Paper 258-2010

Introducing PROC PLM and Postfitting Analysis for Very General Linear Models in SAS/STAT® 9.22

Randy Tobias, Weijie Cai, SAS Institute Inc., Cary NC

ABSTRACT

SAS/STAT software provides many powerful methods to fit general linear models with complicated effects. Some SAS/STAT procedures also enable you to estimate, test, compare, or plot functions of the estimated parameters. Such postprocessing can be as straightforward as predicting responses for given observations, or as complex as plotting multiplicity-adjusted comparisons of LS-means.

In the past, extensive postprocessing facilities have been limited to a few procedures, such as the GLM and MIXED procedures. This paper discusses how the PLM procedure, new in SAS/STAT 9.22, works in conjunction with a new STORE statement in many familiar SAS/STAT procedures to provide a full complement of postprocessing features for a wide spectrum of linear models. You can save the model fit information in any one of a dozen procedures and use it later in PROC PLM to perform new tests and analyses on the fit model.

The paper also discusses the new EFFECT statement, newly available in a number of procedures in SAS/STAT 9.22, which makes it easy for you to add many more types of effects to your general linear models—splines, polynomials, lags, and more.

INTRODUCTION

It was a huge project at the time. It took months for you to carefully plan and collect the data, and due to their sensitivity you only had a short window of access. Meeting after meeting went into determining the complex potential model chosen that was to explain the response, and the computer ground away for hours to fit it. That's all in the past and now you're presenting the results of the study to the decision maker. "That's great," she says, "but can you just show me a plot of how the system changes with respect to Cycle and Lift? And did you drill down into that Filter-by-Monitor interaction to figure out what was really going on?"

What do you do? The data are all locked down again, and even if you could get access to the data, it would take more time than you've got to refit the model and add these extra predictions and tests. But not to worry: you planned for this contingency by storing away the model that you fit. "No worries," you suavely say, and you deftly slip the model fit into the new PROC PLM. With a quick EFFECTPLOT statement and a SLICE statement, you've got the required information:

```
proc plm restore=sasuser.SystemModelFit;
   effectplot contourfit(x=Cycle y=Lift);
   slice Filter*Monitor;
run;
```

"Ah, I see," the boss lady says. "OK, I guess we're ready to sign off on this." PROC PLM saved the day!

The setting is overdramatized, but the situation is familiar— the need to squeeze extra statistical results out of a model you've already fit. The extra results could be as straightforward as predicting responses for given observations, or as complex as plotting multiplicity-adjusted comparisons of LS-means. Of course, many procedures in SAS/STAT software have always had nuts-and-bolts facilities for saving the fit. For example, in most modeling procedures you can save the parameter estimates and their covariance matrix into SAS data sets, and then use these in, say, SAS/IML® software to get the analyses you need. But the downsides to this approach are obvious. It's labor-intensive, it's error prone, and it requires you to juggle the original context of the model fit with the necessary calculations.

The PLM procedure, new in SAS/STAT 9.22, in conjunction with the STORE statement also new in many other SAS/STAT 9.22 procedures, eases all that. The PLM procedure performs postfitting statistical analyses for the contents of a SAS item store that was previously created in some other SAS/STAT procedure, and was stored using the STORE statement. The statements available in the PLM procedure are designed to reveal the contents of the source item store via the Output Delivery System (ODS) and to perform postfitting tasks such as the following:

- testing hypotheses
- computing confidence intervals
- producing prediction plots
- scoring a new data set

The new PLM procedure enables you to separate common postprocessing tasks, such as testing for treatment differences and predicting new observations under a fitted model, from the process of model building and fitting. You can apply a numerically expensive model fitting technique once to produce the source item store. The PLM procedure can then be called multiple times and the results of the fitted model can be analyzed without incurring the model fitting expenditure again.

The postprocessing techniques that PROC PLM offers are among the most advanced available in SAS/STAT software. New enhancements include step-down multiplicity adjustments for *p*-values, *F* tests with order restrictions, analysis of means (ANOM), and sampling-based linear inference based on Bayes posterior estimates. In addition, if the source procedure that performs the model fit includes the new EFFECT statement, then PROC PLM knows how to incorporate constructed effects in its results. This statement makes it easy for you to add many more types of effects to your general linear models—splines, polynomials, lags, and more.

PROC PLM doesn't replace the traditional close connection in SAS/STAT procedures between fitting a model and analyzing it. In fact, the statistical assumptions that underlie PROC PLM make it possible to transparently surface these same tasks directly in many SAS/STAT model procedures. Accordingly in SAS/STAT 9.22 there are over 30 instances of a procedure acquiring a full-featured postprocessing statement, using the same architecture as is used in PROC PLM. The section "ASSOCIATED DEVELOPMENT IN SAS/STAT PROCEDURES" on page 6 details these instances, and discusses how PROC PLM points the way to important elements of the next generation of SAS statistical tools.

LINEAR POSTPROCESSING IN VERY GENERAL LINEAR MODELS

PROC PLM has to do with linear estimation and inference within a very general class of linear models. This section gives a precise definition for these types of models and this type of inference.

Statistical modeling is generally concerned with the dependence of a variate of interest **y** on certain independent variates **x** according to a certain mathematical model. For example, **y** might be the yield of an industrial process and **x** the settings for particular ways of running the process; or **y** might be the time required for a new therapy to cure a patient and **x** the various physical characteristics of the patient. Many procedures in SAS/STAT software model this dependence by using a random distribution that depends linearly on certain parameters $\boldsymbol{\beta}$ of interest. This is written as

$$p(\mathbf{y}) = f(\mathbf{y}; \mathbf{x}'\boldsymbol{\beta}, \mathbf{z}, \phi) \tag{1}$$

where $p(\mathbf{y})$ is the random distribution for **y** and $f(\,)$ is some function. There might be other parameters ϕ and other independent variates **z** that affect the distribution of **y**, but the parameters $\boldsymbol{\beta}$ and hypotheses that concern them are of particular interest. The key point is that $p(\mathbf{y})$ depends on the parameters of interest $\boldsymbol{\beta}$ *linearly* through $\mathbf{x}'\boldsymbol{\beta}$.

For the purposes of this paper, models like (1) are called *very general linear models* or VGLMs. Familiar linear regression models, as fit by PROC REG, are members of the class of VGLMs; so are standard general linear models with classification, polynomial, and interaction effects (PROC GLM), mixed models (PROC MIXED), logistic linear models (PROC LOGISTIC), generalized linear models (PROC GENMOD), generalized linear mixed models (PROC GLIMMIX), and proportional hazards regression models (PROC PHREG).

All the methods implemented in PROC PLM apply to linear estimation and inference—that is, estimation and inference for linear combinations $\mathbf{L}\boldsymbol{\beta}$ of the linear parameters $\boldsymbol{\beta}$ in a VGLM, for some matrix **L**. For example, consider logistic regression for the probability that a customer will buy a product: if the parameters $\boldsymbol{\beta} = [\beta_M,\ \beta_F]$ correspond to males and females, respectively, then the individual components $\beta_M = [1\ 0]\boldsymbol{\beta}$ and $\beta_F = [0\ 1]\boldsymbol{\beta}$ and their difference $\beta_M - \beta_F = [1\ -1]\boldsymbol{\beta}$ are all linear combinations of the parameters. Other common linear inferences of interest concern predicted values ($\mathbf{x}'\boldsymbol{\beta}$), Type III tests, and population marginal means (LS-means).

Procedures that fit VGLMs have various computationally involved ways of fitting the model: PROC REG and PROC GLM use linear algebra, while most of the others mentioned previously employ some form of nonlinear optimization. But a key feature of all of these procedures is that once the fit has been determined, the estimate $\hat{\boldsymbol{\beta}}$ for $\boldsymbol{\beta}$ has an approximate multivariate normal distribution with a covariance $\hat{\sigma}^2\mathbf{G}$ that is known up to a chi-square-distributed constant. In mathematical terms,

$$\hat{\boldsymbol{\beta}} \sim N(\boldsymbol{\beta}, \hat{\sigma}^2\mathbf{G}),\ \hat{\sigma}^2 \sim \sigma^2\chi^2_{\mathrm{DFE}}/\mathrm{DFE} \tag{2}$$

where **G** is a known matrix. This is a big assumption and the fact that it does indeed hold true for so many different analyses represents the triumph of maximum likelihood theory and, ultimately, the central limit theorem. Under (2), linear inferences about $\boldsymbol{\beta}$ can be approximated using normal-theory methods. *F* tests, *t* tests, χ^2 tests—all the familiar tools of linear regression and analysis of variance apply. That's the kernel of what PROC PLM does: the procedure applies the familiar tools of linear regression and analysis of variance to the linear parameters of VGLMs.

FITTING, STORING, LOADING, AND ANALYZING A VGLM

The PLM procedure provides analysis of VGLMs. As such, the primary input to PROC PLM is not a data set, as is the case for most SAS statistical procedures, but rather a fitted VGLM model. You fit this model to data with another SAS/STAT procedure and save the results using the STORE statement, and then you provide the results to PROC PLM via the RESTORE= option.

The format in which the VGLM results are stored is called an item store. An item store is a special SAS binary file that is used to store and restore information that has a hierarchical structure. For example, ODS template definitions created by PROC TEMPLATE are stored in item stores. Item stores that are saved with the STORE statement in procedures that fit VGLMs can be processed with the PLM procedure. New in SAS/STAT 9.22, the STORE statement is available in 10 procedures: GENMOD, GLIMMIX, GLM, LOGISTIC, MIXED, ORTHOREG, PHREG, SURVEYLOGISTIC, SURVEYPHREG, and SURVEYREG. Note that while item store files are portable between machines of the same type, you cannot take an item store created on, say, a Unix machine and use it on a PC.

The following statements fit a general linear model to the standard data set sashelp.Class and save the resulting VGLM fit in a SAS *item store* named sasuser.StudyResults:

```
proc glm data=sashelp.Class;
   class Sex;
   model Weight = Height Sex|Age;
   store sasuser.StudyResults;
run;
```

Having stored the fitted model, you can subsequently use PROC PLM to restore and further analyze it. For example, suppose you want to reexamine the weights of boys and girls in the class, adjusting for height and age. An LSMEANS analysis of the Sex factor is the appropriate tool, and the following statements perform such an analysis on the model you fit previously. The output is shown in Figure 1.

```
proc plm restore=sasuser.StudyResults;
   lsmeans Sex / diff;
run;
```

Figure 1 Analyzing Weights of Boys and Girls, Adjusted for Age and Height

The PLM Procedure

Store Information

Item Store	SASUSER.STUDYRESULTS
Data Set Created From	SASHELP.CLASS
Created By	PROC GLM
Date Created	26FEB10:09:59:59
Response Variable	Weight
Class Variable	Sex
Model Effects	Intercept Height Sex Age Age*Sex

Class Level Information

Class	Levels	Values
Sex	2	F M

Sex Least Squares Means

Sex	Estimate	Standard Error	DF	t Value	Pr > |t|
F	95.4748	4.0787	14	23.41	<.0001
M	104.31	3.8482	14	27.10	<.0001

Differences of Sex Least Squares Means

Sex	_Sex	Estimate	Standard Error	DF	t Value	Pr > |t|
F	M	-8.8303	5.9687	14	-1.48	0.1612

With statements introduced in the following section, you can perform various statistical estimation and inference tasks from a saved item store, whenever the task is applicable within the context of the model. The model context means how the original data relate to the original model fit. Does the original model have classification effects? If so, then LS-means are available in PROC PLM. Did the original fit come from a mixed model procedure that used the Satterthwaite or Kenward-Roger degree-of-freedom method? If so, these methods continue to be available when the store contents are

processed with the PLM procedure.

Because the PLM procedure does not read data to fit a model, the processing time of this procedure is usually considerably less than the processing time of the procedure that originally fits the model and generates the item store.

PLM PROCEDURE SYNTAX

The syntax of the PLM procedure follows familiar conventions for SAS/STAT procedures:

```
proc plm restore=store;
   analysis-statement specification </options>;
run;
```

Here the RESTORE= option in the PROC PLM statement takes the place of the usual DATA= option that is used in most other procedures. It specifies the source item store with which you want to perform postprocessing.

PROC PLM provides 10 analysis statements for postprocessing and analysis. Generally they can be categorized into two groups:

- Statements for estimation, inference, and visualization based on fitted VGLMs:

EFFECTPLOT	produces a display of a complex fitted model.
ESTIMATE	estimates and tests custom linear functions of the form $\mathbf{L}\boldsymbol{\beta}$.
LSMEANS	computes and compares predicted population margins (LS-means) of fixed effects.
LSMESTIMATE	estimates and tests custom linear functions of LS-means.
SCORE	computes predicted values and other observationwise statistics for a SAS data set.
SLICE	performs a partitioned analysis of the LS-means for an interaction effect.
TEST	performs Type I, II, and III F tests for model effects.

- Statements for displaying and filtering analysis contents and results:

FILTER	filters the results of the PLM procedure.
SHOW	displays the contents of the source item store.
WHERE	selects a subset of BY groups from the source item store to which to apply the PROC PLM statements.

Many of these analysis statements are familiar from more established procedures, such as PROC GLM and PROC MIXED. As a rule, PROC PLM offers all of the usual features for these statements plus more. For example, the ESTIMATE statement in PROC PLM provides the combined features of the ESTIMATE and CONTRAST statements in PROC GLM, and also includes multiplicity adjustments for inferences on the estimates and joint tests under one-sided constraints (Silvapulle and Sen, 2004).

ADVANCED STATISTICAL METHODOLOGY IN PROC PLM

As mentioned previously, the postprocessing techniques that PROC PLM offers are among the most advanced available in SAS/STAT software. This section highlights some of these techniques. Also, the EFFECT statement is discussed, for defining new types of "constructed effects". The EFFECT statement is new in SAS/STAT 9.22 for many of the VGLM-fitting procedures that feed into PROC PLM.

Multiple Comparisons

Many of the postfitting analysis features of PROC PLM focus on comparing linear functions of the model parameters. You can compute and display standard differences between LS-means with the LSMEANS statement, dissect interaction effects via cell means with the SLICE statement, estimate and test custom linear combinations of the parameters with the ESTIMATE statement, and estimate and test custom linear combinations of the LS-means with the LSMESTIMATE statement. When you compute these comparisons, your aim to make some inference about the nature of the model; and whenever you've got multiple inferences to consider, you need to adjust for *multiplicity*.

The problem of multiple comparisons is that if many things all have, say, a 5% probability of happening, then the probability of *any one* of them happening is likely to be quite a bit higher than 5%. Multiple comparisons methods adjust

the individual *p*-values to control the probability of making even a single erroneous inference. There are many different methods for performing such adjustments, all the way from simple and general ones like Bonferroni adjustments to complex simulation-based step-down procedures. In the past, SAS/STAT tools for multiple comparisons have surfaced in such traditional workhorses as PROC GLM, PROC MIXED, and most recently PROC GLIMMIX, but with SAS/STAT 9.22 they are much more broadly available (see page 6), and you can also use them for postfit analysis in PROC PLM. See the section "Comparing Multiple B-Splines with PROC PLM" on page 14 for an example that uses PROC PLM to perform multiple comparisons of two nonparametric functions over a range of values.

Joint Tests with Constraints

Silvapulle and Sen (2004) propose a test statistic for inferences where the null hypothesis or the alternative hypothesis (or both) involve inequalities. You can test special cases of these hypotheses with the JOINT option in the ESTIMATE and the LSMESTIMATE statement. The statistic of Silvapulle and Sen (2004) takes into account the projection of the observed estimate onto the convex cone that is formed by the alternative parameter space. This test statistic is called the *chi-bar-square* statistic, and *p*-values are obtained by simulation; see, in particular, Chapter 3.4 in Silvapulle and Sen (2004).

Example 38.7 (*SAS/STAT User's Guide*) shows the JOINT option in action in the LSMESTIMATE statement of PROC GLIMMIX, to test isotonic contrasts for ordered alternatives. With PROC PLM, you can perform the same test for a broad variety of models—a model from an analysis of survival data (PROC PHREG), or of survey data (PROC SURVEYREG), or even of survey survival data (PROC SURVEYPHREG)!

Graphics

PROC PLM provides graphics, created with ODS Statistical Graphics functionality, in two modes. You can use the EFFECTPLOT statement, which is dedicated to visualization, to produce a display of the fitted model. You can also use the PLOTS= option for the four analysis statements (ESTIMATE, LSMEANS, LSMESTIMATE, and SLICE) to visualize linear functions $\mathbf{L}\boldsymbol{\beta}$, LS-means, linear functions of LS-means, and LS-mean differences. Both methods, in addition to the PLOTS option in the PROC PLM statement, provide you with flexibility to change and enhance the graphical displays.

Constructed Effects

VGLM-fitting procedures in SAS/STAT software have traditionally used the familiar and powerful syntax of classification and continuous variables and their interactions to specify the model for the linear parameters $\boldsymbol{\beta}$ in equation (1). In SAS/STAT 9.22 many procedures that fit VGLMs also support the experimental EFFECT statement, which greatly extends the ways you can build collections of columns $\mathbf{x}$ in linear models. These collections are referred to as constructed effects to distinguish them from the usual model effects that are formed from continuous or classification variables. You can supply the PLM procedure with an input item store that was created by using a model that incorporates the EFFECT statement: appropriate estimates and tests are available for such models in PROC PLM.

The following procedures support the EFFECT statement in SAS/STAT 9.22: GLIMMIX, GLMSELECT, HPMIXED, LOGISTIC, ORTHOREG, PHREG, PLS, QUANTREG, ROBUSTREG, SURVEYLOGISTIC, and SURVEYREG.

For example, in the following program, the EFFECT statement defines a constructed effect named SmoothX, and the MODEL statement uses it to form an extended analysis of covariance model:

```
proc glimmix;
   class A B Sub;
   effect SmoothX = spline(x);
   model y = A B A*SmoothX;
   random A*B / subject=Sub;
run;
```

In this model, the columns of SmoothX are formed from the data set variable x as a cubic B-spline basis with three equally spaced interior knots. The A*SmoothX term in the model fits a different spline smooth for each level of the CLASS variable A.

The following effect-types are available:

LAG	specifies the class level in the preceding period; this effect-type is useful for modeling carryover.
MULTIMEMBER	specifies classifications in which an observation can have more than one class level—for example, the effect of a teacher on student grades, since students can have multiple teachers.

POLYNOMIAL	defines standard polynomials. This effect-type is essentially a convenience: familiar VGLM syntax can also define polynomial models, but polynomial EFFECT statements do so much more succinctly.
SPLINE	defines regression splines.

There is also a COLLECTION effect-type that enables you to bind together arbitrary variables as a single effect with multiple degrees of freedom.

WORKING WITH MODEL FIT ITEM STORES

PROC PLM takes as input not a data set but an item store that contains a model fit. This makes it unique among SAS/STAT statistical procedures. This section points out and discusses particularly distinctive features of this item store processing.

- **Displaying item store contents:** Before you use the saved item store to make postprocessing inference, you might want to use the SHOW statement to display the contents of the item store. The SHOW statement enables you to display a range of information that is hierarchically saved in the item store. The default "Store Information" table displays information about the variables and model effects that were used in the original analysis. In addition, you can request other information such as fit statistics, parameter estimates and their covariance matrices, and the original program that was used to create the item store. This statement is useful for verifying that the contents of the item store apply to the analysis and for generating ODS tables.
- **Filtering item store contents:** You can also use the FILTER statement to select PROC PLM results that match certain criteria, making ODS tables and output data sets show just the results you're interested in. With a supplied *keyword* and *expression*, you can easily tailor the output to fit your needs. See the section "Comparing Multiple B-Splines with PROC PLM" on page 14 for an example that uses the FILTER statement to highlight comparisons of interest in a nonparametric analysis of covariance.
- **BY processing in the PLM procedure:** When you use a BY statement in the analysis that creates an item store, the information about BY variables and BY-group-specific modeling results is also transferred to the item store, and PROC PLM also automatically processes the item store in BY-group order. You can use the WHERE statement to restrict the analysis to specific BY groups that meet the conditions of the WHERE expression.
- **Analysis based on posterior estimates:** If you save an item store from a Bayesian analysis by PROC GENMOD or PROC PHREG, then PROC PLM can perform sampling-based inference based on Bayes posterior estimates that were saved in the item store. The majority of postprocessing tasks involve inference based on an estimable linear function, which often requires its mean and variance. When the traditional non-Bayesian analyses are performed, the mean and variance have explicit forms because the parameter estimate is analytically tractable. However, explicit forms are not usually available when Bayesian models are fitted. Instead, empirical means and variance-covariance matrices for the estimable function are constructed from the posterior sample.

ASSOCIATED DEVELOPMENT IN SAS/STAT PROCEDURES

The internal architecture that enables PROC PLM to consume a VGLM fit and perform further postfitting analysis also enables other procedures to do the same—to eat their own cooking, as it were. This makes it possible for the PLM procedure's statistical features to be incorporated directly into many other SAS/STAT procedures. A VGLM-fitting procedure can take its results, abstracted from the particular way that they were produced (see expression 2 of this paper), and then it can use the underlying architecture of PROC PLM to implement statistical features directly, with no need for an intervening item store.

Accordingly, in SAS/STAT 9.22 many familiar procedures have acquired entirely new statements that embody PROC PLM's functionality. For example, almost every procedure in SAS/STAT 9.22 that fits a VGLM now includes the LSMEANS statement, along with advanced methods for multiplicity-corrected comparisons. As another example, the new SURVEYPHREG procedure, for proportional hazards regression in the context of survey sample data, springs to life with a full complement of postprocessing (Mukhopadhyay, 2010). Overall, SAS/STAT 9.22 includes a total of 36 new procedure/statement combinations, as shown in Table 1.

Table 1 New SAS/STAT 9.22 Procedure/Statement Combinations That Use PROC PLM's Architecture

PROC	Statement					
GENMOD	EFFECTPLOT		LSMEANS	LSMESTIMATE	SLICE	
GLIMMIX					SLICE	
LOGISTIC	EFFECTPLOT	ESTIMATE	LSMEANS	LSMESTIMATE	SLICE	
MIXED				LSMESTIMATE	SLICE	
ORTHOREG	EFFECTPLOT	ESTIMATE	LSMEANS	LSMESTIMATE	SLICE	TEST
PHREG		ESTIMATE	LSMEANS	LSMESTIMATE	SLICE	
SURVEYLOGISTIC		ESTIMATE	LSMEANS	LSMESTIMATE	SLICE	
SURVEYPHREG		ESTIMATE	LSMEANS	LSMESTIMATE	SLICE	TEST
SURVEYREG		ESTIMATE	LSMEANS	LSMESTIMATE	SLICE	TEST

Note that the procedure/statement combinations depicted are just those which use the same underlying architecture as PROC PLM. There are gaps in the table, and the explanation for most of them is that they correspond to statements that were already available *before* SAS/STAT 9.22; for example, with PROC GLIMMIX this is the case for all statements except SLICE. Other gaps correspond to research and development work yet to be done, such as effect plots for survey analysis procedures.

When should you use a given analysis statement in the procedure directly and when should you use it in a subsequent PROC PLM call? It's a matter not only of what you want, but of *when* you know you want it. If you're fitting a logistic regression model with PROC LOGISTIC and you know you want to see an effect plot, go ahead and include the EFFECTPLOT statement in PROC LOGISTIC. But if you want to look at the regression analysis first before deciding on which kinds of plots to look at, you can store the fit in an item store and restore it in PROC PLM for as much further postfitting analysis as you like.

EXAMPLES USING PROC PLM

Since PROC PLM performs postfitting analysis, examples of its use require some discussion of how the fit was performed in the first place. Thus, each of the examples in this section spend some time describing the data analytic situation in order to perform the primary model fit, because it's in precisely such situations that it's most convenient to save the model fit and perform postfitting analysis separately. The following examples use PROC LOGISTIC for selecting a semiparametric model for predicting a binary response, PROC GENMOD for fitting a Bayesian logistic regression model, and PROC ORTHOREG for fitting an ANCOVA model by using splines.

Scoring with PROC PLM

Model selection is often used to extract useful information from a large amount of legacy data and to build interpretable models for classification problems with many variables. The selected model is then used to make predictions on new data. This example demonstrates how you can take a classification model that was selected using PROC LOGISTIC and use it later in PROC PLM to classify new observations.

You manufacture cookies, and for quality control you use 100 chemical and physical measurements to predict whether a given cookie would be classified as "Good" or "Bad" by human taste testers. Assume for now that this prediction model has been formulated and fit, based on a large library of past taste tests, and you've stored it in an item store called sasuser.CookieModel. A little later you'll see how to query the item store itself to see how the fitted model was produced.

In order to use the fitted model to score 10 new cookies whose chemical and physical measurements are contained in the file CookieMeasurements.dat, you simply need to read this data in and provide it to the SCORE statement in PROC PLM, like so:

```
data CookieMeasurements;
   infile 'CookieMeasurements.dat';
   input x1-x100 @@;
proc plm restore=sasuser.CookieModel;
   score data=CookieMeasurements out=CookieEval predicted / ilink;
run;
```

The ILINK option in the SCORE statement requests that predicted values be inversely transformed to the response scale. In this case, this is the predicted probability of a bad cookie. Figure 2 shows the predicted probability for the new observations.

Figure 2 Predicted Probability of a Bad Cookie

Obs	Predicted
1	1.00000
2	0.99997
3	0.00862
4	0.97723
5	0.00000
6	0.94821
7	1.00000
8	0.99959
9	0.00317
10	0.11710

Most of the probabilities are very high and a few are very low: you're pretty sure you can tell a good cookie from a bad one!

There's no data or model in sight, and you're already scoring! That's the point: with PROC PLM, you can separately deploy the item store that encapsulates the fitted model, with no need to access the original data. In practice, you use this fitted model for scoring new cookies until evidence accumulates that it has somehow become outdated, at which time you can again undertake the computationally intensive process of selecting a new model. You can rely on PROC PLM to remind you how to do this, using the SHOW statement to display the original program that produced the model fit item store, as demonstrated with the following statements and shown in Figure 3:

```
proc plm restore=sasuser.CookieModel;
   show program;
run;
```

Figure 3 Statements for Building Cookie Quality Model

```
                          The PLM Procedure

                          Store Information

Item Store                SASUSER.COOKIEMODEL
Data Set Created From     SASUSER.PASTTASTETESTS
Created By                PROC LOGISTIC
Date Created              20JAN10:12:13:04
Response Variable         Good
Link Function             Logit
Distribution              Binary
Class Variable            Good
Constructed Effect        splines
Model Effects             Intercept splines_x1 splines_x2
                          splines_x3 splines_x4 splines_x5
                          splines_x6 splines_x7 splines_x8 ..

                      SAS Program Information

   proc logistic data=sasuser.PastTasteTests;
      effect splines = spline(x1-x100/separate);
      model Good = splines/selection=stepwise;
      store sasuser.CookieModel;
   run;
```

The prediction model is based on your large library of past cookie taste tests (sasuser.PastTasteTests): each test consisting of 100 measurements, x1, . . . , x100, and the human taste test result, Good. The measurements are continuous variables and Good is a binary variable that indicates whether the taste tester liked the cookie. The statements shown in Figure 3 use the STEPWISE option in the LOGISTIC procedure to find a model by using the measurements to predict the human taste test results. The EFFECT statement requests that the effect of every predictor be modeled using smooth splines, and the SEPARATE option allows each variable's spline basis to enter or leave the model individually.

Since the spline effect produces seven columns for each predictor in the design matrix, stepwise regression with 100 such predictors is computationally intensive. For example, a typical Pentium 4 workstation takes around 10 minutes to run the preceding statements. You definitely don't want to rerun this model selection every time you have a new cookie to judge, and with PROC PLM, you don't have to.

One final remark: In the spirit of full disclosure, we've got to admit that the data for this example was simulated. We didn't really taste 5,000 cookies! The following statements simulate data for both the original and the new cookies, where the probability that a cookie is judged to be good depends on only the first 10 measurements, most according to nonlinear functions:

```
data sasuser.PastTasteTests;
   keep x1-x100 Good;
   array x{100};
   format x1-x100 5.3;
   do Cookie=1 to 5000;
      do j=1 to dim(x);
         x{j} = ranuni(1);
         end;
      linp =  10 + 11*x1 - 10*sqrt(x2) + 2/x3 - 8*exp(x4) + 7*x5*x5
             - 6*x6**1.5 + 5*log(x7) - 4*sin(3.14*x8) + 3*x9 - 2*x10;
      TrueProb = 1/(1+exp(-linp));

      if ranuni(1) < TrueProb then Good = 1;
      else                         Good = 0;
      output;
   end;
run;

data _null_;
   file 'CookieMeasurements.dat';
   array x{100};
   format x1-x100 5.3;
   do Cookie=1 to 10;
      do j=1 to dim(x);
         x{j} = ranuni(2);
         end;
      drop j;
      put x1-x100;
    end;
run;
```

Postfitting Analysis of Sensitive Data

You are a university administrator who studies faculty pay, and you collect information about the salaries of a small sample of faculty members who are randomly selected from different departments. For each of the faculty members, you obtain the annual salary (in thousand dollars), gender, and academic rank information. The following DATA step shows the data which are obtained from a simulated data set in Stockburger (2001):

```
data Salary;
   length Rank$ 9;
   input Salary Gender $ Rank $ @@;
   datalines;
38  Male     Full       58  Female   Associate
80  Female   Full       30  Female   Assistant
50  Female   Assistant  49  Female   Assistant
45  Male     Full       42  Female   Assistant
59  Male     Full       47  Female   Associate
34  Male     Assistant  53  Male     Associate
35  Female   Assistant  42  Male     Assistant
42  Male     Assistant  51  Male     Full
51  Female   Associate  40  Male     Assistant
48  Female   Associate  34  Female   Assistant
46  Female   Associate  45  Male     Assistant
50  Female   Assistant  61  Male     Full
62  Female   Full       51  Male     Assistant
59  Male     Full       65  Female   Associate
49  Male     Assistant  37  Female   Assistant
;
```

Your objective for this project is to investigate the dependency of salary on gender and rank with all other factors ignored. You use the following statements to perform linear regression on the data set and save the fitted results to an item store SalaryModel:

```
proc orthoreg data=Salary;
   class Gender Rank;
   model Salary=Gender|Rank;
   store SalaryModel;
run;
```

The analysis of variance from this fit (shown as repeated by PROC PLM in Figure 4) indicates that there is a highly significant Gender effect on Salary (uh oh!), but also that there might be an interaction between Rank and Gender. You finish the project by writing a report and sending it to the advisory board, and the sensitive Salary data set goes back under lock and key. No sooner have you done this than the advisory board asks you for additional analysis of the interaction between Gender and Rank. It seems a daunting task because you no longer have access to the original data. However, your foresight in having saved the analysis in an item store and PROC PLM save the day.

In the following statements, the PLM procedure reads the item store SalaryModel, the TEST statement performs Type III tests of fixed effects, and the SLICE statement performs inferences on the two-way interaction term Gender*Rank. The EFFECTPLOT statement requests a plot of predicted Salary values versus Gender and Rank with ODS Graphics enabled by the ODS GRAPHICS ON statement.

```
ods graphics on;
proc plm restore=SalaryModel;
   test;
   slice Gender*Rank;
   effectplot;
run;
ods graphics off;
```

The Type III test result (Figure 4) shows that the Rank effect is significant and the Gender effect is not significant, but the interaction effect Gender*Rank is significant. You wonder whether Gender*Rank plays different roles at different levels of these two classification variables.

Figure 4 Type III Tests Result

The PLM Procedure

Type III Tests of Model Effects

Effect	Num DF	Den DF	F Value	Pr > F
Gender	1	24	1.91	0.1792
Rank	2	24	13.52	0.0001
Gender*Rank	2	24	3.93	0.0333

Figure 5 shows the *F* test results for the interaction effect. For female faculty members, there is strong evidence that salary amounts differ at three academic rank levels. For male faculty members, the difference is not significant. For both assistant and associate professors, salary difference is not significant between males and females. For full professors, however, there is strong evidence that salary for male professors is different from salary for female professors.

Figure 5 Inference on Gender*Rank

F Test for Gender*Rank Least Squares Means Slice

Slice	Num DF	Den DF	F Value	Pr > F
Gender Female	2	24	12.45	0.0002

F Test for Gender*Rank Least Squares Means Slice

Slice	Num DF	Den DF	F Value	Pr > F
Gender Male	2	24	2.25	0.1268

F Test for Gender*Rank Least Squares Means Slice

Slice	Num DF	Den DF	F Value	Pr > F
Rank Assistant	1	24	0.34	0.5629

Figure 5 *continued*

F Test for Gender*Rank Least Squares Means Slice

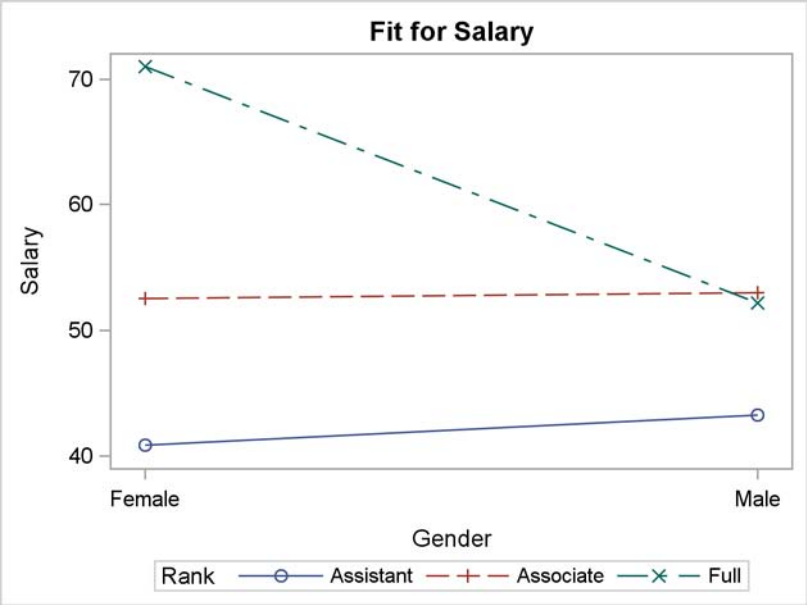
Slice	Num DF	Den DF	F Value	Pr > F
Rank Associate	1	24	0.00	0.9540

F Test for Gender*Rank Least Squares Means Slice

Slice	Num DF	Den DF	F Value	Pr > F
Rank Full	1	24	8.44	0.0078

The "InteractionPlot" produced by the EFFECTPLOT statement, displayed in Figure 6, reveals what's going on. Although there is little difference between the salaries of male and female assistant and associate professors, there is a strong difference for full professors: based on this sample, female full professors make over 35% more than their male counterparts. If this result seems unrealistic, recall that the original sample was small, and the data is simulated anyway!

Figure 6 Predicted Salary versus Gender and Rank



Bayesian Posterior Inference with PROC PLM

This example demonstrates how you can use PROC PLM to perform posterior inference from a Bayesian analysis. The data for this example are taken from Weisberg (1985) and concern the effect of small electrical currents on farm animals. The goal of the experiment is to understand the effects of high-voltage power lines on livestock, in order to better protect them.

The following DATA step lists the data from an experiment on the effects of small-to-moderate shocks on cattle. Current is the shock intensity level, Responses is the number of shock responses out of a total of Trials=35, and Experiment is 1 for the initial experiment and 2 for the follow-up experiment:

```
data Cows;
   input Current Responses Trials Experiment @@;
datalines;
0  0 35 1   0  0 35 2
1  6 35 1   1  3 35 2
2 13 35 1   2  8 35 2
3 26 35 1   3 21 35 2
4 33 35 1   4 27 35 2
5 34 35 1   5 29 35 2
;
```

You model the distribution of the shock response based on the level of current and the experiment number, and the analysis reflects your prior belief that the logit of Responses has a positive association with the shock level. You use the GENMOD procedure to fit a Bayesian logistic regression model that takes this prior information into account, as shown in the following statements:

```
data prior;
   _type_ = 'mean'; Current = 10; output;
   _type_ = 'var '; Current =  5; output;
proc genmod data=Cows;
   class Experiment;
   bayes coeffprior=normal(input=prior) seed=1;
   model Responses/Trials = Current|Experiment / dist=binomial;
   store sasuser.CowModel;
   title 'Bayesian Logistic Analysis of Cow Responses';
run;
```

The DATA step before the GENMOD procedure specifies your prior information about the Current effect, and the BAYES statement in the subsequent PROC GENMOD step uses it in the Bayesian analysis of a logistic ANCOVA model for the effects of Current, Experiment, and their interaction. See Chapter 37, "The GENMOD Procedure" (*SAS/STAT User's Guide*), for more information about this analysis.

The Bayesian analysis isn't very computationally intensive in this case, but you do have to be careful to examine the convergence diagnostics to ensure that the Markov chain simulation has converged. So once you've confirmed that, you use the STORE statement to save the fitted results in an item store named sasuser.CowModel, so that you can return to them again for further Bayesian results based on the same sample.

The analysis of the regression coefficients in this initial GENMOD fit indicate that, while the level of Current has an unambiguously positive effect on the number of responses across experiments, the Experiment effect itself and the interaction are less clear, because their Posterior Intervals contain zero. More investigation about whether the shock reaction was different between two experiments is warranted. Since you've saved the Bayesian analysis in the item store, you don't need to refit the model to perform this further posterior inference. The model fit item store contains the sampled posterior regression coefficients, as you can see using the SHOW statement in PROC PLM:

```
proc plm restore=sasuser.CowModel;
   show parms;
run;
```

A partial listing of the sample parameter estimates is displayed, as shown in Figure 7.

Figure 7 Bayesian Posterior Sample Parameter Estimates

Bayesian Logistic Analysis of Cow Responses

The PLM Procedure

Sample Parameter Estimates (First 20 of 10000 shown)

Parm1	Parm2	Parm3	Parm4	Parm5	Parm6
-4.9518	1.4762	1.3566	0	0.1498	0
-4.3709	1.3350	0.8377	0	0.2591	0
-3.6855	1.0719	0.8713	0	0.1425	0
-3.9691	1.2764	0.6867	0	0.3643	0
-4.5343	1.2963	0.8153	0	0.2937	0
-4.6303	1.4776	0.4115	0	0.2866	0
-4.4807	1.4326	0.6612	0	0.1564	0
-3.1451	1.0903	0.3130	0	0.2955	0
-3.3132	1.1271	0.005655	0	0.2898	0
-3.6783	1.2125	0.3454	0	0.2570	0
-3.4659	1.1874	0.2476	0	0.3117	0
-4.7383	1.5894	0.06146	0	0.2762	0
-3.1960	1.1440	-0.2248	0	0.4934	0
-3.6107	1.1750	-0.5488	0	0.4648	0
-3.5160	1.2373	-0.3322	0	0.2606	0
-3.1860	1.1460	-0.00279	0	0.3794	0
-3.2685	1.0569	-0.2477	0	0.3767	0
-3.0958	1.0385	-0.1845	0	0.3524	0
-3.0205	1.0341	-0.1591	0	0.3776	0
-3.2006	1.0724	-0.1825	0	0.3326	0

In order to determine whether the shock reaction for current level 0 is different between the two experiments, you use PROC PLM with the LSMEANS statement:

```
proc plm restore=sasuser.CowModel;
   lsmeans Experiment / diff exp cl at Current = 0;
run;
```

This analysis compares the predicted population means between the two groups at current level 0. The EXP option exponentiates the results to display an odds ratio as well, and finally, the CL option requests confidence limits. Figure 8 lists the resulting posterior sample estimate for the difference between experiments at this current level.

Figure 8 Comparison between Experiments at Current Level 0

Bayesian Logistic Analysis of Cow Responses

The PLM Procedure

Sample Differences of Experiment Least Squares Means

Experiment	_Experiment	Current	N	Estimate	Standard Deviation
1	2	0.00	10000	0.03659	0.7101

Sample Differences of Experiment Least Squares Means

		Percentiles			
Experiment	_Experiment	25th	50th	75th	Alpha
1	2	-0.4398	0.0467	0.5215	0.05

Sample Differences of Experiment Least Squares Means

Experiment	_Experiment	Lower HPD	Upper HPD	Exponentiated	Standard Error of Exponentiated
1	2	-1.3467	1.4080	1.3266	1.018218

Sample Differences of Experiment Least Squares Means

		Percentiles for Exponentiated		
Experiment	_Experiment	25th	50th	75th
1	2	0.6442	1.0478	1.6845

Sample Differences of Experiment Least Squares Means

Experiment	_Experiment	Lower HPD of Exponentiated	Upper HPD of Exponentiated
1	2	0.1095	3.3065

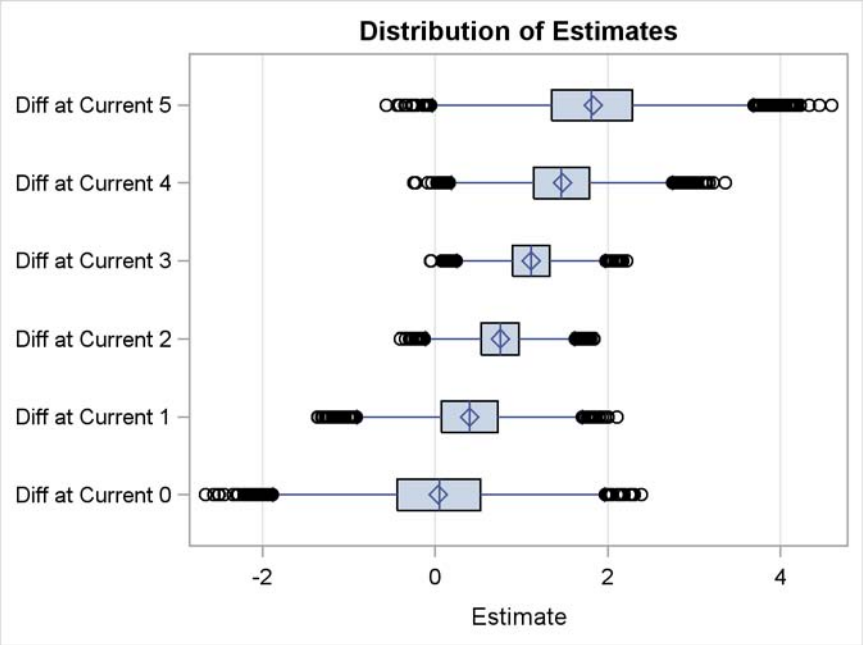
The sample statistics for the estimate are constructed from the posterior sample of regression coefficients that are saved in the item store sasuser.CowModel. The fact that the value of 1 is well within the limits for the odds ratio indicates little to no difference between the responses in the two successive experiments at the lowest level of current.

PROC PLM can also help you visualize the distribution of the experiment differences over the posterior sample of log odds ratios. The following statements use the ESTIMATE statement in PROC PLM to compute comparisons for all current levels, specifying the PLOTS= option to display the results graphically:

```
ods graphics on;
proc plm restore=sasuser.CowModel;
   estimate
   'Diff at Current 0' Experiment 1 -1 Current*Experiment [1, 0 1] [-1, 0 2],
   'Diff at Current 1' Experiment 1 -1 Current*Experiment [1, 1 1] [-1, 1 2],
   'Diff at Current 2' Experiment 1 -1 Current*Experiment [1, 2 1] [-1, 2 2],
   'Diff at Current 3' Experiment 1 -1 Current*Experiment [1, 3 1] [-1, 3 2],
   'Diff at Current 4' Experiment 1 -1 Current*Experiment [1, 4 1] [-1, 4 2],
   'Diff at Current 5' Experiment 1 -1 Current*Experiment [1, 5 1] [-1, 5 2]
    / plots=boxplot(orient=horizontal);
run;
ods graphics off;
```

Figure 9 shows little to no difference between the responses in the two successive experiments at low current levels (0, 1, and 2), but more evidence of a difference at higher current levels. Apparently, the higher the current level, the more likely cows are to get used to it. However, at the very highest current level, the lower HPD (highest posterior density) limit for the odds ratio (the exponentiated log odds difference) again dips below 1. Moreover, the distribution of posterior differences indicates that at this high level, while it is possible that animals will react very differently in the second experiment, it is also possible that they will not react differently at all. Maybe some cows just give up.

Figure 9 Box Plot of Difference between Two Experiments



A key point is that these successive PROC PLM runs don't involve a reanalysis of the original data. Because you stored the fitted model away in the first place, what you're really doing is a sequential posterior investigation of the original analysis.

Comparing Multiple B-Splines with PROC PLM

This example performs multiple comparisons among predicted values in a model with group-specific trends that are modeled through regression splines. The hypothetical setting for the example is botanical, but the data are simulated and indeed completely fictitious, so don't consider using them to learn how to garden!

You are studying the effect of different amounts of fertilizer on the size of two flower species. You try different amounts of fertilizer on each of 100 plants of different species and measure the size of their flowers. The following DATA step shows the (simulated) data:

```
data Flowers;
   input Species Size @@;
   Fertilizer = _n_;
datalines;
 1 -.020    1 0.199    2 -1.36    1 -.026    2 -.397    1 0.065    2 -.861    1 0.251
 1 0.253    2 -.460    2 0.195    2 -.108    1 0.379    1 0.971    1 0.712    2 0.811
 2 0.574    2 0.755    1 0.316    2 0.961    2 1.088    2 0.607    2 0.959    1 0.653
 1 0.629    2 1.237    2 0.734    2 0.299    2 1.002    2 1.201    1 1.520    1 1.105
 1 1.329    1 1.580    2 1.098    1 1.613    2 1.052    2 1.108    2 1.257    2 2.005
 2 1.726    2 1.179    2 1.338    1 1.707    2 2.105    2 1.828    2 1.368    1 2.252
 1 1.984    2 1.867    1 2.771    1 2.052    2 1.522    2 2.200    1 2.562    1 2.517
 1 2.769    1 2.534    2 1.969    1 2.460    1 2.873    1 2.678    1 3.135    2 1.705
 1 2.893    1 3.023    1 3.050    2 2.273    2 2.549    1 2.836    2 2.375    2 1.841
 1 3.727    1 3.806    1 3.269    1 3.533    1 2.948    2 1.954    2 2.326    2 2.017
 1 3.744    2 2.431    2 2.040    1 3.995    2 1.996    2 2.028    2 2.321    2 2.479
 2 2.337    1 4.516    2 2.326    2 2.144    2 2.474    2 2.221    1 4.867    2 2.453
 1 5.253    2 3.024    2 2.403    1 5.498
;
```

The following statements fit an ANCOVA model by using splines, with separate Size-by-Fertilizer trends for the two species; the trends are modeled as B-splines. The TEST statement (also new for PROC ORTHOREG in SAS/STAT 9.22) tests the effects in the model: an overall species difference, an overall fertilizer trend, and the interaction between the two.

```
proc orthoreg data=Flowers;
   class Species;
   effect SmoothF = spline(Fertilizer);
   model Size = Species|SmoothF;
   test Species|SmoothF;
   store FlowerModel;
   title 'Flower Species Fertilizer Effects';
run;
```

The TEST statement is new to PROC ORTHOREG in SAS/STAT 9.22; as discussed on page 6, it shares its underlying architecture with the same statement in PROC PLM. Results from the TEST statement are shown in Figure 10.

Figure 10 Tests for Effects in a Group-Specific Spline Model

Flower Species Fertilizer Effects

The ORTHOREG Procedure

Dependent Variable: Size

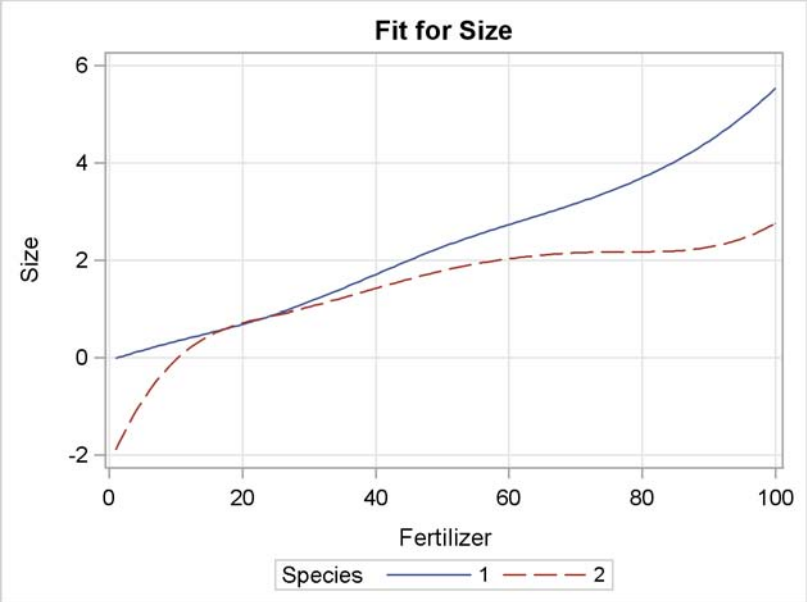
Type III Tests of Model Effects

Effect	Num DF	Den DF	F Value	Pr > F
Species	1	86	13.30	0.0005
SmoothF	6	86	290.42	<.0001
SmoothF*Species	6	86	30.47	<.0001

You examine the results from the PROC ORTHOREG fit to decide whether further postfit analysis is called for. All the model effects are significant, so you go ahead and use PROC PLM to visualize the model fit. All it takes is the EFFECTPLOT statement, with no other options: the fitted model saved in the FlowerModel item store provides enough information for PROC PLM to decide on the best way to plot the effects for this model.

```
ods graphics on;
proc plm restore=FlowerModel;
   effectplot;
run;
ods graphics off;
```

Figure 11 Predicted Values by Group



The prediction plot in Figure 11 depicts the model fit with different smooth curves, one for each species. It suggests that there is some difference between species flower sizes for small amounts of fertilizer and for values that exceed about 40.

Had you known that an effect plot would be of interest, you could have included an EFFECTPLOT statement in the original PROC ORTHOREG program that produced the fit. Using the FlowerModel item store with PROC PLM enables you to choose postfitting analyses interactively. The rest of this section demonstrates performing further postfitting analyses using the same fit model.

In order to determine the range in which the flower size trends separate significantly, you can use the PLM procedure with an ESTIMATE statement that applies species comparisons at a number of values for Fertilizer. In the following statements, a macro makes it easy to estimate differences over an entire range of fertilizer values:

```
%macro GroupDiff;
   %do x=0 %to 75 %by 5;
      "Diff at Fertilizer=&x" Species 1 -1 Species*SmoothF [1,1  &x] [-1,2  &x],
   %end;
   'Diff at Fertlizer=80' Species 1 -1 Species*SmoothF [1,1 80] [-1,2 80]
%mend;

proc plm restore=FlowerModel;
   estimate %GroupDiff / adjust=simulate seed=1 stepdown;
run;
```

The ESTIMATE statement uses nonpositional syntax for the Species*SmoothF interaction. For example, the specification $[-1,\ 2\quad 25]$ requests that the spline be computed for Species $= 2$ at Fertilizer $= 25$. The resulting coefficients are added to the bL vector for the estimate after being multiplied by -1.

Because comparisons are made at a large number of values for Fertlizer, a multiplicity correction is in order to adjust the *p*-values to reflect familywise error control. In this case, the ADJUST=SIMULATE and STEPDOWN options give step-down adjusted simulated *p*-values, one of the tightest multiple-comparison methods available (Westfall and Tobias, 2007).

Figure 12 displays the results from the ESTIMATE statement.

Figure 12 Estimate Results with Multiplicity Correction

Flower Species Fertilizer Effects

The PLM Procedure

Estimates
Adjustment for Multiplicity: Holm-Simulated

Label	Estimate	Standard Error	DF	t Value	Pr > |t|	Adj P
Diff at Fertilizer=0	12.4124	4.2130	86	2.95	0.0041	0.0206
Diff at Fertilizer=5	1.0376	0.1759	86	5.90	<.0001	<.0001
Diff at Fertilizer=10	0.3778	0.1540	86	2.45	0.0162	0.0545
Diff at Fertilizer=15	0.05822	0.1481	86	0.39	0.6952	0.9101
Diff at Fertilizer=20	-0.02602	0.1243	86	-0.21	0.8346	0.9565
Diff at Fertilizer=25	0.02014	0.1312	86	0.15	0.8783	0.9565
Diff at Fertilizer=30	0.1023	0.1378	86	0.74	0.4600	0.7418
Diff at Fertilizer=35	0.1924	0.1236	86	1.56	0.1231	0.2925
Diff at Fertilizer=40	0.2883	0.1114	86	2.59	0.0113	0.0450
Diff at Fertilizer=45	0.3877	0.1195	86	3.24	0.0017	0.0096
Diff at Fertilizer=50	0.4885	0.1308	86	3.74	0.0003	0.0024
Diff at Fertilizer=55	0.5903	0.1231	86	4.79	<.0001	<.0001
Diff at Fertilizer=60	0.7031	0.1125	86	6.25	<.0001	<.0001
Diff at Fertilizer=65	0.8401	0.1203	86	6.99	<.0001	<.0001
Diff at Fertilizer=70	1.0147	0.1348	86	7.52	<.0001	<.0001
Diff at Fertilizer=75	1.2400	0.1326	86	9.35	<.0001	<.0001
Diff at Fertlizer=80	1.5237	0.1281	86	11.89	<.0001	<.0001

Figure 12 shows that at the 5% significance level the trends are significantly different for Fertilizer ≤ 10 and for Fertilizer ≥ 40. Between those values you cannot reject the hypothesis of trend congruity.

To see this effect more clearly, you can filter the results by adding the FILTER statement to the previous PROC PLM run:

```
proc plm restore=FlowerModel;
   estimate %GroupDiff / adjust=simulate seed=1 stepdown;
   filter adjp > 0.05;
run;
```

This produces Figure 13, which displays the subset of the results in Figure 12 that meets the condition in the FILTER expression.

Figure 13 Filtered Estimate Results

Flower Species Fertilizer Effects

The PLM Procedure

Estimates
Adjustment for Multiplicity: Holm-Simulated

Label	Estimate	Standard Error	DF	t Value	Pr > |t|	Adj P
Diff at Fertilizer=10	0.3778	0.1540	86	2.45	0.0162	0.0545
Diff at Fertilizer=15	0.05822	0.1481	86	0.39	0.6952	0.9101
Diff at Fertilizer=20	-0.02602	0.1243	86	-0.21	0.8346	0.9565
Diff at Fertilizer=25	0.02014	0.1312	86	0.15	0.8783	0.9565
Diff at Fertilizer=30	0.1023	0.1378	86	0.74	0.4600	0.7418
Diff at Fertilizer=35	0.1924	0.1236	86	1.56	0.1231	0.2925

In this example, you used the new EFFECT statement in PROC ORTHOREG in conjunction with the plotting and advanced multiple-comparison methods in PROC PLM to zero in on the precise region of interest for a complicated nonparametric model.

DISCUSSION

In this paper you've seen how PROC PLM enables you to restore an entire complicated model fit from a very general linear model and perform postfit processing. You've also seen examples of some of the versatile new postfit statistical techniques that PROC PLM provides. PROC PLM provides this functionality with a development approach that is radically different from other SAS/STAT procedures, in two particular ways.

The first key development is the item store as a tool for storing and exchanging complex, articulated information between procedures. Item stores are used to communicate VGLM fits from various procedures to PROC PLM, which means that PROC PLM requires no input data set in order to perform a statistical analysis. Item stores are proving to be good vehicles for other types of complex, articulated information. For example, item stores are used in the SAS/STAT 9.22 spatial analysis procedures. After you use automatic nonlinear model selection in PROC VARIOGRAM to find the best model for a spatial autocorrelation function, you can use an item store to pass this model directly to PROC KRIGE2D or PROC SIM2D for further spatial prediction and inference. Kolovos (2010) provides an overview of the use of item stores in the SAS/STAT 9.22 spatial analysis procedures.

The second key development behind PROC PLM is the internal architecture that enables it to share so much functionality with so many other SAS/STAT procedures, as discussed on page 6. A great deal of research and development has gone into the particular characterization of VGLMs that is embodied in the PROC PLM item store, in order to make the model fit information complete for as broad a class of models as possible and yet also efficient for performing further analysis. A corresponding degree of research and development is behind the various PROC PLM methods that consume the item store information and produce such a wide variety of postfit analyses. Many of these methods appeared first in such SAS/STAT workhorses as GLIMMIX, GLM, and MIXED, and it was a critical step to deliver these methods in a more general context.

REFERENCES

Kenward, M. G. and Roger, J. H. (1997), "Small Sample Inference for Fixed Effects from Restricted Maximum Likelihood," *Biometrics*, 53, 983–997.

Kolovos, A. (2010), "Everything in Its Place: Efficient Geostatistical Analysis with SAS/STAT Spatial Procedures," *Proceedings of the SAS Global Forum 2010 Conference,* Cary, NC: SAS Institute Inc.

McCullagh, P. and Nelder, J. A. (1989), *Generalized Linear Models*, Second Edition, London: Chapman & Hall.

Mukhopadhyay, P. (2010), "Not Hazardous to Your Health: Proportional Hazards Modeling for Survey Data with the SURVEYPHREG Procedure," *Proceedings of the SAS Global Forum 2010 Conference,* Cary, NC: SAS Institute Inc.

SAS Institute Inc. (2010), *SAS/STAT 9.22 User's Guide*, Cary, NC: SAS Institute Inc.

Silvapulle, M. J. and Sen, P. K. (2004), *Constrained Statistical Inference: Order, Inequality, and Shape Constraints*, New York: John Wiley & Sons.

Stockburger, D. W. (2001), *Multivariate Statistics: Concepts, Models, and Applications*, WWW Version 2.0, http://www.psychstat.missouristate.edu/multibook2/mlt.htm.

Weisberg, S. (1985), *Applied Linear Regression*, Second Edition. New York: John Wiley & Sons.

Westfall, P. H., Tobias, R. D., Rom, D., Wolfinger, R. D., and Hochberg, Y. (1999), *Multiple Comparisons and Multiple Tests Using the SAS System*, Cary, NC: SAS Institute Inc.

Westfall, P. H. and Tobias, R. D. (2007), "Multiple Testing of General Contrasts: Truncated Closure and the Extended Shaffer-Royen Method," *Journal of the American Statistical Association*, 478, 487–494.

CONTACT INFORMATION

Randy Tobias,
SAS Institute Inc.
600 Research Drive
Cary, NC 27513
Work Phone: (919) 531-7933
E-mail: Randy.Tobias@sas.com
Web: http://www.sas.com

Weijie Cai,
SAS Institute Inc.
600 Research Drive
Cary, NC 27513
Work Phone: (919) 531-0359
E-mail: Weijie.Cai@sas.com
Web: http://www.sas.com