

Chapter 64

The TPSPLINE Procedure

Chapter Table of Contents

OVERVIEW	3323
The Penalized Least Squares Estimate	3323
PROC TPSPLINE with Large Data Sets	3325
GETTING STARTED	3325
SYNTAX	3332
PROC TPSPLINE Statement	3333
BY Statement	3333
FREQ Statement	3333
ID Statement	3334
MODEL Statement	3334
SCORE Statement	3336
OUTPUT Statement	3336
DETAILS	3337
Computational Formulas	3337
ODS Tables Produced by PROC TPSPLINE	3341
EXAMPLES	3341
Example 64.1 Partial Spline Model Fit	3341
Example 64.2 Spline Model With Higher-Order Penalty	3343
Example 64.3 Multiple Minima of the GCV Function	3347
Example 64.4 Large Data Set Application	3352
Example 64.5 Computing a Bootstrap Confidence Interval	3357
REFERENCES	3365

Chapter 64

The TPSPLINE Procedure

Overview

The TPSPLINE procedure uses the penalized least squares method to fit a nonparametric regression model. It computes thin-plate smoothing splines to approximate smooth multivariate functions observed with noise. The TPSPLINE procedure allows great flexibility in the possible form of the regression surface. In particular, PROC TPSPLINE makes no assumptions of a parametric form for the model. The generalized cross validation (GCV) function may be used to select the amount of smoothing.

The TPSPLINE procedure complements the methods provided by the standard SAS regression procedures such as the GLM, REG and NLIN procedures. These procedures can handle most situations in which you specify the regression model and the model is known up to a fixed number of parameters. However, when you have no prior knowledge about the model, or when you know that the data cannot be represented by a model with a fixed number of parameters, you can use the TPSPLINE procedure to model the data.

The TPSPLINE procedure uses the penalized least squares method to fit the data with a flexible model in which the number of effective parameters can be as large as the number of unique design points. Hence, as the sample size increases, the model space increases as well, enabling the thin-plate smoothing spline to fit more complicated situations.

The main features of the TPSPLINE procedure are as follows:

- provides penalized least squares estimates
- supports the use of multidimensional data
- supports multiple SCORE statements
- fits both semiparametric models and nonparametric models
- provides options for handling large data sets
- supports multiple dependent variables
- enables you to choose a particular model by specifying the model degrees of freedom or smoothing parameter

The Penalized Least Squares Estimate

Penalized least squares estimates provide a way to balance fitting the data closely and avoiding excessive roughness or rapid variation. A penalized least squares estimate

is a surface that minimizes the penalized least squares over the class of all surfaces satisfying sufficient regularity conditions.

Define \mathbf{x}_i as a d -dimensional covariate vector, \mathbf{z}_i as a p -dimensional covariate vector, and y_i as the observation associated with $(\mathbf{x}_i, \mathbf{z}_i)$. Assuming that the relation between \mathbf{z}_i and y_i is linear but the relation between \mathbf{x}_i and y_i is unknown, you can fit the data using a semiparametric model as follows:

$$y_i = f(\mathbf{x}_i) + \mathbf{z}_i\beta + \epsilon_i$$

where f is an unknown function that is assumed to be reasonably smooth, $\epsilon_i, i = 1, \dots, n$ are independent, zero-mean random errors, and β is a p -dimensional unknown parametric vector.

This model consists of two parts. The $\mathbf{z}_i\beta$ is the parametric part of the model, and the \mathbf{z}_i are the regression variables. The $f(\mathbf{x}_i)$ is the nonparametric part of the model, and the \mathbf{x}_i are the smoothing variables.

The ordinary least squares method estimates $f(x_i)$ and β by minimizing the quantity:

$$\frac{1}{n} \sum_{i=1}^n (y_i - f(\mathbf{x}_i) - \mathbf{z}_i\beta)^2$$

However, the functional space of $f(\mathbf{x})$ is so large that you can always find a function f that interpolates the data points. In order to obtain an estimate that fits the data well and has some degree of smoothness, you can use the penalized least squares method.

The penalized least squares function is defined as

$$S_\lambda(f) = \frac{1}{n} \sum_{i=1}^n (y_i - f(\mathbf{x}_i) - \mathbf{z}_i\beta)^2 + \lambda J_2(f)$$

where $J_2(f)$ is the penalty on the roughness of f and is defined, in most cases, as the integral of the square of the second derivative of f .

The first term measures the goodness of fit and the second term measures the smoothness associated with f . The λ term is the smoothing parameter, which governs the tradeoff between smoothness and goodness of fit. When λ is large, it heavily penalizes estimates with large second derivatives. Conversely, a small value of λ puts more emphasis on the goodness of fit.

The estimate f_λ is selected from a reproducing kernel Hilbert space, and it can be represented as a linear combination of a sequence of basis functions. Hence, the final estimates of f can be written as

$$f_\lambda(\mathbf{x}_i) = \theta_0 + \sum_{j=1}^d \theta_j x_{ij} + \sum_{j=1}^n \delta_j B_j(\mathbf{x}_i)$$

where B_j is the basis function, which depends on where the data \mathbf{x}_j is located, and θ_j and δ_j are the coefficients that need to be estimated.

For a fixed λ , the coefficients (θ, δ, β) can be estimated by solving an $n \times n$ system.

The smoothing parameter can be chosen by minimizing the generalized cross validation (GCV) function.

If you write

$$\hat{\mathbf{y}} = \mathbf{A}(\lambda)\mathbf{y}$$

then $\mathbf{A}(\lambda)$ is referred to as the *hat* or *smoothing* matrix, and the GCV function $V(\lambda)$ is defined as

$$V(\lambda) = \frac{(1/n)\|(\mathbf{I} - \mathbf{A}(\lambda))\mathbf{y}\|^2}{[(1/n)\text{tr}(\mathbf{I} - \mathbf{A}(\lambda))]^2}$$

PROC TPSPLINE with Large Data Sets

The calculation of the penalized least squares estimate is computationally intensive. The amount of memory and CPU time needed for the analysis depend on the number of unique design points, which corresponds to the number of unknown parameters to be estimated.

You can specify the `D= value` option in the `MODEL` statement to reduce the number of unknown parameters. The option groups design points by the specified range (see the `D=` option on page 3335).

PROC TPSPLINE selects one design point from the group and treats all observations in the group as replicates of that design point. Calculation of the thin-plate smoothing spline estimates are based on the reprocessed data. The way to choose the design point from a group depends on the order of the data. Therefore, different orders of input data may result in different estimates.

This option, by combing several design points into one, reduces the number of unique design points. Therefore, it provides an approximate estimate to the original data. The value you specify determines the range used to group the data.

Getting Started

The following example demonstrates how you can use the TPSPLINE procedure to fit a semiparametric model.

Suppose that y is a continuous variable and $x1$ and $x2$ are two explanatory variables of interest. To fit a smoothing spline model, you can use a `MODEL` statement similar to that used in many regression procedures in the SAS System.

```
proc tpspline;
  model y = (x1 x2);
run;
```

The TPSPLINE procedure can fit semiparametric models; the parentheses in the preceding MODEL statement separates the smoothing variables from the regression variables. The following statements illustrates this syntax.

```
proc tpspline;
  model y = x3 (x1 x2);
run;
```

This model assumes a linear relation with x3 and an unknown functional relation with x1 and x2.

If you want to fit several responses using the same explanatory variables, you can save computation time by using the multiple responses feature in the MODEL statement. For example, if y1 and y2 are two response variables, the following MODEL statement can be used to fit two models. Separate analyses are then performed for each response variable.

```
proc tpspline;
  model y1 y2 = (x1 x2);
run;
```

The following example illustrates the use of PROC TPSPLINE. The data are from Bates, Lindstrom, Wahba, and Yandell (1987).

```
data Measure;
  input x1 x2 y @@;
  datalines;
-1.0 -1.0 15.54483570 -1.0 -1.0 15.76312613
-.5 -1.0 18.67397826 -.5 -1.0 18.49722167
.0 -1.0 19.66086310 .0 -1.0 19.80231311
.5 -1.0 18.59838649 .5 -1.0 18.51904737
1.0 -1.0 15.86842815 1.0 -1.0 16.03913832
-1.0 -.5 10.92383867 -1.0 -.5 11.14066546
-.5 -.5 14.81392847 -.5 -.5 14.82830425
.0 -.5 16.56449698 .0 -.5 16.44307297
.5 -.5 14.90792284 .5 -.5 15.05653924
1.0 -.5 10.91956264 1.0 -.5 10.94227538
-1.0 .0 9.61492010 -1.0 .0 9.64648093
-.5 .0 14.03133439 -.5 .0 14.03122345
.0 .0 15.77400253 .0 .0 16.00412514
.5 .0 13.99627680 .5 .0 14.02826553
1.0 .0 9.55700164 1.0 .0 9.58467047
-1.0 .5 11.20625177 -1.0 .5 11.08651907
-.5 .5 14.83723493 -.5 .5 14.99369172
.0 .5 16.55494349 .0 .5 16.51294369
.5 .5 14.98448603 .5 .5 14.71816070
1.0 .5 11.14575565 1.0 .5 11.17168689
-1.0 1.0 15.82595514 -1.0 1.0 15.96022497
-.5 1.0 18.64014953 -.5 1.0 18.56095997
.0 1.0 19.54375504 .0 1.0 19.80902641
```

```

      .5  1.0  18.56884576      .5  1.0  18.61010439
      1.0  1.0  15.86586951      1.0  1.0  15.90136745
;

```

The data set `Measure` contains three variables `x1`, `x2`, and `y`. Suppose that you want to fit a surface by using the variables `x1` and `x2` to model the response `y`. The variables `x1` and `x2` are spaced evenly on a $[-1 \times 1] \times [-1 \times 1]$ square, and the response `y` is generated by adding a random error to a function $f(x_1, x_2)$. The raw data are plotted using the `G3D` procedure. In order to plot those replicates, the data are jittered a little bit.

```

data Measure1;
  set Measure;
run;

proc sort data=Measure1;
  by x2 x1;
run;

data measure1;
  set measure1; by x1;
  if last.x1 then x1=x1+0.00001;
run;

proc g3d data=Measure1;
  scatter x2*x1=y /size=.5
          zmin=9
          zmax=21
          zticknum=4;
  title "Raw Data";
run;

```

Figure 64.1 displays the raw data.

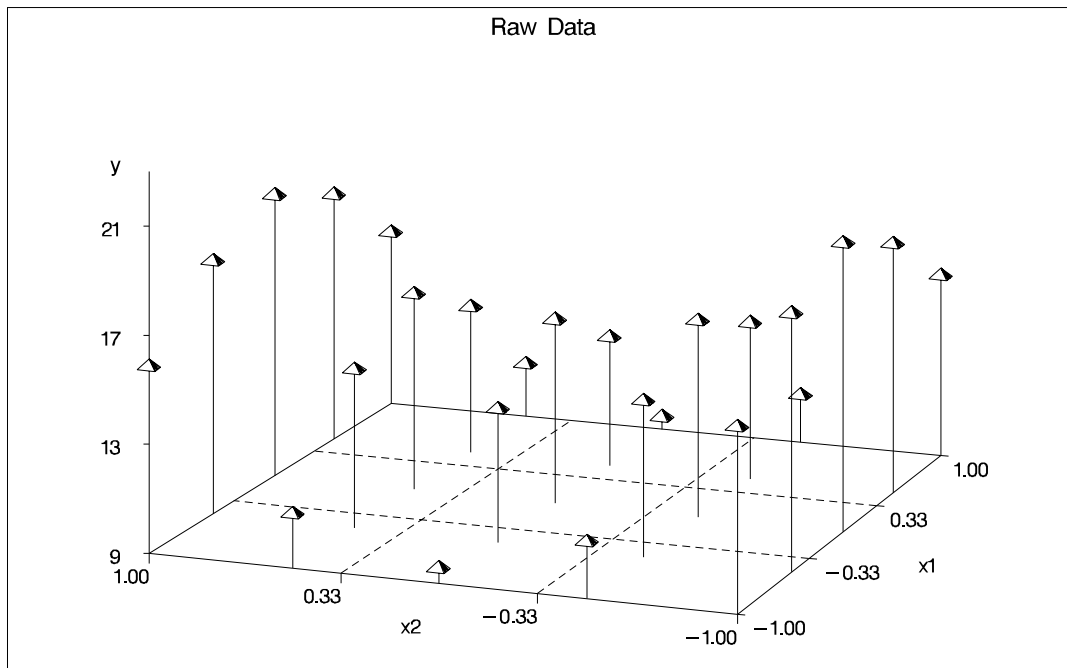


Figure 64.1. Plot of Data Set MEASURE

The following statements invoke the TPSPLINE procedure, using the *Measure* data set as input. In the MODEL statement, the *x1* and *x2* variables are listed as smoothing variables. The LOGNLAMBDA= option returns a list of GCV values with $\log_{10}(n\lambda)$ ranging from -4 to -2 . The OUTPUT statement creates the data set *estimate* to contain the predicted values and the 95% upper and lower confidence limits.

```
proc tpspline data=Measure;
  model y=(x1 x2) /lognlambda=(-4 to -2 by 0.1);
  output out=estimate pred uclm lclm;
run;

proc print data=estimate;
run;
```

The results of this analysis are displayed in the following figures. Figure 64.2 shows that the data set *Measure* contains 50 observations with 25 unique design points. The GCV values are listed along with the \log_{10} of $n\lambda$. The value of $\log_{10}(n\lambda)$ that minimizes the GCV function is around -3.5 . The final thin-plate smoothing spline estimate is based on LOGNLAMBDA = -3.4762 . The residual sum of squares is 0.246110, and the degrees of freedom is 24.593203. The standard deviation, defined as $\text{RSS}/(\text{Tr}(\mathbf{I}-\mathbf{A}))$, is 0.098421. The predictions and 95% confidence limits are displayed in Figure 64.3.


```

The TPSPLINE Procedure
Dependent Variable: y

Summary of Input Data Set

Number of Non-Missing Observations    50
Number of Missing Observations       0
Unique Smoothing Design Points       25

Summary of Final Model

Number of Regression Variables         0
Number of Smoothing Variables         2
Order of Derivative in the Penalty    2
Dimension of Polynomial Space          3

```

Figure 64.2. Output from PROC TPSPLINE

```

The TPSPLINE Procedure
Dependent Variable: y

GCV Function

log10(n*Lambda)      GCV
-4.000000            0.019215
-3.900000            0.019183
-3.800000            0.019148
-3.700000            0.019113
-3.600000            0.019082
-3.500000            0.019064*
-3.400000            0.019074
-3.300000            0.019135
-3.200000            0.019286
-3.100000            0.019584
-3.000000            0.020117
-2.900000            0.021015
-2.800000            0.022462
-2.700000            0.024718
-2.600000            0.028132
-2.500000            0.033165
-2.400000            0.040411
-2.300000            0.050614
-2.200000            0.064699
-2.100000            0.083813
-2.000000            0.109387

Note: * indicates minimum GCV value.

Summary Statistics
of Final Estimation

log10(n*Lambda)      -3.476189
Smoothing Penalty    2558.143232
Residual SS          0.246110
Tr(I-A)              25.406797
Model DF              24.593203
Standard Deviation    0.098421

```

Estimates from Proc TPSPLINE						
Obs	x1	x2	y	P_y	LCLM_y	UCLM_y
1	-1.0	-1.0	15.5448	15.6474	15.5115	15.7832
2	-1.0	-1.0	15.7631	15.6474	15.5115	15.7832
3	-0.5	-1.0	18.6740	18.5783	18.4430	18.7136
4	-0.5	-1.0	18.4972	18.5783	18.4430	18.7136
5	0.0	-1.0	19.6609	19.7270	19.5917	19.8622
6	0.0	-1.0	19.8023	19.7270	19.5917	19.8622
7	0.5	-1.0	18.5984	18.5552	18.4199	18.6905
8	0.5	-1.0	18.5190	18.5552	18.4199	18.6905
9	1.0	-1.0	15.8684	15.9436	15.8077	16.0794
10	1.0	-1.0	16.0391	15.9436	15.8077	16.0794
11	-1.0	-0.5	10.9238	11.0467	10.9114	11.1820
12	-1.0	-0.5	11.1407	11.0467	10.9114	11.1820
13	-0.5	-0.5	14.8139	14.8246	14.6896	14.9597
14	-0.5	-0.5	14.8283	14.8246	14.6896	14.9597
15	0.0	-0.5	16.5645	16.5102	16.3752	16.6452
16	0.0	-0.5	16.4431	16.5102	16.3752	16.6452
17	0.5	-0.5	14.9079	14.9812	14.8461	15.1162
18	0.5	-0.5	15.0565	14.9812	14.8461	15.1162
19	1.0	-0.5	10.9196	10.9497	10.8144	11.0850
20	1.0	-0.5	10.9423	10.9497	10.8144	11.0850
21	-1.0	0.0	9.6149	9.6372	9.5019	9.7724
22	-1.0	0.0	9.6465	9.6372	9.5019	9.7724
23	-0.5	0.0	14.0313	14.0188	13.8838	14.1538
24	-0.5	0.0	14.0312	14.0188	13.8838	14.1538
25	0.0	0.0	15.7740	15.8822	15.7472	16.0171
26	0.0	0.0	16.0041	15.8822	15.7472	16.0171
27	0.5	0.0	13.9963	14.0006	13.8656	14.1356
28	0.5	0.0	14.0283	14.0006	13.8656	14.1356
29	1.0	0.0	9.5570	9.5769	9.4417	9.7122
30	1.0	0.0	9.5847	9.5769	9.4417	9.7122
31	-1.0	0.5	11.2063	11.1614	11.0261	11.2967
32	-1.0	0.5	11.0865	11.1614	11.0261	11.2967
33	-0.5	0.5	14.8372	14.9182	14.7831	15.0532
34	-0.5	0.5	14.9937	14.9182	14.7831	15.0532
35	0.0	0.5	16.5549	16.5386	16.4036	16.6736
36	0.0	0.5	16.5129	16.5386	16.4036	16.6736
37	0.5	0.5	14.9845	14.8549	14.7199	14.9900
38	0.5	0.5	14.7182	14.8549	14.7199	14.9900
39	1.0	0.5	11.1458	11.1727	11.0374	11.3080
40	1.0	0.5	11.1717	11.1727	11.0374	11.3080
41	-1.0	1.0	15.8260	15.8851	15.7493	16.0210
42	-1.0	1.0	15.9602	15.8851	15.7493	16.0210
43	-0.5	1.0	18.6401	18.5946	18.4593	18.7299
44	-0.5	1.0	18.5610	18.5946	18.4593	18.7299
45	0.0	1.0	19.5438	19.6729	19.5376	19.8081
46	0.0	1.0	19.8090	19.6729	19.5376	19.8081
47	0.5	1.0	18.5688	18.5832	18.4478	18.7185
48	0.5	1.0	18.6101	18.5832	18.4478	18.7185
49	1.0	1.0	15.8659	15.8761	15.7402	16.0120
50	1.0	1.0	15.9014	15.8761	15.7402	16.0120

Figure 64.3. Data Set ESTIMATE

The fitted surface is plotted with PROC G3D as follows.

```
proc g3d data=estimate;
  plot x2*x1=p_y/grid
      zmin=9
      zmax=21
      zticknum=4;
```

```

title 'Plot of Fitted Surface';
run;

```

The resulting plot is displayed in Figure 64.4.

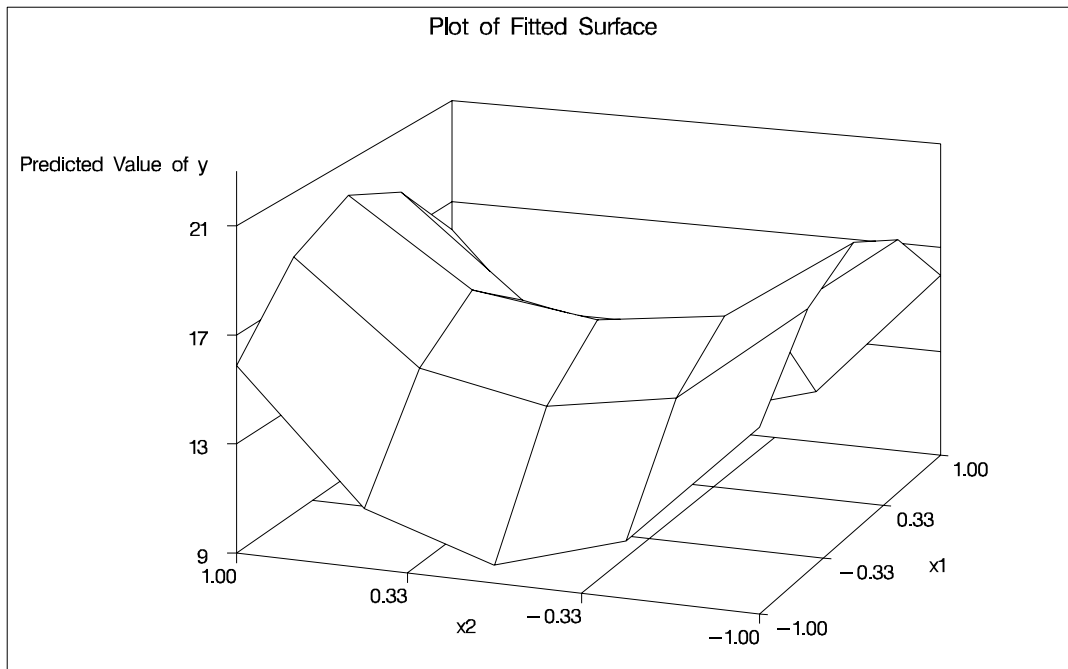


Figure 64.4. Plot of TPSPLINE Fit of Data Set Measure

Because the data in data set `Measure` are very sparse, the fitted surface is not smooth. To produce a smoother surface, the following statements generate the data set `Pred` in order to obtain a finer grid. The `SCORE` statement evaluates the fitted surface at those new design points.

```

data pred;
  do x1=-1 to 1 by 0.1;
    do x2=-1 to 1 by 0.1;
      output;
    end;
  end;
run;

proc tpspline data=measure;
  model y=(x1 x2)/lognlambda=(-4 to -2 by 0.1);
  score data=pred out=predy;
run;

proc g3d data=predy;
  plot x2*x1=p_y/grid
      zmin=9
      zmax=21
      zticknum=4;

```

```

title 'Plot of Fitted Surface on a Fine Grid';
run;

```

The surface plot based on the finer grid is displayed in Figure 64.5. The plot shows that a parametric model with quadratic terms of x_1 and x_2 provides a reasonable fit to the data.

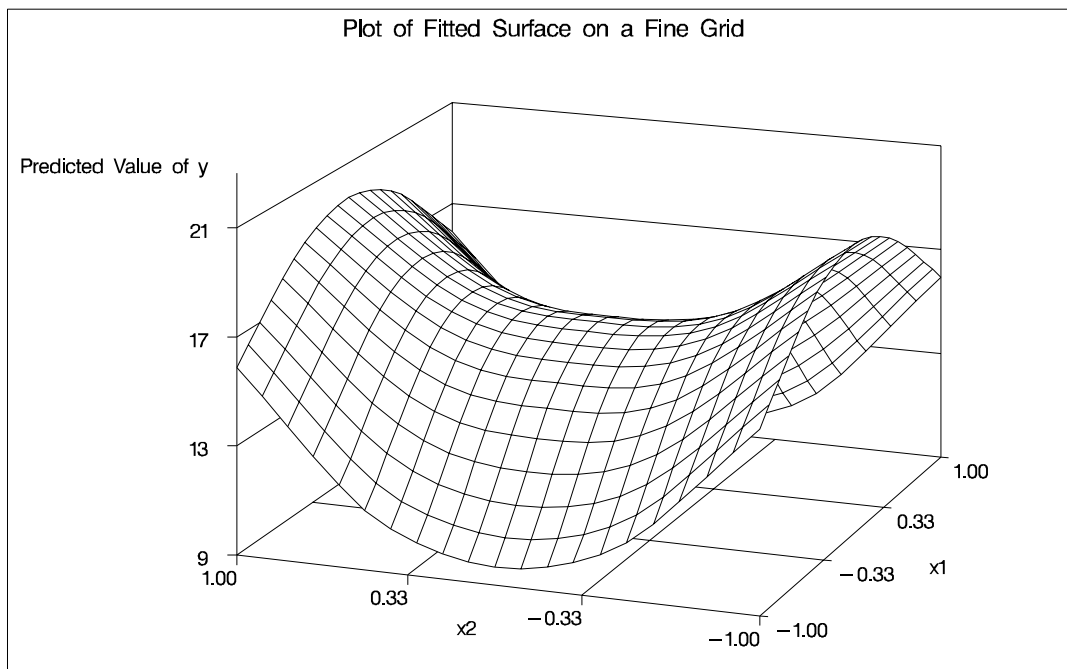


Figure 64.5. Plot of TPSPLINE fit

Syntax

```

PROC TPSPLINE < option > ;
  MODEL dependents = < variables > (variables) < /options > ;
  SCORE data=SAS-data-set out=SAS-data-set ;
  OUTPUT < out=SAS-data-set > keyword < ... keyword > ;
  BY variables ;
  FREQ variable ;
  ID variables ;

```

The syntax in PROC TPSPLINE is similar to that of other regression procedures in the SAS System. The PROC TPSPLINE and MODEL statements are required. The SCORE statement can appear multiple times; all other statements appear only once.

The syntax for PROC TPSPLINE is described in the following sections in alphabetical order after the description of the PROC TPSPLINE statement.

PROC TPSPLINE Statement

PROC TPSPLINE < option > ;

The PROC TPSPLINE statement invokes the procedure. You can specify the following option.

DATA=SAS-data-set

specifies the SAS data set to be read by PROC TPSPLINE. The default value is the most recently created data set.

BY Statement

BY variables ;

You can specify a BY statement with PROC TPSPLINE to obtain separate analysis on observations in groups defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data using the SORT procedure with a similar BY statement.
- Specify the BY statement option NOTSORTED or DESCENDING in the BY statement for the TPSPLINE procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables using the DATASETS procedure.

For more information on the BY statement, refer to the discussion in *SAS Language Reference: Concepts*. For more information on the DATASETS procedure, refer to the discussion in the *SAS Procedures Guide*.

FREQ Statement

FREQ variable ;

If one variable in your input data set represents the frequency of occurrence for other values in the observation, specify the variable's name in a FREQ statement. PROC TPSPLINE treats the data as if each observation appears n times, where n is the value of the FREQ variable for the observation. If the value of the FREQ variable is less than one, the observation is not used in the analysis. Only the integer portion of the value is used.

ID Statement

ID *variables* ;

The variables in the ID statement are copied from the input data set to the OUT= data set. If you omit the ID statement, only the variables used in the MODEL statement and requested statistics are included in the output data set.

MODEL Statement

MODEL *dependents* = < *regression variables* > (*smoothing variables*) < /*options* > ;

The MODEL statement specifies the dependent variables, the independent regression variables, which are listed with no parentheses, and the independent smoothing variables, which are listed inside parentheses.

The regression variables are optional. At least one smoothing variable is required, and it must be listed after the regression variables. No variables can be listed in both the regression variable list and the smoothing variable list.

If you specify more than one dependent variable, PROC TPSPLINE calculates a thin-plate smoothing spline estimate for each dependent variable, using the regression variables and smoothing variables specified on the right-hand side.

If you specify regression variables, PROC TPSPLINE fits a semiparametric model using the regression variables as the linear part of the model.

You can specify the following options in the MODEL statement.

ALPHA=*number*

specifies the significance level α of the confidence limits on the final thin-plate smoothing spline estimate when you request confidence limits to be included in the output data set. Specify *number* as a value between 0 and 1. The default value is 0.05. See the “OUTPUT Statement” section on page 3336 for more information on the OUTPUT statement.

DF=*number*

specifies the degrees of freedom of the thin-plate smoothing spline estimate, defined as

$$df = \text{trace}(\mathbf{A}(\lambda))$$

where $\mathbf{A}(\lambda)$ is the *hat* matrix. Specify *number* as a value between zero and the number of unique design points.

DISTANCE=*number*

D=*number*

defines a range such that if two data points $(\mathbf{x}_i, \mathbf{z}_i)$ and $(\mathbf{x}_j, \mathbf{z}_j)$ satisfy

$$\max_k |x_{ik} - x_{jk}| \leq D/2$$

then these data points are treated as replicates, where \mathbf{x}_i are the smoothing variables and \mathbf{z}_i are the regression variables.

You can use the DISTANCE= option to reduce the number of unique design points by treating nearby data as replicates. This can be useful when you have a large data set. The default value is 0.

LAMBDA0=*number*

specifies the smoothing parameter, λ_0 , to be used in the thin-plate smoothing spline estimate. By default, PROC TPSPLINE uses the λ parameter that minimizes the GCV function for the final fit. The LAMBDA0= value must be positive.

LAMBDA=*list-of-values*

specifies a set of values for the λ parameter. PROC TPSPLINE returns a GCV value for each λ point that you specify. You can use the LAMBDA= option to study the GCV function curve for a set of values for λ . All values listed in the LAMBDA= option must be positive.

LOGNLAMBDA0=*number*

LOGNL0=*number*

specifies the smoothing parameter λ_0 on the $\log_{10}(n\lambda)$ scale. If you specify both the LOGNL0= and LAMBDA0= options, only the value provided by the LOGNL0= option is used. By default, PROC TPSPLINE uses the λ parameter that minimizes the GCV function for the estimate.

LOGNLAMBDA=*list-of-values*

LOGNL=*list-of-values*

specifies a set of values for the λ parameter on the $\log_{10}(n\lambda)$ scale. PROC TPSPLINE returns a GCV value for each λ point that you specify. You can use the LOGNLAMBDA= option to study the GCV function curve for a set of λ values. If you specify both the LOGNL= and LAMBDA= options, only the list of values provided by LOGNL= option is used.

In some cases, the LOGNL= option may be preferred over the LAMBDA= option. Because the LAMBDA= value must be positive, a small change in that value can result in a major change in the GCV value. If you instead specify λ on the \log_{10} scale, the allowable range is enlarged to include negative values. Thus, the GCV function is less sensitive to changes in LOGNLAMBDA.

M=*number*

specifies the order of the derivative in the penalty term. The M= value must be a positive integer. The default value is the $\max(2, \text{INT}(d/2) + 1)$, where d is the number of smoothing variables.

SCORE Statement

SCORE *DATA=SAS-data-set OUT=SAS-data-set* ;

The SCORE statement calculates predicted values for a new data set. If you have multiple data sets to predict, you can specify multiple SCORE statements. You must use a SCORE statement for each data set.

The following keywords must be specified in the SCORE statement.

DATA=SAS-data-set

specifies the input SAS data set containing the smoothing variables **x** and regression variables **z**. The predicted response (**y**) value is computed for each (**x**, **z**) pair. The data set must include all independent variables specified in the MODEL statement.

OUT=SAS-data-set

specifies the name of the SAS data set to contain the predictions.

OUTPUT Statement

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

The OUTPUT statement creates a new SAS data set containing diagnostic measures calculated after fitting the model.

You can request a variety of diagnostic measures that are calculated for each observation in the data set. The new data set contains the variables specified in the MODEL statement in addition to the requested variables. If no *keyword* is present, the data set contains only the predicted values.

Details on the specifications in the OUTPUT statement are as follows.

OUT=SAS-data-set

specifies the name of the new data set to contain the diagnostic measures. This specification is required.

keyword

specifies the statistics to include in the output data set. The names of the new variables that contain the statistics are formed by using a prefix of one or more characters that identify the statistic, followed by an underscore (_), followed by the dependent variable name.

For example, suppose that you have two dependent variables, say **y1** and **y2**, and you specify the keywords PRED, ADIAG, and UCLM. The output SAS data set will contain the following variables.

- P_y1 and P_y2
- ADIAG_y1 and ADIAG_y2
- UCLM_y1 and UCLM_y2

The keywords and the statistics they represent are as follows:

RESID R	residual values, calculated as ACTUAL - PREDICTED
PRED	predicted values
STD	standard error of the mean predicted value
UCLM	upper limit of the confidence interval for the expected value of the dependent variables. By default, PROC TPSPLINE computes 95% confidence limits.
LCLM	lower limit of the confidence interval for the expected value of the dependent variables. By default, PROC TPSPLINE computes 95% confidence limits.
ADIAG	diagonal element of the hat matrix associated with the observation

Details

Computational Formulas

The theoretical foundations for the thin-plate smoothing spline are described in Duchon (1976, 1977) and Meinguet (1979). Further results and applications are given in Wahba and Wendelberger (1980), Hutchinson and Bischof (1983), and Seaman and Hutchinson (1985).

Suppose that \mathcal{H}_m is a space of functions whose partial derivatives of total order m are in $L_2(E^d)$ where E^d is the domain of \mathbf{x} .

Now, consider the data model

$$y_i = f(x_1(i), \dots, x_d(i)) + \epsilon_i, i = 1, \dots, n$$

where $f \in \mathcal{H}_m$.

Using the notation from the section “The Penalized Least Squares Estimate” on page 3323, for a fixed λ , estimate f by minimizing the penalized least squares function

$$\frac{1}{n} \sum_{i=1}^n (y_i - f(\mathbf{x}_i) - \mathbf{z}_i \beta)^2 + \lambda J_m(f)$$

There are several ways to define $J_m(f)$. For the thin-plate smoothing spline, with x of dimension d , define $J_m(f)$ as

$$J_m(f) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \sum \frac{m!}{\alpha_1! \cdots \alpha_d!} \left[\frac{\partial^m f}{\partial x_1^{\alpha_1} \cdots \partial x_d^{\alpha_d}} \right]^2 dx_1 \cdots dx_d$$

where $\sum_i \alpha_i = m$.

When $d = 2$ and $m = 2$, $J_m(f)$ is as follows:

$$J_2(f) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \left(\left[\frac{\partial^2 f}{\partial x_1^2} \right]^2 + 2 \left[\frac{\partial^2 f}{\partial x_1 \partial x_2} \right]^2 + \left[\frac{\partial^2 f}{\partial x_2^2} \right]^2 \right) dx_1 dx_2$$

In general, m and d must satisfy the condition that $2m - d > 0$. For the sake of simplicity, the formulas and equations that follow assume $m = 2$. Refer to Wahba (1990) and Bates et al. (1987) for more details.

Duchon (1976) showed that f_λ can be represented as

$$f_\lambda(\mathbf{x}_i) = \theta_0 + \sum_{j=1}^d \theta_j \mathbf{x}_{ij} + \sum_{j=1}^n \delta_j \mathbf{E}_2(\mathbf{x}_i - \mathbf{x}_j)$$

where $E_2(s) = \frac{1}{2^3 \pi} \|s\|^2 \ln(\|s\|)$.

If you define $\mathbf{K} = (\mathbf{K})_{ij} = \mathbf{E}_2(\mathbf{x}_i - \mathbf{x}_j)$ and $\mathbf{T} = (\mathbf{T})_{ij} = (\mathbf{x}_{ij})$, the goal is to find coefficients β , θ , and δ that minimize

$$S_\lambda(\beta, \theta, \delta) = \frac{1}{n} \|\mathbf{y} - \mathbf{T}\theta - \mathbf{K}\delta - \mathbf{Z}\beta\|^2 + \lambda \delta^T \mathbf{K} \delta$$

A unique solution is guaranteed if the matrix \mathbf{T} is of full rank and $\delta^T \mathbf{K} \delta \geq 0$.

If $\alpha = \begin{pmatrix} \theta \\ \beta \end{pmatrix}$ and $\mathbf{X} = (\mathbf{T} : \mathbf{Z})$, the expression for S_λ becomes

$$\frac{1}{n} \|\mathbf{y} - \mathbf{X}\alpha - \mathbf{K}\delta\|^2 + \lambda \delta^T \mathbf{K} \delta$$

The coefficients α and δ can be obtained by solving

$$\begin{aligned} (\mathbf{K} + n\lambda \mathbf{I}_n) \delta + \mathbf{X}\alpha &= \mathbf{y} \\ \mathbf{X}^T \delta &= \mathbf{0} \end{aligned}$$

To compute α and δ , let the QR decomposition of \mathbf{X} be

$$\mathbf{X} = (\mathbf{Q}_1 : \mathbf{Q}_2) \begin{pmatrix} \mathbf{R} \\ \mathbf{0} \end{pmatrix}$$

where $(\mathbf{Q}_1 : \mathbf{Q}_2)$ is an orthogonal matrix and \mathbf{R} is upper triangular, with $\mathbf{X}^T \mathbf{Q}_2 = \mathbf{0}$ (Dongarra et al. 1979).

Since $\mathbf{X}^T \delta = \mathbf{0}$, δ must be in the column space of \mathbf{Q}_2 . Therefore, δ can be expressed as $\delta = \mathbf{Q}_2 \gamma$ for a vector γ . Substituting $\delta = \mathbf{Q}_2 \gamma$ into the preceding equation and multiplying through by \mathbf{Q}_2^T gives

$$\mathbf{Q}_2^T (\mathbf{K} + n\lambda \mathbf{I}_n) \mathbf{Q}_2 \gamma = \mathbf{Q}_2^T \mathbf{y}$$

or

$$\delta = \mathbf{Q}_2 \gamma = \mathbf{Q}_2 [\mathbf{Q}_2^T (\mathbf{K} + n\lambda \mathbf{I}_n) \mathbf{Q}_2]^{-1} \mathbf{Q}_2^T \mathbf{y}$$

The coefficient α can be obtained by solving

$$\mathbf{R} \alpha = \mathbf{Q}_1^T [\mathbf{y} - (\mathbf{K} + n\lambda \mathbf{I}_n) \delta]$$

The influence matrix $\mathbf{A}(\lambda)$ is defined as

$$\hat{\mathbf{y}} = \mathbf{A}(\lambda) \mathbf{y}$$

and has the form

$$\mathbf{A}(\lambda) = \mathbf{I} - n\lambda \mathbf{Q}_2 [\mathbf{Q}_2^T (\mathbf{K} + n\lambda \mathbf{I}_n) \mathbf{Q}_2]^{-1} \mathbf{Q}_2^T$$

Similar to the regression case, and if you consider the trace of $\mathbf{A}(\lambda)$ as the degrees of freedom for the information signal and the trace of $(\mathbf{I}_n - \mathbf{A}(\lambda))$ as the degrees of freedom for the noise component, the estimate σ^2 can be represented as

$$\hat{\sigma}^2 = \frac{RSS(\lambda)}{\text{Trace}(\mathbf{I}_n - \mathbf{A}(\lambda))}$$

where $RSS(\lambda)$ is the residual sum of squares. Theoretical properties of these estimates have not yet been published. However, good numerical results in simulation studies have been described by several authors. For more information, refer to O'Sullivan and Wong (1987), Nychka (1986a, 1986b, and 1988), and Hall and Titterton (1987).

Confidence Intervals

Viewing the spline model as a Bayesian model, Wahba (1983) proposed Bayesian confidence intervals for smoothing spline estimates as follows:

$$\hat{f}_\lambda(x_i) \pm z_{\alpha/2} \sqrt{\hat{\sigma}^2 a_{ii}(\lambda)}$$

where $a_{ii}(\lambda)$ is the i th diagonal element of the $\mathbf{A}(\lambda)$ matrix and $z_{\alpha/2}$ is the $\alpha/2$ point of the normal distribution. The confidence intervals are interpreted as intervals “across the function” as opposed to point-wise intervals.

Suppose that you fit a spline estimate to experimental data that consists of a true function f and a random error term, ϵ_i . In repeated experiments, it is likely that about $100(1 - \alpha)\%$ of the confidence intervals cover the corresponding true values, although some values are covered every time and other values are not covered by the confidence intervals most of the time. This effect is more pronounced when the true surface or surface has small regions of particularly rapid change.

Smoothing Parameter

The quantity λ is called the smoothing parameter, which controls the balance between the goodness of fit and the smoothness of the final estimate.

A large λ heavily penalizes the m th derivative of the function, thus forcing $f^{(m)}$ close to 0. The final estimating function satisfies $f^{(m)}(x) = 0$. A small λ places less of a penalty on rapid change in $f^{(m)}(x)$, resulting in an estimate that tends to interpolate the data points.

The smoothing parameter greatly affects the analysis, and it should be selected with care. One method is to perform several analyses with different values for λ and compare the resulting final estimates.

A more objective way to select the smoothing parameter λ is to use the “leave-out-one” cross validation function, which is an approximation of the predicted mean squares error. A generalized version of the leave-out-one cross validation function is proposed by Wahba (1990) and is easy to calculate. This Generalized Cross Validation (GCV) function ($V(\lambda)$) is defined as

$$V(\lambda) = \frac{(1/n) \|(\mathbf{I} - \mathbf{A}(\lambda))\mathbf{y}\|^2}{[(1/n)\text{tr}(\mathbf{I} - \mathbf{A}(\lambda))]^2}$$

The justification for using the GCV function to select λ relies on asymptotic theory. Thus, you cannot expect good results for very small sample sizes or when there is not enough information in the data to separate the information signal from the noise component. Simulation studies suggest that for independent and identically distributed Gaussian noise, you can obtain reliable estimates of λ for n greater than 25 or 30. Note that, even for large values of n (say $n \geq 50$), in extreme Monte Carlo simulations there may be a small percentage of unwarranted extreme estimates in which $\hat{\lambda} = 0$ or $\hat{\lambda} = \infty$ (Wahba 1983). Generally, if σ^2 is known to within an order of magnitude, the occasional extreme case can be readily identified. As n gets larger, the effect becomes weaker.

The GCV function is fairly robust against nonhomogeneity of variances and non-Gaussian errors (Villalobos and Wahba 1987). Andrews (1988) has provided favorable theoretical results when variances are unequal. However, this selection method is likely to give unsatisfactory results when the errors are highly correlated.

The GCV value may be suspect when λ is extremely small because computed values may become indistinguishable from zero. In practice, calculations with $\lambda = 0$ or λ near 0 can cause numerical instabilities resulting in an unsatisfactory solution. Simulation studies have shown that a λ with $\log_{10}(n\lambda) > -8$ is small enough that the final estimate based on this λ almost interpolates the data points. A GCV value based on a $\lambda \leq -8$ may not be accurate.

ODS Tables Produced by PROC TPSPLINE

PROC TPSPLINE assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in the following table. For more information on ODS, see Chapter 15, “Using the Output Delivery System.”

Table 64.1. ODS Tables Produced by PROC TPSPLINE

ODS Table Name	Description	Statement	Option
DataSummary	Data summary	PROC	default
FitSummary	Fit parameters and fit summary	PROC	default
FitStatistics	Model fit statistics	PROC	default
GCVFunction	GCV table	MODEL	LOGNLAMBDA, LAMBDA

By referring to the names of such tables, you can use the ODS OUTPUT statement to place one or more of these tables in output data sets.

For example, the following statements create an output data set named `FitStats` containing the `FitStatistics` table, an output data set named `DataInfo` containing the `DataSummary` table, an output data set named `ModelInfo` containing the `FitSummary` and an output data set named `GCVFunc` containing the `GCVFunction`.

```
proc tpspline data=Melanoma;
  model Incidences=Year /LOGNLAMBDA=(-4 to 0 by 0.2);
  ods output FitStatistics = FitStats
             DataSummary   = DataInfo
             FitSummary    = ModelInfo
             GCVFunction   = GCVFunc;
run;
```

Examples

Example 64.1. Partial Spline Model Fit

The following example analyzes the data set `Measure` that was introduced in the “Getting Started” section on page 3325. That analysis determined that the final estimated surface can be represented by a quadratic function for one or both of the independent variables. This example illustrates how you can use PROC TPSPLINE to fit a partial spline model. The data set `Measure` is fit using the following model:

$$f(x_1, x_2) = 1 + x_1 + x_1^2 + h(x_2)$$

The model has a parametric component (associated with the x_1 variable) and a non-parametric component (associated with the x_2 variable). The following statements fit a partial spline model.

```

data Measure;
  set Measure;
  xlsq = x1*x1;
run;

data pred;
  do x1=-1 to 1 by 0.1;
    do x2=-1 to 1 by 0.1;
      xlsq = x1*x1;
      output;
    end;
  end;
run;

proc tpspline data= measure;
  model y = x1 xlsq (x2);
  score data = pred
        out = predy;
run;

```

Output 64.1.1 displays the results from these statements.

Output 64.1.1. Output from PROC TPSPLINE

The TPSPLINE Procedure	
Dependent Variable: y	
Summary of Input Data Set	
Number of Non-Missing Observations	50
Number of Missing Observations	0
Unique Smoothing Design Points	5
Summary of Final Model	
Number of Regression Variables	2
Number of Smoothing Variables	1
Order of Derivative in the Penalty	2
Dimension of Polynomial Space	4
Summary Statistics of Final Estimation	
log10(n*Lambda)	-2.237410
Smoothing Penalty	205.346097
Residual SS	8.582131
Tr(I-A)	43.153394
Model DF	6.846606
Standard Deviation	0.445954

As displayed in Output 64.1.1, there are five unique design points for the smoothing variable x_2 and two regression variables in the model (x_1, x_1sq). The dimension of the null space (polynomial space) is 4. The standard deviation of the estimate is much larger than the one based on the model with both x_1 and x_2 as smoothing variables (0.445954 compared to 0.098421). One of the many possible explanations may be that the number of unique design points of the smoothing variable is too small to warrant an accurate estimate for $h(x_2)$.

The following statements produce a surface plot for the partial spline model:

```

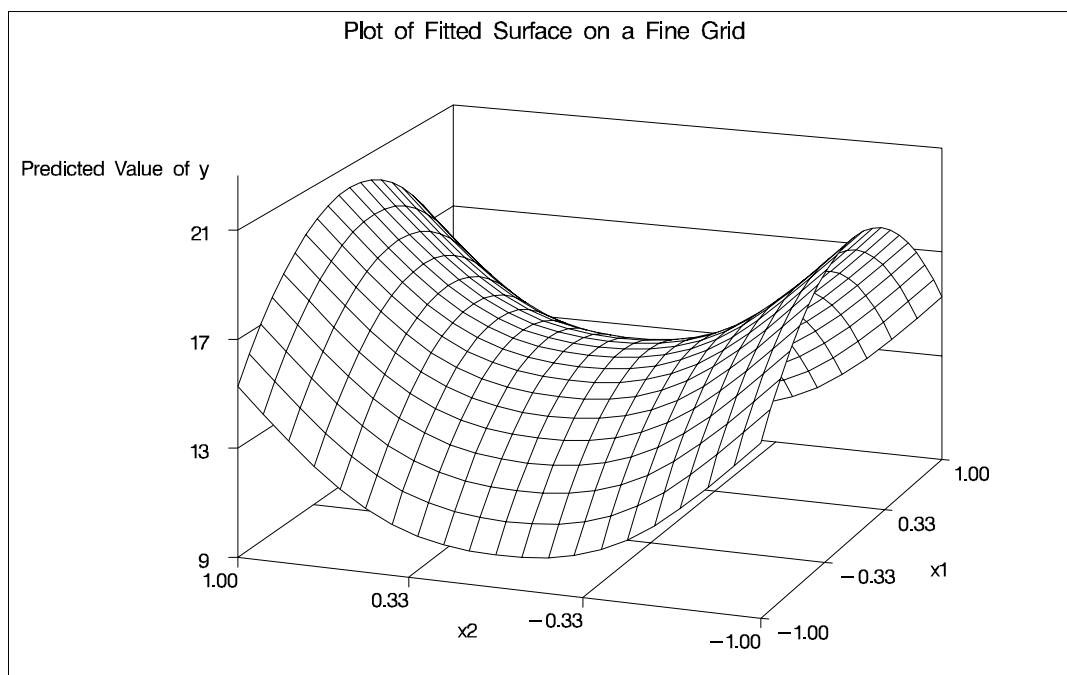
title 'Plot of Fitted Surface on a Fine Grid';

proc g3d data=predy;
  plot x2*x1=p_y/grid
      zmin=9
      zmax=21
      zticknum=4;
run;

```

The surface displayed in Output 64.1.2 is similar to the one estimated by using the full nonparametric model (displayed in Figure 64.5).

Output 64.1.2. Plot of TPSPLINE Fit from the Partial Spline Model



Example 64.2. Spline Model With Higher-Order Penalty

The following example continues the analysis of the data set **Measure** to illustrate how you can use PROC TPSPLINE to fit a spline model with a higher-order penalty term. Spline models with high-order penalty terms move low-order polynomial terms into the null space. Hence, there is no penalty for these terms, and they can vary without constraint.

As shown in the previous analyses, the final model for the data set `Measure` must include quadratic terms for both `x1` and `x2`. This example fits the following model:

$$f(x_1, x_2) = \theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_2 + \theta_4 x_2^2 + \theta_5 x_1 * x_2 + g(x_1, x_2)$$

The model includes quadratic terms for both variables, although it differs from the usual linear model. The nonparametric term $g(x_1, x_2)$ explains the variation of the data unaccounted for by a simple quadratic surface.

To modify the order of the derivative in the penalty term, specify the `M=` option. The following statements specify the option `M=3` in order to include the quadratic terms in the null space:

```
data measure;
  set measure;
  x1sq = x1*x1;
  x2sq = x2*x2;
  x1x2 = x1*x2;
;

proc tpspline data= measure;
  model y = (x1 x2) /m=3;
  score data = pred
          out = predy;
run;
```

The output resulting from these statements is displayed in Output 64.2.1.

Output 64.2.1. Output from PROC TPSPLINE with M=3

The TPSPLINE Procedure	
Dependent Variable: y	
Summary of Input Data Set	
Number of Non-Missing Observations	50
Number of Missing Observations	0
Unique Smoothing Design Points	25
Summary of Final Model	
Number of Regression Variables	0
Number of Smoothing Variables	2
Order of Derivative in the Penalty	3
Dimension of Polynomial Space	6
Summary Statistics of Final Estimation	
log10(n*Lambda)	-3.783100
Smoothing Penalty	2092.449492
Residual SS	0.273146
Tr(I-A)	29.171609
Model DF	20.828391
Standard Deviation	0.096765

The model contains six terms in the null space. Compare Output 64.2.1 with Figure 64.2: the LOGNLAMBDA value and the smoothing penalty differ significantly. Note that, in general, these terms are not directly comparable for different models. The final estimate based on this model is close to the estimate based on the model using the default, M=2.

In the following statements, the REG procedure fits a quadratic surface model to the data set Measure.

```
proc reg data= measure;
  model y = x1 x1sq x2 x2sq x1x2;
run;
```

The results are displayed in Output 64.2.2.

Output 64.2.2. Quadratic Surface Model: The REG Procedure

The REG Procedure					
Model: MODEL1					
Dependent Variable: y					
Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	5	443.20502	88.64100	436.33	<.0001
Error	44	8.93874	0.20315		
Corrected Total	49	452.14376			
	Root MSE	0.45073	R-Square	0.9802	
	Dependent Mean	15.08548	Adj R-Sq	0.9780	
	Coeff Var	2.98781			
Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > t
Intercept	1	14.90834	0.12519	119.09	<.0001
x1	1	0.01292	0.09015	0.14	0.8867
x1sq	1	-4.85194	0.15237	-31.84	<.0001
x2	1	0.02618	0.09015	0.29	0.7729
x2sq	1	5.20624	0.15237	34.17	<.0001
x1x2	1	-0.04814	0.12748	-0.38	0.7076

The REG procedure produces slightly different results. To fit a similar model with PROC TPSPLINE, you can use a MODEL statement specifying the degrees of freedom with the DF= option. You can also use a large value for the LOGNLAMBDA0= option to force a parametric model fit.

Because there is one degree of freedom for each of the following terms, Intercept, x1, x2, x1sq, x2sq, and x1x2, the DF=6 option is used.

```

proc TPSPLINE data=measure;
  model y=(x1 x2) /m=3 df=6 lognlambda=(-4 to 1 by 0.2);
  score data = pred
        out = predy;
run;

```

The results are displayed in Output 64.2.3. PROC TPSPLINE displays the list of GCV values for comparison.

Output 64.2.3. Output from PROC TPSPLINE Using M=3 and DF=6

The TPSPLINE Procedure	
Dependent Variable: y	
Summary of Input Data Set	
Number of Non-Missing Observations	50
Number of Missing Observations	0
Unique Smoothing Design Points	25
Summary of Final Model	
Number of Regression Variables	0
Number of Smoothing Variables	2
Order of Derivative in the Penalty	3
Dimension of Polynomial Space	6
GCV Function	
log10(n*Lambda)	GCV
-4.000000	0.016330
-3.800000	0.016051*
-3.600000	0.016363
-3.400000	0.017770
-3.200000	0.021071
-3.000000	0.027496
-2.800000	0.038707
-2.600000	0.056292
-2.400000	0.080613
-2.200000	0.109714
-2.000000	0.139642
-1.800000	0.166338
-1.600000	0.187437
-1.400000	0.202625
-1.200000	0.212871
-1.000000	0.219512
-0.800000	0.223727
-0.600000	0.226377
-0.400000	0.228041
-0.200000	0.229085
0	0.229740
0.200000	0.230153
0.400000	0.230413
0.600000	0.230576
0.800000	0.230680
1.000000	0.230745
Note: * indicates minimum GCV value.	

The TPSPLINE Procedure	
Dependent Variable: y	
Summary Statistics of Final Estimation	
log10(n*Lambda)	2.382955
Smoothing Penalty	0.000000646
Residual SS	8.938429
Tr(I-A)	43.999670
Model DF	6.000330
Standard Deviation	0.450719

The final estimate is based on 6.000330 degrees of freedom because there are already 6 degrees of freedom in the null space and the search range for lambda is not large enough (in this case, setting DF=6 is equivalent to setting lambda = ∞).

The standard deviation and RSS (Output 64.2.3) are close to the sum of squares for the error term and the root MSE from the the linear regression model (Output 64.2.2), respectively.

For this model, the optimal LOGNLAMBDA is around -3.8 , which produces a standard deviation estimate of 0.096765 (see Output 64.2.1) and a GCV value of 0.016051, while the model specifying DF=6 results in a LOGNLAMBDA larger than 1 and a GCV value larger than 0.23074. The nonparametric model, based on the GCV, should provide better prediction, but the linear regression model can be more easily interpreted.

Example 64.3. Multiple Minima of the GCV Function

The following data represent the deposition of sulfate (SO_4) at 179 sites in 48 contiguous states of the United States in 1990. Each observation records the latitude and longitude of the site as well as the SO_4 deposition at the site measured in gram per square meter (g/m^2).

You can use PROC TPSPLINE to fit a surface that reflects the general trend and that reveals underlying features of the data.

```

data so4;
  input latitude longitude so4 @@;
  datalines;
32.45833 87.24222 1.403 34.28778 85.96889 2.103
33.07139 109.86472 0.299 36.07167 112.15500 0.304
31.95056 112.80000 0.263 33.60500 92.09722 1.950
34.17944 93.09861 2.168 36.08389 92.58694 1.578
.
.
.
162 additional observations
.
.
.
45.82278 91.87444 0.984 41.34028 106.19083 0.335
42.73389 108.85000 0.236 42.49472 108.82917 0.313
42.92889 109.78667 0.182 43.22278 109.99111 0.161
43.87333 104.19222 0.306 44.91722 110.42028 0.210
45.07611 72.67556 2.646
;

data pred;
  do latitude = 25 to 47 by 1;
    do longitude = 68 to 124 by 1;
      output;
    end;
  end;
run;

```

The preceding statements create the SAS data set `so4` and the data set `pred` in order to make predictions on a regular grid. The following statements fit a surface for SO_4 deposition. The ODS OUTPUT statement creates a data set called `GCV` to contain the GCV values for `LOGNLAMBDA` in the range from -6 to 1 .

```

proc tpspline data=so4;
  ods output GCVFunction=gcv;
  model so4 = (latitude longitude) /lognlambda=(-6 to 1 by 0.1);
  score data=pred out=prediction1;
run;

```

Partial output from these statements is displayed in Output 64.3.1.

Output 64.3.1. Partial Output from PROC TPSPLINE for Data Set SO4

The TPSPLINE Procedure	
Dependent Variable: so4	
Summary of Input Data Set	
Number of Non-Missing Observations	179
Number of Missing Observations	0
Unique Smoothing Design Points	179
Summary of Final Model	
Number of Regression Variables	0
Number of Smoothing Variables	2
Order of Derivative in the Penalty	2
Dimension of Polynomial Space	3
Summary Statistics of Final Estimation	
log10(n*Lambda)	0.277005
Smoothing Penalty	2.458790
Residual SS	12.444975
Tr(I-A)	140.274968
Model DF	38.725032
Standard Deviation	0.297856

The following statements produce Output 64.3.2:

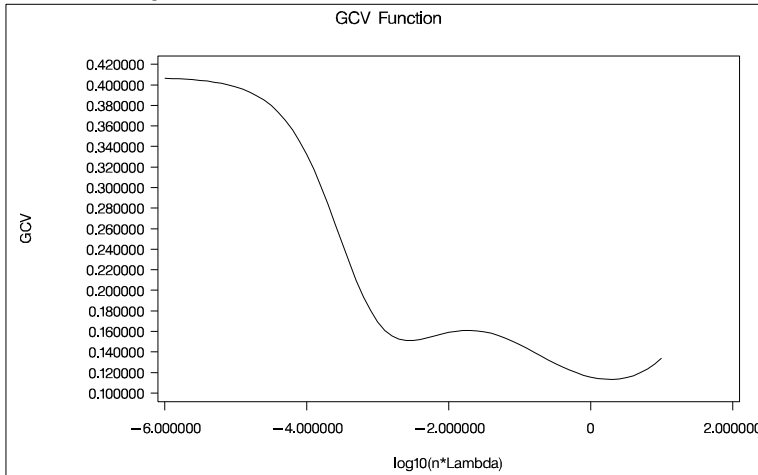
```

symbol1 interpol=join value=none;
title "GCV Function";

proc gplot data=gcv;
  plot gcv*lognlambd/frame cframe=ligr
      vaxis=axis1 haxis=axis2;
run;

```

Output 64.3.2 displays the plot of the GCV function versus nlambd in \log_{10} scale. The GCV function has two minima. PROC TPSPLINE locates the minimum at 0.277005. The figure also displays a local minimum located around -2.56 . Note that the TPSPLINE procedure may not always find the global minimum, although it did in this case.

Output 64.3.2. GCV Function of SO4 Data Set

The following analysis specifies the option `LOGNLAMBDA0=-2.56`. The output is displayed in Output 64.3.3.

```
proc tpspline data=so4;
  model so4 = (latitude longitude) /lognlambda0=-2.56;
  score data=pred out=prediction2;
run;
```

Output 64.3.3. Output from PROC TPSPLINE for Data Set SO4 with `LOGNLAMBDA=-2.56`

```

The TPSPLINE Procedure
Dependent Variable: so4

Summary of Input Data Set

Number of Non-Missing Observations    179
Number of Missing Observations       0
Unique Smoothing Design Points       179

Summary of Final Model

Number of Regression Variables        0
Number of Smoothing Variables        2
Order of Derivative in the Penalty   2
Dimension of Polynomial Space        3

Summary Statistics
of Final Estimation

log10(n*Lambda)          -2.560000
Smoothing Penalty        177.214368
Residual SS              0.043790
Tr(I-A)                  7.208638
Model DF                  171.791362
Standard Deviation       0.077940
```

The smoothing penalty is much smaller in Output 64.3.3 than that displayed in Output 64.3.1. The estimate in Output 64.3.1 uses a large lambda value and, therefore, the surface is smoother than the estimate using `LOGNLAMBDA=-2.56` (Output 64.3.3).

The estimate based on LOGNLAMBDA=-2.56 has a larger value for the degrees of freedom, and it has a much smaller standard deviation.

However, a smaller standard deviation in nonparametric regression does not necessarily mean that the estimate is good: a small λ value always produces an estimate closer to the data and, therefore, a smaller standard deviation.

The following statements produce two contour plots of the estimates using the GCONTOUR procedure. In the final step, the plots are placed into a single graphic with the GREPLAY procedure.

```

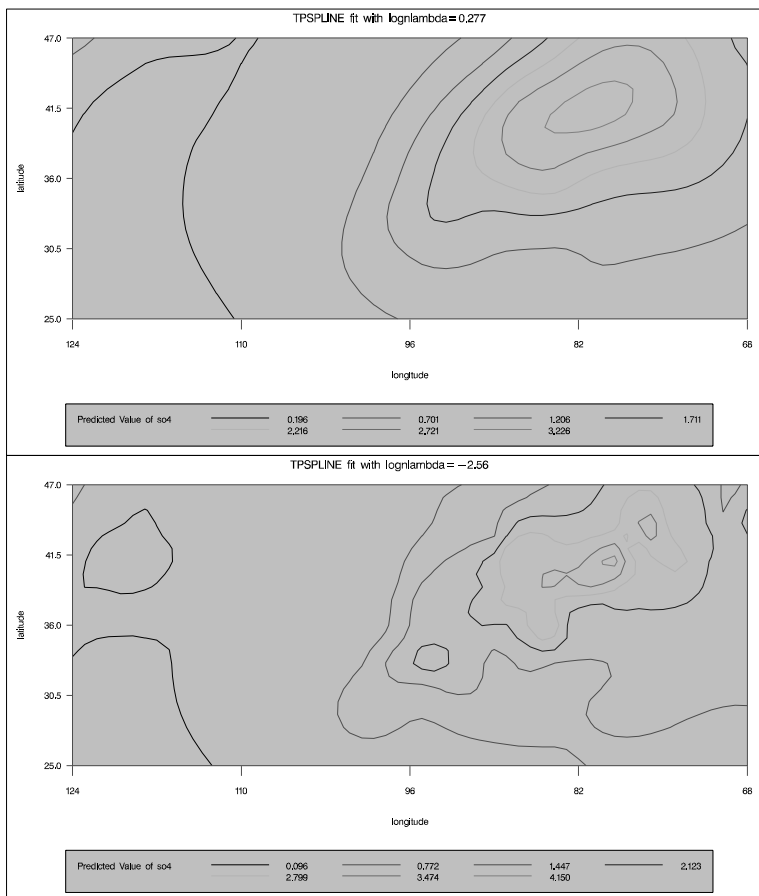
title "TPSPLINE fit with lognlambda=0.277";
proc gcontour data=prediction1 gout=grafcat;
  plot latitude*longitude = P_so4/
         name="tpscon1" legend=legend1
         vaxis=axis1 haxis=axis2
         cframe=ligr hreverse;
run;

title "TPSPLINE fit with lognlambda=-2.56";
proc gcontour data=prediction2 gout=grafcat;
  plot latitude*longitude = P_so4/
         name="tpscon2" legend=legend1
         vaxis=axis1 haxis=axis2
         cframe=ligr hreverse;
run;

title;
proc greplay igout=grafcat tc=sashelp.templt template=v2 nofs;
  treplay 1:tpscon1
         2:tpscon2;
quit;
run;

```

Compare the two estimates by examining the contour plots of both estimates (Output 64.3.4).

Output 64.3.4. Contour Plot of TPSPLINE Estimates with Different Lambdas

As the contour plots show, the estimate with `LOGNLAMBDA=0.277` may represent the underlying trend, while the estimate with the `LOGNLAMBDA=-2.56` is very rough and may be modeling the noise component.

Example 64.4. Large Data Set Application

The following example illustrates how you can use the `D=` option to decrease the computation time needed by the TPSPLINE procedure. Note that, while the `D=` option can be helpful in decreasing computation time for large data sets, it may produce unexpected results when used with small data sets.

The following statements generate the data set `large`:

```
data large;
  do x=-5 to 5 by 0.02;
    y=5*sin(3*x)+1*rannor(57391);
    output;
  end;
run;
```


The data set `large` contains 501 observations with one independent variable `x` and one dependent variable `y`. The following statements invoke PROC TPSPLINE to produce a thin-plate smoothing spline estimate and the associated 99% confidence interval. The output statistics are saved in the data set `fit1`.

```
proc tpspline data=large;
  model y =(x) /lambda=(-5 to -1 by 0.2) alpha=0.01;
  output out=fit1 pred LCLM UCLM;
run;
```

The results from this MODEL statement are displayed in Output 64.4.1.

Output 64.4.1. Output from PROC TPSPLINE without the D= Option

The TPSPLINE Procedure	
Dependent Variable: y	
Summary of Input Data Set	
Number of Non-Missing Observations	501
Number of Missing Observations	0
Unique Smoothing Design Points	501
Summary of Final Model	
Number of Regression Variables	0
Number of Smoothing Variables	1
Order of Derivative in the Penalty	2
Dimension of Polynomial Space	2
GCV Function	
log10(n*Lambda)	GCV
-5.000000	1.258653
-4.800000	1.228743
-4.600000	1.205835
-4.400000	1.188371
-4.200000	1.174644
-4.000000	1.163102
-3.800000	1.152627
-3.600000	1.142590
-3.400000	1.132700
-3.200000	1.122789
-3.000000	1.112755
-2.800000	1.102642
-2.600000	1.092769
-2.400000	1.083779
-2.200000	1.076636
-2.000000	1.072763*
-1.800000	1.074636
-1.600000	1.087152
-1.400000	1.120339
-1.200000	1.194023
-1.000000	1.344213
Note: * indicates minimum GCV value.	

The TPSPLINE Procedure	
Dependent Variable: y	
Summary Statistics of Final Estimation	
log10(n*Lambda)	-1.948303
Smoothing Penalty	9953.706749
Residual SS	475.098382
Tr(I-A)	471.086071
Model DF	29.913929
Standard Deviation	1.004250

The following statements specify an identical model, but with the additional specification of the D= option. The estimates are obtained by treating nearby points as replicates.

```
proc tpspline data=large;
  model y =(x) /lambda=(-5 to -1 by 0.2) d=0.05 alpha=0.01;
  output out=fit2 pred LCLM UCLM;
run;
```

The output is displayed in Output 64.4.2.

Output 64.4.2. Output from PROC TPSPLINE with the D= Option

```

The TPSPLINE Procedure
Dependent Variable: y

Summary of Input Data Set

Number of Non-Missing Observations    501
Number of Missing Observations       0
Unique Smoothing Design Points       251

Summary of Final Model

Number of Regression Variables         0
Number of Smoothing Variables         1
Order of Derivative in the Penalty    2
Dimension of Polynomial Space         2

GCV Function

log10(n*Lambda)      GCV

-5.000000            1.306536
-4.800000            1.261692
-4.600000            1.226881
-4.400000            1.200060
-4.200000            1.179284
-4.000000            1.162776
-3.800000            1.149072
-3.600000            1.137120
-3.400000            1.126220
-3.200000            1.115884
-3.000000            1.105766
-2.800000            1.095730
-2.600000            1.085972
-2.400000            1.077066
-2.200000            1.069954
-2.000000            1.066076*
-1.800000            1.067929
-1.600000            1.080419
-1.400000            1.113564
-1.200000            1.187172
-1.000000            1.337252

Note: * indicates minimum GCV value.

```

```

The TPSPLINE Procedure
Dependent Variable: y

Summary Statistics
of Final Estimation

log10(n*Lambda)      -1.947711
Smoothing Penalty    9943.561350
Residual SS          472.142409
Tr(I-A)              471.090128
Model DF             29.909872
Standard Deviation   1.001116

```

The difference between the two estimates is minimal. However, the CPU time for the second MODEL statement is only about 1/8 of the CPU time used in the first model fit.

The following statements produce a plot for comparison of the two estimates:

```

data fit2;
  set fit2;
  P1_y      = P_y;
  LCLM1_y  = LCLM_y;
  UCLM1_y  = UCLM_y;
  drop P_y
         LCLM_y
         UCLM_y;

proc sort data=fit1;
  by x y;
proc sort data=fit2;
  by x y;

data comp;
  merge fit1 fit2;
  by x y;
  label p1_y      ="Yhat1" p_y="Yhat0"
        lclm_y   ="Lower CL"
        uclm_y   ="Upper CL";

symbol1  i=join v=none ;
symbol2  i=join v=none ;
symbol3  i=join v=none color=cyan;
symbol4  i=join v=none color=cyan;

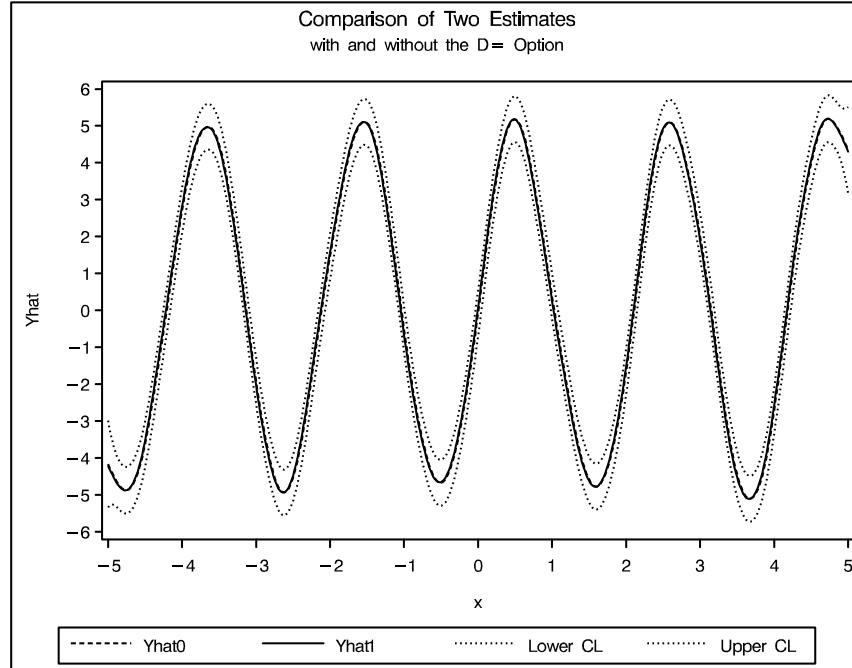
title 'Comparison of Two Estimates';
title2 'with and without the D= Option';

proc gplot data=comp;
  plot P_y*x=1
        P1_y*x=2
        LCLM_y*x=4
        UCLM_y*x=4/overlay      legend=legend1
                                vaxis=axis1 haxis=axis2
                                frame      cframe=ligr;
run;

```

The estimates fit1 and fit2 are displayed in Output 64.4.3 with the 99% confidence interval from the fit1 output data set.

Output 64.4.3. Comparison of Two Fits with and without the D= Option



Example 64.5. Computing a Bootstrap Confidence Interval

The following example illustrates how you can construct a bootstrap confidence interval by using the multiple responses feature in PROC TPSPLINE.

Numerous epidemiological observations have indicated that exposure to solar radiation is an important factor in the etiology of melanoma. The following data present age-adjusted melanoma incidences for 37 years from the Connecticut Tumor Registry (Houghton, Flannery, and Viola 1980). The data are analyzed by Ramsay and Silverman (1997).

```

data melanoma;
  input year incidences @@;
  datalines;
  1936 0.9 1937 0.8 1938 0.8 1939 1.3
  1940 1.4 1941 1.2 1942 1.7 1943 1.8
  1944 1.6 1945 1.5 1946 1.5 1947 2.0
  1948 2.5 1949 2.7 1950 2.9 1951 2.5
  1952 3.1 1953 2.4 1954 2.2 1955 2.9
  1956 2.5 1957 2.6 1958 3.2 1959 3.8
  1960 4.2 1961 3.9 1962 3.7 1963 3.3
  1964 3.7 1965 3.9 1966 4.1 1967 3.8
  1968 4.7 1969 4.4 1970 4.8 1971 4.8
  1972 4.8
  ;
run;

```

The variable `incidences` records the number of melanoma cases per 100,000 people for the years 1936 to 1972. The following model fits the data and requests a 90% Bayesian confidence interval along with the estimate.

```
proc tpspline data=melanoma;
  model incidences = (year) /alpha = 0.1;
  output out = result pred uclm lclm;
run;
```

The output is displayed in Output 64.5.1

Output 64.5.1. Output from PROC TPSPLINE for the Melanoma Data Set

The TPSPLINE Procedure	
Dependent Variable: incidences	
Summary of Input Data Set	
Number of Non-Missing Observations	37
Number of Missing Observations	0
Unique Smoothing Design Points	37
Summary of Final Model	
Number of Regression Variables	0
Number of Smoothing Variables	1
Order of Derivative in the Penalty	2
Dimension of Polynomial Space	2
Summary Statistics of Final Estimation	
log10(n*Lambda)	-0.060735
Smoothing Penalty	0.517132
Residual SS	1.224264
Tr(I-A)	22.585173
Model DF	14.414827
Standard Deviation	0.232823

The following statements produce a plot of the estimated curve:

```
symbol1 h=1pct ;
symbol2 i=join v=none;
symbol3 i=join v=none;
symbol4 i=join v=none c=cyan;

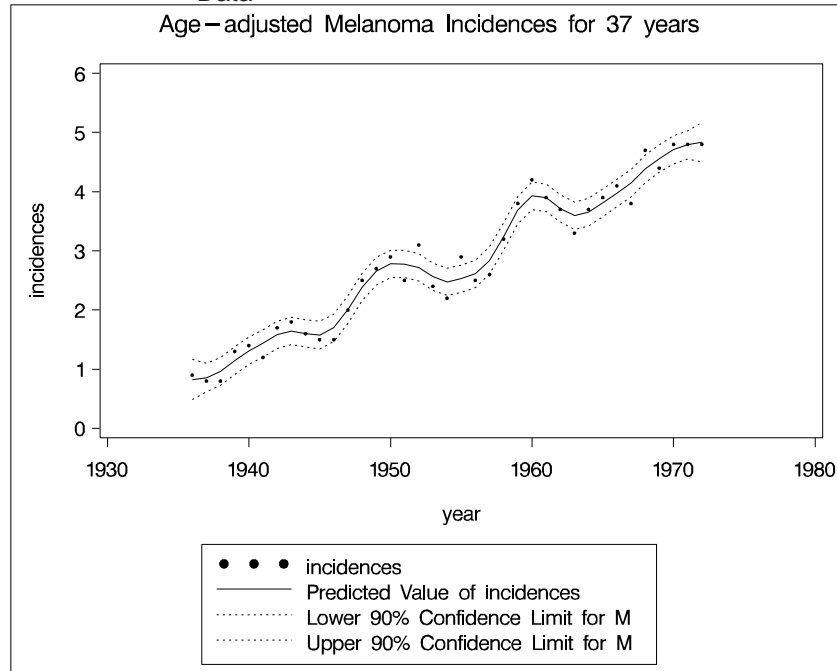
legend1 frame cframe=ligr cborder=black
  label=none position=center;
axis1 label=(angle=90 rotate=0) minor=none;
axis2 minor=none;

title1 'Age-adjusted Melanoma Incidences for 37 years';

proc gplot data=result;
  plot incidences*year=1
  p_incidences*year=2
  lclm_incidences*year=3
  uclm_incidences*year=4 /overlay legend=legend1
  vaxis=axis1 haxis=axis2
  frame cframe=ligr;
run;
```

The estimated curve is displayed with 90% confidence interval bands in Output 64.5.2. The number of melanoma incidences exhibits a periodic pattern and increases over the years. The periodic pattern is related to sunspot activity and the accompanying fluctuations in solar radiation.

Output 64.5.2. TPSPLINE Estimate and 90% Confidence Interval of Melanoma Data



Wang and Wahba (1995) compared several bootstrap confidence intervals to Bayesian confidence intervals for smoothing splines. Both bootstrap and Bayesian confidence intervals are across-the-curve intervals, not point-wise intervals. They concluded that bootstrap confidence intervals work as well as Bayesian intervals concerning average coverage probability. Additionally, bootstrap confidence intervals appear to be better for small sample sizes. Based on their simulation, the “percentile-*t* interval” bootstrap interval performs better than the other types of bootstrap intervals.

Suppose that \hat{f}_λ and $\hat{\sigma}$ are the estimates of f and σ from the data. Assume that \hat{f}_λ is the “true” f , and generate the bootstrap sample as follows:

$$y_i^* = \hat{f}_\lambda(x_i) + \epsilon_i^*, \quad i = 1, \dots, n$$

where $\epsilon^* = (\epsilon_1^*, \dots, \epsilon_n^*)^T \approx \mathbf{N}(\mathbf{0}, \hat{\sigma} \mathbf{I}_{n \times n})$. Denote $f_\lambda^*(x_i)$ as the random variable of the bootstrap estimate at x_i . Repeat this process K times, so that at each point x_i , you have K bootstrap estimates $\hat{f}_\lambda^*(x_i)$ or K realizations of $f_\lambda^*(x_i)$. For each fixed x_i , consider the following statistic D_i^* , which is similar to a Student’s t statistic:

$$D_i^* = (f_\lambda^*(x_i) - \hat{f}_\lambda(x_i)) / \hat{\sigma}_i^*$$

where $\hat{\sigma}_i^*$ is the estimate of $\hat{\sigma}$ based on the i th bootstrap sample.

Suppose $\chi_{\alpha/2}$ and $\chi_{1-\alpha/2}$ are the lower and upper $\alpha/2$ points of the empirical distribution of D_i^* . The $(1 - \alpha)100\%$ bootstrap confidence interval is defined as

$$(\hat{f}_{\hat{\lambda}}(x_i) - \chi_{1-\alpha/2}\hat{\sigma}, \hat{f}_{\hat{\lambda}}(x_i) - \chi_{\alpha/2}\hat{\sigma})$$

Bootstrap confidence intervals are easy to interpret and can be used with any distribution. However, because they require K model fits, their construction is computationally intensive.

The multiple dependent variables feature in PROC TPSPLINE enables you to fit multiple models with the same independent variables. The procedure calculates the matrix decomposition part of the calculations only once regardless of the number of dependent variables in the model. These calculations are responsible for most of the computing time used by the TPSPLINE procedure. This feature is particularly useful when you need to generate a bootstrap confidence interval.

To construct a bootstrap confidence interval, perform the following tasks:

- Fit the data using PROC TPSPLINE and obtain estimates $\hat{f}_{\hat{\lambda}}(x_i)$ and $\hat{\sigma}$.
- Generate K bootstrap samples based on $\hat{f}_{\hat{\lambda}}(x_i)$ and $\hat{\sigma}$.
- Fit the K bootstrap samples with the TPSPLINE procedure to obtain estimates of $\hat{f}_{\hat{\lambda}}^*(x_i)$ and $\hat{\sigma}_i^*$.
- Compute D_i^* and the values $\chi_{\alpha/2}$ and $\chi_{1-\alpha/2}$.

The following statements illustrate this process:

```
proc tpspline data=melanoma;
  model incidences = (year) /alpha = 0.05;
  output out = result pred uclm lclm;
run;
```

The output from the initial PROC TPSPLINE analysis is displayed in Output 64.5.3. The data set **result** contains the predicted values and confidence limits from the analysis.

Output 64.5.3. Output from PROC TPSPLINE for the Melanoma Data Set

The TPSPLINE Procedure	
Dependent Variable: incidences	
Summary of Input Data Set	
Number of Non-Missing Observations	37
Number of Missing Observations	0
Unique Smoothing Design Points	37
Summary of Final Model	
Number of Regression Variables	0
Number of Smoothing Variables	1
Order of Derivative in the Penalty	2
Dimension of Polynomial Space	2
Summary Statistics of Final Estimation	
log10(n*Lambda)	-0.060735
Smoothing Penalty	0.517132
Residual SS	1.224264
Tr(I-A)	22.585173
Model DF	14.414827
Standard Deviation	0.232823

The following statements illustrate how you can obtain a bootstrap confidence interval for the Melanoma data. The following statements create the data set `bootstrap`. The observations are created with information from the preceding PROC TPSPLINE execution; as displayed in Output 64.5.3, $\hat{\sigma} = 0.232823$. The values of $\hat{f}_{\lambda}(x_i)$ are stored in the data set result in the variable `P_incidence`.

```

data bootstrap;
  set result;
  array y{1070} y1-y1070;
  do i=1 to 1070;
    y{i} = p_incidences + 0.232823*rannor(123456789);
  end;
  keep y1-y1070 p_incidences year;
run;

ods listing close;

proc tpspline data=bootstrap;
  ods output FitStatistics=FitResult;
  id p_incidences;
  model y1-y1070 = (year);
  output out=result2;
run;
ods listing;

```

The DATA step generates 1,070 bootstrap samples based on the previous estimate from PROC TPSPLINE. For this data set, some of the bootstrap samples result in λ

(selected by the GCV function) that cause problematic behavior. Thus, an additional 70 bootstrap samples are generated.

The ODS listing destination is closed before PROC TPSPLINE is invoked. The model fits all the y_1 – y_{1070} variables as dependent variables, and the models are fit for all bootstrap samples simultaneously. The output data set `result2` contains the variables `year`, y_1 – y_{1070} , p_{y_1} – $p_{y_{1070}}$, and `P_incidences`.

The ODS OUTPUT statement writes the FitStatistics table to the data set `FitResult`. The data set `FitResult` contains the two variables. They are `Parameter` and `Value`. The `FitResult` data set is used in subsequent calculations for D_i^* .

In the data set `FitResult`, there are 63 estimates with a standard deviation of zero, suggesting that the estimates provide perfect fits of the data and are caused by $\hat{\lambda}$ s that are approximately equal to zero. For small sample sizes, there is a positive probability that the λ chosen by the GCV function will be zero (refer to Wang and Wahba 1995).

In the following steps, these cases are removed from the bootstrap samples as “bad” samples: they represent failure of the GCV function.

The following SAS statements manipulate the data set `FitResult`, retaining the standard deviations for all bootstrap samples and merging `FitResult` with the data set `result2`, which contains the estimates for bootstrap samples. In the final data set `boot`, the D_i^* statistics are calculated.

```

data FitResult; set FitResult;
  if Parameter="Standard Deviation";
  keep Value;
run;

proc transpose data=sum2 out=sd prefix=sd;

data result2;
  if _N_ = 1 then set sd;
  set result2;

data boot;
  set result2;
  array y{1070} p_y1-p_y1070;
  array sd{1070} sd1-sd1070;
  do i=1 to 1070;
    if sd{i} > 0 then do;
      d = (y{i} - P_incidences)/sd{i};
      obs = _N_;
      output;
    end;
  end;
  keep d obs P_incidences year;
run;

```

The following SAS statements retain the first 1000 bootstrap samples and calculate the values $\chi_{\alpha/2}$ and $\chi_{1-\alpha/2}$ with $\alpha = 0.1$.

```

proc sort data=boot;
  by obs;
run;

data boot;
  set boot;
  by obs;
  retain n;

  if first.obs then n=1;
  else n=n+1;
  if n > 1000 then delete;
run;

proc sort data=boot;
  by obs error;
run;

data chil chi2 ;
  set boot;
  if (_N_ = (obs-1)*1000+50) then output chil;
  if (_N_ = (obs-1)*1000+950) then output chi2;
run;

proc sort data=result;
  by year;
run;

proc sort data=chil;
  by year;
run;

proc sort data=chi2;
  by year;
run;

data result;
  merge result
    chil(rename=(d=chi05))
    chi2(rename=(d=chi95));
  keep year incidences P_incidences lower upper
    LCLM_incidences UCLM_incidences;

  lower = -chi95*0.232823 + P_incidences;
  upper = -chi05*0.232823 + P_incidences;

  label lower="Lower 95% CL (Bootstrap)"
    upper="Upper 95% CL (Bootstrap)"
    lclm_incidences="Lower 95% CL (Bayesian)"
    uclm_incidences="Upper 95% CL (Bayesian)";
run;

```

The data set `result` contains the variables `year`, `incidences`, the TPSPLINE estimate `P_incidences`, and the 90% Bayesian and 90% bootstrap confidence intervals.

The following statements produce Output 64.5.4:

```

symbol1 v=dot h=1pct ;
symbol2 i=join v=none l=1;
symbol3 i=join v=none l=33;
symbol4 i=join v=none l=33;
symbol5 i=join v=none l=43 c=green;
symbol6 i=join v=none l=43 c=green;

title1 'Age-adjusted Melanoma Incidences for 37 years';
proc gplot data=result;
  plot      incidences * year=1
            p_incidences * year=2
            lclm_incidences * year=3
            uclm_incidences * year=3
            lower * year=4
            upper * year=4
  /overlay  legend=legend1
  vaxis=axis1 haxis=axis2
  frame     cframe=ligr;

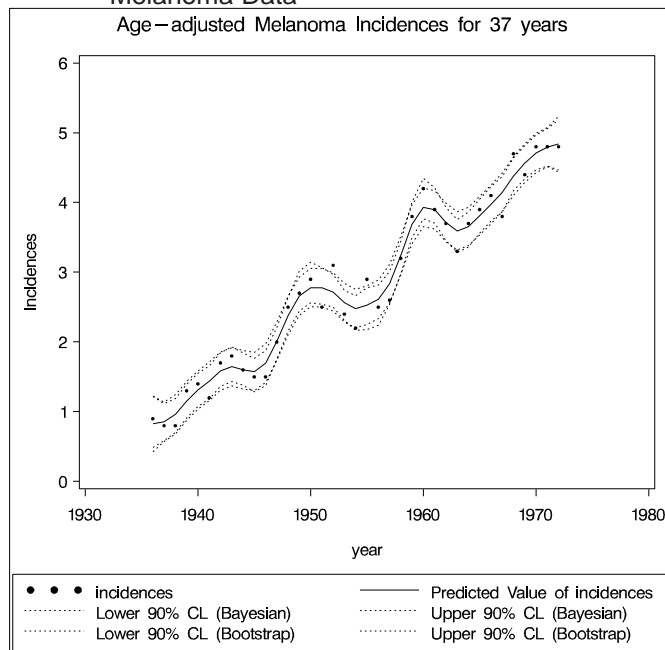
run;

```

Output 64.5.4 displays the plot of the variable `incidences`, the predicted values, and the Bayesian and bootstrap confidence intervals.

The plot shows that the bootstrap confidence interval is similar to the Bayesian confidence interval. However, the Bayesian confidence interval is symmetric around the estimates, while the bootstrap confidence interval is not.

Output 64.5.4. Comparison of Bayesian and Bootstrap Confidence Interval for Melanoma Data



References

- Andrews, D (1988), *Asymptotic Optimality of GC_L , Cross-Validation, and GCV in Regression with Heteroscedastic Errors*, Cowles Foundation, Yale University, New Haven, CT, manuscript.
- Bates, D.; Lindstrom, M.; Wahba, G. and Yandell, B. (1987), “GCVPACK-Routines for Generalized Cross Validation,” *Comm. Statist. B-Simulation Comput.*, 16, 263–297.
- Duchon, J. (1976), “Fonctions-Spline et Esperances Conditionnelles de Champs Gaussiens,” *Ann. Sci. Univ. Clermont Ferrand II Math.*, 14, 19–27.
- Duchon, J. (1977), “Splines Minimizing Rotation-Invariant Semi-Norms in Sobolev Spaces,” in *Constructive Theory of Functions of Several Variables*, eds. W. Schempp and K. Zeller, 85–100.
- Dongarra, J.; Bunch, J.; Moler, C. and Stewart, G. (1979), *Linpack Users’ Guide*, Philadelphia: Society for Industrial and Applied Mathematics.
- Hall, P. and Titterton, D. (1987), “Common Structure of Techniques for Choosing Smoothing Parameters in Regression Problems,” *J. Roy. Statist. Soc. Ser. B*, 49, 184–198.
- Houghton, A. N.; Flannery, J. and Viola, M. V. (1980), “Malignant Melanoma in Connecticut and Denmark,” *International Journal of Cancer*, 25, 95–104.
- Hutchinson, M. and Bischof, R. (1983), “A New Method for Estimating the Spatial Distribution of Mean Seasonal and Annual Rainfall Applied to the Hunter Valley,” *New South Wales, Aust. Met. Mag.*, 31, 179–184.
- Meinguet, J. (1979), “Multivariate Interpolation at Arbitrary Points Made Simple,” *J. Appl. Math. Phys. (ZAMP)*, 5, 439–468.
- Nychka, D (1986a), “The Average Posterior Variance of a Smoothing Spline and a Consistent Estimate of the Mean Square Error,” Tech. Report 168, The Institute of Statistics, North Carolina State University, Raleigh, NC.
- Nychka, D (1986b), “A Frequency Interpretation of Bayesian “Confidence” Interval for Smoothing Splines,” Tech. Report 169, The Institute of Statistics, North Carolina State University, Raleigh, NC.
- Nychka, D (1988), “Confidence Intervals for Smoothing Splines,” *J. Amer. Statist. Assoc.*, 83, 1134–1143.
- O’Sullivan, F. and Wong, T. (1987), “Determining a Function Diffusion Coefficient in the Heat Equation,” Tech. Report 98, Department of Statistics, University of California, Berkeley, CA.
- Ramsay, J. and Silverman, B (1997), *Functional Data Analysis*, New York: Springer-Verlag.
- Seaman, R. and Hutchinson, M. (1985), “Comparative Real Data Tests of Some Objective Analysis Methods by Withholding,” *Aust. Met. Mag.*, 33, 37–46.

- Villalobos, M. and Wahba, G. (1987), “Inequality Constrained Multivariate Smoothing Splines with Application to the Estimation of Posterior Probabilities,” *J. Amer. Statist. Assoc.*, 82 239–248.
- Wahba, G. (1983), “Bayesian “Confidence Intervals” for the Cross Validated Smoothing Spline,” *J. Roy. Statist. Soc. Ser. B*, 45, 133–150.
- Wahba, G., (1990), *Spline Models for Observational Data*, Philadelphia: Society for Industrial and Applied Mathematics.
- Wahba, G. and Wendelberger, J. (1980), “Some New Mathematical Methods for Variational Objective Analysis Using Splines and Cross Validation,” *Monthly Weather Rev.*, 108, 1122–1145.
- Wang, Y. and Wahba, G. (1995), “Bootstrap Confidence Intervals for Smoothing Splines and their Comparison to Bayesian Confidence Intervals,” *J. Statistical Computation and Simulation*, 51, 263–279.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS/STAT® User's Guide, Version 8*, Cary, NC: SAS Institute Inc., 1999.

SAS/STAT® User's Guide, Version 8

Copyright © 1999 by SAS Institute Inc., Cary, NC, USA.

ISBN 1-58025-494-2

All rights reserved. Produced in the United States of America. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, or otherwise, without the prior written permission of the publisher, SAS Institute Inc.

U.S. Government Restricted Rights Notice. Use, duplication, or disclosure of the software and related documentation by the U.S. government is subject to the Agreement with SAS Institute and the restrictions set forth in FAR 52.227-19 Commercial Computer Software-Restricted Rights (June 1987).

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

1st printing, October 1999

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries.® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.

The Institute is a private company devoted to the support and further development of its software and related services.