SAS/STAT® 13.2 User's Guide

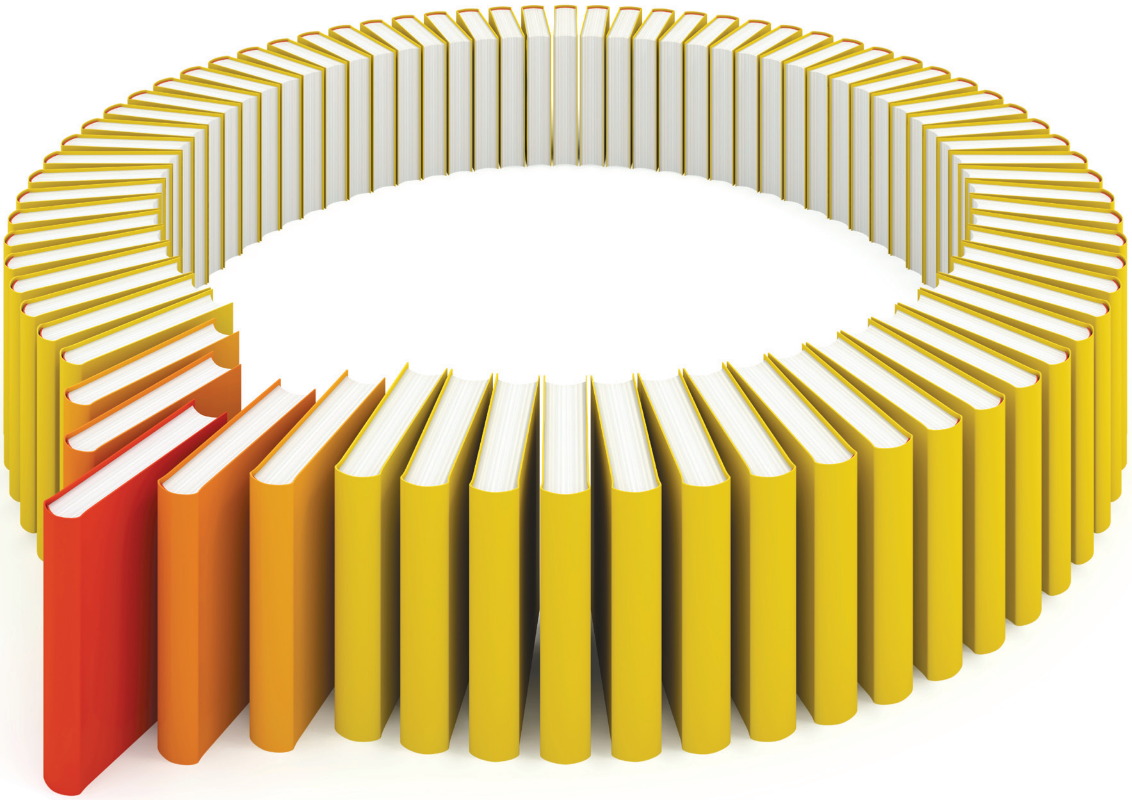
The GLMMOD Procedure

This document is an individual chapter from *SAS/STAT® 13.2 User's Guide*.

The correct bibliographic citation for the complete manual is as follows: SAS Institute Inc. 2014. *SAS/STAT® 13.2 User's Guide*. Cary, NC: SAS Institute Inc.

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

August 2014



Chapter 46
The GLMMOD Procedure

Contents

Overview: GLMMOD Procedure

The GLMMOD procedure constructs the design matrix for a general linear model; it essentially constitutes the model-building front end for the GLM procedure. You can use the GLMMOD procedure in conjunction with other SAS/STAT software regression procedures or with SAS/IML software to obtain specialized analyses for general linear models that you cannot obtain with the GLM procedure.

While some of the regression procedures in SAS/STAT software provide for general linear effects modeling with classification variables and interaction or polynomial effects, many others do not. For such procedures, you must specify the model directly in terms of distinct variables. For example, if you want to use the REG procedure to fit a polynomial model, you must first create the crossproduct and power terms as new variables, usually in a DATA step. Alternatively, you can use the GLMMOD procedure to create a data set that contains the design matrix for a model as specified using the effects modeling facilities of the GLM procedure.

Note that the TRANSREG procedure provides alternative methods to construct design matrices for full-rank and less-than-full-rank models, polynomials, and splines. See Chapter 104, “The TRANSREG Procedure,” for more information.

Getting Started: GLMMOD Procedure

A One-Way Design

A one-way analysis of variance considers one treatment factor with two or more treatment levels. This example employs PROC GLMMOD together with PROC REG to perform a one-way analysis of variance to study the effect of bacteria on the nitrogen content of red clover plants. The treatment factor is bacteria strain, and it has six levels. Red clover plants are inoculated with the treatments, and nitrogen content is later measured in milligrams. The data are derived from an experiment by Erdman (1946) and are analyzed in Chapters 7 and 8 of Steel and Torrie (1980). PROC GLMMOD is used to create the design matrix. The following DATA step creates the SAS data set Clover.

```
title 'Nitrogen Content of Red Clover Plants';
data Clover;
   input Strain $ Nitrogen @@;
   datalines;
3DOK1  19.4 3DOK1  32.6 3DOK1  27.0 3DOK1  32.1 3DOK1  33.0
3DOK5  17.7 3DOK5  24.8 3DOK5  27.9 3DOK5  25.2 3DOK5  24.3
3DOK4  17.0 3DOK4  19.4 3DOK4   9.1 3DOK4  11.9 3DOK4  15.8
3DOK7  20.7 3DOK7  21.0 3DOK7  20.5 3DOK7  18.8 3DOK7  18.6
3DOK13 14.3 3DOK13 14.4 3DOK13 11.8 3DOK13 11.6 3DOK13 14.2
COMPOS 17.3 COMPOS 19.4 COMPOS 19.1 COMPOS 16.9 COMPOS 20.8
;
```

The variable Strain contains the treatment levels, and the variable Nitrogen contains the response. The following statements produce the design matrix:

```
proc glmmod data=Clover;
   class Strain;
   model Nitrogen = Strain;
run;
```

The classification variable, or treatment factor, is specified in the CLASS statement. The MODEL statement defines the response and independent variables. The design matrix produced corresponds to the model

$$Y_{i,j} = \mu + \alpha_i + \epsilon_{i,j}$$

where $i = 1, \ldots, 6$ and $j = 1, \ldots, 5$.

Figure 46.1 and Figure 46.2 display the output produced by these statements. Figure 46.1 displays information about the data set, which is useful for checking your data.

Figure 46.1 Class Level Information and Parameter Definitions

Nitrogen Content of Red Clover Plants

The GLMMOD Procedure

Class Level Information		
Class	**Levels**	**Values**
Strain	6	3DOK1 3DOK13 3DOK4 3DOK5 3DOK7 COMPOS

Number of Observations Read	30
Number of Observations Used	30

Parameter Definitions

		CLASS Variable Values
Column Number	**Name of Associated Effect**	**Strain**
1	Intercept	
2	Strain	3DOK1
3	Strain	3DOK13
4	Strain	3DOK4
5	Strain	3DOK5
6	Strain	3DOK7
7	Strain	COMPOS

The design matrix, shown in Figure 46.2, consists of seven columns: one for the mean and six for the treatment levels. The vector of responses, Nitrogen, is also displayed.

Figure 46.2 Design Matrix

Design Points

		Column Number						
Observation Number	**Nitrogen**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	19.4	1	1	0	0	0	0	0
2	32.6	1	1	0	0	0	0	0
3	27.0	1	1	0	0	0	0	0
4	32.1	1	1	0	0	0	0	0
5	33.0	1	1	0	0	0	0	0
6	17.7	1	0	0	0	1	0	0
7	24.8	1	0	0	0	1	0	0
8	27.9	1	0	0	0	1	0	0
9	25.2	1	0	0	0	1	0	0
10	24.3	1	0	0	0	1	0	0
11	17.0	1	0	0	1	0	0	0
12	19.4	1	0	0	1	0	0	0
13	9.1	1	0	0	1	0	0	0
14	11.9	1	0	0	1	0	0	0
15	15.8	1	0	0	1	0	0	0
16	20.7	1	0	0	0	0	1	0
17	21.0	1	0	0	0	0	1	0
18	20.5	1	0	0	0	0	1	0
19	18.8	1	0	0	0	0	1	0
20	18.6	1	0	0	0	0	1	0
21	14.3	1	0	1	0	0	0	0
22	14.4	1	0	1	0	0	0	0
23	11.8	1	0	1	0	0	0	0
24	11.6	1	0	1	0	0	0	0
25	14.2	1	0	1	0	0	0	0
26	17.3	1	0	0	0	0	0	1
27	19.4	1	0	0	0	0	0	1
28	19.1	1	0	0	0	0	0	1
29	16.9	1	0	0	0	0	0	1
30	20.8	1	0	0	0	0	0	1

Usually, you will find PROC GLMMOD most useful for the data sets it can create rather than for its displayed output. For example, the following statements use PROC GLMMOD to save the design matrix for the clover study to the data set CloverDesign instead of displaying it.

```
proc glmmod data=Clover outdesign=CloverDesign noprint;
   class Strain;
   model Nitrogen = Strain;
run;
```

Now you can use the REG procedure to analyze the data, as the following statements demonstrate:

```
proc reg data=CloverDesign;
   model Nitrogen = Col2-Col7;
run;
```

The results are shown in Figure 46.3.

Figure 46.3 Regression Analysis Using the REG Procedure

```
                  Nitrogen Content of Red Clover Plants

                          The REG Procedure
                           Model: MODEL1
                    Dependent Variable: Nitrogen

                   Number of Observations Read  30
                   Number of Observations Used  30

                          Analysis of Variance
                                  Sum of        Mean
  Source              DF         Squares      Square  F Value  Pr > F
  Model                5       847.04667   169.40933    14.37  <.0001
  Error               24       282.92800    11.78867
  Corrected Total     29      1129.97467

            Root MSE           3.43346  R-Square  0.7496
            Dependent Mean    19.88667  Adj R-Sq  0.6975
            Coeff Var         17.26515

Note: Model is not full rank. Least-squares solutions for the parameters are not unique. Some statistics will be misleading. A reported
      DF of 0 or B means that the estimate is biased.

Note: The following parameters have been set to 0, since the variables are a linear combination of other variables as shown.

            Col7 =  Intercept - Col2 - Col3 - Col4 - Col5 - Col6

                          Parameter Estimates
                                  Parameter  Standard
  Variable   Label           DF    Estimate     Error  t Value  Pr > |t|
  Intercept  Intercept        B    18.70000   1.53549    12.18   <.0001
  Col2       Strain 3DOK1     B    10.12000   2.17151     4.66   <.0001
  Col3       Strain 3DOK13    B    -5.44000   2.17151    -2.51   0.0194
  Col4       Strain 3DOK4     B    -4.06000   2.17151    -1.87   0.0738
  Col5       Strain 3DOK5     B     5.28000   2.17151     2.43   0.0229
  Col6       Strain 3DOK7     B     1.22000   2.17151     0.56   0.5794
  Col7       Strain COMPOS    0           0         .        .        .
```

Syntax: GLMMOD Procedure

The following statements are available in the GLMMOD procedure.

```
PROC GLMMOD < options > ;
    BY variables ;
    CLASS variables ;
    FREQ variable ;
    MODEL dependents = independents < / options > ;
    WEIGHT variable ;
```

The PROC GLMMOD and MODEL statements are required. If classification effects are used, the classification variables must be declared in a CLASS statement, and the CLASS statement must appear before the MODEL statement.

PROC GLMMOD Statement

PROC GLMMOD < *options* > ;

The PROC GLMMOD statement invokes the GLMMOD procedure. Table 46.1 summarizes the options available in the PROC GLMMOD statement.

Table 46.1 PROC GLMMOD Statement Options

Statement	Description
DATA=	Specifies the SAS data set to be used
NAMELEN=	Specifies the maximum length for an effect name
NOPRINT	Suppresses the normal display of results
OUTPARM=	Names an output data set describing the design matrix columns
OUTDESIGN=	Names an output data set to contain the columns of the design matrix
PREFIX=	Specifies a prefix to use in naming the columns of the design matrix
ZEROBASED	Modifies the numbering for the columns of the design matrix

It has the following options:

DATA=*SAS-data-set*

specifies the SAS data set to be used by the GLMMOD procedure. If you do not specify the DATA= option, the most recently created SAS data set is used.

NAMELEN=*n*

specifies the maximum length for an effect name. Effect names are listed in the table of parameter definitions and stored in the EFFNAME variable in the OUTPARM= data set. By default, $n = 20$. You can specify $20 < n \leq 200$ if 20 characters are not enough to distinguish between effects, which might be the case if the model includes a high-order interaction between variables with relatively long, similar names.

NOPRINT

suppresses the normal display of results. This option is generally useful only when one or more output data sets are being produced by the GLMMOD procedure. Note that this option temporarily disables the Output Delivery System (ODS); see Chapter 20, "Using the Output Delivery System," for more information.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the classification variables (which are specified in the CLASS statement).

This option applies to the levels for all classification variables, except when you use the (default) ORDER=FORMATTED option with numeric classification variables that have no explicit format. In that case, the levels of such variables are ordered by their internal value.

The ORDER= option can take the following values:

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 ORDER=FORMATTED and ORDER=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*.

OUTPARM=*SAS-data-set*

names an output data set to contain the information regarding the association between model effects and design matrix columns.

OUTDESIGN=*SAS-data-set*

names an output data set to contain the columns of the design matrix.

PREFIX=*name*

specifies a prefix to use in naming the columns of the design matrix in the OUTDESIGN= data set. The default prefix is Col and the column name is formed by appending the column number to the prefix, so that by default the columns are named Col1, Col2, and so on. If you specify the ZEROBASED option, the column numbering starts at zero, so that with the default value of PREFIX= the columns of the design matrix in the OUTDESIGN= data set are named Col0, Col1, and so on.

ZEROBASED

specifies that the numbering for the columns of the design matrix in the OUTDESIGN= data set begin at 0. By default it begins at 1, so that with the default value of PREFIX= the columns of the design matrix in the OUTDESIGN= data set are named Col1, Col2, and so on. If you use the ZEROBASED option, the column names are instead Col0, Col1, and so on.

BY Statement

BY *variables* **;**

You can specify a BY statement with PROC GLMMOD 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 GLMMOD 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 *variables* < / **TRUNCATE** > **;**

The CLASS statement names the classification variables to be used in the model. Typical classification variables are Treatment, Sex, Race, Group, and Replication. If you use the CLASS statement, it must appear before the MODEL statement.

Classification variables can be either character or numeric. By default, class levels are determined from the entire set of formatted values of the CLASS variables.

NOTE: Prior to SAS 9, class levels were determined by using no more than the first 16 characters of the formatted values. To revert to this previous behavior, you can use the TRUNCATE option in the CLASS statement.

In any case, you can use formats to group values into levels. See the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS*

Formats and Informats: Reference. You can adjust the order of CLASS variable levels with the ORDER= option in the PROC GLMMOD statement.

You can specify the following *option* in the CLASS statement after a slash (/):

TRUNCATE

specifies that class levels should be determined by using only up to the first 16 characters of the formatted values of CLASS variables. When formatted values are longer than 16 characters, you can use this option to revert to the levels as determined in releases prior to SAS 9.

FREQ and WEIGHT Statements

FREQ *variable* **;**

WEIGHT *variable* **;**

FREQ and WEIGHT variables are transferred to the output data sets without change.

MODEL Statement

MODEL *dependents* **=** *independents* < / *options* > **;**

The MODEL statement names the dependent variables and independent effects. For the syntax of effects, see the section “Specification of Effects” on page 3453 in Chapter 45, “The GLM Procedure.”

You can specify the following option in the MODEL statement after a slash (/):

NOINT

requests that the intercept parameter not be included in the model.

Details: GLMMOD Procedure

Displayed Output

For each pass of the data (that is, for each BY group and for each pass required by the pattern of missing values for the dependent variables), the GLMMOD procedure displays the definitions of the columns of the design matrix along with the following:

- the number of the column
- the name of the associated effect
- the values that the classification variables take for this level of the effect

The design matrix itself is also displayed, along with the following:

- the observation number
- the dependent variable values
- the FREQ and WEIGHT values, if any
- the columns of the design matrix

Missing Values

If some variables have missing values for some observations, then PROC GLMMOD handles missing values in the same way as PROC GLM; see the section "Missing Values" on page 3505 in Chapter 45, "The GLM Procedure," for further details.

OUTPARM= Data Set

An output data set containing information regarding the association between model effects and design matrix columns is created whenever you specify the OUTPARM= option in the PROC GLMMOD statement. The OUTPARM= data set contains an observation for each column of the design matrix with the following variables:

- a numeric variable, _COLNUM_, identifying the number of the column of the design matrix corresponding to this observation
- a character variable, EFFNAME, containing the name of the effect that generates the column of the design matrix corresponding to this observation
- the CLASS variables, with the values they have for the column corresponding to this observation, or blanks if they are not involved with the effect associated with this column

If there are BY-group variables or if the pattern of missing values for the dependent variables requires it, the single data set defines several design matrices. In this case, for each of these design matrices, the OUTPARM= data set also contains the following:

- the current values of the BY variables, if you specify a BY statement
- a numeric variable, _YPASS_, containing the current pass of the data, if the pattern of missing values for the dependent variables requires multiple passes

OUTDESIGN= Data Set

An output data set containing the design matrix is created whenever you specify the OUTDESIGN= option in the PROC GLMMOD statement. The OUTDESIGN= data set contains an observation for each observation in the DATA= data set, with the following variables:

- the dependent variables
- the FREQ variable, if any
- the WEIGHT variable, if any
- a variable for each column of the design matrix, with names COL1, COL2, and so forth

If there are BY-group variables or if the pattern of missing values for the dependent variables requires it, the single data set contains several design matrices. In this case, for each of these, the OUTDESIGN= data set also contains the following:

- the current values of the BY variables, if you specify a BY statement
- a numeric variable, _YPASS_, containing the current pass of the data, if the pattern of missing values for the dependent variables requires multiple passes

ODS Table Names

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

Table 46.2 ODS Tables Produced by PROC GLMMOD

ODS Table Name	Description	Statement
ClassLevels	Table of class levels	CLASS statement
DependentInfo	Simultaneously analyzed dependent variables	default when there are multiple dependent variables
DesignPoints	Design matrix	default
NObs	Number of observations	default
Parameters	Parameters and associated column numbers	default

Examples: GLMMOD Procedure

Example 46.1: A Two-Way Design

The following program uses the GLMMOD procedure to produce the design matrix for a two-way design. The two classification factors have seven and three levels, respectively, so the design matrix contains $1 + 7 + 3 + 21 = 32$ columns in all. Output 46.1.1, Output 46.1.2, and Output 46.1.3 display the output produced by the following statements.

```
data Plants;
   input Type $ @;
   do Block=1 to 3;
      input StemLength @;
      output;
   end;
   datalines;
Clarion  32.7 32.3 31.5
Clinton  32.1 29.7 29.1
Knox     35.7 35.9 33.1
O'Neill  36.0 34.2 31.2
Compost  31.8 28.0 29.2
Wabash   38.2 37.8 31.9
Webster  32.5 31.1 29.7
;

proc glmmod data=Plants outparm=Parm outdesign=Design;
   class Type Block;
   model StemLength = Type|Block;
run;

proc print data=Parm;
run;

proc print data=Design;
run;
```

Output 46.1.1 A Two-Way Design

The GLMMOD Procedure

Class Level Information

Class	Levels	Values
Type	7	Clarion Clinton Compost Knox O'Neill Wabash Webster
Block	3	1 2 3

Number of Observations Read	21
Number of Observations Used	21

Parameter Definitions

		CLASS Variable Values	
Column Number	**Name of Associated Effect**	**Type**	**Block**
1	Intercept		
2	Type	Clarion	
3	Type	Clinton	
4	Type	Compost	
5	Type	Knox	
6	Type	O'Neill	
7	Type	Wabash	
8	Type	Webster	
9	Block		1
10	Block		2
11	Block		3
12	Type*Block	Clarion	1
13	Type*Block	Clarion	2
14	Type*Block	Clarion	3
15	Type*Block	Clinton	1
16	Type*Block	Clinton	2
17	Type*Block	Clinton	3
18	Type*Block	Compost	1
19	Type*Block	Compost	2
20	Type*Block	Compost	3
21	Type*Block	Knox	1
22	Type*Block	Knox	2
23	Type*Block	Knox	3
24	Type*Block	O'Neill	1
25	Type*Block	O'Neill	2
26	Type*Block	O'Neill	3
27	Type*Block	Wabash	1
28	Type*Block	Wabash	2
29	Type*Block	Wabash	3
30	Type*Block	Webster	1
31	Type*Block	Webster	2
32	Type*Block	Webster	3

Output 46.1.1 *continued*

Design Points

Observation Number	StemLength
1	32.7
2	32.3
3	31.5
4	32.1
5	29.7
6	29.1
7	35.7
8	35.9
9	33.1
10	36.0
11	34.2
12	31.2
13	31.8
14	28.0
15	29.2
16	38.2
17	37.8
18	31.9
19	32.5
20	31.1
21	29.7

Output 46.1.1 *continued*

Design Points

	Column Number																															
Observation Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
1	1	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	1	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	1	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
8	1	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
9	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
10	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
11	1	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
12	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
13	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
15	1	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
16	1	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
17	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
18	1	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
19	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
20	1	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
21	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Output 46.1.2 The OUTPARM= Data Set

Obs	_COLNUM_	EFFNAME	Type	Block
1	1	Intercept		
2	2	Type	Clarion	
3	3	Type	Clinton	
4	4	Type	Compost	
5	5	Type	Knox	
6	6	Type	O'Neill	
7	7	Type	Wabash	
8	8	Type	Webster	
9	9	Block		1
10	10	Block		2
11	11	Block		3
12	12	Type*Block	Clarion	1
13	13	Type*Block	Clarion	2
14	14	Type*Block	Clarion	3
15	15	Type*Block	Clinton	1
16	16	Type*Block	Clinton	2
17	17	Type*Block	Clinton	3
18	18	Type*Block	Compost	1
19	19	Type*Block	Compost	2
20	20	Type*Block	Compost	3
21	21	Type*Block	Knox	1
22	22	Type*Block	Knox	2
23	23	Type*Block	Knox	3
24	24	Type*Block	O'Neill	1
25	25	Type*Block	O'Neill	2
26	26	Type*Block	O'Neill	3
27	27	Type*Block	Wabash	1
28	28	Type*Block	Wabash	2
29	29	Type*Block	Wabash	3
30	30	Type*Block	Webster	1
31	31	Type*Block	Webster	2
32	32	Type*Block	Webster	3

Output 46.1.3 The OUTDESIGN= Data Set

Obs	StemLength	Col1	Col2	Col3	Col4	Col5	Col6	Col7	Col8	Col9	Col10	Col11	Col12	Col13	Col14	Col15	Col16
1	32.7	1	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0
2	32.3	1	1	0	0	0	0	0	0	0	1	0	0	1	0	0	0
3	31.5	1	1	0	0	0	0	0	0	0	0	1	0	0	1	0	0
4	32.1	1	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0
5	29.7	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	1
6	29.1	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0
7	35.7	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0
8	35.9	1	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
9	33.1	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0
10	36.0	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0
11	34.2	1	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0
12	31.2	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
13	31.8	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0
14	28.0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0
15	29.2	1	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0
16	38.2	1	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0
17	37.8	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0
18	31.9	1	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0
19	32.5	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
20	31.1	1	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0
21	29.7	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0

Obs	Col17	Col18	Col19	Col20	Col21	Col22	Col23	Col24	Col25	Col26	Col27	Col28	Col29	Col30	Col31	Col32
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
13	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Example 46.2: Factorial Screening

Screening experiments are undertaken to select from among the many possible factors that might affect a response the few that actually do, either simply (main effects) or in conjunction with other factors (interactions). One method of selecting significant factors is forward model selection, in which the model is built by successively adding the most statistically significant effects. Forward selection is an option in the REG procedure, but the REG procedure does not allow you to specify interactions directly (as the GLM procedure does, for example). You can use the GLMMOD procedure to create the screening model for a design and then use the REG procedure on the results to perform the screening.

The following statements create the SAS data set Screening, which contains the results of a screening experiment:

```
title 'PROC GLMMOD and PROC REG for Forward Selection Screening';
data Screening;
   input a b c d e y;
   datalines;
-1 -1 -1 -1  1  -6.688
-1 -1 -1  1 -1 -10.664
-1 -1  1 -1 -1  -1.459
-1 -1  1  1  1   2.042
-1  1 -1 -1 -1  -8.561
-1  1 -1  1  1  -7.095
-1  1  1 -1  1   0.553
-1  1  1  1 -1  -2.352
 1 -1 -1 -1 -1  -4.802
 1 -1 -1  1  1   5.705
 1 -1  1 -1  1  14.639
 1 -1  1  1 -1   2.151
 1  1 -1 -1  1   5.884
 1  1 -1  1 -1  -3.317
 1  1  1 -1 -1   4.048
 1  1  1  1  1  15.248
;
```

The data set contains a single dependent variable (y) and five independent factors (a, b, c, d, and e). The design is a half-fraction of the full 2^5 factorial, the precise half-fraction having been chosen to provide uncorrelated estimates of all main effects and two-factor interactions.

The following statements use the GLMMOD procedure to create a design matrix data set containing all the main effects and two-factor interactions for the preceding screening design.

```
ods output DesignPoints = DesignMatrix;
proc glmmod data=Screening;
   model y = a|b|c|d|e@2;
run;
```

Notice that the preceding statements use ODS to create the design matrix data set, instead of the OUTDESIGN= option in the PROC GLMMOD statement. The results are equivalent, but the columns of the data set produced by ODS have names that are directly related to the names of their corresponding effects.

Finally, the following statements use the REG procedure to perform forward model selection for the screening design. Two MODEL statements are used, one without the selection options (which produces the regression analysis for the full model) and one with the selection options. Output 46.2.1 and Output 46.2.2 show the results of the PROC REG analysis.

```
proc reg data=DesignMatrix;
   model y = a--d_e;
   model y = a--d_e / selection = forward
                      details   = summary
                      slentry   = 0.05;
run;
```

Output 46.2.1 PROC REG Full Model Fit

PROC GLMMOD and PROC REG for Forward Selection Screening

The REG Procedure
Model: MODEL1
Dependent Variable: y

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	15	861.48436	57.43229	.	.
Error	0	0	.		
Corrected Total	15	861.48436			

Root MSE	.	**R-Square**	1.0000
Dependent Mean	0.33325	**Adj R-Sq**	.
Coeff Var	.		

Output 46.2.1 *continued*

Parameter Estimates

Variable	Label	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	Intercept	1	0.33325	.	.	.
a		1	4.61125	.	.	.
b		1	0.21775	.	.	.
a_b	a*b	1	0.30350	.	.	.
c		1	4.02550	.	.	.
a_c	a*c	1	0.05150	.	.	.
b_c	b*c	1	-0.20225	.	.	.
d		1	-0.11850	.	.	.
a_d	a*d	1	0.12075	.	.	.
b_d	b*d	1	0.18850	.	.	.
c_d	c*d	1	0.03200	.	.	.
e		1	3.45275	.	.	.
a_e	a*e	1	1.97175	.	.	.
b_e	b*e	1	-0.35625	.	.	.
c_e	c*e	1	0.30900	.	.	.
d_e	d*e	1	0.30750	.	.	.

Output 46.2.2 PROC REG Screening Results

Summary of Forward Selection

Step	Variable Entered	Label	Number Vars In	Partial R-Square	Model R-Square	C(p)	F Value	Pr > F
1	a		1	0.3949	0.3949	.	9.14	0.0091
2	c		2	0.3010	0.6959	.	12.87	0.0033
3	e		3	0.2214	0.9173	.	32.13	0.0001
4	a_e	a*e	4	0.0722	0.9895	.	75.66	<.0001

The full model has 16 parameters (the intercept + 5 main effects + 10 interactions). These are all estimable, but since there are only 16 observations in the design, there are no degrees of freedom left to estimate error; consequently, there is no way to use the full model to test for the statistical significance of effects. However, the forward selection method chooses only four effects for the model: the main effects of factors a, c, and e, and the interaction between a and e. Using this reduced model enables you to estimate the underlying level of noise, although note that the selection method biases this estimate somewhat.

References

Erdman, L. W. (1946), “Studies to Determine If Antibiosis Occurs among Rhizobia,” *Journal of the American Society of Agronomy*, 38, 251–258.

Steel, R. G. D. and Torrie, J. H. (1980), *Principles and Procedures of Statistics*, 2nd Edition, New York: McGraw-Hill.

Subject Index

Syntax Index