Modified: July 28, 2008

Multivariate Adaptive Regression Spline Modeling Using SCAB34S SPLINES and SCA WorkBench

Houston H. Stokes
Department of Economics
University of Illinois at Chicago

William J. Lattyak
Scientific Computing Associates Corp.

Table of Contents

1. MARSPLINE MODELING USING SCAB34S AND WORKBENCH 4

1.1 Interpreting the MARSPLINE Model Output Summaries 7
1.2 Nonlinear Testing 7
1.3 Confusion Matrix 8
1.4 Lift-Gain Table 10

2 SCA WORKBENCH: A GRAPHICAL USER INTERFACE 13

2.1 Model tab 13
2.2 Options tab 17
2.3 Validation tab 19
2.4 Results tab 22
2.5 Graphs tab 24

3. EXAMPLES OF MARSPLINE MODELING USING SCA WORKBENCH 27

3.1 Modeling Daily Electricity Load Using MARSPLINE 27
3.1.1 Specification of the MARSPLINE Model for Daily Electricity Load 29
3.1.2 MARSPLINE Model Options for the Daily Electricity Load Example 30
3.1.3 MARSPLINE Model Validation for the Daily Electricity Load Example 31
3.1.4 MARSPLINE Model Results for the Daily Electricity Load Example 33
3.2 Modeling Nitrous Oxide Production Using MARSPLINE 38
3.2.1 Specification of the MARSPLINE Model for Nitrous Oxide Production 39
3.2.2 MARSPLINE Model Options for Nitrous Oxide Production Example 40
3.2.3 MARSPLINE Model Validation for Nitrous Oxide Production Example 41
3.3 Modeling Capital Punishment as a Deterrent to Murder Using MARS-Probit 45
3.3.1 Specification of the MARS-Probit Model for the McManus Data 46
3.3.2 MARS-Probit Model Options for the McManus Data Example 46
3.3.3 MARS-Probit Model Validation for the McManus Data Example 47

4. DETAILED DESCRIPTION OF THE MARSPLINE ROUTINE IN SCAB34S 55

Usage: 55
Required subroutine arguments: 55
Optional keywords and associated arguments: 55
Variables created in MARSPLINE subroutine call: 57

5. A DETAILED DESCRIPTION OF THE OLSQ COMMAND 61

Usage: 61
Required subroutine arguments: 61
Optional keywords and associated arguments: 61
Variables created specific to OLSQ OPTIONS specified: 62

6. A DETAILED DESCRIPTION OF THE PROBIT COMMAND 65

Usage: 65
Required subroutine arguments: 65
Optional keywords and associated arguments: 65
Variables created specific to PROBIT OPTIONS specified: 66

7. A DETAILED DESCRIPTION OF CUSTOMIZABLE SUBROUTINES USED IN THE MARSPLINE APPLICATION 67

7.1 P_L_EST User Subroutine 67
Usage: 67
Required subroutine arguments: 67
Example: 68
7.2 DISP_LGT User Subroutine 69

Usage: *69*
Required subroutine arguments: *69*
Example: *69*
7.3 DISP_OLS User Subroutine **70**
Usage: *70*
Required subroutine arguments: *70*
Example: *71*
7.4 DSP_TBL User Subroutine **71**
Usage: *71*
Required subroutine arguments: *71*
Example: *71*
7.5 TLGTR User Subroutine **72**
Usage: *72*
Required subroutine arguments: *73*
Example: *74*
7.6 TLOGIT User Subroutine **75**
Usage: *75*
Required subroutine arguments: *75*
Example: *76*
7.7 LIFTGAIN User Subroutine **76**
10 10 0 0.0% 0.0% 100 25 25.0% 100.0% 0.0% 100 0.0% *76*
Usage: *76*
Required subroutine arguments: *77*
Example: *77*
7.8 GRFLIFT User Subroutine **77**
Usage: *78*
Required subroutine arguments: *78*
Example: *78*
7.9 DISP_HIN User Subroutine **79**
Usage: *79*
Required subroutine arguments: *79*
Example: *79*
7.10 CONTRIB User Subroutine **79**
The CONTRIB subroutine creates contribution charts for all explanatory variables. The charts are saved in Windows-Meta-File (WMF) format. The number of charts created is dependent upon the number of explanatory variables in the final model and the number of interactions in combination with the target. An example of a contribution chart is provided below: *80*
Usage: *80*
Required subroutine arguments: *80*
Example: *81*

8. EXAMPLES OF SCAB34S COMMAND FILES FOR MARSPLINE MODELING **82**

8.1 SCAB34S Commands Generated for the Electricity Load Example **82**
8.2 SCAB34S Commands for the Nitrous Oxide Example **85**
8.3 SCAB34S Commands Generated for the MARS-Probit Example **87**
8.4 SCAB34S Commands Used to Display Graphs in the Examples **90**

REFERENCES **92**

Multivariate Adaptive Regression Spline Modeling Using SCAB34S SPLINES and SCA WorkBench

In this document, we discuss *Multivariate Adaptive Regression Spline* (MARSPLINE) modeling and estimation provided by the B34S® ProSeries Econometric System and SCAB34S SPLINES software products. We also discuss the SCA WorkBench companion product and its user interface to shell a MARSPLINE modeling and validation environment in the B34S program suite. The use of the MARS acronym in this document is solely intended as a reference to the methodology of multivariate adaptive regression splines.

SCAB34S SPLINES provides a subset of the capabilities in the B34S® ProSeries Econometric System and we refer to these products interchangeably within this document. SCAB34S SPLINES runs conveniently as an integrated component to SCA WorkBench. The WorkBench product is a companion to the SCA Statistical System and SCAB34S software, providing a graphical user interface for several applications including MARSPLINE modeling for cross sectional data, MARS-Probit modeling for ordered data, and TSMARS (or ASTAR) for time series data. Within the context of MARSPLINE and TSMARS model validation, the predictive performance of these models may be validated by comparing the in-sample and out-of-sample predictive values to linear regression models using OLS, MINIMAX, or L1 estimation methods. Within the context of MARS-Probit model validation, the predictive performance of these models may be validated by viewing the Confusion Matrices and Lift-Gains between the MARS-Probit model and an OLS, Probit, or Logistic model.

The SCAB34S SPLINES product provides a number of procedures to perform common data manipulation tasks, organizational tasks, and statistical/econometric analysis tasks. It also contains a comprehensive matrix programming language that may be used to customize procedures for specialized use. No attempt will be made to cover all features of the SCAB34S product in this document nor the full range of applications that may be solved using the B34S matrix programming facilities.[1] Instead, we shall exclusively use the graphical user interface of SCA WorkBench to specify, estimate, and diagnostically test MARSPLINE and TSMARS models in SCAB34S SPLINES. SCA WorkBench automatically specifies the command script executed in the SCAB34S SPLINES product based on menu selections. A command file is then executed in the SCAB34S engine and the results are read back into WorkBench for examination. The user may save the program file and modify the command script to address additional analysis requirements that may arise.

A major assumption of any linear process is that the coefficients are stable across all levels of the explanatory variables and, in the case of a time series model, across all time periods. The MARSPLINE model is a very useful method of analysis when it is suspected that the model's coefficients have different optimal values across different levels of the explanatory variables. There are many theoretical reasons consistent with this occurring in many different applications including energy, finance, economics, social science, and manufacturing.

[1] The text, *Specifying and Diagnostically Testing Econometric Models*, by Houston H. Stokes Greenwood Press (1997) documents the basic B34S capability. A comprehensive document covering the B34S matrix command facilities is under preparation.

SCAB34S SPLINES integrates the General Public License (GPL) code from Trevor Hastie and Robert Tibshirani for multivariate adaptive regression spline (MARSPLINE) modeling. These routines were coded from scratch by the authors and do not use any of Friedman's routines.

Friedman (1991) originally introduced multivariate adaptive regression splines methodology. He shows how to systematically identify and estimate a MARSPLINE model whose coefficients differ based on the levels of the explanatory variables. Lewis and Stevens (1991) discussed the application of MARSPLINE models on lagged values of a time series as Adaptive Spline Autoregression (ASTAR). The ASTAR model in later published work is also referred to more generally as Time Series Multivariate Adaptive Regression Splines (TSMARS). The SCAB34S SPLINES product is applicable to data organized in both cross-sectional and time series formats.

The breakpoints or thresholds that define a change in a model coefficient are termed spline-knots and can be thought of similar to a piecewise regression. An advantage of the MARSPLINE approach is that the spline knots are determined automatically by the procedure. In addition, complex nonlinear interactions between variables can also be specified. The MARSPLINE model is particularly powerful in situations where there are large numbers of explanatory variables and low-order interaction effects. The equation switching model, in which the slope of the model abruptly changes for a given value of the X variable, is a special case of the MARSPLINE model. The MARSPLINE methodology can detect and fit models in situations where there are distinct breaks in the data, such as are found if there is a change in the underlying probability density function of the coefficients and where there are complicated variable interactions. After an overview of MARSPLINE modeling, some examples will be presented to illustrate the use of these procedures.

Model specification in SCA WorkBench is intuitive and easy to use. It is also quite flexible, providing options to control knot optimization, perform model validation, and evaluate forecast performance.

1. MARSPLINE MODELING USING SCAB34S SPLINES AND WORKBENCH

Assume a nonlinear model of the form

$$y = f(x_1, \cdots, x_m) + e \tag{1}$$

involving N observations on m explanatory variables, $x_1, \cdots, x_m$. The **MARSPLINE** procedure attempts to approximate the nonlinear function $f(\cdot)$ by

$$\hat{f}(\underline{X}) = \sum_{j=1}^{s} c_j K_j(\underline{X}), \tag{2}$$

where $\hat{f}(\underline{X})$ is an additive function of the product basis functions $\{K_j(\underline{X})\}_{j=1}^{s}$ associated with the *s* sub-regions $\{R_j\}_{j=1}^{s}$ and c_j is the coefficient for the jth product basis function. If all sub-regions include the complete range of each of the explanatory variables, then the coefficients $\{c_j\}_{j=1}^{s}$ can be interpreted as just OLS coefficients of variables or interactions among variables. The **MARSPLINE** procedure can identify the sub-regions under which the coefficients are stable and detect any possible interactions up to a maximum number of interactions controllable by the user. For example, assume the model

$$\begin{aligned} &y = \alpha + \beta_1 x + e \quad \text{for } x > 100, \textit{ and} \\ &y = \alpha + \beta_2 x + e \quad \text{for } x < 100 \end{aligned} \tag{3}$$

In terms of the MARSPLINE model notation, this is written

$$y = \alpha' + c_1(x - \tau^*)_+ - c_2(\tau^* - x)_+ + e, \tag{4}$$

where $\tau^* = 100$ and $(\cdot)_+$ is the right (+) truncated spline function which takes on the value 0 if the expression inside $(\cdot)_+$ is negative and its actual value if the expression inside $(\cdot)_+$ is > 0. Here $c_1 \equiv \beta_1$ and $c_2 \equiv \beta_2$. In terms of equation (3), $K_1(X) \equiv (x - \tau^*)$ and $K_2(X) \equiv (\tau^* - x)$. Note that the derivative of the spline function is not defined for values of *x* at the knot value of 100. Friedman (1991b) suggests using either a linear or cubic approximation to determine the *y* value. In the results reported later, both evaluation techniques were tested and the one with the lowest residual sum of squares value was selected. The user selects the maximum number of knots to consider and the highest order interaction to investigate. An example of an interaction model for $y = f(x, z)$ follows

$$y = \alpha + c_1(x - \tau_1^*)_+ + c_2(\tau_1^* - x)_+ + c_3(x - \tau_1^*)_+(z - \tau_2^*)_+ + e \tag{5}$$

implies that

$$\begin{aligned} &y = \alpha + c_1 x - c_1\tau_1^* + e && \text{for } x > \tau_1^* \text{ and } z < \tau_2^* \\ &y = \alpha - c_2 x + c_2\tau_1^* + e && \text{for } x < \tau_1^* \\ &y = \alpha + c_1 x - c_1\tau_1^* + c_3(xz - \tau_1^* z - \tau_2^* x + \tau_1^*\tau_2^*) + e && \text{for } x > \tau_1^* \text{ and } z > \tau_2^*. \end{aligned} \tag{6}$$

As an aid in determining the degree of model complexity, Friedman (1991b) suggests using a modified form of the generalized cross validation criterion (MGCV).

$$MGCV = [(1/N)\sum_{i=1}^{N}(y_i - \hat{f}(X))^2]/[1-[C(M)^*/N]^2], \quad (7)$$

where N is the number of observations, $\hat{f}(X_i) \equiv \hat{y}_i$ and $C(M)^*$ is a complexity penalty. The default is to set $C(M)^*$ equal to a function of the effective number of parameters. The formula used is

$$C(M)^* = C(M) + \delta M. \quad (8)$$

The parameter δ, which is user controlled, is typically set to the default value of 2 for the GPL MARSPLINE routines from Hastie and Tibshirani . $C(M)$ is the number of parameters being fit and M is the number of non-constant basis functions in the model. The MARSPLINE approach starts out by investigating where to place the knots for a non-interaction model. Next more complex interactions are investigated up to a user-controlled maximum number of interactions and maximum number of parameters in the model. Once the forward selection is completed, the MGCV statistic is used to eliminate parameters that do not improve the model. The MGCV value controls how many parameters will finally remain in the model and can be used to form an estimate of the relative importance of each x_i variable in the model.[3]

Unlike researchers who use recursive partitioning models with disjoint sub-regions and hence have problems approximating a linear model, Friedman recommends that the parent region not be eliminated when sibling sub-regions are formed. As Lewis and Stevens (1991) note "an immediate result of retaining parent regions is overlapping sub-regions of the domain. Moreover, each parent region may have multiple sets of sibling sub-regions. With this modification, recursive partitioning can produce linear models with the repetitive partitioning of the initial region R_1 by different predictor variables. Additive models with functions of more than one predictor variable can result from successive partitioning with different predictor variables. This modification also allows for multiple partitions of the same predictor variable from the same parent region."

The MARSPLINE routine requires that the user specify the candidate variables $x_1, \cdots, x_m$ to use in (1). For time series models with a few variables, it is proposed that one first uses a VAR or constrained VAR model to determine the maximum number and placement of the lags to filter the y and x_i series in the linear domain. If the resulting residuals show evidence of nonlinearity, as measured by the Hinich (1982) test, then significant (or all) lags of the y series and lags of the other x_i variables in the VAR model are used as explanatory variables in the MARSPLINE model equation. It is important to note that such a procedure would not be strictly appropriate if there were evidence of feedback found in the VAR model.

1.1 Interpreting the MARSPLINE Model Output Summaries

The structure of a MARSPLINE model can be quite complex considering that it may contain multiple basis functions, spline knots, and interaction terms. SCAB34S SPLINES displays the final MARSPLINE model using a "coding style" format. The coding style format can be used to implement the final MARSPLINE model outside of the SCAB34S SPLINES engine for a variety of applications such as scoring a database, simulation study, and predictive modeling. Below is an example of the MARSPLINE model output from SCAB34S SPLINES:

```
MARSpline Model Coefficients                                          SE                t

Y         =        4.5279553                                      0.83783455E-01       54.04
+   -12.314783    * max(  0.90700000     -  X1{  0}   , 0.0)   0.37832170          -32.55
+  -0.12798485    * max(   15.000000     -  X2{  0}   , 0.0)   0.10927898E-01      -11.71
+   0.68449752    * max(   15.000000     -  X2{  0}   , 0.0)   0.21982449           3.114
                  * max(  0.71500000     -  X1{  0}   , 0.0)
+    11.088234    * max( X1{  0}    -    1.1440000    , 0.0)    1.8244547           6.078
+   -21.967842    * max( X1{  0}    -   0.99000000    , 0.0)    5.7715069          -3.806
+   -7.1267043    * max( X1{  0}    -   0.86800000    , 0.0)    1.1136820          -6.399
+    13.768145    * max( X1{  0}    -    1.0160000    , 0.0)    5.4674066           2.518
+   0.58145323    * max( X2{  0}    -    15.000000    , 0.0)   0.13338996           4.359
                  * max(  0.90700000     -  X1{  0}   , 0.0)
```

The model summary in the above coding-style format can be easily identified as additive components. In examining the first model component

```
+   -12.314783    * max(  0.90700000     -  X1{  0}   , 0.0)
```

the product (-12.314783)(.907 – $X1_t$) is used in the model if (.907 – $X1_t$) is greater than zero. If the model component is an interaction such as

```
+   0.68449752    * max(   15.000000     -  X2{  0}   , 0.0)
                  * max(  0.71500000     -  X1{  0}   , 0.0)
```

then both the (15.0 – $X2_t$) and the (.715 – $X1_t$) conditions must be greater than zero for this interaction term to be used. The standard errors and psuedo t-ratios for the estimated coefficients are also presented for convenience.

1.2 Nonlinear Testing

The Hinich (1982) test has proved invaluable in detecting whether nonlinearity has been removed from the residuals. The crucial issue is whether after the nonlinearity is removed, can the model adequately forecast? If in the process of removing the nonlinearity the model is over-fit, the forecasting performance may deteriorate. While the Hinich (1982) test may miss some types on nonlinearity that could be filtered by the MARSPLINE model, any model failing the Hinich test is certainly a candidate for MARSPLINE modeling.

Since the Hinich (1982) test only detects third-order nonlinearity, and MARSPLINE can address nonlinearity of orders greater than three, the MARSPLINE model can accomodate nonlinearity in data that cannot be detected with the Hinich diagnostic test. The Cleveland and Devlin (1988) nitrous oxide dataset discussed later illustrates just such a situation. For this case the Hinich test on the OLS model does not reject linearity, yet substantial improvements in fit are possible with the MARSPLINE procedures.

```
Hinich82 Nonlinear Tests  -  OLS Residuals
--------------------------------------------------
Gaussality (Mean)    :        -1.208
Linearity  (Mean)    :        -0.365
```

The Hinich test is a one-tailed test. A test value of 2.0 (or greater) indicates that nonlinearity is detected at the 5% level.

1.3 Confusion Matrix

When a MARS-Probit model is considered, information about actual and predicted values can be classified in a confusion matrix (Kohavi and Provost, 1998). The confusion matrix adopted for the MARSPLINE modeling application has the following structure:

CONFUSION MATRIX		**Predicted**		
		Negative $prob < p_n$	*Positive* $prob > p_p$	*Unclassified* $p_n > prob < p_p$
Actual	*Negative*	a	b	U1
	Positive	c	d	U2

The user must provide threshold values to classify the predicted probability values as negative or positive. The default is $p_n = p_p = 0.50$. When $p_n = p_p$ then all predicted probability values will be classified as either negative or positive instances. If threshold values are specified such that $p_n < p_p$, then there is a possibility of predicted probability values falling outside the defined range and therefore becoming unclassified. The confusion matrix reports if any cases cannot be classified.

In addition to the above table which displays a cross tabulation of the number of predicted negative/positive classifications to the number of actual negative/positive instances, several standard ratios are computed:

Accuracy Rate	$\frac{a+d}{a+b+c+d}$	The accuracy rate is the proportion of the total number of correct predictions.
True Positive Rate (TP)	$\frac{d}{c+d}$	The true positive rate is also called the recall rate and is defined as the proportion of positive cases that were correctly identified by the model.

False Positive Rate	$\frac{b}{a+b}$	The false positive rate is the proportion of negative cases that were not identified correctly by the model.
True Negative Rate (TN)	$\frac{a}{a+b}$	The true negative rate is the proportion of negative cases that were identified correctly by the model.
False Negative Rate	$\frac{c}{c+d}$	The false negative rate is the proportion of positive cases that were not classified as negative by the model.
Precision rate (P)	$\frac{d}{b+d}$	The precision rate is the proportion of predicted positive cases that were correctly classified by the model.
g-mean1	$\sqrt{TP \bullet P}$	When the number of negative cases is much greater than the number of positive cases, the above accuracy rate may not be a good performance measure. For example, if there are 100 cases of which 5 are positive and the remaining negative, the accuracy rate of a model that predicted all negative would be 95% even though the model failed to correctly classify a single positive case. To account for this situation, Kubat et al. (1998) suggest the geometric mean where the true positive rate is in the product.
g-mean2	$\sqrt{TP \bullet TN}$	The g-mean2 equation is based on the same principle as the g-mean1 equation. Kubat et al (1998).

The determination of probability threshold values for classification purposes is subjective. It is often set based on achieving the highest True Positive (recall) rate while minimizing/maximizing another rate such as False Positive rate or True Negative rate. The final choice of threshold cut-off is most often dictated by the purpose of the classification model.

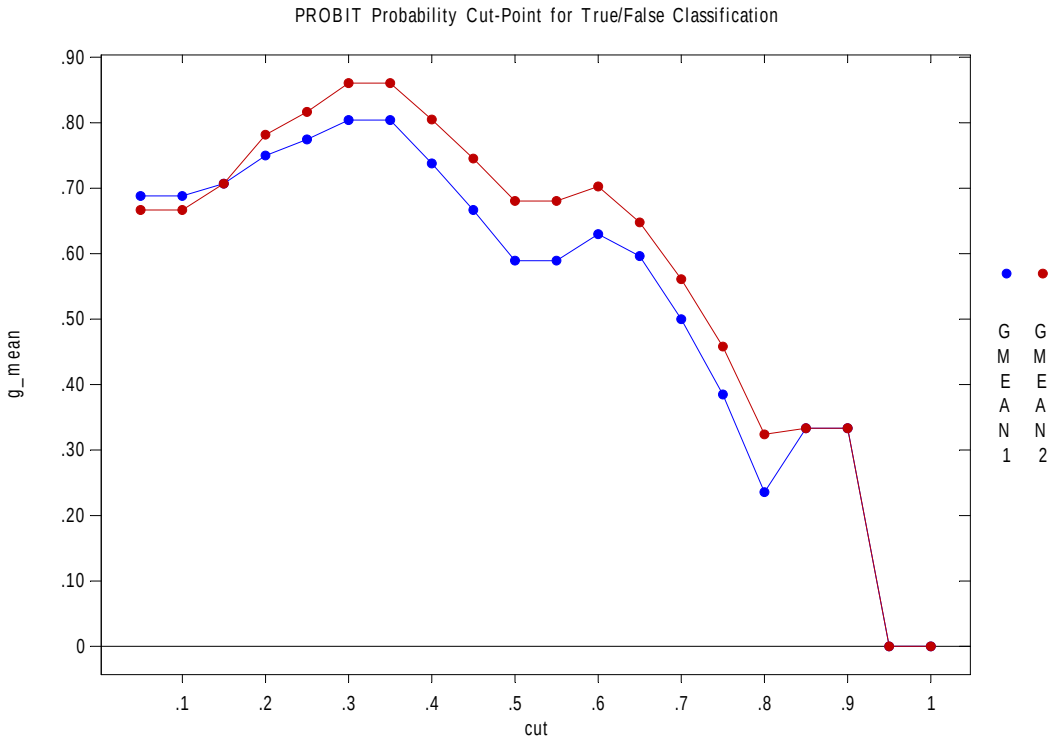
For purposes of evaluating the impact of various probability threshold values on the classification results, the MARS-Probit application provides an extended table that computes the various ratios across the probability cut-off range of 0-1 as shown below.

PROBIT True/False Probability CutOff Range

Cut	True-Pos	True-Neg	False-Pos	False-Neg	Accuracy	Precision	GMean1	GMean2
0.050	1.000	0.444	0.556	0.000	0.630	0.474	0.688	0.667
0.100	1.000	0.444	0.556	0.000	0.630	0.474	0.688	0.667
0.150	1.000	0.500	0.500	0.000	0.667	0.500	0.707	0.707
0.200	1.000	0.611	0.389	0.000	0.741	0.562	0.750	0.782
0.250	1.000	0.667	0.333	0.000	0.778	0.600	0.775	0.816
0.300	0.889	0.833	0.167	0.111	0.852	0.727	0.804	0.861
0.350	0.889	0.833	0.167	0.111	0.852	0.727	0.804	0.861
0.400	0.778	0.833	0.167	0.222	0.815	0.700	0.738	0.805
0.450	0.667	0.833	0.167	0.333	0.778	0.667	0.667	0.745
0.500	0.556	0.833	0.167	0.444	0.741	0.625	0.589	0.680
0.550	0.556	0.833	0.167	0.444	0.741	0.625	0.589	0.680
0.600	0.556	0.889	0.111	0.444	0.778	0.714	0.630	0.703
0.650	0.444	0.944	0.056	0.556	0.778	0.800	0.596	0.648

0.700	0.333	0.944	0.056	0.667	0.741	0.750	0.500	0.561
0.750	0.222	0.944	0.056	0.778	0.704	0.667	0.385	0.458
0.800	0.111	0.944	0.056	0.889	0.667	0.500	0.236	0.324
0.850	0.111	1.000	0.000	0.889	0.704	1.000	0.333	0.333
0.900	0.111	1.000	0.000	0.889	0.704	1.000	0.333	0.333
0.950	0.000	1.000	0.000	1.000	0.667	0.000	0.000	0.000
1.000	0.000	1.000	0.000	1.000	0.667	0.000	0.000	0.000

By default, the MARS-Probit application uses a cut-off probability threshold that maximizes G-MEAN1 or G-MEAN2. In the above table, the cut-off would be computed as 0.325 = (.30+.35)/2. In addition to the table, a graph is generated to chart the G-MEANs against the cut-off probabilities.



1.4 Lift-Gain Table

A lift-gain table measures how well a classification system (e.g. logistic model) predicts the probability of a positive outcome. The data are sorted from highest probability to lowest probability and grouped into deciles. The primary statistics displayed in the Lift-Gain table are computed as follows

$$\text{Cumulative Gain}_d = \frac{\sum_{t=1}^{d} \text{Positives}_t}{\text{Total Positives}}, \quad d=1,2,...,10$$

$$\text{Lift-Index}_d = \frac{\sum_{t=1}^{d} \text{Positives}_t}{d\left(\text{Total Positives}/10\right)}, \quad d=1,2,...,10$$

$$\text{K-S Spread}_d = \frac{\sum_{t=1}^{d} \text{Positives}_t}{\text{Total Positives}} - \frac{\sum_{t=1}^{d} \text{Negatives}_t}{\text{Total Negatives}}, \quad d=1,2,...,10$$

$$\text{Gain-over-Random}_d = 1 - \left(\frac{d}{10} \Big/ \text{Cumulative Gain}_d \right), \quad d=1,2,...,10$$

The K-S (Kolmogrov-Smirnov) spread is a nonparametric statistic that tests whether two samples are from the same population. By comparing the proportion of positive and negative cases at each decile, the maximum K-S spread indicates the decile where the model classifies the greatest number of positive cases in the scoring population while minimizing the number of misclassifications. In acquisition modeling, the K-S spread is used to determine a cut-off percentage of the scoring universe to be targeted with consideration to opportunity costs. In the example below, a holdout sample of 100 cases have been scored using a MARS-Probit model. From the 100 sample, there are a total of 25 positive outcomes. Based on a non-modeling approach and the assumption that the number of positive outcomes are distributed evenly across the sample, one would expect to observe 10% of the total positive outcomes to be cumulated in the first decile, 20% of the total positive outcomes to be cumulated in the second decile, and so on. This expected cumulative gain is compared against the cumulative positive outcomes obtained by the classification system for each decile. Below, the logistic MARS-Probit model classified 40% of the total positive outcomes in the first decile, 80% in the second decile, and 100% in the fourth decile. Furthermore, the K-S spread is greatest in the third decile which indicates that the classification system provides greatest returns within the top 30% scored.

The lift can simply be thought of as the ratio between the cumulative gain of the classification system and the cumulative expected gain.

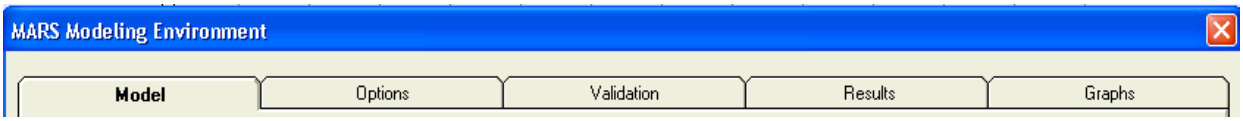
MARSpline Lift-Gain Table

Decile	#Obs in Decile	#Pos in Decile	%Pos in Decile	Pctg of Total Pos	Cum. #Obs	Cum. #Pos	Cum. %Pos	Cum. Gain	K_S Spread	Lift Index	Gain over Random
1	10	10	100.0%	40.0%	10	10	10.0%	40.0%	40.0%	400	75.0%
2	10	10	100.0%	40.0%	20	20	20.0%	80.0%	80.0%	400	75.0%
3	10	4	40.0%	16.0%	30	24	24.0%	96.0%	88.0%	320	68.8%
4	10	1	10.0%	4.0%	40	25	25.0%	100.0%	80.0%	250	60.0%
5	10	0	0.0%	0.0%	50	25	25.0%	100.0%	66.7%	200	50.0%
6	10	0	0.0%	0.0%	60	25	25.0%	100.0%	53.3%	166	40.0%
7	10	0	0.0%	0.0%	70	25	25.0%	100.0%	40.0%	142	30.0%
8	10	0	0.0%	0.0%	80	25	25.0%	100.0%	26.7%	125	20.0%
9	10	0	0.0%	0.0%	90	25	25.0%	100.0%	13.3%	111	10.0%
10	10	0	0.0%	0.0%	100	25	25.0%	100.0%	0.0%	100	0.0%

2. SCA WorkBench: A Graphical User Interface

SCA WorkBench provides a convenient graphical user interface to SCAB34S SPLINES for MARSPLINE modeling. The WorkBench interface builds the data loading steps and MARSPLINE commands based on the user's menu selections. The associated commands are then organized as an SCAB34S program file and submitted to the SCAB34S engine.

The MARSPLINE modeling environment in WorkBench is organized by tabs shown below.



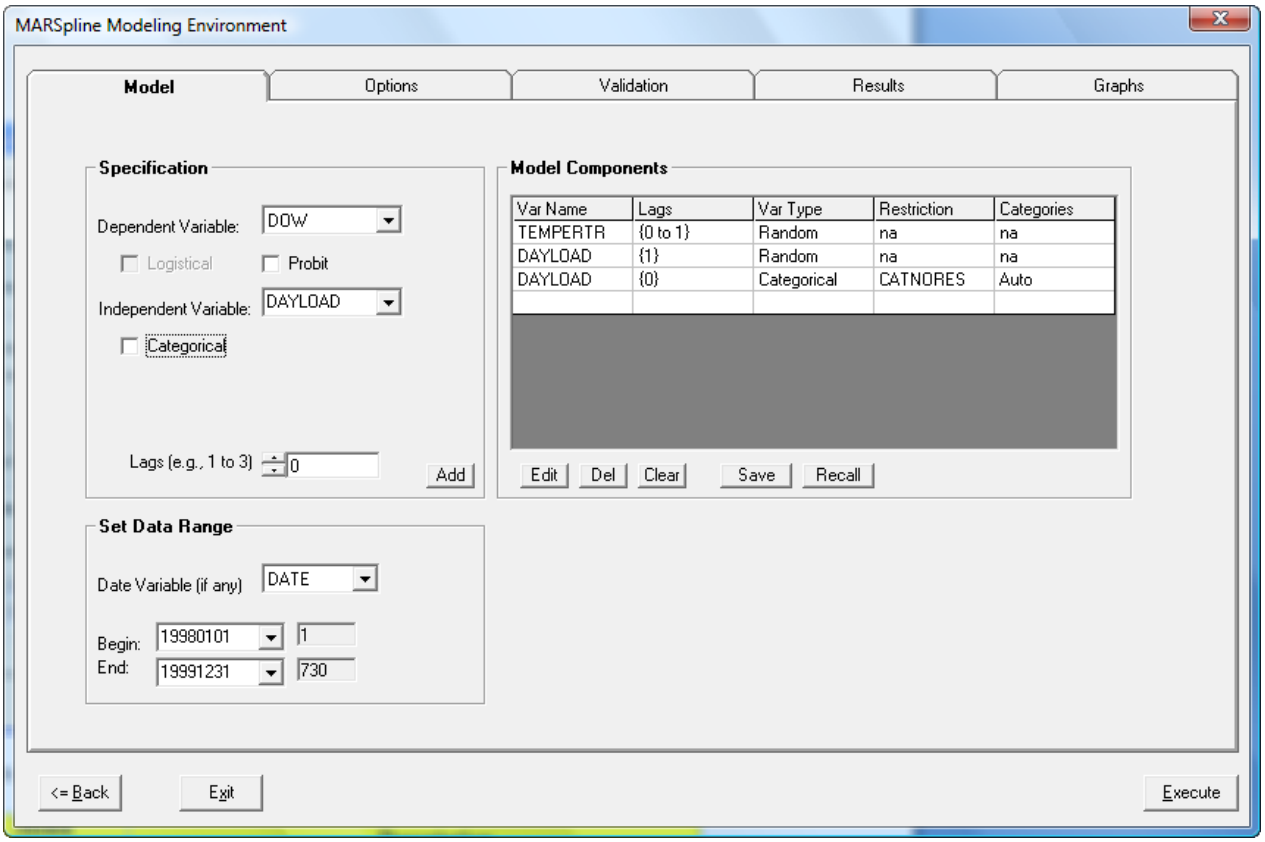
The *Model* tab is used to specify the variables, variable types, and lagged components of the MARSPLINE model. The *Options* tab sets the estimation limits placed on a MARSPLINE model, and controls the detail of output and graphics that are produced. The *Validation* tab provides settings to evaluate the performance of MARSPLINE model prediction and to compare the results with a linear regression model using simple OLS, MINIMAX or L1 estimation. The *Results* tab displays the output from the model estimation, diagnostics, and forecasting. The *Graphs* tab displays a variety of high resolution graphics such as time series plots, residual plots, autocorrelation plots, and others.

Once the SCAB34S program file is created by SCA WorkBench, you may save the file for future reference or make changes directly to the commands and re-execute the script from SCA WorkBench.

2.1 Model Specification Tab

This tab is central to specifying the variables and lagged components of the MARSPLINE model. Use the dropdown combo boxes to select your dependent variable and independent variables. Click on the **Add** button to add an independent variable component to the model. The components will appear in the *Model Components* grid as they are added. In the example below, DAYLOAD is selected as the dependent variable. TEMPERTR is selected as an independent variable with a contemporaneous effect and a lag 1 effect. The lags are specified in the **Lags** textbox. Multiple lags for explanatory variable components can be specified using the word "TO" to separate contiguous lags (e.g., 0 TO 1) or commas to separate non-contiguous lags (e.g., 0, 1, 3).

A component may be deleted or modified by placing your cursor on the specific grid row and then by clicking on the **Del** or **Edit** buttons. If you click on **Edit**, the **Add** button will be replaced by the **Mod** button. You may make changes using the dropdown combo box for the independent variable and other components in the *Specification* frame. Click on the **Mod** button to complete the modification.



The features of the Model specification tab are presented below.

Menu Item	Description
Specification Frame	This frame organizes various controls that you may use to specify MARSPLINE model components including the dependent variable, independent variables, and lag coefficients. If a categorical variable is specified for an independent variable, the MARSPLINE routine will automatically identify it as categorical when it is processed.
Dependent Variable	Use this drop-down list to specify the series that you wish to analyze.
Probit Checkbox	Specifies that the independent variable is a 0-1 variable. When specified, the MARSPLINE model estimates the probability of success/failure based on the independent variables in the model.
Categorical Checkbox	Specifies that the dependent variable is a categorical variable. When specified, the application will automatically determine the number of categories (must be coded as integer) and expand the categorical variable into multiple binary (0-1) variables.
Independent Variable	Use this drop-down list to specify a random or categorical regressor variable component in the model.

Lags	Specifies the lag parameters associated with a random variables or categorical variables. A categorical variable may contain more than one lag parameter; however only one lag specification may be added to the model at a time. For random variables, multiple lag parameters may be added to the model as a group. Multiple lags may be specified using the "TO" keyword to separate contiguous lags. Individual lags may be separated by commas. For example, the user could specify contiguous lags as "0, 1, 3" or as "0 TO 1, 3".
Add	Clicking on Add appends a new component to the MARSPLINE model which is displayed in the model component grid. Multiple instances of the same independent variable may be added to the model as long as the lag operators are unique. For example, in the above form, the user could have optioned to specify the TEMPERTR{0} and TEMPERTR{1} components separately.
Model Components Frame	The model components frame organizes form controls to display the MARSPLINE model components in a grid format, as well as to edit and delete model components.
Model Component Grid	The components of the MARSPLINE model and their attributes are displayed in this grid. The first column displays the independent variable name, the second column displays the individual or grouped lag operators within braces, the third column indicates whether the independent variable is predetermined as random or categorical, the fourth and fifth columns are not used at this time.
Edit	The user can modify a model component by first placing the mouse cursor on the grid row of interest and then clicking on the Edit button. The Specification Frame will reflect the current attributes of the model component and the Add button will be replaced by the Mod button. Make the necessary changes in the Specification Frame and then click on the Mod button to complete the changes.
Del	The user can delete a model component by placing the mouse cursor on the grid row of interest and then clicking on the Del button.
Clear	Clears all model components from the model component grid.
Save	Saves the information in the model component grid to a specified tab-delimited file.
Recall	Recalls the model component grid information from a specified tab-delimited file created (see Save option above).
Set Data Range Frame	This frame organizes form controls related to how the data is indexed (by date or none), and what data span is modeled and analyzed.
Date Variable	Use this drop-down list to specify the date variable associated with your series. If your SCA Data Macro contains a variable named "DATE", it is automatically assigned by SCA WorkBench. If you have an alternative index variable or date variable, you may select it from the drop-down list. If your SCA Data Macro does not

contain a DATE variable, leave the dropdown list empty. WorkBench will then use the observation number as a date index.

If your time series is more than 10,000 observations, WorkBench will not use your DATE variable for indexing. Instead, observation number will be used.

Begin Span Use the Begin drop-down list to omit observations from the beginning of a time series being analyzed.

End Span Use the End drop-down list to omit observations from the back of a time series being analyzed.

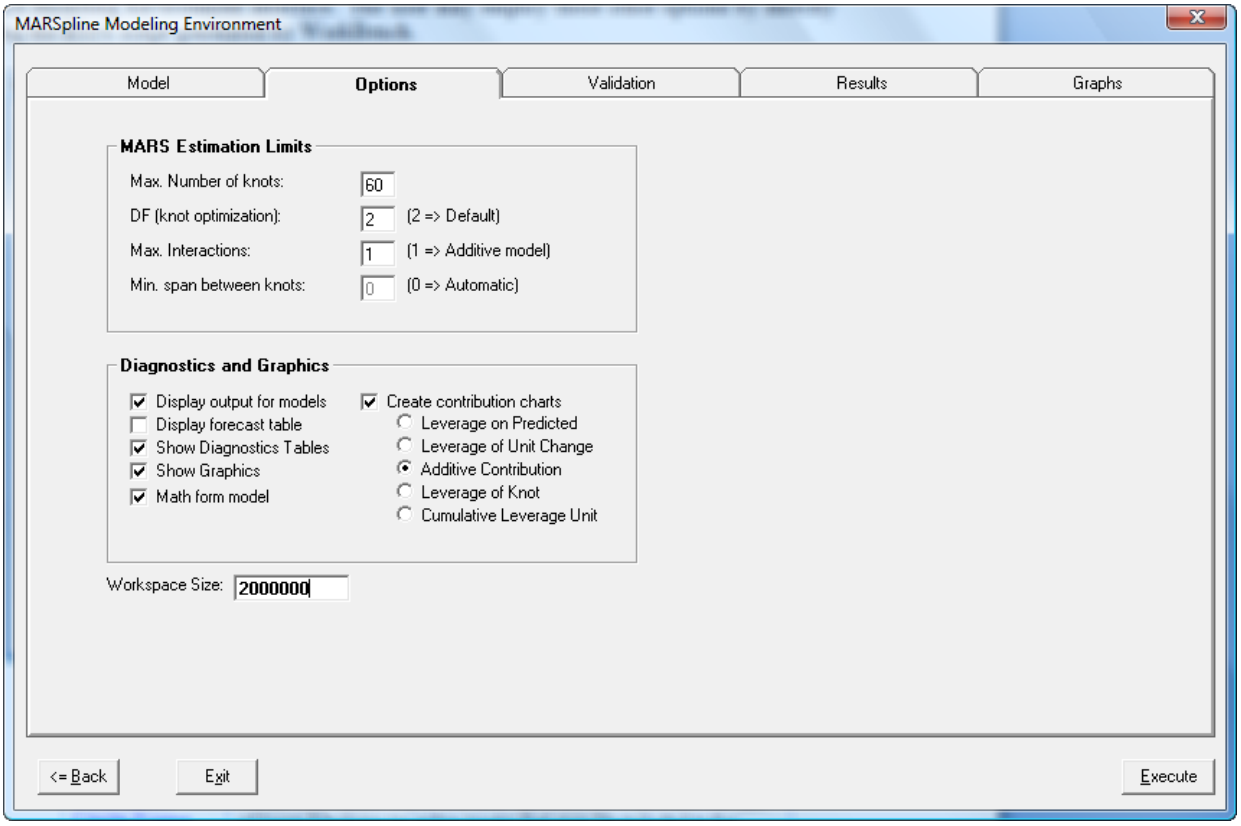
Back Depending on the tab you are currently working in, clicking on the Back button will move you one tab to the left. If you are in the Model tab, you will move to the MARSPLINE Data Viewer dialog box where you may choose a new SCA data macro or leave the MARSPLINE Modeling Environment.

Exit Exits the MARSPLINE modeling environment.

Execute Executes MARSPLINE model estimation, validation, linear model comparison, diagnostics, and graphs by submitting a dynamically created program script to SCAB34S SPLINES. When completed, you will automatically be placed in the Results tab.

2.2 Options Tab

The Options tab sets the estimation limits placed on a MARSPLINE model, controls the detail of output and graphics that is produced, and allocates the workspace size of the SCAB34S SPLINES product. More estimation options are available in the MARSPLINE matrix subroutine that are not exposed in this MARSPLINE Modeling Environment interface. The user may employ these other options by directly editing the B34S script generated by WorkBench.



Menu Item	Description
MARSPLINE Estimation Limits Frame	This frame organizes various controls that set options in MARSPLINE model estimation. Here, the user specifies the maximum number of basis functions (or spline knots) that may be included in the MARSPLINE model, the degrees of freedom (or penalty) imposed for knot optimization, and the maximum number of interactions that may occur between regressor variables. The minimum span between knot placements is automatically determined by MARSPLINE.
Maximum Number of Knots	The maximum number of possible knots limits the number of knots that may be included in the MARSPLINE model during its selection process. Increasing this setting allows for a greater number of basis functions to be evaluated before pruning and sometimes results in a better performing model. The default is 5.

DF (knot optimization) Sets the number of degrees of freedom charged for unrestricted knot optimization.

Maximum Interactions Sets the maximum number of interactions between variables for any given basis function.

Minimum Span between Knots The minimum span allowed between spline knots is automatically determined by the MARSPLINE subroutine when this option is set to 0. The number of regressor variables and the number of observations in the series determine the minimum span between knots. This setting currently can not be modified by the user.

Diagnostics and Graphics Frame This frame organizes controls related to the amount of output produced for MARSPLINE estimation and diagnostics. The math form option will display the model in a form that is easily transferred into a standard program language. The alternative is a model displayed in summation form. The contribution chart option produces contribution (or leverage) charts for all variables that are used in the final model. If the variable is additive (no interations with other variables) the companion variables are set to their median values. If interactions exist, three charts are produced that set the companion variables to their minimum, median, and maximum. The vertical axis displays the predicted values when the target variable takes on a range between its minimum and maximum while all other variables are held constant to their median values.

Display Output for Model Typically, you want to see the MARSPLINE model summary and the OLS model summary.

Display Forecast Table The forecast table displays the original series and the predicted series for both the MARSPLINE model and OLS models.

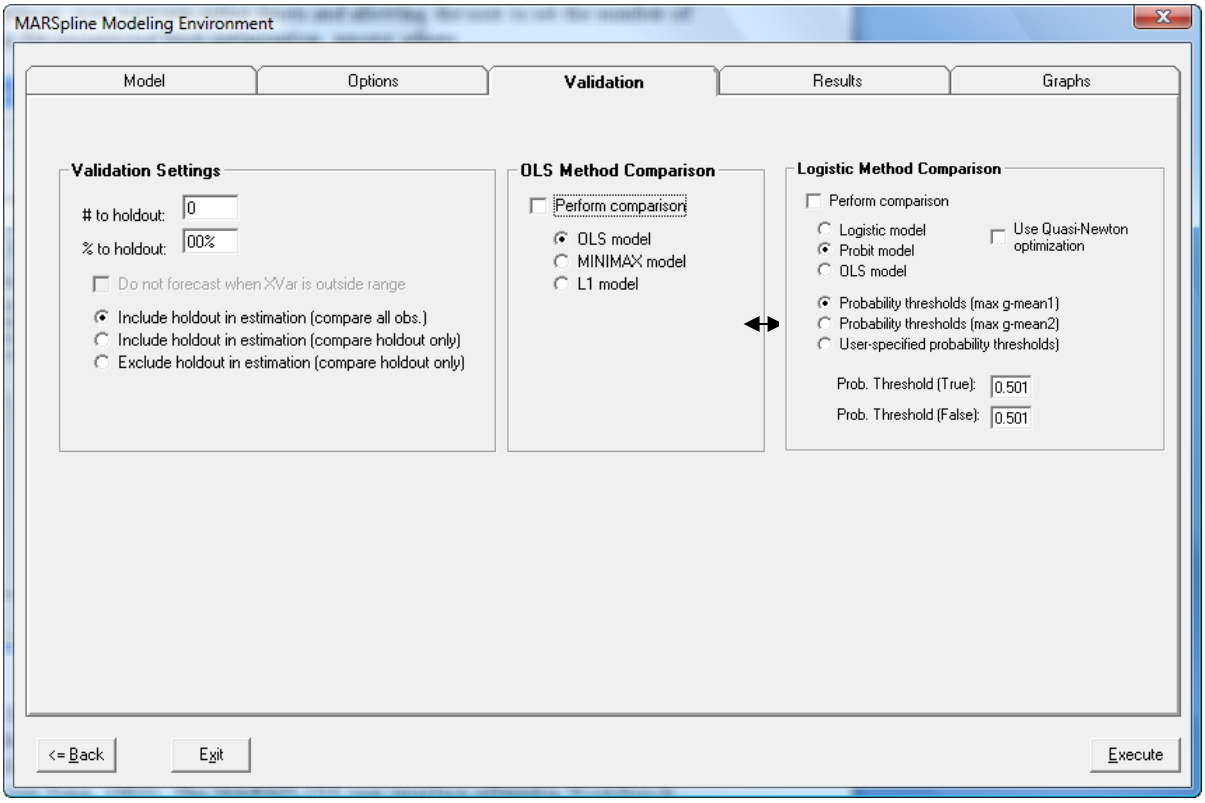
Show Diagnostic Tables Several diagnostics are available for the dependent variable and the residuals from the estimated models. Among the diagnostics are a statistical description tables, sample autocorrelation tables, and Hinich nonlinear testing. The Hinich test wil only be displayed for residual series greater than 50 cases.

Show Graphics Several graphics are created including time plot of the dependent variable, Actual vs. Predicted, ACF and PACF plots, and modified Q-Statistic plot.

Workspace Size The SCAB34S SPLINES product requires its workspace size to be set when the program is initiated. The default workspace is of 2000000 is adequate to handle moderate size datasets. The user may increase the workspace size if needed. Please note that workspace limit is imposed by the amount of available RAM memory of the computer.

2.3 Validation Tab

This tab allows you to evaluate the performance of MARSPLINE model prediction and validate the MARSPLINE model against a linear regression model method using simple OLS, MINIMAX or L1 estimation. A common problem with most nonlinear modeling methods is over-fitting. Models that over-fit the data often perform well within the sample, but do substantially worse when predicting out of sample. The MARSPLINE routine addresses this problem by penalizing overly complex models; controlling the minimum span between spline knots and allowing the user to set the number of degrees of freedom for unrestricted knot optimization, among others.



Although the MARSPLINE modeling approach is well suited to cross-sectional data, it can also be employed successfully on time series data. Lewis and Stevens (1991) discuss the application of MARSPLINE models (ASTAR) on lagged values of Y as an alternative to Threshold Autoregressive (SETAR) models (see Tong, 1983). The MARSPLINE user interface offered in WorkBench leverages its utility in time series applications by allowing the dependent variable to be lagged in the model thus addressing some issues related to serial correlation in the data.

The default validation setting compares the in-sample fit of the estimated MARSPLINE model against the in-sample fit of a simple OLS regression model. All available observations are used to evaluate fit using root mean squared error (RMSE) and mean absolute percentage error (MAPE) criteria.

Other options are available to validate the MARSPLINE model. For example, if the user is primarily interested in evaluating the fit of the model in the later part of the series, a holdout sample can be specified by typing the number of observations (or percentage) to be marked from the back of the series. After specifying the holdout, the user can evaluate in-sample fit for the "holdout period" only by setting the option "Include holdout in estimation (compare holdout only)". The user also has two choices to evaluate the prediction performance of the model where the holdout period is not used in training the model. Those options are to estimate the model up to the first forecast origin only or to re-estimate the model at each forecast origin. Note that re-estimating the model at each forecast origin is computing intensive and the spline knot placements and model structure may change across the sample period since the knots and variable inclusion is performed automatically during MARSPLINE estimation.

The MARSPLINE model can be visually examined as a multi-dimensional surface. The boundaries of this estimated surface are defined by the response of the dependent variable to a range of levels in the regressor variables. Problems arise from out of sample forecasting when the values of the regressor variables fall outside their minimum and maximum range in model estimation. When this occurs, the dependent variable's responses to these new levels are not known. Consequently, it is important that the MARSPLINE model be re-examined occasionally and the model retrained if new maximums or minimums are present in the updated data.

As another validation criterion, the user can compare the improvement of a MARSPLINE model versus a regression model with the same right-hand side variables. Diagnostics are produced for both the MARSPLINE and regression models. If the dependent variable is nonlinear in its response to levels of the regressor variables, the MARSPLINE model should reveal vast improvement in model fit and out-of-sample forecasting performance criteria.

A confusion matrix is produced for the MARSPLINE-Probit model and the comparison linear model for evaluating classification power of the models. The user has a choice for determining the probability cut-off value for classification of positive and negative cases for the final confusion matrix. The user can allow the system to set the probability cut-off automatically using the maximum G-MEAN values as the criteria, or using specific cut-off values. If GMEAN1 is used, the cut-off will slightly favor True-Positive classifications and if GMEAN2 is used, the cut-off will consider equally True-Positive and True-Negative classifications. Since the determination of cut-off probability thresholds is subjective, a table of ratio statistics for a range of cut-off probability values is also provided in the output.

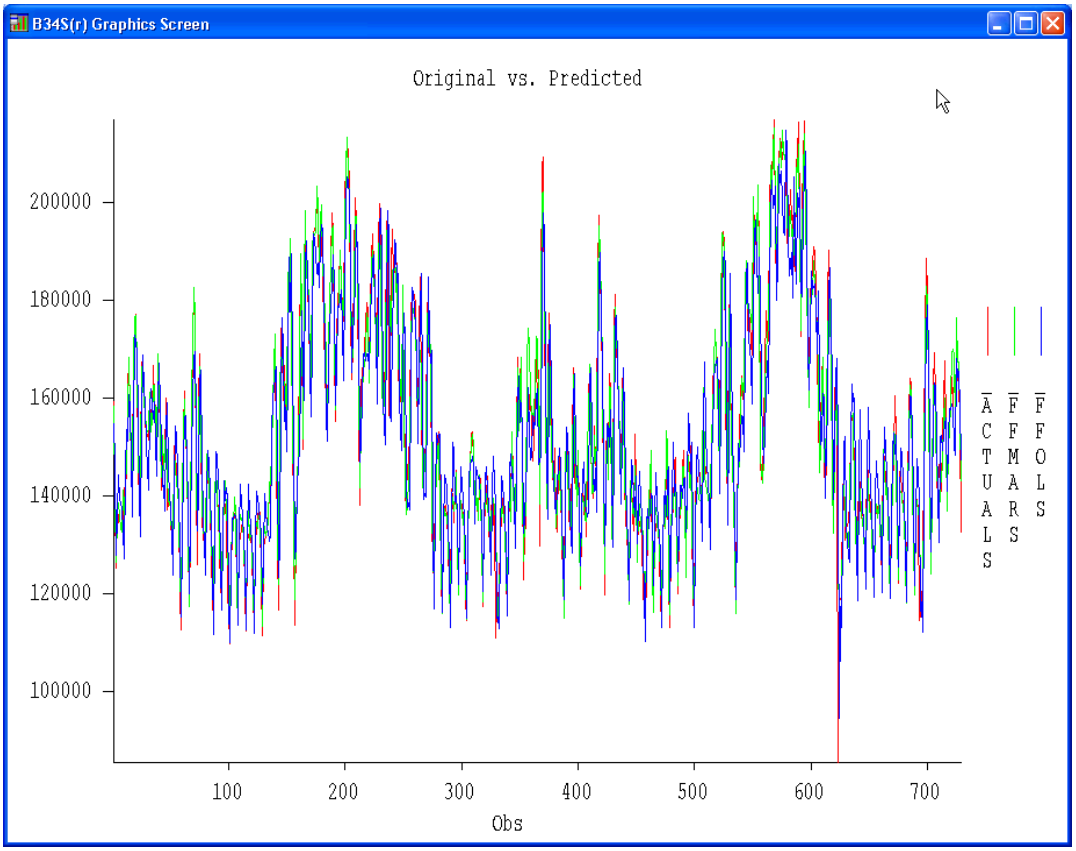
Menu Item	Description
Validation Settings Frame	This frame organizes controls for specifying a holdout sample for forecast performance and model validation. It also provides controls for the user to specify the type of validation for in-sample or out-of-sample forecasting.
# to holdout	Specifies the number of observations that are to be reserved from the back of the dependent variable for evaluating forecast performance. The percentage of the holdout sample relative to the series length is computed and is displayed in % to holdout.
% to holdout	Specifies the size of the holdout sample as a percentage of the

	length of the dataset. The actual number of observations reserved from the back of the series is computed and displayed in # to holdout.
Compare all obs	Evaluate the in-sample fit of the model for all observations.
Compare holdout only for in-sample fit	Evaluate the in-sample fit of the model for the defined holdout sample only
Compare holdout for out-of-sample fit	Evaluate the out-of-sample forecasts defined by the holdout sample. The model is estimated using observations up to the first forecast origin only
OLS Method Comparison Frame	This frame organizes controls to validate the MARSPLINE model against a regression model with the same right-hand-side variables used in the MARSPLINE model.
Logistic Method Comparsion Frame	This frame organizes controls to validate the MARS-Probit model against a Logit or Probit model with the same right-hand side variables used in the MARSPLINE model.
Perform comparison	By default a comparison is made to MARSPLINE using a simple OLS regression estimation method if the dependent variable is random. A comparison is not automatically performed if the dependent variable is specified as a logistical variable.
OLS model	Estimates a regression model using the ordinary least squares (OLS) method.
MINIMAX model	Estimates a regression model using the MINIMAX method which minimizes $\max\left(\text{abs}\left(Y_t - \hat{Y}_{t-1}\right)\right)$. This estimation method is more sensitive to outliers.
L1 model	Estimates a regression model using the L1 method which minimizes $\text{sum}\left(\text{abs}\left(Y_t - \hat{Y}_{t-1}\right)\right)$. This estimation method is not as sensitive to outliers as OLS or MINIMAX.
Logistic model	Estimates a logistic regression model in comparison to a MARSprobit model.
Probit model	Estimates a probit regression model in comparison to a MARSprobit model.
Probability thresholds	The threshold values for classifying a predicted case as a positive or negative instance.

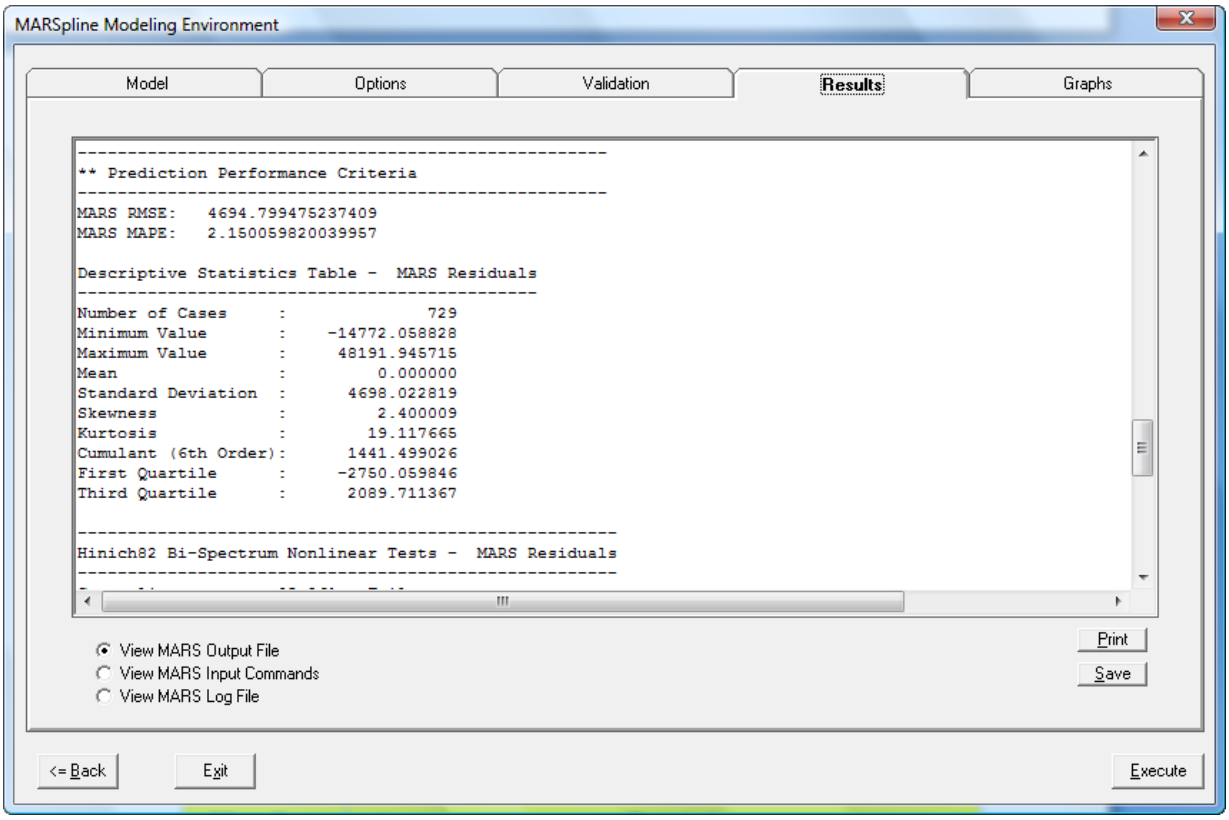
2.4 Results Tab

The results tab provides a convenient facility to view output from MARSPLINE model estimation. It also allows you to view the input commands for SCAB34S SPLINES execution. If there are errors during estimation, you can view the log file for a detailed account of all commands executed and error messages.

After the user executes the MARSPLINE model application by clicking on the **Execute** button, SCAB34S SPLINES will display a graph of the actual versus fitted data. This indicates that the MARSPLINE procedure has completed. The user should click anywhere on the graph (an example is shown below) to close it.



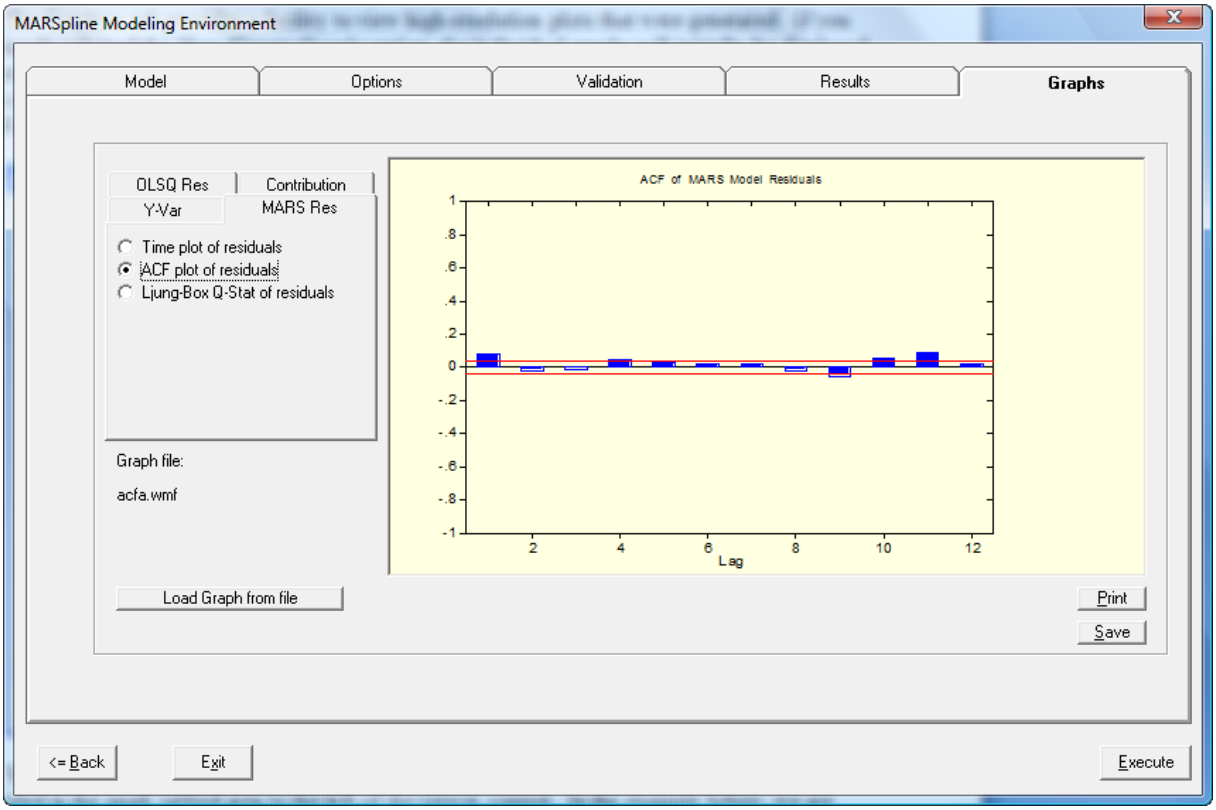
After the graph disappears, the user will be placed on the *Results* tab of the MARSPLINE Modeling environment where the output is listed.



Menu Item	Description
View MARSPLINE Output File	Displays the MARSPLINE modeling results and tabulated diagnostics.
View MARSPLINE Input Commands	Displays the input commands submitted to SCAB34S SPLINES. You can modify the commands directly in this window and submit the modified command file by clicking on the Execute button.
View MARSPLINE Log File	Displays a detailed command and error log for jobs submitted to SCAB34S SPLINES
Print	Send information displayed in the viewer to the printer.
Save	Saves the information in the viewer to a file. You may want to use this feature to save the modeling script with intentions of executing it later from the System -> Run SCA with Macro menu, or the System -> Run SCAB34S Program File menu.
Execute	While you are in the Results tab, if you click on Execute, you will send the information in the viewer to SCAB34S SPLINES for processing.

2.5 Graphs Tab

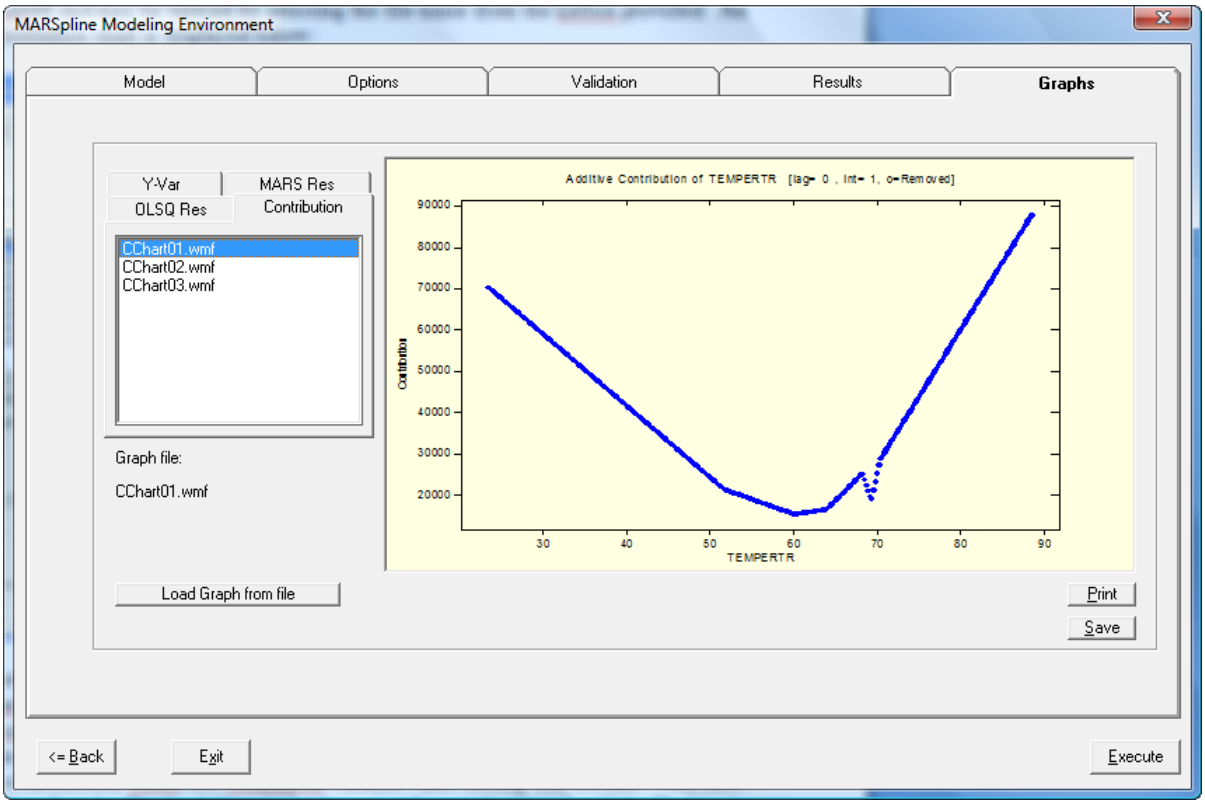
The Graphics tab provides a facility to view high-resolution plots that were generated. If you previously selected the *Show/Create Graphs* option, the individual graphs will initially be displayed on screen. When you click on the graph, the next generated graph will appear until all graphics have been created. As the graphs are displayed, they are also being saved as Windows Meta Files using fixed names such as "yvar.wmf" or "acfa.wmf".



You can review all created graphic files by selecting the graph from the set of radio buttons provided in the small tabbed area to the left of the viewer control. In the example below, we are viewing the sample autocorrelations of the MARSPLINE model residuals. The name of the graphic file (acfa.wmf) is displayed for reference. Since the graphs are saved to fixed file names, they are overwritten each time you generate a new set of graphs from the MARSPLINE modeling environment. If you wish to save the graphic file for future reference, please use the **Save** button on this tab to copy the file to a new name. Please do not rename the file extension because the **Save** button only renames the file. It does not convert it to a new format. You can view those renamed files by using the **Load Graph from File** facility. You may send the graph to the printer by clicking on the Print command button. If you double-click on the graph image it will load in the external program that is associated with WMF files on your computer (e.g., Windows FAX/Picture Viewer).

If you elected to create contribution charts, at least one contribution (leverage) chart will be generated for each variable used in the final model. Since a variable number of charts may be created based on number of explanatory varaiables and interactions, the file names are sequenced from

CChart01 – CChart## and may be viewed by selecting the file name from the listbox provided. An example of a contribution chart is displayed below:



The contribution chart reveals how the predicted value (vertical axis) is influenced by a particular explanatory variable (horizontal axis) in the model when all other variables are held constant. In the above graph, we are viewing the leverage of temperature at lag=0 when all other variables in the model are set to their median values. The title in the chart includes information on the lag, number of interaction terms, and companion variable value setting. The CChart*.wmf files are overwritten, therefore the file should be renamed or moved to another location if the file is to be saved for future reference.

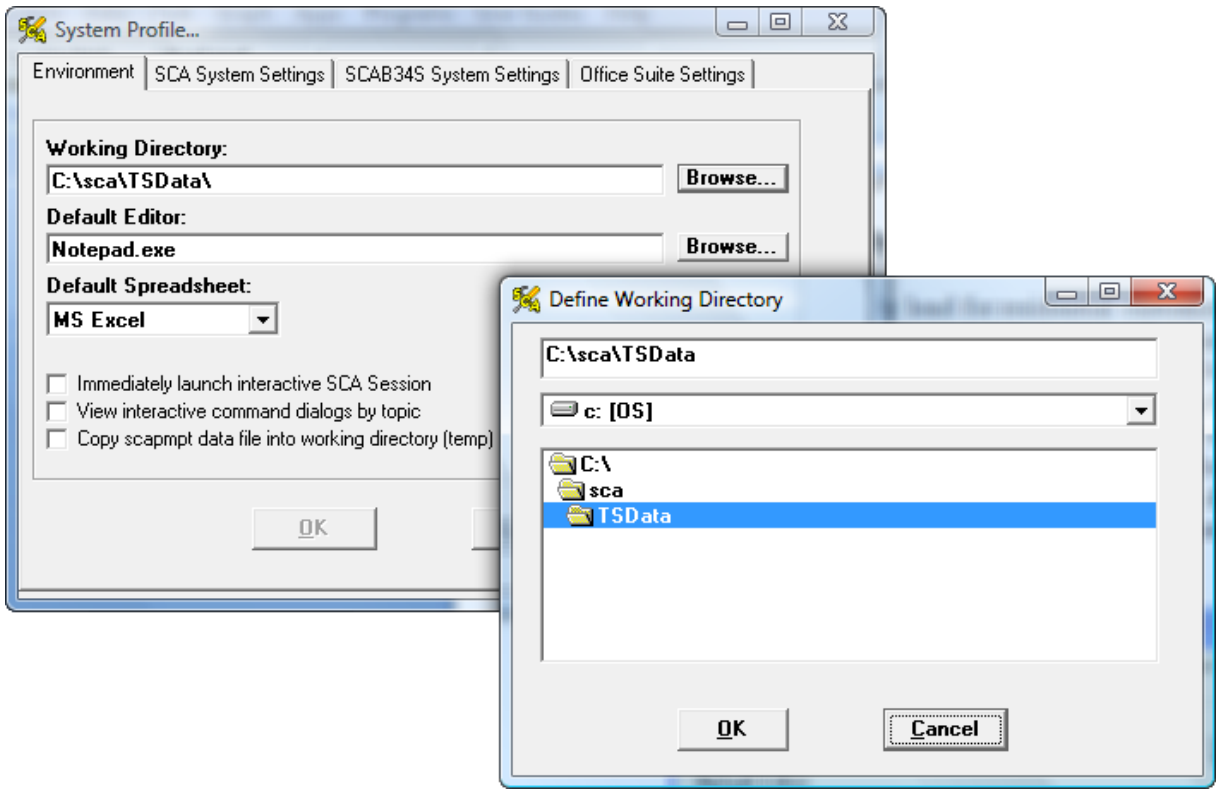
3. EXAMPLES OF MARSPLINE MODELING USING SCA WORKBENCH

This section provides various MARSPLINE modeling examples using SCA WorkBench and its interface to SCAB34S SPLINES. The first example explores the efficacy of MARSPLINE models in electricity load forecasting. A second example examines MARSPLINE models in process control using a nitrous oxide dataset. Finally, a third example demonstrates MARS-Probit modeling.

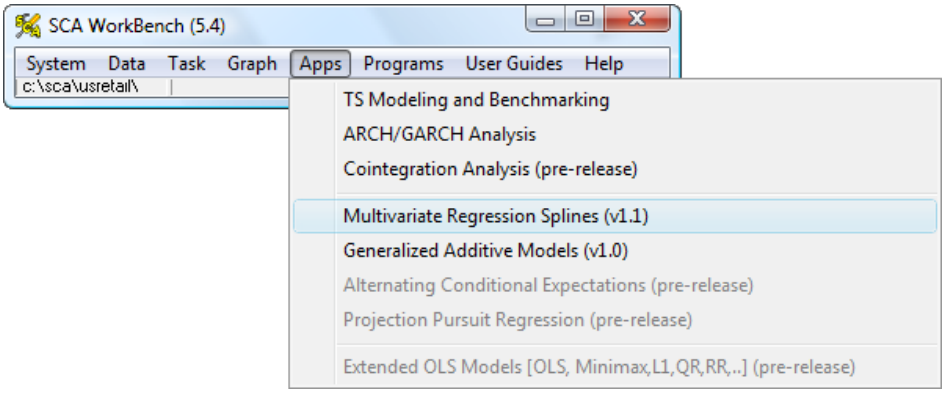
The data files used for the examples discussed in this section are available under the WorkBench installation folder in a sub-directory named **TSDATA**. The command files built by SCA WorkBench for the illustrated examples are presented later in Section 8 of this document.

3.1 Modeling Daily Electricity Load Using MARSPLINE

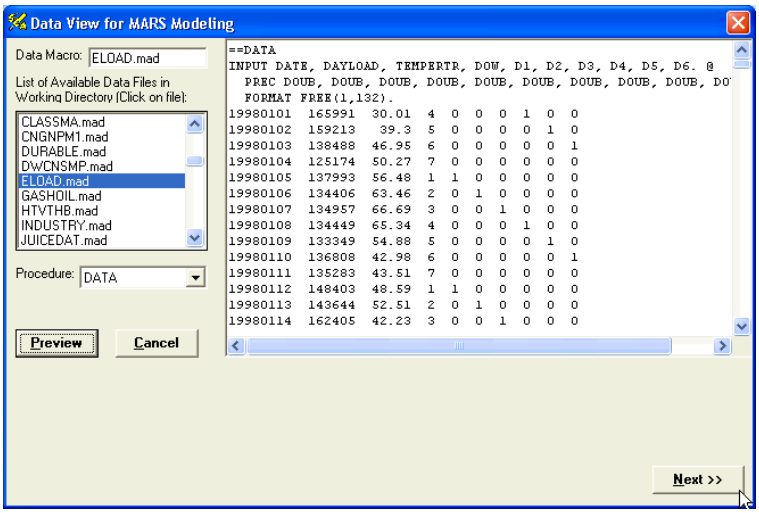
Daily total electricity load for residential customers is used to demonstrate Multivariate Adaptive Regression Spline (MARSPLINE) models in business forecasting solutions. If you are working through this example, the first step is to set the working directory by selecting the *System Profile* item under WorkBench's *System* menu. In the *Environment* tab of the *System Profile* dialog box, click on the *Browse* button associated with the working directory text box. Using the *Define Working Directory* dialog box, move to the **C:\SCA\TSData** directory that contains various sample data sets distributed with WorkBench. If WorkBench is installed under a different directory, the **TSData** subdirectory is located under the SCA installation directory. An example of the dialog boxes associated with modifying the working directory is shown below.



After the working directory is modified and the new profile is saved, click on the *Multivariate Regression Spline Modeling* item under the *Apps* menu to enter the graphical user interface for MARSPLINE modeling as demonstrated below.



Once you click on the *Multivariate Regression Spline Modeling* item, it is necessary to select the data to analyze. The *Data View for MARSPLINE Modeling* dialog box will automatically pop-up and display all SCA data macro files in the working directory. The daily electricity load data (DAYLOAD series) is located in the **ELOAD** data macro file under the **DATA** procedure. Please select this data set as illustrated below.



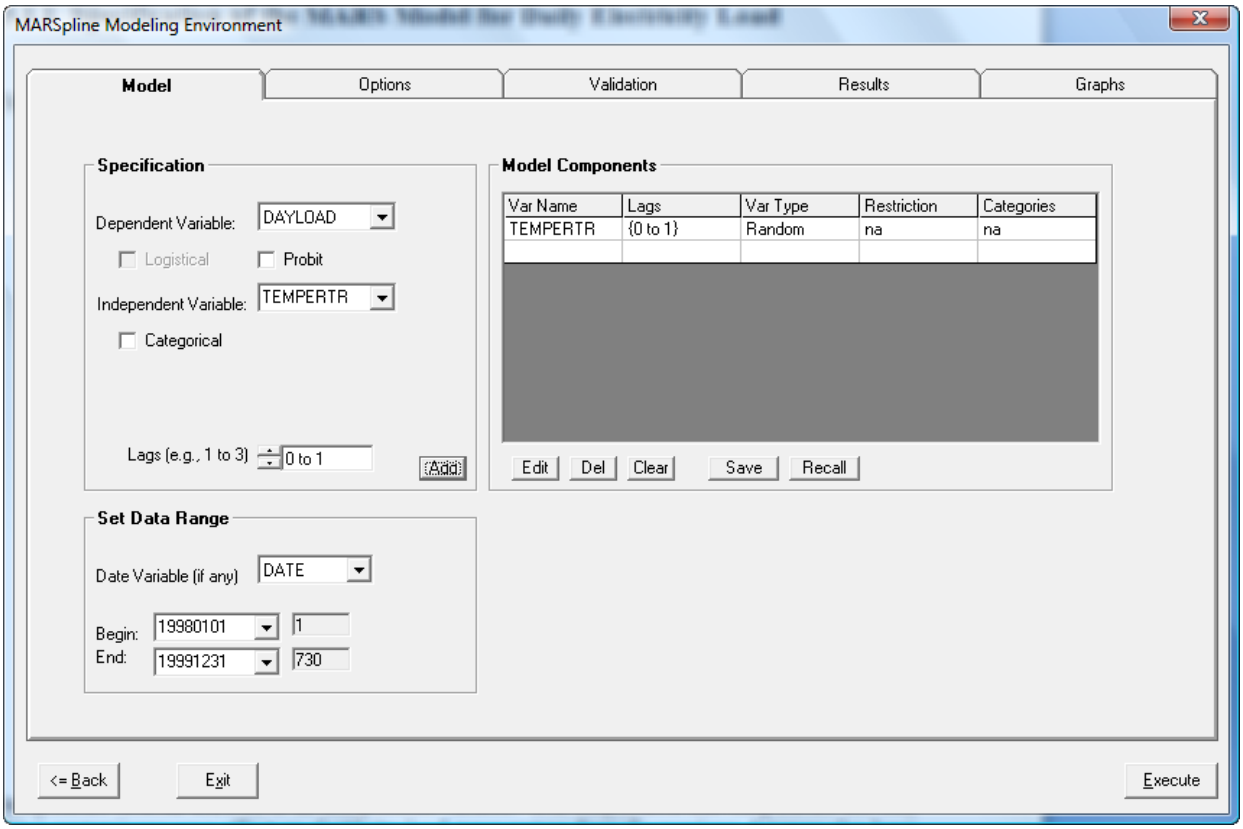
The data may be viewed by clicking on the ***Preview*** button. Here, we see that the ELOAD data macro contains ten variables:

DATE	→	YYYYMMDD
DAYLOAD	→	Daily total electricity load for residential customers
TEMPERTR	→	Average daily temperature
DOW	→	Day of week categorical variable (1=Monday, 2=Tuesday, etc.)
D1 to D6	→	Dummy variables for day of week (D1=Monday, D2=Tuesday, etc.)

In this example, we specify a MARSPLINE model for the DAYLOAD time series using average daily temperature as a predictor variable. Dummy variables D1-D6 are used as a categorical variables since electricity load tends to differ based on day of week. The DOW variable could also be used instead of the dummy variables in the MARSPLINE model. However, the OLS model estimation would not be valid since it would be treated as a random variable in the OLS model specification. Click on the *Next* button to enter the *MARSpline Modeling Environment*.

3.1.1 Specification of the MARSPLINE Model for Daily Electricity Load

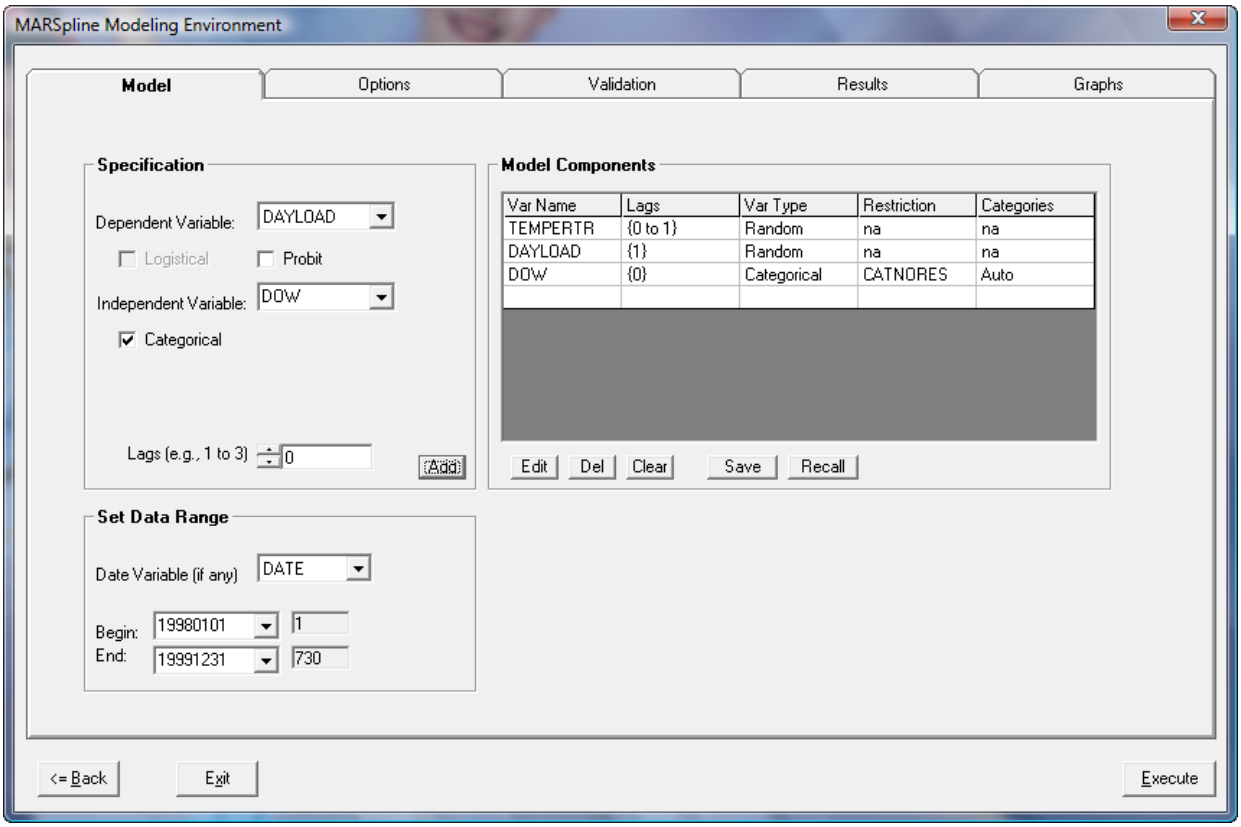
Once in the *MARSpline Modeling Environment*, click on the *Model* tab as shown below to specify a MARSPLINE model specification.



Use the drop down list box to select the dependent variable as **DAYLOAD**. Next, select temperature (**TEMPERTR**) as an independent variable. It is conjectured that temperature not only has a contemporaneous effect on electricity load, temperature from the previous day may also have some effect on electricity load. To accommodate a contemporaneous effect and lag effect, we specify "0 to 1" in the lags text box provided. Multiple lags may also be specified using commas. After specifying the **TEMPERTR** model component, click on the ***Add*** button to include it in the model. The model components are displayed in the Model Components grid. To modify or delete an existing model component, place your cursor on the grid row, and click on the ***Edit*** or ***Del*** buttons.

In addition to temperature, **DAYLOAD** at lag one is specified in the MARSPLINE model to partially offset the existence of autocorrelation from day to day electricity load. Finally, the **DOW** (day-of-week) categorical variables are included since day of week has a strong inference on electricity load (i.e., weekday and weekend patterns for electricity load differ considerably).

After specifying the MARSPLINE model for daily electricity load, the *Model* tab should like the one below:

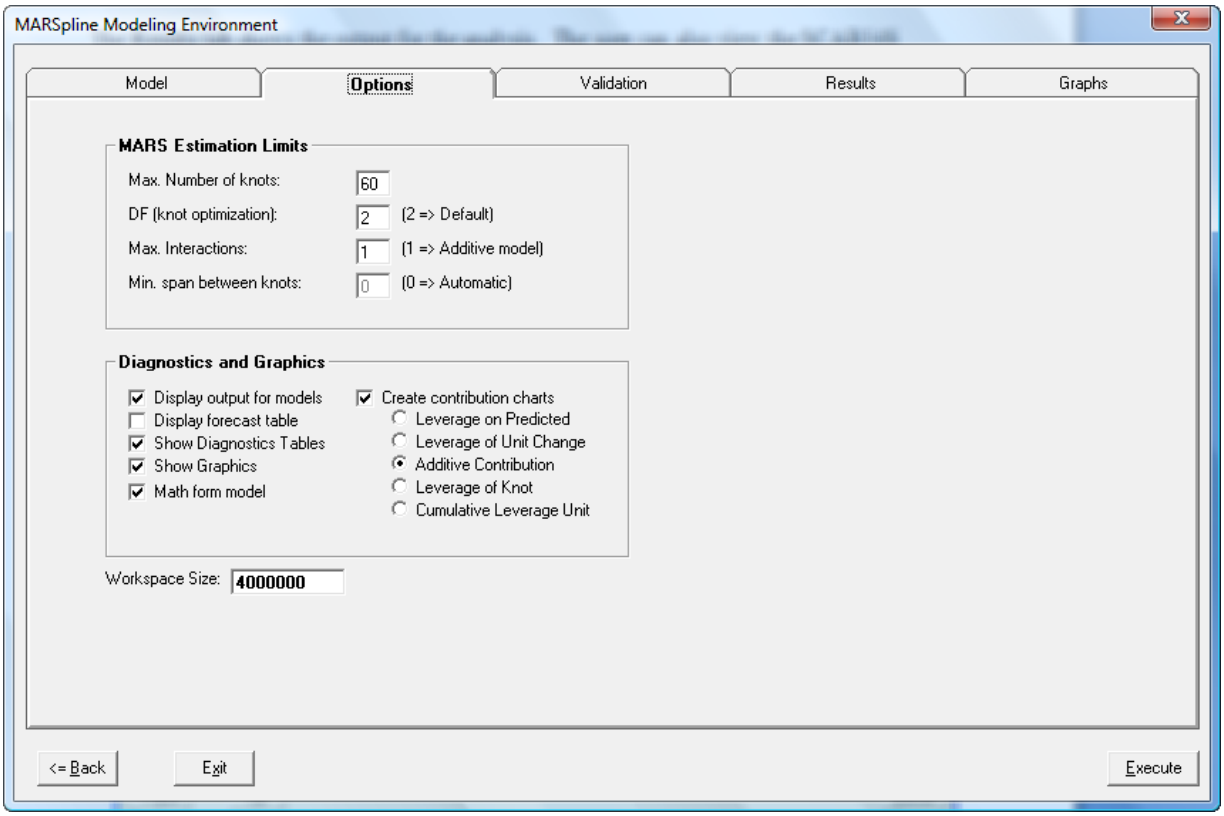


3.1.2 MARSPLINE Model Options for the Daily Electricity Load Example

After specifying an initial MARSPLINE model, click on the *Options* tab. Here, frequently used options to set estimation criteria in MARSPLINE model estimation are provided. This tab also allows the user to select the level of detail for estimation summaries, diagnostics, and graphics.

The maximum number of basis functions or knots allowed in the final MARSPLINE model has a default setting of five. It is recommended that the user experiment with this setting. For the daily electricity load example, set the maximum number of knots to 60 and the number of degrees of freedom for knot optimization to 2. To simplify the example, the variables will only be allowed to enter the model additively. Therefore, the maximum number of interactions per basis function should remain at its default value of 1. The above settings will produce a good overall fit for this example for both in-sample and out-of-sample periods.

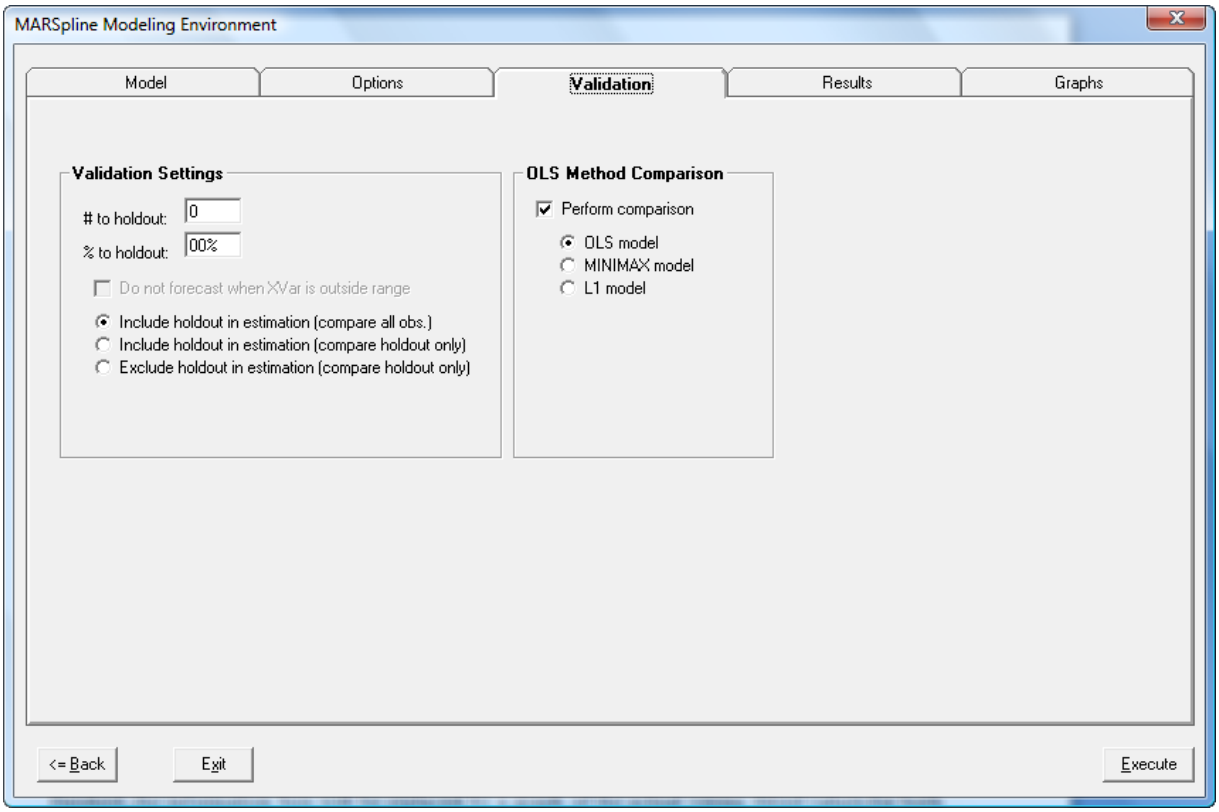
It is prudent to initially entertain a simpler model and add complexity later based on model validation and diagnostic tests.



3.1.3 MARSPLINE Model Validation for the Daily Electricity Load Example

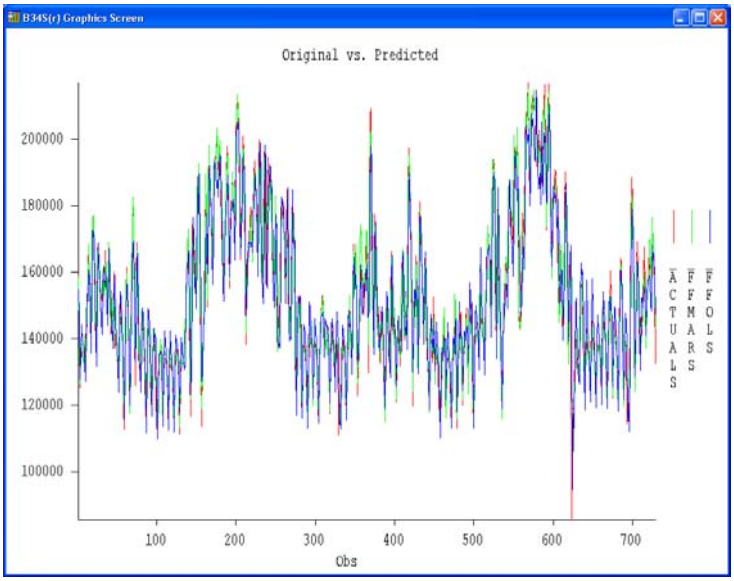
The *Validation* tab allows the user to evaluate the in-sample fit or out-of-sample predictive power of the MARSPLINE model compared to a linear regression model. As a first step, the overall in-sample fit of the MARSPLINE model will be examined and compared against a regression model that uses ordinary least squares (OLS) estimation.

Select the radio button that compares all observations and set the number of observation in the holdout period to zero as shown below. Next, click on the *Execute* button to run the analysis. The program script file will be automatically generated by SCA WorkBench and submitted to the SCAB34S engine for execution.



Once the SCAB34S engine is activated and the program script is loaded, an information box will appear on the computer screen indicating that calculations are in progress. When the analysis has finished, the information box will be replaced by a graph of the actual versus fitted values for both the MARSPLINE and OLS regression model.

An example of the graph is shown below. Click on the graph and it will disappear. However, this graph and other diagnostic graphs will be created in windows metafile format (WMF) and be saved to the user's working directory. The file names for the graphs are static and will be overwritten for each execution run. The plots can be viewed, printed, or saved under the *Graphs* tab.

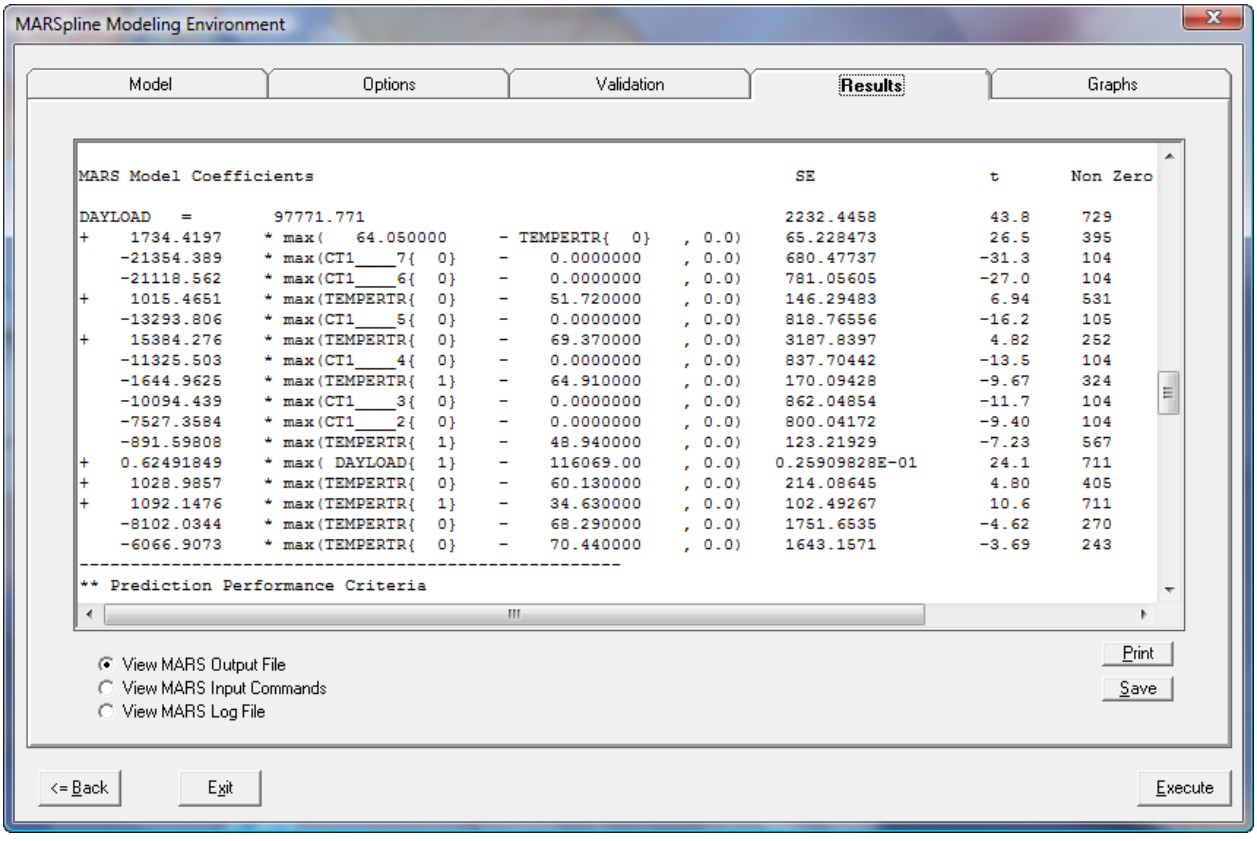


The _FFMARS series is the fitted values from the MARSPLINE model and the _FFOLS series is the fitted values from the linear regression model. It looks like both models have a decent fit, but it is not clear whether one model is much better than the other, at least from this graph. After clicking on the graph, the user will automatically be placed in the *Result* tab.

3.1.4 MARSPLINE Model Results for the Daily Electricity Load Example

The Results tab shows the output for the analysis. The user can also view the SCAB34S commands that were built by the SCA WorkBench interface. If desired, the input commands can be modified directly from the Results tab and re-submitted to SCAB34S SPLINES by clicking on the *Execute* button in the *Results* tab. The modified script can then be saved to a new file which can be executed from the System->Run SCAB34S Applet Program File menu.

Clicking on the *Execute* button from any other tab will trigger SCA WorkBench to rebuild the SCAB34S command script based on current settings in the *Model*, *Options*, and *Validation* tabs.



The output generated for a MARSPLINE analysis is fairly extensive and may be reduced by appropriate settings provided in the *Options* tab. The output always begins by summarizing the data series that has been brought into the workspace. This is followed by the OLS regression model summary, the MARSPLINE model details, a table of actual and predicted, followed by various diagnostic tests on the residuals (summary statistics, autocorrelation testing, nonlinear testing, etc.).

Below are segments of the output generated from the above analysis. Comments are also provided to help explain some of the results obtained. The first segment is a model summary of the

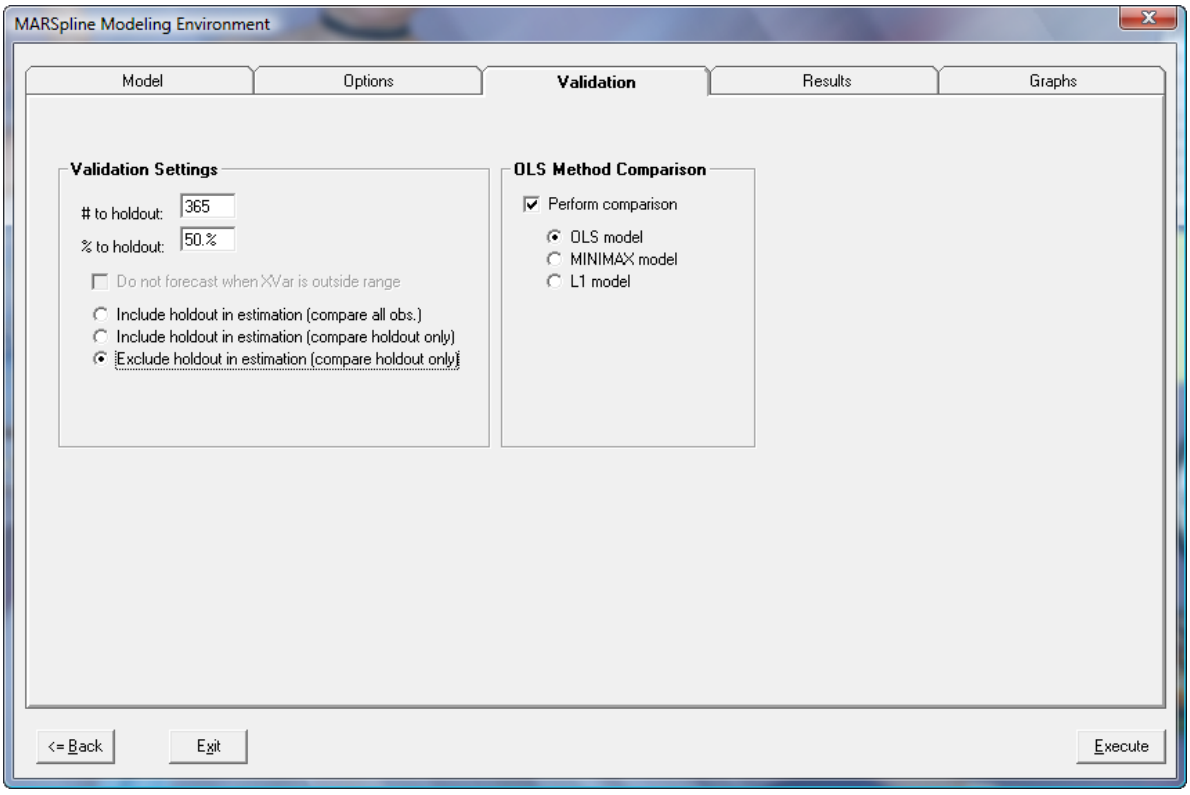
OLS regression model based on the same regressor variables included in the MARSPLINE model. The linear regression model is used as a benchmark in evaluating the performance gain of a non-linear MARSPLINE model. It also provides insight regarding the importance of the parameters being considered. For example, longer lags could be considered for the **TEMPERTR** variable and their significance examined using OLS. If additional lags are found to be significant, they can be introduced into the MARSPLINE model as well. In the OLS regression summary displayed, all model parameters considered are statistically significant.

```
*****************************************************
** Analysis Performed on Variable: DAYLOAD
*****************************************************

Categorical variable  DOW      split into 0-1 variables

Categorical  Value   Reference    Cases-On
 DOW           1.0    CT1____1         *** dropped in model
 DOW           2.0    CT1____2         104
 DOW           3.0    CT1____3         104
 DOW           4.0    CT1____4         105
 DOW           5.0    CT1____5         105
 DOW           6.0    CT1____6         104
 DOW           7.0    CT1____7         104

Final Model Estimation Summary -  OLS
Dependent Variable:               DAYLOAD
-----------------------------------------------------------------
  # Variable   Lag     Coefficient      std.Err      t-value     Comments
-----------------------------------------------------------------
  1 TEMPERTR     0        -394.922       70.404       -5.609
  2 TEMPERTR     1         446.461       69.130        6.458
  3 DAYLOAD      1           0.897        0.017       51.765
  4 CT1____2     0      -13094.057     1350.394       -9.696
  5 CT1____3     0      -15689.850     1364.489      -11.499
  6 CT1____4     0      -17482.286     1363.351      -12.823
  7 CT1____5     0      -18937.460     1353.272      -13.994
  8 CT1____6     0      -25767.282     1345.041      -19.157
  9 CT1____7     0      -22905.939     1319.950      -17.354
 10 CONSTANT     0       28844.713     2478.529       11.638

Number of observations:                    729
Adjusted R-Square:                       0.838
Sum of Squared Residuals:            0.647E+11
Schwartz Information Criteria:       15483.108


Multivariate Autoregressive Splines Analysis
Routines Based on Hastie-Tibshirani GPL routines
Version - 1 March 2006.

Left Hand Side Variable                         DAYLOAD
Penalty cost per degree of freedom            2.000
Threshold for Forward stepwise Stopping   0.1000E-03
Rank Test Tolerance                       0.1000E-12
Max # of Knots  (nk)                       60
Max interaction (mi)                       1
Number of Observations                     729
Number of right hand Variables             9


GCV with only the constant                 555487921.7267956
Total sum of squares                       403840481081.5556
Final gcv                                  24180800.39781882
Variance of Y Variable                     554725935.5515872
R**2 (1 - (var(res)/var(y)))               0.9602120308565458
Residual Sum of Squares                    16067992600.15061
Residual Variance                          22071418.40680029
Residual Standard Error                    4698.022818888845
Sum Absolute Residuals                     2321732.022322142
Max Absolute Residual                      48191.94571455222
# of coefficients after last fwd step      17
```

```
MARSPLINE Model Coefficients                                                  SE              t     --Non Zero---  Importance  Coef#
                                                                                                      #         %        %
DAYLOAD   =        97771.771                                                2232.4458          43.8     729      100.0              1
+    1734.4197    * max(   64.050000     - TEMPERTR{  0}   , 0.0)    65.228473          26.5     395       54.      84.7     2
    -21354.389    * max(CT1____7{  0}    -    0.0000000    , 0.0)    680.47737         -31.3     104       14.     100.0     3
    -21118.562    * max(CT1____6{  0}    -    0.0000000    , 0.0)    781.05605         -27.0     104       14.      86.2     4
+    1015.4651    * max(TEMPERTR{  0}    -    51.720000    , 0.0)    146.29483          6.94     531       72.      22.1     5
    -13293.806    * max(CT1____5{  0}    -    0.0000000    , 0.0)    818.76556         -16.2     105       14.      51.7     6
+    15384.276    * max(TEMPERTR{  0}    -    69.370000    , 0.0)    3187.8397          4.82     252       34.      15.4     7
    -11325.503    * max(CT1____4{  0}    -    0.0000000    , 0.0)    837.70442         -13.5     104       14.      43.1     8
    -1644.9625    * max(TEMPERTR{  1}    -    64.910000    , 0.0)    170.09428         -9.67     324       44.      30.8     9
    -10094.439    * max(CT1____3{  0}    -    0.0000000    , 0.0)    862.04854         -11.7     104       14.      37.3    10
    -7527.3584    * max(CT1____2{  0}    -    0.0000000    , 0.0)    800.04172         -9.40     104       14.      30.0    11
    -891.59808    * max(TEMPERTR{  1}    -    48.940000    , 0.0)    123.21929         -7.23     567       77.      23.1    12
+   0.62491849    * max( DAYLOAD{  1}    -    116069.00    , 0.0)   0.25909828E-01      24.1     711       97.      76.9    13
+    1028.9857    * max(TEMPERTR{  0}    -    60.130000    , 0.0)    214.08645          4.80     405       55.      15.3    14
+    1092.1476    * max(TEMPERTR{  1}    -    34.630000    , 0.0)    102.49267          10.6     711       97.      34.0    15
    -8102.0344    * max(TEMPERTR{  0}    -    68.290000    , 0.0)    1751.6535         -4.62     270       37.      14.7    16
    -6066.9073    * max(TEMPERTR{  0}    -    70.440000    , 0.0)    1643.1571         -3.69     243       33.      11.8    17

-----------------------------------------------------
** Prediction Performance Criteria
-----------------------------------------------------
OLS RMSE:         9421.463493903346
OLS MAPE:            4.515850911762661
MARSPLINE RMSE:   4694.799475237409
MARSPLINE MAPE:      2.150059820039957

Descriptive Statistics Table -  MARSPLINE Residuals
----------------------------------------------
Number of Cases     :              729
Minimum Value       :    -14772.058828
Maximum Value       :     48191.945715
Mean                :         0.000000
Standard Deviation  :      4698.022819
Skewness            :         2.400009
Kurtosis            :        19.117665
Cumulant (6th Order):      1441.499026
First Quartile      :     -2750.059846
Third Quartile      :      2089.711367
```

It is clear that the MARSPLINE model exhibits a significantly better fit compared to the OLS model. Under the section of the output labeled as “Prediction Performance Criteria”, the root mean square error (RMSE) of the residuals for the MARSPLINE model is much smaller than the OLS regression model. This may be deceptive though since we are evaluating the in-sample fit of the data and it is quite possible that the MARSPLINE model is over-fitting the data.

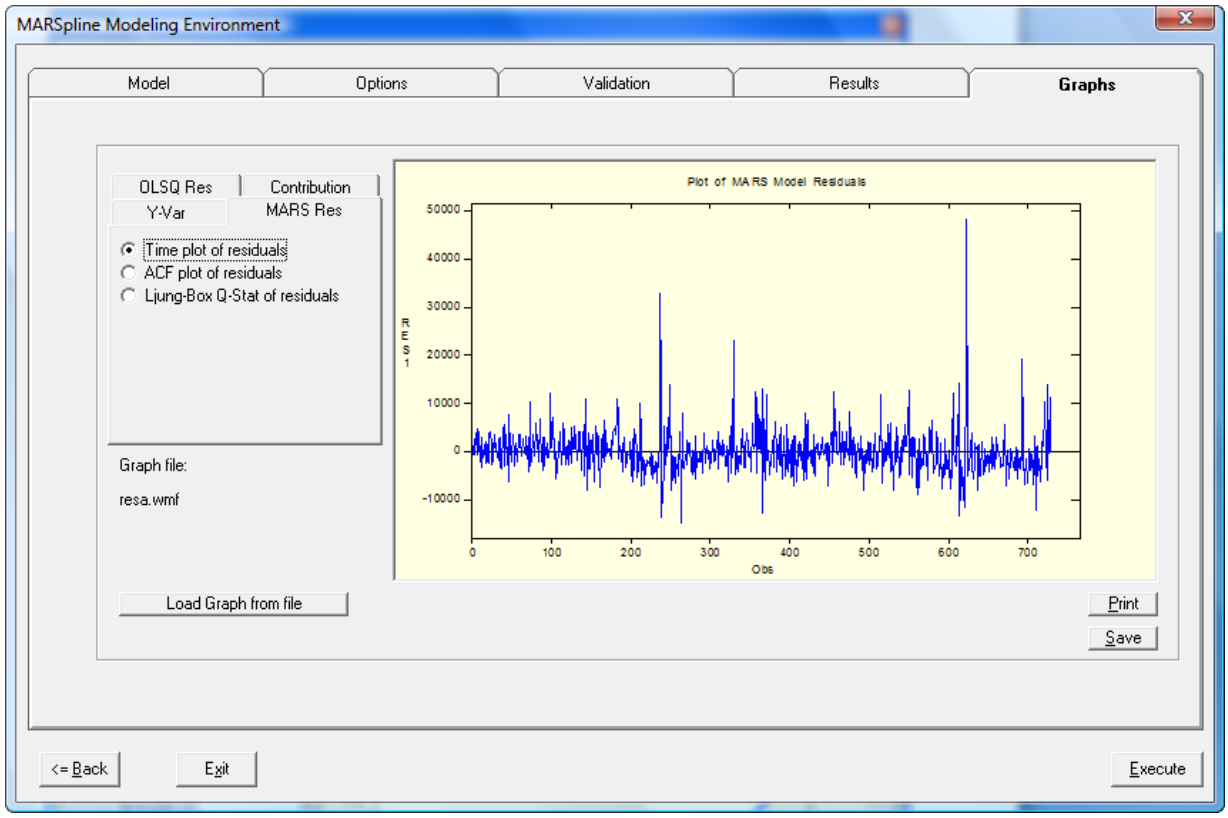
To examine the out-of-sample predictive performance of the models, a percentage of the data should be retained for post-sample forecast comparison. To accomplish this, go back to the *Validation* tab. The last half of the data will be retained as the post-sample period.

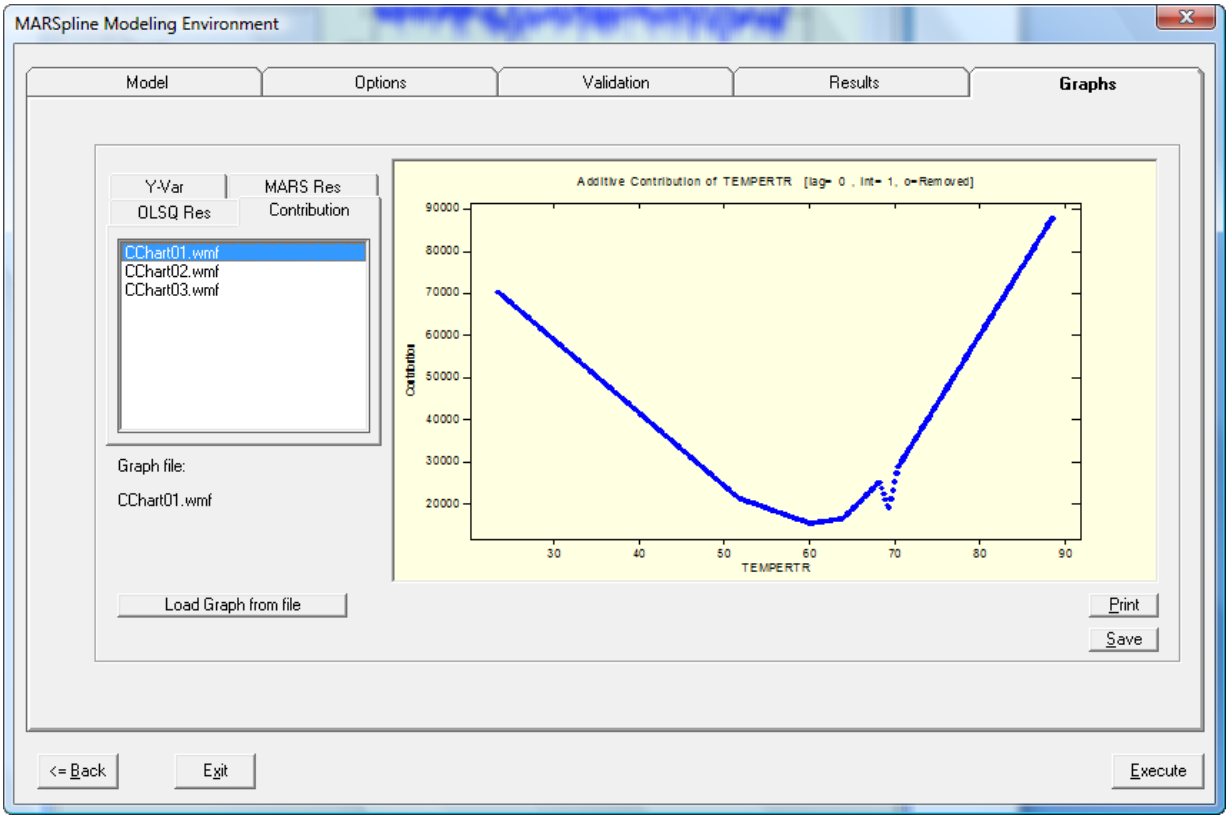


After changing the settings on the *Validation* tab, click on the *Execute* button to perform the analysis. After the analysis has completed, the performance criteria can once again be evaluated to determine if the model's true predictive power holds for the holdout period. Since the holdout sample is rather large in this example, we can be assured that outliers in the data will not severely skew our conclusions. The excerpt from the output is displayed below.

```
-----------------------------------------------------
** Prediction Performance Criteria
-----------------------------------------------------
OLS RMSE:          10236.36655180456
OLS MAPE:              4.949096693159521
MARSPLINE RMSE:    5491.451907109587
MARSPLINE MAPE:        2.505546603055314
```

We can conclude that the MARSPLINE modeling approach is a viable approach to predict daily electricity load based on changing levels of temperature, day of week, and observed load from the previous period. The model can be further refined to include dummy variables for seasons (Winter, Spring, Summer and Fall) and known events (holidays, power grid failures, etc.). An important tool to lend direction to possible model improvements is to examine residual graphs and diagnostics. Several diagnostic tables have been presented in the above output summary. By clicking on the *Graph* tab, the user can examine various graphs including a time plot of the dependent variable, actual versus predicted, time plots of residuals, residual autocorrelation function, Q-Stat charts, and various contribution charts. Two of these charts are displayed next.


MARSpline Modeling Environment
Model
Options
Validation
Results
Graphs
OLSQ Res
Contribution
Y-Var
MARS Res
Time plot of residuals
ACF plot of residuals
Ljung-Box Q-Stat of residuals
Graph file:
resa.wmf
Load Graph from file
Plot of MARS Model Residuals
50000
40000
30000
20000
10000
0
-10000
0
100
200
300
400
500
600
700
Obs
Print
Save
<= Back
Exit
Execute


MARSpline Modeling Environment
Model
Options
Validation
Results
Graphs
Y-Var
MARS Res
OLSQ Res
Contribution
CChart01.wmf
CChart02.wmf
CChart03.wmf
Graph file:
CChart01.wmf
Load Graph from file
Additive Contribution of TEMPERTR [lag= 0 , Int= 1, o=Removed]
90000
80000
70000
60000
50000
40000
30000
20000
Contribution
30
40
50
60
70
80
90
TEMPERTR
Print
Save
<= Back
Exit
Execute

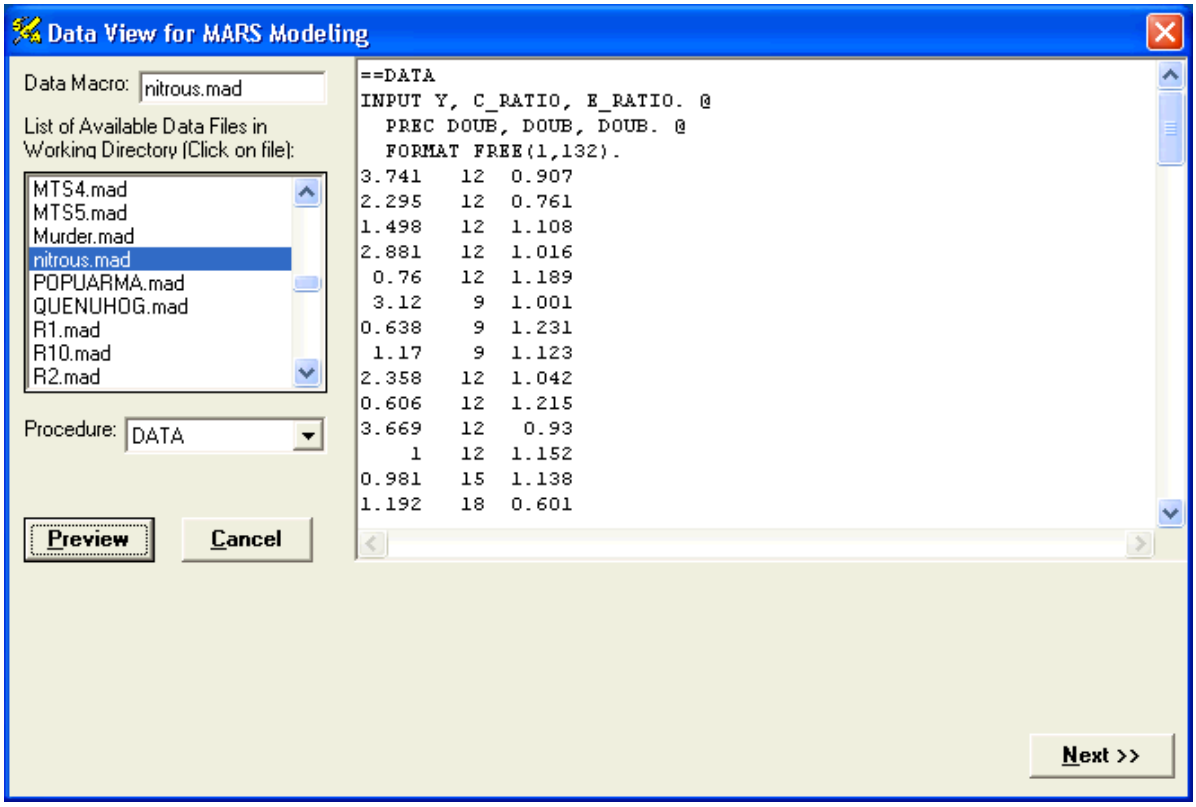
3.2 Modeling Nitrous Oxide Production Using MARSPLINE

At room temperature, nitrous oxide, N2O, is quite non-reactive with most substances, including alkali metals, halogens, and even ozone. It is therefore widely used as a propellant in aerosol cans in place of the CFCs which can damage the ozone layer. It is also used as a safe analgesic vapor for minor medical procedures by relaxing the patient and taking the edge off pain sensory functions. This vapor has become to be known as "Laughing Gas".

When heated sufficiently, however, N2O decomposes exothermically to N2 and O2. If this reaction occurs in the combustion chamber of an automobile, 3 moles of gas would be produced from 2 moles, providing extra power to the piston. In addition, the increased oxygen from the process provides more efficient combustion of fuel, the nitrogen buffers the increased cylinder pressure controlling the combustion, and the latent heat of vaporization of the N2O reduces the intake temperature. Therefore N2O is occasionally injected into the fuel lines of racing cars to give more power to the engine and to give the car exceptional acceleration.

Breiman (1991) analyzed the production of nitrous oxide in a single-cylinder engine powered by ethanol using equivalence ratio and compression ratio of the engine as regressor variables. The data was also originally analyzed by Cleveland and Devlin (1988).

The nitrous oxide data (Y series) is located in the **NITROUS** data macro file under the **DATA** procedure. Please select this data set as illustrated below.



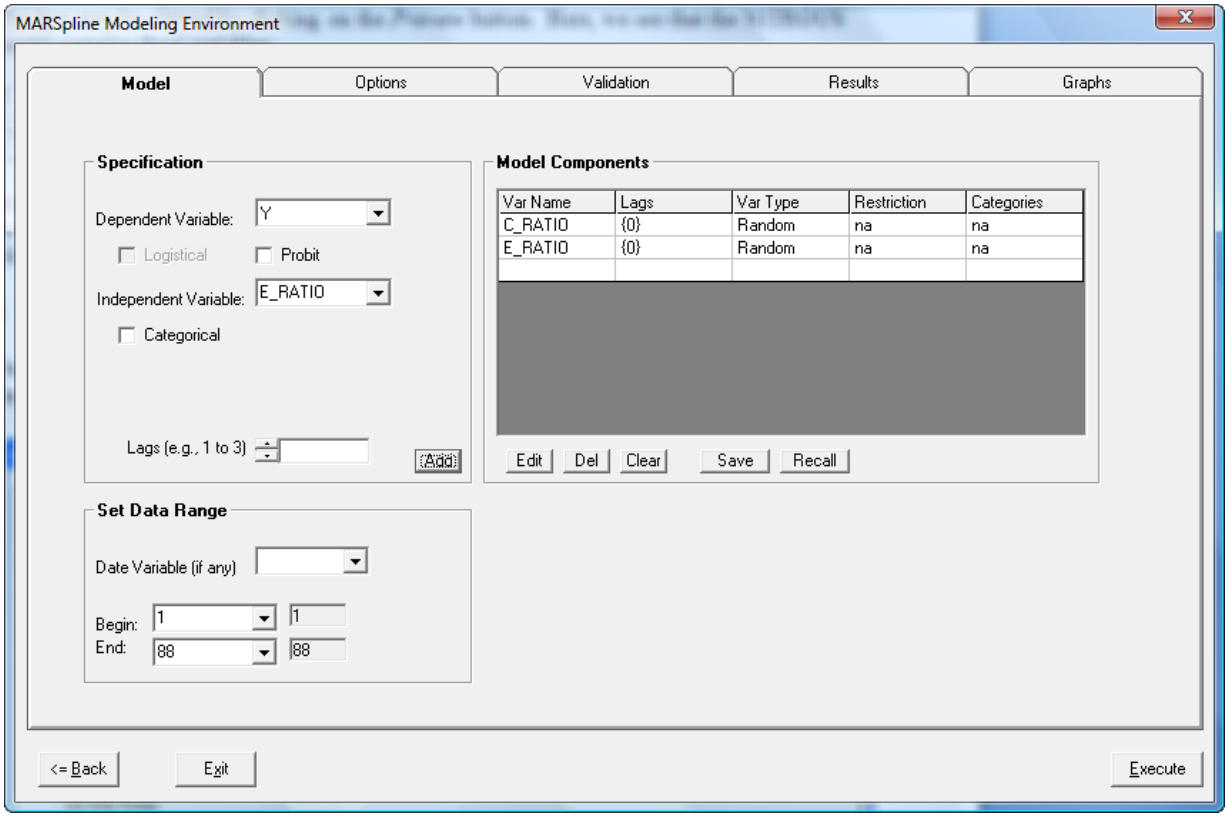
The data may be viewed by clicking on the *Preview* button. Here, we see that the NITROUS data macro contains three variables:

Y	→	Nitrous oxide measures
E_RATIO	→	Equivalence ratio
C_RATIO	→	Compression ratio

In this example, we specify a MARSPLINE model for nitrous oxide based on the contemporaneous effects of the equivalence and compression ratios. Click on the *Next* button to enter the *MARSPLINE Modeling Environment.*

3.2.1 Specification of the MARSPLINE Model for Nitrous Oxide Production

Once in the *MARSPLINE Modeling Environment*, click on the *Model* tab to specify the model components.

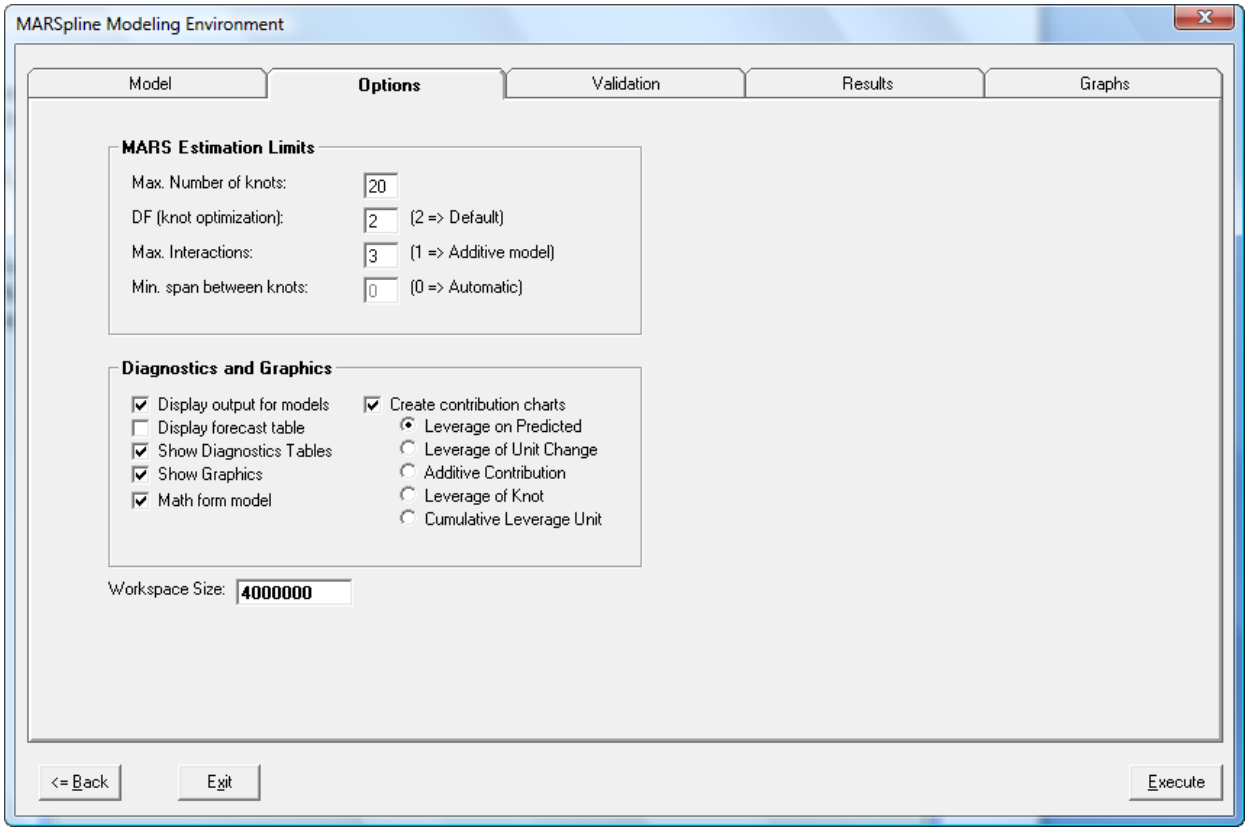


Use the drop down list box to select the dependent variable as **Y**. Next, select compression ratio (**C_RATIO**) as an independent variable. To specify a contemporaneous effect for compression ratio, type "0" in the *Lags* text box provided. If you leave the *Lags* text box empty a contemporaneous effect will be assumed. After specifying the **C_RATIO** model component, click on the *Add* button to include it in the model. The model components are displayed in the Model Components grid. To modify or delete an existing model component, place your cursor on the grid row, and click on the *Edit* or *Del* buttons.

In addition to compression ratio, specify the equivalence ratio variable **E_RATIO** in the same manner. After specifying the MARSPLINE model for nitrous oxide, the *Model* tab should like the one above.

3.2.2 MARSPLINE Model Options for Nitrous Oxide Production Example

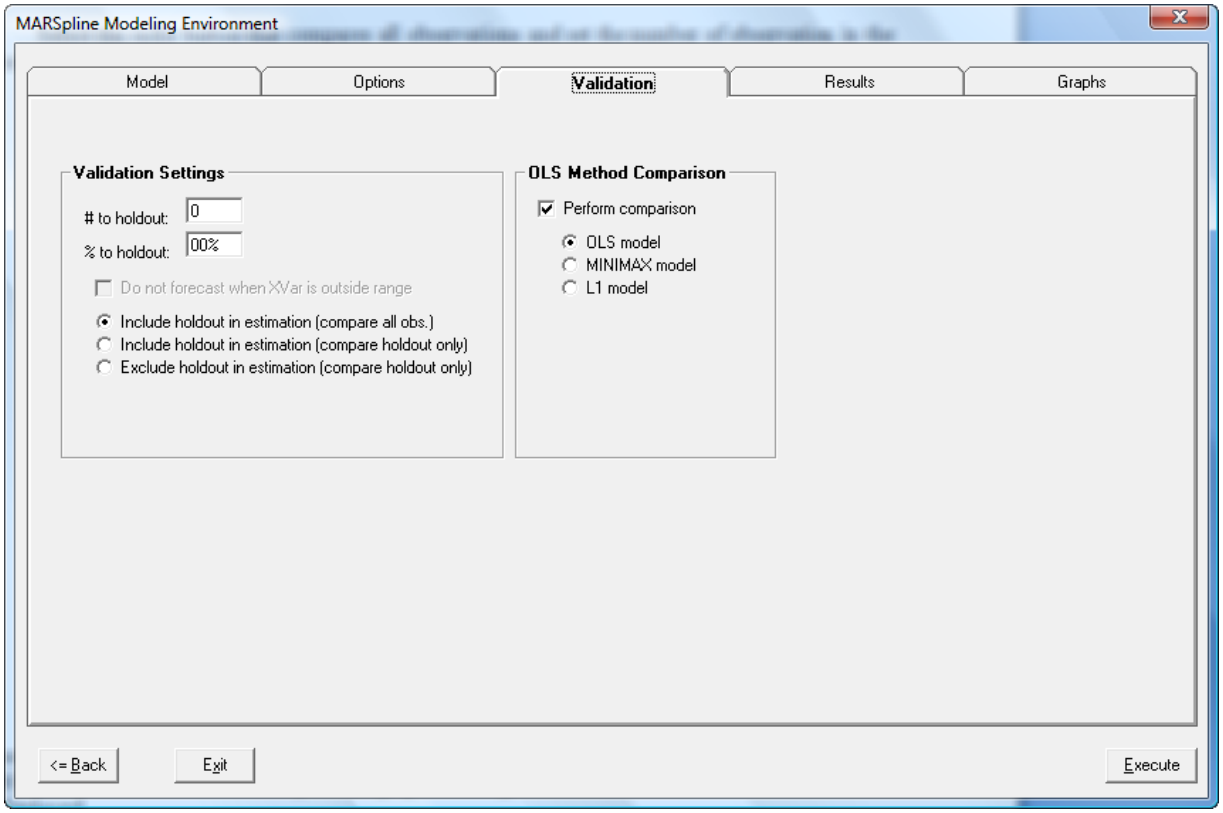
After specifying the MARSPLINE model, click on the *Options* tab to set the maximum number of basis functions to 25 and the maximum number of interactions allowed to 3. By allowing interactions, the MARSPLINE model will determine if different blends of compression ratio and equivalence ratio levels impact nitrous oxide production.



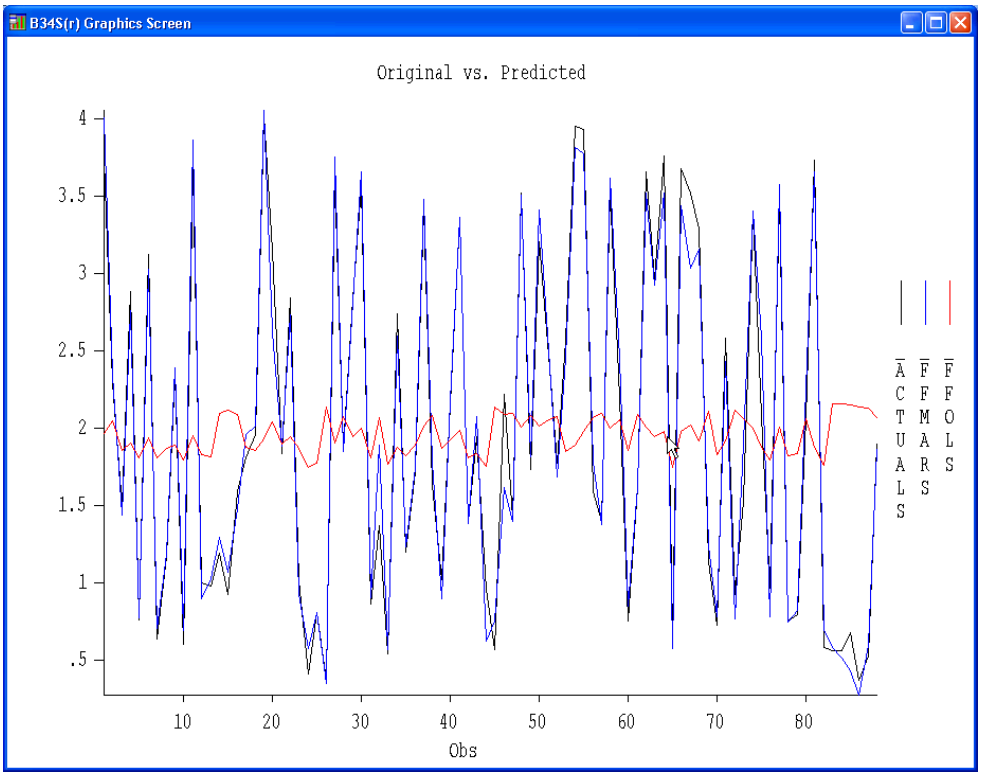
3.2.3 MARSPLINE Model Validation for Nitrous Oxide Production Example

The *Validation* tab allows the user to evaluate the in-sample fit or out-of-sample predictive power of the MARSPLINE model compared to a linear regression model. As a first step, the overall in-sample fit of the MARSPLINE model will be examined and compared against a regression model that uses ordinary least squares (OLS) estimation.

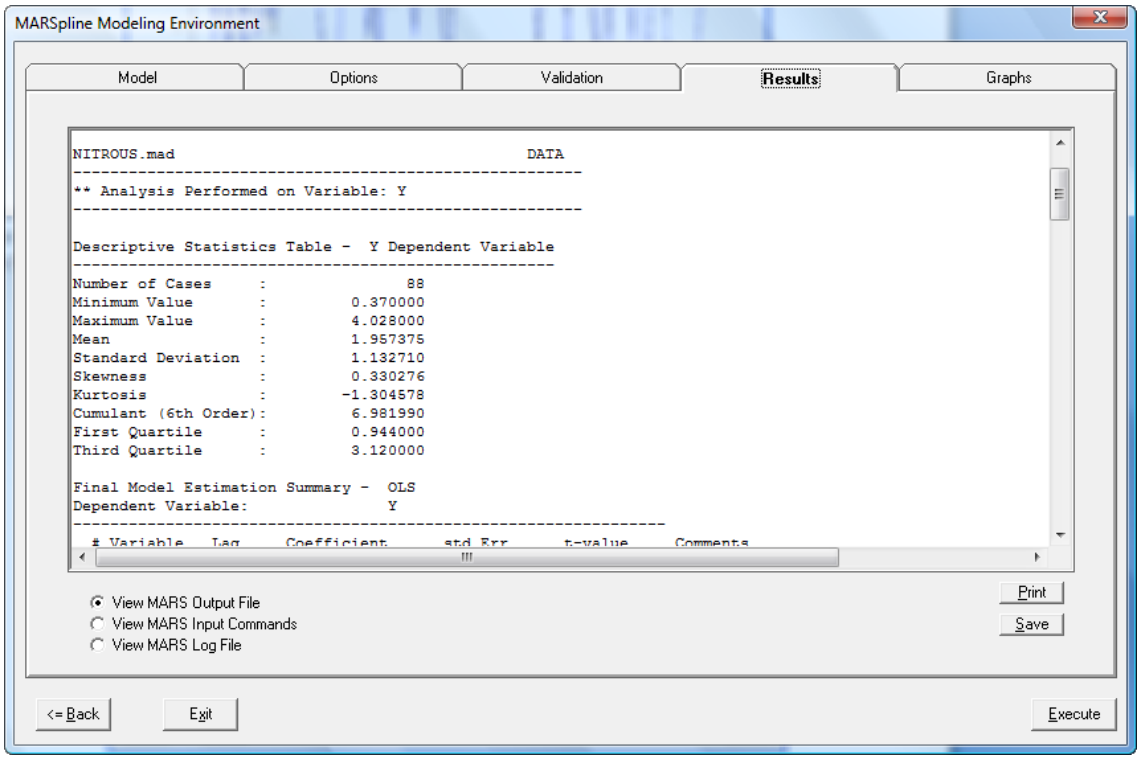
Select the radio button that compares all observations and set the number of observation in the holdout period to zero as shown below.



Next, click on the *Execute* button to run the analysis. The program script file will be automatically generated by SCA WorkBench and submitted to the SCAB34S engine for execution. When the MARSPLINE estimation completes, a graph showing the actual and fitted values will be displayed.



Examining this graph clearly shows that MARSPLINE fits the data very well and that linear regression methods perform poorly. It would seem that nitrous oxide production responds in a nonlinear fashion to levels of compression and equivalence ratios. To view the output, first click on the graph to close it focus will automatically be set to the *Results* tab as shown below.



Use the scroll bar on the right-hand side of the text box to review the detailed output from the execution run. The sample output below has been reduced for brevity. Here, if the OLS regression model summary is examined, the C_RATIO and E_RATIO variables are not significant, leaving the constant as the only significant component in the model. Our conclusion is that nitrous oxide production can not be explained adequately as a linear process using compression and equivalence ratios as predictor variables. However, nitrous oxide production can be explained as a nonlinear process based on varied levels of compression and equivalence ratios; and their interaction. This was substantiated by visually examining the fit of the OLS regression model in the previous graph.

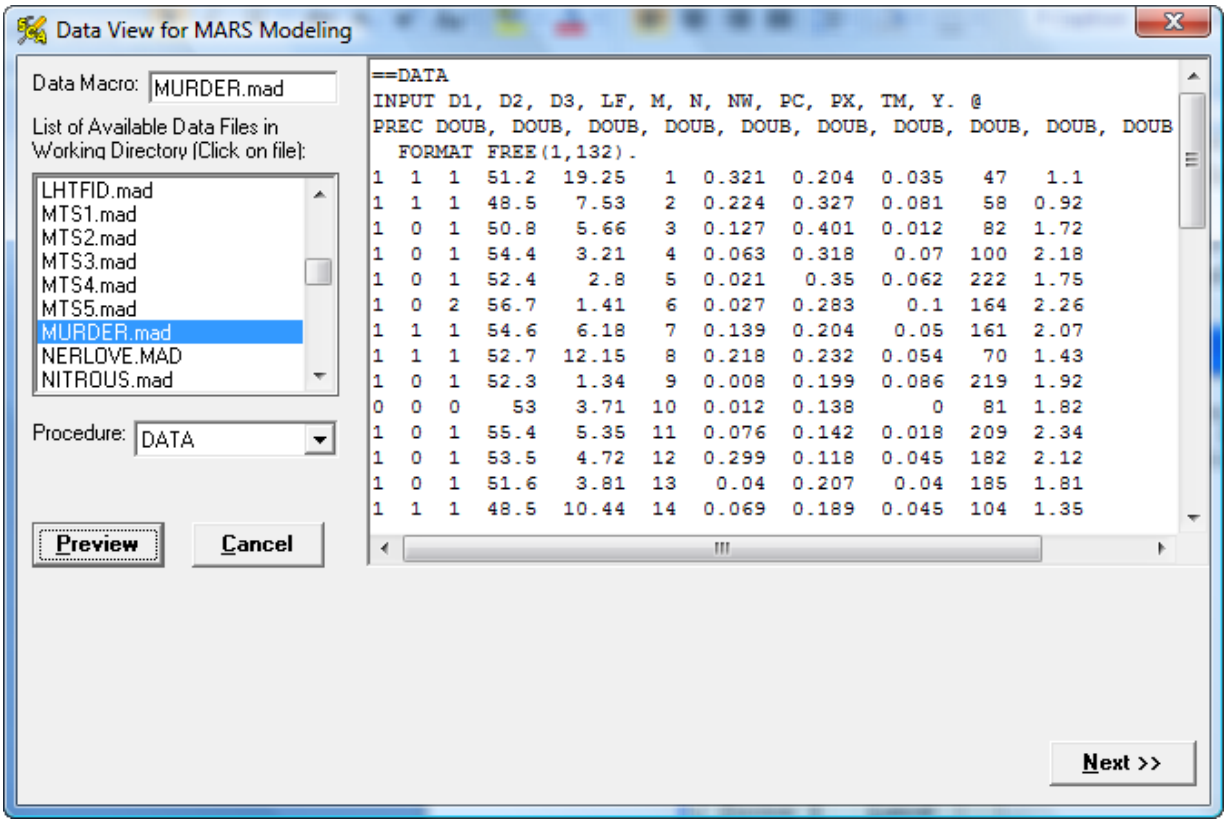
```
-------------------------------------------------------
** Analysis Performed on Variable: Y
-------------------------------------------------------

Descriptive Statistics Table -  Y Dependent Variable
-------------------------------------------------
Number of Cases     :               88
Minimum Value       :         0.370000
Maximum Value       :         4.028000
Mean                :         1.957375
Standard Deviation  :         1.132710
Skewness            :         0.330276
Kurtosis            :        -1.304578
Cumulant (6th Order):         6.981990
First Quartile      :         0.944000
Third Quartile      :         3.120000

Final Model Estimation Summary -  OLS
Dependent Variable:               Y
-----------------------------------------------------------------
  # Variable   Lag     Coefficient      std.Err      t-value     Comments
-----------------------------------------------------------------
  1 C_RATIO      0          -0.007        0.031       -0.228
  2 E_RATIO      0          -0.557        0.601       -0.926
  3 CONSTANT     0           2.559        0.662        3.863

Number of observations:                     88
Adjusted R-Square:                      -0.012
Sum of Squared Residuals:              110.
Schwartz Information Criteria:          287.600


Multivariate Autoregressive Splines Analysis
Routines Based on Hastie-Tibshirani GPL routines
Version - 1 March 2006.

Left Hand Side Variable                          Y
Penalty cost per degree of freedom          2.000
Threshold for Forward stepwise Stopping   0.1000E-03
Rank Test Tolerance                       0.1000E-12
Max # of Knots  (nk)                        20
Max interaction (mi)                        3
Number of Observations                      88
Number of right hand Variables              2
tolbx    set as                             1.000000000000000E-09
stopfac  gcv/gcvnull > stopfac => stop      10.00000000000000
prevcrit set as                             10000000000.00000

Series   Lag  Mean          Max           Min
C_RATIO   0      12.03         18.00          7.500
E_RATIO   0     0.9265         1.232         0.5350

GCV with only the constant                  1.297779827057735
Total sum of squares                        111.6238126250000
Final gcv                                   4.980617316973167E-02
Variance of Y Variable                      1.283032329022989
R**2 (1 - (var(res)/var(y)))                0.9758597462229390
Residual Sum of Squares                     2.694627164330599
Residual Variance                           3.097272602678850E-02
Residual Standard Error                     0.1759906986939608
```

```
Sum Absolute Residuals                      10.75317643991678
Max Absolute Residual                       0.6017260845076515
# of coefficients after last fwd step       10

MARSPLINE Model Coefficients                                                SE              t     --Non Zero---  Importance  Coef#
                                                                                                  #         %        %
Y         =        4.5279553                                        0.83783455E-01       54.0      88      100.0              1
    -12.314783    * max(  0.90700000     -  E_RATIO{  0}   , 0.0)   0.37832170          -32.5      41       46.     100.0     2
   -0.12798485    * max(   15.000000     -  C_RATIO{  0}   , 0.0)   0.10927898E-01      -11.7      53       60.      36.0     3
+   0.68449752    * max(   15.000000     -  C_RATIO{  0}   , 0.0)   0.21982449           3.11       7        8.       9.6     4
                  * max(  0.71500000     -  E_RATIO{  0}   , 0.0)
+    11.088234    * max( E_RATIO{  0}    -    1.1440000    , 0.0)    1.8244547           6.07      16       18.      18.7     5
    -21.967842    * max( E_RATIO{  0}    -   0.99000000    , 0.0)    5.7715069          -3.80      40       45.      11.7     6
    -7.1267043    * max( E_RATIO{  0}    -   0.86800000    , 0.0)    1.1136820          -6.39      52       59.      19.7     7
+    13.768145    * max( E_RATIO{  0}    -    1.0160000    , 0.0)    5.4674066           2.51      34       38.       7.7     8
+   0.58145323    * max( C_RATIO{  0}    -    15.000000    , 0.0)   0.13338996           4.35       6        6.      13.4     9
                  * max(  0.90700000     -  E_RATIO{  0}   , 0.0)
+   0.32277939    * max(   15.000000     -  C_RATIO{  0}   , 0.0)   0.40592246E-01       7.95      39       44.      24.4    10
                  * max( E_RATIO{  0}    -   0.76500000    , 0.0)

-----------------------------------------------------
** Prediction Performance Criteria
-----------------------------------------------------
OLS RMSE:          1.120070955746104
OLS MAPE:         82.75249532340079
MARSPLINE RMSE:    0.1749878945271059
MARSPLINE MAPE:    8.343364824891751
```

3.3 Modeling Capital Punishment as a Deterrent to Murder Using MARS-Probit

McManus (1985) studied the use of capital punishment as a deterrent to murder using a logistic regression approach. In this example, we extend the use of logistic regression to MARS-Probit to evaluate the presence of thresholds and/interactions in the data that may provide further insight into the McManus study.

The McManus data are located in the **MURDER** data macro file under the **DATA** procedure. Please select this data set as illustrated below.



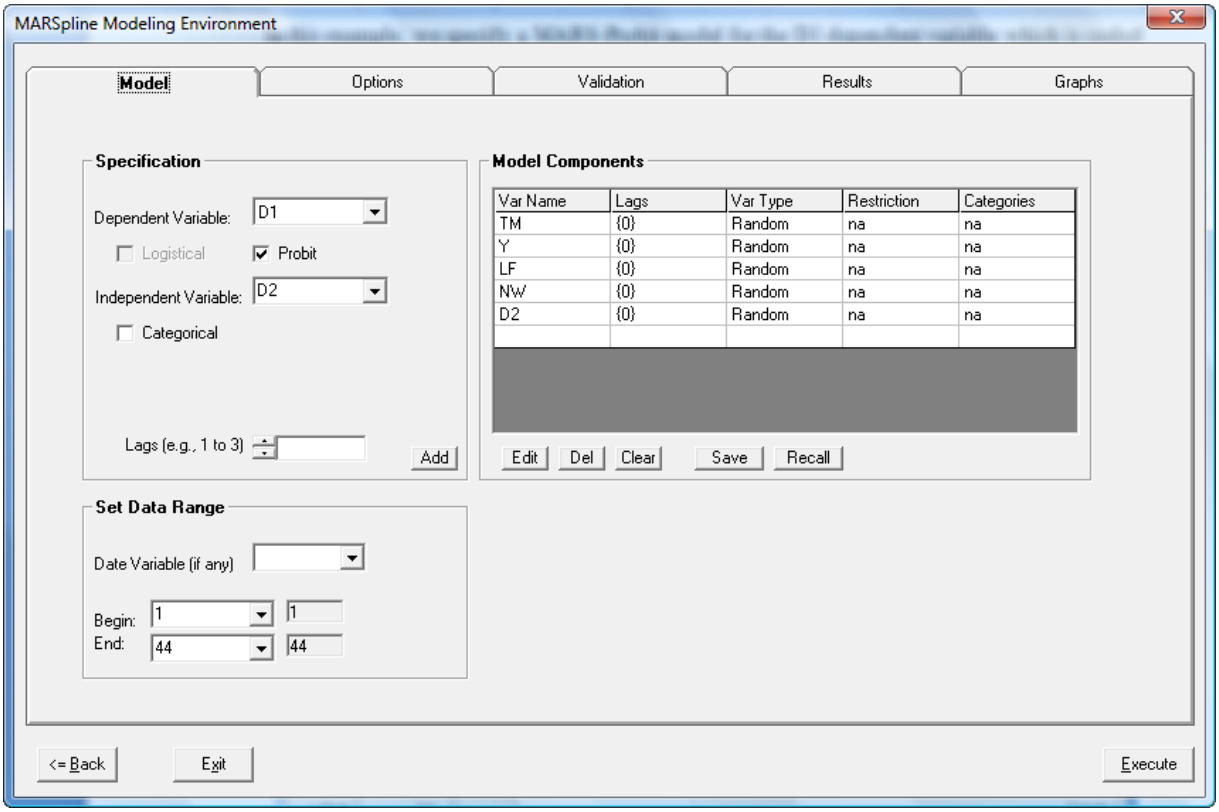
The data may be viewed by clicking on the *Preview* button. Here, we see that the MURDER data macro contains several variables:

N	→	Observation number
M	→	Murder rate per 100,000 in year 1950
PC	→	Number of convictions / Number of murders
PX	→	Number of executions from 1946-1950 / Number of convictions
Y	→	Median family income (1949) / 1000
LF	→	Labor force participation as a percent in 1950
NW	→	Proportion of population that is non-white
D2	→	Dummy variable (1=southern state)
D1	→	Dummy variable (1=state has capital punishment policy)
TM	→	Median time served by convicted murderers
D3	→	Indicator variable (0=>D1=0, D3=1,1=>D1=1,2+>PX GE .1

In this example, we specify a MARS-Probit model for the D1 dependent variable which is coded as 1 for states that have capital punishment and 0 otherwise. Click on the *Next* button to enter the *MARSPLINE Modeling Environment*.

3.3.1 Specification of the MARS-Probit Model for the McManus Data

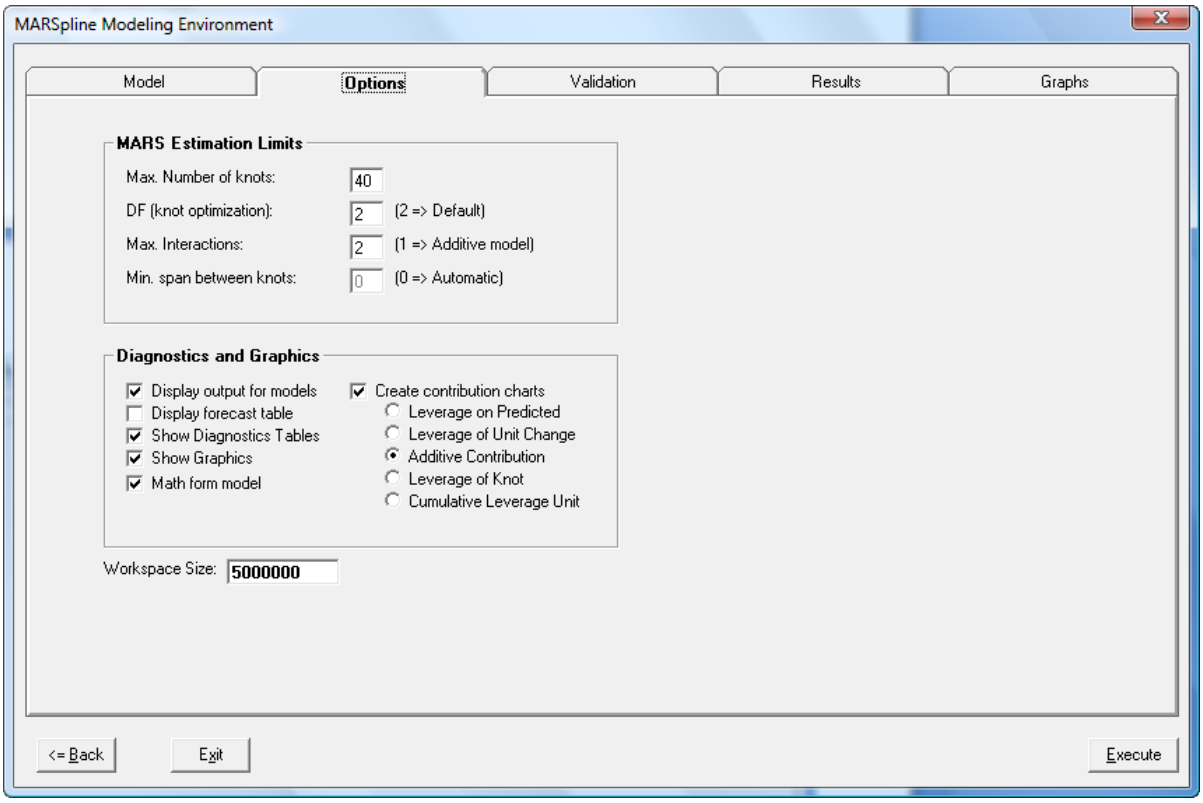
Once in the *MARSpline Modeling Environment*, click on the *Model* tab to specify the model components.



Use the drop down list box to select the dependent variable as **D1**. Next, select median time served (**TM**) as an independent variable. Clicking on the Add button will add the component to the model. Repeat this for the variable **Y**, **LF**, **NW,** and **D2**. After specifying the model for D1, the *Model* tab should look like the one above.

3.3.2 MARS-Probit Model Options for McManus Data Example

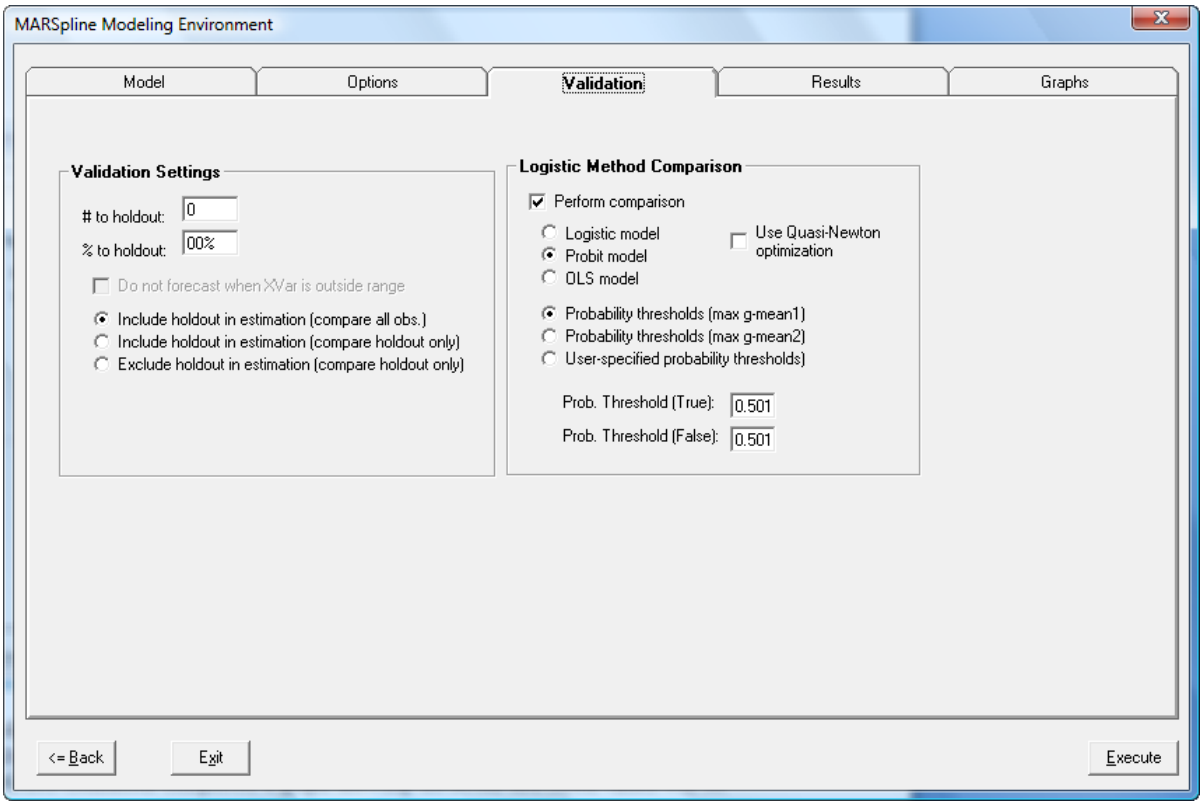
After specifying the model, click on the *Options* tab to set the maximum number of knots to 40 and the maximum number of interactions allowed to 2. By allowing interactions, the MARS-Probit model will determine if the combination of certain independent variables improve model performance.



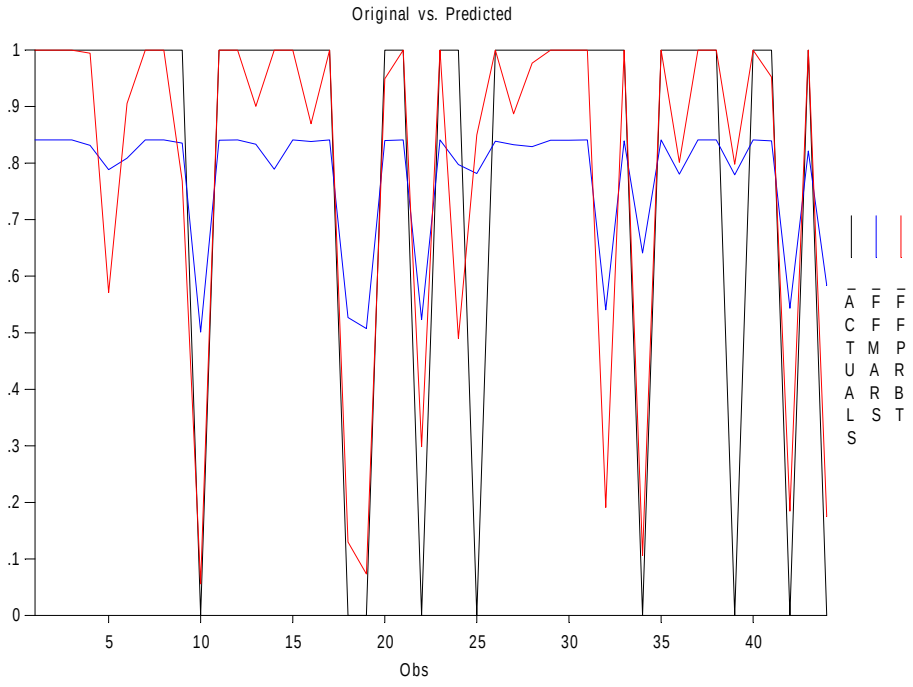
3.3.3 MARS-Probit Model Validation for the McManus Data Example

The *Validation* tab allows the user to evaluate the predictive power of the MARS-Probit model compared to a linear model. Select the radio button that compares all observations and set the number of observation in the holdout period to zero as shown below. By putting a check in the *Perform Comparison* checkbox and the *Probit* option, the linear probit model will be used as a benchmark to compare with the MARS-Probit model.

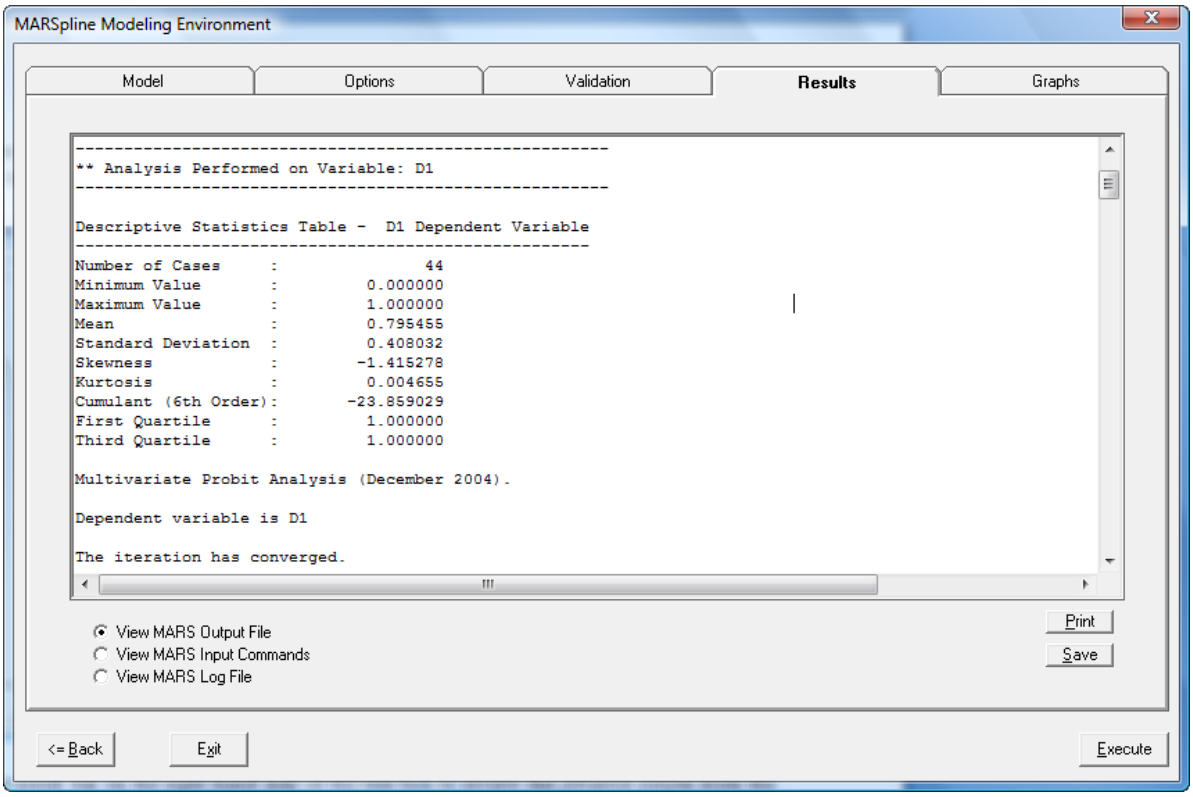
Finally, we will use the maximum g-mean1 statistic to determine the probability cut-off value for the True and False classification of D1. The g-mean 1 statistic is a geometric mean computed as $g-mean_1 = \sqrt{\text{True Positive Rate} * \text{Precision Rate}}$.



Next, click on the *Execute* button to run the analysis. The program script file will be automatically generated by SCA WorkBench and submitted to the SCAB34S engine for execution. When the MARS-Probit estimation completes, a graph showing the actual and fitted values will be displayed.



To view the output, first click on the graph to close it focus will automatically be set to the *Results* tab as shown below.



Use the scroll bar on the right-hand side of the text box to review the detailed output from the execution run.

```
** Analysis Performed on Variable: D1
-----------------------------------------------------

Descriptive Statistics Table -  D1 Dependent Variable
-----------------------------------------------------
Number of Cases     :               44
Minimum Value       :         0.000000
Maximum Value       :         1.000000
Mean                :         0.795455
Standard Deviation  :         0.408032
Skewness            :        -1.415278
Kurtosis            :         0.004655
Cumulant (6th Order):       -23.859029
First Quartile      :         1.000000
Third Quartile      :         1.000000

1/ Cond of variance covariance of coef     1.550633447891420E-09
# of Iterations                           11
Log of likelihood function                -8.640412513548712
Convergence tolerance                     9.999999747378752E-06

  **Summary of results**
                                                                  Partial Derivatives
Variable Lag  Max Likelihood       Est. Std. Error     t score     At Max Den.   At X Mean
TM         0      0.11312245E-01     0.56557361E-02   2.0001366      0.45129328E-02  0.58416776E-06
Y          0       6.4610744          3.1483250       2.0522260       2.5775957      0.33365184E-03
LF         0     -0.40928369         0.25721235      -1.5912288     -0.16328057     -0.21135534E-04
NW         0       42.498076          20.749605       2.0481390       16.954279      0.21946135E-02
D2         0       4.3900449          46.323736      0.94768801E-01   1.7513745      0.22670325E-03
CONSTANT   0       6.9154055          11.322386      0.61077279       2.7588477      0.35711364E-03
```

At point of means, E(dependent variable) 0.9999883866761383

# of observations	44
# limits(=0)	9
# nonlimits(=1)	35
(-2.0) times the log likelihood ratio	27.30345606139865
Distributed as Chi squared with DF	5
Significance of Chi squared statistic	0.9999502123677709

Final Model Estimation Summary - PROBIT
Method => Quasi-Newton (CMAXF2)
Functional Value= 27.303

#	Variable	Lag	Coefficient	std.Err	t-value
1	TM	0	0.113E-01	0.566E-02	2.00
2	Y	0	6.46	3.15	2.05
3	LF	0	-0.409	0.257	-1.59
4	NW	0	42.5	20.7	2.05
5	D2	0	4.39	46.3	0.948E-01
6	CONSTANT	0	6.92	11.3	0.611

PROBIT True/False Probability CutOff Range

Cut	True-Pos	True-Neg	False-Pos	False-Neg	Accuracy	Precision	GMean1	GMean2
0.050	1.000	0.000	1.000	0.000	0.795	0.795	0.892	0.000
0.100	1.000	0.222	0.778	0.000	0.841	0.833	0.913	0.471
0.150	1.000	0.444	0.556	0.000	0.886	0.875	0.935	0.667
0.200	0.971	0.667	0.333	0.029	0.909	0.919	0.945	0.805
0.250	0.971	0.667	0.333	0.029	0.909	0.919	0.945	0.805
0.300	0.971	0.778	0.222	0.029	0.932	0.944	0.958	0.869
0.350	0.971	0.778	0.222	0.029	0.932	0.944	0.958	0.869
0.400	0.971	0.778	0.222	0.029	0.932	0.944	0.958	0.869
0.450	0.971	0.778	0.222	0.029	0.932	0.944	0.958	0.869
0.500	0.943	0.778	0.222	0.057	0.909	0.943	0.943	0.856
0.550	0.943	0.778	0.222	0.057	0.909	0.943	0.943	0.856
0.600	0.914	0.778	0.222	0.086	0.886	0.941	0.928	0.843
0.650	0.914	0.778	0.222	0.086	0.886	0.941	0.928	0.843
0.700	0.914	0.778	0.222	0.086	0.886	0.941	0.928	0.843
0.750	0.914	0.778	0.222	0.086	0.886	0.941	0.928	0.843
0.800	0.886	0.889	0.111	0.114	0.886	0.969	0.926	0.887
0.850	0.857	1.000	0.000	0.143	0.886	1.000	0.926	0.926
0.900	0.800	1.000	0.000	0.200	0.841	1.000	0.894	0.894
0.950	0.714	1.000	0.000	0.286	0.773	1.000	0.845	0.845
1.000	0.000	1.000	0.000	1.000	0.265	0.000	0.000	0.000

Confusion Matrix
Performance Evaluation for PROBIT

	Predicted Negative	Predicted Positive	Unclassified
Negative	7	2	0
Positive	1	34	0

Total Number Cases	44
Total Classified Cases	44
Unclassified Positive Cases	0
Unclassified Negative Cases	0
Cut-off for True (>)	0.375
Cut-off for False (<)	0.375
Accuracy Rate	0.932
True Positive Rate	0.971
False Positive Rate	0.222
True Negative Rate	0.778
False Negative Rate	0.029
Precision Rate	0.944
g-mean1	0.958
g-mean2	0.869

PROBIT Lift-Gain Table

Decile	#Obs in Decile	#Pos in Decile	%Pos in Decile	Pctg of Total Pos	Cum. #Obs	Cum. #Pos	Cum. %Pos	Cum. Gain	K_S Spread	Lift Index	Gain over Random
1	4	4	100.0%	11.4%	4	4	9.1%	11.4%	11.4%	114	12.5%
2	5	5	100.0%	14.3%	9	9	20.5%	25.7%	25.7%	128	22.2%
3	4	4	100.0%	11.4%	13	13	29.5%	37.1%	37.1%	123	19.2%
4	5	5	100.0%	14.3%	18	18	40.9%	51.4%	51.4%	128	22.2%
5	4	4	100.0%	11.4%	22	22	50.0%	62.9%	62.9%	125	20.5%
6	4	4	100.0%	11.4%	26	26	59.1%	74.3%	74.3%	123	19.2%
7	5	4	80.0%	11.4%	31	30	68.2%	85.7%	74.6%	122	18.3%
8	4	3	75.0%	8.6%	35	33	75.0%	94.3%	72.1%	117	15.2%
9	5	2	40.0%	5.7%	40	35	79.5%	100.0%	44.4%	111	10.0%
10	4	0	0.0%	0.0%	44	35	79.5%	100.0%	0.0%	100	0.0%

Multivariate MARS-Probit Analysis (December 2004).

1/ Cond of variance covariance of coef	6.392614080808697E-06
# of Iterations	5
Log of likelihood function	-7.565011372386165
Convergence tolerance	9.999999747378752E-06

Summary of results

Variable	Lag	Max Likelihood	Est. Std. Error	t score	Partial Derivatives At Max Den.	At X Mean
Col____1	0	3.0869839	0.90014043	3.4294469	1.2315284	0.33075794
Col____2	0	-0.56479864	0.20463391	-2.7600442	-0.22532206	-0.60515910E-01
Col____3	0	-107.43487	46.908596	-2.2903023	-42.860310	-11.511215
Col____4	0	-6.8685071	3.6703310	-1.8713590	-2.7401379	-0.73593299

At point of means, E(dependent variable)	0.9475443333109783
# of observations	44
# limits(=0)	9
# nonlimits(=1)	35
(-2.0) times the log likelihood ratio	29.45425834372375
Distributed as Chi squared with DF	3
Significance of Chi squared statistic	0.9999982007019489

MARSPLINE Model Coefficients

					SE	t	--Non Zero--- #	%	Importance %	Coef#
D1	=	3.0869839			0.90014043	3.42	44	100.000		1
-0.56479864	* max(	161.00000	-	TM{ 0} , 0.0)	0.20463391	-2.76	16	36.364	100.000	2
	* max(	0.90000000E-01	-	NW{ 0} , 0.0)						
-107.43487	* max(	2.0400000	-	Y{ 0} , 0.0)	46.908596	-2.29	17	38.636	73.838	3
	* max(	0.90000000E-01	-	NW{ 0} , 0.0)						
-6.8685071	* max(	LF{ 0}	-	52.200000 , 0.0)	3.6703310	-1.87	21	47.727	49.660	4
	* max(	0.90000000E-01	-	NW{ 0} , 0.0)						

MARSPLINE True/False Probability CutOff Range

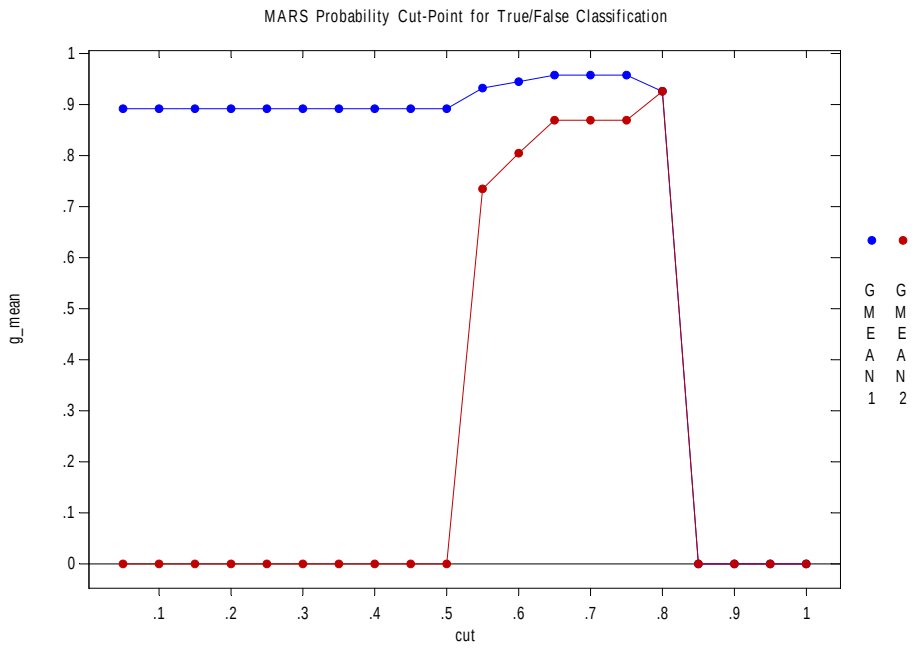
Cut	True-Pos	True-Neg	False-Pos	False-Neg	Accuracy	Precision	GMean1	GMean2
0.050	1.000	0.000	1.000	0.000	0.795	0.795	0.892	0.000
0.100	1.000	0.000	1.000	0.000	0.795	0.795	0.892	0.000
0.150	1.000	0.000	1.000	0.000	0.795	0.795	0.892	0.000
0.200	1.000	0.000	1.000	0.000	0.795	0.795	0.892	0.000
0.250	1.000	0.000	1.000	0.000	0.795	0.795	0.892	0.000
0.300	1.000	0.000	1.000	0.000	0.795	0.795	0.892	0.000
0.350	1.000	0.000	1.000	0.000	0.795	0.795	0.892	0.000
0.400	1.000	0.000	1.000	0.000	0.795	0.795	0.892	0.000
0.450	1.000	0.000	1.000	0.000	0.795	0.795	0.892	0.000
0.500	1.000	0.000	1.000	0.000	0.795	0.795	0.892	0.000
0.550	0.971	0.556	0.444	0.029	0.886	0.895	0.932	0.735
0.600	0.971	0.667	0.333	0.029	0.909	0.919	0.945	0.805
0.650	0.971	0.778	0.222	0.029	0.932	0.944	0.958	0.869
0.700	0.971	0.778	0.222	0.029	0.932	0.944	0.958	0.869
0.750	0.971	0.778	0.222	0.029	0.932	0.944	0.958	0.869
0.800	0.857	1.000	0.000	0.143	0.886	1.000	0.926	0.926
0.850	0.000	1.000	0.000	1.000	0.205	0.000	0.000	0.000
0.900	0.000	1.000	0.000	1.000	0.205	0.000	0.000	0.000
0.950	0.000	1.000	0.000	1.000	0.205	0.000	0.000	0.000
1.000	0.000	1.000	0.000	1.000	0.205	0.000	0.000	0.000

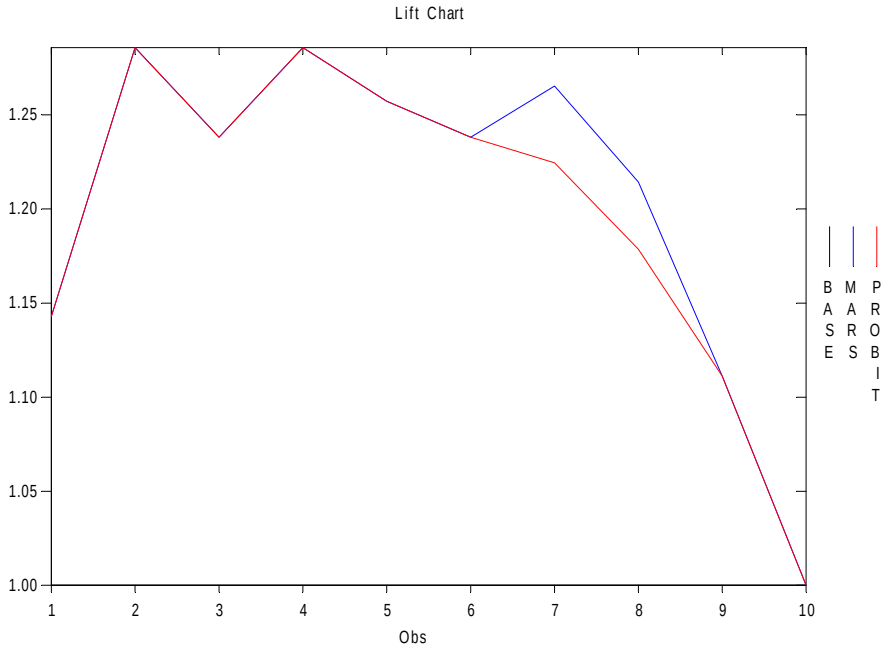
```
-----------------------------------------------------
                  Confusion Matrix
          Performance Evaluation for MARSPLINE
-----------------------------------------------------
                          Predicted
              Negative    Positive    Unclassified
Negative             7           2               0
Positive             1          34               0

Total Number Cases          :         44
Total Classified Cases      :         44
Unclassified Positive Cases :          0
Unclassified Negative Cases :          0
Cut-off for True  (>)       :      0.700
Cut-off for False (<)       :      0.700
Accuracy Rate               :      0.932
True  Positive Rate         :      0.971
False Positive Rate         :      0.222
True  Negative Rate         :      0.778
False Negative Rate         :      0.029
Precision Rate              :      0.944
g-mean1                     :      0.958
g-mean2                     :      0.869
```

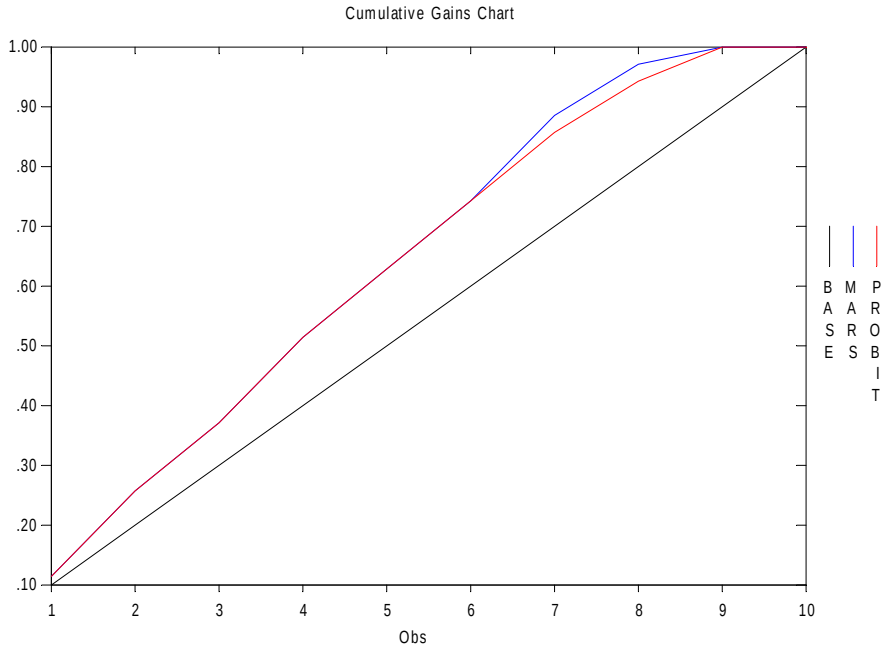
MARSPLINE Lift-Gain Table

Decile	#Obs in Decile	#Pos in Decile	%Pos in Decile	Pctg of Total Pos	Cum. #Obs	Cum. #Pos	Cum. %Pos	Cum. Gain	K_S Spread	Lift Index	Gain over Random
1	4	4	100.0%	11.4%	4	4	9.1%	11.4%	11.4%	114	12.5%
2	5	5	100.0%	14.3%	9	9	20.5%	25.7%	25.7%	128	22.2%
3	4	4	100.0%	11.4%	13	13	29.5%	37.1%	37.1%	123	19.2%
4	5	5	100.0%	14.3%	18	18	40.9%	51.4%	51.4%	128	22.2%
5	4	4	100.0%	11.4%	22	22	50.0%	62.9%	62.9%	125	20.5%
6	4	4	100.0%	11.4%	26	26	59.1%	74.3%	74.3%	123	19.2%
7	5	5	100.0%	14.3%	31	31	70.5%	88.6%	88.6%	126	21.0%
8	4	3	75.0%	8.6%	35	34	77.3%	97.1%	86.0%	121	17.6%
9	5	1	20.0%	2.9%	40	35	79.5%	100.0%	44.4%	111	10.0%
10	4	0	0.0%	0.0%	44	35	79.5%	100.0%	0.0%	100	0.0%

In addition to the Lift-Gain and cut-off probability evaluation tables, the corresponding Lift, Cumulative Gains , and G-Mean evaluation charts are accessible from the Graphs tab. The charts show the predictive power of the MARS-Probit model for this example.




Lift Chart
1.25
1.20
1.15
1.10
1.05
1.00
1 2 3 4 5 6 7 8 9 10
Obs
BASE
MARS
PROBIT


Cumulative Gains Chart
1.00
.90
.80
.70
.60
.50
.40
.30
.20
.10
1 2 3 4 5 6 7 8 9 10
Obs
BASE
MARS
PROBIT

4 A DETAILED DESCRIPTION OF THE MARSPLINE ROUTINE

The SCAB34S SPLINES product integrates the GPL subroutines for Multivariate Adaptive Regression Splines (MARSPLINE) models developed by Hastie and Tribshirani. A detailed description of the **MARSPLINE** subroutine is listed next and should be studied prior to any model building.

Usage:

```
CALL MARSPLINE(Y X1 X2 :OPTIONS);
```

Controls estimation of Multivariate Adaptive Regression Splines following methods suggested by Friedman (1991) but using GPL code developed by Hastie and Tribshirani. Lags can be entered as

```
X{1} or  X{1 to 20}
```

Required subroutine arguments:

Arguments	Intent	Comments
Y	*Input*	Dependent variable in the MARSPLINE model. The dependent variable may be specified as a random variable or a binary variable.
X	*Input*	One or more regressor variables in the MARSPLINE model. The regressor variables may be specified as random or categorical. If a lagged regressor variable is included in the model, the lags must be specified within braces as shown in the usage section above.

Optional keywords and associated arguments:

Keyword	# Args	Type(s)	Intent	Comments
:LOGIT	1	*Logical*	*input*	Accommodates a MARSlogit model where the left hand side variable is a logit (0-1) variable. Y=1/(1-exp(-xb)). **Not yet released.**
:PROBIT	1	*Logical*	*input*	Accommodates a MARSprobit model where the left hand side variable is a logit (0-1) variable. The MARSprobit estimation requires a two-step procedure which is handled automatically in the software application. Y=cumulative normal probability.
:PRINT	1	*Logical*	*input*	Print model header and minimal output for the MARSPLINE model
:DISPMARS	1	*Logical*	*input*	Displays most current MARSPLINE model estimation results in the SCAB34S SPLINES workspace.

:SAMPLE	1	*Real number array*	*input*	Specifies a mask real*8 variable where mask= 0.0 drops that observation. Unless the mask is the number of observations after any lags, an error message will be generated. The sample variable must be used with great caution when there are lags. A much better choice is the :holdout option when lagged regression variables are used.
:HOLDOUT	1	*Integer scalar*	*input*	Specifies the number of observations to hold out from the back of the time series for evaluating prediction power of the model. The :holdout option can not be used when the :sample option is employed.
:MI	1	*Integer scalar*	*input*	Sets the maximum number of variables per basis function. MI=1 → Additive model (default) MI>1 → Number of interactions allowed
:NK	1	*Integer scalar*	*input*	Sets the maximum number of basis functions allowed in the MARSPLINE model. Default=5.
:DF	1	*Real number scalar*	*input*	Sets the number of degrees of freedom charged for unrestricted knot optimization. Default=3.0.
:WEIGHT	1	*Logical*	*input*	Uses the last series specified in the model as a weight variable vector.
:SAVEMODEL	1	*Character*10*	*input*	Saves the estimated MARSPLINE model to a file in a B34S format that can be later read back into the program. The default file name is 'marss.psv'
:GETMODEL	1	*Character*10*	*input*	Imports a previously saved MARSPLINE model
:FORECAST	1	*Real number matrix*	*input*	Allows users to supply observations on the right hand side variables outside the sample period so that forecasts can be calculated. The same number of observations must be supplied for all right hand series. Due to the way that splines are calculated, it is imperative that any values of the x variables NOT lie outside the ranges of the original data. The forecast sentence produces the temporary variables %fore - forecast values %foreobs - index of the forecast value The forecast input matrix must be aligned and pre-lagged if lags are used in the model. The LAGMATRIX command can be used to conveniently create the forecast input matrix when lags are present.
:TRACE	1	*Logical*	*input*	Traces the solution of a MARSPLINE model. This option is only used for debugging purposes and is usually not needed.
:NOFWDSTEP	1	*Logical*	*Input*	Turns off forward step search.

:NOPRUNE	1	*Logical*	*Input*	Turns off model prunning step. This is not recommended in most cases.
:THRESH	1	*Real number scalar*	*Input*	Sets the threshold for forward step selection. Smaller numbers will allow for more complex models. The default value is 0.0001
:RANKTOL	1	*Real number scalar*	*Input*	Sets the threshold for pruning of a multicolinear basis. If the model seems overly complex, this threshold should by lowered. The default value is .1d-13
:TOLBX	1	*Real number scalar*	*Input*	Sets the threshold for variable inclusion into the model. The default value is .1d-8
:STOPFAC	1	*Integer sclar*	*Input*	Default value is set to 10. This value should not be modified under normal circumstances.
:SAVEBX	1	*Logical*	*Input*	Saves the matrix of basic functions with internal storage name %bx.

Variables created in MARSPLINE subroutine call:

Variable	Comments
%YVAR	*Character scalar; output only* Name of the dependent variable in the MARSPLINE model.
%NAMES	*Character array; output only* Names of the independent variables in the MARSPLINE model.
%TYPEVAR	*Integer array; output only* *0 if variable is continuous* *1 if variable is categorical*
%LAG	*Integer array; output only* Lags of the independent variables.
%COEF	*Real number array; output only* Estimates of the final MARSPLINE model. Constant is in location one. Size of the array is nk+1.
%SE	*Real number array; output only* Estimates of the standard errors of the coefficients assuming that the knot matrix is given. Users may elect to use alternative standard error estimates by recalculating based on the %BX matrix or from %VAR.
%MINVAR	*Real number array; output only* The minimum value of the independent variables for the estimation range.
%MAXVAR	*Real number array; output only* The maximum value of the independent variables for the estimation range.
%FLAG	*Integer array; output only* The FLAG(I , J) indicates that for %COEF(I) a knot was found within the variable(J)
%DIR	*Integer array; output only* Indicates the direction of a knot in the model for %COEF(I) and the variable(J). DIR(i,j) = 1 indicates the knot is of the form max(var-knot,0)

	DIR(I,j) = -1 indicates the knot is of the form max(knot-var,0)
%VARRIMP	*Real number array; output only* *The relative importance of the variable in the MARSPLINE model. The array is in the same order as the variables specified in the model.*
%BESTIN	*Real number array; output only* If BESTIN(i) <> 0 then %COEF(i) is in the final model
%CUT	*Real number array; output only* CUT(i , j) shows the ith coefficient knot for the jth variable

%BX	*Real number matrix; output only*
	n by nk matrix of knot products. First pick off basis vectors in the final model using %bestin array. Then estimated coefficients are obtainable from:
	
	s_bestin=sum(%bestin);
	xx=matrix(norows(%bx),s_bestin:);
	ihave=0;
	
	do i=1,norows(%bestin);
	if(%bestin(i).eq.1)then;
	ihave=ihave+1;
	xx(,ihave)=%bx(,i);
	endif;
	enddo;
	
	call olsq(vfam(%y) xx :noint : print);

%VAR	*Real number matrix; output only* The covariance of the parameters after QR step. The square root of the diagonal elements multiplied by (norows(%y) – norows(%coef)) are the standard errors of the coefficients.
%FWDINFO	*Real number array; output only* This array is of dimension (nfk,3) where (i,1) = GVC (i,2) = residual sum of squares (i,3) = cut Analysis of this array may give insight into the relative importance of a given knot.
%IWDINFO	*Real number array; output only* This array is of dimension (nfk,3) where (i,1) = i cut (i,2) = j cut (i,3) = parent index
%GVCNULL	*Real number scalar; output only* The GVC with only a constant in the model
%RSSNULL	*Real number scalar; output only* The residual sum of squares with only a constant in the model
%NK	*Integer scalar; output only* The :NK setting (max number of knots) used for the modeling procedure
%MI	*Integer scalar; output only* The :MI setting (max number of interactions in a basis function) used for the modeling procedure
%DF	*Real number scalar; output only*

	The :DF setting (number of degrees of freedom) used for the modeling procedure
%K	*Integer scalar; output only* The number of independent variables included in the final MARSPLINE model.
%NOBS	*Integer scalar; output only* The number of observations used in the estimated model
%RSS	*Real number scalar; output only* Residual sum of squares value.
%SUMRE	*Real number scalar; output only* Sum of absolute residuals value.
%REMAX	*Real number scalar; output only* Maximum absolute residual value.
%RESVAR	*Real number scalar; output only* Residual variance of the estimated MARSPLINE model.
%YHAT	*Real number array; output only* Fitted values of the dependent variable.
%Y	*Real number array; output only* *Actual values of the dependent variable having the same number of observations as %YHAT*
%RES	*Real number array; output only* Residuals of the estimated MARSPLINE model.
%FORE	*Real number array; output only* The forecast values for the holdout sample including forecasts that can be computed past the holdout if employing a lagged MARSPLINE model that does not contain any contemporaneous parameters.
%FOREOBS	*Integer array; output only* The observation index of the forecasts. If the MARSPLINE model contains lags, the maximum lag specified in the MARSPLINE model should be added to the %FOREOBS values in order to map back to the observation index of the dependent variable in the model.
%MODTYPE	*Integer scalar; output only* =0 is ordinal =1 is logistic =2 is probit

5 A DETAILED DESCRIPTION OF THE OLSQ COMMAND

The OLSQ command is a linear regression capability using OLS, L1, and MINIMAX estimation methods. The variables in the model can be real*8 or real*16. Real*16 capability is designed to handle difficult problems. Cholesky factorization is used to perform calculations although the QR (Quantile Regression) approach is an option. Recursive residuals can be optionally calculated.

Usage:

```
CALL OLSQ(Y X1 X2 :OPTIONS);
```

Lags can be entered as

```
X{1} or  X{1 to 20}
```

Required subroutine arguments:

Arguments	Intent	Comments
Y	*Input*	Dependent variable in the regression model. The dependent variable may be a random variable.
X	*Input*	One or more regressor variables in the regression model. The regressor variables may be specified also be random. If a lagged regressor variable is included in the model, the lags must be specified within braces as shown in the usage section above.

Optional keywords and associated arguments:

Keyword	# Args	Type(s)	Intent	Comments
:NOINT	1	*Logical*	*input*	Estimate the model without an intercept term.
:PRINT	1	*Logical*	*input*	Print estimation results for OLSQ
:DIAG	1	*Logical*	*input*	Print/save extensive diagnostics for the estimated model
:L1	1	*Logical*	*input*	Estimates a regression model using the L1 method which minimizes $\text{sum}\left(\text{abs}\left(Y_t - \hat{Y}_{t-1}\right)\right)$. This estimation method is not as sensitive to outliers as OLS or MINIMAX.
:MINIMAX	1	*Logical*	*input*	Estimates a regression model using the MINIMAX method which minimizes $\max\left(\text{abs}\left(Y_t - \hat{Y}_{t-1}\right)\right)$. This estimation method is more sensitive to outliers.

:QR	1	*Logical*	*input*	Uses the quantile regression method to obtain ordinary least squares (OLS) estimates. This option is slower and more computationally intensive than the default Cholesky decomposition method. Use this option when more accuracy is needed.
:WHITE **:WHITE1** **:WHITE2** **:WHITE3**	1	*Logical*	*input*	Computes the standard errors and t-values using method introduced by White (:WHITE) or its variants (:WHITE1, :WHITE2, :WHITE3). Results are saved in temporary variables, %whitese and %whitet. For details on alternative formulas see Davidson-MacKinnon (2004) page 199-200. See also Greene (2003) page 220 for added detail.
:SAVEX	1	*Logical*	*input*	Saves the X matrix in %X. This is useful for threshold autoregressive (TAR) modeling.
:SAMPLE	1	*Real number array*	*input*	Specifies a mask real*8 variable that if = 0.0 drops that observation from estimation. The sample variable must be used with great caution when there are lags specified in the model. If lags are specified, a much better option is the :holdout option.
:HOLDOUT	1	*Integer scalar*	*input*	Sets the number of observations to omit from the back of the data during estimation. When this option is used, the %XFUTURE temporary variable is created that saved the holdout portion of the data as an X-Matrix for post-sample forecasting purposes.
:RR	1	*Integer scalar*	*input*	Calculates recursive residuals for up to a specified maximum order. When this option is employed, several temporary variables are created. These temporary variables are indicated in the next table. The :RR option is computationally intensive. The user may consider using the RR command in the B34S ProSeries Econometric System for datasets greater than 10,000 observations.

Variables created specific to OLSQ OPTIONS specified:

Variable	If (option)	Comments
%YVAR		*Character scalar; output only* Name of the dependent variable in the model
%NAMES		*Character array; output only* Names of the regressor variables in the model
%LAG		*Integer array; output only* Lag operator associated with the regressor variable
%COEF		*Real number array; output only* The estimated OLS coefficients in the model
%SE		*Real number array; output only* The standard errors of the OLS coefficient estimates
%T		*Real number array; output only* The t-values for the OLS coefficient estimates
%RSS		*Real number scalar; output only* The residual sum of squares value for OLS estimation
%SUMRE		*Real number scalar; output only*

		The sum of absolute residuals for OLS estimation
%REMAX		*Real number scalar; output only* The maximum absolute residual value for OLS estimation
%RESVAR		*Real number scalar; output only* Residual variance for OLS estimation
%RSQ		*Real number scalar; output only* Centered R-Square value for OLS estimation
%RCOND		*Real number scalar; output only* Inverse of the condition of the X'X matrix
%NOB		*Integer scalar; output only* Number of observations used in the model
%K		*Integer scalar; output only* Number of regressor variables in the model
%XPXINV		*Real number matrix; output* Inverse X'X matrix
%YHAT		*Real number array; output only* The OLS fitted values for the dependent variable
%Y		*Real number array; output only* The dependent variable
%RES		*Real number array; output only* The residuals from the OLS model estimation
%ALMXK		*Real number scalar; output only* -2.0 * log maximum likelihood function for OLS estimation
%AICSTAT		*Real number scalar; output only* Akaike Information Criteria for OLS estimation
%SICSTAT		*Real number scalar; output only* Schwartz Information Criteria for OLS estimation
%FPETEST		*Real number scalar; output only* Akaike Finite Prediction Error (1970) for OLS estimation
%GVCTEST		*Real number scalar; output only* Generalized cross validation test for OLS estimation
%HGTEST		*Real number scalar; output only* Hannan - Quinn (1979) for OLS estimation
%SHTEST		*Real number scalar; output only* Shibata test for OLS estimation
%RICETST		*Real number scalar; output only* Rice test for OLS estimation
%XFUTURE		*Real number matrix; output only* The X-Matrix for the holdout sample (if specified)
%XFOBS		*Integer array; output only* The observation number of the holdout sample (if specified)
%WHITESE	*:WHITE or* *:WHITE1 or* *:WHITE2 or* *:WHITE3*	*Real number array; output only* White standard errors for estimates
%WHITET	*:WHITE or* *:WHITE1 or* *:WHITE2 or* *:WHITE3*	*Real number array; output only* White t-values for estimates
%X	*:SAVEX*	*Real number matrix; output only* The X-Matrix
%L1COEF	*:L1*	*Real number array; output only* The estimated coefficients using L1 estimation
%L1SUMRE	*:L1*	*Real number scalar; output only* The sum of absolute errors for L1 estimation
%L1RES	*:L1*	*Real number array; output only* The residuals from L1 model estimation

%L1YHAT	*:L1*	*Real number array; output only* The fitted values from L1 estimation
%L1RSS	*:L1*	*Real number scalar; output only* Residual sum of squares from L1 estimation
%L1REMAX	*:L1*	*Real number scalar; output only* The maximum absolute residual from L1 estimation
%MMCOEF	*:MINIMAX*	*Real number array; output only* The estimated coefficients using MINIMAX estimation
%MMSUMRE	*:MINIMAX*	*Real number scalar; output only* The sum of absolute errors for MINIMAX estimation
%MMRES	*:MINIMAX*	*Real number array; output only* The residuals from MINIMAX model estimation
%MMYHAT	*:MINIMAX*	*Real number array; output only* The fitted values from MINIMAX estimation
%MMRSS	*:MINIMAX*	*Real number scalar; output only* Residual sum of squares from MINIMX estimation
%MMREMAX	*:MINIMAX*	*Real number scalar; output only* The maximum absolute residual from MINIMAX estimation
%RROBS	*:RR*	*Integer array; output only* Recursive residual observation base
%RR	*:RR*	*Real number matrix; output only* Recursive residuals matrix of dimension (N-K) by maximum order specified for the RR option
%RRSTD	*:RR*	*Real number matrix; output only* Standardized recursive residuals matrix of dimension (N-K) by maximum order specified for the RR option
%RRCOEF	*:RR*	*Real number matrix; output only* Coefficients used for recursive residuals of dimension (N-K) by maximum order specified for the RR option
%RRCOEFT	*:RR*	*Real number matrix; output only* t-Values of coefficients used for recursive residuals of dimension (N-K) by maximum order specified for the RR option
%SSR1	*:RR*	*Real number array; output only* Sumof square residuals going forward. N-2K obs
%SSR2	*:RR*	*Real number array; output only* Sum of squares residuals backward. N-2K obs

6 A DETAILED DESCRIPTION OF THE PROBIT COMMAND

The PROBIT capability estimates the probability of a dichotomous (0-1) response variable based on the cumulative normal probability distribution. The log-likelihood function is

$$\ln L = \sum w_j \ln \Phi(x_j b) + \sum w_j \ln(1 - \Phi(x_j b))$$

The variables in the model can be real*8 and the X-variables may be specified as a matrix if desired.

Usage:

```
CALL PROBIT(Y X1 X2 :OPTIONS);
```

Lags can be entered as

```
X{1} or  X{1 to 20}
```

Required subroutine arguments:

Arguments	Intent	Comments
Y	*Input*	Dependent variable in the probit model. The dependent variable must contain values of 0 and 1 only.
X	*Input*	One or more regressor variables in the probit model. The regressor variables may be specified as random or as dummies. If a lagged regressor variable is included in the model, the lags must be specified within braces as shown in the usage section above.

Optional keywords and associated arguments:

Keyword	# Args	Type(s)	Intent	Comments
:PRINT	1	*Logical*	*input*	Print estimation results for probit
:PRINTVCV	1	*Logical*	*input*	Print Variance-Covariance matrix
:SECD	1	*Logical*	*input*	Print second derivatives matrix
:NSTRT	1	*Integer Scalar*	*input*	Beginning observation for output of predicted and actual dependent variable and its density.
:NSTOP	1	*Integer Scalar*	*input*	Ending observation for output of predicted and actual dependent variable and its density.
:TOLA	1	*Real Number Scalar*	*input*	Convergence tolerance Default = .00001
:IITLK	1	*Logical*	*Input*	Print the Log-likelihood function after each iteration
:IIESK	1	*Logical*	*Input*	Print estimated coefficients after each iteration
:SAVEX	1	*Logical*	*input*	Saves the X-Variable matrix in the temporary variable %X
:HOLDOUT	1	*Integer scalar*	*input*	Sets the number of observations to omit from the back of the data during estimation. When this option is used, the %XFUTURE temporary variable is created that saved the holdout portion of the data as an X-Matrix for post-sample forecasting purposes.

Variables created specific to PROBIT OPTIONS specified:

Variable	If (option)	Comments
%YVAR		*Character scalar; output only* Name of the dependent variable in the model
%Y		*Real number array; output only* The dependent variable
%YHAT		*Real number array; output only* The probit fitted values for the dependent variable
%NAMES		*Character array; output only* Names of the regressor variables in the model
%LAG		*Integer array; output only* Lag operator associated with the regressor variable
%COEF		*Real number array; output only* The estimated coefficients in the probit model
%SE		*Real number array; output only* The standard errors of the coefficient estimates
%T		*Real number array; output only* The t-values for the coefficient estimates
%FUNC		*Real number scalar; output only* -2.0 times Log-likelihood function
%FUNCSIG		*Real number scalar; output only* Statistical significance of %FUNC
%DFFUNC		*Real number scalar; output only* Degrees of freedom for %FUNC
%LIMITS		*Integer scalar; output only* The number of 0's in the dependent variable
%RCOND		*Real number scalar; output only* 1 / condition of the Variance-Covariance matrix
%HESSIAN		*Real number matrix; output only* Hessian matrix
%XFUTURE		*Real number matrix; output only* The X-Matrix for the holdout sample (if specified)
%XFOBS		*Integer array; output only* The observation number of the holdout sample (if specified)
%XPXINV		*Real number matrix; output* Inverse X'X matrix

7 A DETAILED DESCRIPTION OF CUSTOMIZABLE SUBROUTINES USED IN THE MARSPLINE APPLICATION

The SCAB34S SPLINES application uses some customized user subroutines that were developed in the B34S matrix language. These subroutines perform similar to a macro procedure and are stored in the STAGING.MAC and WBSUPPL.MAC files located in the B34SLM installation folder. Since these subroutines are loaded and called from the script generated by SCA WorkBench for MARSPLINE modeling, the user can modify the scripts to handle some specific needs, as long as the nature of the arguments and calling sequence of the arguments is not changed.

For example, the P_L_EST subroutine performs LOGIT and PROBIT model estimation using two available optimization routines. The user can conceivably modify the functional forms of the model being optimized or include additional sequences inside the subroutine and have them automatically employed in the MARSPLINE Modeling application.

It is highly recommended that you make a backup of the original files before attempting to customize the subroutines.

7.1 P_L_EST User Subroutine

The P_L_EST routine estimates a Logit (or Probit) model using the CMAXF2 (Quasi-Newton) or MAXF2 which is a safeguarded quadratic interpolation method written by M.J.D. Powell of Cambridge University. The Logit estimation procedure in this user subroutine is much slower than the Probit matrix subroutine. It is recommended that the Probit option be used for faster execution.

By default, the MARSPLINE interface application will use the MAXF2 routine when estimating a Logit model as a validation against the MARSPLINE method. The standard errors for the estimated model parameters may differ between the CMAXF2 and MAXF2 routines, however since prediction accuracy is of primary interest in validating the MARSPLINE model, the MAXF2 routine seems to converge faster and with more reliability using default convergence criteria. The Probit option is not used by the MARSPLINE interface since a more comprehensive and faster Probit routine is available directly as a B34S matrix subroutine.

Usage:

```
CALL LOAD(P_L_EST  :STAGING);
CALL P_L_EST(TYPEMOD,Y,X,COEF,SE,T,YHAT,IPRINT,ROUTINE);
```

Required subroutine arguments:

Arguments	Type(s)	Intent	Comments
TYPEMOD	*Character keyword*	*input*	Sets the model type as a Logit or a Probit model. LOGIT => Logit Model PROBIT => Probit model
Y	*Real number vector*	*input*	The name of the dependent variable (0-1) in the model

X	*Real number matrix*	*input*	The name of the X-Matrix of regressor variables in the model. If lags are used, the regressor variables must be already pre-lagged in the X-Matrix. Use the LAGMATRIX routine to perform the lagging operation.
COEF	*Real number vector*	*output*	The name of the variable that the model coefficients are to be stored after estimation.
SE	*Real number vector*	*output*	The name of the variable that the standard errors of the coefficients are to be stored after estimation.
T	*Real number vector*	*output*	The name of the variable that the t-values of the model coefficients are to be stored after estimation.
YHAT	*Real number vector*	*output*	The name of the variable that the fitted values are to be stored after estimation.
IPRINT	*keyword*	*input*	Sets whether to print the output from the estimation or run silent. The keywords are PRINT => print estimation NOPRINT => silent
ROUTINE	*Integer key*	*input*	Sets the optimization routine to use for estimating the model. The integer keys are 0 => CMAXF2 (Quasi-Newton) 1 => MAXF2 (Powell)

Example:

```
call p_l_est('LOGIT',_ytmp, _xlogit,_func,_logparm,_logse,_logt,
_fflogt,'noprint',1);
```

7.2 DISP_LGT User Subroutine

The DISP_LGT subroutine displays a formatted model summary of parameter estimates, standard errors, and t-values for the Logit and Probit model estimated using the P_L_EST routine. An example of the information it generates is shown below:

```
Final Model Estimation Summary -  LOGIT
Method =>   Powell Quadratic Interpolation (MAXF2)
Functional Value=       -7.2955
-----------------------------------------------------------------
  # Variable   Lag     Coefficient      std.Err      t-value
-----------------------------------------------------------------
  1 Y            0            14.1    0.321E-01       438.00
  2 M            0            1.54        0.909         1.69
  3 LF           0          -0.743        0.110        -6.77
  4 NW           0            60.0         3.12         19.2
  5 TM           0       0.316E-01    0.104E-01         3.04
  6 CONSTANT     0            3.48         4.30        0.810
```

Usage:

```
CALL LOAD(DISP_LGT   :WBSUPPL);
CALL DISP_LGT(PNAMES,LAGS,COEF,SE,TVAL,FUNC,MDLDESC,METHOD);
```

Required subroutine arguments:

Arguments	Type(s)	Intent	Comments
PNAMES	*Character array*	*input*	The names of the parameters in the same order as the coefficients, standard errors, and t-values.
LAGS	*Integer array*	*input*	The lags associated with the parameters in the model.
COEF	*Real number vector*	*input*	The estimated coefficients in the model.
SE	*Real number vector*	*input*	The standard errors of the estimated coefficients in the model.
TVAL	*Real number vector*	*input*	The t-values of the estimated coefficients in the model.
FUNC	*Real number scalar*	*input*	The functional value of the estimated model.
MDLDESC	*Character string*	*input*	One or two words describing the model (e.g., LOGIT)
METHOD	*Integer key*	*input*	An integer key value assigned to the estimation method used. This should match the ROUTINE integer key in the P_L_EST routine. 0 => CMAXF2 (Quasi-Newton) 1 => MAXF2 (Powell)

Example:

```
call disp_lgt(%lmatvar,%lmatlag,_logparm,_logse,_logt,_func,'LOGIT',1);
```

7.3 DISP_OLS User Subroutine

The DISP_OLS subroutine displays a formatted model summary of parameter estimates, standard errors, and t-values for the linear regression model estimated using the OLSQ routine. An example of the information it generates is provided below.

```
Final Model Estimation Summary -  OLS
Dependent Variable:               DAYLOAD
-------------------------------------------------------------
  # Variable   Lag     Coefficient      std.Err      t-value     Comments
-------------------------------------------------------------
  1 TEMPERTR     0          70.929       34.874        2.034
  2 DAYLOAD      1           0.835        0.021       39.507
  3 CONSTANT     0       20776.779     3121.907        6.655

Number of observations:                    729
Adjusted R-Square:                       0.726
Sum of Squared Residuals:            0.110E+12
Schwartz Information Criteria:       15825.765
```

Usage:

```
CALL LOAD(COINT2       :WBSUPPL);
CALL DISP_OLS(PNAMES,LAGS,COEF,SE,TVAL,NUMOBS,ADJR2,SIC,RSS,
  MDLDESC,CMNT,ESTTYP);
```

Required subroutine arguments:

Arguments	Type(s)	Intent	Comments
PNAMES	*Character array*	*input*	The names of the parameters in the same order as the coefficients, standard errors, and t-values.
LAGS	*Integer array*	*input*	The lags associated with the parameters in the model.
COEF	*Real number vector*	*input*	The estimated coefficients in the model.
SE	*Real number vector*	*input*	The standard errors of the estimated coefficients in the model.
TVAL	*Real number vector*	*input*	The t-values of the estimated coefficients in the model.
NUMOBS	*Integer scalar*	*input*	Number of observations used in estimation
ADJR2	*Real number scalar*	*input*	The Adjusted R-Square value
SIC	*Real number scalar*	*input*	The Schwartz information criteia
RSS	*Real number scalar*	*input*	Residual Sum of Squares
MDLDESC	*Character string*	*input*	One or two words describing the model (e.g., LOGIT)
CMNT	*Character*1 array*	*input*	Row comments for a coefficient
ESTTYP	*Integer key*	*input*	An integer key value assigned to the estimation

			method used. 1 => OLS 2 => L1 3 => MiniMax

Example:

```
call olsq(DAYLOAD argument(_rxmdl)  :diag);
ccomment=c1array(norows(%coef):);
call character(vnamea,'DAYLOAD');
call disp_ols(vnamea,%names,%lag,%coef,%se,%t,%nob,
%adjrsq,%sicstat,%rss,'OLS',ccomment,1);
_ffols=goodrow(vfam(%yhat));
```

7.4 DSP_TBL User Subroutine

The DISP_TBL subroutine displays a formatted numeric table. An example of the information it generates is shown below:

```
Obs         MARS_          OLS_       DAYLOAD
  2   165338.4039   162142.7985   159213.0000
  3   150036.4320   157026.7541   138488.0000
  4   136420.1296   139959.8565   125174.0000
  5   128691.0769   129285.0584   137993.0000
  6   131868.7082   140482.1561   134406.0000
  7   133968.5805   137716.6297   134957.0000
  8   132311.4424   138080.8810   134449.0000
  9   132753.3339   136914.8574   133349.0000
 10   146452.7486   135152.4616   136808.0000
```

Usage:

```
CALL LOAD(DISP_TBL     :WBSUPPL);
CALL DISP_TBL(DMATRIX,NNAMES,ISPACE,IPREC,DESC);
```

Required subroutine arguments:

Arguments	Type(s)	Intent	Comments
DMATRIX	*Real number matrix (n,p)*	*input*	A rectangular matrix of numeric data
NNAMES	*Character array (p)*	*input*	The labels for the columns of data
ISPACE	*Integer scalar*	*Input*	The size of each column of data
IPREC	*Integer scalar*	*Input*	The number of decimals of precision displayed
DESC	*Character*1 array*	*Input*	The title description of the table

Example:

```
call dsp_tbl(_dmat,nnames,_nbegn,14,4,_desc);
```

7.5 TLGTR User Subroutine

The TLGTR subroutine produces the ratio statistics of a confusion matrix over a range of probability cut-off values. This subroutine also calls the TLOGIT subroutine to display the final Confusion Matrix to evaluate the performance of a logistic model. An example of the information it generates is shown below:

```
--------------------------------------------------------------------------------
                    PROBIT True/False Probability CutOff Range
--------------------------------------------------------------------------------
   Cut  True-Pos    True-Neg  False-Pos  False-Neg  Accuracy Precision   GMean1   GMean2
 0.050     1.000       0.444      0.556      0.000     0.630     0.474    0.688    0.667
 0.100     1.000       0.444      0.556      0.000     0.630     0.474    0.688    0.667
 0.150     1.000       0.500      0.500      0.000     0.667     0.500    0.707    0.707
 0.200     1.000       0.611      0.389      0.000     0.741     0.562    0.750    0.782
 0.250     1.000       0.667      0.333      0.000     0.778     0.600    0.775    0.816
 0.300     0.889       0.833      0.167      0.111     0.852     0.727    0.804    0.861
 0.350     0.889       0.833      0.167      0.111     0.852     0.727    0.804    0.861
 0.400     0.778       0.833      0.167      0.222     0.815     0.700    0.738    0.805
 0.450     0.667       0.833      0.167      0.333     0.778     0.667    0.667    0.745
 0.500     0.556       0.833      0.167      0.444     0.741     0.625    0.589    0.680
 0.550     0.556       0.833      0.167      0.444     0.741     0.625    0.589    0.680
 0.600     0.556       0.889      0.111      0.444     0.778     0.714    0.630    0.703
 0.650     0.444       0.944      0.056      0.556     0.778     0.800    0.596    0.648
 0.700     0.333       0.944      0.056      0.667     0.741     0.750    0.500    0.561
 0.750     0.222       0.944      0.056      0.778     0.704     0.667    0.385    0.458
 0.800     0.111       0.944      0.056      0.889     0.667     0.500    0.236    0.324
 0.850     0.111       1.000      0.000      0.889     0.704     1.000    0.333    0.333
 0.900     0.111       1.000      0.000      0.889     0.704     1.000    0.333    0.333
 0.950     0.000       1.000      0.000      1.000     0.667     0.000    0.000    0.000
 1.000     0.000       1.000      0.000      1.000     0.667     0.000    0.000    0.000

--------------------------------------------------
                Confusion Matrix
        Performance Evaluation for PROBIT
--------------------------------------------------
                          Predicted
               Negative    Positive    Unclassified
Negative             15           3              0
Positive              1           8              0

Total Number Cases          :         27
Total Classified Cases      :         27
Unclassified Positive Cases :          0
Unclassified Negative Cases :          0
Cut-off for True  (>)       :      0.325
Cut-off for False (<)       :      0.325
Accuracy Rate               :      0.852
True  Positive Rate         :      0.889
False Positive Rate         :      0.167
True  Negative Rate         :      0.833
False Negative Rate         :      0.111
Precision Rate              :      0.727
g-mean1                     :      0.804
g-mean2                     :      0.861
```

Usage:

```
CALL LOAD(TLOGIT    :STAGING);
CALL TLGTR(YVAL,YHAT,NCUT, PTRUER,PFALSER, PFPOS, PFFALS,
ACCURACY, PRECISION, GMEAN1, GMEAN2, RPROB1, RPROB2,
UPPER,LOWER,DESC,IPROBSET,IMDL,IPRINT);
```

Required subroutine arguments:

Arguments	Type(s)	Intent	Comments
YVAL	*Real number array*	*Input*	The dependent variable (0-1) containing the actual series.
YHAT	*Real number array*	*Input*	The predicted series to be evaluated corresponding to the dependent variable.
NCUT	*Integer scalar*	*Input*	The number of probability cut-off values to evaluate between the range 0-1
PTRUER	*Real number array*	*Output*	True positive ratio
PFALSER	*Real number array*	*Output*	True negative ratio
PFPOS	*Real number array*	*Output*	False positive ratio
PFFALS	*Real number array*	*Output*	False negative ratio
ACCURACY	*Real number array*	*Output*	Accuracy rate
PRECISION	*Real number array*	*Output*	Precision rate
GMEAN1	*Real number array*	*Output*	g-mean 1 => SQRT(PTRUER*PRECISION)
GMEAN2	*Real number array*	*Output*	g-mean 2 => SQRT(PTRUER*PFALSER)
RPROB1	*Real number scalar*	*Output*	Max(GMEAN1) value
RPROB2	*Real number scalar*	*Output*	Max(GMEAN2) value
UPPER	*Real number scalar*	*Input/Output*	The threshold value used to classify a prediction as a positive instance. YHAT >= UPPER
LOWER	*Real number scalar*	*Input/Output*	The threshold value used to classify a prediction as a negative instance. YHAT < LOWER
DESC	*Character string*	*Input*	A character string that provides a description for the confusion matrix.
IPROBSET	*Integer scalar*	*Input*	Controls what cut-off probability values are used for the confusion matrix 0 ➔ UPPER and LOWER (user specified) 1 ➔ RPROB1 is used (max(gmean1)) 2 ➔ RPROB2 is used (max(gmean2))
IMDL	*Integer scalar*	*Input*	Specify the ID of model used for classification. This ID is used to set the table descriptions and graph titles. 1 ➔MARSPLINE, 2 ➔LOGIT, 3 ➔PROBIT, 4 ➔OLS, 5 ➔GAM

IPRINT	*Integer key*	*Input*	Specify whether or not to print the confusion matrix Key =0 => no print Key =1 => print Key =2 => print and graph

Example:

```
_fflogt=afam(_fflogt);
_iprint=2;
_iprobset=1;
call tlgtr(_actuals,_fflogt,_ncut,_ptruer,_pfalser,_pFPos,_pFFals,
           _acc,_pprecise,_gmean1,_gmean2,rprob1,rprob2,_upper,_lower,
           _desc,_iprobSet,_iprobset,_iprint);
```

7.6 TLOGIT User Subroutine

The TLOGIT subroutine produces a confusion matrix (Kohavi and Provost, 1998) that compares actual to predicted classifications from a classification system. TLOGIT presents information to evaluate the predictive performance of a logistic model. An example of the information it generates is shown below:

```
-----------------------------------------------------
                 Confusion Matrix
          Performance Evaluation for LOGIT
-----------------------------------------------------
                           Predicted
               Negative    Positive    Unclassified
Negative              7           2           0
Positive              1          34           0

Total Number Cases          :         44
Total Classified Cases      :         44
Unclassified Positive Cases :          0
Unclassified Negative Cases :          0
Cut-off for True  (>=)      :      0.500
Cut-off for False (<)       :      0.500
Accuracy Rate               :      0.932
True  Positive Rate         :      0.971
False Positive Rate         :      0.222
True  Negative Rate         :      0.778
False Negative Rate         :      0.222
Precision Rate              :      0.944
g-mean1                     :      0.958
g-mean2                     :      0.869
```

Usage:

```
CALL LOAD(TLOGIT    :STAGING);
CALL TLOGIT(YVAL,YHAT,UPPER,LOWER,DESC,NTRUER,NTRUEP,NFALSER,
  NFALSEP,NUNCLEAR,PTRUER,PFALSER,IPRINT);
```

Required subroutine arguments:

Arguments	Type(s)	Intent	Comments
YVAL	*Real number array*	*Input*	The dependent variable (0-1) containing the actual series.
YHAT	*Real number array*	*Input*	The predicted series to be evaluated corresponding to the dependent variable.
UPPER	*Real number scalar*	*Input*	The threshold value used to classify a prediction as a positive instance. YHAT >= UPPER
LOWER	*Real number scalar*	*Input*	The threshold value used to classify a prediction as a negative instance. YHAT < LOWER
DESC	*Character string*	*Input*	A character string that provides a description for the confusion matrix.
NTRUER	*Real number*	*Output*	The number of positive cases correctly classified

	scalar		
NTRUEP	*Real number scalar*	*Output*	The total number of predicted positive cases
NFALSER	*Real number scalar*	*Output*	The number of negative cases correctly classified
NFALSEP	*Real number scalar*	*Output*	The total number of predicted negative cases
NUNCLEAR	*Real number scalar*	*Output*	The total number of cases that could not be classified. This may occur if there is a gap in the upper and lower threshold values
PTRUER	*Real number scalar*	*Output*	True positive ratio
PFALSER	*Real number scalar*	*Output*	True negative ratio
IPRINT	*Integer key*	*Input*	Specify whether or not to print the confusion matrix Key =0 => no print Key<>0 => print

Example:

```
_fflogt=afam(_fflogt);
_iprint=1;
call tlogit(_actuals,_fflogt,_upper,_lower,_desc2,_ntruer,_ntruep
   _nfalser,_nfalsep,_nunclear,_ptruer,_pfalser,_iprint);
```

7.7 LIFTGAIN User Subroutine

The LIFTGAIN subroutine displays a formatted table of the cumulative gains and lift of a classification system. LIFTGAIN presents information to evaluate the predictive performance of a logistic model compared to base level expectations. An example of the information it generates is shown below:

GAM Lift-Gain Table

Decile	#Obs in Decile	#Pos in Decile	%Pos in Decile	Pctg of Total Pos	Cum. #Obs	Cum. #Pos	Cum. %Pos	Cum. Gain	K_S Spread	Lift Index	Gain over Random
1	10	10	100.0%	40.0%	10	10	10.0%	40.0%	40.0%	400	75.0%
2	10	10	100.0%	40.0%	20	20	20.0%	80.0%	80.0%	400	75.0%
3	10	4	40.0%	16.0%	30	24	24.0%	96.0%	88.0%	320	68.8%
4	10	1	10.0%	4.0%	40	25	25.0%	100.0%	80.0%	250	60.0%
5	10	0	0.0%	0.0%	50	25	25.0%	100.0%	66.7%	200	50.0%
6	10	0	0.0%	0.0%	60	25	25.0%	100.0%	53.3%	166	40.0%
7	10	0	0.0%	0.0%	70	25	25.0%	100.0%	40.0%	142	30.0%
8	10	0	0.0%	0.0%	80	25	25.0%	100.0%	26.7%	125	20.0%
9	10	0	0.0%	0.0%	90	25	25.0%	100.0%	13.3%	111	10.0%
10	**10**	**0**	**0.0%**	**0.0%**	**100**	**25**	**25.0%**	**100.0%**	**0.0%**	**100**	**0.0%**

Usage:

```
CALL LOAD(LIFTGAIN     :WBSUPPL);
CALL LIFTGAIN(PREDICT1,ACTUALS1,DESC,NPT,NET,CPGAIN,CPLIFT);
```

Required subroutine arguments:

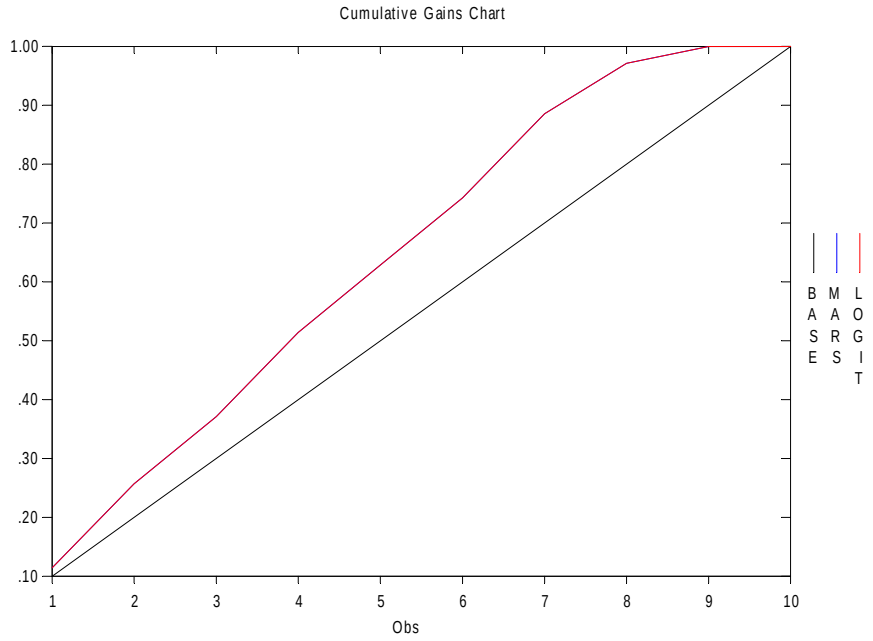
Arguments	Type(s)	Intent	Comments
PREDICT1	*Real number array*	*Input*	The predicted probabilities of positive outcomes for the sample period.
ACTUALS1	*Real number array*	*Input*	The actuals (0/1) for the sample period.
DESC	*Character string*	*Input*	A description of the table or method being displayed. For example, "MARSPLINE Lift-Gains".
NPT	*Integer array*	*Output*	The number of positive outcomes classified by the classification system by deciles.
NET	*Integer array*	*Output*	The number of expected positive outcomes by deciles.
CPGAIN	*Real number array*	*Output*	The cumulative gains of the classification system by deciles
CPLIFT	*Real number array*	*Output*	The cumulative lifts of the classification system by deciles

Example:

```
call character(_desc1b, 'MARSPLINE Lift-Gain Table');
call liftgain(_ffmars,_actuals,_desc1b,npt,nt,margain,marlift);
```

7.8 GRFLIFT User Subroutine

The GRFLIFT subroutine creates the cumulative gains chart and lift chart using the results of the LIFTGAIN subroutine. The subroutine will charts up to two comparative methods on the same chart. The charts are saved to windows-metafile files under the names GAINS.WMF and LIFT.WMF respectively. An example of the gains chart is shown below:



Usage:

```
CALL LOAD(LIFTGAIN    :WBSUPPL);
CALL GRFLIFT(GAIN1,GAIN2,LIFT1,LIFT2,NAME1,NAME2,M);
```

Required subroutine arguments:

Arguments	Type(s)	Intent	Comments
GAIN1	*Real number array*	*Input*	The cumulative gains by deciles for method 1.
GAIN2	*Real number array*	*Input*	The cumulative gains by deciles for method 2. If only one method is being displayed, pass the same array as the GAIN1 argument.
LIFT1	*Real number array*	*Input*	The cumulative lift by deciles for method 1.
LIFT2	*Real number array*	*Input*	The cumulative lift by deciles for method 2. If only one method is being displayed, pass the same array as the LIFT1 argument.
NAME1	*Integer Keyword*	*Input*	1=MARSPLINE, 2=LOGIT, 3=PROBIT, 4=OLS
NAME2	*Integer Keyword*	*Input*	1=MARSPLINE, 2=LOGIT, 3=PROBIT, 4=OLS
M	*Integer*	*Input*	1=>Graph Lift/Gain for method 1 only 2=>Graph Lift/Gain for method 1 and method 2

Example:

```
call grflift(margain,marlift,lgtgain,lgtlift,1,2,2);
```

7.9 DISP_HIN User Subroutine

The DISP_HIN subroutine displays a formatted table of the Hinich (1982) nonlinearity tests(subroutine hinich82()). An example of the information it generates is shown below:

```
-----------------------------------------------------
Hinich82 Bi-Spectrum Nonlinear Tests -  MARSPLINE Residuals
-----------------------------------------------------
Gaussality  :       -0.370
Linearity   :       -0.363
```

Usage:

```
CALL LOAD(DISP_HIN    :WBSUPPL);
CALL DISP_HIN(GAUSS,LINEAR,MDLTYPE);
```

Required subroutine arguments:

Arguments	Type(s)	Intent	Comments
GAUSS	*Real number array*	*Input*	An array(2) that contains the Hinich Gaussality tests from the HINICH82 subroutine.
LINEAR	*Real number array*	*Input*	An array(2) that contains the Hinich Linearity tests from the HINICH82 subroutine.
MDLTYPE	*Character Scalar*	*Input*	A character description of the variable that is being tested (e.g., MARSPLINE Residuals).

Example:

```
call hinich82(res2,_mols,_gols,_lols :meanonly);
call disp_hin(_gols,_lols,'OLS Residuals');
```

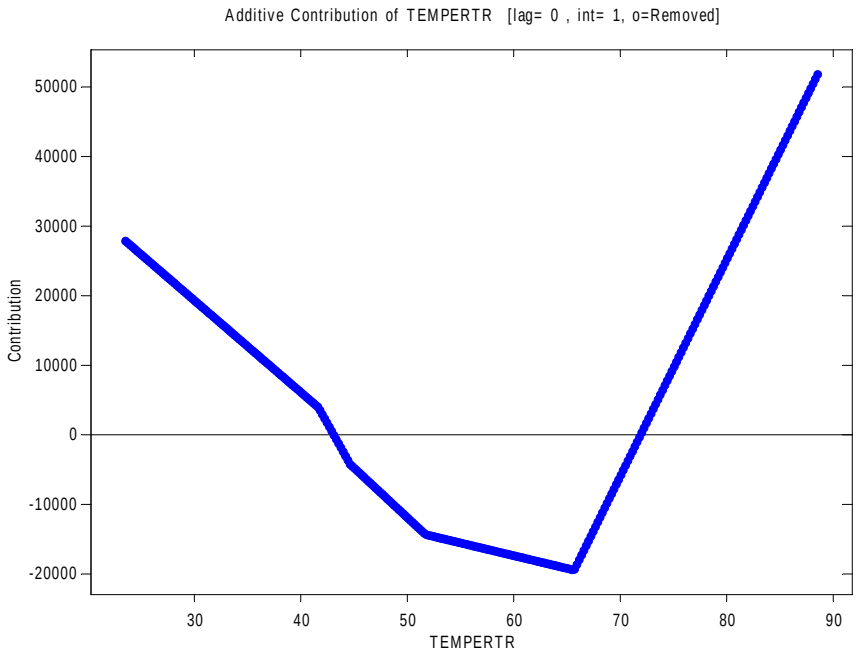
7.10 CONTRIB User Subroutine

The CONTRIB subroutine creates contribution charts for all explanatory variables. The charts are saved in Windows-Meta-File (WMF) format. The number of charts created is dependent upon the number of explanatory variables in the final model and the number of interactions in combination with the target contribution variable. If the variable is additive, a single chart will be created for that target contribution variable. Here, the contribution variable is plotted on the horizontal axis over its minimum to maximum range in the training data. The other variables in the model are either set to their median values or are effectively removed from the model depending upon the type of contribution/leverage chart being created. If the contribution variable has interaction with other variables, up to three contribution charts are created for the target variable where the other variables in the model are set to their minimum, median, and maximum values. In the future enhancement,

surface or mesh plots will be created for variables with interactions. The user may select from the following contribution chart types:

1. Leverage effect of a contribution variable on the predicted response holding all other variables in the model constant
2. Leverage effect of a unit increase in the contribution variable holding all other variables in the model constant. Computed as LEV(t)=YHat(t)-YHat(1) where YHat(1) is the predicted value when the contribution variable at its minimum.
3. Additive contribution of the contribution variable removing all other variables in the model. This option is most appropriate when the contribution variable is additive (i.e., no interactions).
4. Leverage effect of the knots (thresholds) of the contribution variable holding all other variables in the model constant. Computed as LEV(t)=YHat(t)-YHat(t-1) where YHat is the predicted value.
5. Cumulative leverage effect of a unit increase in the contribution variable holding all other variables in the model constant. Computed as LEV(t)=Csum(YHat(t))-(YHat(1)*t) where YHat(1) is the predicted value when the contribution variable at its minimum.

The CONTRIB subroutine creates contribution charts for all explanatory variables. The charts are saved in Windows-Meta-File (WMF) format. The number of charts created is dependent upon the number of explanatory variables in the final model and the number of interactions in combination with the target. An example of a contribution chart is provided below:



Usage:

```
CALL LOAD(CONTRIB     :WBSUPPL);
CALL CONTRIB(MEDIANS, MEANS, IVERSION);
```

Required subroutine arguments:

Arguments	Type(s)	Intent	Comments
MEDIANS	*Real number array(,p)*	*input*	The median values of the explanatory variables in the model to which the explanatory variables will be held constant
MEANS	*Real number array(,p)*	*input*	The mean values of the explanatory variables in the model to which the explanatory variables will be held constant
IVERSION	*Integer scalar*	*Input*	The type of contribution charts to create (1-5)

Example:

```
call contrib(_medians,_means,3);
```

8 EXAMPLES OF SCAB34S COMMAND FILES FOR MARSPLINE MODELING

SCA WorkBench was used to automatically generate the command files for the SCAB34S engine for MARSPLINE modeling, OLS modeling, Logit-Probit modeling, estimation, diagnostics, and graphics. This section presents the command files generated by WorkBench for the examples used in this document.

8.1 SCAB34S Commands Generated for the Electricity Load Example

```
b34sexec matrix;
call echooff;
call getsca('ELOAD.mad'
     :mad :member DATA);
call load(dispmars :staging);
call load(disp_hin 'C:\B34SLM\wbsuppl.mac');
call load(dsp_acf  'C:\B34SLM\wbsuppl.mac');
call load(coint2   'C:\B34SLM\wbsuppl.mac');
call load(dsp_tbl  'C:\B34SLM\wbsuppl.mac');
call load(contrib  'C:\B34SLM\wbsuppl.mac');
call print('-----------------------------------------------------':);
call print('** Analysis Performed on Variable: DAYLOAD':);
call print('-----------------------------------------------------':);

call dspdscrb('DAYLOAD Dependent Variable',DAYLOAD);

/$ ************************************************************$/
/$ Set span for sample: DAYLOAD
/$ ************************************************************$/
iorigins=integers(1, 730);
DAYLOAD = DAYLOAD(iorigins);
DATE = DATE(iorigins);
TEMPERTR = TEMPERTR(iorigins);
DOW = DOW(iorigins);
D1 = D1(iorigins);
D2 = D2(iorigins);
D3 = D3(iorigins);
D4 = D4(iorigins);
D5 = D5(iorigins);
D6 = D6(iorigins);

/$ ****************************************************
/$ _holdout is the number of forecasts
/$ _maxlag is the longest lag in model
/$ _nlags is number of lags for diagnostics
/$ _knots is the maximum number of basis function
/$ _mi is the maximum number of interactions
/$ _ms is the minimum span between each knot
/$ _ffmars holds the MARSPLINE forecasts
/$ _ffols holds the OLS forecasts for comparison
/$ _actuals holds the actuals for comparison
/$ _rxmdl holds the random X-variable model components
/$ ****************************************************
_holdout=0;
_nlags=12;
if((_holdout.le._nlags).and.(_holdout.gt.0)) _nlags=_holdout-1;
_maxlag=1;
_knots=60;
_mi=1;
_ms=0;
_df=2.0;
imax=1;
if(imax .le. _maxlag) imax=_maxlag+1;
 nn=integers(imax,norows(DAYLOAD));
```

```
 nnact=integers(imax,norows(DAYLOAD));
_actuals=vfam(DAYLOAD(nnact));


/$ ***************************************************************
/$  Specify model components using character string variables
/$ ***************************************************************

call character(_rxmdl,
'TEMPERTR{ 0 TO 1 }
DAYLOAD{ 1 }
D1
D2
D3
D4
D5
D6 '
);


call olsq(DAYLOAD argument(_rxmdl)  :diag);
ccomment=c1array(norows(%coef):);
call character(vnamea,'DAYLOAD');
call disp_ols(vnamea,%names,%lag,%coef,%se,%t,%nob,
%adjrsq,%sicstat,%rss,'OLS',ccomment,1);
_ffols=goodrow(vfam(%yhat));


/$ ******************************************************
/$ Specify MARSPLINE model
/$ ******************************************************

call marspline(DAYLOAD argument(_rxmdl)  :mathform :print :nk _knots
:mi _mi :df _df  :savemodel );
_ffmars=goodrow(vfam(%yhat));
_dmat=array(3,norows(_actuals):);
_dmat(1,)=_ffmars;
_dmat(2,)=_ffols;
_dmat(3,)=_actuals;
nnames=c1array(3:);
nnames(1)='MARSPLINE_';
nnames(2)='OLS_';
nnames(3)='DAYLOAD';
call character(_desc,
'Table of Actual vs. Predicted');
_nbegn=_holdout+_maxlag;
call dsp_tbl(_dmat,nnames,_nbegn,14,4,_desc);

/$ ************************************************************$/
/$ Compute performance criteria for prediction
/$ ************************************************************$/
marssq=sumsq(_actuals-_ffmars);
m_rmse=dsqrt(mean(afam(_actuals-_ffmars)**2.));
m_mape=mean(afam(dabs(_actuals-_ffmars))/afam(dabs(_actuals)))*100.;
olsssq=sumsq(_actuals-_ffols);
o_rmse=dsqrt(mean(afam(_actuals-_ffols)**2.));
o_mape=mean(afam(dabs(_actuals-_ffols))/afam(dabs(_actuals)))*100.;

call print('-----------------------------------------------------':);
call print('** Prediction Performance Criteria':);
call print('-----------------------------------------------------':);
call print('OLS RMSE: ', o_rmse :format '(f12.3)');
call print('OLS MAPE: ', o_mape :format '(f12.3)');
call print('MARSPLINE RMSE: ', m_rmse :format '(f12.3)');
call print('MARSPLINE MAPE: ', m_mape :format '(f12.3)');
res1=_ffmars - _actuals;
res2=_ffols - _actuals;
call graph(_actuals,_ffmars,_ffols :file 'yfit.wmf'
:heading 'Original vs. Predicted' :nolabel
```

```
        :colors black black bblue bred);
    call dspdscrb('MARSPLINE Residuals',res1);

    /$ ************************************************************$/
    /$ Compute ACF, PACF, Q-stats for model residuals
    /$ ************************************************************$/
    call dsp_acf('MARSPLINE Residuals',res1,_nlags,acfa,pacfa,sea,
    mqa,pmqa,-1,0,0);
    if(norows(res1).gt.50)then;
    call hinich82(res1,_mmars,_gmars,_lmars :meanonly);
    call disp_hin(_gmars,_lmars,'MARSPLINE Residuals');
    endif;
    call dspdscrb('OLS Residuals',res2);

    /$ ************************************************************$/
    /$ Compute ACF, PACF, Q-stats for model residuals
    /$ ************************************************************$/
    call dsp_acf('OLS Residuals',res2,_nlags,acfb,pacfb,seb,
    mqb,pmqb,-1,0,0);
    if(norows(res2).gt.50)then;
    call hinich82(res2,_mols,_gols,_lols :meanonly);
    call disp_hin(_gols,_lols,'OLS Residuals');
    endif;

    b34srun;.
    . The commands related to building graphics in SCAB34S omitted here for brevity. It is
    . a common script that will be presented later in this section.
    .
    .

b34srun;
```

8.2 SCAB34S Commands for the Nitrous Oxide Example

```
b34sexec scaio readsca file('nitrous.mad') dataset(DATA); b34srun;
b34sexec matrix;
call loaddata; call echooff;
call load(dispmars :staging);
call load(disp_hin 'C:\B34SLM\wbsuppl.mac');
call load(dsp_acf  'C:\B34SLM\wbsuppl.mac');
call load(coint2   'C:\B34SLM\wbsuppl.mac');
call load(dsp_tbl  'C:\B34SLM\wbsuppl.mac');
call print('----------------------------------------------------------':);
call print('** Analysis Performed on Variable: Y':);
call print('----------------------------------------------------------':);

call dspdscrb('Y Dependent Variable',Y);

/$ ************************************************************$/
/$ Set span for sample: Y
/$ ************************************************************$/
iorigins=integers(1, 88);
Y = Y(iorigins);
C_RATIO = C_RATIO(iorigins);
E_RATIO = E_RATIO(iorigins);

/$ ******************************************************
/$ _holdout is the number of forecasts
/$ _maxlag is the longest lag in model
/$ _nlags is number of lags for diagnostics
/$ _knots is the maximum number of basis function
/$ _mi is the maximum number of interactions
/$ _ms is the minimum span between each knot
/$ _ffmars holds the MARSPLINE forecasts
/$ _ffols holds the OLS forecasts for comparison
/$ _actuals holds the actuals for comparison
/$ _rxmdl holds the random X-variable model components
/$ ******************************************************
_holdout=0;
_nlags=12;
if((_holdout.le._nlags).and.(_holdout.gt.0)) _nlags=_holdout-1;
_maxlag=0;
_knots=25;
_mi=3;
_ms=0;
_df=2.0;
imax=1;
if(imax .le. _maxlag) imax=_maxlag+1;
 nn=integers(imax,norows(Y));
 nnact=integers(imax,norows(Y));
_actuals=vfam(Y(nnact));


/$ ************************************************************
/$  Specify model components using character string variables
/$ ************************************************************

call character(_rxmdl, 'C_RATIO E_RATIO ');


call olsq(Y argument(_rxmdl)  :diag);
ccomment=c1array(norows(%coef):);
call character(vnamea,'Y');
call disp_ols(vnamea,%names,%lag,%coef,%se,%t,%nob,
%adjrsq,%sicstat,%rss,'OLS',ccomment,1);
_ffols=goodrow(vfam(%yhat));
```

```
/$ ******************************************************
/$ Specify MARSPLINE model
/$ ******************************************************

call marspline(Y argument(_rxmdl)  :print :nk _knots :mi _mi :df _df  );
_ffmars=goodrow(vfam(%yhat));

/$ ******************************************************
/$ Build labels for actual vs. predicted output table
/$ ******************************************************

_dmat=array(3,norows(_actuals):);
_dmat(1,)=_ffmars;
_dmat(2,)=_ffols;
_dmat(3,)=_actuals;
nnames=c1array(3:);
nnames(1)='MARSPLINE_';
nnames(2)='OLS_';
nnames(3)='Y';
call character(_desc,
'Table of Actual vs. Predicted');
_nbegn=_holdout+_maxlag;
call dsp_tbl(_dmat,nnames,_nbegn,14,4,_desc);

/$ ************************************************************$/
/$ Compute performance criteria for prediction
/$ ************************************************************$/
marssq=sumsq(_actuals-_ffmars);
m_rmse=dsqrt(mean(afam(_actuals-_ffmars)**2.));
m_mape=mean(afam(dabs(_actuals-_ffmars))/afam(dabs(_actuals)))*100.;
olsssq=sumsq(_actuals-_ffols);
o_rmse=dsqrt(mean(afam(_actuals-_ffols)**2.));
o_mape=mean(afam(dabs(_actuals-_ffols))/afam(dabs(_actuals)))*100.;

call print('-----------------------------------------------------':);
call print('** Prediction Performance Criteria':);
call print('-----------------------------------------------------':);
call print('OLS RMSE: ', o_rmse :format '(f12.3)');
call print('OLS MAPE: ', o_mape :format '(f12.3)');
call print('MARSPLINE RMSE: ', m_rmse :format '(f12.3)');
call print('MARSPLINE MAPE: ', m_mape :format '(f12.3)');
res1=_ffmars - _actuals;
res2=_ffols - _actuals;

/$ ************************************************************$/
/$ Show descriptive summary statistics for MARSPLINE model residuals
/$ ************************************************************$/

call dspdscrb('MARSPLINE Residuals',res1);

/$ ************************************************************$/
/$ Compute ACF, PACF, Q-stats for model residuals for MARSPLINE model
/$ ************************************************************$/
call dsp_acf('MARSPLINE Residuals',res1,_nlags,acfa,pacfa,sea,
mqa,pmqa,-1,0,0);
if(norows(res1).gt.50)then;
call hinich82(res1,_mmars,_gmars,_lmars :meanonly);
call disp_hin(_gmars,_lmars,'MARSPLINE Residuals');
endif;
call dspdscrb('OLS Residuals',res2);

/$ ************************************************************$/
/$ Compute ACF, PACF, Q-stats for model residuals for OLS model
/$ ************************************************************$/
call dsp_acf('OLS Residuals',res2,_nlags,acfb,pacfb,seb,
mqb,pmqb,-1,0,0);
if(norows(res2).gt.50)then;
call hinich82(res2,_mols,_gols,_lols :meanonly);
call disp_hin(_gols,_lols,'OLS Residuals');
```

```
     endif;.
     .
     . The commands related to building graphics in SCAB34S omitted here for brevity. It is
     . a common script that will be presented later in this section.
     .
     .

b34srun;
```

8.3 SCAB34S Commands Generated for the MARS-Probit Example

```
b34sexec matrix;
call echooff;
call getsca('MURDER.mad'
     :mad :member DATA);
call load(dispmars :staging);
call load(disp_hin :wbsuppl);
call load(dsp_acf  :wbsuppl);
call load(coint2   :wbsuppl);
call load(dsp_tbl  :wbsuppl);
call load(contrib  :wbsuppl);
call load(catbuild :wbsuppl);
call load(tlogit   :staging);
call load(liftgain :wbsuppl);
call load(disp_lgt :wbsuppl);
call print('-----------------------------------------------------':);
call print('** Analysis Performed on Variable: D1':);
call print('-----------------------------------------------------':);

call dspdscrb('D1 Dependent Variable',D1);

/$ ************************************************************$/
/$ Set span for sample: D1
/$ ************************************************************$/
iorigins=integers(1, 44);
D1 = D1(iorigins);
D2 = D2(iorigins);
D3 = D3(iorigins);
LF = LF(iorigins);
M = M(iorigins);
N = N(iorigins);
NW = NW(iorigins);
PC = PC(iorigins);
PX = PX(iorigins);
TM = TM(iorigins);
Y = Y(iorigins);

/$ ****************************************************
/$ _holdout is the number of forecasts
/$ _maxlag is the longest lag in model
/$ _nlags is number of lags for diagnostics
/$ _knots is the maximum number of basis function
/$ _mi is the maximum number of interactions
/$ _ms is the minimum span between each knot
/$ _ffmars holds the MARSPLINE forecasts
/$ _ffprbt holds the PROBIT forecasts for comparison
/$ _actuals holds the actuals for comparison
/$ _rxmdl holds the random X-variable model components
/$ ****************************************************
_holdout=0;
_nlags=12;
if((_holdout.le._nlags).and.(_holdout.gt.0)) _nlags=_holdout-1;
_maxlag=0;
_upper=.501;
_lower=.501;
_ncut=20;
_iprobset=1;
_knots=40;
```

```
_mi=2;
_ms=0;
_df=2.0;
call character(_desc1,
'Performance Evaluation for MARSPLINE');

call character(_desc2,
'Performance Evaluation for PROBIT');
imax=1;
if(imax .le. _maxlag) imax=_maxlag+1;
 nn=integers(imax,norows(D1));
 nnact=integers(imax,norows(D1));
_actuals=vfam(D1(nnact));


/$ ************************************************************
/$  Specify model components using character string variables
/$ ************************************************************

call character(_rxmdl,
'TM Y LF NW D2 ');

call probit(D1 argument(_rxmdl)  :print );
%lmatvar=%names;
%lmatlag=%lag;
_logparm=%coef;
_logse=%se;
_logt=%t;
_func=%func;

_ffprbt=goodrow(vfam(%yhat));
call disp_lgt(%lmatvar,%lmatlag,_logparm,_logse,_logt,_func,'PROBIT',0);

_ffprbt=vfam(_ffprbt);
_iprint=2;
call tlgtr(_actuals,_ffprbt,_ncut,_ptruer,_pfalser,_pFPos,_pFFals,
           _acc,_pprecise, _gmean1,_gmean2,rprob1,rprob2,_upper,_lower,
           _desc2,_iprobSet,3,_iprint);

call character(_desc2b, 'PROBIT Lift-Gain Table');
call liftgain(_ffprbt,_actuals,_desc2b,npt,nt,lgtgain,lgtlift);

/$ ****************************************************
/$ Specify MARSPLINE model
/$ ****************************************************

call marspline(D1 argument(_rxmdl) :probit :savebx  :mathform :nk
_knots :mi _mi :df _df  :savemodel );

/$ ************************************************
/$ * Adjust coef, se, a t-vals under Probit assumptions
/$ ************************************************
call probit(D1 %probitx  :noint :print);
coefnew=%coef;
senew=%se;
tnew=%t;
yhatnew=probnorm(%yhat);

/$ ************************************************
/$ * Save adjusted probit information into Marspline model
/$ ************************************************
call restore(:file 'marss.psv');
%coef=coefnew;
%se=senew;
%t=tnew;
%yhat=yhatnew;
call checkpoint(:file 'marszero.psv' :var
%BESTIN %FLAG %DIR %CUT %YVAR %NAMES %TYPEVAR
%LAG %COEF %T %MINVAR %MAXVAR %K %NOB %RSS %SUMRE
```

```
%REMAX %RESVAR %MARS_VR %SE %MODTYPE %YHAT %NK);

/$ ************************************************
/$ * Display the final MarsProbit model
/$ ************************************************
call marspline(D1 argument(_rxmdl) :probit :savebx  :mathform :nk
_knots :mi _mi :df _df  :getmodel 'marszero.psv'
 :print );
call dispmars;

_ffmars=goodrow(vfam(%yhat));

_iprint=2;
call tlgtr(_actuals,_ffmars,_ncut,_ptruer,_pfalser,_pFPos,_pFFals,
           _acc,_pprecise, _gmean1,_gmean2,rprob1,rprob2,_upper,_lower,
           _desc1,_iprobSet,1,_iprint);

call character(_desc1b, 'MARSPLINE Lift-Gain Table');
call liftgain(_ffmars,_actuals,_desc1b,npt,nt,margain,marlift);

res1=_ffmars - _actuals;
res2=_ffprbt - _actuals;
call graph(_actuals,_ffmars,_ffprbt :file 'yfit.wmf'
:heading 'Original vs. Predicted' :nolabel
    :colors black black bblue bred);
call grflift(margain,marlift,lgtgain,lgtlift,1,3,2);


. The commands related to building graphics in SCAB34S omitted here for brevity. It is
. a common script that will be presented later in this section.
.
.


/$ ************************************************************$/
/$ Create contribution charts for righthand-side variables
/$ ************************************************************$/
call lagmatrix( argument(_rxmdl)
               :noint :matrix tmat);
_medians=array(nocols(tmat):);
_means=_medians;

do i=1,norows(_medians);
 call describe(tmat(,i));
 _medians(i)=%median;
 _means(i)=%mean;
enddo;

call contrib(_medians,_means,3);

b34srun;
```

8.4 SCAB34S Commands Used to Display Graphs in the Examples

```
/$ ****************************************************
/$ Display graphics for Dependent Variable
/$ ****************************************************
call graph(DAYLOAD :file 'yvar.wmf' :noshow
     :pspaceon
     :pgyscaleright 'i'
     :pgborder
     :pgxscaletop 'i'
     :nocontact
     :colors black bblue
     :heading 'Plot of DAYLOAD');

/$****************************************************
/$ Display graphics for MARSPLINE Residuals
/$ ****************************************************
call graph(res1  :file 'resa.wmf' :noshow
     :pspaceon
     :pgyscaleright 'i'
     :pgborder
     :pgxscaletop 'i'
     :nocontact
     :colors black bblue
     :heading 'Plot of MARSPLINE Model Residuals');
call graph(acfa, sea  :file 'acfa.wmf' :noshow
     :pspaceon
     :pgyscaleright 'i'
     :pgborder
     :pgxscaletop 'i'
     :histscale integers(0,_nlags,2)
     :overlay acfplot
     :colors black bblue bred
     :heading 'ACF of MARSPLINE Model Residuals');
call graph(mqa  :file 'mqa.wmf' :noshow
     :pspaceon
     :pgyscaleright 'i'
     :pgborder
     :pgxscaletop 'i'
     :nocontact
     :colors black bblue
     :heading 'Q-Stats from MARSPLINE Model Residuals');

/$****************************************************
/$ Display graphs for OLSQ model residuals
/$ ****************************************************
call graph(res2 :file 'resb.wmf' :noshow
     :pspaceon
     :pgyscaleright 'i'
     :pgborder
     :pgxscaletop 'i'
     :nocontact
     :colors black bblue
     :heading 'Plot of OLSQ Model Residuals');
call graph(acfb, seb :file 'acfb.wmf' :noshow
     :pspaceon
     :pgyscaleright 'i'
     :pgborder
     :pgxscaletop 'i'
     :histscale integers(0,_nlags,2)
     :overlay acfplot
     :colors black bblue bred
     :heading 'ACF of OLSQ Residuals');
call graph(mqb  :file 'mqb.wmf' :noshow
     :pspaceon
     :pgyscaleright 'i'
```

```
 :pgborder
 :pgxscaletop 'i'
 :nocontact
 :colors black bblue
 :heading 'Q-Stats from OLSQ Residuals');\

/$ ************************************************************$/
/$ Create contribution charts for righthand-side variables
/$ ************************************************************$/
call lagmatrix( argument(_rxmdl)
               :noint :matrix tmat);
_medians=array(nocols(tmat):);
_means=_medians;

do i=1,norows(_medians);
 call describe(tmat(,i));
 _medians(i)=%median;
 _means(i)=%mean;
enddo;

call contrib(_medians,_means,3);
```

REFERENCES

Breiman, L. (1991a). "The Π Method for Estimating Multivariate Functions from Noisy Data.". *Technometrics* 33: 125-143

Breiman, L. (1991b). "Response.". *Technometrics* 33: 155-160

Breiman, L. (1991c). "Discussion.". *Annals of Statistics* 19: 82-91

Faraway, J. (2006). *Extending the Linear Model with R,* New York: Chapman & Hall/CRC

Friedman, J. (1991a). "Discussion". *Technometrics* 33: 145-148

Friedman, J. (1991b). "Multivariate Adaptive Regression Splines". *Annals of Statistics* 19: 1-67

Friedman, J. (1991c). "Rejoinder". *Annals of Statistics* 19: 123-141

Hastie, T. and Tibshirani, R. (1990). *Generalized Additive Models*. Chapman & Hall

Hinich, M. (1982). "Testing for Gaussianity and Linearity of a Stationary Time Series". *Journal of Time Series Analysis* 3: 169-176

Kubat, M. and Matwin, S. (1997). "Addressing the Curse of Imbalanced Training Sets: One Sided Selection". In *Proceedings of the Fourteenth International Conference on Machine Learning*: 179–186. Nashville, Morgan Kaufmann.

Kubat, M. and Holte, R. and Matwin, S. (1998). "Machine Learning for the Detection of Oil Spills in Satellite Radar Images". *Machine Learning* 30: 195–215.

Lattyak, W.J. and Liu, L-M. (2005). *SCA WorkBench User's Guide*. Scientific Computing Associates Corp.

Lewis, P.A.W. and Stevens, J.G. (1991). "Nonlinear Modeling of Time Series using Multivariate Adaptive Regression Splines (MARS)". *Journal of the American Statistical Association* 86: 864-877.

Ljung, G.M. and Box, G.E.P. (1978). "On a Measure of Lack of Fit in Time Series Models". *Biometrika* 66: 265-270

Maddala, G. S. (1992) *Introduction to Econometrics*. Second Edition. MacMillan Press.

McManus, W.S. (1985). "Estimates of the Deterrent Effect of Capital Punishment: The Importance of the Researcher's Prior Beliefs". *Journal of Political Economy* 93: 416-425

Provost, F., Fawcett, T. and Kohavi, R. (1998). "The Case Against Accuracy Estimation for Comparing Induction Algorithms". In *Proceedings of the Fifteenth International Conference on Machine Learning*: 445–453. San Francisco, Morgan Kaufmann.

Stokes, H.H. (2004) *The B34S ProSeries Econometric System* (software).

Stokes, H.H. (2004). "On the advantage of using two or more econometric software systems to solve the same problem". *Journal of Economic and Social Measurement* 29: 307–320

Stokes, H.H. (1997). *Specifying and Diagnostically Testing Econometric Models*. Second Edition. New York, Quorum Books.

Stokes, H.H. (2002) “Chapter 16: Programming in the Matrix Command" in *Specifying and Diagnostically Testing Econometric Models,Volume II*. (manuscript).