Paper 437-2013

Having an EFFECT: More General Linear Modeling and Analysis with the New EFFECT Statement in SAS/STAT® Software

Phil Gibbs, Randy Tobias, Kathleen Kiernan, and Jill Tao, SAS Institute Inc.

ABSTRACT

Linear models relate a response to a linear function of a design matrix **X**. General linear models, long available in standard SAS/STAT® 9.3 procedures such as GLM and MIXED, incorporate classification, interaction, and crossproduct effects to define **X**. The new EFFECT statement, which is available in many SAS/STAT 9.3 procedures, extends how you can define **X**. It enables you to fit models with nonparametric regression effects, crossover and carryover effects, and complicated inheritance effects.

This paper first shows how the EFFECT statement fits into the general architecture of SAS/STAT linear modeling tools and then explains and demonstrates specific *effect-types*. You will see how this powerful new feature easily enhances the statistical analyses that you can perform.

INTRODUCTION

SAS/STAT software has many tools for fitting, analyzing, and using linear models. A linear model consists of more than just simple linear relationships between a response and the predictors. Rather, a general linear model is linear in carefully chosen functions of the predictors. Linear models in statistics are powerful because many phenomena can be practically, if approximately, represented by such general models.

Many SAS/STAT procedures already have very versatile ways of defining such useful functions of the predictors, namely, with classification effects for grouping observed responses and with interaction and crossproduct effects for describing the joint effect of several predictors. The purpose of this paper is to introduce and demonstrate a new linear modeling tool that is available in many SAS/STAT 9.22 procedures. That tool is the EFFECT statement.

The EFFECT statement gives you an easy way to add more complex modeling terms to your linear model. Such terms can include classification beyond simple grouping (so-called *multimember* and *lag* effects), continuous modeling beyond simple polynomials (*polynomial* and *spline* effects), and general terms that you define yourself (*collection* effects). Using these new types of terms enables you to easily fit models that more closely match the real behavior of your data. Moreover, SAS/STAT also recognizes these types of linear modeling components during inferential analysis of the fit, enabling you to produce estimates, tests, and graphical summaries that are appropriate for their particular form. In particular, the EFFECT statement has been added to many of the modeling procedures in SAS/STAT software. The EFFECT statement also works well together with the new EFFECTPLOT statement, providing a way to visualize the relationship between these complex modeling terms and the response variable.

This paper briefly discusses the new *effect-types* introduced in the EFFECT statement. The paper illustrates the power and flexibility of the EFFECT statement through a series of examples.

GETTING STARTED: ADDING A POLYNOMIAL EFFECT TO A MODEL

A brief look at a concrete example demonstrates how to use the EFFECT statement and what it can do for you. Consider modeling the data that are shown in Figure 1. The sample plot is not a simple linear relationship between the variables **X** and **Y**; there is curvature present. A polynomial function of the predictor variable, **X**, enables you to define this curvature in a model. In order to capture the curvature present in Figure 1, you suspect that a seventh-degree polynomial is an appropriate fit. Therefore, the model includes **X** and all powers up to $\mathbf{X}^7$.

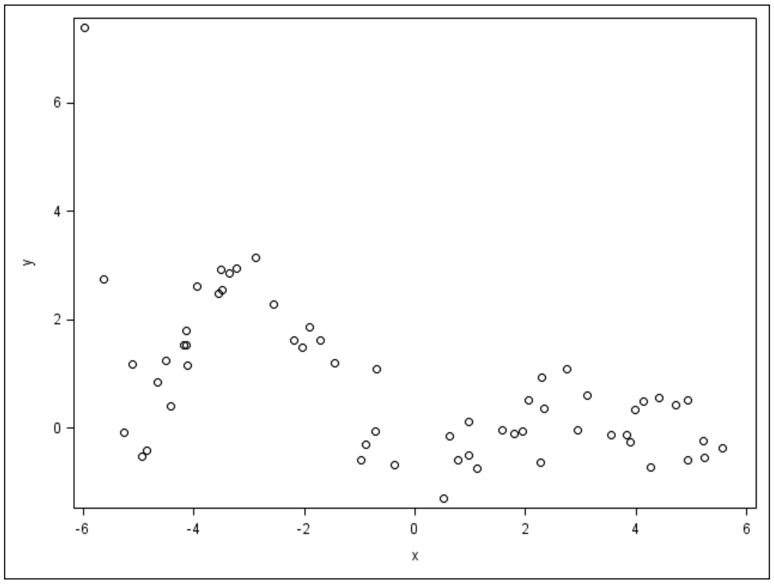

Figure 1. A Sample Plot of Data for a Polynomial Fit

If your model procedure allows for the specification of effects, similar to those in the GLM procedure, then you can define the model to fit a seventh-degree polynomial in the variable **X** as follows:

```
model y = x x*x x*x*x x*x*x*x x*x*x*x*x x*x*x*x*x*x x*x*x*x*x*x*x;
```

If you use this approach, then you must ensure that the largest term in the MODEL statement really does have a degree of precisely 7 and that all the intervening terms are included correctly. There can be numerical dangers, however, in fitting models with polynomial terms this large.

As an alternative, you can use the EFFECT statement to define this model with a more compact syntax:

```
effect p7=polynomial(x / degree=7);
 model y=p7;
```

Instead of defining the model in a single statement, this syntax uses two statements:

- the EFFECT statement to define the polynomial effect itself, called p7, formed from the variable **X** and having a degree of 7
- the MODEL statement to associate that p7 effect with the response variable **Y**

An EFFECTPLOT statement gives a quick check of how well the polynomial model performs, overlaying the fitted polynomial curve to the data, as shown in Figure 2. Note that using the EFFECT statement to define the polynomial model has an added benefit; special care is taken with some of the numerical pitfalls of fitting polynomials of high degree.

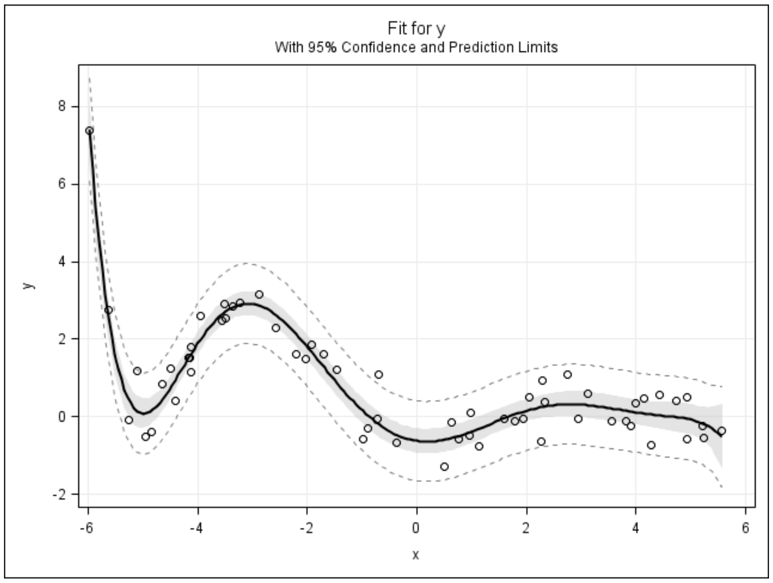

Figure 2. A Fitted Polynomial Effect

GETTING TO KNOW THE EFFECT STATEMENT

The general purpose of the EFFECT statement is to use designated variables from an existing data set to construct a set of columns to use in the regression design matrix. This is an extension of the method, familiar in many SAS/STAT procedures, of constructing design columns using CLASS variables. The EFFECT statement was first available as an experimental feature in SAS/STAT 9.22 and as a production statement in SAS/STAT 9.3. The statement is included in the following procedures:

GLIMMIX	GLMSELECT	HPMIXED
LOGISTIC	ORTHOREG	PHREG
PLS	QUANTREG	ROBUSTREG
SURVEYLOGISTIC	SURVEYREG	

The previous example demonstrates four general aspects of the EFFECT statement that you should keep in mind:

- An EFFECT statement does not itself define the model that is to be fitted. Rather, it defines components that you refer to in a MODEL statement, much as you would any variable.
- There are different types of EFFECT statements. The preceding example used a POLYNOMIAL *effect-type*. However, other *effect-types* are available.
- EFFECT statements might have a considerable amount of underlying processing, such as centering and scaling for polynomials. However, for other *effect-types*, such as SPLINE, the amount of processing can be considerably more than that.
- Procedures can do more with an EFFECT statement than only fit models. The preceding example showed how the EFFECTPLOT statement can plot the results of an EFFECT statement. EFFECT statements can also affect the results of Type 3 tests, LS-means, model predictions, and other procedure output.

This is the basic syntax for the EFFECT statement:

```
effect effect-name = effect-type (var-list < / effect-options >) ;
```

You can use the *effect-name* as you would use a variable in the MODEL statement of your modeling procedure. The *effect-name* can even be used in interactions with other traditionally formed effects in the model.

There are five *effect-types* in the EFFECT statement.

- SPLINE defines B-splines, truncated power function splines, and natural splines. Using such linear modeling components in your model-fitting repertoire can be very useful when working with data that might be nonlinear in nature.
- LAG constructs a new effect from an existing variable in the data set, where the value for the current observation in the new effect takes on the value of the previous observation from the existing variable. LAG effects are defined only within observations from the same subject. This *effect-type* is common in certain clinical-trial models, when treatments might have an impact not only for the current period but also for succeeding periods.
- POLYNOMIAL, as seen in the introductory example, constructs a set of columns that represent a polynomial expansion across one or more of the existing variables in a data set. Besides removing the need for cumbersome coding in the DATA step or MODEL statement, using polynomial effects can also be more numerically stable and can lead to more appropriate post-fit analyses (for example, with the EFFECTPLOT statement).
- MULTIMEMBER is used when observations in the input data can belong to more than one category. For example, this feature can be useful in modeling a teacher effect in education studies when students have multiple instructors.
- COLLECTION creates a model effect from an existing set of variables in the input data set. This *effect-type* groups individual predictors together. This feature can be used to force groups of variables to stay together during model selection or to test a joint effect of a set of predictors. Perhaps more importantly, a collection effect provides an alternative way for you to define your own *effect-types*.

IMPORTANT EFFECT STATEMENT OPTIONS

Each *effect-type* has options that enable you to add complexity to how your model fits. This section of the paper presents the syntax for the most important of these options. The "Examples" section of this paper explores these options more fully.

Here is the syntax for the LAG *effect-type*:

```
EFFECT lag-effect-name = LAG(effect-variable
                                  / WITHIN=(subject-variables)
                                    PERIOD=period-variable);
```

Both the WITHIN= and PERIOD= options are required in the EFFECT statement for a LAG *effect-type*. The WITHIN= option defines a single variable or set of variables that identify the subjects in your lag design. Each unique setting of the specified variables defines a new subject. Effects are lagged only within a level of subject, never across levels of subject. The PERIOD= option defines a variable that represents the ordering of the observations within a subject. The data are treated as if they are in period order when the lag effect is created.

Here is the syntax for the SPLINE *effect-type*:

```
EFFECT spline-effect-name = SPLINE(var-list </spline-options>);
```

The default spline basis for each variable that is listed in the SPLINE variable list is a cubic B-spline with three equally spaced knots. Both the BASIS= and DEGREE= options are useful for adjusting the SPLINE *effect-type*. BASIS= enables you to set the spline basis to either BSPLINE (the default) or TPF (truncated power function). DEGREE= sets the degree of the spline transformation. Essentially, you are setting the flexibility and the number of required parameters for the spline. The default value of DEGREE=3 provides for reasonable flexibility without requiring too many parameters to be estimated.

You can specify a polynomial effect with this syntax:

```
EFFECT polynomial-effect-name = POLY<nomial>(var-list </polynomial-options>);
```

This effect is constructed from design matrix columns that represent each term of the specified polynomial for each variable in *var-list*. You can control the degree of the polynomial fit with the DEGREE= option. By default, polynomials of DEGREE=1 are used. The STANDARDIZE= option is useful when the scale of the effect variable could lead to numerical stability issues.

The MULTIMEMBER *effect-type* uses the following syntax:

```
      EFFECT multimember-effect-name = MM(var-list </mm-options>);
```

You can use the WEIGHT= option to specify a list of variables that define a set of weights used to weigh the contributions of each of the classification variables that make up the multimember effect. For example, in the educational modeling example mentioned above, you might decide to weight different teachers' effects on student scores by how many hours of class time each teacher actually had with the students.

THE EFFECTPLOT STATEMENT: THE EFFICIENT AND TRUSTY COMPANION TO THE EFFECT STATEMENT

One way that you can assess the impact of a model effect is to use the new EFFECTPLOT statement. The EFFECTPLOT statement enables you to visually inspect the relationship between the response and either a traditional formed effect or a constructed effect created with the EFFECT statement. You can use the EFFECTPLOT statement directly in the GENMOD, LOGISTIC, or ORTHOREG procedures or in a post-model fit analysis with PROC PLM. The plots that are produced by this statement give you the ability to explore the model fit overlaid with the actual data and to slice out classification effects and grouping effects.

The EFFECTPLOT statement gives you the ability to generate a single plot that can show differences in your model effects. There are several *plot-types* available.

- FIT shows the fitted predicted curve plotted against a continuous variable from the model.
- BOX displays a box plot of the response against a classification effect.
- SLICEFIT extends the FIT plot to show the predicted fit grouped by the values of a classification effect in the model.
- CONTOUR shows a graph of the predicted fit against two continuous model variables.
- INTERACTION displays predicted values against multiple categorical variables.

EXAMPLES

The following examples illustrate the use of the various *effect-types* in the EFFECT statement. The examples also show how the EFFECTPLOT statement can complement model effects. Models with multiple EFFECT statements are also introduced.

ADDING A SPLINE EFFECT

Using a spline effect in your model is one common way to handle observed nonlinearity in your data. Splines are very flexible tools for improving model fit and are usually superior to adding polynomial terms, because they avoid scaling issues that can occur when accommodating powers of the covariates in a model. It could be argued that spline parameters are not directly interpretable, but so are those associated with high-order polynomial powers.

For example, you have sales data for three line items at a particular store. The data show a definite nonlinear trend over time, probably indicating a seasonal relationship (Figure 3).

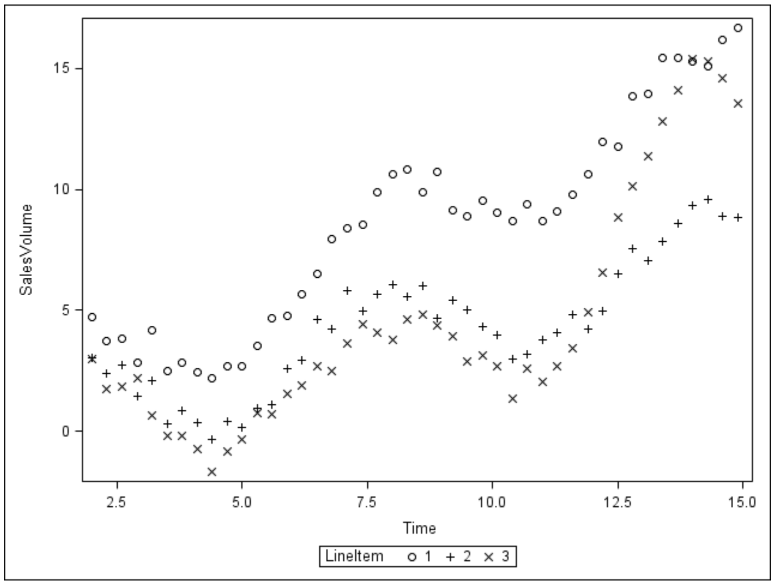

Figure 3. A Scatter Plot of Sales Data

You could try a variety of nonlinear transformations of time to create a good fit to these data, but the nonstandard form of the trend does not seem to promise much from this approach. This is where splines come in handy. You can use PROC ORTHOREG to fit this model:

```
proc orthoreg data=SalesData;
   class LineItem;
   effect SplTime=spline(Time);
   model SalesVolume = LineItem*SplTime;
   effectplot slicefit / clm;
run;
```

This ORTHOREG procedure code fits a model with an interaction between the line item and spline effect, which defines a separate spline transformation of time for each level of line item. The EFFECTPLOT statement generates a plot of the spline effect across each level of the sales line item (Figure 4).

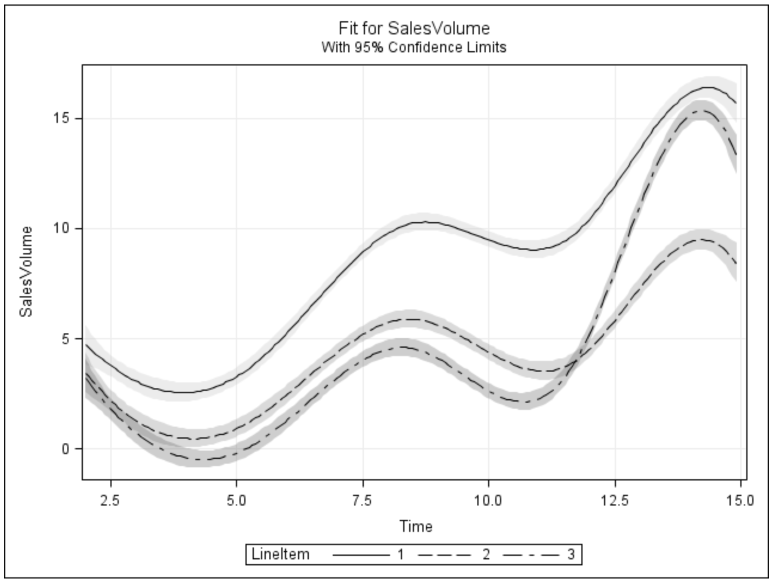

Figure 4. A Model Fit for Sales Volume

ADDING POLYNOMIAL EFFECTS

The example in the "Getting Started: Adding a Polynomial Effect to a Model" section introduced the use of the EFFECT statement to add polynomial effects to your models. That example also showed how much coding the EFFECT statement can save you when you add even a simple polynomial expansion. What if your data have more than one covariate for which you want to fit polynomial terms? The EFFECT statement can help with that task.

The following code simulates a data set that has two covariates, *x1* and *x2*. Both *x1* and *x2* are involved in predicting *y*, with *x1* used in a seventh-degree polynomial and *x2* in a second-degree polynomial, and with no interaction across those polynomials:

```
 data test;
   call streaminit(6432524);
   do rep=1 to 4000;
      x1=rand('uniform')*12 - 6;
      x2=rand('uniform')*12 - 6;
       y=(-1/5600)*((x1+5)**2)*(x1+1)*((x1-4)**3)*(x1-3)
         + (x2+4)**2 + rand('normal')*.5;
       output;
   end;
run;
```

You can use one EFFECT statement to model both covariates in a model, as shown in the following code:

```
proc orthoreg data=test;
  effect p3_x1_x2=polynomial(x1 x2 / degree=3);
  model y=p3_x1_x2;
run;
```

This model fits polynomials of degree 3 for both *x1* and *x2*. This model also includes the crossproduct effect of each term in *x1* with each term in *x2*. You can see this by viewing the parameter estimate table that is produced for this model, as shown in Table 1.

Parameter	DF	Parameter Estimate	Standard Error	*t* Value	Pr > |*t*|
Intercept	1	16.2916458053253	0.0363457344	448.24	<.0001
x1	1	-0.27694186577372	0.015180661	-18.24	<.0001
x2	1	8.02299481702874	0.0153379341	523.08	<.0001
x1^2	1	0.02003874608114	0.0017998641	11.13	<.0001
x1*x2	1	0.00104497607529	0.0015949378	0.66	0.5124
x2^2	1	0.99727764417477	0.0018085303	551.43	<.0001
x1^3	1	-0.00135028869352	0.0005876105	-2.30	0.0216
x1^2*x2	1	-0.00046102736184	0.0005213873	-0.88	0.3766
x1*x2^2	1	0.00094934371235	0.0005165785	1.84	0.0662
x2^3	1	-0.00097330837759	0.0005948148	-1.64	0.1019

Table 1. Parameter Estimates Output from PROC ORTHOREG

Here, you can see the polynomial terms for both *x1* and *x2*. The ^ operator indicates the exponent on each effect. You can also see the crossproduct between each term from each polynomial. None of the crossproduct terms nor the *x2^3* term are significant. This is as expected, because the simulation did not involve these effects.

To model the data more accurately, you can use two EFFECT statements—one to model the seventh-degree polynomial in *x1* and a second to model the second-degree polynomial in *x2*. The code for this model is as follows:

```
proc orthoreg data=test;
   effect p7_x1=polynomial(x1 / degree=7);
   effect p2_x2=polynomial(x2 / degree=2);
   model y=p7_x1 p2_x2;
run;
```

The parameter estimates for this model are displayed in Table 2, with all effects significant. Again, this significance is consistent with how the data are generated.

Parameter	**DF**	**Parameter Estimate**	**Standard Error**	***t* Value**	**Pr > |*t*|**
Intercept	1	15.1581018333388	0.0198051599	765.36	<.0001
x1	1	-0.26634812659721	0.0150751134	-17.67	<.0001
x1^2	1	0.54956341200902	0.0058446504	94.03	<.0001
x1^3	1	-0.08644770083848	0.0030026849	-28.79	<.0001
x1^4	1	-0.03878276815741	0.0004397655	-88.19	<.0001
x1^5	1	0.00841003633528	0.000172499	48.75	<.0001
x1^6	1	0.00071993917595	8.7460566E-6	82.32	<.0001
x1^7	1	-0.00018212159003	2.9205824E-6	-62.36	<.0001
x2	1	7.9976624909875	0.0022905256	3491.63	<.0001
x2^2	1	0.99909114176582	0.0007410834	1348.15	<.0001

Table 2. Parameter Estimates from PROC ORTHOREG

The EFFECTPLOT statement can still show the relationship between one of the covariates and the dependent variable. The AT option in the EFFECTPLOT statement enables you to control the values of the other covariates in the model for this 2-dimensional plot. Adding the following statement produces a plot of *y* versus *x1*, controlling for *x2=1* as shown in Figure 5.

```
effectplot fit / at (x2=1);
```

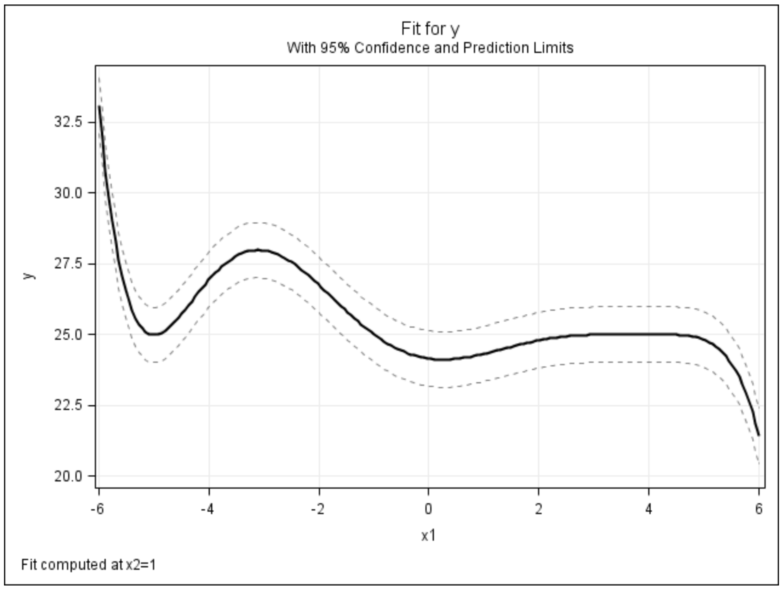

Figure 5. A Fit Plot for *y* versus *x1*, Controlling for *x2*

There is one last interesting point to mention about these constructed effects. The effects *p7_x1* and *p2_x2* can enter the model just like any other effect. In particular, you can specify interactions between these effects and any other effect in your model, including themselves! Adding *p7_x1*p2_x2* to the model includes the crossproducts of each pair of terms in the two polynomial expansions.

To see this general interaction feature in action, consider this example. You do not know whether the model should involve crossproducts between the two variables. The following code proposes a maximal potential model that includes all crossproducts between the two polynomials, and uses the GLMSELECT procedure to see whether they are really needed through the procedure's model selection process:

```
proc glmselect data=test;
      effect p7_x1=polynomial(x1 / degree=7);
      effect p2_x2=polynomial(x2 / degree=2);
      model y=p7_x1|p2_x2 / selection=stepwise;
run;
```

The preceding code gives a model with only the main effects for each polynomial term. GLMSELECT, through the model selection process, decided on its own that the crossproduct and interaction terms were not necessary. The final parameters chosen by the selection process are shown in Table 3.

Parameter Estimates				
Parameter	**DF**	**Estimate**	**Standard Error**	***t* Value**
Intercept	1	15.158102	0.019805	765.36
x1	1	-0.266348	0.015075	-17.67
x1^2	1	0.549563	0.005845	94.03
x1^3	1	-0.086448	0.003003	-28.79
x1^4	1	-0.038783	0.000440	-88.19
x1^5	1	0.008410	0.000172	48.75
x1^6	1	0.000720	0.000008746	82.32
x1^7	1	-0.000182	0.000002921	-62.36
x2	1	7.997662	0.002291	3491.63
x2^2	1	0.999091	0.000741	1348.15

Table 3. Parameter Estimates for Interacted Polynomials

ADDING LAG EFFECTS

In a crossover design, multiple treatments are given to a set of subjects, with the subjects randomized to a set of treatment sequences. One of the easiest crossover designs to study is a 2x2 design, in which two treatments are given to each subject. One treatment is given in the first period of the study, with the second treatment following in a second period. If the treatments are labeled as A and B, then each subject is randomized to one of the two treatment sequences AB or BA. Researchers often assume that treatments "washout"—that is, sufficient time passes between the two periods so that the treatment given in period 1 has no effect on the treatment given in period 2. If this treatment washout is not a reasonable assumption, then you should model the carryover effect of the first treatment on the second.

Grizzle (1965) presented a two-period, two-treatment, crossover design, and the data from his study are included in the following SAS® DATA step:

```
data Grizzle;
   input y Sequence Subject Period Treatment $ @@;
   cards;
 0.2 1 11 1 a   1.0 1 11 2 b
 0.0 1 12 1 a  -0.7 1 12 2 b
-0.8 1 13 1 a   0.2 1 13 2 b
 0.6 1 14 1 a   1.1 1 14 2 b
 0.3 1 15 1 a   0.4 1 15 2 b
 1.5 1 16 1 a   1.2 1 16 2 b
 1.3 2 21 1 b   0.9 2 21 2 a
-2.3 2 22 1 b   1.0 2 22 2 a
 0.0 2 23 1 b   0.6 2 23 2 a
-0.8 2 24 1 b  -0.3 2 24 2 a
-0.4 2 25 1 b  -1.0 2 25 2 a
-2.9 2 26 1 b   1.7 2 26 2 a
-1.9 2 27 1 b  -0.3 2 27 2 a
-2.9 2 28 1 b   0.9 2 28 2 a
;
```

Subjects in sequence 1 were given the treatments in AB order, whereas the subjects in sequence 2 received the treatments in BA order.

The old way to model the carryover effect involves creating a new variable in the data set that indicates the prior period's treatment value. This approach involves some programming and a fair amount of careful and selective interpretation of the analysis results. Alternatively, using the EFFECT statement enables you to replace all the programming with a single line that you add to your modeling procedure.

```
proc glimmix data=Grizzle;
   class Sequence Subject Period Treatment;
   effect Carryover=lag(Treatment / within=Subject period=Period);
   model y=Sequence Treatment Carryover / solution ddfm=kr htype=1;
   random Intercept / subject=Subject(Sequence);
run;
```

This model includes the effect Carryover, which represents the carryover effect between the two treatment periods in the design. The Type I tests on Carryover in Table 4 indicates that the carryover is significant in this study.

Type I Tests of Fixed Effects				
Effect	**Num DF**	**Den DF**	**F Value**	**Pr > F**
Sequence	1	24	4.07	0.0549
Treatment	1	24	4.58	0.0427
Carryover	1	24	5.56	0.0269

Table 4. Type I Tests for Crossover Design

ADDING MULTIMEMBER EFFECTS

Educational studies often factor in the effect of the instructor on student performance. Simple classification effects are used to add this instructor effect to a statistical model. However, the same student could have the same instructor multiple times over a period of several teaching sessions. Multimember effects handle this situation easily.

The North Carolina Science Olympiad tests elementary and middle school students on a variety of scientific subject matter in a weekend tournament. To prepare for the tournament, parent volunteers meet with students to coach them on the Olympiad events.

A local elementary school used three parent volunteers to work in four preparatory sessions with each child. Students were not assigned to a particular parent coach. Any child could work with any parent in each of the four sessions. A final quiz was given to each student and a standardized measure of each student's quiz score was recorded by the school's Olympiad coordinator. A partial list of the student data is shown in Table 5.

ID	HomeRoom	Coach1	Coach2	Coach3	Coach4	QuizScore
1	1	Mark	Bill	Bill	Mark	5.8734
2	1	Mark	Bill	Bill	Bill	10.1424
3	1	Mark	Bill	Mark	Angela	5.9590

Table 5. Student Scores for the Science Olympiad Preparation **(continued)**

ID	HomeRoom	Coach1	Coach2	Coach3	Coach4	QuizScore
4	1	Bill	Mark	Mark	Mark	3.2442
5	1	Angela	Mark	Bill	Angela	8.3787
6	2	Angela	Bill	Angela	Bill	9.3448
7	2	Bill	Bill	Angela	Mark	8.0779
8	2	Mark	Mark	Mark	Mark	0.6274
9	2	Mark	Bill	Mark	Mark	4.5242
10	2	Mark	Angela	Mark	Mark	4.5633

Table 5 (continued). Student Scores for the Science Olympiad Preparation

You can create a multimember effect for the parent coach to help assess the effectiveness of the coach in preparing the students for the Olympiad tournament. The multimember effect here counts the number of times each student saw each coach. Student 1 saw coach Mark and coach Bill twice each. Student 2 saw coach Mark one time and coach Bill three times. Note that not all students saw every coach.

The following EFFECT statement creates the multimember effect Coach that counts the number of times each student saw each coach:

```
effect Coach = mm(coach1 coach2 coach3 coach4);
```

New columns for the design matrix are constructed to represent these counts, as shown in Table 6.

Obs	Coach_Angela	Coach_Bill	Coach_Mark
1	0	2	2
2	0	3	1
3	1	1	2
4	0	1	3
5	2	1	1
6	2	2	0
7	1	2	1
8	0	0	4
9	0	1	3
10	1	0	3

Table 6. The Multimember Effect for Parent Coaches

The following code uses PROC GLIMMIX to fit the multimember effect to the response, showing a significant effect for the parent coach; the model also includes a random effect for the class within each student's school and specifies that the LS-means for the Coach effect be displayed.

```
proc glimmix data=test;
   class Coach1 Coach2 Coach3 Coach4 HomeRoom;
   effect Coach = mm(Coach1 Coach2 Coach3 Coach4);
   model QuizScore=Coach;
   random HomeRoom;
   lsmeans Coach / pdiff adjust=simulate;
run;
```

Table 7 shows that the overall effect of Coach is significant. The LS-means (Table 8) show that both Coach Angela and Coach Bill were more effective in preparing the students than Coach Mark while there is only a marginally significant difference between Coach Angela and Coach Bill (Table 9).

Type III Tests of Fixed Effects				
Effect	**Num DF**	**Den DF**	**F Value**	**Pr > F**
Coach	3	26	80.44	<.0001

Table 7. Type III Tests of the Multimember Effect

Coach Least Squares Means					
Coach	**Estimate**	**Standard Error**	**DF**	***t* Value**	**Pr > |*t*|**
Angela	9.1512	0.5515	26	16.59	<.0001
Bill	11.4339	0.5332	26	21.45	<.0001
Mark	1.0948	0.3930	26	2.79	0.0098

Table 8. The Least Squares Means for the Coach Effect

Differences of Coach Least Squares Means Adjustment for Multiple Comparisons: Simulated							
Coach	**_Coach**	**Estimate**	**Standard Error**	**DF**	***t* Value**	**Pr > |*t*|**	**Adj P**
Angela	**Bill**	-2.2826	0.9291	26	-2.46	0.0210	0.0549
Angela	**Mark**	8.0564	0.7353	26	10.96	<.0001	<.0001
Bill	**Mark**	10.3391	0.7750	26	13.34	<.0001	<.0001

Table 9. Differences in Least Squares Means Across Coaches

Note that the original definition of LS-means in Searle, Speed, and Milliken (1980) had nothing to do with multimember effects, but SAS/STAT procedures have extended that concept of population marginal means to this new application. In this case, these values estimate what the average quiz score would be had all students been coached by that person. Other familiar analytical constructs for SAS/STAT modeling procedures such as Type III tests and predicted values have also been made “EFFECT-aware” in this way.

CONCLUSION

In this paper, you have seen how the new EFFECT statement can add complexity to your model-fitting tool bag. Adding polynomials or spline effects for continuous variables to your models and adding lagged or multimember effects for your categorical variables increases the range of models that SAS/STAT software fits. The EFFECTPLOT statement gives you a quick visual assessment of how well your new effects work with your data and model.

However, the flexibility of the EFFECT statement does not end there. While the existing syntax gives you a great start, the COLLECTION *effect-type* enables you to design your own model effects. Perhaps you can even develop a new *effect-type* that can enhance the EFFECT statement's capabilities.

REFERENCES

Grizzle, J. E. 1965. "The Two-Period Change-Over Design and Its Use in Clinical Trials." *Biometrics* 21:461-480.

Searle, S. R., F. M. Speed, and G. A. Milliken. 1980. "Population Marginal Means in the Linear Model: An Alternative to Least Squares Means." *The American Statistician* 34 (4):216–221.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at:

Phil Gibbs
SAS Institute Inc.
SAS Campus Drive
Cary, NC 27513
E-mail: support@sas.com
Web: support.sas.com

Randy Tobias
SAS Institute Inc.
SAS Campus Drive
Cary, NC 27513
E-mail: support@sas.com
Web: www.sas.com

Kathleen Kiernan
SAS Institute Inc.
SAS Campus Drive
Cary, NC 27513
E-mail: support@sas.com
Web: support.sas.com

Jill Tao
SAS Institute Inc.
SAS Campus Drive
Cary, NC 27513
E-mail: support@sas.com
Web: support.sas.com