Expanding the Capabilities of Bayesian Regression With Interactions and Smooth Effects

by

Nathan Johnson

B.Sc, Okanagan University College 1998

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

Master of Science

in

THE FACULTY OF GRADUATE STUDIES

(Department of Statistics)

we accept this thesis as conforming
to the required standard

The University of British Columbia

September 2000

Abstract

The B-WISE (Bayesian regression With Interactions and Smooth Effects) method of regression modelling was designed to improve upon other flexible regression schemes in the areas of model interpretability and ease of implementation. B-WISE models have been shown to have good predictive performance, and performance can be even further improved with a natural Bayesian model averaging scheme. In this thesis I will outline some ways in which some of the constraints inherent in a B-WISE model can be relaxed, so that the technique can be used in more general situations and with even greater flexibility. It is shown that much of the interpretability of B-WISE models is retained and implementation is still relatively straightforward.

Contents

Acknowledgements

I would like to thank Paul Gustafson for his initiative and integrity in research, as well as the freedom with guidance he afforded me in the directions I took. Thanks to Bertrand Clarke for being a second reader, and to the rest of the faculty for their high commitment to teaching and mentoring. I have thoroughly enjoyed my time in the Department of Statistics and won't begin to mention the colleagues who contributed to a memorable two years.

NATHAN JOHNSON

The University of British Columbia

September 2000

To my parents — obviously!

Chapter 1

Introduction

Often an adequate summary of the relationship between a response variable Y and predictor variables $\mathbf{X} = (X_1, \ldots, X_p)$ is provided by a model

$$Y = f(X_1, \ldots, X_p) + \epsilon, \ \text{var}(\epsilon) = \sigma^2,$$

where ϵ is usually normally distributed with mean 0. The ability to predict Y for an $\mathbf{X}$ yet to be observed is perhaps the most important aspect of a good model, but often interpretability is a competing aspect. Investigators often want a model which is indeed a summary, making important features of the relationship apparent without conveying the more minute details. Hastie and Tibshirani (1990) give a thorough treatment of *additive models*, which have the form

$$Y = \alpha + \sum_{i=1}^{p} f_i(X_i) + \epsilon, \ \text{var}(\epsilon) = \sigma^2, \tag{1.1}$$

where $f_1, \ldots, f_p$ are unspecified smooth functions. The flexibility of the functions $f_1, \ldots, f_p$, coupled with the clarity and interpretability provided by the additive structure in (1.1), have made the additive model a standard tool for statistical analysis. Having obtained an additive model which adequately summarizes the relationship between $\mathbf{X}$ and Y, an investigator can find in f_i all of the model's information about the effect of X_i on the response Y. A scatterplot of Y against X_i, together with what Hastie and Tibshirani call the *scatterplot smooth*, f_i, is one appealing way to represent this information and assess its quality.

A large number of modelling schemes which go beyond additivity have been proposed for situations where the emphasis is on the predictive ability of $f(X_1, \ldots, X_p)$. These include surface smoothers, regression trees, MARS, projection pursuit regression, neural networks, gaussian processes, to name a few. Any of these methods can perform well in terms of prediction, but seldom do they lead to models that are interpretable. In particular, given a good fit $f(X_1, \ldots, X_p)$ it may be very difficult to elicit information about the effect of a particular predictor, yet at the same time the additive model may be too simplistic to be useful.

Significant research has been done to take the simple additive model one step further towards predictive ability, by describing possible bivariate interactions in predictor variables. We can write this extended additive model as

$$Y = \sum_i f_i(X_i) + \sum_{j<k} g_{jk}(X_j, X_k) + \epsilon, \ \operatorname{var}(\epsilon) = \sigma^2, \tag{1.2}$$

where both the univariate and bivariate effect functions are unspecified. Note that model (1.2) retains much of the simplicity and transparency of the simple additive model. In particular, the estimated effect of a predictor X_i on the response can still be graphed, either as a curve in two dimensions as before, or a set of surfaces in three dimensions, depending on the number of bivariate interactions in which X_i is involved. In this thesis we develop some extensions to the B-WISE scheme originally proposed by Gustafson (2000). B-WISE is a loose acronym for Bayesian Regression With Interactions and Smooth Effects. In a Bayesian framework this scheme models data using the expanded additive model (1.2) and uses a novel structure for the bivariate effect functions. Before giving a more thorough outline of the B-WISE method we describe cubic splines, which are fundamental to B-WISE, and we discuss briefly some of the more popular methods for multiple regression.

1.1 Cubic Smoothing Splines and Cubic Regression Splines

A scatterplot smooth is curve that to some extent summarizes the information contained in a scatterplot. A curve that is too rough and overfits the data will not be useful as a summary, and it will perform poorly as a predictor for new observations. On the other hand, a curve that is too smooth may not contain enough information about local features or curvatures. Among all functions with two continuous derivatives, it seems that a function f which could minimize the penalized residual sum of squares,

$$\sum_{i=1}^{n}(y_i - f(x_i))^2 + \lambda \int \{f''(t)\}^2 dt, \tag{1.3}$$

where $(x_1, y_1), \ldots, (x_n, y_n)$ are the data points, would satisfy both features of a good summary. The first term in (1.3) ensures that f contains enough information, that it follows the data closely; the second term ensures that it is not overly rough. Quantifying the roughness of f as $\int (f'')^2$ has intuitive appeal. If f is constant or linear then it has zero roughness. Green and Silverman (1994) tell us that a mechanical spline (e.g. a thin strip of wood) constrained to pass through a set of points will assume the shape of the function f which minimizes bending energy, a quantity which to first order is proportional to $\int f''^2$, provided the points are reasonably close to a straight line. Thus we have a loose physical interpretation for this quantification of roughness.

For the following discussion we assume the values of the predictor variable are ordered so that $x_1 < x_2 \cdots < x_n$ and there are no ties. Green and Silverman also tell us that among all functions with two continuous derivatives, the function which minimizes (1.3) is the *cubic smoothing spline.* A cubic smoothing spline is made up of $n-1$ cubic polynomials, one for each region $[x_i,\ x_{i+1}]$, $i = 1, \ldots, n-1$. The cubics are joined together in a smooth way by requiring that at each of the interior knots $x_2, \ldots, x_{n-1}$ the two cubics that join there must be equal in value and first and second derivatives. The second derivatives at the exterior knots x_1 and x_n are set to zero, making the spline a *natural cubic spline* (NCS). Outside of the exterior knots the NCS is linear. With these constraints in place the minimizer g of (1.3) is found by solving an $n \times n$ linear system whose solution is the vector of spline values $\mathbf{g} = (g(x_1), \ldots, g(x_n))^T$. A NCS is fully specified given its vector of knot-point values. To see this, note that there are originally $4(n-1)$ parameters to be specified. The constraints for smoothness number three for each of the $n-2$ interior knots, and the additional two requirements at the exterior knots bring the number of constraints to $3n-4$. Thus there are n free parameters to be determined by the n data points.

While the cubic smoothing spline does minimize (1.3) and there is an elegant algorithm available to solve the $n \times n$ linear system, yet there are difficulties in interpreting a curve that requires such a large number of parameters in its description. For our purposes a better smooth is the *cubic regression spline*, which utilizes a smaller number of knots, and thus requires that fewer parameters be estimated than the cubic smoothing spline. For simplicity we let the spline have m equally-spaced knots $t_1 < \ldots < t_m$, and in our examples m is typically close to 7. From the many ways to specify a NCS we choose to parameterize in terms of the knot values $\mathbf{g} := (g(t_1), \ldots, g(t_m))$, which seems to be most amenable to interpretation and specification of a prior distribution. Under this parameterization the values of the spline at arbitrary $\mathbf{x} = (x_1, \ldots, x_n)$ can be expressed as

$$\begin{bmatrix} g(x_1) \\ \cdot \\ \cdot \\ \cdot \\ g(x_n) \end{bmatrix} = Z\mathbf{g}, \tag{1.4}$$

where Z depends only on $\mathbf{x}$ and the knot locations. For the interested reader we close this section with details on the construction of Z, which is an essential procedure in the implementation of B-WISE.

We need to define two matrices Q and R. Let $h_i = t_{i+1} - t_i$ for $i = 1, \ldots, n-1$. The matrix Q is $m \times (m-2)$ with entries q_{ij}, $i = 1, \ldots, m$ and $j = 2, \ldots, m-1$, given by

$$q_{j-1,j} = h_{j-1}^{-1},\ g_{jj} = -h_{j-1}^{-1} - h_j^{-1},\ q_{j+1,j} = h_j^{-1}$$

for $j = 2, \ldots, m-1$, and all other entries are zero. The matrix R is $(m-2) \times (m-2)$ with entries r_{ij}, i and j running from 2 to $m-1$, given by

$$r_{ii} = \frac{1}{3}(h_{i-1} + h_i) \quad \text{for } i = 2, \ldots, m-1$$
$$r_{i,i+1} = r_{i+1,i} = \frac{1}{6}h_i \quad \text{for } i = 2, \ldots, m-2,$$

and all other entries are zero. Finally we let γ be the vector with the $m-2$ second derivative values $g''(t_i)$, $i = 2, \ldots, m-1$. One of the main results in Green and Silverman is stated in the following theorem.

The vectors $\mathbf{g}$ *and* γ *specify a natural cubic spline* g *if and only if*

$$Q^T\mathbf{g} = R\gamma.$$

If this condition is satisfied then the roughness penalty will satisfy

$$\int g''(t)^2 dt = \gamma^T R \gamma = \mathbf{g}^T Q R^{-1} Q^T. \tag{1.5}$$

(Note that R is invertible since it is strictly diagonal dominant and hence strictly positive-definite.)

The theorem gives a useful relation between $\mathbf{g}$ and γ, as well as a computationally convenient expression for the roughness penalty. The reader can verify that the value of the cubic spline at t is given by

$$\begin{aligned} g(t) &= \frac{(t-t_i)g_{i+1} + (t_{i+1}-t)g_i}{h_i} \\ &\quad -\frac{1}{6}(t-t_i)(t_{i+1}-t)\left\{\left(1+\frac{t-t_i}{h_i}\right)\gamma_{i+1} + \left(1+\frac{t_{i+1}-t}{h_i}\right)\gamma_i\right\} \\ &\quad \text{for } t_i \le t \le t_{i+1},\ i = 1, \ldots, n-1. \end{aligned}$$

In vector notation, using the above theorem, this can be written as

$$g(t) = \left[\mathbf{v}^T - \frac{1}{6}\left(\frac{t-t_i}{h_i}\right)\left(1-\frac{t-t_i}{h_i}\right)\mathbf{w}^T R^{-1} Q^T\right]\mathbf{g} = Z\mathbf{g},$$

where $\mathbf{v}$ is a vector with $v_i = \frac{t_{i+1}-t}{h_i}$, $v_{i+1} = \frac{t-t_i}{h_i}$, and all other components zero, and $\mathbf{w}$ is a vector with $w_i = 1 + \frac{t_{i+1}-t}{h_i}$, $w_{i+1} = 1 + \frac{t-t_i}{h_i}$, and all other components zero. Thus we have a construction for Z.

1.2 Some Methods for Multiple Regression

1.2.1 Thin Plate Splines and Tensor Product Splines

Thin plate splines and tensor product splines fall into a category of smoothers that Hastie and Tibshirani call "generic multivariate smoothers," which are basically generalizations of one-dimensional methods to higher dimensions. These methods are often useful for two or three dimensions, but in higher dimensions they lead to models that are difficult to interpret. A more fundamental problem with these methods has been called the "curse of dimensionality." Most smoothing methods rely on the concept of localness. The smoothed value of an observed response is determined mostly by points in a local neighbourhood of the response. But in high dimensions, local neighbourhoods are inherently sparse. For example, in a uniformly distributed data set of p predictors, a hypercube with sides that cover 10% of the range of each predictor will contain on average 0.1^p of the points in the data set. This sparsity has prompted the development of other specialized methods that can deal more effectively with high dimensional data, but the generic multivariate smoothers can be useful in two or three dimensions.

Thin plate splines are motivated by the generalization of the roughness measure $r(f) = \int \{f''(t)\}^2 \, dt$ to higher dimensions. The roughness measure for a bivariate function $f(x, y)$ analogous to the above measure for a univariate function is

$$r(f) = \int \int \left\{ \left(\frac{\partial^2 f}{\partial x^2} \right)^2 + 2 \left(\frac{\partial^2 f}{\partial x \partial y} \right)^2 + \left(\frac{\partial^2 f}{\partial y^2} \right)^2 \right\} dx dy. \tag{1.6}$$

Under this measure any plane in two dimensions has zero roughness. Green and Silverman note a physical interpretation similar to that for the univariate roughness measure. If an infinite elastic flat plate is deformed to the shape of the function ϵf for small ϵ then the bending energy of the plate is to first order proportional to $r(f)$. The minimizer of the penalized residual sum of squares, using this roughness measure, is a thin plate spline. We can generalize the roughness measure (1.6) to still higher dimensions, where again a thin plate spline minimizes the penalized residual sum of squares, but for clarity we limit our discussion to the two-dimensional case. To define a thin plate spline we first define a function $\eta(r)$ as

$$\eta(r) = \frac{1}{16\pi} r^2 \mathrm{log} r^2; \; \eta(0) = 0.$$

Denote the points in the predictor space by $\mathbf{t}_i = (x_i, y_i)$, $i = 1, \ldots, n$, and let $\|\mathbf{t}\|$ denote the usual Euclidean norm of $\mathbf{t}$. Then a function $g(\mathbf{t})$ is a thin plate spline on the data set $\mathbf{t}_1, \ldots, \mathbf{t}_n$ if and only if g is of the form

$$g(\mathbf{t}) = \sum_{i=1}^{n} \delta_i \eta(\|\mathbf{t} - \mathbf{t}_i\|) + a_1 + a_2 x + a_2 y,$$

for some constants $\delta_1, \ldots, \delta_n, a_1, a_2, a_3$. We see that regression models based on thin plate splines are hardly interpretable, but in two and three dimensions they can at least be analysed graphically.

Another, perhaps more natural, way to generalize univariate splines is by using tensor product splines. Following the discussion in Green and Silverman we consider the most useful case of two dimensions. The tensor product of two functions $f(t)$ and $g(u)$ is the function $(f \otimes g)(t, u) = f(t)g(u)$. Now suppose we have a set of linearly independent functions $\{\delta_{j_1} : i = 1, \ldots, q_1\}$ and another basis $\{\epsilon_{j_2} : j_2 = 1, \ldots, q_2\}$. The tensor product of these two bases is the set of all linear combinations of tensor products of basis functions

$$\left\{ \sum_{j_1=1}^{q_1} \sum_{j_2=1}^{q_2} a_{j_1 j_2} \delta_{j_1}(t) \epsilon_{j_2}(u) \right\}.$$

Let $\{\tau_1, \ldots, \tau_{m_1}\}$ and $\{\upsilon_1, \ldots, \upsilon_{m_2}\}$ be sequences such that the range of t is $T = [\tau_1, \tau_{m_1}]$, and the range of u is $U = [\upsilon_1, \upsilon_{m_2}]$. A tensor product cubic spline is constructed by dividing up $T \times U$ into panels of the form $[\tau_r, \tau_{r+1}] \times [\upsilon_s, \upsilon_{s+1}]$. Within each panel the function is the product of a cubic in t and a cubic in u, and at the joins between the panels the function is smooth. We omit the details, but suffice it to say that a large number of parameters must be estimated and, related to that issue, interpreting the function is difficult.

1.2.2 Regression Trees

Regression trees (Breiman, Friedman, Olshen and Stone 1984) carve up the predictor space into disjoint and exhaustive rectangles, modelling the response Y within each rectangle as a constant. The method begins by finding the optimal splitting point among all splitting points among all predictors. The optimal split is the one which divides the space into two regions such that the within region variance is minimal. Then for each of the two subregions the process is repeated until convergence, where further splitting does not significantly improve the within region variance. A regression tree can be presented graphically as a binary tree, or mathematically as in

$$f(\mathbf{X}) = \sum_i a_i I(\mathbf{X} \in R_i),$$

where the regions R_i are defined by the binary splits. There is no standard convergence criterion in use, but one popular strategy is to build a large tree and then prune it to a smaller size using cross-validation. The final tree is the one which has the smallest estimated prediction error.

The main drawback to regression trees is that they are discontinuous functions of the predictor variables. This leads to less than optimal predictive ability in situations where the true underlying relationship between $\mathbf{X}$ and Y is continuous. Also, while small trees are often very interpretable for practitioners, larger trees can be puzzling. However the adaptive placement of the 'knots', or discontinuities, provides efficient use of the data since more parameters are estimated only where the data requires them.

1.2.3 Multivariate Adaptive Regression Splines (MARS)

The recursive partitioning approach outlined above motivated the MARS methodology proposed by Friedman (1991). Like regression trees MARS uses adaptive knot placement, but produces a regression function that is continuous. Starting with the initial basis function $B_1(\mathbf{x}) = 1$, a search is made for the optimal splitting point among all splitting points among all predictors. That splitting point introduces a pair of new basis functions, $B_2(\mathbf{x}) = B_1(\mathbf{x})[+(x_i - t_i)]_+$ and $B_3(\mathbf{x}) = B_1(\mathbf{x})[-(x_i - t_i)]_+$, where x_i is the predictor being split, t_i is the optimal splitting point, and $[z]_+ = \max(0, z)$. Each iteration entails a search for the optimal splitting point for all of the basis functions present, and results in two new basis functions being introduced. The final regression function

$$f(\mathbf{X}) = \sum_i a_i B_i(\mathbf{X})$$

is a continuous function of the predictor variables (since all the basis functions are continuous). The basis functions have the form

$$B_i(\mathbf{x}) = \begin{cases} 1, & i = 1 \\ \prod_{j=1}^{J_i} [s_{ji}(x_{\nu(ji)} - t_{ji})]_+, & i = 2, 3, \ldots \end{cases}$$

where s_{ji} is the sign of each factor, $\nu(ji)$ is the index of the predictor in the jth factor of the ith basis function, t_{ji} is the splitting point, and J_i is the degree of interaction. We do not allow a predictor to occur more than once in a single basis function. As with regression trees, a popular strategy is to run the algorithm until a large number of basis functions have been formed, and then to delete the unimportant functions until some lack-of-fit measure is minimized.

Denison, Mallick and Smith (1998) put MARS in a Bayesian context by placing a prior distribution on the number k of basis functions present, and on the vector

$\theta^{(k)} = (\beta_1, \ldots, \beta_k)$, where $\beta_i = (a_i, T_i, t_{1i}, s_{1i}, \ldots, t_{Ji}, s_{Ji})$ contains information that completely determines the ith basis function. The variable T_i is the "type" of the basis function, defined to be the indicator of which predictors occur in the basis function. They then simulate samples from the joint posterior distribution

$$p(\theta^{(k)}, k|y) \propto p(k, \theta^{(k)}, y) = p(k)p(\theta^{(k)}|k)p(y|k, \theta^{(k)})$$

using an MCMC method. The main strength in BMARS over MARS is this MCMC sampling scheme by which Bayesian model averaging can be employed. Emperical results have shown that Bayesian model averages obtained from the BMARS algorithm have better predictive ability than good MARS models.

1.2.4 Projection Pursuit Regression

Projection pursuit regression (Friedman and Stuetzle 1981) models are of the form

$$Y = \sum_{k=1}^{K} h_k(\alpha_k^T \mathbf{X}) + \epsilon,$$

where $\alpha_k^T \mathbf{X}$ is a one-dimensional projection of $\mathbf{X}$ and h_k is an arbitrary smooth function. Because all of the smoothing is univariate there are no dimensionality problems. Interactions between predictor variables can be accounted for but are not easily understood from the functional form. In fact the functional form is not very enlightening at all for $K > 1$, but the judicious selection of projections does keep the number of parameters to a minimum.

1.2.5 Neural Networks

The simplest form of a neural network in the regression context can be written as

$$Y = \phi_0 \left(\alpha + \sum_h v_{hk} \phi_h \left(\alpha_h + \sum_i w_{ih} X_i \right) \right) + \epsilon, \tag{1.7}$$

where ϕ_h is almost always the logistic function, and ϕ_0 is a linear, logistic or threshold function. The pictorial representation in Figure (1.1) is helpful. At each of the hidden nodes the inputs $X_1, \ldots, X_p$ are multiplied by weights w_{ih} and summed. The functions ϕ_h are applied and the results are treated in the same way as inputs for the output node. The scheme originated in neuroscience as a model for electronic impulses in the brain. As such it is not a very realistic model, but as a nonlinear regression scheme it can be powerful. Besides model interpretability, which is basically nil, there are some complications in controlling the amount of fitting. Various suggestions have been made in the literature of neural networks for choosing the number of hidden nodes (which controls the amount of fitting). As with regression trees cross-validation is one possibility, but no procedure has been widely agreed upon. Variants of the simple model (1.7) may have more than one hidden layer and may incorporate "skip-layer" connections which link the input layer directly to the output. Parameter estimation is done in an iterative manner using Gauss-Newton steps and can be time consuming. Neal (1996) put parameter estimation in a Bayesian framework, and also showed that a neural network converges to a Gaussian process as the number of hidden nodes approaches infinity.

1.2.6 Gaussian Processes

A Gaussian process can be used to define a prior distribution over functions of one or more input variables. Suppose we have a response variable t which depends on a p parameter input $\mathbf{x}$. Denoting the repeats of t as $t^{(1)}, \ldots, t^{(n)}$ and the corresponding inputs as $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}$, a prior for the relationship between $\mathbf{x}$ and t is specified in terms of a covariance function $\text{Cov}\left[t^{(i)}, t^{(j)}\right]$ which depends on the inputs. For example,

$$\text{Cov}\left[t^{(i)}, t^{(j)}\right] = \sigma_\alpha^2 + \sum_{u=1}^{p} x_u^{(i)} x_u^{(j)} + \delta_{ij}\sigma_\epsilon^2$$

is the covariance function for the simple regression model

$$t^{(i)} = \alpha + \sum_{u=1}^{p} x_u^{(i)}\beta_u + \epsilon^{(i)}.$$

The covariance function

$$\text{Cov}\left[t^{(i)}, t^{(j)}\right] = \eta^2 \exp\left(-\sum_{u=1}^{p} \rho_u^2 (x_u^{(i)} - x_u^{(j)})^2\right) + \delta_{ij}\sigma_\epsilon^2$$

where η and ρ_u are hyperparameters, can be used to obtain a regression function based on arbitrary smooth functions. Letting C be the $n \times n$ matrix of values of the covariance function, it can be shown that the posterior predictive distribution of an unobserved $t^{(n+1)}$ is Gaussian with mean

$$E\left[t^{(n+1)} | t^{(1)}, \ldots, t^{(n)}\right] = \mathbf{k}^T C^{-1}\mathbf{t}$$

and variance

$$\text{Var}\left[t^{(n+1)} | t^{(1)}, \ldots, t^{(n)}\right] = \nu - \mathbf{k}^T C^{-1}\mathbf{k},$$

where $\mathbf{k}$ is the vector of covariances between $t^{(n+1)}$ and the n known targets, and ν is the prior variance of $t^{(n+1)}$ (given by $\text{Cov}\left[t^{(n+1)}, t^{(n+1)}\right]$). More details and illustrations can be found in Neal (1997).

The main drawback to using Gaussian processes is that they become computationally expensive as n increases. Memory requirements grow as n^2 and time requirements grow as n^3, but Neal suggests Gaussian processes can handle data sets with up to a thousand cases with modest computational resources.

1.3 B-WISE

B-WISE models a continuous response Y as a function of continuous predictors $X_1, \ldots, X_p$. For ease of interpretation we assume that the predictors have been rescaled so that 0 corresponds to a low value of the predictor and 1 corresponds to a high value. We assume the response Y is rescaled to have mean zero. Most variable selection techniques are aimed at deciding which of the p predictors should be included in the model and which should be excluded. In B-WISE, a predictor can be included as smooth effect, included as a linear effect, or excluded from the model. Thus a B-WISE model can be written as

$$Y = \alpha + \sum_{i \in M} f_i(X_i) + \sum_{(j,k) \in I} g_{jk}(X_j, X_k) + \epsilon, \tag{1.8}$$

where $\epsilon \sim N(0, \sigma^2)$, M indicates which predictors are included as main effects only and I indicates which predictors are involved in a bivariate interaction. A main effect function f_i can be either linear,

$$f_i(x_i) = \beta_i x_i,$$

or smooth,

$$f_i(x_i) = \beta_i x_i + s_i(x_i),$$

where $s_i(x_i)$ is a cubic regression spline constrained to be zero at the exterior knots 0 and 1.

The bivariate effect functions include a main effect for each variable, so a predictor involved in an interaction will appear only in the second summation of (1.8). Moreover a predictor may only appear once in (1.8), so that 'overlapping' interactions, as when g_{12} and g_{13} are both included, are not allowed. Several proposals have been made for the structure of nonparametric bivariate functions. One possibility is to let $g_{jk}(x_j, x_k) = g_{jk}(x_j)h_{jk}(x_k)$, where the two univariate functions are nonparametric (Cuzick 1993). The main drawback associated with this technique is that a large number of smooth functions must be estimated. Hastie and Tibshirani (1993) proposed varying-coefficient models, in which the univariate function for one of the predictors is nonparametric and acts as a coefficient for the other predictor. Thus fewer parameters need to be estimated and much flexibility is retained.

In the B-WISE scheme, if two variables x_j and x_k are both included as linear effects then their bivariate effect function g_{jk} would be modelled as the usual

$$g_{jk}(x_j, x_k) = \beta_j x_j + \beta_k x_k + \beta_{jk} x_j x_k.$$

Under this structure the effect of one predictor can be viewed as a linear effect, with slope and intercept that depend on the value of the other predictor. For intuition into the effect of x_j we might plot $g_{jk}(x_j, 0) = \beta_j x_j$ and $g_{jk}(x_j, 1) = \beta_k + (\beta_j + \beta_{jk})x_j$ on the same axes, with the understanding that the effect at an arbitrary x_k is a linear interpolation,

$$g_{jk}(x_j, x_k) = (1 - x_k)g_{jk}(x_j, 0) + x_k g_{jk}(x_j, 1).$$

This is a useful feature, since it can be difficult for a picture of a surface in three dimensions to communicate information about a bivariate effect. B-WISE models retain this feature when x_j is include as a linear effect and x_k is included as a smooth effect. In this case the bivariate effect function would be modelled as

$$g_{jk}(x_j, x_k) = \beta_j x_j + \beta_k x_k + s_k(x_k) + \beta_{jk} x_j x_k + x_j t_{jk}(x_k),$$

where $t_{jk}(x_k)$ is another cubic regression spline with exterior knots constrained to be zero. We could plot $g_{jk}(0, x_k) = \beta_k x_k + s_k(x_k)$ and $g_{jk}(1, x_k) = \beta_j + (\beta_k + \beta_{jk})x_k +$

$s_k(x_k) + t_{jk}(x_k)$ on the same axis, with the understanding that the effect of x_k at arbitrary values of x_j is a linear interpolation between the two smooth curves,

$$g_{jk}(x_j, x_k) = (1 - x_j)g_{jk}(0, x_k) + x_j g_{jk}(1, x_k).$$

Alternatively we might plot the two linear functions $g_{jk}(x_j, 0)$ and $g_{jk}(x_j, 1)$, but the interpolation for general values of x_k is no longer linear. In the case where both predictors are included as smooth effects then their bivariate effect function is modelled as

$$g_{jk}(x_j, x_k) = \beta_j x_j + s_j(x_j) + \beta_k x_k + s_k(x_k) + \beta_{jk} x_j x_k + x_j t_{jk}(x_k) + x_k t_{kj}(x_j).$$

In this situation we can plot the bivariate effect as a function of either variable while holding the other variable fixed. However, it would be prudent to plot more than just two curves since there is no linear interpolation interpretation. In all three cases for the bivariate effect function the surface is determined by the univariate functions at the edges of the unit square, thus facilitating interpretability and ease of computation.

The aim of the B-WISE scheme is to rank all, or many, possible models according to their simplicity and predictive ability. This ranking is made possible by means of a Bayesian paradigm in which a posterior probability is calculated for each model. This probability then serves as a measure of a model's quality, in terms of the criteria emphasized under this scheme. We assume that $Y|\theta, \sigma^2, \lambda \sim N(A_\lambda \theta, \sigma^2 I)$, where θ is a parameter vector with an intercept, slope terms, and the values of the splines at the knot points. A_λ is the design matrix for model λ. We use a prior of the form $p(\theta, \sigma^2, \lambda) = p(\theta|\sigma^2, \lambda)p(\sigma^2)p(\lambda)$, where $p(\sigma^2) \propto \sigma^{-2}$, and $p(\lambda)$ is uniform on the model space Λ. For $p(\theta|\sigma^2, \lambda)$ we use the g-prior,

$$\theta|\sigma^2, \lambda \sim N\left(0, \frac{\sigma^2}{c}\left(\frac{A_\lambda^T A_\lambda}{n}\right)^{-1}\right),$$

but in a slightly modified form that penalizes models according to their degree of roughness, $r(f) = \int\{f''(t)\}^2 dt$. The modified g-prior makes use of a roughness matrix R such that

$$v^T R v = r(f),$$

where v is the vector of spline values at the knot points (see equation 1.5). To penalize roughness we give v the prior distribution $v \sim N\{0, (kR)^{-1}\}$, where k determines the strength of the roughness penalty. Now to quantify the roughness of an entire model we use the matrix R to construct an overall roughness matrix R_λ that satisfies

$$\theta^T R_\lambda \theta = \sum_j r(s_j) + \sum_k \frac{r(s_k) + r(s_k + t_k)}{2}, \qquad (1.9)$$

where j ranges over smooth predictors involved in a main effect only, and k ranges over smooth predictors involved in an interaction. This is, in some sense, the sum of the average roughnesses of all the predictors. Here a predictor involved in an interaction has an average roughness that depends on its effect when the interacting variable is at a 'high' value and on its effect when the interacting variable is at a 'low' value. One can verify that R_λ is everywhere zero except for R on the diagonal blocks corresponding to the parameters for s_i, $R/2$ on the diagonal blocks for t_i, and $R/2$ on the off-diagonal blocks for the s_i-t_i pairs. Then the final form of the prior is

$$\theta|\sigma^2, \lambda \sim N\left(0, \sigma^2 \left(\frac{cA_\lambda^T A_\lambda}{n} + kR_\lambda\right)^{-1}\right),$$

where the hyperparameters c and k must be set in order for the prior to be specified completely.

With the prior thus specified, the conditional $(\theta|\sigma^2, \lambda, y)$ is normally distributed with mean

$$\hat{\theta}_\lambda = \left[\left(1+\frac{c}{n}\right) A_\lambda^T A_\lambda + kR_\lambda\right]^{-1} A_\lambda^T y,$$

and variance $\sigma^2[(1+c/n)A_\lambda^T A_\lambda + kR_\lambda]^{-1}$. The distribution of $\sigma^2|\lambda, y$ is inverse gamma with shape parameter $n/2$ and scale parameter $\{y^T(I-H_\lambda)y\}/2$, where

$$H_\lambda = A_\lambda \left[\left(1+\frac{c}{n}\right) A_\lambda^T A_\lambda + kR_\lambda\right]^{-1} A_\lambda^T$$

is the hat matrix for model λ. The distribution of $\lambda|y$ is

$$p(\lambda|y) \propto \frac{\left|\left(\frac{c}{n}\right) A_\lambda^T A_\lambda + kR_\lambda\right|^{1/2}}{\left|\left(1+\frac{c}{n}\right) A_\lambda^T A_\lambda + kR_\lambda\right|^{1/2}} \{y^T(I-H_\lambda)y\}^{-n/2}. \qquad (1.10)$$

Note that (1.10) has some intuitive appeal. We can write $y^T(I-H_\lambda)y$ as

$$y^T(I-H_\lambda)y = \{||y-\hat{y}_\lambda||^2 + k\hat{\theta}_\lambda^T R_\lambda \hat{\theta}_\lambda\} + \frac{c}{n}||\hat{y}_\lambda||^2,$$

where $\hat{y}_\lambda = H_\lambda y$ are the fitted values. The term in braces is just the penalized residual sum of squares. The other factor in (1.10) can be interpreted as a penalty for model complexity. In the simple case where $k=0$ we have

$$p(\lambda|y) \propto \exp\left\{-\frac{n}{2}\log\left(||y-\hat{y}_\lambda||^2 + \frac{c}{n}||\hat{y}_\lambda||^2\right) - \frac{d(\lambda)}{2}\log\left(1+\frac{n}{c}\right)\right\},$$

where $d(\lambda)$ is the dimension of θ under model λ. Here we can easily see how the posterior probability involves a tradeoff between goodness of fit and the number of parameters in the model.

1.4 Identifying Models With High Posterior Probability

Given predictor variables $X_1, \ldots, X_p$ the collection of models of form (1.8) make up a model space Λ. When p is small it is possible to evaluate every model in Λ. For larger p it may not be feasible to evaluate every model, yet there are techniques available for locating models with high posterior probabilities. George and McCulloch (1993) describe one such Markov chain Monte Carlo (MCMC) technique, and George and McCulloch (1997) give advice for fast model space exporation. For B-WISE Gustafson proposes an MCMC inspired algorithm which makes use of the nonnormalized posterior probability (1.10). When this probability is available up to a normalizing constant the search algorithm need not contain the accept-reject step usually found in MCMC algorithms. To describe the B-WISE search algorithm we define a *neighbourhood* of a particular model to contain all models obtained by

- including a predictor not already in the model
- upgrading the linear effect of a predictor to a smooth effect
- including an interaction between two variables present in the model and not already involved in interactions
- downgrading the smooth effect of a predictor to a linear effect
- removing a predictor included in the model as a linear effect
- removing an interaction term
- partner switching two interaction terms (eg. changing $A \times B$ and $C \times D$ interactions to $A \times C$ and $B \times D$ interactions.

The algorithm begins by evaluating the nonnormalized posterior probability of all models in the neighbourhood of an initial model λ_1 chosen by the investigator. The nonnormalized probabilities are normalized relative to this neighbourhood and the next model λ_2 is chosen randomly from the neighbourhood according to these probabilities. The same procedure is performed for λ_2, and is repeated until some convergence criterion is satisfied. Gustafson suggests that the algorithm has likely converged if it fails to find a model in

$$\Lambda_\alpha = \{\lambda : p(\lambda|y) \geq \alpha^{-1}\text{max}_\lambda p(\lambda|y)\} \tag{1.11}$$

for t iterations in a row. The maximum in (1.11) is taken over all models evaluated by the algorithm so far, and Gustafson used $\alpha = 20$ and $t = 5$. The model search algorithm is more fully detailed in Appendix A.

Having performed a complete run of the algorithm we may find that a single best model will suit our purposes, or we may form a Bayesian model average using the Occam's window approach described in Raftery, Madigan and Hoeting (1997). Bayesian model averages are often found to have better predictive ability than a single best model, but in the B-WISE scheme dramatic improvements have not been seen.

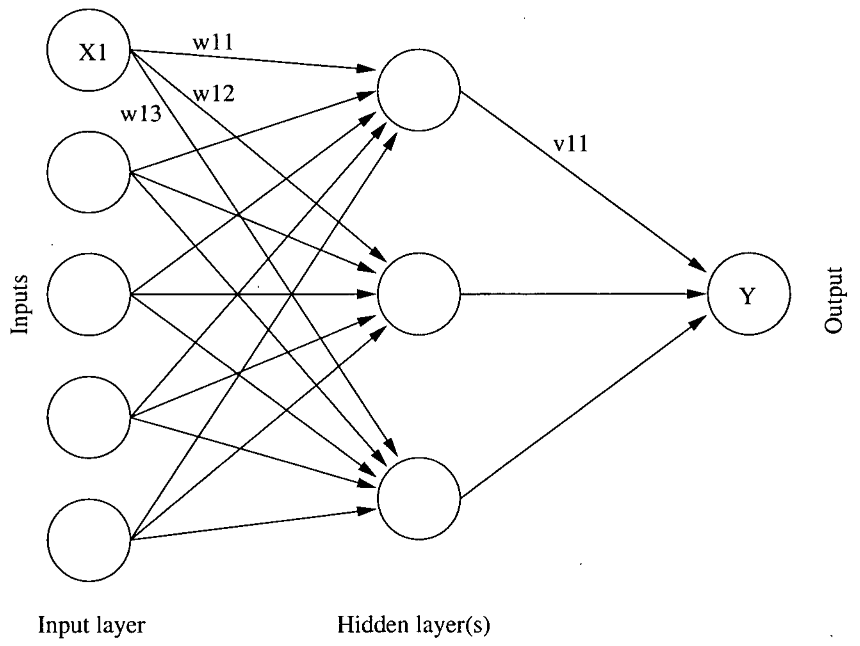


Figure 1.1: A pictorial representation of model (1.7). At node h of the hidden layer the inputs are multiplied by weights and summed. The result is inputted to a function ϕ_h, whose value is then treated as an input to the next layer of the network.

Chapter 2

Extensions to B-WISE

There are at least two limitations of the B-WISE scheme as it stands. The first is that overlapping interactions are not permitted, so a given predictor variable can interact with at most one other. Yet in some studies a predictor variable may well interact with several other predictor variables. One example might be a study in human health with gender as a predictor variable. Here an investigator may well expect the gender variable to interact with every other predictor variable. The second limitation of B-WISE is also apparent in this example. As of yet no provision has been made for categorical predictor variables. In this section we discuss how B-WISE can be extended to allow overlapping interactions and categorical predictors.

2.1 Overlapping Interactions

The model (1.8) has an appealing structure. To investigate the model's information about a particular predictor one need examine only one term of the model, which will be either a univariate function f_i or a bivariate function g_{ij}. To incorporate overlapping interactions we must assume a different structure underlying (1.8). The problem is that if both g_{ij} and g_{ik} appear in the model, there is no obvious place to put the main effect for predictor X_i, except perhaps in a separate main effect term f_i. This is the approach we take, so that the first summation in (1.8) contains the main effects for all the predictors in the model, and the second summation contains the bivariate interactions for any pairs of variables which interact. Thus, to investigate the model's information about a particular predictor, one must collect all of the terms involving that predictor.

With a predictor being able to interact with several others, there comes the question of how to quantify the roughness of a predictor's effect function. The old way is given by equation (1.9), where the roughness is given by $r(s_i)$ if X_i is not involved in any interactions, and by $(r(s_i) + r(s_i + t_i))/2$ if X_i is involved in an interaction. We can build on this idea of averaging over the high and low levels of the interacting variable. If predictor X_i interacts with d other predictors $X_{j_1}, \ldots, X_{j_d}$ then there are 2^d different high-low combinations over which to average. If we put all of the interaction splines into a vector $\mathbf{T}_i = (t_{ij_1}, \ldots, t_{ij_d})$ then we can write the roughness of the effect for X_i as

$$2^{-d} \sum_I r(s_i + I^T \mathbf{T}_i)$$

where I ranges over all vectors of length d with elements from $\{0, 1\}$. This simplifies to the expression given for the non-overlapping case where $d = 1$.

To specify the prior for the parameter vector θ we need a matrix R_λ that satisfies

$$\theta^T R_\lambda \theta = \sum_k \left\{ 2^{-\dim(\mathbf{T}_k)} \sum_I r(s_k + I^T \mathbf{T}_k) \right\},$$

where k ranges over all predictors with smooth effects. Fortunately such an R_λ is fairly easy to construct. If a predictor X_i interacts with d others then we put a

$(d+1) \times (d+1)$ block matrix into R_λ, each block being the roughness matrix R multiplied by a special factor. This special factor will be 1 in the upper left block, 1/2 in the rest of the top row, left column and diagonal, and 1/4 in the remaining blocks.

Example: To show that the above construction works we give an example where a smooth predictor X_1 interacts with both X_2 and X_3. Let $\mathbf{a}$ be the parameter vector for the spline $s_1(x_1)$, $\mathbf{b}$ be the parameter vector for the spline $t_{12}(x_1)$ and $\mathbf{c}$ be the parameter vector for the spline $t_{13}(x_1)$. Then

$$\begin{aligned}
\left[\mathbf{a}^T \mathbf{b}^T \mathbf{c}^T\right] R_\lambda \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \end{bmatrix} &= \left[\mathbf{a}^T \mathbf{b}^T \mathbf{c}^T\right] \begin{bmatrix} R & \frac{1}{2}R & \frac{1}{2}R \\ \frac{1}{2}R & \frac{1}{2}R & \frac{1}{4}R \\ \frac{1}{2}R & \frac{1}{4}R & \frac{1}{2}R \end{bmatrix} \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \end{bmatrix} \\
&= \frac{1}{4}[\mathbf{a}^T R\mathbf{a} + (\mathbf{a}+\mathbf{b})^T R(\mathbf{a}+\mathbf{b}) + (\mathbf{a}+\mathbf{c})^T R(\mathbf{a}+\mathbf{c}) \\
&\quad + (\mathbf{a}+\mathbf{b}+\mathbf{c})^T R(\mathbf{a}+\mathbf{b}+\mathbf{c})] \qquad (2.1) \\
&= \frac{1}{4}\left[r(s_1) + r(s_1+t_{12}) + r(s_1+t_{13}) + r(s_1+t_{12}+t_{13})\right],
\end{aligned}$$

which is the average roughness of the effect function for X_1.

To show that the construction for R_λ works in general, suppose X_i interacts with $d = \dim(\mathbf{T}_i)$ other predictors and consider the form of equation (2.1). All 2^d terms have an $\mathbf{a}^T R\mathbf{a}$. When we divide by 2^d in forming the average the result is that $\mathbf{a}^T R\mathbf{a}$ has a coefficient of 1. Hence the entry in the upper left block of R_λ. There are three other kinds of terms found in equation (2.1): terms of the form $\mathbf{x}^T R\mathbf{x}$ where $\mathbf{x}$ is a spline other than a, terms of the form $\mathbf{a}^T R\mathbf{x}$ (or $\mathbf{x}^T R\mathbf{a}$), and terms of the form $\mathbf{x}^T R\mathbf{y}$ where both $\mathbf{x}$ and $\mathbf{y}$ are splines other than $\mathbf{a}$. Terms of the first kind come from a larger term $(\mathbf{a}+\cdots+\mathbf{x}+\cdots)^T R(\mathbf{a}+\cdots+\mathbf{x}+\cdots)$, and there are 2^{d-1} possibilities for the exact form of this term (there are $d-1$ splines other than $\mathbf{x}$ which are either present or absent from this larger term). Dividing by 2^d in taking the average gives $\mathbf{x}^T R\mathbf{x}$ a coefficient of $2^{d-1}/2^d = 1/2$. Hence the diagonal entries of R_λ. The same reasoning applies to $\mathbf{a}^T R\mathbf{x}$ (or $\mathbf{x}^T R\mathbf{a}$), which explains the top row and left column of R_λ. Terms of the third kind come from a larger term $(\mathbf{a} + \cdots + \mathbf{x} + \cdots + \mathbf{y} + \cdots)^T R(\mathbf{a} + \cdots + \mathbf{x} + \cdots + \mathbf{y} + \cdots)$, and there are 2^{d-2} possibilities for the exact form of this term. Dividing by 2^d means the term $\mathbf{x}^T R\mathbf{y}$ has a coefficient of $1/4$. Hence all the remaining blocks of R_λ.

2.2 Categorical Variables

Incorporating categorical variables into the B-WISE setup is fairly straightforward given the work done so far. The key assumption is that each predictor has an additive effect on the response variable, and that effect may depend on the values of several other predictors. We will keep equation (1) as our model equation and remain in the wider setting where overlapping interactions are allowed.

If a categorical predictor X_i has a main effect only, we write its effect function as

$$f_i(X_i) = \beta_{i2} I(X_i = x_{i2}) + \cdots + \beta_{im} I(X_i = x_{im}),$$

where $x_{i1}, \ldots, x_{im}$ are the m mutually exclusive values which X_i can take. The effect of X_i being at level x_{i1} is now hidden in the intercept α of equation (1.8). Note that when X_i is a binary (0/1) categorical variable, we can take $f_i(X_i) = \beta_i X_i$, as we do for continuous variables.

As with all other types of interactions, when we allow a categorical predictor to interact with another predictor, we also include main effect functions for each of the variables involved in the interaction. These main effect functions have the same form as they would if the same two predictors were in the model but did not interact with each other. Thus, allowing an interaction between two variables already in a model amounts to only a modest increase in the complexity of the design matrix. We need to consider three types of interactions in which a categorical predictor X_i may be involved: an interaction with a linear predictor, an interaction with a smooth predictor, and an interaction with another categorical predictor.

If X_i interacts with a linear predictor X_j we code the bivariate effect function as

$$g_{ij}(X_i, X_j) = I(X_i = x_{i2})(\alpha_{j2} + \beta_{j2}X_j) + \cdots + I(X_i = x_{im})(\alpha_{jm} + \beta_{jm}X_j).$$

In this case the effect of X_i being at level x_{i1} is hidden in the intercept α and the main effect function $f_j(X_j)$.

If categorical predictor X_i interacts with a smooth predictor X_j, one possibility is to code the bivariate effect function as

$$\begin{aligned} g_{ij}(X_i, X_j) &= I(X_i = x_{i2})(\alpha_{j2} + \beta_{j2}X_j + s_{j2}(X_j)) + \cdots \\ &\quad + I(X_i = x_{im})(\alpha_{jm} + \beta_{jm}X_j + s_{jm}(X_j)), \end{aligned}$$

but this causes complications when we try to measure the average roughness of the smooth predictor X_j. We return to this issue later. A better choice, perhaps, is to model the bivariate effect function as

$$g_{ij}(X_i, X_j) = I(X_i = x_{i2})(\alpha_{j2} + \beta_{j2}X_j) + \cdots + I(X_i = x_{im})(\alpha_{jm} + \beta_{jm}X_j),$$

so that the same spline (which would appear in the main effect function $f_j(X_j)$) is used for each category, but the linear effect is allowed to vary. This is the approach we favoured, with the hope that under this model the savings in the number of parameters to be estimated would make up for any loss in flexibility. The roughness matrix to be constructed under this approach is no different than that in the case where the categorical and smooth predictor do not interact, since there are no additional splines in the model to contribute to the model's roughness.

In the final case, we may have X_i interacting with another categorical predictor X_j. In this case we model the bivariate effect function as

$$g_{ij}(X_i, X_j) = \sum_{k=2}^{m}\sum_{l=2}^{n}\gamma_{ij}I(X_i = x_{ik}, X_j = x_{jl}),$$

m being the number of categories of X_i, and n being the number of categories of X_j. The effect when either of these predictors is in its first category shows up in the main effect functions.

In extending B-WISE to allow categorical predictors, we have made provision for bivariate interactions between categorical and smooth predictors, but we have constrained the smooth predictor's effect function to use the same spline for each level of the categorical variable while allowing the linear piece to differ. We concede that important features of data could be missed under this modelling technique. In the remainder of this chapter we outline the construction of the roughness matrix R_λ under the more flexible conditions where a smooth effect function for a predictor is allowed to use a different spline term for each of an interacting categorical predictor's levels. We do this in such a way that $\theta R_\lambda \theta$ may still be interpreted as the sum of the average roughnesses of all the predictors, and hence (1.10) remains interpretable. Now suppose a smooth predictor X_1 interacts with a number of other continuous predictors $X_2, \ldots, X_{d+1}$ and a number of categorical predictors $W_1, \ldots, W_t$ with $c_1, \ldots, c_t$ categories, respectively. For each of the 2^d high-low combinations of the continuous predictors there are $c_1 \cdots c_t$ category combinations of the categorical predictors. We define the average roughness of X_1 to be the average of the roughnesses of all $2^d c_1 \cdots c_t$ effect functions for X_1. Let $w_{i2}, w_{i3}, \ldots, w_{ic_i}$ be parameter vectors for the splines corresponding to the different levels of predictor W_i. To build R_λ we reconsider the form of equation (2.1). Each of the $2^d c_1 \cdots c_t$ terms would contain an $a^T Ra$, and dividing by the number of terms gives the upper left block of R_λ a coefficient of 1. The diagonal blocks corresponding to continuous interacting variables will have a coefficient of $\frac{1}{2}$, since there are $2^{d-1}c_1 \cdots c_t$ terms of the form $(a+\ldots+x+\ldots)^T R(a+\ldots+x+\ldots)$, where x is the spline corresponding to the high value of an interacting continuous predictor. In fact one can see fairly easily that all blocks corresponding to continuous predictors will have the same multipliers as before. Now, blocks on the diagonal corresponding to categorical predictor W_i will have a coefficient of $\frac{1}{c_i}$, since a term $(a+\ldots+w_{ij}+\ldots)^T R(a+\ldots+w_{ij}+\ldots)$ occurs $2^d c_1 \cdots c_{i-1}c_{i+1} \cdots c_t$ times. The same reasoning applies to the top row and left column blocks. Blocks off the diagonal corresponding to categorical variables W_i and W_j, $i \neq j$ will have a coefficient of $\frac{1}{c_i c_j}$ and blocks off the diagonal corresponding to the same categorical predictor, eg. to the splines w_{i2} and w_{i3}, will be zero. Lastly, blocks corresponding to categorical predictor W_i and a smooth predictor will have a coefficient of $2^{d-1}c_1 \cdots c_{i-1}c_{i+1} \cdots c_t$. Obviously the computer code will have to be meticulous. This further increase in flexibility has good potential but of course the increase in the number of parameters requires a larger amount of data for estimation.

Chapter 3

Examples

3.1 Simulated Data

Denison, Mallick and Smith (1998) tested BMARS on an example from Friedman (1991). Gustafson (2000) compared B-WISE with BMARS using the same example, and we will use it again to see if the extensions to B-WISE discussed above are helpful. The data consist of predictor variables $\mathbf{X} = (X_1, \ldots, X_p)$ simulated from a p-dimensional unit hypercube, and a response variable Y generated independently as $Y|\mathbf{X} \sim N(f(\mathbf{X}), \sigma^2)$, where

$$f(\mathbf{x}) = 10\text{sin}(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5,$$

and $\sigma^2 = 1$. In one version of the problem $(n,p) = (200, 6)$ (one predictor having no relation to Y), and in another version $(n,p) = (100, 10)$ (five predictors having no relation to Y). Rather than simulate the data as prescribed above we use the data made available by Denison, Mallick and Smith, which is also the data that Gustafson analyzed.

Gustafson found that in both versions of the problem B-WISE placed the

Table 3.1: Simulated data set with $(n, p) = (200, 6)$. The top five models found in one run of the model search algorithm.

Model	Posterior Probability
$S(X_1) \otimes S(X_2) + S(X_3) + L(X_4) + L(X_5)$	0.366
$S(X_1) \otimes S(X_2) + S(X_3) + S(X_4) + L(X_5)$	0.186
$S(X_1) \otimes S(X_2) + S(X_3) + L(X_4) + S(X_5)$	0.080
$S(X_1) \otimes S(X_2) + S(X_3) + L(X_4) \otimes L(X_5)$	0.061
$S(X_1) \otimes S(X_2) + S(X_3) + S(X_4) + S(X_5)$	0.041

highest posterior probability on the qualitatively correct model, that is, the model with smooth effects for X_1, X_2 and X_3, linear effects for X_4 and X_5, and an interaction between X_1 and X_2. Computational resources do not conveniently allow us to evaluate every model in the $p = 6$ case, let alone the $p = 10$ case, but on several runs of the model search algorithm the qualitatively correct model was located and had the highest posterior probability of any model. Apparently the scheme penalizes model complexity in a way that prevents overfitting, and yet can describe fairly complicated data structures in detail. Table 3.1 lists the top five models found in one run of the model search algorithm, along with their posterior probabilities normalized relative to all models visited on that run. These are the same as the top five models found by Gustafson with the original B-WISE scheme.

Table 3.2 lists the top five models for the problem involving 100 points and 10 predictors. These models are not the same as the top five models in Gustafson, but the results there also show that X_{10} makes some contribution to predictive ability. We think the models listed in Table 3.2 represent a more compact cluster of models around the best model than those found by Gustafson.

Figure 3.1 shows the univariate effects of the B-WISE best model and a BMARS model average for the $p = 6$ case. By "univariate effect" we mean the func-

Table 3.2: Simulated data set with $(n, p) = (100, 10)$. The top five models found in one run of the model search algorithm.

Model	Posterior Probability
$S(X_1) \otimes S(X_2) + S(X_3) + L(X_4) + L(X_5)$	0.112
$S(X_1) \otimes S(X_2) + S(X_3) + L(X_4) + L(X_5) + S(X_1) \otimes L(X_{10})$	0.105
$S(X_1) \otimes S(X_2) + S(X_3) + L(X_4) + L(X_5) + L(X_{10})$	0.055
$S(X_1) \otimes S(X_2) + S(X_3) + L(X_4) \otimes S(X_5) + S(X_1) \otimes L(X_{10})$	0.046
$S(X_1) \otimes S(X_2) + S(X_3) + S(X_4) + L(X_5) + S(X_1) \otimes L(X_{10})$	0.045

tion defined by holding all but one predictor variable constant in the fitted model. As such, a univariate effect can only be defined up to a vertical translation, since it is not clear what the intercept should be. The effects in Figure 3.1 have been vertically translated by eye so as to line up with the data. Much information regarding each of the fitted functions cannot be seen in the univariate effects, but the pictures do help to characterize the differences between the two regression methods. One apparent consequence of the model parsimony associated with B-WISE is that B-WISE curves are less flexible. The hyperparameters c and k are certainly determining factors here, and could be changed to make the univariate effects somewhat more dramatic than those shown in Figure 3.1. It is also interesting that even a BMARS model average has sharp bends. This could be due to a lack of diversity in the models that make up the average.

Also of interest are the joint effect functions for X_1 and X_2, shown in Figures 3.2 and 3.3. Here again B-WISE is seen to produce a smoother, less flexible surface. Yet we will see in the next example that maintaining interpretable algebraic forms for models does not necessarily result in models with little predictive ability, as B-WISE can actually outperform BMARS in this respect.

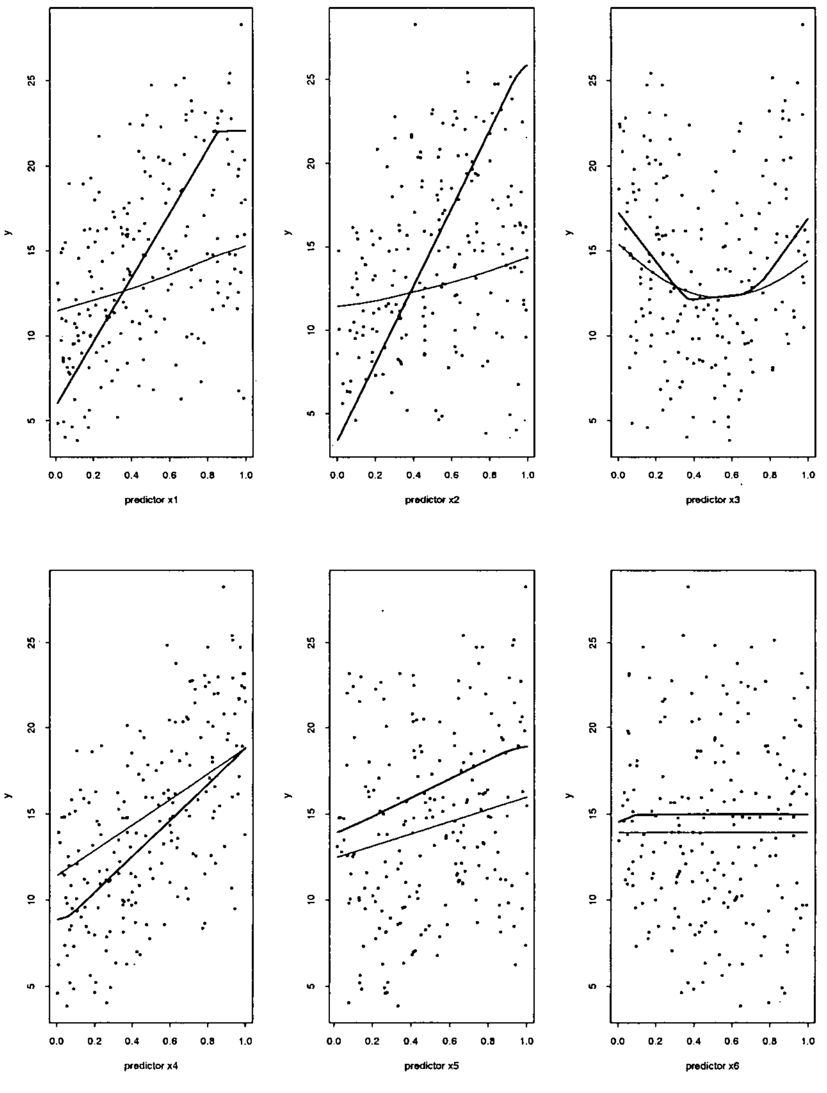


Figure 3.1: The B-WISE univariate effects are shown with a lighter line, and the BMARS effects with a heavier line. Vertical translations are not of interest here; only the shapes of the curves are.

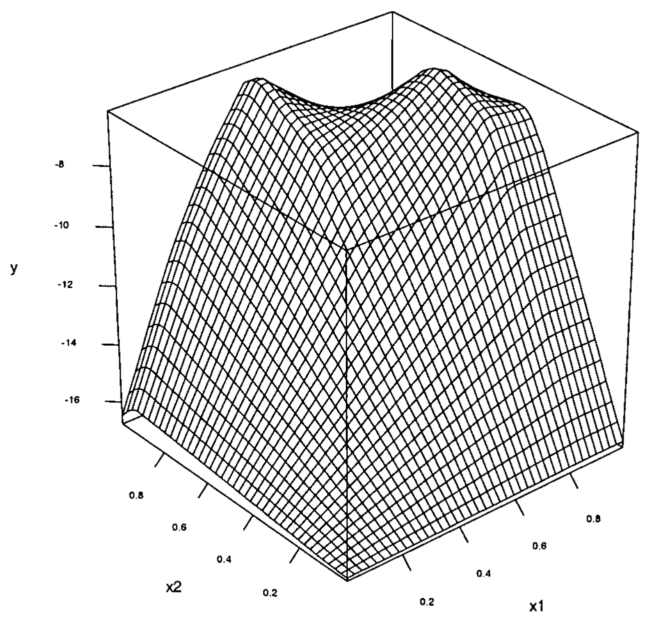


Figure 3.2: The BMARS joint effect function for X_1 and X_2 seems to be quite flexible.

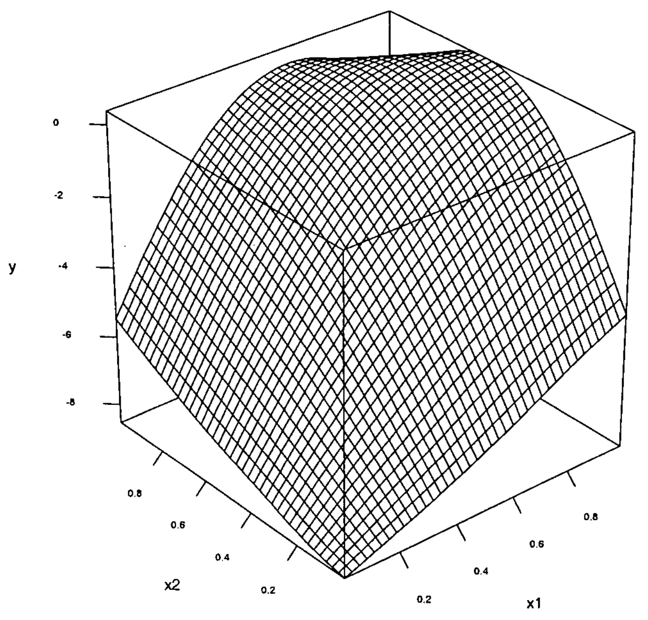


Figure 3.3: The B-WISE joint effect function for X_1 and X_2 is smoother than the corresponding BMARS function.

3.2 Abalone Data

The data for this example are made available by the Department of Computer Science at the University of Toronto. Here 9 variables are recorded for each of 4177 abalone (molluscs). Our goal is to predict the number of rings in each abalone using the other variables as predictors (the age in years is roughly the number of rings plus 1.5). We treat the number of rings as a continuous variable, but in fact it has values on the integers ranging from 1 to 29. To facilitate comparison with regression methods that do not allow categorical variables we will for the present omit the "gender" variable, which classifies each abalone as either male, female, or infant.

An analysis of the residuals from a B-WISE fit to all of the data showed that a log transformation of the response gave better agreement with the constant variance assumption. Figures 3.4 and 3.5 show diagnostics for a B-WISE fit to all of the data, using a log transformation on the response. The assumption of constant variance is tenable, as is the assumption of normal errors.

We performed a five-fold cross-validation, using roughly 4/5 of the data as a training set to fit a model and 1/5 of the data as a validation set to assess the model. Denoting the i-th observation of the validation set as $(x_i^*,\ y_i^*)$, we assess the predictive ability of a model $\hat{f}$ using the root-mean-squared prediction error,

$$RMSPE = \left[\frac{1}{m} \sum_{i=1}^{m} \{\hat{f}(x_i^*) - y_i^*\}^2 \right]^{1/2},$$

where m is the number of points in the validation set. We terminated the B-WISE model search algorithm if for five consecutive iterations it failed to locate a model in Λ_2, and we terminated the BMARS algorithm after $20,000$ iterations, following the recommendation of the BMARS creators.

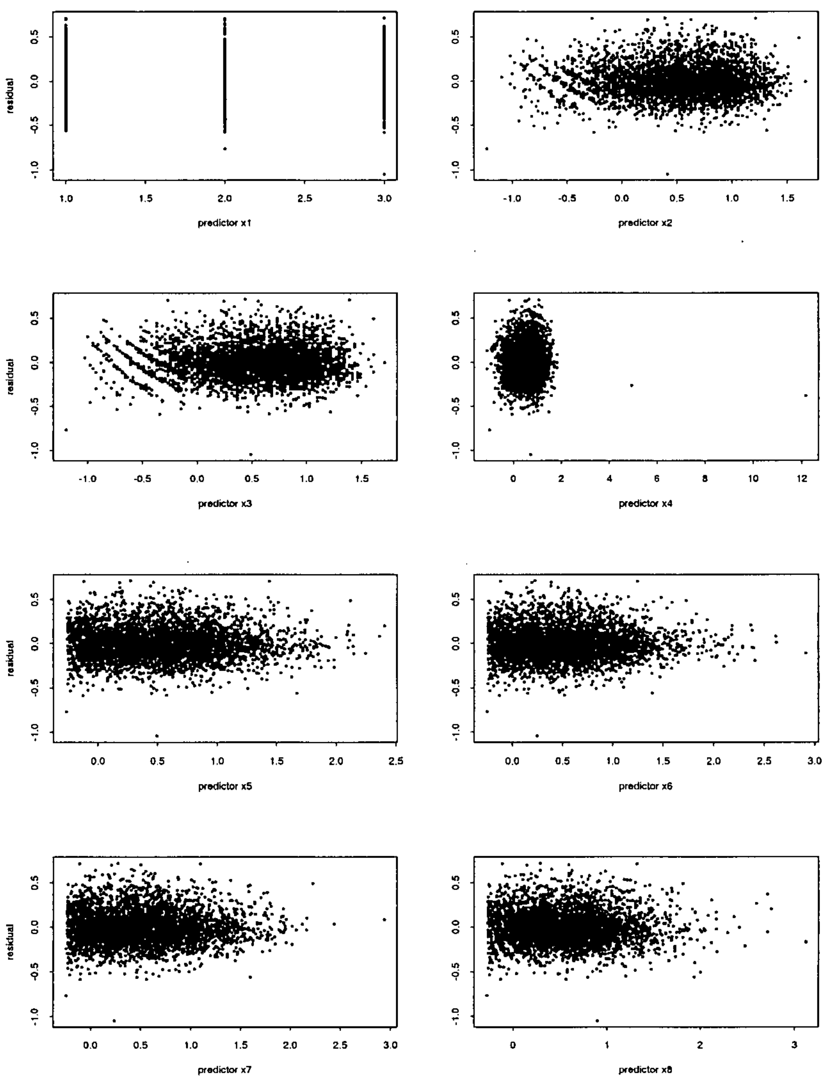


Figure 3.4: The assumption of constant variance is nearly satisfied, but some funnelling occurs in the latter predictors. We are satisfied because of the simplicity of the log transformation.

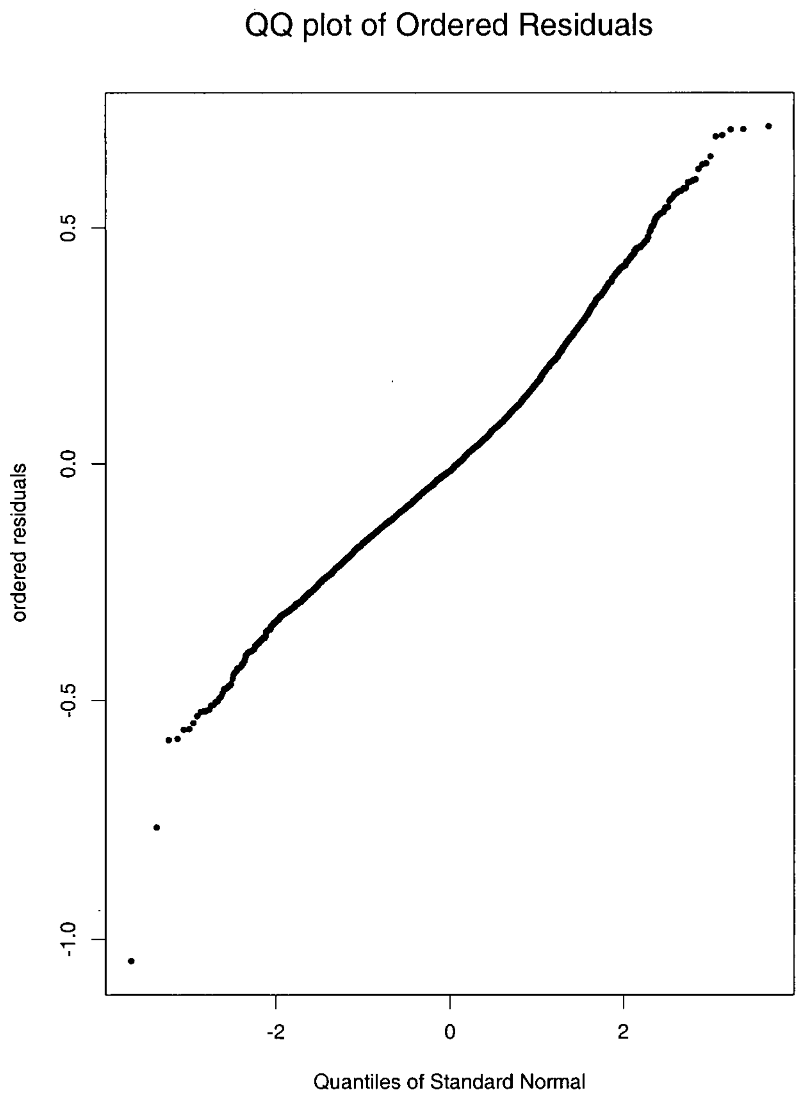


Figure 3.5: The normality assumption is tenable, apart from two outliers.

Figure 3.6 shows that in all 5 splits of the data, the extended B-WISE method produced models with better predictive ability than the B-WISE method. Likewise, B-WISE models performed better than BMARS model averages in all but one split where a BMARS model average performed better than both extended B-WISE and B-WISE. Also shown are the results for the extended B-WISE method which include the "gender" categorical predictor variable. It is unfair to compare these results to those obtained by the other methods which use one less predictor, but this example does show the usefulness of an ability to incorporate categorical variables.

We also note some differences between the BMARS and the B-WISE algorithms. While the BMARS algorithm may step from one model to another more quickly than B-WISE, the difference is usually not enough to make the overall running time shorter. For this example $20,000$ iterations of BMARS translated into a computing time of about 11 hours per split of data. In contrast, the B-WISE algorithm converged in less than an hour for every split of data.

We have found that a convenient way to represent B-WISE models is with an upper diagonal $p \times p$ matrix, where p is the number of possible predictors in the model. The diagonal elements can be either 0, 1 or 2. If the (i, i)-th element is 0 then the ith predictor is excluded from the model, if 1 then it is included as a linear effect, and if 2 then it is included as a smooth effect. The off-diagonal elements can be either 0 or 1. If the (i, j)-th element is 0 then predictors i and j do not interact, and if 1 then they do interact. For example, in the fifth split of the abalone data,

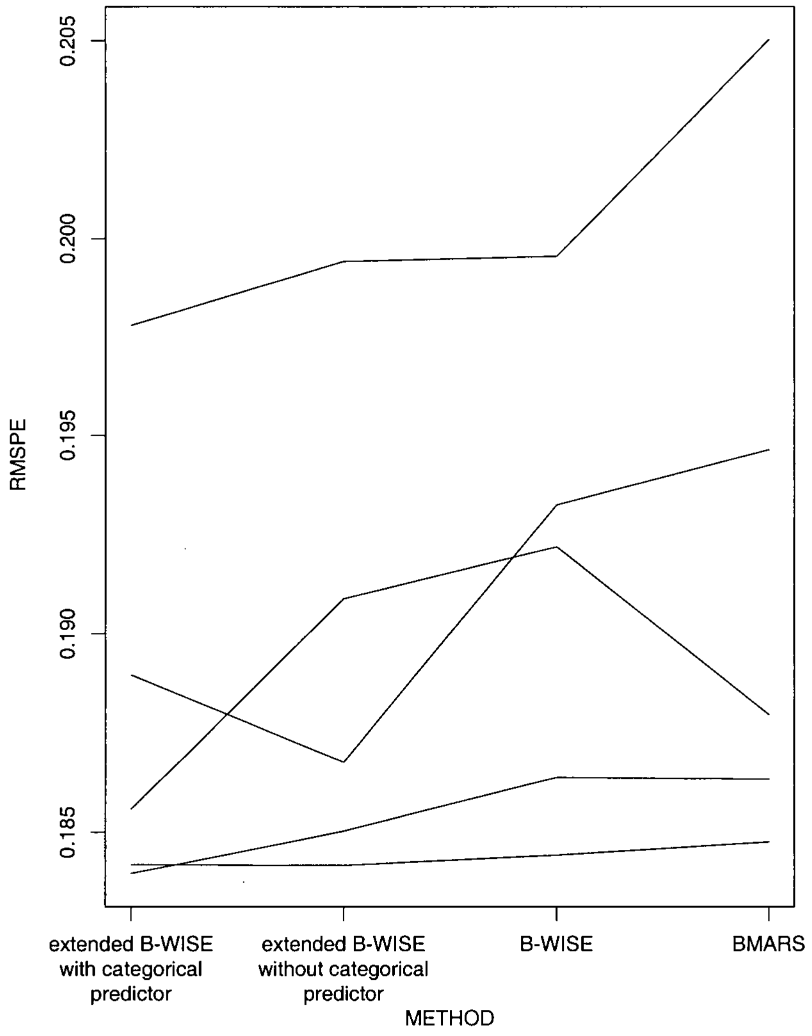


Figure 3.6: Each line represents a particular split of the data. The extended B-WISE method has a lower $RMSPE$ than both B-WISE and BMARS in four of the five splits.

the best B-WISE model found could be represented as the matrix

$$\begin{pmatrix} 2 & 1 & 0 & 1 & 0 & 0 & 0 \\ & 2 & 0 & 0 & 0 & 0 & 1 \\ & & 1 & 0 & 0 & 0 & 0 \\ & & & 2 & 1 & 0 & 0 \\ & & & & 2 & 0 & 1 \\ & & & & & 2 & 1 \\ & & & & & & 2 \end{pmatrix}. \tag{3.1}$$

Here we can immediately see that all predictors are included in the model. The interaction pairs are predictors 1 and 2, 1 and 4, 2 and 7, 4 and 5, 5 and 7, and 6 and 7. The number of predictors is high enough that there is some sophistication required, but B-WISE models are certainly more accessible than those of other adaptive regression schemes.

3.3 Census Housing Data

The data for this example were collected as part of the 1990 US census, and are made available by the Computer Science Department at the University of Toronto. Here 16 features for each of 256 geographical regions are used to predict the median price of the houses in the region. The 16 predictor variables are

1. total number of families in the region
2. total number of households (HH's)
3. percentage of black people
4. percentage of people between 25-64 years of age

5. percentage of widowed females

6. percentage of HH's with 1 person

7. percentage of HH's with asian householder

8. percentage of HH's with Hispanic householder

9. percentage of HH's in which more then one non-relative lives

10. percentage of HH's which are non-family with 2 or more people

11. percentage of housing units (HU's) occupied

12. percentage of occupied HU's which are owner-occupied

13. percentage of vacant HU's which are not for rent, sale, migrant workers nor for seasonal, recreational or occasional use

14. percentage of occupied HU's with white householder

15. percentage of occupied HU's with householder not of Hispanic origin

16. percentage of HU's with 1 to 4 rooms

Before performing the regression analysis we applied a log transformation to the response variable and fourth-root transformations to the two count variables, predictors 1 and 2, to achieve distributions which were closer to Gaussian. The remaining predictors are proportions on $[0, 1]$, and since some are highly skewed towards one end of the interval, we applied a logit transformation to all of these predictors. There were some values at the endpoints of $[0, 1]$, so before applying the logit transformation we applied a small shrinkage towards 0.5. Figures 3.7 and 3.8 show the plots of the response against predictors using the original data and the transformed data, respectively. Predictor 15 is eliminated from the analysis because during the initial fit we found that this predictor was perfectly negatively correlated with predictor 8. From the description of each predictor this detail is not entirely obvious, but it becomes obvious upon attempting to run the model search algorithm. At this point we added a safety feature that keeps the algorithm from stepping through models for which $A_\lambda^T A_\lambda$ is non-invertible. In some cases this may occur because there are more parameters than data points, and in other cases because of linear dependence or near linear dependence between continuous predictor variables. Also, with several categorical predictors in the model there are often zero counts in some category combinations. If the right interaction is present this gives rise to a non-invertible $A_\lambda^T A_\lambda$.

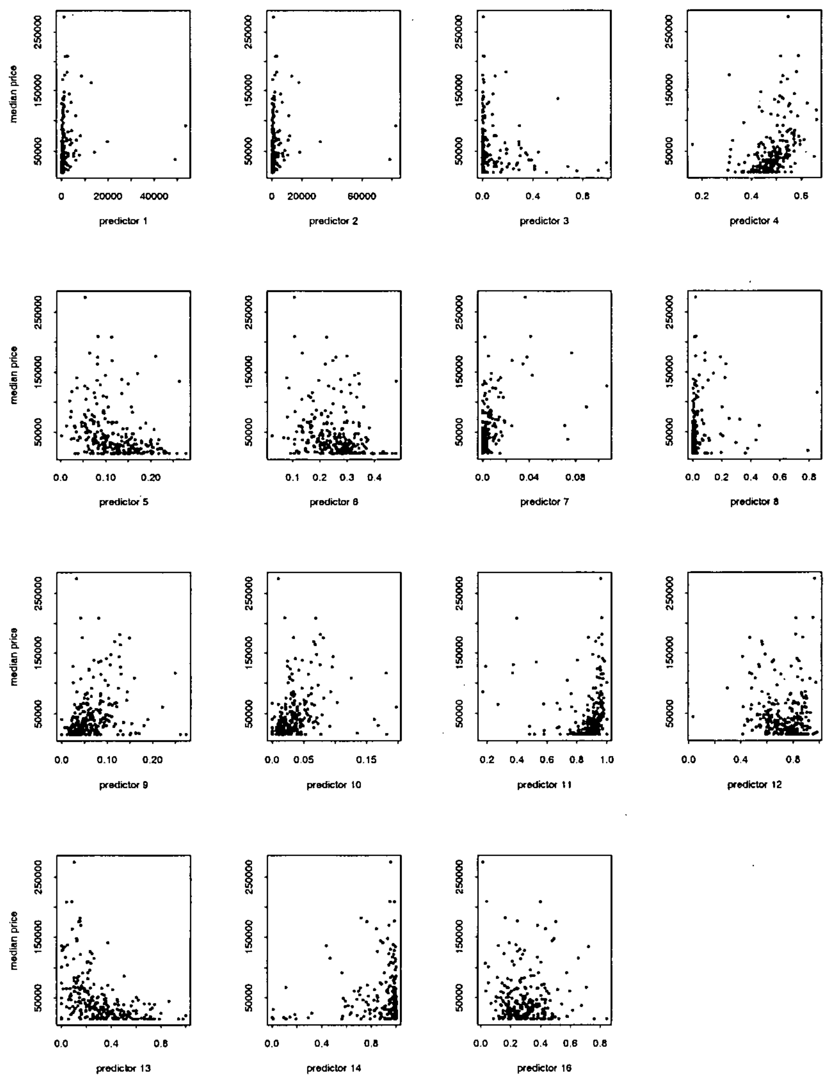


Figure 3.7: Plots of median house price against each of the untransformed predictor variables.

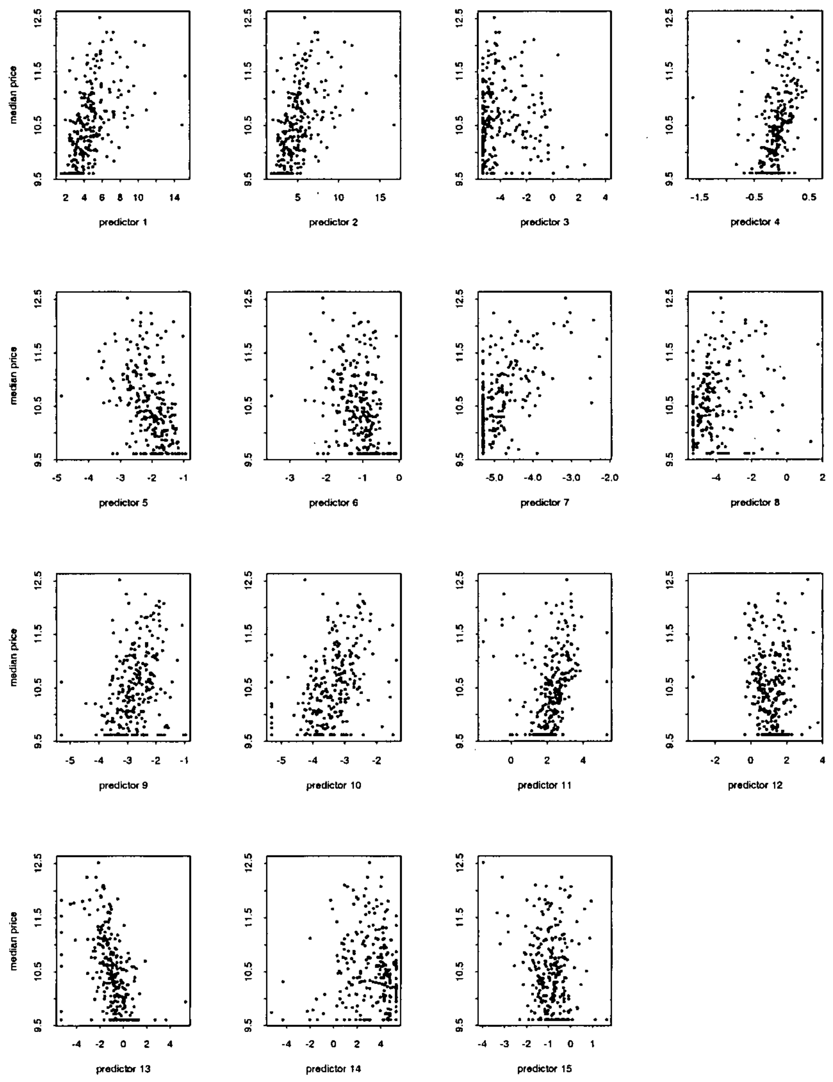


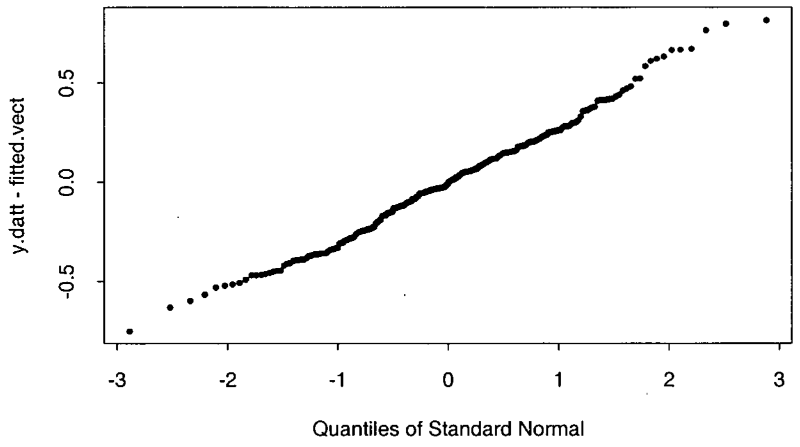
Figure 3.8: Plots of the transformed response variable against each of the transformed predictor variables. Some skewness in the predictors has been eliminated.

The best model found by the model search algorithm was

$$\left(\begin{array}{ccccccccccccccc}
2 & & & & & & & & & 1 & & & & & \\
 & 0 & & & & & & & & & & & & & \\
 & & 0 & & & & & & & & & & & & \\
 & & & 2 & & 1 & & & & & & 1 & & & \\
 & & & & 0 & & & & & & & & & & \\
 & & & & & 1 & & & & & & & & & \\
 & & & & & & 1 & & & & & & & & \\
 & & & & & & & 1 & & 1 & 1 & & & & \\
 & & & & & & & & 2 & & & & & & \\
 & & & & & & & & & 1 & & & & 1 & \\
 & & & & & & & & & & 2 & & & & \\
 & & & & & & & & & & & 1 & & & 1 \\
 & & & & & & & & & & & & 2 & & \\
 & & & & & & & & & & & & & 1 & \\
 & & & & & & & & & & & & & & 1
\end{array}\right).$$

Figure 3.9 shows that the assumption of normal errors is tenable, but the assumption of constant variance may be problematic. As it happens, about 14% of the responses take on the minimum response value of 14999, while almost all of the other response values are unique. Apparently a lower bound was imposed on the median house price when the data were recorded.

It was hoped that the model structure (3.3) would give some insight into the underlying influences of the median house price, but because of the large number of predictors this seems to be an example in which the model structure can not shed much understanding on the problem. Indeed, trying to interpret the above model



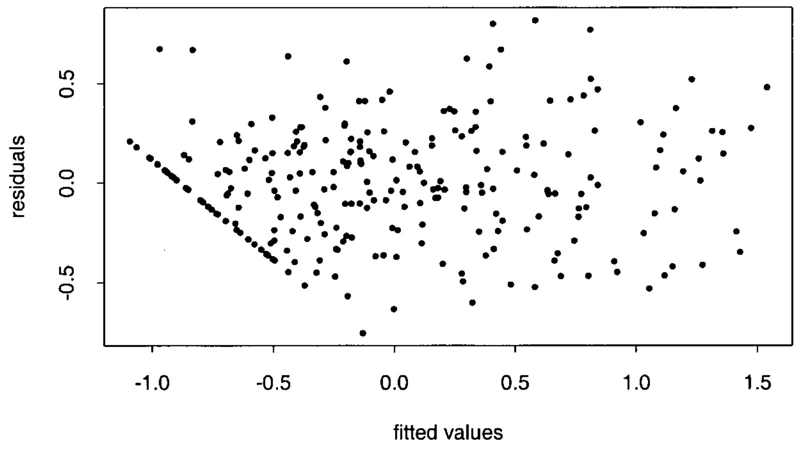


Figure 3.9: The normality assumption is reasonable, but the assumption of constant variance is problematic. Apparently a lower bound was imposed on the median house price ie. values less than 14999 were recorded as 14999, which produced the line of data points at the left of the lower plot. There does not seem to be an obvious remedy for the situation.

would be putting too much faith in the B-WISE modelling scheme. Assuming there is some true underlying model that accounts for all systematic variation in the median house price, the most we can hope is that a good approximation to this model exists within the class of B-WISE models. There is certainly some error in the structure of any B-WISE model, and for that reason it may be prudent to average over several good models. A Bayesian model average is an average of several or all models in the model space, where each model is weighted by its posterior probability. For this example we constructed Bayesian model averages using only the models in Λ_{20} as this speeds up computation and reduces round-off error. Of course, the model weights were normalized relative to Λ_{20}.

We performed five-fold cross-validation and assessed predictive ability using $RMSPE$. In all five splits the model average had a lower $RMSPE$ than the best single model though the gains were perhaps not dramatic. This could be because, as Gustafson suggested, the B-WISE models that make up the average are in some sense quite similar.

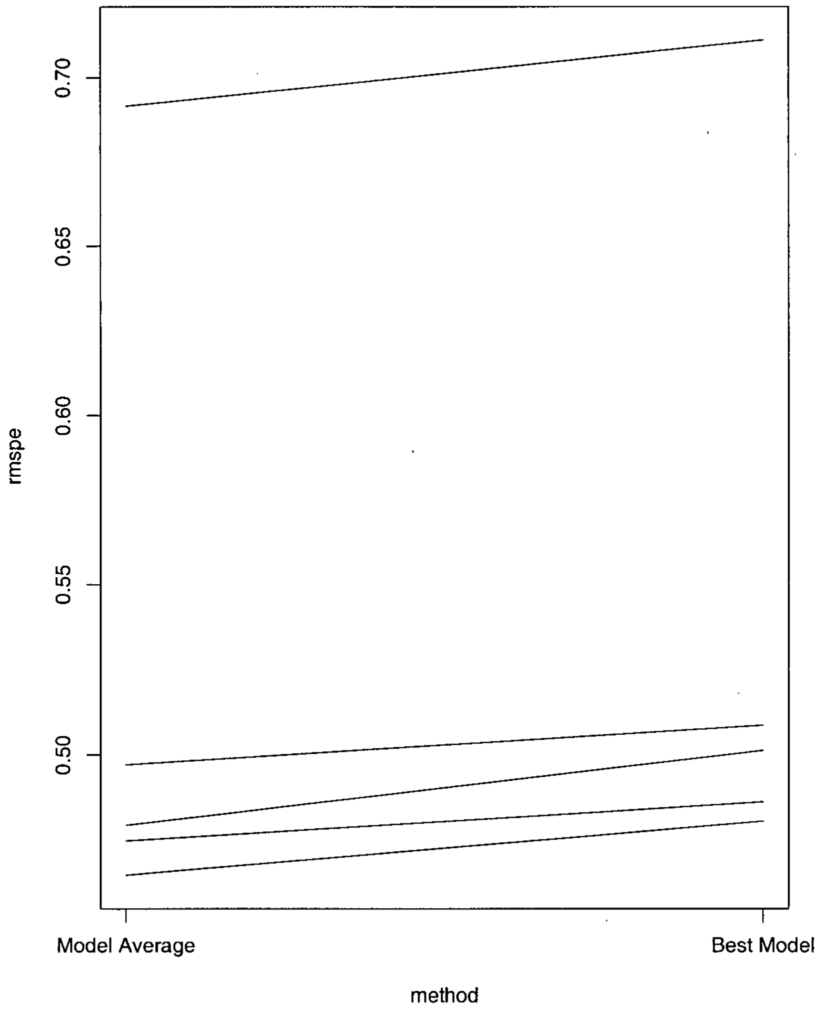


Figure 3.10: $RMSPE$ for the five splits of the census housing data.

Chapter 4

Conclusion and Further Work

B-WISE improves on many existing flexible regression schemes in the area of interpretation by limiting the number of ways a predictor can enter the model, yet it allows enough flexibility to capture non-trivial features of data and has been shown to have excellent predictive ability. The interpretability comes mainly from the easy solicitation of univariate effects, even for predictors involved in an interaction. The flexibility is provided by cubic splines for the univariate effects and a structured approach to bivariate interactions. In this thesis we have implemented two important extensions to the original B-WISE introduced by Gustafson, but there are still other directions in which B-WISE could be extended.

In our first example we hinted at some arbitrariness in the choice of the hyperparameters c and k. Gustafson provided a reasonable rationale for $c = 1$ and $k = 0.1$, but recommended some sensitivity analysis. Another option is to put a prior on these parameters. The main hurdle to using this approach would probably be choosing the prior distributions, and most likely we would lose the closed form for the model posterior probability (1.10). As a consequence, a full MCMC algorithm (i.e. an algorithm with an accept-reject step) would be required for the model search. Because of these drawbacks we favour the simplicity of fixing the hyperparameters and perhaps tuning them after a sensitivity analysis.

A vast number of data sets have a non-normally distributed response variable, in which case B-WISE cannot be used. But we think some of the ideas used in B-WISE would be useful for discrimination in the case of a binary response. As a starting point one could implement a deterministic, non-Bayesian version of the model search algorithm, which at each step moves towards the model which achieves the most separation

$$D^2 = (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)^T S_{pooled}^{-1} (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)$$

between the two groups. A Bayesian implementation would take steps towards neighbouring models in accordance with the posterior probabilities of those models, and the model posterior probability would favour models which achieve large D^2 and would penalize models according to their complexity. We think there are discrimination problems where the flexibility of B-WISE models would be useful.

Currently the rate determining step in the software we have developed is the evaluation of $A_\lambda^T A_\lambda$, which has an operation count of $O(nq^2)$, where q is the number of columns in the design matrix. This can easily be lowered to $O(nq)$ for many of the model evaluations in the search algorithm. Suppose we have calculated $A_0^T A_0$ for a certain model λ_0, and then we form a new model λ_1 by adding a term to λ_0, that is, inserting a column c into $A_0 = [A_a|A_b]$. Then the design matrix for λ_1 is $[A_a|c|A_b]$ and $A_1^T A_1$ has the form

$$A_1^T A_1 = \left(\begin{array}{c|c|c} A_a^T A_b & A_a^T c & A_a^T A_b \\ \hline c^T A_a & c^T c & c^T A_b \\ \hline A_b^T A_a & A_b^T c & A_b^T A_b \end{array} \right)$$

Most of the entries in this matrix have already been calculated. The remaining entries can be calculated in $O(nq)$ steps. Clearly, models formed by deleting columns from A_0 are even easier to compute. We suspect there are several places in the model search code where fast updating methods could be employed. Once a certain quantity has been calculated for a model, the corresponding quantity for a model in the immediate neighbourhoud might be a simple update to the information already obtained.

Bibliography

[1] Breiman, L., J. Friedman, R. Olshen and C. Stone (1984). *Classification and Regression Trees.* Belmont, Calif.: Wadsworth, Inc.

[2] Cuzick, J. (1993). Discussion of "Varying-coefficient Models" by T. J. Hastie and R. J. Tibshirani. *Journal of the Royal Statistical Society, Series B*, **55**, 757–796.

[3] Denison, D. G. T., B.K. Mallick and A. F. M. Smith (1998). Bayesian MARS. *Statistics and Computing*, **8**, 337–346.

[4] Friedman, J. (1991). Multivariate Adaptive Regression Splines (with discussion). *The Annals of Statistics*, **19**, 1–141.

[5] Friedman, J. and W. Stuetzle (1981). Projection Pursuit Regression. *Journal of the American Statistical Association*, **76**, 817–823.

[6] George E. I. and R. McCulloch (1993). Variable Selection via Gibbs Sampling. *Journal of the American Statistical Association*, **88**, 881–889.

[7] George E. I. and R. McCulloch (1997). Approaches for Bayesian Variable Selection. it Statistica Sinica, **7**, 339–373.

[8] Green, P.J. and B.W. Silverman (1994). *Nonparametric Regression and Generalized Linear Models.* London: Chapman and Hall.

[9] Gustafson, P. (2000). Bayesian Regression Modelling with Interactions and Smooth Effects. To appear in *Journal of the American Statistical Association*, **95**.

[10] Hastie, T. J. and R. J. Tibshirani (1990). *Generalized Additive Models.* London: Chapman and Hall.

[11] Hastie, T. J. and R. J. Tibshirani (1993). Varying-coefficient Models (with discussion). *Journal of the Royal Statistical Society, Series B*, **55**, 757–796.

[12] Neal, R. M. (1996). *Bayesian Learning For Neural Networks.* New York: Springer-Verlag.

[13] Neal, R. M. (1997). Monte Carlo Implementation of Gaussian Process Models for Bayesian Regression and Classification. Technical Report 9702, Department of Statistics, University of Toronto.

[14] Raftery, A. E., D. M. Madigan and J. Hoeting (1997). Model Selection and Accounting for Model Uncertainty in Linear Regression Models. *Journal of the American Statistical Association*, **92**, 179-191.

Appendix A

Details of the Model Search Algorithm

In Section 1.4 we described an algorithm with which one can locate B-WISE models of high posterior probability. In this appendix we give a pseudocode for the software we developed.

Before the model search begins, the program must read in from files all of the relevant data, including the $n \times p$ matrix X of predictor variables, the $n \times 1$ vector $\mathbf{y}$ of responses, the spline structure (Z from equation (1.4)) for each of the continuous variables, the roughness matrix (the R used in the construction of Z), and the initial model (specified using a matrix, as in (3.1)). We found it more convenient to construct Z and R using the matrix friendly routines in Splus and then to store them in files, rather than to construct them with the same C program that performs the model search. Having acquired the relevant quantities the algorithm proceeds as follows.

```
print "Model 1";
```

```
  print model;  // here model is the user-specified initial model
  print "log of non-normalized posterior probability = ";
  bestlpp=designmatrix(model,p,X,y,Z,R,n,numknots);
  print bestlpp;
```

The function `designmatrix` is the workhorse of the whole algorithm. It builds the design matrix A_λ based on the specifications in `model`, and ultimately computes and returns the model posterior probability (1.10).

```
  stopflag=1;  // the algorithm terminates if stopflag reaches 5
  numvisits=1; // this is the number of models visited
  while(stopflag<5 and numvisits<=maxit)
// maxit is the maximum number of iterations allowed
  {
    stopflag=stopflag+1;
    numnbhd=sizenbhd(model);
// sizenbhd computes the number of models in the nbhd of model
    modlist=list of all models in neighbourhood of model;
// In the loop below we will compute the model posterior
// for all of the models in modlist.
    for(lp from 1 to numnbhd)
    {
      curmodel=modlist[lp];
      lookup curmodel in modfile;
      if found then lpplist[lp]=log post prob found in modfile;
      else
      {
```

```
// need to compute log post prob for curmodel and store it in modfile
        lpplist[lp]=design_matrix(curmodel,p,X,y,Z,R,n,numknots);
        if lpplist[lp]>bestlpp then
        {
          bestlpp=lpplist[lp];
          bestmodel=curmodel;
          print "best so far = ";
          print bestlpp;
        }
        if lpplist[lp+1]>bestlpp-log(20.0) then stopflag=0;
// If a model is found which has post prob greater then 5% of the
// highest post prob then stopflag is reset to 0.
        write curmodel to modfile;
        write lpplist[lp] to modfile;
      }
    }
// Now we exponentiate and normalize the posterior probabilities
// relative to the immediate neighbourhood.
    lpplist=exp(lpplist);
    s=sum(lpplist);
    lpplist=lpplist/s;
    draw=runif(0,1);
    cumulative=0.0;
    for(i from 1 to numnbhd)
    {
      cumulative=cumulative+lpplist[i];
      if draw<=cumulative then
      {
        temp=i;
        exit for loop;
      }
    }
    print "Model ";
    numvisits=numvisits+1;
    print numvisits;
    model=modlist[temp];
    print model;
    print "log of non-normalized posterior probability = ";
    print log(s*lpplist[temp]);
  }
  print "The model with highest posterior density was ";
  print bestmodel;
```

We have also written simpler programs to compute predicted or fitted values for a single B-WISE model and for a Bayesian model average which uses the `modfile` created by the model search algorithm to construct the average.