SANDIA REPORT
SAND2006-4693
Unlimited Release
Printed September 2006

Multiple Predictor Smoothing Methods For Sensitivity Analysis

Curtis B. Storlie and Jon C. Helton

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under Contract DE-AC04-94AL85000.

Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865)576-8401
Facsimile: (865)576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.doc.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd.
Springfield, VA 22161

Telephone: (800)553-6847
Facsimile: (703)605-6900
E-Mail: orders@ntis.fedworld.gov
Online ordering: http://www.ntis.gov/ordering.htm

SAND2006-4693
Unlimited Release
Printed September 2006

Multiple Predictor Smoothing Methods for Sensitivity Analysis

Curtis B. Storlie[a] and Jon C. Helton[b]
[a]Department of Statistics, North Carolina State University, Raleigh, NC 27695-8203 USA
[b]Department of Mathematics and Statistics, Arizona State University, Tempe, AZ 85287-1804 USA

Abstract

The use of multiple predictor smoothing methods in sampling-based sensitivity analyses of complex models is investigated. Specifically, sensitivity analysis procedures based on smoothing methods employing the stepwise application of the following nonparametric regression techniques are described: (i) locally weighted regression (LOESS), (ii) additive models, (iii) projection pursuit regression, and (iv) recursive partitioning regression. The indicated procedures are illustrated with both simple test problems and results from a performance assessment for a radioactive waste disposal facility (i.e., the Waste Isolation Pilot Plant). As shown by the example illustrations, the use of smoothing procedures based on nonparametric regression techniques can yield more informative sensitivity analysis results than can be obtained with more traditional sensitivity analysis procedures based on linear regression, rank regression or quadratic regression when nonlinear relationships between model inputs and model predictions are present.

Key Words: Additive models, Epistemic uncertainty, Locally weighted regression, Nonparametric regression, Projection pursuit regression, Recursive partitioning regression, Scatterplot smoothing, Sensitivity analysis, Stepwise selection, Uncertainty analysis.

Acknowledgements

Work performed for Sandia National Laboratories (SNL), which is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Security Administration under contract DE-AC04-94AL-85000. Review at SNL provided by L. Swiler and C. Sallaberry. Editorial support provided by F. Puffer, K. Best, M. Spielman, and J. Ripple of Tech Reps, a division of Ktech Corporation.

Contents

Figures

Tables

This page intentionally left blank.

1. Introduction

The importance of uncertainty analysis and sensitivity analysis as components of analyses for complex systems is almost universally recognized, where uncertainty analysis designates the determination of the uncertainty in analysis results that derives from the uncertainty in analysis inputs and sensitivity analysis designates the determination of the contributions of individual uncertain analysis inputs to the uncertainty in analysis results.[1-11] A number of approaches to uncertainty and sensitivity analysis have been developed, including differential analysis,[12-17] response surface methodology,[18-26] Monte Carlo analysis,[27-38] and variance decomposition procedures.[39-43] Overviews of these approaches are available in several reviews.[44-52]

The focus of this presentation is on Monte Carlo (i.e., sampling-based) approaches to uncertainty and sensitivity analysis. Such analyses involve the consideration of models of the form

$$\mathbf{y} = \mathbf{f}(\mathbf{x}), \tag{1.1}$$

where

$$\mathbf{y} = [y_1, y_2, \ldots, y_{nY}] \tag{1.2}$$

is a vector of analysis results and

$$\mathbf{x} = [x_1, x_2, \ldots, x_{nX}] \tag{1.3}$$

is a vector of imprecisely known analysis inputs. In general, the model $\mathbf{f}$ can be quite large and involved (e.g., a system of nonlinear partial differential equations requiring numerical solution (see Ref. 53) or possibly a sequence of complex, linked models as is the case in a probabilistic risk assessment for a nuclear power plant (see Refs. 54, 55) or a performance assessment for a radioactive waste disposal facility (see Refs. 56, 57)); the vector $\mathbf{y}$ of analysis results can be of high dimension and complex structure (e.g., the elements of $\mathbf{y}$ might be several hundred temporally or spatially dependent functions); and the vector $\mathbf{x}$ of analysis inputs can also be of high dimension and complex structure (e.g., several hundred variables, with some variables corresponding to physical properties of the system under study and other variables corresponding to parameters in probability distributions or perhaps to designators for alternative models).

The uncertainty in the elements of $\mathbf{x}$ is characterized by a sequence of probability distributions

$$D_1, D_2, \ldots, D_{nX}, \tag{1.4}$$

where D_j is a probability distribution characterizing the uncertainty in x_j. Correlations and other restrictions involving the relations between the x_j are also possible. Such distributions and any associated restrictions are intended to numerically capture the existing knowledge about the elements of $\mathbf{x}$ and are often developed through an expert review process.[58-73]

The uncertainty characterized by the distributions $D_1, D_2, \ldots, D_{nX}$ in Eq. (1.4) is often referred to as epistemic uncertainty. Alternate designations for epistemic uncertainty include state of knowledge, subjective, reducible, and type B.[74-82] In particular, epistemic uncertainty derives from a lack of knowledge about the appropriate value to use for a quantity that is assumed to have a fixed value in the context of a particular analysis. In the conceptual and computational organization of an analysis, epistemic uncertainty is generally considered to be distinct from aleatory uncertainty, which arises from an inherent randomness in the behavior of the system under study.[74-83]

Sampling-based uncertainty and sensitivity analyses are based on a sample

$$\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{i,nX}], i = 1, 2, \ldots, nS, \tag{1.5}$$

from the possible values for $\mathbf{x}$ generated in consistency with the distributions in Eq. (1.4) and any associated restrictions. Random sampling is one possibility for the generation of this sample. However, owing to its efficient stratification properties, Latin hypercube sampling is widely used in analyses of this type, especially when computationally intensive models are involved.[27, 37, 38]

The analysis evaluations

$$\mathbf{y}_i = \mathbf{y}(\mathbf{x}_i) = \mathbf{f}(\mathbf{x}_i), i = 1, 2, \ldots, nS, \tag{1.6}$$

provide a mapping between analysis inputs (i.e., $\mathbf{x}_i$) and analysis results (i.e., $\mathbf{y}_i$) that forms the basis for both uncertainty analysis and sensitivity analysis. Once the preceding mapping is available, the determination of uncertainty analysis results is generally straightforward and involves the generation of summary results such as histograms, density functions, cumulative distribution functions (CDFs), complementary cumulative distribution functions (CCDFs), and box plots for individual elements of $\mathbf{y}$ (Sect. 6.5, Ref. 29). The determination of sensitivity analysis results involves the exploration of the preceding mapping with techniques such as ex-

amination of scatterplots, regression analysis, correlation and partial correlation analysis, and searches for nonrandom patterns (Sect. 6.6, Ref. 29).

The determination of sensitivity analysis results is generally more demanding than the determination of uncertainty analysis results. In particular, the popular regression and correlation based techniques can fail to appropriately identify the effects of the individual elements of **x** on the elements of **y** when nonlinear and nonmonotonic relations are present (Sect. 6.6, Ref. 29). Possible approaches to sensitivity analysis to use in such situations include grid-based statistical analyses of scatterplots,[30, 84] distance-based statistical analyses of scatterplots,[85-98] multidimensional Kolmogorov-Smirnov tests,[99-102] rank-concordance tests,[103, 104] and classification trees.[105, 106] However, the preceding approaches lack the intuitive appeal of regression-based approaches to sensitivity analysis. In particular, regression-based sensitivity analysis can be carried out in a sequential manner with variable importance being indicated by the order in which variables enter the regression model and by the fraction of total variance that can be accounted for as successive variables enter the regression model.

The purpose of this presentation is to describe regression-based techniques for sensitivity analysis that are based on multiple predictor smoothing methods. Such methods are conceptually consistent with regression-based methods that have been widely used in the past in sensitivity analysis (Sect. 6.6, Ref. 29), but have the important advantage that they are capable of incorporating local changes in the relationship between a dependent variable (i.e., an element of **y**) and multiple independent variables (i.e., elements of **x**). As a result, these methods can be successfully applied in situations involving nonlinear relationships between analysis inputs and analysis results where more traditional regression-based approaches would fail to appropriately capture these relationships.

The presentation is organized as follows. First, traditional approaches to regression-based sensitivity are briefly described (Sect. 2), and then nonparametric approaches to regression analysis based on local data smoothing are introduced (Sect. 3). Next, technical details related to the implementation of the techniques described in Sect. 3 to a sequence of example sensitivity analyses are described (Sect. 4), and the results of these examples are presented (Sect. 5). The presentation concludes with a summary discussion (Sect. 6).

Although analyses for real systems almost always involve multiple output variables as indicated in conjunction with Eqs. (1.1) – (1.3), the following discussions assume that a single real-valued result of the form

$$y = f(\mathbf{x}) \tag{1.7}$$

is under consideration. Similarly,

$$y_i = f(\mathbf{x}_i), i = 1, 2, \ldots, nS, \tag{1.8}$$

is used to represent the result of evaluating y with the sample in Eq. (1.5). This simplifies the notation and results in no loss in generality as the results under discussion are valid for individual elements of **y**. All statistical analyses in this presentation are carried out within the *R* statistical computing environment,[107] which is an open source equivalent to the *S*-Plus statistical package.[108]

2. Traditional Parametric Regression Models

Several parametric regression models used in sensitivity analysis are briefly reviewed. More information on such models can be obtained in a number of excellent texts (e.g., Ref. 109-113).

2.1 Linear Regression

Linear regression has long been the method of choice for researchers wishing to approximate a surface. This regression model is predicated on a relation of the form

$$y = \beta_0 + \sum_{j=1}^{nX} \beta_j x_j + \varepsilon, \tag{2.1}$$

where ε is a random error term with an expected value of zero (i.e., $E(\varepsilon) = 0$).

The approximate form of the relation in Eq. (2.1) is

$$\hat{y} = b_0 + \sum_{j=1}^{nX} b_j x_j, \tag{2.2}$$

where the b_j are typically estimated with least squares procedures from observations of the form $[\mathbf{x}_i, y_i]$, $i = 1, 2, \ldots, nS$. In turn, the preceding approximation is often algebraically reformulated as

$$(\hat{y} - \bar{y})/\hat{s} = \sum_{j=1}^{nX} \left(b_j \hat{s}_j / \hat{s}\right)\left(x_j - \bar{x}_j\right)/\hat{s}_j, \tag{2.3}$$

where

$$\bar{y} = \sum_{i=1}^{nS} y_i / nS, \quad \hat{s} = \left[\sum_{i=1}^{nS} (y_i - \bar{y})^2 / (nS - 1)\right]^{1/2}$$

$$\bar{x}_j = \sum_{i=1}^{nS} x_{ij} / nS, \quad \hat{s}_j = \left[\sum_{i=1}^{nS} \left(x_{ij} - \bar{x}_j\right)^2 / (nS - 1)\right]^{1/2}.$$

The coefficients $b_j \hat{s}_j / \hat{s}$ in Eq. (2.3) are called standardized regression coefficients. When the x_j are independent, $|b_j \hat{s}_j / \hat{s}|$ can be used as a measure of variable importance. Specifically, $|b_j \hat{s}_j / \hat{s}|$ indicates the effect of moving a variable away from its expected value by a fixed fraction of its standard deviation while holding all other variables fixed at their expected values. Statistical tests can be used to indicate if the coefficients in Eqs. (2.2) and (2.3) appear to be different from zero. However, in the context of sensitivity analysis, it is important to recognize that such tests are simply one form of guidance with respect to variable importance as the underlying distributional assumptions with respect to the error term ε are not satisfied when deterministic models are under consideration (Sect. 6.6.3, Ref. 29).

The following identity holds when the relation in Eq. (2.2) is estimated with least squares procedures:

$$\sum_{i=1}^{nS} (y_i - \bar{y})^2 = \sum_{i=1}^{nS} (\hat{y}_i - \bar{y})^2 + \sum_{i=1}^{nS} (\hat{y}_i - y_i)^2, \tag{2.4}$$

where $\hat{y}_i$ denotes the estimate of y_i obtained from the regression model (Sect. 3.4, Ref. 109). In order from left to right, the three summations in the preceding equation are referred to as the total sum of squares (SS_{tot}), the regression sum of squares (SS_{reg}), and the residual sum of squares (SS_{res}). Since SS_{res} provides a measure of variability about the regression model,

$$R^2 = SS_{reg} / SS_{tot} = \sum_{i=1}^{nS} (\hat{y}_i - \bar{y})^2 \Big/ \sum_{i=1}^{nS} (y_i - \bar{y})^2 \tag{2.5}$$

provides a measure of the extent to which the regression model can match the observed results. Specifically, R^2 is close to 1 when the variation about the regression model is small (i.e., when SS_{res} is small relative to SS_{tot}), which indicates that the regression model is successful in matching the observed results. Similarly, R^2 is close to 0 when the variation about the regression model is large (i.e., when SS_{reg} is small relative to SS_{tot}), which indicates that the regression model is not successful in matching the observed results.

When linear regression is used as a sensitivity analysis technique, the regression is usually performed in a stepwise manner (Sect. 6.6.4, Ref. 29). With this approach, the most influential variable is added to the model first (producing a model of the form in Eq. (2.2) with one independent variable); then the next most influential variable is added to the model (producing a model of the form in Eq. (2.2) with two independent variables); and the process is continued in this manner until no more influential variables can be identified. Variable importance is then indicated by the order in which variables entered the regression model, the changes in R^2 values as successive variables entered the regression model, and the standardized regression coefficients for the variables in the final regression model.

However, it is important to recognize that standardized regression coefficients can produce very misleading indications of variable importance when highly correlated variables are included in the regression model (Sect. 6.6.7, Ref. 29).

An important special situation exists when the values for the x_j used in construction of the regression model in Eq. (2.2) (i.e., in the sample in Eq. (1.5)) are independent. Technically, this is equivalent to $\mathbf{X}^T\,\mathbf{X}$ being a diagonal matrix, where

$$\mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1,nX} \\ 1 & x_{21} & x_{22} & \cdots & x_{2,nX} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{nS,1} & x_{nS,2} & & x_{nS,nX} \end{bmatrix}. \qquad (2.6)$$

In this situation,

$$R^2 = R_1^2 + R_2^2 + \cdots + R_{nX}^2, \qquad (2.7)$$

where R_j^2 is the R^2 value that results from regressing y on only x_j (p. 99, Ref. 110). Thus, R_j^2 is the contribution of x_j to R^2 when the sampled inputs are independent (i.e., when the design matrix $\mathbf{X}$ is orthogonal). As a result, the incremental R^2 values in a stepwise regression are equal to the contributions of the individual independent variables to the total R^2 value for the regression.

There are many favorable properties of linear regression such as computational speed and interpretability. Hypothesis testing for input variable importance can be performed with ease. When the surface to be approximated is nearly linear in the inputs (i.e., the x_j), there is no better technique. However, in situations where the underlying relationship (i.e., the model in Eq. (1.7)) is far from linear, linear regression will produce a very poor approximation (Fig. 1). As a result, a number of alternatives to linear regression have been developed, including rank regression (Sect. 2.2), quadratic regression (Sect. 2.3), and nonlinear regression (Sect. 2.4).

The results in Fig. 1 come from an uncertainty and sensitivity analysis carried out for a two phase fluid flow model. This analysis will be described in greater detail in Sect. 5.2 where it is used to illustrate multiple predictor smoothing methods. This analysis involved 31 uncertain variables (i.e., $nX = 31$ in Eq. (1.3)). The regression line in Fig. 1 involves only one uncertain variable. Owing to the extreme nonlinearity of the relationships involved, the inclusion of additional uncertain

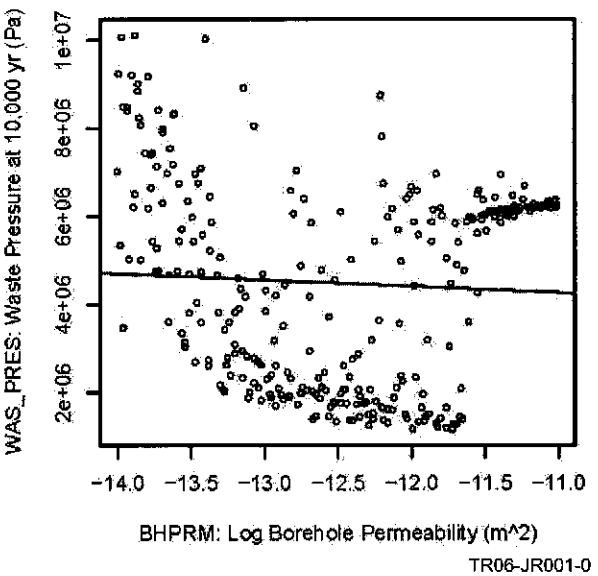


Fig. 1. Linear regression on results generated in a sensitivity analysis of a two-phase fluid flow model.

variables in the regression model fails to produce a satisfactory representation. For example, use of an α-value cutoff of 0.02 for entry of a variable into the regression model produces a model with five variables and an R^2 value of only 0.27. Additional discussion is given in Sect. 5.2.6.

2.2 Rank Regression

The results obtained with linear regression can often be improved with suitable transformations of the independent (i.e., y) and dependent (i.e., x_1, x_2, ..., x_{nX}) variables. For example, logarithmic or square root transformations may make the underlying relationships more linear and hence more amenable to analysis by linear regression. The identification of effective transformations is often subjective and thus difficult to automate. As a result, the effective use of transformations in a large sensitivity study can be difficult due to the large number of independent and dependent variables under consideration.

One broadly applicable transformation is the rank transformation, which is effective when the relationships between independent and dependent variables are monotonic (Ref. 114; Sect. 6.6.6, Ref. 29). The use of the rank transformation in conjunction with linear regression is straightforward. The smallest value of a variable is given a rank of 1; the next largest value is given a rank of 2; and so on up to the largest value

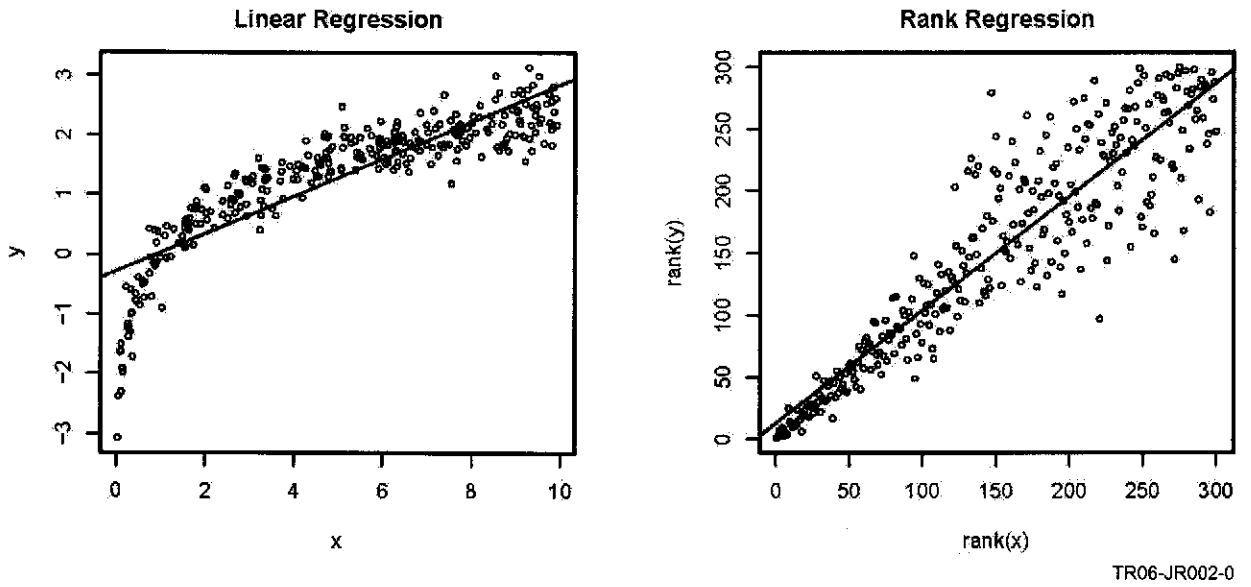


Fig. 2. Rank regression on an example monotonic relationship.

which is given a rank of nS, where nS is the sample size. Equal variable values are assigned the average of what their ranks would have been. Then, the usual regression procedures are carried out with the original variable values replaced by their ranks (Ref. 114; Sect. 6.6.6, Ref. 29).

The rank transformation converts monotonic relationships into linear relationships (Fig. 2). As a result, a linear regression in this situation with rank transformed data (i.e., a rank regression) provides a better approximation to the underlying relationships than would be obtained with a linear regression on the original (i.e., raw) data. Rank regressions have been successfully used in a large number of sensitivity analyses (e.g., Refs. 115-117). However, rank regressions cannot significantly improve the quality of a regression analysis when the underlying relations are nonlinear and nonmonotonic (Fig. 3).

2.3 Quadratic Regression

Quadratic regression is used as a designator for linear regression that includes individual variables (i.e., the x_j), variable squares (i.e., x_j^2), and multiplicative interaction terms (i.e., $x_j x_k$). Formally, quadratic regression is predicated on a model of the form

$$y = \alpha + \sum_{j=1}^{nX} \left(\beta_j x_j + \beta_{jj} x_j^2 \right) + \sum_{j=1}^{nX} \sum_{k=j+1}^{nX} \beta_{jk} x_j x_k + \varepsilon. \quad (2.8)$$

More generally, polynomial regression models that involve additional powers of the x_j and more complex multiplicative interaction terms are also possible.

Quadratic regression removes the assumption that the effects of the individual x_j are completely additive but still cannot model completely general interactions. Further, quadratic regression has difficulty representing functions with asymptotes and other complex behavior. Still, quadratic regression has been used with considerable success in industrial applications for many years.[20, 118]

A quadratic regression model (Fig. 4) shows significant improvement over the results previously shown for the application of linear and rank regression to a nonlinear and nonmonotonic relationship (Figs. 1, 3).

2.4 Nonlinear Regression

Nonlinear regression involves estimating the coefficients in a nonlinear relationship between y and the independent variables under consideration.[119] In particular, the regression models introduced in Sects. 2.1 – 2.3 are referred to as linear models because y is expressed as a linear combination of the variables in the regression model. In contrast, nonlinear regression involves estimating the coefficients β_j, $j = 0, 1, 2, \ldots$, in a hypothesized relationship such as

$$y = \beta_0 + \beta_1 x + \beta_2 \exp(\beta_3 x) + \varepsilon, \quad (2.9)$$

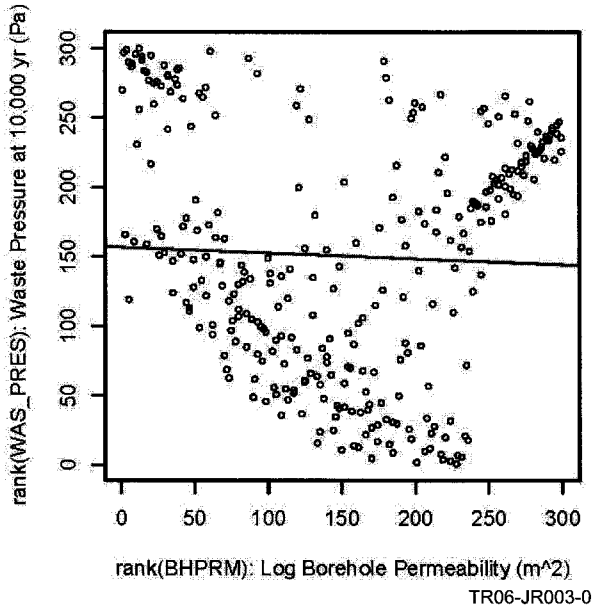


Fig. 3. Rank regression on a nonlinear and non-monotonic relationship generated in a sensitivity analysis of a two-phase fluid flow model.

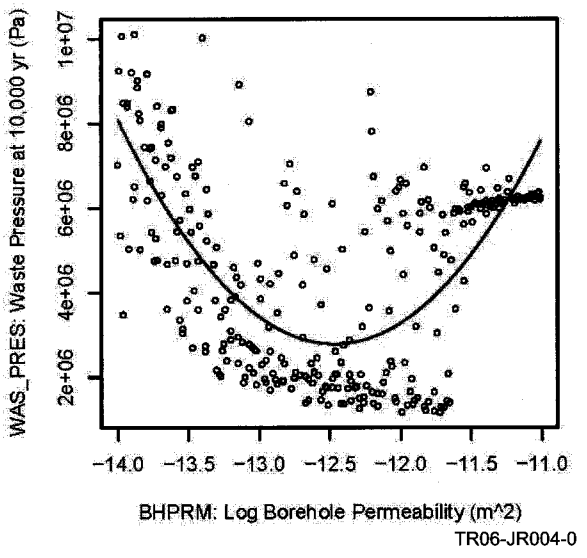


Fig. 4. Quadratic regression on a nonlinear and non-monotonic relationship generated in a sensitivity analysis of a two-phase fluid flow model.

where the relationships between y and at least some of the independent variables are nonlinear in the sense that y is not represented as a linear combination of these variables. Once the candidate form for the nonlinear regression model is decided on (e.g., the relationship in Eq. (2.9)), the β_j's can be estimated with techniques based on least squares, which is the maximum likelihood estimate when the ε's are normally distributed.

A major drawback to nonlinear regression is the requirement to decide on the form of the nonlinear regression model before the regression process can be initiated. This can be a particularly daunting challenge in a sensitivity analysis where several hundred different dependent variables (i.e., y's) may be under consideration with each dependent variable potentially requiring the formulation of a different nonlinear regression model. Further, model fitting, hypothesis testing, and interpreting of results is more difficult than is the case for linear regression. For the proceeding reasons, nonlinear regression models are not considered in this study. The nonparametric regression approaches introduced in the next section (Sect. 3) have advantages over nonlinear regression in that they can incorporate nonlinear relationships without the need to provide *a priori* specifications of model form.

3. Nonparametric Regression

Linear regression analysis has many desirable properties. When the underlying relationships are close to linear, no better technique is available. However, when nonlinear relationships are present, linear regression analysis can give misleading results and possibly no results at all. This potential failing provides the motivation for nonparametric regression.

Nonparametric regression, which is often called smoothing, is a form of surface approximation that is based on an assumed relationship of the form

$$y = f(\mathbf{x}) + \varepsilon, \mathbf{x} = [x_1, x_2, ..., x_{nX}], \quad (3.1)$$

where $E(\varepsilon) = 0$ and, as a result, $E(y|\mathbf{x}) = f(\mathbf{x})$. Usually, very few restrictions or assumptions are made about the properties of f. In particular, f is not assumed to take a particular parametric form such as a multivariate polynomial involving the elements of $\mathbf{x}$. Sometimes f is assumed to be "smooth" in the sense that certain continuity restrictions are imposed on f and possibly its derivatives.

To facilitate the introduction of the concept of smoothing, $\mathbf{x}$ is initially assumed to be univariate and smoothing is discussed in this context (Sect. 3.1); that is, the relation in Eq. (3.1) is assumed to be of the form $y = f(x) + \varepsilon$. Such univariate smoothing is often referred to as scatterplot smoothing. Next, the concept of degrees of freedom in association with smoothing is discussed (Sect. 3.2). Then, multivariate smoothing is described for relationships of the form in Eq. (3.1); that is, for the case where $\mathbf{x}$ is a vector rather than a scalar (Sect. 3.3). Finally, hypothesis testing for variable importance in nonparametric regression is discussed (Sect. 3.4).

3.1 Univariate Scatterplot Smoothers

The following provides a brief overview of scatterplot smoothing. More information is available in several references.[120-123] As previously indicated, scatterplot smoothers are used when there is one independent variable (i.e., x) and one dependent variable (i.e., y). Specifically, a data set of the form (x_i, y_i), $i = 1, 2, \ldots, nS$, is under consideration throughout this section. As suggested by the name, scatterplot smoothing involves fitting a curve to the data represented in a scatterplot. There are many ways to construct (i.e., fit) such a curve. The most familiar approach to such construction is simple linear regression (Sect. 2.1), although this approach is hardly nonparametric. In contrast to the parametric character of linear regression, the following nonparametric approaches to scatterplot smoothing are introduced: running means (Sect. 3.1.1), locally weighted means (Sect. 3.1.2), locally weighted regression (Sect. 3.1.3), and smoothing splines (Sect. 3.1.4).

3.1.1 Running Means

With running means, or possibly running medians, the predicted (i.e., estimated) value $\hat{y}(x_0)$ of $y(x_0)$ at a value x_0 of x is given by the mean (or median) of the y_i's associated with x_i's close to x_0. Typically, a fixed number r of values for x_i is selected for use. Then, $\hat{y}(x_0)$ is defined on the basis of the r values for x_i that are closest to x_0. For running means, this leads to the following approximation for an arbitrary value of x:

$$\hat{y}(x) = \hat{f}(x) = \frac{1}{r}\sum_{i=1}^{nS} I_{[0, d_r(x)]}(|x_i - x|)\, y_i, \quad (3.2)$$

where $d_r(x)$ denotes the distance along the x-axis to the r^{th} nearest neighbor of x (i.e., r values of x_i satisfy $|x_i - x| \le d_r(x)$) and

$$I_{[0, d_r(x)]}(|x_i - x|) = \begin{cases} 1 & \text{if } 0 \le |x_i - x| \le d_r(x) \\ 0 & \text{otherwise.} \end{cases} \quad (3.3)$$

A minor modification is required if multiple observations satisfy $|x_i - x| = d_r(x)$ (e.g., increase the value for r to incorporate these values or leave r fixed and average the corresponding y_i values). An analogous relationship holds for running medians except that medians over the y_i's associated with the r x_i's closest to x are calculated rather than means.

Although running means or medians are appealing because of their simplicity, they tend to produce a very wiggly function $\hat{f}(x)$. Specifically, as the values for x move along the x-axis, the sets of x_i's in use change, with these changes resulting in discontinuities in $\hat{f}(x)$. This behavior is illustrated in Fig. 5 for running means with the previously introduced two-phase flow data and $r = 20$.

3.1.2 Locally Weighted Means: Kernel Smoothers

Smoothing based on locally weighted means is employed to keep the intuitively appealing idea of a moving average while, concurrently, producing less small-scale erratic behavior in $\hat{f}(x)$. Specifically, locally weighted

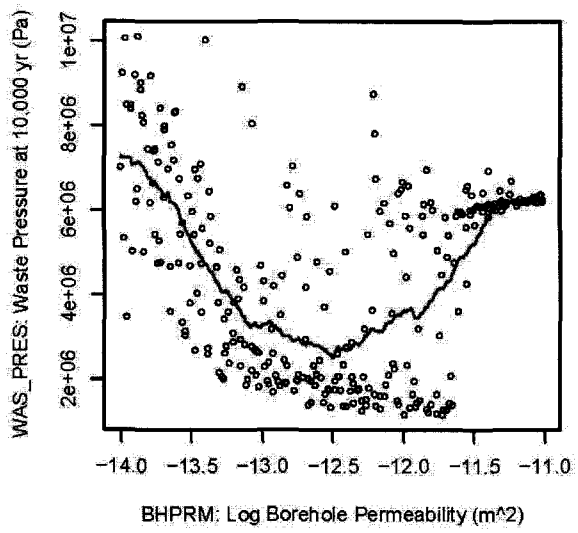


Fig. 5. Running means with $r = 20$ on results generated in a sensitivity analysis of a two-phase fluid flow model.

averaging with a kernel function $k(z; h)$ produces the approximation

$$\hat{f}(x) = \sum_{i=1}^{nS} k(x_i - x; h)\, y_i \Big/ \sum_{i=1}^{nS} k(x_i - x; h). \tag{3.4}$$

The role of $k(z; h)$ is to place more weight on the y_i's associated with x_i's close to x and less weight on y_i's associated with x_i's farther away from x. The kernel function $k(z; h)$ is usually chosen to have a maximum at $z = 0$ and to decrease monotonically to zero as $|z|$ increases. If $k(z; h)$ is a continuous function of z, then $\hat{f}(x)$ will be a continuous function of x. The bandwidth h, also known as the smoothing parameter, determines the amount of smoothing to be done to the data. Larger values of h result in more smoothing and smaller values of h result in greater fidelity to the data. A commonly used kernel function is

$$k(z; h) = \left(1/h\sqrt{2\pi}\right)\exp\left(-z^2/2h^2\right), \tag{3.5}$$

which corresponds to the normal density function with $\mu = 0$ and $\sigma = h$. Other viable choices for $k(z; h)$ also exist (e.g., see Sect. 2.6, Ref. 120). As discussed in more detail in Sect. 3.2, there is no universally accepted approach to determining the best value for h for a given kernel function and data set. However, it is widely accepted that the choice of the bandwidth h has more effect on the smoothing process than the choice of the kernel function (p. 19, Ref. 120).

The use of a kernel smoother with the kernel function in Eq. (3.5) and a bandwidth of $h = 0.6$ is illustrated in Fig. 6. Comparison of Figs. 5 and 6 illustrates the smoother form for $\hat{f}(x)$ produced by the use of locally weighted means than is the case for running means.

Kernel smoothers are "linear smoothers" in the sense that

$$\hat{\mathbf{y}} = [\hat{f}(x_1), \hat{f}(x_2), \ldots, \hat{f}(x_{nS})]^T \tag{3.6}$$

can be represented by

$$\hat{\mathbf{y}} = \mathbf{S}\mathbf{y}, \tag{3.7}$$

where $\mathbf{y} = [y_1, y_2, \ldots, y_{nS}]^T$ and the i^{th} row of the matrix $\mathbf{S}$ contains the kernel weights in the linear combination in Eq. (3.4). Specifically, the value in row i and column j of $\mathbf{S}$ is

$$s_{ij} = k\left(x_j - x_i; h\right) \Big/ \sum_{k=1}^{nS} k\left(x_k - x_i; h\right), \tag{3.8}$$

which is the weight on the j^{th} observation for prediction at x_i.

Edge effects are a potential drawback with locally weighted means. Such effects can be manifested near the largest and smallest observed values for x and result because of the unequal numbers of observations to the left and right of such values. Specifically, there are few observations to the left of small values for x_i and few observations to the right of large values for x_i. This imbalance in the number of observations can result in an overemphasis in the averaging process of observations on one side of such values and thus distort $\hat{f}(x)$ for values of x near the upper or lower ends of the range of values for the x_i. This effect can be seen for the smaller values of $x = BHPRM$ in Fig. 6, where the value for $\hat{f}(x)$ determined with locally weighted means appears to fall below the overall trend of the data.

3.1.3 Locally Weighted Regression

An approach similar to the kernel smoother (Sect. 3.1.2) that reduces the problem of edge effects involves the use of a locally weighted regression line.[124] With locally weighted regression,

$$\hat{f}(x) = \hat{\alpha}(x) + \hat{\beta}(x)x, \tag{3.9}$$

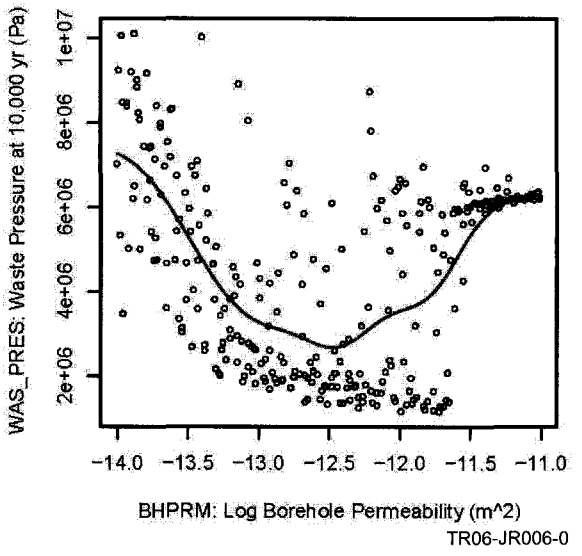


Fig. 6. Locally weighted means with kernel function $k(z; h)$ in Eq. (3.5) and bandwidth $h = 0.6$ on results generated in a sensitivity analysis of a two-phase fluid flow model.

where $\hat{\alpha}(x)$ and $\hat{\beta}(x)$ are estimated for individual values of x. In particular and for a specific value of x, the quantities $\hat{\alpha}(x)$ and $\hat{\beta}(x)$ are defined to be the values for α and β that minimize the sum

$$\sum_{i=1}^{nS}(\alpha+\beta x_i-y_i)^2 k(x-x_i;h), \tag{3.10}$$

where $k(x - x_i; h)$ is an appropriately defined kernel function. The indicated minimization of α and β is straightforward with an appropriate matrix formulation of the problem (p. 84, Ref. 123).

Locally weighted regression is actually equivalent to the determination of a locally weighted mean (Sect. 3.1.2) with a complicated kernel function that derives from the estimation of $\hat{\alpha}(x)$ and $\hat{\beta}(x)$. This kernel function will not be given here but can be found elsewhere (p. 241, Ref. 125). Thus, locally weighted regression is also a linear smoother as it can be put in the form $\hat{\mathbf{y}} = \mathbf{Sy}$ indicated in Eq. (3.7).

Both kernel smoothing of the mean and locally weighted regression have a problem when data are sparse in a particular region. In this situation, there are few points close to some x values to use in the averaging process and, depending on the kernel in use, $\hat{f}(x)$ may not even be defined for such x values. Cleveland recognized this problem and mitigated its effects by incorporating a nearest neighbors approach with the locally weighted regression line.[124] This procedure is often referred to by the designator LOESS, which is short for *local regression* and was chosen in allusion to the fact that LOESS is a deposit of fine clay or silt along a river valley and is thus a surface of sorts (p. 314, Ref. 126).

With LOESS, the kernel function is modified to take into account the distance $d_r(x)$ to the r^{th} nearest neighbor of a point x. Specifically, Cleveland[124] proposed that $\hat{\alpha}(x)$ and $\hat{\beta}(x)$ should be estimated by minimizing the expression in Eq. (3.10) with the kernel function $k(z; h)$ defined by

$$k(z;h)=\left[1-\left(|z|/h\right)^3\right]^3 I_{[0,h)}(|z|), \tag{3.11}$$

where $I_{[0,h)}(|z|)$ is defined analogously to the expression in Eq. (3.3) (i.e., $I_{[0,h)}(|z|) = 1$ if $0 \le |z| < h$ and 0 otherwise) and h corresponds to $d_r(x)$. With this formulation, $\hat{\alpha}(x)$ and $\hat{\beta}(x)$ are defined to be the values for α and β that minimize the expression

$$\sum_{i=1}^{nS}(\alpha+\beta x_i-y_i)^2\left\{1-\left[|x-x_i|/d_r(x)\right]^3\right\}^3 \times I_{[0,d_r(x))}(|x-x_i|). \tag{3.12}$$

The use of $h = d_r(x)$ in the definition of $k(z; h)$ allows the bandwidth to vary along the x-axis. This assures that $r - 1$ of the nS observations will have nonzero weights when computing the local regression line $\hat{f}(x)$ for each x regardless of how sparse the data is. If several points are tied for being the r^{th} nearest neighbor to x, then there will actually be less than $r - 1$ points with nonzero weight for this special case. An analysis employing LOESS is often described by its span, which is the ratio r/nS. Intuitively, the span is the ratio of the number of observations with nonzero weight used in the estimation of $\hat{\alpha}(x)$ and $\hat{\beta}(x)$ to the total number of observations although this is not quite correct as only $r - 1$ observations typically have nonzero weight.

The improvement in the estimate of $\hat{f}(x)$ with LOESS over the estimate obtained with locally weighted means can be seen by comparing the results in Figs. 6 and 7. In particular, the estimate for $\hat{f}(x)$ in Fig. 7 is obtained from LOESS with $r = 60$ and a corresponding span of 0.20. This estimate tracks the data near the ends of the range for $x = BHPRM$ more faithfully than is the case for $\hat{f}(x)$ in Fig. 6 obtained with locally weighted means. This is particularly evident for the smaller values of x. Due to its good performance, LOESS has become one of the most popular scatterplot smoothers.

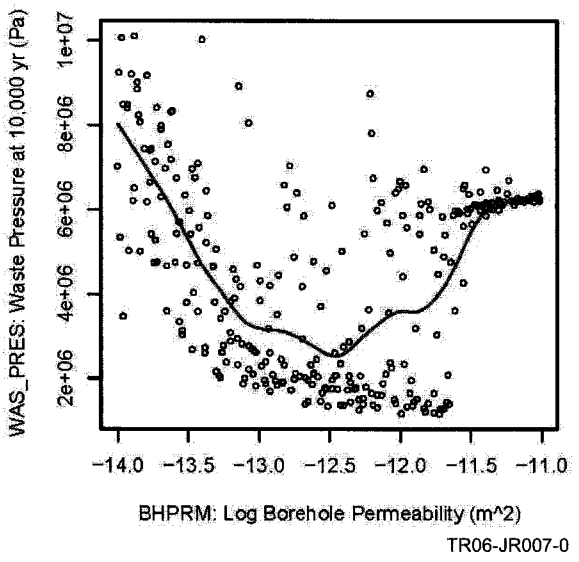


Fig. 7. Analysis with LOESS for kernel function $k(z; h)$ in Eq. (3.11) and $r = 60$ (i.e., a span of 0.20) on results generated in a sensitivity analysis of a two-phase fluid flow model.

3.1.4 Smoothing Splines

Another popular scatterplot smoother is the cubic smoothing spline. A cubic smoothing spline is a function $\hat{f}$ that minimizes the penalized residual sum of squares

$$\sum_{i=1}^{nS}\left[y_i - \hat{f}(x_i)\right]^2 + \lambda \int_a^b \left[\mathrm{d}^2\hat{f}(x)/\mathrm{d}x^2\right]^2 \mathrm{d}x \qquad (3.13)$$

over all continuously differentiable functions f, where $a \le x_{(1)} = \min\{x_i\colon 1 \le i \le nS\}$, $\max\{x_i\colon 1 \le i \le nS\} = x_{(nS)} \le b$, and λ is a constant (Sect. 2.10, Ref. 120). The first term in the preceding expression is the residual sum of squares and measures fidelity to the data; the second term constitutes a penalty for $\hat{f}$ having too much curvature.

There is a unique, explicit solution to the minimization problem associated with Eq. (3.13). This solution is a natural cubic polynomial spline with knots (i.e., locations of change in the structure of the spline) at the observed values for x (Sect. 2.10, Ref. 120). A cubic polynomial spline is a function that is a cubic polynomial on any interval defined by adjacent knots, has two continuous derivatives, and has a third derivative that is a step function with jumps at the knots. A natural cubic spline is a cubic spline that is restricted to be linear on $(-\infty, x_{(1)})$ and $(x_{(nS)}, \infty)$.

The quantity λ in Eq. (3.13) plays the role of a smoothing parameter. As with the smoothing parameters associated with the previously introduced methods, the appropriate value to use for λ is not intuitively apparent. Typically, the equivalent degrees of freedom (*df*) described in the next section (Sect. 3.2) is used to determine the value for λ for smoothing splines. Fig. 8 shows a cubic smoothing spline involving *BHPRM* with $df = 8$. As comparison of Figs. 7 and 8 shows, the behavior of this cubic smoothing spline is similar to that of LOESS.

3.2 Equivalent Degrees of Freedom and Smoothing Parameters

Automated methods of selecting smoothing parameters for the techniques presented in Section 3.1 are now discussed. To do this, the related topic of degrees of freedom is introduced. In linear model theory, the degrees of freedom *df* of a model is defined to be the number of linearly independent columns in the design matrix $\mathbf{X}$ defined in Eq. (2.6). This is the same as the number of parameters included in the associated linear model. An equivalent definition is

$$df = tr(\mathbf{H}), \qquad (3.14)$$

where $\mathbf{H} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$, often called the hat matrix, is the perpendicular projection matrix that projects a vector onto the $\mathbf{X}$ space (i.e., the space spanned by the vectors corresponding to the columns in $\mathbf{X}$) and $tr(\mathbf{H})$ denotes the trace of $\mathbf{H}$ (i.e., the sum of the diagonal elements of $\mathbf{H}$). Predicted values for the linear model $\mathbf{y} = \mathbf{X}\beta$ are obtained from the relationship $\hat{\mathbf{y}} = \mathbf{H}\mathbf{y}$. More on the projection matrix $\mathbf{H}$ can be found elsewhere (p. 68, Ref. 127; p. 393, Ref. 128).

The nonparametric techniques discussed in Sects. 3.1.1 – 3.1.3 can be put in the form $\hat{\mathbf{y}} = \mathbf{S}\mathbf{y}$ indicated in Eq. (3.7). Such techniques are said to be linear. For convenience, $\mathbf{S}$ is referred to as the smoother matrix. The symbol $\mathbf{S}$ is used to denote the smoother matrix to distinguish it from $\mathbf{H}$ since $\mathbf{S}$ is not, in general, a perpendicular projection onto the $\mathbf{X}$ space. A natural generalization of the concept of degrees of freedom is to define the degrees of freedom associated with a smoother matrix $\mathbf{S}$ to be

$$df = tr(\mathbf{S}), \qquad (3.15)$$

where $tr(\mathbf{S})$ denotes the trace of $\mathbf{S}$. In turn, the degrees of freedom for error *dferr* can then be defined by

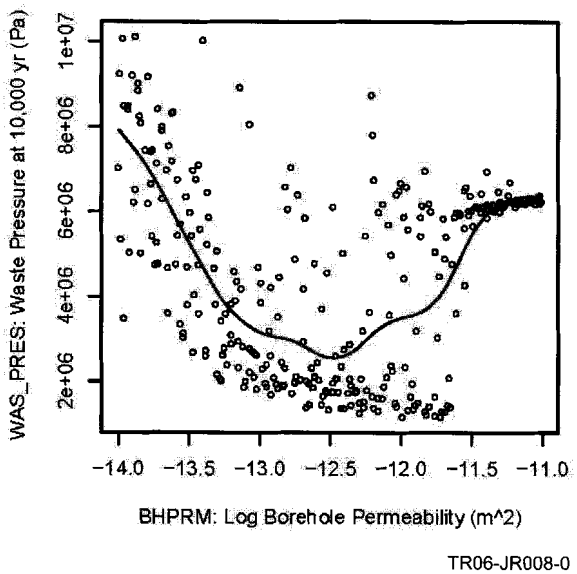


Fig. 8. Analysis with smoothing spline with $a = x_{(1)}$, $b = x_{(nS)}$ and $df = 8$ (see Eq. (3.13)) on results generated in a sensitivity analysis of a two-phase fluid flow model.

$$dferr = nS - tr(\mathbf{S}) \tag{3.16}$$

in analogy to the corresponding definition

$$dferr = nS - tr(\mathbf{H}) \tag{3.17}$$

for linear models.

The preceding definitions make some intuitive sense if the two extreme prediction cases, simple averaging and interpolation, are considered. For a simple average, the diagonal elements of **S** are given by $s_{ii} = 1/nS$. As a result, $tr(\mathbf{S}) = 1$ or, equivalently, one degree of freedom (i.e., $df = 1$) is being used to estimate the overall mean value. In the interpolation case, $s_{ii} = 1$ and the other weights in a row must be 0 so that the predicted value is given by $\hat{f}(x_i) = y_i$. In this case, each observation has its own value and $tr(\mathbf{S}) = nS$, which implies a model with nS degrees of freedom (i.e., $df = nS$). Most models fall somewhere in between these two extremes. Additional discussion of degrees of freedom in the context of nonparametric regression is available elsewhere (pp. 52 – 55, Ref. 120).

Degrees of freedom will be used for inference later in this presentation. However, degrees of freedom can also be used to obtain some insight with respect to appropriate values to use for smoothing parameters. For a particular kernel, a desired value of *df* for the smoother matrix can be specified, and then the value of the smoothing parameter that produces this value can be determined. This still leaves open the question of what is an appropriate value for *df*. The approaches below offer a better guide to smoothing parameter selection.

A widely used automatic selection procedure for smoothing parameters is the cross validation (CV) approach. With this approach, the jackknifed (or leave one out) residuals are obtained by fitting the model without the i^{th} observation and then predicting y_i. The deleted residual is then

$$r_{(i)} = y_i - \hat{y}_{(i)}, \tag{3.18}$$

where $\hat{y}_{(i)}$ is the jackknifed (i.e., predicted) value for y_i obtained with y_i omitted from the prediction process, and the predicted residual sum of squares (PRESS)[129] is given by

$$PRS = \sum_{i=1}^{nS} r_{(i)}^2 = \sum_{i=1}^{nS} \left[y_i - \hat{y}_{(i)} \right]^2. \tag{3.19}$$

The PRESS value *PRS* is then used in the selection of the smoothing parameter. In particular, different values for the smoothing parameter result in different values for *PRS*. The preferred smoothing parameter value is the value that minimizes *PRS*.

For representations of the form $\hat{\mathbf{y}} = \mathbf{Sy}$, it is not necessary to fit the model multiple times to obtain the deleted residuals. Instead, all the deleted residuals can be obtained from the usual residuals

$$r_i = y_i - \hat{y}_i \tag{3.20}$$

and the leverage values s_{ii}, which are the diagonal elements of **S**. In particular, the deleted residuals are given by

$$r_{(i)} = r_i / (1 - s_{ii}) = (y_i - \hat{y}_i) / (1 - s_{ii}) \tag{3.21}$$

(see p. 47, Ref. 120). This makes cross validation easy to apply for linear smoothing provided **S** is relatively easy to calculate.

In practice, the preceding cross validation criterion tends to result in the selection of smoothing parameters that undersmooth. To correct for this, a generalized cross validation criterion has been suggested (p. 49, Ref. 120). This generalized criterion employs an adjusted PRESS value given by

$$PRS_A = \sum_{i=1}^{nS}\left[\frac{r_i}{1-tr(\mathbf{S})/nS}\right]^2 = \sum_{i=1}^{nS}\left[\frac{y_i-\hat{y}_i}{1-tr(\mathbf{S})/nS}\right]^2 \quad (3.22)$$

in the determination of the smoothing parameter. Given that

$$tr(\mathbf{S}) = \sum_{i=1}^{nS} s_{ii}, \quad (3.23)$$

each deleted residual $r_{(i)}$ is in essence being calculated with an average leverage value given by

$$\bar{s} = tr(\mathbf{S})/nS = \sum_{i=1}^{nS} s_{ii}/nS\,. \quad (3.24)$$

This approach puts less emphasis on observations with high leverage values. Another way to write PRS_A is

$$PRS_A = \left[\frac{1}{1-df/nS}\right]^2 \sum_{i=1}^{nS} r_i^2, \quad (3.25)$$

which shows that PRS_A can be viewed as the error sum of squares penalized by the degrees of freedom associated with the model used in smoothing the data. With this criterion, the preferred smoothing parameter value is the value that minimizes PRS_A.

3.3 Multivariate Smoothers

More general relationships of the form $y = f(\mathbf{x})$ indicated in Eq. (1.7) are now considered. Further, a mapping $y_i = f(\mathbf{x}_i)$, $i = 1, 2, \ldots, nS$, from analysis inputs to analysis results as shown in Eq. (1.8) is assumed to be available for analysis. In this framework, approximations $\hat{f}(\mathbf{x})$ to a relationship of the form

$$E(y|\mathbf{x}) = f(\mathbf{x}) = f(x_1, x_2, \ldots, x_{nX}) \quad (3.26)$$

are sought. The kernel methods described for the univariate case in Sect. 3.1 have immediate and straightforward generalizations to this multivariate context. These generalizations are often referred to as multiple predictor techniques. In particular, the following multiple predictor techniques are considered in this section: locally weighted regression (Sect. 3.3.1), additive models (Sect. 3.3.2), projection pursuit regression (Sect. 3.3.3), and recursive partitioning regression (Sect. 3.3.4).

3.3.1 Locally Weighted Regression: LOESS

The LOESS technique in multiple dimensions is analogous to the same technique in one dimension (Sect. 3.1.3). In particular, the relationship between y and $\mathbf{x}$ is assumed to be of the form

$$y = f(\mathbf{x}) = \alpha(\mathbf{x}) + \boldsymbol{\beta}(\mathbf{x})\mathbf{x} + \varepsilon, \quad (3.27)$$

where $\boldsymbol{\beta}(\mathbf{x}) = [\beta_1(\mathbf{x}), \beta_2(\mathbf{x}), \ldots, \beta_{nX}(\mathbf{x})]$, $\mathbf{x} = [x_1, x_2, \ldots, x_{nX}]^T$, and $E(\varepsilon) = 0$. In turn, an approximate relationship of the form

$$\hat{y} = \hat{f}(\mathbf{x}) = \hat{\alpha}(\mathbf{x}) + \hat{\boldsymbol{\beta}}(\mathbf{x})\mathbf{x} \quad (3.28)$$

is sought with LOESS, with the corresponding one dimensional special case appearing in Eq. (3.9).

The quantities $\hat{\alpha}(\mathbf{x})$ and $\hat{\boldsymbol{\beta}}(\mathbf{x})$ for a given value of $\mathbf{x}$ are defined to be the values for α and $\boldsymbol{\beta} = [\beta_1, \beta_2, \ldots, \beta_{nX}]$ that minimize the sum

$$\sum_{i=1}^{nS}(\alpha + \boldsymbol{\beta}\mathbf{x}_i - y_i)^2\left[1-\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|}{d_r(\mathbf{x})}\right)^3\right]^3 I_{[0,d_r(\mathbf{x}))}(\|\mathbf{x}-\mathbf{x}_i\|), \quad (3.29)$$

where (i) $d_r(\mathbf{x})$ is the distance to the r^{th} nearest neighbor of $\mathbf{x}$ in nX-dimensional Eulidean space, (ii) $I_{[0,d_r(\mathbf{x}))}(\|\mathbf{x}-\mathbf{x}_i\|)$ is defined analogously to $I_{[0,d_r(x))}(|x-x_i|)$ in Eq. (3.12), and (iii) the individual independent variables (i.e., $x_1, x_2, \ldots, x_{nX}$) are normalized to mean zero and standard deviation one so that the value for the norm $\|\cdot\|$ is not dominated by the units used for these variables. The determination of α and $\boldsymbol{\beta}$ is straightforward with the use of appropriate matrix techniques (p. 139, Ref. 121). Except for use of the norm $\|\cdot\|$ instead of absolute value $|\cdot|$, the expression in Eq. (3.29) with LOESS for multidimensional $\mathbf{x}$ is the same as the expression in Eq. (3.12) for the one-dimensional case.

The determination of $\hat{\alpha}$ and $\hat{\boldsymbol{\beta}}(\mathbf{x})$ provides an estimate of $\hat{y}$ for one value of $\mathbf{x}$ as indicated in Eq. (3.28). Estimates of y for additional values of $\mathbf{x}$ require the solution of an additional minimization problem for each $\mathbf{x}$. This may seem computationally demanding but LOESS is actually quite fast computationally even with multiple independent variables.

The obvious benefit to using LOESS in multiple dimensions is that it can capture nonlinear behavior that

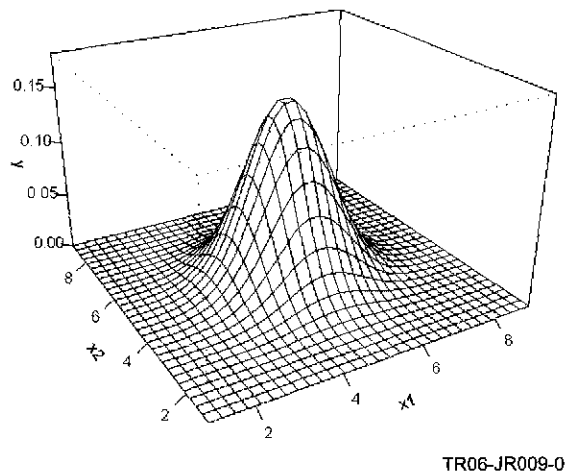
a typical parametric model cannot. A more subtle advantage is the capability to capture very general interactions between input variables. The indicated capabilities derive from the property that LOESS is inherently local in its approximations to the relationship $y = f(\mathbf{x})$. For example, a LOESS surface fitted to two variables is shown in Fig. 9. The actual functional relationship is

$$y = f(x_1, x_2) = (1/2\pi)\exp\left\{-\left[(x_1-5)^2+(x_2-5)^2\right]/2\right\}, \tag{3.30}$$

which corresponds to the density function for a bivariate normal distribution. The surface in Fig. 9 was constructed with LOESS and a random sample of size $nS = 100$ from $\mathbf{x} = [x_1, x_2]$ with x_1 and x_2 uniform on [0, 10]. In this example, LOESS captures the nonlinear interaction between x_1 and x_2 in the determination of y.

The LOESS technique in multiple dimensions is also a linear smoother in the sense that it can be expressed in the form shown in Eq. (3.7). The actual form of the kernel function is a generalization of the univariate case given in Schimek (p.241, Ref. 125).

A drawback with LOESS and other local averaging techniques in higher dimensions is that the closest observed values $\mathbf{x}_i$ to the value $\mathbf{x}$ under consideration are not necessarily local (i.e., nearby) along the axes for the individual variables x_j, $j = 1, 2, \ldots, nX$, contained in $\mathbf{x}$. This is sometimes referred to as the curse of dimensionality. To illustrate this, first consider one independent variable. To include 30% of the data in a local average, it is necessary to span approximately 30% of the corresponding axis if the variable values are approximately uniformly distributed. With the same distributional assumption and two independent variables, including 30% of the data now requires spanning 55% of the range of each of the variables. This requirement results because the joint range of the two variables is now a rectangle and covering 30% of this rectangle requires covering 55% of the range of each of the two variables (i.e., $(0.55)^2 \cong 0.30$). As the number of independent variables increases, the problem becomes worse. With five independent variables, use of 30% of the data requires spanning 79% of the range of each of the individual variables. This hardly constitutes a local average anymore. The span (i.e., percent coverage) can be made smaller but then there is a danger of undersmoothing unless the number of observations is substantially increased.



TR06-JR009-0

Fig. 9. Example of LOESS surface constructed for $y = f(x_1, x_2) = (1/2\pi)\exp\{-[(x_1 - 5)^2 + (x_2 - 5)^2]/2\}$; see Eq. (3.27).

The LOESS procedure will work in higher dimensions and actually works quite well for $nX \leq 3$. For $nX > 3$, however, LOESS starts to be affected by the curse of dimensionality. As will be illustrated later, this can cause LOESS to miss the effects of important variables in the estimation of f (Sect. 5).

Several procedures have been developed in an attempt to overcome the dimensionality problem. These procedures implement one or more of the following strategies as discussed in subsequent sections: additive modeling (Sect. 3.3.2), dimension reduction (Sect. 3.3.3), and recursive partitioning (Sect. 3.3.4).

3.3.2 Additive Models

For additive modeling, the function $f(\mathbf{x})$ in Eq. (3.27) is assumed to have the form

$$f(\mathbf{x}) = \sum_{j=1}^{nX} f_j(x_j), \tag{3.31}$$

where the f_j are arbitrary functions that will be determined as part of the analysis process. This is analogous to multiple linear regression where the effects of the independent variables are additive. The difference is that $y = f(\mathbf{x})$ is not assumed to be a linear function of the x_j. This representation is not completely general as it does not allow for interactions between the independent variables. However, nothing prevents the inclusion of multiplicative interactions $x_r x_s$ as in linear regression.

Additive models are usually constructed with a method known as backfitting suggested by Friedman and Stutzel.[130] The algorithm that is used in the software packages R and S-Plus to implement this method is described in Chambers and Hastie (p. 300, Ref. 126). The indicated algorithm is more efficient than the approach that is described below. However, the described approach provides a more intuitive introduction to the ideas involved in additive model construction.

The observed values for y are assumed to be of the form

$$y_i = f(\mathbf{x}_i) + \varepsilon_i = \sum_{j=1}^{nX} f_j(x_{ij}) + \varepsilon_i. \tag{3.32}$$

Given initial estimates $\hat{f}_2$, $\hat{f}_3$, ..., $\hat{f}_{nX}$ for f_2, f_3, ..., f_{nX}, (e.g., $\hat{f}_j(x_j) = b_j x_j$ for $j = 2, 3, \ldots, nX$, where the b_j are coefficients from a regression model of the form indicated in Eq. (2.2)), an estimate $\hat{f}_1$ for f_1 can be obtained through use of the relationship

$$y_i - \sum_{j=2}^{nX} \hat{f}_j(x_{ij}) \cong f_1(x_{i1}) + \varepsilon_i \tag{3.33}$$

for $i = 1, 2, \ldots, nS$. In particular, one of the scatterplot smoothers introduced in Sect. 3.1 can be used to smooth the partial residuals on the left hand side of Eq. (3.33) across x_1. This produces an estimate $\hat{f}_1$ for f_1 defined across the range of values for x_1. Given this estimate for f_1, the estimate $\hat{f}_2$ for f_2 can be refined in the same manner across the range of values for x_2 with $\hat{f}_1$, $\hat{f}_3$, $\hat{f}_4$, ..., $\hat{f}_{nX}$. This procedure then continues and repetitively cycles through the variables. The cycling continues until convergence is achieved.

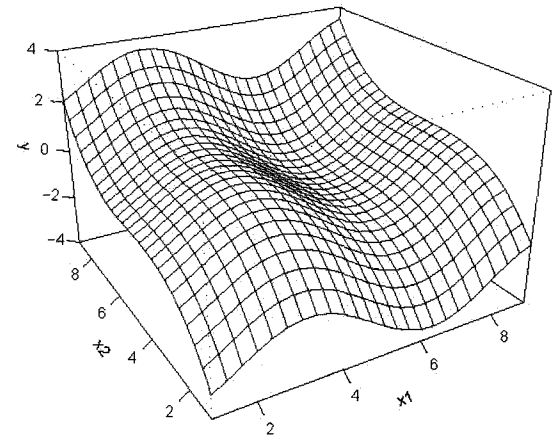
The result is $\hat{f}_j$ defined over the range of x_j for $j = 1, 2, \ldots, nX$. In turn, $y = f(\mathbf{x})$ can be estimated for arbitrary values of $\mathbf{x} = [x_1, x_2, \ldots, x_{nX}]$ by

$$y \cong \sum_{j=1}^{nX} \hat{f}_j(x_j) \tag{3.34}$$

Additional detail is available elsewhere (pp. 90 – 91, Ref. 120; pp. 300 – 302, Ref. 126).

Additive models can be used to develop representations for complex nonlinear behavior as indicated in Fig. 10 by the approximation to

$$y = f(x_1, x_2) = \sin(x_1) + (x_2 - 5)^3 \tag{3.35}$$



TR06-JR010-0

Fig. 10. Example of additive model surface constructed for $y = f(x_1, x_2) = \sin(x_1) + (x_2 - 5)^3$; see Eq. (3.35).

obtained from a random sample of size $nS = 100$ from $\mathbf{x} = [x_1, x_2]$ with x_1 and x_2 uniform on [0, 10]. Additive models also work well in higher dimensions with a large number of independent variables as will be illustrated in the model is dependent on the actual relationship Sect. 5. However, successful construction of an additive model is dependent on the actual relationship between y and $\mathbf{x}$ involving limited interactions between the elements of $\mathbf{x}$.

The procedure indicated in this section to construct an approximation to the function $f(\mathbf{x})$ in Eq. (3.31) is a linear smoother provided a linear scatterplot smoother is used in the backfitting algorithm in the sense that this procedure can be formally represented in the form shown in Eq. (3.7). The smoother matrix $\mathbf{S}$ in Eq. (3.7) is difficult to compute in a closed form as the overall analysis involves an iterative process. An approximation to $tr(\mathbf{S})$, which corresponds to the number of degrees of freedom associated with the procedure, is given by

$$tr(\mathbf{S}) \cong \sum_{j=1}^{nX} df_j, \tag{3.36}$$

where df_j is the degrees of freedom used in the scatterplot smoother for x_j in the backfitting algorithm (p. 129, Ref. 120).

3.3.3 Projection Pursuit Regression

Projection pursuit regression involves both dimension reduction and additive modeling and is based on

the assumption that the function $f(\mathbf{x})$ in Eq. (3.27) has the form

$$f(\mathbf{x}) = \sum_{s=1}^{nD} g_s(\boldsymbol{\alpha}_s \mathbf{x}), \tag{3.37}$$

where $\boldsymbol{\alpha}_s = [\alpha_{1s}, \alpha_{2s}, \ldots, \alpha_{nX,s}]$, $\boldsymbol{\alpha}_s$ and $\boldsymbol{\alpha}_t$ are orthogonal for $s \neq t$, $\mathbf{x} = [x_1, x_2, \ldots, x_{nX}]^T$, $\boldsymbol{\alpha}_s \mathbf{x}$ corresponds to a linear combination of the elements of $\mathbf{x}$, and g_s is an arbitrary function. Values for g_s, $\boldsymbol{\alpha}_s$ and nD are determined as part of the analysis procedure. The expression in Eq. (3.37) is an additive model with the quantities $\boldsymbol{\alpha}_s \mathbf{x}$ replacing the elements x_j of $\mathbf{x}$ as the independent variables. Further, this expression involves a reduction in dimension as nD is usually smaller than nX.

The representation for f in Eq. (3.37) allows for interactions between variables, which is not the case for the additive representation in Eq. (3.31). To see this, consider the example in which $\mathbf{x} = [x_1, x_2]^T$, $\boldsymbol{\alpha}_1 = [1, 1]$ and $g_1(u) = u^2$. The result is

$$g_1(\boldsymbol{\alpha}_1 \mathbf{x}) = (x_1 + x_2)^2 = x_1^2 + 2x_1 x_2 + x_2^2, \tag{3.38}$$

which involves the interaction term $x_1 x_2$.

The entities $\hat{\boldsymbol{\alpha}}_1, \hat{\boldsymbol{\alpha}}_2, \ldots, \hat{\boldsymbol{\alpha}}_{nD}$ and $\hat{g}_1, \hat{g}_2, \ldots, \hat{g}_{nD}$ are estimated as part of the construction process. This is accomplished by first estimating $\boldsymbol{\alpha}_1$ and g_1. Specifically, $\hat{\boldsymbol{\alpha}}_1$ and $\hat{g}_1$ are defined to be the values for $\boldsymbol{\alpha}$ and $g_{\boldsymbol{\alpha}}$ that minimize the sum

$$\sum_{i=1}^{nS} \left[y_i - g_{\boldsymbol{\alpha}}(\boldsymbol{\alpha} \mathbf{x}_i) \right]^2, \tag{3.39}$$

where $\boldsymbol{\alpha} \in R^{nX}$, $\|\boldsymbol{\alpha}\| = 1$, and $g_{\boldsymbol{\alpha}}$ is the outcome of using a scatterplot smoother (e.g., LOESS; see Sect. 3.1.3) on the points $[y_i, \boldsymbol{\alpha} \mathbf{x}_i]$, $i = 1, 2, \ldots, nS$. Once $\hat{\boldsymbol{\alpha}}_1$ and $\hat{g}_1$ are estimated, the partial residuals $y_i - \hat{g}_1(\hat{\boldsymbol{\alpha}}_1 \mathbf{x}_i)$, $i = 1, 2, \ldots, nS$, are used to obtain $\hat{\boldsymbol{\alpha}}_2$ and $\hat{g}_2$. Specifically, $\hat{\boldsymbol{\alpha}}_2$ and $\hat{g}_2$ are defined to be the values for $\boldsymbol{\alpha}$ and $g_{\boldsymbol{\alpha}}$ that minimize the sum

$$\sum_{i=1}^{nS} \left\{ \left[y_i - \hat{g}_1(\hat{\boldsymbol{\alpha}}_1 \mathbf{x}_1) \right] - g_{\boldsymbol{\alpha}}(\boldsymbol{\alpha} \mathbf{x}_i) \right\}^2, \tag{3.40}$$

where $\boldsymbol{\alpha} \in R^{nX}$, $\|\boldsymbol{\alpha}\| = 1$, $\boldsymbol{\alpha}$ and $\hat{\boldsymbol{\alpha}}_1$ are orthogonal, and $g_{\boldsymbol{\alpha}}$ is the outcome of using a scatterplot smoother on the points $[y_i - \hat{g}_1(\hat{\boldsymbol{\alpha}}_1 \mathbf{x}), \boldsymbol{\alpha} \mathbf{x}_i]$, $i = 1, 2, \ldots, nS$. This process continues until no appreciable improvement based on a relative error criterion is observed. Unlike additive models, backfitting is typically not used in projection pursuit regression.

The scatterplot smoother typically used at each step in projection pursuit regression is a variable span version of LOESS, called the supersmoother (*supsmu*) in R and *S*-Plus (p. 318, Ref. 108). This presentation will actually use smoothing splines instead. Further discussion on smoothing in the context of projection pursuit regression, smoothing parameter selection and determination of the number of projection terms nD is given in Sect. 4. Additional information on projection pursuit regression is available elsewhere.[130]

As indicated in Eq. (3.37), the outcome of a projection pursuit regression consists of the vectors $\boldsymbol{\alpha}_s$ defined for $s = 1, 2, \ldots, nD$ and corresponding functions g_s defined for $\boldsymbol{\alpha}_s \mathbf{x}$. Predictions of $y = f(\mathbf{x})$ are then given by

$$y \cong \sum_{s=1}^{nD} g_s(\boldsymbol{\alpha}_s \mathbf{x}), \tag{3.41}$$

for $\mathbf{x} = [x_1, x_2, \ldots, x_{nX}]$.

Projection pursuit regression can represent very general situations involving nonlinearity and variable interactions. Further, it avoids the dimensionality problem by using projection terms and additive modeling. However, this generality can come at a price. Results in Sect. 5 suggest that projection pursuit regression has a tendency to overfit the data by including spurious variables in the model.

3.3.4 Recursive Partitioning Regression

Recursive partitioning regression is most commonly known in the form of regression trees.[131] A regression tree splits the data into subgroups where the observations within each subgroup are more homogeneous than they are over the set of all observations. Then, $f(\mathbf{x})$ in Eq. (3.27) is estimated by the sample mean over each subgroup. The resultant estimate for f is a piecewise constant function, which is also known as a simple function. More precisely, the estimate $\hat{f}(\mathbf{x})$ is given by

$$\hat{f}(\mathbf{x}) = \sum_{s=1}^{nP} c_s I_s(\mathbf{x}), \tag{3.42}$$

where (i) $\mathcal{A}_s$, $s = 1, 2, \ldots, nP$, are the disjoint sets into which the observed values $\mathbf{x}_i$, $i = 1, 2, \ldots, nS$, are partitioned (usually on the basis of the values for y_i), (ii) the mean c_s over each set $\mathcal{A}_s$ is defined by

$$c_s = \sum_{\mathbf{x}_i \in \mathcal{A}_s} y_i / C(\mathcal{A}_s) \tag{3.43}$$

with $C(\mathcal{A}_s)$ denoting the cardinality of $\mathcal{A}_s$, and (iii) $I_s(\mathbf{x})$ is the indicator function such that $I_s(\mathbf{x}) = 1$ if $\mathbf{x} \in \mathcal{A}_s$ and 0 otherwise. The use of regression trees in sensitivity analysis is illustrated in Mishra et al.[106]

Regression trees can be generalized by replacing the mean c_s in Eq. (3.42) with a linear function. In particular, $\hat{f}(\mathbf{x})$ can be defined by

$$\hat{f}(\mathbf{x}) = \sum_{s=1}^{nP} \left(\hat{\alpha}_s + \hat{\beta}_s \mathbf{x}\right) I_s(\mathbf{x}), \tag{3.44}$$

where $\hat{\alpha}_s + \hat{\beta}_s \mathbf{x}$ is the least squares linear fit to the data associated with $\mathcal{A}_s$ and I_s is defined the same as in Eq. (3.42). An example of $\hat{f}(\mathbf{x})$ for a single independent variable is given in Fig. 11. The individual regressions can also be constrained so that the regression lines (in one dimension) and regression surfaces (in two or more dimensions) meet continuously. Examples for one and two dimensions are given in Figs. 12 and 13.

The individual regression lines in Figs. 11b and 12b are constructed with a robust regression procedure in which the sum of squares is minimized over the middle two quartiles of the deviations from the regression line (see Ref. 132 for additional information on robust regression). In contrast, the individual regression lines in Figs. 11a and 12a are constructed with the traditional least squares procedure in which the sum of squares is minimized over all deviations from the regression line. The robust regression procedure reduces the effects of large deviations from the overall trend in the data. The effect of this reduction in the examples presented in Figs. 11 and 12 is to produce regression lines that more closely match a visual impression of the trends in the data. The visually appealing nature of the results in Figs. 11b and 12b suggests that robust regression procedures could have a useful role to play in sensitivity analysis due to their effectiveness in reducing the influence of outliers. Although all of the least squares procedures in this presentation are carried out in the traditional manner, the use of robust regression procedures in sensitivity analysis is an area that merits additional investigation.

The linear fit associated with $\hat{f}(\mathbf{x})$ in Eq. (3.44) reduces the need to split the data as many times as is typically the case when a regression tree is used. This approach will certainly outperform a regression tree when the relationship between y and the $\mathbf{x}_i$'s is close to linear for each partition set $\mathcal{A}_s$. The interpretation of the representation for $\hat{f}(\mathbf{x})$ in Eq. (3.44) with the linear fit is perhaps less obvious than the interpretation for $\hat{f}(\mathbf{x})$ in Eq. (3.42) with means. However, the primary concern in this presentation is constructing close approximations to the function $f(\mathbf{x})$ that defines y. Which independent variables are important in this approximation can be easily determined by observing the fidelity of $\hat{f}(\mathbf{x}_i)$ to the corresponding values y_i when $\hat{f}(\mathbf{x})$ is constructed with and without the inclusion of individual independent variables. In the examples of Section 5.1, the recursive partitioning approach given here outperformed the regression tree approach indicated in Eqs. (3.42) and (3.43), particularly in terms of estimation of η^2 defined in Eq. (5.13).

The determination of the partition sets $\mathcal{A}_s$, $s = 1, 2, \ldots, nP$, and the associated function $\hat{f}(\mathbf{x})$ is now considered. Let $x_{(r)j}$, $r = 1, 2, \ldots, nS$, represent the sampled values for x_j ordered by size (i.e., $x_{(r)j} \le x_{(r+1)j}$ for $r = 1, 2, \ldots, nS - 1$), and let $\mathcal{A}_{rj1}$ and $\mathcal{A}_{rj2}$ denote the sets defined by

$$\mathcal{A}_{rj1} = \left\{\mathbf{x}_i : x_{ij} \le x_{(r)j}\right\} \tag{3.45}$$

and

$$\mathcal{A}_{rj2} = \left\{\mathbf{x}_i : x_{ij} > x_{(r)j}\right\} \tag{3.46}$$

for $r = 1, 2, \ldots, nS$ and $j = 1, 2, \ldots, nX$. A separate linear regression is performed for the set of $(y_i, \mathbf{x}_i)$ pairs associated with each of the sets $\mathcal{A}_{rj1}$ and $\mathcal{A}_{rj2}$. Some of the sets will have too few data pairs (i.e., less than $nX + 1$) to fit a linear regression model and are excluded from consideration. This results in a total of $nX(nS - 2nX - 1)$ pairs $[\mathcal{A}_{rj1}, \mathcal{A}_{rj2}]$ that are candidates to define initial values for $\mathcal{A}_1$ and $\mathcal{A}_2$.

The pair $[\mathcal{A}_{rj1}, \mathcal{A}_{rj2}]$ with regressions that together provide the best representation for y are selected as the initial values for $\mathcal{A}_1$ and $\mathcal{A}_2$. This determination is made on the basis of the R^2 value given by

$$R_{rj}^2 = \frac{SST - SSE_{rj}}{SST} \tag{3.47}$$

for each pair $[\mathcal{A}_{rj1}, \mathcal{A}_{rj2}]$, where

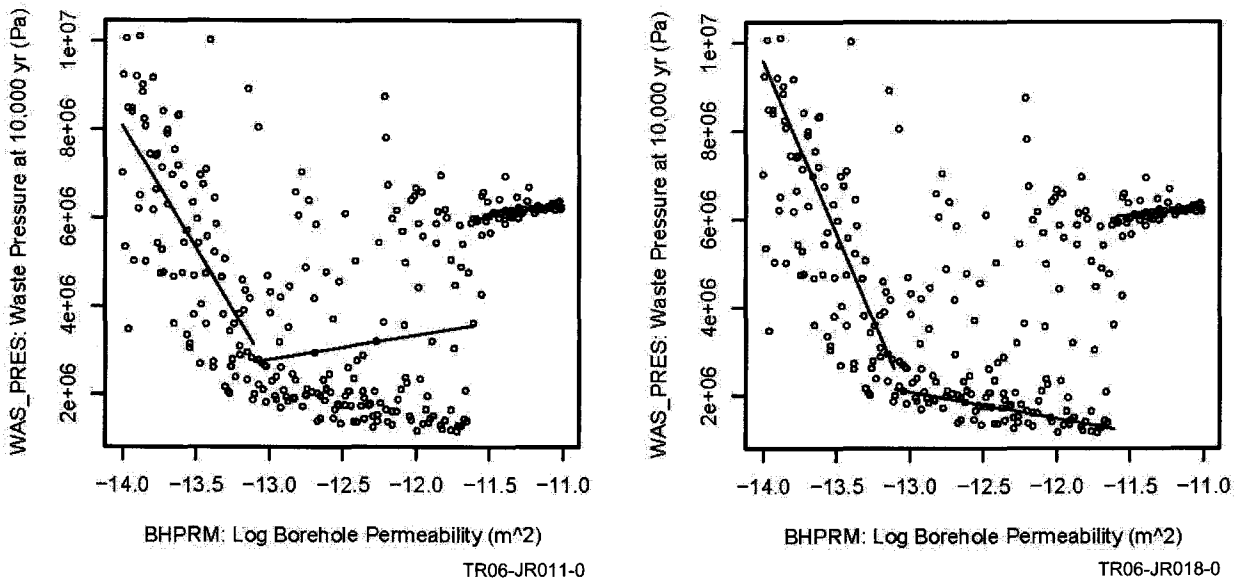


Fig. 11. Recursive partitioning regression on results generated in a sensitivity analysis of a two-phase fluid flow model: (a) Individual regression lines generated with traditional least squares regression, and (b) Individual regression lines generated with robust regression in which the sum of squares is minimized over the middle two quartiles of the deviations from the regression line.

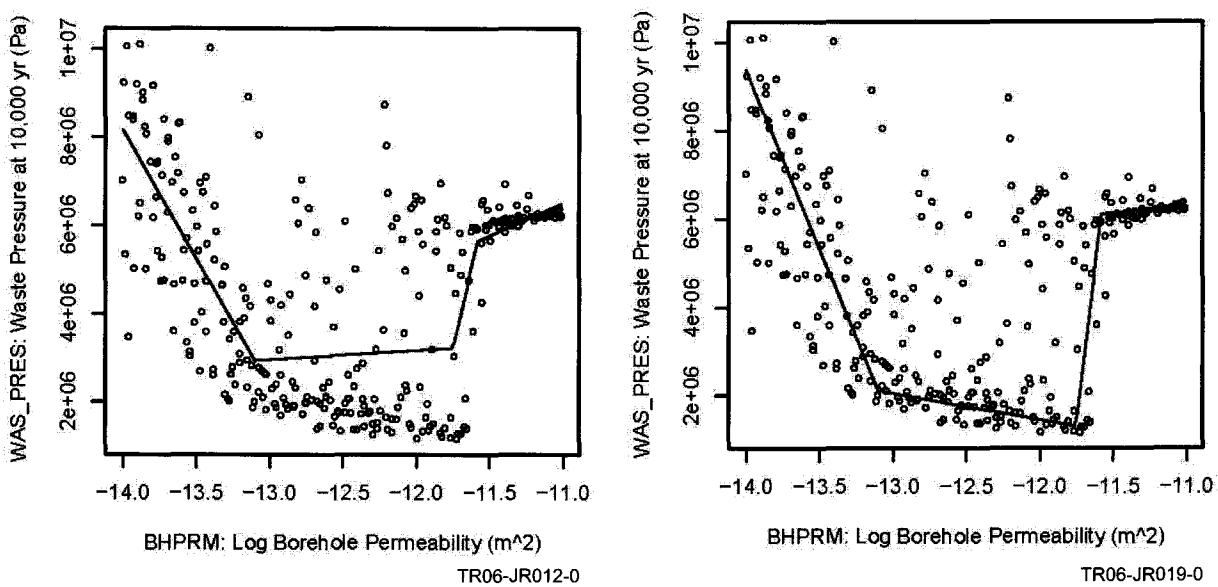
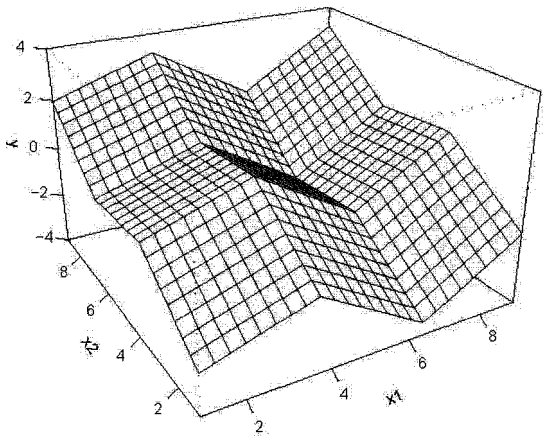


Fig. 12. Recursive partitioning regression on results generated in a sensitivity analysis of a two-phase fluid flow model with individual regression lines constrained to meet continuously: (a) Individual regression lines generated with traditional least squares regression, and (b) Individual regression lines generated with robust regression in which the sum of squares is minimized over the middle two quartiles of the deviations from the regression line.



TR06-JR013-0

Fig. 13. Recursive partitioning regression constructed for $y = f(x_1, x_2) = \sin x_1 + (x_2 - 5)^3$ with individual regression surfaces constrained to meet continuously; see Eq. (3.35).

$$SST = \sum_{i=1}^{nS}(y_i - \bar{y})^2 = \sum_{i=1}^{nS}\left(y_i - \sum_{i=1}^{nS} y_i \Big/ nS\right)^2,$$

$$SSE_{rj} = SSE\left(\mathcal{A}_{rj1}\right) + SSE\left(\mathcal{A}_{rj2}\right),$$

and $SSE(\mathcal{A}_{rj1})$ and $SSE(\mathcal{A}_{rj2})$ denote the error sum of squares for the linear regressions associated with $\mathcal{A}_{rj1}$ and $\mathcal{A}_{rj2}$, respectively. The selection

$$[\mathcal{A}_1, \mathcal{A}_2] = [\mathcal{A}_{rj1}, \mathcal{A}_{rj2}] \tag{3.48}$$

is then made for the pair $[\mathcal{A}_{rj1}, \mathcal{A}_{rj2}]$ that has the largest value for R^2_{rj}.

With initial values for $\mathcal{A}_1$ and $\mathcal{A}_2$ determined, consideration is given to splitting $\mathcal{A}_1$ and $\mathcal{A}_2$ into two subsets to produce three sets of $\mathbf{x}_i$ values. This involves consideration of triples of sets of the form $[\mathcal{U}_1, \mathcal{V}_1, \mathcal{A}_2]$ and $[\mathcal{A}_1, \mathcal{U}_2, \mathcal{V}_2]$, where (i) $\mathcal{U}_1$ and $\mathcal{V}_1$ correspond to subsets of $\mathcal{A}_1$ obtained in a manner analogous to that used in the definition of $\mathcal{A}_{rj1}$ and $\mathcal{A}_{rj2}$ and (ii) $\mathcal{U}_2$ and $\mathcal{V}_2$ correspond to subsets of $\mathcal{A}_2$ also obtained in a manner analogous to that used in the definition of $\mathcal{A}_{rj1}$ and $\mathcal{A}_{rj2}$. A triple of sets $[\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3]$ is then defined that is equal to the $[\mathcal{U}_1, \mathcal{V}_1, \mathcal{A}_2]$ or $[\mathcal{A}_1, \mathcal{U}_2, \mathcal{V}_2]$ triple that has the highest R^2 value obtained in a manner analogous to that described in conjunction with Eq. (3.47) except that results obtained from regressions involving three sets are involved. This process of constructing additional sets is then continued in an analogous manner until further splitting would not be beneficial as determined by some stopping criterion.

Prediction of $y = f(\mathbf{x})$ for arbitrary values of $\mathbf{x}$ is straightforward once the construction process to obtain $\mathcal{A}_s$, $\hat{\alpha}_s$ and $\hat{\beta}_s$ is complete. Specifically, the desired prediction follows directly from Eq. (3.44).

Since the determination of the partition regions is data driven (i.e., based on the observed y values), the smoother matrix $\mathbf{S}$ for recursive partitioning regression depends on the y values and is hence not a linear smoother. Because of this, an equivalent degrees of freedom is hard to define. However, a possible definition is to use the degrees of freedom from the model obtained as if the partitions had been specified *a priori*, and then add a certain number of degrees of freedom for each partition.

If the partitions were known *a priori*, then the smoother matrix derives from the regression analyses carried out for each set $\mathcal{A}_s$, $s = 1, 2, \ldots, nP$, and can be constructed from the design matrices $\mathbf{X}_s$ associated with these regressions (see Eq. (2.6)). In particular, $\mathbf{S}$ is constructed from the matrices

$$\mathbf{H}_s = \mathbf{X}_s\left(\mathbf{X}_s^T\mathbf{X}_s\right)^{-1}\mathbf{X}_s^T \tag{3.49}$$

and has the form

$$\mathbf{S} = \begin{bmatrix} \mathbf{H}_1 & 0 & \cdots & 0 \\ 0 & \mathbf{H}_2 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & \mathbf{H}_{nP} \end{bmatrix} \tag{3.50}$$

when the rows of $\mathbf{X}$ are rearranged so rows that correspond to elements of $\mathcal{A}_s$ are next to each other. Further, the degrees of freedom for the model with the sets $\mathcal{A}_s$ specified *a priori* is

$$df_{ap} = tr(\mathbf{S}) = \sum_{s=1}^{nP} tr(\mathbf{H}_s), \tag{3.51}$$

which corresponds to the number of degrees of freedom associated with $\mathbf{S}$. Given that the regression for each set $\mathcal{A}_s$ involves the determination of $nX + 1$ parameters (i.e., the coefficients in the regression model), $tr(\mathbf{H}_s) = nX + 1$; as a result,

$$df_{ap} = nP(nX + 1) \tag{3.52}$$

is the degrees of freedom for $\mathbf{S}$.

However, as the sets $\mathcal{A}_s$, $s = 1, 2, \ldots, nP$, are not specified *a priori*, additional degrees of freedom are involved in the estimation of these partitions. Since the complexity of the partitioned regions increases with the number of independent variables, each additional partition can be viewed as involving another nX degrees of freedom. As a result,

$$df = nP(nX+1)+nX(nP-1) \tag{3.53}$$

is an estimate of the degrees of freedom for the entire recursive partitioning model.

In our experience this rule works quite well for determining equivalent degrees of freedom for the recursive partitioning procedure described above. Additional discussion about equivalent degrees of freedom for adaptive or "data driven" approaches such as recursive partitioning and Multivariate Adaptive Regression Splines (MARS) is given in Hastie et al.[133]

Determining the number of sets $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_{nP}$ to use in recursive partitioning regression is analogous to choosing the smoothing parameter in previously described methods. Therefore, a reasonable approach is to determine a stopping point in the partitioning process with a criterion similar to that used for the selection of a smoothing parameter such as cross validation or generalized cross validation (Sect. 3.2). With cross validation, the PRESS value PRS is calculated as indicated in Eq. (3.19); similarly, with generalized cross validation, the adjusted PRESS value PRS_A is calculated as indicated in Eqs. (3.22) and (3.25). Then, if PRS_s, $s = 1, 2, \ldots$, and PRS_{As}, $s = 1, 2, \ldots$, represent values for PRS and PRS_A, respectively, calculated at successive steps in the partitioning process, an appropriate stopping point would be the last step before these values begin to increase as such an increase is indicative of an overfitting of the data.

As is the case for LOESS (Sect. 3.3.1), additive models (Sect. 3.3.2), and projection pursuit regression (Sect. 3.3.3), recursive partitioning regression can model very general nonlinear relationships. It also models very general interactions and performs well in higher dimensions. Unlike projection pursuit regression, the results in Sect. 5 do not indicate a tendency to overfit the data. However, these desirable properties come at a cost as recursive partitioning regression can require an order of magnitude more computational effort than the other indicated methods.

3.4 Hypothesis Testing for Variable Importance

A number of possibilities exist for hypothesis testing for smoothing methods, including the use of approximate distributions and bootstrapping.[122] For reasons of computational efficiency, the following approach to hypothesis testing is practicable for use in sensitivity analyses employing stepwise nonparametric regression and is used in this presentation for that reason.

It is desired to compare results (i.e., estimates of y) obtained with a model constructed from n independent variables (i.e., variables corresponding to n different elements of $\mathbf{x} = [x_1, x_2, \ldots, x_{nX}]$) with results obtained from a model constructed without one of these n variables, say x_j. The goal is to test the following hypotheses:

H_0: Results obtained with and without the inclusion of x_j are the same.

H_a: Results obtained with the inclusion of x_j are different from the results obtained with the exclusion of x_j.

If H_0 can be rejected in favor of H_a, then x_j is an important variable and should be included in the model being constructed by the smoothing process.

The usual test statistic for choosing between H_0 and H_a for linear models is

$$F^* = \frac{(SSE_R - SSE_F)/(df_F - df_R)}{SSE_F/df_F}, \tag{3.54}$$

where SSE_R and df_R are the error sum of squares and degrees of freedom for the reduced model (i.e., the model without x_j) and SSE_F and df_F are defined similarly for the full model (i.e., the model with x_j included) (Ref. 134, p. 169; Ref. 135, Sect. 4.4). In the testing of linear models with normally distributed data, F^* has an F-distribution with $m = df_F - df_R$ and $n = df_F$ degrees of freedom when H_0 is true. As a result, a p-value equal to $prob(F > F^*|H_0)$ can be used to test H_0 against H_a, where $prob(F > F^*|H_0)$ is the probability that a value F for the F-statistic greater than F^* will be obtained by chance if the null hypothesis H_0 is satisfied.

The statistic F^* in Eq. (3.54) can also be defined for models with $n - 1$ and n variables constructed in a smoothing process with df_R and df_F defined by

$$df_R = tr(\mathbf{S}_R) \text{ and } dF_F = tr(\mathbf{S}_F), \tag{3.55}$$

where $\mathbf{S}_R$ and $\mathbf{S}_F$ are the smoother matrices associated with the reduced model (i.e., the model without x_j) and the full model (i.e., the model with x_j included), respectively. Unfortunately, the true distribution for F^* is not known for any of the smoothing methods considered in this presentation. However, the distribution for F^* for these smoothing methods can be approximated by an F-distribution with $df_F - df_R$ and df_F degrees of freedom (i.e., F_{rs} with $r = df_F - df_R$ and $s = df_F$; see pp. 66 – 67, Ref. 120).

Determination of whether or not a particular variable should be included in a smoothing process can be made by fitting the associated model with and without the variable and then performing the appropriate F test. This is particularly useful in sensitivity analysis where the objective is to identify the important variables.

Performance of a comprehensive robustness study of this approach would be very beneficial. Our experience with the results contained in this presentation indicates that the approach is quite reasonable. Other approximate tests for H_0 are also available (pp. 87 – 89, Ref. 122); however, none of these tests are exact. Fortunately, such tests in and of themselves do not have a large bearing on which variables are identified as being important in the stepwise procedures described in this presentation. Rather, it is the contribution of a variable to the model R^2 value that serves as the metric for variable importance (see discussion of R^2 in Sect. 5.1). Hypothesis testing merely serves as a model building tool in the stepwise variable selection discussed in Sect. 4.1.

4. Implementation of Smoothing Methods for Sensitivity Analysis

Explanations are now given on how the different smoothing methods for surface approximation can be used in sensitivity analysis. Details about the forward (i.e., stepwise) model building process and smoothing parameter selection are given. In particular, the following topics are considered in the context of sensitivity analysis: stepwise variable selection (Sect. 4.1), traditional regression methods (Sect. 4.2), locally weighted regression, i.e., LOESS (Sect. 4.3), generalized additive models (Sect. 4.4), projection pursuit regression (Sect. 4.5), and recursive partitioning regression (Sect. 4.6). All of the techniques discussed here and used in the examples of Sect. 5 were implemented using the R language, which is an open source language very similar to S-Plus.

4.1 Stepwise Variable Selection

For purposes of sensitivity analysis, all of the presented regression (i.e., smoothing) methods can be implemented with a forward stepwise variable selection procedure. An approach of this type is essential for sensitivity analyses as there are usually a large number of uncertain analysis inputs under consideration (e.g., $nX \cong 150$ in the NUREG-1150 probabilistic risk assessments,[55, 136-139] $nX \cong 60$ in the compliance certification application for the Waste Isolation Pilot Plant,[140] and $nX \cong 250$ in an analysis for the proposed Yucca Mountain facility for the disposal of high level radioactive waste[141, 142]). Nonparametric regression techniques are not suitable for constructing models that contain a large number of independent variables unless the sample size is very large. Hence, it is essential to have a method that does not include all the variables under consideration in a model at once. Further, the order in which variables are selected in an appropriately designed stepwise procedure provides important sensitivity information.

A forward stepwise selection procedure operates in the following manner. A single variable model is constructed using each of the independent variables. Thus, if nX independent variables are under consideration, this results in the construction of nX single variable models. The variable, say $\tilde{x}_1$, associated with the best of these models is identified and retained. Then, two variable models are constructed using $\tilde{x}_1$ and each of the remaining $nX - 1$ variables. This results in the construction of $nX - 1$ two variable models. The variable, say $\tilde{x}_2$, associated with the best of these models is identified and retained. The process then continues with the construction of three variable models with $\tilde{x}_1$, $\tilde{x}_2$, and the remaining $nX - 2$ variables, and so on. This process continues until some stopping criterion is reached that indicates that no additional predictive capability is provided by models with additional variables.

Two important questions are left unanswered in the preceding paragraph: (i) What determines which model, and hence which variable, is best in a set of models?, and (ii) What is an appropriate stopping criterion? The best model is usually determined on the basis of a p-value (Sect. 3.4). The variable associated with the model with the smallest p-value is considered to provide the most predictive capability and is thus the variable retained for use in the next step in the model construction process. The p-value is also used to provide a stopping criterion. In particular, when the minimum p-value over all models is greater than some cutoff value (e.g., 0.02), no variable is selected and the model construction process terminates with the model constructed at the preceding step.

Variable selection can also be based on the PRESS statistic (see Eq. (3.19)). At each step in the selection process, the variable whose inclusion results in the smallest PRESS value is the variable retained. If the minimum PRESS value at a step is larger than the minimum PRESS value of the preceding step, then no variable is selected and the model construction process terminates with the model constructed at the preceding step. Other selection and stopping criteria are also possible, including Akaike's information criterion (p. 158, Ref. 120), adjusted R^2 values (described at the beginning of Sect. 5), and the adjusted PRESS statistic (described in Sect. 3.2). An enhancement of the forward selection procedure is to allow for the possibility of a previously selected variable being dropped from the modeling process if it no longer contributes significant predictive capability as additional variables are selected and included in the model.

Backward stepwise selection involves fitting a model with all nX variables. Then, unimportant variables are sequentially removed until the removal of additional variables reduces the predictive capability of the model. At this point the process is terminated. This selection procedure is not appropriate for sensitivity analysis with nonparametric regression models for two reasons. First, the construction of nonparametric regression models with a large number of variables is not possible with relatively small sample sizes. Second, the backward selection procedure is not useful for identify-

ing the importance of individual variables, which is the primary goal of sensitivity analysis. In contrast, a well designed forward selection procedure identifies the most important variable at the first step, then the next most important variable at the second step, and so on.

The sensitivity results presented in Sect. 5 use a p-value criterion (Sect. 3.4) for both individual variable selection and termination of the model construction process. Preliminary results indicated that use of a PRESS criterion was too computationally demanding for some of the regression methods and also resulted in models that tended to overfit the data. Our experience is that using either the p-value with a cutoff of $\alpha = 0.02$ or the adjusted PRESS statistic PRS_A for model selection usually results in the same model.

4.2 Traditional Regression: Linear Regression (LIN_REG), Rank Regression (RANK_REG) and Quadratic Regression (QUAD_REG)

Each of the traditional regression approaches (i.e., LIN_REG, RANK_REG and QUAD_REG) is implemented the same way with a forward stepwise selection procedure and a p-value criterion of $\alpha = 0.02$ (Table 1). The forward selection procedure with QUAD_REG requires some additional explanation as there are many ways to structure this procedure to incorporate variable interactions and squares. The approach taken is to consider a variable, its square, and all two-way interactions at each step of the selection procedure. Thus, if $\tilde{x}_1$ is the first variable selected, then the corresponding model would be of the form

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 \tilde{x}_1 + \hat{\beta}_{11} \tilde{x}_1^2. \tag{4.1}$$

Then, if $\tilde{x}_2$ is the second variable selected, the corresponding model would be of the form

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 \tilde{x}_1 + \hat{\beta}_2 \tilde{x}_2 + \hat{\beta}_{12} \tilde{x}_1 \tilde{x}_2 + \hat{\beta}_{11} \tilde{x}_1^2 + \hat{\beta}_{22} \tilde{x}_2^2, \tag{4.2}$$

and so on.

4.3 Locally Weighted Regression (LOESS)

The forward stepwise procedure with LOESS uses a two stage variable selection process at each step (Table 2). First, multiple spans (i.e., r/nS, where $r - 1$ is the number of points averaged over and nS is the sample size; see Sect. 3.1.3) are considered for each candidate variable, and a LOESS model is constructed for each span. Specifically, models are constructed for the following spans: 10, 0.7, 0.3, 0.1, 0.07 and 0.05, where 10 is simply an indicator for the use of a linear regression model. This results in five models for each candidate variable. Then, the "best" of these five models is selected on the basis of generalized cross validation using the adjusted PRESS value PRS_A defined in Eqs. (3.22) and (3.25). This produces one selected model (i.e., the model with the smallest value for PRS_A) for each candidate variable. Second, the "best" of the selected models for the individual candidate variables is identified with the approximate hypothesis test indicated in conjunction with Eqs. (3.54). Specifically, the model with the smallest p-value is identified, and the associated candidate variable is the variable selected at that step in the stepwise procedure. The procedure terminates with no variable selected if all p-values exceed $\alpha = 0.02$.

The flexibility provided by the different choices for span causes a potential problem with the approximate hypothesis test indicated in conjunction with Eq. (3.54). In particular, adding a variable to an existing model could result in a new model with the same or fewer degrees of freedom. This can happen if the span selected for the new model is much larger than the span for the previous model. This possibility exists because smaller spans produce more complex models and thus result in models with larger numbers of equivalent degrees of freedom. In turn, this would result in the numerator degrees of freedom (i.e., $df_F - df_R$) for the F statistic in Eq. (3.54) being less than zero.

This problem is handled in the following manner with SSE_R, SSE_F, df_R and df_F defined as in Eq. (3.54):

(i) If $df_F > df_R$, define the p-value with the F test as usual and test against α.

(ii) If $df_F - df_R = 0$, then define the p-value by

$$p\text{-value} = \begin{cases} 0 & \text{if } SSE_F < SSE_R \\ 1 & \text{if } SSE_F \geq SSE_R \end{cases} \tag{4.3}$$

and test against α.

(iii) If $df_F \leq df_R$, a p-value is defined with an F test involving

$$F^* = \frac{(SSE_F - SSE_R)/(df_R - df_F)}{SSE_R / df_R}, \tag{4.4}$$

Table 1. Forward Stepwise Variable Selection Algorithm for Sensitivity Analysis with LIN_REG, RANK_REG and QUAD_REG

Step 1. Estimate $y \cong f_{1j}(x_j)$ with regression procedure in use (i.e., LIN_REG, RANK_REG or QUAD_REG) for j = 1, 2, ..., nX. For each of the models $y \cong f_{1j}(x_j)$, determine (i) degrees of freedom df_j (i.e., $df_j = 2$ for LIN_REG and RANK_REG and $df_j = 3$ for QUAD_REG), (ii) F-statistic F_j for comparison against mean only model, and (iii) resultant p-value p_j (see Eq. (3.54)). Variable $\tilde{x}_1$ with smallest p-value is selected as most important variable at Step 1; corresponding model and degrees of freedom are represented by $y \cong f_1(\tilde{x}_1)$ and $\widetilde{df}_1$, respectively. The process terminates with no variable selected if all p_j are greater than a specified cutoff (e.g., $\alpha = 0.02$).

Step 2. Estimate $y \cong f_{2j}(\tilde{x}_1, x_j)$ with regression procedure in use (i.e., LIN_REG, RANK_REG or QUAD_REG) for j = 1, 2, ..., nX and $x_j \neq \tilde{x}_1$. For each of the models $y \cong f_{2j}(\tilde{x}_1, x_j)$, determine (i) degrees of freedom df_j (i.e., $df_j = 3$ for LIN_REG and RANK_REG and $df_j = 5$ for QUAD_REG), (ii) F-statistic F_j for comparison against model $y \cong f_1(\tilde{x}_1)$ selected at Step 1, and (iii) resultant p-value p_j (see Eq. (3.54)). Variable $\tilde{x}_2$ with smallest p-value is selected as most important variable at Step 2; corresponding model and degrees of freedom are represented by $y \cong f_2(\tilde{x}_1, \tilde{x}_2)$ and $\widetilde{df}_2$, respectively. The process terminates with no variable selected at Step 2 if all p_j are greater than a specified cutoff (e.g., $\alpha = 0.02$).

Step 3. Estimate $y \cong f_{3j}(\tilde{x}_1, \tilde{x}_2, x_j)$ with regression procedure in use (i.e., LIN_REG, RANK_REG or QUAD_REG) for j = 1, 2, ..., nX, $x_j \neq \tilde{x}_1$ and $x_j \neq \tilde{x}_2$. Continue as in Step 2.

...

Step N. Terminate process when no variable satisfies specified cutoff.

which is the original F statistic (see Eq. (3.54)) with the roles of the two models (i.e., the reduced or old model and the full or new model) reversed. The new variable should be added to the model only if evidence exists that the resultant model is better than the previous model. Such evidence is provided if the new model results in a significant reduction in the degrees of freedom (i.e., $df_R - df_F > 0$) without significantly increasing the error sum of squares (i.e., if $SSE_F - SSE_R$ is "small"). The preceding implies that the old model should be rejected in favor of the new model for sufficiently small values of F^* as defined in Eq. (4.4). In particular, the associated p-value is given by $prob(F < F^*)$ for an F distribution with $m = df_R - df_F$ and $n = df_R$ degrees of freedom. For the special case $SSE_F - SSE_R \leq 0$, the corresponding p-value is assumed to be zero. The usual test against α is made (i.e., the new model is accepted if the resultant p-value is less than α).

4.4 Generalized Additive Models (GAMs)

Additive models are now considered. Such models are designated as GAMs (generalized additive models) after the *gam* function in the R and S-Plus languages (p. 252, Ref. 120). The descriptor "generalized" is used to indicate fitting of a discontinuous response. All models constructed in this presentation assume continuous responses; however, the designator GAM is used to be consistent with the *gam* function in R and S-Plus.

Similarly to the forward stepwise procedure with LOESS (Table 2), the forward stepwise procedure with GAMs uses a two stage variable selection process at each step (Table 3). First, multiple additive models are constructed for each candidate variable, and the "best" of these models is selected for each candidate variable. Second, the "best" of the selected models for the individual variables is identified with the approximate hypothesis test indicated in conjunction with Eq. (3.54). Specifically, the model with the smallest p-value is identified, and the associated candidate variable is the variable selected at that step. The process terminates with no variable selected if all p-values exceed $\alpha = 0.02$.

Table 2. Forward Stepwise Variable Selection Algorithm for Sensitivity Analysis with LOESS

Notation: Variables s_k, $k = 1, 2, \ldots, 5$, represent the candidate spans 10, 0.7, 0.3, 0.1, 0.07 and 0.05 described in Sect. 4.3 with $s_1 = 10$ designating the use of linear regression.

Step 1. Estimate $y \cong f_{1jk}\left(x_j \middle| s_k\right)$ with LOESS for $j = 1, 2, \ldots, nX$ and $k = 1, 2, \ldots, 6$. For each x_j, identify span $\tilde{s}_j$ that results in the smallest value for the adjusted PRESS statistic PRS_A (see Eqs. (3.22) and (3.25)) for the models $y \cong f_{1jk}\left(x_j \middle| s_k\right)$, $k = 1, 2, \ldots, 6$. For each of the models $y \cong f_{1j}\left(x_j \middle| \tilde{s}_j\right)$, determine (i) degrees of freedom df_j (i.e., $df_j = tr\,(\mathbf{S}_{Fj})$, where $\mathbf{S}_{Fj}$ is the smoother matrix associated with the selected span $\tilde{s}_j$ for x_j (see Eq. (3.54) and associated discussion in Sect. 3.4), (ii) F-statistic F_j for comparison against mean only model, and (iii) resultant p-value p_j (see Eq. (3.54) and associated discussion in Sect. 4.3). Variable $\tilde{x}_1$ with smallest p-value is selected as most important variable at Step 1; corresponding model and degrees of freedom are represented by $y \cong f_1\left(\tilde{x}_1 \middle| \tilde{s}_1\right)$ and $\widetilde{df}_1$, respectively. The process terminates with no variable selected if all p_j are greater than a specified cutoff (e.g., $\alpha = 0.02$).

Step 2. Estimate $y \cong f_{2j}\left(\tilde{x}_1, x_j \middle| s_k\right)$ with LOESS for $j = 1, 2, \ldots, nX$, $x_j \neq \tilde{x}_1$, and $k = 1, 2, \ldots, 6$. For each x_j, identify span $\tilde{s}_j$ that results in smallest value for the adjusted PRESS statistic PRS_A (see Eqs. (3.22) and (3.25)) for the models $y \cong f_{2jk}\left(\tilde{x}_1, x_j \middle| s_k\right)$, $k = 1, 2, \ldots, 6$. For each of the models $y \cong f_{2j}\left(\tilde{x}_1, x_j \middle| \tilde{s}_j\right)$, determine (i) degrees of freedom df_j, (ii) F-statistic F_j for comparison against model $y \cong f_1\left(\tilde{x}_1 \middle| \tilde{s}_1\right)$ selected in Step 1, and (iii) resultant p-value p_j (see Eq. (3.54) and associated discussion in Sect. 4.3). Variable $\tilde{x}_2$ with smallest p-value is selected as most important variable at Step 2; corresponding model and degrees of freedom are represented by $y \cong f_2\left(\tilde{x}_1, \tilde{x}_2 \middle| \tilde{s}_2\right)$ and $\widetilde{df}_2$, respectively. The process terminates with no variable selected at Step 2 if all p_j are greater than a specified cutoff (e.g., $\alpha = 0.02$).

Step 3. Estimate $y \cong f_{2jk}\left(\tilde{x}_1, \tilde{x}_2, x_j \middle| s_k\right)$ with LOESS for $j = 1, 2, \ldots, nX$, $x_j \neq \tilde{x}_1$, $x_j \neq \tilde{x}_2$, and $k = 1, 2, \ldots, 6$. Continue as in Step 2.

...

Step N. Terminate process when no variable satisfies specified cutoff.

An additive model is constructed by repeatedly smoothing residuals across one independent variable at a time (Sect. 3.3.2). In concept, any scatterplot smoother could be used. However, the *gam* function in R and *S*-Plus is restricted to LOESS (Sect. 3.1.3) and/or smoothing splines (Sect. 3.1.4). Both smoothers usually gave similar results in some preliminary analyses, but an occasional convergence problem was encountered with LOESS. Therefore, smoothing splines are indicated in Table 3 and used in the generation of the GAM results presented in Sect. 5. Specifically, at a given step in the stepwise procedure, multiple degrees of freedom are considered (i.e., 1, 2, 4, 7, 10, 15) for each candidate variable, and a GAM is constructed using smoothing splines for each of these degrees of freedom. This results in six models for each candidate variable. After this construction, it is then necessary to select that "best" of the these six models for each candidate variable.

The indicated selection is made on the basis of generalized cross validation (Sect. 3.2) employing the adjusted PRESS values PRS_A (see Eqs. (3.22) – (3.25)). This criterion for model selection was picked because there is not an option associated with the *gam* function to use cross validation (Sect. 3.2) inside the back fitting algorithm. Further, computing PRESS (see Eq. (3.19)) is difficult because the leverage values s_{ii} are not obtainable for a *gam* fit in R and thus cannot be used in a computationally efficient calculation of PRESS (see Eq. (3.21)). As a result, obtaining the PRESS statistic in R would require fitting a model nS times, where nS is the sample size, and then making nS predictions. Thus, use of cross validation with PRESS in the generation of GAMs with R is computationally very expensive.

Table 3. Forward Stepwise Variable Selection Algorithm for Sensitivity Analysis with GAMs

Notation. Variables λ_k, $k = 1, 2, \ldots, 6$, represent candidate smoothing parameters used in smoothing splines (see Eq. (3.13)) in the sequential construction of GAMs, with λ_k resulting in a smoothing process with approximately δ_k degrees of freedom. Specifically, $\lambda_1 \sim \delta_1 = 1$, $\lambda_2 \sim \delta_2 = 2$, $\lambda_3 \sim \delta_3 = 4$, $\lambda_4 \sim \delta_4 = 7$, $\lambda_5 \sim \delta_5 = 10$, and $\lambda_6 \sim \delta_6 = 15$. The actual value used for λ_k in Eq. (3.13) is determined from the specified value for δ_k (see Sect. 3.5, Ref. [120] and Sect. 7.4.1, Ref. [126]).

Step 1. Estimate $y \cong f_{1jk}\left(x_j \middle| \lambda_k\right)$ with a smoothing spline on (x_{ij}, y_i), $i = 1, 2, \ldots, nS$, for $j = 1, 2, \ldots, nX$ and $k = 1, 2, \ldots, 6$. For each x_j, select model $f_{1j}\left(x_j \middle| \tilde{\lambda}_j\right)$ with smoothing parameter $\tilde{\lambda}_j$ from the models $y \cong f_{1jk}\left(x_j \middle| \lambda_k\right)$, $k = 1, 2, \ldots, 5$, that results in the smallest value for the adjusted PRESS statistic PRS_A (see Eqs. (3.22) and (3.25) and discussion in Sect. 4.4). For each of the models $y \cong f_{1j}\left(x_j \middle| \tilde{\lambda}_j\right)$, determine (i) degrees of freedom (i.e., $df_j = \tilde{\delta}_j$, with $\tilde{\lambda}_j \sim \tilde{\delta}_j$; see discussion in Sect. 4.4), (ii) F-statistic F_j for comparison against mean only model, and (iii) resultant p-value p_j (see Eq. (3.54)). Variable $\tilde{x}_1$ with smallest p-value is selected as the most important variable at Step 1; corresponding model, smoothing parameter, and degrees of freedom are represented by $y \cong f_1\left(\tilde{x}_1 \middle| \tilde{\lambda}_1\right)$, $\tilde{\lambda}_1$ and $\widetilde{df}_1$, respectively. The process terminates with no variable selected if all p_j are greater than a specified cutoff (e.g., $\alpha = 0.02$).

Step 2. Estimate $y \cong f_{2jk}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \lambda_k\right)$ through a sequence of smoothing operations for $j = 1, 2, \ldots, nX$, $x_j \neq \tilde{x}_1$ and $k = 1, 2, \ldots, 6$; see Step 2′ for details. For each x_j, select model $f_{2j}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \tilde{\lambda}_j\right)$ with smoothing parameter $\tilde{\lambda}_j$ from the models $y \cong f_{2jk}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \lambda_k\right)$, $k = 1, 2, \ldots, 6$, that results in the smallest value for the adjusted PRESS statistic PRS_A (see Eqs. (3.22) and (3.25) and discussion in Sect. 4.4). For each of the models $y \cong f_{2j}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \tilde{\lambda}_j\right)$, determine (i) degrees of freedom (i.e., $df_j = \tilde{\delta}_1 + \tilde{\delta}_j$ with $\tilde{\lambda}_1 \sim \tilde{\delta}_1$ and $\tilde{\lambda}_j \sim \tilde{\delta}_j$; see discussion in Sect. 4.4), (ii) F-statistic F_j for comparison against model $y \cong f_1\left(\tilde{x}_1 \middle| \tilde{\lambda}_1\right)$ constructed in Step 1, and (iii) resultant p-value p_j (see Eq. (3.54)). Variable $\tilde{x}_2$ with smallest p-value is selected as most important variable at Step 2; corresponding model, smoothing parameter and degrees of freedom are represented by $y \cong f_2\left(\tilde{x}_1, \tilde{x}_2 \middle| \tilde{\lambda}_1, \tilde{\lambda}_2\right) = f_{21}\left(\tilde{x}_1 \middle| \tilde{\lambda}_1\right) + f_{22}\left(\tilde{x}_2 \middle| \tilde{\lambda}_2\right)$, $\tilde{\lambda}_2$ and $\widetilde{df}_2$, respectively, where $f_{21}\left(\tilde{x}_1 \middle| \tilde{\lambda}_1\right)$ is a smoothed estimate of y as function of $\tilde{x}_1$ (i.e., $f_{21}\left(\tilde{x}_1 \middle| \tilde{\lambda}_1\right)$ corresponds to $F_{jk,2l}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \lambda_j\right)$ in Step 2.4′ for the selected values for $\tilde{x}_2$ and $\tilde{\lambda}_2$) and $f_{22}\left(\tilde{x}_2 \middle| \tilde{\lambda}_2\right)$ is a smoothed estimate of y as a function of $\tilde{x}_2$ (i.e., $f_{22}\left(\tilde{x}_2 \middle| \tilde{\lambda}_2\right)$ corresponds to $F_{jk,2l+1}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \lambda_j\right)$ in Step 2.4′ for the selected values of $\tilde{x}_2$ and $\tilde{\lambda}_2$). The process terminates with no variable selected at Step 2 if all p_j are greater than a specified cutoff (e.g., $\alpha = 0.02$).

Step 2′. Procedure for obtaining smoothed model $f_{2jk}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \lambda_k\right)$ on the basis of a relative error criterion for variable x_j, $x_j \neq \tilde{x}_1$, and smoothing parameter λ_k in Step 2.

Step 2.1′. Estimate $F_{jk1}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \lambda_k\right) \cong y - f_1\left(\tilde{x}_1 \middle| \tilde{\lambda}_1\right)$ by smoothing on $\left(x_{ij}, y_i - f_1\left(\tilde{x}_{i1} \middle| \tilde{\lambda}_1\right)\right)$, $i = 1, 2, \ldots, nS$, with a smoothing spline and smoothing parameter λ_k. Result is estimate $y_i \cong G_{jk1}\left(\tilde{x}_{i1}, x_{ij} \middle| \tilde{\lambda}_1, \lambda_k\right) = f_1\left(\tilde{x}_{i1} \middle| \tilde{\lambda}_1\right) + F_{jk1}\left(\tilde{x}_{i1}, x_{ij} \middle| \tilde{\lambda}_1, \lambda_k\right)$.

Table 3. Forward Stepwise Variable Selection Algorithm for Sensitivity Analysis with GAMs (Continued)

Step 2.2′. Estimate $F_{jk2}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \lambda_k\right) \cong y - F_{jk1}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \lambda_k\right)$ by smoothing on $\left(\tilde{x}_{i1}, y_i - F_{jk1}\left(\tilde{x}_{i1}, x_{ij} \middle| \tilde{\lambda}_1, \lambda_k\right)\right)$, i = 1, 2, …, nS, with a smoothing spline and smoothing parameter $\tilde{\lambda}_1$. Then, estimate $F_{jk3}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \lambda_k\right) \cong y - F_{jk2}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \lambda_k\right)$ by smoothing on $\left(x_{ij}, y_i - F_{ij2}\left(\tilde{x}_i, x_{ij} \middle| \tilde{\lambda}_1, \lambda_k\right)\right)$, i = 1, 2, …, nS, with a smoothing spline and smoothing parameter λ_k. Result is estimate $y_i \cong G_{jk2}\left(\tilde{x}_{i1}, x_{ij} \middle| \tilde{\lambda}_1, \lambda_k\right) = F_{jk2}\left(\tilde{x}_{i1}, x_{ij} \middle| \tilde{\lambda}_1, \lambda_k\right) + F_{jk3}\left(\tilde{x}_{i1}, x_{ij} \middle| \tilde{\lambda}_1, \lambda_k\right)$.

Step 2.3′. Similar to Step 2.2′. First, estimate $F_{jk4}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \lambda_k\right) \cong y - F_{jk3}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \lambda_k\right)$ by smoothing on $\left(\tilde{x}_{i1}, y_i - F_{jk3}\left(\tilde{x}_{i1}, x_{ij} \middle| \tilde{\lambda}_1, \lambda_k\right)\right)$, i = 1, 2, …, nS, with a smoothing spline and associated smoothing parameter $\tilde{\lambda}_1$. Then, estimate $F_{jk5}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \lambda_k\right) \cong y_2 - F_{jk4}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \lambda_k\right)$ by smoothing on $\left(x_{ij}, y_i - F_{jk4}\left(\tilde{x}_{i1}, x_{ij} \middle| \tilde{\lambda}_1, \lambda_k\right)\right)$, i = 1, 2, …, nS, with a smoothing spline and associated smoothing parameter λ_k. Result is estimate $y_i \cong G_{jk3}\left(\tilde{x}_{i1}, x_{ij} \middle| \tilde{\lambda}_1, \lambda_k\right) = F_{jk4}\left(\tilde{x}_{i1}, x_{ij} \middle| \tilde{\lambda}_1, \lambda_k\right) + F_{jk5}\left(\tilde{x}_{i1}, x_{ij} \middle| \tilde{\lambda}_1, \lambda_k\right)$.

Step 2.4′. Continue as in Step 2.3′ until the relative error criterion $\left\|\mathbf{G}_{jk,l+1} - \mathbf{G}_{jkl}\right\| \le rerr \left\|\mathbf{G}_{jkl}\right\|$ is satisfied for $\mathbf{G}_{jkr} = \left[G_{jkr}\left(\tilde{x}_{11}, x_{1j} \middle| \tilde{\lambda}_1, \lambda_k\right), G_{jkr}\left(\tilde{x}_{21}, x_{2j} \middle| \tilde{\lambda}_1, \lambda_k\right), \ldots, G_{jkr}\left(\tilde{x}_{nS,1}, x_{nS,j} \middle| \tilde{\lambda}_1, \lambda_2\right)\right]$, $r = l$, l + 1, and $rerr = 10^{-7}$. At this point, the construction process stops; f_{2jk} is defined by $f_{2jk}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \lambda_k\right) = G_{jk,l+1}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \lambda_k\right) = F_{jk,2l}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \lambda_k\right) + F_{jk,2l+1}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \lambda_k\right)$, where $F_{jk,2l}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \lambda_k\right)$ is a smoothed estimate of y as a function of $\tilde{x}_1$ and $F_{jk,2l+1}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \lambda_k\right)$ is a smoothed estimate of y as a function of x_j; and the adjusted PRESS statistic PRS_{Ajk} is determined for the approximation to y defined by f_{2jk}.

Step 3. Similar to Step 2 with $y \cong f_{3jk}\left(\tilde{x}_1, \tilde{x}_2, x_j \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right)$ being estimated through a sequence of smoothing operations for j = 1, 2, …, nX, $x_j \ne \tilde{x}_1, x_j \ne \tilde{x}_2$, and k = 1, 2, …, 6; details of the estimation of $f_{3jk}\left(\tilde{x}_1, \tilde{x}_2, x_j \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right)$ are described in Step 3′ and are similar to those described in Step 2′ for the estimation of $f_{2jk}\left(\tilde{x}_1, x_j \middle| \tilde{\lambda}_1, \lambda_k\right)$ with the addition that the intermediate smoothings indicated in Steps 2.2′ and 2.3′ now involve $\tilde{x}_1$, $\tilde{x}_2$ and x_j rather than $\tilde{x}_1$ and x_j. Remainder of Step 3 is the same as in Step 2 and results in the selection of $\tilde{x}_3$ as the most important variable at Step 3.

Step 3′. Procedure for obtaining smoothed model $f_{3jk}\left(\tilde{x}_1, \tilde{x}_2, x_j \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right)$ on the basis of a relative error criterion for variable x_j, $x_j \ne \tilde{x}_1$, $x_j \ne \tilde{x}_2$, and smoothing parameter λ_k in Step 3.

Step 3.1′. Estimate $F_{jkl}\left(\tilde{x}_1, \tilde{x}_2 \middle| \tilde{\lambda}_1, \tilde{\lambda}_2\right) \cong y - f_1\left(\tilde{x}_1 \middle| \tilde{\lambda}_1\right)$ by smoothing on $\left(\tilde{x}_{i2}, y_i - f_1\left(\tilde{x}_{i1} \middle| \tilde{\lambda}_1\right)\right)$, i = 1, 2, …, nS, with a smoothing spline and smoothing parameter $\tilde{\lambda}_2$ (Note: $F_{jk1}\left(\tilde{x}_1, \tilde{x}_2 \middle| \tilde{\lambda}_1, \tilde{\lambda}_2\right)$ was previously determined in Step 2.1′ for $\tilde{x}_2 = x_j$ and $\tilde{\lambda}_2 = \lambda_k$). Estimate $f_{jk2}\left(\tilde{x}_1, \tilde{x}_2, x_j \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right) \cong y - f_1\left(\tilde{x}_1 \middle| \tilde{\lambda}_1\right) - F_{jkl}\left(\tilde{x}_1, \tilde{x}_2 \middle| \tilde{\lambda}_1, \tilde{\lambda}_2\right)$ by smoothing on $\left(x_{ij}, y_i - f_1\left(\tilde{x}_{i1} \middle| \tilde{\lambda}_1\right) - F_{jk1}\left(\tilde{x}_{i1}, \tilde{x}_{i2} \middle| \tilde{\lambda}_1, \tilde{\lambda}_2\right)\right)$, i = 1, 2, …, nS, with a smoothing spline and smoothing parameter λ_k. Result is estimate $y_i \cong G_{jk1}\left(\tilde{x}_{i1}, \tilde{x}_{i2}, x_{ij} \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right) = f_1\left(\tilde{x}_1 \middle| \tilde{\lambda}_1\right) + F_{jkl}\left(\tilde{x}_{i1}, \tilde{x}_{i2} \middle| \tilde{\lambda}_1, \tilde{\lambda}_2\right) + F_{jk2}\left(\tilde{x}_{i1}, \tilde{x}_{i2}, x_{ij} \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right)$.

Table 3. Forward Stepwise Variable Selection Algorithm for Sensitivity Analysis with GAMs (Continued)

Step 3.2′. Estimate $F_{jk3}\left(\tilde{x}_1, \tilde{x}_2, x_j \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right) \cong y - F_{jk1}\left(\tilde{x}_1, \tilde{x}_2 \middle| \tilde{\lambda}_1, \tilde{\lambda}_2\right) - F_{jk2}\left(\tilde{x}_1, \tilde{x}_2, x_j \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right)$ with a smoothing spline on $\tilde{x}_1$ and smoothing parameter $\tilde{\lambda}_1$. Then, estimate $F_{jk4}\left(\tilde{x}_1, \tilde{x}_2, x_j \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right) \cong y - F_{jk2}\left(\tilde{x}_1, \tilde{x}_2, x_j \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right) - F_{jk3}\left(\tilde{x}_1, \tilde{x}_2, x_j \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right)$ with a smoothing spline on $\tilde{x}_2$ and smoothing parameter $\tilde{\lambda}_2$, and estimate $F_{jk5}\left(\tilde{x}_1, \tilde{x}_2, x_j \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_j\right) \cong y - F_{jk3}\left(\tilde{x}_1, \tilde{x}_2, x_j \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right) - F_{jk4}\left(\tilde{x}_1, \tilde{x}_2, x_j \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right)$ with a smoothing spline on x_j and smoothing parameter λ_k. Result is estimate $y_i \cong G_{jk2}\left(\tilde{x}_{i1}, \tilde{x}_{i2}, x_{ij} \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right) = \sum_{r=3}^{5} F_{jkr}\left(\tilde{x}_{i1}, \tilde{x}_{i2}, x_{ij} \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right)$.

Step 3.3′. Similar to Step 3.2′ with $F_{jkr}\left(\tilde{x}_1, \tilde{x}_2, x_j \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right) \cong y - F_{jk,r-2}\left(\tilde{x}_1, \tilde{x}_2, x_j \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right) - F_{jk,r-1}\left(\tilde{x}_1, \tilde{x}_2, x_j \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right)$ being estimated for r = 6, 7 and 8 by smoothing on $\tilde{x}_1$, $\tilde{x}_2$ and x_j, respectively, with corresponding smoothing parameters $\tilde{\lambda}_1$, $\tilde{\lambda}_2$ and λ_k. Result is estimate $y_i \cong G_{jk3}\left(\tilde{x}_{i1}, \tilde{x}_{i2}, x_{ij} \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right) = \sum_{r=6}^{8} F_{jkr}\left(\tilde{x}_{i1}, \tilde{x}_{i2}, x_{ij} \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right)$.

Step 3.4′. Continue as in Step 3.4′ until the relative error criterion $\left\|\mathbf{G}_{jk,l+1} - \mathbf{G}_{jkl}\right\| \le rerr \left\|G_{jkl}\right\|$ is satisfied for $\mathbf{G}_{jkr} = \left[G_{jkr}\left(\tilde{x}_{i1}, \tilde{x}_{i2}, \tilde{x}_{ij} \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right), G_{jkr}\left(\tilde{x}_{21}, \tilde{x}_{22}, \tilde{x}_{2j} \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right), \ldots, G_{jkr}\left(\tilde{x}_{nS,1}, \tilde{x}_{nS,2}, x_{nS,j} \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right)\right]$, $r = l, l+1$,and $rerr = 10^{-7}$. At this point, the construction process stops; f_{3jk} is defined by $f_{3jk}\left(\tilde{x}_1, \tilde{x}_2, x_j \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right) = G_{jk,l+1}\left(\tilde{x}_1, \tilde{x}_2, x_j \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right) = \sum_{r=3l}^{3l+2} F_{jkr}\left(\tilde{x}_1, \tilde{x}_2, x_j \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right)$, where $F_{jkr}\left(\tilde{x}_1, \tilde{x}_2, x_j \middle| \tilde{\lambda}_1, \tilde{\lambda}_2, \lambda_k\right)$ is a smoothed estimate of y as function of $\tilde{x}_1$, $\tilde{x}_2$ and x_j for $\tilde{\lambda}_1$, $\tilde{\lambda}_2$ and λ_k, respectively; and the adjusted PRESS statistic PRS_{Ajk} is determined for the approximation to y defined by f_{3jk}.

...

Step N. Terminate process when no variable satisfies specified cutoff.

In contrast, use of generalized cross validation allows a more computationally efficient determination of the "best" model associated with each candidate variable (Sect. 3.2). In particular, deleted residuals are not needed in generalized cross validation (see Eq. (3.21)). Instead, generalized cross validation is based on the adjusted PRESS value PRS_A, which uses $tr(\mathbf{S})$ in its evaluation (see Eqs. (3.22) – (3.25)). The value for $tr(\mathbf{S})$ can be estimated as indicated in Eq. (3.36). This estimation requires the degrees of freedom df_j (i.e., 1, 2, 4, 7, 10 or 15) used for each variable x_j in the scatterplot smoother employed in the backfitting algorithm. Because the values for df_j are known for each GAM constructed for a given candidate variable (see description of backfitting algorithm in Table 3), the determination of $tr(\mathbf{S})$ (see Eq. (3.36)) and hence PRS_A (see Eq. (3.22)) is straightforward for each of these GAMs. In turn, the selected (i.e., "best") GAM for a given candidate variable is the model with the smallest value for PRS_A. As already indicated, once the "best" GAM for each variable is identified, the "best" GAM overall is determined on the basis of the p-value associated with the approximate hypothesis test in Eq. (3.54), and the variable selected at the step under consideration is the variable associated with that model.

4.5 Projection Pursuit Regression (PP_REG)

Similarly to the forward stepwise procedures with LOESS (Table 2) and GAMs (Table 3), the forward stepwise procedure with PP_REG uses a two stage variable selection process at each step (Table 4). First, multiple PP_REG models are constructed for each candidate variable, and the "best" of these models is selected for each candidate variable. Second, the "best" of the selected models for the individual variables is identified with the approximate hypothesis test indicated in conjunction with Eq. (3.54). Specifically, the model with the smallest p-value is identified, and the associated variable is the variable selected at that step. The process terminates with no variable selected if all p-values exceed $\alpha = 0.02$.

Table 4. Forward Stepwise Variable Selection Algorithm for Sensitivity Analysis with PP_REG

Notation. Variables λ_k, $k = 1, 2, \ldots, 6$, represent candidate smoothing parameters used in smoothing splines (see Eq. (3.13)) in the sequential construction of PP_REG models, with λ_k resulting in a smoothing process with approximately δ_k degrees of freedom. Specifically, $\lambda_1 \sim \delta_1 = 1$, $\lambda_2 \sim \delta_2 = 2$, $\lambda_3 \sim \delta_3 = 4$, $\lambda_4 \sim \delta_4 = 7$, $\lambda_5 \sim \delta_5 = 10$, and $\lambda_6 \sim \delta_6 = 15$. However, unlike the stepwise construction procedure for GAMs described in Table 3, the degrees of freedom associated with smoothing splines in the stepwise construction of PP_REG models is obtained directly from the *ppr* subroutine in R rather than approximated from the δ_k's.

Step 1. Estimate $y \cong f_{1jk}\left(x_j \middle| \lambda_k\right)$ with a smoothing spline on (x_{ij}, y_i), $i = 1, 2, \ldots, nS$, for $j = 1, 2, \ldots, nX$ and $k = 1, 2, \ldots, 6$. For each x_j, select model $f_{1j}\left(x_j \middle| \tilde{\lambda}_j\right)$ with smoothing parameter $\tilde{\lambda}_j$ from the models $y \cong f_{1jk}\left(x_j \middle| \lambda_k\right)$, $k = 1, 2, \ldots, 6$, that results in the smallest value for the adjusted PRESS statistic PRS_A (see Eqs. (3.22) and (3.25) and discussion in Sect. 4.4). For each of the models $y \cong f_{1j}\left(x_j \middle| \tilde{\lambda}_j\right)$, determine (i) degrees of freedom, (ii) F-statistic F_j

for comparison against mean only model, and (iii) resultant p-value p_j (see Eq. (3.54)). Variable $\tilde{x}_1$ with smallest p value is selected as the most important variable at Step 1; corresponding model, smoothing parameter, and degrees of freedom are represented by $y \cong f_1\left(\tilde{x}_1 \middle| \tilde{\lambda}_1\right)$, $\tilde{\lambda}_1$ and $\widetilde{df}_1$, respectively. The process terminates with no variable selected if all p_j are greater than a specified cutoff (e.g., $\alpha = 0.02$).

Step 2. Estimate $y \cong f_{2j}\left(\tilde{x}_1, x_j \middle| \lambda_{j1}, \lambda_{j2}\right) = F_{j1}\left(\tilde{x}_1, x_j \middle| \lambda_{j1}\right) + F_{j2}\left(\tilde{x}_1, x_j \middle| \lambda_{j2}\right)$ for $j = 1, 2, \ldots, nX$ and $x_j \neq \tilde{x}_1$ through a sequential application of PP_REG described in Steps 2.1′ – 2.3′. For each of the models $y \cong f_{2j}\left(\tilde{x}_1, x_j \middle| \lambda_{j1}, \lambda_{j2}\right)$, determine (i) degrees of freedom, (ii) F statistic for comparison against model $y \cong f_1\left(\tilde{x}_1 \middle| \tilde{\lambda}_{11}\right)$ constructed in Step 1, and (iii) resultant p-value p_j (see Eq. (3.54)). Variable $\tilde{x}_2$ with smallest p-value is selected as most important variable at Step 2; corresponding model, smoothing parameters, and degrees of freedom are represented by $y \cong f_2\left(\tilde{x}_1, \tilde{x}_2 \middle| \tilde{\lambda}_{21}, \tilde{\lambda}_{22}\right)$, $\tilde{\lambda}_{21}$, $\tilde{\lambda}_{22}$, and $\widetilde{df}_2$, respectively. The process terminates with no variable selected at Step 2 if all p_j are greater than a specified cutoff (e.g., $\alpha = 0.02$).

Step 2′. Procedure for obtaining model $f_{2j}\left(\tilde{x}_1, x_j \middle| \lambda_{j1}, \lambda_{j2}\right)$ and smoothing parameters λ_{j1} and λ_{j2} for variable x_j, $x_j \neq \tilde{x}_1$, in Step 2 through a sequential application of PP_REG.

Step 2.1′. Estimate $y \cong F_{1jk}\left(\tilde{x}_1, x_j \middle| \lambda_k\right)$ for $k = 1, 2, \ldots, 6$ from the observations $\left(\left[\tilde{x}_{i1}, x_{ij}\right], y_i\right)$, $i = 1, 2, \ldots, nS$, with PP_REG as indicated in conjunction with Eq. (3.39). Determine adjusted PRESS value PRS_A (see Eqs. (3.22) and (3.25)) for each of the six models and select model $y \cong F_{j1}\left(\tilde{x}_1, x_j \middle| \lambda_{j1}\right)$ and associated smoothing parameter λ_{1j} with smallest value for PRS_A.

Step 2.2′. Estimate $F_{2jk}\left(\tilde{x}_1, x_j \middle| \lambda_k\right)$ for $k = 1, 2, \ldots, 6$ from the observations $\left(\left[\tilde{x}_{i1}, x_{ij}\right], y_i - F_{j1}\left(\tilde{x}_{i1}, x_{ij} \middle| \lambda_{j1}\right)\right)$, $i = 1, 2, \ldots, nS$, with PP_REG as indicated in conjunction with Eq. (3.39); this corresponds to the second step in a PP_REG as indicated in Eq. (3.40).

Table 4. Forward Stepwise Variable Selection Algorithm for Sensitivity Analysis with PP_REG (Continued)

Step 2.3′. Approximations to y are given by $F_{j1}\left(\tilde{x}_1, x_j \middle| \lambda_{j1}\right) + F_{2jk}\left(\tilde{x}_1, x_j \middle| \lambda_k\right)$ for $k = 1, 2, \ldots, 6$. For each of these six models, determine the adjusted PRESS value PRS_A (see Eqs. (3.22) and (3.25) and select the model with the smallest value for PRS_A. Specifically, with $F_{j2}\left(\tilde{x}_1, x_j \middle| \lambda_{j2}\right)$ and λ_{j2} representing the selected model and smoothing parameter, the desired approximation to y for x_j is given by $y \cong f_{2j}\left(\tilde{x}_1, x_j \middle| \lambda_{j1}, \lambda_{j2}\right) = F_{j1}\left(\tilde{x}_1, x_j \middle| \lambda_{j1}\right) + F_{2j}\left(\tilde{x}_1, x_j \middle| \lambda_{j2}\right)$ as indicated at the beginning of Step 2.

Step 3. Same as Step 2 but starting with estimate $y \cong f_{3j}\left(\tilde{x}_1, \tilde{x}_2, x_j \middle| \lambda_{j1}, \lambda_{j2}, \lambda_{j3}\right) = \sum_{s=1}^{3} F_{j3}\left(\tilde{x}_1, \tilde{x}_2, x_j \middle| \lambda_{js}\right)$ for $j = 1, 2, \ldots, nX$, $x_j \neq \tilde{x}_1$ and $x_j \neq \tilde{x}_2$ developed through a sequential application of PP_REG analogous to that described in Steps 2.1′ – 2.3′.

...

Step N. Terminate process when no variable satisfies specified cutoff.

The default implementation of PP_REG in the function *ppr* in R and S-Plus uses a scatterplot smoother called *supsmu*, which is a variable span smoother that usually provides a better fit to data than a fixed span smoother.[130] However, the bandwidth at a particular value for $\mathbf{x}_i$ depends on the values for y, which makes this smoother nonlinear. In turn, this makes the equivalent degrees of freedom difficult to define. Without the degrees of freedom, it is difficult to assess the quality of the associated model. Unfortunately, a high R^2 value by itself is not very informative because there is no way to know if an overfit of the data has occurred. A possibility is to use the PRESS statistic to assess the quality of the fit, but this can be very time consuming for moderately large samples. The preceding complication is avoided in this study by using the option of employing smoothing splines as the scatterplot smoother in *ppr* with a degrees of freedom δ_k specified for each smoothing operation (see Table 4). With this option, the resultant degrees of freedom associated with the smoothing operations can be obtained directly from *ppr* rather than approximated from the δ_k's as is done in the stepwise procedure from the construction of GAMs (see Table 3).

As iterative smoothing operations are applied, the possibility exists that the degrees of freedom will decrease when a variable is added to a model. This situation occurs when the successor model is less complex (i.e., involves less smoothing) than the predecessor model. When this occurs, the procedure described for use in the same situation with LOESS is applied (Sect. 4.3).

4.6 Recursive Partitioning Regression (RP_REG)

As for LOESS (Table 2), GAMs (Table 3) and PP_REG (Table 4), the forward stepwise procedure with RP_REG uses a two stage variable selection process at each step (Table 5). First, multiple RP_REG models are constructed for each candidate variable, and the "best" of these models is selected for each candidate variable. Second, the "best" of the selected models for the individual variables is identified with the approximate hypothesis test indicated in conjunction with Eq. (3.54). Specifically, the model with the smallest p-value is identified, and the associated candidate variable is the variable selected at that step. The process terminates with no variable selected if all p-values exceed $\alpha = 0.02$.

For each candidate variable at each step in the partitioning process, it is necessary to investigate a large number of possible split points (Sect. 3.3.4). Each possible split point requires the construction of a regression model. For example, if the partitioning process has reached the point that five variables are under consideration, then each possible split point requires the construction of a regression model with six parameters. To reduce the number of required regression constructions, every observation for a variable is not investigated as a possible split point. Instead, for every variable, a split point is considered at the smallest sampled value possible for use in splitting and then at every k^{th} observed value after that up to the largest sampled

Table 5. Forward Stepwise Variable Selection Algorithm for Sensitivity Analysis with RP_REG

Step 1. Estimate $y \cong f_{1j}(x_j)$ by performing RP_REG on (x_{ij}, y_i), $i = 1, 2, \ldots, nS$, for $j = 1, 2, \ldots, nX$ as described in Sect. 3.3.4. This results in partition sets $\mathcal{A}_{jk}$, $k = 1, 2, \ldots, nP_j$, of the values x_{ij}, $i = 1, 2, \ldots, nS$, and associated regressions $\hat{y}_{jk} = \hat{\beta}_{0jk} + \hat{\beta}_{1jk} x_j$, $k = 1, 2, \ldots, nP_j$, for each x_j. At each step in the partitioning process for x_j (i.e., for $nP_j = 2$, then $nP_j = 3$, and so on), the partition that results in the highest R^2 value is retained (see Eqs. (3.47) – (3.48) and associated discussion); the partitioning process for x_j is stopped when the partitioning of $\mathcal{A}_{jk}$, $k = 1, 2, \ldots, nP_j$, into $\tilde{\mathcal{A}}_{jk}$, $k = 1, 2, \ldots, nP_j + 1$, results in the model associated with the partitions $\tilde{\mathcal{A}}_{jk}$ that having a higher adjusted PRESS value PRS_A (see Eqs. (3.22) and (3.25)) than the model associated with the partitions $\mathcal{A}_{jk}$ (Note: Because of the sequential partitioning process, only two of the partitions in the sequence $\tilde{\mathcal{A}}_{jk}$, $k = 1, 2, \ldots, nP_j + 1$, differ from partitions in the sequence $\mathcal{A}_{jk}$, $k = 1, 2, \ldots, nP_j$). For the model constructed with x_j, determine (i) degrees of freedom $df_j = 3nP_j - 1$, (ii) F-statistic F_j for comparison against mean only model, and (iii) resultant p-value p_j (see Eq. (3.54)). Variable $\tilde{x}_1$ with smallest p-value is selected as most important variable at Step 1; the corresponding model and degrees of freedom are represented by $y \cong f_1(\tilde{x}_1)$ and $\widetilde{df}_1$, respectively. The process terminates with no variable selected if all p_j are greater than a specified cutoff (e.g., $\alpha = 0.02$).

Step 2. Estimate $y \cong f_{2j}(\tilde{x}_1, x_j)$ by performing RP_REG on $([\tilde{x}_{i1}, x_{ij}], y_i)$, $i = 1, 2, \ldots, nS$, for $j = 1, 2, \ldots, nX$ and $x_j \neq \tilde{x}_1$ as described in Sect. 3.3.4. This results in partition sets $\mathcal{A}_{jk}$, $k = 1, 2, \ldots, nP_j$, for the vectors $[\tilde{x}_{i1}, x_{ij}]$, $i = 1, 2, \ldots, nS$, and associated regressions $\hat{y}_{jk} = \hat{\beta}_{0jk} + \hat{\beta}_{1jk} \tilde{x}_1 + \hat{\beta}_{2jk} x_j$, $k = 1, 2, \ldots, nP_j$, for each x_j (Note: Construction of the partition sets for $[\tilde{x}_{i1}, x_{ij}]$, $i = 1, 2, \ldots, nS$, starts *ab initio* and does not involve the partition sets constructed for $\tilde{x}_1$ in Step 1). At each step in the partitioning process for x_j (i.e., for $nP_j = 2$, then $nP_j = 3$, and so on), the partition that results in the highest R^2 value is retained (see Eqs. (3.47) – (3.48) and associated discussion); the partitioning process associated with x_j is stopped when the partitioning of $\mathcal{A}_{jk}$, $k = 1, 2, \ldots, nP_j$, into $\tilde{\mathcal{A}}_{jk}$, $k = 1, 2, \ldots, nP_j + 1$, results in the model associated with the partitions $\tilde{\mathcal{A}}_{jk}$ having a higher adjusted PRESS value PRS_A (see Eqs. (3.22) and (3.25)) than the model associated with the partitions $\mathcal{A}_{jk}$ (Note: Because of the sequential partitioning process, only two of the partitions in the sequence $\tilde{\mathcal{A}}_{jk}$, $k = 1, 2, \ldots, nP_j + 1$, differ from partitions in the sequence $\mathcal{A}_{jk}$, $k = 1, 2, \ldots, nP_j$). For the model constructed with x_j, determine (i) degrees of freedom $df_j = 5nP_j - 2$, (ii) F-statistic F_j for comparison against model $y \cong f_1(\tilde{x}_1)$ selected in Step 1, and (iii) resultant p-value p_j (see Eq. (3.54)). Variable $\tilde{x}_2$ with smallest p-value is selected as most important variable at Step 2; the corresponding model and degrees of freedom are represented by $y \cong f_2(\tilde{x}_1, \tilde{x}_2)$ and $\widetilde{df}_2$, respectively. The process terminates with no variable selected if all p_j are greater than a specified cutoff (e.g., $\alpha = 0.02$).

Step 3. Estimate $y \cong f_{3j}(\tilde{x}_1, \tilde{x}_2, x_j)$ by performing RP_REG on $([\tilde{x}_{i1}, \tilde{x}_{i2}, x_{ij}], y_i)$, $i = 1, 2, \ldots, nS$, for $j = 1, 2, \ldots, nX$, $x_j \neq \tilde{x}_1$ and $x_j \neq \tilde{x}_2$ as described in Sect. 3.3.4. Continue as in Step 2.

...

Step N. Terminate process when no variable satisfies specified cutoff.

value possible for use in splitting. For example, if a sample of size $nS = 300$ is under consideration, $k = 3$ and the partitioning process has reached the point at which five independent variables are under consideration in the regression model construction, then the possible split points for the variable x_j would be $x_{(6),j}$, $x_{(9),j}$, ..., $x_{(291),j}$, $x_{(294),j}$, where $x_{(i)j}$, $i = 1, 2, \ldots, 300$, denotes a rank ordering of the observed values for variable x_j. In this example, the smallest possible value for splitting is $x_{(6),j}$ because at least six observations are required to estimate the six parameters in the associated regression model. In the examples presented in Sect. 5, $k = 2$ is used when $nS = 100$, and $k = 3$ is used when $nS = 300$.

As indicated in Table 5, the split point that results in the largest increase in R^2 defines the split point to be used (see Eqs. (3.47) – (3.48)). If the adjusted PRESS value PRS_A is smaller after the split than before, the split is kept and the search continues for the next possible split point. The construction process continues in this manner until PRS_A increases after a split, at which point the split is not kept and the model is completed for that step and the particular candidate variable under consideration. Then, the F-static and the associated p-value are determined for each model constructed at this step in a comparison with the model retained at the preceding step. As with the other procedures, a cutoff of $\alpha = 0.02$ for the approximate p-value is used in the stepwise variable selection procedure (Sect. 4.1) to determine whether or not a new variable should be retained in the stepwise procedure.

This page intentionally left blank.

5. Example Sensitivity Analysis Results

The efficacy of the various methods described in previous sections as procedures for sensitivity analysis is now investigated with both analytic test model data (Sect. 5.1) and real data (Sect. 5.2). The analytic test models were assembled as part of a review volume on sensitivity analysis.[48, 143] The real data comes from a performance assessment for the Waste Isolation Pilot Plant (WIPP).[56, 57] The methods are compared on the basis of fidelity to the data, overfitting of the data, and reproducibility.

The R^2 statistic (see Eq. (2.5)) provides one measure of the fidelity of a regression model to the data from which it was constructed. In particular, the closer R^2 is to one, the better the model reproduces the data. However, the R^2 statistic can be misleading in that its value can be unrealistically inflated by overfitting the data. The adjusted R^2 statistic R_A^2 provides a measure of fidelity that attempts to correct the effects of overfitting the data (pp. 91 – 92, Ref. 110). Specifically, R_A^2 is defined by

$$R_A^2 = 1 - \frac{(nS-1)\sum_{i=1}^{nS}(\hat{y}_i - y_i)^2}{(nS-p)\sum_{i=1}^{nS}(y_i - \bar{y})^2} = 1 - (1-R^2)\left(\frac{nS-1}{nS-p}\right), \quad (5.1)$$

where p is the number of degrees of freedom associated with the fitted model. However, the values for R^2 and R_A^2 are similar when p is small relative to nS.

The PRESS statistic PRS (see Eq. (3.19)) provides a way to test for an overfitting of the data. In particular, a decrease in PRS with the addition of a variable to a model indicates an improvement in the predictive capability of the model (i.e., the fidelity of the model to the data has increased). In contrast, an increase in PRS indicates that an overfitting of the data has taken place. This property results because the PRESS statistic is very sensitive to the effects of a limited number of highly influential observations (typically observations with extreme values for one or a few independent variables). Monitoring PRESS values as variables are added to a model provides a way to check for overfitting of the data, with such overfitting indicated when the addition of a variable results in an increase in the PRESS value over the PRESS value obtained before the addition of that variable. Such a jump in the PRESS statistic indicates the model is starting to "chase" results associated with individual observations rather than following actual patterns in the data.

The PRESS statistic can also be used to compare the fidelity of models constructed from the same data set but with different procedures. In particular, a model with a lower PRESS value is preferable to a model with a higher PRESS value. However, there are two drawbacks in using PRESS to compare models obtained with different procedures. First, PRESS values can be very sensitive to the effects of a limited number of extreme observations. Second, there is no "absolute" standard against which a PRESS value can be compared to indicate whether or not a model is providing a good match to the data. In contrast, R^2 values approach one as the fidelity of the model to the data increases; unfortunately, there is no such limiting value for the PRESS statistic.

The adjusted PRESS value PRS_A (see Eqs. (3.22) – (3.25)) reduces the effects of highly influential observations by using an average leverage value in its definition. The adjusted PRESS value PRS_A is similar in concept to the adjusted R^2 value R_A^2 in that it penalizes a model for the use of an excessive number of degrees of freedom in its construction. However, as with the original statistic PRS, there is no limiting value for PRS_A that provides a standard by which the fidelity of a model to the underlying data can be judged. Although PRS_A can be more useful than the original PRESS statistic in comparing models constructed with different procedures, it is less effective in checking for overfitting because of the reduction in the effects of extreme observations.

The top-down coefficient of concordance (TDCC) provides a way to assess the reproducibility of sensitivity analysis results obtained with individual analysis procedures.[103, 144] In particular, the TDCC provides a measure of the agreement between results obtained with independently generated samples in a manner that emphasizes agreement in the identification of the most important variables and places less emphasis on agreement in the identification of the less important variables. For notational purposes in the definition of the TDCC, suppose (i) nR independently generated samples of the same size involving a vector $\mathbf{x} = [x_1, x_2, \ldots, x_{nX}]$ of independent variables are under consideration, (ii) a sensitivity analysis to rank variable importance is carried out for each sample, and (iii) r_{jk} denotes the rank assigned to variable j in the indicated sensitivity

analysis for sample k, where the most important variable is assigned a rank of 1, the next most important variable is assigned a rank of 2, and so on, with variables of the same importance assigned their average rank (the preceding is the reverse of the ranking procedure described in Sect. 2.2 for rank regression). The TDCC is then defined by

$$C_T = \left\{ \sum_{j=1}^{nX} \left[\sum_{k=1}^{nR} ss\left(r_{jk}\right) \right]^2 - (nR)^2 \, nX \right\} \Bigg/ \left\{ (nR)^2 \left(nX - \sum_{j=1}^{nX} 1/j \right) \right\}, \tag{5.2}$$

where $ss(r_{jk})$ is the Savage score given by

$$ss\left(r_{jk}\right) = \sum_{i=r_{jk}}^{nX} 1/i$$

for variable j in a sample k and average Savage scores are assigned in the event of ties. Use of the Savage scores $ss(r_{jk})$ rather than the ranks r_{jk} in the definition of the TDCC in Eq. (5.2) results in the previously indicated emphasis on agreement on the most important variables and deemphasis on disagreement on the less important variables.

In the examples that follow, variable importance is defined by the order in which variables enter the model under construction, with the first variable entering the model ranked 1, the second variable entering the model ranked 2, and so on. The variables that are not selected for entry into the model are all assigned the same average rank. The preceding ranking is used in the calculation of the TDCC. Values for the TDCC close to one indicate a high level of reproducibility for the sensitivity analysis method under consideration, with a decrease in reproducibility indicated as the value for the TDCC decreases away from one.

The primary emphasis of this presentation is on regression-based procedures for sensitivity analysis. For comparison, a nonregression-based procedure for sensitivity analysis is also included. This procedure is referred to as the SRD/RCC test and is the result of combining a test for nonrandomness in the relationship between an independent and a dependent variable called the squared rank differences (SRD) test with the Spearman rank correlation coefficient (RCC). This test is effective at identifying linear and very general nonlinear patterns in analysis results. However, unlike the regression procedures under consideration, the SRD/RCC test does not involve the development of a model that approximates the relationship between independent and dependent variables.

A brief description of the SRD/RCC test follows. The test is used to assess the relationships between individual elements x_j of $\mathbf{x} = [x_1, x_2, \ldots, x_{nX}]$ and a predicted variable y of interest for a random or LHS and a functional relationship of the form indicated in Eq. (1.8). The SRD component of the test is based on the statistic

$$Q_j = \sum_{i=1}^{nS-1} \left(r_{j,i+1} - r_{ji} \right)^2, \tag{5.3}$$

where r_{ji}, $i = 1, 2, \ldots, nS$, is the rank of y obtained with the sample element in which x_j has rank i and the indicated ranks are defined as described in Sect. 2.2. Under the null hypothesis of no relationship between x_j and y, the quantity

$$S_j = \left\{ Q_j - \left[nS\left(nS^2 - 1\right)/6 \right] \right\} \Big/ \left\{ \sqrt{nS^5}/6 \right\} \tag{5.4}$$

approximately follows a standard normal distribution for $nS > 40$. Thus, a p-value p_{rj} indicative of the strength of the nonlinear relationship between x_j and y can be obtained from Q_j. Specifically, p_{rj} is the probability that a value $\tilde{Q}_j > Q_j$ would occur due to chance if there was no relationship between x_j and y. The RCC component of the test is based on the rank (i.e., Spearman) correlation coefficient

$$R_j = \left\{ \sum_{i=1}^{nS} \left[R\left(x_{ij}\right) - (nS+1)/2 \right] \left[R\left(y_i\right) - (nS+1)/2 \right] \right\} \times \left\{ \sum_{i=1}^{nS} \left[R\left(x_{ij}\right) - (nS+1)/2 \right]^2 \right\}^{-1/2} \times \left\{ \sum_{i=1}^{nS} \left[R\left(y_i\right) - (nS+1)/2 \right]^2 \right\}^{-1/2}, \tag{5.5}$$

where $R(x_{ij})$ and $R(y_i)$ are the ranks associated x_j and y for sample element i. Under the null hypothesis of no rank correlation between x_j and y, the quantity R_j has a known distribution (Table A10, Ref. 145). Thus, a p-value p_{cj} indicative of the strength of the monotonic relationship between x_j and y can be obtained from R_j. The SRD/RCC test is obtained from combining the p-values p_{rj} and p_{cj} to obtain the statistic

$$\chi_4^2 = -2\left[\ln\left(p_{rf}\right) + \ln\left(p_{cj}\right)\right], \tag{5.6}$$

which has a chi-squared distribution with four degrees of freedom. The p-value associated with χ_4^2 constitutes the SRD/RCC test for the strength of the relationship between x_j and y. A detailed description of the SRD/RCC test and the determination of the associated p-value is available elsewhere.[104]

5.1 Example Results: Analytic Test Models

Results obtained with the following four analytic test models are now presented:

$$y_1 = f_1(x_1, x_2) = 5x_1 + (5x_2)^4, \tag{5.7}$$

$$y_2 = f_2(x_1, x_2) = (x_2 + 0.5)^4 / (x_1 + 0.5)^2, \tag{5.8}$$

$$y_3 = f_3(x_1, x_2, \ldots, x_8) = \prod_{j=1}^{8} \left\{ \frac{\left|4x_j - 2\right| + a_j}{1 + a_j} \right\} \tag{5.9}$$

with $[a_1, a_2, \ldots, a_8] = [0, 1, 4.5, 9, 99, 99, 99, 99]$, and

$$\begin{aligned} y_4 &= f_4(x_1, x_2, x_3) \\ &= \sin(2\pi x_1 - \pi) + 7\sin^2(2\pi x_2 - \pi) \\ &\quad + 0.1(2\pi x_3 - \pi)^4 \sin(2\pi x_1 - \pi). \end{aligned} \tag{5.10}$$

The individual models have from 2 to 8 input variables that are assumed to be uniformly and independently distributed on [0, 1]. The functions f_1, f_2, f_3 and f_4 and the associated distributional assumptions for the x_j's correspond to Model 4c, 6b, 7 and 9, respectively, in Ref. 143. The functions f_1, f_3 and f_4 are also considered in Sects. 4 and 5 of Ref. 28.

The example analyses use three replicated random samples of size 100 each from 10 variables (i.e., the x_j) with uniform distributions on [0, 1]. This results in the analysis for each model including from 2 to 8 completely spurious variables. The presence of such variables provides an indication of whether or not the individual regression procedures have a tendency to include spurious variables in model construction. As for the WIPP example (Sect. 5.2), the replicated sampling results in the three samples of the form indicated in Eq. (1.5) and three mappings of the form indicated in Eq. (1.6). As in Sect. 5.2, the individual replicates are referred to as replicates R1, R2 and R3, respectively.

In concept, the example results can be thought of as the outcome of evaluating a model of the form

$$y = f(\mathbf{x}) = \left[f_1(\mathbf{x}), f_2(\mathbf{x}), f_3(\mathbf{x}), f_4(\mathbf{x})\right] \tag{5.11}$$

with $\mathbf{x} = [x_1, x_2, \ldots, x_{10}]$. Such multiple outcomes are usually the case in analyses of real systems (e.g., see the analyses in Refs. 53, 146, 147 from which the examples in Sect. 5.2 are derived). Further, it is also typical of such analyses that individual results are not affected by all of the uncertain variables under consideration.

The analytic models introduced in this section (Sect. 5.1) have an advantage over the real model considered in the following section (Sect. 5.2) in that it is possible to unambiguously determine the contributions of individual analysis inputs to the uncertainty in analysis results. This is not possible with a computationally demanding model of the type considered in Sect. 5.2. In particular, such determinations make comparisons between truth and sensitivity results obtained with the procedures under consideration possible. The method used to determine the actual effects of individual variables is described in the next paragraph.

The R^2 value is the primary quantity used in this presentation to assess the contribution of the uncertainty associated with a group of variables to the uncertainty in an analysis result. In particular, if $\tilde{\mathbf{x}} = [\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_p]$ is a vector of variables taken from the variables $x_1, x_2, \ldots, x_{nX}$ under consideration in a particular analysis (i.e., $\mathbf{x} = [x_1, x_2, \ldots, x_{nX}]$ is the vector of uncertain inputs under consideration), $\hat{f}(\tilde{\mathbf{x}}) = \hat{f}(\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_p)$ is an approximation to the real model $f(\mathbf{x}) = f(x_1, x_2, \ldots, x_{nX})$ estimated with a particular procedure from a mapping $[\mathbf{x}_i, y_i]$, $i = 1, 2, \ldots nS$, from analysis inputs to analysis results, and $\tilde{\mathbf{x}}_i = [\tilde{x}_{i1}, \tilde{x}_{i2}, \ldots, \tilde{x}_{ip}]$ for $i = 1, 2, \ldots, nS$, then

$$R^2 = 1 - \sum_{i=1}^{nS}\left[y_i - \hat{f}(\tilde{\mathbf{x}}_i)\right]^2 \Big/ \sum_{i=1}^{nS}\left[y_i - \bar{y}\right]^2 \tag{5.12}$$

provides an estimate of the fraction of the uncertainty in y that derives from the uncertainty associated with the variables in $\tilde{\mathbf{x}}$.

The contribution of $\tilde{\mathbf{x}}$ to the uncertainty in y that is estimated by R^2 is formally defined by the correlation ratio

$$\begin{aligned}\eta^2 &= 1 - E\left(\left[y - E(y|\tilde{\mathbf{x}})\right]^2\right) \Big/ E\left(\left[y - E(y)\right]^2\right) \\ &= 1 - E\left[Var(y|\tilde{\mathbf{x}})\right] / Var(y) \\ &= Var\left[E(y|\tilde{\mathbf{x}})\right] / Var(y), \end{aligned} \tag{5.13}$$

where (i)

$$E(y) = \int_X f(\mathbf{x}) d_X(\mathbf{x})\, dX$$

$$E(y|\tilde{\mathbf{x}}) = \int_{\tilde{X}^c} f\left(\tilde{\mathbf{x}}^c, \tilde{\mathbf{x}}\right) d_{\tilde{X}^c}\left(\tilde{\mathbf{x}}^c\right) d\tilde{X}^c$$

$$\begin{aligned} E\left(\left[y - E(y)\right]^2\right) &= \int_X \left[f(\mathbf{x}) - E(y)\right]^2 d_X(\mathbf{x})\, dX \\ &= Var(y) \end{aligned}$$

$$\begin{aligned} E\left(\left[y - E(y|\tilde{\mathbf{x}})\right]^2\right) &= \int_X \left[f(\mathbf{x}) - E(y|\tilde{\mathbf{x}})\right]^2 d_X(\mathbf{x})\, dX \\ &= E\left[Var(y|\tilde{x})\right] \end{aligned}$$

$$Var\left[E(y|\tilde{\mathbf{x}})\right] = \int_{\tilde{X}} \left[E(y|\tilde{\mathbf{x}}) - E(y)\right]^2 d_{\tilde{X}}(\tilde{\mathbf{x}})\, d\tilde{X},$$

(ii) $(X, \mathbb{X}, p_X)$, $(\tilde{X}, \tilde{\mathbb{X}}, p_{\tilde{X}})$ and $(\tilde{X}^c, \tilde{\mathbb{X}}^c, p_{\tilde{X}^c})$ are the probability spaces associated with $\mathbf{x}$, $\tilde{\mathbf{x}}$, and $\tilde{\mathbf{x}}^c$, where $\tilde{\mathbf{x}}^c$ contains the elements of $\mathbf{x}$ not contained in $\tilde{\mathbf{x}}$, and (iii) $d_X(\mathbf{x})$, $d_{\tilde{X}}(\tilde{\mathbf{x}})$ and $d_{\tilde{X}^c}(\tilde{\mathbf{x}}^c)$ are the corresponding density functions for $\mathbf{x}$, $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{x}}^c$ (Sect. 8.2, Ref. 148). The quantity η^2 is based on the analysis of variance (ANOVA) decomposition

$$Var(y) = Var\left[E(y|\tilde{\mathbf{x}})\right] + E\left[Var(y|\tilde{\mathbf{x}})\right] \tag{5.14}$$

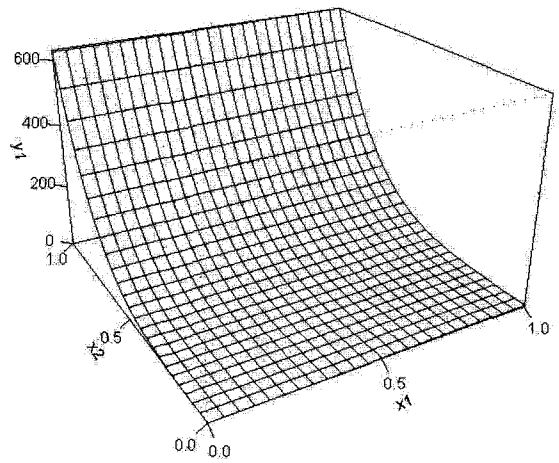
and corresponds to the fraction of the variance of y that derives from the uncertainty associated with the variables that constitute the elements of $\tilde{\mathbf{x}}$.[39-42, 148] For the simple functions considered in this section, η^2 can be calculated and used in comparisons with its corresponding estimate R^2 defined in Eq. (5.12). In some cases, the estimate R^2 can be shown to converge in probability to η^2 as $n \to \infty$.[149, 150]

In the following, η^2 is calculated in a stepwise manner for use in determining variable importance. The most important variable, designated $\tilde{x}_1$, is the element of $\mathbf{x} = [x_1, x_2, \ldots, x_{nX}]$ that gives the largest value for η^2. That is, $\tilde{\mathbf{x}} = [x_1]$, $\tilde{\mathbf{x}} = [x_2]$, ..., $\tilde{\mathbf{x}} = [x_{nX}]$ are considered in the definition of η^2 in Eq. (5.13), and the x_j that gives the highest value for η^2 is deemed to be the most important variable and taken to be $\tilde{x}_1$. The second most important variable, designated $\tilde{x}_2$, is the element of $\mathbf{x} = [x_1, x_2, \ldots, x_{nX}]$ that gives the largest value for η^2 when all possible values for $\tilde{\mathbf{x}} = [\tilde{x}_1, x_j]$, $\tilde{x}_1 \neq x_j$, are considered. The third most important variable, designated $\tilde{x}_3$, is determined in like manner from consideration of vectors of the form $\tilde{\mathbf{x}} = [\tilde{x}_1, \tilde{x}_2, x_j]$, $\tilde{x}_1 \neq x_j$ and $\tilde{x}_2 \neq x_j$, and so on through all nX elements of $\mathbf{x}$.

The individual analytical test models are now considered. For each test problem, R^2, R_A^2, PRS, PRS_A, and the TDCC C_T are calculated for the following methods: LIN_REG, RANK_REG, QUAD_REG, LOESS, PP_REG, RP_REG and GAM. The TDCC is calculated from the three replicated samples of size 100, and the rest of the results are calculated from the pooled sample of size 300. For comparison, the true η^2 values are also presented. The TDCC score comparing the variable rankings obtained with each method with the rankings based on the true model are also given.

5.1.1 Monotonic Relationships: $y_1 = f_1(x_1, x_2)$

The uncertainty in y_1 is mainly driven by x_2 as can be seen in Fig. 14. The results of the various regression methods applied to y_1 are given in Table 6. As indicated by the analysis for the true model in Table 6 (i.e., in the value of η^2 defined in Eq. (5.13)), 99.99% of the uncertainty in y_1 is due to x_2. All the analysis methods agree with the true model in the identification of x_2 as the most important variable. The analysis with LIN_REG has some trouble with a failure to include x_1 in the model and an R^2-value of only 0.76. In contrast, RANK_REG does better as the underlying relationships are monotonic and results in a model containing both x_1 and x_2 and a final R^2 value of 0.98. The analyses with QUAD_REG and LOESS successfully estimate the contribution of x_2 with R^2 values of 0.98 and 1.00, respectively, but fail to identify the effect of x_1. The analyses with PP_REG, GAM and RP_REG all do well in that they include x_1 and x_2 and also give R^2 values for x_1 and x_2 that are equal to the values obtained for the true model. However, PP_REG includes the spurious variables x_5 and x_7. The non-regression based method SRD/RCC also identifies both x_1 and x_2 as important variables. The analysis of y_1 is challenging with respect to the identification of x_1 due to the very small effect associated with this variable.



TR06-JR014-0

Fig. 14. Analytic test model $y_1 = f_1(x_1, x_2) = 5x_1 + (5x_2)^2$

Table 6. Sensitivity Analyses for Analytic Test Model $y_1 = f_1(x_1, x_2)$

Var[a]	R^{2b}	df^c	p-val[d]	PRS[e]
LIN_REG				
x_2	0.7552	1.0	0.0000	1.87E6
$R_A^2 = 0.7544$[f], $PRS_A = 1.86E6$[g]				
$C_T = 1.0000$,[h] C_T w/true = 0.9294[i]				
LOESS				
x_2	0.9999	27.1	0.0000	6.74E2
$R_A^2 = 0.9999$, $PRS_A = 6.79E2$				
$C_T = 1.0000$, C_T w/true = 0.9294				
GAM				
x_2	0.9999	15.0	0.0000	6.80E2
x_1	1.0000	1.0	0.0000	7.55E1
$R_A^2 = 1.0000$, $PRS_A = 4.50E1$				
$C_T = 1.0000$, C_T w/true = 1.0000				

Var[a]	R^{2b}	df^c	p-val[d]	PRS[e]
RANK_REG				
x_2	0.9774	1.0	0.0000	5.19E4
x_1	0.9842	1.0	0.0000	3.66E4
$R_A^2 = 0.9841$, $PRS_A = 3.64E4$				
$C_T = 0.9744$, C_T w/true = 1.0000				
PP_REG				
x_2	0.9999	13.2	0.0000	2.21E3
x_1	1.0000	14.4	0.0000	1.79E3
x_5	1.0000	25.3	0.0000	1.79E3
x_7	1.0000	−18.7	0.0009	1.79E3
$R_A^2 = 1.0000$, $PRS_A = 2.00E0$				
$C_T = 0.9097$, C_T w/true = 0.9453				
SRD/RCC TEST				
x_2	NA	4.0	0.0000	NA
x_1	NA	4.0	0.0101	NA
$R_A^2 =$ NA, $PRS_A =$ NA				
$C_T = 0.9135$, C_T w/true = 1.0000				

Var[a]	R^{2b}	df^c	p-val[d]	PRS[e]
QUAD_REG				
x_2	0.9789	2.0	0.0000	1.63E5
$R_A^2 = 0.9787$, $PRS_A = 1.62E5$				
$C_T = 1.0000$, C_T w/true = 0.9294				
RP_REG				
x_2	0.9999	46.0	0.0000	7.63E2
x_1	1.0000	41.0	0.0000	5.06E1
$R_A^2 = 1.0000$, $PRS_A = 4.01E1$				
$C_T = 1.0000$, C_T w/true = 1.0000				
TRUE MODEL				
x_2	0.9999	NA[j]	NA	NA
x_1	1.0000	NA	NA	NA
$R_A^2 =$ NA, $PRS_A =$ NA				
$C_T =$ NA, C_T w/true = 1.0000				

[a] Variables listed in order of selection with sample of size $nS = 300$.

[b] Cumulative R^2 value with entry of each variable into model (see Eq. (5.13) for True Model and Eq. (5.12) for all other cases).

[c] Incremental degrees of freedom with entry of each variable into model for all cases except SRD/RCC test; df fixed at 4.0 for all variables for SRD/RCC test (see Eq. (5.6)).

[d] p-value for model with addition of each new variable (see Sect. 3.4 and related discussion for individual modeling cases).

[e] PRESS value for model with addition of each new variable (see Eq. (3.19)).

[f] Adjusted R^2 value for final model (see Eq. (5.1)).

[g] Adjusted PRESS value for final model (see Eqs. (3.22) and (3.25)).

[h] TDCC calculated between results for three replicated samples of size $nS = 100$ (see Eq. (5.2)).

[i] TDCC calculated between results obtained for case under consideration with a sample of size $nS = 300$ and the results obtained for the True Model (see Eq. (5.2)).

[j] NA indicates that result is not applicable.

5.1.2 Monotonic Relationships: $y_2 = f_2(x_1, x_2)$

All of the regression methods identify the two important variables (i.e., x_1 and x_2) for y_2 (Fig. 15). As shown in Table 7, the regression methods all indicate that x_2 is the most important variable followed by x_1, which results in a high TDCC with the true model for all the methods. The analysis with LIN_REG underestimates the contribution of x_1. The analysis with RANK_REG overestimates the contribution of x_2; this is likely because rank transformed data instead of actual the y values are being used to compute R^2. The analysis with GAM underestimates x_1's contribution because of its inability to model interactions. The analyses with QUAD_REG, PP_REG, LOESS, and RP_REG give good estimates of the R^2 contribution of x_2 and x_1. However, the analyses with GAM and RP_REG each include one spurious variable, which prevents the TDCC with the true model from being 1.00. The analysis with PP_REG again includes two spurious variables, x_5 and x_9. It is possible that an adjustment to increase the degrees of freedom in a similar fashion to that for RP_REG is required to account for estimating the projections in PP_REG. The analysis with SRD/RCC also identifies x_1 and x_2 as the important variables in the correct order.

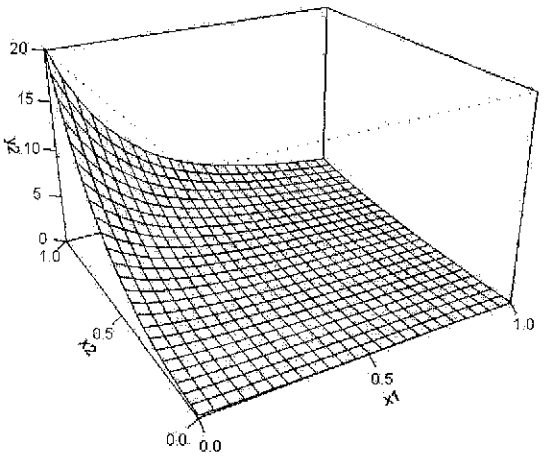


Fig. 15. Analytic test model $y_2 = f_2(x_1, x_2) = (x_2 + 0.5)^4/(x_1 + 0.5)^2$.

Table 7. Sensitivity Analysis for Analytic Test Model $y_2 = f_2(x_1, x_2)$[a]

Var	R^2	*df*	*p*-val	PRS
LIN_REG				
x_2	0.4550	1.0	0.0000	1.11E3
x_1	0.6605	1.0	0.0000	7.03E2
R^2_A = 0.6582, PRS_A = 6.94E2				
C_T = 1.0000, C_T w/true = 1.0000				
LOESS				
x_2	0.6199	27.1	0.0000	1.06E3
x_1	0.9995	45.7	0.0000	2.13E0
R^2_A = 0.9993, PRS_A = 1.88E0				
C_T = 1.0000, C_T w/true = 1.0000				
GAM				
x_2	0.5340	4.0	0.0000	9.79E2
x_1	0.8046	15.0	0.0000	4.81E2
x_3	0.8225	10.0	0.0033	4.64E2
R^2_A = 0.8034, PRS_A = 4.39E2				
C_T = 0.9460, C_T w/true = .9712				

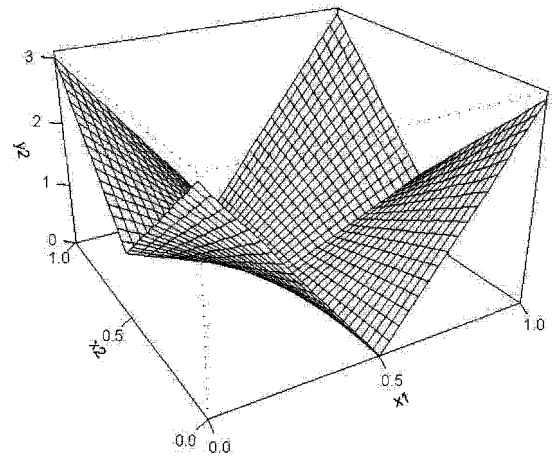
Var	R^2	*df*	*p*-val	PRS
RANK_REG				
x_2	0.8013	1.0	0.0000	4.52E5
x_1	0.9784	1.0	0.0000	4.99E4
R^2_A = 0.9783, PRS_A = 4.95E4				
C_T = 1.0000, C_T w/true = 1.0000				
PP_REG				
x_2	0.5323	3.0	0.0000	9.84E2
x_1	0.9994	32.5	0.0000	5.08E0
x_5	0.9999	4.6	0.0000	4.70E0
x_9	0.9999	11.5	0.0002	4.32E0
R^2_A = 0.9999, PRS_A = 1.78E-1				
C_T = 0.8727, C_T w/true = 0.9511				
SRD/RCC TEST				
x_2	NA	4.0	0.0000	NA
x_1	NA	4.0	0.0000	NA
R^2_A = NA, PRS_A = NA				
C_T = 0.9317 C_T w/true = 1.0000				

Var	R^2	*df*	*p*-val	PRS
QUAD_REG				
x_2	0.5282	2.0	0.0000	9.75E2
x_1	0.9295	3.0	0.0000	1.55E2
R^2_A = 0.9283, PRS_A = 1.47E2				
C_T = 0.9235, C_T w/true = 1.0000				
RP_REG				
x_2	0.5597	16.0	0.0000	1.26E3
x_1	0.9987	56.0	0.0000	1.28E1
x_{10}	0.9991	29.0	0.0000	1.24E1
R^2_A = 0.9986, PRS_A = 4.33E0				
C_T = 0.9712, C_T w/true = 0.9712				
TRUE MODEL				
x_2	0.5196	NA	NA	NA
x_1	1.0000	NA	NA	NA
R^2_A = NA, PRS_A = NA				
C_T = NA, C_T w/true = 1.0000				

[a] Table structure same as described in footnotes to Table 6.

5.1.3 Nonmonotonic Relationships: $y_3 = f_3(x_1, x_2, \ldots, x_8)$

Result y_3 is severely nonlinear in behavior as illustrated in Fig. 16, which contains a plot of y_3 versus the two most important variables, x_1 and x_2. The true model summary in Table 8 indicates that x_1 and x_2 are responsible for most of the uncertainty (95%) in y_3; further, x_3 accounts for about an additional 3% and x_4 an additional 1% of the uncertainty in y_3. The analyses with LIN_REG and RANK_REG demonstrate that these methods are not capable of modeling this data. Both analyses result in models with no variables selected as important. Hence, both analyses result in R^2 values of 0.00. The analysis with PP_REG does a decent job of picking the two most important variables and giving reasonable estimates of their contribution to the uncertainty in y_3 but fails to identify the variables x_3



TR06-JR016-0

Fig. 16. Analytic test model $y_3 = f_3(x_1, x_2, \ldots, x_8)$ (see Eq. (5.9)) with surface averaged over x_3, x_4, $\ldots$, x_8.

Table 8. Sensitivity Analyses for Analytic Test Model $y_3 = f_3(x_1, x_2, \ldots, x_8)$[a]

Var	R^2	*df*	*p*-val	PRS
LIN_REG				
None	0.0000	0.0	NA	1.18E2
$R^2_A = 0.0000$, $PRS_A = 1.18E2$				
$C_T = 0.3333$, C_T w/true = 0.5000				
LOESS				
x_1	0.7047	5.8	0.0000	3.68E1
x_2	0.9503	44.2	0.0000	9.64E0
x_3	0.9819	72.2	0.0000	1.96E1
$R^2_A = 0.9694$, $PRS_A = 6.11E0$				
$C_T = 1.0000$, C_T w/true = 0.9726				
GAM				
x_1	0.7164	10.0	0.0000	3.64E1
x_2	0.9089	15.0	0.0000	1.37E1
x_3	0.9324	4.0	0.0000	1.05E1
x_4	0.9414	4.0	0.0000	9.45E0
$R^2_A = 0.9341$, $PRS_A = 8.76E0$				
$C_T = 0.9730$, C_T w/true = 0.9863				

Var	R^2	*df*	*p*-val	PRS
RANK_REG				
None	0.0000	0.0	NA	2.27E6
$R^2_A = 0.0000$, $PRS_A = 2.27E6$				
$C_T = 0.3333$, C_T w/true = 0.500				
PP_REG				
x_1	0.7177	9.9	0.0000	3.69E1
x_2	0.9123	13.6	0.0000	1.79E1
x_5	0.9329	12.7	0.0000	1.43E1
x_7	0.9493	13.8	0.0000	1.54E1
$R^2_A = 0.9392$, $PRS_A = 8.71E0$				
$C_T = 0.8541$, C_T w/true = 0.9146				
SRD/RCC TEST				
x_1	NA	4.0	0.0000	NA
x_2	NA	4.0	0.0003	NA
$R^2_A = NA$, $PRS_A = NA$				
$C_T = 0.9373$, C_T w/true = 0.9453				

Var	R^2	*df*	*p*-val	PRS
QUAD_REG				
x_1	0.6540	2.0	0.0000	4.19E1
x_2	0.8459	3.0	0.0000	1.92E1
x_3	0.8733	4.0	0.0000	1.63E1
x_4	0.8803	5.0	0.0063	1.61E1
x_6	0.8870	6.0	0.0123	1.60E1
$R^2_A = 0.8789$, $PRS_A = 1.53E1$				
$C_T = 0.9730$, C_T w/true = 0.9866				
RP_REG				
x_1	0.7201	10.0	0.0000	4.06E1
x_2	0.9719	62.0	0.0000	8.17E0
x_3	0.9818	36.0	0.0000	9.23E0
$R^2_A = 0.9715$, $PRS_A = 5.99E0$				
$C_T = 0.9726$, C_T w/true = 0.9726				
TRUE MODEL				
x_1	0.7115	NA	NA	NA
x_2	0.9546	NA	NA	NA
x_3	0.9891	NA	NA	NA
x_4	0.9996	NA	NA	NA
x_5	0.9997	NA	NA	NA
x_6	0.9998	NA	NA	NA
x_7	0.9999	NA	NA	NA
x_8	1.0000	NA	NA	NA
$R^2_A = NA$, $PRS_A = NA$				
$C_T = NA$, C_T w/true = 1.0000				

[a] Table structure same as described in footnotes to Table 6.

and x_4. The analysis with the SRD/RCC test also identifies the two most important variables correctly. The analyses with QUAD_REG and GAM do an even better job by picking out and ordering the four most important variables correctly with reasonably good R^2 estimates, although the R^2 estimates from GAM are closer to the true values than those from QUAD_REG. The analysis with RP_REG and LOESS both do an excellent job of accurately estimating the contribution of the three most important variables x_1 , x_2 and x_3, but fail to identify variable x_4. In this particular example, the standard regression tree defined in Eqs. (3.42) and (3.43) (results not displayed) gives sequential R^2 estimates of 0.72, 0.92, and 0.90 as x_1, x_2 and x_3 enter the model. For RP_REG, these values are 0.72, 0.97, and 0.98, which are closer to the true values. In addition, the RP_REG procedure provided superior results to the standard regression tree approach in most of the other examples considered in this presentation.

5.1.4 Nonmonotonic Relationship: $y_4 = f_4(x_1, x_2, x_3)$

Result y_4 is the most difficult outcome to analyze for all of the regression methods. As shown in Table 9, the linear methods (i.e., LIN_REG, RANK_REG and QUAD_REG) have an R^2 below 0.2 The analysis with QUAD_REG fails because the sinusoidal relationship that can be seen in Fig. 17 departs too much from a quadratic. The oscillating behavior of y_4 is also difficult for LIN_REG and RANK_REG to model. The analysis with PP_REG, which often overfits the data, this time identified only x_2 and x_3 for inclusion in the model; the reasons for this are unclear. The analysis with GAM has an R^2 value of 0.79 and identifies the two most important variables correctly. However, GAM also includes the spurious variable x_4. The analysis with LOESS was the most successful on this example. It has an R^2 value of 0.95 and identifies the three important inputs correctly. The analysis with RP_REG has an R^2 value of 0.90, and also identifies all three important inputs correctly. Both LOESS and RP_REG give reasonable estimates of the R^2 contribution of each variable as well. The analysis with the SRD/RCC test also identifies x_2 and x_1 as the two most important variables in the correct order but fails to identify x_3 and includes x_6 spuriously.

5.2 Example Results: Two-Phase Fluid Flow

The regression-based sensitivity analysis procedures are now illustrated with results from an uncertainty/sensitivity analysis of a model for two phase fluid flow[53, 146, 147] carried out as part of the 1996 compliance certification application (CCA) for the Waste Isolation Pilot Plant (WIPP).[56] The CCA involved $nX = 57$ uncertain variables,[140] with 31 of these variables used in the two-phase fluid flow analysis considered in this section (Table 10). The two-phase fluid flow analysis considered six different scenarios (i.e., modeling cases; see Table 6, Ref. 140) and generated several hundred time-dependent analysis results for each modeling case (e.g., see Table 1, Ref. 146, for a

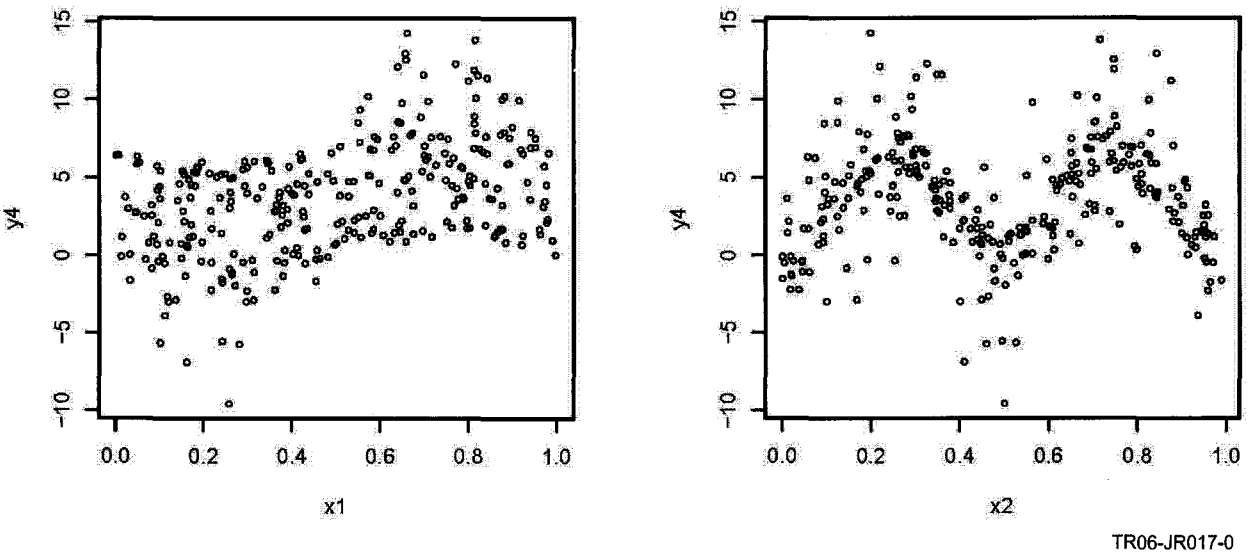


Fig. 17. Scatterplots for x_1 and x_2 for analytic test model $y_4 = f_4(x_1, x_2, x_3)$ (see Eq. (5.10)).

Table 9. Sensitivity Analyses for Analytic Test Model $y_4 = f_4(x_1, x_2, x_3)$[a]

Var	R^2	*df*	*p*-val	PRS
LIN_REG				
x_1	0.1756	1.0	0.0000	3.26E3
R_A^2 = 0.1728, PRS_A = 3.26E3				
C_T = 1.0000, C_T w/true = 0.6949				
LOESS				
x_2	0.5030	18.1	0.0000	2.21E3
x_1	0.7982	31.9	0.0000	1.13E3
x_3	0.9519	72.2	0.0000	1.39E3
R_A^2 = 0.9187, PRS_A = 5.40E2				
C_T = 1.0000, C_T w/true = 1.0000				
GAM				
x_2	0.4736	10.0	0.0000	2.21E3
x_1	0.7753	7.0	0.0000	9.81E2
x_4	0.7920	10.0	0.0188	9.72E2
R_A^2 = 0.7714, PRS_A = 9.87E2				
C_T = 0.9744, C_T w/true = 0.9360				

Var	R^2	*df*	*p*-val	PRS
RANK_REG				
x_1	0.1599	1.0	0.0000	1.92E6
R_A^2 = 0.1571, PRS_A = 1.92E6				
C_T = 0.9373, C_T w/true = 0.6949				
PP_REG				
x_2	0.4669	7.9	0.0000	2.21E3
x_3	0.5483	21.0	0.0012	2.68E3
R_A^2 = 0.4999, PRS_A = 2.19E3				
C_T = 0.7675, C_T w/true = 0.9171				
SRD/RCC TEST				
x_1	NA	4.0	0.0000	NA
x_2	NA	4.0	0.0000	NA
x_6	NA	4.0	0.0017	NA
R_A^2 = NA, R_A^2 = NA				
C_T = 0.9207 C_T w/true = 0.8586				

Var	R^2	*df*	*p*-val	PRS
QUAD_REG				
x_1	0.1768	2.0	0.0000	3.28E3
R_A^2 = 0.1713, PRS_A = 3.28E3				
C_T = 0.9373, C_T w/true = 0.6949				
RP_REG				
x_2	0.5114	19.0	0.0000	2.53E3
x_1	0.8180	48.0	0.0000	1.57E3
x_3	0.9033	41.0	0.0000	1.79E3
R_A^2 = 0.8486, PRS_A = 1.00E3				
C_T = 1.0000, C_T w/true = 1.0000				
TRUE MODEL				
x_2	0.4463	NA	NA	NA
x_1	0.7593	NA	NA	NA
x_3	1.0000	NA	NA	NA
R_A^2 = NA, PRS_A = NA				
C_T = NA, C_T w/true = 1.0000				

[a] Table structure same as described in footnotes to Table 6.

partial listing of these results). A small subset of these results is considered in this presentation. In particular, the modeling case corresponding to a drilling intrusion at 1000 yr that penetrates both the repository and an underlying region of pressurized brine is used as an example (i.e., an E1 intrusion at 1000 yr in the terminology of the 1996 WIPP CCA; see Table 6, Ref. 140), and three time-dependent analysis results are used for illustration (Table 11).

The example analysis used Latin hypercube sampling to generate a mapping between analysis inputs and analysis results of the form indicated in Eqs. (1.5) and (1.6). In particular, three replicated (i.e., independently generated) Latin hypercube samples[27, 38] of size nS = 100 were used. Thus, the analysis actually had three samples of the form indicated in Eq. (1.5) and three mappings of the form indicated in Eq. (1.6). This replication was performed to provide a way to test the stability (i.e., reproducibility) of analysis results (Sect. 7, Ref. 140). For convenience, the individual replicates are referred to as replicate R1, R2 and R3, respectively. The 100 time-dependent values for the variables indicated in Table 11 (i.e., *BRNREPTC*, *REP_SATB*, *WAS_PRES*) that result for replicate R1 are shown in Fig. 18.

The three time-dependent results indicated in Table 11 are analyzed at 1000 yr and 10,000 yr. The results at 1000 yr are for undisturbed conditions immediately prior to the drilling intrusion at 1000 yr. Because of this timing, the 1000 yr results are unaffected by the drilling intrusion and thus are very different from the 10,000 yr results.

5.2.1 Cumulative Brine Flow at 1000 yr (*BRNREPTC.1K*)

All of the analysis methods perform well for *BRNREPTC.1K* (Table 12), with all methods identifying *HALPOR* as the most important variable and all the regression-based methods identifying *HALPOR*, *WMICDFLG*, *ANHPRM* and *HALPRM* as the four most important variables. Specifically, all regression-based methods indicate that *HALPOR* accounts for approximately 96% of the uncertainty in *BRNREPTC.1K* and that the four most important variables collectively account for between 98% and 99% of the uncertainty in *BRNREPTC.1K*. The examination of scatterplots shows the dominant effect of *HALPOR* and also the more subtle effects associated with *WMICDFLG*, *ANHPRM* and *HALPRM* (Fig. 19). The similarity of the results obtained with LIN_REG and RANK_REG indicates that the relationships between *BRNREPTC.1K* and the sampled variables affecting *BRNREPTC.1K* are effectively linear. In this situation, all of the regression-based methods are producing models of

Table 10. Independent (i.e., sampled) Variables Considered in Example Sensitivity Analyses for Two-Phase Fluid Flow (Source: Table 1, Ref. 103, and Table 1, Ref. 140)

ANHBCEXP—Brooks-Corey pore distribution parameter for anhydrite (dimensionless). Distribution: Student's with 5 degrees of freedom. Range: 0.491 to 0.842. Mean, Median: 0.644, 0.644.

ANHBCVGP—Pointer variable for selection of relative permeability model for use in anhydrite. Distribution: Discrete with 60% 0, 40% 1. Value of 0 implies Brooks-Corey model; value of 1 implies van Genuchten-Parker model.

ANHCOMP—Bulk compressibility of anhydrite (Pa^{-1}). Distribution: Student's with 3 degrees of freedom. Range: 1.09×10^{-11} to 2.75×10^{-10} Pa^{-1}. Mean, Median: 8.26×10^{-11} Pa^{-1}, 8.26×10^{-11} Pa^{-1}. Correlation: –0.99 rank correlation[151, 152] with *ANHPRM*.

ANHPRM—Logarithm of anhydrite permeability (m^2). Distribution: Student's with 5 degrees of freedom. Range: –21.0 to –17.1 (i.e., permeability range is 1×10^{-21} to $1 \times 10^{-17.1}$ m^2). Mean, Median: –18.9, –18.9. Correlation : –0.99 rank correlation with *ANHCOMP*.

ANRBRSAT—Residual brine saturation in anhydrite (dimensionless). Distribution: Student's with 5 degrees of freedom. Range: 7.85×10^{-3} to 1.74×10^{-1}. Mean, Median: 8.36×10^{-2}, 8.36×10^{-2}.

ANRGSSAT—Residual gas saturation in anhydrite (dimensionless). Distribution: Student's with 5 degrees of freedom. Range: 1.39×10^{-2} to 1.79×10^{-1}. Mean, median: 7.71×10^{-2}, 7.71×10^{-2}.

BHPRM—Logarithm of borehole permeability (m^2). Distribution: Uniform. Range: –14 to –11 (i.e., permeability range is 1×10^{-14} to 1×10^{-11} m^2). Mean, median: –12.5, –12.5.

BPCOMP—Logarithm of bulk compressibility of brine pocket (Pa^{-1}). Distribution: Triangular. Range: –11.3 to –8.00 (i.e., bulk compressibility range is $1 \times 10^{-11.3}$ to 1×10^{-8} Pa^{-1}). Mean, mode: –9.80, –10.0. Correlation: –0.75 rank correlation with *BPPRM*.

BPINTPRS— Initial pressure in brine pocket (Pa). Distribution: Triangular. Range: 1.11×10^7 to 1.70×10^7 Pa. Mean, mode: 1.36×10^7 Pa, 1.27×10^7 Pa.

BPPRM— Logarithm of intrinsic brine pocket permeability (m^2). Distribution: Triangular. Range: –14.7 to –9.80 (i.e., permeability range is $1 \times 10^{-14.7}$ to $1 \times 10^{-9.80}$ m^2). Mean, mode: –12.1, –11.8. Correlation: –0.75 rank correlation with *BPCOMP*.

BPVOL— Pointer variable for selection of brine pocket volume. Distribution: Discrete, with integer values 1, 2, ..., 32 equally likely.

HALCOMP—Bulk compressibility of halite (Pa^{-1}). Distribution: Uniform. Range: 2.94×10^{-12} to 1.92×10^{-10} PA^{-1}. Mean, median: 9.75×10^{-11} Pa^{-1}, 9.75×10^{-11} Pa^{-1}. Correlation: –0.99 rank correlation with *HALPRM*.

HALPOR—Halite porosity (dimensionless). Distribution: Piecewise uniform. Range: 1.0×10^{-3} to 3×10^{-2}. Mean, median: 1.28×10^{-2}, 1.00×10^{-2}.

HALPRM—Logarithm of halite permeability (m^2). Distribution: Uniform. Range: –24 to –21 (i.e., permeability range is 1×10^{-24} to 1×10^{-21} m^2). Mean, median: –22.5, –22.5. Correlation: –0.99 rank correlation with *HALCOMP*.

Table 10. Independent (i.e., sampled) Variables Considered in Example Sensitivity Analyses for Two-Phase Fluid Flow (Source: Table 1, Ref. 103, and Table 1, Ref. 140) (Continued)

SALPRES—Initial brine pressure, without the repository being present, at a reference point located in the center of the combined shafts at the elevation of the midpoint of Marker Bed (MB) 139 (Pa). Distribution: Uniform. Range: 1.104×10^7 to 1.389×10^7 Pa. Mean, median: 1.247×10^7 Pa, 1.247×10^7 Pa.

SHBCEXP—Brooks-Corey pore distribution parameter for shaft (dimensionless). Distribution: Piecewise uniform. Range: 0.11 to 8.10. Mean, median: 2.52, 0.94.

SHPRMASP—Logarithm of permeability (m^2) of asphalt component of shaft seal (m^2). Distribution: Triangular. Range: –21 to –18 (i.e., permeability range is 1×10^{-21} to 1×10^{-18} m^2). Mean, mode: –19.7, –20.0.

SHPRMCLY—Logarithm of permeability (m^2) for clay components of shaft. Distribution: Triangular. Range: –21 to –17.3 (i.e., permeability range is 1×10^{-21} to $1 \times 10^{-17.3}$ m^2). Mean, mode: –18.9, –18.3.

SHPRMCON—Same as *SHPRMASP* but for concrete component of shaft seal for 0 to 400 yr. Distribution: Triangular. Range: –17.0 to –14.0 (i.e., permeability range is 1×10^{-17} to 1×10^{-14} m^2). Mean, mode: –15.3, –15.0.

SHPRMDRZ—Logarithm of permeability (m^2) of DRZ surrounding shaft. Distribution: Triangular. Range: –17.0 to –14.0 (i.e., permeability range is 1×10^{-17} to 1×10^{-14} m^2). Mean, mode: –15.3, –15.0.

SHPRMHAL—Pointer variable (dimensionless) used to select permeability in crushed salt component of shaft seal at different times. Distribution: Uniform. Range: 0 to 1. Mean, mode: 0.5, 0.5. A distribution of permeability (m^2) in the crushed salt component of the shaft seal is defined for each of the following time intervals: [0, 10 yr], [10, 25 yr], [25, 50 yr], [50, 100 yr], [100, 200 yr], [200, 10000 yr]. *SHPRMHAL* is used to select a permeability value from the cumulative distribution function for permeability for each of the preceding time intervals with result that a rank correlation of 1 exists between the permeabilities used for the individual time intervals.

SHRBRSAT—Residual brine saturation in shaft (dimensionless). Distribution: Uniform. Range: 0 to 0.4. Mean, median: 0.2, 0.2.

SHRGSSAT—Residual gas saturation in shaft (dimensionless). Distribution: Uniform. Range: 0 to 0.4. Mean, median: 0.2, 0.2.

WASTWICK—Increase in brine saturation of waste due to capillary forces (dimensionless). Distribution: Uniform. Range: 0 to 1. Mean, median: 0.5, 0.5.

WFBETCEL—Scale factor used in definition of stoichiometric coefficient for microbial gas generation (dimensionless). Distribution: Uniform. Range: 0 to 1. Mean, median: 0.5, 0.5.

WGRCOR—Corrosion rate for steel under inundated conditions in the absence of CO_2 (m/s). Distribution: Uniform. Range: 0 to 1.58×10^{-14} m/s. Mean, median: 7.94×10^{-15} m/s, 7.94×10^{-15} m/s.

WGRMICH—Microbial degradation rate for cellulose under humid conditions (mol/kg•s). Distribution: Uniform. Range: 0 to 1.27×10^{-9} mol/kg•s. Mean, median: 6.34×10^{-10} mol/kg•s, 6.34×10^{-10} mol/kg•s.

WGRMICI—Microbial degradation rate for cellulose under inundated conditions (mol/kg•s). Distribution: Uniform. Range: 3.17×10^{-10} to 9.51×10^{-9} mol/kg•s. Mean, median: 4.92×10^{-9} mol/kg•s, 4.92×10^{-9} mol/kg•s.

Table 10. Independent (i.e., sampled) Variables Considered in Example Sensitivity Analyses for Two-Phase Fluid Flow (Source: Table 1, Ref. 103, and Table 1, Ref. 140) (Continued)

WMICDFLG—Pointer variable for microbial degradation of cellulose. Distribution: Discrete, with 50% 0, 25% 1, 25% 2. *WMICDFLG* = 0, 1, 2 implies no microbial degradation of cellulose, microbial degradation of only cellulose, microbial degradation of cellulose, plastic, and rubber.

WRBRNSAT—Residual brine saturation in waste (dimensionless). Distribution: Uniform. Range: 0 to 0.552. Mean, median: 0.276, 0.276.

WRGSSAT—Residual gas saturation in waste (dimensionless). Distribution: Uniform. Range: 0 to 0.15. Mean, median: 0.075, 0.075.

Table 11. Time-Dependent Two-Phase Fluid Flow Results for a Drilling Intrusion at 1000 yr that Penetrates the Repository and an Underlying Region of Pressurized Brine (i.e., an E1 intrusion at 1000 yr) Used to Illustrate Sensitivity Analysis Results

BRNREPTC: Cumulative brine flow (m^3) into repository (i.e., into region corresponding to Cells 596 – 625, 638 – 640 in Fig. 3, Ref. 53).

REP_SATB: Average brine saturation in waste panels not penetrated by the drilling intrusion (i.e., in the region corresponding to Cells 617 – 625 in Fig. 3, Ref. 53).

WAS_PRES: Pressure (Pa) in waste panel penetrated by the drilling intrusion (i.e., in the region corresponding to Cells 596 – 616 in Fig. 3, Ref. 53).

Note 1: Effects of the drilling intrusion are only manifested for times greater than 1000 yr. Conditions for times less than or equal to 1000 yr are the same as for undisturbed conditions (i.e., E0 conditions in the terminology of the 1996 WIPP CCA).

Note 2: Suffixes of *.1K* and *.10K* are appended to variable names to indicate results at 1000 and 10,000 years, respectively.

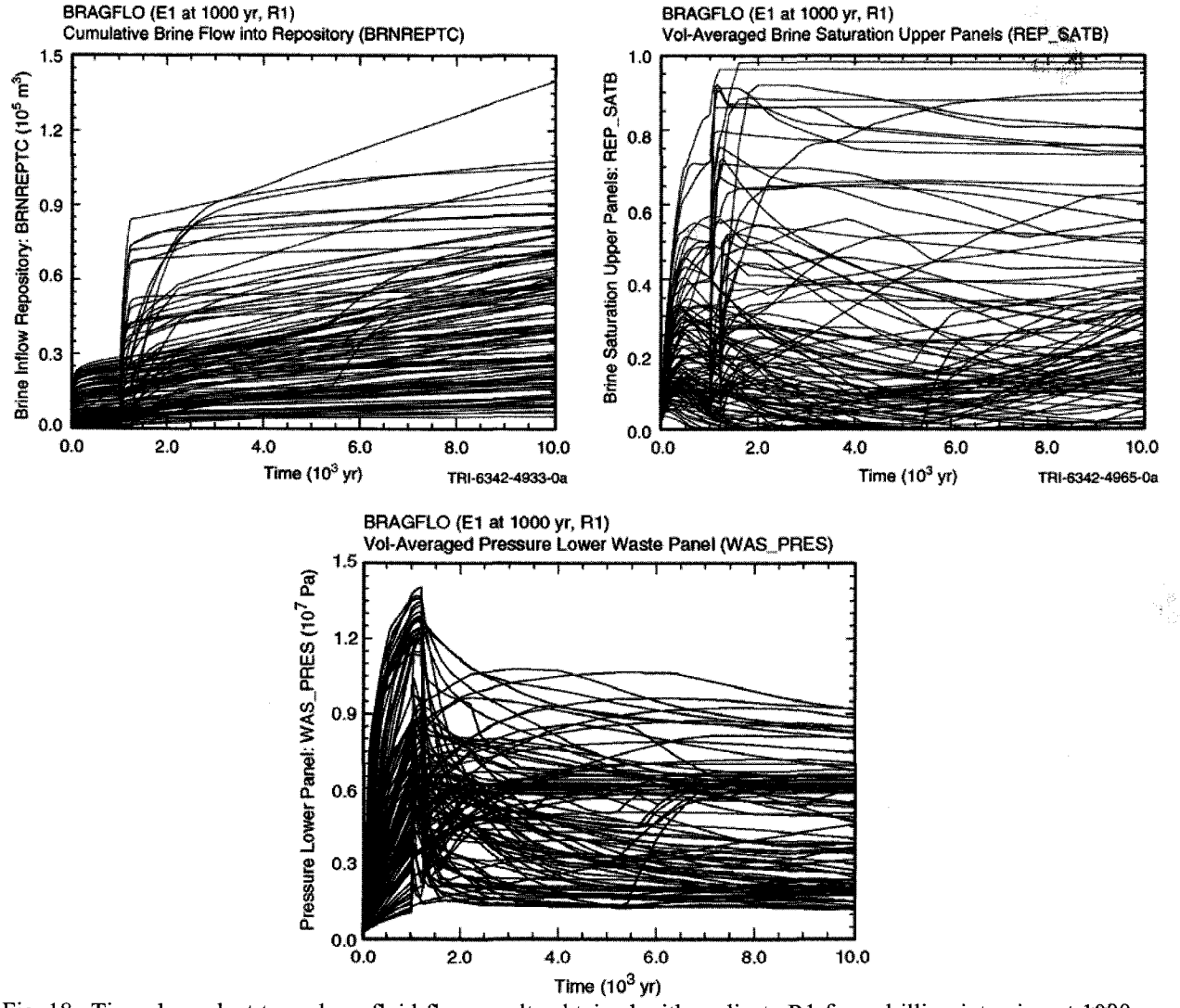


Fig. 18. Time-dependent two-phase fluid flow results obtained with replicate R1 for a drilling intrusion at 1000 yr that penetrates the repository and an underlying region of pressurized brine (i.e., an E1 intrusion at 1000 yr).

similar predictive capability. However, as suggested by the incremental changes in the number of degrees of freedom, the nonparametric regression procedures (i.e., LOESS, PP_REG, RP_REG, GAM) are producing models that are more complicated than those produced by the parametric regression procedures (i.e., LIN_REG, RANK_REG, QUAD_REG). For PP_REG and RP_REG, the negative value for the incremental degrees of freedom associated with the addition of *HALPRM* and *WASTWICK* to the respective models indicates a reduction in complexity for the constructed model with the addition of this variable.

It is likely that some of the variables added near the ends of the stepwise procedures for the individual regression procedures are spurious. For example, the variable *BHPRM* added at the end of the analysis with PP_REG is obviously spurious because *BHPRM* does not affect *BRNREPTC.1K*. With approximately 30 uncertain variables under consideration and use of an α-value cutoff of 0.02, the selection of spurious variables near the end of a stepwise analysis is always a possibility.

Table 12. Sensitivity Analyses for Cumulative Brine Flow at 1000 yr into Repository (*BRNREPTC.1K*) for Undisturbed Conditions[a]

Var	R^2	*df*	*p*-value	*PRS*
LIN_REG				
HALPOR	0.9607	1.0	0.0000	8.09E8
WMICDFLG	0.9704	1.0	0.0000	6.12E8
ANHPRM	0.9785	1.0	0.0000	4.57E8
HALPRM	0.9801	1.0	0.0000	4.26E8
WRBRNSAT	0.9813	1.0	0.0000	4.02E8
WASTWICK	0.9825	1.0	0.0000	3.80E8
SALPRES	0.9831	1.0	0.0019	3.70E8
WGRCOR	0.9836	1.0	0.0032	3.62E8
R_A^2 = 0.9831, PRS_A = 3.55E8, C_T = 0.9221				
QUAD_REG				
HALPOR	0.9657	2.0	0.0000	7.09E8
ANHPRM	0.9813	3.0	0.0000	4.15E8
WMICDFLG	0.9897	4.0	0.0000	2.42E8
HALPRM	0.9916	5.0	0.0000	2.08E8
WGRCOR	0.9934	6.0	0.0000	1.75E8
WASTWICK	0.9944	7.0	0.0000	1.56E8
SALPRES	0.9955	8.0	0.0000	1.40E8
WRBRNSAT	0.9964	9.0	0.0000	1.17E8
SHPRMDRZ	0.9968	10.0	0.0046	1.14E8
WRGSSAT	0.9971	11.0	0.0041	1.07E8
SHPRMCLY	0.9975	12.0	0.0025	1.04E8
R_A^2 = 0.9966, PRS_A = 9.32E7, C_T = 0.9211				
PP_REG				
HALPOR	0.9659	2.9	0.0000	7.10E8
ANHPRM	0.9868	12.4	0.0000	3.47E8
WMICDFLG	0.9933	1.9	0.0000	1.97E8
SHPRMCON	0.9962	46.4	0.0000	3.07E8
HALPRM	0.9962	-42.1	0.0000	1.84E8
SALPRES	0.9984	51.3	0.0000	1.97E8
SHPRMASP	0.9989	32.0	0.0000	2.58E8
BPVOL	0.9992	3.3	0.0000	1.79E8
WRBRNSAT	0.9997	35.0	0.0000	2.68E8
R_A^2 = 0.9995, PRS_A = 1.92E7, C_T = 0.8103				
SRD/RCC TEST				
HALPOR	NA	4.0	0.0000	NA
R_A^2 = NA, PRS_A = NA, C_T = 1.0000				

Var	R^2	*df*	*p*-value	*PRS*
RANK_REG				
HALPOR	0.9550	1.0	0.0000	1.03E5
WMICDFLG	0.9658	1.0	0.0000	7.85E4
ANHPRM	0.9726	1.0	0.0000	6.37E4
HALPRM	0.9749	1.0	0.0000	5.87E4
WRBRNSAT	0.9764	1.0	0.0000	5.57E4
WASTWICK	0.9778	1.0	0.0000	5.29E4
SALPRES	0.9791	1.0	0.0000	5.00E4
WGRCOR	0.9800	1.0	0.0003	4.82E4
R_A^2 = 0.9795, PRS_A = 4.78E4, C_T = 0.9224				
LOESS				
HALPOR	0.9657	2.3	0.0000	7.12E8
ANHPRM	0.9878	36.0	0.0000	3.75E8
WMICDFLG	0.9945	27.7	0.0000	2.49E8
WASTWICK	0.9963	24.6	0.0000	1.57E8
HALPRM	0.9979	44.2	0.0000	2.48E8
R_A^2 = 0.9961, PRS_A = 1.45E8, C_T = 0.9203				
RP_REG				
HALPOR	0.9684	7.0	0.0000	7.26E8
ANHPRM	0.9870	15.0	0.0000	4.47E8
WMICDFLG	0.9924	2.0	0.0000	3.12E8
BPCOMP	0.9954	34.0	0.0000	3.80E8
ANRGSSAT	0.9960	13.0	0.0060	5.61E8
SALPRES	0.9965	0.0	0.0000	6.51E8
WASTWICK	0.9965	-19.0	0.0000	2.71E8
HALPRM	0.9974	7.0	0.0000	1.58E8
WGRCOR	0.9980	7.0	0.0000	9.18E7
SHBCEXP	0.9984	28.0	0.0119	1.54E8
SHPRMCON	0.9985	8.0	0.0085	7.69E7
R_A^2 = 0.9978, PRS_A = 6.92E7, C_T = 0.9223				
GAM				
HALPOR	0.9661	4.0	0.0000	7.10E8
ANHPRM	0.9875	15.0	0.0000	3.10E8
WMICDFLG	0.9932	2.0	0.0000	1.75E8
HALPRM	0.9944	1.0	0.0000	1.47E8
WASTWICK	0.9951	2.0	0.0000	1.31E8
SALPRES	0.9956	2.0	0.0000	1.21E8
WGRCOR	0.9961	1.0	0.0000	1.10E8
WRBRNSAT	0.9964	1.0	0.0000	1.02E8
R_A^2 = 0.9961, PRS_A = 8.90E7, C_T = 0.9593				

[a] Table structure same as described in footnotes to Table 6.

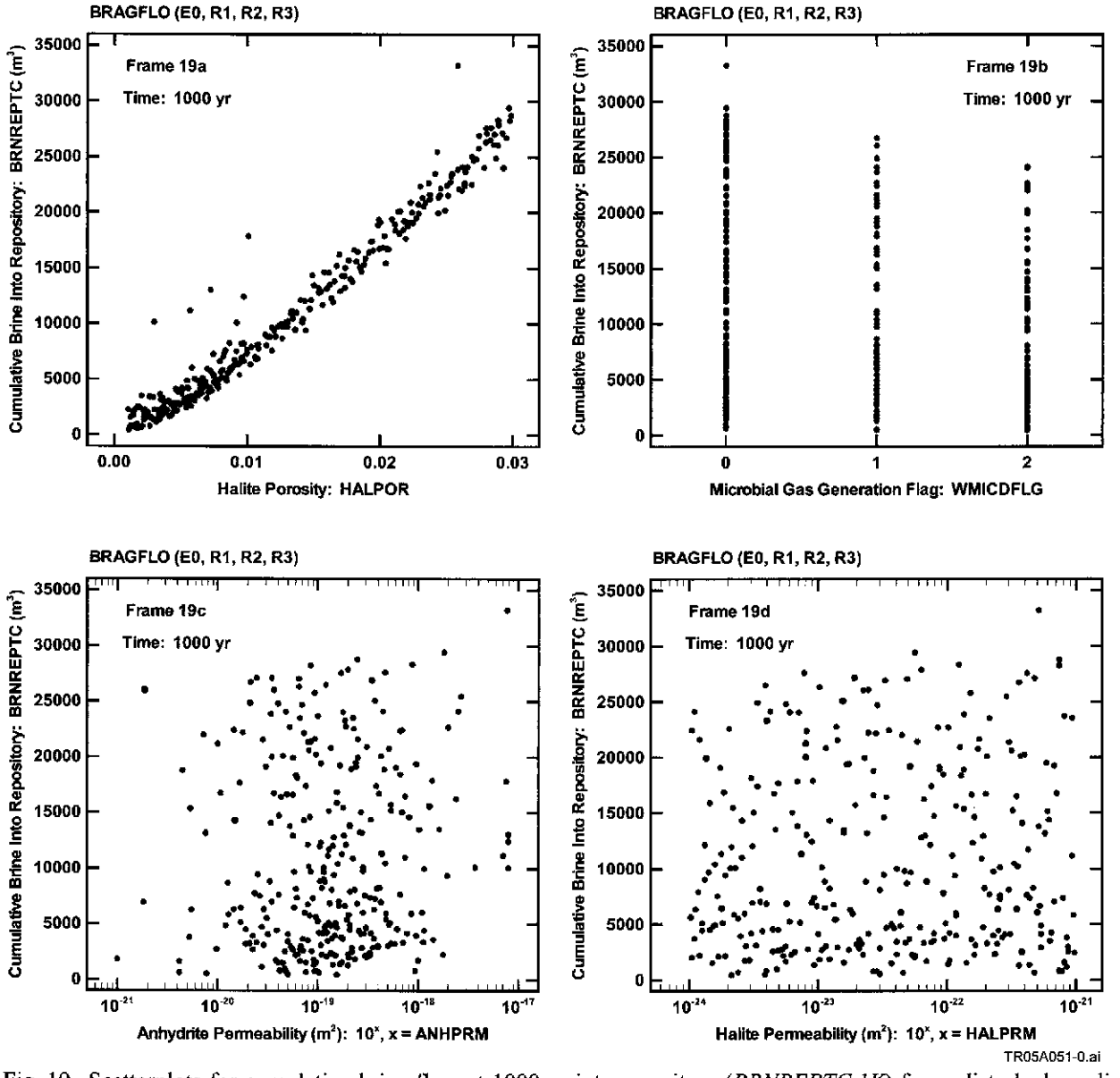


Fig. 19. Scatterplots for cumulative brine flow at 1000 yr into repository (*BRNREPTC.1K*) for undisturbed conditions.

The decreasing PRESS values for LIN_REG, RANK_REG, QUAD_REG and GAM indicate that the data is not being overfitted. However, there are some jumps in the PRESS values for LOESS, PP_REG and RP_REG as variables with small indicated effects are added to the regression model, which suggests that some overfitting of the data is taking place. Further, with the exception of PP_REG, the values for the TDCC (i.e., C_T) range from 0.92 to 0.96 for the individual regression procedures and indicate a high degree of reproducibility for results obtained with the three replicated LHSs of size 100. The PP_REG procedure has a lower level of reproducibility as indicated by a TDCC value of 0.81.

The SRD/RCC test identifies the dominant effect associated with *HALPOR* but misses the smaller effects associated with *WMICDFLG*, *ANHPRM* and *HALPRM*.

5.2.2 Cumulative Brine Flow at 10,000 yr (*BRNREPTC.10K*)

For *BRNREPTC.10K*, the methods generally agree on the three most important variables (i.e., *BHPRM*, *BPCOMP*, and *HALPOR*, with *HALPOR* selected fourth with QUAD_REG and RP_REG) but there is some inconsistency with respect to the fourth most important variable (Table 13). The methods also do not agree on

the total amount of uncertainty that can be explained. As shown by the scatterplots of the four most important input variables (Fig. 20), there is a definite monotonic relationship between these variables and *BRNREPTC.10K*. The linear methods LIN_REG and RANK_REG each have a relatively low final R^2 value of 0.71. In addition, GAM has a final R^2 value of 0.79, which suggests that in addition to nonlinearity there is also interaction between input variables. The remaining methods, QUAD_REG, LOESS, PP_REG, and RP_REG, all have R^2 values of about 0.8 or higher after inclusion of the fifth most important variable. After that, there is considerable disagreement on which inputs make additional contributions to the uncertainty in *BRNREPTC.10K*. Thus, the only safe inference that can be drawn from the collective analyses is that these first five inputs (i.e., *BHPRM, BPCOMP, HALPOR, WMICDFLG,* and *ANHPRM*) are giving rise to about 80 – 90% of the uncertainty in *BRNREPTC.10K*. For PP_REG and RP_REG, increases in PRESS values near the end of the stepwise process indicate that overfitting of the data could be taking place as variables with small effects are added to the models.

The SRD/RCC test agrees with the regression methods on four of the first five variables but also includes *BPPRM* as the fifth most important input, which is not in any of the other models except RP_REG. This difference probably results from the –0.75 rank correlation between *BPPRM* and *BPCOMP* (see Table 10). All the regression-based methods select *BPCOMP* as the second variable in the stepwise procedure. Because of the indicated correlation, the resultant regression model includes effects that derive from both *BPCOMP* and *BPPRM*, which reduces the likelihood that *BPPRM* will be selected at a later step. In contrast, the SRD/RCC test examines the effects of variables individually, which makes it more effective in identifying the effects of correlated variables than is the case for stepwise regression procedures.

The PP_REG procedure has a very low reproducibility with a TDCC value of 0.40. The LOESS and RP_REG procedures also have relatively low TDCC values of 0.72 and 0.71, respectively. In contrast, the TDCC values for the other methods range between 0.87 and 0.96, which indicates fairly high levels of reproducibility.

5.2.3 Brine Saturation at 1000 yr (*REP_SATB.1K*)

The analysis for *REP_SATB.1K* (Table 14) produce results very similar to those *BRNREPTC.1K* (Table 12), where the linear methods performed well. All the methods agree on the four most important input variables (i.e., *HALPOR, WGRCOR, WMICDFLG, WASTWICK*). All the regression-based methods indicate that *HALPOR* accounts for about 60% of the uncertainty in *REP_SATB.1K*. They also indicate that *WGRCOR* is responsible for an additional 20% of the uncertainty in *REP_SATB.1K*. The dominant effects of *HALPOR* and *WGRCOR* are clearly evident in the corresponding scatterplots in Fig. 21. In addition, *WMICDFLG* and *WASTWICK* account for about another 10% and 5%, respectively, of the uncertainty in *REP_SATB.1K*. The SRD/RCC test also identifies these four variables in the same order as the regression-based methods. All methods also have high TDCC values, indicating a high level of reproducibility. However, PP_REG and RP_REG have jumps in PRESS values at the end of the stepwise process as variables with small effects are added to the models, which indicates that an overfitting of the data could be taking place.

5.2.4 Brine Saturation at 10,000 yr (*REP_SATB.10K*)

All methods identify *WGRCOR* and *BHPRM* as the two most important contributors to the uncertainty in *REP_SATB.10K* (Table 15). The linear methods (i.e., LIN_REG and RANK_REG) underperform the other regression methods in that they appear to underestimate the contributions of *WGRCOR* and *BHPRM* to the uncertainty in *REP_SATB.10K* (i.e., compare R^2 values for the different regression procedures in Table 15). Specifically, LIN_REG and RANK_REG indicate that *WGRCOR* accounts for about 22 – 28% of the uncertainty in *REP_SATB.10K* while the other methods indicate that *WGRCOR* contributes in the range of 42 – 48% of the uncertainty in *REP_SATB.10K*. From this, it is then clear that *BHPRM* accounts for another 15 – 20% of the uncertainty above and beyond that accounted for by *WGRCOR*.

After *WGRCOR* and *BHPRM*, the individual analyses generally indicate that additional contributions to the uncertainty in *REP_SATB.10K* are made primarily by *HALPOR* (~10 – 15%) and *BPCOMP* (~5%), with smaller contributions from *WMICDFLG, ANHPRM* and *WASTWICK* (~ 2% each). As shown by the scatterplots in Fig. 22, *WGRCOR* and *BHPRM* have visually discernable effects on *REP_SATB.10K*, while the less important contributors to the uncertainty in *REP_SATB.10K* have effects that are identifiable by the analysis procedures but are less apparent in a visual examination.

Table 13. Sensitivity Analyses for Cumulative Brine Flow at 10,000 yr into Repository (*BRNREPTC.10K*) for an E1 Intrusion at 1000 yr[a]

Var	R^2	*df*	*p*-value	*PRS*
LIN_REG				
BHPRM	0.2868	1.0	0.0000	1.64E11
BPCOMP	0.4590	1.0	0.0000	1.26E11
HALPOR	0.5645	1.0	0.0000	1.02E11
WMICDFLG	0.6267	1.0	0.0000	8.83E10
ANHPRM	0.6556	1.0	0.0000	8.24E10
BPVOL	0.6797	1.0	0.0000	7.73E10
SHRGSSAT	0.6886	1.0	0.0041	7.57E10
BPINTPRS	0.6976	1.0	0.0035	7.41E10
WGRCOR	0.7045	1.0	0.0098	7.30E10
WASTWICK	0.7109	1.0	0.0118	7.20E10
R_A^2 = 0.7009, PRS_A = 7.10E10, C_T = 0.9467				
QUAD_REG				
BHPRM	0.2923	2.0	0.0000	1.64E11
BPCOMP	0.4890	3.0	0.0000	1.23E11
WMICDFLG	0.6088	4.0	0.0000	9.62E10
HALPOR	0.7182	5.0	0.0000	7.18E10
ANHPRM	0.7831	6.0	0.0000	6.03E10
BPVOL	0.8215	7.0	0.0000	5.30E10
WGRCOR	0.8477	8.0	0.0000	4.89E10
BPINTPRS	0.8709	9.0	0.0000	4.67E10
SHPRMDRZ	0.8866	10.0	0.0004	4.44E10
SHPRMCON	0.8978	11.0	0.0093	4.43E10
SHPRMCLY	0.9087	12.0	0.0129	4.36E10
R_A^2 = 0.8770, PRS_A = 3.80E10, C_T = 0.9174				
PP_REG				
BHPRM	0.2916	1.7	0.0000	1.65E11
BPCOMP	0.4768	1.3	0.0000	1.29E11
HALPOR	0.6822	25.0	0.0000	1.09E11
WMICDFLG	0.8256	19.0	0.0000	7.06E10
ANRGSSAT	0.8389	2.6	0.0001	7.80E10
R_A^2 = 0.8069 PRS_A = 5.35E10, C_T = 0.3972				
SRD/RCC TEST				
BHPRM	NA	4.0	0.0000	NA
BPCOMP	NA	4.0	0.0000	NA
HALPOR	NA	4.0	0.0000	NA
BPPRM	NA	4.0	0.0000	NA
WMICDFLG	NA	4.0	0.0000	NA
BPVOL	NA	4.0	0.0057	NA
R_A^2 = NA, PRS_A = NA, C_T = 0.9225				

Var	R^2	*df*	*p*-value	*PRS*
RANK_REG				
BHPRM	0.3415	1.0	0.0000	1.50E6
BPCOMP	0.4874	1.0	0.0000	1.18E6
HALPOR	0.6065	1.0	0.0000	9.12E5
WMICDFLG	0.6778	1.0	0.0000	7.52E5
BPVOL	0.6974	1.0	0.0000	7.12E5
ANHPRM	0.7104	1.0	0.0003	6.87E5
BPINTPRS	0.7117	1.0	0.0063	6.75E5
R_A^2 = 0.7109, PRS_A = 6.70E5, C_T = 0.9629				
LOESS				
BHPRM	0.2933	2.3	0.0000	1.64E11
BPCOMP	0.5242	10.8	0.0000	1.22E11
HALPOR	0.7473	50.6	0.0000	1.03E11
ANHPRM	0.7404	-21.7	0.0012	8.87E10
WMICDFLG	0.8379	15.5	0.0000	6.17E10
BPVOL	0.8814	22.5	0.0000	5.83E10
R_A^2 = 0.8382, PRS_A = 5.07E10, C_T = 0.7243				
RP_REG				
BHPRM	0.2868	1.0	0.0000	1.66E11
BPCOMP	0.5244	11.0	0.0000	1.25E11
WMICDFLG	0.6582	12.0	0.0000	1.01E11
HALPOR	0.7899	16.0	0.0000	7.37E10
ANHPRM	0.8909	42.0	0.0000	5.95E10
HALPRM	0.9281	41.0	0.0002	9.02E10
BPPRM	0.9461	19.0	0.0003	7.32E10
R_A^2 = 0.8973, PRS_A = 4.49E10, C_T = 0.7110				
GAM				
BHPRM	0.2928	2.0	0.0000	1.64E11
BPCOMP	0.4880	2.0	0.0000	1.22E11
HALPOR	0.6011	4.0	0.0000	9.74E10
ANHPRM	0.6876	7.0	0.0000	8.03E10
WMICDFLG	0.7450	2.0	0.0000	6.60E10
BPVOL	0.7596	1.0	0.0000	6.28E10
SHRBRSAT	0.7846	10.0	0.0008	6.02E10
SHRGSSAT	0.7951	4.0	0.0099	5.90E10
WGRCOR	0.8042	2.0	0.0024	5.71E10
WASTWICK	0.8107	1.0	0.0028	5.58E10
SHPRMCLY	0.8181	2.0	0.0057	5.44E10
R_A^2 = 0.7924, PRS_A = 5.44E10, C_T = 0.8729				

[a] Table structure same as described in footnotes to Table 6.

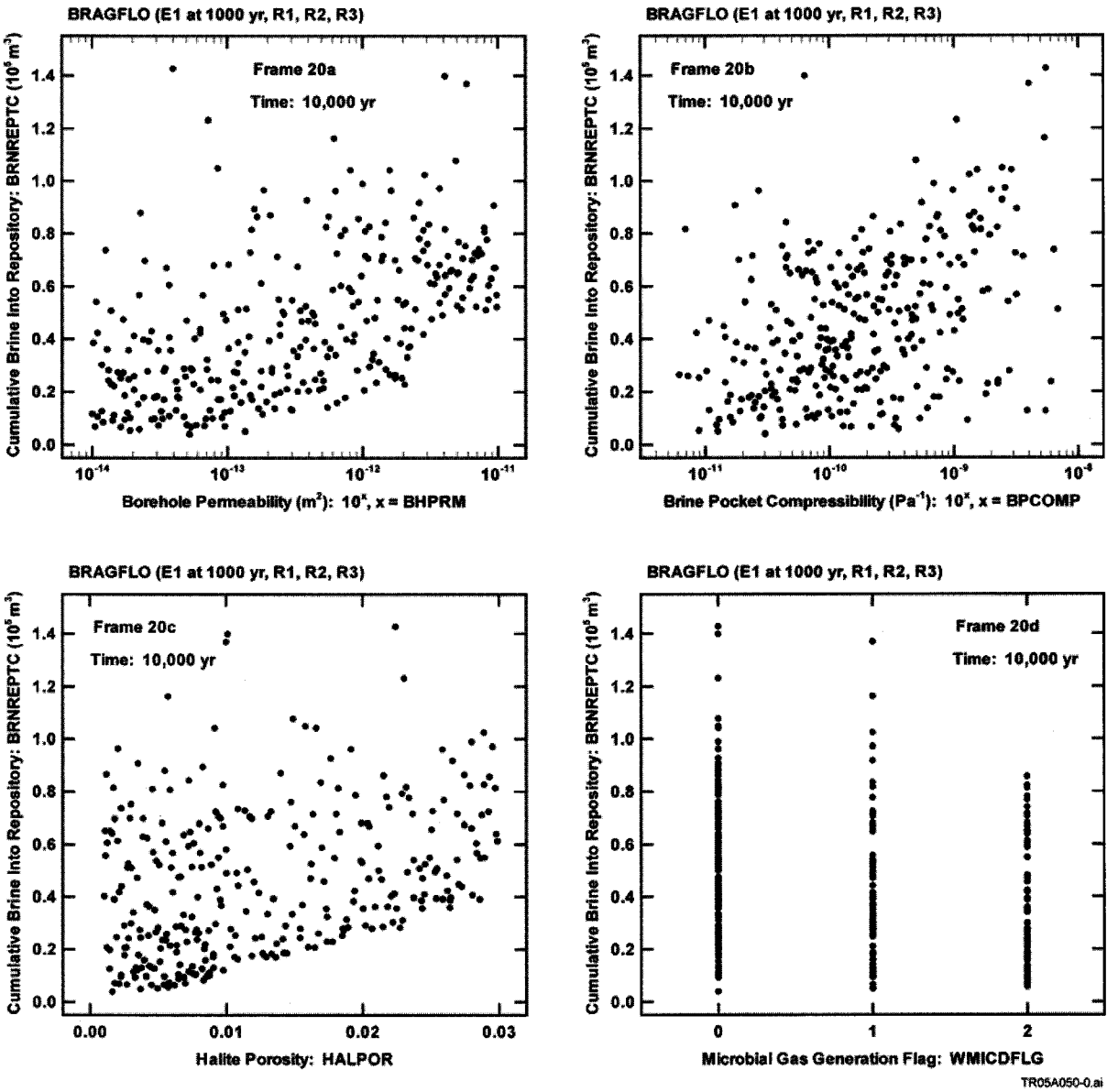


Fig. 20. Scatterplots for cumulative brine flow at 10,000 yr into repository (*BRNREPTC.10K*) for an E1 intrusion at 1000 yr.

Table 14. Sensitivity Analyses for Average Brine Saturation at 1000 yr in Waste Panels Not Penetrated by a Drilling Intrusion (*REP_SATB.1K*) for Undisturbed Conditions[a]

Var	R^2	*df*	*p*-value	*PRS*
LIN_REG				
HALPOR	0.5739	1.0	0.0000	1.30E2
WGRCOR	0.7398	1.0	0.0000	7.99E1
WMICDFLG	0.8267	1.0	0.0000	5.36E1
WASTWICK	0.8792	1.0	0.0000	3.77E1
SHRGSSAT	0.8819	1.0	0.0092	3.71E1
R_A^2 = 0.8799, PRS_A = 3.68E1, C_T = 0.9834				
QUAD_REG				
HALPOR	0.5835	2.0	0.0000	1.28E2
WGRCOR	0.7904	3.0	0.0000	6.59E1
WMICDFLG	0.9211	4.0	0.0000	2.53E1
WASTWICK	0.9804	5.0	0.0000	6.74E0
WRBRNSAT	0.9825	6.0	0.0000	6.35E0
SALPRES	0.9841	7.0	0.0002	6.04E0
ANHPRM	0.9853	8.0	0.0124	6.04E0
SHPRMDRZ	0.9864	9.0	0.0114	5.88E0
R_A^2 = 0.9841, PRS_A = 5.62E0, C_T = 0.9243				
PP_REG				
HALPOR	0.5822	2.0	0.0000	1.29E2
WGRCOR	0.8001	4.0	0.0000	6.44E1
WMICDFLG	0.9430	25.4	0.0000	2.31E1
WASTWICK	0.9926	12.7	0.0000	4.22E0
SHRGSSAT	0.9946	25.4	0.0000	5.43E0
BHPRM	0.9961	16.6	0.0000	5.74E0
R_A^2 = 0.9945 PRS_A = 2.36E0, C_T = 0.8598				
SRD/RCC TEST				
HALPOR	NA	4.0	0.0000	NA
WGRCOR	NA	4.0	0.0000	NA
WMICDFLG	NA	4.0	0.0000	NA
WASTWICK	NA	4.	0.0000	NA
R_A^2 = NA, PRS_A = NA, C_T = 0.9102				

Var	R^2	*df*	*p*-value	*PRS*
RANK_REG				
HALPOR	0.6141	1.0	0.0000	8.79E5
WGRCOR	0.7704	1.0	0.0000	5.27E5
WASTWICK	0.8516	1.0	0.0000	3.44E5
WMICDFLG	0.9201	1.0	0.0000	1.87E5
R_A^2 = 0.9190, PRS_A = 1.86E5, C_T = 0.9973				
LOESS				
HALPOR	0.5821	2.3	0.0000	1.28E2
WGRCOR	0.8025	9.7	0.0000	6.52E1
WMICDFLG	0.9593	67.3	0.0000	2.40E1
WASTWICK	0.9919	4.0	0.0000	5.10E0
R_A^2 = 0.9888, PRS_A = 4.69E0, C_T = 1.000				
RP_REG				
HALPOR	0.6461	10.0	0.0000	1.30E2
WGRCOR	0.8161	7.0	0.0000	7.84E1
WMICDFLG	0.9487	31.0	0.0000	2.82E1
WASTWICK	0.9896	42.0	0.0000	9.96E0
ANHBCVGP	0.9929	31.0	0.0000	1.09E1
R_A^2 = 0.9881, PRS_A = 6.01E0, C_T = 0.9664				
GAM				
HALPOR	0.5821	2.0	0.0000	1.28E2
WGRCOR	0.7564	2.0	0.0000	7.60E1
WMICDFLG	0.8409	2.0	0.0000	5.02E1
WASTWICK	0.8953	2.0	0.0000	3.36E1
R_A^2 = 0.8925, PRS_A = 3.33E1, C_T = 0.9546				

[a] Table structure same as described in footnotes to Table 6.

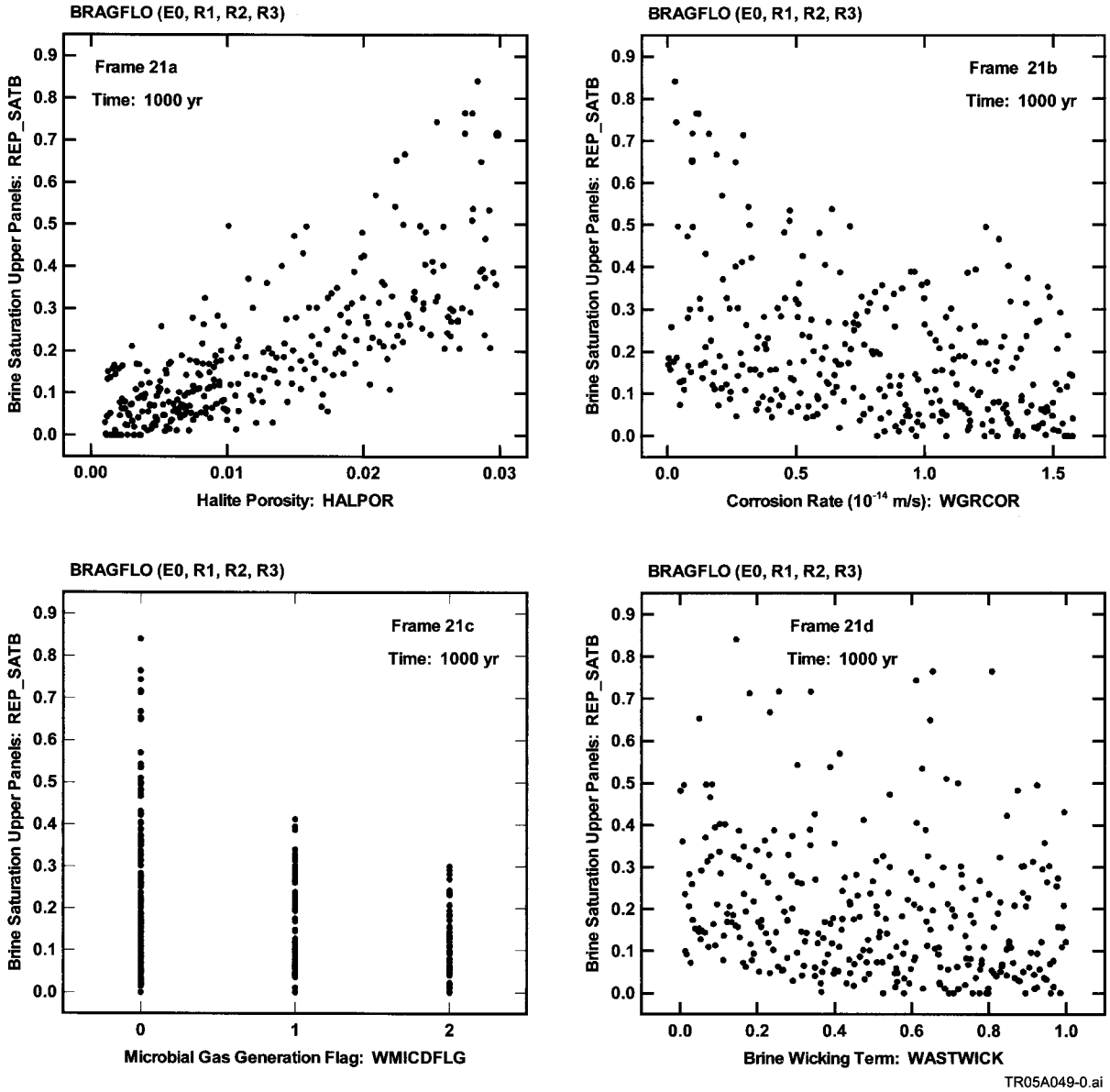


Fig. 21. Scatterplots for average brine saturation at 1000 yr in waste panels not penetrated by a drilling intrusion (*REP_SATB.1K*) for undisturbed conditions.

Table 15. Sensitivity Analyses for Average Brine Saturation at 10,000 yr in Waste Panels Not Penetrated by a Drilling Intrusion (*REP_SATB.10K*) for an E1 Intrusion at 1000 yr[a]

Var	R^2	*df*	*p*-value	*PRS*
LIN_REG				
WGRCOR	0.2803	1.0	0.0000	2.18E2
BHPRM	0.4359	1.0	0.0000	1.72E2
HALPOR	0.5111	1.0	0.0000	1.50E2
BPCOMP	0.5811	1.0	0.0000	1.30E2
SHRGSSAT	0.6048	1.0	0.0000	1.23E2
WASTWICK	0.6258	1.0	0.0001	1.18E2
WMICDFLG	0.6443	1.0	0.0001	1.12E2
ANHPRM	0.6601	1.0	0.0003	1.08E2
BPVOL	0.6696	1.0	0.0041	1.06E2
R_A^2 = 0.6594, PRS_A = 1.06E2, C_T = 0.9003				
QUAD_REG				
WGRCOR	0.4232	2.0	0.0000	1.76E2
BHPRM	0.5992	3.0	0.0000	1.24E2
HALPOR	0.6717	4.0	0.0000	1.04E2
BPCOMP	0.7449	5.0	0.0000	8.48E1
WMICDFLG	0.7835	6.0	0.0000	7.48E1
WASTWICK	0.8101	7.0	0.0000	7.04E1
ANHPRM	0.8329	8.0	0.0000	6.66E1
SHRGSSAT	0.8496	9.0	0.0012	6.40E1
R_A^2 = 0.8237, PRS_A = 6.22E1, C_T = 0.9120				
PP_REG				
WGRCOR	0.4736	4.3	0.0000	1.64E2
BHPRM	0.6472	5.5	0.0000	1.14E2
SHPRMCLY	0.7382	27.4	0.0000	1.14E2
BPCOMP	0.7954	-2.5	0.0000	9.98E1
HALPRM	0.8657	30.0	0.0000	9.16E1
BPVOL	0.8985	12.6	0.0000	1.06E2
R_A^2 = 0.8632 PRS_A = 5.60E1, C_T = 0.6226				
SRD/RCC TEST				
WGRCOR	NA	4.0	0.0000	NA
BHPRM	NA	4.0	0.0000	NA
HALPOR	NA	4.0	0.0000	NA
BPCOMP	NA	4.0	0.0001	NA
BPPRM	NA	4.0	0.0144	
R_A^2 = NA, PRS_A = NA, C_T = 0.9292				

Var	R^2	*df*	*p*-value	*PRS*
RANK_REG				
BHPRM	0.2619	1.0	0.0000	1.68E6
WGRCOR	0.4790	1.0	0.0000	1.19E6
HALPOR	0.5584	1.0	0.0000	1.02E6
BPCOMP	0.6226	1.0	0.0000	8.78E5
WASTWICK	0.6488	1.0	0.0000	8.23E5
WMICDFLG	0.6703	1.0	0.0000	7.78E5
SHRGSSAT	0.6815	1.0	0.0015	7.56E5
ANHPRM	0.6928	1.0	0.0012	7.35E5
BPVOL	0.7023	1.0	0.0026	7.17E5
R_A^2 = 0.6931 PRS_A = 7.17E5, C_T = 0.8966				
LOESS				
WGRCOR	0.4751	5.6	0.0000	1.64E2
BHPRM	0.7118	29.1	0.0000	1.07E2
BPCOMP	0.7401	-11.3	0.0000	9.21E1
HALPOR	0.7968	14.1	0.0000	8.10E1
R_A^2 = 0.7676, PRS_A = 8.00E1, C_T = 0.9203				
RP_REG				
WGRCOR	0.4836	7.0	0.0000	1.68E2
BHPRM	0.6905	10.0	0.0000	1.20E2
HALPOR	0.7689	14.0	0.0000	1.06E2
BPCOMP	0.8336	9.0	0.0000	7.66E1
WASTWICK	0.8740	20.0	0.0000	8.39E1
SHRGSSAT	0.8971	24.0	0.0047	7.77E1
R_A^2 = 0.8570, PRS_A = 5.99E1, C_T = 0.8300				
GAM				
WGRCOR	0.4719	4.0	0.0000	1.63E2
BHPRM	0.6523	2.0	0.0000	1.09E2
HALPOR	0.7227	4.0	0.0000	8.88E1
BPCOMP	0.7732	1.0	0.0000	7.35E1
WASTWICK	0.7994	4.0	0.0000	6.71E1
ANHPRM	0.8169	2.0	0.0000	6.24E1
WMICDFLG	0.8443	2.0	0.0000	5.35E1
R_A^2 = 0.8338, PRS_A = 5.34E1, C_T = 0.9000				

[a] Table structure same as described in footnotes to Table 6.

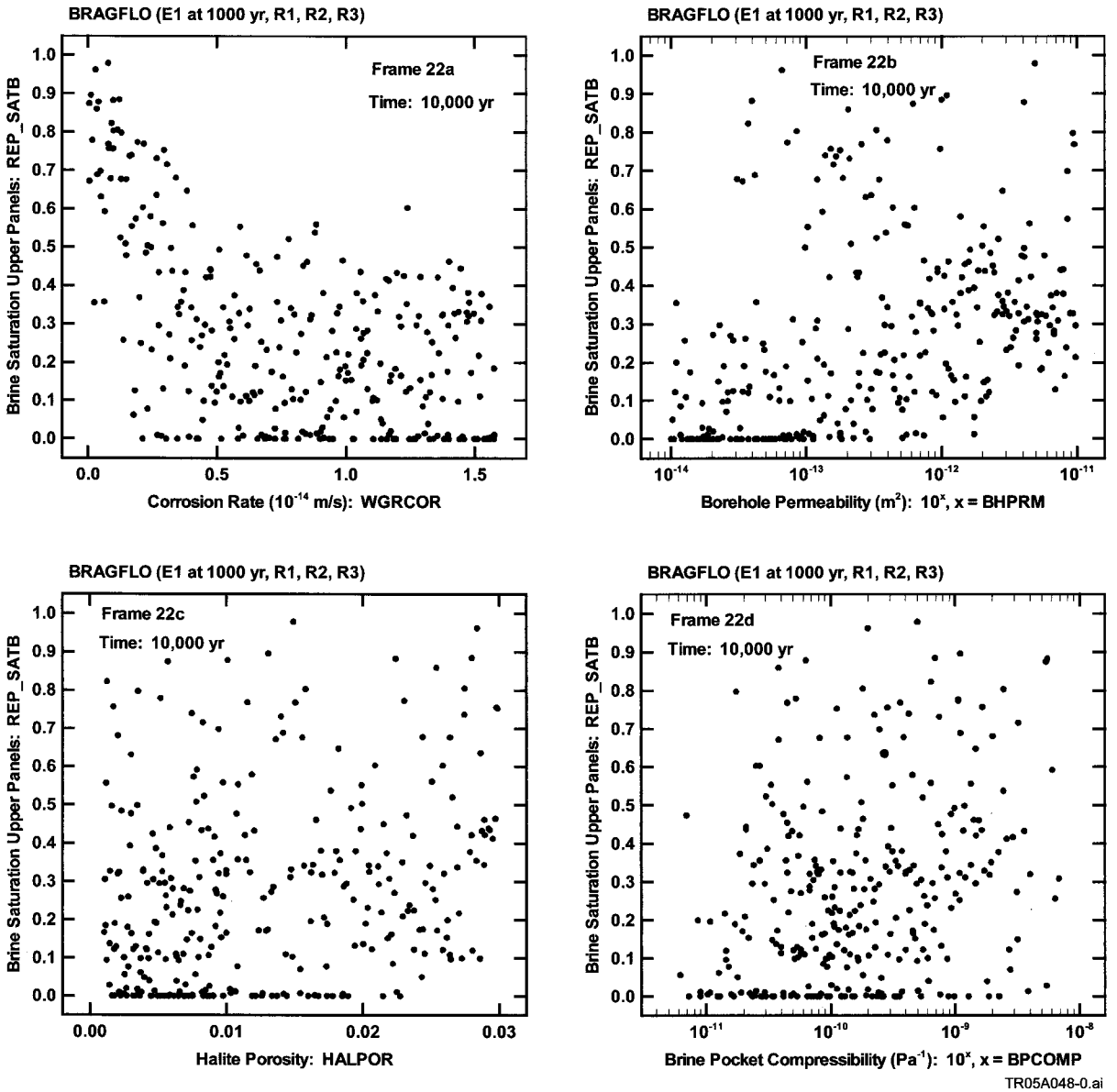


Fig. 22. Scatterplots for average brine saturation at 10,000 yr in waste panels not penetrated by a drilling intrusion (*REP_SATB.10K)* for an E1 intrusion at 1000 yr.

Although PP_REG has a reasonably high R^2 value, it has a low TDCC of 0.62. The reasons for the lack of reproducibility, and thus overall poor performance of the PP_REG procedure in this example, are not clear at this time. Also, PP_REG and RP_REG again have jumps in PRESS values at the end of the stepwise process as variables with very small effects are added to the models, which indicates that an overfitting of the data could be taking place.

The other regression methods have TDCCs between 0.83 and 0.92, which suggests that they are providing more reproducible results than the PP_REG procedure. The SRD/RCC test agrees with the regression methods on the first four variables and also has a high TDCC of 0.93. It also includes *BPPRM* when none of the other methods do. As discussed in conjunction with *BRNREPTC.10K* in Sect. 5.2.2, this difference in variable selection probably results from the –0.75 rank correlation between *BPPRM* and *BPCOMP*.

5.2.5 Pressure at 1000 yr (*WAS_PRES.1K*)

The analyses for *WAS_PRES.1K* (Table 16) show again that linear models can work quite well in some

Table 16. Sensitivity Analysis for Pressure at 1000 yr in Waste Panel Penetrated by a Drilling Intrusion (*WAS_PRES.1K*) for Undisturbed Conditions[a]

Var	R^2	*df*	*p*-value	*PRS*
LIN_REG				
WMICDFLG	0.8457	1.0	0.0000	4.67E1
WGRCOR	0.9193	1.0	0.0000	2.46E1
WASTWICK	0.9503	1.0	0.0000	1.53E1
HALPOR	0.9535	1.0	0.0000	1.44E1
ANHPRM	0.9559	1.0	0.0001	1.38E1
WGRMICI	0.9573	1.0	0.0016	1.34E1
ANHBCVGP	0.9582	1.0	0.0161	1.32E1
R_A^2 = 0.9572, PRS_A = 1.32E1, C_T = 0.9401				
QUAD_REG				
WMICDFLG	0.8564	2.0	0.0000	4.37E1
WGRCOR	0.9451	3.0	0.0000	1.71E1
WASTWICK	0.9769	4.0	0.0000	7.44E0
IIALPOR	0.9861	5.0	0.0000	4.64E0
WGRMICI	0.9886	6.0	0.0000	4.10E0
ANHPRM	0.9900	7.0	0.0000	3.74E0
WGRMICII	0.9908	8.0	0.0067	3.70E0
R_A^2 = 0.9895, PRS_A = 3.57E0, C_T = 0.9569				
PP_REG				
WMICDFLG	0.8564	2.0	0.0000	4.37E1
WGRCOR	0.9458	4.3	0.0000	1.71E1
WASTWICK	0.9718	3.8	0.0000	8.29E0
WRBRNSAT	0.9827	30.5	0.0000	9.15E0
HALPOR	0.9904	9.9	0.0000	5.49E0
SHBCEXP	0.9947	27.8	0.0000	6.68E0
HALPRM	0.9960	19.7	0.0000	6.16E0
ANHBCVGP	0.9963	1.5	0.0002	6.72E0
SHRBRSAT	0.9978	21.3	0.0000	8.47E0
R_A^2 = 0.9963 PRS_A = 1.87E0, C_T = 0.8671				
SRD/RCC TEST				
WMICDFLG	NA	4.0	0.0000	NA
WGRCOR	NA	4.0	0.0000	NA
WASTWICK	NA	4.0	0.0001	NA
SHPRMASP	NA	4.0	0.0120	NA
R_A^2 = NA, PRS_A = NA, C_T = 0.8697				

Var	R^2	*df*	*p*-value	*PRS*
RANK_REG				
WMICDFLG	0.7830	1.0	0.0000	4.94E5
WGRCOR	0.8909	1.0	0.0000	2.51E5
WASTWICK	0.9366	1.0	0.0000	1.47E5
HALPOR	0.9427	1.0	0.0000	1.34E5
ANHPRM	0.9456	1.0	0.0001	1.28E5
WGRMICI	0.9468	1.0	0.0094	1.26E5
R_A^2 = 0.9457, PRS_A = 1.26E5, C_T = 0.9834				
LOESS				
WMICDFLG	0.8564	2.0	0.0000	4.37E1
WGRCOR	0.9528	25.4	0.0000	1.74E1
WASTWICK	0.9852	47.0	0.0000	8.04E0
HALPOR	0.9924	9.0	0.0000	5.25E0
WGRMICI	0.9949	44.3	0.0018	9.02E0
R_A^2 = 0.9911, PRS_A = 4.68E0, C_T = 0.9755				
RP_REG				
WMICDFLG	0.8564	4.0	0.0000	4.45E1
WGRCOR	0.9505	13.0	0.0000	1.78E1
WASTWICK	0.9782	13.0	0.0000	9.54E0
HALPOR	0.9929	52.0	0.0000	6.55E0
SHPRMCON	0.9940	19.0	0.0157	8.38E0
WGRMICI	0.9955	19.0	0.0001	1.03E1
R_A^2 = 0.9924, PRS_A = 3.80E0, C_T = 0.9593				
GAM				
WMICDFLG	0.8457	1.0	0.0000	4.67E1
WGRCOR	0.9338	3.0	0.0000	2.05E1
WASTWICK	0.9654	2.0	0.0000	1.09E1
HALPOR	0.9695	1.0	0.0000	9.72E0
ANHPRM	0.9721	2.0	0.0000	8.99E0
WGRMICI	0.9732	2.0	0.0042	8.79E0
HALPRM	0.9763	15.0	0.0025	8.61E0
R_A^2 = 0.9741, PRS_A = 8.55E0, C_T = 0.8834				

[a] Table structure same as described in footnotes to Table 6.

situations. The dominant variable contributing to the uncertainty in *WAS_PRES.1K* is *WMICDFLG*, with the regression methods indicating that *WMICDFLG* accounts for approximately 85% of the uncertainty in *WAS_PRES.1K*. After *WMICDFLG*, the variable *WGRCOR* contributes an additional 10% of the uncertainty to *WAS_PRES.1K*. Owing to the linearity of the relationships between *WMICDFLG*, *WGRCOR* and *WAS_PRES.1K* (Fig. 23), the estimated contributions of *WMICDFLG* and *WGRCOR* to the uncertainty in *WAS_PRES.1K* are approximately the same for all regression methods. Further, the next two most important contributors to the uncertainty in *WAS_PRES.1K* (i.e., *WASTWICK* and *HALPOR*) are also consistently identified by all the regression methods. However, the effects of *WASTWICK* and *HALPOR* are small relative to the effects associated with *WMICDFLG* and *WGRCOR* as indicated by the incremental R^2 values associated with individual regressions and the scatterplots in Fig. 23. As indicated by the incremental degrees of freedom for individual regression models, the nonparametric regression models are considerably more complex than the models constructed with the linear regression procedures.

The SRD/RCC test produces results consistent with the regression methods in that it identifies *WMICDFLG*, *WGRCOR* and *WASTWICK*, in that order, as the three dominant contributors to the uncertainty in *WAS_PRES.1K*. However, the identification of an effect for *SHPRMASP* by the SRD/RCC test is probably spurious.

All of the procedures result in TDCCs close to or above 0.9. Thus, reproducible results for all procedures are being obtained for the dominant contributors to the uncertainty in *WAS_PRES.1K*. However, the large number of variables with marginal effects selected at the end of the analysis with PP_REG and the associated increases in PRESS values suggest that an overfitting of the data is taking place. Some increases in PRESS values near the end of the stepwise process also takes place for LOESS and RP_REG.

5.2.6 Pressure at 10,000 yr (*WAS_PRES.10K*)

The limitations of linear methods for sensitivity analysis are shown in the analyses for *WAS_PRES.10K* (Table 17). The dominant variable contributing to the uncertainty in *WAS_PRES.10K* is *BHPRM* (Fig. 24) and provided the illustrative example used for scatterplot smoothers in Sect. 3.1. The relationship between *BHPRM* is both nonlinear and nonmonotonic. Linear regression with raw or rank-transformed data is essentially useless in this case and fails to even include *BHPRM* in the model when it is clearly the input most responsible for the uncertainty in the output (Table 17). While linear regression with raw or rank-transformed data had final R^2 values of about 0.27, the nonparametric methods and also QUAD_REG had R^2 values in the 0.8 – 0.9 range. Because of its limitations in higher dimensions, LOESS will sometimes include too few variables in the model, which may be the case here.

The analyses with QUAD_REG, LOESS, GAM, PP_REG and RP_REG all had reasonably high R^2 values (i.e., 0.85, 0.84, 0.79, 0.83, 0.95) and generally agreed on the four most important variables (i.e., *BHPRM*, *HALPRM*, *BPCOMP*, *ANHPRM*), although RP_REG and PP_REG include *WGRCOR* as the second and third variable, respectively, in the model (see Table 17). All methods indicated that *BHPRM* was responsible for about 50% of the uncertainty in *WAS_PRES.10K*. However, PP_REG had a TDCC of 0.64, which is low. The analysis with RP_REG has a high R^2 value of 0.95 and a TDCC of 0.86. The analyses with GAM and QUAD_REG have TDCC values of 0.75 and 0.86, respectively, but have lower R^2 values than RP_REG. Based on the methods with high R^2 values, a breakdown for percentage contributors to the uncertainty in *WAS_PRES.10K* would be *BHPRM* with 50%, *BPCOMP* with about 10%, *HALPRM* with 5 – 10%, WGRCOR with 5 – 10%, and *ANHPRM* with 5 – 10%. After that, *HALPOR* may account for as much as another 5%. Again, the SRD/RCC test agrees with the regression methods on the first four variables and has a high TDCC of 0.91.

As seen in other analyses, jumps in PRESS values occur for LOESS, PP_REG and RP_REG near the end of the stepwise process.

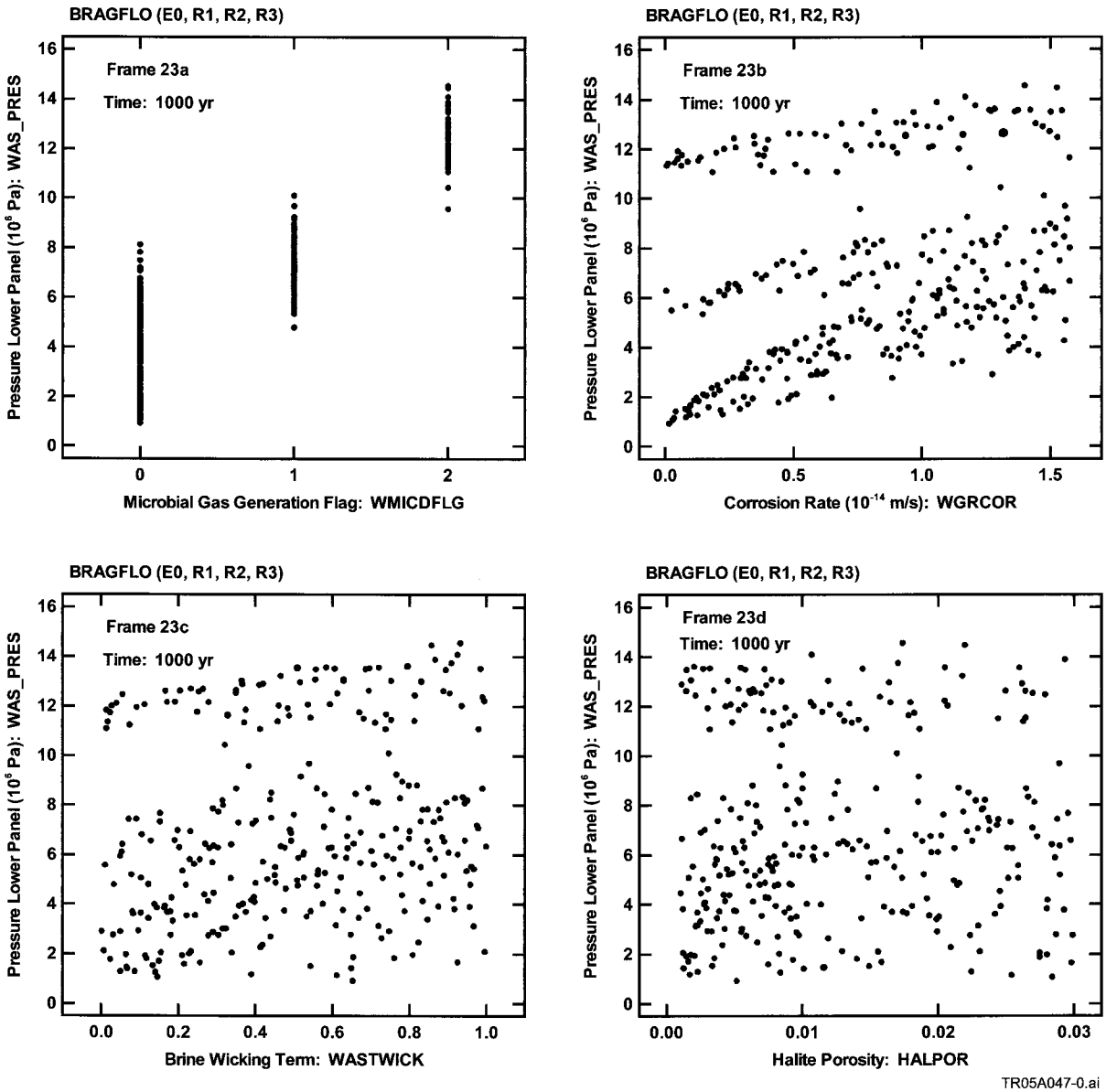


Fig. 23. Scatterplots for pressure at 1000 yr in waste panel penetrated by a drilling intrusion (*WAS_PRES.1K*) for undisturbed conditions.

Table 17. Sensitivity Analyses for Pressure at 10,000 yr in Waste Panel Penetrated by a Drilling Intrusion (*WAS_PRES.10K*) for an E1 Intrusion at 1000 yr[a]

Var	R^2	*df*	*p*-value	*PRS*
LIN_REG				
HALPRM	0.1188	1.0	0.0000	2.67E2
BPCOMP	0.1724	1.0	0.0000	2.53E2
ANHPRM	0.2168	1.0	0.0001	2.41E2
HALPOR	0.2428	1.0	0.0016	2.35E2
BPVOL	0.2679	1.0	0.0017	2.28E2
R_A^2 = 0.2554, PRS_A = 2.28E2, C_T = 0.7730				
QUAD_REG				
BHPRM	0.4550	2.0	0.0000	1.66E2
HALPRM	0.5499	3.0	0.0000	1.39E2
BPCOMP	0.6201	4.0	0.0000	1.22E2
ANHPRM	0.6873	5.0	0.0000	1.05E2
HALPOR	0.7299	6.0	0.0000	9.52E1
WGRCOR	0.7713	7.0	0.0000	8.59E1
WMICDFLG	0.8030	8.0	0.0000	7.83E1
BPVOL	0.8273	9.0	0.0001	7.41E1
BPINTPRS	0.8456	10.0	0.0018	7.28E1
R_A^2 = 0.8116, PRS_A = 6.92E1, C_T = 0.8646				
PP_REG				
BHPRM	0.4992	9.7	0.0000	1.62E2
HALPRM	0.5882	4.7	0.0000	1.40E2
WGRCOR	0.6794	20.5	0.0000	1.45E2
BPCOMP	0.7181	-15.0	0.0000	1.13E2
HALPOR	0.8261	24.9	0.0000	1.14E2
R_A^2 = 0.7955, PRS_A = 7.30E1, C_T = 0.6399				
SRD/RCC TEST				
BHPRM	NA	4.0	0.0000	NA
HALPRM	NA	4.0	0.0000	NA
BPCOMP	NA	4.0	0.0002	NA
ANHPRM	NA	4.0	0.0011	NA
BPVOL	NA	4.0	0.0149	NA
R_A^2 = NA, PRS_A = NA, C_T = 0.9074				

Var	R^2	*df*	*p*-value	*PRS*
RANK_REG				
HALPRM	0.1207	1.0	0.0000	2.00E6
BPCOMP	0.1716	1.0	0.0000	1.90E6
ANHPRM	0.2023	1.0	0.0008	1.84E6
BPVOL	0.2258	1.0	0.0030	1.80E6
HALPOR	0.2494	1.0	0.0026	1.76E6
SHRGSSAT	0.2636	1.0	0.0182	1.74E6
R_A^2 = 0.2485, PRS_A = 1.74E6, C_T = 0.8835				
LOESS				
BHPRM	0.5312	17.5	0.0000	1.59E2
HALPRM	0.6332	16.4	0.0000	1.39E2
ANHPRM	0.7444	32.4	0.0000	1.23E2
BPCOMP	0.8371	39.8	0.0000	1.35E2
R_A^2 = 0.7477, PRS_A = 1.18E2, C_T = 0.7733				
RP_REG				
BHPRM	0.5294	16.0	0.0000	1.69E2
WGRCOR	0.6588	26.0	0.0000	1.73E2
BPCOMP	0.7628	17.0	0.0000	1.45E2
ANHPRM	0.8049	8.0	0.0000	1.21E2
HALPRM	0.8540	4.0	0.0000	1.70E2
WRBRNSAT	0.9230	52.0	0.0000	8.21E1
BPINTPRS	0.9495	34.0	0.0007	1.16E2
R_A^2 = 0.8937, PRS_A = 6.74E1, C_T = 0.8632				
GAM				
BHPRM	0.4992	10.0	0.0000	1.61E2
HALPRM	0.5613	1.0	0.0000	1.42E2
ANHPRM	0.6305	4.0	0.0000	1.23E2
BPCOMP	0.6884	2.0	0.0000	1.06E2
HALPOR	0.7296	4.0	0.0000	9.46E1
WGRCOR	0.7564	4.0	0.0000	8.78E1
BPVOL	0.7666	1.0	0.0007	8.47E1
SHRBRSAT	0.7776	4.0	0.0111	8.30E1
SHRGSSAT	0.7833	1.0	0.0084	8.16E1
BPINTPRS	0.7891	1.0	0.0075	7.99E1
R_A^2 = 0.7638, PRS_A = 7.96E1, C_T = 0.7460				

[a] Table structure same as described in footnotes to Table 6.

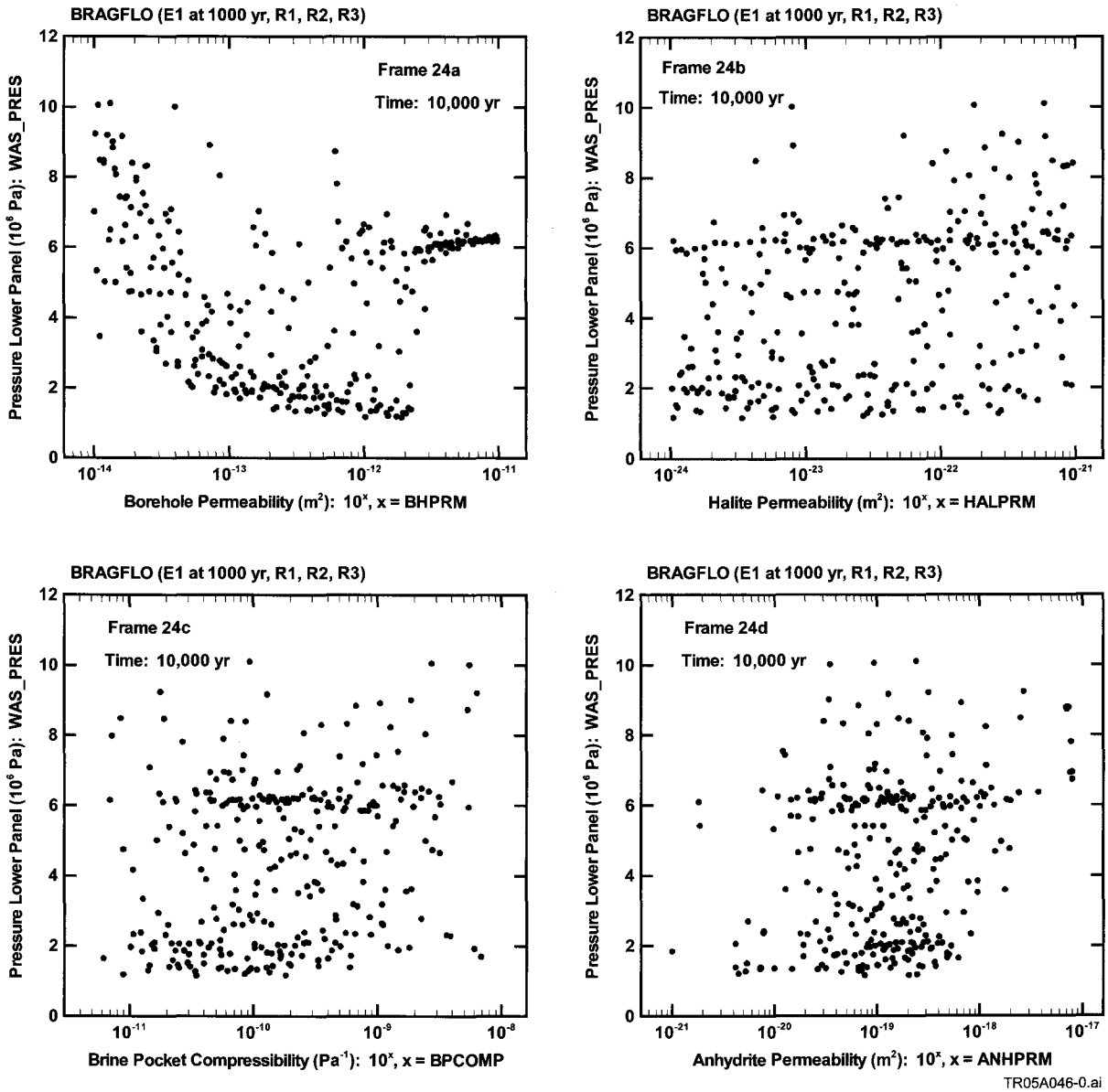


Fig. 24. Scatterplots for pressure at 10,000 yr in waste panel penetrated by a drilling intrusion (*WAS_PRES.10K*) for an E1 intrusion at 1000 yr.

This page intentionally left blank.

6. Observations and Insights

The following observations and insights are based on the examples described in this presentation. Nonparametric methods worked quite well for sensitivity analysis and provide a useful addition to currently employed sampling-based sensitivity analysis procedures.

The overall best method considered in this study is RP_REG. In the test cases, it almost always ordered the input variables correctly and estimated the contributions to R^2 accurately. The drawback is that it generally takes longer to apply than any of the other methods.

The GAM and QUAD_REG procedures displayed good performance on the test data and are fast computationally. The QUAD_REG procedure can model a certain degree of interaction while GAM does not. However, GAM can model more general nonlinearity than QUAD_REG. Also, multiplicative interaction terms could be used in GAM to make it a more general method.

The LOESS and PP_REG procedures exhibited some problems that could reduce their usefulness for sensitivity analysis. Specifically, LOESS sometimes failed to identify important input variables, although it usually identified the two most important variables. The PP_REG procedure showed a tendency to err in the opposite direction and often included insignificant input variables in the model. This tendency was indicated by the jumps in PRESS values that often occurred near the end of the stepwise implementation of PP_REG.

The SRD/RCC test also performed well and identified the dominant variables in all the analyses. The drawback to this test is that it does not provide the fraction of the uncertainty in the dependent variable explained by each of the identified independent variables. However, it has an advantage over the non-parametric regression procedures in being both conceptually simple and computationally quick.

Given the nonlinear relationships that can be present in analyses with complex computer models, one should be cautious about using only linear methods for sensitivity analysis. However, when a linear regression with raw or rank-transformed data is appropriate, it should be used as it is the easiest method to implement and interpret.

A reasonable analysis strategy is initially to fit linear regressions with raw and rank-transformed data and observe the R^2 values. If these values are below 0.9, then fit a QUAD_REG surface. If QUAD_REG also has an R^2 below 0.9, then fit a GAM surface. If the GAM surface still has a low R^2, then fit a RP_REG model. This approach restricts the use of the more computationally demanding RP_REG procedure to situations where its use is necessary. This is important because real analyses can involve carrying out sensitivity analyses for hundreds of time-dependent analysis results (e.g., see the sensitivity analyses summarized in Ref. 56).

If the resources are not available to carry out the indicated sequence of nonparametric regressions, then the SRD/RCC test provides a computationally efficient way to identify nonlinear relationships. Another analysis possibility is to use the SRD/RCC test to identify the dominant contributors to the uncertainty in a dependent variable, and then consider only these dominant variables in a nonparametric regression analysis.

The authors' experience is that linear regression with rank-transformed data and examination of associated scatterplots are usually sufficient to carry out a successful sensitivity analysis. However, there are situations where this approach will not be successful. Then, nonparametric regression procedures can often provide the needed techniques to determine the relationships between uncertain analysis inputs and analysis results.

Additional generalized regression procedures also exist that merit investigation for their potential usefulness in sensitivity analysis. For example, additional procedures for additive modeling include the Alternating Conditional Expectation (ACE) algorithm[153] and the Additivity and Variance Stabilization (AVAS) algorithm[154] (see Ref. 120, pp. 175 – 194, for additional discussion of the ACE and AVAS algorithms). There are also more sophisticated forms of recursive partitioning such as Multivariate Adaptive Regression Splines (MARS) (Ref. 155; also Ref. 120, pp. 275 – 277) and Smoothed and Unsmoothed Piecewise-Polynomial Regression Trees (SUPPORT).[156] As the recursive partitioning technique (Sect. 3.3.4) was the best of the presented nonparametric regression methods, these two techniques merit investigation for use in sensitivity analysis. Gaussian process models have also been proposed for use in sensitivity analysis.[157-159] A comparison of the performance of Gaussian process models and nonparametric regression models in sensitivity analysis would be interesting.

This page intentionally left blank.

7. References

1. Christie, M.A., J. Glimm, J.W. Grove, D.M. Higdon, D.H. Sharp, and M.M. Wood-Schultz. 2005. "Error Analysis and Simulations of Complex Phenomena," *Los Alamos Science*. Vol. 29, pp. 6-25.

2. Sharp, D.H. and M.M. Wood-Schultz. 2003. "QMU and Nuclear Weapons Certification: What's Under the Hood?," *Los Alamos Science*. Vol. 28, pp. 47-53.

3. Wagner, R.L. 2003. "Science, Uncertainty and Risk: The Problem of Complex Phenomena," *APS News*. Vol. 12, no. 1, pp. 8.

4. Oberkampf, W.L., S.M. DeLand, B.M. Rutherford, K.V. Diegert, and K.F. Alvin. 2002. "Error and Uncertainty in Modeling and Simulation," *Reliability Engineering and System Safety*. Vol. 75, no. 3, pp. 333-357.

5. Risk Assessment Forum. 1997. *Guiding Principles for Monte Carlo Analysis*, EPA/630/R-97/001. Washington DC: U.S. Environmental Protection Agency. (Available from the NTIS as PB97-188106/XAB.).

6. NCRP (National Council on Radiation Protection and Measurements). 1996. *A Guide for Uncertainty Analysis in Dose and Risk Assessments Related to Environmental Contamination*, NCRP Commentary No. 14. Bethesda, MD: National Council on Radiation Protection and Measurements.

7. NRC (National Research Council). 1994. *Science and Judgment in Risk Assessment*, Washington, DC: National Academy Press.

8. NRC (National Research Council). 1993. *Issues in Risk Assessment*. Washington, DC: National Academy Press.

9. U.S. EPA (U.S. Environmental Protection Agency). 1993. *An SAB Report: Multi-Media Risk Assessment for Radon, Review of Uncertainty Analysis of Risks Associated with Exposure to Radon*, EPA-SAB-RAC-93-014. Washington, DC: U.S. Environmental Protection Agency.

10. IAEA (International Atomic Energy Agency). 1989. *Evaluating the Reliability of Predictions Made Using Environmental Transfer Models*, Safety Series No. 100. Vienna: International Atomic Energy Agency.

11. Beck, M.B. 1987. "Water-Quality Modeling: A Review of the Analysis of Uncertainty," *Water Resources Research*. Vol. 23, no. 8, pp. 1393-1442.

12. Cacuci, D.G. 2003. *Sensitivity and Uncertainty Analysis, Vol. 1: Theory*. Boca Raton, FL: Chapman and Hall/CRC Press.

13. Turányi, T. 1990. "Sensitivity Analysis of Complex Kinetic Systems. Tools and Applications," *Journal of Mathematical Chemistry*. Vol. 5, no. 3, pp. 203-248.

14. Rabitz, H., M. Kramer, and D. Dacol. 1983. "Sensitivity Analysis in Chemical Kinetics," *Annual Review of Physical Chemistry*. Vol. 34. Eds. B.S. Rabinovitch, J.M. Schurr, and H.L. Strauss. Palo Alto, CA: Annual Reviews Inc, pp. 419-461.

15. Lewins, J. and M. Becker, eds. 1982. *Sensitivity and Uncertainty Analysis of Reactor Performance Parameters*. Vol. 14. New York, NY: Plenum Press.

16. Frank, P.M. 1978. *Introduction to System Sensitivity Theory*. New York, NY: Academic Press.

17. Tomovic, R. and M. Vukobratovic. 1972. *General Sensitivity Theory*. New York, NY: Elsevier.

18. Myers, R.H., D.C. Montgomery, G.G. Vining, C.M. Borror, and S.M. Kowalski. 2004. "Response Surface Methodology: A Retrospective and Literature Review," *Journal of Quality Technology*. Vol. 36, no. 1, pp. 53-77.

19. Myers, R.H. 1999. "Response Surface Methodology - Current Status and Future Directions," *Journal of Quality Technology*. Vol. 31, no. 1, pp. 30-44.

20. Andres, T.H. 1997. "Sampling Methods and Sensitivity Analysis for Large Parameter Sets," *Journal of Statistical Computation and Simulation*. Vol. 57, no. 1-4, pp. 77-110.

21. Kleijnen, J.P.C. 1997. "Sensitivity Analysis and Related Analyses: A Review of Some Statistical Techniques," *Journal of Statistical Computation and Simulation*. Vol. 57, no. 1-4, pp. 111-142.

22. Kleijnen, J.P.C. 1992. "Sensitivity Analysis of Simulation Experiments: Regression Analysis and Statistical Design," *Mathematics and Computers in Simulation*. Vol. 34, no. 3-4, pp. 297-315.

23. Sacks, J., W.J. Welch, T.J. Mitchel, and H.P. Wynn. 1989. "Design and Analysis of Computer Experiments," *Statistical Science*. Vol. 4, no. 4, pp. 409-435.

24. Morton, R.H. 1983. "Response Surface Methodology," *Mathematical Scientist*. Vol. 8, pp. 31-52.

25. Mead, R. and D.J. Pike. 1975. "A Review of Response Surface Methodology from a Biometric Viewpoint," *Biometrics*. Vol. 31, pp. 803-851.

26. Myers, R.H. 1971. *Response Surface Methodology*. Boston, MA: Allyn and Bacon.

27. Helton, J.C. and F.J. Davis. 2003. "Latin Hypercube Sampling and the Propagation of Uncertainty in Analyses of Complex Systems," *Reliability Engineering and System Safety*. Vol. 81, no. 1, pp. 23-69.

28. Helton, J.C. and F.J. Davis. 2002. "Illustration of Sampling-Based Methods for Uncertainty and Sensitivity Analysis," *Risk Analysis*. Vol. 22, no. 3, pp. 591-622.

29. Helton, J.C. and F.J. Davis. 2000. "Sampling-Based Methods," *Sensitivity Analysis*. Ed. A. Saltelli, K. Chan, and E.M. Scott. New York, NY: Wiley. pp. 101-153.

30. Kleijnen, J.P.C. and J.C. Helton. 1999. "Statistical Analyses of Scatterplots to Identify Important Factors in Large-Scale Simulations, 1: Review and Comparison of Techniques," *Reliability Engineering and System Safety*. Vol. 65, no. 2, pp. 147-185.

31. Blower, S.M. and H. Dowlatabadi. 1994. "Sensitivity and Uncertainty Analysis of Complex Models of Disease Transmission: an HIV Model, as an Example," *International Statistical Review*. Vol. 62, no. 2, pp. 229-243.

32. Saltelli, A., T.H. Andres, and T. Homma. 1993. "Sensitivity Analysis of Model Output. An Investigation of New Techniques," *Computational Statistics and Data Analysis*. Vol. 15, no. 2, pp. 445-460.

33. Iman, R.L. 1992. "Uncertainty and Sensitivity Analysis for Computer Modeling Applications," *Reliability Technology - 1992, The Winter Annual Meeting of the American Society of Mechanical Engineers, Anaheim, California, November 8-13, 1992*. Eds. T.A. Cruse. Vol. 28, pp. 153-168. New York, NY: American Society of Mechanical Engineers, Aerospace Division.

34. Saltelli, A. and J. Marivoet. 1990. "Non-Parametric Statistics in Sensitivity Analysis for Model Output. A Comparison of Selected Techniques," *Reliability Engineering and System Safety*. Vol. 28, no. 2, pp. 229-253.

35. Iman, R.L., J.C. Helton, and J.E. Campbell. 1981. "An Approach to Sensitivity Analysis of Computer Models, Part 1. Introduction, Input Variable Selection and Preliminary Variable Assessment," *Journal of Quality Technology*. Vol. 13, no. 3, pp. 174-183.

36. Iman, R.L., J.C. Helton, and J.E. Campbell. 1981. "An Approach to Sensitivity Analysis of Computer Models, Part 2. Ranking of Input Variables, Response Surface Validation, Distribution Effect and Technique Synopsis," *Journal of Quality Technology*. Vol. 13, no. 4, pp. 232-240.

37. Iman, R.L. and W.J. Conover. 1980. "Small Sample Sensitivity Analysis Techniques for Computer Models, with an Application to Risk Assessment," *Communications in Statistics: Theory and Methods*. Vol. A9, no. 17, pp. 1749-1842.

38. McKay, M.D., R.J. Beckman, and W.J. Conover. 1979. "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code," *Technometrics*. Vol. 21, no. 2, pp. 239-245.

39. Li, G., C. Rosenthal, and H. Rabitz. 2001. "High-Dimensional Model Representations," *The Journal of Physical Chemistry*. Vol. 105, no. 33, pp. 7765-7777.

40. Rabitz, H. and O.F. Alis. 1999. "General Foundations of High-Dimensional Model Representations," *Journal of Mathematical Chemistry*. Vol. 25, no. 2-3, pp. 197-233.

41. Saltelli, A., S. Tarantola, and K.P.-S. Chan. 1999. "A Quantitative Model-Independent Method for Global Sensitivity Analysis of Model Output," *Technometrics*. Vol. 41, no. 1, pp. 39-56.

42. Sobol', I.M. 1993. "Sensitivity Estimates for Nonlinear Mathematical Models," *Mathematical Modeling & Computational Experiment*. Vol. 1, no. 4, pp. 407-414.

43. Cukier, R.I., H.B. Levine, and K.E. Shuler. 1978. "Nonlinear Sensitivity Analysis of Multiparameter Model Systems," *Journal of Computational Physics*. Vol. 26, no. 1, pp. 1-42.

44. Saltelli, A., M. Ratto, S. Tarantola, and F. Campolongo. 2005. "Sensitivity Analysis for Chemical Models," *Chemical Reviews*. Vol. 105, no. 7, pp. 2811-2828.

45. Ionescu-Bujor, M. and D.G. Cacuci. 2004. "A Comparative Review of Sensitivity and Uncertainty Analysis of Large-Scale Systems--I: Deterministic Methods," *Nuclear Science and Engineering*. Vol. 147, no. 3, pp. 189-2003.

46. Cacuci, D.G. and M. Ionescu-Bujor. 2004. "A Comparative Review of Sensitivity and Uncertainty Analysis of Large-Scale Systems--II: Statistical Methods," *Nuclear Science and Engineering*. Vol. 147, no. 3, pp. 204-217.

47. Frey, H.C. and S.R. Patil. 2002. "Identification and Review of Sensitivity Analysis Methods," *Risk Analysis*. Vol. 22, no. 3, pp. 553-578.

48. Saltelli, A., K. Chan, and E.M. Scott (eds). 2000. *Sensitivity Analysis*. New York, NY: Wiley.

49. Hamby, D.M. 1994. "A Review of Techniques for Parameter Sensitivity Analysis of Environmental Models," *Environmental Monitoring and Assessment*. Vol. 32, no. 2, pp. 135-154.

50. Helton, J.C. 1993. "Uncertainty and Sensitivity Analysis Techniques for Use in Performance Assessment for Radioactive Waste Disposal," *Reliability Engineering and System Safety*. Vol. 42, no. 2-3, pp. 327-367.

51. Ronen, Y. 1988. *Uncertainty Analysis.*, Boca Raton, FL: CRC Press, Inc.

52. Iman, R.L. and J.C. Helton. 1988. "An Investigation of Uncertainty and Sensitivity Analysis Techniques for Computer Models," *Risk Analysis*. Vol. 8, no. 1, pp. 71-90.

53. Vaughn, P., J.E. Bean, J.C. Helton, M.E. Lord, R.J. MacKinnon, and J.D. Schreiber. 2000. "Representation of Two-Phase Flow in the Vicinity of the Repository in the 1996 Performance Assessment for the Waste Isolation Pilot Plant," *Reliability Engineering and System Safety*. Vol. 69, no. 1-3, pp. 205-226.

54. Helton, J.C. and R.J. Breeding. 1993. "Calculation of Reactor Accident Safety Goals," *Reliability Engineering and System Safety*. Vol. 39, no. 2, pp. 129-158.

55. Breeding, R.J., J.C. Helton, E.D. Gorham, and F.T. Harper. 1992. "Summary Description of the Methods Used in the Probabilistic Risk Assessments for NUREG-1150," *Nuclear Engineering and Design*. Vol. 135, no. 1, pp. 1-27.

56. Helton, J.C. and M.G. Marietta. 2000. "Special Issue: The 1996 Performance Assessment for the Waste Isolation Pilot Plant," *Reliability Engineering and System Safety*. Vol. 69, no. 1-3, pp. 1-451.

57. Helton, J.C., D.R. Anderson, H.-N. Jow, M.G. Marietta, and G. Basabilvazo. 1999. "Performance Assessment in Support of the 1996 Compliance Certification Application for the Waste Isolation Pilot Plant," *Risk Analysis*. Vol. 19, no. 5, pp. 959 - 986.

58. Garthwaite, P.H., J.B. Kadane, and A. O'Hagan. 2005. "Statistical Methods for Eliciting Probability Distributions," *Journal of the American Statistical Association*. Vol. 100, no. 470, pp. 680-700.

59. Cooke, R.M. and L.H.J. Goossens. 2004. "Expert Judgement Elicitation for Risk Assessment of Critical Infrastructures," *Journal of Risk Research*. Vol. 7, no. 6, pp. 643-656.

60. Ayyub, B.M. 2001. *Elicitation of Expert Opinions for Uncertainty and Risks*, Boca Raton, FL: CRC Press.

61. McKay, M. and M. Meyer. 2000. "Critique of and Limitations on the use of Expert Judgements in Accident Consequence Uncertainty Analysis," *Radiation Protection Dosimetry*. Vol. 90, no. 3, pp. 325-330.

62. Goossens, L.H.J., F.T. Harper, B.C.P. Kraan, and H. Metivier. 2000. "Expert Judgement for a Probabilistic Accident Consequence Uncertainty Analysis," *Radiation Protection Dosimetry*. Vol. 90, no. 3, pp. 295-301.

63. Budnitz, R.J., G. Apostolakis, D.M. Boore, L.S. Cluff, K.J. Coppersmith, C.A. Cornell, and P.A. Morris. 1998. "Use of Technical Expert Panels: Applications to Probabilistic Seismic Hazard Analysis," *Risk Analysis*. Vol. 18, no. 4, pp. 463-469.

64. Goossens, L.H.J. and F.T. Harper. 1998. "Joint EC/USNRC Expert Judgement Driven Radiological Protection Uncertainty Analysis," *Journal of Radiological Protection*. Vol. 18, no. 4, pp. 249-264.

65. Siu, N.O. and D.L. Kelly. 1998. "Bayesian Parameter Estimation in Probabilistic Risk Assessment," *Reliability Engineering and System Safety*. Vol. 62, no. 1-2, pp. 89-116.

66. Evans, J.S., G.M. Gray, R.L. Sielken Jr., A.E. Smith, C. Valdez-Flores, and J.D. Graham. 1994. "Use of Probabilistic Expert Judgement in Uncertainty Analysis of Carcinogenic Potency," *Regulatory Toxicology and Pharmacology*. Vol. 20, no. 1, pt. 1, pp. 15-36.

67. Thorne, M.C. 1993. "The Use of Expert Opinion in Formulating Conceptual Models of Underground Disposal Systems and the Treatment of Associated Bias," *Reliability Engineering and System Safety*. Vol. 42, no. 2-3, pp. 161-180.

68. Chhibber, S., G. Apostolakis, and D. Okrent. 1992. "A Taxonomy of Issues Related to the Use of Expert Judgments in Probabilistic Safety Studies," *Reliability Engineering and System Safety*. Vol. 38, no. 1-2, pp. 27-45.

69. Otway, H. and D.V. Winterfeldt. 1992. "Expert Judgement in Risk Analysis and Management: Process, Context, and Pitfalls," *Risk Analysis*. Vol. 12, no. 1, pp. 83-93.

70. Thorne, M.C. and M.M.R. Williams. 1992. "A Review of Expert Judgement Techniques with Reference to Nuclear Safety," *Progress in Nuclear Safety*. Vol. 27, no. 2-3, pp. 83-254.

71. Cooke, R.M. 1991. *Experts in Uncertainty: Opinion and Subjective Probability in Science*. Oxford; New York: Oxford University Press.

72. Meyer, M.A. and J.M. Booker. 1991. *Eliciting and Analyzing Expert Judgment: A Practical Guide*. New York, NY: Academic Press.

73. Hora, S.C. and R.L. Iman. 1989. "Expert Opinion in Risk Analysis: The NUREG-1150 Methodology," *Nuclear Science and Engineering*. Vol. 102, no. 4, pp. 323-331.

74. Helton, J.C. 1997. "Uncertainty and Sensitivity Analysis in the Presence of Stochastic and Subjective Uncertainty," *Journal of Statistical Computation and Simulation*. Vol. 57, no. 1-4, pp. 3-76.

75. Helton, J.C. and D.E. Burmaster. 1996. "Guest Editorial: Treatment of Aleatory and Epistemic Uncertainty in Performance Assessments for Complex Systems," *Reliability Engineering and System Safety*. Vol. 54, no. 2-3, pp. 91-94.

76. Paté-Cornell, M.E. 1996. "Uncertainties in Risk Analysis: Six Levels of Treatment," *Reliability Engineering and System Safety*. Vol. 54, no. 2-3, pp. 95-111.

77. Winkler, R.L. 1996. "Uncertainty in Probabilistic Risk Assessment," *Reliability Engineering and System Safety*. Vol. 54, no. 2-3, pp. 127-132.

78. Hoffman, F.O. and J.S. Hammonds. 1994. "Propagation of Uncertainty in Risk Assessments: The Need to Distinguish Between Uncertainty Due to Lack of Knowledge and Uncertainty Due to Variability," *Risk Analysis*. Vol. 14, no. 5, pp. 707-712.

79. Helton, J.C. 1994. "Treatment of Uncertainty in Performance Assessments for Complex Systems," *Risk Analysis*. Vol. 14, no. 4, pp. 483-511.

80. Apostolakis, G. 1990. "The Concept of Probability in Safety Assessments of Technological Systems," *Science*. Vol. 250, no. 4986, pp. 1359-1364.

81. Haan, C.T. 1989. "Parametric Uncertainty in Hydrologic Modeling," *Transactions of the ASAE*. Vol. 32, no. 1, pp. 137-146.

82. Parry, G.W. and P.W. Winter. 1981. "Characterization and Evaluation of Uncertainty in Probabilistic Risk Analysis," *Nuclear Safety*. Vol. 22, no. 1, pp. 28-42.

83. Kaplan, S. and B.J. Garrick. 1981. "On the Quantitative Definition of Risk," *Risk Analysis*. Vol. 1, no. 1, pp. 11-27.

84. Kleijnen, J.P.C. and J.C. Helton. 1999. "Statistical Analyses of Scatterplots to Identify Important Factors in Large-Scale Simulations, 2: Robustness of Techniques," *Reliability Engineering and System Safety*. Vol. 65, no. 2, pp. 187-197.

85. Assunçao, R. 1994. "Testing Spatial Randomness by Means of Angles," *Biometrics*. Vol. 50, pp. 531-537.

86. Ripley, B.D. 1987. "Spatial Point Pattern Analysis in Ecology," *Developments in Numerical Ecology. NATO ASI Series, Series G: Ecological Sciences*. Vol. 14. Eds. P. Legendre and L. Legendre. Berlin; New York: Springer-Verlag, pp. 407-430.

87. Zeng, G. and R.C. Dubes. 1985. "A Comparison of Tests for Randomness," *Pattern Recognition*. Vol. 18, no. 2, pp. 191-198.

88. Diggle, P.J. and T.F. Cox. 1983. "Some Distance-Based Tests of Independence for Sparsely-Sampled Multivariate Spatial Point Patterns," *International Statistical Review*. Vol. 51, no. 1, pp. 11-23.

89. Byth, K. 1982. "On Robust Distance-Based Intensity Estimators," *Biometrics*. Vol. 38, no. 1, pp. 127-135.

90. Byth, K. and B.D. Ripley. 1980. "On Sampling Spatial Patterns by Distance Methods," *Biometrics*. Vol. 36, no. 2, pp. 279-284.

91. Ripley, B.D. 1979. "Tests of Randomness for Spatial Point Patterns," *Journal of the Royal Statistical Society*. Vol. 41, no. 3, pp. 368-374.

92. Diggle, P.J. 1979. "Statistical Methods for Spatial Point Patterns in Ecology," *Spatial and Temporal Analysis in Ecology*. Ed. R.M. Cormack and J.K. Ord. Fairfield, MD: International Cooperative Pub. House. 95-150.

93. Diggle, P.J. 1979. "On Parameter Estimation and Goodness-of-Fit Testing for Spatial Point Patterns," *Biometrics*. Vol. 35, no. 1, pp. 87-101.

94. Besag, J. and P.J. Diggle. 1977. "Simple Monte Carlo Tests for Spatial Pattern," *Applied Statistics*. Vol. 26, no. 3, pp. 327-333.

95. Diggle, P.J., J. Besag, and J.T. Gleaves. 1976. "Statistical Analysis of Spatial Point Patterns by Means of Distance Methods," *Biometrics*. Vol. 32, pp. 659-667.

96. Cox, T.F. and T. Lewis. 1976. "A Conditioned Distance Ratio Method for Analyzing Spatial Patterns," *Biometrika*. Vol. 63, no. 3, pp. 483-491.

97. Holgate, P. 1972. "The Use of Distance Methods for the Analysis of Spatial Distribution of Points," *Stochastic Point Processes: Statistical Analysis, Theory, and Applications*. Eds. P.A.W. Lewis. New York, NY: Wiley-Interscience. 122-135.

98. Holgate, P. 1965. "Tests of Randomness Based on Distance Methods," *Biometrika*. Vol. 52, no. 3-4, pp. 345-353.

99. Garvey, J.E., E.A. Marschall, and R.A. Wright. 1998. "From Star Charts to Stoneflies: Detecting Relationships in Continuous Bivariate Data," *Ecology*. Vol. 79, no. 2, pp. 442-447.

100. Fasano, G. and A. Franceschini. 1987. "A Multidimensional Version of the Kolmogorov-Smirnov Test," *Monthly Notices of the Royal Astronomical Society*. Vol. 225, no. 1, pp. 155-170.

101. Gosset, E. 1987. "A 3-Dimensional Extended Kolmogorov-Smirnov Test as a Useful Tool in Astronomy," *Astronomy and Astrophysics*. Vol. 188, no. 1, pp. 258-264.

102. Peacock, J.A. 1983. "Two-Dimensional Goodness-Of-Fit Testing in Astronomy," *Monthly Notices of the Royal Astronomical Society*. Vol. 202, no. 2, pp. 615-627.

103. Helton, J.C., F.J. Davis, and J.D. Johnson. 2005. "A Comparison of Uncertainty and Sensitivity Analysis Results Obtained with Random and Latin Hypercube Sampling," *Reliability Engineering and System Safety*. Vol. 89, no. 3, pp. 305-330.

104. Hora, S.C. and J.C. Helton. 2003. "A Distribution-Free Test for the Relationship Between Model Input and Output when Using Latin Hypercube Sampling," *Reliability Engineering and System Safety*. Vol. 79, no. 3, pp. 333-339.

105. Mokhtari, A., H.C. Frey, and L.A. Jaykus. 2006. "Application of Classification and Regression Trees for Sensitivity Analysis of the Escherichia coli O157: H7 Food Safety Process Risk Model," *Journal of Food Protection*. Vol. 69, no. 3, pp. 609-618.

106. Mishra, S., N.E. Deeds, and B.S. RamaRao. 2003. "Application of Classification Trees in the Sensitivity Analysis of Probabilistic Model Results," *Reliability Engineering and System Safety*. Vol. 79, no. 2, pp. 123-129.

107. Maindonald, J. and J. Braun. 2003. *Data Analysis and Graphics Using R*. Cambridge: Cambridge University Press.

108. Insightful Corporation. 2001. *S-PLUS 6 for Windows Guide to Statistics, Volume 1*, Seattle, WA: Insightful Corporation.

109. Myers, R.H. 1990. *Classical and Modern Regression with Applications*. 2nd ed. Boston: Duxbury Press.

110. Draper, N.R. and H. Smith. 1981. *Applied Regression Analysis*. 2nd ed. New York, NY: John Wiley & Sons.

111. Daniel, C., F.S. Wood, and J.W. Gorman. 1980. *Fitting Equations to Data: Computer Analysis of Multifactor Data*. 2nd ed. New York, NY: John Wiley & Sons.

112. Seber, G.A. 1977. *Linear Regression Analysis*. New York, NY: John Wiley & Sons.

113. Neter, J. and W. Wasserman. 1974. *Applied Linear Statistical Models: Regression, Analysis of Variance, and Experimental Designs*. Homewood, IL: Richard D. Irwin.

114. Iman, R.L. and W.J. Conover. 1979. "The Use of the Rank Transform in Regression," *Technometrics*. Vol. 21, no. 4, pp. 499-509.

115. Helton, J.C. 1999. "Uncertainty and Sensitivity Analysis in Performance Assessment for the Waste Isolation Pilot Plant," *Computer Physics Communications*. Vol. 117, no. 1-2, pp. 156-180.

116. Helton, J.C., J.D. Johnson, M.D. McKay, A.W. Shiver, and J.L. Sprung. 1995. "Robustness of an Uncertainty and Sensitivity Analysis of Early Exposure Results with the MACCS Reactor Accident Consequence Model," *Reliability Engineering and System Safety*. Vol. 48, no. 2, pp. 129-148.

117. Iman, R.L. and J.C. Helton. 1991. "The Repeatability of Uncertainty and Sensitivity Analyses for Complex Probabilistic Risk Assessments," *Risk Analysis*. Vol. 11, no. 4, pp. 591-606.

118. Myers, R.H. and D.C. Montgomery. 1995. *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*. New York, NY: Wiley.

119. Huet, S., A. Bouvier, M.-A. Gruet, and E. Jolivet. 2003. *Statistical Tools for Nonlinear Regression: A Practical Guide with S-PLUS and R Examples*. New York, NY: Springer-Verlag.

120. Hastie, T.J. and R.J. Tibshirani. 1990. *Generalized Additive Models*. London: Chapman & Hall.

121. Simonoff, J.S. 1996. *Smoothing Methods in Statistics*. New York, NY: Springer-Verlag.

122. Bowman, A.W. and A. Azzalini. 1997. *Applied Smoothing Techniques for Data Analysis*. Oxford: Clarendon.

123. Ruppert, D., M.P. Ward, and R.J. Carroll. 2003. *Semiparametric Regression*. Cambridge: Cambridge University Press.

124. Cleveland, W.S. 1979. "Robust Locally Weighted Regression and Smoothing Scatterplots," *Journal of the American Statistical Association*. Vol. 14, no. 368, pp. 829-836.

125. Schimek, M.G. 2000. *Smoothing and Regression: Approaches, Computation, and Application*. New York, NY: Wiley.

126. Chambers, J.M. and T.J. Hastie. 1992. *Statistical Models in S*. Pacific Grove, CA: Wadsworth & Brooks.

127. Hocking, R.R. 1996. *Methods and Applications of Linear Models*. New York, NY: Wiley.

128. Christensen, R. 1996. *Plane Answers to Complex Questions*. New York, NY: Springer-Verlag.

129. Allen, D.M. 1971. *The Prediction Sum of Squares as a Criterion for Selecting Predictor Variables*, 23. Lexington: University of Kentucky, Department of Statistics.

130. Friedman, J.H. and W. Stuetzle. 1981. "Projection Pursuit Regression," *Journal of the American Statistical Association*. Vol. 76, no. 376, pp. 817-823.

131. Breiman, L., J.H. Friedman, R.A. Olshen, and C.J. Stone. 1984. *Classification and Regression Trees*. Belmont, CA: Wadsworth Intl.

132. Rousseeuw, P.J. and A.M. Leroy. 1987. *Robust Regression and Outlier Detection*. New York, NY: Wiley.

133. Hastie, T., R. Tibshirani, and J. Friedman. 2001. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York, NY: Springer.

134. Press, W.H., S.A. Teukolsky, W.T. Vetterlings, and B.P. Flannery. 1992. *Numerical Recipes: the Art of Scientific Computing (FORTRAN Version)*. 2nd ed. Cambridge: Cambridge University Press.

135. Hogg, R.V. and A.T. Craig. 1978. *Introduction to Mathematical Statistics, 4th Ed.* New York, NY: Macmillan.

136. Breeding, R.J., J.C. Helton, W.B. Murfin, L.N. Smith, J.D. Johnson, H.-N. Jow, and A.W. Shiver. 1992. "The NUREG-1150 Probabilistic Risk Assessment for the Surry Nuclear Power Station," *Nuclear Engineering and Design*. Vol. 135, no. 1, pp. 29-59.

137. Payne, A.C., Jr., R.J. Breeding, J.C. Helton, L.N. Smith, J.D. Johnson, H.-N. Jow, and A.W. Shiver. 1992. "The NUREG-1150 Probabilistic Risk Assessment for the Peach Bottom Atomic Power Station," *Nuclear Engineering and Design*. Vol. 135, no. 1, pp. 61-94.

138. Gregory, J.J., R.J. Breeding, J.C. Helton, W.B. Murfin, S.J. Higgins, and A.W. Shiver. 1992. "The NUREG-1150 Probabilistic Risk Assessment for the Sequoyah Nuclear Plant," *Nuclear Engineering and Design*. Vol. 135, no. 1, pp. 92-115.

139. Brown, T.D., R.J. Breeding, J.C. Helton, H.-N. Jow, S.J. Higgins, and A.W. Shiver. 1992. "The NUREG-1150 Probabilistic Risk Assessment for the Grand Gulf Nuclear Station," *Nuclear Engineering and Design*. Vol. 135, no. 1, pp. 117-137.

140. Helton, J.C., M.-A. Martell, and M.S. Tierney. 2000. "Characterization of Subjective Uncertainty in the 1996 Performance Assessment for the Waste Isolation Pilot Plant," *Reliability Engineering and System Safety*. Vol. 69, no. 1-3, pp. 191-204.

141. CRWMS M&O (Civilian Radioactive Waste Management System Management and Operating Contractor). 2000. *Total System Performance Assessment for the Site Recommendation*, TDR-WIS-PA-000001 REV 00. Las Vegas, NV: CRWMS M&O.

142. CRWMS M&O (Civilian Radioactive Waste Management System Management and Operating Contractor). 2000. *Total System Performance Assessment (TSPA) Model for Site Recommendation*, MDL-WIS-PA-000002 REV 00. Las Vegas, NV: CRWMS M&O.

143. Campolongo, F., A. Saltelli, T. Sorensen, and S. Tarantola. 2000. "Hitchhiker's Guide to Sensitivity Analysis," *Sensitivity Analysis*. Ed. A. Saltelli, K. Chan, and M. Scott. New York, NY: John Wiley & Sons.

144. Iman, R.L. and W.J. Conover. 1987. "A Measure of Top-Down Correlation," *Technometrics*. Vol. 29, no. 3, pp. 351-357.

145. Conover, W.J. 1980. *Practical Nonparametric Statistics*. 2nd ed. New York, NY: John Wiley & Sons.

146. Helton, J.C., J.E. Bean, K. Economy, J.W. Garner, R.J. MacKinnon, J. Miller, J.D. Schreiber, and P. Vaughn. 2000. "Uncertainty and Sensitivity Analysis for Two-Phase Flow in the Vicinity of the Repository in the 1996 Performance Assessment for the Waste Isolation Pilot Plant: Undisturbed Conditions," *Reliability Engineering and System Safety*. Vol. 69, no. 1-3, pp. 227-261.

147. Helton, J.C., J.E. Bean, K. Economy, J.W. Garner, R.J. MacKinnon, J. Miller, J.D. Schreiber, and P. Vaughn. 2000. "Uncertainty and Sensitivity Analysis for Two-Phase Flow in the Vicinity of the Repository in the 1996 Performance Assessment for the Waste Isolation Pilot Plant: Disturbed Conditions," *Reliability Engineering and System Safety*. Vol. 69, no. 1-3, pp. 263-304.

148. Chan, K., S. Tarantola, and A. Saltelli. 2000. "Variance-Based Methods," *Sensitivity Analysis*. Ed. A. Saltelli, K. Chan, and E.M.S. Scott. New York, NY: Wiley. 167-197.

149. Yao, Q. and H. Tong. 2000. "Nonparametric Estimation of Ratios of Noise to Signal in Stochastic Regression," *Statistica Sinica*. Vol. 10, pp. 751-770.

150. Doskum, K. and A. Samaroz. 1995. "Nonparametric Estimation of Global Functionals and a Measure of the Explanatory Power of Covariates in Regression," *Annals of Statistics*. Vol. 23, no. 5, pp. 1443-1473.

151. Iman, R.L. and W.J. Conover. 1982. "A Distribution-Free Approach to Inducing Rank Correlation Among Input Variables," *Communications in Statistics: Simulation and Computation*. Vol. B11, no. 3, pp. 311-334.

152. Iman, R.L. and J.M. Davenport. 1982. "Rank Correlation Plots for Use with Correlated Input Variables," *Communications in Statistics: Simulation and Computation*. Vol. B11, no. 3, pp. 335-360.

153. Breiman, L. and J.H. Friedman. 1985. "Estimating Optimal Transformers for Multiple Regression and Correlation (with discussion)," *Journal of American Statistical Association*. Vol. 80, pp. 580-619.

154. Tibshirani, R. 1988. "Estimating Optimal Transformations for Regression via Additivity and Variance Stabilization," *Journal of American Statistical Association*. Vol. 83, pp. 394-405.

155. Friedman, J.H. 1991. "Multivariate Adaptive Regression Splines (with discussion)," *The Annals of Statistics*. Vol. 19, no. 1, pp. 1-141.

156. Chaudhuri, P., M. Huang, W. Loh, and R. Yao. 1994. "Piecewise Polynomial Regression Trees," *Statistica Sinica*. Vol. 4, pp. 143-167.

157. O'Hagan, A. 2006. "Bayesian Analysis of Computer Code Outputs: A Tutorial," *Reliability Engineering and System Safety*. Vol. 91, no. 10-11, pp. 1290-1300.

158. Kennedy, M.C., C.W. Anderson, S. Conti, and A. O'Hagan. 2006. "Case Studies in Gaussian Process Modelling of Computer Codes," *Reliability Engineering and System Safety*. Vol. 91, no. 10-11, pp. 1301-1309.

159. Oakley, J.E. and A. O'Hagan. 2004. "Probabilistic Sensitivity Analysis of Complex Models: A Bayesian Approach," *Journal of The Royal Statistical Society*. Vol. B66, pp. 751-769.

Appendix A: R Code

This document is a supplement to the Sand Report entitled "Multiple Predictor Smoothing Methods for Sensitivity Analysis". Here we list the R functions and scripts used to generate the analyses in the paper. Although these functions were written for the R language, they should work in S-Plus as well. R is an open source language that can be downloaded at `http://www.r-project.org/`.

1. Example R script

The following R Script will perform a sensitivity analysis on examples 1-4 from the paper. This can be executed by putting all of the functions listed in Section 3 in a file called *functions.R* and this script in a file called *run_examples.R*. Assuming both files are in the present working directory, typing the command

```
> source('run_examples.R')
```

at the R prompt will perform the sensitivity analysis and output the results to the files *y1_table.out*, *y2_table.out*, *y3_table.out*, and *y4_table.out*. The content of the file *y1_table.out* that is generated is given in Section 2. Equivalently one could copy and paste the lines of the following script at the R prompt in pieces or all at once. The script should take anywhere from 10 minutes to an hour to run depending on the computer. You may need to install the R packages `locfit` and `gam` if they are not already installed. This can be accomplished by typing the commands

```
> install.packages('locfit')
> install.packages('gam')
```

at the R prompt.

Notice that the first few lines of the *run_examples.R* script randomly generates data to be used for the sensitivity analysis. If we were to perform a sensitivity analysis on real data, we would simply replace lines 5-23 with a statement that imports our data. If the data is stored in a tab delimited ascii file called *real_data.txt* then the following command will read the file into the data frame `sens.dat`.

```
> sens.dat <- read.table('real_data.txt',header=T)
```

Type `?read.table` at the R prompt for more help on how to read in the data. The rest of this section is the *run_examples.R* script.

```
#####################  run_examples.R  ############################

source('functions.R')
library('locfit')
library('gam')

set.seed(12)

a <- c(0,1,4.5,9,99,99,99,99)
```

```
X <- matrix(runif(3000),ncol=10)
Y <- matrix(nrow=300,ncol=4)
Y[,1] <- 5*X[,1]+(5*X[,2])^4
Y[,2] <- (X[,2]+.5)^4/(X[,1]+.5)^2
ans <- rep(1,300)
for(j in 1:8){
  ans <- ans*(abs(4*X[,j]-2)+a[j])/(1+a[j])
}
Y[,3] <- ans
Y[,4] <- sin(2*pi*X[,1]-pi) + 7*(sin(2*pi*X[,2]-pi))^2 +
         0.1*(2*pi*X[,3]-pi)^4*sin(2*pi*X[,1]-pi)

sens.dat <- as.data.frame(cbind(X,Y))
names(sens.dat) <- c(paste("x",1:10,sep=''),paste("y",1:4,sep=''))
sens.dat1 <<- sens.dat[1:100,]
sens.dat2 <<- sens.dat[101:200,]
sens.dat3 <<- sens.dat[201:300,]

x.loc <- 1:10
y.loc <- 11:14


################# Perform SA on entire sample of size 300  ###################

sim.true <<- true.order()


sim.reg <<- sensitivity(sens.dat, x.loc, y.loc, surface='reg',
                        smooth='s', span='cv', df='cv', n.terms='cv',
                        summary=F, maxterms=20, crit='pval', alpha=.02,
                        inc.press=T, CV=T)

sim.rreg <<- sensitivity(sens.dat, x.loc, y.loc, surface='rank',
                        smooth='s', span='cv', df='cv', n.terms='cv',
                        summary=F, maxterms=20, crit='pval', alpha=.02,
                        inc.press=T, CV=T)

sim.rsreg <<- sensitivity(sens.dat, x.loc, y.loc, surface='rs.reg',
                        smooth='s', span='cv', df='cv', n.terms='cv',
                        summary=F, maxterms=20, crit='pval', alpha=.02,
                        inc.press=T, CV=T)

sim.addreg <<- sensitivity(sens.dat, x.loc, y.loc, surface='add.reg',
                        smooth='s', span='cv', df='cv', n.terms='cv',
                        summary=F, maxterms=20, crit='pval', alpha=.02,
                        inc.press=F, CV=F)

sim.locreg <<- sensitivity(sens.dat, x.loc, y.loc, surface='loc.reg',
                        smooth='s', span='cv', df='cv', n.terms='cv',
                        summary=F, maxterms=20, crit='pval', alpha=.02,
                        inc.press=T, CV=T)

sim.ppreg <<- sensitivity(sens.dat, x.loc, y.loc, surface='ppr',
                        smooth='s', span='cv', df='cv', n.terms='cv',
                        summary=F, maxterms=20, crit='pval', alpha=.02,
                        inc.press=F, CV=F)

sim.rpreg <<- sensitivity(sens.dat, x.loc, y.loc, surface='rpr',
                         nsplit='gcv', space=3, summary=F, maxterms=20,
                         crit='pval', alpha=.02, inc.press=F, CV=F)


sim.srdrcc <<- srd.sens(sens.dat, x.loc, y.loc, alpha=.02, summary=F)
```

```
################ Perform SA on the 3 subsamples of size 100  #################


sim.reg1 <<- sensitivity(sens.dat1, x.loc, y.loc, surface='reg',
                         smooth='s', span='cv', df='cv', n.terms='cv',
                         summary=F, maxterms=20, crit='pval', alpha=.02, CV=F)

sim.rreg1 <<- sensitivity(sens.dat1, x.loc, y.loc, surface='rank',
                         smooth='s', span='cv', df='cv', n.terms='cv',
                         summary=F, maxterms=20, crit='pval', alpha=.02, CV=F)

sim.rsreg1 <<- sensitivity(sens.dat1, x.loc, y.loc, surface='rs.reg',
                         smooth='s', span='cv', df='cv', n.terms='cv',
                         summary=F, maxterms=20, crit='pval', alpha=.02, CV=F)

sim.addreg1 <<- sensitivity(sens.dat1, x.loc, y.loc, surface='add.reg',
                         smooth='s', span='cv', df='cv', n.terms='cv',
                         summary=F, maxterms=20, crit='pval', alpha=.02, CV=F)

sim.locreg1 <<- sensitivity(sens.dat1, x.loc, y.loc, surface='loc.reg',
                         smooth='s', span='cv', df='cv', n.terms='cv',
                         summary=F, maxterms=20, crit='pval', alpha=.02, CV=F)

sim.ppreg1 <<- sensitivity(sens.dat1, x.loc, y.loc, surface='ppr',
                         smooth='s', span='cv', df='cv', n.terms='cv',
                         summary=F, maxterms=20, crit='pval', alpha=.02, CV=F)

sim.rpreg1 <<- sensitivity(sens.dat1, x.loc, y.loc, surface='rpr',
                           nsplit='gcv', space=2, summary=F, maxterms=20,
                           crit='pval', alpha=.02, CV=F)

sim.srdrcc1 <<- srd.sens(sens.dat1, x.loc, y.loc, alpha=.02, summary=F)

##################


sim.reg2 <<- sensitivity(sens.dat2, x.loc, y.loc, surface='reg',
                         smooth='s', span='cv', df='cv', n.terms='cv',
                         summary=F, maxterms=20, crit='pval', alpha=.02, CV=F)

sim.rreg2 <<- sensitivity(sens.dat2, x.loc, y.loc, surface='rank',
                         smooth='s', span='cv', df='cv', n.terms='cv',
                         summary=F, maxterms=20, crit='pval', alpha=.02, CV=F)

sim.rsreg2 <<- sensitivity(sens.dat2, x.loc, y.loc, surface='rs.reg',
                         smooth='s', span='cv', df='cv', n.terms='cv',
                         summary=F, maxterms=20, crit='pval', alpha=.02, CV=F)

sim.addreg2 <<- sensitivity(sens.dat2, x.loc, y.loc, surface='add.reg',
                         smooth='s', span='cv', df='cv', n.terms='cv',
                         summary=F, maxterms=20, crit='pval', alpha=.02, CV=F)

sim.locreg2 <<- sensitivity(sens.dat2, x.loc, y.loc, surface='loc.reg',
                         smooth='s', span='cv', df='cv', n.terms='cv',
                         summary=F, maxterms=20, crit='pval', alpha=.02, CV=F)

sim.ppreg2 <<- sensitivity(sens.dat2, x.loc, y.loc, surface='ppr',
                         smooth='s', span='cv', df='cv', n.terms='cv',
                         summary=F, maxterms=20, crit='pval', alpha=.02, CV=F)

sim.rpreg2 <<- sensitivity(sens.dat2, x.loc, y.loc, surface='rpr',
                           nsplit='gcv', space=2, summary=F, maxterms=20,
                           crit='pval', alpha=.02, CV=F)
```

```
sim.srdrcc2 <<- srd.sens(sens.dat2, x.loc, y.loc, alpha=.02, summary=F)

#################


sim.reg3 <<- sensitivity(sens.dat3, x.loc, y.loc, surface='reg',
                         smooth='s', span='cv', df='cv', n.terms='cv',
                         summary=F, maxterms=20, crit='pval', alpha=.02, CV=F)

sim.rreg3 <<- sensitivity(sens.dat3, x.loc, y.loc, surface='rank',
                         smooth='s', span='cv', df='cv', n.terms='cv',
                         summary=F, maxterms=20, crit='pval', alpha=.02, CV=F)

sim.rsreg3 <<- sensitivity(sens.dat3, x.loc, y.loc, surface='rs.reg',
                         smooth='s', span='cv', df='cv', n.terms='cv',
                         summary=F, maxterms=20, crit='pval', alpha=.02, CV=F)

sim.addreg3 <<- sensitivity(sens.dat3, x.loc, y.loc, surface='add.reg',
                         smooth='s', span='cv', df='cv', n.terms='cv',
                         summary=F, maxterms=20, crit='pval', alpha=.02, CV=F)

sim.locreg3 <<- sensitivity(sens.dat3, x.loc, y.loc, surface='loc.reg',
                         smooth='s', span='cv', df='cv', n.terms='cv',
                         summary=F, maxterms=20, crit='pval', alpha=.02, CV=F)

sim.ppreg3 <<- sensitivity(sens.dat3, x.loc, y.loc, surface='ppr',
                         smooth='s', span='cv', df='cv', n.terms='cv',
                         summary=F, maxterms=20, crit='pval', alpha=.02, CV=F)

sim.rpreg3 <<- sensitivity(sens.dat3, x.loc, y.loc, surface='rpr',
                           nsplit='gcv', space=2, summary=F, maxterms=20,
                           crit='pval', alpha=.02, CV=F)

sim.srdrcc3 <<- srd.sens(sens.dat3, x.loc, y.loc, alpha=.02, summary=F)


######################## Add TDCC to sensitivity objects ######################

sim.reg <<- tdcc.list(sim.reg1, sim.reg2,sim.reg3, nx=10,
                         alt.obj=sim.reg)
sim.rreg <<- tdcc.list(sim.rreg1, sim.rreg2,sim.rreg3, nx=10,
                         alt.obj=sim.rreg)
sim.rsreg <<- tdcc.list(sim.rsreg1, sim.rsreg2,sim.rsreg3, nx=10,
                         alt.obj=sim.rsreg)
sim.addreg <<- tdcc.list(sim.addreg1, sim.addreg2,sim.addreg3, nx=10,
                         alt.obj=sim.addreg)
sim.locreg <<- tdcc.list(sim.locreg1, sim.locreg2,sim.locreg3, nx=10,
                         alt.obj=sim.locreg)
sim.ppreg <<- tdcc.list(sim.ppreg1, sim.ppreg2,sim.ppreg3, nx=10,
                        alt.obj=sim.ppreg)
sim.rpreg <<- tdcc.list(sim.rpreg1, sim.rpreg2,sim.rpreg3, nx=10,
                        alt.obj=sim.rpreg)
sim.srdrcc <<- tdcc.list(sim.srdrcc1, sim.srdrcc2,sim.srdrcc3, nx=10,
                         alt.obj=sim.srdrcc)


#################### Add TDCCw/true to sensitivity objects ###################

sim.reg <<- tdcc.list(sim.reg, sim.true, nx=10, wtrue=T, alt.obj=sim.reg)
sim.rreg <<- tdcc.list(sim.rreg, sim.true, nx=10, wtrue=T, alt.obj=sim.rreg)
sim.rsreg<<- tdcc.list(sim.rsreg, sim.true, nx=10, wtrue=T, alt.obj=sim.rsreg)
sim.addreg <<- tdcc.list(sim.addreg, sim.true, nx=10, wtrue=T,
                         alt.obj=sim.addreg)
```

```
sim.locreg <<- tdcc.list(sim.locreg, sim.true, nx=10, wtrue=T,
                         alt.obj=sim.locreg)
sim.ppreg<<- tdcc.list(sim.ppreg, sim.true, nx=10, wtrue=T, alt.obj=sim.ppreg)
sim.rpreg<<- tdcc.list(sim.rpreg, sim.true, nx=10, wtrue=T, alt.obj=sim.rpreg)
sim.srdrcc <<- tdcc.list(sim.srdrcc, sim.true, nx=10, wtrue=T,
                         alt.obj=sim.srdrcc)


###################### Create tables for each output ##########################

sens.tables(sim.reg, sim.rreg, sim.rsreg, sim.locreg, sim.ppreg,
            sim.rpreg, sim.addreg, sim.srdrcc, sim.true,
            ncol=1, Rsq=T, APS=T, CT=T, CTwT=T)
```

2. Example Output

The script from Section 1 will produce 4 output files, 1 file for each response variable in the sensitivity analysis. The contents that are written to *y1_table.out* are given below.

```
Var    R2          df     p-val        PRESS
             reg
x2     0.7552      1.0    0.0000       1.87E6
Rsq_A = 0.7544
PRS_A = 1.86E6
C_T = 1.0000, C_T w/true = 0.9294

             rank
x2     0.9774      1.0    0.0000       5.19E4
x1     0.9842      1.0    0.0000       3.66E4
Rsq_A = 0.9841
PRS_A = 3.64E4
C_T = 0.9744, C_T w/true = 1.0000

             rs.reg
x2     0.9789      2.0    0.0000       1.63E5
Rsq_A = 0.9787
PRS_A = 1.62E5
C_T = 1.0000, C_T w/true = 0.9294

             loc.reg
x2     0.9999      27.1   0.0000       6.74E2
Rsq_A = 0.9999
PRS_A = 6.79E2
C_T = 1.0000, C_T w/true = 0.9294

             ppr
x2     0.9999      13.2   0.0000       NA
x1     1.0000      14.4   0.0000       NA
x5     1.0000      25.3   0.0000       NA
x7     1.0000      -18.7  0.0009       NA
x4     1.0000      38.7   0.0000       NA
x10    1.0000      0.1    0.0000       NA
x6     1.0000      -1.9   0.0000       NA
x9     1.0000      37.8   0.0000       NA
Rsq_A = 1.0000
PRS_A = 2.00E0
C_T = 0.9097, C_T w/true = 0.9453

             rpr
x2     0.9999      38.0   0.0000       NA
```

```
x1    1.0000       115.0 0.0000       NA
Rsq_A = 1.0000
PRS_A = 1.20E0
C_T = 1.0000, C_T w/true = 1.0000

            add.reg
x2    0.9999       15.0  0.0000       NA
x1    1.0000       1.0   0.0000       NA
Rsq_A = 1.0000
PRS_A = 4.50E1
C_T = 1.0000, C_T w/true = 1.0000

            srd_rcc
x2    NA     NA     0.0000       NA
x1    NA     NA     0.0164       NA
Rsq_A = NA
PRS_A = NA
C_T = 0.9135, C_T w/true = 1.0000

            TRUE
x2    0.9999       NA     NA     NA
x1    0.0001       NA     NA     NA
Rsq_A = NA
PRS_A = NA
C_T = NA, C_T w/true = 1.0000
```

3. R functions

The following functions are used by the *run_examples.R* script given in Section 1.

```
############################  functions.R  ###################################


stand <- function(x){
# standardize x
  return((x-mean(x))/sd(x))
}

#############################################################################
#

stand.data <- function(X, Min=10E-6, Max=10E6){
# standardize columns if max|column| > Max or min|column| < Min
  X <- as.data.frame(X)
  for(i in 1:ncol(X)){
    if( (max(abs(X[,i])) > Max) || (min(abs(X[,i])) < Min) )
      X[,i] <- stand(X[,i])
  }
  return(X)
}

#############################################################################
#

rs.reg <- function(X, y, summary=T){
#calculate a response surface regression
  X <- as.matrix(X)
  n <- nrow(X)
  if(n != length(y))
    stop("y must have length nrow(X)")
  #first center the predictors
```

```
Xc <- X - matrix(colMeans(X),nrow(X),ncol(X),byrow=T)
nx <- ncol(X)
if(length(dimnames(X)[[2]]) == 0){
  dimnames(Xc)[[2]] <- list()
  dimnames(Xc)[[2]] <- paste("x", 1:nx, sep="")
}
Xr <- cbind(1,Xc)
#Compute squares and cross products
for(i in 1:nx){
  old.ncol <- ncol(Xr)
  if(length(table(Xc[,i]))>2){
    Xr <- cbind(Xr, Xc[,i]*Xc[,c(i:nx)])
      dimnames(Xr)[[2]] <- c(dimnames(Xr)[[2]][1:old.ncol],
         paste(dimnames(Xr)[[2]][i],"*",dimnames(Xr)[[2]][i:nx],sep=""))
  }
  else if(length(table(Xc[,i]))==2){
    if(i<nx){
      Xr <- cbind(Xr, Xc[,i]*Xc[,(i+1):nx])
      dimnames(Xr)[[2]] <- c(dimnames(Xr)[[2]][1:old.ncol],
         paste(dimnames(Xr)[[2]][i],"*",dimnames(Xr)[[2]][(i+1):nx],sep=""))
    }
  }
  else
    Xr <- Xr[,-(i+1)]
}
if(ncol(Xr)>=n)
  stop("more variables than rows")
#Now calculate RSS, PRESS, Rsq, dfmod, and dferr

qrx <- qr(Xr)
y.hat <- qr.fitted(qrx, y)
coef <- as.vector(qr.coef(qrx,y))
lev <- hat(qrx)
r <- y - y.hat
r.del <- r/(1-lev)
SSE <- sum(r^2)
PRESS <- sum(r.del^2)
SSTot <- sum((y - mean(y))^2)
SSReg <- SSTot - SSE
dfmod <- qrx$rank-1
dferr <- n - dfmod -1
APS <- SSE/(1-(dfmod+1)/n)^2
Rsq <- SSReg/SSTot

if(summary == F)
  return(list(SSReg=SSReg, SSE=SSE, dferr=dferr, PRESS=PRESS, APS=APS,
              Rsq=Rsq, coef=coef))

else{  # Create summary objects
  SS1 <- numeric(nx)
  SS3 <- numeric(nx)
  df1 <- numeric(nx)
  df3 <- numeric(nx)
  F.stat1 <- numeric(nx)
  F.stat3 <- numeric(nx)
  P.val1 <- numeric(nx)
  P.val3 <- numeric(nx)
  frac.var1 <- rep(0,nx)
  cum.var <- numeric(nx)
  frac.var3 <- numeric(nx)
  step.PRESS <- numeric(nx)
  prev.model <- list(SSE = SSTot, dferr = n-1)

  if(nx>1){
```

```
      for(i in 1:nx){
    #sequential summary
        next.model <- rs.reg(X[,c(1:i)],y,F)
        SS1[i] <- prev.model$SSE - next.model$SSE
        df1[i] <- prev.model$dferr - next.model$dferr
        F.stat1[i] <- (SS1[i]/df1[i])/(next.model$SSE/next.model$dferr)
        P.val1[i] <- 1-pf(F.stat1[i],df1[i],next.model$dferr)
        frac.var1[i] <- SS1[i]/SSTot
        cum.var[i] <- sum(frac.var1)
        step.PRESS[i] <- next.model$PRESS
        prev.model <- next.model
    #Last term summary
        r.model <- rs.reg(X[,-i],y,F)
        SS3[i] <- r.model$SSE - SSE
        df3[i] <- r.model$dferr - dferr
        F.stat3[i] <- (SS3[i]/df3[i])/(SSE/dferr)
        P.val3[i] <- 1-pf(F.stat3[i],df3[i],dferr)
        frac.var3[i] <- SS3[i]/SSTot
      }
    }
    else if(nx==1){
      SS1 <- SS3 <- SSReg
      df1 <- df3 <- dfmod
      F.stat1 <- F.stat3 <- (SSReg/df1)/(SSE/dferr)
      P.val1 <- P.val3 <- 1-pf(F.stat1,df1,dferr)
      frac.var1 <- frac.var3 <- cum.var <- SS3/SSTot
      step.PRESS <- PRESS
    }
    else{  # nx=0
      SS1 <- SS3 <- SSReg
      df1 <- df3 <- dfmod
      F.stat1 <- F.stat3 <- (SS1/df1)/(SSE/dferr)
      P.val1 <- P.val3 <- 1-pf(F.stat1,df1,dferr)
      frac.var1 <- frac.var3 <- SS3/SSTot
    }

    Rsq <- SSReg/SSTot
    Adj.Rsq <- 1-((n-1)/dferr)*(SSE/SSTot)
    MSReg <- SSReg/dfmod
    MSE <- SSE/dferr
    F.mod <- MSReg/MSE
    P.mod <- 1-pf(F.mod,dfmod,dferr)
    var.table1 <- data.frame(var= dimnames(Xc)[[2]], df1=df1, SS1=SS1,
                             F=F.stat1, P.val=P.val1,
frac.var=round(frac.var1,4),
                             Rsq=round(cum.var,4), PRESS=step.PRESS)
    var.table3 <- data.frame(var= dimnames(Xc)[[2]], df3=df3, SS3=SS3,
                             F=F.stat3, P.val=P.val3,
frac.var=round(frac.var3,4))
    mod.table <- data.frame(Source=c("Model","Error"), df=c(dfmod,dferr),
                            SS=c(SSReg,SSE), MS=c(MSReg,MSE), F=c(F.mod,""),
                            P.val=c(P.mod,""))
    return(list(mod.sum = mod.table, Rsq = Rsq, Adj.Rsq=Adj.Rsq, PRESS =
PRESS,
                APS=APS, span=NA, seq.sum = var.table1, last.sum=var.table3,
                coef=coef))
  }
}

#############################################################################
#

reg <- function(X, y, summary=T){
#calculate a regression
```

```
if(missing(X))
  X <- numeric(0)

X <- as.matrix(X)

if(nrow(X)==0){
  n <- length(y)
  nx <- 0
  Xr <- as.matrix(rep(1,n))
}
else{
  n <- nrow(X)
  nx <- ncol(X)
  if(n != length(y))
    stop("y must have length nrow(X)")
  if(nx+1 > n)
    warning("more variables than observations")
  if(length(dimnames(X)[[2]]) == 0){
    dimnames(X)[[2]] <- list()
    dimnames(X)[[2]] <- paste("x", 1:nx, sep="")
  }
  Xr <- cbind(1,X)
}
#Calculate RSS, PRESS, Rsq, dfmod, and dferr

qrx <- qr(Xr)
y.hat <- qr.fitted(qrx, y)
coef <- as.vector(qr.coef(qrx,y))
lev <- hat(qrx)
r <- y - y.hat
r.del <- r/(1-lev)
SSE <- sum(r^2)
PRESS <- sum(r.del^2)
SSTot <- sum((y - mean(y))^2)
SSReg <- SSTot - SSE
dfmod <- qrx$rank-1
dferr <- n - dfmod -1
APS <- SSE/(1-(dfmod+1)/n)^2
Rsq <- SSReg/SSTot
if(summary == F)
  return(list(SSReg=SSReg, SSE=SSE, dferr=dferr, PRESS=PRESS, Rsq=Rsq,
              coef=coef, fitted=y.hat))

else{  ################ Create summary objects #############
  SS1 <- numeric(nx)
  SS3 <- numeric(nx)
  df1 <- numeric(nx)
  df3 <- numeric(nx)
  F.stat1 <- numeric(nx)
  F.stat3 <- numeric(nx)
  P.val1 <- numeric(nx)
  P.val3 <- numeric(nx)
  frac.var1 <- rep(0,nx)
  cum.var <- numeric(nx)
  frac.var3 <- numeric(nx)
  step.PRESS <- numeric(nx)
  prev.model <- list(SSE = SSTot, dferr = n-1)

  if(nx>1){
    for(i in 1:nx){
  #sequential summary
      next.model <- reg(X[,c(1:i)],y,F)
      SS1[i] <- prev.model$SSE - next.model$SSE
```

```
        df1[i] <- prev.model$dferr - next.model$dferr
        F.stat1[i] <- (SS1[i]/df1[i])/(next.model$SSE/next.model$dferr)
        P.val1[i] <- 1-pf(F.stat1[i],df1[i],next.model$dferr)
        frac.var1[i] <- SS1[i]/SSTot
        cum.var[i] <- sum(frac.var1)
        step.PRESS[i] <- next.model$PRESS
        prev.model <- next.model
    #Last term summary
        r.model <- reg(X[,-i],y,F)
        SS3[i] <- r.model$SSE - SSE
        df3[i] <- r.model$dferr - dferr
        F.stat3[i] <- (SS3[i]/df3[i])/(SSE/dferr)
        P.val3[i] <- 1-pf(F.stat3[i],df3[i],dferr)
        frac.var3[i] <- SS3[i]/SSTot
      }
    }
    else if(nx==1){
      SS1 <- SS3 <- SSReg
      df1 <- df3 <- dfmod
      F.stat1 <- F.stat3 <- (SSReg/df1)/(SSE/dferr)
      P.val1 <- P.val3 <- 1-pf(F.stat1,df1,dferr)
      frac.var1 <- frac.var3 <- cum.var <- SS3/SSTot
      step.PRESS <- PRESS
    }
    else{  # nx=0
      SS1 <- SS3 <- 0
      df1 <- df3 <- 0
      F.stat1 <- F.stat3 <- NA
      P.val1 <- P.val3 <- NA
      frac.var1 <- frac.var3 <- cum.var <- 0
      step.PRESS <- PRESS
    }

    if(nx > 0){
      var.names <- dimnames(X)[[2]]
      Rsq <- SSReg/SSTot
      Adj.Rsq <- 1-((n-1)/dferr)*(SSE/SSTot)
      MSReg <- SSReg/dfmod
      MSE <- SSE/dferr
      F.mod <- MSReg/MSE
      P.mod <- 1-pf(F.mod,dfmod,dferr)
    }
    else{
      var.names <- "None"
      Rsq <- 0
      Adj.Rsq <- 0
      MSReg <- NA
      MSE <- SSE/dferr
      F.mod <- NA
      P.mod <- NA
    }
    var.table1 <- data.frame(var= var.names, df1=df1, SS1=SS1,
                             F=F.stat1, P.val=P.val1,
frac.var=round(frac.var1,4),
                             Rsq=round(cum.var,4), PRESS=step.PRESS)
    var.table3 <- data.frame(var= var.names, df3=df3, SS3=SS3,
                             F=F.stat3, P.val=P.val3,
frac.var=round(frac.var3,4))
    mod.table <- data.frame(Source=c("Model","Error"), df=c(dfmod,dferr),
                            SS=c(SSReg,SSE), MS=c(MSReg,MSE), F=c(F.mod,""),
                            P.val=c(P.mod,""))
    return(list(mod.sum = mod.table, Rsq = Rsq, Adj.Rsq=Adj.Rsq, PRESS =
PRESS,
                APS=APS, span=NA, seq.sum = var.table1, last.sum=var.table3,
```

```
                coef=coef, fitted=y.hat))
  }
}


##############################################################################

predict.reg <- function(X, object){

  coef <- object$coef
  Xr <- cbind(1,X)
  yhat <- as.numeric(Xr%*%coef)
  return(yhat)
}

#predict.reg <- function(X,object){
#
# X <- as.matrix(X)
# yhat <- as.numeric(apply(X, MARGIN=1, FUN=predict1.reg, object))
# return(yhat)
#}


##############################################################################

predict.rsreg <- function(X, object){

  coef <- object$coef
  #first center the predictors
  Xc <- X - matrix(colMeans(X),nrow(X),ncol(X),byrow=T)
  nx <- ncol(X)
  if(length(dimnames(X)[[2]]) == 0){
    dimnames(Xc)[[2]] <- list()
    dimnames(Xc)[[2]] <- paste("x", 1:nx, sep="")
  }
  Xr <- cbind(1,Xc)
  #Compute squares and cross products
  for(i in 1:nx){
    old.ncol <- ncol(Xr)
    if(length(table(Xc[,i]))>2){
      Xr <- cbind(Xr, Xc[,i]*Xc[,c(i:nx)])
        dimnames(Xr)[[2]] <- c(dimnames(Xr)[[2]][1:old.ncol],
           paste(dimnames(Xr)[[2]][i],"*",dimnames(Xr)[[2]][i:nx],sep=""))
    }
    else if(length(table(Xc[,i]))==2){
      if(i<nx){
        Xr <- cbind(Xr, Xc[,i]*Xc[,(i+1):nx])
        dimnames(Xr)[[2]] <- c(dimnames(Xr)[[2]][1:old.ncol],
           paste(dimnames(Xr)[[2]][i],"*",dimnames(Xr)[[2]][(i+1):nx],sep=""))
      }
    }
    else
      Xr <- Xr[,-(i+1)]
  }

  yhat <- as.numeric(Xr%*%coef)
  return(yhat)
}

##############################################################################


loess.formula <- function(loc.data){
```

```
#Takes a data frame X and creates a formula for use in loess
# ie. y~names(loc.data)[1]*names(loc.data)[2]) * ...

  npred <- ncol(loc.data)-1

  #Create a string that contains the appropriate formula

  form <- paste(names(loc.data)[npred+1], "~", names(loc.data)[1])

  if(npred>1){
    for(i in 2:npred)
      form <- paste(form, "*", names(loc.data)[i])
  }

  #Convert string into a formula object
  return(as.formula(form))
}

##############################################################################
#

loc.reg <- function(X, y, span='cv', degree=1, summary=T){
# creates a formula then applies loess(formula) to obtain desired statistics

  X <- as.matrix(X)
  n <- nrow(X)
  nx <- ncol(X)
  if(n != length(y))
    stop("y must have length nrow(X)")
  if(nx+1 >= n)
    stop("more variables than observations")
  if(length(dimnames(X)[[2]]) == 0){
    dimnames(X)[[2]] <- list()
    dimnames(X)[[2]] <- paste("x", 1:nx, sep="")
  }
  loc.data <- data.frame(cbind(X,y))
  l.span <- length(span)
  if(l.span==1 && span=='cv'){
    span <- c(10,.7,.3,.1,.07,.05)
    l.span <- length(span)
  }

  # scan for variables with 6 or less distinct values. Treat them as factors
  nfac <- 0
  for(i in 1:nx){
    if(length(table(loc.data[,i]))<=6){
      loc.data[,i] <- as.factor(loc.data[,i])
      nfac <- nfac+1
    }
  }
  if(nfac==nx){
    cv.span <- NA
    form <- as.formula(loess.formula(loc.data))
    lf.model <- lm(form,data=loc.data,qr=T)
    r <- lf.model$residuals
    lev <- hat(lf.model$qr)
    r.del <- r/(1-lev)
    SSE <- sum(r^2)
    PRESS <- sum(r.del^2)
    t.dev <- y - mean(y)
    SSTot <- sum(t.dev^2)
    SSReg <- SSTot - SSE
    dfmod <- sum(lev)-1
```

```
    dferr <- n - dfmod - 1
    APS <- SSE/(1-(dfmod+1)/n)^2
  }


  else if(l.span==1){
    cv.span <- span
    form <- as.formula(loess.formula(loc.data))
#    lf.model <- locfit.raw(X, y, alpha=c(.1,span), deg=degree, scale=T,
#                           kern="gauss", maxk=1000)

    lf.model <- locfit.raw(X, y, alpha=span, deg=degree, scale=T,
                           kern="tcub", maxk=1000)

    r <- residuals(lf.model)
#    lev <- diag(hatmatrix(form, data=loc.data, alpha=c(.1,span), deg=degree,
#                           scale=T, kern="gauss", maxk=1000, ev="data"))

    lev <- diag(hatmatrix(form, data=loc.data, alpha=span, deg=degree,
                          scale=T, kern="tcub", maxk=1000, ev="data"))

    r.del <- r/(1-lev)
    SSE <- sum(r^2)
    PRESS <- sum(r.del^2)
    t.dev <- y - mean(y)
    SSTot <- sum(t.dev^2)
    SSReg <- SSTot - SSE
    dfmod <- sum(lev)-1
    dferr <- n - dfmod - 1
    APS <- SSE/(1-(dfmod+1)/n)^2
  }

  else{  #Perform Generalized Cross Validation to determine optimal span
    APS <- Inf
#    alpha <- cbind(rep(.1,l.span),span)
#    APS.vec <- gcvplot(X, y, alpha=alpha, deg=degree, scale=T,
#                       kern="gauss", maxk=1000)$values

    APS.vec <- gcvplot(X, y, alpha=span, deg=degree, scale=T,
                       kern="tcub", maxk=1000)$values

    sort.span <- span[order(-span)]
    cv.ind <- order(APS.vec)[1]
    cv.span <- sort.span[cv.ind]
    model <- loc.reg(X, y, cv.span, degree, summary=F)
    SSE <- model$SSE
    SSReg <- model$SSReg
    dferr <- model$dferr
    PRESS <- model$PRESS
    SSTot <- model$SSTot
    dfmod <- model$dfmod
    APS <- model$APS
    lf.model <- model$lf.model
  }

  if(summary == F)
    return(list(SSReg=SSReg, SSE=SSE, dferr=dferr, PRESS=PRESS, APS=APS,
                SSTot=SSTot, Rsq=SSReg/SSTot, dfmod=dfmod, lf.model=lf.model))

  else{  ################ Create summary objects #############
    SS1 <- numeric(nx)
    SS3 <- numeric(nx)
    df1 <- numeric(nx)
    df3 <- numeric(nx)
```

```
F.stat1 <- numeric(nx)
F.stat3 <- numeric(nx)
P.val1 <- numeric(nx)
P.val3 <- numeric(nx)
frac.var1 <- rep(0,nx)
cum.var <- numeric(nx)
frac.var3 <- numeric(nx)
step.PRESS <- numeric(nx)
prev.model <- list(SSE = SSTot, dferr = n-1)

if(nx>1){
  for(i in 1:nx){
#sequential summary
    next.model <- loc.reg(X[,c(1:i)],y, span, degree, summary=F)
    SS1[i] <- prev.model$SSE - next.model$SSE
    df1[i] <- prev.model$dferr - next.model$dferr
    if(df1[i]>0){
      F.stat1[i] <- (SS1[i]/df1[i])/(next.model$SSE/next.model$dferr)
      P.val1[i] <- 1-pf(F.stat1[i],df1[i],next.model$dferr)
    }
    else if(SS1[i]>0){
      F.stat1[i] <- NA
      P.val1[i] <- 0
    }
    else if(SS1[i]==0&&df1[i]==0){
      F.stat1[i] <- NA
      P.val1[i] <- NA
    }
    else{   # SS1[i]<0 but df1[i]<0 perform lower tailed F test

      F.stat1[i] <- (SS1[i]/df1[i])/(next.model$SSE/next.model$dferr)
      P.val1[i] <- pf(F.stat1[i],-df1[i],next.model$dferr)
    }
    frac.var1[i] <- SS1[i]/SSTot
    cum.var[i] <- sum(frac.var1)
    step.PRESS[i] <- next.model$PRESS
    prev.model <- next.model
#Last term summary
    r.model <- loc.reg(X[,-i],y, span, degree, summary=F)
    SS3[i] <- r.model$SSE - SSE
    df3[i] <- r.model$dferr - dferr
    F.stat3[i] <- (SS3[i]/df3[i])/(SSE/dferr)
    P.val3[i] <- 1-pf(F.stat3[i],df3[i],dferr)
    frac.var3[i] <- SS3[i]/SSTot
  }
}
else{
  SS1 <- SS3 <- SSReg
  df1 <- df3 <- dfmod
  F.stat1 <- F.stat3 <- (SS3/df1)/(SSE/dferr)
  P.val1 <- P.val3 <- 1-pf(F.stat1,df1,dferr)
  frac.var1 <- frac.var3 <- SS3/SSTot
  cum.var <- frac.var1
  step.PRESS <- PRESS
}

Rsq <- SSReg/SSTot
Adj.Rsq <- 1-((n-1)/dferr)*(SSE/SSTot)
MSReg <- SSReg/dfmod
MSE <- SSE/dferr
F.mod <- MSReg/MSE
P.mod <- 1-pf(F.mod,dfmod,dferr)
var.table1 <- data.frame(var= dimnames(X)[[2]], df1=df1, SS1=SS1,
                         F=F.stat1, P.val=P.val1,
```

```
frac.var=round(frac.var1,4),
                          Rsq=round(cum.var,4), PRESS=step.PRESS)
    var.table3 <- data.frame(var= dimnames(X)[[2]], df3=df3, SS3=SS3,
                          F=F.stat3, P.val=P.val3,
frac.var=round(frac.var3,4))
    mod.table <- data.frame(Source=c("Model","Error"), df=c(dfmod,dferr),
                            SS=c(SSReg,SSE), MS=c(MSReg,MSE), F=c(F.mod,""),
                            P.val=c(P.mod,""))
    return(list(mod.sum = mod.table, Rsq = Rsq, Adj.Rsq=Adj.Rsq, PRESS =
PRESS,
                APS=APS, span=cv.span, seq.sum = var.table1,
                last.sum=var.table3, lf.model=lf.model))
  }
}

##############################################################################
#

predict.locreg <- function(newdata, object){

  lf.model <- object$lf.model
  ans <- as.numeric(predict(lf.model, newdata))
  return(ans)
}

##############################################################################
#


gam.formula <- function(data, method="lo", span=.3, df=6, nd.min=5){

#Takes a data frame X and creates a formula for use in gam
# ie. y~lo(names(data)[1]) + lo(names(data)[2]) + ...

  if(method!="lo"&&method!="s")
    stop("Smoothing method must be either lo or s.")
  nx <- ncol(data)-1
  if(length(span)==1)
    span <- rep(span, nx)
  if(length(df)==1)
    df <- rep(df, nx)
  if(length(span)!=nx||length(df)!=nx)
    stop("span/df must have same length as nx")

  #Create a string that contains the appropriate formula

  if(method == "lo"){
    nd.i <- length(table(data[,1]))
    if(nd.i>nd.min)
      form <- paste(names(data)[nx+1], "~ lo(", names(data)[1], ",
                    span=", span[1], ")")
    else if(length(table(data[,1]))>=3)
      form <- paste(names(data)[nx+1],"~ poly(",names(data)[1],",",nd.i-1,
")")
    else if(length(table(data[,1]))==2)
      form <- paste(names(data)[nx+1], "~", names(data)[1])

    if(nx>1){
      for(i in 2:nx){
        nd.i <- length(table(data[,i]))
        if(nd.i>nd.min)
          form <- paste(form, "+ lo(", names(data)[i], ",span=", span[i], ")")
        else if(length(table(data[,i]))>=3)
          form <- paste(form, "+ poly(", names(data)[i], ",", nd.i-1, ")")
```

```
        else if(length(table(data[,i]))==2)
          form <- paste(form, "+", names(data)[i])
      }
    }
  }

  else{  # method == "s"
    nd.i <- length(table(data[,1]))
    if(nd.i>nd.min)
      form <- paste(names(data)[nx+1], "~ s(", names(data)[1], ",", df[1],
")")
    else if(length(table(data[,1]))>=3)
      form<-paste(names(data)[nx+1], "~ poly(", names(data)[1], ",",nd.i-
1,")")
    else if(length(table(data[,1]))==2)
      form <- paste(names(data)[nx+1], "~", names(data)[1])

    if(nx>1){
      for(i in 2:nx){
        nd.i <- length(table(data[,i]))
        if(nd.i>nd.min)
          form <- paste(form, "+ s(", names(data)[i], ",", df[i], ")")
        else if(length(table(data[,i]))>=3)
          form <- paste(form, "+ poly(", names(data)[i], ",", nd.i-1, ")")
        else if(length(table(data[,i]))==2)
          form <- paste(form, "+", names(data)[i])
      }
    }
  }
  #Convert string into a formula object
  return(as.formula(form))
}

##############################################################################

add.reg <- function(X, y, smooth="s", span=df, df="cv", summary=T, CV=F,
                    step=F, int.terms=F, order, maxit=30, inc.press=F){

# creates a formula then applies gam(formula) to obtain desired statistics
  X <- as.matrix(X)
  n <- nrow(X)
  nx <- ncol(X)
  if(length(dimnames(X)[[2]]) == 0){
    dimnames(X)[[2]] <- list()
    dimnames(X)[[2]] <- paste("x", 1:nx, sep="")
  }
  if(int.terms){ ## Include Multiplicitive Interaction Terms
    ## first center the predictors
    Xc <- X - matrix(colMeans(X),nrow(X),ncol(X),byrow=T)
    Xr <- Xc
    ## Now compute cross products
    for(i in 1:(nx-1)){
      old.ncol <- ncol(Xr)
      if(length(table(Xc[,i]))>=2){
        Xr <- cbind(Xr, Xc[,i]*Xc[,(i+1):nx])
        dimnames(Xr)[[2]] <- c(dimnames(Xr)[[2]][1:old.ncol],
           paste(dimnames(Xr)[[2]][i],"*",dimnames(Xr)[[2]][(i+1):nx],sep=""))
      }
      else
        Xr <- Xr[,-(i+1)]
    }
    if(missing(order))
      order <- 1:ncol(Xr)
    X <- Xr[,order]
```

```
    nx <- ncol(X)
  }
  if(n != length(y))
    stop("y must have length nrow(X)")
  if(nx+1 >= n)
    stop("more variables than observations")
  data <- data.frame(cbind(X,y))

 ## scan for variables with 5 or less distinct values. If all <=5, fit lm
  nfac <- 0
  nd <- numeric(nx)
  for(i in 1:nx){
    nd[i] <- length(table(data[,i]))
    if(nd[i]<=5){
      nfac <- nfac+1
    }
  }
  if(nfac==nx){
    form <- loess.formula(data)
    model <- lm(form,data=data,qr=T)
    r <- model$residuals
    lev <- hat(model$qr)
    r.del <- r/(1-lev)
    SSE <- sum(r^2)
    PRESS <- sum(r.del^2)
    t.dev <- y - mean(y)
    SSTot <- sum(t.dev^2)
    SSReg <- SSTot - SSE
    dfmod <- sum(lev)-1
    dferr <- n - dfmod - 1
    APS <- SSE/(1-(dfmod+1)/n)^2
    Rsq <- SSReg/SSTot
    if(step==T){
      df <- c(df,nd[nx]-1)
      span <- c(span,1)
    }
    else{
      df <- nd-1
      span <- rep(1,nx)
    }
    gam.model <- model
  }

  else if((is.numeric(df)&&smooth=="s")||(is.numeric(span)&&smooth=="lo")){

    gam.form <- as.formula(gam.formula(data, smooth, span, df))
    gam.model <- gam(gam.form, data=data, control=gam.control(bf.maxit=maxit),
                     x=T)    #x=model.mat)
    SSE <- gam.model$deviance
    SSTot <- sum((y-mean(y))^2)
    SSReg <- SSTot - SSE
    dferr <- gam.model$df.residual
    dfmod <- n-dferr-1
    r <- gam.model$residuals
    APS <- SSE/(1-(dfmod + 1)/n)^2

    if(CV){
      r.del <- numeric(n)
cat("\n")
      for(i in 1:n){
cat('\b\b\b\b\b',i)
        Xi <- X[-i,]
        yi <- y[-i]
        gam.model.i <- add.reg(Xi, yi, smooth=smooth, span=span, df=df,
```

```
                                summary=F, CV=F, inc.press=F)
          r.del[i] <- y[i]-predict.addreg(X, gam.model.i)[i]
          ### xpred <- t(as.matrix(X[i,]))
          ### r.del[i] <- y[i]-predict.addreg(xpred, gam.model.i)
        }
        PRESS <- sum(r.del^2)
      }
      else
         PRESS <- NA
    }

    else{ #compute optimal span or df by gcv

      APS <- Inf
      df.cv <- c(1,2,4,7,10,15)
      sp.cv <- c(10,.8,.5,.3,.1,.05)
      if(!step){    #  Cross-validate df/span for all predictors
        span.mat <- matrix(rep(sp.cv, times=nx), ncol=nx)
        df.mat <- matrix(rep(df.cv, times=nx), ncol=nx)
      }
      else{   # cross-validate df/span on last predictor only
        l.sp <- length(sp.cv)
        l.df <- length(df.cv)
        span.mat <- matrix(rep(span, times=l.sp), nrow=l.sp, ncol=nx, byrow=T)
        span.mat[,nx] <- sp.cv
        span.mat <- matrix(as.numeric(span.mat),ncol=nx)
        df.mat <- matrix(rep(df, times=l.df), nrow=l.df, ncol=nx, byrow=T)
        df.mat[,nx] <- df.cv
        df.mat <- matrix(as.numeric(df.mat),ncol=nx)
      }

      for(i in 1:nrow(df.mat)){
        model.i <- add.reg(X, y, smooth, span=span.mat[i,], df=df.mat[i,],
                           summary=F, CV=F)
        if(model.i$APS < APS){
          APS <- model.i$APS
          span <- span.mat[i,]
          df <- df.mat[i,]
        }
      }
      model <- add.reg(X, y, smooth, span=span, df=df, summary=F, CV=CV)
      SSReg <- model$SSReg
      SSE <- model$SSE
      dferr <- model$dferr
      PRESS <- model$PRESS
      APS <- model$APS
      SSTot <- model$SSTot
      dfmod <- model$dfmod
      r <- model$r
      gam.model <- model$gam.model
    }
    if(smooth=="s")
      lsp <- df[nx]
    else
      lsp <- span[nx]

    if(summary == F)
      return(list(SSReg=SSReg, SSE=SSE, dferr=dferr, PRESS=PRESS, APS=APS,
                  SSTot=SSTot, Rsq=SSReg/SSTot, dfmod=dfmod, lsp=lsp,
span.vec=span,
                  df.vec=df, gam.model=gam.model))

    else{  ################ Create summary objects #############
      SS1 <- numeric(nx)
```

```
    SS3 <- numeric(nx)
    df1 <- numeric(nx)
    df3 <- numeric(nx)
    F.stat1 <- numeric(nx)
    F.stat3 <- numeric(nx)
    P.val1 <- numeric(nx)
    P.val3 <- numeric(nx)
    frac.var1 <- rep(0,nx)
    cum.var <- numeric(nx)
    frac.var3 <- numeric(nx)
    step.APS <- numeric(nx)
    step.PRESS <- numeric(nx)
    prev.model <- list(SSE = SSTot, dferr = n-1)

    if(nx>1){
      for(i in 1:nx){
    #sequential summary
        next.model <- add.reg(X[,1:i],y, smooth, span[1:i], df[1:i],
                              summary=F, CV=inc.press)
        SS1[i] <- prev.model$SSE - next.model$SSE
        df1[i] <- prev.model$dferr - next.model$dferr
        if(df1[i]>0){
          F.stat1[i] <- (SS1[i]/df1[i])/(next.model$SSE/next.model$dferr)
          P.val1[i] <- 1-pf(F.stat1[i],df1[i],next.model$dferr)
        }
        else if(SS1[i]>0){
          F.stat1[i] <- NA
          P.val1[i] <- 0
        }
        else if(SS1[i]==0&&df1[i]==0){
          F.stat1[i] <- NA
          P.val1[i] <- NA
        }
        else{   # SS1[i]<0 but df1[i]<0 perform lower tailed F test

          F.stat1[i] <- (SS1[i]/df1[i])/(next.model$SSE/next.model$dferr)
          P.val1[i] <- pf(F.stat1[i],-df1[i],next.model$dferr)
        }
        frac.var1[i] <- SS1[i]/SSTot
        cum.var[i] <- sum(frac.var1)
        step.APS[i] <- next.model$APS
        step.PRESS[i] <- next.model$PRESS
        prev.model <- next.model
    #Last term summary
        r.model <- add.reg(X[,-i],y, smooth, span[-i], df[-i], summary=F,
CV=F)
        SS3[i] <- r.model$SSE - SSE
        df3[i] <- r.model$dferr - dferr
        F.stat3[i] <- (SS3[i]/df3[i])/(SSE/dferr)
        P.val3[i] <- 1-pf(F.stat3[i],df3[i],dferr)
        frac.var3[i] <- SS3[i]/SSTot
      }
    }
    else{
      SS1 <- SS3 <- SSReg
      df1 <- df3 <- dfmod
      F.stat1 <- F.stat3 <- (SS3/df3)/(SSE/dferr)
      P.val1 <- P.val3 <- 1-pf(F.stat1,df3,dferr)
      frac.var1 <- frac.var3 <- SS3/SSTot
      cum.var <- frac.var1
      step.APS <- APS
      step.PRESS <- PRESS
    }
```

```
    Rsq <- SSReg/SSTot
    Adj.Rsq <- 1-((n-1)/dferr)*(SSE/SSTot)
    MSReg <- SSReg/dfmod
    MSE <- SSE/dferr
    F.mod <- MSReg/MSE
    P.mod <- 1-pf(F.mod,dfmod,dferr)

    var.table1 <- data.frame(var= dimnames(X)[[2]], df1=df1, SS1=SS1,
                             F=F.stat1, P.val=P.val1,
frac.var=round(frac.var1,4),
                             Rsq=round(cum.var,4), APS=step.APS,
PRESS=step.PRESS)
    var.table3 <- data.frame(var= dimnames(X)[[2]], df3=df3, SS3=SS3,
                             F=F.stat3, P.val=P.val3,
frac.var=round(frac.var3,4))
    mod.table <- data.frame(Source=c("Model","Error"), df=c(dfmod,dferr),
                            SS=c(SSReg,SSE), MS=c(MSReg,MSE), F=c(F.mod,""),
                            P.val=c(P.mod,""))
    return(list(mod.sum = mod.table, Rsq = Rsq, Adj.Rsq=Adj.Rsq, PRESS =
PRESS,
            APS=APS, smooth=smooth, seq.sum = var.table1, last.sum=var.table3,
            gam.model=gam.model))
  }
}

##############################################################################
#


predict.addreg <- function(X, object, int.terms=F, order){

  X <- as.matrix(X)
  n <- nrow(X)
  nx <- ncol(X)
  if(length(dimnames(X)[[2]]) == 0){
    dimnames(X)[[2]] <- list()
    dimnames(X)[[2]] <- paste("x", 1:nx, sep="")
  }
  if(int.terms){ ## Include Multiplicitive Interaction Terms
    ## first center the predictors
    Xc <- X - matrix(colMeans(X),nrow(X),ncol(X),byrow=T)
    Xr <- Xc
    ## Now compute cross products
    for(i in 1:(nx-1)){
      old.ncol <- ncol(Xr)
      if(length(table(Xc[,i]))>=2){
        Xr <- cbind(Xr, Xc[,i]*Xc[,(i+1):nx])
        dimnames(Xr)[[2]] <- c(dimnames(Xr)[[2]][1:old.ncol],
           paste(dimnames(Xr)[[2]][i],".",dimnames(Xr)[[2]][(i+1):nx],sep=""))
      }
      else
        Xr <- Xr[,-(i+1)]
    }
    if(missing(order))
      order <- 1:ncol(Xr)
    X <- Xr[,order]
    nx <- ncol(X)
  }
  X <- as.data.frame(X)

  gam.model <- object$gam.model
  ans <- predict(gam.model, X, type='terms')
  constant <- attributes(ans)$constant
  yhat <- rowSums(ans)+constant
```

```
  return(yhat)
}

#############################################################################
#

ppr2 <- function(X, y, df="gcv", nterms="gcv", summary=T, CV=F,
                 maxterms=ncol(X)+1, inc.press=F)
{

  # calls ppr  Then obtains desired statistics
  # X: matrix of predictors, y: vector of responses,
  # nterms: the number of projection terms to include.  default"cv" to
  #  determine by generalized cross validation on 1:max.terms.
  # df: is degrees of freedom for splines fit to projection terms, default is
  #  "gcv" which will choose span by GCV while choosing nterms

  X <- as.matrix(X)
  n <- nrow(X)
  nx <- ncol(X)
  if(n != length(y))
    stop("y must have length nrow(X)")
  if(nx+1 >= n)
    stop("more variables than observations")
  if(length(dimnames(X)[[2]]) == 0){
    dimnames(X)[[2]] <- list()
    dimnames(X)[[2]] <- paste("x", 1:nx, sep="")
  }

  # scan for variables with 5 or less distinct values. Treat them as factors
  nfac <- 0
  data <- data.frame(cbind(X,y))
  for(i in 1:nx){
    if(length(table(data[,i]))<=5){
      data[,i] <- as.factor(data[,i])
      nfac <- nfac+1
    }
  }
  if(nfac==nx){
    cv.span <- NA
    cv.terms <- NA
    form <- loess.formula(data)
    model <- lm(form,data=data,qr=T)
    r <- model$residuals
    lev <- hat(model$qr)
    r.del <- r/(1-lev)
    SSE <- sum(r^2)
    PRESS <- sum(r.del^2)
    t.dev <- y - mean(y)
    SSTot <- sum(t.dev^2)
    SSReg <- SSTot - SSE
    dfmod <- sum(lev)-1
    dferr <- n - dfmod - 1
    APS <- SSE/(1-(dfmod+1)/n)^2
    Rsq <- SSReg/SSTot
    df.vec <- NA
    pp.model <- model
  }

  else if(is.numeric(nterms)){
    cv.terms <- nterms
    if(is.numeric(df))
      pp.model <- ppr(X,
y,nterms=cv.terms,sm.method='spline',df=df,optlevel=2)
```

```
    else
      pp.model <- ppr(X, y, nterms=cv.terms, sm.method='gcvspline',
optlevel=2)

    SSE <- sum(pp.model$residuals^2)
    SSTot <- sum((y-mean(y))^2)
    SSReg <- SSTot - SSE
    dfmod <- sum(pp.model$edf)
    dferr <- n-dfmod
    APS <- SSE/(max(0,1-(dfmod+1)/n))^2
    Rsq <- SSReg/SSTot
    df.vec <- pp.model$edf
    pp.model <- pp.model
  }

  else{ #Cross Validate on nterms using GCV

    APS <- Inf
    for(i in 1:maxterms){
      if(is.numeric(df))
        pp.model.i <- ppr(X, y, nterms=i, sm.method='spline',df=df,optlevel=2)
      else
        pp.model.i <- ppr(X, y, nterms=i, sm.method='gcvspline', optlevel=2)

      APS.i<-sum(pp.model.i$residuals^2)/(max(0,1-
(sum(pp.model.i$edf)+1)/n))^2
#cat("\n nterms =",i,"   APS =",APS.i)
      if(APS.i < APS){
        APS <- APS.i
        cv.terms <- i
      }
    }
    model <- ppr2(X, y, df, cv.terms, summary=F, CV=F)
    SSReg <- model$SSReg
    SSE <- model$SSE
    dferr <- model$dferr
    PRESS <- model$PRESS
    APS <- model$APS
    SSTot <- model$SSTot
    dfmod <- model$dfmod
    Rsq <- model$Rsq
    df.vec <- model$df.vec
    pp.model <- model$pp.model
  }

  if(CV){
    r.del <- numeric(n)
cat("\n")
    for(i in 1:n){
cat('\b\b\b\b\b',i)
      Xi <- X[-i,]
      yi <- y[-i]
      xpred <- t(as.matrix(X[i,]))
      pp.model.i <- ppr2(Xi, yi, df, nterms, summary=F, CV=F)
      r.del[i] <- y[i]-predict.ppr2(xpred, pp.model.i)
    }
    PRESS <- sum(r.del^2)
  }
  else
    PRESS <- NA

  if(!summary){
    return(list(SSReg=SSReg, SSE=SSE, dferr=dferr, SSTot=SSTot, PRESS=PRESS,
                dfmod=dfmod, APS=APS, Rsq=Rsq, nterms=cv.terms,df.vec=df.vec,
```

```
            pp.model=pp.model))
}

else{  ################ Create summary objects #############
  SS1 <- numeric(nx)
  SS3 <- numeric(nx)
  df1 <- numeric(nx)
  df3 <- numeric(nx)
  F.stat1 <- numeric(nx)
  F.stat3 <- numeric(nx)
  P.val1 <- numeric(nx)
  P.val3 <- numeric(nx)
  frac.var1 <- rep(0,nx)
  cum.var <- numeric(nx)
  frac.var3 <- numeric(nx)
  step.APS <- numeric(nx)
  step.PRESS <- numeric(nx)
  prev.model <- list(SSE = SSTot, dferr = n-1)

  if(nx>1){
    for(i in 1:nx){
  #sequential summary
      next.model <- ppr2(X[,c(1:i)], y, df, nterms, summary=F, CV=inc.press)
      SS1[i] <- prev.model$SSE - next.model$SSE
      df1[i] <- prev.model$dferr - next.model$dferr
      if(df1[i]>0){
        F.stat1[i] <- (SS1[i]/df1[i])/(next.model$SSE/next.model$dferr)
        P.val1[i] <- 1-pf(F.stat1[i],df1[i],next.model$dferr)
      }
      else if(SS1[i]>0){
        F.stat1[i] <- NA
        P.val1[i] <- 0
      }
      else if(SS1[i]==0&&df1[i]==0){
        F.stat1[i] <- NA
        P.val1[i] <- NA
      }
      else{   # SS1[i]<0 but df1[i]<0 perform lower tailed F test

        F.stat1[i] <- (SS1[i]/df1[i])/(next.model$SSE/next.model$dferr)
        P.val1[i] <- pf(F.stat1[i],-df1[i],next.model$dferr)
      }
      frac.var1[i] <- SS1[i]/SSTot
      cum.var[i] <- sum(frac.var1)
      step.APS[i] <- next.model$APS
      step.PRESS[i] <- next.model$PRESS
      prev.model <- next.model
  #Last term summary
      r.model <- ppr2(X[,-i], y, df, nterms, summary=F, CV=F)
      SS3[i] <- r.model$SSE - SSE
      df3[i] <- r.model$dferr - dferr
      if(df3[i]<0)
        df3[i] <- 1
      F.stat3[i] <- (SS3[i]/df3[i])/(SSE/dferr)
      P.val3[i] <- 1-pf(F.stat3[i],df3[i],dferr)
      frac.var3[i] <- SS3[i]/SSTot
    }
  }
  else{
    SS1 <- SS3 <- SSReg
    df1 <- df3 <- dfmod
    F.stat1 <- F.stat3 <- (SS3/df3)/(SSE/dferr)
    P.val1 <- P.val3 <- 1-pf(F.stat1,df3,dferr)
    frac.var1 <- frac.var3 <- cum.var <- SS3/SSTot
```

```
      step.APS <- APS
      step.PRESS <- PRESS
    }

    Rsq <- SSReg/SSTot
    Adj.Rsq <- 1-((n-1)/dferr)*(SSE/SSTot)
    MSReg <- SSReg/dfmod
    MSE <- SSE/dferr
    F.mod <- MSReg/MSE
    P.mod <- 1-pf(F.mod,dfmod,dferr)
    var.table1 <- data.frame(var= dimnames(X)[[2]], df1=df1, SS1=SS1,
                             F=F.stat1, P.val=P.val1,
frac.var=round(frac.var1,4),
                             Rsq=round(cum.var,4), APS=step.APS,
PRESS=step.PRESS)
    var.table3 <- data.frame(var= dimnames(X)[[2]], df3=df3, SS3=SS3,
                             F=F.stat3, P.val=P.val3,
frac.var=round(frac.var3,4))
    mod.table <- data.frame(Source=c("Model","Error"), df=c(dfmod,dferr),
                            SS=c(SSReg,SSE), MS=c(MSReg,MSE), F=c(F.mod,""),
                            P.val=c(P.mod,""))
    return(list(mod.sum = mod.table, Adj.Rsq=Adj.Rsq, seq.sum = var.table1,
                last.sum=var.table3,SSReg=SSReg, SSE=SSE, dferr=dferr,
                SSTot=SSTot, PRESS=PRESS, dfmod=dfmod, APS=APS, Rsq=Rsq,
                nterms=cv.terms,df.vec=df.vec, pp.model=pp.model))
  }
}


##############################################################################
#


predict.ppr2 <- function(newdata, ppr2.obj){

  yhat <- predict(ppr2.obj$pp.model, newdata)
  return(yhat)
}



##############################################################################
#


part <- function(X, y, nfew, space, region){

#calculate the best variable and split point for a linear regression partition
  X <- as.matrix(X)
  n <- nrow(X)
  if(n < 2*nfew)
    stop("too few observations")
  if(n != length(y))
    stop("y must have length nrow(X)")
  nx <- ncol(X)
  poss.i <- unique(c(seq(nfew,n-nfew,by=space),n-nfew))

  SSE <- Inf
  PRESS <- Inf
  for(j in 1:nx){
    Xs <- as.matrix(X[order(X[,j]),])
    ys <- y[order(X[,j])]
    for(i in poss.i){
      model1 <- reg(Xs[1:i,], ys[1:i], sum=F)
```

```
      model2 <- reg(Xs[(i+1):n,], ys[(i+1):n], sum=F)
      SSEij <- model1$SSE + model2$SSE

      PRESSij <- model1$PRESS + model2$PRESS
      if(is.na(PRESSij))
        next;
#      if(PRESSij <= PRESS){
      if(SSEij <= SSE){
        SSE <- SSEij
        PRESS <- PRESSij
        SSE1 <- model1$SSE
        SSE2 <- model2$SSE
        dferr1 <- model1$dferr
        dferr2 <- model2$dferr
        PRESS1 <- model1$PRESS
        PRESS2 <- model2$PRESS
        coef1 <- model1$coef
        coef2 <- model2$coef
        splt.var <- j
        splt.pt <- i
      }
    }
  }

  ind <- order(X[,splt.var])
  ind1 <- ind[1:splt.pt]
  ind2 <- ind[(splt.pt+1):n]
  region1 <- region2 <- region
  new.lim <- (X[ind,splt.var][splt.pt]+X[ind,splt.var][splt.pt+1])/2
  region1[splt.var,2] <- new.lim
  region2[splt.var,1] <- new.lim

  return(list(ind1=ind1, ind2=ind2, coef1=coef1, coef2=coef2, region1=region1,
              region2=region2, SSE1=SSE1, SSE2=SSE2, SSE=SSE,dferr1=dferr1,
              dferr2=dferr2, PRESS1=PRESS1, PRESS2=PRESS2, PRESS=PRESS))
}

##############################################################################
#

rpreg <- function(X, y, nfew=ncol(X)+2, space=3, cv.space=3, nsplit="gcv",
                  summary=T, CV=F, inc.press=F){
#calculate a recursive partition linear regression

  X <- as.matrix(X)
  n <- nrow(X)
  nx <- ncol(X)
  if(n != length(y))
    stop("y must have length nrow(X)")

  if(length(dimnames(X)[[2]]) == 0){
    dimnames(X)[[2]] <- list()
    dimnames(X)[[2]] <- paste("x", 1:nx, sep="")
  }
  part.ind <- list(1:n)
  dfmod <- nx
  model <- reg(X,y,sum=F)
  region <- list(matrix(c(-Inf,Inf),ncol=2,nrow=nx,byrow=T))
  coef <- list(model$coef)
  SSE <- part.SSE <- model$SSE
  part.df <- model$dferr
  APS <- SSE/(1-2/n)^2
  PRESS <- part.PRESS <- model$PRESS
  repeat{
```

```
    PRESSdiff <- 0
    APSdiff <- 0
    flag <- F
    for(i in 1:length(part.ind)){
      if(length(part.ind[[i]])<2*nfew)
        next;
      ind <- part.ind[[i]]
      modeli <- part(X[ind,], y[ind], nfew, space, region[[i]])
      SSE.now <- sum(part.SSE[-i])+modeli$SSE1+modeli$SSE2
      df.now <- dfmod+2*nx+1
      APS.now <- SSE.now/(1-(df.now+1)/n)^2
#      if((part.PRESS[i]-modeli$PRESS) > PRESSdiff){
      if(APS-APS.now > APSdiff){
        part.num <- i
        PRESSdiff <- part.PRESS[i]-modeli$PRESS
        APSdiff <- APS-APS.now
        SSE1 <- modeli$SSE1
        SSE2 <- modeli$SSE2
        dferr1 <- modeli$dferr1
        dferr2 <- modeli$dferr2
        PRESS1 <- modeli$PRESS1
        PRESS2 <- modeli$PRESS2
        ind1 <- modeli$ind1
        ind2 <- modeli$ind2
        region1 <- modeli$region1
        region2 <- modeli$region2
        coef1 <- modeli$coef1
        coef2 <- modeli$coef2
        flag <- T
      }
    }
    if(!flag){
#       warning("Not enough data to continue partitioning")
      break;
    }

    part.SSE <- c(part.SSE[-part.num],SSE1,SSE2)
    part.df <- c(part.df[-part.num],dferr1,dferr2)
    part.PRESS <- c(part.PRESS[-part.num],PRESS1,PRESS2)
#    dfnew <- n-sum(part.df)-1
    dfnew <- length(part.SSE)*nx + nx*(length(part.SSE)-1)-1
    SSE.new <- sum(part.SSE)
    APS.new <- SSE.new/(1-(dfnew+1)/n)^2
    PRESS.new <- sum(part.PRESS)
    if(nsplit=="cv"&&PRESS.new >= PRESS)
      break;
    if(nsplit=="gcv"&&APS.new >= APS)
      break;
    SSE <- SSE.new
    dfmod <- dfnew
    APS <- APS.new
    PRESS <- PRESS.new
    nr <- length(part.ind)

    part.ind <- c(part.ind[-part.num],
                  list(part.ind[[part.num]][ind1],part.ind[[part.num]][ind2]))
    region <- c(region[-part.num], list(region1, region2))
    coef <- c(coef[-part.num], list(coef1, coef2))

  }
  if(CV){
cat("\n")
    r.del <- rep(NA,n)
    for(i in 1:n){
```

```
cat('\b\b\b\b\b',i)
      Xi <- X[-i,]
      yi <- y[-i]
      xpred <- t(as.matrix(X[i,]))
      model.i <- rpreg(Xi, yi, nfew=nfew, space=cv.space, nsplit=nsplit,
                       summary=F, CV=F)
      r.del[i] <- y[i]-predict.rpr(xpred, model.i)
    }
    PRESS <- sum(r.del^2)
  }
  else
    PRESS <- NA

  SSTot <- sum((y - mean(y))^2)
  SSReg <- SSTot - SSE
  dferr <- n - dfmod -1
  Rsq <- SSReg/SSTot

  if(!summary){
    return(list(SSReg=SSReg, SSE=SSE, dferr=dferr, SSTot=SSTot, PRESS=PRESS,
                dfmod=dfmod, Rsq=Rsq, APS=APS, region=region,
                coef=coef, part.ind=part.ind))
  }

  else{  ################ Create summary objects #############
    SS1 <- numeric(nx)
    SS3 <- numeric(nx)
    df1 <- numeric(nx)
    df3 <- numeric(nx)
    F.stat1 <- numeric(nx)
    F.stat3 <- numeric(nx)
    P.val1 <- numeric(nx)
    P.val3 <- numeric(nx)
    frac.var1 <- rep(0,nx)
    cum.var <- numeric(nx)
    frac.var3 <- numeric(nx)
    step.APS <- numeric(nx)
    step.PRESS <- rep(NA,nx)
    prev.model <- list(SSE = SSTot, dferr = n-1)

    if(nx>1){
      for(i in 1:nx){
    #sequential summary
        next.model <- rpreg(X[,1:i], y,space=space,nsplit=nsplit,summary=F,
                            CV=inc.press)
        SS1[i] <- prev.model$SSE - next.model$SSE
        df1[i] <- prev.model$dferr - next.model$dferr
        if(df1[i]>0){
          F.stat1[i] <- (SS1[i]/df1[i])/(next.model$SSE/next.model$dferr)
          P.val1[i] <- 1-pf(F.stat1[i],df1[i],next.model$dferr)
        }
        else if(SS1[i]>0){
          F.stat1[i] <- NA
          P.val1[i] <- 0
        }
        else if(SS1[i]==0&&df1[i]==0){
          F.stat1[i] <- NA
          P.val1[i] <- NA
        }
        else{   # SS1[i]<0 but df1[i]<0 perform lower tailed F test

          F.stat1[i] <- (SS1[i]/df1[i])/(next.model$SSE/next.model$dferr)
          P.val1[i] <- pf(F.stat1[i],-df1[i],next.model$dferr)
        }
```

```
        frac.var1[i] <- SS1[i]/SSTot
        cum.var[i] <- sum(frac.var1)
        step.APS[i] <- next.model$APS
        step.PRESS[i] <- next.model$PRESS
        prev.model <- next.model
    #Last term summary
        r.model <- rpreg(X[,-i], y, space=space, nsplit=nsplit, summary=F)
        SS3[i] <- r.model$SSE - SSE
        df3[i] <- r.model$dferr - dferr
        if(df3[i]>0){
          F.stat3[i] <- (SS3[i]/df3[i])/(SSE/dferr)
          P.val3[i] <- 1-pf(F.stat3[i],df3[i],dferr)
        }
        else if(SS3[i]>0){
          F.stat3[i] <- NA
          P.val3[i] <- 0
        }
        else if(SS3[i]==0&&df3[i]==0){
          F.stat3[i] <- NA
          P.val3[i] <- NA
        }
        else{   # SS3[i]<0 but df3[i]<0 perform lower tailed F test
          F.stat3[i] <- (SS3[i]/df3[i])/(SSE/dferr)
          P.val3[i] <- pf(F.stat3[i],-df3[i],dferr)
        }
        frac.var3[i] <- SS3[i]/SSTot
      }
    }
    else{
      SS1 <- SS3 <- SSReg
      df1 <- df3 <- dfmod
      F.stat1 <- F.stat3 <- (SS3/df3)/(SSE/dferr)
      P.val1 <- P.val3 <- 1-pf(F.stat1,df3,dferr)
      frac.var1 <- frac.var3 <- cum.var <- SS3/SSTot
      step.APS <- APS
      step.PRESS <- PRESS
    }

    Rsq <- SSReg/SSTot
    Adj.Rsq <- 1-((n-1)/dferr)*(SSE/SSTot)
    MSReg <- SSReg/dfmod
    MSE <- SSE/dferr
    F.mod <- MSReg/MSE
    P.mod <- 1-pf(F.mod,dfmod,dferr)
    var.table1 <- data.frame(var= dimnames(X)[[2]], df1=df1, SS1=SS1,
                             F=F.stat1, P.val=P.val1,
frac.var=round(frac.var1,4),
                             Rsq=round(cum.var,4), APS=step.APS,
PRESS=step.PRESS)
    var.table3 <- data.frame(var= dimnames(X)[[2]], df3=df3, SS3=SS3,
                             F=F.stat3, P.val=P.val3,
frac.var=round(frac.var3,4))
    mod.table <- data.frame(Source=c("Model","Error"), df=c(dfmod,dferr),
                            SS=c(SSReg,SSE), MS=c(MSReg,MSE), F=c(F.mod,""),
                            P.val=c(P.mod,""))
    return(list(mod.sum = mod.table, Rsq = Rsq, Adj.Rsq=Adj.Rsq, PRESS=PRESS,
                APS=APS, span=NA, seq.sum=var.table1, last.sum=var.table3,
                region=region, coef=coef))
  }
}


##############################################################################
#
```

```
summary.rpreg <- function(object){

  cat(paste("Model Summary for Recursive Partitioning Regression \n"))
  print(object$mod.sum)
  cat("\n")
  cat(paste("Rsq =",round(object$Rsq, digits=4)))
  cat("       ")
  cat(paste("Adj-Rsq =",round(object$Adj.Rsq, digits=4)))
  cat("\n")
  cat(paste("PRESS =",round(object$PRESS,4)))
  cat("       ")
  cat(paste("Adj-PRESS =",round(object$APS,4)))
  cat("\n")
  cat("\n")
  cat("Sequential Summary\n")
  print(object$seq.sum)
  cat("\nLast Variable Summary\n")
  print(object$last.sum)
  invisible(1)
}


#############################################################################
#


predict1.rpr <- function(x, object){

  region <- object$region
  coef <- object$coef
  nx <- length(x)
  xr <- c(1,as.numeric(x))
  nr <- length(region)
  for(i in 1:nr){
    flag <- 1
    for(j in 1:nx){
      if(x[j]<region[[i]][j,1] || x[j]>=region[[i]][j,2]){
        flag <- 0
        break
      }
    }
    if(flag){
      reg.ind <- i
      break
    }
  }
  yhat <- t(xr)%*%(coef[[reg.ind]])
  return(yhat)
}


#############################################################################
#

predict.rpr <- function(X,object){

  X <- as.matrix(X)
  ans <- as.numeric(apply(X, MARGIN=1, FUN=predict1.rpr, object))
  return(ans)
}
```

```
#############################################################################
#


mars <- function(X, y, gcvpen=2, summary=T, CV=F, maxsize=100, tol=1.0e-16,
                 inc.press=F)
{

  X <- as.matrix(X)
  n <- nrow(X)
  nx <- ncol(X)
  if(n != length(y))
    stop("y must have length nrow(X)")
  if(nx+1 >= n)
    stop("more variables than observations")
  if(length(dimnames(X)[[2]]) == 0){
    dimnames(X)[[2]] <- list()
    dimnames(X)[[2]] <- paste("x", 1:nx, sep="")
  }

  # scan for variables with 5 or less distinct values. Treat them as factors
  nfac <- 0
  data <- data.frame(cbind(X,y))
  for(i in 1:nx){
    if(length(table(data[,i]))<=5){
      data[,i] <- as.factor(data[,i])
      nfac <- nfac+1
    }
  }
  if(nfac==nx){
    cv.span <- NA
    cv.terms <- NA
    form <- loess.formula(data)
    model <- lm(form,data=data,qr=T)
    r <- model$residuals
    lev <- hat(model$qr)
    r.del <- r/(1-lev)
    SSE <- sum(r^2)
    PRESS <- sum(r.del^2)
    t.dev <- y - mean(y)
    SSTot <- sum(t.dev^2)
    SSReg <- SSTot - SSE
    dfmod <- sum(lev)-1
    dferr <- n - dfmod - 1
    APS <- SSE/(1-(dfmod+1)/n)^2
    Rsq <- SSReg/SSTot
    pm.model <- NA
  }
  else{
    model <- polymars(y, X,
gcv=gcvpen,maxsize=maxsize,tolerance=tol,verbose=F)
    SSE <- sum(model$residuals^2)
    SSTot <- sum((y-mean(y))^2)
    SSReg <- SSTot - SSE
    dfmod <- model$model.size
    dferr <- n-dfmod
    APS <- SSE/(max(0,1-(dfmod+1)/n))^2
    Rsq <- SSReg/SSTot
    pm.model <- model
  }

  r.del <- rep(NA,n)
  if(CV){
cat("\n")
```

```
    for(i in 1:n){
cat('\b\b\b\b\b',i)
      Xi <- X[-i,]
      yi <- y[-i]
      xpred <- t(as.matrix(X[i,]))
      model.i <- mars(Xi, yi, gcvpen, maxsize=maxsize, tol=tol,summary=F,
CV=F)
      r.del[i] <- y[i]-predict.mars(xpred, model.i)
    }
    PRESS <- sum(r.del^2)
  }
  else
    PRESS <- NA

  if(!summary){
    return(list(SSReg=SSReg, SSE=SSE, dferr=dferr, SSTot=SSTot, PRESS=PRESS,
                dfmod=dfmod, APS=APS, Rsq=Rsq, pm.model=pm.model,
r.del=r.del))
  }

  else{  ################ Create summary objects ##############
    SS1 <- numeric(nx)
    SS3 <- numeric(nx)
    df1 <- numeric(nx)
    df3 <- numeric(nx)
    F.stat1 <- numeric(nx)
    F.stat3 <- numeric(nx)
    P.val1 <- numeric(nx)
    P.val3 <- numeric(nx)
    frac.var1 <- rep(0,nx)
    cum.var <- numeric(nx)
    frac.var3 <- numeric(nx)
    step.APS <- numeric(nx)
    step.PRESS <- numeric(nx)
    prev.model <- list(SSE = SSTot, dferr = n-1)

    if(nx>1){
      for(i in 1:nx){
    #sequential summary
        next.model <- mars(X[,c(1:i)], y, gcvpen, maxsize=maxsize, tol=tol,
                           summary=F, CV=inc.press)
        SS1[i] <- prev.model$SSE - next.model$SSE
        df1[i] <- prev.model$dferr - next.model$dferr
        if(df1[i]>0){
          F.stat1[i] <- (SS1[i]/df1[i])/(next.model$SSE/next.model$dferr)
          P.val1[i] <- 1-pf(F.stat1[i],df1[i],next.model$dferr)
        }
        else if(SS1[i]>0){
          F.stat1[i] <- NA
          P.val1[i] <- 0
        }
        else if(SS1[i]==0&&df1[i]==0){
          F.stat1[i] <- NA
          P.val1[i] <- NA
        }
        else{   # SS1[i]<0 but df1[i]<0 perform lower tailed F test

          F.stat1[i] <- (SS1[i]/df1[i])/(next.model$SSE/next.model$dferr)
          P.val1[i] <- pf(F.stat1[i],-df1[i],next.model$dferr)
        }
        frac.var1[i] <- SS1[i]/SSTot
        cum.var[i] <- sum(frac.var1)
        step.APS[i] <- next.model$APS
        step.PRESS[i] <- next.model$PRESS
```

```
        prev.model <- next.model
    #Last term summary
        r.model <- mars(X[,-i], y, gcvpen, maxsize=maxsize, tol=tol,
                        summary=F, CV=F)
        SS3[i] <- r.model$SSE - SSE
        df3[i] <- r.model$dferr - dferr
        if(df3[i]>0){
          F.stat3[i] <- (SS3[i]/df3[i])/(SSE/dferr)
          P.val3[i] <- 1-pf(F.stat3[i],df3[i],dferr)
        }
        else if(SS3[i]>0){
          F.stat3[i] <- NA
          P.val3[i] <- 0
        }
        else if(SS3[i]==0&&df3[i]==0){
          F.stat3[i] <- NA
          P.val3[i] <- NA
        }
        else{   # SS3[i]<0 but df3[i]<0 perform lower tailed F test

          F.stat3[i] <- (SS3[i]/df3[i])/(SSE/dferr)
          P.val3[i] <- pf(F.stat3[i],-df3[i],dferr)
        }
      }
    }
    else{
      SS1 <- SS3 <- SSReg
      df1 <- df3 <- dfmod
      F.stat1 <- F.stat3 <- (SS3/df3)/(SSE/dferr)
      P.val1 <- P.val3 <- 1-pf(F.stat1,df3,dferr)
      frac.var1 <- frac.var3 <- cum.var <- SS3/SSTot
      step.APS <- APS
      step.PRESS <- PRESS
    }

    Rsq <- SSReg/SSTot
    Adj.Rsq <- 1-((n-1)/dferr)*(SSE/SSTot)
    MSReg <- SSReg/dfmod
    MSE <- SSE/dferr
    F.mod <- MSReg/MSE
    P.mod <- 1-pf(F.mod,dfmod,dferr)
    var.table1 <- data.frame(var= dimnames(X)[[2]], df1=df1, SS1=SS1,
                             F=F.stat1, P.val=P.val1,
frac.var=round(frac.var1,4),
                             Rsq=round(cum.var,4), APS=step.APS,
PRESS=step.PRESS)
    var.table3 <- data.frame(var= dimnames(X)[[2]], df3=df3, SS3=SS3,
                             F=F.stat3, P.val=P.val3,
frac.var=round(frac.var3,4))
    mod.table <- data.frame(Source=c("Model","Error"), df=c(dfmod,dferr),
                            SS=c(SSReg,SSE), MS=c(MSReg,MSE), F=c(F.mod,""),
                            P.val=c(P.mod,""))
    return(list(mod.sum = mod.table, Adj.Rsq=Adj.Rsq, seq.sum = var.table1,
                last.sum=var.table3,SSReg=SSReg, SSE=SSE, dferr=dferr,
                SSTot=SSTot, PRESS=PRESS, dfmod=dfmod, APS=APS, Rsq=Rsq,
                pm.model=pm.model, r.del=r.del))
  }
}


##############################################################################
#
```

```
predict.mars <- function(newdata, mars.obj){

  yhat <- predict(mars.obj$pm.model, newdata)
  return(yhat)
}



##############################################################################
#

step.rs <- function(X, y, surface="rs.reg", smooth="s", span="cv", df="cv",
                    n.terms="cv", nsplit="gcv", space=1, summary=T, maxterms=20,
                    crit="pval", alpha=.02, CV=F, int.terms=F, sqr.terms=F,
                    gcvpen=2, maxsize=100, tol=1.0e-16, inc.press=F, cv.space=1)
{
# performs stepwise response surface and outputs the best model according to
# the specified criterion

  nx <- ncol(X)
  n <- nrow(X)
  if(int.terms){ ## Include Multiplicitive Interaction Terms
    ## first center the predictors
    Xc <- X - matrix(colMeans(X),nrow(X),ncol(X),byrow=T)
    if(length(dimnames(X)[[2]]) == 0){
      dimnames(Xc)[[2]] <- list()
      dimnames(Xc)[[2]] <- paste("x", 1:nx, sep="")
    }
    Xr <- Xc
    ## Now compute cross products
    for(i in 1:(nx-1)){
      old.ncol <- ncol(Xr)
      if(length(table(Xc[,i]))>=2){
        Xr <- cbind(Xr, Xc[,i]*Xc[,(i+1):nx])
        dimnames(Xr)[[2]] <- c(dimnames(Xr)[[2]][1:old.ncol],
           paste(dimnames(Xr)[[2]][i],"*",dimnames(Xr)[[2]][(i+1):nx],sep=""))
      }
      else
        Xr <- Xr[,-(i+1)]
    }
    X <- Xr
    nx <- ncol(X)
  }

  if(nx<maxterms)
    maxterms <- nx
  if(n != length(y))
    stop("y must have length nrow(X)")
  if((surface!="rs.reg")&&(surface!="reg")&&(surface!="rank")&&
     (surface!="add.reg")&&(surface!="loc.reg")&&(surface!="ppr")&&
     (surface!="rpr")&&(surface!="rpr2")&&(surface!="mars"))
    stop("not a recognized surface type")
  x.index <- 1:nx
  ## label factors
  factor <- rep(0,nx)
  for(i in 1:nx){
    if(length(table(X[,i]))<=5){
      factor[i] <- 1
    }
  }
  keep <- x.index<0
  order <- numeric(0)
  X.new <- matrix(ncol=0, nrow=n)
```

```
  PRESS <- Inf

  ####### Case 1: surface = (rs.reg), (reg), or (rank) ########

  if(surface=="rs.reg"||surface=="reg"||surface=="rank"){

    if(surface=="rs.reg")
      method <- rs.reg
    else if(surface=="reg")
      method <- reg
    else{     #assign ranks to the X columns and y
      for(i in 1:nx)
        X[,i] <- rank(X[,i])
      y <- rank(y)
      method <- reg
    }
    old.model <- list(SSE=sum((y-mean(y))^2), dferr=n-1)

    repeat{
      SSE <- Inf
      for(i in x.index[!keep]){
        Xi <- cbind(X.new,X[,i])
        model <- method(Xi,y,F)
        if(model$SSE<SSE){
          SSE <- model$SSE
          next.var <- i
        }
      }
      X.new <- cbind(X.new, X[,next.var])
      if(crit=="press"){
        model <- method(X.new, y, F)
        if(model$PRESS < PRESS)
          PRESS <- model$PRESS
        else
          break;
      }
      else if(crit=="pval"){
        new.model <- method(X.new, y, F)
        SSnew <- old.model$SSE - new.model$SSE
        dfnew <- old.model$dferr - new.model$dferr
        F.stat <- (SSnew/dfnew)/(new.model$SSE/new.model$dferr)
        p.val <- 1-pf(F.stat, dfnew, new.model$dferr)
        old.model <- new.model

        if(p.val > alpha)
          break;
      }
      else
        stop("crit must be press or pval")

      keep[next.var] <- T
      order <- c(order, next.var)
cat(paste("", next.var))
      if(ncol(X.new) == maxterms)
        break;
    }
cat("\n")
    if(length(order)>1)
      model <- method(X[,order], y, T)
    else if(length(order)==1){
      X.reg <- as.matrix(X[,order])
      dimnames(X.reg)[[2]] <- list(dimnames(X)[[2]][order])
      model <- method(X.reg, y, T)
    }
```

```
  else
    model <- reg(, y, summary=T)

  object <- list(mod.sum = model$mod.sum, Rsq = model$Rsq, span=NA,
                 Adj.Rsq=model$Adj.Rsq, PRESS = model$PRESS, APS=model$APS,
                 seq.sum = model$seq.sum, last.sum = model$last.sum,
                 order=order, surface=surface, crit=crit)
  if(summary==T)
    summary.rs(object)
  return(invisible(object))
}

###############  Case2: surface = add.reg ############################

if(surface=="add.reg"){
  old.model <- list(SSE=sum((y-mean(y))^2), dferr=n-1)
  if(smooth=="lo"&&crit=="press")
    crit <- "gcv"
  if(span=="cv"||df=="cv"){
    s.span <- .1
    l.df <- 15
  }
  else{
    s.span <- span
    l.df <- df
  }
  span.vec <- df.vec <- numeric(0)

  repeat{
    SSE <- Inf
    for(i in x.index[!keep]){
      Xi <- cbind(X.new,X[,i])
      model <- add.reg(Xi, y, smooth, c(span.vec,s.span), c(df.vec,l.df),
                       summary=F, CV=F, step=T)
      if(model$SSE<SSE){
        SSE <- model$SSE
        next.var <- i
      }
    }
    X.new <- cbind(X.new, X[,next.var])

    if(crit=="gcv"){
      model <- add.reg(X.new, y, smooth, c(span.vec,span), c(df.vec,df),
                       summary=F, CV=F, step=T)
      if(model$APS < PRESS){
        PRESS <- model$APS
        span.vec <- model$span.vec
        df.vec <- model$df.vec
      }
      else
        break;
    }

    else if(crit=="pval"){
      new.model <- add.reg(X.new, y, smooth, c(span.vec,span), c(df.vec,df),
                           summary=F, CV=F, step=T)
      SSnew <- old.model$SSE - new.model$SSE
      dfnew <- old.model$dferr - new.model$dferr
      if(dfnew>0){
        F.stat <- (SSnew/dfnew)/(new.model$SSE/new.model$dferr)
        p.val <- 1-pf(F.stat, dfnew, new.model$dferr)
      }
      else if(SSnew>0){
        p.val <- 0
```

```
        }
        else if(SSnew<=0&&dfnew==0){
          p.val <- 1
        }
        else{   # SSnew<0 but dfnew<0 perform lower tailed F test
          F.stat <- (SSnew/dfnew)/(new.model$SSE/new.model$dferr)
          p.val <- pf(F.stat, -dfnew, new.model$dferr)
        }
        if(p.val <= alpha){
          old.model <- new.model
          span.vec <- new.model$span.vec
          df.vec <- new.model$df.vec
        }
        else
          break;
      }

      else if(crit=="press"){
        model <- add.reg(X.new, y, smooth, c(span.vec,span), c(df.vec,df),
                         summary=F, CV=T, step=T)
        if(model$PRESS < PRESS){
          PRESS <- model$PRESS
          span.vec <- model$span.vec
          df.vec <- model$df.vec
        }
        else
          break;
      }

      else
        stop("crit must be press, gcv, or pval")

      keep[next.var] <- T
      order <- c(order, next.var)
cat(paste("", next.var))
      if(ncol(X.new) == maxterms)
        break;
    }
cat("\n")

    if(length(order)>1){
      model <- add.reg(X[,order], y, smooth, span.vec, df.vec,
summary=T,CV=CV,
                       inc.press=inc.press)
    }
    else if(length(order)==1){
      X.reg <- as.matrix(X[,order])
      dimnames(X.reg)[[2]] <- list(dimnames(X)[[2]][order])
      model <- add.reg(X.reg, y, smooth, span.vec, df.vec, summary=T, CV=CV,
                       inc.press=inc.press)
    }
    else
      model <- reg(, y, summary=T)

    object <- list(mod.sum = model$mod.sum, Rsq = model$Rsq, span=model$span,
                   Adj.Rsq=model$Adj.Rsq, PRESS = model$PRESS, APS=model$APS,
                   seq.sum = model$seq.sum, last.sum = model$last.sum,
                   order=order, surface=surface, crit=crit)
    if(summary==T)
      summary.rs(object)
    return(invisible(object))
  }

  ###############  Case3: surface = loc.reg ############################
```

```
  if(surface=="loc.reg"){
    old.model <- list(SSE=sum((y-mean(y))^2), dferr=n-1)

    repeat{
      SSE <- Inf
      for(i in x.index[!keep]){
        Xi <- cbind(X.new,X[,i])
        model <- loc.reg(Xi, y, span=c(.1), summary=F)
        if(model$SSE<SSE){
          SSE <- model$SSE
          next.var <- i
        }
      }
      X.new <- cbind(X.new, X[,next.var])
      if(crit=="press"){
        model <- loc.reg(X.new, y, span, summary=F)
        if(model$PRESS < PRESS)
          PRESS <- model$PRESS
        else
          break;
      }
      else if(crit=="gcv"){
        model <- loc.reg(X.new, y, span, summary=F)
        if(model$APS < PRESS){
          PRESS <- model$APS
        }
        else
          break;
      }

      else if(crit=="pval"){
        new.model <- loc.reg(X.new, y, span, summary=F)
        SSnew <- old.model$SSE - new.model$SSE
        dfnew <- old.model$dferr - new.model$dferr
        if(dfnew>0){
          F.stat <- (SSnew/dfnew)/(new.model$SSE/new.model$dferr)
          p.val <- 1-pf(F.stat, dfnew, max(1E-16,new.model$dferr))
        }
        else if(SSnew>0){
          p.val <- 0
        }
        else if(SSnew<=0&&dfnew==0){
          p.val <- 1
        }
        else{   # SSnew<0 but dfnew<0 perform lower tailed F test
          F.stat <- (SSnew/dfnew)/(new.model$SSE/new.model$dferr)
          p.val <- pf(F.stat, -dfnew, new.model$dferr)
        }
        old.model <- new.model

        if(p.val > alpha)
          break;
      }
      else
        stop("crit must be gcv, press, or pval")

      keep[next.var] <- T
      order <- c(order, next.var)
cat(paste("", next.var))
      if(ncol(X.new) == maxterms)
        break;
    }
cat("\n")
```

```
  if(length(order)>1)
    model <- loc.reg(X[,order], y, span, summary=T)
  else if(length(order)==1){
    X.reg <- as.matrix(X[,order])
    dimnames(X.reg)[[2]] <- list(dimnames(X)[[2]][order])
    model <- loc.reg(X.reg, y, span, summary=T)
  }
  else
    model <- reg(, y, summary=T)

  object <- list(mod.sum = model$mod.sum, Rsq = model$Rsq, span=model$span,
                 Adj.Rsq=model$Adj.Rsq, PRESS = model$PRESS, APS=model$APS,
                 seq.sum = model$seq.sum, last.sum = model$last.sum,
                 order=order, surface=surface, crit=crit)
  if(summary==T)
    summary.rs(object)
  return(invisible(object))
}

################  Case4: surface = ppr ##############################

if(surface=="ppr"){
  old.model <- list(SSE=sum((y-mean(y))^2), dferr=n-1)

  repeat{
    SSE <- Inf
    for(i in x.index[!keep]){

      if(ncol(X.new)==1&&factor[i])
        next
      Xi <- cbind(X.new,X[,i])
      model <- ppr2(Xi, y, df=df, nterms=n.terms, summary=F, CV=F)
      if(model$SSE<SSE){
        SSE <- model$SSE
        next.var <- i
      }
    }
    X.new <- cbind(X.new, X[,next.var])

    if(crit=="gcv"){
      model <- ppr2(X.new, y, df=df, nterms=n.terms, summary=F, CV=F)
      if(model$APS < PRESS)
        PRESS <- model$APS
      else
        break;
    }

    else if(crit=="pval"){
      new.model <- ppr2(X.new, y, df=df, nterms=n.terms, summary=F, CV=F)
      SSnew <- old.model$SSE - new.model$SSE
      dfnew <- old.model$dferr - new.model$dferr
      if(dfnew>0){
        F.stat <- (SSnew/dfnew)/(new.model$SSE/new.model$dferr)
        p.val <- 1-pf(F.stat, dfnew, new.model$dferr)
      }
      else if(SSnew>0){
        p.val <- 0
      }
      else if(SSnew<=0&&dfnew==0){
        p.val <- 1
      }
      else{    # SSnew<0 but dfnew[i]<0 perform lower tailed F test
        F.stat <- (SSnew/dfnew)/(new.model$SSE/new.model$dferr)
```

```
        p.val <- pf(F.stat, -dfnew, new.model$dferr)
      }
      old.model <- new.model

      if(p.val > alpha)
        break;
    }

    else if(crit=="press"){
      model <- ppr2(X.new, y, df=df, nterms=n.terms, summary=F, CV=T)
      if(model$PRESS < PRESS)
        PRESS <- model$PRESS
      else
        break;
    }

    else
      stop("crit must be press, gcv, or pval")

    keep[next.var] <- T
    order <- c(order, next.var)
cat(paste("", next.var))
    if(ncol(X.new) == maxterms)
      break;
  }
cat("\n")

  if(length(order)>1)
    model <- ppr2(X[,order], y, df=df, nterms=n.terms, summary=T, CV=CV,
                  inc.press=inc.press)
  else if(length(order)==1){
    X.reg <- as.matrix(X[,order])
    dimnames(X.reg)[[2]] <- list(dimnames(X)[[2]][order])
    model <- ppr2(X.reg, y, df=df, nterms=n.terms, summary=T, CV=CV,
                  inc.press=inc.press)
  }
  else
    model <- reg(, y, summary=T)

  object <- list(mod.sum = model$mod.sum, Rsq = model$Rsq, span=model$span,
                 Adj.Rsq=model$Adj.Rsq, PRESS = model$PRESS, APS=model$APS,
                 seq.sum = model$seq.sum, last.sum = model$last.sum,
                 order=order, surface=surface, crit=crit)
  if(summary==T)
    summary.rs(object)
  return(invisible(object))
}

################## Case 5: surface = rpr ############################

if(surface=="rpr"){
  old.model <- list(SSE=sum((y-mean(y))^2), dferr=n-1)

  repeat{
    SSE <- Inf
    for(i in x.index[!keep]){
      Xi <- cbind(X.new,X[,i])
      model <- rpreg(Xi, y, space=space, nsplit=nsplit, summary=F, CV=F)
      if(model$SSE<SSE){
        SSE <- model$SSE
        next.var <- i
      }
    }
    X.new <- cbind(X.new, X[,next.var])
```

```
      if(crit=="gcv"){
        model <- rpreg(X.new, y, space=space, nsplit=nsplit, summary=F, CV=F)
        if(model$APS < PRESS)
          PRESS <- model$APS
        else
          break;
      }

      else if(crit=="pval"){
        new.model <- rpreg(X.new, y, space=space,
nsplit=nsplit,summary=F,CV=F)
        SSnew <- old.model$SSE - new.model$SSE
        dfnew <- old.model$dferr - new.model$dferr
        if(dfnew>0){
          F.stat <- (SSnew/dfnew)/(new.model$SSE/new.model$dferr)
          p.val <- 1-pf(F.stat, dfnew, new.model$dferr)
        }
        else if(SSnew>0){
          p.val <- 0
        }
        else if(SSnew<=0&&dfnew==0){
          p.val <- 1
        }
        else{   # SSnew<0 but dfnew[i]<0 perform lower tailed F test
          F.stat <- (SSnew/dfnew)/(new.model$SSE/new.model$dferr)
          p.val <- pf(F.stat, -dfnew, new.model$dferr)
        }
        old.model <- new.model

        if(p.val > alpha)
          break;
      }

      else
        stop("crit must be gcv or pval")

      keep[next.var] <- T
      order <- c(order, next.var)
cat(paste("", next.var))
      if(ncol(X.new) == maxterms)
        break;
    }
cat("\n")

    if(length(order)>1)
      model <- rpreg(X[,order], y, space=space, nsplit=nsplit, summary=T,
                     CV=CV, inc.press=inc.press, cv.space=cv.space)
    else if(length(order)==1){
      X.reg <- as.matrix(X[,order])
      dimnames(X.reg)[[2]] <- list(dimnames(X)[[2]][order])
      model <- rpreg(X.reg, y, space=space, nsplit=nsplit, summary=T, CV=CV,
                     inc.press=inc.press, cv.space=cv.space)
    }
    else
      model <- reg(, y, summary=T)

    object <- list(mod.sum = model$mod.sum, Rsq = model$Rsq, span=model$span,
                   Adj.Rsq=model$Adj.Rsq, PRESS = model$PRESS, APS=model$APS,
                   seq.sum = model$seq.sum, last.sum = model$last.sum,
                   order=order, surface=surface, crit=crit)
    if(summary==T)
      summary.rs(object)
    return(invisible(object))
```

```
  }

  ################## Case 6: surface = rpr2 ##########################

  if(surface=="rpr2"){
    old.model <- list(SSE=sum((y-mean(y))^2), dferr=n-1)

    repeat{
      SSE <- Inf
      for(i in x.index[!keep]){
        Xi <- cbind(X.new,X[,i])
        model <- rpreg2(Xi, y, space=space, nsplit=nsplit,
summary=F,verbose=F)
        if(model$SSE<SSE){
          SSE <- model$SSE
          next.var <- i
        }
      }
      X.new <- cbind(X.new, X[,next.var])

      if(crit=="gcv"){
        model <- rpreg2(X.new, y, space=space, nsplit=nsplit, summary=F)
        if(model$APS < PRESS)
          PRESS <- model$APS
        else
          break;
      }

      else if(crit=="pval"){
        new.model <- rpreg2(X.new, y, space=space, nsplit=nsplit, summary=F)
        SSnew <- old.model$SSE - new.model$SSE
        dfnew <- old.model$dferr - new.model$dferr
        if(dfnew>0){
          F.stat <- (SSnew/dfnew)/(new.model$SSE/new.model$dferr)
          p.val <- 1-pf(F.stat, dfnew, new.model$dferr)
        }
        else if(SSnew>0){
          p.val <- 0
        }
        else if(SSnew<=0&&dfnew==0){
          p.val <- 1
        }
        else{   # SSnew<0 but dfnew[i]<0 perform lower tailed F test
          F.stat <- (SSnew/dfnew)/(new.model$SSE/new.model$dferr)
          p.val <- pf(F.stat, -dfnew, new.model$dferr)
        }
        old.model <- new.model

        if(p.val > alpha)
          break;
      }

      else if(crit=="press"){
        model <- rpreg2(X.new, y, space=space, nsplit=nsplit, summary=F)
        if(model$PRESS < PRESS)
          PRESS <- model$PRESS
        else
          break;
      }

      else
        stop("crit must be press, gcv, or pval")

      keep[next.var] <- T
```

```
      order <- c(order, next.var)
cat(paste("", next.var))
      if(ncol(X.new) == maxterms)
        break;
    }
cat("\n")

    if(length(order)>1)
      model <- rpreg2(X[,order], y, space=space, nsplit=nsplit, summary=T,
                      CV=CV, inc.press=inc.press, cv.space=cv.space)
    else if(length(order)==1){
      X.reg <- as.matrix(X[,order])
      dimnames(X.reg)[[2]] <- list(dimnames(X)[[2]][order])
      model <- rpreg2(X.reg, y, space=space, nsplit=nsplit, summary=T, CV=CV,
                      inc.press=inc.press, cv.space=cv.space)
    }
    else
      model <- reg(, y, summary=T)

    object <- list(mod.sum = model$mod.sum, Rsq = model$Rsq, span=model$span,
                   Adj.Rsq=model$Adj.Rsq, PRESS = model$PRESS, APS=model$APS,
                   seq.sum = model$seq.sum, last.sum = model$last.sum,
                   order=order, surface=surface, crit=crit)
    if(summary==T)
      summary.rs(object)
    return(invisible(object))
  }

  ################  Case7: surface = mars ##############################

  if(surface=="mars"){
    old.model <- list(SSE=sum((y-mean(y))^2), dferr=n-1)

    repeat{
      SSE <- Inf
      for(i in x.index[!keep]){
        Xi <- cbind(X.new,X[,i])
        model <- mars(Xi, y, gcvpen=gcvpen, maxsize=maxsize, tol=tol,
                      summary=F, CV=F)
        if(model$SSE<SSE){
          SSE <- model$SSE
          next.var <- i
        }
      }
      X.new <- cbind(X.new, X[,next.var])

      if(crit=="gcv"){
        model <- mars(X.new, y, gcvpen=gcvpen, maxsize=maxsize, tol=tol,
                      summary=F, CV=F)
        if(model$APS < PRESS)
          PRESS <- model$APS
        else
          break;
      }

      else if(crit=="pval"){
        new.model <- mars(X.new, y, gcvpen=gcvpen, maxsize=maxsize, tol=tol,
                      summary=F, CV=F)
        SSnew <- old.model$SSE - new.model$SSE
        dfnew <- old.model$dferr - new.model$dferr
        if(dfnew>0){
          F.stat <- (SSnew/dfnew)/(new.model$SSE/new.model$dferr)
          p.val <- 1-pf(F.stat, dfnew, new.model$dferr)
        }
```

```
        else if(SSnew>0){
          p.val <- 0
        }
        else if(SSnew<=0&&dfnew==0){
          p.val <- 1
        }
        else{   # SSnew<0 but dfnew[i]<0 perform lower tailed F test
          F.stat <- (SSnew/dfnew)/(new.model$SSE/new.model$dferr)
          p.val <- pf(F.stat, -dfnew, new.model$dferr)
        }
        old.model <- new.model

        if(p.val > alpha)
          break;
      }

      else if(crit=="press"){
        model <- mars(X.new, y, gcvpen=gcvpen, maxsize=maxsize, tol=tol,
                      summary=F, CV=F)
        if(model$PRESS < PRESS)
          PRESS <- model$PRESS
        else
          break;
      }

      else
        stop("crit must be press, gcv, or pval")

      keep[next.var] <- T
      order <- c(order, next.var)
cat(paste("", next.var))
      if(ncol(X.new) == maxterms)
        break;
    }
cat("\n")

    if(length(order)>1)
      model <- mars(X[,order], y, gcvpen=gcvpen, maxsize=maxsize, tol=tol,
                    summary=T, CV=CV, inc.press=inc.press)
    else if(length(order)==1){
      X.reg <- as.matrix(X[,order])
      dimnames(X.reg)[[2]] <- list(dimnames(X)[[2]][order])
      model <- mars(X.reg, y, gcvpen=gcvpen, maxsize=maxsize, tol=tol,
                    summary=T, CV=CV, inc.press=inc.press)
    }
    else
      model <- reg(, y, summary=T)

    object <- list(mod.sum = model$mod.sum, Rsq = model$Rsq, span=model$span,
                   Adj.Rsq=model$Adj.Rsq, PRESS = model$PRESS, APS=model$APS,
                   seq.sum = model$seq.sum, last.sum = model$last.sum,
                   order=order, surface=surface, crit=crit, r.del=model$r.del)
    if(summary==T)
      summary.rs(object)
    return(invisible(object))
  }


}

#############################################################################
#

summary.rs <- function(object){
```

```
# Takes a rs or reg object and displays the model and variable summaries along
# with Rsq and PRESS
  cat(paste("Model Summary for surface =", object$surface, "\n"))
  cat(paste("  stepwise criterion =", object$crit, "\n"))
  print(object$mod.sum)
  cat("\n")
  cat(paste("Rsq =",round(object$Rsq, digits=4)))
  cat("       ")
  cat(paste("Adj-Rsq =",round(object$Adj.Rsq, digits=4)))
  cat("\n")
  cat(paste("PRESS =",round(object$PRESS,4)))
  cat("       ")
  cat(paste("Adj-PRESS =",round(object$APS,4)))
  cat("\n")
  cat(paste("span =",object$span))
  cat("\n")
  cat("\n")
  cat("Sequential Summary\n")
  print(object$seq.sum)
  cat("\nLast Variable Summary\n")
  print(object$last.sum)
  invisible(1)
}

#############################################################################
#

sensitivity <- function(data, x.pos, y.pos, surface="rs",
smooth="s",span="cv",
                        df="cv", n.terms="cv", nsplit="gcv",
space=1,summary=F,
                        maxterms=20, crit="pval", alpha=.02, CV=F, gcvpen=2,
                        maxsize=100, tol=1.0e-16, inc.press=F, cv.space=1)
{
  #data: a data frame
  #y.pos: vector of positions of responses to be analyzed in data
  #x.pos: vector of positions of predictors in data
  #other parameters are as in step.rs

  data <- stand.data(data)
  X <- data[,x.pos]
  out.list <- numeric(0)
  for(i in y.pos){
    yi <- data[,i]
    print(names(data)[i])
    object.i <- step.rs(X, yi, surface=surface, smooth=smooth, span=span,
                        df=df, n.terms=n.terms, nsplit=nsplit, space=space,
                        summary=summary, maxterms=maxterms, crit=crit,
                        alpha=alpha, CV=CV, gcvpen=gcvpen, maxsize=maxsize,
                        tol=tol, inc.press=inc.press)

    out.list <- c(out.list, list(object.i))
  }
  names(out.list) <- names(data)[y.pos]
  if(summary)
    summary.sens(out.list)
  invisible(out.list)
}

#############################################################################
#

summary.sens <- function(object, file){
# Takes a sensitivity object ie. a list of step.rs objects and displays the
```

```
#   summaries for all responses (the default) or a subset

  if(!missing(file))
    sink(file=file)

  for(i in 1:length(object)){
    cat(paste("\n\n********************* Response:", names(object)[i],
              "*********************\n\n"))
    summary.rs(object[[i]])
  }

  if(!missing(file))
    sink()
  invisible(1)
}


###############################################################################
#


summary.table <- function(objects, fname, methods, ncol=3, Rsq=F, APS=F,
                          PRESS=F, CT=F, CTwT=F){

  #objects: list ofstep.rs objects
  nmeth <- length(objects)
  if(!missing(fname))
    sink(file=fname)
  if(missing(methods)){
    methods <- numeric(nmeth)
    for(i in 1:nmeth)
      methods[i] <- objects[[i]]$surface
  }
  nrow <- ceiling(nmeth/ncol)

  header <- rep(c("Var","R2","df","p-val","PRESS"),ncol)
  cat(header,sep='\t')
  cat('\n')
  for(i in 1:nrow){
    ind.m <- (ncol*(i-1)+1):(min(nmeth,ncol*i))
    nvars <- numeric(0)
    for(j in ind.m){
      nvars <- c(nvars, length(objects[[j]]$seq.sum$var))
      cat("\t\t", methods[j],"\t\t\t",sep='')
    }
    cat('\n')
    for(j in 1:max(nvars)){
      for(k in 1:length(ind.m)){
        if(j<= nvars[k]){
         cat("",as.character(objects[[ind.m[k]]]$seq.sum$var[j]),"\t",
             sep='')
         cat("",format.pval(objects[[ind.m[k]]]$seq.sum$Rsq[j],4),
             "\t",sep='')
         cat("",format(round(objects[[ind.m[k]]]$seq.sum$df[j],1),nsmall=1),
             "\t",sep='')
         cat("",format.pval(objects[[ind.m[k]]]$seq.sum$P.val[j],4),
             "\t",sep='')
#         if(is.na( objects[[ind.m[k]]]$seq.sum$PRESS[j]))
#           cat("",scientific(objects[[ind.m[k]]]$seq.sum$APS[j],digits=2),
#               "\t",sep="")
#         else
          cat("",scientific(objects[[ind.m[k]]]$seq.sum$PRESS[j],digits=2),
              "\t",sep="")
        }
```

```
        else
          cat("\t\t\t\t\t")
      }
      cat('\n')
    }
    if(Rsq){
      for(k in 1:length(ind.m))
        cat("Rsq_A = ",format.pval(objects[[ind.m[k]]]$Adj.Rsq,4),
            "\t\t\t\t\t",sep='')
      cat('\n')
    }
    if(APS){
      for(k in 1:length(ind.m))
        cat("PRS_A = ",scientific(objects[[ind.m[k]]]$APS,digits=2),
            "\t\t\t\t\t",sep='')
      cat('\n')
    }
    if(PRESS){
      for(k in 1:length(ind.m))
        cat("PRESS = ",scientific(objects[[ind.m[k]]]$PRESS,digits=2),
            "\t\t\t\t\t",sep='')
      cat('\n')
    }
    if(CT){
      for(k in 1:length(ind.m)){
        cat("C_T = ",format.pval(objects[[ind.m[k]]]$CT,4),sep='')
        if(!CTwT)
          cat("\t\t\t\t\t")
        else
          cat(", C_T w/true = ",format.pval(objects[[ind.m[k]]]$CTwtrue,4),
              "\t\t\t\t\t",sep='')
      }
      cat('\n')
    }
  }
  if(!missing(fname))
    sink()
  invisible(1)

}


##############################################################################
#


sens.tables <- function(..., fnames, methods, ncol=3, Rsq=F, APS=F,
                            PRESS=F, CT=F, CTwT=F){

## ... sensitivity objects

  objects <- list(...)
  nmeth <- length(objects)
  if(missing(fnames))
    fnames <- paste(names(objects[[1]]),'_table.out',sep='')
  flag <- missing(methods)
  nvar <- length(objects[[1]])
  for(i in 1:nvar){
      rs.obs <- list()
    for(j in 1:nmeth)
      rs.obs[[j]] <- objects[[j]][[i]]
    if(flag)
      summary.table(rs.obs, fnames[i], ncol=ncol, Rsq=Rsq, APS=APS,
                    PRESS=PRESS, CT=CT, CTwT=CTwT)
```

```
    else
      summary.table(rs.obs, fnames[i], methods, ncol=ncol, Rsq=Rsq, APS=APS,
                    PRESS=PRESS, CT=CT, CTwT=CTwT)
  }
  invisible(1)
}


################################################################################
#


scientific <- function(x, digits=4){

  if(is.na(x))
    return(rep('NA',length(x)))
  m <- floor(log10(x))
  base <- x/(10^m)
  base <- format(round(base,digits),nsmall=digits)
  ans <- paste(base,'E',m,sep='')
  return(ans)
}


################################################################################
#

format.pval <- function(x, digits=4){

  if(is.na(x))
    return('NA')
  if(x>=10^(-digits+1) || x < 5*10^(-digits-1))
    ans <- format(round(x, digits), nsmall=digits)
  else{
    y <- round(x*10^(digits), 0)
    z.string <- '0.'
    for(i in 1:(digits-1))
    z.string <- paste(z.string,0,sep='')
    ans <- paste(z.string,y,sep="")
  }
  return(ans)
}


################################################################################
#


get_sens_data <- function(fname, rows){

  header <- scan(fname,nlines=1)
  n <- header[1]
  if(missing(rows))
    rows <- c(1,n)
  nx <- header[2]
  ny <- header[3]
  nv <- nx+ny

  ## Read in Column Names and # of lines of column names to skip
  flag<-1
  nlines.cn <- 1
  while(flag){
```

```
    colnames <- scan(fname,skip=1,nlines=nlines.cn,what='character')
    if(length(colnames)==nv)
      flag <- 0
    else
      nlines.cn <- nlines.cn+1
  }

  ## Read in first obs to determine # of lines for each obs
  flag<-1
  nlines.obs <- 1
  while(flag){
    row1 <- scan(fname,skip=nlines.cn+1,nlines=nlines.obs)
    if(length(row1)==nv)
      flag <- 0
    else
      nlines.obs <- nlines.obs+1
  }

  skip <- nlines.cn+1+(rows[1]-1)*nlines.obs
  nlines <- (rows[2]-rows[1]+1)*nlines.obs
  data.vec <- scan(fname,skip=skip, nlines=nlines)
  data.mat <- matrix(data.vec, ncol=nv, byrow=T)
  sens.dat <- as.data.frame(data.mat)
  names(sens.dat) <- colnames
  return(sens.dat)
}


##############################################################################
#


srd.test <- function(x, y)
{
# performs SRD test and returns test statistic and p-val for SRD test,
#   Spearman Rank test, and joint test.

  n <- length(y)
  if(n != length(x))
    stop("length y must = length x")

  ###### Perform SRD Test ######
  ys <- y[order(x)]
  ry <- rank(ys)
  Q <- sum((ry[-1]-ry[-n])^2)

  # Determine sigma
  if(n>40) sigma <- 1/6
  else if(n>10) sigma <- 1/6 - .136/n
  else if(n==10) sigma <- .152
  else if(n==9) sigma <- .150
  else if(n==8) sigma <- .147
  else if(n==7) sigma <- .143
  else if(n==6) sigma <- .138
  else if(n==5) sigma <- .130
  else if(n==4) sigma <- .117
  else if(n==3) sigma <- .091
  else stop("n <= 2, Get Real!")

  s <- (Q-n*(n^2-1)/6)/(n^(5/2)*sigma)
  ps <- pnorm(s)

  ###### Perform RCC Test ######
  rcc <- cor.test(x, y, method="spearman")
```

```
  r <- rcc$stat
  names(r) <- NULL
  pr <- rcc$p.value
  names(pr) <- NULL

  ###### perform combined test ######

  chi <- -2*(log(pr)+log(ps))
  pc <- 1-pchisq(chi,4)

  return(list(s=s,r=r,c=chi,ps=ps, pr=pr, pc=pc))
}

##############################################################################
#

srd <- function(X, y, alpha=.05, summary=T){
# takes a matrix of predictors and performs SRD, RCC, and combined tests
# returns a sorted list of variables whose p.val <= alpha
  X <- as.matrix(X)
  n <- nrow(X)
  nx <- ncol(X)
  if(n != length(y))
    stop("y must have length nrow(X)")
  if(nx+1 >= n)
    stop("more variables than observations")
  if(length(dimnames(X)[[2]]) == 0){
    dimnames(X)[[2]] <- list()
    dimnames(X)[[2]] <- paste("x", 1:nx, sep="")
  }
  ps <- numeric(nx)
  pr <- numeric(nx)
  pc <- numeric(nx)
  for(i in 1:nx){
    test <- srd.test(X[,i],y)
    ps[i] <- test$ps
    pr[i] <- test$pr
    pc[i] <- test$pc
  }

  table.s <- data.frame(SRD= dimnames(X)[[2]], p.val.1=ps)
  table.r <- data.frame(RCC= dimnames(X)[[2]], p.val.2=pr)
  table.c <- data.frame(Combined= dimnames(X)[[2]], p.val=pc)

  table.s <- table.s[order(table.s$p.val.1),]
  table.r <- table.r[order(table.r$p.val.2),]
  table.c <- table.c[order(table.c$p.val),]

  table.s$p.val <- round(table.s$p.val.1, 6)
  table.r$p.val <- round(table.r$p.val.2, 6)
  table.c$p.val <- round(table.c$p.val, 6)

  table <- cbind(table.s, table.r, table.c)
  logical <-
table.s$p.val.1<=alpha|table.r$p.val.2<=alpha|table.c$p.val<=alpha
  table <- table[logical,]

  ## Stuff to just get sens.tables and TDCC to work
  ## This is needed for the TDCC  ( a seq.sum$frac.var and order entry )
  logical.c <- table.c$p.val<=alpha
  order <- as.numeric(row.names(table.c[logical.c,]))
  if(length(order)>0){
    frac.var <- length(order):1
  }
```

```
  var <- table$Combined
  P.val <- table$p.val
  df <- rep(NA,length(var))
  Rsq <- rep(NA,length(var))
  PRESS <- rep(NA,length(var))
  seq.sum <- list(frac.var=frac.var,var=var,P.val=P.val,df=df,Rsq=Rsq,
                  PRESS=PRESS)
  ans <- list(table=table, seq.sum=seq.sum, order=order, APS=NA, Adj.Rsq=NA,
              PRESS=NA, surface='srd_rcc')

  return(ans)
}

##############################################################################
#

srd.sens <- function(data, x.pos, y.pos, alpha=.02, summary=T){
  #Performs srd test for all variables in x.pos on all responses in y.pos
  #data: a data frame
  #y.pos: vector of positions of responses to be analyzed in data
  #x.pos: vector of positions of predictors in data
  #other parameters are as in step.rs

  data <- stand.data(data)
  X <- data[,x.pos]
  out.list <- numeric(0)
  for(i in y.pos){
    yi <- data[,i]
    object.i <- srd(X,yi,alpha,sum=F)
    out.list <- c(out.list, list(object.i))
  }
  names(out.list) <- names(data)[y.pos]
  if(summary)
    summary.srd(out.list)
  return(out.list)
}

##############################################################################
#

summary.srd <- function(object, file){
# Takes an srd.sens object ie. a list of srd objects and displays the
#   summaries for all responses (the default) or a subset

  if(!missing(file))
    sink(file=file)

  for(i in 1:length(object)){
    cat(paste("\n\n********************* Response:", names(object)[i],
              "*********************\n\n"))
    print(object[[i]])
    cat("\n")
  }

  if(!missing(file))
    sink()
  invisible(1)
}


##############################################################################
#
```

```
savage <- function(x){
  nx <- length(x)
  ss.val <- ss <- numeric(nx)
  for(i in 1:nx)
    ss.val[i] <- sum(1/(i:nx))
  ss[sort.list(-x)] <- ss.val
  for(i in unique(x[duplicated(x)])){
    which <- x == i
    ss[which] <- mean(ss[which])
  }
  return(ss)
}

################################################################################

tdcc <- function(..., nx){
  #...: step.rs objects, typically replicates, with the same predictors
  #  on the same response variable
  # nx: number of predictors in the Sensitivity Analysis
  objects <- list(...)
  nr <- length(objects)
  SS <- matrix(numeric(nr*nx), nrow=nx)

  for(j in 1:nr){
    frac.var <- rep(-1,nx)
    if(length(objects[[j]]$order)>0)
      frac.var[objects[[j]]$order] <- length(objects[[j]]$order):1

#######  Checking this
#    frac.var[objects[[j]]$order] <- objects[[j]]$seq.sum$frac.var
#######

    SS[,j] <- savage(frac.var)
  }

  if(all(SS[,1]<(1+1E-12) & SS[,1]>(1-1E-12))){
    warning("There are no variables selected in the model")
    cor.mat <- diag(1,nr)
  }
  else
    cor.mat <- cor(SS)
  avg.cor <- (sum(cor.mat)-nr)/(nr*(nr-1))
  CT <- ((nr-1)*avg.cor+1)/nr
  #CT <- ( sum((rowSums(SS))^2) - nr^2*nx )/( nr^2 * (nx - sum(1/(1:nx))) )
  T.stat <- nr*(nx-1)*CT
  p.val <- 1 - pchisq(T.stat,nx-1)

  return(list(CT=CT, T.stat=T.stat, p.val=p.val))
}


################################################################################

tdcc2 <- function(objects, nx){
  # objects: list of step.rs objects
  # nx: number of predictors in the Sensitivity Analysis
  nr <- length(objects)
  SS <- matrix(numeric(nr*nx), nrow=nx)

  for(j in 1:nr){
    frac.var <- rep(-1,nx)
    if(length(objects[[j]]$order)>0)
      frac.var[objects[[j]]$order] <- length(objects[[j]]$order):1
```

```
#######  Checking this
#    frac.var[objects[[j]]$order] <- objects[[j]]$seq.sum$frac.var
#######

    SS[,j] <- savage(frac.var)
  }

  if(all(SS[,1]<(1+1E-12) & SS[,1]>(1-1E-12))){
    warning("There are no variables selected in the model")
    cor.mat <- diag(1,nr)
  }
  else
    cor.mat <- cor(SS)
  avg.cor <- (sum(cor.mat)-nr)/(nr*(nr-1))
  CT <- ((nr-1)*avg.cor+1)/nr
  T.stat <- nr*(nx-1)*CT
  p.val <- 1 - pchisq(T.stat,nx-1)

  return(list(CT=CT, T.stat=T.stat, p.val=p.val))
}

##############################################################################
#



tdcc.list <- function(..., nx, alt.obj, wtrue=F){
  #...: sensitivity objects, typically replicates, with the same predictors
  #  on the same response variables
  #alt.obj: optionally will add TDCC stats to "alt.obj" if provided
  # Computes the TDCC statistic between ... objects for each response

  objects <- list(...)
  ny <- length(objects[[1]])
  if(missing(alt.obj)){
    alt.obj <- list()
    for(i in 1:ny)
      alt.obj[[i]] <- list()
  }
  nr <- length(objects)
  rs.obs.i <- list()
  for(i in 1:ny){
    for(j in 1:nr){
      rs.obs.i[[j]] <- objects[[j]][[i]]
    }
    CT.i<- tdcc2(rs.obs.i,nx)$CT
    if(wtrue)
      alt.obj[[i]]$CTwtrue <- CT.i
    else
      alt.obj[[i]]$CT <- CT.i
  }
  return(alt.obj)
}


##############################################################################
#


## Create a sensitivity object of the truth for variable importance in
##  4 test models
## This is needed for the TDCC  ( a seq.sum$frac.var and order entry )
```

```
true.order <- function(){

  order1 <- c(2,1)
  var1 <- paste('x',order1,sep='')
  frac.var1 <- c(.9999,1.0000)
  Rsq1 <- frac.var1-c(0,frac.var1[-2])
  df1 <- rep(NA,length(frac.var1))
  pval1 <- rep(NA,length(frac.var1))
  press1 <- rep(NA,length(frac.var1))
  aps1 <- rep(NA,length(frac.var1))
  seq.sum1 <- list(frac.var=frac.var1, var=var1, Rsq=Rsq1, df=df1,
P.val=pval1,
                   APS=aps1, PRESS=press1)
  ans1 <- list(seq.sum=seq.sum1, order=order1, Adj.Rsq=NA, APS=NA, PRESS=NA,
               surface='TRUE',CT=NA, CTwtrue=1)

  order2 <- c(2,1)
  var2 <- paste('x',order2,sep='')
  frac.var2 <- c(.5196,1.0000)
  Rsq2 <- frac.var2-c(0,frac.var2[-2])
  df2 <- rep(NA,length(frac.var2))
  pval2 <- rep(NA,length(frac.var2))
  press2 <- rep(NA,length(frac.var2))
  aps2 <- rep(NA,length(frac.var2))
  seq.sum2 <- list(frac.var=frac.var2, var=var2, Rsq=Rsq2, df=df2,
P.val=pval2,
                   APS=aps2, PRESS=press2)
  ans2 <- list(seq.sum=seq.sum2, order=order2, Adj.Rsq=NA, APS=NA, PRESS=NA,
               surface='TRUE',CT=NA, CTwtrue=1)

  order3 <- 1:8
  var3 <- paste('x',order3,sep='')
  frac.var3 <- c(.7115,.9546,.9891,.9996,.9997,.9998,.9999,1.0000)
  Rsq3 <- frac.var3-c(0,frac.var3[-2])
  df3 <- rep(NA,length(frac.var3))
  pval3 <- rep(NA,length(frac.var3))
  press3 <- rep(NA,length(frac.var3))
  aps3 <- rep(NA,length(frac.var3))
  seq.sum3 <- list(frac.var=frac.var3, var=var3, Rsq=Rsq3, df=df3,
P.val=pval3,
                   APS=aps3, PRESS=press3)
  ans3 <- list(seq.sum=seq.sum3, order=order3, Adj.Rsq=NA, APS=NA, PRESS=NA,
               surface='TRUE',CT=NA, CTwtrue=1)

  order4 <- c(2,1,3)
  var4 <- paste('x',order4,sep='')
  frac.var4 <- c(.4463,.7593,1.0000)
  Rsq4 <- frac.var4-c(0,frac.var4[-2])
  df4 <- rep(NA,length(frac.var4))
  pval4 <- rep(NA,length(frac.var4))
  press4 <- rep(NA,length(frac.var4))
  aps4 <- rep(NA,length(frac.var4))
  seq.sum4 <- list(frac.var=frac.var4, var=var4, Rsq=Rsq4, df=df4,
P.val=pval4,
                   APS=aps4, PRESS=press4)
  ans4 <- list(seq.sum=seq.sum4, order=order4, Adj.Rsq=NA, APS=NA, PRESS=NA,
               surface='TRUE',CT=NA, CTwtrue=1)

  ans <- list(y1=ans1,y2=ans2,y3=ans3,y4=ans4)
  return(ans)
}

##############################################################################
```

This page intentionally left blank.