Paper 212-30

Parameterizing Models to Test the Hypotheses You Want: Coding Indicator Variables and Modified Continuous Variables

David J. Pasta, Ovation Research Group, San Francisco, CA

ABSTRACT

When statistical models include categorical explanatory variables, you can specify them on the CLASS statement (if the procedure supports the CLASS statement), or you can create a set of "dummy" or "indicator" variables. When you specify a categorical variable on the CLASS statement, there is a default parameterization that SAS® uses. But the default parameterization may not produce what you expect (for example, the default parameterization for PROC LOGISTIC is different from the default for PROC GLM). Beginning with Version 8, SAS provides more power and flexibility for categorical variables in the CLASS statement, specifically by allowing you to specify the method of parameterization and the reference category. The selected parameterization method has a profound effect on how CONTRAST statements are specified and on the interpretation of the parameter estimates and the associated hypothesis tests. An alternative to using the CLASS statement for categorical variables is to code your own indicator variables, which allows you even more control over the parameterization and associated hypotheses. This paper provides a series of examples to allow you to become comfortable using the CLASS statement to specify and test desired hypotheses (using the new syntax in PROC LOGISTIC for the examples) and illustrates the corresponding indicator variables. It also shows how the same principles can be used with modified continuous variables to construct piecewise linear models and test hypotheses about them. Much of the material presented here is taken from an earlier paper by Michelle Pritchard and David Pasta, "Head of the CLASS: Impress your colleagues with a superior understanding of the CLASS statement in PROC LOGISTIC" (Pritchard and Pasta, 2004).

INTRODUCTION

Generally students are not taught very much about parameterization or how to code indicator (dummy) variables. They are simply told, "this is how the model is parameterized," or are taught simple rules for creating indicator variables. Occasionally there is some discussion of some alternative parameterizations, but generally the topic is treated as something that's not quite proper to discuss in polite company. One book that covers this material in some detail, with an explicit example, is Hardy (1993).

Even the name usually given – dummy variables – makes it sound like something you would not necessarily be proud of. Perhaps the more mathematically-oriented term "indicator variables" makes it sound a little more respectable. Strictly speaking, an indicator variable takes on the value 1 when the condition is true and the value 0 when the condition is not true. In this context, we will use the term also for variables that may take on the value –1, which allows the variables to parameterize categorical variables in a different way than if only strict indicator variables were used.

Because SAS has added convenient syntax to the CLASS statement (for PROC LOGISTIC and coming soon to other procedures) for specifying alternative parameterizations, it is more important than ever to understand the choices of parameterizations and the implications of each. Understanding how the corresponding indicator variables are coded and how to interpret the results can be very useful in your modeling.

Different parameterizations arise very naturally in the context of categorical variables, but choices about parameterizations arise in the context of continuous variables as well. Adding or subtracting a constant from a continuous variable can change the hypotheses being tested. Creating a set of modified continuous variables from a single original variable can be very useful in testing specific hypotheses about the structure of a relationship between that variable and others. For example, using a set of modified continuous variables permits the construction of piecewise linear models, which are a natural generalization of the linear models most of you are familiar with. Piecewise linear models can be fit

using the usual linear modeling procedures – they are not "nonlinear models" in the sense that they require special algorithms (the models are still linear in the parameters to be estimated). I believe piecewise linear models are a better way to model deviations from linearity than the use of ordinary polynomials. Piecewise constant and piecewise linear models are often very useful for answering the sorts of questions researchers have about their data and are underutilized. They can be extended to the sophisticated cubic spline functions when smooth curves are wanted, but in my work I find that piecewise linear functions are more useful.

THE CLASS STATEMENT IN PROC LOGISTIC

Beginning in SAS Version 8, the CLASS statement in the LOGISTIC procedure enables programmers to specify a full-rank parameterization (with the choice of Effect, Reference, Polynomial, or Orthogonal Polynomial coding), or a less than full-rank parameterization (as in the GLM procedure). This is a big step forward from the days of doing your own coding of indicator variables, like you would for regression.

The GLM parameterization is only available as a global option (i.e., for all variables in the CLASS statement), but the full-rank parameterizations can be specified either globally or for individual variables. **Global parameterization** is specified by the PARAM= *<EFFECT GLM ORTHPOLY POLYNOMIAL REFERENCE>* option after the forward slash (/) in the CLASS statement, as follows:

```
proc logistic data = temp01;
  class <classvar1> <classvar2>/param = glm;
  model <response> = <classvar1> <classvar2>;
  title3 "Example 1: PARAM = GLM Global Option";
run;
quit;

proc logistic data = temp01;
  class <classvar1> <classvar2>/param = ref;
  model <response> = <classvar1> <classvar2>;
  title3 "Example 2: PARAM = REF Global Option";
run;
quit;
```

Alternatively, **parameterization for individual variables** can be specified in parentheses immediately following the variable in the CLASS statement, as follows:

```
proc logistic data = temp01;
  class <classvar1> (param = ref) <classvar2> (param = effect);
  model <response> = <classvar1> <classvar2>;
  title3 "Example 3: Individual Variable Parameterization Options";
run;
quit;
```

As an extra bit of flexibility, the REF= option in the CLASS statement determines the **reference level** for the EFFECT and REFERENCE coding. As with the PARAM= option, the REF= option can be coded globally or separately for each classification variable. To program the **reference level for each variable individually**, use the REF=" " option in parentheses immediately following the variable in the CLASS statement, or use the keyword FIRST or LAST (FIRST designates the first ordered category as the reference and LAST designates the last ordered category as the reference). Unless you use FIRST or LAST, the reference level should be placed in either single or double quotation marks, as follows:

```
proc logistic data = temp01;
  class <classvar1> (param = ref ref = "<refcat>") <classvar2>
      (param = effect ref = last);
  model <response> = <classvar1> <classvar2>;
  title3 "Example 4: Individual Variable Parameterization Options with
      Individual Specification of Reference Category";
run;
quit;
```

Alternatively, the **reference category can be applied to all variables** in the CLASS statement by using the keyword FIRST or LAST in the REF= option after the forward slash (/) in the CLASS statement:

```
proc logistic data = temp01;
  class <classvar1> (param = ref) <classvar2> (param = effect)/
      ref = first;
  model <response> = <classvar1> <classvar2>;
  title3 "Example 5: Individual Variable Parameterization Options with
      Global Specification of Reference Category";
run;
quit;
```

Noteworthy notes:

1. If you neglect to specify a coding method and/or reference category, then the **default parameterization** method is PARAM=EFFECT and the **default reference category** is the last ordered category (REF=LAST).
2. Individual parameterization trumps the global option, unless the global option is GLM.
3. REF= is only valid when PARAM=EFFECT or PARAM=REF.
4. If a format has been applied to the classification variables, then either the **formatted value** (or FIRST or LAST) must be used when specifying the reference category. If the keyword FIRST or LAST is used in this situation, then it pertains to the first or last ordered formatted category.
5. We recommend limiting the classification variables to no more than one parameterization method, unless there are specific multiple hypotheses that you want to test **and** you are confident in what you are doing **and** you won't be showing the output to anyone else **and** you are sure you'll remember later why you did what you did **and** … you get the idea.

DATA

Now that you have seen the new PARAM= and REF= syntax, let's consider a model with a dichotomous response and three classification variables. The variables involved are distributed as follows:

response	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	329	59.82	329	59.82
1	221	40.18	550	100.00

female	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0. Male	259	47.09	259	47.09
1. Female	291	52.91	550	100.00

smoker	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1. Never	261	47.45	261	47.45
2. Past	214	38.91	475	86.36
3. Current	75	13.64	550	100.00

agecat	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1. 18-34	210	38.18	210	38.18
2. 35-49	167	30.36	377	68.55
3. 50-64	117	21.27	494	89.82
4. 65+	56	10.18	550	100.00

female	smoker	agecat	response	Frequency	Percent
0	1	1	0	34	6.18
0	1	1	1	21	3.82
0	1	2	0	16	2.91
0	1	2	1	10	1.82
0	1	3	0	15	2.73
0	1	3	1	9	1.64
0	1	4	0	7	1.27
0	1	4	1	3	0.55
0	2	1	0	26	4.73
0	2	1	1	19	3.45
0	2	2	0	27	4.91
0	2	2	1	17	3.09
0	2	3	0	10	1.82
0	2	3	1	5	0.91
0	2	4	0	1	0.18
0	2	4	1	1	0.18
0	3	1	0	3	0.55
0	3	1	1	2	0.36
0	3	2	0	6	1.09
0	3	2	1	6	1.09
0	3	3	0	6	1.09
0	3	3	1	6	1.09
0	3	4	0	5	0.91
0	3	4	1	4	0.73
1	1	1	0	28	5.09
1	1	1	1	18	3.27
1	1	2	0	22	4.00
1	1	2	1	20	3.64
1	1	3	0	25	4.55
1	1	3	1	17	3.09
1	1	4	0	8	1.45
1	1	4	1	8	1.45
1	2	1	0	26	4.73
1	2	1	1	20	3.64
1	2	2	0	25	4.55
1	2	2	1	12	2.18
1	2	3	0	15	2.73
1	2	3	1	7	1.27
1	2	4	0	2	0.36
1	2	4	1	1	0.18
1	3	1	0	7	1.27
1	3	1	1	6	1.09
1	3	2	0	5	0.91
1	3	2	1	1	0.18
1	3	3	0	1	0.18
1	3	3	1	1	0.18
1	3	4	0	9	1.64
1	3	4	1	7	1.27

MODEL A: DEFAULT PARAMETERIZATION (EFFECT)

```
proc logistic data = temp01  descending;
  class female smoker agecat;
  model response = female smoker agecat;
  title3 "Model A: Logistic regression with three categorical predictors
      and default options PARAM=EFFECT and REF=LAST";
run;
quit;
```

In Model A, the method of parameterization is not specified, so the default EFFECT parameterization will be used. (Also, by default the last ordered category is used as the reference category.) The columns of the design matrix are created based on the EFFECT coding scheme, as follows: For each variable, *c*–1 columns comprise the design matrix, where *c* is the number of levels of the classification variable. For the non-reference levels, the columns represent group membership (1=member, 0 =non-member), and for the reference level, the row is populated with a–1. In our Model A, the reference level is the last ordered category, so the design matrix is:

Class Level Information

		Design Variables		
Class	Value	1	2	3
female	0	1		
	1	-1		
smoker	1	1	0	
	2	0	1	
	3	-1	-1	
agecat	1	1	0	0
	2	0	1	0
	3	0	0	1
	4	-1	-1	-1

Using EFFECT coding, the beta estimates are estimating the difference in the effect of each non-reference level compared to the average effect over all levels.

Analysis of Maximum Likelihood Estimates

Parameter		DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept		1	-0.3705	0.1057	12.2880	0.0005
female	0	1	-0.0170	0.0877	0.0374	0.8466
smoker	1	1	-0.0120	0.1253	0.0091	0.9240
smoker	2	1	-0.1098	0.1353	0.6578	0.4174
agecat	1	1	0.0470	0.1431	0.1079	0.7425
agecat	2	1	-0.0108	0.1524	0.0051	0.9433
agecat	3	1	-0.0753	0.1677	0.2017	0.6534

Therefore, the odds of having response = 1 for never vs. current smokers is: exp(-0.0120)/exp[-(-0.012 – 0.1098)] = 0.875.

Odds Ratio Estimates

Effect	Point Estimate	95% Wald Confidence Limits	
female 0 vs 1	0.967	0.685	1.363
smoker 1 vs 3	0.875	0.511	1.499
smoker 2 vs 3	0.793	0.451	1.397
agecat 1 vs 4	1.008	0.536	1.897
agecat 2 vs 4	0.951	0.498	1.818
agecat 3 vs 4	0.892	0.456	1.745

MODEL B: GLM PARAMETERIZATION

```
proc logistic data = temp01  descending;
  class female smoker agecat/param = glm;
  model response = female smoker agecat;
  title3 "Model B: Logistic regression with three categorical predictors
       and PARAM = GLM option";
run;
quit;
```

In Model B, the GLM parameterization has been specified. With GLM parameterization, the programmer does not have control over the reference category using the REF= option; instead, SAS includes variables as long as they are linearly independent from those that went before. This has the effect of making the last ordered category the omitted (reference) category. The columns of the design matrix are created based on the GLM coding scheme, as follows: For each variable, *c* columns comprise the design matrix, where *c* is the number of levels of the classification variable. For all levels, the columns represent group membership (1=member, 0 =non-member). The design matrix for Model B is:

Class Level Information

		Design Variables			
Class	Value	1	2	3	4
female	0	1	0		
	1	0	1		
smoker	1	1	0	0	
	2	0	1	0	
	3	0	0	1	
agecat	1	1	0	0	0
	2	0	1	0	0
	3	0	0	1	0
	4	0	0	0	1

Using GLM coding, the beta estimates are estimating the difference in the effect of each non-reference level compared to the reference (last) level.

Analysis of Maximum Likelihood Estimates

Parameter		DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
intercept		1	-0.1927	0.3163	0.3712	0.5423
female	0	1	-0.0339	0.1754	0.0374	0.8466
female	1	0	0	.	.	.
smoker	1	1	-0.1337	0.2747	0.2368	0.6266
smoker	2	1	-0.2315	0.2887	0.6431	0.4226
smoker	3	0	0	.	.	.
agecat	1	1	0.00790	0.3226	0.0006	0.9805
agecat	2	1	-0.0500	0.3304	0.0229	0.8798
agecat	3	1	-0.1144	0.3424	0.1117	0.7382
agecat	4	0	0	.	.	.

Therefore, the odds of having response = 1 for never vs. current smokers is: exp(-0.1337) = 0.875.

Odds Ratio Estimates

Effect	Point Estimate	95% Wald Confidence Limits	
female 0 vs 1	0.967	0.685	1.363
smoker 1 vs 3	0.875	0.511	1.499
smoker 2 vs 3	0.793	0.451	1.397
agecat 1 vs 4	1.008	0.536	1.897
agecat 2 vs 4	0.951	0.498	1.818
agecat 3 vs 4	0.892	0.456	1.745

One of the reasons this parameterization may be desirable is that it corresponds to the behavior of GLM and MIXED and is therefore familiar to long-time users of those procedures. One disadvantage is that in order to control which category is omitted (treated as the reference category), it is necessary to order the categories so the category to be omitted is last.

MODEL C: REFERENCE PARAMETERIZATION

The REFERENCE parameterization is similar to GLM except you actually omit one of the categories (so the resulting design matrix is full rank), and you have explicit control over which category is to be omitted — it need not be the last category.

```
proc logistic data = temp01  descending;
  class female smoker agecat/param = ref;
  model response = female smoker agecat;
  title3 "Model C: Logistic regression with three categorical predictors
      and PARAM = REF option";
run;
quit;
```

In Model C, the REFERENCE parameterization has been specified. (Also, by default the last ordered category is used as the reference category.) The columns of the design matrix are created based on the REFERENCE coding scheme, as follows: For each variable, *c*–1 columns comprise the design matrix, where *c* is the number of levels of the classification variable. For the non-reference levels, the columns represent group membership (1=member, 0 =non-member), and for the reference level, the row is populated with a 0. In our Model C, the reference level is the last ordered category, so the design matrix is:

Class Level Information

Class	Value	Design Variables		
female	0	1		
	1	0		
smoker	1	1	0	
	2	0	1	
	3	0	0	
agecat	1	1	0	0
	2	0	1	0
	3	0	0	1
	4	0	0	0

Using REFERENCE coding, the beta estimates are estimating the difference in the effect of each non-reference level compared to the effect of the reference level.

Analysis of Maximum Likelihood Estimates

Parameter		DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept		1	-0.1927	0.3163	0.3712	0.5423
female	0	1	-0.0339	0.1754	0.0374	0.8466
smoker	1	1	-0.1337	0.2747	0.2368	0.6266
smoker	2	1	-0.2315	0.2887	0.6431	0.4226
agecat	1	1	0.00790	0.3226	0.0006	0.9805
agecat	2	1	-0.0500	0.3304	0.0229	0.8798
agecat	3	1	-0.1144	0.3424	0.1117	0.7382

Therefore, as with the GLM parameterization, the odds of having response = 1 for never vs. current smokers is: exp(-0.1337) = 0.875.

Odds Ratio Estimates

Effect	Point Estimate	95% Wald Confidence Limits	
female 0 vs 1	0.967	0.685	1.363
smoker 1 vs 3	0.875	0.511	1.499
smoker 2 vs 3	0.793	0.451	1.397
agecat 1 vs 4	1.008	0.536	1.897
agecat 2 vs 4	0.951	0.498	1.818
agecat 3 vs 4	0.892	0.456	1.745

MODEL D: REFERENCE PARAMETERIZATION WITH REFERENCE CATEGORY SPECIFICATION

In this model we use the reference parameterization but specify REF=FIRST. This changes the design matrix and the odds ratios are inverted for binary variables.

```
proc logistic data = temp01  descending;
  class female smoker agecat/ param = ref ref = first ;
  model response = female smoker agecat;
  title3 "Model D: Logistic regression with three categorical predictors
      and PARAM=REF and REF=first";
run;
quit;
```

Class Level Information

Class	Value	Design Variables 1	2	3
female	0	0		
	1	1		
smoker	1	0	0	
	2	1	0	
	3	0	1	
agecat	1	0	0	0
	2	1	0	0
	3	0	1	0
	4	0	0	1

Analysis of Maximum Likelihood Estimates

Parameter		DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept		1	-0.3524	0.1915	3.3869	0.0657
female	1	1	0.0339	0.1754	0.0374	0.8466
smoker	2	1	-0.0978	0.1920	0.2597	0.6103
smoker	3	1	0.1337	0.2747	0.2368	0.6266
agecat	2	1	-0.0579	0.2121	0.0745	0.7850
agecat	3	1	-0.1223	0.2376	0.2650	0.6067
agecat	4	1	-0.00790	0.3226	0.0006	0.9805

Therefore, the odds of having response = 1 for current vs. never smokers is: exp(0.1337) = 1.143.

Odds Ratio Estimates

Effect	Point Estimate	95% Wald Confidence Limits	
female 1 vs 0	1.035	0.734	1.459
smoker 2 vs 1	0.907	0.622	1.321
smoker 3 vs 1	1.143	0.667	1.958
agecat 2 vs 1	0.944	0.623	1.430
agecat 3 vs 1	0.885	0.555	1.410
agecat 4 vs 1	0.992	0.527	1.867

MODEL E: DEFAULT WITH INTERACTIONS

One thing that analysts new to the CLASS statement find mysterious is that when they specify interactions, sometimes SAS changes the order of the variables. They specified FEMALE*SMOKER, for example, but in the output they see SMOKER*FEMALE. What is SAS doing? It turns out SAS is paying attention to the order of the variables in the CLASS statement.

```
proc logistic data = temp01  descending;
  class female smoker agecat;
  model response = female smoker agecat female*smoker female*agecat
      smoker*agecat;
  title3 "Model E1: Logistic regression with three categorical
      predictors plus two-way interactions, default options";
run;
quit;

proc logistic data = temp01  descending;
  class smoker agecat female;
  model response = female smoker agecat female*smoker female*agecat
      smoker*agecat;
  title3 "Model E2: Logistic regression with three categorical
      predictors plus two-way interactions, default options";
run;
quit;
```

MODEL F: PARAM=GLM WITH INTERACTIONS

```
proc logistic data = temp01  descending;
  class female smoker agecat/param = glm;
  model response = female smoker agecat female*smoker female*agecat
      smoker*agecat;
  title3 "Model F: Logistic regression with three categorical predictors
      plus two-way interactions, with PARAM = GLM option";
run;
```

HYPOTHESIS TESTING IN PROC LOGISTIC WITH CONTRAST STATEMENTS

The CONTRAST statement in PROC LOGISTIC works much like the corresponding statement in other procedures such as GLM and MIXED. It allows you to specify one or more linear combinations of the parameters to test against zero. You can test each of the linear combinations against zero or specify that several linear combinations be tested against zero simultaneously (a multiple-degrees-of-freedom test). In order to correctly specify the CONTRAST statement, you need to pay close attention to the parameterization. For example, consider the AGECAT variable as parameterized in Model A, the default EFFECT coding. To test group 1 against group 2, the contrast statement would be:

```
contrast '1 vs. 2' agecat 1 –1 0 / e;
```

The 0 at the end is not necessary but is a good reminder that there are three parameters for the AGECAT variable. The E option is extremely useful: it requests that the linear combination be displayed, giving you a chance to make sure the coefficients are lined up with the proper parameters. I strongly recommend using the E option when testing new CONTRAST statements.

Testing that groups 1, 2, and 3 are simultaneously equal requires a two-degree-of-freedom test:

```
contrast '1 = 2 = 3' agecat 1 –1 0 , agecat 1 0 -1 / e;
```

or, equivalently,

```
contrast '1 = 2 = 3' agecat 1 –1 0 , agecat 0 1 -1 / e;
```

or

```
contrast '1 = 2 = 3' agecat 1 0 –1 , agecat 0 1 -1 / e;
```

all of which test the same hypothesis, that the first three AGECAT groups are equal.

Because the fourth AGECAT group is coded as the negation of the first three, contrasts involving that group look rather different. If you label the parameters for the first three groups A1, A2, and A3, then the parameter for the fourth group is -A1-A2-A3. Thus, if you want to compare group 1 to group 4, you want to test (A1)–(-A1-A2-A3)=0, or 2*A1+A2+A3=0:

```
contrast '1 vs. 4' agecat 2 1 1 / e;
```

It is clear how useful it is to label each contrast — it would be easy to forget what this contrast is designed to test!

The construction of CONTRAST statements can be very tricky in the presence of many interactions. To be sure you are testing the hypothesis you want to test, use the E option and write down the resulting equation separately. It may be worth seeking advice from a statistician for especially complex models.

INTERACTIONS OF CATEGORICAL AND CONTINUOUS VARIABLES

Another time where parameterization is important is when there is a mix of categorical and continuous variables and you are interested in interactions between them. The continuous variable could be

anything, but often it is time or something that varies systematically over time (such as age). For convenience, we will assume the variable TIME has been added to our dataset. If TIME is added to the model along with FEMALE, SMOKER, and AGECAT (plus interactions among those categorical variables), this implicitly assumes that the TIME effect is the same for all individuals, regardless of their values for the categorical variables. That is, the model fits parallel lines, with the slope of the lines given by the parameter estimate for TIME, and the distance between the lines corresponding to the estimated difference between the corresponding subgroups (male smokers, say, compared to female nonsmokers).

In order to have different slopes for different groups, the categorical variables need to be interacted with TIME. Consider the model that includes FEMALE, TIME, and FEMALE*TIME. How are the coefficients of these three terms interpreted? It turns out the coefficient of TIME is the slope of the line for males, and the coefficient for FEMALE*TIME is the difference between the slope for males and slope for females. Thus, the latter coefficient tests the hypothesis that there is no difference in slope between males and females. There is no convenient test of the overall question of whether both slopes are zero (that test would be a combined test of TIME and FEMALE*TIME). If you included a model that had just FEMALE and FEMALE*TIME, then the test for FEMALE*TIME (which would have two degrees of freedom) would be a test that both slopes were zero.

But how do you interpret the coefficient of FEMALE in the model with FEMALE, TIME, and FEMALE*TIME? When you just have FEMALE and TIME, the coefficient of FEMALE represents the difference between the (parallel) male and female lines, but when we add the FEMALE*TIME interaction, the lines are no longer parallel. The coefficient of FEMALE still represents the difference between the two lines, but now we need to specify at what time. The answer is: at TIME=0. Thus the FEMALE variable tests whether the intercepts of the two lines are the same.

Sometimes this test is very instructive; other times, it is meaningless. Consider when TIME is measured in years. We are then testing whether the lines have the same intercept back in the Year 0. (OK, the year 1 BCE if you want to get technical.) Not necessarily riveting. But if we measure TIME as the year minus 2002, say, then we are measuring whether the two lines have approximately the same value in 2002, potentially a much more interesting hypothesis to test.

More complicated interactions produce similarly complicated interpretation. Consider a model with FEMALE, AGECAT, TIME, FEMALE*AGECAT, FEMALE*TIME, AGECAT*TIME. We allow the eight gender-by-age categories to have any intercept they want, but we are imposing some structure on the slopes (because we omitted the FEMALE*AGECAT*TIME term). There is hardly anything to do about understanding the model except to draw some graphs. Look at Silva (2003) for guidance on making this process less painful. But what are the hypotheses being tested? You need to think hard about what each of the terms is contributing and about what the model looks like if that term is omitted in order to figure it out. And remember that the categorical variables and their interactions are looking at the lines at the intercept, i.e., when TIME=0.

PIECEWISE CONSTANT AND PIECEWISE LINEAR MODELS

Extensions to the linear model that are very useful to consider include piecewise constant and piecewise linear models. These extensions are easily implemented by coding what are called basis functions (special variables that capture a piece of the final functional form), and then fitting a linear model using those new variables. The book by Hastie, Tibshirani, and Friedman (2001) provides a reasonably clear although somewhat mathematical treatment of this in Chapter 5, "Basis Expansions and Regularization." The basic idea is as follows. Define some points on the X axis (we will continue to assume that variable is TIME) at which the fitted function is allowed to change. These are called "knots" but you can think of them simply as boundaries between pieces of the piecewise function. Define variables that have nonzero values only between pairs of knots or which change slopes (for example) at the knots. These variables are then put together into a fitted function made up of the pieces.

PIECEWISE CONSTANT MODELS

Consider a model where TIME is measured in months since surgery. We might be interested in looking at six-month time intervals going out two years. So we would have data from TIME=0 to TIME=24, with knots at TIME=6, 12, and 18. If we wanted to fit a piecewise constant function, we could calculate new variables as follows:

```
tcon0006=(0 le time le 6);
tcon0612=(6 lt time le 12);
tcon1218=(12 lt time le 18);
tcon1824=(18 lt time le 24);
```

These indicator variables would take on the value 1 when the expression was true (i.e., TIME was within the bounds given) and 0 otherwise. Coding I would be more comfortable with would go along the following lines:

```
if (time lt 0) then error 'error: time lt 0';
else if (time le 6) then per=1;
else if (time le 12) then per=2;
else if (time le 18) then per=3;
else if (time le 24) then per=4;
else error 'error: time gt 24';
tcon0006=(per eq 1);
tcon0612=(per eq 2);
tcon1218=(per eq 3);
tcon1824=(per eq 4);
```

This code has the advantage of creating a variable PER (for time period) that is likely to be useful. It also creates the variable with checks that TIME is never outside the range 0 to 24 and makes it easy to ensure consistent coding of the indicator variables.

By using these four variables in a model, a piecewise constant model would be fit. Each coefficient tests whether that particular time period has a zero value (see Figure 1). It may be more useful to fit an overall mean and include deviations from that mean so that you are testing deviations from the global mean rather than deviations from zero.

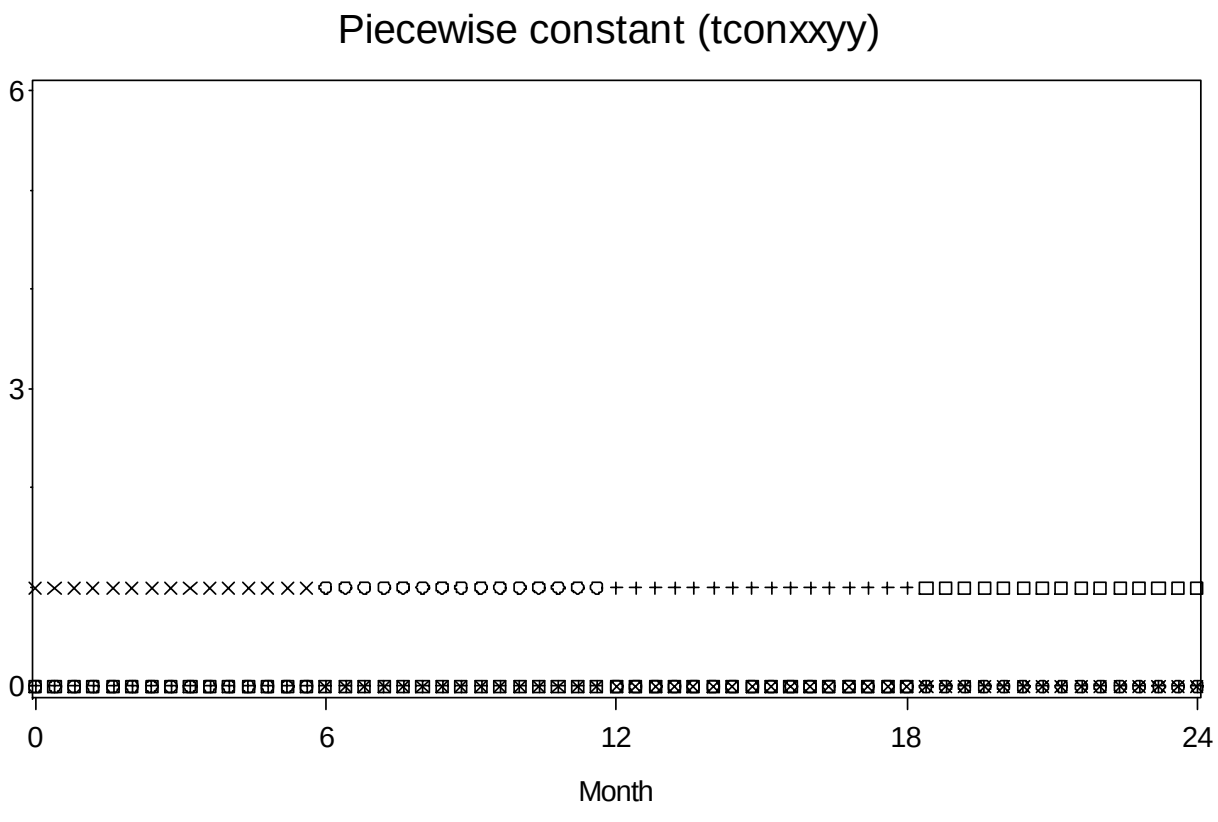

FIGURE 1

PIECEWISE LINEAR 1: CHANGE IN SLOPE

A similar approach can be used to fit piecewise linear functions, except now the variable is not a constant. There are two main ways to parameterize piecewise linear functions. The first way is designed to test for a change in slope. The variables are defined to be nonzero over its assigned piece and all later pieces. Consider the variables defined by:

```
time00on=max(time-00,0);
time06on=max(time-06,0);
time12on=max(time-12,0);
time18on=max(time-18,0);
```

The variable TIME00ON is exactly the same as TIME if TIME never takes on negative values (as we are assuming here). The other variables are zero up to the time indicated by their name, and then rise linearly thereafter (see Figure 2). Why is this form convenient? Consider a model with TIME00ON, TIME06ON, and TIME12ON. The coefficient of TIME06ON is the difference in the slope from 00 to 06 compared to the slope from 06 to 12. The test of that coefficient is a test of whether the slope changes at time 06. The test of the coefficient of TIME12ON is the test that the slope changes at time 12.

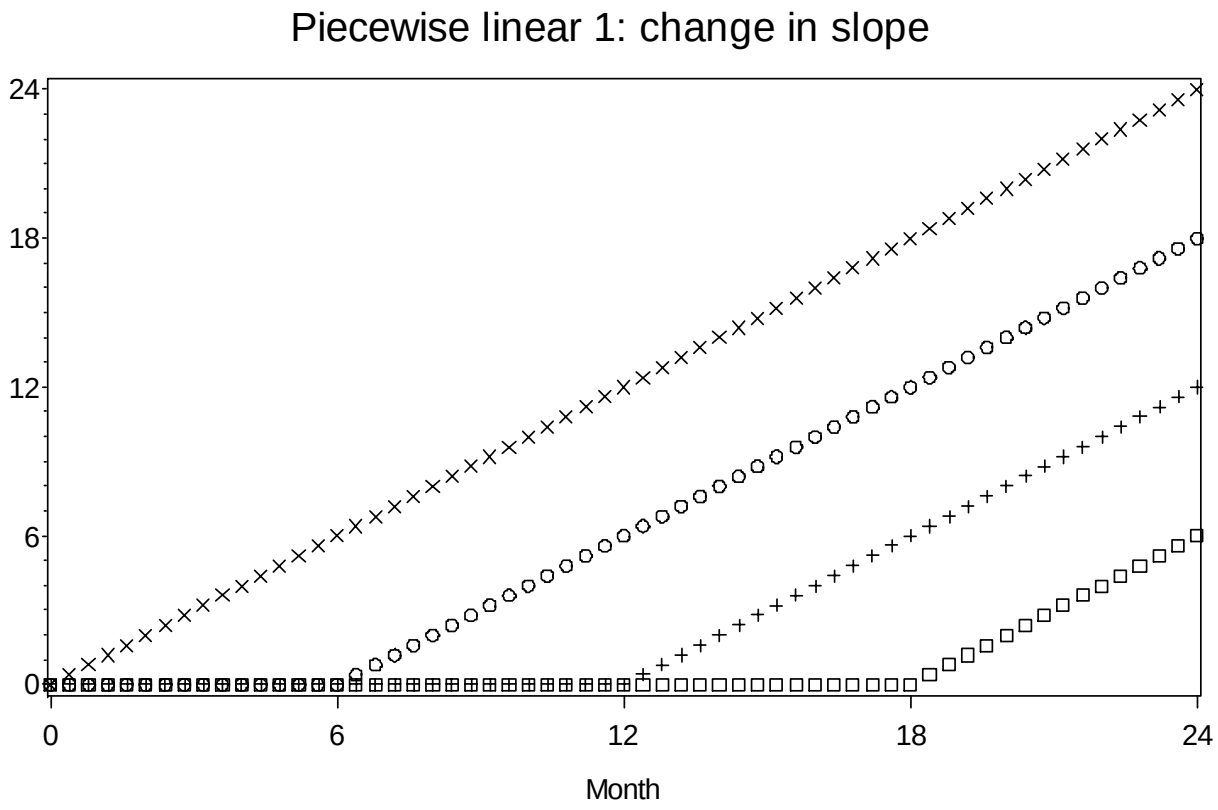

FIGURE 2

PIECEWISE LINEAR 2: NONZERO SLOPE

Sometimes you are more interested, not in a change of slope, but whether the slope is zero. Then the basis functions might be:

```
time0006=min(max(time-00,0),06);
time0612=min(max(time-06,0),06);
time1218=min(max(time-12,0),06);
time1824=min(max(time-18,0),06);
```

Then the coefficient of TIME1218 in a model with TIME0006, TIME0612, and TIME1218 asks whether the slope is nonzero from time 12 to time 18. Figure 3 shows the separate functions. It is worth convincing yourself that these coefficients add up to the overall slope at each time for specific individuals.

There are many variations on these ways of parameterizing piecewise linear models. This same approach can be used with cubic splines to fit smooth curves; Hastie, Tibshirani, and Friedman (2001) have details.

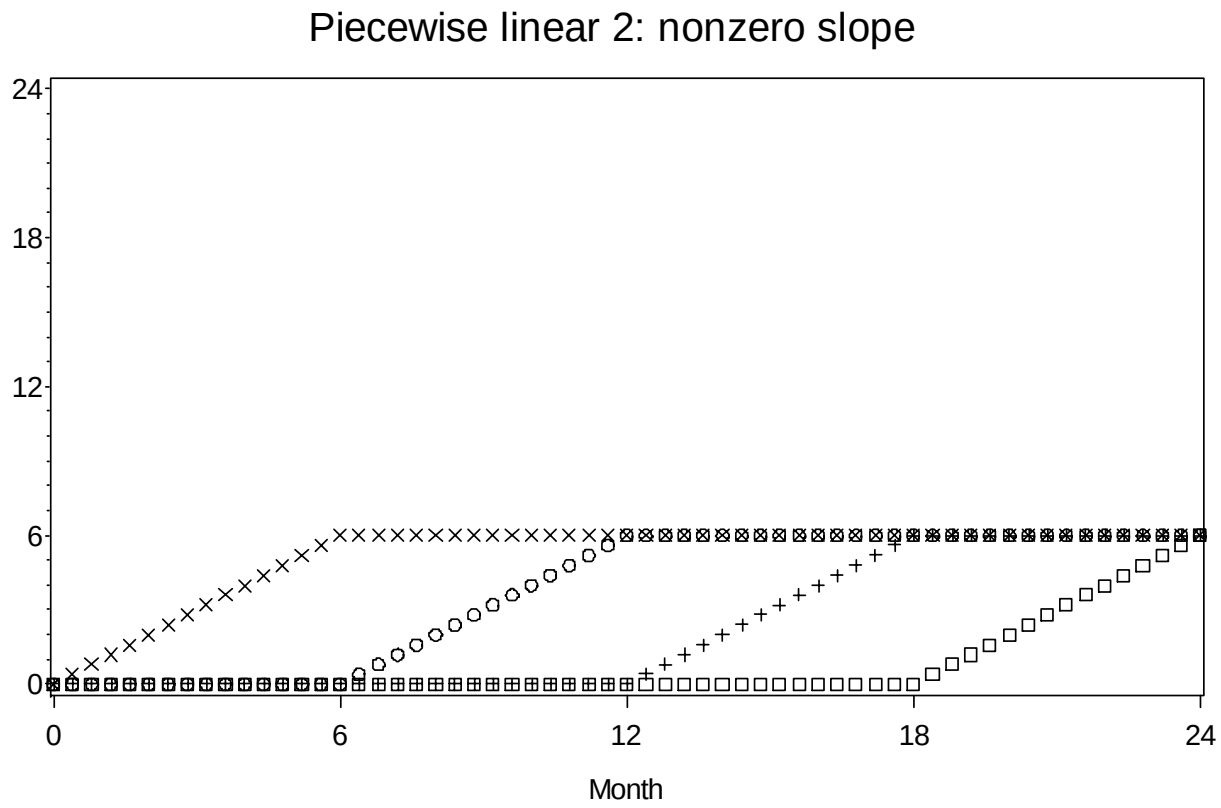

FIGURE 3

AN EXAMPLE OF FITTING WITH PIECEWISE LINEAR FUNCTIONS

How do you actually go about fitting these functions in practice? Consider the following artificial example. Two sets of data were generated, one from a piecewise linear function, and one from a quadratic function designed to be very similar. Figures 4 and 5 show the actual data to be fitted and Figure 6 shows the two underlying functions. We can fit a piecewise constant that fits surprisingly well, but a piecewise linear function fits better and is much smoother (Figure 7). The difference between a fitted piecewise linear function and a fitted quadratic is negligible (Figure 8).

If you fit a full model, you get the same piecewise linear function whether you use the first parameterization (*XX*ON, which tests change in slope) or the second (*XXYY*, which tests slopes against zero). However, you do get different parameter estimates (and associated tests) and if you decide to simplify the model you can get different models.

When we fit using the first parameterization, TIME*XX*ON, we get the following results for a full model:

Variable	Parameter Estimate	Standard Error	t Value	P-value
Intercept	0.11813	0.64896	0.17	0.8562
time00on	1.07358	0.15751	6.82	<.0001
time06on	-0.68941	0.25940	-2.66	0.0102
time12on	-0.18266	0.23624	-0.77	0.4427
time18on	-0.02919	0.25940	-0.10	0.9108

If we remove the least significant variable, TIME18ON, the variable TIME12ON is still nonsignificant. Omitting both those variables leads to the following reduced model:

Variable	Parameter Estimate	Standard Error	t Value	P-value
Intercept	-0.06120	0.62812	-0.10	0.9227
time00on	1.16965	0.13406	8.72	<.0001
time06on	-0.91829	0.15674	-5.86	<.0001

We have found that we cannot reject the null hypothesis that the slope is the same from time 06 onward, i.e., that there is no "kink" at time 12 or at time 18.

When instead we fit using the second parameterization, TIME*XXYY*, we get the following full and reduced models:

Variable	Parameter Estimate	Standard Error	t Value	P-value
Intercept	0.11813	0.64896	0.17	0.8562
time0006	1.07358	0.15751	6.82	<.0001
time0612	0.38418	0.13092	2.93	0.0048
time1218	0.20152	0.13092	1.54	0.1294
time1824	0.17233	0.15751	1.10	0.2786

Variable	Parameter Estimate	Standard Error	t Value	P-value
Intercept	0.10408	0.64995	0.17	0.8733
time0006	1.08111	0.15764	6.86	<.0001
time0612	0.36045	0.12933	2.79	0.0072
time1218	0.28952	0.10347	2.80	0.0070

And here we cannot reject the null hypothesis that the slope is nonzero from time 18 onward. Note that once we drop the variable TIME1824, which has the highest p-value, the variable TIME1218 becomes statistically significant.

How different are these two models and the conclusions you would reach based on them? Figure 9 shows the two models. The model with TIME*XX*ON shows a steady linear increase from time 06 onward, but the model with TIME*XXYY* shows a plateau beginning with time 18. Both fit the data well, but lead to very different extrapolations beyond time 24 and perhaps lead to a different theoretical interpretation.

Maybe even more interesting is that when you apply the same approach to the Z data, the two partial piecewise linear functions come out a little bit different — and both now with a kink at month 12, where neither did before (see Figure 10). What this illustrates is that very slight differences in the data can make a differences in the fit you obtain, just as differences in the parameterizations can: both the Y and Z data could have dome from either model.

Is this potential instability a major drawback of fitting piecewise linear functions? I don't think so. I think it is something you need to be aware of and pay particular attention to the null hypothesis that you are testing. This example is also a good reminder of the dangers of extrapolation fitted models beyond the edge of the data – we really do not know if the data beyond time 24 stays level (as implied by the fitted functions with zero slope from 18 to 24), rises (as implied by the fitted functions with a positive slope from 18 to 24), or declines (as would be implied by a fitted quadratic function).

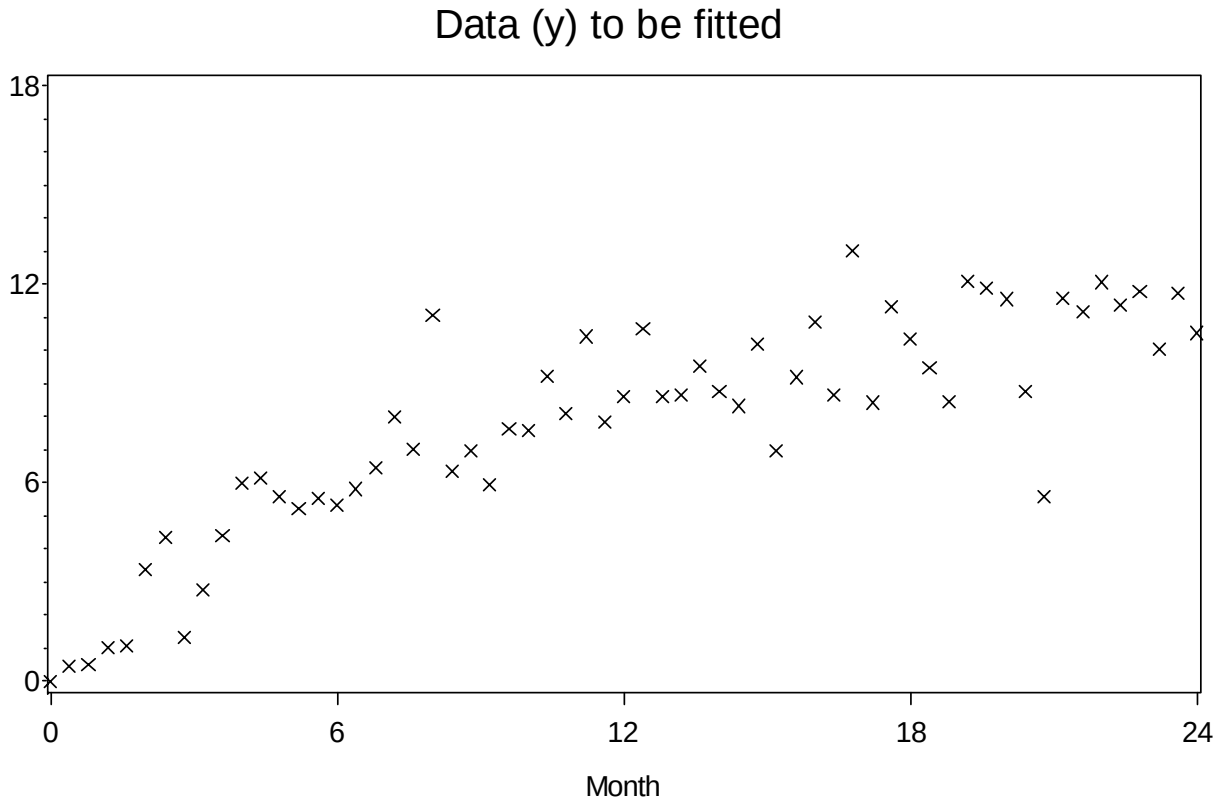

FIGURE 4

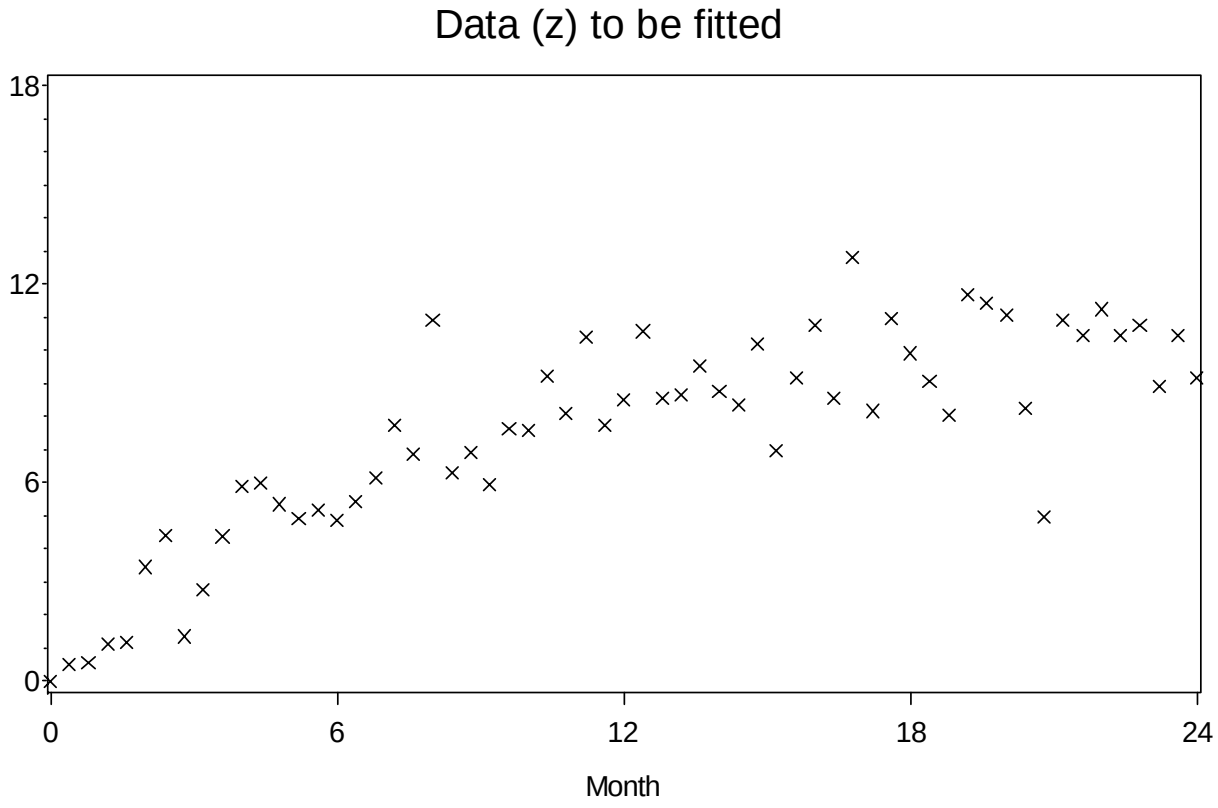

FIGURE 5

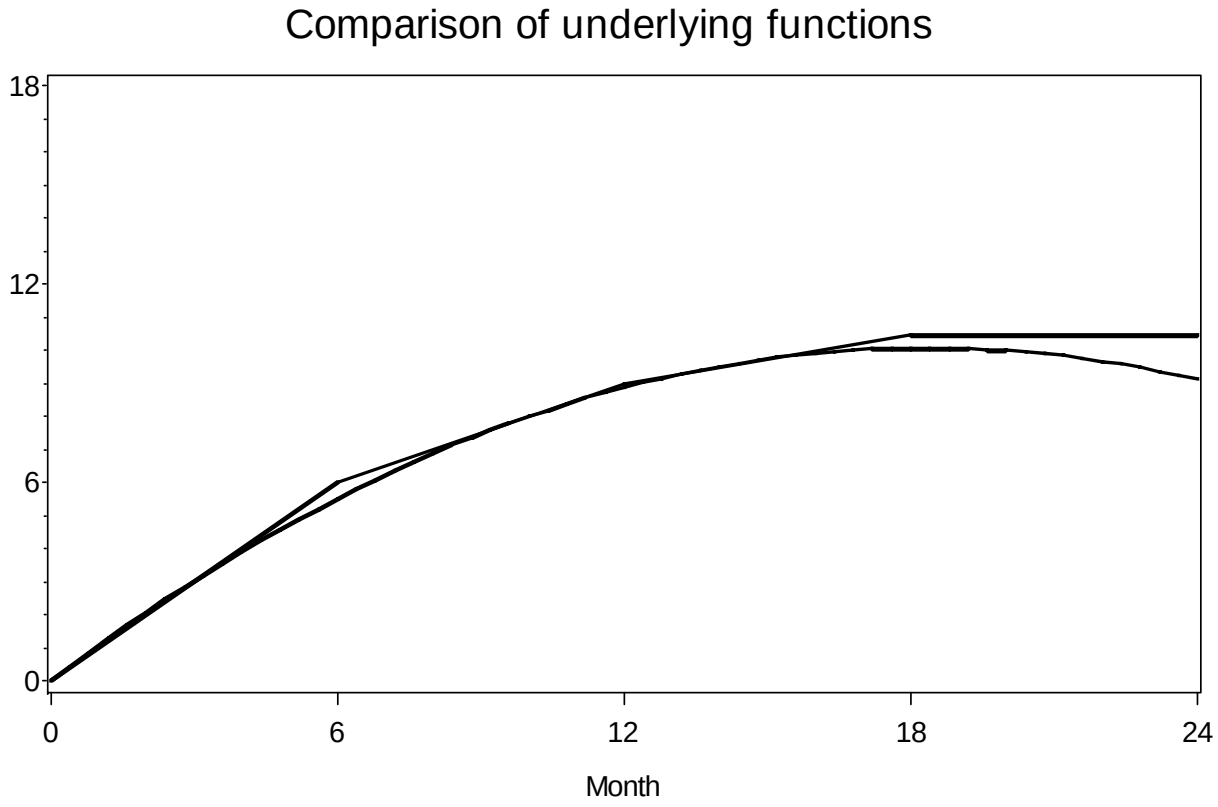

FIGURE 6

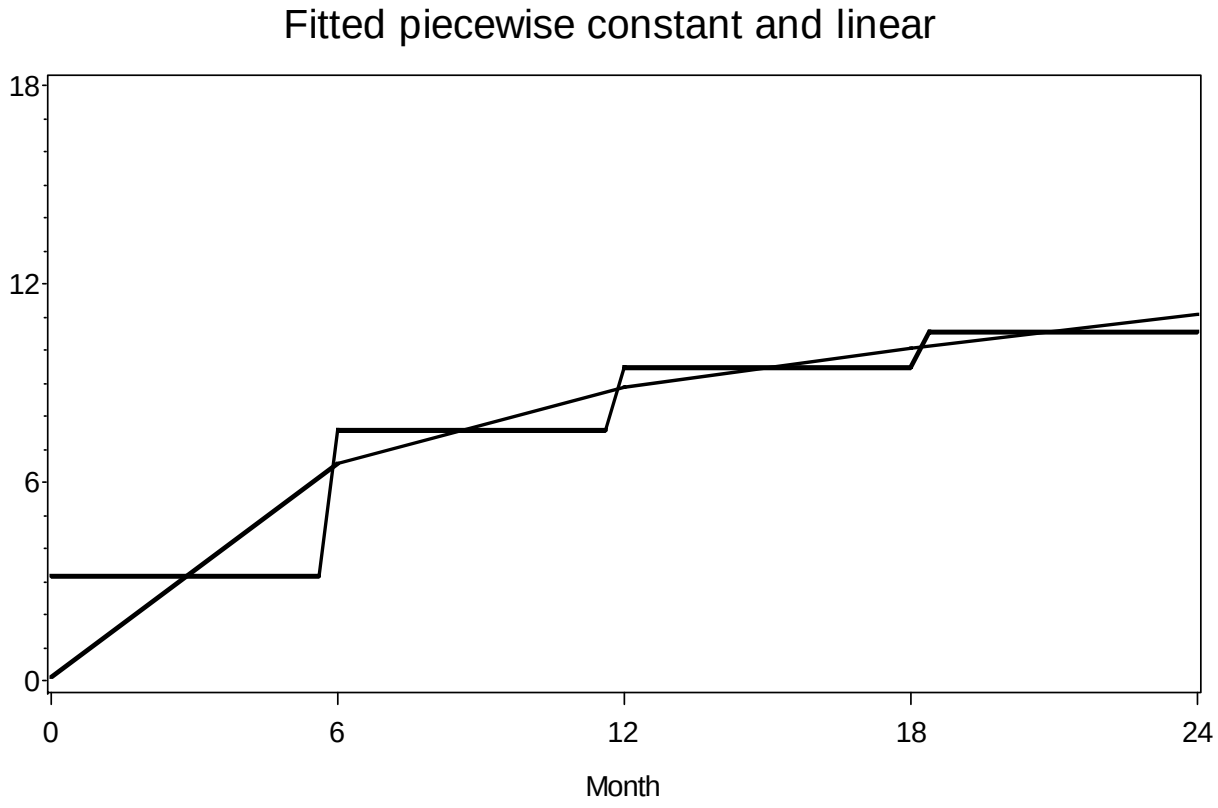

FIGURE 7

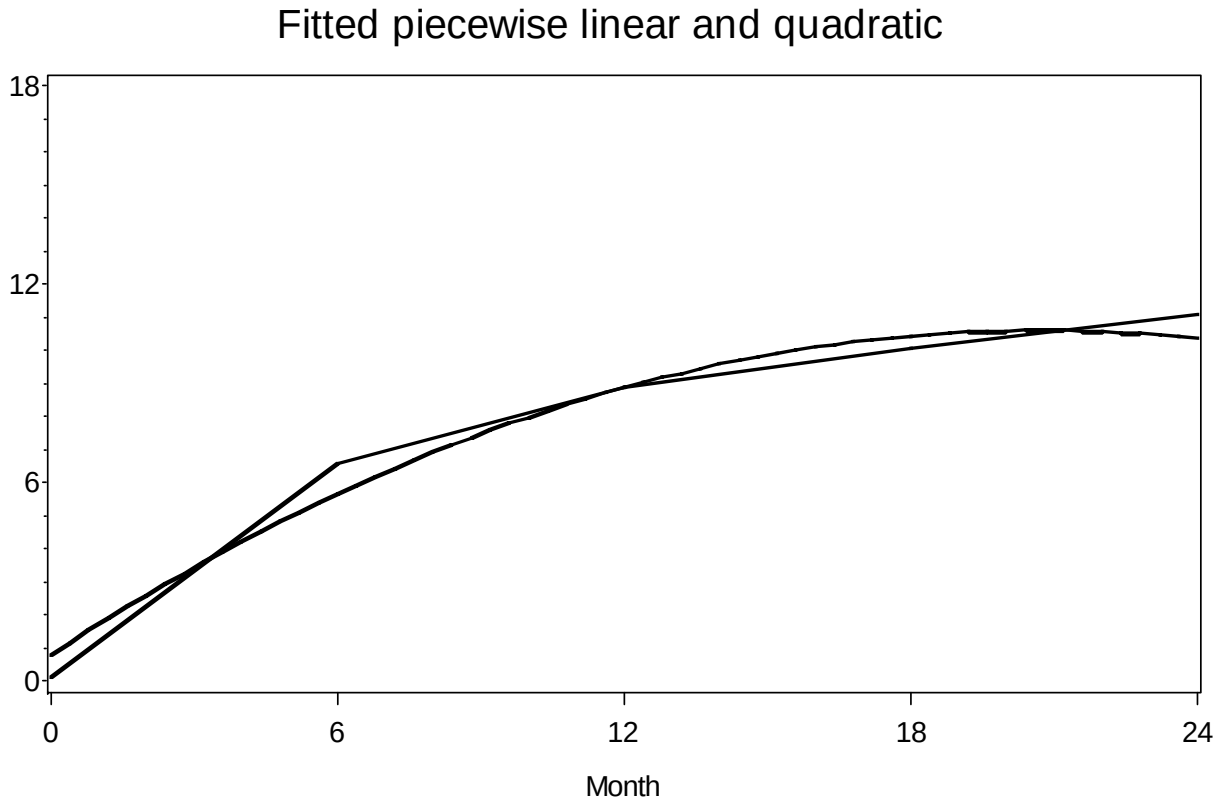

FIGURE 8

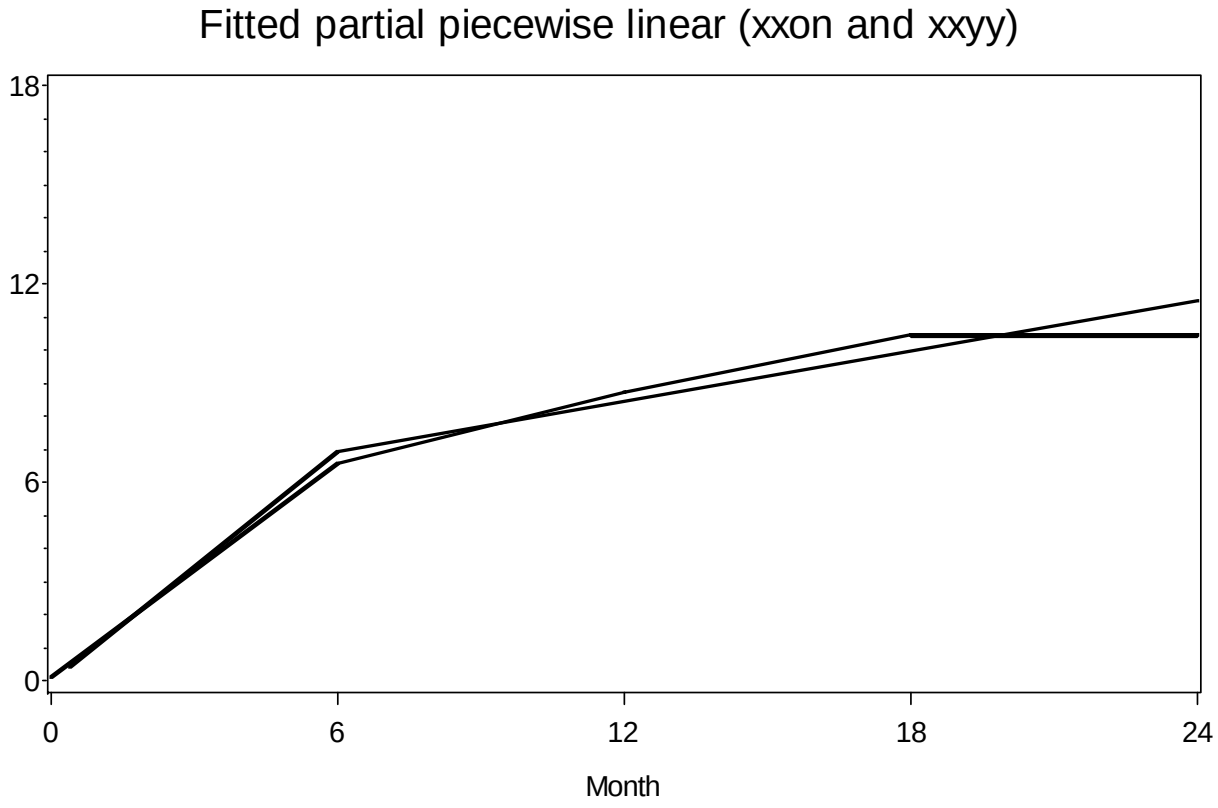

FIGURE 9

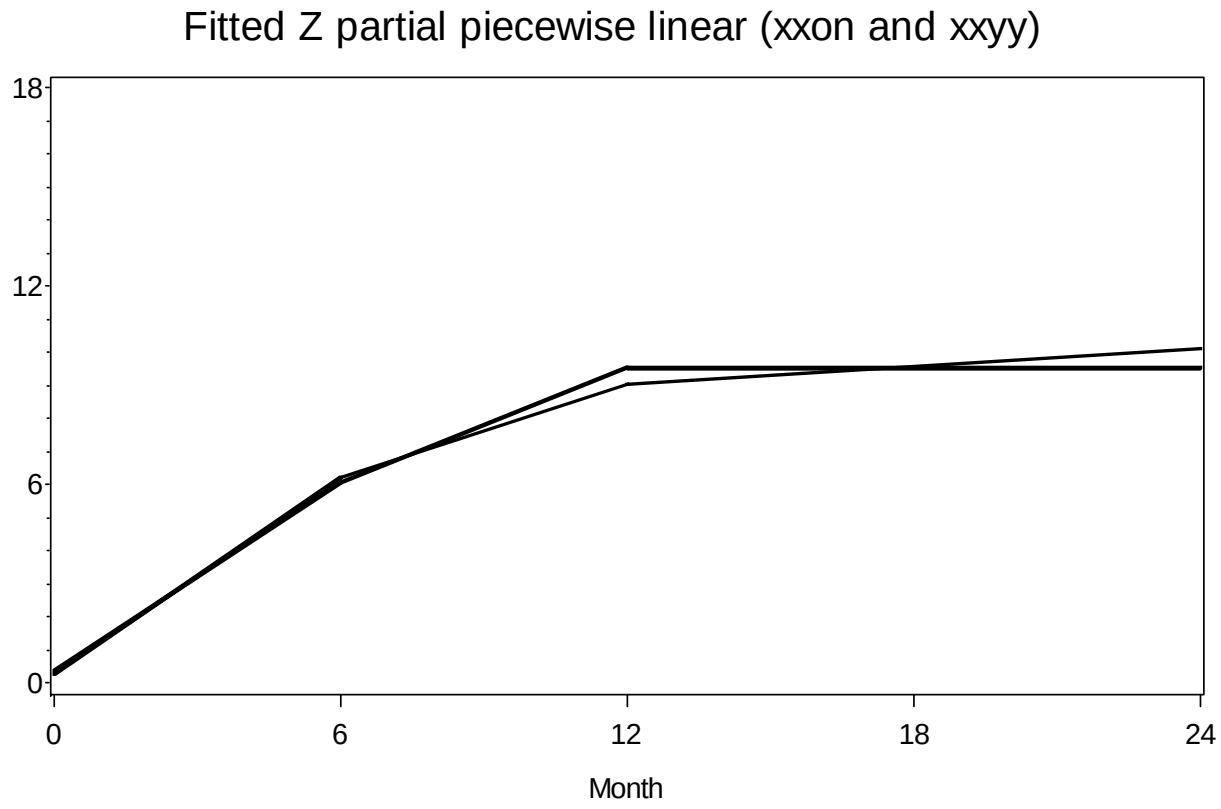

FIGURE 10

CONCLUSION

Different model parameterizations are a powerful tool for specifying models and testing hypotheses. With that power comes responsibility – responsibility to be sure the results you get are what you intended, so you should:

- Pay attention to the order of the levels of the class variables
- Pay attention to the order of the variables in the CLASS statement
- Pay attention to the design matrix, especially in the presence of interactions
- Understand the beta estimates and hypothesis tests
- Understand the syntax and defaults, and what options are and are not available for each method of parameterization
- Be aware of how formatted data are treated

You can fit non-parallel lines by interacting continuous and categorical variables. When you do that, the parameter test that a categorical variable is not statistically significant is a test of whether the lines cross (approximately) at X=0 (the intercept). You may want to consider redefining "zero" for continuous variables to test more interesting hypotheses.

Consider fitting piecewise constant or piecewise linear models. To do so, you need to:

- Decide on a set of "knots" (boundaries) for the pieces
- Define variables that change level or slope at the knots
- Construct the model from these variables using ordinary fitting methods

It is my view that piecewise constant and piecewise linear models are greatly underutilized. I think they are far more valuable in practice than polynomial models. They are a natural stepping-stone to smooth functions defined by cubic splines but are easier to work with and can be used to test very natural hypotheses.

REFERENCES

Hardy, Melissa A. (1993), *Regression with Dummy Variables* (Sage University Paper series on Quantitative Applications in the Social Sciences, 07-093), Newbury Park, CA: Sage.

Hastie, Trevor, Tibshirani, Robert, and Friedman, Jerome (2001), *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, New York: Springer-Verlag.

Pritchard, Michelle L. and Pasta, David J. (2004), "Head of the CLASS: Impress your colleagues with a superior understanding of the CLASS statement in PROC LOGISTIC," Proceedings of the Twenty-Ninth Annual SAS Users Group International Conference, Paper 194-29.

Silva, Stefanie (2003), "How to Get a Graph from a Complex Linear Model," Proceedings of the Western Users of SAS Software Eleventh Annual Conference.

CONTACT INFORMATION

The author welcomes questions and comments. Please direct inquiries to:

David J. Pasta
Vice President, Statistics & Data Operations
Ovation Research Group
120 Howard St., Ste. 600
San Francisco, CA 94105
dpasta@ovation.org
(415) 371-2111