effect	Candidate SBC		Compare SBC
Entry	Division	-268.6094	>	-269.7804
Removal	nBB	-262.8353	>	-269.7804

The coefficient panel in Figure 49.10 enables you to visualize the selection process. In this plot, standardized coefficients of all the effects selected at some step of the stepwise method are plotted as a function of the step number. This enables you to assess the relative importance of the effects selected at any step of the selection process as well as providing information as to when effects entered the model. The lower plot in the panel shows how the criterion used to choose the selected model changes as effects enter or leave the model.

Figure 49.10 Coefficient Progression



The selected effects, analysis of variance, fit statistics, and parameter estimates tables shown in Figure 49.11 give details of the selected model.

Figure 49.11 Details of the Selected Model

The GLMSELECT Procedure
Selected Model

The selected model is the model at the last step (Step 5).

Effects:	Intercept nHits nBB YrMajor

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value
Model	3	120.52553	40.17518	120.12
Error	259	86.62820	0.33447	
Corrected Total	262	207.15373		

Root MSE	0.57834
Dependent Mean	5.92722
R-Square	0.5818
Adj R-Sq	0.5770
AIC	-19.06903
AICC	-18.83557
BIC	-282.17004
C(p)	11.35235
PRESS	89.54336
SBC	-269.78041
ASE	0.32938

Parameter Estimates

Parameter	DF	Estimate	Standard Error	t Value
Intercept	1	4.013911	0.111290	36.07
nHits	1	0.007929	0.000994	7.98
nBB	1	0.007280	0.002049	3.55
YrMajor	1	0.100663	0.007551	13.33

PROC GLMSELECT provides you with the flexibility to use several selection methods and many fit criteria for selecting effects that enter or leave the model. You can also specify criteria to determine when to stop the selection process and to choose among the models at each step of the selection process. You can find continued exploration of the baseball data that uses a variety of these methods in Example 49.1.

Syntax: GLMSELECT Procedure

The following statements are available in the GLMSELECT procedure:

PROC GLMSELECT < *options* > **;**
- **BY** *variables* **;**
- **CLASS** *variable* < **(***v-options***)** > < *variable* < **(***v-options . . .***)** > > < / *v-options* > < *options* > **;**
- **CODE** < *options* > **;**
- **EFFECT** *name* **=** *effect-type* **(***variables* < / *options* >**)** **;**
- **FREQ** *variable* **;**
- **MODEL** *variable* **=** < *effects* > < / *options* > **;**
- **MODELAVERAGE** < *options* > **;**
- **OUTPUT** < **OUT=***SAS-data-set* > < *keyword*< **=***name* > > < . . . *keyword*< **=***name* > > **;**
- **PARTITION** < *options* > **;**
- **PERFORMANCE** < *options* > **;**
- **SCORE** < **DATA=***SAS-data-set* > < **OUT=***SAS-data-set* > **;**
- **STORE** < **OUT=** >*item-store-name* < / **LABEL=**'*label*' > **;**
- **WEIGHT** *variable* **;**

All statements other than the MODEL statement are optional and multiple SCORE statements can be used. CLASS and EFFECT statements, if present, must precede the MODEL statement.

The STORE and CODE statements are also used by many other procedures. A summary description of functionality and syntax for these statements is also shown after the PROC GLMSELECT statement in alphabetical order, but you can find full documentation about them in the section "STORE Statement" on page 512 in Chapter 19, "Shared Concepts and Topics."

PROC GLMSELECT Statement

PROC GLMSELECT < *options* > **;**

The PROC GLMSELECT statement invokes the GLMSELECT procedure. Table 49.1 summarizes the *options* available in the PROC GLMSELECT statement.

Table 49.1 PROC GLMSELECT Statement Options

Option	Description
Data Set Options	
DATA=	Names a data set to use for the regression
MAXMACRO=	Sets the maximum number of macro variables produced
TESTDATA=	Names a data set that contains test data
VALDATA=	Names a data set that contains validation data

Table 49.1 *continued*

Option	Description
ODS Graphics Options	
PLOTS=	Produces ODS graphical displays
Other Options	
OUTDESIGN=	Requests a data set that contains the design matrix
NAMELEN=	Sets the length of effect names in tables and output data sets
NOPRINT	Suppresses displayed output including plots
SEED=	Sets the seed used for pseudo-random number generation

Following are explanations of the *options* that you can specify in the PROC GLMSELECT statement (in alphabetical order).

DATA=*SAS-data-set*

names the SAS data set to be used by PROC GLMSELECT. If the DATA= option is not specified, PROC GLMSELECT uses the most recently created SAS data set. If the named data set contains a variable named _ROLE_, then this variable is used to assign observations for training, validation, and testing roles. See the section "Using Validation and Test Data" on page 3867 for details on using the _ROLE_ variable.

MAXMACRO=*n*

specifies the maximum number of macro variables with selected effects to create. By default, MAXMACRO=100. PROC GLMSELECT saves the list of selected effects in a macro variable, &_GLSIND. Say your input effect list consists of x1-x10. Then &_GLSIND would be set to x1 x3 x4 x10 if, for example, the first, third, fourth, and tenth effects were selected for the model. This list can be used, for example, in the model statement of a subsequent procedure. If you specify the OUTDESIGN= option in the PROC GLMSELECT statement, then PROC GLMSELECT saves the list of columns in the design matrix in a macro variable named &_GLSMOD.

With BY processing, one macro variable is created for each BY group, and the macro variables are indexed by the BY group number. The MAXMACRO= option can be used to either limit or increase the number of these macro variables when you are processing data sets with many BY groups.

With no BY processing, PROC GLMSELECT creates the following:

_GLSIND	selected effects
_GLSIND1	selected effects
_GLSMOD	design matrix columns
_GLSMOD1	design matrix columns
_GLSNUMBYS	number of BY groups
_GLSNUMMACROBYS	number of _GLSIND*i* macro variables actually made

With BY processing, PROC GLMSELECT creates the following:

_GLSIND	selected effects for BY group 1
_GLSIND1	selected effects for BY group 1
_GLSIND2	selected effects for BY group 2
.	
.	
.	
_GLSIND*m*	selected effects for BY group *m*, where a number is substituted for *m*
_GLSMOD	design matrix columns for BY group 1
_GLSMOD1	design matrix columns for BY group 1
_GLSMOD2	design matrix columns for BY group 2
.	
.	
.	
_GLSMOD*m*	design matrix columns for BY group *m*, where a number is substituted for *m*
_GLSNUMBYS	*n*, the number of BY groups
_GLSNUMMACROBYS	the number *m* of _GLSIND*i* macro variables actually made. This value can be less than _GLSNUMBYS = *n*, and it is less than or equal to the MAXMACRO= value.

See the section "Macro Variables Containing Selected Models" on page 3861 for further details.

NOPRINT

suppresses all displayed output including plots.

NAMELEN=*n*

specifies the length of effect names in tables and output data sets to be *n* characters long, where *n* is a value between 20 and 200 characters. The default length is 20 characters.

OUTDESIGN < (*options*) >< =*SAS-data-set* >

creates a data set that contains the design matrix. By default, the GLMSELECT procedure includes in the OUTDESIGN data set the **X** matrix corresponding to the parameters in the selected model. Two schemes for naming the columns of the design matrix are available. In the first scheme, names of the parameters are constructed from the parameter labels that appear in the ParameterEstimates table. This naming scheme is the default when you do not request BY processing and is not available when you do use BY processing. In the second scheme, the design matrix column names consist of a prefix followed by an index. The default naming prefix is _X.

You can specify the following *options* in parentheses to control the contents of the OUTDESIGN data set:

ADDINPUTVARS

requests that all variables in the input data set be included in the OUTDESIGN= data set.

FULLMODEL

specifies that parameters corresponding to all the effects specified in the MODEL statement be included in the OUTDESIGN= data set. By default, only parameters corresponding to the selected model are included.

NAMES

produces a table associating columns in the OUTDESIGN data set with the labels of the parameters they represent.

PREFIX< =prefix >

requests that the design matrix column names consist of a prefix followed by an index. The default naming prefix is _X. You can optionally specify a different prefix.

PARMLABELSTYLE=*options*

specifies how parameter names and labels are constructed for nested and crossed effects.

The following *options* are available:

INTERLACED < (SEPARATOR=*quoted-string*) >

forms parameter names and labels by positioning levels of classification variables and constructed effects adjacent to the associated variable or constructed effect name and using " * " as the delimiter for both crossed and nested effects. This style of naming parameters and labels is used in the TRANSREG procedure. You can request truncation of the classification variable names used in forming the parameter names and labels by using the CPREFIX= and LPREFIX= options in the CLASS statement. You can use the SEPARATOR= suboption to change the delimiter between the crossed variables in the effect. PARMLABELSTYLE=INTERLACED is not supported if you specify the SPLIT option in an EFFECT statement or a CLASS statement. The following are examples of the parameter labels in this style (Age is a continuous variable, Gender and City are classification variables):

```
Age
Gender male * City Beijing
City London * Age
```

SEPARATE

specifies that in forming parameter names and labels, the effect name appears before the levels associated with the classification variables and constructed effects in the effect. You can control the length of the effect name by using the NAMELEN= option in the PROC GLMSELECT statement. In forming parameter labels, the first level that is displayed is positioned so that it starts at the same offset in every parameter label—this enables you to easily distinguish the effect name from the levels when the parameter labels are displayed in a column in the "Parameter Estimates" table. This style of labeling is used in the GLM procedure and is the default if you do not specify the PARMLABELSTYLE option. The following are examples of the parameter labels in this style (Age is a continuous variable, Gender and City are classification variables):

```
Age
Gender*City male Beijing
Age*City    London
```

SEPARATECOMPACT

requests the same parameter naming and labeling scheme as PARMLABELSTYLE=SEPARATE except that the first level in the parameter label is separated from the effect name by a single blank. This style of labeling is used in the PLS procedure. The following are examples of the parameter labels in this style (Age is a continuous variable, Gender and City are classification variables):

```
Age
Gender*City male Beijing
Age*City London
```

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

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

controls the plots produced through ODS Graphics. When you specify only one plot request, you can omit the parentheses around the plot request. Here are some examples:

```
plots=all
plots=coefficients(unpack)
plots(unpack)=(criteria candidates)
```

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

```
ods graphics on;

proc glmselect plots=all;
   model y = x1-x100;
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."

Global Plot Options

The *global-options* apply to all plots generated by the GLMSELECT procedure, unless it is altered by a *specific-plot-option*.

ENDSTEP=*n*

specifies that the step ranges shown on the horizontal axes of plots terminates at specified step. By default, the step range shown terminates at the final step of the selection process. If you specify the ENDSTEP= option as both a global plot option and a specific plot option, then the ENDSTEP= value on the specific plot is used.

LOGP | LOGPVALUE

displays the natural logarithm of the entry and removal significance levels when the SELECT=SL option is specified.

MAXSTEPLABEL=*n*

specifies the maximum number of characters beyond which labels of effects on plots are truncated.

MAXPARMLABEL= *n*

specifies the maximum number of characters beyond which parameter labels on plots are truncated.

STARTSTEP=*n*

specifies that the step ranges shown on the horizontal axes of plots start at the specified step. By default, the step range shown starts at the initial step of the selection process. If you specify the STARTSTEP= option both as a global plot option and a specific plot option, then the STARTSTEP= value on the specific plot is used.

STEPAXIS=EFFECT | NORMB | NUMBER

specifies the horizontal axis to be used on the plots, where this axis represents the sequence of entering or departing effects.

STEPAXIS=EFFECT

requests that each step be labeled by a prefix followed by the name of the effect that enters or leaves at that step. The prefix consists of the step number, followed by a "+" sign or a "-" sign, depending on whether the effect enters (+) or leaves (-) at that step.

STEPAXIS=NORMB

is valid only with the LAR, LASSO, and elastic net selection methods and requests that the horizontal axis value at step *i* be the L1 norm of the parameters at step *i*, normalized by the L1 norm of the parameters at the final step.

STEPAXIS=NUMBER

requests that each step be labeled by the step number.

UNPACK

suppresses paneling. By default, multiple plots can appear in some output panels. Specify UNPACK to see each plot individually. You can also specify UNPACK as a suboption of the CRITERIA and COEFFICIENTS options.

Specific Plot Options

The following listing describes the specific plots and their *options*.

ALL

requests that all default plots be produced. Note that candidate plots are produced only if you specify DETAILS=STEPS or DETAILS=ALL in the MODEL statement.

ASE | ASEPLOT <(*aseplot-option***)>**

plots the progression of the average square error on the training data, and the test and validation data whenever these data are provided with the TESTDATA= and VALDATA= options or are produced by using a PARTITION statement. You can specify the following *aseplot-option*:

STEPAXIS=EFFECT | NORMB | NUMBER

specifies the horizontal axis to be used.

CANDIDATES | CANDIDATESPLOT < (*candidatesplot-options*) >

produces a needle plot of the SELECT= criterion values for the candidates for entry or removal at each step of the selection process, ordered from best to worst. Candidates plots are not available if you specify SELECTION=NONE, SELECTION=LAR, SELECTION=LASSO, SELECTION=ELASTICNET, or SELECTION=GROUPLASSO in the MODEL statement, or if you have not specified DETAILS=ALL or DETAILS=STEPS in the MODEL statement. The following *candidatesplot-options* are available:

LOGP | LOGPVALUE

displays the natural logarithm of the entry and removal significance levels when the SELECT=SL option is specified.

SHOW=*number*

specifies the maximum number of candidates displayed at each step. The default is SHOW=10.

COEFFICIENTS | COEFFICIENTPANEL < (*coefficientPanel-options*) >

plots a panel of two plots. The upper plot shows the progression of the parameter values as the selection process proceeds. The lower plot shows the progression of the CHOOSE= criterion. If no CHOOSE= criterion is in effect, then the AICC criterion is displayed. The following *coefficientPanel-options* are available:

LABELGAP=*percentage*

specifies the percentage of the vertical axis range that forms the minimum gap between successive parameter labels at the final step of the coefficient progression plot. If the values of more than one parameter at the final step are closer than this gap, then the labels on all but one of these parameters is suppressed. The default value is LABELGAP=5. Planned enhancements to the automatic label collision avoidance algorithm will obviate the need for this option in future releases of the GLMSELECT procedure.

LOGP | LOGPVALUE

displays the natural logarithm of the entry and removal significance levels when the SELECT=SL option is specified.

STEPAXIS=EFFECT | NORMB | NUMBER

specifies the horizontal axis to be used.

UNPACK | UNPACKPANEL

displays the coefficient progression and the CHOOSE= criterion progression in separate plots.

CRITERIA | CRITERIONPANEL < (*criterionPanel-options*) >

plots a panel of model fit criteria. The criteria that are displayed are ADJRSQ, AIC, AICC, and SBC, as well as any other criteria that are named in the CHOOSE=, SELECT=, STOP=, or STATS= option in the MODEL statement. The following *criterionPanel-options* are available:

STEPAXIS=EFFECT | NORMB | NUMBER

specifies the horizontal axis to be used.

UNPACK | UNPACKPANEL
displays each criterion progression on a separate plot.

EFFECTSELECTPCT < (*effectSelectPct-options*) >
requests a bar chart whose bars correspond to effects that are selected in at least one sample when you use the MODELAVERAGE statement. The length of a bar corresponds to the percentage of samples where the selected model contains the effect the bar represents. The EFFECTSELECTPCT option is ignored if you do not specify a MODELAVERAGE statement. The following *effectSelectPct-options* are available:

MINPCT=*percent*
specifies that effects that appear in fewer than the specified percentage of the sample selected models not be included in the plot. By default, effects that are shown in the EffectSelectPct table are displayed.

ORDER=ASCENDING | DESCENDING | MODEL
specifies the ordering of the effects in the bar chart. ORDER=MODEL specifies that effects appear in the order in which they appear in the MODEL statement. ORDER=ASCENDING | DESCENDING specifies that the effects are shown in ascending or descending order of the number of samples in which the effects appear in the selected model. The default is ORDER=DESCENDING.

NONE
suppresses all plots.

PARMDIST < (*parmDist-options*) >
produces a panel that shows histograms and box plots of the parameter estimate values across samples when you use a MODELAVERAGE statement. There is a histogram and box plot for each parameter that appears in the AvgParmEst table. The PARMDIST option is ignored if you do not specify a MODELAVERAGE statement. The following *parmDist-options* are available:

MINPCT=*percent*
specifies that distributions be shown only for parameters whose estimates are nonzero in at least the specified percentage of the selected models. By default, distributions are shown for the all parameters that appear in the AvgParmEst table.

ORDER=ASCENDING | DESCENDING | MODEL
specifies the ordering of the parameters in the panels. ORDER=MODEL specifies that parameters be shown in the order in which the corresponding effects appear in the MODEL statement. ORDER=ASCENDING | DESCENDING specifies that the parameters be shown in an ascending or descending order of the number of samples in which the parameter estimate is nonzero. The default is ORDER=DESCENDING.

NOBOXPLOTS
suppress the box plots.

PLOTSPERPANEL=*number*
specifies the maximum number of parameter distributions that appear in a panel. If the number of relevant parameters is greater than *number*, then multiple panels are produced. Valid values are 1–16 with 9 as the default.

UNPACK
specifies that the distribution for each relevant parameter be shown in a separate plot.

SEED=*number*
specifies an integer used to start the pseudo-random number generator for resampling the data, random cross validation, and random partitioning of data for training, testing, and validation. If you do not specify a seed, or if you specify a value less than or equal to zero, the seed is generated from reading the time of day from the computer's clock.

TESTDATA=*SAS-data-set*
names a SAS data set containing test data. This data set must contain all the variables specified in the MODEL statement. Furthermore, when a BY statement is used and the TESTDATA=data set contains any of the BY variables, then the TESTDATA= data set must also contain all the BY variables sorted in the order of the BY variables. In this case, only the test data for a specific BY group is used with the corresponding BY group in the analysis data. If the TESTDATA= data set contains none of the BY variables, then the entire TESTDATA = data set is used with each BY group of the analysis data.

If you specify a TESTDATA=data set, then you cannot also reserve observations for testing by using a PARTITION statement.

VALDATA=*SAS-data-set*
names a SAS data set containing validation data. This data set must contain all the variables specified in the MODEL statement. Furthermore, when a BY statement is used and the VALDATA=data set contains any of the BY variables, then the VALDATA= data set must also contain all the BY variables sorted in the order of the BY variables. In this case, only the validation data for a specific BY group are used with the corresponding BY group in the analysis data. If the VALDATA= data set contains none of the BY variables, then the entire VALDATA = data set is used with each BY group of the analysis data.

If you specify a VALDATA=data set, then you cannot also reserve observations for validation by using a PARTITION statement.

BY Statement

BY *variables* **;**

You can specify a BY statement with PROC GLMSELECT 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 GLMSELECT 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*.

CLASS Statement

CLASS *variable* < **(***v-options***)** > . . . < *variable* < **(***v-options***)** > > < / *options* > **;**

The CLASS statement names the classification variables to be used in the analysis. The CLASS statement must precede the MODEL statement.

Table 49.2 summarizes the *options* available in the CLASS statement.

Table 49.2 CLASS Statement Options

Option	Description
CPREFIX=	Specifies maximum number of CLASS variable name characters
DELIMITER=	Specifies the delimiter
DESCENDING	Reverses the sort order
LPREFIX=	Labels design variables
MISSING	Allows for missing values
ORDER=	Specifies the sort order
PARAM=	Specifies the parameterization method
REF=	Specifies the reference level
SHOW	Requests a table for each CLASS variable
SPLIT	Splits CLASS variables into independent effects

The following *options* can be specified after a slash (/):

DELIMITER=*quoted character*
: specifies the delimiter that is used between levels of classification variables in building parameter names and lists of class level values. The default if you do not specify DELIMITER= is a space. This option is useful if the levels of a classification variable contain embedded blanks.

SHOW | SHOWCODING
: requests a table for each classification variable that shows the coding used for that variable.

You can specify various *v-options* for each variable by enclosing them in parentheses after the variable name. You can also specify global *v-options* for the CLASS statement by placing them after a slash (/). Global *v-options* are applied to all the variables specified in the CLASS statement. If you specify more than one CLASS statement, the global *v-options* specified in any one CLASS statement apply to all CLASS statements. However, individual CLASS variable *v-options* override the global *v-options* except for the PARAM=GLM option. The global PARAM=GLM option overrides all individual PARAM= options.

The following *v-options* are available:

CPREFIX=*n*

specifies that, at most, the first *n* characters of a CLASS variable name be used in creating names for the corresponding design variables. The default is $32 - \min(32, \max(2, f))$, where f is the formatted length of the CLASS variable. The CPREFIX= applies only when you specify the PARMLABEL-STYLE=INTERLACED option in the PROC GLMSELECT statement.

DESCENDING

DESC

reverses the sort order of the classification variable.

LPREFIX=*n*

specifies that, at most, the first *n* characters of a CLASS variable label be used in creating labels for the corresponding design variables. The default is $256 - \min(256, \max(2, f))$, where f is the formatted length of the CLASS variable. The LPREFIX= applies only when you specify the PARMLABEL-STYLE=INTERLACED option in the PROC GLMSELECT statement.

MISSING

allows missing value ('.' for a numeric variable and blanks for a character variables) as a valid value for the CLASS variable.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of classification variables. This ordering determines which parameters in the model correspond to each level in the data, so the ORDER= option might be useful when you use the CONTRAST or ESTIMATE statement. If ORDER=FORMATTED for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values. Note that this represents a change from previous releases for how class levels are ordered. Before SAS 8, numeric class levels with no explicit format were ordered by their BEST12. formatted values, and in order to revert to the previous ordering you can specify this format explicitly for the affected classification variables. The change was implemented because the former default behavior for ORDER=FORMATTED often resulted in levels not being ordered numerically and usually required the user to intervene with an explicit format or ORDER=INTERNAL to get the more natural ordering. The following table shows how PROC GLMSELECT interprets values of the ORDER= option.

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For FORMATTED and INTERNAL, the sort order is machine dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Language Reference: Concepts*.

PARAM=*keyword*

specifies the parameterization method for the classification variable or variables. Design matrix columns are created from CLASS variables according to the following coding schemes. If the PARAM= option is not specified with any individual CLASS variable, by default, PARAM=GLM. Otherwise, the default is PARAM=EFFECT. If PARAM=ORTHPOLY or PARAM=POLY, and the CLASS levels are numeric, then the ORDER= option in the CLASS statement is ignored, and the internal, unformatted values are used. See the section "CLASS Variable Parameterization and the SPLIT Option" on page 3859 for further details.

EFFECT	specifies effect coding.
GLM	specifies less-than-full-rank, reference-cell coding; this option can be used only as a global option.
ORDINAL **THERMOMETER**	specifies the cumulative parameterization for an ordinal CLASS variable.
POLYNOMIAL **POLY**	specifies polynomial coding.
REFERENCE **REF**	specifies reference-cell coding.
ORTHEFFECT	orthogonalizes PARAM=EFFECT.
ORTHORDINAL **ORTHOTHERM**	orthogonalizes PARAM=ORDINAL.
ORTHPOLY	orthogonalizes PARAM=POLYNOMIAL.
ORTHREF	orthogonalizes PARAM=REFERENCE.

The EFFECT, POLYNOMIAL, REFERENCE, and ORDINAL schemes and their orthogonal parameterizations are full rank. The REF= option in the CLASS statement determines the reference level for the EFFECT and REFERENCE schemes and their orthogonal parameterizations.

REF=*'level' | keyword*

specifies the reference level for PARAM=EFFECT, PARAM=REFERENCE, and their orthogonalizations. For an individual (but not a global) variable REF= option, you can specify the *level* of the variable to use as the reference level. For a global or individual variable REF= option, you can use one of the following *keywords*. The default is REF=LAST.

FIRST	designates the first-ordered level as reference.
LAST	designates the last-ordered level as reference.

SPLIT

splits the columns of the design matrix that correspond to any effect that contains a split classification variable so that they can enter or leave a model independently of the other design columns for that effect. For example, suppose a variable named temp has three levels with values 'hot', 'warm', and 'cold', and a variable named sex has two levels with values 'M' and 'F' are used in a PROC GLMSELECT job as follows:

```
proc glmselect;
   class temp sex/split;
   model depVar = sex sex*temp;
run;
```

The two effects named in the MODEL statement are split into eight independent effects. The effect 'sex' is split into two effects labeled 'sex_M' and 'sex_F'. The effect 'sex*temp' is split into six effects labeled 'sex_M*temp_hot', 'sex_F*temp_hot', 'sex_M*temp_warm', 'sex_F*temp_warm', 'sex_M*temp_cold', and 'sex_F*temp_cold'. Thus the previous PROC GLMSELECT step is equivalent to the following step:

```
proc glmselect;
   model depVar =  sex_M sex_F sex_M*temp_hot  sex_F*temp_hot
                               sex_M*temp_warm sex_F*temp_warm
                               sex_M*temp_cold sex_F*temp_cold;
run;
```

The split option can be used on individual classification variables. For example, consider the following PROC GLMSELECT step:

```
proc glmselect;
   class temp(split) sex;
   model depVar = sex sex*temp;
run;
```

In this case the effect 'sex' is not split and the effect 'sex*temp' is split into three effects labeled 'sex*temp_hot', 'sex*temp_warm', and 'sex*temp_cold'. Furthermore each of these three split effects now has two parameters that correspond to the two levels of 'sex', and the PROC GLMSELECT step is equivalent to the following:

```
proc glmselect;
   class sex;
   model depVar = sex sex*temp_hot sex*temp_warm sex*temp_cold;
run;
```

CODE Statement

CODE < *options* > **;**

The CODE statement writes SAS DATA step code for computing predicted values of the fitted model either to a file or to a catalog entry. This code can then be included in a DATA step to score new data.

Table 49.3 summarizes the *options* available in the CODE statement.

Table 49.3 CODE Statement Options

Option	Description
CATALOG=	Names the catalog entry where the generated code is saved
DUMMIES	Retains the dummy variables in the data set
ERROR	Computes the error function
FILE=	Names the file where the generated code is saved
FORMAT=	Specifies the numeric format for the regression coefficients
GROUP=	Specifies the group identifier for array names and statement labels
IMPUTE	Imputes predicted values for observations with missing or invalid covariates
LINESIZE=	Specifies the line size of the generated code
LOOKUP=	Specifies the algorithm for looking up CLASS levels
RESIDUAL	Computes residuals

For details about the syntax of the CODE statement, see the section "CODE Statement" on page 399 in Chapter 19, "Shared Concepts and Topics."

EFFECT Statement

EFFECT *name*=*effect-type* **(***variables* < / *options* >**)** **;**

The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are referred to as *constructed effects* to distinguish them from the usual model effects that are formed from continuous or classification variables, as discussed in the section "GLM Parameterization of Classification Variables and Effects" on page 391 in Chapter 19, "Shared Concepts and Topics."

You can specify the following *effect-types*:

COLLECTION	specifies a collection effect that defines one or more variables as a single effect with multiple degrees of freedom. The variables in a collection are considered as a unit for estimation and inference.
LAG	specifies a classification effect in which the level that is used for a particular period corresponds to the level in the preceding period.
MULTIMEMBER | MM	specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement.

POLYNOMIAL | POLY	specifies a multivariate polynomial effect in the specified numeric variables.
SPLINE	specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

Table 49.4 summarizes the *options* available in the EFFECT statement.

Table 49.4 EFFECT Statement Options

Option	Description
Collection Effects Options	
DETAILS	Displays the constituents of the collection effect
Lag Effects Options	
DESIGNROLE=	Names a variable that controls to which lag design an observation is assigned
DETAILS	Displays the lag design of the lag effect
NLAG=	Specifies the number of periods in the lag
PERIOD=	Names the variable that defines the period. This option is required.
WITHIN=	Names the variable or variables that define the group within which each period is defined. This option is required.
Multimember Effects Options	
NOEFFECT	Specifies that observations with all missing levels for the multimember variables should have zero values in the corresponding design matrix columns
WEIGHT=	Specifies the weight variable for the contributions of each of the classification effects
Polynomial Effects Options	
DEGREE=	Specifies the degree of the polynomial
MDEGREE=	Specifies the maximum degree of any variable in a term of the polynomial
STANDARDIZE=	Specifies centering and scaling suboptions for the variables that define the polynomial
Spline Effects Options	
BASIS=	Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect
DEGREE=	Specifies the degree of the spline effect
KNOTMETHOD=	Specifies how to construct the knots for the spline effect

For more information about the syntax of these *effect-types* and how columns of constructed effects are computed, see the section “EFFECT Statement” on page 401 in Chapter 19, “Shared Concepts and Topics.”

FREQ Statement

FREQ *variable* **;**

The variable that is specified in the FREQ statement identifies a variable in the input data set that contains the frequency of occurrence of each observation. PROC GLMSELECT treats each observation as if it appears *n* times, where *n* is the value of the FREQ variable for the observation. If it is not an integer, the frequency value is truncated to an integer. If it is less than 1 or missing, the observation is not used.

MODEL Statement

MODEL *dependent* **=** < *effects* > **/** < *options* > **;**

The MODEL statement names the dependent variable and the explanatory effects, including covariates, main effects, constructed effects, interactions, and nested effects; for more information, see the section "Specification of Effects" on page 3593 in Chapter 46, "The GLM Procedure." If you omit the explanatory effects, the procedure fits an intercept-only model.

After the keyword MODEL, the dependent (response) variable is specified, followed by an equal sign. The explanatory effects follow the equal sign.

Table 49.5 summarizes the *options* available in the MODEL statement.

Table 49.5 MODEL Statement Options

Option	Description
CVDETAILS=	Requests details when cross validation is used
CVMETHOD=	Specifies how subsets for cross validation are formed
DETAILS=	Specifies details to be displayed
FUZZ=	Specifies the tolerance range for criterion comparisons
HIERARCHY=	Specifies the hierarchy of effects to impose
NOINT	Specifies models without an explicit intercept
ORDERSELECT	Requests that parameter estimates be displayed in the order in which the parameters first entered the model
SELECTION=	Specifies the model selection method
SHOWPVALUES	Requests p-values in "ANOVA" and "Parameter Estimates" tables
STATS=	Specifies additional statistics to be displayed
STB	Adds standardized coefficients to "Parameter Estimates" tables

You can specify the following *options* in the MODEL statement after a slash (/).

CVDETAILS=ALL | COEFFS | CVPRESS

specifies the details that are produced when cross validation is requested as the CHOOSE=, SELECT=, or STOP= criterion in the MODEL statement. If n-fold cross validation is being used, then the training data are subdivided into n parts, and at each step of the selection process, models are obtained on each of the n subsets of the data obtained by omitting one of these parts. CVDETAILS=COEFFS requests that the parameter estimates obtained for each of these *n* subsets be included in the parameter estimates table. CVDETAILS=CVPRESS requests a table containing the predicted residual sum of squares of each of these models scored on the omitted subset. CVDETAILS=ALL requests both CVDETAILS=COEFFS and CVDETAILS=CVPRESS. If DETAILS=STEPS or DETAILS=ALL has been specified in the MODEL statement, then the requested CVDETAILS are produced for every step of the selection process.

CVMETHOD=BLOCK< (*n*) > | RANDOM< (*n*) > | SPLIT< (*n*) > | INDEX (*variable*)

specifies how the training data are subdivided into *n* parts when you request *n*-fold cross validation by using any of the CHOOSE=CV, SELECT=CV, and STOP=CV suboptions of the SELECTION= option in the MODEL statement.

- **BLOCK** requests that parts be formed of *n* blocks of consecutive training observations.
- **SPLIT** requests that the *i*th part consist of training observations $i, i + n, i + 2n, \ldots$.
- **RANDOM** assigns each training observation randomly to one of the *n* parts.
- **INDEX(*variable*)** assigns observations to parts based on the formatted value of the named variable. This input data set variable is treated as a classification variable and the number of parts *n* is the number of distinct levels of this variable. By optionally naming this variable in a CLASS statement you can use the CLASS statement options ORDER= and MISSING to control how the levelization of this variable is done.

n defaults to 5 with CVMETHOD=BLOCK, CVMETHOD=SPLIT, or CVMETHOD=RANDOM. If you do not specify the CVMETHOD= option, then the CVMETHOD defaults to CVMETHOD=RANDOM(5).

DETAILS=*level* | STEPS < (*step options*) >

specifies the level of detail produced, where *level* can be ALL, STEPS, or SUMMARY. The default if the DETAILS= option is omitted is DETAILS=SUMMARY. The DETAILS=ALL option produces the following:

- entry and removal statistics for each variable selected in the model building process
- ANOVA, fit statistics, and parameter estimates
- entry and removal statistics for the top 10 candidates for inclusion or exclusion at each step
- a selection summary table

The DETAILS=SUMMARY option produces only the selection summary table.

The option DETAILS=STEPS < (*step options*) > provides the step information and the selection summary table. The following *options* can be specified within parentheses after the DETAILS=STEPS option:

ALL

requests ANOVA, fit statistics, parameter estimates, and entry or removal statistics for the top 10 candidates for inclusion or exclusion at each selection step.

ANOVA

requests ANOVA at each selection step.

FITSTATISTICS | FITSTATS | FIT

requests fit statistics at each selection step. The default set of statistics includes all the statistics named in the CHOOSE=, SELECT=, and STOP= suboptions specified in the MODEL statement SELECTION= option, but you can request additional statistics by specifying the STATS= option in the MODEL statement.

PARAMETERESTIMATES | PARMEST

requests parameter estimates at each selection step.

CANDIDATES < (SHOW= ALL | *n*) >

requests entry or removal statistics for the best *n* candidate effects for inclusion or exclusion at each step. If you specify SHOW=ALL, then all candidates are shown. If SHOW= is not specified. then the best 10 candidates are shown. The entry or removal statistic is the statistic named in the SELECT= option that is specified in the SELECTION= option in the MODEL statement.

FUZZ=*value*

specifies the tolerance range for criterion comparisons. Criterion values that differ by less than the tolerance are regarded as equal. If you specify FUZZ=0, then the comparisons are based on simple equality. The default is 10^{-12}.

HIERARCHY=NONE | SINGLE | SINGLECLASS

HIER=NONE | SINGLE | SINGLECLASS

specifies whether and how the model hierarchy requirement is applied. This option also controls whether a single effect or multiple effects are allowed to enter or leave the model in one step. You can specify that only classification effects, or both classification and continuous effects, be subject to the hierarchy requirement. The HIERARCHY= option is ignored unless you also specify one of the following *options*: SELECTION=FORWARD, SELECTION=BACKWARD, or SELECTION=STEPWISE.

Model hierarchy refers to the requirement that for any term to be in the model, all model effects contained in the term must be present in the model. For example, in order for the interaction A*B to enter the model, the main effects A and B must be in the model. Likewise, neither effect A nor effect B can leave the model while the interaction A*B is in the model.

You can specify the following values:

NONE

specifies that model hierarchy not be maintained. Any single effect can enter or leave the model at any given step of the selection process.

SINGLE

specifies that only one effect enter or leave the model at one time, subject to the model hierarchy requirement. For example, suppose that the model contains the main effects A and B and the interaction A*B. In the first step of the selection process, either A or B can enter the model. In the second step, the other main effect can enter the model. The interaction effect can enter the model only when both main effects have already entered. Also, before A or B can be removed from the model, the A*B interaction must first be removed. All effects (CLASS and interval) are subject to the hierarchy requirement.

SINGLECLASS
is the same as HIERARCHY=SINGLE except that only CLASS effects are subject to the hierarchy requirement.

By default, HIERARCHY=NONE.

NOINT
suppresses the intercept term that is otherwise included in the model.

ORDERSELECT
specifies that for the selected model, effects be displayed in the order in which they first entered the model. If you do not specify the ORDERSELECT option, then effects in the selected model are displayed in the order in which they appeared in the MODEL statement.

SELECTION=*method* **<(***method-options***)>**
specifies the method used to select the model, optionally followed by parentheses enclosing options applicable to the specified method. The default if the SELECTION= option is omitted is SELECTION=STEPWISE.

You can specify the following *methods*, which are explained in detail in the section "Model-Selection Methods" on page 3846:

NONE	specifies no model selection.
FORWARD	specifies forward selection. This method starts with no effects in the model and adds effects.
BACKWARD	specifies backward elimination. This method starts with all effects in the model and deletes effects.
STEPWISE	specifies stepwise regression. This is similar to the forward selection method except that effects already in the model do not necessarily stay there.
LAR	specifies least angle regression. This method, like forward selection, starts with no effects in the model and adds effects. The parameter estimates at any step are "shrunk" when compared to the corresponding least squares estimates. If the model contains classification variables, then these classification variables are split. For more information, see the SPLIT option in the CLASS statement.
LASSO	specifies the LASSO method, which adds and deletes parameters based on a version of ordinary least squares where the sum of the absolute regression coefficients is constrained. If the model contains classification variables, then these classification variables are split. For more information, see the SPLIT option in the CLASS statement.
ELASTICNET	specifies the elastic net method, an extension of LASSO that estimates parameters based on a version of ordinary least squares in which both the sum of the absolute regression coefficients and the sum of the squared regression coefficients are constrained. If the model contains classification variables, then these classification variables are split. For more information, see the SPLIT option in the CLASS statement.
GROUPLASSO	specifies the group LASSO method, a variant of LASSO that estimates parameters based on a version of ordinary least squares in which the sum of the Euclidean norms of a group of regression coefficients is constrained.

Table 49.6 lists the applicable *method-options* for each of these methods.

Table 49.6 Applicable SELECTION= Options by Method

Option	FORWARD	BACKWARD	STEPWISE	LAR	LASSO	ELASTICNET	GROUPLASSO
STOP=	x	x	x	x	x	x	x
CHOOSE=	x	x	x	x	x	x	x
STEPS=	x	x	x	x	x	x	x
MAXSTEP=	x	x	x	x	x	x	x
SELECT=	x	x	x				
INCLUDE=	x	x	x				
SLENTRY=	x		x	x	x	x	
SLSTAY=		x	x		x	x	
DROP=			x				
ADAPTIVE					x		
LSCOEFFS				x	x		
L1=					x	x	x
L1CHOICE=					x	x	
L2=						x	
L2STEPS=						x	
L2LOW=						x	
L2HIGH=						x	
L2SEARCH=						x	
ENSCALE						x	
RHO=							x
SCREEN=					x	x	
TOL=							x

The syntax of the *method-options* follows. Note that, as described in Table 49.6, not all selection *method-options* are available for every SELECTION= method.

ADAPTIVE < (< GAMMA=*nonnegative number* **> < INEST=***SAS-data-set* **>) >**

requests that adaptive weights be applied to each of the coefficients in the LASSO method. You use the optional INEST= option to name the SAS data set that contains estimates that are used to form the adaptive weights for all the parameters in the model. If you do not specify an INEST= data set, then ordinary least squares estimates of the parameters in the model are used in forming the adaptive weights. You use the GAMMA= option to specify the power transformation that is applied to the parameters in forming the adaptive weights. By default, GAMMA=1.

CHOOSE=*criterion*

specifies the criterion for choosing the model. The specified *criterion* is evaluated at each step of the selection process, and the model that yields the best value of the criterion is chosen. If the optimal value of the criterion occurs for models at more than one step, then the model that has the smallest number of parameters is chosen. If you do not specify the CHOOSE= option, then the model at the final step in the selection process is selected.

The criteria that you can specify in the CHOOSE= option are shown in Table 49.7. For more information about these criteria, see the section "Criteria Used in Model Selection Methods" on page 3857.

Table 49.7 Criteria for the CHOOSE= Option

Criterion	**Criterion**
ADJRSQ	Adjusted R-square statistic
AIC	Akaike's information criterion
AICC	Corrected Akaike's information criterion
BIC	Sawa Bayesian information criterion
CP	Mallows' C_p statistic
CV	Predicted residual sum of square with k-fold cross validation
CVEX	Predicted residual sum of square with k-fold external cross validation
PRESS	Predicted residual sum of squares
SBC	Schwarz Bayesian information criterion
VALIDATE	Average square error for the validation data

For ADJRSQ, the chosen value is the largest one; for all other criteria, the smallest value is chosen. You can use the CHOOSE=VALIDATE option only if you have specified a VALDATA= data set in the PROC GLMSELECT statement or if you have reserved part of the input data for validation by using either a PARTITION statement or a _ROLE_ variable in the input data. The PRESS criterion is not available for SELECTION=ELASTICNET. The CHOOSE=PRESS option cannot be used with SELECTION=LAR or SELECTION=LASSO unless the LSCOEFFS suboption is also specified. The BIC, CP, CV, and PRESS criteria are not available for SELECTION=GROUPLASSO.

DROP=BEFOREADD | COMPETITIVE

specifies when effects are eligible to be dropped in the STEPWISE method. You can specify the following values:

BEFOREADD requests that currently in the model be examined to see if any meet the requirements to be removed from the model. If so, the effect that gives the best value of the removal criterion is dropped from the model and the stepwise method proceeds to the next step. Only when no effect currently in the model meets the requirement to be removed from the model are any effects added to the model.

COMPETITIVE requests that the SELECT= criterion be evaluated for all models in which an effect currently in the model is dropped or an effect not yet in the model is added. The effect whose removal or addition to the model yields the maximum improvement to the SELECT= criterion is dropped or added. You can specify DROP=COMPETITIVE only if the SELECT= criterion is not SL.

By default, DROP=BEFOREADD. If SELECT=SL, PROC GLMSELECT uses the traditional stepwise method as implemented in PROC REG.

ENSCALE

requests that the solution to SELECTION=ELASTICNET be scaled to offset bias because of the double shrinkage inherent in the elastic net method (Zou and Hastie 2005). This option applies only when SELECTION=ELASTICNET. The default is not to rescale the solution: this is the so-called naive elastic net.

INCLUDE=*n*

forces the first *n* effects listed in the MODEL statement to be included in all models. The selection methods are performed on the other effects in the MODEL statement. The INCLUDE= option is available only when SELECTION=FORWARD, SELECTION=STEPWISE, and SELECTION=BACKWARD.

L1=*value*

specifies the LASSO regularization or constraint parameter that is used when SELECTION=LASSO, SELECTION=ELASTICNET or SELECTION=GROUPLASSO. This option is available only when you specify the STOP=L1 option with SELECTION=LASSO, SELECTION=ELASTICNET, or SELECTION=GROUPLASSO.

L1CHOICE=NORM | RATIO | VALUE

specifies both the criterion used in the L1=*value* option and the criterion used in aggregating the results of *k*-fold external cross validation for computing the CVEXPRESS statistic. This option is available only when you specify SELECTION=LASSO or SELECTION=ELASTICNET. You can specify the following values:

NORM	indicates that the value specified in the L1=*value* option corresponds to the sum of the absolute values of the coefficients (the so-called L1 norm), and the *k*-fold external cross validation aggregation is based on the L1 norms.
RATIO	indicates that the value specified in the L1=*value* option corresponds to the ratio obtained by scaling the value of the LASSO regularization parameter to lie in the interval [0,1], and the *k*-fold external cross validation aggregation is based on the scaled ratios.
VALUE	indicates that the value specified in the L1=*value* option corresponds to the actual value of the LASSO regularization parameter, and the *k*-fold external cross validation aggregation is based on the actual LASSO regularization parameters.

By default, L1CHOICE=RATIO. If you specify SELECTION=GROUPLASSO, L1CHOICE=RATIO is used.

L2=*value*

specifies the ridge regularization parameter that is used when SELECTION=ELASTICNET. The L2= option is available only when SELECTION=ELASTICNET. If you specify the L2= option, then the value that you specify is used in defining the elastic net method, and the L2HIGH=, L2LOW=, L2SEARCH=, and L2STEPS= options are ignored. If you do not specify the L2 = option with SELECTION=ELASTICNET, then PROC GLMSELECT searches for the suitable value of L2 according to the L2HIGH=, L2LOW=, L2SEARCH=, and L2STEPS= options.

L2HIGH=*value*

specifies the highest value used in the search of the ridge regression parameter L2 when SELECTION=ELASTICNET. If you specify the L2= option, then the L2HIGH= option is ignored. By default, L2HIGH=1.

L2LOW=*value*

specifies the lowest value used in the search of the ridge regression parameter L2 when SELECTION=ELASTICNET. If you specify the L2= option, then the L2LOW= option is ignored. By default, L2LOW=0.

L2SEARCH=GOLDEN | GRID

specifies the approach for the search of the ridge regression parameter L2 for the SELECTION=ELASTICNET option. You can specify the following values:

GOLDEN	requests a golden section search of L2 in the range [low, high], where low and high are specified by the L2LOW= and L2HIGH= options, respectively.
GRID	requests a log scale grid search of L2 in the range [low, high], where low and high are specified by the L2LOW= and L2HIGH= options, respectively. If L2LOW=0, then the log scale grid search for L2 is in the range [10^{-8}, high] plus 0.

If you specify the L2= option with SELECTION=ELASTICNET, then the L2SEARCH= option is ignored. By default, L2SEARCH=GRID.

L2STEPS=*n*

specifies the number of steps in the search of the ridge regression parameter L2 when SELECTION=ELASTICNET. If you specify the L2 = option, then the L2STEPS= option is ignored. By default, L2STEPS=50.

LSCOEFFS

requests a hybrid version of the LAR or LASSO method, in which the sequence of models is determined by the LAR or LASSO method but the coefficients of the parameters for the model at any step are determined by using ordinary least squares.

MAXSTEP=*n*

specifies the maximum number of selection steps that are performed. The default value of *n* is the number of effects in the model statement for the forward, backward, and LAR methods, two times the number of effects for the stepwise, LASSO, and elastic net methods and 100 for the group LASSO method.

SELECT=*criterion*

specifies the criterion that PROC GLMSELECT uses to determine the order in which effects enter or leave at each step of the specified selection method. The SELECT= option is not valid with the LAR, LASSO, elastic net, and group LASSO methods. The criteria that you can specify with the SELECT= option are ADJRSQ, AIC, AICC, BIC, CP, CV, PRESS, RSQUARE, SBC, SL, and VALIDATE. For more information about these criteria, see the section "Criteria Used in Model Selection Methods" on page 3857. The default value of the SELECT= criterion is SELECT=SBC. You can use SELECT=SL to request the traditional approach, in which effects enter and leave the model based on the significance level. For other SELECT= criteria, the effect that is selected to enter or leave at a step of the selection process is the effect whose addition to or removal from the current model gives the maximum improvement in the specified criterion.

RHO=*value*

specifies the value of ρ that determines the sequence of regularization parameters $[\rho, \rho^2, \rho^3, \rho^4, \ldots]$ used in SELECTION=GROUPLASSO. By default, RHO=0.9.

SCREEN=NONE | SASVI | SIS< (*sis-options***) >**

specifies which screening method to apply. This option is ignored unless you also specify SELECTION=LASSO or SELECTION=ELASTICNET. In addition, the SCREEN= option is ignored when the SSCP matrix is computed in full. For more information about the SSCP matrix, see "Building the SSCP Matrix" on page 3865.

You can specify the followings values:

NONE
specifies no screening.

SASVI
uses the SASVI safe screening technique of Liu et al. (2014). The resulting solution is identical to the one that results from SCREEN=NONE, but it can often be several times faster to obtain. For more information about the SASVI technique, see the section "Safe Screening via SCREEN=SASVI" on page 3874.

SIS< (*sis-options***) >**
uses the sure independence screening (SIS) technique of Fan and Lv (2008). Because SIS is a heuristic screening rule, the resulting solution is not necessarily identical to the one that results from SCREEN=NONE. However, it can be much faster to obtain when the number of potential predictors is very large.

You can specify the following *sis-options*:

KEEPNUM=*n*
keeps *n* effects that have the largest screening statistic values. The kept effects are then processed by the method that is specified in the SELECTION= option for model selection.

KEEPRATIO=*p*
specifies a value in the range 0 to 1 for *p*, and keeps (*p**100)% effects that have the largest screening statistic values. The kept effects are then processed by the method that is specified in the SELECTION= option for model selection.

If you do not specify any *sis-options*, KEEPNUM=100 by default. If you specify both KEEPNUM=*n* and KEEPRATIO=*value*, KEEPRATIO=*value* is used.

By default, SCREEN=NONE, so that no screening is done. In this case, the LASSO or elastic net method is performed on all the potential predictors.

SLENTRY=*value*
SLE=*value*
specifies the significance level for entry, which is used when the STOP=SL or SELECT=SL option is specified. The default is 0.50 when SELECTION=FORWARD, SELECTION=LAR, SELECTION=LASSO or SELECTION=ELASTICNET. The default is 0.15 when SELECTION=STEPWISE.

SLSTAY=*value*
SLS=*value*
specifies the significance level for staying in the model, which is used when the STOP=SL or SELECT=SL option is specified. The default is 0.10 when SELECTION=BACKWARD, SELECTION=LAR, SELECTION=LASSO or SELECTION=ELASTICNET. The default is 0.15 when SELECTION=STEPWISE.

STEPS=*n*
specifies the number of selection steps to be done. If the STEPS= option is specified, the STOP= and MAXSTEP= options are ignored.

STOP=*n*

STOP=*criterion*

specifies when PROC GLMSELECT is to stop the selection process. If the STEPS= option is specified, then the STOP= option is ignored. If the STOP=option does not cause the selection process to stop before the maximum number of steps for the selection method, then the selection process terminates at the maximum number of steps.

If you do not specify the STOP= option but do specify the SELECT= option, then the criterion named in the SELECT=option is also used as the STOP= criterion. If you do not specify either the STOP= or SELECT= option, then by default STOP=SBC.

If you specify STOP=*n*, then PROC GLMSELECT stops selection at the first step for which the selected model has *n* effects.

The nonnumeric arguments that you can specify in the STOP= option are shown in Table 49.8. For more information about these criteria, see the section "Criteria Used in Model Selection Methods" on page 3857.

Table 49.8 Nonnumeric Criteria for the STOP= Option

Option	Criteria
NONE	
ADJRSQ	Adjusted R-square statistic
AIC	Akaike's information criterion
AICC	Corrected Akaike's information criterion
BIC	Sawa Bayesian information criterion
CP	Mallows' C_p statistic
CV	Predicted residual sum of square with k-fold cross validation
L1	The LASSO regularization or constraint parameter
PRESS	Predicted residual sum of squares
SBC	Schwarz Bayesian information criterion
SL	Significance level
VALIDATE	Average square error for the validation data

When you use the SL criterion, selection stops at the step where the significance level for entry of all the effects not yet in the model is greater than the SLE= value for addition steps in the forward and stepwise methods and where the significance level for removal of any effect in the current model is smaller than the SLS= value in the backward and stepwise methods. When you use the ADJRSQ criterion, selection stops at the step where the next step would yield a model that has a smaller value of the adjusted R-square statistic; for all other criteria, selection stops at the step where the next step would yield a model that has a larger value of the criteria. You can use the VALIDATE option only if you have specified a VALDATA= data set in the PROC GLMSELECT statement or if you have reserved part of the input data for validation by using either a PARTITION statement or a _ROLE_ variable in the input data.

The L1 criterion is available only when SELECTION=LASSO or SELECTION=ELASTICNET. When you use the L1 criterion, selection stops at the step where the LASSO regularization parameter is equal to the value specified by the L1=*value* option. The PRESS criterion is not available for SELECTION=ELASTICNET. The STOP=PRESS option cannot be used with SELECTION=LAR or

SELECTION=LASSO unless the LSCOEFFS suboption is also specified. The BIC, CP, CV, PRESS, and SL criteria are not available for SELECTION=GROUPLASSO.

STAT|STATS=*name*

STATS=(*names***)**

specifies which model fit statistics are displayed in the fit summary table and fit statistics tables. If you omit the STATS= option, the default set of statistics that are displayed in these tables includes all the criteria specified in any of the CHOOSE=, SELECT=, and STOP= options specified in the MODEL statement SELECTION= option.

You can specify the following statistics:

ADJRSQ	specifies the adjusted R-square statistic.
AIC	specifies Akaike's information criterion.
AICC	specifies corrected Akaike's information criterion.
ASE	specifies the average square errors for the training, test, and validation data. The ASE statistics for the test and validation data are reported only if you specify TESTDATA= or VALDATA= in the PROC GLMSELECT statement or if you have reserved part of the input data for testing or validation by using either a PARTITION statement or a _ROLE_ variable in the input data.
BIC	specifies the Sawa Bayesian information criterion.
CP	specifies the Mallows' C_p statistic.
FVALUE	specifies the F statistic for entering or departing effects.
PRESS	specifies the predicted residual sum of squares statistic.
RSQUARE	specifies the R-square statistic.
SBC	specifies the Schwarz Bayesian information criterion.
SL	specifies the significance level of the F statistic for entering or departing effects.

The statistics ADJRSQ, AIC, AICC, FVALUE, RSQUARE, SBC, and SL can be computed with little computation cost. However, computing BIC, CP, CVPRESS, PRESS, and ASE for test and validation data when these are not used in any of the CHOOSE=, SELECT=, and STOP= options specified in the MODEL statement SELECTION= option can hurt performance.

SHOWPVALUES

SHOWPVALS

displays p-values in the "ANOVA" and "Parameter Estimates" tables. These p-values are generally liberal because they are not adjusted for the fact that the terms in the model have been selected.

STB

produces standardized regression coefficients. A standardized regression coefficient is computed by dividing a parameter estimate by the ratio of the sample standard deviation of the dependent variable to the sample standard deviation of the regressor.

TOL=*value*

specifies the tolerance value that determines the optimization precision in SELECTION= GROUPLASSO. The default *value* is 10^{-6}.

MODELAVERAGE Statement

MODELAVERAGE < *options* > **;**

The MODELAVERAGE statement requests that model selection be repeated on resampled subsets of the input data. An average model is produced by averaging the parameter estimates of the selected models that are obtained for each resampled subset of the input data.

Table 49.9 summarizes the *options* available in the MODELAVERAGE statement.

Table 49.9 MODELAVERAGE Statement Options

Option	Description
ALPHA=	Specifies lower and upper quantiles of the sample parameter
DETAILS	Displays model selection details
NSAMPLES=	Specifies the number of samples used for the refit averaging
REFIT	Performs a second round of model averaging
SAMPLING=	Specifies how to generate the samples taken from the training data
SUBSET	Uses only a subset of the selected models in forming the average model
TABLES	Controls the displayed tables

The following *options* are available:

ALPHA=α

controls which lower and upper quantiles of the sample parameter estimates are displayed. The ALPHA= option also controls which quantiles of the predicted values are added to the output data set when the LOWER= and UPPER= options are specified in the OUTPUT statement. The lower and upper quantiles used are $\alpha/2$ and $1-\alpha/2$, respectively. The value specified must lie in the interval [0, 1]. The default value is ALPHA=0.5.

DETAILS

requests that model selection details be displayed for each sample of the data. The level of detail shown is controlled by the DETAILS= option in the MODEL statement.

NSAMPLES=*n*

specifies the number of samples to be used. The default value is NSAMPLES=100.

REFIT < **(***refit-options***)** >

requests that a second round of model averaging, referred to as the refit averaging, be performed. Usually, the initial round of model averaging produces a model that contains a large number of effects. You can use the refit option to obtain a more parsimonious model. For each data sample in the refit, a least squares model is fit with no effect selection. The effects that are used in the refit depend on the results of the initial round of model averaging. If you do not specify any *refit-options*, then effects that are selected in at least twenty percent of the samples in the initial round of model averaging are used in the refit model average. The following *refit-options* are available:

BEST=*n*

specifies that the *n* most frequently selected effects in the initial round of model averaging be used in the refit averaging.

MINPCT=*percent*

specifies that the effects that are selected at least the specified percentage of times in the initial round of model averaging be used in the refit averaging.

NSAMPLES=*n*

specifies the number of samples to be used for the refit averaging. The default value is the number of samples used in the initial round of model averaging.

SAMPLING=SRS | URS < (*sampling-options***) >**

specifies how the samples of the usable observations in the training data are generated. SAMPLING=SRS specifies simple random sampling in which samples are generated by randomly drawing without replacement. SAMPLING=URS specifies unrestricted random sampling in which samples are generated by randomly drawing with replacement. Model averaging with samples drawn without replacement corresponds to the bootstrap methodology. The default is SAMPLING=URS. If you specify a frequency variable by using a FREQ statement, then the *i*th observation is sampled f_i times, where f_i is the frequency of the *i*th observation.

You can specify one of the following *sampling-options*:

PERCENT=*percent*

specifies the percentage of the training data that is used in each sample. The default value is 75% for SAMPLING=SRS and 100% for SAMPLING=URS.

SIZE=*n*

specifies the sum of frequencies in each sample.

SUBSET(*subset-options***)**

specifies that only a subset of the selected models be used in forming the average model and producing predicted values. The following *subset-options* are available:

BEST=*n*

specifies that only the best *n* models be used, where the model ranking criterion used is the frequency score. See the section "Model Selection Frequencies and Frequency Scores" on page 3867 for the definition of the frequency score. If multiple models with the same frequency score correspond to the *n*th best model, then all these tied models are used in forming the average model and producing predicted values.

MINMODELFREQ=*freq*

specifies that only models that are selected at least *freq* times be used in forming the average model and producing predicted values.

TABLES < (ONLY) > < =*table-request* **< (***options***) > >**

TABLES < (ONLY) > < = (*table-request* **< (***options***) > < ...** *table-request* **< (***options***) > >) >**

controls the displayed output that is produced in the initial round of model averaging. By default, the following tables are produced:

EFFECTSELECTPCT displays the percentage of times that effects appear in the selected models.

MODELSELECTFREQ displays the frequency with which models are selected.

AVGPARMEST displays the mean, standard deviation, and quantiles of the parameter estimates of the parameters that appear in the selected models.

When you specify only one *table-request*, you can omit the outer parentheses. Here are some examples:

```
tables=none
tables=(all parmest(minpct=10))
tables(only)=effectselectpct(order=model minpct=15)
```

The following *table-request* options are available:

ALL

requests that all model averaging output tables be produced. You can specify other options with ALL; for example, to request all tables and to require that effects are displayed in decreasing order of selection frequency in the EffectSelectPct table, specify TABLES=(ALL EFFECTSELECTPCT(ORDER=DESCENDING)).

EFFECTSELECTPCT < (*effectSelectPct-options***) >**

specifies how the effects in the EffectSelectPct table are displayed. The following *effectSelectPct-options* are available:

ALL

specifies that effects that appear in the selected model for any sample be displayed.

MINPCT=*percent*

specifies that the effects displayed must appear in the selected model for at least the specified percentage of the samples. By default, this table includes effects that appear in at least twenty percent of the selected models. The MINPCT= option is ignored if you also specify the ALL option as a *effectSelectPct-option*.

ORDER=ASCENDING | DESCENDING | MODEL

specifies the order in which the effects are displayed. ORDER=MODEL specifies that effects be displayed in the order in which they appear in the MODEL statement. ORDER=ASCENDING | DESCENDING specifies that the effects be displayed in ascending or descending order of their selection frequency.

MODELSELECTFREQ < (*modelSelectFreq-options***) >**

specifies how the models in the ModelSelectFreq table are displayed. The following *modelSelectFreq-options* are available:

ALL
specifies that all selected models be displayed in the ModelSelectFreq table.

BEST=*n*
specifies that only the best *n* models be displayed, where the model ranking criterion used is the frequency score. See the section “Model Selection Frequencies and Frequency Scores” on page 3867 for the definition of the frequency score. The default value is BEST=20. The BEST= option is ignored if you also specify the ALL option as a *modelSelectFreq-option*.

ONLY
suppresses the default output. If you specify the ONLY option within parentheses after the TABLES option, then only the tables specifically requested are produced.

PARMEST < (*parmEst-options*) >
specifies how the parameters in the AvgParmEst table are displayed. The following *parmEst-options* are available:

ALL
specifies that parameters that are nonzero in the selected model for any sample be displayed.

MINPCT=*percent*
specifies that the parameters displayed must have nonzero estimates in the selected model for at least the specified percentage of the samples. By default, this table includes parameters that appear in at least twenty percent of the selected models. The MINPCT= option is ignored if you also specify the ALL option as a *parmEst* option.

NONZEROPARMS
specifies that for each parameter, the sample that is used to compute the estimate mean, standard deviation, and quantiles consist of just the nonzero values of that parameter in the selected models. If you do not specify the NONZEROPARMS option, then parameters that do not appear in a selected model are assigned the value zero in that model and these zero values are retained when computing the estimate means, standard deviations, and quantiles.

ORDER=ASCENDING | DESCENDING | MODEL
specifies the order in which the effects are displayed. ORDER=MODEL specifies that effects are displayed in the order in which they appear in the MODEL statement. ORDER=ASCENDING | DESCENDING specifies that the effects are displayed in ascending or descending order of their selection frequency.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set*> < *keyword*< **=***name*> > . . . < *keyword*< **=***name*> > **;**

The OUTPUT statement creates a new SAS data set that saves diagnostic measures calculated for the selected model. If you do not specify a *keyword*, then the only diagnostic included is the predicted response.

All the variables in the original data set are included in the new data set, along with variables created in the OUTPUT statement. These new variables contain the values of a variety of statistics and diagnostic measures that are calculated for each observation in the data set. If you specify a BY statement, then a variable _BY_ that indexes the BY groups is included. For each observation, the value of _BY_ is the index of the BY group to which this observation belongs. This variable is useful for matching BY groups with macro variables that PROC GLMSELECT creates. See the section "Macro Variables Containing Selected Models" on page 3861 for details.

If you have requested *n*-fold cross validation by requesting CHOOSE=CV, SELECT=CV, or STOP=CV in the MODEL statement, then a variable _CVINDEX_ is included in the output data set. For each observation used for model training the value of _CVINDEX_ is *i* if that observation is omitted in forming the *i*th subset of the training data. See the CVMETHOD= for additional details. The value of _CVINDEX_ is 0 for all observations in the input data set that are not used for model training.

If you have partitioned the input data with a PARTITION statement, then a character variable _ROLE_ is included in the output data set. For each observation the value of _ROLE_ is as follows:

ROLE	Observation Role
TEST	testing
TRAIN	training
VALIDATE	validation

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*.

Details on the specifications in the OUTPUT statement follow.

keyword< **=***name* >

specifies the statistics to include in the output data set and optionally names the new variables that contain the statistics. Specify a *keyword* for each desired statistic (see the following list of *keywords*), followed optionally by an equal sign, and a variable to contain the statistic.

If you specify *keyword***=***name*, the new variable that contains the requested statistic has the specified name. If you omit the optional **=***name* after a *keyword*, then the new variable name is formed by using a prefix of one or more characters that identify the statistic. For residuals and predicted values, the prefix is followed by an underscore (_), followed by the dependent variable name.

The *keywords* allowed and the statistics they represent are as follows:

PREDICTED | PRED | P predicted values. The prefix for the default name is *p*.

RESIDUAL | RESID | R residual, calculated as ACTUAL – PREDICTED. The prefix for the default name is *r*.

When you also use the MODELAVERAGE statement, the following *keywords* and the statistics that they represent are also available:

LOWER the 100($\alpha/2$)th percentile of the sample predicted values. By default, $\alpha = 0.5$, which yields the 25th percentile. You can change the value of α by using the ALPHA= option in the MODELAVERAGE statement. The default name is *LOWER*.

MEDIAN median of the sample predicted values. The default name is *median*.

SAMPLEFREQ | SF sample frequencies. For the *i*th sample, a column that contains the frequencies used for that sample is added. The name of this column is formed by appending an index *i* to the name that you specify. If you do not specify a name, then the default prefix is sf.

SAMPLEPRED | SP sample predictions. For the *i*th sample, a column that contains the predicted values produced by the model selected for that sample is added. The name of this column is formed by appending an index *i* to the name that you specify. If you do not specify a name, then the default prefix is sp.

STANDARDDEVIATION | STDDEV standard deviation of the sample predicted values. The default name is *stdDev*.

UPPER the $100(1-\alpha/2)$th percentile of the sample predicted values. By default, $\alpha = 0.5$, which yields the 75th percentile. You can change the value of α by using the ALPHA= option in the MODELAVERAGE statement. The default name is *UPPER*.

OUT=*SAS data set*

specifies the name of the new data set. By default, the procedure uses the DATA*n* convention to name the new data set.

PARTITION Statement

The PARTITION statement specifies how observations in the input data set are logically partitioned into disjoint subsets for model training, validation, and testing. Either you can designate a variable in the input data set and a set of formatted values of that variable to determine the role of each observation, or you can specify proportions to use for random assignment of observations for each role.

An alternative to using a PARTITION statement is to provide a variable named _ROLE_ in the input data set to define roles of observations in the input data. If you specify a PARTITION statement then the _ROLE_ variable if present in the input data set is ignored. If you do not use a PARTITION statement and the input data do not contain a variable named _ROLE_, then all observations in the input data set are assigned to model training.

The following mutually exclusive *options* are available:

ROLEVAR | ROLE=*variable* **(< TEST=***'value'* **> < TRAIN=***'value'* **> < VALIDATE=***'value'* **>)**

names the variable in the input data set whose values are used to assign roles to each observation. The formatted values of this variable that are used to assign observations roles are specified in the TEST=, TRAIN=, and VALIDATE= suboptions. If you do not specify the TRAIN= suboption, then all observations whose role is not determined by the TEST= or VALIDATE= suboptions are assigned to training. If you specify a TESTDATA= data set in the PROC GLMSELECT statement, then you cannot also specify the TEST= suboption in the PARTITION statement. If you specify a VALDATA= data set in the PROC GLMSELECT statement, then you cannot also specify the VALIDATE= suboption in the PARTITION statement.

FRACTION(< TEST=*fraction* **> < VALIDATE=***fraction* **>)**

requests that specified proportions of the observations in the input data set be randomly assigned training and validation roles. You specify the proportions for testing and validation by using the TEST= and VALIDATE= suboptions. If you specify both the TEST= and the VALIDATE= suboptions, then the sum of the specified fractions must be less than one and the remaining fraction of the observations are assigned to the training role. If you specify a TESTDATA= data set in the PROC GLMSELECT statement, then you cannot also specify the TEST= suboption in the PARTITION statement. If you specify a VALDATA= data set in the PROC GLMSELECT statement, then you cannot also specify the VALIDATE= suboption in the PARTITION statement.

PERFORMANCE Statement

PERFORMANCE < *options* > **;**

The PERFORMANCE statement is used to change default options that affect the performance of PROC GLMSELECT and to request tables that show the performance options in effect and timing details.

The following *options* are available:

DETAILS
requests the PerfSettings table that shows the performance settings in effect and the Timing table that provides a broad timing breakdown of the PROC GLMSELECT step.

BUILDSSCP=FULL | INCREMENTAL
specifies whether the SSCP matrix is built incrementally as the selection process progresses or whether the SCCP matrix for the full model is built at the outset. Building the SSCP matrix incrementally can significantly reduce the memory required and the time taken to perform model selection in cases where the number of parameters in the selected model is much smaller than the number of parameters in the full model, but it can hurt performance in other cases since it requires at least one pass through the model training data at each step. If you use backward selection or no selection, or if the BIC or CP statistics are required in the selection process, then the BUILDSSCP=INCREMENTAL option is ignored. In other cases, BUILDSSCP=INCREMENTAL is used by default if the number of effects is greater than 100. See the section “Building the SSCP Matrix” on page 3865 for further details.

SCORE Statement

SCORE < **DATA=***SAS-data-set* > < **OUT=***SAS-data-set* > < *keyword*< **=***name* > > . . . < *keyword*< **=***name* > >
;

The SCORE statement creates a new SAS data set containing predicted values and optionally residuals for data in a new data set that you name. If you do not specify a DATA= data set, then the input data are scored. If you have multiple data sets to predict, you can specify multiple SCORE statements. 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*.

When a BY statement is used, the score data set must either contain all the BY variables sorted in the order of the BY variables or contain none of the BY variables. If the score data set contains all of the BY variables, then the model selected for a given BY group is used to score just the matching observations in the score data set. If the score data set contains none of the BY variables, then the entire score data set is scored for each BY group.

All observations in the score data set are retained in the output data set. However, only those observations that contain nonmissing values for all the continuous regressors in the selected model and whose levels of the classification variables appearing in effects of the selected model are represented in the corresponding classification variables in the procedure’s input data set are scored. All the variables in the input data set are included in the output data set, along with variables containing predicted values and optionally residuals.

Details on the specifications in the SCORE statement follow:

DATA=*SAS data set*

names the data set to be scored. If you omit this option, then the input data set named in the DATA= option in the PROC GLMSELECT statement is scored.

keyword< **=***name* >

specifies the statistics to include in the output data set and optionally names the new variables that contain the statistics. Specify a *keyword* for each desired statistic (see the following list of *keywords*), followed optionally by an equal sign, and a variable to contain the statistic.

If you specify *keyword*=*name*, the new variable that contains the requested statistic has the specified name. If you omit the optional **=***name* after a *keyword*, then the new variable name is formed by using a prefix of one or more characters that identify the statistic, followed by an underscore (_), followed by the dependent variable name.

The *keywords* allowed and the statistics they represent are as follows:

PREDICTED | PRED | P predicted values. The prefix for the default name is *p*.

RESIDUAL | RESID | R residual, calculated as ACTUAL – PREDICTED. The prefix for the default name is *r*.

OUT=*SAS data set*

gives the name of the new output data set. By default, the procedure uses the DATA*n* convention to name the new data set.

STORE Statement

STORE < **OUT=** >*item-store-name* < / **LABEL=**'*label*' > **;**

The STORE statement requests that the procedure save the context and results of the statistical analysis. The resulting item store has a binary file format that cannot be modified. The contents of the item store can be processed with the PLM procedure. For details about the syntax of the STORE statement, see the section "STORE Statement" on page 512 in Chapter 19, "Shared Concepts and Topics."

WEIGHT Statement

WEIGHT *variable* **;**

A WEIGHT statement names a variable in the input data set with values that are relative weights for a weighted least squares fit. If the weight value is proportional to the reciprocal of the variance for each observation, then the weighted estimates are the best linear unbiased estimates (BLUE).

Values of the weight variable must be nonnegative. If an observation's weight is zero, the observation is deleted from the analysis. If a weight is negative or missing, it is set to zero, and the observation is excluded from the analysis. A more complete description of the WEIGHT statement can be found in Chapter 46, "The GLM Procedure."

Details: GLMSELECT Procedure

Model-Selection Methods

The model selection methods implemented in PROC GLMSELECT are specified with the SELECTION= option in the MODEL statement.

Full Model Fitted (NONE)

The complete model specified in the MODEL statement is used to fit the model and no effect selection is done. You request this by specifying SELECTION=NONE in the MODEL statement.

Forward Selection (FORWARD)

The forward selection technique begins with just the intercept and then sequentially adds the effect that most improves the fit. The process terminates when no significant improvement can be obtained by adding any effect.

In the traditional implementation of forward selection, the statistic used to gauge improvement in fit is an F statistic that reflects an effect's contribution to the model if it is included. At each step, the effect that yields the most significant F statistic is added. Note that because effects can contribute different degrees of freedom to the model, it is necessary to compare the p-values corresponding to these F statistics.

More precisely, if the current model has p parameters excluding the intercept, and if you denote its residual sum of squares by RSS_p and you add an effect with k degrees of freedom and denote the residual sum of squares of the resulting model by RSS_{p+k}, then the F statistic for entry with k numerator degrees of freedom and $n-(p+k)-1$ denominator degrees of freedom is given by

$$F = \frac{(\text{RSS}_p - \text{RSS}_{p+k})/k}{\text{RSS}_{p+k}/(n-(p+k)-1)}$$

where n is number of observations used in the analysis.

The process stops when the significance level for adding any effect is greater than some specified entry significance level. A well-known problem with this methodology is that these F statistics do not follow an F distribution (Draper, Guttman, and Kanemasu 1971). Hence these p-values cannot reliably be interpreted as probabilities. Various ways to approximate this distribution are described by Miller (2002). Another issue when you use significance levels of entering effects as a stopping criterion arises because the entry significance level is an a priori specification that does not depend on the data. Thus, the same entry significance level can result in overfitting for some data and underfitting for other data.

One approach to address the critical problem of when to stop the selection process is to assess the quality of the models produced by the forward selection method and choose the model from this sequence that "best" balances goodness of fit against model complexity. PROC GLMSELECT supports several criteria that you can use for this purpose. These criteria fall into two groups—information criteria and criteria based on out-of-sample prediction performance.

You use the CHOOSE= option of forward selection to specify the criterion for selecting one model from the sequence of models produced. If you do not specify a CHOOSE= criterion, then the model at the final step is the selected model.

For example, if you specify

```
selection=forward(select=SL choose=AIC SLE=0.2)
```

then forward selection terminates at the step where no effect can be added at the 0.2 significance level. However, the selected model is the first one with the minimal value of Akaike's information criterion. Note that in some cases this minimal value might occur at a step much earlier that the final step, while in other cases the AIC criterion might start increasing only if more steps are done (that is, a larger value of SLE is used). If what you are interested in is minimizing AIC, then too many steps are done in the former case and too few in the latter case. To address this issue, PROC GLMSELECT enables you to specify a stopping criterion with the STOP= option. With a stopping criterion specified, forward selection continues until a local extremum of the stopping criterion in the sequence of models generated is reached. You can also specify STOP= number, which causes forward selection to continue until there are the specified number of effects in the model.

For example, if you specify

```
selection=forward(select=SL stop=AIC)
```

then forward selection terminates at the step where the effect to be added at the next step would produce a model with an AIC statistic larger than the AIC statistic of the current model. Note that in most cases, provided that the entry significance level is large enough that the local extremum of the named criterion occurs before the final step, specifying

```
selection=forward(select=SL choose=CRITERION)
```

or

```
selection=forward(select=SL stop=CRITERION)
```

selects the same model, but more steps are done in the former case. In some cases there might be a better local extremum that cannot be reached if you specify the STOP= option but can be found if you use the CHOOSE= option. Also, you can use the CHOOSE= option in preference to the STOP= option if you want examine how the named criterion behaves as you move beyond the step where the first local minimum of this criterion occurs.

Note that you can specify both the CHOOSE= and STOP= options. You might want to consider models generated by forward selection that have at most some fixed number of effects but select from within this set based on a criterion you specify. For example, specifying

```
selection=forward(stop=20 choose=ADJRSQ)
```

requests that forward selection continue until there are 20 effects in the final model and chooses among the sequence of models the one that has the largest value of the adjusted R-square statistic. You can also combine these options to select a model where one of two conditions is met. For example,

```
selection=forward(stop=AICC choose=PRESS)
```

chooses whatever occurs first between a local minimum of the predicted residual sum of squares (PRESS) and a local minimum of corrected Akaike's information criterion (AICC).

It is important to keep in mind that forward selection bases the decision about what effect to add at any step by considering models that differ by one effect from the current model. This search paradigm cannot guarantee reaching a "best" subset model. Furthermore, the add decision is greedy in the sense that the effect deemed most significant is the effect that is added. However, if your goal is to find a model that is best in terms of some selection criterion other than the significance level of the entering effect, then even this one step choice might not be optimal. For example, the effect you would add to get a model with the smallest value of the PRESS statistic at the next step is not necessarily the same effect that has the most significant entry F statistic. PROC GLMSELECT enables you to specify the criterion to optimize at each step by using the SELECT= option. For example,

```
selection=forward(select=CP)
```

requests that at each step the effect that is added be the one that gives a model with the smallest value of the Mallows' C_p statistic. Note that in the case where all effects are variables (that is, effects with one degree of freedom and no hierarchy), using ADJRSQ, AIC, AICC, BIC, CP, RSQUARE, or SBC as the selection criterion for forward selection produces the same sequence of additions. However, if the degrees of freedom contributed by different effects are not constant, or if an out-of-sample prediction-based criterion is used, then different sequences of additions might be obtained.

You can use SELECT= together with CHOOSE= and STOP=. If you specify only the SELECT= criterion, then this criterion is also used as the stopping criterion. In the previous example where only the selection criterion is specified, not only do effects enter based on the Mallows' C_p statistic, but the selection terminates when the $C(p)$ statistic first increases.

You can find discussion and references to studies about criteria for variable selection in Burnham and Anderson (2002), along with some cautions and recommendations.

Examples of Forward Selection Specifications

```
selection=forward
```

adds effects that at each step give the lowest value of the SBC statistic and stops at the step where adding any effect would increase the SBC statistic.

```
selection=forward(select=SL)
```

adds effects based on significance level and stops when all candidate effects for entry at a step have a significance level greater than the default entry significance level of 0.50.

```
selection=forward(select=SL stop=validation)
```

adds effects based on significance level and stops at a step where adding any effect increases the error sum of squares computed on the validation data.

```
selection=forward(select=AIC)
```

adds effects that at each step give the lowest value of the AIC statistic and stops at the step where adding any effect would increase the AIC statistic.

```
selection=forward(select=ADJRSQ stop=SL SLE=0.2)
```

adds effects that at each step give the largest value of the adjusted R-square statistic and stops at the step where the significance level corresponding to the addition of this effect is greater than 0.2.

Backward Elimination (BACKWARD)

The backward elimination technique starts from the full model including all independent effects. Then effects are deleted one by one until a stopping condition is satisfied. At each step, the effect showing the smallest contribution to the model is deleted. In traditional implementations of backward elimination, the contribution of an effect to the model is assessed by using an F statistic. At any step, the predictor producing the least significant F statistic is dropped and the process continues until all effects remaining in the model have F statistics significant at a stay significance level (SLS).

More precisely, if the current model has p parameters excluding the intercept, and if you denote its residual sum of squares by RSS_p and you drop an effect with k degrees of freedom and denote the residual sum of squares of the resulting model by RSS_{p-k}, then the F statistic for removal with k numerator degrees of freedom and $n - p - 1$ denominator degrees of freedom is given by

$$F = \frac{(\text{RSS}_{p-k} - \text{RSS}_p)/k}{\text{RSS}_p/(n - p - 1)}$$

where n is number of observations used in the analysis.

Just as with forward selection, you can change the criterion used to assess effect contributions with the SELECT= option. You can also specify a stopping criterion with the STOP= option and use a CHOOSE= option to provide a criterion used to select among the sequence of models produced. See the discussion in the section "Forward Selection (FORWARD)" on page 3846 for additional details.

Examples of Backward Selection Specifications

```
selection=backward
```

removes effects that at each step produce the largest value of the Schwarz Bayesian information criterion (SBC) statistic and stops at the step where removing any effect increases the SBC statistic.

```
selection=backward(stop=press)
```

removes effects based on the SBC statistic and stops at the step where removing any effect increases the predicted residual sum of squares (PRESS).

```
selection=backward(select=SL)
```

removes effects based on significance level and stops when all candidate effects for removal at a step have a significance level less than the default stay significance level of 0.10.

```
selection=backward(select=SL choose=validate SLS=0.1)
```

removes effects based on significance level and stops when all effects in the model are significant at the 0.1 level. Finally, from the sequence of models generated, choose the one that gives the smallest average square error when scored on the validation data.

Stepwise Selection(STEPWISE)

The stepwise method is a modification of the forward selection technique that differs in that effects already in the model do not necessarily stay there.

In the traditional implementation of stepwise selection method, the same entry and removal F statistics for the forward selection and backward elimination methods are used to assess contributions of effects as they are added to or removed from a model. If at a step of the stepwise method, any effect in the model is not significant at the SLSTAY= level, then the least significant of these effects is removed from the model and the algorithm proceeds to the next step. This ensures that no effect can be added to a model while some effect currently in the model is not deemed significant. Only after all necessary deletions have been accomplished can another effect be added to the model. In this case the effect whose addition yields the most significant F

value is added to the model and the algorithm proceeds to the next step. The stepwise process ends when none of the effects outside the model has an *F* statistic significant at the SLENTRY= level and every effect in the model is significant at the SLSTAY= level. In some cases, neither of these two conditions for stopping is met and the sequence of models cycles. In this case, the stepwise method terminates at the end of the second cycle.

Just as with forward selection and backward elimination, you can change the criterion used to assess effect contributions, with the SELECT= option. You can also specify a stopping criterion with the STOP= option and use a CHOOSE= option to provide a criterion used to select among the sequence of models produced. See the discussion in the section "Forward Selection (FORWARD)" on page 3846 for additional details.

For selection criteria other than significance level, PROC GLMSELECT optionally supports a further modification in the stepwise method. In the standard stepwise method, no effect can enter the model if removing any effect currently in the model would yield an improved value of the selection criterion. In the modification, you can use the DROP=COMPETITIVE option to specify that addition and deletion of effects should be treated competitively. The selection criterion is evaluated for all models obtained by deleting an effect from the current model or by adding an effect to this model. The action that most improves the selection criterion is the action taken.

Examples of Stepwise Selection Specifications

```
selection=stepwise
```

requests stepwise selection based on the SBC criterion. First, if removing any effect yields a model with a lower SBC statistic than the current model, then the effect producing the smallest SBC statistic is removed. When removing any effect increases the SBC statistic, then provided that adding some effect lowers the SBC statistic, the effect producing the model with the lowest SBC is added.

```
selection=stepwise(select=SL)
```

requests the traditional stepwise method. First, if the removal of any effect yields an *F* statistic that is not significant at the default stay level of 0.15, then the effect whose removal produces the least significant *F* statistic is removed and the algorithm proceeds to the next step. Otherwise the effect whose addition yields the most significant *F* statistic is added, provided that it is significant at the default entry level of 0.15.

```
selection=stepwise(select=SL stop=SBC)
```

is the traditional stepwise method, where effects enter and leave based on significance levels, but with the following extra check: If any effect to be added or removed yields a model whose SBC statistic is greater than the SBC statistic of the current model, then the stepwise method terminates at the current model. Note that in this case, the entry and stay significance levels still play a role as they determine whether an effect is deleted from or added to the model. This might result in the selection terminating before a local minimum of the SBC criterion is found.

```
selection=stepwise(select=SL SLE=0.1 SLS=0.08 choose=AIC)
```

selects effects to enter or drop as in the previous example except that the significance level for entry is now 0.1 and the significance level to stay is 0.08. From the sequence of models produced, the selected model is chosen to yield the minimum AIC statistic.

```
selection=stepwise(select=AICC drop=COMPETITIVE)
```

requests stepwise selection based on the AICC criterion with steps treated competitively. At any step, evaluate the AICC statistics corresponding to the removal of any effect in the current model or the addition of any effect to the current model. Choose the addition or removal that produced this minimum value, provided that this minimum is lower than the AICC statistic of the current model.

```
selection=stepwise(select=SBC drop=COMPETITIVE stop=VALIDATE)
```

requests stepwise selection based on the SBC criterion with steps treated competitively and where stopping is based on the average square error over the validation data. At any step, SBC statistics corresponding to the removal of any effect from the current model or the addition of any effect to the current model are evaluated. The addition or removal that produces the minimum SBC value is made. The average square error on the validation data for the model with this addition or removal is evaluated. If this average square error is greater than the average square error on the validation data prior to this addition or deletion, then the algorithm terminates at this prior model.

Least Angle Regression (LAR)

Least angle regression was introduced by Efron et al. (2004). Not only does this algorithm provide a selection method in its own right, but with one additional modification it can be used to efficiently produce LASSO solutions. Just like the forward selection method, the LAR algorithm produces a sequence of regression models where one parameter is added at each step, terminating at the full least squares solution when all parameters have entered the model.

The algorithm starts by centering the covariates and response, and scaling the covariates so that they all have the same corrected sum of squares. Initially all coefficients are zero, as is the predicted response. The predictor that is most correlated with the current residual is determined and a step is taken in the direction of this predictor. The length of this step determines the coefficient of this predictor and is chosen so that some other predictor and the current predicted response have the same correlation with the current residual. At this point, the predicted response moves in the direction that is equiangular between these two predictors. Moving in this direction ensures that these two predictors continue to have a common correlation with the current residual. The predicted response moves in this direction until a third predictor has the same correlation with the current residual as the two predictors already in the model. A new direction is determined that is equiangular between these three predictors and the predicted response moves in this direction until a fourth predictor joins the set having the same correlation with the current residual. This process continues until all predictors are in the model.

As with other selection methods, the issue of when to stop the selection process is crucial. You can specify a criterion to use to choose among the models at each step with the CHOOSE= option. You can also specify a stopping criterion with the STOP= option. See the section "Criteria Used in Model Selection Methods" on page 3857 for details and Table 49.10 for the formulas for evaluating these criteria. These formulas use the approximation that at step k of the LAR algorithm, the model has k degrees of freedom. See Efron et al. (2004) for a detailed discussion of this so-called simple approximation.

A modification of LAR selection suggested in Efron et al. (2004) uses the LAR algorithm to select the set of covariates in the model at any step, but uses ordinary least squares regression with just these covariates to obtain the regression coefficients. You can request this hybrid method by specifying the LSCOEFFS suboption of SELECTION=LAR.

Lasso Selection (LASSO)

LASSO (least absolute shrinkage and selection operator) selection arises from a constrained form of ordinary least squares regression where the sum of the absolute values of the regression coefficients is constrained to be smaller than a specified parameter. More precisely let $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m)$ denote the matrix of covariates and let $\mathbf{y}$ denote the response, where the $\mathbf{x}_i$s have been centered and scaled to have unit standard deviation and mean zero, and $\mathbf{y}$ has mean zero. Then for a given parameter t, the LASSO regression coefficients $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_m)$ are the solution to the constrained optimization problem

$$\min ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 \qquad \text{subject to} \quad \sum_{j=1}^{m} |\beta_j| \le t$$

Provided that the LASSO parameter t is small enough, some of the regression coefficients will be exactly zero. Hence, you can view the LASSO as selecting a subset of the regression coefficients for each LASSO parameter. By increasing the LASSO parameter in discrete steps, you obtain a sequence of regression coefficients where the nonzero coefficients at each step correspond to selected parameters.

Early implementations (Tibshirani 1996) of LASSO selection used quadratic programming techniques to solve the constrained least squares problem for each LASSO parameter of interest. Later Osborne, Presnell, and Turlach (2000) developed a "homotopy method" that generates the LASSO solutions for all values of t. Efron et al. (2004) derived a variant of their algorithm for least angle regression that can be used to obtain a sequence of LASSO solutions from which all other LASSO solutions can be obtained by linear interpolation. This algorithm for SELECTION=LASSO is used in PROC GLMSELECT. It can be viewed as a stepwise procedure with a single addition to or deletion from the set of nonzero regression coefficients at any step.

As with the other selection methods that PROC GLMSELECT supports, you can specify a criterion to choose among the models at each step of the LASSO algorithm by using the CHOOSE= option. You can also specify a stopping criterion by using the STOP= option. For more information, see the discussion in the section "Forward Selection (FORWARD)" on page 3846. The model degrees of freedom that PROC GLMSELECT uses at any step of the LASSO are simply the number of nonzero regression coefficients in the model at that step. Efron et al. (2004) cite empirical evidence for doing this but do not give any mathematical justification for this choice.

A modification of LASSO selection that is suggested in Efron et al. (2004) uses the LASSO algorithm to select the set of covariates in the model at any step, but it uses ordinary least squares regression with just these covariates to obtain the regression coefficients. You can request this hybrid method by specifying the LSCOEFFS suboption of the SELECTION=LASSO option.

Adaptive LASSO Selection

Adaptive LASSO selection is a modification of LASSO selection; in adaptive LASSO selection, weights are applied to each of the parameters in forming the LASSO constraint (Zou 2006). More precisely, suppose that the response $\mathbf{y}$ has mean zero and the regressors $\mathbf{x}$ are scaled to have mean zero and common standard deviation. Furthermore, suppose you can find a suitable estimator $\hat{\boldsymbol{\beta}}$ of the parameters in the true model and you define a weight vector by $w = 1/|\hat{\boldsymbol{\beta}}|^{\gamma}$, where $\gamma \ge 0$. Then the adaptive LASSO regression coefficients $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_m)$ are the solution to the constrained optimization problem

$$\min ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 \qquad \text{subject to} \quad \sum_{j=1}^{m} |w_j \beta_j| \leq t$$

You can specify $\hat{\boldsymbol{\beta}}$ by using the INEST=suboption of the SELECTION=LASSO option in the MODEL statement. The INEST= data set has the same structure as the OUTEST= data set that is produced by several SAS/STAT procedures, including the REG and LOGISTIC procedures. The INEST= data set must contain all explanatory variables in the MODEL statement. It must also contain an intercept variable named Intercept unless you specify the NOINT option in the MODEL statement. If BY processing is used, the INEST= data set must also include the BY variables, and there must be one observation for each BY group. If the INEST= data set also contains the _TYPE_ variable, only observations whose _TYPE_ value is `'PARMS'` are used.

If you do not specify an INEST= data set, then PROC GLMSELECT uses the solution to the unconstrained least squares problem as the estimator $\hat{\boldsymbol{\beta}}$. This is appropriate unless collinearity is a concern. If the regressors are collinear or nearly collinear, then Zou (2006) suggests using a ridge regression estimate to form the adaptive weights.

Elastic Net Selection (ELASTICNET)

The elastic net method bridges the LASSO method and ridge regression. It balances having a parsimonious model with borrowing strength from correlated regressors, by solving the least squares regression problem with constraints on both the sum of the absolute coefficients and the sum of the squared coefficients. More specifically, the elastic net coefficients $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_m)$ are the solution to the constrained optimization problem

$$\min ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 \qquad \text{subject to} \quad \sum_{j=1}^{m} |\beta_j| \leq t_1, \sum_{j=1}^{m} \beta_j^2 \leq t_2$$

The method can be written as the equivalent Lagrangian form

$$\min ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 + \lambda_1 \sum_{j=1}^{m} |\beta_j| + \lambda_2 \sum_{j=1}^{m} \beta_j^2$$

If t_1 is set to a very large value or, equivalently, if λ_1 is set to 0, then the elastic net method reduces to ridge regression. If t_2 is set to a very large value or, equivalently, if λ_2 is set to 0, then the elastic net method reduces to LASSO. If t_1 and t_2 are both large or, equivalently, if λ_1 and λ_2 are both set to 0, then the elastic net method reduces to ordinary least squares regression.

As stated by Zou and Hastie (2005), the elastic net method can overcome the limitations of LASSO in the following three scenarios:

- In the case where you have more parameters than observations, $m > n$, the LASSO method selects at most n variables before it saturates, because of the nature of the convex optimization problem. This can be a defect for a variable selection method. By contrast, the elastic net method can select more than n variables in this case because of the ridge regression regularization.
- If there is a group of variables that have high pairwise correlations, then whereas LASSO tends to select only one variable from that group, the elastic net method can select more than one variable.
- In the $n > m$ case, if there are high correlations between predictors, it has been empirically observed that the prediction performance of LASSO is dominated by ridge regression. In this case, the elastic net method can achieve better prediction performance by using ridge regression regularization.

An elastic net fit is achieved by building on LASSO estimation, in the following sense. Let $\tilde{\mathbf{X}}$ be a matrix obtained by augmenting $\mathbf{X}$ with a scaled identity matrix,

$$\tilde{\mathbf{X}} = [\mathbf{X}; \sqrt{\lambda_2} I]$$

Let $\tilde{\mathbf{y}}$ be a vector correspondingly obtained by augmenting the response $\mathbf{y}$ with m 0's,

$$\tilde{\mathbf{y}} = [\mathbf{y}; \mathbf{0}]$$

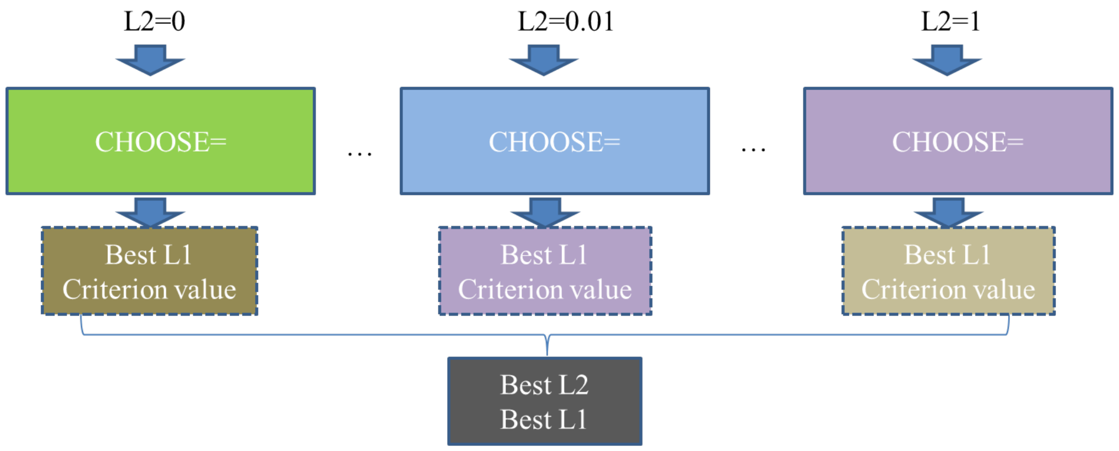
Then the Lagrangian form of the elastic net optimization problem can be reformulated as

$$\min ||\tilde{\mathbf{y}} - \tilde{\mathbf{X}}\boldsymbol{\beta}||^2 + \lambda_1 \sum_{j=1}^{m} |\beta_j|$$

In other words, you can solve the elastic net method in the same way as LASSO by using this augmented design matrix $\tilde{\mathbf{X}}$ and response $\tilde{\mathbf{y}}$. Therefore, for given λ_2, the coefficients of the elastic net fit follow the same piecewise linear path as LASSO. Zou and Hastie (2005) suggest rescaling the coefficients by $1 + \lambda_2$ to deal with the double amount of shrinkage in the elastic net fit, and such rescaling is applied when you specify the ENSCALE option in the MODEL statement.

If you have a good estimate of λ_2, you can specify the value in the L2= option. If you do not specify a value for λ_2, then by default PROC GLMSELECT searches for a value between 0 and 1 that is optimal according to the current CHOOSE= criterion. Figure 49.12 illustrates the estimation of the ridge regression parameter λ_2 (L2). Meanwhile, if you do not specify the CHOOSE= option, then the model at the final step in the selection process is selected for each λ_2 (L2), and the criterion value shown in Figure 49.12 is the one at the final step that corresponds to the specified STOP= option (STOP=SBC by default).

Figure 49.12 Estimation of the Ridge Regression Parameter λ_2 (L2) in the Elastic Net Method



Note that when you specify the L2SEARCH=GOLDEN, it is assumed that the criterion curve that corresponds to the CHOOSE= option with respect to λ_2 is a smooth and bowl-shaped curve. However, this assumption is not checked and validated. Hence, the default value for the L2SEARCH= option is set to GRID.

Group LASSO Selection (GROUPLASSO)

The group LASSO method proposed by Yuan and Lin (2006) is a variant of LASSO that is specifically designed for linear models defined in terms of effects that have multiple degrees of freedom, such as the main effects of CLASS variables, interactions between CLASS variables, and effects defined using an EFFECT statement.

Recall that LASSO selection depends on solving a constrained least squares problem of the form

$$\min ||\mathbf{y}-\mathbf{X}\boldsymbol{\beta}||^2 \qquad \text{subject to} \quad \sum_{j=1}^{m} |\beta_j| \le t$$

In this formulation, you can include or exclude individual parameters from the model independently, subject only to the overall constraint. In contrast, the group LASSO method uses a constraint that forces all parameters that correspond to the same effect to be included or excluded simultaneously. For a model that has k effects, let β_{G_j} be the group of linear coefficients that correspond to effect j in the model. Then group LASSO depends on solving a constrained optimization problem of the form

$$\min ||\mathbf{y}-\mathbf{X}\boldsymbol{\beta}||^2 \qquad \text{subject to} \quad \sum_{j=1}^{k} \sqrt{|G_j|}||\beta_{G_j}|| \le t$$

where $|G_j|$ is the number of parameters that correspond to effect j, and $||\beta_{G_j}||$ denotes the Euclidean norm of the parameters β_{G_j}. That is, instead of constraining the sum of the absolute value of individual parameters, group LASSO constrains the Euclidean norm of groups of parameters, where groups are defined by effects.

You can write the group LASSO method in the equivalent Lagrangian form

$$\min ||\mathbf{y}-\mathbf{X}\boldsymbol{\beta}||^2 + \lambda \sum_{j=1}^{k} \sqrt{|G_j|}||\beta_{G_j}||$$

The weight $\sqrt{|G_j|}$, as suggested by Yuan and Lin (2006), should take the size of the group into consideration in group LASSO.

Unlike LASSO, group LASSO does not allow a piecewise linear constant solution path as generated by a LAR algorithm. Instead, the method that Nesterov (2013) proposes is adopted to solve the Lagrangian form of the group LASSO problem that corresponds to a prespecified regularization parameter, λ. Nesterov's method is known to have an optimal convergence rate for first-order black box optimization. Because the optimal λ is usually unknown, a sequence of regularization parameters, $\rho, \rho^2, \rho^3, \ldots,$ is employed, where ρ is a positive value less than 1. You can specify ρ by using the RHO= suboption of the SELECTION= option in the MODEL statement; by default, RHO=0.9. In the ith step of group LASSO selection, the value used for λ is ρ^i. If you want the solution that corresponds to a prespecified λ, you can specify the value of λ by using the L1= option together with STOP=L1.

Another unique feature of the group LASSO method is that it does not necessarily add or remove precisely one effect at each step of the process. This is different from the forward, stepwise, backward, LAR, LASSO, and elastic net selection methods.

Model Selection Issues

Many authors caution against the use of "automatic variable selection" methods and describe pitfalls that plague many such methods. For example, Harrell (2001) states that "stepwise variable selection has been a very popular technique for many years, but if this procedure had just been proposed as a statistical method, it would most likely be rejected because it violates every principle of statistical estimation and hypothesis testing." He lists and discusses several of these issues and cites a variety of studies that highlight these problems. He also notes that many of these issues are not restricted to stepwise selection but affect forward selection and backward elimination, as well as methods based on all-subset selection.

In their introductory chapter, Burnham and Anderson (2002) discuss many issues involved in model selection. They also strongly warn against "data dredging," which they describe as "the process of analyzing data with few or no a priori questions, by subjectively and iteratively searching the data for patterns and 'significance.'" However, Burnham and Anderson also discuss the desirability of finding parsimonious models. They note that using "full models" that contain many insignificant predictors might avoid some of the inferential problems arising in models with automatically selected variables but will lead to overfitting the particular sample data and produce a model that performs poorly in predicting data not used in training the model.

One problem in the traditional implementations of forward, backward, and stepwise selection methods is that they are based on sequential testing with specified entry (SLE) and stay (SLS) significance levels. However, it is known that the "F-to-enter" and "F-to-delete" statistics do not follow an F distribution (Draper, Guttman, and Kanemasu 1971). Hence the SLE and SLS values cannot reliably be viewed as probabilities. One way to address this difficulty is to replace hypothesis testing as a means of selecting a model with information criteria or out-of-sample prediction criteria. While Harrell (2001) points out that information criteria were developed for comparing only prespecified models, Burnham and Anderson (2002) note that AIC criteria have routinely been used for several decades for performing model selection in time series analysis.

Problems also arise when the selected model is interpreted as if it were prespecified. There is a "selection bias" in the parameter estimates that is discussed in detail in Miller (2002). This bias occurs because a parameter is more likely to be selected if it is above its expected value than if it is below its expected value. Furthermore, because multiple comparisons are made in obtaining the selected model, the *p*-values obtained for the selected model are not valid. When a single best model is selected, inference is conditional on that model.

Model averaging approaches provide a way to make more stable inferences based on a set of models. PROC GLMSELECT provides support for model averaging by averaging models that are selected on resampled data. Other approaches for performing model averaging are presented in Burnham and Anderson (2002), and Bayesian approaches are discussed in Raftery, Madigan, and Hoeting (1997).

Despite these difficulties, careful and informed use of variable selection methods still has its place in modern data analysis. For example, Foster and Stine (2004) use a modified version of stepwise selection to build a predictive model for bankruptcy from over 67,000 possible predictors and show that this yields a model whose predictions compare favorably with other recently developed data mining tools. In particular, when the goal is prediction rather than estimation or hypothesis testing, variable selection with careful use of validation to limit both under and over fitting is often a useful starting point of model development.

Criteria Used in Model Selection Methods

PROC GLMSELECT supports a variety of fit statistics that you can specify as criteria for the CHOOSE=, SELECT=, and STOP= options in the MODEL statement. The following statistics are available:

ADJRSQ	adjusted R-square statistic (Darlington 1968; Judge et al. 1985)
AIC	Akaike's information criterion (Darlington 1968; Judge et al. 1985)
AICC	corrected Akaike's information criterion (Hurvich and Tsai 1989)
BIC	Sawa Bayesian information criterion (Sawa 1978; Judge et al. 1985)
CP	Mallows' C_p statistic (Mallows 1973; Hocking 1976)
PRESS	predicted residual sum of squares statistic
SBC	Schwarz Bayesian information criterion (Schwarz 1978; Judge et al. 1985)
SL	significance level of the *F* statistic used to assess an effect's contribution to the fit when it is added to or removed from a model
VALIDATE	average square error over the validation data

Table 49.10 provides formulas and definitions for the fit statistics.

Table 49.10 Formulas and Definitions for Model Fit Summary Statistics

Statistic	Definition or Formula
n	Number of observations
p	Number of parameters including the intercept
$\hat{\sigma}^2$	Estimate of pure error variance from fitting the full model
SST	Total sum of squares corrected for the mean for the dependent variable
SSE	Error sum of squares
ASE	$\frac{\text{SSE}}{n}$
MSE	$\frac{\text{SSE}}{n-p}$
R^2	$1 - \frac{\text{SSE}}{\text{SST}}$
ADJRSQ	$1 - \frac{(n-1)(1-R^2)}{n-p}$
AIC	$n \log\left(\frac{\text{SSE}}{n}\right) + 2p + n + 2$
AICC	$n \log\left(\frac{\text{SSE}}{n}\right) + \frac{n(n+p)}{n-p-2}$
BIC	$n \log\left(\frac{\text{SSE}}{n}\right) + 2(p+2)q - 2q^2$ where $q = \frac{n\hat{\sigma}^2}{\text{SSE}}$
CP (C_p)	$\frac{\text{SSE}}{\hat{\sigma}^2} + 2p - n$
PRESS	$\sum_{i=1}^{n} \frac{r_i^2}{(1-h_i)^2}$ where r_i = residual at observation i and h_i = leverage of observation i = $\mathbf{x}_i(\mathbf{X}'\mathbf{X})^{-}\mathbf{x}_i'$
RMSE	$\sqrt{\text{MSE}}$
SBC	$n \log\left(\frac{\text{SSE}}{n}\right) + p \log(n)$

Formulas for AIC and AICC

There is some inconsistency in the literature on the precise definitions for the AIC and AICC statistics. The definitions used in PROC GLMSELECT changed between the experimental and the production release of the procedure in SAS 9.2. The definitions now used in PROC GLMSELECT yield the same final models as before, but PROC GLMSELECT makes the connection between the AIC statistic and the AICC statistic more transparent.

In the context of linear regression, several different versions of the formulas for AIC and AICC appear in the statistics literature. However, for a fixed number of observations, these different versions differ by additive and positive multiplicative constants. Because the model selected to yield a minimum of a criterion is not affected if the criterion is changed by additive and positive multiplicative constants, these changes in the formula for AIC and AICC do not affect the selection process.

The following section provides details about these changes. Formulas used in the experimental download release are denoted with a superscript of (d) and n, p and SSE are defined in Table 49.10.

The experimental download release of PROC GLMSELECT used the following formulas for AIC (Darlington 1968; Judge et al. 1985) and AICC (Hurvich, Simonoff, and Tsai 1998):

$$\text{AIC}^{(d)} = n\log\left(\frac{\text{SSE}}{n}\right) + 2p$$

and

$$\text{AICC}^{(d)} = \log\left(\frac{\text{SSE}}{n}\right) + 1 + \frac{2(p+1)}{n-p-2}$$

PROC GLMSLECT now uses the definitions of AIC and AICC found in Hurvich and Tsai (1989):

$$\text{AIC} = n\log\left(\frac{\text{SSE}}{n}\right) + 2p + n + 2$$

and

$$\text{AICC} = \text{AIC} + \frac{2(p+1)(p+2)}{n-p-2}$$

Hurvich and Tsai (1989) show that the formula for AICC can also be written as

$$\text{AICC} = n\log\left(\frac{\text{SSE}}{n}\right) + \frac{n(n+p)}{n-p-2}$$

The relationships between the alternative forms of the formulas are

$$\text{AIC} = \text{AIC}^{(d)} + n + 2$$

$$\text{AICC} = n\ \text{AICC}^{(d)}$$

CLASS Variable Parameterization and the SPLIT Option

The GLMSELECT procedure supports nonsingular parameterizations for classification effects. A variety of these nonsingular parameterizations are available. You use the PARAM= option in the CLASS statement to specify the parameterization. See the section "Other Parameterizations" on page 395 in Chapter 19, "Shared Concepts and Topics," for details.

PROC GLMSELECT also supports the ability to split classification effects. You can use the SPLIT option in the CLASS statement to request that the columns of the design matrix that correspond to any effect that contains a split classification variable can be selected to enter or leave a model independently of the other design columns of that effect. The following statements illustrate the use of SPLIT option together with other features of the CLASS statement:

```
data codingExample;
   drop i;
   do i=1 to 1000;
      c1 = 1 + mod(i,6);
      if      i < 50  then c2 = 'very low ';
      else if i < 250 then c2 = 'low';
      else if i < 500 then c2 = 'medium';
      else if i < 800 then c2 = 'high';
      else                 c2 = 'very high';
      x1 = ranuni(1);
      x2 = ranuni(1);
      y = x1 + 10*(c1=3) +5*(c1=5) +rannor(1);
      output;
   end;
run;
proc glmselect data=codingExample;
   class c1(param=ref split) c2(param=ordinal order=data) /
          delimiter = ',' showcoding;
   model y = c1 c2 x1 x2/orderselect;
run;
```

The “Class Level Information” table shown in Figure 49.13 is produced by default whenever you specify a CLASS statement.

Figure 49.13 Class Level Information

The GLMSELECT Procedure

Class Level Information

Class	Levels		Values
c1	6	*	1,2,3,4,5,6
c2	5		very low,low,medium,high,very high

*** Associated Parameters Split**

Note that because the levels of the variable c2 contain embedded blanks, the DELIMITER=',' option has been specified. The SHOWCODING option requests the display of the “Class Level Coding” table shown in Figure 49.14. An ordinal parameterization is used for c2 because its levels have a natural order. Furthermore, because these levels appear in their natural order in the data, you can preserve this order by specifying the ORDER=DATA option.

Figure 49.14 Class Level Coding

Class Level Coding

c1 Level	Design Variables 1	2	3	4	5
1	1	0	0	0	0
2	0	1	0	0	0
3	0	0	1	0	0
4	0	0	0	1	0
5	0	0	0	0	1
6	0	0	0	0	0

The SPLIT option has been specified for the classification variable c1. This permits the parameters associated with the effect c1 to enter or leave the model individually. The "Parameter Estimates" table in Figure 49.15 shows that for this example the parameters that correspond to only levels 3 and 5 of c1 are in the selected model. Finally, note that the ORDERSELECT option in the MODEL statement specifies that the parameters are displayed in the order in which they first entered the model.

Figure 49.15 Parameter Estimates

Parameter Estimates

Parameter	DF	Estimate	Standard Error	t Value
Intercept	1	-0.216680	0.068650	-3.16
c1_3	1	10.160900	0.087898	115.60
c1_5	1	5.018015	0.087885	57.10
x1	1	1.315468	0.109772	11.98

Macro Variables Containing Selected Models

Often you might want to perform postselection analysis by using other SAS procedures. To facilitate this, PROC GLMSELECT saves the list of selected effects in a macro variable. This list does not explicitly include the intercept so that you can use it in the MODEL statement of other SAS/STAT regression procedures.

The following table describes the macro variables that PROC GLMSELECT creates. Note that when BY processing is used, one macro variable, indexed by the BY group number, is created for each BY group.

Macro Variable	Description
No BY processing	
_GLSIND1	Selected model
BY processing	
_GLSNUMBYS	Number of BY groups
_GLSIND1	Selected model for BY group 1
_GLSIND2	Selected model for BY group 2
. . .	

You can use the macro variable _GLSIND as a synonym for _GLSIND1. If you do not use BY processing, _GLSNUMBYS is still defined and has the value 1.

To aid in associating indexed macro variables with the appropriate observations when BY processing is used, PROC GLMSELECT creates a variable _BY_ in the output data set specified in an OUTPUT statement (see the section "OUTPUT Statement" on page 3841) that tags observations with an index that matches the index of the appropriate macro variable.

The following statements create a data set with two BY groups and run PROC GLMSELECT to select a model for each BY group.

```
data one(drop=i j);
   array x{5} x1-x5;
   do i=1 to 1000;
      classVar = mod(i,4)+1;
      do j=1 to 5;
         x{j} = ranuni(1);
      end;
      if i<400 then do;
         byVar = 'group 1';
         y     = 3*classVar+7*x2+5*x2*x5+rannor(1);
      end;
      else do;
         byVar = 'group 2';
         y     = 2*classVar+x5+rannor(1);
      end;
      output;
   end;
run;

proc glmselect data=one;
   by     byVar;
   class  classVar;
   model  y = classVar x1|x2|x3|x4|x5 @2 /
                  selection=stepwise(stop=aicc);
   output out=glmselectOutput;
run;
```

The preceding PROC GLMSELECT step produces three macro variables:

Macro Variable	Value	Description
_GLSNUMBYS	2	Number of BY groups
_GLSIND1	classVar x2 x2*x5	Selected model for the first BY group
_GLSIND2	classVar x5	Selected model for the second BY group

You can now leverage these macro variables and the output data set created by PROC GLMSELECT to perform postselection analyses that match the selected models with the appropriate BY-group observations. For example, the following statements create and run a macro that uses PROC GLM to perform LSMeans analyses.

```
%macro LSMeansAnalysis;
   %do i=1 %to &_GLSNUMBYS;
      title1  "Analysis Using the Selected Model for BY group number &i";
      title2 "Selected Effects: &&_GLSIND&i";

      ods select LSMeans;
      proc glm data=glmselectOutput(where = (_BY_ = &i));
         class classVar;
         model y = &&_GLSIND&i;
         lsmeans classVar;
      run;quit;
   %end;
%mend;
%LSMeansAnalysis;
```

The LSMeans analysis output from PROC GLM is shown in Output 49.16.

Figure 49.16 LS-Means Analyses for Selected Models

Analysis Using the Selected Model for BY group number 1
Selected Effects: classVar x2 x2*x5

The GLM Procedure
Least Squares Means

classVar	y LSMEAN
1	7.8832052
2	10.9528618
3	13.9412216
4	16.7929355

Figure 49.16 *continued*

Analysis Using the Selected Model for BY group number 2
Selected Effects: classVar x5

The GLM Procedure
Least Squares Means

classVar	y LSMEAN
1	2.46805014
2	4.52102826
3	6.53369479
4	8.49354763

Using the STORE Statement

The preceding section shows how you can use macro variables to facilitate performing postselection analysis by using other SAS procedures. An alternative approach is to use the STORE statement to save the results of the PROC GLMSELECT step in an *item store*. You can then use the PLM procedure to obtain a rich set of postselection analyses. The following statements show how you can use this approach to obtain the same LSMeans analyses as shown in section "Macro Variables Containing Selected Models" on page 3861:

```
proc glmselect data=one;
   by     byVar;
   class  classVar;
   model  y = classVar x1|x2|x3|x4|x5 @2 /
                  selection=stepwise(stop=aicc);
   store out=glmselectStore;
run;

proc plm source=glmselectStore;
   lsmeans classVar;
run;
```

The LSMeans analysis output for the first BY group is shown in Figure 49.17.

Figure 49.17 LS-Means Analysis Produced by PROC PLM

The PLM Procedure

classVar Least Squares Means

classVar	Estimate	Standard Error	DF	t Value	Pr > |t|
1	7.8832	0.1050	393	75.11	<.0001
2	10.9529	0.1043	393	104.99	<.0001
3	13.9412	0.1043	393	133.70	<.0001
4	16.7929	0.1042	393	161.09	<.0001

Building the SSCP Matrix

Traditional implementations of FORWARD and STEPWISE selection methods start by computing the augmented crossproduct matrix for all the specified effects. This initial crossproduct matrix is updated as effects enter or leave the current model by sweeping the columns corresponding to the parameters of the entering or departing effects. Building the starting crossproduct matrix can be done with a single pass through the data and requires $O(m^2)$ storage and $O(nm^2)$ work, where n is the number of observations and m is the number of parameters. If k selection steps are done, then the total work sweeping effects in and out of the model is $O(km^2)$. When $n >> m$, the work required is dominated by the time spent forming the crossproduct matrix. However, when m is large (tens of thousands), just storing the crossproduct matrix becomes intractable even though the number of selected parameters might be small. Note also that when interactions of classification effects are considered, the number of parameters considered can be large, even though the number of effects considered is much smaller.

When the number of selected parameters is smaller than the total number of parameters, it turns out that many of the crossproducts are not needed in the selection process. Let $\mathbf{y}$ denote the dependent variable, and suppose at some step of the selection process that $\mathbf{X}$ denotes the $n \times p$ design matrix columns corresponding to the currently selected model. Let $\mathbf{Z} = \mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_{m-p}$ denote the design matrix columns corresponding to the $m - p$ effects not yet in the model. Then in order to compute the reduction in the residual sum of squares when $\mathbf{z}_j$ is added to the model, the only additional crossproducts needed are $\mathbf{z}'_j y$, $\mathbf{z}'_j\mathbf{X}$, and $\mathbf{z}'_j\mathbf{z}_j$. Note that it is not necessary to compute any of $\mathbf{z}'_j\mathbf{z}_i$ with $i \neq j$ and if $p << m$, and this yields a substantial saving in both memory required and computational work. Note, however, that this strategy does require a pass through the data at any step where adding an effect to the model is considered.

PROC GLMSELECT supports both of these strategies for building the crossproduct matrix. You can choose which of these strategies to use by specifying the BUILDSSCP=FULL or BUILDSSCP=INCREMENTAL option in the PERFORMANCE statement. If you request BACKWARD selection, then the full SSCP matrix is required. Similarly, if you request the BIC or CP criterion as the SELECT=, CHOOSE=, or STOP= criterion, or if you request the display of one or both of these criteria with the STATS=BIC, STATS=CP, or STATS=ALL option, then the full model needs to be computed. If you do not specify the BUILDSSCP= option, then PROC GLMSELECT switches to the incremental strategy if the number of effects is greater than one hundred. This default strategy is designed to give good performance when the number of selected parameters is less than about 20% of the total number of parameters. Hence if you choose options that you know will cause the selected model to contain a significantly higher percentage of the total number of candidate parameters, then you should consider specifying BUILDSSCP=FULL. Conversely, if you specify fewer than 100 effects in the MODEL statement but many of these effects have a large number of associated parameters, then specifying BUILDSSCP=INCREMENTAL might result in improved performance.

Model Averaging

As discussed in the section "Model Selection Issues" on page 3856, some well-known issues arise in performing model selection for inference and prediction. One approach to address these issues is to use resampled data as a proxy for multiple samples that are drawn from some conceptual probability distribution. A model is selected for each resampled set of data, and a predictive model is built by averaging the predictions of these selected models. You can perform this method of model averaging by using the MODELAVERAGE statement. Resampling-based methods, in which samples are obtained by drawing with replacement from your data, fall under the umbrella of the widely studied methodology known as the bootstrap (Efron and Tibshirani 1993). For use of the bootstrap in the context of variable selection, see Breiman (1992).

By default, when the average is formed, models that are selected in multiple samples receive more weight than infrequently selected models. Alternatively, you can start by fitting a prespecified set of models on your data, then use information-theoretic approaches to assign a weight to each model in building a weighted average model. You can find a detailed discussion of this methodology in Burnham and Anderson (2002), in addition to some comparisons of this approach with bootstrap-based methods.

In the linear model context, the average prediction that you obtain from a set of models is the same as the prediction that you obtain with the single model whose parameter estimates are the averages of the corresponding estimates of the set of models. Hence, you can regard model averaging as a selection method that selects this average model. To show this, denote by $\beta^{(i)}$ the parameter estimates for the sample i where $\beta_j^{(i)} = 0$ if parameter j is not in the selected model for sample i. Then the predicted values $\hat{y}^{(i)}$ for average model i are given by

$$\hat{y}^{(i)} = \mathbf{X}\boldsymbol{\beta}^{(i)}$$

where $\mathbf{X}$ is the design matrix of the data to be scored. Forming averages gives

$$\hat{y}^{(*)} = \frac{1}{N}\sum_{i=1}^{N} \hat{y}^{(i)} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{X}\boldsymbol{\beta}^{(i)} = \mathbf{X}\left(\frac{1}{N}\sum_{i=1}^{N} \beta^{(i)}\right) = \mathbf{X}\beta^{(*)}$$

where for parameter j, $\beta_j^{(*)} = \frac{1}{N}\sum_{i=1}^{N} \beta_j^{(i)}$.

You can see that if a parameter estimate is nonzero for just a few of the sample models, then averaging the estimates for this parameter shrinks this estimate towards zero. It is this shrinkage that ameliorates the bias that a parameter is more likely to be selected if it is above its expected value rather than below it. This reduction in bias often produces improved predictions on new data that you obtain with the average model. However, the average model is not parsimonious since it has nonzero estimates for any parameter that is selected in any sample.

One resampling-based approach for obtaining a parsimonious model is to use the number of times that regressors are selected as an indication of importance and then to fit a new model that uses just the regressors that you deem to be most important. This approach is not without risk. One possible problem is that you might have several regressors that, for purposes of prediction, can be used as surrogates for one another. In this case it is possible that none of these regressors individually appears in a large enough percentage of the sample models to be deemed important, even though every model contains at least one of them. Despite such potential problems, this strategy is often successful. You can implement this approach by using the REFIT option in the MODELAVERAGE statement. By default, the REFIT option performs a second round of model averaging, where a fixed model that consists of the effects that are selected in a least twenty percent of the samples in the initial round of model averaging is used. The average model is obtained by averaging the ordinary least squares estimates obtained for each sample in the refit. Note that the default selection frequency cutoff of twenty percent is merely a heuristic guideline that often produces reasonable models.

Another approach to obtaining a parsimonious average model is to form the average of just the frequently selected models. You can implement this strategy by using the SUBSET option in the MODELAVERAGE statement. However, in situations where there are many irrelevant regressors, it is often the case that most of the selected models are selected just once. In such situations, having a way to order the models that are selected with the same frequency is desirable. The following section discusses a way to do this.

Model Selection Frequencies and Frequency Scores

The model frequency score orders models by their selection frequency, but also uses effect selection frequencies to order different models that are selected with the same frequency. Let $\mathbf{x}_i$ denote the ith effect and let f_i denote the selection fraction for this effect. f_i is computed as the number of samples whose selected model contains effect $\mathbf{x}_i$ divided by the number of samples. Suppose the jth model that consists of the K effects $\mathbf{x}_{j_1}, \mathbf{x}_{j_2}, \ldots, \mathbf{x}_{j_K}$ is selected m_j times. Then the model *frequency score*, s_j, for this model is computed as the sum of the model selection frequency and the average selection fraction for this model; that is,

$$s_j = m_j + \frac{\sum_{k=1}^{K} f_{j_k}}{K}$$

When you use the BEST=b suboption of the SUBSET option in the MODELAVERAGE statement, then the average model is formed from the b models with the largest model frequency scores.

Using Validation and Test Data

When you have sufficient data, you can subdivide your data into three parts called the training, validation, and test data. During the selection process, models are fit on the training data, and the prediction error for the models so obtained is found by using the validation data. This prediction error on the validation data can be used to decide when to terminate the selection process or to decide what effects to include as the selection process proceeds. Finally, once a selected model has been obtained, the test set can be used to assess how the selected model generalizes on data that played no role in selecting the model.

In some cases you might want to use only training and test data. For example, you might decide to use an information criterion to decide what effects to include and when to terminate the selection process. In this case no validation data are required, but test data can still be useful in assessing the predictive performance of the selected model. In other cases you might decide to use validation data during the selection process but forgo assessing the selected model on test data. Hastie, Tibshirani, and Friedman (2001) note that it is difficult to give a general rule on how many observations you should assign to each role. They note that a typical split might be 50% for training and 25% each for validation and testing.

PROC GLMSELECT provides several methods for partitioning data into training, validation, and test data. You can provide data for each role in separate data sets that you specify with the DATA=, TESTDATA=, and VALDATA= options in the PROC GLMSELECT procedure. An alternative method is to use a PARTITION statement to logically subdivide the DATA= data set into separate roles. You can name the fractions of the data that you want to reserve as test data and validation data. For example, specifying

```
proc glmselect data=inData;
   partition fraction(test=0.25 validate=0.25);
   ...
run;
```

randomly subdivides the inData data set, reserving 50% for training and 25% each for validation and testing.

In some cases you might need to exercise more control over the partitioning of the input data set. You can do this by naming a variable in the input data set as well as a formatted value of that variable that correspond to each role. For example, specifying

```
proc glmselect data=inData;
   partition roleVar=group(test='group 1' train='group 2')
   ...
run;
```

assigns all roles observations in the inData data set based on the value of the variable named group in that data set. Observations where the value of group is ’group 1’ are assigned for testing, and those with value ’group 2’ are assigned to training. All other observations are ignored.

You can also combine the use of the PARTITION statement with named data sets for specifying data roles. For example,

```
proc glmselect data=inData testData=inTest;
   partition fraction(validate=0.4);
   ...
run;
```

reserves 40% of the inData data set for validation and uses the remaining 60% for training. Data for testing is supplied in the inTest data set. Note that in this case, because you have supplied a TESTDATA= data set, you cannot reserve additional observations for testing with the PARTITION statement.

When you use a PARTITION statement, the output data set created with an OUTPUT statement contains a character variable _ROLE_ whose values `'TRAIN'`, `'TEST'`, and `'VALIDATE'` indicate the role of each observation. _ROLE_ is blank for observations that were not assigned to any of these three roles. When the input data set specified in the DATA= option in the PROC GLMSELECT statement contains an _ROLE_ variable and no PARTITION statement is used, and TESTDATA= and VALDATA= are not specified, then the _ROLE_ variable is used to define the roles of each observation. This is useful when you want to rerun PROC GLMSELECT but use the same data partitioning as in a previous PROC GLMSELECT step. For example, the following statements use the same data for testing and training in both PROC GLMSELECT steps:

```
proc glmselect data=inData;
   partition fraction(test=0.5);
   model y=x1-x10/selection=forward;
   output out=outDataForward;
run;

proc glmselect data=outDataForward;
   model y=x1-x10/selection=backward;
run;
```

When you have reserved observations for training, validation, and testing, a model fit on the training data is scored on the validation and test data, and the average squared error, denoted by ASE, is computed separately for each of these subsets. The ASE for each data role is the error sum of squares for observations in that role divided by the number of observations in that role.

Using the Validation ASE as the STOP= Criterion

If you have provided observations for validation, then you can specify STOP=VALIDATE as a suboption of the SELECTION= option in the MODEL statement. At step *k* of the selection process, the best candidate effect to enter or leave the current model is determined. Note that here “best candidate” means the effect that gives the best value of the SELECT= criterion that need not be based on the validation data. The validation

ASE for the model with this candidate effect added is computed. If this validation ASE is greater than the validation ASE for the model at step k, then the selection process terminates at step k.

Using the Validation ASE as the CHOOSE= Criterion

When you specify the CHOOSE=VALIDATE suboption of the SELECTION= option in the MODEL statement, the validation ASE is computed for the models at each step of the selection process. The model at the first step yielding the smallest validation ASE is selected.

Using the Validation ASE as the SELECT= Criterion

You request the validation ASE as the selection criterion by specifying the SELECT=VALIDATE suboption of the SELECTION= option in the MODEL statement. At step k of the selection process, the validation ASE is computed for each model where a candidate for entry is added or candidate for removal is dropped. The selected candidate for entry or removal is the one that yields a model with the minimal validation ASE.

Cross Validation

Deciding when to stop a selection method is a crucial issue in performing effect selection. Predictive performance of candidate models on data not used in fitting the model is one approach supported by PROC GLMSELECT for addressing this problem (see the section “Using Validation and Test Data” on page 3867). However, in some cases, you might not have sufficient data to create a sizable training set and a validation set that represent the predictive population well. In these cases, cross validation is an attractive alternative for estimating prediction error.

In k-fold cross validation, the data are split into k roughly equal-sized parts. One of these parts is held out for validation, and the model is fit on the remaining $k - 1$ parts. This fitted model is used to compute the predicted residual sum of squares on the omitted part, and this process is repeated for each of k parts. The sum of the k predicted residual sum of squares so obtained is the estimate of the prediction error that is denoted by CVPRESS. Note that computing the CVPRESS statistic for k-fold cross validation requires fitting k different models, and so the work and memory requirements increase linearly with the number of cross validation folds.

You can use the CVMETHOD= option in the MODEL statement to specify the method for splitting the data into k parts. CVMETHOD=BLOCK(k) requests that the k parts be made of blocks of floor(n/k) or floor(n/k) + 1 successive observations, where n is the number of observations. CVMETHOD=SPLIT(k) requests that parts consist of observations $\{1, k+1, 2k+1, 3k+1, \ldots\}$, $\{2, k+2, 2k+2, 3k+2, \ldots\}$, . . . , $\{k, 2k, 3k, \ldots\}$. CVMETHOD=RANDOM(k) partitions the data into random subsets each with roughly floor(n/k) observations. Finally, you can use the formatted value of an input data set variable to define the parts by specifying CVMETHOD=*variable*. This last partitioning method is useful in cases where you need to exercise extra control over how the data are partitioned by taking into account factors such as important but rare observations that you want to “spread out” across the various parts.

You can request details of the CVPRESS computations by specifying the CVDETAILS= option in the MODEL statement. When you use cross validation, the output data set created with an OUTPUT statement contains an integer-valued variable, _CVINDEX_, whose values indicate the subset to which an observation is assigned.

The widely used special case of n-fold cross validation when you have n observations is known as *leave-one-out* cross validation. In this case, each omitted part consists of one observation, and CVPRESS statistic can be efficiently obtained without refitting the model n times. In this case, the CVPRESS statistic is denoted simply by PRESS and is given by

$$\text{PRESS} = \sum_{i=1}^{n} \left(\frac{r_i}{1 - h_i} \right)^2$$

where r_i is the residual and h_i is the leverage of the ith observation. You can request *leave-one-out* cross validation by specifying PRESS instead of CV with the options SELECT=, CHOOSE=, and STOP= in the MODEL statement. For example, if the number of observations in the data set is 100, then the following two PROC GLMSELECT steps are mathematically equivalent, but the second step is computed much more efficiently:

```
proc glmselect;
   model y=x1-x10/selection=forward(stop=CV) cvMethod=split(100);
run;

proc glmselect;
   model y=x1-x10/selection=forward(stop=PRESS);
run;
```

Hastie, Tibshirani, and Friedman (2001) include a discussion about choosing the cross validation fold. They note that as an estimator of true prediction error, cross validation tends to have decreasing bias but increasing variance as the number of folds increases. They recommend five- or tenfold cross validation as a good compromise. By default, PROC GLMSELECT uses CVMETHOD=RANDOM(5) for cross validation.

Using Cross Validation as the STOP= Criterion

You request cross validation as the stopping criterion by specifying the STOP=CV suboption of the SELECTION= option in the MODEL statement. At step k of the selection process, the best candidate effect to enter or leave the current model is determined. Note that here "best candidate" means the effect that gives the best value of the SELECT= criterion that need not be the CV criterion. The CVPRESS score for the model with this candidate effect added or removed is determined. If this CVPRESS score is greater than the CVPRESS score for the model at step k, then the selection process terminates at step k.

Using Cross Validation as the CHOOSE= Criterion

When you specify the CHOOSE=CV suboption of the SELECTION= option in the MODEL statement, the CVPRESS score is computed for the models at each step of the selection process. The model at the first step that yields the smallest CVPRESS score is selected.

Using Cross Validation as the SELECT= Criterion

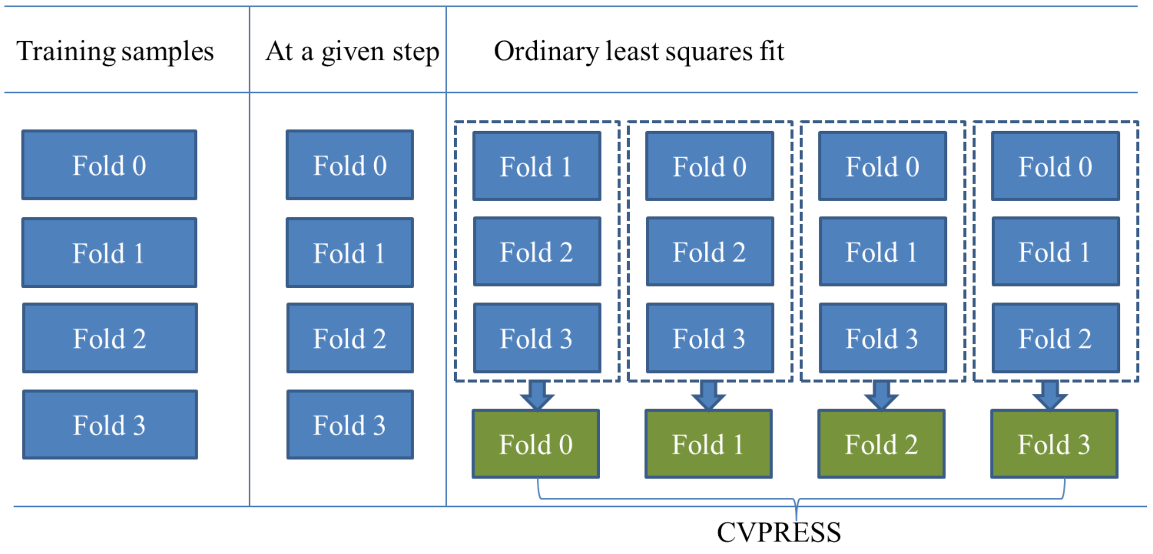
You request cross validation as the selection criterion by specifying the SELECT=CV suboption of the SELECTION= option in the MODEL statement. At step k of the selection process, the CVPRESS score is computed for each model in which a candidate for entry is added or a candidate for removal is dropped. The selected candidate for entry or removal is the one that yields a model that has the minimal CVPRESS score. At each step of the selection process, this requires forming the CVPRESS statistic for all possible candidate models at the next step. Because forming the CVPRESS statistic for k-fold requires fitting k models, using cross validation as the selection criterion is computationally very demanding compared to using other selection criteria.

External Cross Validation

The method of cross validation that is discussed in the previous section judges models by their performance with respect to ordinary least squares. An alternative to ordinary least squares is to use the penalized regression that is defined by the LASSO or elastic net method. This method is called *external cross validation*, and you can specify external cross validation with CHOOSE=CVEX, new in SAS/STAT 13.1. CHOOSE=CVEX applies only when SELECTION=LASSO or SELECTION=ELASTICNET.

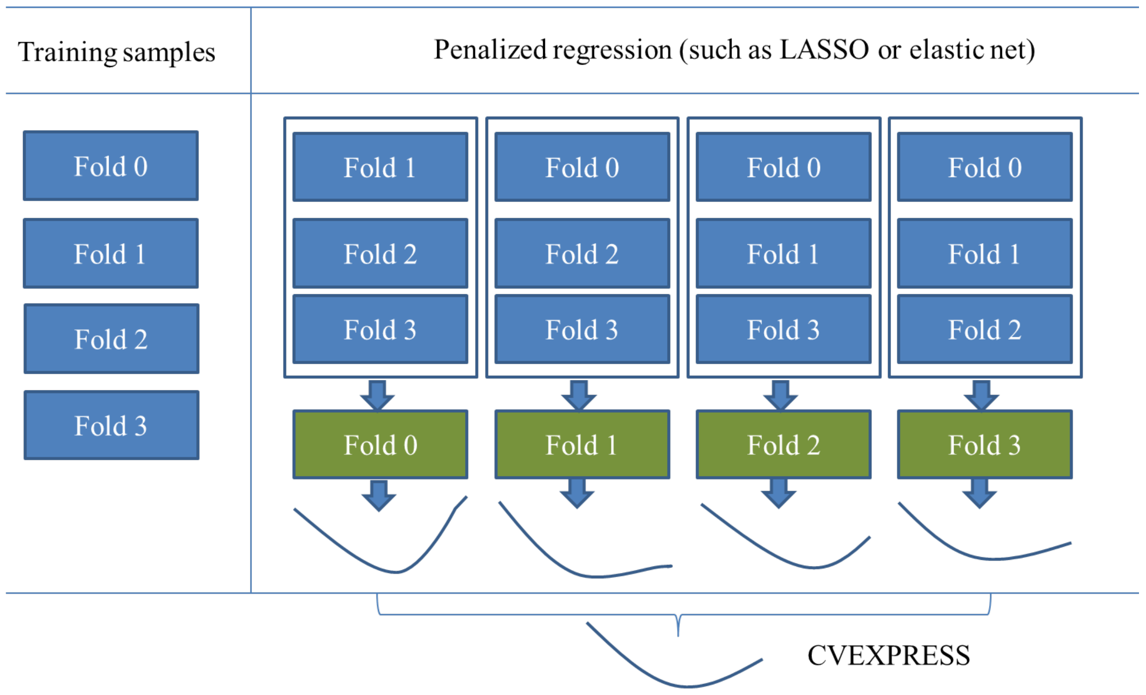
To understand how k-fold external cross validation works, first recall how k-fold cross validation works, as shown in Figure 49.18. The first column in this figure illustrates dividing the training samples into four folds, the second column illustrates the same training samples with reduced numbers of variables at a given step, and the third column illustrates applying ordinary least squares to compute the CVPRESS statistic. For the SELECTION=LASSO and SELECTION=ELASTICNET options, the CVPRESS statistic that is computed by k-fold cross validation uses an ordinary least squares fit, and hence it does not directly depend on the coefficients obtained by the penalized least squares regression.

Figure 49.18 Applying k-fold Cross Validation to Computing the CVPRESS Statistic



If you want a statistic that is directly based on the coefficients obtained by a penalized least squares regression, you can specify CHOOSE=CVEX to use k-fold external cross validation. External cross validation directly applies the coefficients obtained by a penalized least squares regression to computing the predicted residual sum of squares. Figure 49.19 depicts k-fold external cross validation.

Figure 49.19 *k*-fold External Cross Validation



In k-fold external cross validation, the data are split into k approximately equal-sized parts, as illustrated in the first column of Figure 49.19. One of these parts is held out for validation, and the model is fit on the remaining $k-1$ parts by the LASSO method or the elastic net method. This fitted model is used to compute the predicted residual sum of squares on the omitted part, and this process is repeated for each of the k parts. More specifically, for the ith model fit ($i = 1, 2, \ldots, k$), let $\mathbf{X}_{-i}$ denote the held-out part of the matrix of covariates and $\mathbf{y}_{-i}$ denote the corresponding response, and let $\mathbf{X}_i$ denote the remaining $k-1$ parts of the matrix of covariates and $\mathbf{y}_i$ denote the corresponding response. The LASSO method is applied on $\mathbf{X}_i$ and $\mathbf{y}_i$ to solve the optimization problem

$$\min ||\mathbf{y}_i - \mathbf{X}_i \boldsymbol{\beta}||^2 + \lambda_1 \sum_{j=1}^{m} |\beta_j|$$

Note that, as discussed in the section “Elastic Net Selection (ELASTICNET)” on page 3853, the elastic net method can be solved in the same way as LASSO, by augmenting the design matrix $\mathbf{X}_i$ and the response $\mathbf{y}_i$.

Following the piecewise linear solution path of LASSO, the coefficients

$$\boldsymbol{\beta}^{i1}, \boldsymbol{\beta}^{i2}, \ldots, \boldsymbol{\beta}^{ip}, \ldots$$

are computed to correspond to the LASSO regularization parameter

$$\lambda_1^{i1}, \lambda_1^{i2}, \ldots, \lambda_1^{ip}, \ldots$$

Based on the computed coefficients, the predicted residual sum of squares is computed on the held-out part $\mathbf{X}_{-i}$ and $\mathbf{y}_{-i}$ as

$$r^{ip} = ||\mathbf{y}_{-i} - \mathbf{X}_{-i} \boldsymbol{\beta}^{ip}||^2$$

The preceding process can be summarized as

$$(\mathbf{X}_i, \mathbf{y}_i) \rightarrow (\lambda_1^{ip}, \boldsymbol{\beta}^{ip}) \rightarrow (\lambda_1^{ip}, r^{ip}), i = 1, 2, \ldots, k, p = 1, 2, \ldots$$

For the illustration in Figure 49.19, the results (λ_1^{ip}, r^{ip}) correspond to four curves. The knots $\lambda_1^{i1}, \lambda_1^{i2}, \ldots, \lambda_1^{ip}, \ldots$ are usually different among different model fits $i = 1, 2, \ldots, k$. To merge the results of the k model fits for computing the CVEXPRESS statistic, perform the following three steps:

1. Identify distinct knots among all $\lambda_1^{ip}, i = 1, 2, \ldots, k, p = 1, 2, \ldots$
2. Apply interpolation to compute the predicted residual sum of squares at all the knots. Here the interpolation is done in a closed quadratic function by using the piecewise linear solutions of LASSO.
3. Add the predicted residual sum of squares of k model fits according to the identified knots, and the sum of the k predicted residual sum of squares so obtained is the estimate of the prediction error that is denoted by CVEXPRESS.

The bottom curve in Figure 49.19 illustrates the curve between the CVEXPRESS statistics and the value of λ_1 (L1). Note that computing the CVEXPRESS statistic for k-fold external cross validation requires fitting k different LASSO or elastic net models, and so the work and memory requirements increase linearly with the number of cross validation folds.

In addition to characterizing the piecewise linear solutions of the coefficients $\boldsymbol{\beta}^{i1}, \boldsymbol{\beta}^{i2}, \ldots, \boldsymbol{\beta}^{ip}, \ldots$ by the LASSO regularization parameters $\lambda_1^{i1}, \lambda_1^{i2}, \ldots, \lambda_1^{ip}, \ldots$, you can also characterize the solutions by the sum of the absolute values of the coefficients or the scaled regularization parameter. For a detailed discussion of the different options, see the L1CHOICE= option in the MODEL statement.

Like k-fold cross validation, you can use the CVMETHOD= option in the MODEL statement to specify the method for splitting the data into k parts in k-fold external cross validation. CVMETHOD=BLOCK(k) requests that the k parts be made of blocks of floor(n/k) or floor(n/k) + 1 successive observations, where n is the number of observations. CVMETHOD=SPLIT(k) requests that parts consist of observations $\{1, k+1, 2k+1, 3k+1, \ldots\}, \{2, k+2, 2k+2, 3k+2, \ldots\}, \ldots, \{k, 2k, 3k, \ldots\}$. CVMETHOD=RANDOM($k$) partitions the data into random subsets, each with approximately floor(n/k) observations. Finally, you can use the formatted value of an input data set variable to define the parts by specifying CVMETHOD=*variable*. This last partitioning method is useful in cases where you need to exercise extra control over how the data are partitioned by taking into account factors such as important but rare observations that you want to "spread out" across the various parts. By default, PROC GLMSELECT uses CVMETHOD=RANDOM(5) for external cross validation.

For the elastic net method, if the ridge regression parameter λ_2 is not specified by the L2= option and you use k-fold external cross validation for the CHOOSE= option, then the optimal λ_2 is searched over an interval (see Figure 49.12 for an illustration) and it is set to the value that achieves the minimum CVEXPRESS statistic. You can use the L2SEARCH=, L2LOW=, L2HIGH=, and L2STEPS= options to control the search of λ_2 (L2).

Difference between Cross Validation and External Cross Validation

If you specify SELECTION=LASSO or SELECTION=ELASTICNET, the penalized model is fit only once using the same training samples in k-fold cross validation, whereas the penalized model is fit k times by using different training samples in k-fold external cross validation. External cross validation also requires identifying the knots that result from the different solution paths. The CVPRESS statistic that is computed in *k*-fold cross validation is based on ordinary least squares regression, whereas the CVEXPRESS statistic that is computed in *k*-fold external cross validation is based on the penalized regression.

Using External Cross Validation as the CHOOSE= Criterion

When you specify the CHOOSE=CVEX suboption of the SELECTION= option in the MODEL statement, the CVEXPRESS statistics are computed for the models at each step of the selection process. The model at the first step that has the smallest CVEXPRESS score is selected.

Screening

Model selection from a very large number of effects is computationally demanding. For example, in analyzing microarray data (where each dot in the array corresponds to a regressor), it is not uncommon to have 35,000 such regressors. Large regression problems also arise when you want to consider all possible interactions of your main effects as candidates for inclusion in a selected model. For an example that uses this approach to build a predictive model for bankruptcy, see Foster and Stine (2004).

In recent years, there has been a resurgence of interest in combining variable selection methods with a screening approach that reduces the large number of regressors to a much smaller subset from which the model selection is performed. There are two categories of screening methods:

- safe screening methods, by which the resulting solution is exactly the same as the solution when no screening is performed
- heuristic screening methods, by which the resulting solution is not necessarily the same as the solution when no screening is performed

The heuristic screening approaches are usually much faster than the safe screening methods, but they are not guaranteed to reproduce the true LASSO or elastic net solution.

Safe Screening via SCREEN=SASVI

The safe screening approaches are developed mainly for the LASSO method and its extensions, which solve a well-defined convex optimization problem. For these methods, safe screening works as follows:

1. Given a solution $\boldsymbol{\beta}_1$ that corresponds to a regularization parameter λ_1, safe screening approaches aim to identify the effects that are guaranteed to have zero coefficients in the solution $\boldsymbol{\beta}_2$, which corresponds to the regularization parameter λ_2 ($0 < \lambda_2 < \lambda_1$).
2. The computation of $\boldsymbol{\beta}_2$ can exclude such inactive effects, thus saving computation cost.

The idea of safe screening was pioneered by El Ghaoui, Viallon, and Rabbani (2012), and improved subsequently by other researchers (Liu et al. 2014; Wang et al. 2013; Xiang, Xu, and Ramadge 2011).

If you specify SCREEN=SASVI in the model statement, PROC GLMSELECT uses the SASVI technique of Liu et al. (2014) to speed up LAR-type LASSO. The computation cost can usually be reduced while the solution is the same when you specify SCREEN=NONE.

Heuristic Screening via SCREEN=SIS

Heuristic screening approaches (Fan and Lv 2008; Tibshirani et al. 2012) use a screening statistic that is inexpensive to compute in order to eliminate regressors that are unlikely to be selected. For linear regression, you can use the magnitude of the correlation between each individual regressor and the response as such a screening statistic. The square of the correlation between a regressor and the response is the R-square value for the univariate regression of the response on this regressor. Hence, screening by the magnitude of the pairwise correlations is equivalent to fitting univariate models to do the screening.

The SIS (sure independence screening) approach proposed by Fan and Lv (2008) is a well-known heuristic screening approach that applies to model selection methods such as forward selection, backward selection, LASSO, and so on. When you specify SCREEN=SIS in the MODEL statement, PROC GLMSELECT first chooses only the subset of regressors whose screening statistics are among a specified number or percentage of the largest screening statistic values. When you specify SCREEN=SIS, PROC GLMSELECT uses the screening statistic that is the magnitude of the correlation between each individual regressor and the response. Then it performs model selection for the response from this screened subset of the original regressors.

Displayed Output

The following sections describe the displayed output produced by PROC GLMSELECT. The output is organized into various tables, which are discussed in order of appearance. Note that the contents of a table might change depending on the options that you specify.

Model Information

The “Model Information” table displays basic information about the data sets and the settings used to control effect selection. These settings include the following:

- the selection method
- the criteria used to select effects, stop the selection, and choose the selected model
- the effect hierarchy enforced

The ODS name of the “Model Information” table is ModelInfo.

Performance Settings

The “Performance Settings” table displays settings that affect performance. These settings include whether threading is enabled and the number of CPUs available as well as the method used to build the crossproduct matrices. This table is displayed only if you specify the DETAILS option in the PERFORMANCE statement. The ODS name of the “Performance Settings” table is PerfSettings.

Number of Observations

The “Number of Observations” table displays the number of observations read from the input data set and the number of observations used in the analysis. If you specify a FREQ statement, the table also displays the sum of frequencies read and used. If you use a PARTITION statement, the table also displays the number of observations used for each data role. If you specify TESTDATA= or VALDATA= data sets in the PROC GLMSELECT statement, then "Number of Observations" tables are also produced for these data sets. The ODS name of the "Number of Observations" table is NObs.

Class Level Information

The "Class Level Information" table lists the levels of every variable specified in the CLASS statement. The ODS name of the "Class Level Information" table is ClassLevelInfo.

Class Level Coding

The "Class Level Coding" table shows the coding used for variables specified in the CLASS statement. The ODS name of the "Class Level Coding" table is ClassLevelCoding.

Dimensions

The "Dimensions" table displays information about the number of effects and the number of parameters from which the selected model is chosen. If you use split classification variables, then this table also includes the number of effects after splitting is taken into account. The ODS name of the "Dimensions" table is Dimensions.

Candidates

The "Candidates" table displays the effect names and values of the criterion used to select entering or departing effects at each step of the selection process. The effects are displayed in sorted order from best to worst of the selection criterion. You request this table with the DETAILS= option in the MODEL statement. The ODS name of the "Candidates" table is Candidates.

Selection Summary

The "Selection Summary" table displays details about the sequence of steps of the selection process. For each step, the effect that was entered or dropped is displayed along with the statistics used to select the effect, stop the selection, and choose the selected model. You can request that additional statistics be displayed with the STATS= option in the MODEL statement. For all criteria that you can use for model selection, the steps at which the optimal values of these criteria occur are also indicated. The ODS name of the "Selection Summary" table is SelectionSummary.

Stop Reason

The "Stop Reason" table displays the reason why the selection stopped. To facilitate programmatic use of this table, an integer code is assigned to each reason and is included if you output this table by using an ODS OUTPUT statement. The reasons and their associated codes follow:

Code	Stop Reason
1	maximum number of steps done
2	specified number of steps done
3	specified number of effects in model
4	stopping criterion at local optimum
5	model is an exact fit
6	all entering effects are linearly dependent on those in the model
7	all effects are in the model
8	all effects have been dropped
9	requested full least squares fit completed
10	stepwise selection is cycling
11	dropping any effect does not improve the selection criterion
12	no effects are significant at the specified SLE or SLS levels
13	adding or dropping any effect does not improve the selection criterion
14	all remaining effects are required

The ODS name of the "Stop Reason" table is StopReason.

Stop Details

The "Stop Details" table compares the optimal value of the stopping criterion at the final model with how it would change if the best candidate effect were to enter or leave the model. The ODS name of the "Stop Details" table is StopDetails.

Selected Effects

The "Selected Effects" table displays a string containing the list of effects in the selected model. The ODS name of the "Selected Effects" table is SelectedEffects.

ANOVA

The "ANOVA" table displays an analysis of variance for the selected model. This table includes the following:

- the Source of the variation, Model for the fitted regression, Error for the residual error, and C Total for the total variation after correcting for the mean. The Uncorrected Total Variation is produced when the NOINT option is used.
- the degrees of freedom (DF) associated with the source
- the Sum of Squares for the term
- the Mean Square, the sum of squares divided by the degrees of freedom
- the F Value for testing the hypothesis that all parameters are zero except for the intercept. This is formed by dividing the mean square for Model by the mean square for Error.
- the Prob>F, the probability of getting a greater F statistic than that observed if the hypothesis is true. Note that these p-values are displayed only if you specify the "SHOWPVALUES" option in the MODEL statement. These p-values are generally liberal because they are not adjusted for the fact that the terms in the model have been selected.

You can request “ANOVA” tables for the models at each step of the selection process with the DETAILS= option in the MODEL statement. The ODS name of the “ANOVA” table is ANOVA.

Fit Statistics

The “Fit Statistics” table displays fit statistics for the selected model. The statistics displayed include the following:

- Root MSE, an estimate of the standard deviation of the error term. It is calculated as the square root of the mean square error.
- Dep Mean, the sample mean of the dependent variable
- R-square, a measure between 0 and 1 that indicates the portion of the (corrected) total variation attributed to the fit rather than left to residual error. It is calculated as SS(Model) divided by SS(Total). It is also called the *coefficient of determination*. It is the square of the multiple correlation—in other words, the square of the correlation between the dependent variable and the predicted values.
- Adj R-Sq, the adjusted R^2, a version of R^2 that has been adjusted for degrees of freedom. It is calculated as

  $$\bar{R}^2 = 1 - \frac{(n-i)(1-R^2)}{n-p}$$

  where i is equal to 1 if there is an intercept and 0 otherwise, n is the number of observations used to fit the model, and p is the number of parameters in the model.
- fit criteria AIC, AICC, BIC, CP, and PRESS if they are used in the selection process or are requested with the STATS= option. See the section “Criteria Used in Model Selection Methods” on page 3857 for details and Table 49.10 for the formulas for evaluating these criteria.
- the CVPRESS statistic when cross validation is used in the selection process. See the section “Cross Validation” on page 3869 for details.
- the average square errors (ASE) on the training, validation, and test data. See the section “Using Validation and Test Data” on page 3867 for details.

You can request “Fit Statistics” tables for the models at each step of the selection process with the DETAILS= option in the MODEL statement. The ODS name of the “Fit Statistics” table is FitStatistics.

Cross Validation Details

The “Cross Validation Details” table displays the following:

- the fold number
- the number of observations used for fitting
- the number of observations omitted
- the predicted residual sum of squares on the omitted observations

You can request this table with the CVDETAILS= option in the MODEL statement whenever cross validation is used in the selection process. This table is displayed for the selected model, but you can request this table at each step of the selection process by using the DETAILS= option in the MODEL statement. The ODS name of the “Cross Validation Details” table is CVDetails.

Parameter Estimates

The “Parameter Estimates” table displays the parameters in the selected model and their estimates. The information displayed for each parameter in the selected model includes the following:

- the parameter label that includes the effect name and level information for effects containing classification variables
- the degrees of freedom (DF) for the parameter. There is one degree of freedom unless the model is not full rank.
- the parameter estimate
- the standard error, which is the estimate of the standard deviation of the parameter estimate
- T for H0: Parameter=0, the *t* test that the parameter is zero. This is computed as the parameter estimate divided by the standard error.
- the Prob > |T|, the probability that a *t* statistic would obtain a greater absolute value than that observed given that the true parameter is zero. This is the two-tailed significance probability. Note that these *p*-values are displayed only if you specify the SHOWPVALUES option in the MODEL statement. These *p*-values are generally liberal because they are not adjusted for the fact that the terms in the model have been selected.

If cross validation is used in the selection process, then you can request that estimates of the parameters for each cross validation fold be included in the “Parameter Estimates” table by using the CVDETAILS= option in the MODEL statement. You can request “Parameter Estimates” tables for the models at each step of the selection process with the DETAILS= option in the MODEL statement. The ODS name of the “Parameter Estimates” table is ParameterEstimates.

Score Information

For each SCORE statement, the “Score Information” table displays the names of the score input and output data sets, and the number of observations that were read and successfully scored. The ODS name of the “Score Information” table is ScoreInfo.

Timing Breakdown

The “Timing Breakdown” table displays a broad breakdown of where time was spent in the PROC GLMSELECT step. This table is displayed only if you specify the DETAILS option in the PERFORMANCE statement. The ODS name of the “Timing Breakdown” table is Timing.

ODS Table Names

PROC GLMSELECT assigns a name to each table it creates. You can use these names to reference the table when you use the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 49.11.

For more information about ODS, see Chapter 20, “Using the Output Delivery System.”

Table 49.11 ODS Tables Produced by PROC GLMSELECT

ODS Table Name	Description	Statement	Option
ANOVA	Selected model ANOVA table	MODEL	Default
AvgParmEst	Average parameter estimates	MODELAVERAGE	Default
BSplineDetails	B-spline basis details	EFFECT	DETAILS
Candidates	Entry/removal effect ranking	MODEL	DETAILS=
ClassLevelCoding	Classification variable coding	CLASS	SHOWCODING
ClassLevelInfo	Classification variable levels	CLASS	Default
CollectionLevelInfo	Levels of collection effects	EFFECT	DETAILS
CVDetails	Cross validation PRESS by fold	MODEL	CVDETAILS=
Dimensions	Number of effects and parameters	MODEL	Default
EffectSelectPct	Effect selection percentages	MODELAVERAGE	Default
FitStatistics	Selected model fit statistics	MODEL	Default
MMLevelInfo	Levels of multimember effects	EFFECT	DETAILS
ModelAvgInfo	Model averaging information	MODELAVERAGE	Default
ModelInfo	Model information	MODEL	Default
ModelSelectFreq	Model selection frequencies	MODELAVERAGE	Default
NObs	Number of observations	MODEL	Default
ParameterNames	Labels for column names in the design matrix	PROC	OUTDESIGN(names)
ParameterEstimates	Selected model parameter estimates	MODEL	Default
PerfSettings	Performance settings	PERFORMANCE	DETAILS
PolynomialDetails	Polynomial details	EFFECT	DETAILS
PolynomialScaling	Polynomial scaling	EFFECT	DETAILS
RefitAvgParmEst	Refit average parameter estimates	MODELAVERAGE	REFIT
ScoreInfo	Score request information	SCORE	Default
SelectedEffects	List of selected effects	MODEL	Default
SelectionSummary	Selection summary	MODEL	Default
StopDetails	Stopping criterion details	MODEL	Default
StopReason	Reason why selection stopped	MODEL	Default
Timing	Timing details	PERFORMANCE	DETAILS
TPFSplineDeatils	Truncated power function spline basis details	EFFECT	DETAILS

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."

You must also specify the PLOTS= option in the PROC GLMSELECT statement.

The following sections describe the ODS graphical displays produced by PROC GLMSELECT. The examples use the Sashelp.Baseball data set that is described in the section "Getting Started: GLMSELECT Procedure" on page 3804.

ODS Graph Names

PROC GLMSELECT assigns a name to each graph it creates using ODS. You can use these names to reference the graphs when using ODS. The names are listed in Table 49.12.

Table 49.12 Graphs Produced by PROC GLMSELECT

ODS Graph Name	Plot Description	PLOTS Option
AdjRSqPlot	Adjusted R-square by step	CRITERIA(UNPACK)
AICCPlot	Corrected Akaike's information criterion by step	CRITERIA(UNPACK)
AICPlot	Akaike's information criterion by step	CRITERIA(UNPACK)
ASEPlot	Average square errors by step	ASE
BICPlot	Sawa's Bayesian information criterion by step	CRITERIA(UNPACK)
CandidatesPlot	SELECT= criterion by effect	CANDIDATES
ChooseCriterionPlot	CHOOSE= criterion by step	COEFFICIENTS(UNPACK)
CoefficientPanel	Coefficients and CHOOSE= criterion by step	COEFFICIENTS
CoefficientPlot	Coefficients by step	COEFFICIENTS(UNPACK)
CPPlot	Mallows' C_p by step	CRITERIA(UNPACK)
CriterionPanel	Fit criteria by step	CRITERIA
CVPRESSPlot	Cross validation predicted RSS by step	CRITERIA(UNPACK)
EffectSelectPctPlot	Resampling effect selection percentages	EFFECTSELECTPCT
ParmDistPanel	Resampling parameter estimate distributions	PARMDIST
PRESSPlot	Predicted RSS by step	CRITERIA(UNPACK)
SBCPlot	Schwarz Bayesian information criterion by step	CRITERIA(UNPACK)
ValidateASEPlot	Average square error on validation data by step	CRITERIA(UNPACK)

Candidates Plot

You request the "Candidates Plot" by specifying the PLOTS=CANDIDATES option in the PROC GLMSELECT statement and the DETAILS=STEPS option in the MODEL statement. This plot shows the values of selection criterion for the candidate effects for entry or removal, sorted from best to worst from left to right across the plot. The leftmost candidate displayed is the effect selected for entry or removal at that step. You can use this plot to see at what steps the decision about which effect to add or drop is clear-cut. See Figure 49.6 for an example.

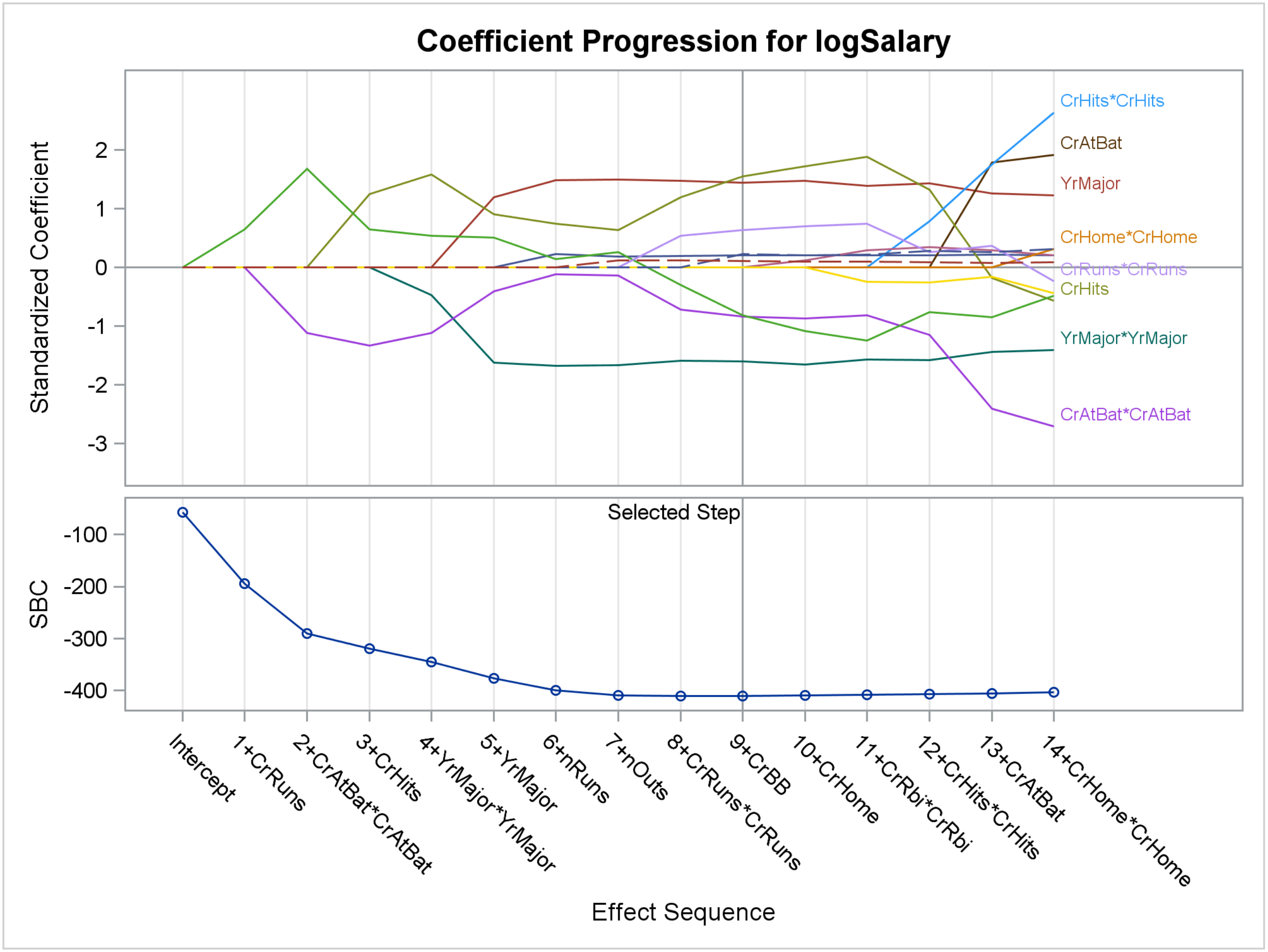
Coefficient Panel

When you specify the PLOTS=COEFFICIENTS option in the PROC GLMSELECT statement, PROC GLMSELECT produces a panel of two plots showing how the standardized coefficients and the criterion used to choose the final model evolve as the selection progresses. The following statements provide an example:

```
ods graphics on;

proc glmselect data=sashelp.baseball plots=coefficients;
   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                     yrMajor|yrMajor crAtBat|crAtBat crHits|crHits
                     crHome|crHome crRuns|crRuns crRbi|crRbi
                     crBB|crBB league division nOuts nAssts nError /
                     selection=forward(stop=AICC CHOOSE=SBC);
run;
```

Figure 49.20 shows the requested graphic. The upper plot in the panel displays the standardized coefficients as a function of the step number. You can request standardized coefficients in the parameter estimates tables by specifying the STB option in the MODEL statement, but this option is not required to produce this plot. To help in tracing the changes in a parameter, the standardized coefficients for each parameter are connected by lines. Coefficients corresponding to effects that are not in the selected model at a step are zero and hence not observable. For example, consider the parameter CrAtBat*CrAtBat in Output 49.20. Because CrAtBat*CrAtBat enters the model at step 2, the line that represents this parameter starts rising from zero at step 1 when CrRuns enters the model. Parameters that are nonzero at the final step of the selection are labeled if their magnitudes are greater than 1% of the range of the magnitudes of all the nonzero parameters at this step. To avoid collision, labels corresponding to parameters with similar values at the final step might get suppressed. You can control when this label collision avoidance occurs by using the LABELGAP= suboption of the PLOTS=COEFFICIENTS option. Planned enhancements to the automatic label collision avoidance algorithm will obviate the need for this option in future releases of the GLMSELECT procedure.

Figure 49.20 Coefficient Panel



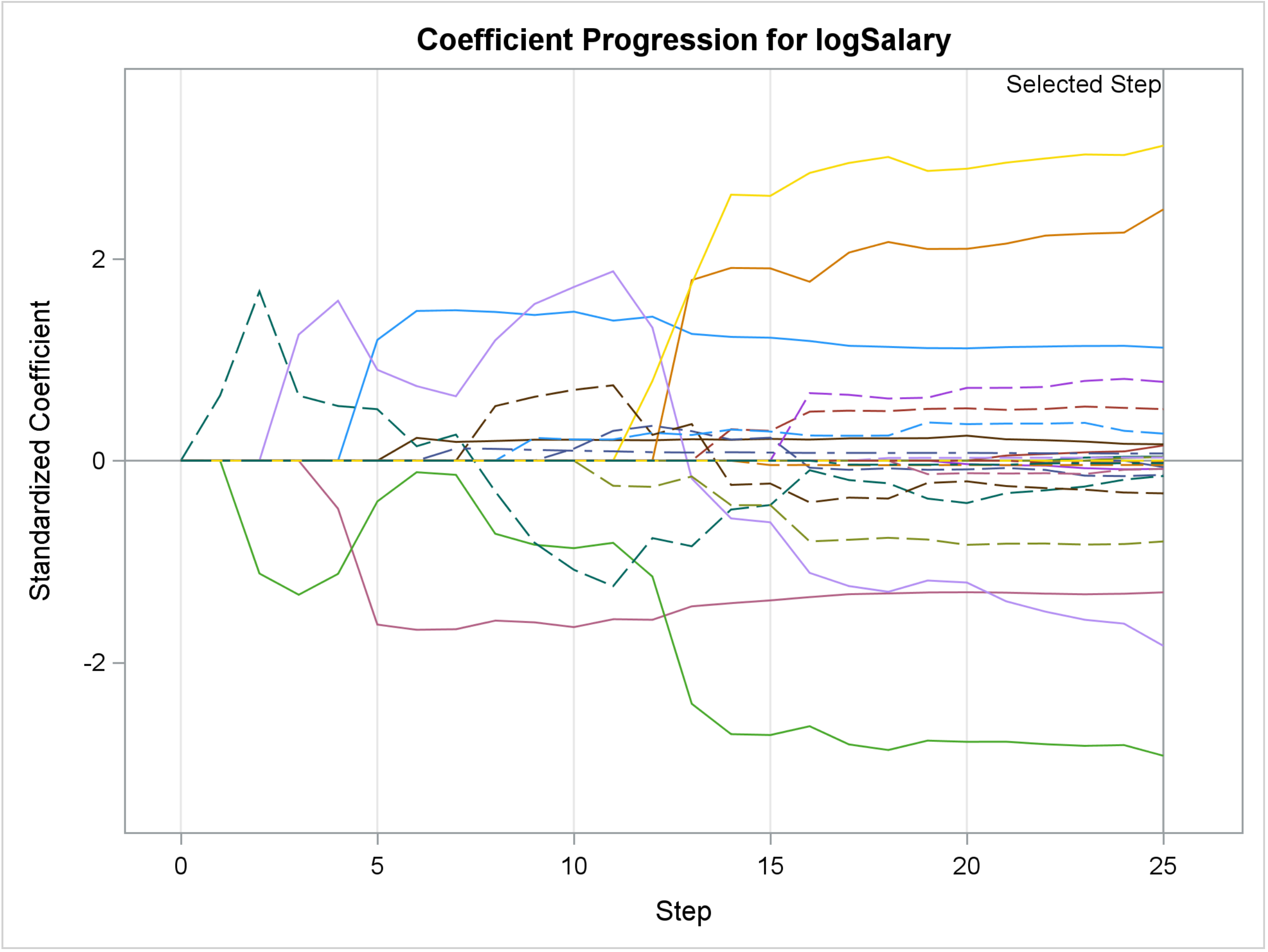
The lower plot in the panel shows how the criterion used to choose among the examined models progresses. The selected step occurs at the optimal value of this criterion. In this example, this criterion is the SBC criterion and it achieves its minimal value at step 9 of the forward selection.

In some cases, particularly when the final step contains a large number of parameters, you might be interested in using this plot only to discern if and when the parameters in the model are essential unchanged beyond a certain step. In such cases, you might want to suppress the labeling of the parameters and use a numeric axis on the horizontal axis of the plot. You can do this using the STEPAXIS= and MAXPARMLABEL= suboptions of the PLOTS=CRITERIA option. The following statements provide an example:

```
proc glmselect data=sashelp.baseball
     plots(unpack maxparmlabel=0 stepaxis=number)=coefficients;

   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                     yrMajor|yrMajor crAtBat|crAtBat crHits|crHits
                     crHome|crHome crRuns|crRuns crRbi|crRbi
                     crBB|crBB league division nOuts nAssts nError /
                     selection=forward(stop=none);
run;
```

The UNPACK = option requests that the plots of the coefficients and CHOOSE= criterion be shown in separate plots. The STEPAXIS=NUMBER option requests a numeric horizontal axis showing step number, and the MAXPAMLABEL=0 option suppresses the labels for the parameters. The "Coefficient Plot" is shown in Figure 49.21. You can see that the standardized coefficients do not vary greatly after step 16.

Figure 49.21 Coefficient Plot



Criterion Panel

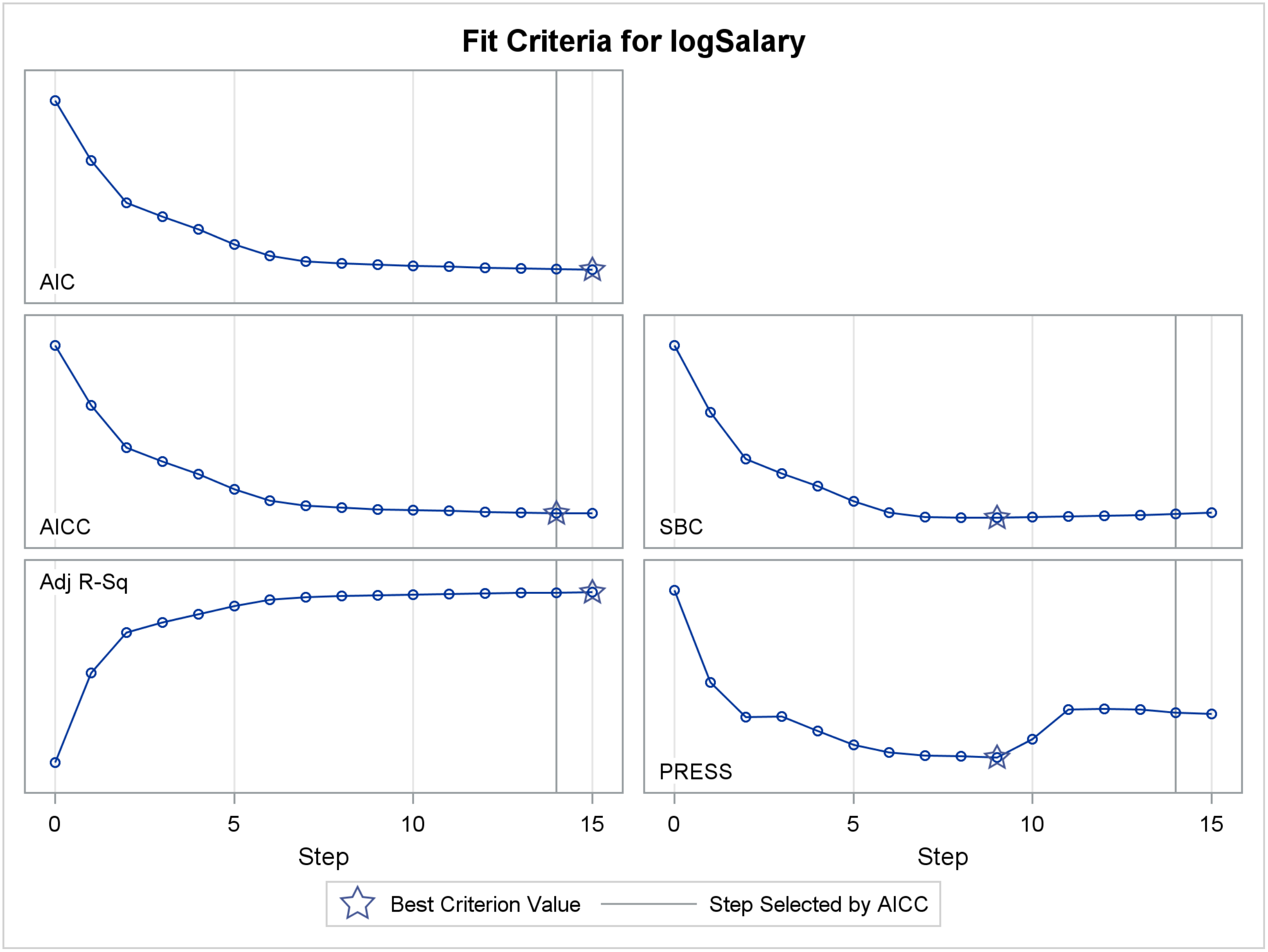
You request the criterion panel by specifying the PLOTS=CRITERIA option in the PROC GLMSELECT statement. This panel displays the progression of the ADJRSQ, AIC, AICC, and SBC criteria, as well as any other criteria that are named in the CHOOSE=, SELECT=, STOP=, or STATS= option in the MODEL statement.

The following statements provide an example:

```
   proc glmselect data=sashelp.baseball plots=criteria;
      class league division;
      model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                        yrMajor|yrMajor crAtBat|crAtBat crHits|crHits
                        crHome|crHome crRuns|crRuns crRbi|crRbi
                        crBB|crBB league division nOuts nAssts nError /
                        selection=forward(steps=15 choose=AICC)
                        stats=PRESS;
   run;
```

Figure 49.22 shows the requested criterion panel. Note that the PRESS criterion is included in the panel because it is named in the STATS= option in the MODEL statement. The selected step is displayed as a vertical reference line on the plot of each criterion, and the legend indicates which of these criteria is used to make the selection. If the selection terminates for a reason other than optimizing a criterion displayed on this plot, then the legend will not report a reason for the selected step. The optimal value of each criterion is indicated with the "Star" marker. Note that it is possible that a better value of a criterion might have been reached had more steps of the selection process been done.

Figure 49.22 Criterion Panel



Average Square Error Plot

You request the average square error plot by specifying the PLOTS=ASE option in the PROC GLMSELECT statement. This plot shows the progression of the average square error (ASE) evaluated separately on the training data, and the test and validation data whenever these data are provided with the TESTDATA= and VALDATA= options or are produced by using a PARTITION statement. You use the plot to detect when overfitting the training data occurs. The ASE decreases monotonically on the training data as parameters are added to a model. However, the average square error on test and validation data typically starts increasing when overfitting occurs. See Output 49.1.9 and Output 49.2.6 for examples.

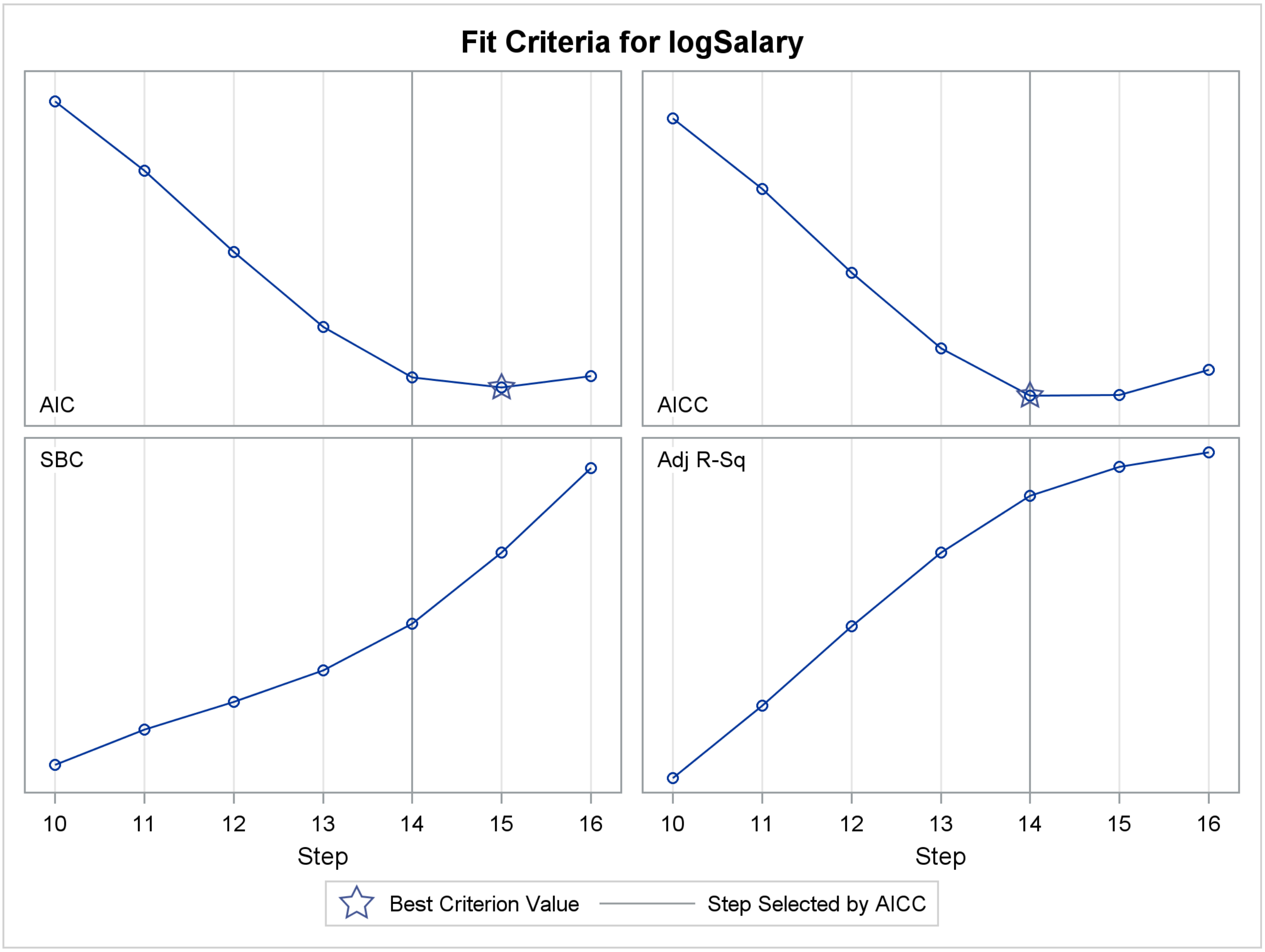
Examining Specific Step Ranges

The coefficient panel, criterion panel, and average square error plot display information for all the steps examined in the selection process. In some cases, you might want to focus attention on just a particular step range. For example, it is hard to discern the variation in the criteria displayed in Figure 49.22 near the selected step because the variation in these criteria in the steps close to the selected step is small relative to the variation across all steps. You can request a range of steps to display using the STARTSTEP= and ENDSTEP= suboptions of the PLOTS= option. You can specify these options as both global and specific plot options, with the specific options taking precedence if both are specified. The following statements provide an example:

```
proc glmselect data=sashelp.baseball plots=criteria(startstep=10 endstep=16);
   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                     yrMajor|yrMajor crAtBat|crAtBat crHits|crHits
                     crHome|crHome crRuns|crRuns crRbi|crRbi
                     crBB|crBB league division nOuts nAssts nError /
                     selection=forward(stop=none choose=AICC);
run;

ods graphics off;
```

Figure 49.23 shows the progression of the fit criteria between steps 10 and 16. Note that if the optimal value of a criterion does not occur in this specified step range, then no optimal marker appears for that criterion. The plot of the SBC criterion in Figure 49.23 is one such case.

Figure 49.23 Criterion Panel for Specified Step Range



Examples: GLMSELECT Procedure

Example 49.1: Modeling Baseball Salaries Using Performance Statistics

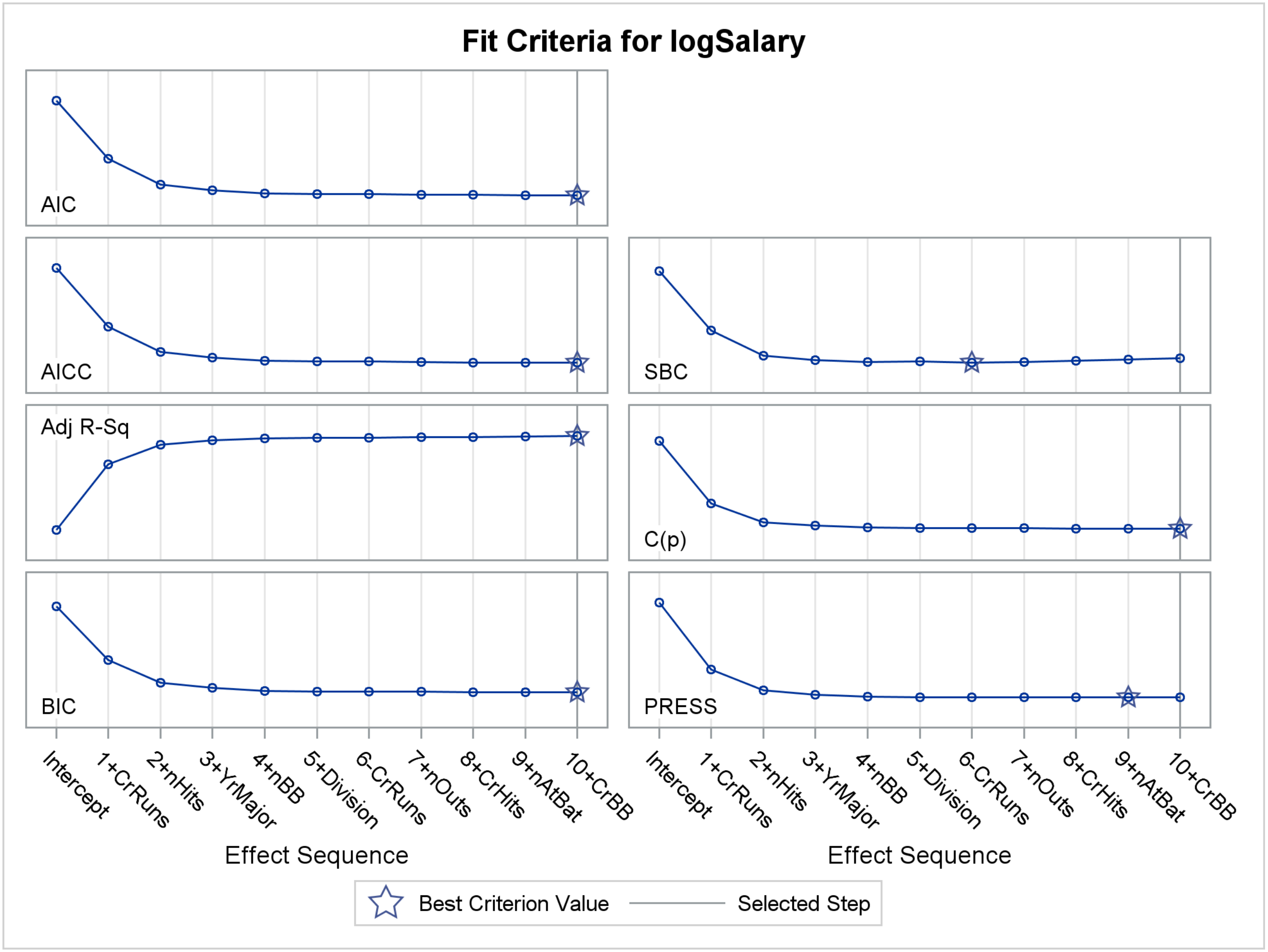
This example continues the investigation of the baseball data set introduced in the section "Getting Started: GLMSELECT Procedure" on page 3804. In that example, the default stepwise selection method based on the SBC criterion was used to select a model. In this example, model selection that uses other information criteria and out-of-sample prediction criteria is explored.

PROC GLMSELECT provides several selection algorithms that you can customize by specifying criteria for selecting effects, stopping the selection process, and choosing a model from the sequence of models at each step. For more details on the criteria available, see the section "Criteria Used in Model Selection Methods" on page 3857. The SELECT=SL suboption of the SELECTION= option in the MODEL statement in the following code requests the traditional hypothesis test-based stepwise selection approach, where effects in the model that are not significant at the stay significance level (SLS) are candidates for removal and effects not yet in the model whose addition is significant at the entry significance level (SLE) are candidates for addition to the model.

```
ods graphics on;

proc glmselect data=sashelp.baseball plot=CriterionPanel;
   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                     yrMajor crAtBat crHits crHome crRuns crRbi
                     crBB league division nOuts nAssts nError
                   / selection=stepwise(select=SL) stats=all;
run;
```

The default SLE and SLS values of 0.15 might not be appropriate for these data. One way to investigate alternative ways to stop the selection process is to assess the sequence of models in terms of model fit statistics. The STATS=ALL option in the MODEL statement requests that all model fit statistics for assessing the sequence of models of the selection process be displayed. To help in the interpretation of the selection process, you can use graphics supported by PROC GLMSELECT. ODS Graphics must be enabled before requesting plots. For general information about ODS Graphics, see Chapter 21, "Statistical Graphics Using ODS." With ODS Graphics enabled, the PLOTS=CRITERIONPANEL option in the PROC GLMSELECT statement produces the criterion panel shown in Output 49.1.1.

Output 49.1.1 Criterion Panel



You can see in Output 49.1.1 that this stepwise selection process would stop at an earlier step if you use the Schwarz Bayesian information criterion (SBC) or predicted residual sum of squares (PRESS) to assess the selected models as stepwise selection progresses. You can use the CHOOSE= suboption of the SELECTION= option in the MODEL statement to specify the criterion you want to use to select among the evaluated models. The following statements use the PRESS statistic to choose among the models evaluated during the stepwise selection.

```
proc glmselect data=sashelp.baseball;
   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                     yrMajor crAtBat crHits crHome crRuns crRbi
                     crBB league division nOuts nAssts nError
                   / selection=stepwise(select=SL choose=PRESS);
run;
```

Note that the selected model is the model at step 9. By default, PROC GLMSELECT displays the selected model, ANOVA and fit statistics, and parameter estimates for the selected model. These are shown in Output 49.1.2.

Output 49.1.2 Details of Selected Model

The GLMSELECT Procedure
Selected Model

The selected model, based on PRESS, is the model at Step 9.

Effects:	Intercept nAtBat nHits nBB YrMajor CrHits Division nOuts

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value
Model	7	124.67715	17.81102	55.07
Error	255	82.47658	0.32344	
Corrected Total	262	207.15373		

Root MSE	0.56872
Dependent Mean	5.92722
R-Square	0.6019
Adj R-Sq	0.5909
AIC	-23.98522
AICC	-23.27376
PRESS	88.55275
SBC	-260.40799

Parameter Estimates

Parameter		DF	Estimate	Standard Error	t Value
Intercept		1	4.176133	0.150539	27.74
nAtBat		1	-0.001468	0.000946	-1.55
nHits		1	0.011078	0.002983	3.71
nBB		1	0.007226	0.002115	3.42
YrMajor		1	0.070056	0.018911	3.70
CrHits		1	0.000247	0.000143	1.72
Division	**East**	1	0.143082	0.070972	2.02
Division	**West**	0	0	.	.
nOuts		1	0.000241	0.000134	1.81

Even though the model that is chosen to give the smallest value of the PRESS statistic is the model at step 9, the stepwise selection process continues to the step where the stopping condition based on entry and stay significance levels is met. If you use the PRESS statistic as the stopping criterion, the stepwise selection process stops at step 9. This ability to stop at the first extremum of the criterion you specify can significantly reduce the amount of computation done, especially in the cases where you are selecting from a large number of effects. The following statements request stopping based on the PRESS statistic. The stop reason and stop details tables are shown in Output 49.1.3.

```
proc glmselect data=sashelp.baseball plot=Coefficients;
   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                  yrMajor crAtBat crHits crHome crRuns crRbi
                  crBB league division nOuts nAssts nError
                / selection=stepwise(select=SL stop=PRESS);
run;
```

Output 49.1.3 Stopping Based on PRESS Statistic

The GLMSELECT Procedure

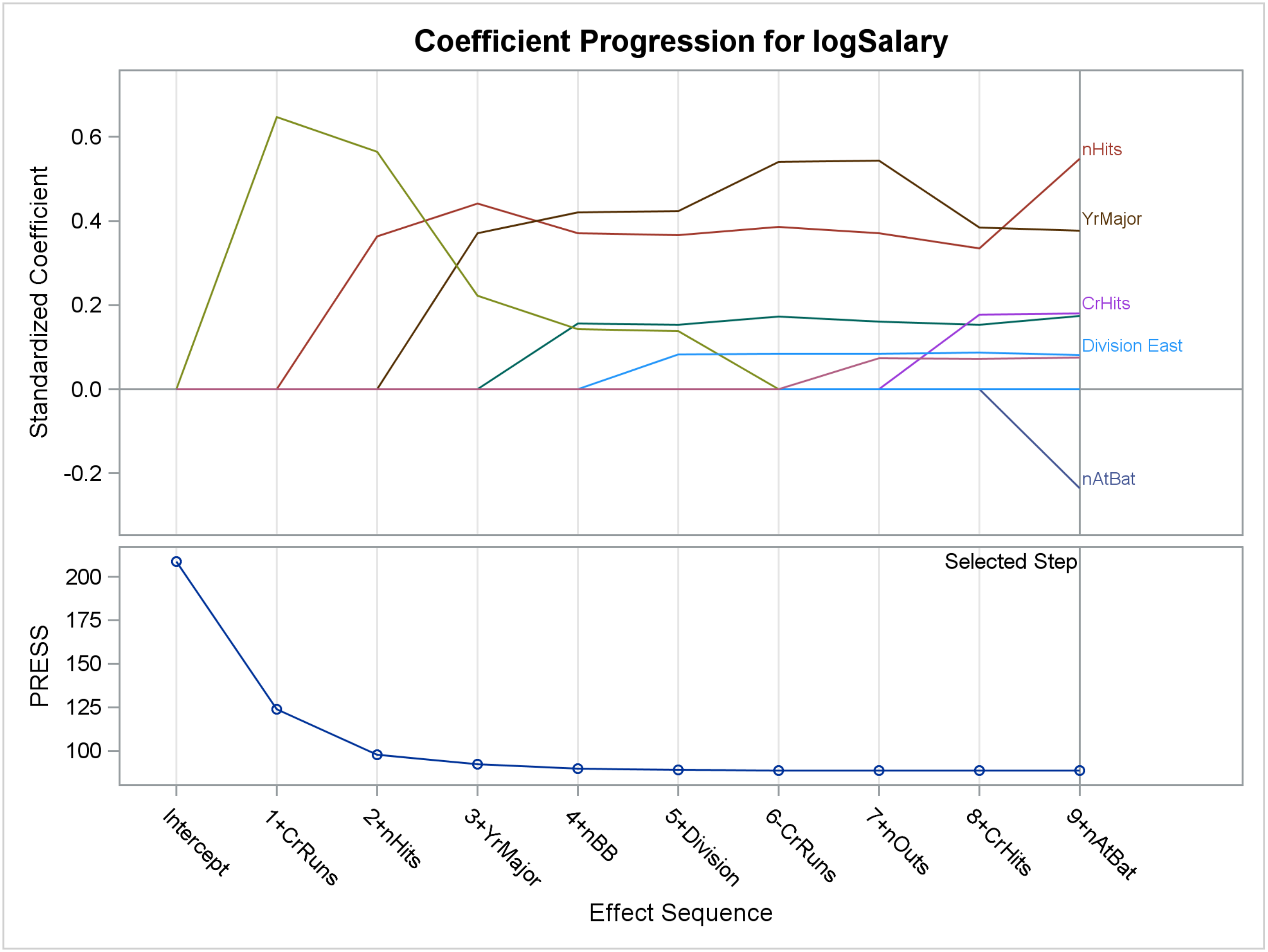
Selection stopped at a local minimum of the PRESS criterion.

Stop Details

Candidate For	Effect	Candidate PRESS		Compare PRESS
Entry	CrBB	88.6321	>	88.5528
Removal	nAtBat	88.6866	>	88.5528

The PLOTS=COEFFICIENTS specification in the PROC GLMSELECT statement requests a plot that enables you to visualize the selection process.

Output 49.1.4 Coefficient Progression Plot



Output 49.1.4 shows the standardized coefficients of all the effects selected at some step of the stepwise method plotted as a function of the step number. This enables you to assess the relative importance of the effects selected at any step of the selection process as well as providing information as to when effects entered the model. The lower plot in the panel shows how the criterion used to choose the selected model changes as effects enter or leave the model.

Model selection is often done in order to obtain a parsimonious model that can be used for prediction on new data. An ever-present danger is that of selecting a model that overfits the "training" data used in the fitting process, yielding a model with poor predictive performance. Using cross validation is one way to assess the predictive performance of the model. Using k-fold cross validation, the training data are subdivided into k parts, and at each step of the selection process, models are obtained on each of the k subsets of the data obtained by omitting one of these parts. The cross validation predicted residual sum of squares, denoted CV PRESS, is obtained by summing the squares of the residuals when each of these submodels is scored on the data omitted in fitting the submodel. Note that the PRESS statistic corresponds to the special case of "leave-one-out" cross validation.

In the preceding example, the PRESS statistic was used to choose among models that were chosen based on entry and stay significance levels. In the following statements, the SELECT=CVPRESS suboption of the SELECTION= option in the MODEL statement requests that the CV PRESS statistic itself be used as the selection criterion. The DROP=COMPETITIVE suboption requests that additions and deletions be considered simultaneously when deciding whether to add or remove an effect. At any step, the CV PRESS statistic for all models obtained by deleting one effect from the model or adding one effect to the model is computed. Among these models, the one yielding the smallest value of the CV PRESS statistic is selected and the process is repeated from this model. The stepwise selection terminates if all additions or deletions increase the CV PRESS statistic. The CVMETHOD=SPLIT(5) option requests five-fold cross validation with the five subsets consisting of observations $\{1, 6, 11, \ldots\}$, $\{2, 7, 12, \ldots\}$, and so on.

```
proc glmselect data=sashelp.baseball plot=Candidates;
   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                  yrMajor crAtBat crHits crHome crRuns crRbi
                  crBB league division nOuts nAssts nError
                / selection=stepwise(select=CV drop=competitive)
                  cvMethod=split(5);
run;
```

The selection summary table is shown in Output 49.1.5. By comparing Output 49.1.5 and Output 49.7 you can see that the sequence of models produced is different from the sequence when the stepwise selection is based on the SBC statistic.

Output 49.1.5 Stepwise Selection Based on Cross Validation

The GLMSELECT Procedure

Stepwise Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	Number Parms In	CV PRESS
0	Intercept		1	1	208.9638
1	CrRuns		2	2	122.5755
2	nHits		3	3	96.3949
3	YrMajor		4	4	92.2117
4	nBB		5	5	89.5242
5		CrRuns	4	4	88.6917
6	League		5	5	88.0417
7	nError		6	6	87.3170
8	Division		7	7	87.2147
9	nHome		8	8	87.0960*

* Optimal Value of Criterion

If you have sufficient data, another way you can assess the predictive performance of your model is to reserve part of your data for testing your model. You score the model obtained using the training data on the test data and assess the predictive performance on these data that had no role in the selection process. You can also reserve part of your data to validate the model you obtain in the training process. Note that the validation data are not used in obtaining the coefficients of the model, but they are used to decide when to stop the selection process to limit overfitting.

PROC GLMSELECT enables you to partition your data into disjoint subsets for training validation and testing roles. This partitioning can be done by using random proportions of the data, or you can designate a variable in your data set that defines which observations to use for each role. See the section "PARTITION Statement" on page 3843 for more details.

The following statements randomly partition the baseball data set, using 50% for training, 30% for validation, and 20% for testing. The model selected at each step is scored on the validation data, and the average residual sums of squares (ASE) is evaluated. The model yielding the lowest ASE on the validation data is selected. The ASE on the test data is also evaluated, but these data play no role in the selection process. Note that a seed for the pseudo-random number generator is specified in the PROC GLMSELECT statement.

```
proc glmselect data=sashelp.baseball plots=(CriterionPanel ASE) seed=1;
   partition fraction(validate=0.3 test=0.2);
   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                     yrMajor crAtBat crHits crHome crRuns crRbi
                     crBB league division nOuts nAssts nError
                   / selection=forward(choose=validate stop=10);
run;
```

Output 49.1.6 Number of Observations Table

The GLMSELECT Procedure

Number of Observations Read	322
Number of Observations Used	263
Number of Observations Used for Training	132
Number of Observations Used for Validation	80
Number of Observations Used for Testing	51

Output 49.1.6 shows the number of observation table. You can see that of the 263 observations that were used in the analysis, 132 (50.2%) observations were used for model training, 80 (30.4%) for model validation, and 51 (19.4%) for model testing.

Output 49.1.7 Selection Summary and Stop Reason

The GLMSELECT Procedure

Forward Selection Summary

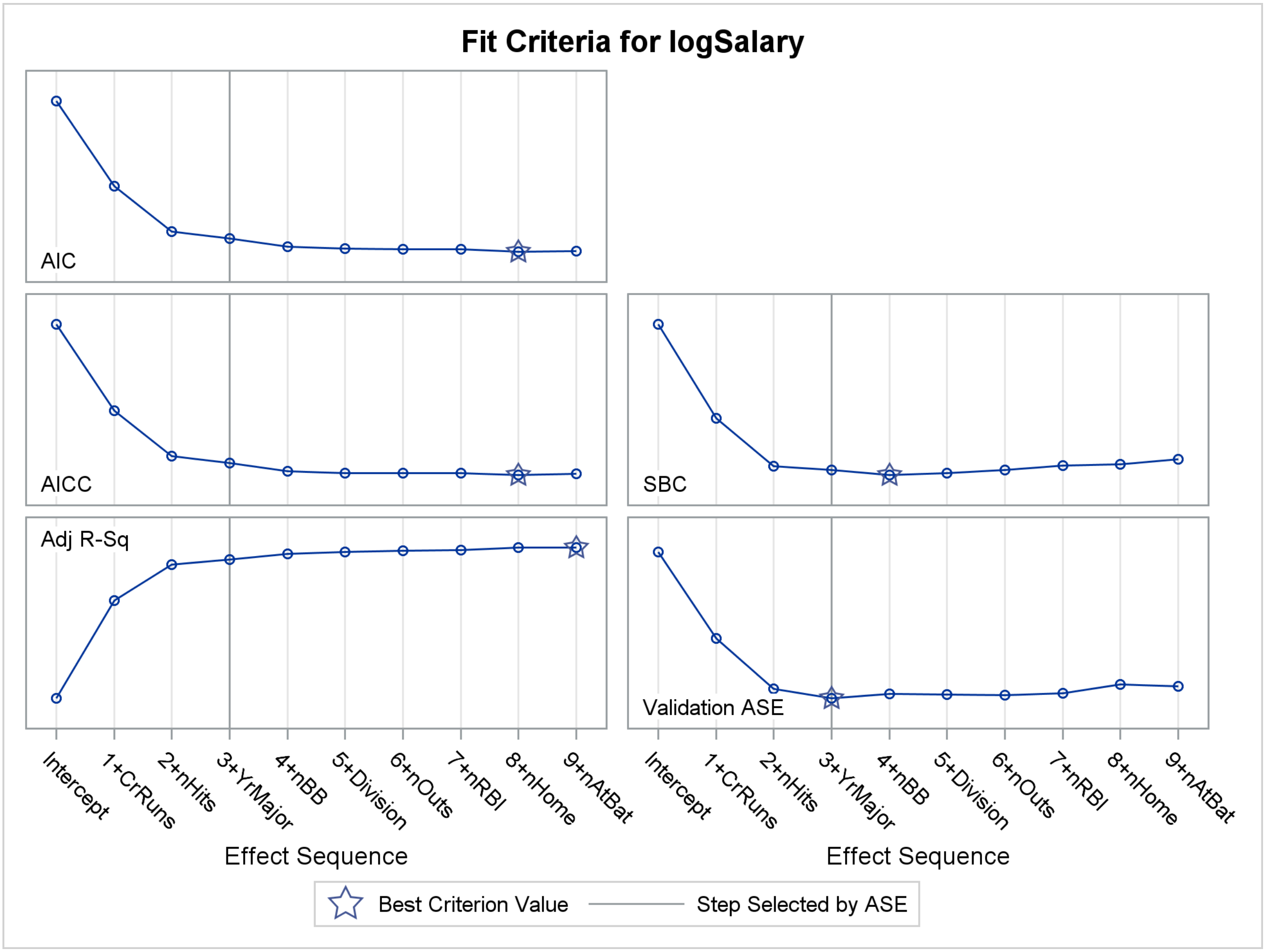
Step	Effect Entered	Number Effects In	Number Parms In	SBC	ASE	Validation ASE	Test ASE
0	**Intercept**	1	1	-30.8531	0.7628	0.7843	0.8818
1	**CrRuns**	2	2	-93.9367	0.4558	0.4947	0.4210
2	**nHits**	3	3	-126.2647	0.3439	0.3248	0.4697
3	**YrMajor**	4	4	-128.7570	0.3252	0.2920*	0.4614
4	**nBB**	5	5	-132.2409*	0.3052	0.3065	0.4297
5	**Division**	6	6	-130.7794	0.2974	0.3050	0.4218
6	**nOuts**	7	7	-128.5897	0.2914	0.3028	0.4186
7	**nRBI**	8	8	-125.7825	0.2868	0.3097	0.4489
8	**nHome**	9	9	-124.7709	0.2786	0.3383	0.4533
9	**nAtBat**	10	10	-121.3767	0.2754	0.3337	0.4580

*** Optimal Value of Criterion**

Selection stopped at the first model containing the specified number of effects (10).

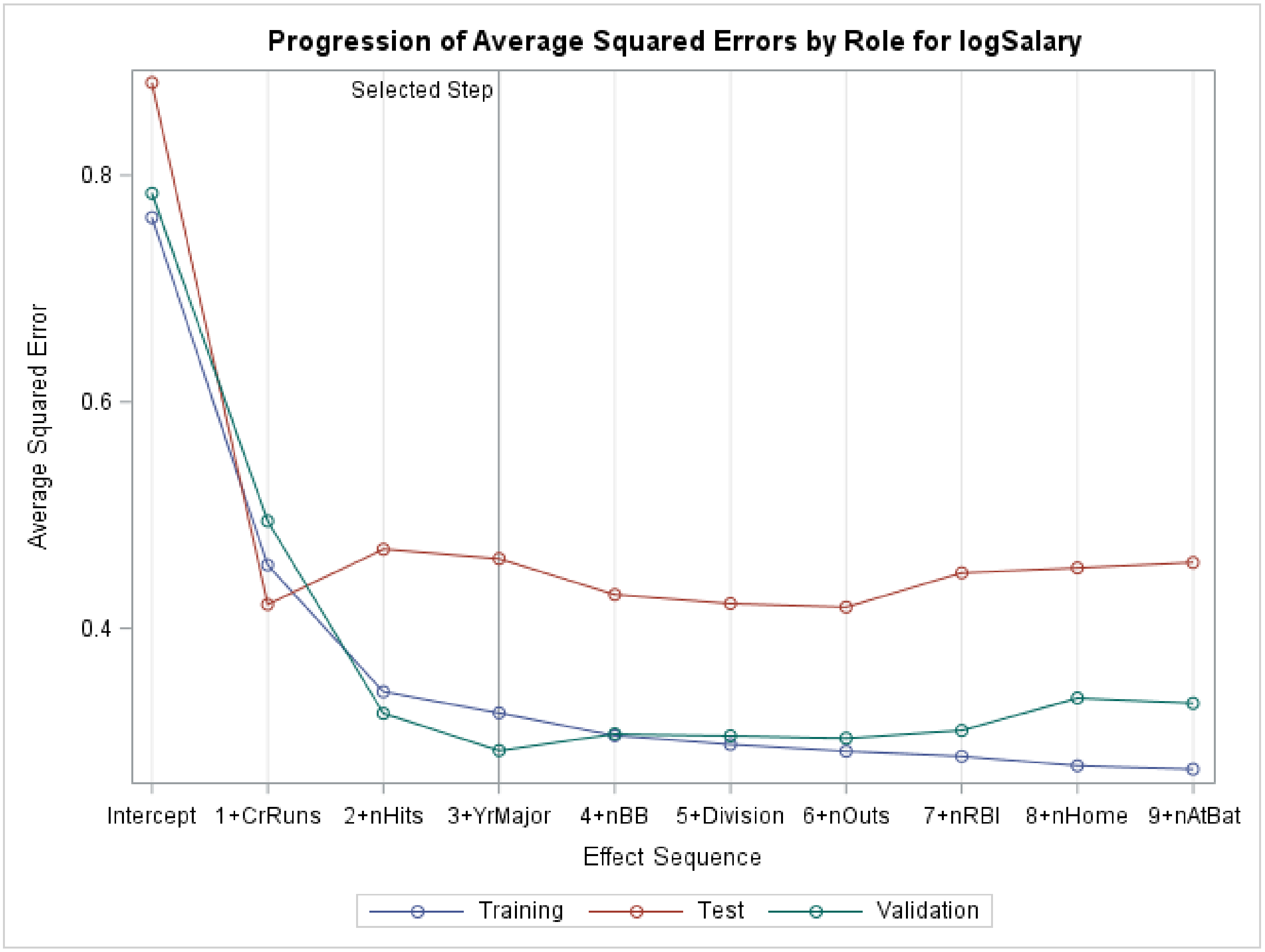
Output 49.1.7 shows the selection summary table and the stop reason. The forward selection stops at step 9 since the model at this step contains 10 effects, and so it satisfies the stopping criterion requested with the STOP=10 suboption. However, the selected model is the model at step 3, where the validation ASE, the CHOOSE= criterion, achieves its minimum.

Output 49.1.8 Criterion Panel



The criterion panel in Output 49.1.8 shows how the various criteria evolved as the stepwise selection method proceeded. Note that other than the ASE evaluated on the validation data, these criteria are evaluated on the training data.

Output 49.1.9 Average Square Errors by Role



Finally, the ASE plot in Output 49.1.9 shows how the average square error evolves on the training, validation, and test data. Note that while the ASE on the training data continued decreasing as the selection steps proceeded, the ASE on the test and validation data behave more erratically.

LASSO selection, pioneered by Tibshirani (1996), is a constrained least squares method that can be viewed as a stepwise-like method where effects enter and leave the model sequentially. You can find additional details about the LASSO method in the section "Lasso Selection (LASSO)" on page 3852. Note that when classification effects are used with LASSO, the design matrix columns for all effects containing classification variables can enter or leave the model individually. The following statements perform LASSO selection for the baseball data. The LASSO selection summary table is shown in Output 49.1.10.

```
proc glmselect data=sashelp.baseball plot=CriterionPanel ;
   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                  yrMajor crAtBat crHits crHome crRuns crRbi
                  crBB league division nOuts nAssts nError
                / selection=LASSO(choose=CP steps=20);
run;

ods graphics off;
```

Output 49.1.10 Selection Summary for LASSO Selection

The GLMSELECT Procedure

LASSO Selection Summary

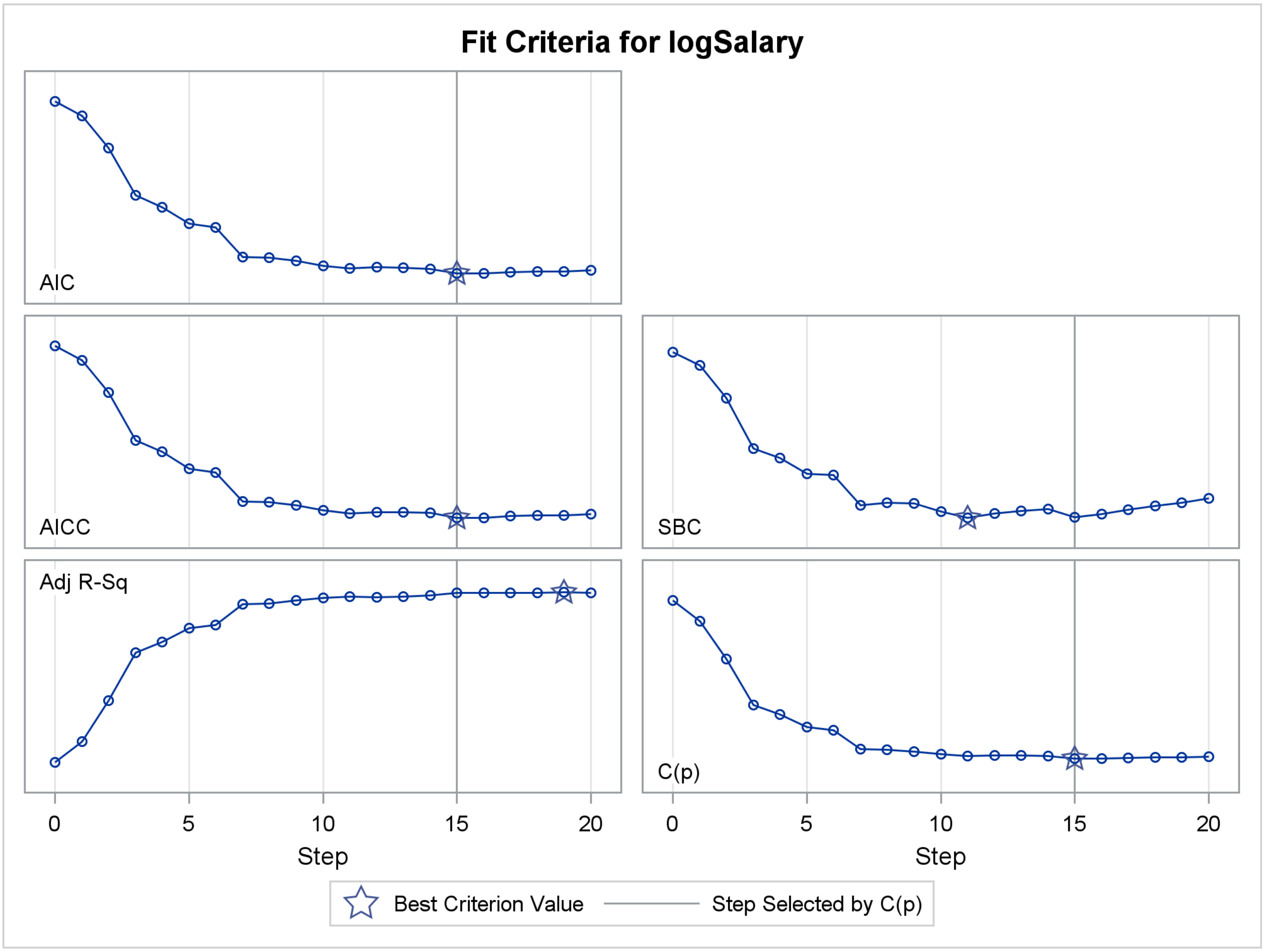
Step	Effect Entered	Effect Removed	Number Effects In	CP
0	Intercept		1	375.9275
1	CrRuns		2	328.6492
2	CrHits		3	239.5392
3	nHits		4	134.0374
4	nBB		5	111.6638
5	CrRbi		6	81.7296
6	YrMajor		7	75.0428
7	nRBI		8	30.4494
8	Division_East		9	29.9913
9	nOuts		10	25.1656
10		CrRuns	9	18.7295
11		CrRbi	8	15.1683
12	nError		9	16.6233
13	nHome		10	16.3741
14	League_American		11	14.8794
15		nRBI	10	8.8477*
16	CrBB		11	9.2242
17	CrRuns		12	10.7608
18	nAtBat		13	11.6266
19	nAssts		14	11.8572
20	CrAtBat		15	13.4020

*** Optimal Value of Criterion**

Selection stopped at the specified number of steps (20).

Note that effects enter and leave sequentially. In this example, the STEPS= suboption of the SELECTION= option specifies that 20 steps of LASSO selection be done. You can see how the various model fit statistics evolved in Output 49.1.11.

Output 49.1.11 Criterion Panel



The CHOOSE=CP suboption specifies that the selected model be the model at step 15 that yields the optimal value of Mallows' C_p statistic. Details of this selected model are shown in Output 49.1.12.

Output 49.1.12 Selected Model

The GLMSELECT Procedure
Selected Model

The selected model, based on C(p), is the model at Step 15.

Effects: Intercept nHits nHome nBB YrMajor CrHits League_American Division_East nOuts nError

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value
Model	9	125.24302	13.91589	42.98
Error	253	81.91071	0.32376	
Corrected Total	262	207.15373		

Output 49.1.12 *continued*

Root MSE	0.56900
Dependent Mean	5.92722
R-Square	0.6046
Adj R-Sq	0.5905
AIC	-21.79589
AICC	-20.74409
BIC	-283.91417
C(p)	8.84767
SBC	-251.07435

Parameter Estimates

Parameter	DF	Estimate
Intercept	1	4.204236
nHits	1	0.006942
nHome	1	0.002785
nBB	1	0.005727
YrMajor	1	0.067054
CrHits	1	0.000249
League_American	1	-0.079607
Division_East	1	0.134723
nOuts	1	0.000183
nError	1	-0.007213

Example 49.2: Using Validation and Cross Validation

This example shows how you can use both test set and cross validation to monitor and control variable selection. It also demonstrates the use of split classification variables.

The following statements produce analysis and test data sets. Note that the same statements are used to generate the observations that are randomly assigned for analysis and test roles in the ratio of approximately two to one.

```
data analysisData testData;
   drop i j c3Num;
   length c3 $ 7;

   array x{20} x1-x20;

   do i=1 to 1500;
      do j=1 to 20;
         x{j} = ranuni(1);
      end;

      c1 = 1 + mod(i,8);
      c2 = ranbin(1,3,.6);

      if      i < 50   then do; c3 = 'tiny';     c3Num=1;end;
      else if i < 250  then do; c3 = 'small';    c3Num=1;end;
      else if i < 600  then do; c3 = 'average';  c3Num=2;end;
      else if i < 1200 then do; c3 = 'big';      c3Num=3;end;
      else                  do; c3 = 'huge';     c3Num=5;end;

      y = 10 + x1 + 2*x5 + 3*x10 + 4*x20  + 3*x1*x7 + 8*x6*x7
             + 5*(c1=3)*c3Num + 8*(c1=7)  + 5*rannor(1);

      if ranuni(1) < 2/3 then output analysisData;
                         else output testData;
   end;
run;
```

Suppose you suspect that the dependent variable depends on both main effects and two-way interactions. You can use the following statements to select a model:

```
ods graphics on;

proc glmselect data=analysisData testdata=testData
               seed=1 plots(stepAxis=number)=(criterionPanel ASEPlot);
   partition fraction(validate=0.5);
   class c1 c2 c3(order=data);
   model y =  c1|c2|c3|x1|x2|x3|x4|x5|x5|x6|x7|x8|x9|x10
             |x11|x12|x13|x14|x15|x16|x17|x18|x19|x20 @2
           / selection=stepwise(choose = validate
                                select = sl)
             hierarchy=single stb;
run;
```

Note that a TESTDATA= data set is named in the PROC GLMSELECT statement and that a PARTITION statement is used to randomly assign half the observations in the analysis data set for model validation and the rest for model training. You find details about the number of observations used for each role in the number of observations tables shown in Output 49.2.1.

Output 49.2.1 Number of Observations Tables

The GLMSELECT Procedure

Observation Profile for Analysis Data	
Number of Observations Read	1010
Number of Observations Used	1010
Number of Observations Used for Training	510
Number of Observations Used for Validation	500

The “Class Level Information” and “Dimensions” tables are shown in Output 49.2.2. The “Dimensions” table shows that at each step of the selection process, 278 effects are considered as candidates for entry or removal. Since several of these effects have multilevel classification variables as members, there are 661 parameters.

Output 49.2.2 Class Level Information and Problem Dimensions

Class Level Information

Class	Levels	Values
c1	8	1 2 3 4 5 6 7 8
c2	4	0 1 2 3
c3	5	tiny small average big huge

Dimensions	
Number of Effects	278
Number of Parameters	661

The model statement options request stepwise selection with the default entry and stay significance levels used for both selecting entering and departing effects and stopping the selection method. The CHOOSE=VALIDATE suboption specifies that the selected model is chosen to minimize the predicted residual sum of squares when the models at each step are scored on the observations reserved for validation. The HIERARCHY=SINGLE option specifies that interactions can enter the model only if the corresponding main effects are already in the model, and that main effects cannot be dropped from the model if an interaction with such an effect is in the model. These settings are listed in the model information table shown in Output 49.2.3.

Output 49.2.3 Model Information

The GLMSELECT Procedure

Data Set	WORK.ANALYSISDATA
Test Data Set	WORK.TESTDATA
Dependent Variable	y
Selection Method	Stepwise
Select Criterion	Significance Level
Stop Criterion	Significance Level
Choose Criterion	Validation ASE
Entry Significance Level (SLE)	0.15
Stay Significance Level (SLS)	0.15
Effect Hierarchy Enforced	Single
Random Number Seed	1

The stop reason and stop details tables are shown in Output 49.2.4. Note that because the STOP= suboption of the SELECTION= option was not explicitly specified, the stopping criterion used is the selection criterion, namely significance level.

Output 49.2.4 Stop Details

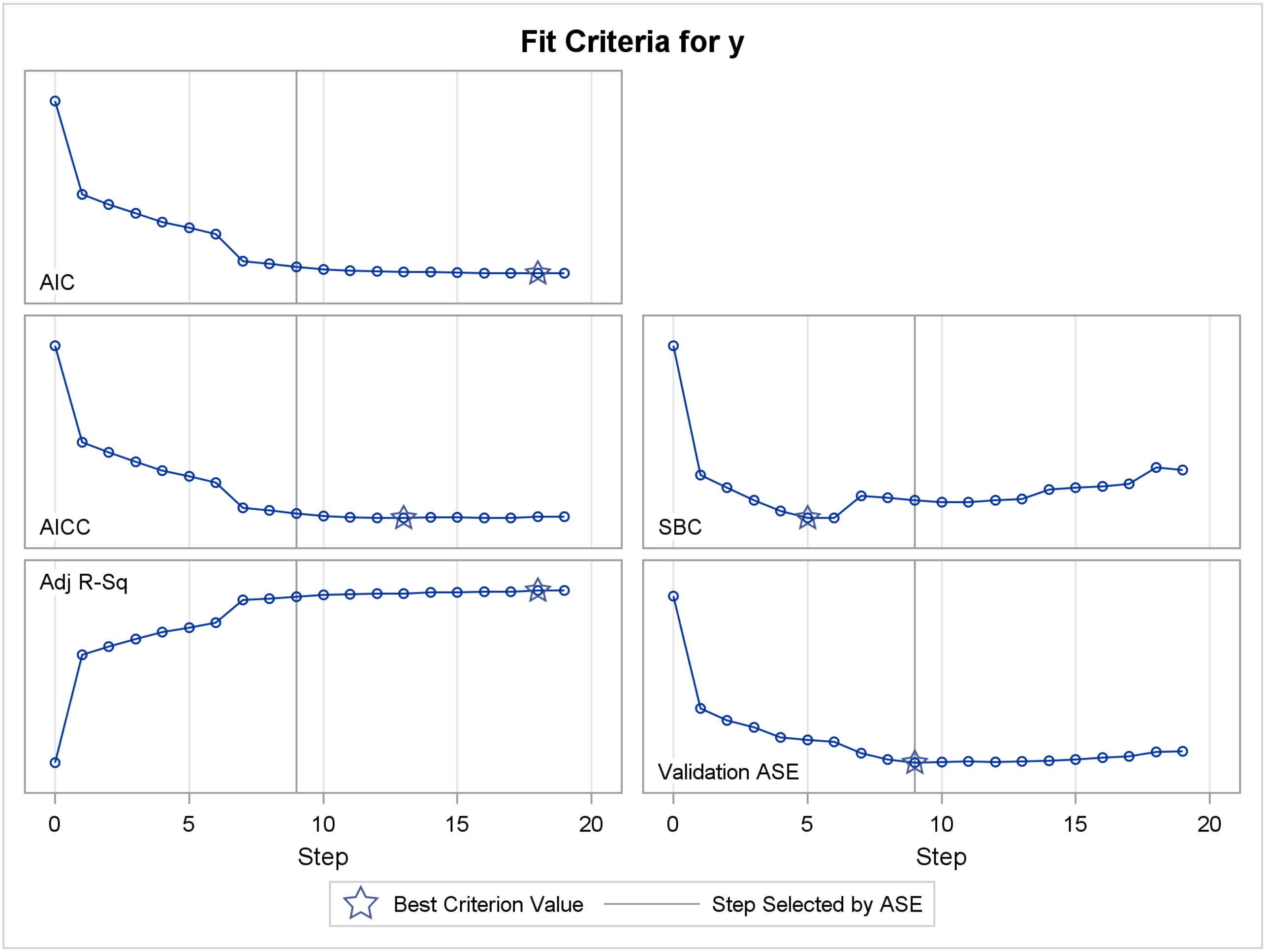
```
Selection stopped because the candidate for entry has SLE > 0.15 and the candidate for removal has SLS < 0.15.
```

Stop Details

Candidate For	Effect	Candidate Significance		Compare Significance	
Entry	x2*x5	0.1742	>	0.1500	(SLE)
Removal	x5*x10	0.0534	<	0.1500	(SLS)

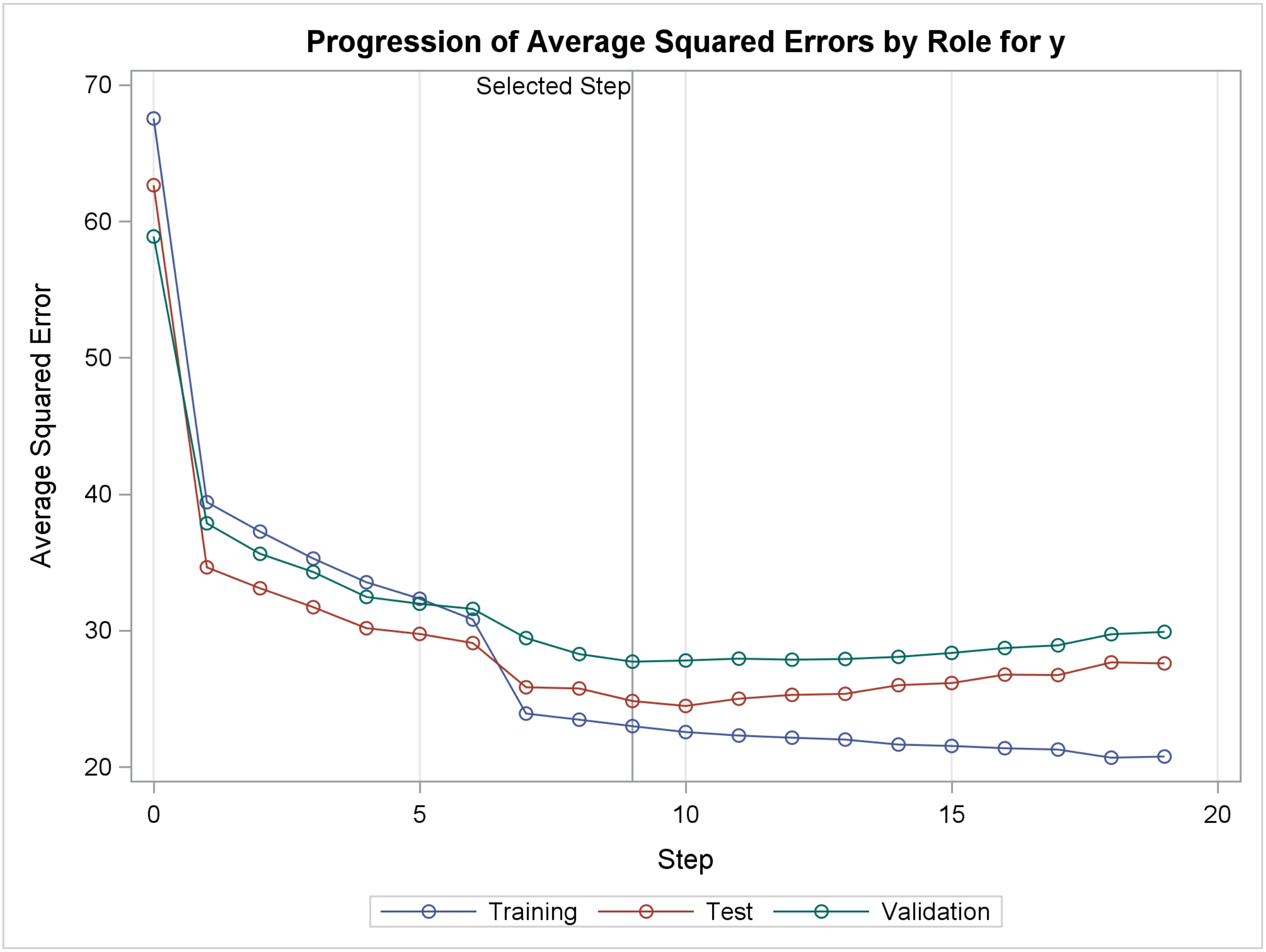
The criterion panel in Output 49.2.5 shows how the various fit criteria evolved as the stepwise selection method proceeded. Note that other than the ASE evaluated on the validation data, these criteria are evaluated on the training data. You see that the minimum of the validation ASE occurs at step 9, and hence the model at this step is selected.

Output 49.2.5 Criterion Panel



Output 49.2.6 shows how the average squared error (ASE) evolved on the training, validation, and test data. Note that while the ASE on the training data decreases monotonically, the errors on both the validation and test data start increasing beyond step 9. This indicates that models after step 9 are beginning to overfit the training data.

Output 49.2.6 Average Squared Errors



Output 49.2.7 shows the selected effects, analysis of variance, and fit statistics tables for the selected model. Output 49.2.8 shows the parameter estimates table.

Output 49.2.7 Selected Model Details

The GLMSELECT Procedure
Selected Model

The selected model, based on Validation ASE, is the model at Step 9.

Effects:	Intercept c1 c3 c1*c3 x1 x5 x6 x7 x10 x20

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value
Model	44	22723	516.43621	20.49
Error	465	11722	25.20856	
Corrected Total	509	34445		

Root MSE	5.02081
Dependent Mean	21.09705
R-Square	0.6597
Adj R-Sq	0.6275
AIC	2200.75319
AICC	2210.09228
SBC	1879.30167
ASE (Train)	22.98427
ASE (Validate)	27.71105
ASE (Test)	24.82947

Output 49.2.8 Parameter Estimates

Parameter Estimates

Parameter		DF	Estimate	Standardized Estimate	Standard Error	t Value
Intercept		1	6.867831	0	1.524446	4.51
c1	**1**	1	0.226602	0.008272	2.022069	0.11
c1	**2**	1	-1.189623	-0.048587	1.687644	-0.70
c1	**3**	1	25.968930	1.080808	1.693593	15.33
c1	**4**	1	1.431767	0.054892	1.903011	0.75
c1	**5**	1	1.972622	0.073854	1.664189	1.19
c1	**6**	1	-0.094796	-0.004063	1.898700	-0.05
c1	**7**	1	5.971432	0.250037	1.846102	3.23
c1	**8**	0	0	0	.	.
c3	**tiny**	1	-2.919282	-0.072169	2.756295	-1.06
c3	**small**	1	-4.635843	-0.184338	2.218541	-2.09
c3	**average**	1	0.736805	0.038247	1.793059	0.41
c3	**big**	1	-1.078463	-0.063580	1.518927	-0.71
c3	**huge**	0	0	0	.	.
c1*c3	**1 tiny**	1	-2.449964	-0.018632	4.829146	-0.51
c1*c3	**1 small**	1	5.265031	0.069078	3.470382	1.52
c1*c3	**1 average**	1	-3.489735	-0.064365	2.850381	-1.22
c1*c3	**1 big**	1	0.725263	0.017929	2.516502	0.29
c1*c3	**1 huge**	0	0	0	.	.
c1*c3	**2 tiny**	1	5.455122	0.050760	4.209507	1.30
c1*c3	**2 small**	1	7.439196	0.131499	2.982411	2.49
c1*c3	**2 average**	1	-0.739606	-0.014705	2.568876	-0.29
c1*c3	**2 big**	1	3.179351	0.078598	2.247611	1.41
c1*c3	**2 huge**	0	0	0	.	.
c1*c3	**3 tiny**	1	-19.266847	-0.230989	3.784029	-5.09
c1*c3	**3 small**	1	-15.578909	-0.204399	3.266216	-4.77
c1*c3	**3 average**	1	-18.119398	-0.395770	2.529578	-7.16
c1*c3	**3 big**	1	-10.650012	-0.279796	2.205331	-4.83
c1*c3	**3 huge**	0	0	0	.	.
c1*c3	**4 tiny**	0	0	0	.	.
c1*c3	**4 small**	1	4.432753	0.047581	3.677008	1.21
c1*c3	**4 average**	1	-3.976295	-0.091632	2.625564	-1.51
c1*c3	**4 big**	1	-1.306998	-0.033003	2.401064	-0.54
c1*c3	**4 huge**	0	0	0	.	.
c1*c3	**5 tiny**	1	6.714186	0.062475	4.199457	1.60
c1*c3	**5 small**	1	1.565637	0.022165	3.182856	0.49
c1*c3	**5 average**	1	-4.286085	-0.068668	2.749142	-1.56
c1*c3	**5 big**	1	-2.046468	-0.045949	2.282735	-0.90
c1*c3	**5 huge**	0	0	0	.	.
c1*c3	**6 tiny**	1	5.135111	0.039052	4.754845	1.08
c1*c3	**6 small**	1	4.442898	0.081945	3.079524	1.44
c1*c3	**6 average**	1	-2.287870	-0.056559	2.601384	-0.88
c1*c3	**6 big**	1	1.598086	0.043542	2.354326	0.68
c1*c3	**6 huge**	0	0	0	.	.
c1*c3	**7 tiny**	1	1.108451	0.010314	4.267509	0.26
c1*c3	**7 small**	1	7.441059	0.119214	3.135404	2.37
c1*c3	**7 average**	1	1.796483	0.038106	2.630570	0.68

Output 49.2.8 *continued*

Parameter Estimates

Parameter		DF	Estimate	Standardized Estimate	Standard Error	t Value
c1*c3	7 big	1	3.324160	0.095173	2.303369	1.44
c1*c3	7 huge	0	0	0	.	.
c1*c3	8 tiny	0	0	0	.	.
c1*c3	8 small	0	0	0	.	.
c1*c3	8 average	0	0	0	.	.
c1*c3	8 big	0	0	0	.	.
c1*c3	8 huge	0	0	0	.	.
x1		1	2.713527	0.091530	0.836942	3.24
x5		1	2.810341	0.098303	0.816290	3.44
x6		1	4.837022	0.167394	0.810402	5.97
x7		1	5.844394	0.207035	0.793775	7.36
x10		1	2.463916	0.087712	0.794599	3.10
x20		1	4.385924	0.156155	0.787766	5.57

The magnitudes of the standardized estimates and the *t* statistics of the parameters of the effect 'c1' reveal that only levels '3' and '7' of this effect contribute appreciably to the model. This suggests that a more parsimonious model with similar or better predictive power might be obtained if parameters corresponding to the levels of 'c1' are allowed to enter or leave the model independently. You request this with the SPLIT option in the CLASS statement as shown in the following statements:

```
proc glmselect data=analysisData testdata=testData
               seed=1 plots(stepAxis=number)=all;
   partition fraction(validate=0.5);
   class c1(split) c2 c3(order=data);
   model y =  c1|c2|c3|x1|x2|x3|x4|x5|x5|x6|x7|x8|x9|x10
             |x11|x12|x13|x14|x15|x16|x17|x18|x19|x20 @2
           / selection=stepwise(stop   = validate
                                select = sl)
             hierarchy=single;
   output out=outData;
run;
```

The “Class Level Information” and “Dimensions” tables are shown in Output 49.2.9. The “Dimensions” table shows that while the model statement specifies 278 effects, after splitting the parameters corresponding to the levels of c1, there are 439 split effects that are considered for entry or removal at each step of the selection process. Note that the total number of parameters considered is not affected by the split option.

Output 49.2.9 Class Level Information and Problem Dimensions

The GLMSELECT Procedure

Class Level Information

Class	Levels	Values
c1	8	* 1 2 3 4 5 6 7 8
c2	4	0 1 2 3
c3	5	tiny small average big huge

*** Associated Parameters Split**

Output 49.2.9 *continued*

Dimensions	
Number of Effects	278
Number of Effects after Splits	439
Number of Parameters	661

The stop reason and stop details tables are shown in Output 49.2.10. Since the validation ASE is specified as the stopping criterion, the selection stops at step 11, where the validation ASE achieves a local minimum and the model at this step is the selected model.

Output 49.2.10 Stop Details

```
Selection stopped at a local minimum of the residual sum of squares of the validation data.
```

Stop Details

Candidate For	Effect	Candidate Validation ASE		Compare Validation ASE
Entry	x18	25.9851	>	25.7462
Removal	x6*x7	25.7611	>	25.7462

You find details of the selected model in Output 49.2.11. The list of selected effects confirms that parameters corresponding to levels `'3'` and `'7'` only of `c1` are in the selected model. Notice that the selected model with classification variable `c1` split contains 18 parameters, whereas the selected model without splitting `c1` has 45 parameters. Furthermore, by comparing the fit statistics in Output 49.2.7 and Output 49.2.11, you see that this more parsimonious model has smaller prediction errors on both the validation and test data.

Output 49.2.11 Details of the Selected Model

The GLMSELECT Procedure
Selected Model

The selected model is the model at the last step (Step 11).

```
Effects: Intercept c1_3 c1_7 c3 c1_3*c3 x1 x5 x6 x7 x6*x7 x10 x20
```

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value
Model	17	22111	1300.63200	51.88
Error	492	12334	25.06998	
Corrected Total	509	34445		

Output 49.2.11 *continued*

Root MSE	5.00699
Dependent Mean	21.09705
R-Square	0.6419
Adj R-Sq	0.6295
AIC	2172.72685
AICC	2174.27787
SBC	1736.94624
ASE (Train)	24.18515
ASE (Validate)	25.74617
ASE (Test)	22.57297

When you use a PARTITION statement to subdivide the analysis data set, an output data set created with the OUTPUT statement contains a variable named _ROLE_ that shows the role each observation was assigned to. See the section "OUTPUT Statement" on page 3841 and the section "Using Validation and Test Data" on page 3867 for additional details.

The following statements use PROC PRINT to produce Output 49.2.12, which shows the first five observations of the outData data set.

```
proc print data=outData(obs=5);
run;
```

Output 49.2.12 Output Data Set with _ROLE_ Variable

Obs	c3	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12
1	tiny	0.18496	0.97009	0.39982	0.25940	0.92160	0.96928	0.54298	0.53169	0.04979	0.06657	0.81932	0.52387
2	tiny	0.47579	0.84499	0.63452	0.59036	0.58258	0.37701	0.72836	0.50660	0.93121	0.92912	0.58966	0.29722
3	tiny	0.51132	0.43320	0.17611	0.66504	0.40482	0.12455	0.45349	0.19955	0.57484	0.73847	0.43981	0.04937
4	tiny	0.42071	0.07174	0.35849	0.71143	0.18985	0.14797	0.56184	0.27011	0.32520	0.56918	0.04259	0.43921
5	tiny	0.42137	0.03798	0.27081	0.42773	0.82010	0.84345	0.87691	0.26722	0.30602	0.39705	0.34905	0.76593

Obs	x13	x14	x15	x16	x17	x18	x19	x20	c1	c2	y	_ROLE_	p_y
1	0.85339	0.06718	0.95702	0.29719	0.27261	0.68993	0.97676	0.22651	2	1	11.4391	VALIDATE	18.5069
2	0.39104	0.47243	0.67953	0.16809	0.16653	0.87110	0.29879	0.93464	3	1	31.4596	TRAIN	26.2188
3	0.52238	0.34337	0.02271	0.71289	0.93706	0.44599	0.94694	0.71290	4	3	16.4294	VALIDATE	17.0979
4	0.91744	0.52584	0.73182	0.90522	0.57600	0.18794	0.33133	0.69887	5	3	15.4815	VALIDATE	16.1567
5	0.54340	0.61257	0.55291	0.73591	0.37186	0.64565	0.55718	0.87504	6	2	26.0023	TRAIN	24.6358

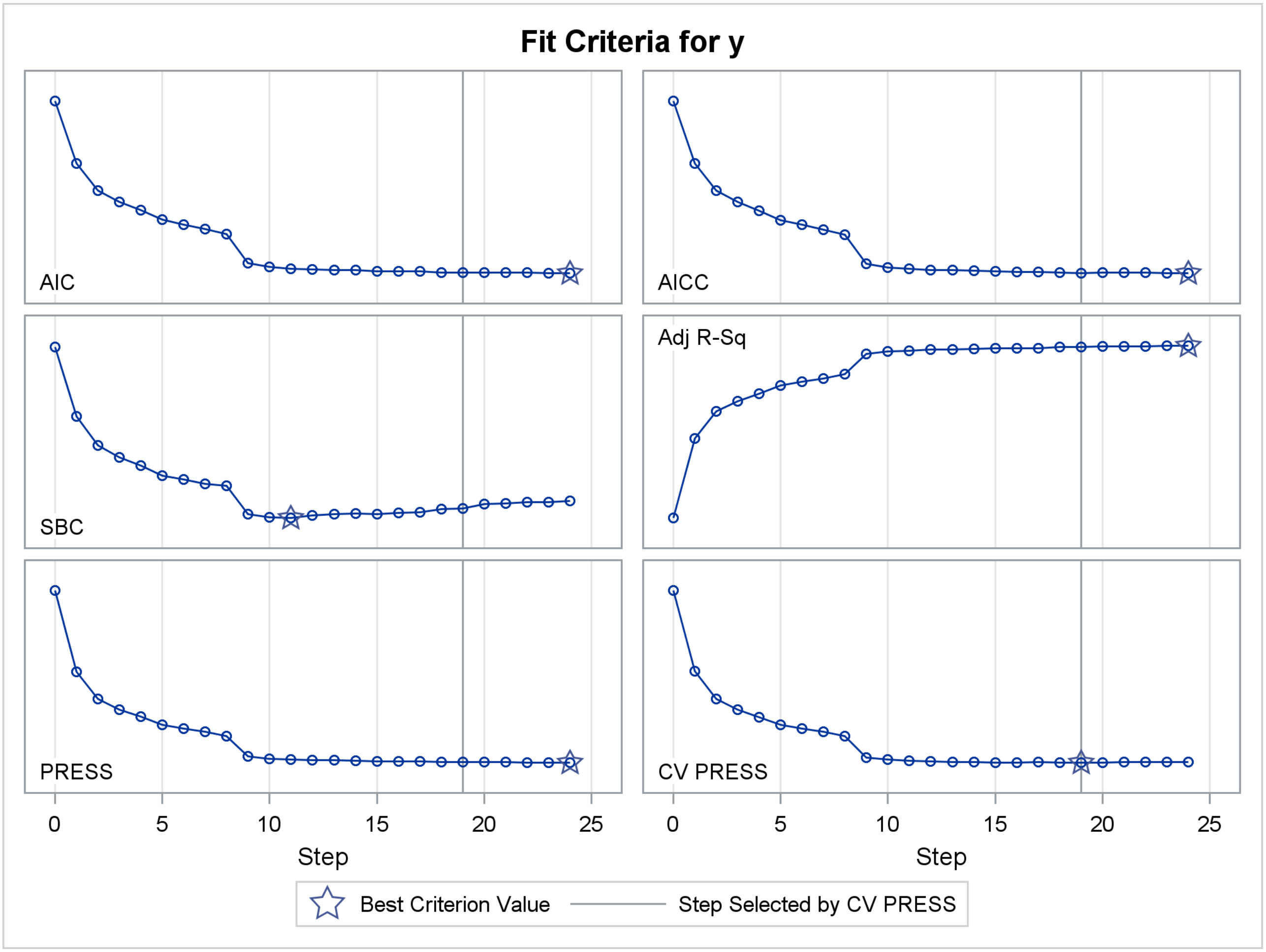
Cross validation is often used to assess the predictive performance of a model, especially for when you do not have enough observations for test set validation. See the section "Cross Validation" on page 3869 for further details. The following statements provide an example where cross validation is used as the CHOOSE= criterion.

```
proc glmselect data=analysisData testdata=testData
               plots(stepAxis=number)=(criterionPanel ASEPlot);
   class c1(split) c2 c3(order=data);
   model y =  c1|c2|c3|x1|x2|x3|x4|x5|x5|x6|x7|x8|x9|x10
             |x11|x12|x13|x14|x15|x16|x17|x18|x19|x20 @2
           / selection = stepwise(choose = cv
                                  select = sl)
             stats     = press
             cvMethod  = split(5)
             cvDetails = all
             hierarchy = single;
   output out=outData;
run;
```

The CVMETHOD=SPLIT(5) option in the MODEL statement requests five-fold cross validation with the five subsets consisting of observations $\{1, 6, 11, \ldots\}$, $\{2, 7, 12, \ldots\}$, and so on. The STATS=PRESS option requests that the leave-one-out cross validation predicted residual sum of squares (PRESS) also be computed and displayed at each step, even though this statistic is not used in the selection process.

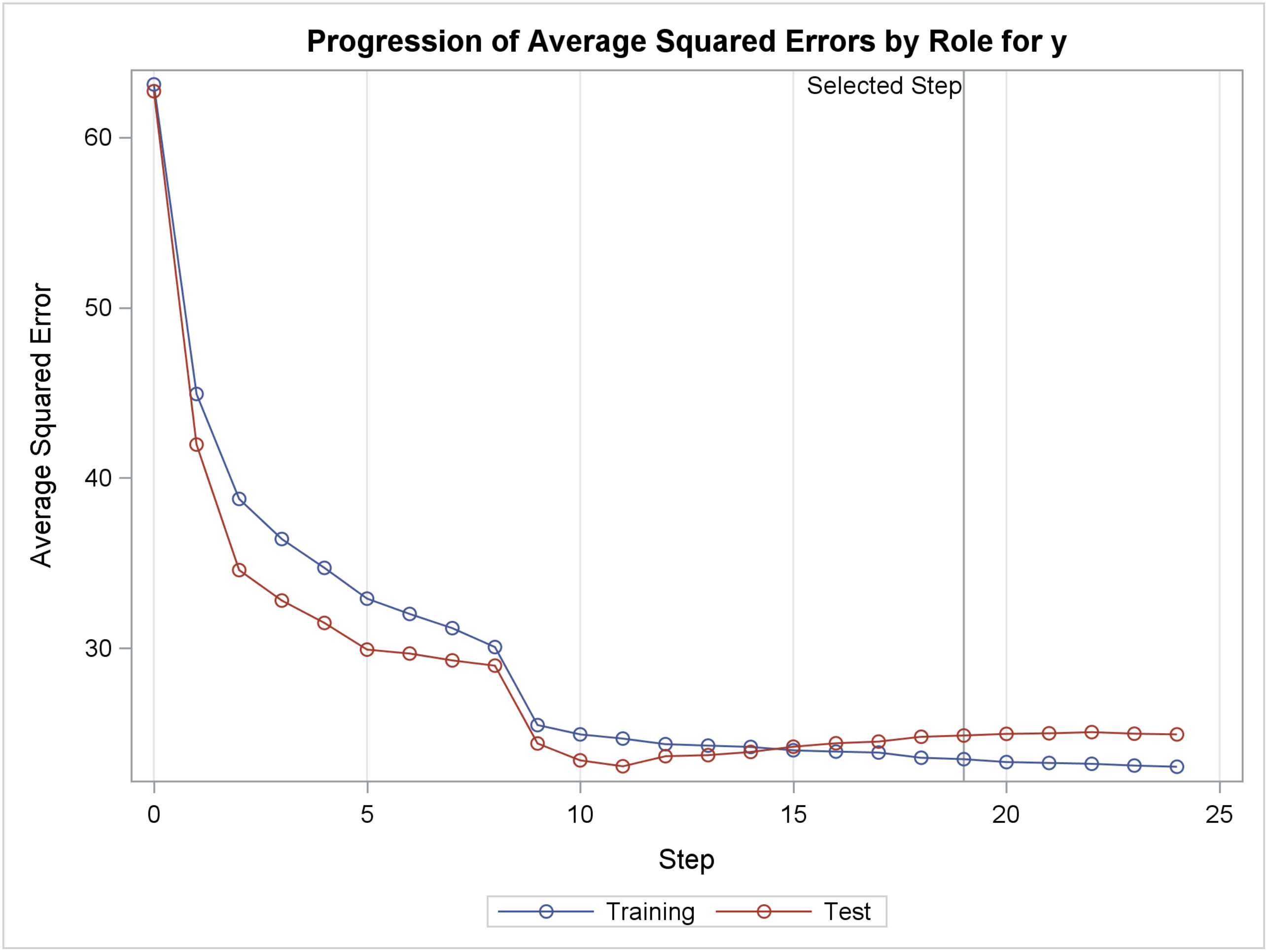
Output 49.2.13 shows how several fit statistics evolved as the selection process progressed. The five-fold CV PRESS statistic achieves its minimum at step 19. Note that this gives a larger model than was selected when the stopping criterion was determined using validation data. Furthermore, you see that the PRESS statistic has not achieved its minimum within 25 steps, so an even larger model would have been selected based on leave-one-out cross validation.

Output 49.2.13 Criterion Panel



Output 49.2.14 shows how the average squared error compares on the test and training data. Note that the ASE error on the test data achieves a local minimum at step 11 and is already slowly increasing at step 19, which corresponds to the selected model.

Output 49.2.14 Average Squared Error Plot



Output 49.2.15 Breakdown of CV Press Statistic by Fold

Cross Validation Details

	Observations		
Index	**Fitted**	**Left Out**	**CV PRESS**
1	808	202	5031.4484
2	808	202	4209.3671
3	808	202	5616.2516
4	808	202	5020.6697
5	808	202	5415.7655
Total			25293.5024

The CVDETAILS=ALL option in the MODEL statement requests the "Cross Validation Details" table in Output 49.2.15 and the cross validation parameter estimates that are included in the "Parameter Estimates" table in Output 49.2.16. For each cross validation index, the predicted residual sum of squares on the observations omitted is shown in the "Cross Validation Details" table and the parameter estimates of the corresponding model are included in the "Parameter Estimates" table. By default, these details are shown for the selected model, but you can request this information at every step with the DETAILS= option in the MODEL statement. You use the _CVINDEX_ variable in the output data set shown in Output 49.2.17 to find out which observations in the analysis data are omitted for each cross validation fold.

Output 49.2.16 Cross Validation Parameter Estimates

Parameter Estimates

		Cross Validation Estimates				
Parameter		**1**	**2**	**3**	**4**	**5**
Intercept		10.7617	10.1200	9.0254	13.4164	12.3352
c1_3		28.2715	27.2977	27.0696	28.6835	27.8070
c1_7		7.6530	7.6445	7.9257	7.4217	7.6862
c3	**tiny**	-3.1103	-4.4041	-5.1793	-8.4131	-7.2096
c3	**small**	2.2039	1.5447	1.0121	-0.3998	1.4927
c3	**average**	0.3021	-1.3939	-1.2201	-3.3407	-2.1467
c3	**big**	-0.9621	-1.2439	-1.6092	-3.7666	-3.4389
c3	**huge**	0	0	0	0	0
c1_3*c3	**tiny**	-21.9104	-21.7840	-22.0173	-22.6066	-21.9791
c1_3*c3	**small**	-20.8196	-20.2725	-19.5850	-20.4515	-20.7586
c1_3*c3	**average**	-16.8500	-15.1509	-15.0134	-15.3851	-13.4339
c1_3*c3	**big**	-12.7212	-12.1554	-12.0354	-12.3282	-13.0174
c1_3*c3	**huge**	0	0	0	0	0
x1		0.9238	1.7286	2.5976	-0.2488	1.2093
x1*c3	**tiny**	-1.5819	-1.1748	-3.2523	-1.7016	-2.7624
x1*c3	**small**	-3.7669	-3.2984	-2.9755	-1.8738	-4.0167
x1*c3	**average**	2.2253	2.4489	1.5675	4.0948	2.0159
x1*c3	**big**	0.9222	0.5330	0.7960	2.6061	1.2694
x1*c3	**huge**	0	0	0	0	0
x5		-1.3562	0.5639	0.3022	-0.4700	-2.5063
x6		-0.9165	-3.2944	-1.2163	-2.2063	-0.5696
x7		5.2295	5.3015	6.2526	4.1770	5.8364
x6*x7		6.4211	7.5644	6.1182	7.0020	5.8730
x10		1.9591	1.4932	0.7196	0.6504	-0.3989
x5*x10		3.6058	1.7274	4.3447	2.4388	3.8967
x15		-0.0079	0.6896	1.6811	0.0136	0.1799
x15*c1_3		-3.5022	-2.7963	-2.6003	-4.2355	-4.7546
x7*x15		-5.1438	-5.8878	-5.9465	-3.6155	-5.3337
x18		-2.1347	-1.5656	-2.4226	-4.0592	-1.4985
x18*c3	**tiny**	2.2988	1.1931	2.6491	6.1615	5.6204
x18*c3	**small**	4.6033	3.2359	4.4183	5.5923	1.7270
x18*c3	**average**	-2.3712	-2.5392	-0.6361	-1.1729	-1.6481
x18*c3	**big**	2.3160	1.4654	2.7683	3.0487	2.5768
x18*c3	**huge**	0	0	0	0	0
x6*x18		3.0716	4.2036	4.1354	4.9196	2.7165
x20		4.1229	4.5773	4.5774	4.6555	4.2655

The following statements display the first eight observations in the outData data set.

```
proc print data=outData(obs=8);
run;
```

Output 49.2.17 First Eight Observations in the Output Data Set

Obs	c3	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12	x13	x14	x15	x16	x17	x18	x19	x20	c1	c2	y	_CVINDEX_	p_y
1	tiny	0.18496	0.97009	0.39982	0.25940	0.92160	0.96928	0.54298	0.53169	0.04979	0.06657	0.81932	0.52387	0.85339	0.06718	0.95702	0.29719	0.27261	0.68993	0.97676	0.22651	2	1	11.4391	1	18.1474
2	tiny	0.47579	0.84499	0.63452	0.59036	0.58258	0.37701	0.72836	0.50660	0.93121	0.92912	0.58966	0.29722	0.39104	0.47243	0.67953	0.16809	0.16653	0.87110	0.29879	0.93464	3	1	31.4596	2	24.7930
3	tiny	0.51132	0.43320	0.17611	0.66504	0.40482	0.12455	0.45349	0.19955	0.57484	0.73847	0.43981	0.04937	0.52238	0.34337	0.02271	0.71289	0.93706	0.44599	0.94694	0.71290	4	3	16.4294	3	16.5752
4	tiny	0.42071	0.07174	0.35849	0.71143	0.18985	0.14797	0.56184	0.27011	0.32520	0.56918	0.04259	0.43921	0.91744	0.52584	0.73182	0.90522	0.57600	0.18794	0.33133	0.69887	5	3	15.4815	4	14.7605
5	tiny	0.42137	0.03798	0.27081	0.42773	0.82010	0.84345	0.87691	0.26722	0.30602	0.39705	0.34905	0.76593	0.54340	0.61257	0.55291	0.73591	0.37186	0.64565	0.55718	0.87504	6	2	26.0023	5	24.7479
6	tiny	0.81722	0.65822	0.02947	0.85339	0.36285	0.37732	0.51054	0.71194	0.37533	0.22954	0.68621	0.55243	0.58182	0.17472	0.04610	0.64380	0.64545	0.09317	0.62008	0.07845	7	1	16.6503	1	21.4444
7	tiny	0.19480	0.81673	0.08548	0.18376	0.33264	0.70558	0.92761	0.29642	0.22404	0.14719	0.59064	0.46326	0.41860	0.25631	0.23045	0.08034	0.43559	0.67020	0.42272	0.49827	1	1	14.0342	2	20.9661
8	tiny	0.04403	0.51697	0.68884	0.45333	0.83565	0.29745	0.40325	0.95684	0.42194	0.78079	0.33106	0.17210	0.91056	0.26897	0.95602	0.13720	0.27190	0.55692	0.65825	0.68465	2	3	14.9830	3	17.5644

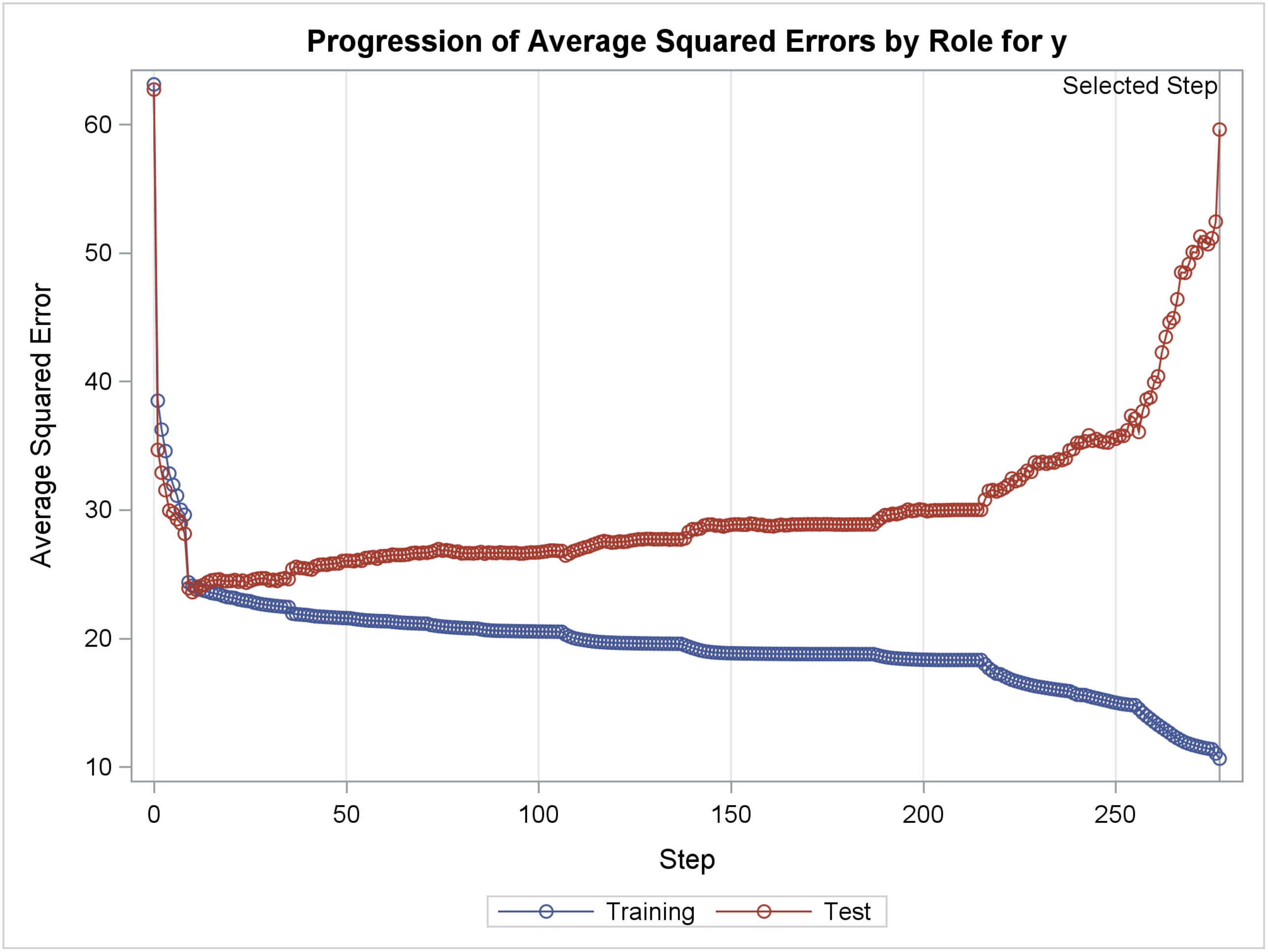
This example demonstrates the usefulness of effect selection when you suspect that interactions of effects are needed to explain the variation in your dependent variable. Ideally, a priori knowledge should be used to decide what interactions to allow, but in some cases this information might not be available. Simply fitting a least squares model allowing all interactions produces a model that overfits your data and generalizes very poorly.

The following statements use forward selection with selection based on the SBC criterion, which is the default selection criterion. At each step, the effect whose addition to the model yields the smallest SBC value is added. The STOP=NONE suboption specifies that this process continue even when the SBC statistic grows whenever an effect is added, and so it terminates at a full least squares model. The BUILDSSCP=FULL option is specified in a PERFORMANCE statement, since building the SSCP matrix incrementally is counterproductive in this case. See the section "BUILDSSCP=FULL | INCREMENTAL" on page 3844 for details. Note that if all you are interested in is a full least squares model, then it is much more efficient to simply specify SELECTION=NONE in the MODEL statement. However, in this example the aim is to add effects in roughly increasing order of explanatory power.

```
proc glmselect data=analysisData testdata=testData plots=ASEPlot;
   class c1 c2 c3(order=data);
   model y =  c1|c2|c3|x1|x2|x3|x4|x5|x5|x6|x7|x8|x9|x10
             |x11|x12|x13|x14|x15|x16|x17|x18|x19|x20 @2
           / selection=forward(stop=none)
             hierarchy=single;
   performance buildSSCP = full;
run;

ods graphics off;
```

The ASE plot shown in Output 49.2.18 clearly demonstrates the danger in overfitting the training data. As more insignificant effects are added to the model, the growth in test set ASE shows how the predictions produced by the resulting models worsen. This decline is particularly rapid in the latter stages of the forward selection, because the use of the SBC criterion results in insignificant effects with lots of parameters being added after insignificant effects with fewer parameters.

Output 49.2.18 Average Squared Error Plot



Example 49.3: Scatter Plot Smoothing by Selecting Spline Functions

This example shows how you can use model selection to perform scatter plot smoothing. It illustrates how you can use the experimental EFFECT statement to generate a large collection of B-spline basis functions from which a subset is selected to fit scatter plot data.

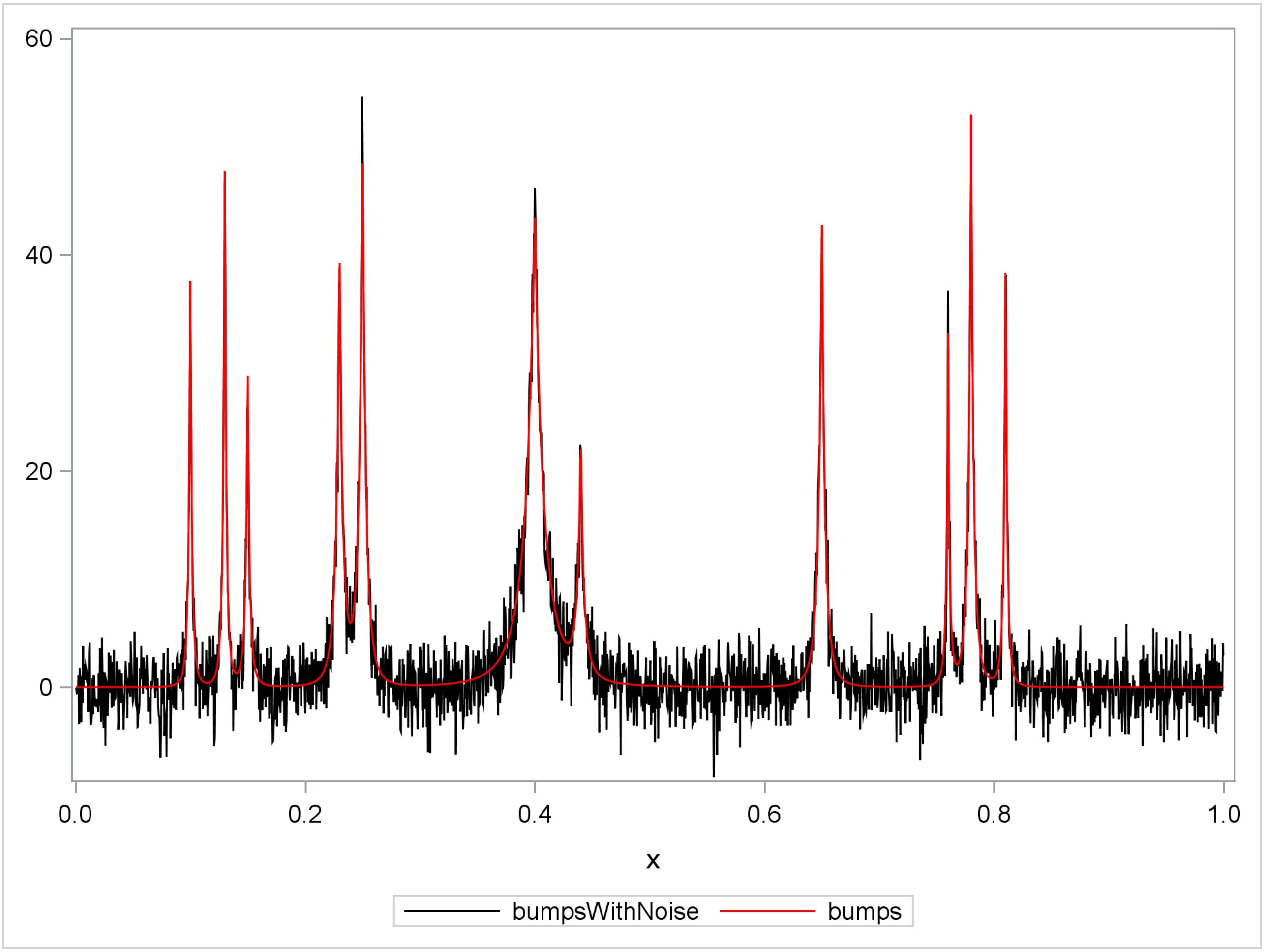
The data for this example come from a set of benchmarks developed by Donoho and Johnstone (1994) that have become popular in the statistics literature. The particular benchmark used is the "Bumps" functions to which random noise has been added to create the test data. The following DATA step, extracted from Sarle (2001), creates the data. The constants are chosen so that the noise-free data have a standard deviation of 7. The standard deviation of the noise is $\sqrt{5}$, yielding bumpsNoise with a signal-to-noise ratio of 3.13 ($7/\sqrt{5}$).

```
%let random=12345;

data DoJoBumps;
   keep x bumps bumpsWithNoise;

   pi = arcos(-1);

   do n=1 to 2048;
      x=(2*n-1)/4096;
      link compute;
      bumpsWithNoise=bumps+rannor(&random)*sqrt(5);
      output;
   end;
   stop;

compute:
   array t(11) _temporary_ (.1 .13 .15 .23 .25 .4 .44 .65 .76 .78 .81);
   array b(11) _temporary_ (   4    5    3    4    5 4.2 2.1 4.3  3.1  5.1  4.2);
   array w(11) _temporary_ (.005 .005 .006 .01 .01 .03 .01 .01 .005 .008 .005);

   bumps=0;
   do i=1 to 11;
      bumps=bumps+b[i]*(1+abs((x-t[i])/w[i]))**-4;
   end;
   bumps=bumps*10.528514619;
   return;
run;
```

The following statements use the SGPLOT procedure to produce the plot in Output 49.3.1. The plot shows the bumps function superimposed on the function with added noise.

```
proc sgplot data=DoJoBumps;
   yaxis display=(nolabel);
   series  x=x y=bumpsWithNoise/lineattrs=(color=black);
   series  x=x y=bumps/lineattrs=(color=red);
run;
```

Output 49.3.1 Donoho-Johnstone Bumps Function

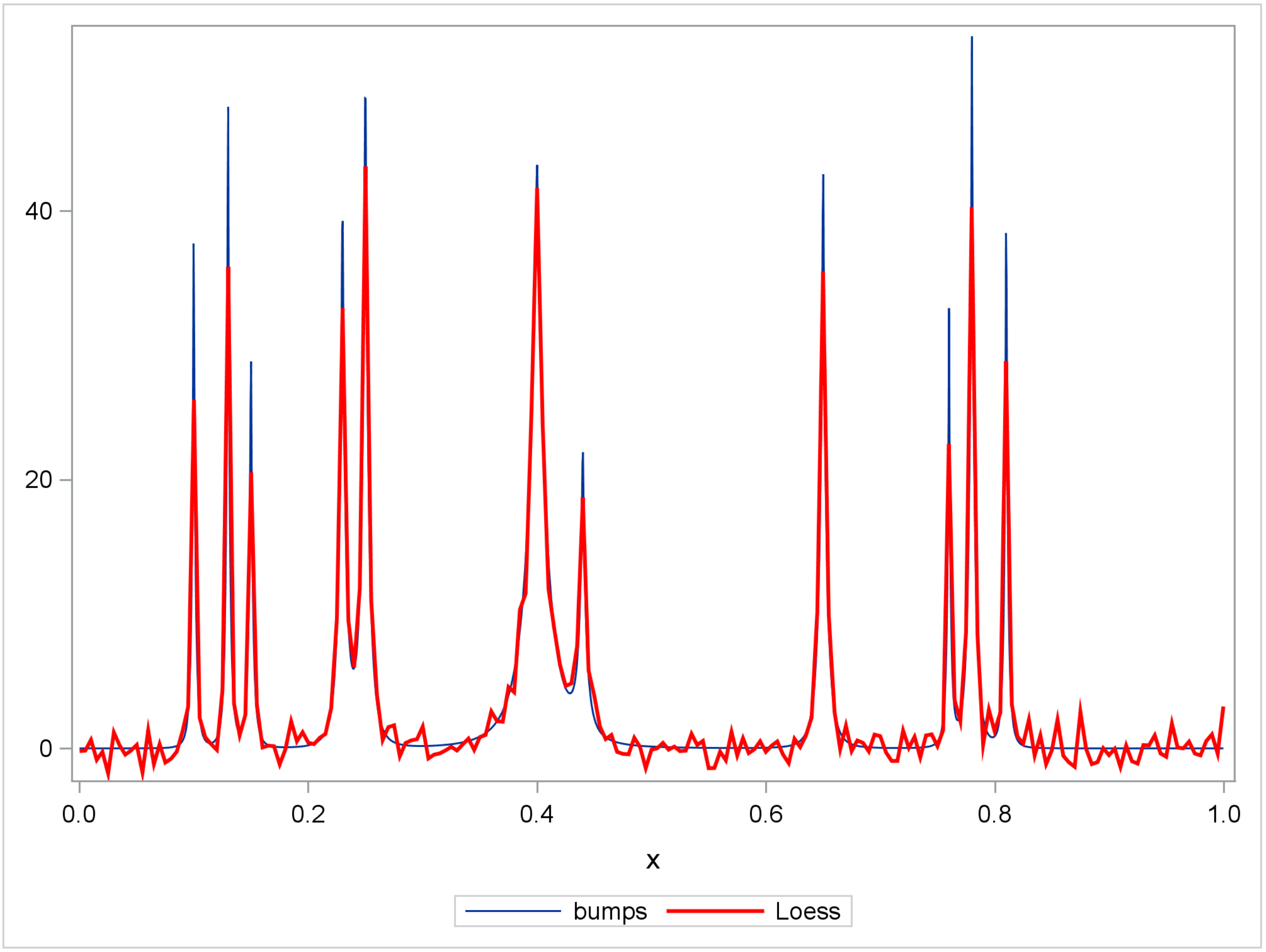


Suppose you want to smooth the noisy data to recover the underlying function. This problem is studied by Sarle (2001), who shows how neural nets can be used to perform the smoothing. The following statements use the LOESS statement in the SGPLOT procedure to show a loess fit superimposed on the noisy data (Output 49.3.2). (See Chapter 71, "The LOESS Procedure," for information about the loess method.)

```
proc sgplot data=DoJoBumps;
   yaxis display=(nolabel);
   series x=x y=bumps;
   loess  x=x y=bumpsWithNoise / lineattrs=(color=red) nomarkers;
run;
```

The algorithm selects a smoothing parameter that is small enough to enable bumps to be resolved. Because there is a single smoothing parameter that controls the number of points for all local fits, the loess method undersmooths the function in the intervals between the bumps.

Output 49.3.2 Loess Fit



Another approach to doing nonparametric fitting is to approximate the unknown underlying function as a linear combination of a set of basis functions. Once you specify the basis functions, then you can use least squares regression to obtain the coefficients of the linear combination. A problem with this approach is that for most data, you do not know a priori what set of basis functions to use. You need to supply a sufficiently rich set to enable the features in the data to be approximated. However, if you use too rich a set of functions, then this approach yields a fit that undersmooths the data and captures spurious features in the noise.

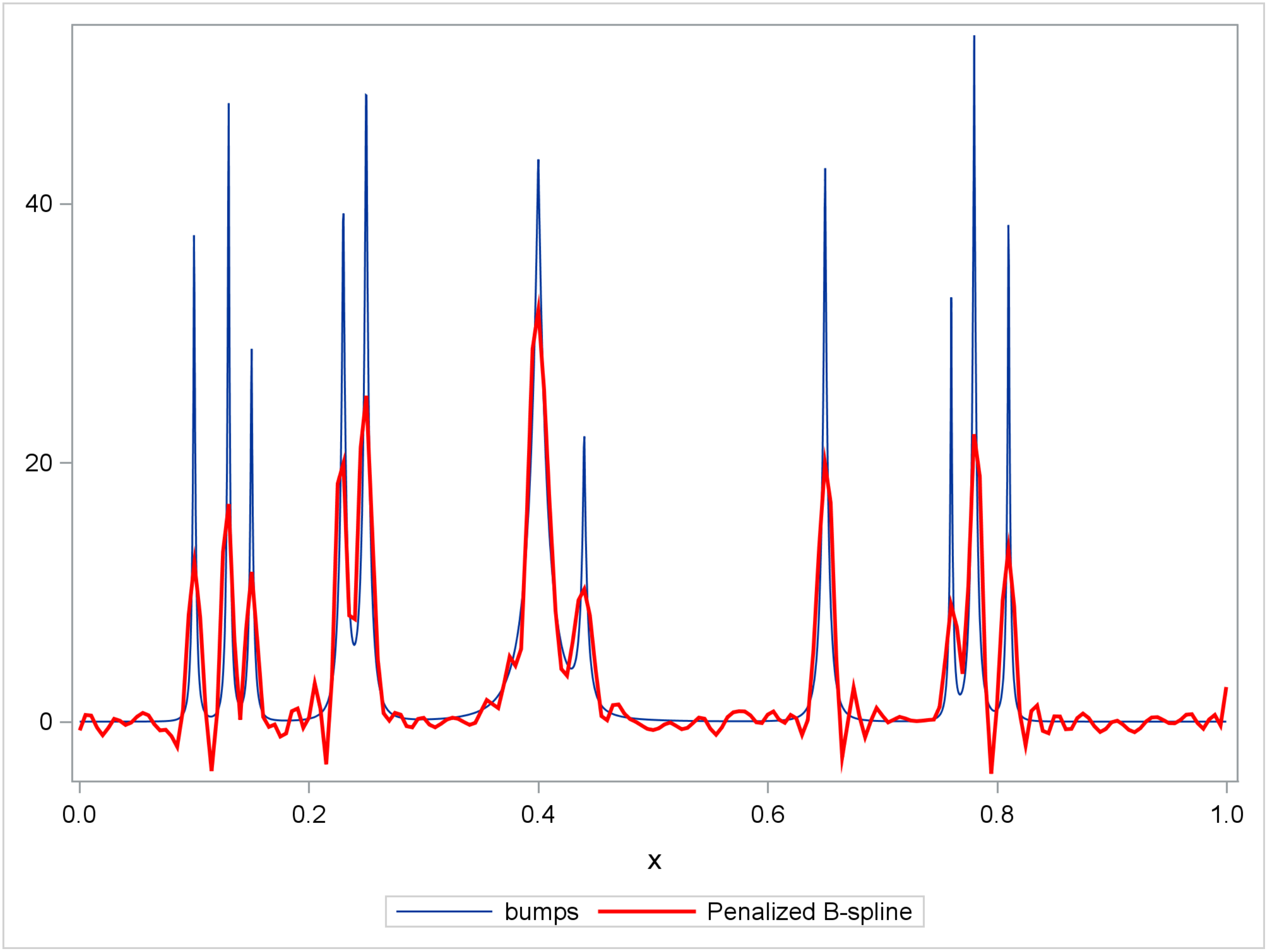
The penalized B-spline method (Eilers and Marx 1996) uses a basis of B-splines (see the section "EFFECT Statement" on page 401 in Chapter 19, "Shared Concepts and Topics") corresponding to a large number of equally spaced knots as the set of approximating functions. To control the potential overfitting, their algorithm modifies the least squares objective function to include a penalty term that grows with the complexity of the fit.

The following statements use the PBSPLINE statement in the SGPLOT procedure to show a penalized B-spline fit superimposed on the noisy data (Output 49.3.3). See Chapter 117, "The TRANSREG Procedure," for details about the implementation of the penalized B-spline method.

```
proc sgplot data=DoJoBumps;
   yaxis display=(nolabel);
   series    x=x y=bumps;
   pbspline  x=x y=bumpsWithNoise /
                lineattrs=(color=red) nomarkers;
run;
```

As in the case of loess fitting, you see undersmoothing in the intervals between the bumps because there is only a single smoothing parameter that controls the overall smoothness of the fit.

Output 49.3.3 Penalized B-spline Fit



An alternative to using a smoothness penalty to control the overfitting is to use variable selection to obtain an appropriate subset of the basis functions. In order to be able to represent features in the data that occur at multiple scales, it is useful to select from B-spline functions defined on just a few knots to capture large scale features of the data as well as B-spline functions defined on many knots to capture fine details of the data. The following statements show how you can use PROC GLMSELECT to implement this strategy:

```
proc glmselect data=dojoBumps;
   effect spl = spline(x / knotmethod=multiscale(endscale=8)
                             split details);
   model bumpsWithNoise=spl;
   output out=out1 p=pBumps;
run;

proc sgplot data=out1;
   yaxis display=(nolabel);
   series    x=x y=bumps;
   series    x=x y=pBumps / lineattrs=(color=red);
run;
```

The KNOTMETHOD=MULTISCALE suboption of the `EFFECT spl = SPLINE` statement provides a convenient way to generate B-spline basis functions at multiple scales. The ENDSCALE=8 option requests that the finest scale use B-splines defined on 2^8 equally spaced knots in the interval $[0, 1]$. Because the cubic B-splines are nonzero over five adjacent knots, at the finest scale, the support of each B-spline basis function is an interval of length about 0.02 (5/256), enabling the bumps in the underlying data to be resolved. The default value is ENDSCALE=7. At this scale you will still be able to capture the bumps, but with less sharp resolution. For these data, using a value of ENDSCALE= greater than eight provides unneeded resolution, making it more likely that basis functions that fit spurious features in the noise are selected.

Output 49.3.4 shows the model information table. Since no options are specified in the MODEL statement, PROC GLMSELECT uses the stepwise method with selection and stopping based on the SBC criterion.

Output 49.3.4 Model Settings

The GLMSELECT Procedure

Data Set	WORK.DOJOBUMPS
Dependent Variable	bumpsWithNoise
Selection Method	Stepwise
Select Criterion	SBC
Stop Criterion	SBC
Effect Hierarchy Enforced	None

The DETAILS suboption in the EFFECT statement requests the display of spline knots and spline basis tables. These tables contain information about knots and basis functions at all scales. The results for scale four are shown in Output 49.3.5 and Output 49.3.6.

Output 49.3.5 Spline Knots

Knots for Spline Effect spl

Knot Number	Scale	Scale Knot Number	Boundary	x
40	4	1	*	-0.11735
41	4	2	*	-0.05855
42	4	3	*	0.00024414
43	4	4		0.05904
44	4	5		0.11783
45	4	6		0.17663
46	4	7		0.23542
47	4	8		0.29422
48	4	9		0.35301
49	4	10		0.41181
50	4	11		0.47060
51	4	12		0.52940
52	4	13		0.58819
53	4	14		0.64699
54	4	15		0.70578
55	4	16		0.76458
56	4	17		0.82337
57	4	18		0.88217
58	4	19		0.94096
59	4	20	*	0.99976
60	4	21	*	1.05855
61	4	22	*	1.11735

Output 49.3.6 Spline Details

Basis Details for Spline Effect spl

Column	Scale	Scale Column	Support		Support Knots
32	4	1	-0.11735	0.05904	1-4
33	4	2	-0.11735	0.11783	1-5
34	4	3	-0.05855	0.17663	2-6
35	4	4	0.00024414	0.23542	3-7
36	4	5	0.05904	0.29422	4-8
37	4	6	0.11783	0.35301	5-9
38	4	7	0.17663	0.41181	6-10
39	4	8	0.23542	0.47060	7-11
40	4	9	0.29422	0.52940	8-12
41	4	10	0.35301	0.58819	9-13
42	4	11	0.41181	0.64699	10-14
43	4	12	0.47060	0.70578	11-15
44	4	13	0.52940	0.76458	12-16
45	4	14	0.58819	0.82337	13-17
46	4	15	0.64699	0.88217	14-18
47	4	16	0.70578	0.94096	15-19
48	4	17	0.76458	0.99976	16-20
49	4	18	0.82337	1.05855	17-21
50	4	19	0.88217	1.11735	18-22
51	4	20	0.94096	1.11735	19-22

The Dimensions table in Output 49.3.7 shows that at each step of the selection process, 548 effects are considered as candidates for entry or removal. Note that although the MODEL statement specifies a single constructed effect spl, the SPLIT suboption causes each of the parameters in this constructed effect to be treated as an individual effect.

Output 49.3.7 Dimensions

Dimensions	
Number of Effects	548
Number of Parameters	548

Output 49.3.8 shows the parameter estimates for the selected model. You can see that the selected model contains 31 B-spline basis functions and that all the selected B-spline basis functions are from scales four though eight. For example, the first basis function listed in the parameter estimates table is spl_S4:9—the ninth B-spline function at scale 4. You see from Output 49.3.6 that this function is nonzero on the interval $(0.29, 0.52)$.

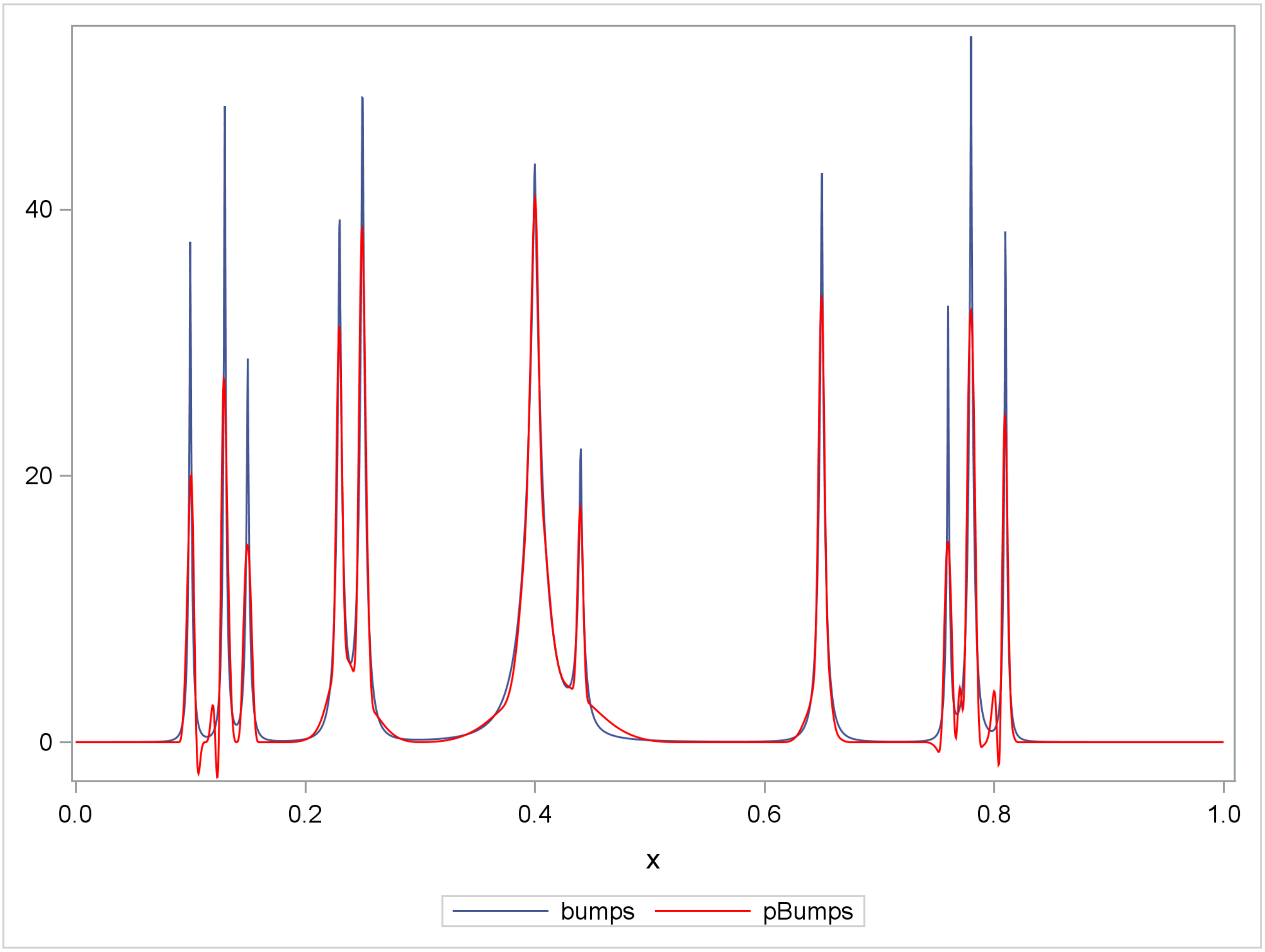
Output 49.3.8 Parameter Estimates

Parameter Estimates

Parameter	DF	Estimate	Standard Error	t Value
Intercept	1	-0.009039	0.077412	-0.12
spl_S4:9	1	7.070207	0.586990	12.04
spl_S5:10	1	5.323121	1.199824	4.44
spl_S6:17	1	5.222808	1.728910	3.02
spl_S6:28	1	24.562103	1.490639	16.48
spl_S6:44	1	4.930829	1.243552	3.97
spl_S6:52	1	-7.046308	2.487700	-2.83
spl_S7:86	1	9.592742	2.626471	3.65
spl_S7:106	1	16.268550	3.334015	4.88
spl_S8:27	1	10.626586	1.752152	6.06
spl_S8:28	1	27.882444	2.004520	13.91
spl_S8:29	1	-6.129939	1.752151	-3.50
spl_S8:33	1	5.855648	1.766912	3.31
spl_S8:34	1	-11.782303	2.092484	-5.63
spl_S8:35	1	38.705178	2.092486	18.50
spl_S8:36	1	13.823256	1.766916	7.82
spl_S8:40	1	15.975124	1.691679	9.44
spl_S8:41	1	14.898716	1.691679	8.81
spl_S8:61	1	37.441965	2.084375	17.96
spl_S8:66	1	47.484506	1.883409	25.21
spl_S8:67	1	16.811502	1.910358	8.80
spl_S8:104	1	11.098484	1.958676	5.67
spl_S8:105	1	26.704556	2.042735	13.07
spl_S8:115	1	21.102920	1.576185	13.39
spl_S8:169	1	36.572294	2.914521	12.55
spl_S8:197	1	20.869716	1.882529	11.09
spl_S8:198	1	16.210987	2.693183	6.02
spl_S8:200	1	13.113942	3.458187	3.79
spl_S8:202	1	38.463549	2.462314	15.62
spl_S8:203	1	34.164644	1.757908	19.43
spl_S8:209	1	-22.645471	3.598587	-6.29
spl_S8:210	1	29.024741	2.557567	11.35

The OUTPUT statement captures the predicted values in a data set named out1, and Output 49.3.9 shows a fit plot produced by PROC SGPLOT.

Output 49.3.9 Fit by Selecting B-splines



Example 49.4: Multimember Effects and the Design Matrix

This example shows how you can use multimember effects to build predictive models. It also demonstrates several features of the OUTDESIGN= option in the PROC GLMSELECT statement.

The simulated data for this example describe a two-week summer tennis camp. The tennis ability of each camper was assessed and ratings were assigned at the beginning and end of the camp. The camp consisted of supervised group instruction in the mornings with a number of different options in the afternoons. Campers could elect to participate in unsupervised practice and play. Some campers paid for one or more individual lessons from 30 to 90 minutes in length, focusing on forehand and backhand strokes and volleying. The aim of this example is to build a predictive model for the rating improvement of each camper based on the times the camper spent doing each activity and several other variables, including the age, gender, and initial rating of the camper.

The following statements produce the TennisCamp data set:

```
data TennisCamp;
   length forehandCoach $6 backhandCoach $6 volleyCoach $6 gender $1;
   input forehandCoach backhandCoach volleyCoach tLessons tPractice tPlay
         gender inRating nPastCamps age tForehand tBackhand tVolley
         improvement;
   label forehandCoach = "Forehand lesson coach"
         backhandCoach = "Backhand lesson coach"
         volleyCoach   = "Volley   lesson coach"
         tForehand     = "time (1/2 hours) of forehand lesson"
         tBackhand     = "time (1/2 hours) of backhand lesson"
         tVolley       = "time (1/2 hours) of volley lesson"
         tLessons      = "time (1/2 hours) of all lessons"
         tPractice     = "total practice time (hours)"
         tPlay         = "total play time (hours)"
         nPastCamps    = "Number of previous camps attended"
         age           = "age (years)"
         inRating      = "Rating at camp start"
         improvement   = "Rating improvement at end of camp";
   datalines;
.        .        Tom      1   30   19   f   44   0   13   0   0   1    6
Greg     .        .        2   12   33   f   48   2   15   2   0   0   14
.        .        Mike     2   12   24   m   53   0   15   0   0   2   13
.        Mike     .        1   12   28   f   48   0   13   0   1   0   11
.        Bruna    .        2   13   34   f   57   0   16   0   2   0   12

   ... more lines ...

.        .        .        0   12   38   m   47   1   15   0   0   0    8
Greg     Tom      Tom      6    3   41   m   48   2   15   2   1   3   19
.        Greg     Mike     5   30   16   m   52   0   13   0   2   3   18
;
```

A multimember effect (see the section “EFFECT Statement” on page 401 in Chapter 19, “Shared Concepts and Topics”) is appropriate for modeling the effect of coaches on the campers’ improvement, because campers might have worked with multiple coaches. Furthermore, since the time a coach spent with each camper varies, it is appropriate to use these times to weight each coach’s contribution in the multimember effect. It is also important not to exclude campers from the analysis if they did not receive any individual instruction. You can accomplish all these goals by using a multimember effect defined as follows:

```
class forehandCoach backhandCoach volleyCoach;
effect coach = MM(forehandCoach backhandCoach volleyCoach/ noeffect
                  weight=(tForehand tBackhand tVolley));
```

Based on similar previous studies, it is known that the time spent practicing should not be included linearly, because there are diminishing returns and perhaps even counterproductive effects beyond about 25 hours. A spline effect with a single knot at 25 provides flexibility in modeling effect of practice time.

The following statements use PROC GLMSELECT to select effects for the model.

```
proc glmselect data=TennisCamp outdesign=designCamp;
   class forehandCoach backhandCoach volleyCoach gender;

   effect coach    = mm(forehandCoach backhandCoach volleyCoach / noeffect
                        details weight=(tForehand tBackhand tVolley));
   effect practice = spline(tPractice/knotmethod=list(25) details);

   model improvement = coach practice tLessons tPlay age gender
                       inRating nPastCamps;
run;
```

Output 49.4.1 shows the class level and MM level information. The levels of the constructed MM effect are the union of the levels of its constituent classification variables. The MM level information is not displayed by default—you request this table by specifying the DETAILS suboption in the relevant EFFECT statement.

Output 49.4.1 Levels of MM EFFECT Coach

The GLMSELECT Procedure

Class Level Information

Class	Levels	Values
forehandCoach	5	Bruna Elaine Greg Mike Tom
backhandCoach	5	Bruna Elaine Greg Mike Tom
volleyCoach	5	Andy Bruna Greg Mike Tom
gender	2	f m

The GLMSELECT Procedure

Level Details for MM Effect coach

Levels	Values
6	Andy Bruna Elaine Greg Mike Tom

Output 49.4.2 shows the parameter estimates for the selected model. You can see that the constructed multimember effect coach and the spline effect practice are both included in the selected model. All coaches provided benefit (all the parameters of the multimember effect coach are positive), with Greg and Mike being the most effective.

Output 49.4.2 Parameter Estimates

Parameter Estimates

Parameter		DF	Estimate	Standard Error	t Value
Intercept		1	0.379873	0.513431	0.74
coach	Andy	1	1.444370	0.318078	4.54
coach	Bruna	1	1.446063	0.110179	13.12
coach	Elaine	1	1.312290	0.281877	4.66
coach	Greg	1	3.042828	0.112256	27.11
coach	Mike	1	2.840728	0.121166	23.45
coach	Tom	1	1.248946	0.115266	10.84
practice	1	1	2.538938	1.015772	2.50
practice	2	1	3.837684	1.104557	3.47
practice	3	1	2.574775	0.930816	2.77
practice	4	1	-0.034747	0.717967	-0.05
practice	5	0	0	.	.
tPlay		1	0.139409	0.023043	6.05

Suppose you want to examine regression diagnostics for the selected model. PROC GLMSELECT does not support such diagnostics, so you might want to use the REG procedure to produce these diagnostics. You can overcome the difficulty that PROC REG does not support CLASS and EFFECT statements by using the OUTDESIGN= option in the PROC GLMSELECT statement to obtain the design matrix that you can use as an input data set for further analysis with other SAS procedures.

The following statements use PROC PRINT to produce Output 49.4.3, which shows the first five observations of the design matrix designCamp.

```
proc print data=designCamp(obs=5);
run;
```

Output 49.4.3 First Five Observations of the designCamp Data Set

Obs	Intercept	coach_Andy	coach_Bruna	coach_Elaine	coach_Greg	coach_Mike	coach_Tom	tPlay
1	1	0	0	0	0	0	1	19
2	1	0	0	0	2	0	0	33
3	1	0	0	0	0	2	0	24
4	1	0	0	0	0	1	0	28
5	1	0	2	0	0	0	0	34

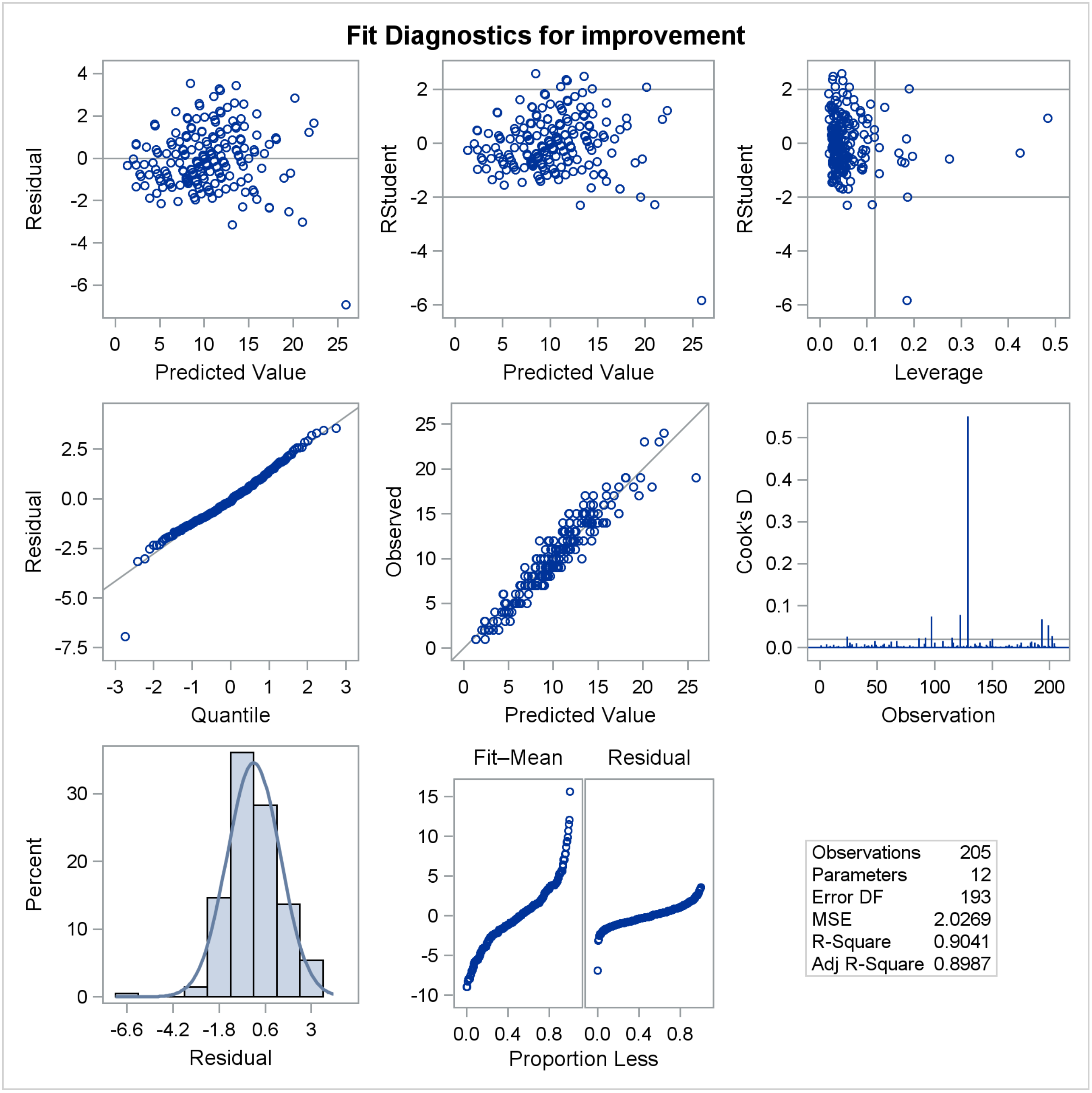
Obs	practice_1	practice_2	practice_3	practice_4	practice_5	improvement
1	0.00000	0.00136	0.05077	0.58344	0.36443	6
2	0.20633	0.50413	0.25014	0.03940	0.00000	14
3	0.20633	0.50413	0.25014	0.03940	0.00000	13
4	0.20633	0.50413	0.25014	0.03940	0.00000	11
5	0.16228	0.49279	0.29088	0.05405	0.00000	12

To facilitate specifying the columns of the design matrix corresponding to the selected model, you can use the macro variable named _GLSMOD that PROC GLMSELECT creates whenever you specify the OUTDESIGN= option. The following statements use PROC REG to produce a panel of regression diagnostics corresponding to the model selected by PROC GLMSELECT.

```
ods graphics on;

proc reg data=designCamp;
   model improvement = &_GLSMOD;
quit;

ods graphics off;
```

The regression diagnostics shown in Output 49.4.4 indicate a reasonable model. However, they also reveal the presence of one large outlier and several influential observations that you might want to investigate.

Output 49.4.4 Fit Diagnostics



Sometimes you might want to use subsets of the columns of the design matrix. In such cases, it might be convenient to produce a design matrix with generic names for the columns. You might also want a design matrix containing the columns corresponding to the full model that you specify in the MODEL statement. By default, the design matrix includes only the columns that correspond to effects in the selected model. The following statements show how to do this.

```
proc glmselect data=TennisCamp
     outdesign(fullmodel prefix=parm names)=designCampGeneric;
   class forehandCoach backhandCoach volleyCoach gender;

   effect coach    = mm(forehandCoach backhandCoach volleyCoach / noeffect
                        details weight=(tForehand tBackhand tVolley));
   effect practice = spline(tPractice/knotmethod=list(25) details);

   model improvement = coach practice tLessons tPlay age gender
                       inRating nPastCamps;
run;
```

The PREFIX=parm suboption of the OUTDESIGN= option specifies that columns in the design matrix be given the prefix parm with a trailing index. The NAMES suboption requests the table in Output 49.4.5 that associates descriptive labels with the names of columns in the design matrix. Finally, the FULLMODEL suboption specifies that the design matrix include columns corresponding to all effects specified in the MODEL statement.

Output 49.4.5 Descriptive Names of Design Matrix Columns

The GLMSELECT Procedure
Selected Model

Parameter Names

Name	Parameter
parm1	Intercept
parm2	coach Andy
parm3	coach Bruna
parm4	coach Elaine
parm5	coach Greg
parm6	coach Mike
parm7	coach Tom
parm8	practice 1
parm9	practice 2
parm10	practice 3
parm11	practice 4
parm12	practice 5
parm13	tLessons
parm14	tPlay
parm15	age
parm16	gender f
parm17	gender m
parm18	inRating
parm19	nPastCamps

The following statements produce Output 49.4.6, displaying the first five observations of the designCampGeneric data set:

```
proc print data=designCampGeneric(obs=5);
run;
```

Output 49.4.6 First Five Observations of designCampGeneric Data Set

Obs	parm1	parm2	parm3	parm4	parm5	parm6	parm7	parm8	parm9	parm10	parm11	parm12
1	1	0	0	0	0	0	1	0.00000	0.00136	0.05077	0.58344	0.36443
2	1	0	0	0	2	0	0	0.20633	0.50413	0.25014	0.03940	0.00000
3	1	0	0	0	0	2	0	0.20633	0.50413	0.25014	0.03940	0.00000
4	1	0	0	0	0	1	0	0.20633	0.50413	0.25014	0.03940	0.00000
5	1	0	2	0	0	0	0	0.16228	0.49279	0.29088	0.05405	0.00000

Obs	parm13	parm14	parm15	parm16	parm17	parm18	parm19	improvement
1	1	19	13	1	0	44	0	6
2	2	33	15	1	0	48	2	14
3	2	24	15	0	1	53	0	13
4	1	28	13	1	0	48	0	11
5	2	34	16	1	0	57	0	12

Example 49.5: Model Averaging

This example shows how you can combine variable selection methods with model averaging to build parsimonious predictive models. This example uses simulated data that consist of observations from the model

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + N(0, \sigma^2)$$

where $\mathbf{X}$ is drawn from a multivariate normal distribution $N(0, \mathbf{V})$ with $V_{i,j} = \rho^{|i-j|}$ where $0 < \rho < 1$. This setup has been widely studied in investigations of variable selection methods. For examples, see Breiman (1992); Tibshirani (1996); Zou (2006).

The following statements define a macro that uses the SIMNORMAL procedure to generate the regressors. This macro prepares a TYPE=CORR data set that specifies the desired pairwise correlations. This data set is used as the input data for PROC SIMNORMAL which produces the sampled regressors in an output data set named Regressors.

```
%macro makeRegressorData(nObs=100,nVars=8,rho=0.5,seed=1);
   data varCorr;
      drop i j;
      array x{&nVars};
      length  _NAME_ $8   _TYPE_ $8;
      _NAME_ = '';

      _TYPE_ = 'MEAN';
      do j=1 to &nVars; x{j}=0; end;
      output;

      _TYPE_ = 'STD';
      do j=1 to &nVars; x{j}=1;end;
      output;

      _TYPE_ = 'N';
      do j=1 to &nVars; x{j}=10000;end;
      output;

      _TYPE_ = 'CORR';
      do i=1 to &nVars;
         _NAME_="x" || trim(left(i));
         do j= 1 to &nVars;
            x{j}=&rho**(abs(i-j));
         end;
         output;
      end;
   run;

   proc simnormal data=varCorr(type=corr) out=Regressors
                  numReal=&nObs seed=&seed;
      var x1-x&nVars;
   run;
%mend;
```

The following statements use the %makeRegressorData macro to generate a sample of 100 observations with 10 regressors, where $E(\mathbf{X}_i\mathbf{X}_j) = 0.5^{|i-j|}$, the true coefficients are $\boldsymbol{\beta}' = (3, 1.5, 0, 0, 2, 0, 0, 0, 0, 0)$, and $\sigma = 3$.

```
%makeRegressorData(nObs=100,nVars=10,rho=0.5);

data simData;
   set regressors;
   yTrue = 3*x1 + 1.5*x2 + 2*x5;
   y     = yTrue + 3*rannor(2);
run;
```

The adaptive LASSO algorithm (see "Adaptive LASSO Selection" on page 3852) is a modification of the standard LASSO algorithm in which weights are applied to each of the parameters in forming the LASSO constraint. Zou (2006) shows that the adaptive LASSO has theoretical advantages over the standard LASSO. Furthermore, simulation studies show that the adaptive LASSO tends to perform better than the standard LASSO in selecting the correct regressors, particularly in high signal-to-noise ratio cases. The following statements fit an adaptive LASSO model to the simData data:

```
proc glmselect data=simData;
   model y=x1-x10/selection=LASSO(adaptive stop=none choose=sbc);
run;
```

The selected model and parameter estimates are shown in Output 49.5.1

Output 49.5.1 Model Selected by Adaptive Lasso

The GLMSELECT Procedure
Selected Model

Effects:	Intercept x1 x2 x5

Output 49.5.1 *continued*

Parameter Estimates

Parameter	DF	Estimate
Intercept	1	-0.243219
x1	1	3.246129
x2	1	1.310514
x5	1	2.132416

You see that the selected model contains only the relevant regressors x1, x2, and x5. You might want to investigate how frequently the adaptive LASSO method selects just the relevant regressors and how stable the corresponding parameter estimates are. In a simulation study, you can do this by drawing new samples and repeating this process many times. What can you do when you only have a single sample of the data available? One approach is to repeatedly draw subsamples from the data that you have, and to fit models for each of these samples. You can then form the average model and use this model for prediction. You can also examine how frequently models are selected, and you can use the frequency of effect selection as a measure of effect importance.

The following statements show how you can use the MODELAVERAGE statement to perform such an analysis:

```
ods graphics on;

proc glmselect data=simData seed=3 plots=(EffectSelectPct ParmDistribution);
   model y=x1-x10/selection=LASSO(adaptive stop=none choose=SBC);
   modelAverage tables=(EffectSelectPct(all) ParmEst(all));
run;
```

The ModelAverageInfo table in Output 49.5.2 shows that the default sampling method is the bootstrap approach of drawing 100 samples with replacement, where the sampling percentage of 100 means that each sample has the same sum of frequencies as the input data. You can use the SAMPLING= and NSAMPLES= options in the MODELAVERAGE statement to modify the sampling method and the number of samples used.

Output 49.5.2 Model Averaging Information

The GLMSELECT Procedure

Model Averaging Information	
Sampling Method	Unrestricted (with replacement)
Sample Percentage	100
Number of Samples	100

Output 49.5.3 shows the percentage of samples where each effect is in the selected model. The ALL option of the EFFECTSELECTPCT request in the TABLES= option specifies that effects that appear in any selected model are shown. (By default, the “Effect Selection Percentage” table displays only those effects that are selected at least 20 percent of the time.)

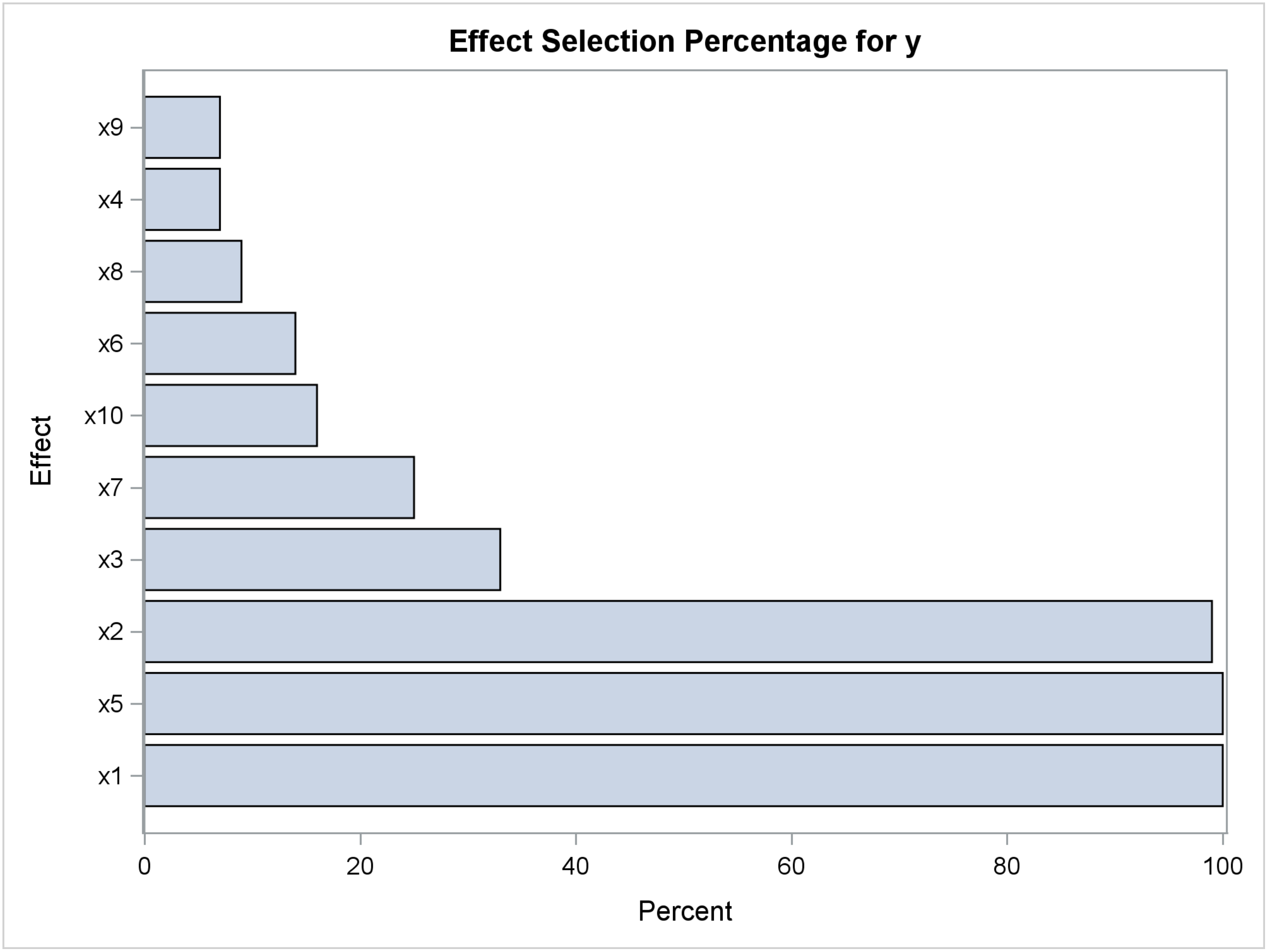
Output 49.5.3 Effect Selection Percentages

Effect Selection Percentage

Effect	**Selection Percentage**
x1	100.0
x2	99.00
x3	33.00
x4	7.00
x5	100.0
x6	14.00
x7	25.00
x8	9.00
x9	7.00
x10	16.00

The EFFECTSELECTPCT request in the PLOTS= option in the PROC GLMSELECT statement produces the bar chart shown in Output 49.5.4, which graphically displays the information in the EffectSelectPct table.

Output 49.5.4 Effect Selection Percentages



Output 49.5.5 shows the frequencies with which models get selected. By default, only the "best" 20 models are shown. See the section "Model Selection Frequencies and Frequency Scores" on page 3867 for details about how these models are ordered.

Output 49.5.5 Model Selection Frequency

Model Selection Frequency

Times Selected	Selection Percentage	Number of Effects	Frequency Score	Effects in Model
44	44.00	4	45.00	Intercept x1 x2 x5
9	9.00	5	9.86	Intercept x1 x2 x3 x5
8	8.00	6	8.76	Intercept x1 x2 x3 x5 x7
4	4.00	5	4.82	Intercept x1 x2 x5 x8
4	4.00	7	4.67	Intercept x1 x2 x3 x5 x6 x7
3	3.00	5	3.85	Intercept x1 x2 x5 x7
2	2.00	5	2.83	Intercept x1 x2 x5 x10
2	2.00	5	2.81	Intercept x1 x2 x4 x5
2	2.00	6	2.74	Intercept x1 x2 x3 x5 x6
2	2.00	7	2.66	Intercept x1 x2 x3 x5 x6 x10
1	1.00	5	1.83	Intercept x1 x2 x5 x6
1	1.00	4	1.81	Intercept x1 x5 x7
1	1.00	5	1.81	Intercept x1 x2 x5 x9
1	1.00	6	1.75	Intercept x1 x2 x3 x5 x10
1	1.00	6	1.74	Intercept x1 x2 x3 x5 x8
1	1.00	6	1.73	Intercept x1 x2 x5 x6 x7
1	1.00	6	1.72	Intercept x1 x2 x5 x7 x9
1	1.00	6	1.71	Intercept x1 x2 x5 x8 x10
1	1.00	6	1.70	Intercept x1 x2 x4 x5 x10
1	1.00	7	1.68	Intercept x1 x2 x3 x5 x7 x10

You can see that the most frequently selected model is the model that contains just the true underlying regressors. Although this model is selected in 44% of the samples, most of the selected models contain at least one irrelevant regressor. This is not surprising because even though the true model has just a few large effects in this example, the regressors have nontrivial pairwise correlations.

When your goal is simply to obtain a predictive model, then a good strategy is to average the predictions that you get from all the selected models. In the linear model context, this corresponds to using the model whose parameter estimates are the averages of the estimates that you get for each sample, where if a parameter is not in a selected model, the corresponding estimate is defined to be zero. This has the effect of shrinking the estimates of infrequently selected parameters towards zero.

Output 49.5.6 shows the average parameter estimates. The ALL option of the PARMEST request in the TABLES= option specifies that all parameters that are nonzero in any selected model are shown. (By default, the "Average Parameter Estimates" table displays only those parameters that are nonzero in at least 20 percent of the selected models.)

Output 49.5.6 Average Parameter Estimates

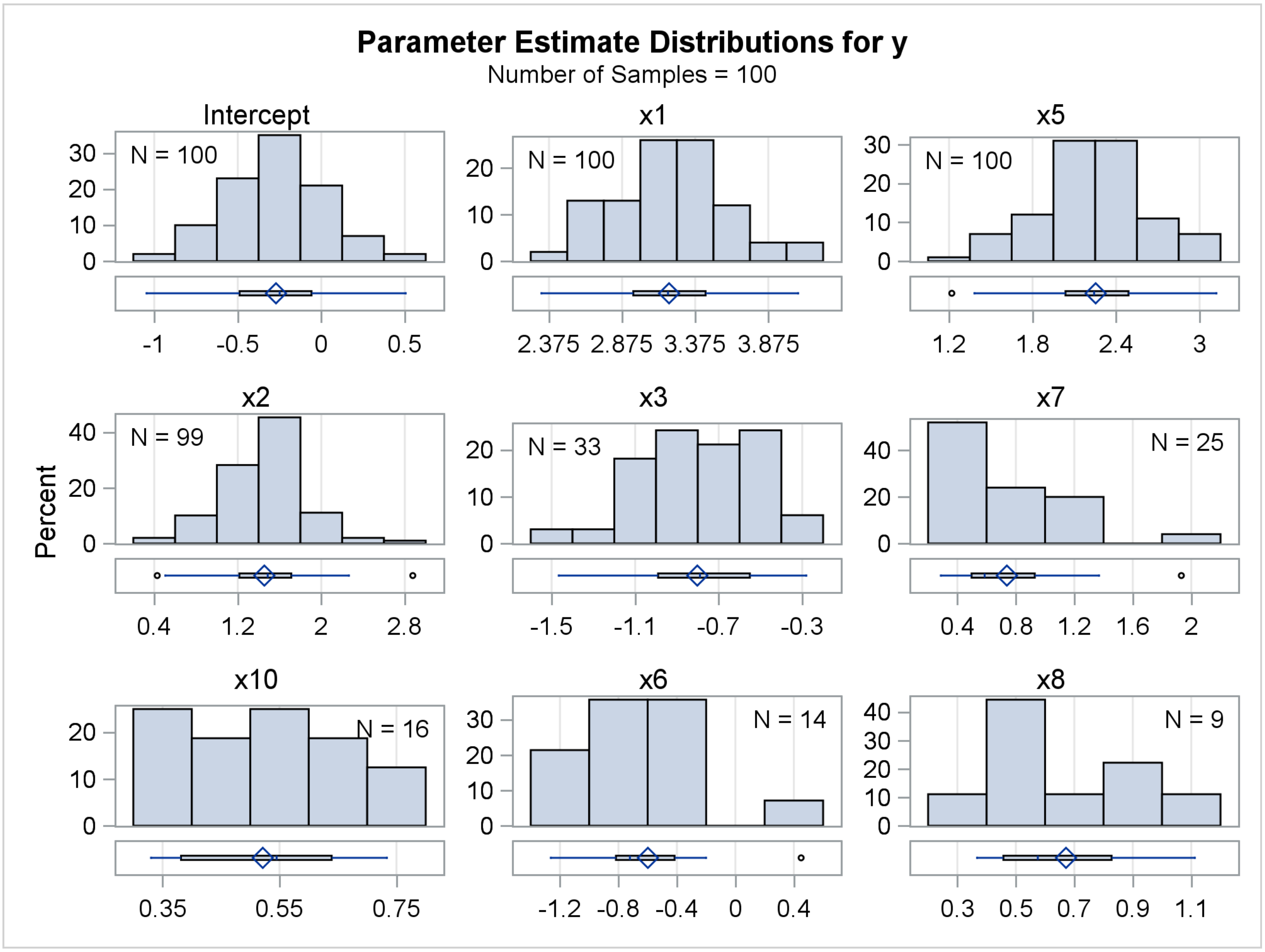
Average Parameter Estimates

					Estimate Quantiles		
Parameter	**Number Non-zero**	**Non-zero Percentage**	**Mean Estimate**	**Standard Deviation**	**25%**	**Median**	**75%**
Intercept	100	100.00	-0.271262	0.308146	-0.489061	-0.249163	-0.058233
x1	100	100.00	3.196392	0.377771	2.951551	3.189078	3.446055
x2	99	99.00	1.439966	0.416054	1.209781	1.484064	1.710275
x3	33	33.00	-0.264831	0.412148	-0.536449	0	0
x4	7	7.00	-0.037810	0.142932	0	0	0
x5	100	100.00	2.253196	0.397032	2.036261	2.242240	2.489068
x6	14	14.00	-0.083823	0.261641	0	0	0
x7	25	25.00	0.184656	0.372813	0	0	0.143317
x8	9	9.00	0.060438	0.206621	0	0	0
x9	7	7.00	-0.043307	0.239940	0	0	0
x10	16	16.00	0.083411	0.199573	0	0	0

The average estimate for a parameter is computed by dividing the sum of the estimate values for that parameter in each sample by the total number of samples. This corresponds to using zero as the estimate value for the parameter in those samples where the parameter does not appear in the selected model. Similarly, these zero estimates are included in the computation of the estimated standard deviation and quantiles that are displayed in the AvgParmEst table. If you want see the estimates that you get if you do not use zero for nonselected parameters, you can specify the NONZEROPARMS suboption of the PARMEST request in the TABLES=option.

The PARMDISTRIBUTION request in the PLOTS= option in the PROC GLMSELECT statement requests the panel in Output 49.5.7, which shows the distribution of the estimates for each parameter in the average model. For each parameter in the average model, a histogram and box plot of the nonzero values of the estimates are shown. You can use this plot to assess how the selected estimates vary across the samples.

Output 49.5.7 Effect Selection Percentages



You can obtain details about the model selection for each sample by specifying the DETAILS option in the MODELAVERAGE statement. You can use an OUTPUT statement to output the mean predicted value and standard deviation, quantiles of the predicted values, as well as the individual sample frequencies and predicted values for each sample. The following statements show how you do this:

```
proc glmselect data=simData seed=3;
   model y=x1-x10/selection=LASSO(adaptive stop=none choose=SBC);
   modelAverage details;
   output out=simOut sampleFreq=sf samplePred=sp
                     p=p stddev=stddev lower=q25 upper=q75 median;
run;
```

Output 49.5.8 shows the selection summary and parameter estimates for sample 1 of the model averaging. Note that you can obtain all the model selection output, including graphs, for each sample.

Output 49.5.8 Selection Details for Sample 1

The GLMSELECT Procedure
Sample 1

LASSO Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	SBC
0	Intercept		1	374.8287
1	x1		2	243.4087
2	x5		3	227.5991
3	x2		4	225.7356*
4	x7		5	229.9135
5	x3		6	233.3660
6	x6		7	237.7447
7	x10		8	235.2171
8	x4		9	238.8085
9	x9		10	239.8353
10	x8		11	244.4236

* Optimal Value of Criterion

Parameter Estimates

Parameter	DF	Estimate
Intercept	1	-0.092885
x1	1	4.079938
x2	1	0.505697
x5	1	1.473929

The following statements display the subset of the variables in the first five observations of the output data set, as shown in Output 49.5.9.

```
proc print data=simOut(obs=5);
   var p stddev q25 median q75 sf1-sf3 sp1-sp3;
run;
```

Output 49.5.9 Part of the Output Data Set

Obs	p	stddev	q25	median	q75	sf1	sf2	sf3	sp1	sp2	sp3
1	10.3569	0.82219	9.95992	10.3878	10.9194	1	0	1	10.1378	11.2104	11.0124
2	-5.5453	0.64544	-6.05563	-5.6455	-5.0829	1	1	1	-4.7517	-6.7191	-6.4413
3	6.5066	0.75289	6.05984	6.5077	6.9099	3	2	0	6.0838	7.4880	6.3466
4	-1.7527	0.85168	-2.26638	-1.8123	-1.3312	1	1	2	-2.1891	-1.4887	-1.7083
5	-7.5840	1.20687	-8.44679	-7.5716	-6.7386	3	1	1	-6.7051	-9.0558	-6.7949

By default, the LOWER and UPPER options in the OUTPUT statement produce the lower and upper quartiles of the sample predicted values. You can change the quantiles produced by using the ALPHA= option in the MODELAVERAGE statement. The variables sf1–sf100 contain the sample frequencies used for each sample, and the variables sp1–sp100 hold the corresponding predicted values. Even if you do not specify the DETAILS option in the MODELAVERAGE statement, you can use the sample frequencies in the output data set to reproduce the selection results for any particular sample. For example, the following statements recover the selection for sample 1:

```
proc glmselect data=simOut;
   freq sf1;
   model y=x1-x10/selection=LASSO(adaptive stop=none choose=SBC);
run;
```

The average model is not parsimonious—it includes shrunken estimates of infrequently selected parameters which often correspond to irrelevant regressors. It is tempting to ignore the estimates of infrequently selected parameters by setting their estimate values to zero in the average model. However, this can lead to a poorly performing model. Even though a parameter might occur in only one selected model, it might be a very important term in that model. Ignoring its estimate but including some of the estimates of the other parameters in this model leads to biased predictions. One scenario where this might occur is when the data contains two highly correlated regressors which are both strongly correlated with the response.

You can obtain a parsimonious model by using the frequency of effect selection as a measure of effect importance and refitting a model that contains just the effects that you deem important. In this example, Output 49.5.3 shows that the effects x1, x2, and x5 all get selected at least 99 percent of the time, whereas all other effects get selected less than 34 percent of the time. This large gap suggests that using 35% as the selection cutoff for this data will produce a parsimonious model that retains good predictive performance. You can use the REFIT option to implement this strategy. The REFIT option requests a second round of model averaging, where you use the MINPCT= suboption to specify the minimum percentage of times an effect must be selected in the initial set of samples to be included in the second round of model averaging. The average model is obtained by averaging the ordinary least squares estimates obtained for each sample. The following statements show how you do this:

```
proc glmselect data=simData seed=3 plots=(ParmDistribution);
   model y=x1-x10/selection=LASSO(adaptive stop=none choose=SBC);
   modelAverage refit(minpct=35 nsamples=1000) alpha=0.1;
run;

ods graphics off;
```

The NSAMPLES=1000 suboption of the REFIT option specifies that 1,000 samples be used in the refit, and the MINPCT=35 suboption specifies the cutoff for inclusion in the refit. The ALPHA=0.1 option specifies that the 5th and 95th percentiles of the estimates be displayed. Output 49.5.10 shows the effects that are used in performing the refit and the resulting average parameter estimates.

Output 49.5.10 Refit Average Parameter Estimates

The GLMSELECT Procedure
Refit Model Averaging Results

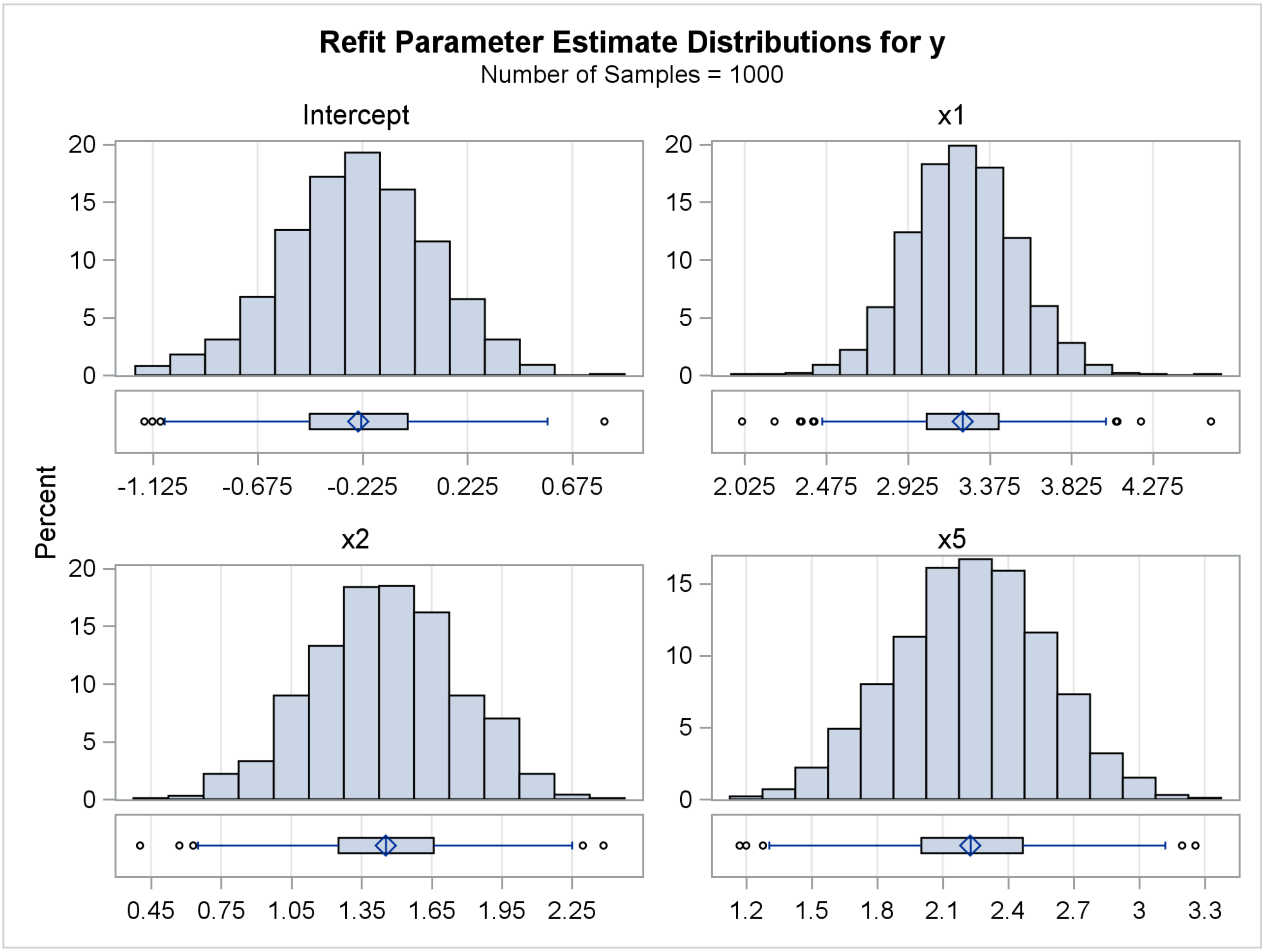
Effects: Intercept x1 x2 x5

Average Parameter Estimates

			Estimate Quantiles		
Parameter	**Mean Estimate**	**Standard Deviation**	**5%**	**Median**	**95%**
Intercept	-0.243514	0.315207	-0.762462	-0.230630	0.271510
x1	3.226252	0.299443	2.737843	3.226758	3.708131
x2	1.453584	0.308062	0.947059	1.454635	1.968231
x5	2.226044	0.345185	1.627491	2.228189	2.780034

Output 49.5.11 displays the distributions of the estimates that are obtained for each parameter in the refit model. Because the distributions are approximately normal and a large number of samples are used, it is reasonable to interpret the range between the 5th and 95th percentiles of each estimate as an approximate 90% confidence interval for that estimate.

Output 49.5.11 Effect Selection Percentages



Example 49.6: Elastic Net and External Cross Validation

This example shows how to use the elastic net method for model selection and compares it with the LASSO method. The example also uses *k*-fold external cross validation as a criterion in the CHOOSE= option to choose the best model based on the penalized regression fit.

This example uses a microarray data set called the leukemia (LEU) data set (Golub et al. 1999), which is used in the paper by Zou and Hastie (2005) to demonstrate the performance of the elastic net method in comparison with that of LASSO. The LEU data set consists of 7,129 genes and 72 samples, and 38 samples are used as training samples. Among the 38 training samples, 27 are type 1 leukemia (acute lymphoblastic leukemia) and 11 are type 2 leukemia (acute myeloid leukemia). The goal is to construct a diagnostic rule based on the expression level of those 7,219 genes to predict the type of leukemia. The remaining 34 samples are used as the validation data for picking the appropriate model or as the test data for testing the performance of the selected model. The training and validation data sets are available from Sashelp.Leutrain and Sashelp.Leutest.

To have a basis for comparison, use the following statements to apply the LASSO method for model selection:

```
ods graphics on;
proc glmselect data=sashelp.Leutrain valdata=sashelp.Leutest
               plots=coefficients;
   model y = x1-x7129/
         selection=LASSO(steps=120  choose=validate);
run;
```

The details about the number of observations for training and validation data are given in Output 49.6.1 and Output 49.6.2, respectively.

Output 49.6.1 Number of Training Observations Table

The GLMSELECT Procedure

Observation Profile for Analysis Data	
Number of Observations Read	38
Number of Observations Used	38
Number of Observations Used for Training	38

Output 49.6.2 Number of Validation Observations Table

Observation Profile for Validation Data	
Number of Observations Read	34
Number of Observations Used	34

Output 49.6.3 shows the “LASSO Selection Summary” table. Although the STEPS= suboption of the SELECTION= option specifies that 120 steps of LASSO selection be performed, the LASSO method terminates at step 81 because the selected model is a perfect fit and the number of effects that can be selected by LASSO is bounded by the number of training samples. Note that a VALDATA= data set is named in the PROC GLMSELECT statement and that the minimum of the validation ASE occurs at step 68. Hence the model at this step is selected, resulting in 35 selected effects. Output 49.6.4 shows the standardized coefficients of all the effects selected at some step of the LASSO method, plotted as a function of the step number.

Output 49.6.3 LASSO Selection Summary Table

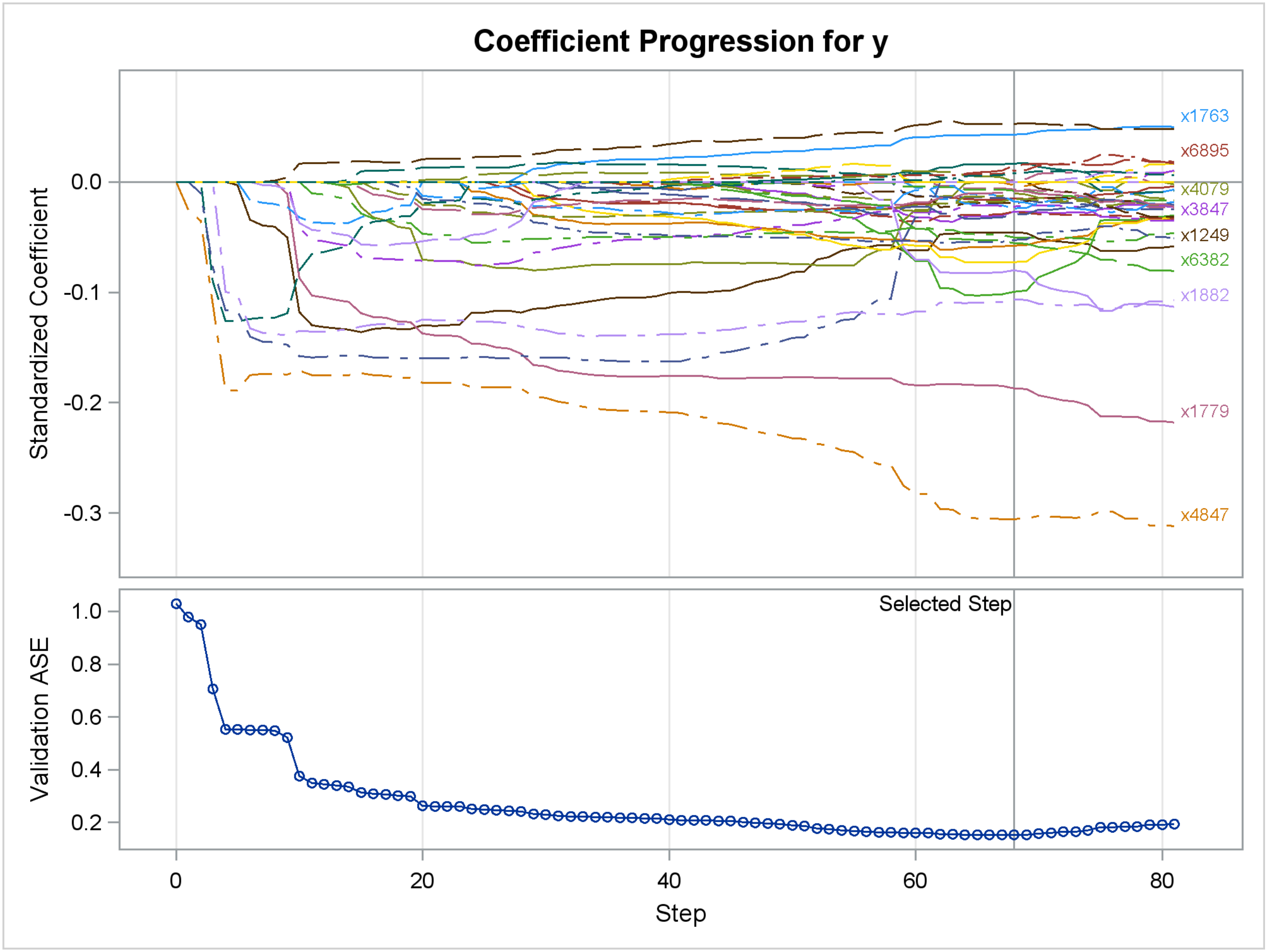
The GLMSELECT Procedure

LASSO Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	ASE	Validation ASE
0	Intercept		1	0.8227	1.0287
1	x4847		2	0.7884	0.9787
2	x3320		3	0.7610	0.9507
3	x2020		4	0.4935	0.7061
4	x5039		5	0.2838	0.5527
5	x1249		6	0.2790	0.5518
6	x2242		7	0.2255	0.5513
			.	.	.
			.	.	.
			.	.	.
62	x2534		35	0.0016	0.1554
63		x3320	34	0.0016	0.1557
64	x5631		35	0.0013	0.1532
65		x6021	34	0.0012	0.1534
66		x1745	33	0.0012	0.1535
67	x6376		34	0.0012	0.1535
68	x4831		35	0.0010	0.1531*
69		x6184	34	0.0009	0.1539
70	x3820		35	0.0006	0.1588
71	x3171		36	0.0005	0.1615
72		x6376	35	0.0004	0.1640
73	x5142		36	0.0004	0.1659
74	x1745		37	0.0003	0.1688
75	x3605		38	0.0001	0.1821
76		x573	37	0.0001	0.1825
77	x3221		38	0.0001	0.1842
78		x1745	37	0.0001	0.1842
79	x6163		38	0.0000	0.1918
80		x4831	37	0.0000	0.1920
81	x1745		38	0.0000	0.1940

*** Optimal Value of Criterion**

Output 49.6.4 LASSO Coefficient Progression Plot



As discussed earlier, the number of effects that can be selected by the LASSO method is upper-bounded by the number of training samples. However, this is not a restriction for the elastic net method, which incorporates an additional ridge regression penalty. You can use the following statements to apply the elastic net method for model selection:

```
proc glmselect data=sashelp.Leutrain valdata=sashelp.Leutest
               plots=coefficients;
   model y = x1-x7129/
         selection=elasticnet(steps=120 L2=0.001 choose=validate);
run;
```

The L2= suboption of the SELECTION= option in the MODEL statement specifies the value of the ridge regression parameter. Output 49.6.5 shows the “Elastic Net Selection Summary” table. As in the example for LASSO, the STEPS= suboption of the SELECTION= option specifies that 120 steps of elastic net selection be performed. However, in contrast to the early termination in the LASSO method at step 81, the elastic net method runs until the specified 120 steps are performed. With the same VALDATA= data set named in the PROC GLMSELECT statement as in the LASSO example, the minimum of the validation ASE occurs at step 105, and hence the model at this step is selected, resulting in 54 selected effects. Compared with the LASSO method, the elastic net method can select more variables, and the number of selected variables is not restricted by the number of samples (38 for this example). Output 49.6.6 shows the standardized coefficients of all the effects that are selected at some step of the elastic net method, plotted as a function of the step number.

Output 49.6.5 Elastic Net Selection Summary Table

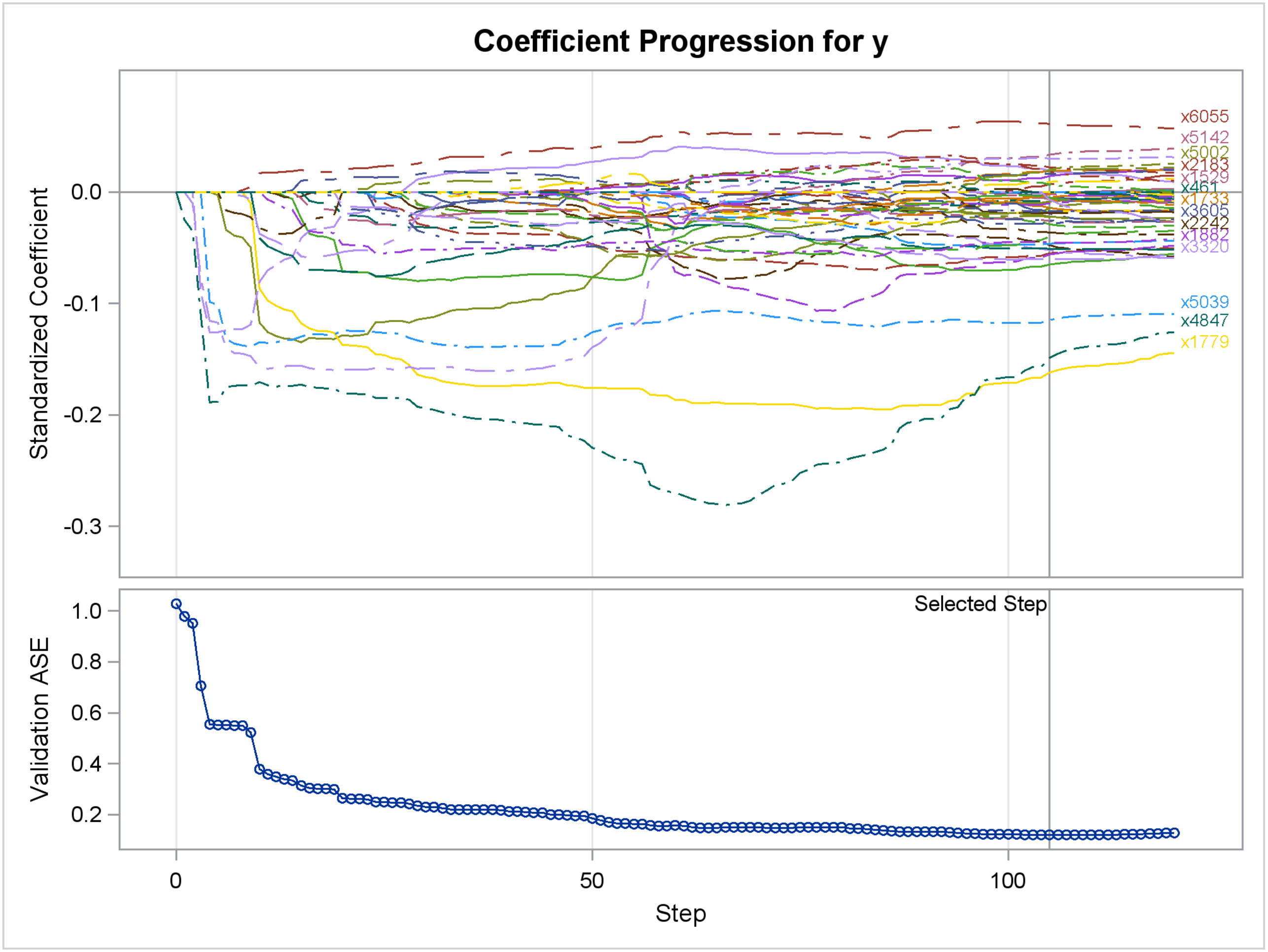
The GLMSELECT Procedure

Elastic Net Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	ASE	Validation ASE
0	Intercept		1	0.8227	1.0287
1	x4847		2	0.7885	0.9790
2	x3320		3	0.7611	0.9510
3	x2020		4	0.4939	0.7065
4	x5039		5	0.2845	0.5533
5	x1249		6	0.2799	0.5525
6	x2242		7	0.2259	0.5517
			.	.	.
			.	.	.
			.	.	.
100	x259		53	0.0000	0.1207
101	x1781		54	0.0000	0.1207
102		x7065	53	0.0000	0.1198
103	x4079		54	0.0000	0.1193
104		x6163	53	0.0000	0.1193
105	x6539		54	0.0000	0.1192*
106	x6071		55	0.0000	0.1192
107	x1829		56	0.0000	0.1192
108		x5598	55	0.0000	0.1192
109	x6169		56	0.0000	0.1192
110	x1529		57	0.0000	0.1194
111	x1306		58	0.0000	0.1194
112	x5954		59	0.0000	0.1202
113	x4271		60	0.0000	0.1203
114	x3312		61	0.0000	0.1207
115	x461		62	0.0000	0.1213
116	x4697		63	0.0000	0.1228
117		x259	62	0.0000	0.1231
118	x4186		63	0.0000	0.1252
119		x4271	62	0.0000	0.1258
120	x4939		63	0.0000	0.1261

* Optimal Value of Criterion

Output 49.6.6 Elastic Net Coefficient Progression Plot



When you have a good estimate of the ridge regression parameter, you can specify the L2= suboption of the SELECTION= option in the MODEL statement. However, when you do not have a good estimate of the ridge regression parameter, you can leave the L2= option unspecified. In this case, PROC GLMSELECT estimates this regression parameter based on the criterion specified in the CHOOSE= option.

If you have a validation data set, you can use the following statements to apply the elastic net method of model selection:

```
proc glmselect data=sashelp.Leutrain valdata=sashelp.Leutest
               plots=coefficients;
   model y = x1-x7129/
         selection=elasticnet(steps=120 choose=validate);
run;
```

PROC GLMSELECT tries a series of candidate values for the ridge regression parameter, which you can control by using the L2HIGH=, L2LOW=, and L2SEARCH= options. The ridge regression parameter is set to the value that achieves the minimum validation ASE (see Figure 49.12 for an illustration). Output 49.6.7 shows the "Elastic Net Selection Summary" table. Compared to Output 49.6.5, where a ridge regression parameter is specified by the L2= option, this table shows that a slightly lower validation ASE is achieved as the value of the ridge regression parameter is optimized on the validation data set.

Output 49.6.7 Elastic Net Selection Summary Table

The GLMSELECT Procedure

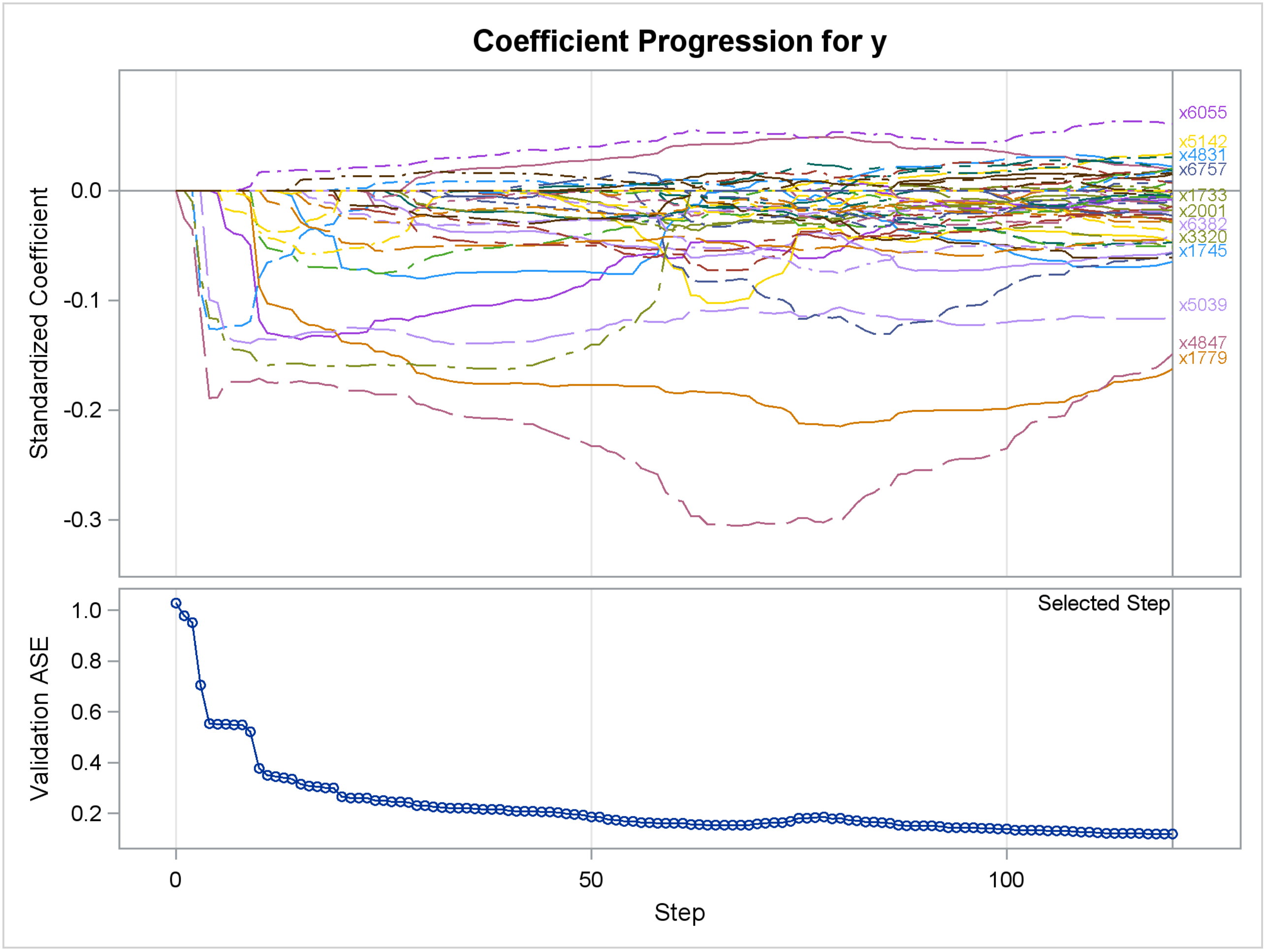
Elastic Net Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	ASE	Validation ASE
0	**Intercept**		1	0.8227	1.0287
1	**x4847**		2	0.7884	0.9787
2	**x3320**		3	0.7610	0.9508
3	**x2020**		4	0.4936	0.7061
4	**x5039**		5	0.2839	0.5527
5	**x1249**		6	0.2790	0.5519
6	**x2242**		7	0.2256	0.5513
			.	.	.
			.	.	.
			.	.	.
110	**x3605**		49	0.0000	0.1252
111	**x1975**		50	0.0000	0.1241
112	**x6757**		51	0.0000	0.1224
113		**x3221**	50	0.0000	0.1206
114	**x2183**		51	0.0000	0.1205
115	**x259**		52	0.0000	0.1202
116	**x157**		53	0.0000	0.1201
117	**x1781**		54	0.0000	0.1197
118		**x7065**	53	0.0000	0.1194
119	**x4079**		54	0.0000	0.1187
120	**x6539**		55	0.0000	0.1186*

*** Optimal Value of Criterion**

Output 49.6.8 shows the standardized coefficients of all the effects that are selected at some step of the elastic net method, plotted as a function of the step number.

Output 49.6.8 Elastic Net Coefficient Progression Plot



When validation data are not available, you can use other criteria supported by the CHOOSE= option for estimating the value of the ridge regression parameter. Criteria such as AIC, AICC, and BIC are defined for the training samples only, and they tend to set the value of the ridge regression parameter to 0 when the number of training samples is less than the number of variables. In this case, the elastic net method reduces to the LASSO method. As an alternative to AIC, AICC, and BIC, you can use either *k*-fold cross validation or *k*-fold external cross validation in the CHOOSE= option, where the criterion is computed for held-out data.

If you want to use *k*-fold cross validation for selecting the model and selecting the appropriate value for the ridge regression parameter, you can use the following statements to apply the elastic net method of model selection:

```
proc glmselect data=sashelp.Leutrain testdata=sashelp.Leutest
               plots=coefficients;
   model y = x1-x7129/
         selection=elasticnet(steps=120 choose=cv)cvmethod=split(4);
run;
```

Output 49.6.9 Elastic Net Selection Summary Table

The GLMSELECT Procedure

Elastic Net Selection Summary

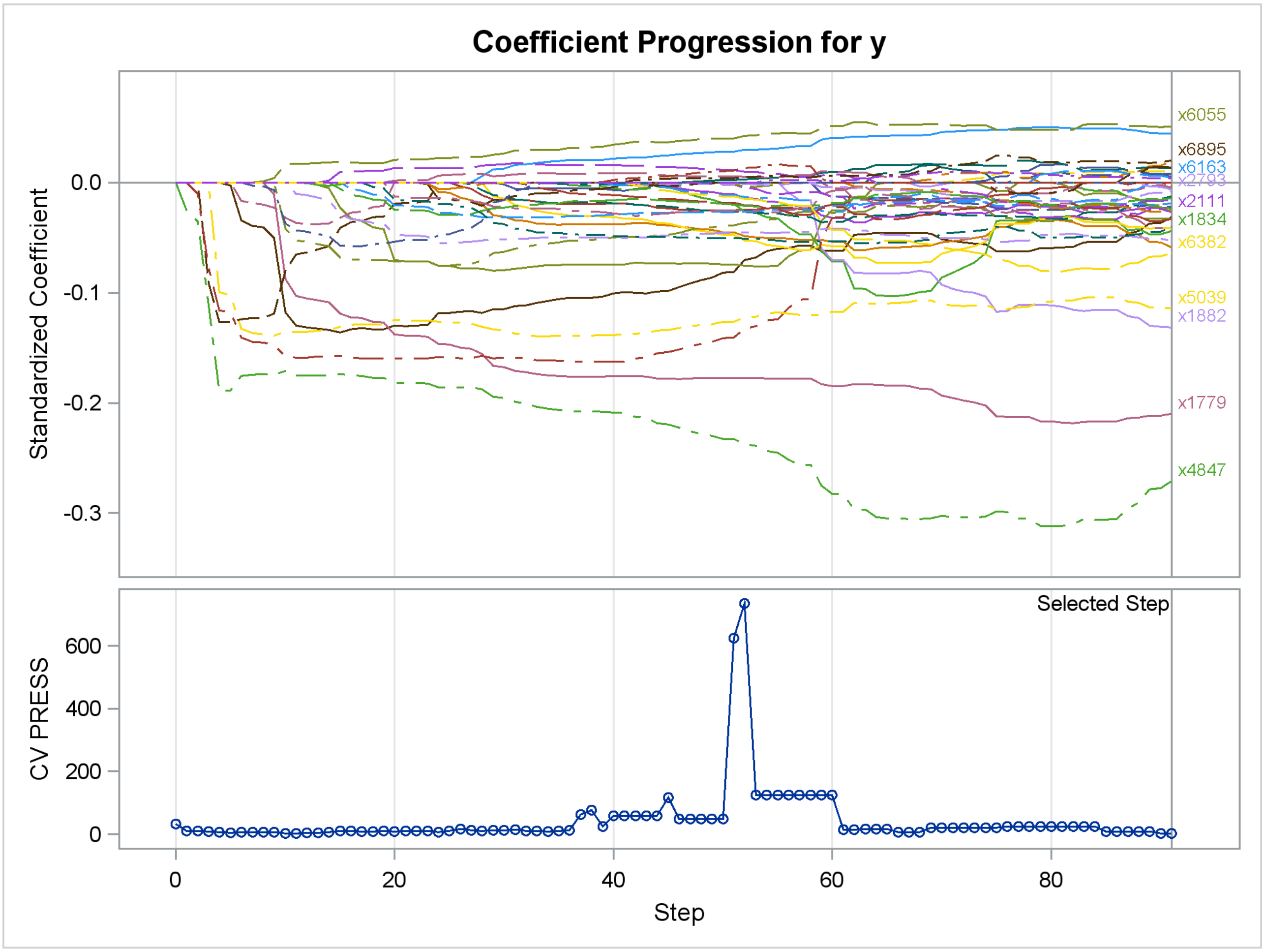
Step	Effect Entered	Effect Removed	Number Effects In	ASE	Test ASE	CV PRESS
0	**Intercept**		1	0.8227	1.0287	31.4370
1	**x4847**		2	0.7884	0.9787	10.6957
2	**x3320**		3	0.7610	0.9507	10.2065
3	**x2020**		4	0.4935	0.7061	7.5478
4	**x5039**		5	0.2838	0.5527	5.7695
5	**x1249**		6	0.2790	0.5518	5.0187
6	**x2242**		7	0.2255	0.5513	5.4050
			.	.	.	.
			.	.	.	.
			.	.	.	.
80		**x4831**	37	0.0000	0.1919	24.0303
81	**x1745**		38	0.0000	0.1933	24.0303
82	**x4831**		39	0.0000	0.1937	24.0303
83	**x2661**		40	0.0000	0.1864	24.0303
84		**x3605**	39	0.0000	0.1857	24.0303
85		**x464**	38	0.0000	0.1857	7.2724
86	**x5766**		39	0.0000	0.1854	7.2724
87	**x464**		40	0.0000	0.1763	7.2724
88	**x6376**		41	0.0000	0.1735	7.2724
89		**x464**	40	0.0000	0.1675	7.2724
90		**x4079**	39	0.0000	0.1674	1.2851
91	**x1987**		40	0.0000	0.1625	1.2846*

*** Optimal Value of Criterion**

A TESTDATA= data set is named in the PROC GLMSELECT statement; this is the same as the validation data set used earlier. Because the L2= option is not specified, PROC GLMSELECT tries a series of candidate values for the ridge regression parameter according to the CHOOSE=CV option, and the ridge regression parameter is set to the value that achieves the minimal CVPRESS score. Output 49.6.9 shows the "Elastic Net Selection Summary" table, which corresponds to the ridge regression parameter selected by *k*-fold cross validation. The elastic net method achieves the smallest CVPRESS score at step 91. Hence the model at this step is selected, resulting in 40 selected effects.

Output 49.6.10 shows the standardized coefficients of all the effects selected at some step of the elastic net method, plotted as a function of the step number. Note that *k*-fold cross validation uses a least squares fit to compute the CVPRESS score. Thus the criterion does not directly depend on the penalized regression used in the elastic net method, and the CVPRESS curve in Output 49.6.10 has abrupt jumps.

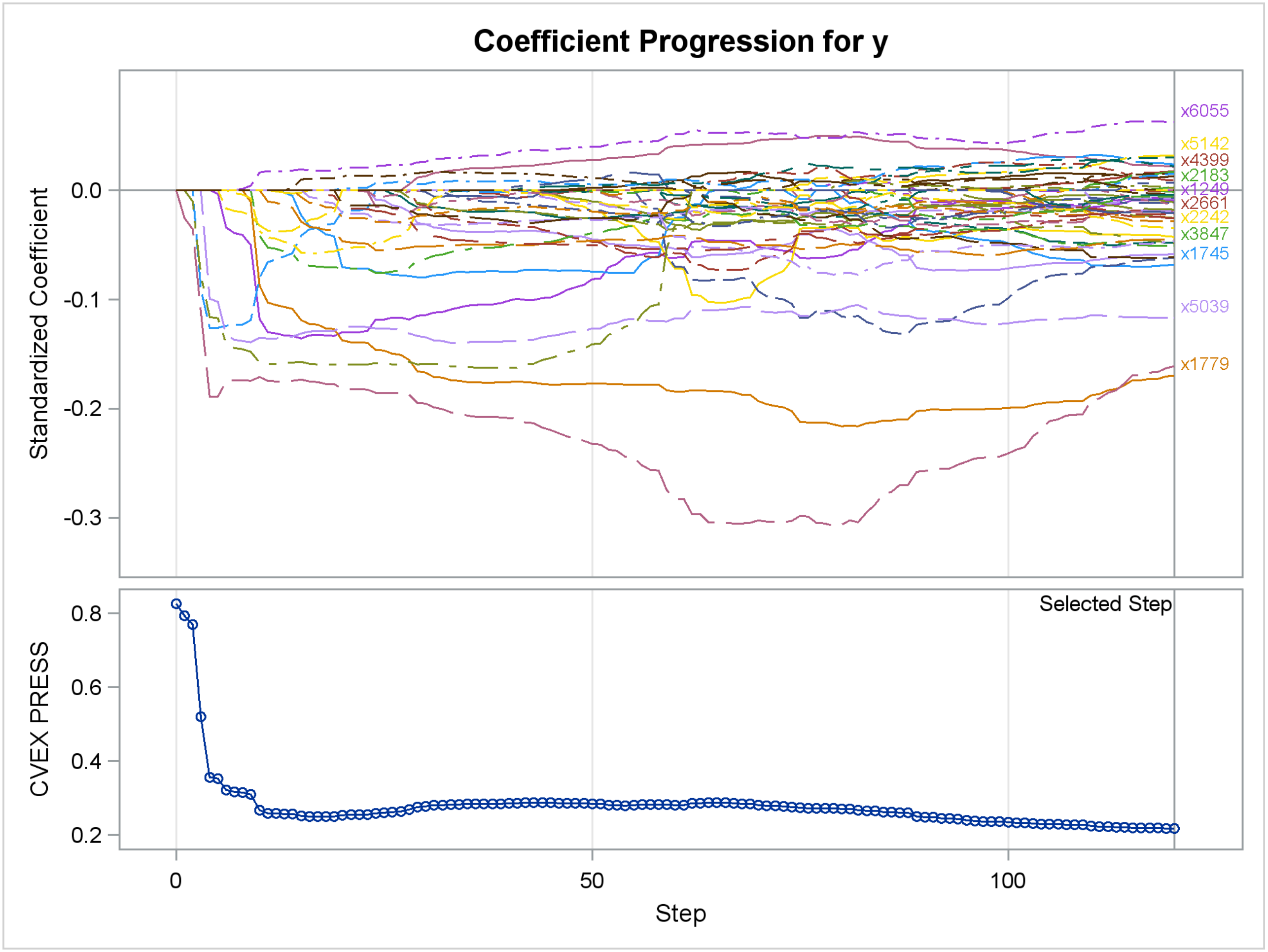
Output 49.6.10 Elastic Net Coefficient Progression Plot



If you want to perform the selection of both the ridge regression parameter and the selected variables in the model based on a criterion that directly depends on the regression used in the elastic net method, you can use *k*-fold external cross validation by specifying the following statements:

```
proc glmselect data=sashelp.Leutrain testdata=sashelp.Leutest
               plots=coefficients;
   model y = x1-x7129/
         selection=elasticnet(steps=120 choose=cvex)cvmethod=split(4);
run;
```

Output 49.6.11 Elastic Net Coefficient Progression Plot



As earlier, a TESTDATA= data set is named in the PROC GLMSELECT statement to test the prediction accuracy of the model on the training samples. Because the L2= option is not specified, PROC GLMSELECT tries a series of candidate values for the ridge regression parameter. Output 49.6.11 shows the standardized coefficients of all the effects selected at some step of the elastic net method, plotted as a function of the step number, and also the curve of the CVEXPRESS statistic as a function of the step number. A comparison between the criterion curves in Output 49.6.10 and Output 49.6.11 shows that the CVEXPRESS statistic is smoother than the CVPRESS statistic. Note that the CVEXPRESS statistic is based on a penalized model, whereas the CVPRESS statistic is based on an ordinary least squares model. Output 49.6.12 shows the "Elastic Net Selection Summary" table, which corresponds to the ridge regression parameter selected by *k*-fold external cross validation. The elastic net method achieves the smallest CVEXPRESS score at step 120, and hence the model at this step is selected, resulting in 53 selected effects.

Output 49.6.12 Elastic Net Selection Summary Table

The GLMSELECT Procedure

Elastic Net Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	ASE	Test ASE	CVEX PRESS
0	**Intercept**		1	0.8227	1.0287	0.8265
1	**x4847**		2	0.7884	0.9787	0.7940
2	**x3320**		3	0.7610	0.9507	0.7698
3	**x2020**		4	0.4936	0.7061	0.5195
4	**x5039**		5	0.2839	0.5527	0.3554
5	**x1249**		6	0.2790	0.5519	0.3526
6	**x2242**		7	0.2255	0.5513	0.3218
7	**x6539**		8	0.2161	0.5498	0.3166
8	**x6055**		9	0.2128	0.5488	0.3148
9	**x1779**		10	0.1968	0.5205	0.3096
10	**x3847**		11	0.1065	0.3765	0.2671
11	**x4951**		12	0.0891	0.3498	0.2586
12	**x1834**		13	0.0868	0.3458	0.2579
13	**x1745**		14	0.0837	0.3397	0.2570
14	**x6021**		15	0.0809	0.3355	0.2561
15	**x6362**		16	0.0667	0.3144	0.2510
16	**x2238**		17	0.0621	0.3075	0.2498
			.	.	.	.
			.	.	.	.
			.	.	.	.
100	**x2121**		45	0.0000	0.1402	0.2350
101	**x6021**		46	0.0000	0.1386	0.2334
102		**x3171**	45	0.0000	0.1377	0.2324
103	**x4461**		46	0.0000	0.1350	0.2302
104	**x4697**		47	0.0000	0.1338	0.2299
105		**x3605**	46	0.0000	0.1331	0.2293
106	**x3221**		47	0.0000	0.1330	0.2289
107	**x5002**		48	0.0000	0.1320	0.2282
108		**x3820**	47	0.0000	0.1319	0.2281
109		**x4697**	46	0.0000	0.1315	0.2278
110	**x4052**		47	0.0000	0.1276	0.2235
111	**x6184**		48	0.0000	0.1266	0.2229
112	**x3605**		49	0.0000	0.1252	0.2215
113	**x1975**		50	0.0000	0.1240	0.2212
114	**x6757**		51	0.0000	0.1222	0.2200
115		**x3221**	50	0.0000	0.1206	0.2185
116	**x2183**		51	0.0000	0.1205	0.2185
117	**x259**		52	0.0000	0.1202	0.2182
118	**x157**		53	0.0000	0.1201	0.2181
119	**x1781**		54	0.0000	0.1197	0.2177
120		**x7065**	53	0.0000	0.1194	0.2177*

*** Optimal Value of Criterion**

Example 49.7: LASSO with Screening

This example shows how you can use the SCREEN= option to speed up model selection when you have a large number of regressors. In order to demonstrate the efficiency in screening model selection, this example uses simulated data in which the response y depends systematically on a relatively small subset of a much larger set of regressors, which is described in Table 49.13.

Table 49.13 Complete Set of Regressors

Regressor Name	Type	Number of Levels	In True Model
xIn1–xIn5	Continuous		Yes
xOut1–xOut1000	Continuous		No
cIn1–cIn2	Classification	2–5	Yes
cOut1–cOut1000	Classification	2–5	No

The labels In and Out, which are part of the regressor names, make it easy to identify whether the selected model succeeds or fails in capturing the true underlying model.

The following DATA step generates the data:

```
%let nObs      = 5000;
%let nContIn   = 5;
%let nContOut  = 1000;
%let nClassIn  = 2;
%let nClassOut = 1000;
%let maxLevs   = 5;
%let noiseScale= 1;

 data ex7Data;
   array xIn{&nContIn};
   array xOut{&nContOut};
   array cIn{&nClassIn};
   array cOut{&nClassOut};

   drop i j sign nLevs xBeta;

   do i=1 to &nObs;
      sign  = -1;
      xBeta = 0;
      do j=1 to dim(xIn);
         xIn{j} = ranuni(1);
         xBeta  = xBeta + sqrt(j)*sign*xIn{j};
         sign   = -sign;
      end;
      do j=1 to dim(xOut);
         xOut{j} = ranuni(1);
      end;

      do j=1 to dim(cIn);
         nLevs  = 2 + mod(j,&maxlevs-1);
         cIn{j} = 1+int(ranuni(1)*nLevs);
```

```
            xBeta   = xBeta + j*sign*(cIn{j}-nLevs/2);
            sign    = -sign;
         end;

         do j=1 to dim(cOut);
            nLevs   = 2 + mod(j,&maxlevs-1);
            cOut{j} = 1+int(ranuni(1)*nLevs);
         end;

         y = xBeta + &noiseScale*rannor(1);

         output;
     end;
  run;
```

The following statements apply the LASSO method for model selection but do not specify any screening technique:

```
  proc glmselect data=ex7Data;
     class c:;
     model y = x: c:/
           selection=lasso;
  run;
```

Output 49.7.1 and Output 49.7.2 show the number of observations and the number of effects, respectively. You can see that 5,000 observations are used for training and the number of effects after splits is 4,513.

Output 49.7.1 Number of Training Observations Table

The GLMSELECT Procedure

Number of Observations Read	5000
Number of Observations Used	5000

Output 49.7.2 Dimensions

Dimensions	
Number of Effects	2008
Number of Effects after Splits	4513
Number of Parameters	4513

Output 49.7.3 shows the “LASSO Selection Summary” table. You can see that the selected model contains all the predictive effects and successfully excludes all the noise effects.

Output 49.7.3 LASSO Selection Summary Table

The GLMSELECT Procedure

Without the SCREEN= option

Step	Effect Entered	Effect Removed	Number Effects In	SBC
0	Intercept		1	10413.6418
1	cln2_1		2	10335.5423
2	cln2_4		3	7237.0701
3	cln1_1		4	7186.9642
4	cln1_3		5	6753.6076
5	xln5		6	6609.0322
6	cln2_3		7	6510.2639
7	xln4		8	5797.1607
8	xln3		9	5487.4432
9	xln2		10	2894.0166
10	xln1		11	310.9340*

* Optimal Value of Criterion

You can improve the computational performance of the LASSO method by using the SCREEN=SASVI option as follows:

```
proc glmselect data=ex7Data;
   class c:;
   model y = x: c:/
         selection=lasso(screen=sasvi);
run;
```

Output 49.7.4 shows that the safe screening approach SASVI is used.

Output 49.7.4 Screening Information

The GLMSELECT Procedure

Screening Information	
Screening Method	SASVI (safe screening)

Output 49.7.5 shows the "LASSO Selection Summary" table, which is exactly the same as Output 49.7.3. If you compare the "Parameter Estimates" tables that correspond to Output 49.7.3 and Output 49.7.5, you can see that the parameter estimates are exactly the same, because SASVI is a safe screening approach that guarantees the same solution, usually in less time. PROC GLMSELECT runs about twice as fast when SCREEN=SASVI as it runs when no screening is specified.

Output 49.7.5 LASSO Selection Summary Table

The GLMSELECT Procedure

SCREEN=SASVI

Step	Effect Entered	Effect Removed	Number Effects In	SBC
0	Intercept		1	10413.6418
1	cln2_1		2	10335.5423
2	cln2_4		3	7237.0701
3	cln1_1		4	7186.9642
4	cln1_3		5	6753.6076
5	xln5		6	6609.0322
6	cln2_3		7	6510.2639
7	xln4		8	5797.1607
8	xln3		9	5487.4432
9	xln2		10	2894.0166
10	xln1		11	310.9340*

* Optimal Value of Criterion

As an alternative to screening by SASVI, you can specify SCREEN=SIS for sure independence screening as follows:

```
proc glmselect data=ex7Data;
   class c:;
   model y = x: c:/
         selection=lasso(screen=sis(keepnum=15));
run;
```

Output 49.7.6 shows that sure independence screening is used and that the number of effects to be kept before applying LASSO for model selection is 15. The last two rows of Output 49.7.6 show the actual number of kept effects and the corresponding ratio with the total number of effects.

Output 49.7.6 Screening Information

The GLMSELECT Procedure

Screening Information	
Screening Method	SIS (sure independence screening)
Screening Keep Number	15
Actual Kept Number	15
Actual Kept Ratio	0.003

Output 49.7.7 shows the absolute correlation between the effects and the response that corresponds to the leading 20 effects. Because you specified KEEPNUM=15, only the leading 15 effects are kept for model selection by LASSO. In this example, you can see that all the predictive effects are among the top 15. Note that if the screening stage does not keep an effect, that effect has no chance of being selected in the final model.

Output 49.7.7 LASSO Selection Summary Table

The GLMSELECT Procedure

SCREEN=SIS

Effect Screening by Correlation

Rank	Effect	Maximum Absolute Correlation
1	cIn2_1	0.621286
2	cIn2_4	0.611998
3	cIn1_1	0.261449
4	cIn1_3	0.259374
5	cIn2_3	0.215103
6	xIn5	0.212433
7	xIn4	0.201704
8	cIn2_2	0.195087
9	xIn3	0.174133
10	xIn2	0.170705
11	xIn1	0.083742
12	cOut447_4	0.061958
13	cOut238_1	0.052981
14	cOut538_4	0.052165
15	cOut747_5	0.051390
16	xOut345	0.050114*
17	cOut511_5	0.047524*
18	cOut630_2	0.047440*
19	cOut672_2	0.045937*
20	cOut672_1	0.045937*

*** Screened Out Effect**

Output 49.7.8 shows the "LASSO Selection Summary" table. As in Output 49.7.3, all the predictive effects enter into the model. However, one noise effect also enters into the model, because SCREEN=SIS does not necessarily guarantee the same solution as the solution that results when SCREEN=NONE. For this example, PROC GLMSELECT runs only slightly faster when SCREEN=SIS than it does when SCREEN=SASVI, although it runs about twice as fast as it does when SCREEN=NONE. However, for problems that have more predictors or that use much more computationally intense CHOOSE= criterion, sure independence screening (SIS) can run faster by orders of magnitude.

Output 49.7.8 LASSO Selection Summary Table

The GLMSELECT Procedure

SCREEN=SIS

Step	Effect Entered	Effect Removed	Number Effects In	SBC
0	Intercept		1	10413.6418
1	cIn2_1		2	10335.5423
2	cIn2_4		3	7237.0701
3	cIn1_1		4	7186.9642
4	cIn1_3		5	6753.6076
5	xIn5		6	6609.0322
6	cIn2_3		7	6510.2639
7	xIn4		8	5797.1607
8	xIn3		9	5487.4432
9	xIn2		10	2894.0166
10	xIn1		11	245.9693
11	cOut747_5		12	238.4427*

*** Optimal Value of Criterion**

Example 49.8: Group LASSO Selection

This example shows how you can use the group LASSO method for model selection. This example treats the parameters that correspond to the same spline and CLASS variable as a group and also uses a collection effect to group otherwise unrelated parameters. The following DATA step generates the data:

```
%macro makeRegressorData(data=,nObs=500,nCont=5,nClass=5,nLev=3);
   data &data;
      drop i j;
      %if &nCont>0  %then %do; array x{&nCont}  x1-x&nCont; %end;
      %if &nClass>0 %then %do; array c{&nClass} c1-c&nClass;%end;
      do i = 1 to &nObs;
         %if &nCont>0 %then %do;
            do j= 1 to &nCont;
               x{j} = rannor(1);
            end;
         %end;
         %if &nClass > 0 %then %do;
            do j=1 to &nClass;
               if      mod(j,3) = 0 then c{j} = ranbin(1,&nLev,.6);
               else if mod(j,3) = 1 then c{j} = ranbin(1,&nLev,.5);
               else if mod(j,3) = 2 then c{j} = ranbin(1,&nLev,.4);
            end;
         %end;
         output;
      end;
   run;
%mend;

%makeRegressorData(data=traindata,nObs=500,nCont=5,nClass=5,nLev=3);
```

In the generated data, there are five continuous effects, x1–x5, and five CLASS variables, c1–c5. The following DATA step generates a response, y, that is dependent on x1–x4 and c1:

```
%macro AddDepVar(data=,modelRHS =,errorStd = 1);
   data &data;
      set &data;
      y = &modelRHS + &errorStd * rannor(1);
   run;
%mend;

%AddDepVar(data    = traindata,
           modelRHS= x1 +
                     0.1*x2 - 0.1*x3 - 0.01* x4 -
                     c1,
           errorStd= 1);
```

The effects x1 and c1 have a stronger influence on the response y than x2 and x3, and x4 has the weakest influence. For the continuous effect x1, a spline effect is constructed using the cubic B-spline basis (see the section "EFFECT Statement" on page 401 in Chapter 19, "Shared Concepts and Topics") as follows:

```
effect s1=spline(x1)
```

Models that contain spline effects are particularly good candidates for group LASSO selection, because the individual parameters in a spline effect have no meaning by themselves. You need them all to be in the model together in order to define the correct spline term in the model.

To have a basis for comparison, first use the following statements to apply LASSO to model selection:

```
ods graphics on;

proc glmselect data=traindata plots=coefficients;
  class c1-c5/split;
  effect s1=spline(x1/split);
  model y = s1 x2-x5 c:/
        selection=lasso(steps=20 choose=sbc);
run;
```

In LASSO selection, effects that have multiple parameters are split into their constituent parameters. In this problem, the spline effect has seven parameters by default and each of the five CLASS effects has four parameters. So together with the intercept and four continuous main effects, your model has a total of 32 parameters, as shown in the "Dimensions" table in Output 49.8.1. Note that the number of effects in the table is 17, because the spline effect s1 is counted as 7 effects.

Output 49.8.1 Dimensions

The GLMSELECT Procedure

Dimensions	
Number of Effects	17
Number of Effects after Splits	32
Number of Parameters	32

Output 49.8.2 shows the "LASSO Selection Summary" table, and Output 49.8.3 shows the standardized coefficients of all the effects selected at some step of the LASSO method, plotted as a function of the step number. Because you specified CHOOSE=SBC to pick the best model, the SBC value for the model at each step is also displayed in Output 49.8.3, with the minimum occurring at step 12. Hence the model at this step is selected, resulting in 13 selected effects. The "Parameter Estimates" table in Figure 49.8.4 shows that x2 and x3 as well as parts of the spline effect and parts of three CLASS effects are selected for the final model. So this selection misses the small true effect of x4 and erroneously includes c3 and c4 because of noise.

Output 49.8.2 LASSO Selection Summary Table

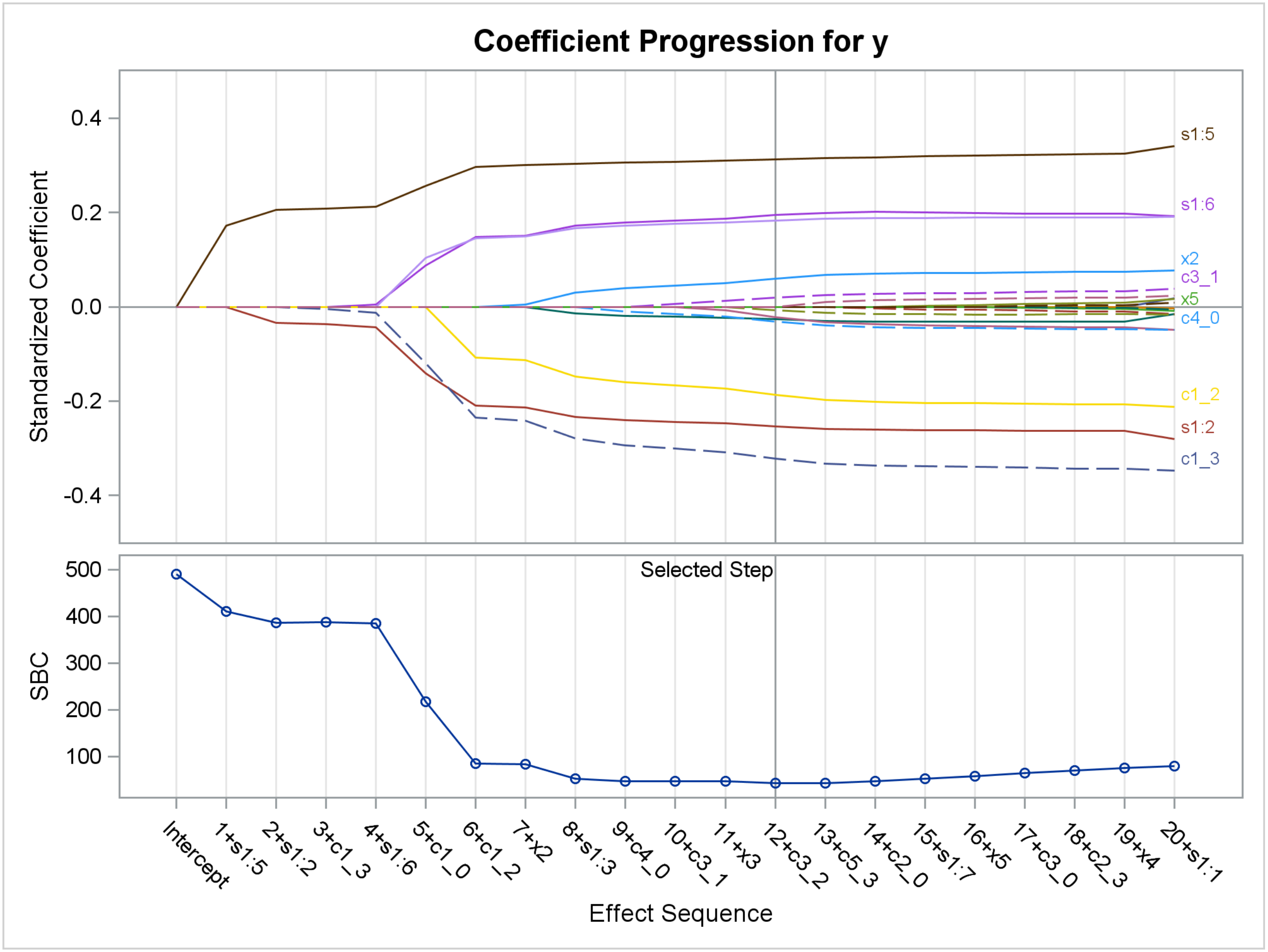
The GLMSELECT Procedure

LASSO Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	SBC
0	Intercept		1	491.1109
1	s1:5		2	410.2008
2	s1:2		3	386.7278
3	c1_3		4	388.1524
4	s1:6		5	385.0045
5	c1_0		6	216.9169
6	c1_2		7	84.3459
7	x2		8	82.8547
8	s1:3		9	52.3145
9	c4_0		10	46.8144
10	c3_1		11	46.6228
11	x3		12	46.2866
12	c3_2		13	42.7335*
13	c5_3		14	42.8659
14	c2_0		15	47.1151
15	s1:7		16	52.5826
16	x5		17	58.3346
17	c3_0		18	63.8812
18	c2_3		19	69.4954
19	x4		20	75.6385
20	s1:1		21	79.8007

* Optimal Value of Criterion

Output 49.8.3 LASSO Coefficient Progression Plot



Output 49.8.4 LASSO Parameter Estimates

Parameter Estimates

Parameter	DF	Estimate
Intercept	1	-1.636852
s1:2	1	-3.829173
s1:3	1	-0.192499
s1:5	1	2.259207
s1:6	1	3.040642
x2	1	0.098525
x3	1	-0.033892
c1_0	1	0.852990
c1_2	1	-0.635575
c1_3	1	-1.643039
c3_1	1	0.070937
c3_2	1	-0.022502
c4_0	1	-0.149375

As opposed to what happens in LASSO selection, spline effects and classification effects are not split by default in group LASSO selection. In addition, you can use a collection effect to construct a group of three of the continuous effects, as shown in the following statements:

```
proc glmselect data=traindata plots=coefficients;
  class c1-c5;
  effect s1=spline(x1);
  effect s2=collection(x2 x3 x4);
  model y = s1 s2 x5 c:/
        selection=grouplasso(steps=20 choose=sbc rho=0.8);
run;
```

Because spline and classification effects are not split by default in group LASSO selection, this model contains nine effects: s1, s2, x5, c1–c5, and the intercept. The number of parameters is the same as before, as shown in Output 49.8.5.

Output 49.8.5 Dimensions

The GLMSELECT Procedure

Dimensions	
Number of Effects	9
Number of Parameters	32

The RHO=0.8 option specifies the value of ρ for determining the regularization parameter $\lambda = \rho^i$ used in the ith step of the group LASSO selection process. If you use a larger value for ρ, you can expect the plot shown in Output 49.8.7 to exhibit a finer coefficient progression.

Output 49.8.6 shows the “Group LASSO Selection Summary” table, and Output 49.8.7 shows the standardized coefficients of all the effects selected at some step of the group LASSO method, plotted as a function of the step number. As you can see in this plot, the CHOOSE=SBC option selects the model at step 10 as the one that has the minimum value of the SBC statistic. The resulting model contains the four effects, including all the true ones, and 15 parameters, as shown in the “Parameter Estimates” table in Output 49.8.8.

Unlike LASSO, which adds or drops an effect at each step, the group LASSO method can add or drop more than one effect. Indeed, you can see in Output 49.8.7 that steps 11 and 16 each added two effects to the model. Simple selection breaks down because group LASSO does not admit a piecewise linear constant solution path, as regular LASSO does.

Output 49.8.6 Group LASSO Selection Summary Table

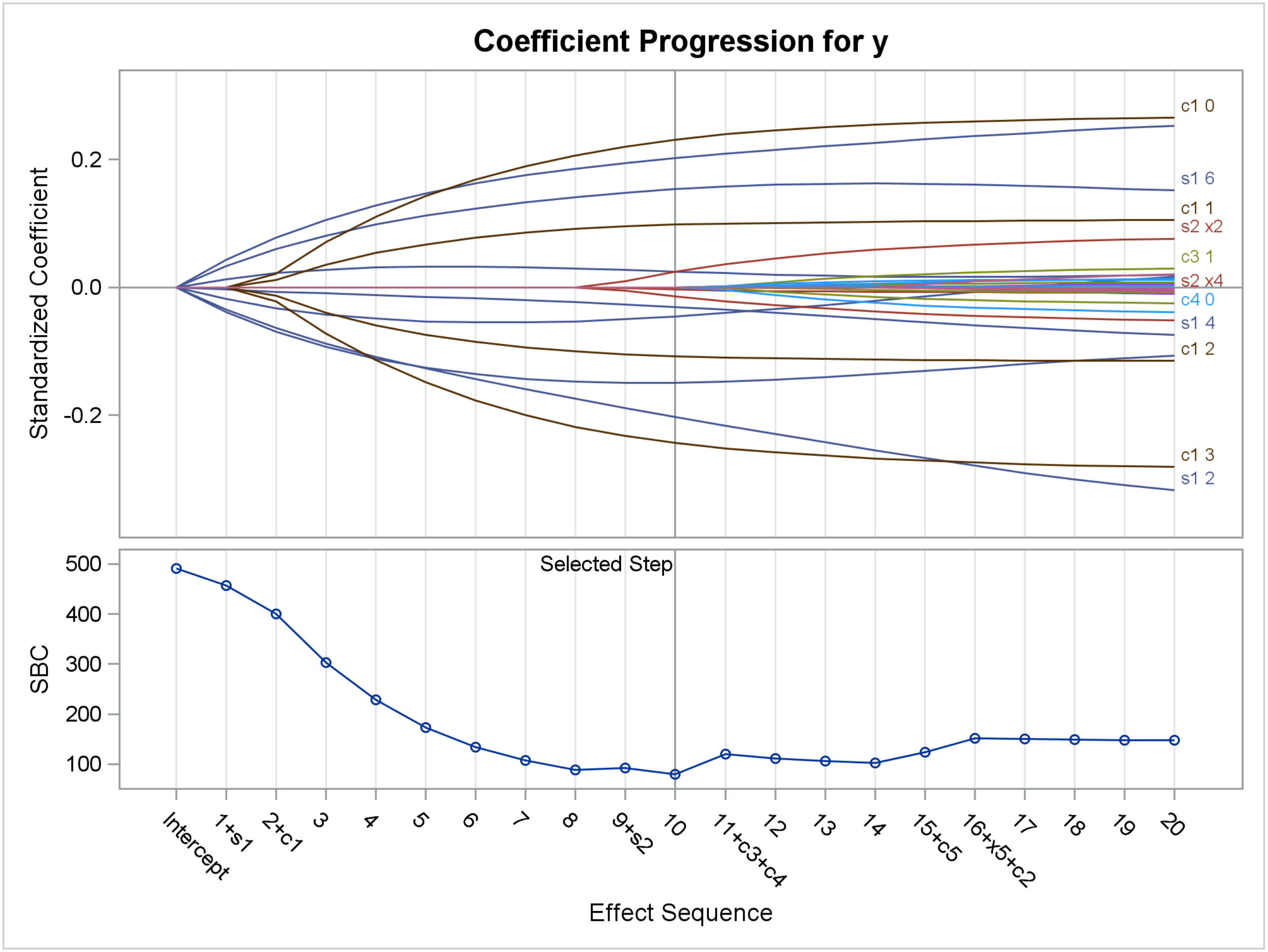
The GLMSELECT Procedure

Group LASSO Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	SBC
0	Intercept		1	491.1109
1	s1		2	457.0038
2	c1		3	400.1165
3			3	302.8975
4			3	228.2883
5			3	173.3913
6			3	134.3346
7			3	107.2038
8			3	88.6178
9	s2		4	91.9567
10			4	79.7022*
11	c3 c4		6	119.8826
12			6	111.5926
13			6	105.8915
14			6	101.9617
15	c5		7	123.6787
16	x5 c2		9	152.2387
17			9	150.4210
18			9	149.1847
19			9	148.3496
20			9	147.7893

* Optimal Value of Criterion

Output 49.8.7 LASSO Coefficient Progression Plot



Output 49.8.8 Group LASSO Parameter Estimates

Parameter Estimates

Parameter		**Estimate**
Intercept		-1.439483
s1	**1**	-5.320998
s1	**2**	-3.058403
s1	**3**	-1.090183
s1	**4**	-0.240067
s1	**5**	1.458828
s1	**6**	2.404019
s1	**7**	3.023258
s2	**x2**	0.040036
s2	**x3**	-0.022261
s2	**x4**	-0.005458
c1	**0**	1.075888
c1	**1**	0.326289
c1	**2**	-0.366125
c1	**3**	-1.240907

References

Breiman, L. (1992). “The Little Bootstrap and Other Methods for Dimensionality Selection in Regression: X-Fixed Prediction Error.” *Journal of the American Statistical Association* 87:738–754.

Burnham, K. P., and Anderson, D. R. (2002). *Model Selection and Multimodel Inference*. 2nd ed. New York: Springer-Verlag.

Darlington, R. B. (1968). “Multiple Regression in Psychological Research and Practice.” *Psychological Bulletin* 69:161–182.

Donoho, D. L., and Johnstone, I. M. (1994). “Ideal Spatial Adaptation via Wavelet Shrinkage.” *Biometrika* 81:425–455.

Draper, N. R., Guttman, I., and Kanemasu, H. (1971). “The Distribution of Certain Regression Statistics.” *Biometrika* 58:295–298.

Efron, B., Hastie, T. J., Johnstone, I. M., and Tibshirani, R. (2004). “Least Angle Regression (with Discussion).” *Annals of Statistics* 32:407–499.

Efron, B., and Tibshirani, R. J. (1993). *An Introduction to the Bootstrap*. New York: Chapman & Hall.

Eilers, P. H. C., and Marx, B. D. (1996). “Flexible Smoothing with *B*-Splines and Penalties.” *Statistical Science* 11:89–121. With discussion.

El Ghaoui, L., Viallon, V., and Rabbani, T. (2012). “Safe Feature Elimination for the Lasso and Sparse Supervised Learning Problems.” *Pacific Journal of Optimization* 8:667–698. Special issue on conic optimization.

Fan, J., and Lv, J. (2008). “Sure Independence Screening for Ultrahigh Dimensional Feature Space.” *Journal of the Royal Statistical Society, Series B* 70:849–911.

Foster, D. P., and Stine, R. A. (2004). “Variable Selection in Data Mining: Building a Predictive Model for Bankruptcy.” *Journal of the American Statistical Association* 99:303–313.

Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., Coller, H., Loh, M., Downing, J. R., Caligiuri, M. A., Bloomfield, C. D., and Lander, E. S. (1999). “Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression.” *Science* 286:531–537.

Harrell, F. E. (2001). *Regression Modeling Strategies*. New York: Springer-Verlag.

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

Hocking, R. R. (1976). “The Analysis and Selection of Variables in a Linear Regression.” *Biometrics* 32:1–50.

Hurvich, C. M., Simonoff, J. S., and Tsai, C.-L. (1998). “Smoothing Parameter Selection in Nonparametric Regression Using an Improved Akaike Information Criterion.” *Journal of the Royal Statistical Society, Series B* 60:271–293.

Hurvich, C. M., and Tsai, C.-L. (1989). "Regression and Time Series Model Selection in Small Samples." *Biometrika* 76:297–307.

Judge, G. G., Griffiths, W. E., Hill, R. C., Lütkepohl, H., and Lee, T.-C. (1985). *The Theory and Practice of Econometrics*. 2nd ed. New York: John Wiley & Sons.

Liu, J., Zhao, Z., Wang, J., and Ye, J. (2014). "Safe Screening with Variational Inequalities and Its Application to Lasso." In *JMLR Workshop and Conference Proceedings, Vol. 32: Proceedings of the Thirty-First International Conference on Machine Learning, Second Cycle*. http://jmlr.org/proceedings/papers/v32/liuc14.pdf.

Mallows, C. L. (1967). "Choosing a Subset Regression." Bell Telephone Laboratories.

Mallows, C. L. (1973). "Some Comments on C_p." *Technometrics* 15:661–675.

Miller, A. J. (2002). *Subset Selection in Regression.* Vol. 95 of Monographs on Statistics and Applied Probability. 2nd ed. Boca Raton, FL: Chapman & Hall/CRC.

Nesterov, Y. (2013). "Gradient Methods for Minimizing Composite Objective Function." *Mathematical Programming* 140:125–161.

Osborne, M. R., Presnell, B., and Turlach, B. A. (2000). "A New Approach to Variable Selection in Least Squares Problems." *IMA Journal of Numerical Analysis* 20:389–404.

Raftery, A. E., Madigan, D., and Hoeting, J. A. (1997). "Bayesian Model Averaging for Linear Regression Models." *Journal of the American Statistical Association* 92:179–191.

Reichler, J. L., ed. (1987). *The 1987 Baseball Encyclopedia Update*. New York: Macmillan.

Sarle, W. S. (2001). "Donoho-Johnstone Benchmarks: Neural Net Results." Accessed March 27, 2007. ftp://ftp.sas.com/pub/neural/dojo/dojo.html.

Sawa, T. (1978). "Information Criteria for Discriminating among Alternative Regression Models." *Econometrica* 46:1273–1282.

Schwarz, G. (1978). "Estimating the Dimension of a Model." *Annals of Statistics* 6:461–464.

Tibshirani, R. (1996). "Regression Shrinkage and Selection via the Lasso." *Journal of the Royal Statistical Society, Series B* 58:267–288.

Tibshirani, R., Bien, J., Friedman, J., Hastie, T., Simon, N., Taylor, J., and Tibshirani, R. (2012). "Strong Rules for Discarding Predictors in Lasso-Type Problems." *Journal of the Royal Statistical Society, Series B* 74:245–266.

Time Inc. (1987). "What They Make." *Sports Illustrated* (April 20): 54–81.

Wang, J., Zhou, J., Wonka, P., and Ye, J. (2013). "Lasso Screening Rules via Dual Polytope Projection." In *Advances in Neural Information Processing Systems 26*, edited by C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, 1070–1078. La Jolla, CA: Neural Information Processing Systems Foundation.

Xiang, Z. J., Xu, H., and Ramadge, P. J. (2011). "Learning Sparse Representations of High Dimensional Data on Large Scale Dictionaries." In *Advances in Neural Information Processing Systems 24*, edited by J. Shawe-Taylor, R. S. Zemel, P. Bartlett, F. C. N. Pereira, and K. Q. Weinberger, 900–908. La Jolla, CA: Neural Information Processing Systems Foundation.

Yuan, M., and Lin, L. (2006). "Model Selection and Estimation in Regression with Grouped Variables." *Journal of the Royal Statistical Society, Series B* 68:49–67.

Zou, H. (2006). "The Adaptive Lasso and Its Oracle Properties." *Journal of the American Statistical Association* 101:1418–1429.

Zou, H., and Hastie, T. (2005). "Regularization and Variable Selection via the Elastic Net." *Journal of the Royal Statistical Society, Series B* 67:301–320.

Subject Index

Syntax Index