

Recursive Partitioning for Kernel Smoothers: A Tree-based Approach for Estimating Variable Bandwidths in Local Linear Regression

December 2, 2004

An Jia and William R. Schucany

SUMMARY

Bandwidth selection is a critical issue in local linear regression. A bandwidth can be chosen to remain constant or to vary with the predictor variable. A constant bandwidth performs poorly whenever the unknown curve has complicated structure. To capture the complexity of such curves a variable bandwidth is needed. A fully variable bandwidth requires the estimation of the same number of bandwidths as the number of data points. A piecewise-constant bandwidth is recommended as a compromise. Piecewise-constant bandwidth selection involves the partition of the predictor interval and the estimation of a bandwidth in each sub-interval. Existing variable bandwidth selection methods all disregard the underlying structure and use equal-in-length partitions. In this article, we develop a tree-based approach for local linear regression. The new methodology recursively finds a partition of the predictor interval based on the structure of the unknown response mean and simultaneously estimates piecewise constant bandwidths. The recursive partitioning is based on an improved version of the well-known Akaike information criterion (AIC). A partition-adjusted version of the same criterion is used to determine the right-sized tree. Simulation results show that this flexible method compares favorably with equal-in-length variable bandwidth selection techniques.

1. INTRODUCTION

Regression is one of the most widely used statistical methods to explore the association between dependent and independent variables. More specifically, the goal of regression analysis is to estimate the conditional mean structure of a response Y for given values of a predictor X ,

$$m(x) = E(Y | X = x), \tag{1.1}$$

where little is known about the underlying regression function $m(\cdot)$. Classical regression, linear or nonlinear, requires the analyst to make a subjective choice of the global parametric form of the regression function, which in practice is almost never known.

Nonparametric regression on the other hand is a set of methods that let the data produce a suitable curve to describe the relationship, instead of being limited to a certain functional form. Fan and Gijbels (1996) call it the “data analytic approach” to emphasize its data-driven nature. One popular class of nonparametric regression methods is known as kernel smoothers. They smooth scatter plots by estimating regression functions. Many useful techniques have been proposed for univariate smoothing. See Fan (2000) for a survey for example. Different techniques have their own different merits. Chapter 2 of Fan and Gijbels (1996) gives a detailed overview of these techniques. There are several relevant chapters in Hastie, Tibshirani & Friedman (2001) as well.

The focus of this paper is on univariate local polynomial regression and in particular on local linear regression. Local polynomial regression applies regression techniques locally to a strip of data

around each regression point. Therefore it does not require knowledge of the global parametric form of the regression function.

Let us now consider the estimation of $m(x_0)$ at a prediction point x_0 , given paired observations from an assumed model

$$Y_i = m(x_i) + \varepsilon_i, \quad i = 1, \dots, n, \quad \varepsilon_i \sim \text{i.i.d. } F(0, \sigma^2), \quad (1.2)$$

where $F(\cdot)$ is some CDF with mean 0 and constant variance, σ^2 . A polynomial of degree p is fit locally by a weighted-least-squares regression criterion

$$SS = \sum_{i=1}^n \left\{ y_i - \sum_{j=0}^p \beta_j (x_i - x_0)^j \right\}^2 K(x_i - x_0, h(x_i)), \quad (1.3)$$

where K is the *kernel function* and h is the *bandwidth*. For example fitting $m(x_0)$ with a *local constant* ($p = 0$) results in the *weighted average* estimate

$$\hat{m}(x_0) = \frac{\sum_{|x_i - x_0| < h(x_i)} y_i w(x_i - x_0)}{\sum_{|x_i - x_0| < h(x_i)} w(x_i - x_0)}, \quad (1.4)$$

where $w(x_i - x_0) = K(x_i - x_0, h(x_i))$, and details of both K and h will be given in Section 2.

Figure 1.1 demonstrates the local constant fit of a sample from a simulated function, m_5 , at prediction point $x_0=2/3$, where $m_5(x) = 4.26(e^{-9.75x} - 4e^{-19.5x} + 3e^{-29.25x})$, $x \in [0, 1]$, is one of the six test functions that we use in our simulations, the details of which are in Section 4.1. We use this example to illustrate two key concepts associated with the local polynomial regression. The weights are controlled by a selected *kernel function* $K(\cdot)$, centered at x_0 as indicated by the red curve in the figure, and the size of the neighborhood is determined by a *bandwidth* h , and $2h$ is the width of the green band. Any data points that fall in the green band contribute to the weighted average for $m(x_0)$. We will examine these two concepts further.

For curve estimation a bandwidth can be chosen to remain constant or to vary with the predictor variable. The *global* bandwidth approach uses the *same* bandwidth throughout the range of x . This does not account for the structure of the relationship, e.g., for varying degrees of curvature. One hopes for a reasonable compromise to reveal enough of the mean structure without going to the extreme of interpolation. But this compromise is often not realized when the underlying regression function has a complicated structure, as in Figure 1.2. This *global* bandwidth results in *oversmoothing* for the steep curve to the left. Oversmoothing under-parameterizes this part of the regression function and causes a large modeling bias, while wiggly estimates to the right indicate *undersmoothing* for the nearly linear part. Undersmoothing over-parameterizes the unknown function and results in noisy estimates.

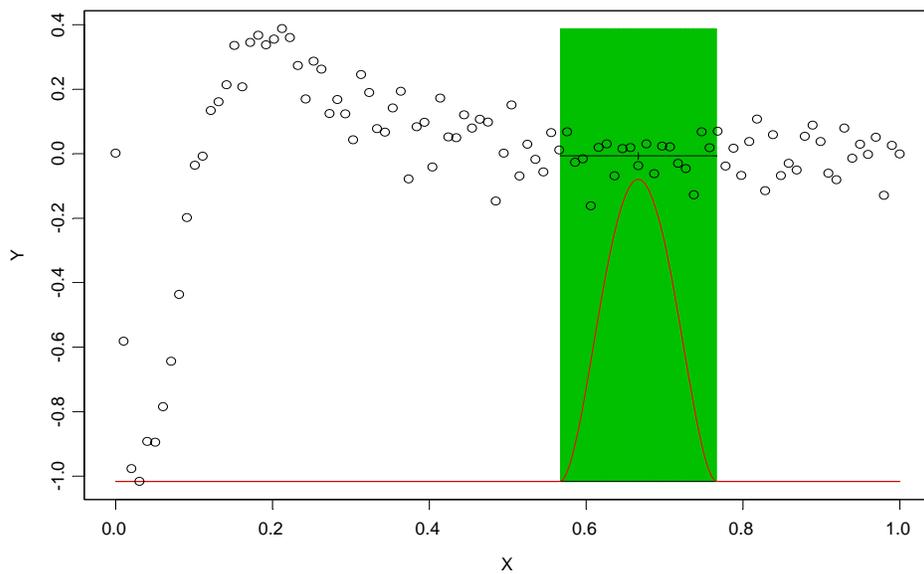


Figure 1.1. Example: Fitting a local constant – weighted average $n = 100$, $x_0 = 2/3$, bandwidth $h = 0.1$, and using biweight kernel (in red)

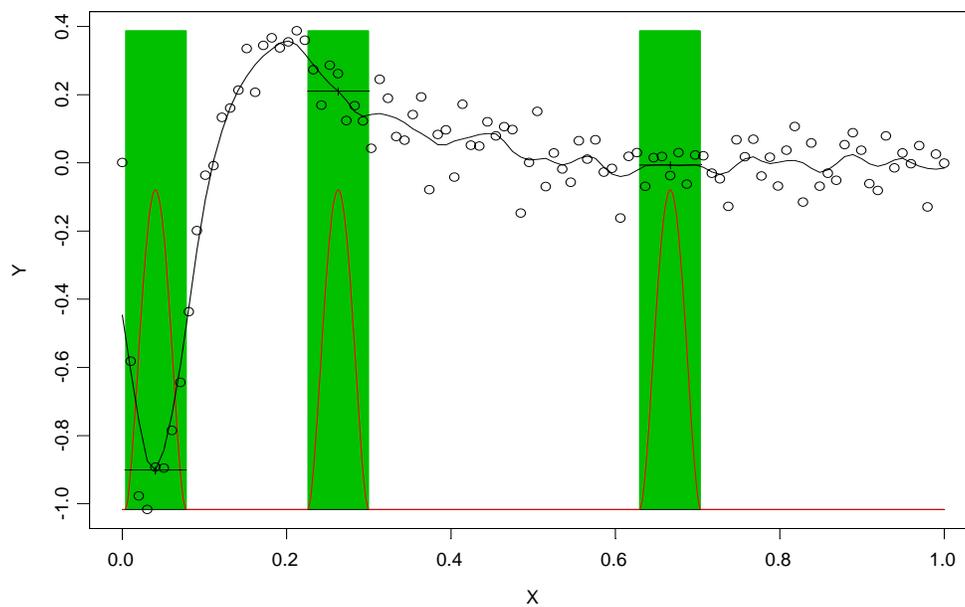


Figure 1.2. Same Example as Figure 1.1: Local constants fit with global bandwidth. The windows and fits are displayed at values of $x_0 = (0.05, 0.27, 0.67)$

A variable bandwidth approach, on the other hand, allows the bandwidth to vary as a function of each predictor's value, $x_{i..}$. The purpose is to account for changes in the structure or curvature of

the function. Hence local linear regression can handle spatially inhomogeneous curves. However, a fully variable bandwidth requires the estimation of the same number of bandwidths as the number of data points. Thus a piecewise-constant bandwidth is recommended as a compromise. Figure 1.3 shows the results of local linear regression of the same function with piecewise constant bandwidths. The oversmoothing in the steep curve to the left is resolved by using smaller bandwidths and the undersmoothing of the linear trend to the right is resolved by using larger bandwidths.

The notation used in Figure 1.3 deserves some explanation since it will be used throughout the rest of the paper. The orange blocks at the base of the chart show the piecewise constant bandwidths corresponding to the locations of the predictor points. The heights of the blocks above the x -axis represent the relative magnitudes of the bandwidths used in the local linear regression. The true mean function $m(x)$ is plotted as the green line and the estimated regression line is in red.

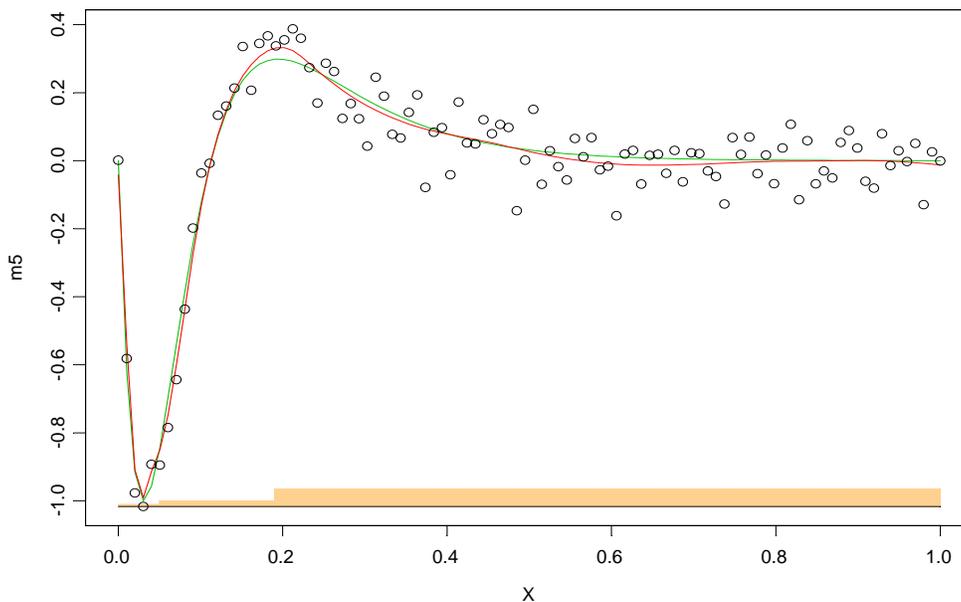


Figure 1.3. Example 1: Local Linear Regression of m_5 with Variable Bandwidths

Piecewise-constant bandwidth selection involves partitioning the predictor interval and estimating bandwidths in each sub-interval of the partition. Existing variable bandwidth selection methods all disregard the underlying structure and use equal-in-length partitions. See, for example, Fan & Gijbels (1996) and Pitblado (2000).

In this article, we develop a tree-based approach for local linear regression. The new methodology recursively partitions the predictor interval based on the structure of the data and simultaneously estimates the piecewise constant bandwidths. Consider the recursive partitioning process as growing a binary tree. Each tree node corresponds to a subinterval of the predictor variable. Starting with a global bandwidth in the predictor interval (the root), the procedure first splits the predictor interval into two children nodes. Instead of splitting the interval equally in length, the position of the split and the bandwidths in the two children intervals is selected to minimize an improved version of the Akaike information criterion (AIC_C), which is defined in Section 2.2. The procedure then continues to grow the tree by recursively partitioning the two children intervals, and

then their children, and so forth. When the tree is fully grown based on the stopping rule, a partition-adjusted version of the AIC_C criterion is used as the cost-complexity measure to prune or recombine some of the tree nodes to produce the right-sized tree. This new regression estimator may be denoted by $RPVB$, for Recursively Partitioned Variable Bandwidths.

Simulation results show that this flexible method compares favorably to other fixed equal-in-length variable bandwidth selection techniques. Comparison of the performance of this method and the equal-in-length partition-based method introduced in Fan and Gijbels (1995) use six simulation functions. A detailed description of $RPVB$ is in Section 3. The three main advantages of our proposed method are summarized below.

Recursive partitioning based on the AIC_C criterion provides a data-driven algorithm to automatically decide the number of partitions, the location of the splits in the partitions, as well as the estimation of piecewise-constant bandwidths based on the structure of the underlying curve. It consistently results in fewer partitions than the equal-length methods while achieving similar MSE. For example $RPVB$ chooses a global bandwidth for function m_1 , which is a quadratic function with constant second derivative (curvature does not change). This is quite consistent even when we introduce large additive error variance.

Consequently, our method successfully captures some fine structure that some other variable bandwidth methods miss. Simulation function m_3 serves as a good example, $m_3 = 5\{1 + e^{(3.2-6.4x)}\}^{-1} + \sin\{30\pi(x-1/3)\}\delta_{[1/3,2/3]}(x)$, $x \in [0, 1]$, another of the six test functions. Our method successfully captures the sine wave in the middle, which is completely smoothed out by the Fully Variable Bandwidths (FVB) method of Fan & Gijbels (1995) (F&G). We will discuss these simulation results in more detail in Section 4. See for example Figure 4.2.

Another advantage of our method is that it leads to smoother estimated curves. This is because our partition and bandwidths estimation are based on a bias corrected version of AIC that imposes a “roughness” penalty, while F&G is based solely on a residual squares criterion (RSC).

Also, this method can be used directly for local polynomial fitting of any order and can be easily modified to estimate the derivatives. It can also be extended to multiple covariate situations in the same way that the regression tree is used for handling multivariate cases.

In Section 2 we discuss the bandwidth selection problem and the AIC_C criterion. In Section 3 we discuss the details of the recursive partition method and implementation. The simulation experimental design and results will be presented in Section 4. Conclusions and discussions are in Section 5.

2. VARIABLE BANDWIDTH SELECTION USING AIC_C IN LOCAL LINEAR REGRESSION

In this section we take a closer look at local linear regression ($p = 1$). The local constant fit was discussed in Section 1 just as a simple illustration of local polynomial regression. Fan & Gijbels (1996) argue that local linear regression ($p = 1$) is generally superior to a local constant fit. Local linear regression solves for β_0 and β_1 to minimize

$$SS = \sum_i \{y_i - \beta_0(x_0) - \beta_1(x_0)(x_i - x_0)\}^2 K(x_i - x_0, h(x_i)). \quad (2.1)$$

The estimated regression curve at a point x in the neighborhood of x_0 is

$$\hat{m}(x) = \hat{\beta}_0(x_0) + \hat{\beta}_1(x_0)(x - x_0). \quad (2.2)$$

In the previous section we briefly introduced two key concepts, *kernel functions* and *bandwidths*. A kernel function is a unimodal, symmetric around 0, nonnegative function denoted by

$$K(t, h) = h^{-1} K_0(h^{-1} t), \quad (2.3)$$

where K_0 integrates to 1 and $h > 0$. For example the *biweight* kernel is given by

$$K_0(u) = \begin{cases} 0.9375(1-u^2)^2 & \text{if } |u| < 1, \\ 0 & \text{otherwise.} \end{cases} \quad (2.4)$$

A kernel function does not have to have compact support, as long as it decays to eliminate the influence of remote x values. There are many choices of kernel functions. Some commonly used kernel functions include *biweight*, *Epanechnikov*, *cosine*, *Laplace*, *normal*, *triangular*, and *uniform*. It has been shown that (see for example Theorem 3.4 in Fan & Gijbels (1996)), the *Epanechnikov* kernel

$$K_0(u) = \begin{cases} 0.75(1-u^2) & \text{if } |u| < 1 \\ 0 & \text{otherwise} \end{cases} \quad (2.5)$$

is the optimal kernel in the sense that it minimizes the asymptotic MSE and MISE over all nonnegative, symmetric, and Lipschitz continuous functions when theoretically optimal bandwidths are used. The *biweight* kernel in (2.4) resolves the non-differentiability of the *Epanechnikov* kernel at $u = \pm 1$ with little sacrifice to the optimality of the latter. Therefore we use the *biweight* kernel in all our simulations. In practice, the kernel function in local polynomial regression works by replacing t by $(x_i - x_0)$, so that the weight of a data point x_i is determined by its distance from the prediction point, x_0 .

Most nonparametric techniques involve selection of smoothing parameters. Bandwidth, commonly denoted by h , controls the size of the local neighborhood of the prediction point. The selection of bandwidth is a bias and variance tradeoff. Smaller h yields smaller bias, larger variance of estimates, and less *smoothing*; while larger h is associated with larger bias, smaller variance in estimates, and more smoothing. For detailed derivations and discussion, please refer to Fan & Gijbels (1996). Another issue in local polynomial fitting is the choice of the order of the local polynomial. Since the modeling bias is primarily controlled by the bandwidth, this issue is less crucial and is not considered further in this paper.

The choice of bandwidth determines the complexity of the model. A bandwidth $h = 0$ results in interpolating the data and hence leads to the most complex model. A bandwidth approaching $+\infty$ corresponds to fitting classical linear regression, the simplest model.

There are many published methods for global bandwidth selection. They are often referred in two categories, namely classical methods and plug-in methods. Classical methods are based primarily on information criteria and cross validation, while plug-in techniques involve the estimation

and substitution of the unknown quantities in the theoretical expression that minimizes Asymptotic Mean Integrated Squared Error (AMISE). See Loader (1999) for the comparisons of these two methods. Ideal theoretical choices of this optimal bandwidth are easy to obtain as shown, for example, in Fan & Gijbels (1996). However this theoretical choice is not directly usable in practice because it depends on unknown quantities. Finding a practical procedure for selecting the bandwidth parameters is one of the most challenging tasks. This is the main focus of this paper. In the next section we address this issue in some detail.

2.1 Variable Bandwidth Selection

A variable bandwidth is introduced to allow for different degrees of smoothing, resulting in a possible reduction of the bias in peaked regions and of the variance in flat regions. This enhances the flexibility of the local polynomial fitting so that it can adapt to spatially inhomogeneous curves.

A fully variable bandwidth requires the estimation of as many smoothing parameters as data points, which increases the model complexity. Such estimates of the optimal bandwidth function are not very stable and may require smoothing themselves. This extra step may reduce their adaptability to curvature. Piecewise-constant bandwidths have proven to be a reasonable compromise. Adams (1998) showed that piecewise constant bandwidths can capture the structure of reasonably complex mean functions with far fewer parameters than the fully variable bandwidths.

For a variable bandwidth the distinction should be made between a *local variable bandwidth* versus a *global variable bandwidth*. A local bandwidth $h(x_0)$ varies with the estimation point, whereas a global variable bandwidth changes with the data points, $h(x_i)$ as indicated in (2.1). The distinction in the case of density estimation is well established by Jones (1990). The theoretical choice of a global variable bandwidth is given in Fan and Gijbels (1992) and that for local variable bandwidth in Fan and Gijbels (1996). Adams (1998) pointed out that the piecewise-constant local variable bandwidth results in discontinuous regression estimates. This is a major flaw, because $m(x)$ is assumed to be continuous. On the other hand the piecewise-constant global variable bandwidth does not have this problem. Therefore it is preferred and will be used in our method. Subsequently, when we use the term “variable bandwidth”, we mean piecewise-constant global variable bandwidth.

Piecewise-constant bandwidth selection involves the partition of the predictor interval and the estimation of a bandwidth in each sub-interval of the partition. Suppose the interval of observed predictor, $x \in I$, is partitioned into J disjoint subintervals denoted by $I_j, j = 1, \dots, J$, i.e., $I = \bigcup_{j=1}^J I_j$ and $I_j \cap I_k = \emptyset, j \neq k, j, k = 1, \dots, J$. J is called the partition number, as described in Pitblado (2000). Given a fixed bandwidth h_j within each subinterval I_j , the piecewise constant bandwidth function can then be represented as

$$h(x) = \sum_{j=1}^J h_j \delta_{I_j}(x), \quad (2.6)$$

where δ_I is the indicator function, which = 1 if $x \in I$ and 0 otherwise.

When piecewise constant bandwidths are used, the local linear ($p=1$) regression estimate of $m(x_0)$, the mean function at a given prediction point x_0 , involves calculating the intercept β_0 and slope β_1 , which minimize a weighted sum of squared deviations, such as

$$SS = \sum_{i=1}^n \{y_i - \beta_0 - \beta_1(x_i - x_0)\}^2 K\{x_i - x_0, h_p(x_i)\}. \quad (2.7)$$

It is convenient to use matrix and vector notation to show the calculations in solving for the intercept and slope in (2.7). They are given in Pitblado (2000). We include them here for completeness. Let $\mathbf{y}^t = (y_1, \dots, y_n)$ and $\mathbf{x}^t = (x_1, \dots, x_n)$ be the vectors of the observed responses and corresponding predictors, where the superscript t denotes matrix transpose. The model in (1.5) can be rewritten as

$$\mathbf{y} = \mathbf{m} + \boldsymbol{\varepsilon}, \quad (2.8)$$

where $\mathbf{m}^t = \{m(x_1), \dots, m(x_n)\}$, $E(\boldsymbol{\varepsilon}) = 0$, $\text{Var}(\boldsymbol{\varepsilon}) = \sigma^2 \mathbf{I}_n$, and \mathbf{I}_n denotes the n -dimensional identity matrix. The design matrix for local linear estimation of $m(x_0)$ is

$$\mathbf{X}(x_0) = \begin{bmatrix} 1 & x_1 - x_0 \\ \vdots & \vdots \\ 1 & x_n - x_0 \end{bmatrix} = [\mathbf{1}_n \quad \mathbf{x} - x_0 \mathbf{1}_n], \quad (2.9)$$

where $\mathbf{1}_n$ denotes an n -dimensional vector of ones. The weight matrix is denoted by

$$\mathbf{W}(x_0, h_p) = \text{diag}\{K(x_1 - x_0, h_p(x_1)), \dots, K(x_n - x_0, h_p(x_n))\}. \quad (2.10)$$

The weighted sum of squares to minimize can be written

$$SS = \{\mathbf{y} - \mathbf{X}(x_0) \boldsymbol{\beta}\}^t \mathbf{W}(x_0, h_p) \{\mathbf{y} - \mathbf{X}(x_0) \boldsymbol{\beta}\}, \quad (2.11)$$

where $\boldsymbol{\beta}^t = (\beta_0, \beta_1)$. The weighted-least-squares estimates of the local intercept and slope, denoted by $\hat{\boldsymbol{\beta}}(x_0, h_p)$, are

$$\hat{\boldsymbol{\beta}}(x_0, h_p) = \{\mathbf{X}(x_0)^t \mathbf{W}(x_0, h_p) \mathbf{X}(x_0)\}^{-1} \mathbf{X}(x_0)^t \mathbf{W}(x_0, h_p) \mathbf{y}. \quad (2.12)$$

Hence the estimate of $m(x_0)$ is

$$\hat{m}(x_0, h_p) = u_1^t \hat{\boldsymbol{\beta}}(x_0, h_p) = u_1^t \{\mathbf{X}(x_0)^t \mathbf{W}(x_0, h_p) \mathbf{X}(x_0)\}^{-1} \mathbf{X}(x_0)^t \mathbf{W}(x_0, h_p) \mathbf{y}, \quad (2.13)$$

where u_i is a unit vector along the i^{th} coordinate axis. Estimating \mathbf{m} , the vector of regression functions at each of the observed predictors, and denoting it by $\hat{\mathbf{m}}(h_p)$, we have

$$\begin{aligned}
\hat{\mathbf{m}}(h_p) &= \{\hat{m}(x_1, h_p) \cdots \hat{m}(x_n, h_p)\}' = \begin{bmatrix} u_1' \hat{\beta}(x_1, h_p) \\ \vdots \\ u_1' \hat{\beta}(x_n, h_p) \end{bmatrix} \\
&= \begin{bmatrix} u_1' \{X(x_1)' W(x_1, h_p) X(x_1)\}^{-1} X(x_1)' W(x_1, h_p) y \\ \vdots \\ u_1' \{X(x_n)' W(x_n, h_p) X(x_n)\}^{-1} X(x_n)' W(x_n, h_p) y \end{bmatrix} \\
&= \mathbf{H}(h_p) \mathbf{y}, \tag{2.14}
\end{aligned}$$

where $\mathbf{H}(h_p)$ is a *smoother* (or *hat*) *matrix*. This matrix will be used in Section 2.2 for the definition of the information criterion AIC_C .

Several data-driven methods have been developed. Cross-validation (Allen 1974; Stone 1974; Rudemo 1982) and generalized cross-validation (Wahba, 1977) are generally applicable methods. Yet, their resulting bandwidths can vary substantially (Hall and Johnstone, 1992). Plug-in methods tend to be more stable. In addition to the methods surveyed in Jones, Marron and Sheather (1996), the pre-asymptotic substitution method by Fan and Gijbels (1995) and the empirical-bias method by Ruppert (1997) provide useful alternatives. See also Marron and Padgett (1987).

We focus on Fan and Gijbels (1995), who proposed a data-driven bandwidth selection procedure for both constant and variable bandwidths with a fixed equal-in-length partition of the predictor interval. Their idea is based on a residual squares criterion (RSC) along with good approximations for the bias and variance of the estimators. Through a wealth of test examples, they successfully demonstrate that local polynomial regression with their data-driven variable bandwidths has spatial adaptation properties that are similar to wavelets. Therefore, we use their method for comparison and show that our method outperforms theirs. Their method to be denoted by *FVB* stands for Fully Variable Bandwidths.

More recently, Pitblado (2000) proposed a data-driven methodology for choosing variable bandwidths that uses an adjusted form of the Akaike Information Criterion (AIC_C). This criterion, described in Section 2.2, yields promising estimates of piecewise constant bandwidths for fixed equal-in-length partitions. It may also be able to choose a parsimonious partition number.

However, both Fan & Gijbels (1995) and Pitblado (2000) are limited to equal-in-length partitions, thereby disregarding the underlying structure. It is reasonable to argue that the bandwidth should remain constant unless the curvature changes noticeably. This not only will lead to more consistent estimates in the area with homogeneous curvature, but also will result in fewer smoothing parameters and hence further reduces the model complexity. A data-based technique to automatically find such a partition in accordance with the underlying structure given the random errors in the observed data is challenging but highly desirable. We will show that a tree-based recursive partitioning method using adjusted forms of the AIC criterion satisfactorily resolves this problem. We first discuss the criterion in the next section and then how recursive partitioning is used to address this issue.

2.2 Improved Akaike Information Criterion (AIC_C)

As we mentioned, some of the classical bandwidth selection methods are based on information criteria. The criterion that we are going to use is a bias-corrected version of the *Akaike Information*

Criterion (AIC). The *AIC* was originally designed for parametric models as an approximately unbiased estimate of the expected Kullback-Leibler information. When adopted for linear regression and time series models, Hurvich and Tsai (1989) showed that the bias of *AIC* could be quite large in small samples. It leads to over-fitting, especially as the dimension of the candidate model approaches the sample size. They proposed a corrected version, termed AIC_C , which was found to be less biased than *AIC*. Hurvich, Simonoff, and Tsai (1998) investigated the use of AIC_C to choose smoothing parameters. They showed that the use of AIC_C avoids the large variability and the tendency to undersmooth (compared to the actual minimizer of average squared error,) that are typical for other classical approaches such as generalized cross-validation or *AIC*. For a more detailed derivation and discussion of this criterion, please refer to these papers and Pitblado (2000).

In local linear regression, we estimate $\mathbf{m}^t = (m(x_1), \dots, m(x_n))$, the regression function at each of the observed predictors $\mathbf{x}^t = (x_1, \dots, x_n)$, with weighted least squares. It has been shown, see for example Hurvich, Simonoff and Tsai (1998), that local linear regression is a linear smoother, which means that $\hat{m}(h_p) = H(h_p)y$, where $\mathbf{y}^t = (y_1, \dots, y_n)$ is the response vector and $H(h_p)$ is the *smoother* (or *hat*) *matrix* defined in (2.14). For simplicity, we denote $H(h_p)$ by H . With the above notation

$$AIC_C = \log(\hat{\sigma}^2) + \psi_n\{tr(H)\}, \quad (2.15)$$

where $\hat{\sigma}^2$ is the mean squared error (MSE) of the residuals at the observed predictors and

$$\psi_n(t) = \begin{cases} \frac{1+t/n}{1-(t+2)/n} & 0 < t < n-2, \text{ or} \\ \infty & \text{otherwise,} \end{cases} \quad (2.16)$$

is the penalty function applied to the trace of the smoother matrix. The estimate $tr(H)$ can be interpreted as an effective number of parameters used in the smoothing, a measure of model complexity. It reflects the “roughness” of the estimated curve. Since the regression function is assumed to be smooth, there is a penalty to be imposed on the “roughness”. As the global bandwidth decreases, $\hat{\sigma}^2$ decreases and we have less bias in the estimates, whereas the $tr(H)$ and $\psi_n\{tr(H)\}$ increase as the estimated curve becomes less smooth. Pitblado (2000) demonstrated this behavior of AIC_C with respect to the choice of the global bandwidth. This is also the case for a variable bandwidth.

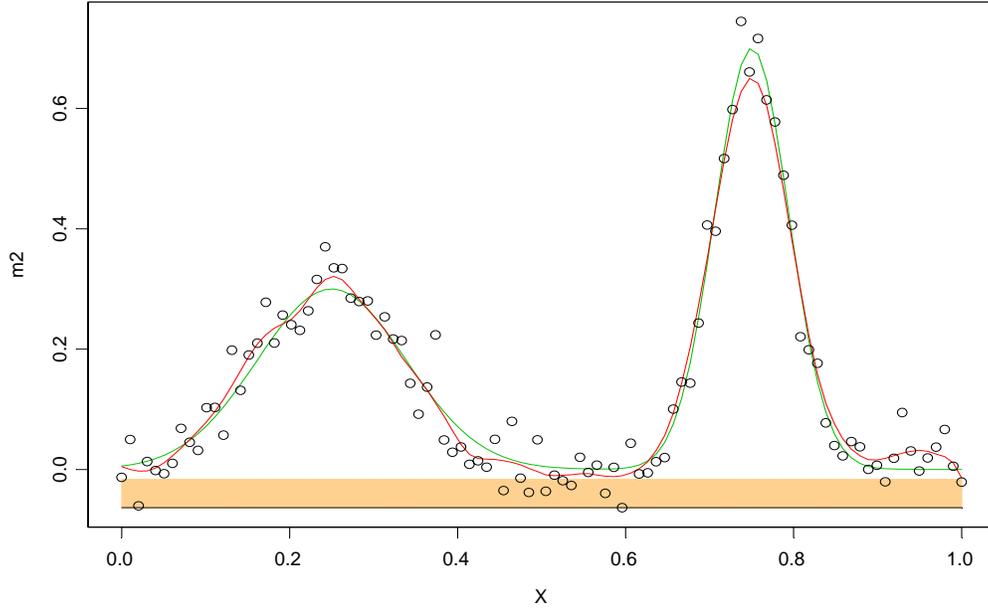


Figure 2.1. The true regression function of m_2 and the local linear fit with global bandwidth

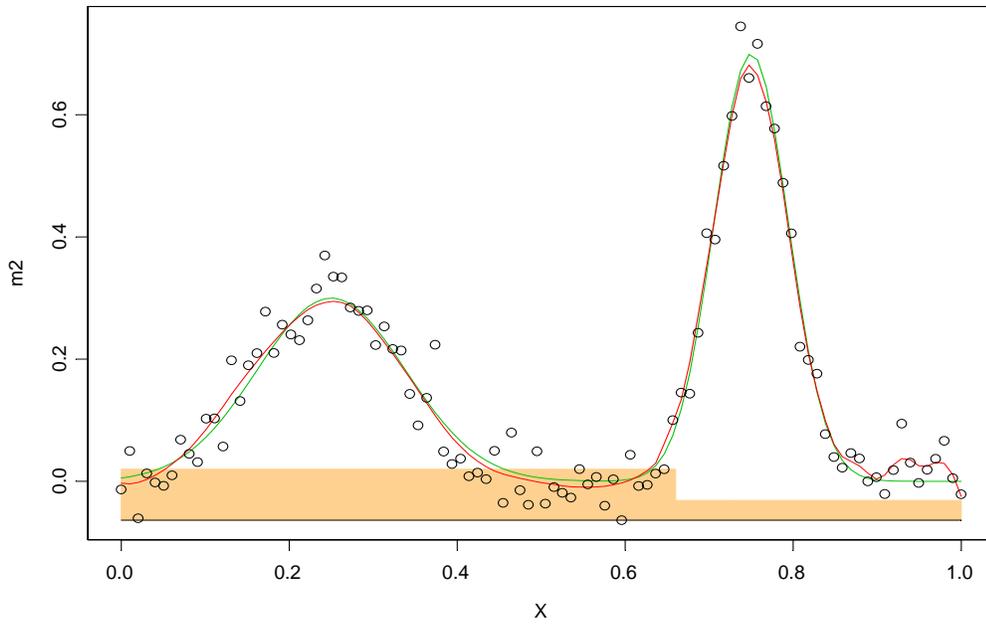


Figure 2.2. The true regression function of m_2 and the local linear fit with variable bandwidths

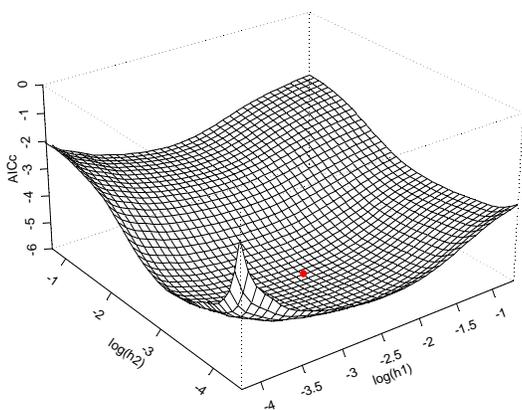
How do variable bandwidths, in particular piecewise constant bandwidths, help us in this respect? Consider estimates using $n=100$ equally spaced evaluations (plus noise) of the function m_2 , which is another of the six test functions that we use in our simulations. Function

$m_2 = 0.3e^{-64(x-0.25)^2} + 0.7e^{-256(x-0.75)^2}$, $x \in [0, 1]$, is a linear combination of two normal density functions with different curvature. Figure 2.1 shows the true regression function and the local linear fit with a global bandwidth. The estimated curve with a global bandwidth exhibits undersmoothing of the left mode and oversmoothing of the right mode. Improvement is obvious in the estimated curve with variable bandwidths in Figure 2.2.

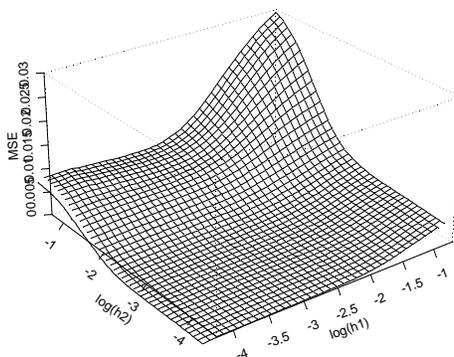
Our new method found a partition with a split at $x = 0.66$ and variable bandwidths $h_p = (h_1, h_2)$, where $h_1 = 0.074$ is for the left subinterval and $h_2 = 0.038$ is for the right subinterval. Figure 2.3 (i) shows AIC_C as a function of $\log(h_1)$ and $\log(h_2)$. Apparently the criterion favors a relatively larger h_1 and smaller h_2 , which reflects the higher degree of curvature to the right. Notice that AIC_C reaches its minimum at $(0.074, 0.038)$, or $(-2.6, -3.27)$ on the log scale, as marked by the red dot. The trend is not seen in the top right panel, Figure 2.3 (ii), plot of MSE against the variable bandwidths.

Recall that there are two components in AIC_C : $\log(\hat{\sigma}^2)$ is the penalty for bias and $\psi_n\{tr(H)\}$ is the penalty for “roughness”. The bias term decreases when bandwidths get smaller and reaches its minimum when both bandwidths equal zero, in which case local linear regression becomes interpolation. This is obvious in Figure 2.3 (ii). On the other hand, $tr(H)$ and $\psi_n\{tr(H)\}$ are plotted in Figure 2.3 (iii) and (iv), respectively. Obviously, they both decrease as bandwidths get larger, where the estimated curve becomes smoother.

When bandwidths are chosen to minimize AIC_C , it allows one to strike a balance between the bias and the smoothness of the regression estimates. Therefore, it is used as the information criterion in our search for variable bandwidths. For a given partition, we select the variable bandwidths that minimize the AIC_C . Moreover, we will show how recursive partitioning is used to resolve the challenging problem of finding the best partition tree for the variable bandwidth. Here AIC_C is used as the cost function and a partition size adjusted version of AIC_C is used for minimum cost-complexity pruning. This will be the topic of Section 3.



(i)



(ii)

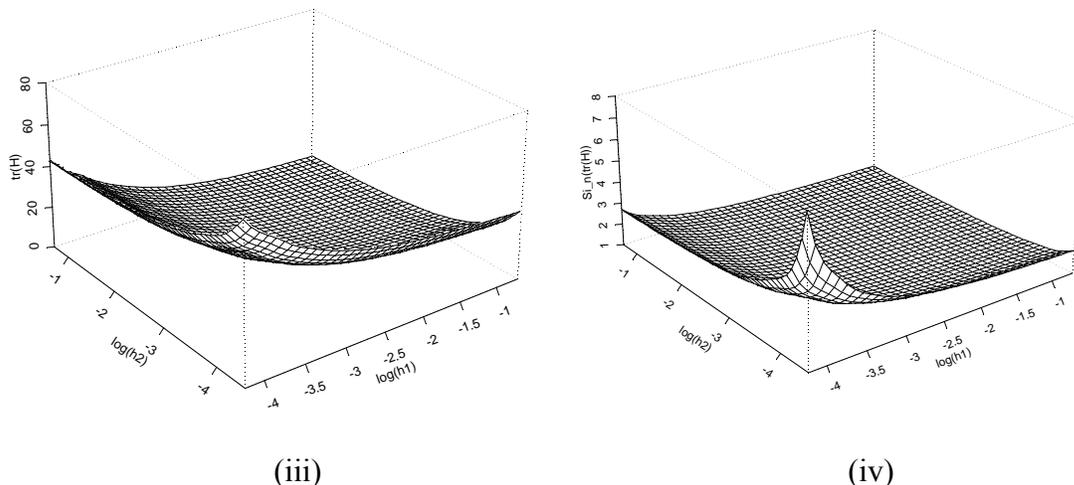


Figure 2.3. (i) AIC_C , (ii) MSE, (iii) $tr(H)$, and (iv) $\psi_n\{tr(H)\}$ as functions of the variable bandwidths

3. RECURSIVE PARTITIONING FOR VARIABLE BANDWIDTHS

A tree-based approach is proposed here to find an appropriate partition for a piecewise-constant bandwidth. The predictor interval is recursively partitioned into subintervals using binary splits. The resulting partition can be represented as a binary tree. Each node of the tree represents a subinterval with a bandwidth to be used for the x values in that subinterval. The bandwidths are estimated concurrently in the recursive partitioning process. This procedure is not limited to equal-length partitions as in Fan and Gijbels (1995) and Pitblado (2000). Furthermore it imposes a penalty for too many partitions.

Breiman et al (1984) contains a detailed introduction to tree-based methodologies for classification and regression. The same approach is used to estimate a piecewise-constant bandwidth, where the independent variable is recursively partitioned into a tree of subintervals and a different constant bandwidth is used in each subinterval. We grow a large tree first and then selectively prune (recombine) upward. The reason is that some intermediate nodes themselves may fail to minimize AIC_C , but they provide a basis for important subsequent splits that do reduce AIC_C greatly because of adaptation to curvature change.

Our goodness-of-split criterion is AIC_C . In Figure 1.3 why should the interval $[0, 1]$ be split at $x = 0.19$ rather than anywhere else? Similar questions arise for subsequent splits. This is done by exhausting every possible split at each of the observed x values. For example, given (x_i, y_i) , $i = 1, \dots, n$. Suppose that 5 is the minimum node size, i.e., each sub interval must have at least five observations. The recursive algorithm must then evaluate the values of the goodness-of-split criterion at $x_6, x_7, x_8, \dots, x_{n-5}$, and select the position where the goodness-of-split criterion is maximized.

In our method the partitioning process stops when AIC_C cannot be improved. There are a couple of differences between AIC_C and the common risk measures like *Gini* and SSE. One is that AIC_C is not additive. SSE is additive because the total SSE in an interval is the sum of the SSE's in its subintervals, i.e.,

$$\Delta SSE = SSE(t) - SSE(T_t) = SSE(t) - SSE_{\text{left}}(T_t) - SSE_{\text{right}}(T_t). \quad (3.1)$$

This is also the case for Gini. In both cases the change in risk, ΔR , is always positive. But this is not the case for ΔAIC_C , which could be negative. This is because that AIC_C is not only a function of SSE, but also the penalty for “roughness”. This results in ΔR possibly being negative, which means that a split fails to improve AIC_C . So further partitioning of this node would not appear to be helpful.

Another difference is that the choice of a bandwidth in a subinterval may contribute to the SSE throughout the entire data range. This is because we are using a global variable bandwidth to eliminate the discontinuity of local variable bandwidth (recall Subsection 2.1). Therefore, we need to impose a restriction on the magnitude of any bandwidth to control the influence of the bandwidth in distant subintervals. We address this issue in Section 4 in discussing the implementation of this new *RP* *VB*.

In the *minimum cost-complexity* pruning used in *CART*, the *cost-complexity* is defined as

$$R_\alpha = R(T) + \alpha \|T\|,$$

where $R(T)$ is the risk of a tree or a branch of a tree T , $\|T\|$ is the size of the branch, i.e., the number of descendent nodes (or equivalently the number of splits), and α is the penalty imposed for each tree node. *CART* contains a procedure to seek the *smallest subtree* that *minimizes* R_α by recursively cutting the *weakest links*. For a detailed discussion of the procedure see Chapter 3 of Breiman et al (1984).

AIC_C is a compact *cost-complexity* measure in its own right. Recall (2.15)

where a penalty is imposed on the roughness of the local linear regression based on the variable bandwidths selected. Our new stopping rule requires a split to improve AIC_C at least $100C_0$ percent, i.e.,

$$C_0 > \{AIC_{C(t)} - AIC_{C(T)}\} / AIC_{C(t)}, \quad (3.2)$$

where $AIC_{C(t)}$ is the AIC_C value calculated after a split is introduced. Preliminary simulations show that the cutoff value of $C_0 = 0.05$ is sufficient for most samples that we tested, but there are cases where it failed in adjusting to some of the curvature changes. On the other hand, by choosing a smaller cutoff value, one grows a large tree, which tends to pick up random patterns as well. But as we discussed in the previous section, it has been proven that growing a large tree at first, and then selectively pruning back, or recombining those unnecessary nodes can overcome this problem. Therefore, it is appropriate to choose a small cutoff value for C_0 so as not to miss any critical splits and $C_0 = 0.01$ is used in our simulation.

However, AIC_C is not directly related to the size of the partition. Hence it does not allow the *minimum cost-complexity* pruning that is recommended for the right-sized tree. Therefore, in our implementation we use a small value of C_0 to grow a big tree, and then use a tree-size adjusted version of AIC_C as our *cost-complexity* measure. Our measure is defined as

$$AIC_p(\alpha) = \log(\hat{\sigma}^2) + \psi_n \{tr(H) + \alpha(\|T\| - 1)\}, \quad (3.3)$$

where α , and $\|T\|$ are the unit penalty and the size of the subtree T , respectively. We propose a *minimum cost-complexity* pruning algorithm based on AIC_p . In the algorithm, α_0 is the maximum unit penalty to be imposed for nodes to be pruned. This is a threshold which may be selected by the user. In the algorithm ψ_n^{-1} is the inverse function of ψ_n defined in (2.16), specifically,

$$\psi_n^{-1}(s) = \frac{(n-2)s - n}{s+1}, \quad s > 0. \quad (3.6)$$

The implementation of our method is outlined as Algorithm 3.1.

Algorithm 3.1. Minimum cost-complexity pruning using AIC_p in (3.3).

```

repeat
  Loop for each non-terminal node  $t$ ,
    i.  $\alpha_{min} = \text{infinity}$ 
    ii. Calculate the  $AIC_p(t)$  assuming all children of  $t$  are pruned.
    iii.  $\alpha(t) = \{\psi_n^{-1}(AIC_p(t) - \log(\hat{\sigma}^2)) - tr(H)\} / (\|T\| - 1)$ 
    iv. If  $0 < \alpha(t) < \alpha_{min}$ ,
            $\alpha_{min} = \alpha(t)$ 
    v. end if
  end loop
if  $\alpha_{min} < \alpha_0$ 
  Prune all children of  $t = \text{argmin}(\alpha_{min})$ , the branch that is
  associated with  $\alpha_{min}$ .
end if
while  $\alpha_{min} < \alpha_0$ .

```

4. SIMULATION RESULTS

4.1. Simulation Functions and Setup

Pitblado (2000) uses five simulation functions to evaluate his equal-in-length variable bandwidth selection methods. He names these m_1 through m_5 . These functions range from a quadratic function m_1 , which has constant second derivative and hence curvature does not change, to m_5 , which mimics the motorcycle impact data with linear combinations of exponential functions. The response of the motorcycle impact data is the acceleration (y) of the head of a motorcycle passenger during impact over time (x), see for example Fan and Gijbels (1996). These functions are also used by Adams (1998), and m_2 appears in Hurvich, Simonoff, and Tsai (1998). We have shown m_2 , m_3 , and m_5 in previous examples and graph the other functions in this section. In addition we add one more function and name it m_6 . It also appears in Fan and Gijbels (1995) and helps compare our method with their FVB method. For completeness the functions are defined in Table 4.1 and plotted in Figure 4.1.

The ranges of the respective mean functions, denoted by R_m , are also given in Table 4.1 (adapted from Pitblado (2000)). However, we scale the domains of the mean functions, the range of

x to be in the same interval $[0, 1]$, so that the bandwidths we select are comparable among the functions as well. This means that the same bandwidth covers the same number of point pairs for samples of the same size from all of the six functions.

As discussed in Pitblado (2000), all these mean functions except m_1 have varying degrees of curvature, which is a function of their corresponding second derivatives. Some of them like m_5 have large spatial variability, which implies extensive curvature changes. Function m_1 has no spatial variability since $m_1''(x) = -2$, a constant. So a global bandwidth is most appropriate for m_1 . Therefore, it serves as a good test of the algorithms for overfitting. The function m_2 may require two to three bandwidth partitions since it is a linear combination of two different normal densities. A partition of three is most appropriate for m_3 and m_4 to model the sine wave and the normal density in the middle, respectively. On the other hand, m_5 and m_6 may require more partitions. They do not have the clear-cut patterns of the first four functions. We will show how the proposed *RPVB* algorithm will automatically find a sensible partition for all of these functions in the presence of varying amounts of random error.

It is worth mentioning that the proposed *RPVB* method minimizes AIC_C by balancing the residual sum of squares and the smoothness of the regression. Specifically, for m_1 , where a global bandwidth is most appropriate, a good result is achieved by trading some roughness for smaller variance. For m_3 , the variance is a little larger for the variable bandwidth in order to capture the sine wave in the middle. For m_3, m_4, m_5 , and m_6 , both SSE and $\text{tr}(H)$ are smaller for *RPVB* due to the fact that variable bandwidth captures the changes of curvature by varying the bandwidth accordingly. All of these results are accomplished with a completely data-driven method.

Table 4.1. Simulation functions

$m_1(x) = x(1-x), x \in [0, 1], R_m = 0.25$
$m_2(x) = .3 \exp\{-64(x-.25)^2\} + .7 \exp\{-256(x-.75)^2\}, x \in [0, 1], R_m = 0.74$
$m_3(x) = 5 \{1 + \exp\{3.2 - 6.4x\}\}^{-1} + \sin\{30\pi(x - 1/3)\} \delta_{[1/3, 2/3]}(x), x \in [0, 1], R_m = 4.60$
$m_4(x) = 2(2x - 1) + 2 \exp\{-40(2x - 1)^2\}, x \in [0, 1], R_m = 4.00$
$m_5(x) = 4.26 \{ \exp\{-9.75x\} - 4 \exp\{-19.5x\} + 3 \exp\{-29.25x\} \}, x \in [0, 1], R_m = 1.30$
$m_6(x) = \sin(2x) + 2 \exp(-16x^2), x \in [0, 1], R_m = 3.03$

We use FVB as the competing method for the performance of our new *RPVB*. It represents the class of equal-in-length methods with published results. It has the necessary flexibility for capturing complicated shapes and curves. The authors argue that their method has spatial adaptation properties that are similar to wavelets. Also Professor Fan provided us with the software to calculate FVB estimates for our simulation, for which we acknowledge our deep appreciation.

The simulation takes the following steps: First $M = 400$ Monte Carlo replications independently for each function. Pilot tests at $M = 100$ indicate that the standard errors would be sufficiently small at $M = 400$. Then for each Monte Carlo run, sample sizes of $n = 50$ and 100 are used. Because *RPVB* is computationally intensive, larger sample sizes are not used. The equally spaced predictor points (x_1, \dots, x_n) are also used as the estimation points. The standard deviations of the normal error terms are 5%, 15%, and 20% R_m , the range of the corresponding mean function.

In the simulation, AIC_p is our cost-complexity measure. Initial tree growing stops when a split fails to reduce AIC_C by $C_0 = 0.01$, or an interval includes only 5 or fewer data points. The stopping

rule for tree pruning is $\alpha_0 = 1.0$, where α_0 is the maximum per-node penalty for a node to be pruned as described in Section 3.3.

4.2. Minimization Consideration

One successful feature of *RPVB* is the method for minimization of AIC_C in both the tree growing and the tree pruning processes. For each candidate split of an interval, we search for the bandwidths (h_1, h_2) for the two subintervals generated by the split that minimize AIC_C , while holding the bandwidths in all other intervals constant. Since this optimization is done for all candidate splits, it has a big impact on the stability and speed of this new method. We investigated two completely different approaches.

One approach uses one of the conventional numerical optimization routines, specifically the Downhill Simplex algorithm in multidimensions. For a detailed discussion of this method, see §10.4 of Press, et al. (1992). This method requires only function evaluations not derivatives.

The AIC_C function of the h 's, which must be minimized, has a very complicated form. Hence such numerical methods for finding a minimum do not work well. Newton-Raphson type methods do not always succeed in finding a minimum, not to mention not finding the true global minimum. Recall that the criterion value is based on estimation for all predictor values. Failure at a single point not only means the failure in finding the split, but also the whole recursive partitioning process. A feasible alternative was two-dimensional grid of points.

In each dimension, those grid proposed in Fan and Gijbels (1995) for the one-dimensional case, which we extend into two dimensions to serve our purposes. This method is to compare AIC_C values at a finite points are set to be of geometric type, i.e., $h_j = d^j h_{\min}$, where h_{\min} denotes the first grid point and d is the grid spacing. Suppose we want to minimize AIC_C for $(h^{(\text{left})}, h^{(\text{right})})$ in a $[h_{\min}, h_{\max}] \times [h_{\min}, h_{\max}]$ region. Each of the two h 's starts from h_{\min} , are inflated separately by a factor d , and the AIC_C values calculated at the full array of grid points.

Choose the minimizer of AIC_C as the grid point having the smallest computed AIC_C value. The advantage of this method is that it is stable. One always finds a minimum or a solution that is close enough to the minimum. This is very important for the success and credibility of our simulation. The disadvantage is that it is much slower than the Newton-Raphson type methods in our two-dimensional case.

Fan and Gijbels (1995) use $h_{\min} = (X_{(n)} - X_{(1)}) / n$, $h_{\max} = (X_{(n)} - X_{(1)}) / 2$, and $d = 1.1$. Using the same settings in our simulation, $h_{\min} = 0.01$ and $h_{\max} = 0.5$, we need 42^2 , or 1764 evaluations for each minimization. This is quite slow. So in our simulation, we use $h_{\min} = 3(X_{(n)} - X_{(1)}) / n$, $h_{\max} = (X_{(n)} - X_{(1)}) / 4$. The number of evaluations needed for each optimization reduces to 23^2 , or 529. These approximations reduce the adaptability of our proposed method. However, we see later in this section that our method still compares favorably to the FVB method.

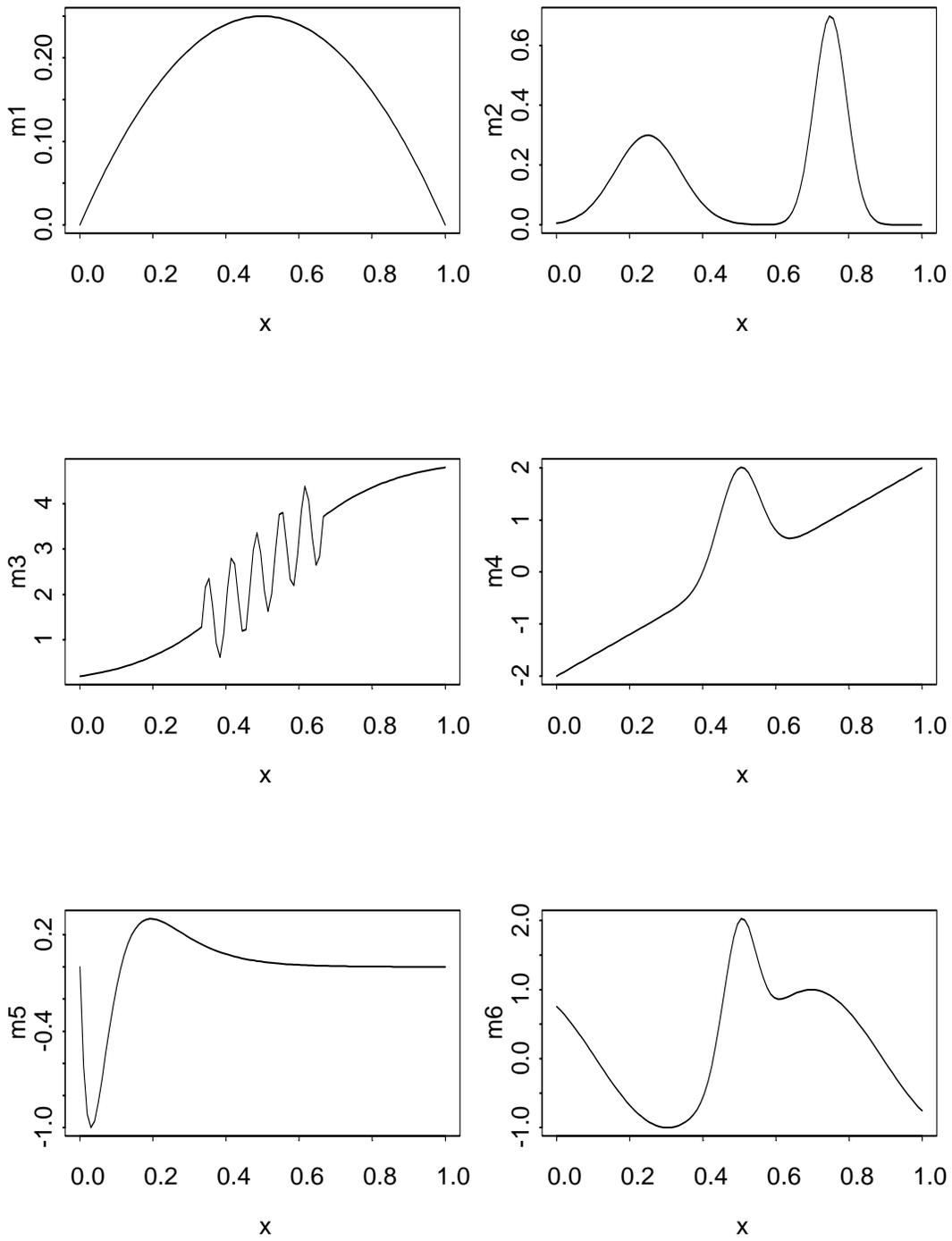


Figure 4.1. Plots of Simulation Functions

4.3. Performance Comparison Criteria

In this section we define two measures for comparison. The familiar Relative Efficiency (RE) is a global measure. Mean Absolute Difference Ratio (MADR) is introduced to compare the local performance at a given estimation point of interest. Unless otherwise specified we use a level of $\alpha=0.05$ to test whether *RPVB* performs significantly different from *FVB*, i.e. $H_0: \theta = 1$, where θ is either RE or MADR.

4.3.1. Global comparison measure - RE

Relative Efficiency (RE) is a conventional measure to compare the overall performance of two smoothing techniques. For each random sample $j = 1, 2, \dots, M$, we estimate $\mathbf{m}^t = (m(x_1), \dots, m(x_n))$, the regression function at each of the observed predictors $\mathbf{x}^t = (x_1, \dots, x_n)$. We define Mean Squared Residuals (MSR) as

$$MSR = \frac{1}{Mn} \sum_{j=1}^M \left\{ \sum_{i=1}^n [\hat{m}_j(x_i) - m(x_i)]^2 \right\}, \quad (4.1)$$

where $\hat{m}_j(x_i)$ denotes the estimated mean at x_i for sample j . To compare the performance of two methods, say *RPVB* and *FVB*, the relative efficiency (*RE*) of the two methods is defined as the ratio of the corresponding MSR, i.e.,

$$RE = \frac{MSR_{RPVB}}{MSR_{FVB}}. \quad (4.2)$$

This is our measure for global performance. A *RE* less than 1 (or 100%) indicates that *RPVB* compares favorably to *FVB*. The smaller *RE* is, the more superior *RPVB* is relative to *FVB*.

4.3.2. Local comparison measure - MADR

With any smoothing technique it is important to investigate its performance with respect to oversmoothing. However, relative efficiency is not helpful in this regard. This can be achieved by comparing the two methods at local minima or maxima, where such oversmoothing occurs. In Section 4.5 we select one such point from each simulation function to do this comparison. We defined a statistic for this purpose, the *Mean Absolute Difference Ratio (MADR)*. Suppose we want to compare the two methods at a mode x_0 based on m Monte Carlo samples. Let $m(x_0)$, $\hat{m}_j^{RPVB}(x_0)$, and $\hat{m}_j^{FVB}(x_0)$ denote the true mean, *RPVB* estimate for sample j , and *FVB* estimate for sample j at x_0 , respectively. Then *MADR* is defined by

$$MADR = \frac{\sum_{j=1}^M |\hat{m}_j^{RPVB}(x_0) - m(x_0)|}{\sum_{j=1}^M |\hat{m}_j^{FVB}(x_0) - m(x_0)|}, \quad (4.3)$$

which measures the relative closeness of the estimates to the true mean at a given peak or valley. *MADR* like *RE* is a ratio statistic. If it is significantly less than 1, we have evidence to say that *RPVB* compares favorably to *FVB*. Another comparison that should be conducted is the undersmoothing,

but there is no conventional measure for this purpose. However, *RPVB* uses AIC_C as the bandwidth selection criterion, which has the built-in “roughness” penalty. *FVB* on the other hand only exploits residual sum of squares, which estimates the local mean-squared error.. Therefore, *RPVB* is expected to produce smoother estimates than *FVB*. This is substantiated by visual inspection of the plots of their estimated curves.

4.4. Overall Comparison of *RPVB* versus *FVB*

In this section we present one set of results for both *RPVB* and *FVB* with the six simulation functions in Section 4.1. The additive error standard deviation is 5% of the respective range of the mean functions. Sample size is $n = 100$, and $M = 400$ independent samples were analyzed for each function. Minimum and maximum bandwidths are set to be 0.03 and 0.25, respectively. Tables 4.2 shows the relative efficiency of the two methods. The standard errors (s.e.) of the ratios in (4.2) are based on the sample variances and covariances of the 400 pairs of inner sums in (4.1).

Table 4.2. Relative Efficiency of *RPVB* to *FVB*. Each based on $M = 400$ Monte Carlo simulations, sample size $n = 100$, bandwidth $h > 0.03$.

Function	MSR of <i>FVB</i>	MSR of <i>RPVB</i>	RE	s.e. of RE	Mean (med) partition
m_1	0.0000157	0.0000164	1.043	0.0223	1.07 (1)
m_2	0.0003138	0.0002938	0.936	0.0062	2.21 (2)
m_3	0.1680993	0.0464788	0.277	0.0025	3.29 (3)
m_4	0.0073811	0.0068286	0.925	0.0099	3.24 (3)
m_5	0.0012078	0.0009767	0.809	0.0058	3.02 (3)
m_6	0.0053930	0.0050313	0.933	0.0091	3.58 (3)

For all six sample functions, except m_1 , the RE are significantly less than 1 (at 95% confidence level), which implies that *RPVB* is preferred to *FVB*. RE is 1.043 in the case of m_1 , which is not significantly greater than 1 due to the large variance. However *RPVB* elects to use global bandwidth more than 88.6% of the time with mean partition size 1.07, while *FVB* uses a fixed five-interval partition. Therefore this is expected and should not be considered as a drawback of *RPVB*.

RPVB outperforms *FVB* significantly for m_3 and m_5 . For m_3 *FVB* failed to detect the sine wave in the middle for 99% of the random samples while our new method detected every one of them. Figure 4.3 shows a typical fit. The four cases in 400 runs that *FVB* does detect the sine waves are because the sine waves in those samples are almost perfect, meaning that essentially there was very little random noise. Even in these cases however, *FVB* has some trouble by oversmoothing the sine waves and undersmoothing both left and right sides. We examine these comparisons function by function in the next section.

4.5. Size of Partitions of *RPVB*

Size of partition means the number of intervals in the partition. In this section we analyze the size of partitions that *RPVB* yields. *FVB* have a fixed size of partitions. Fan & Gijbels (1995) in their simulation used an empirical value $\lceil n / 10 \log(n) \rceil$, or a constant five interval partitions in our simulation setting where $n = 100$. The number of intervals in the partition increases only with sample size, regardless of the data relationship. .

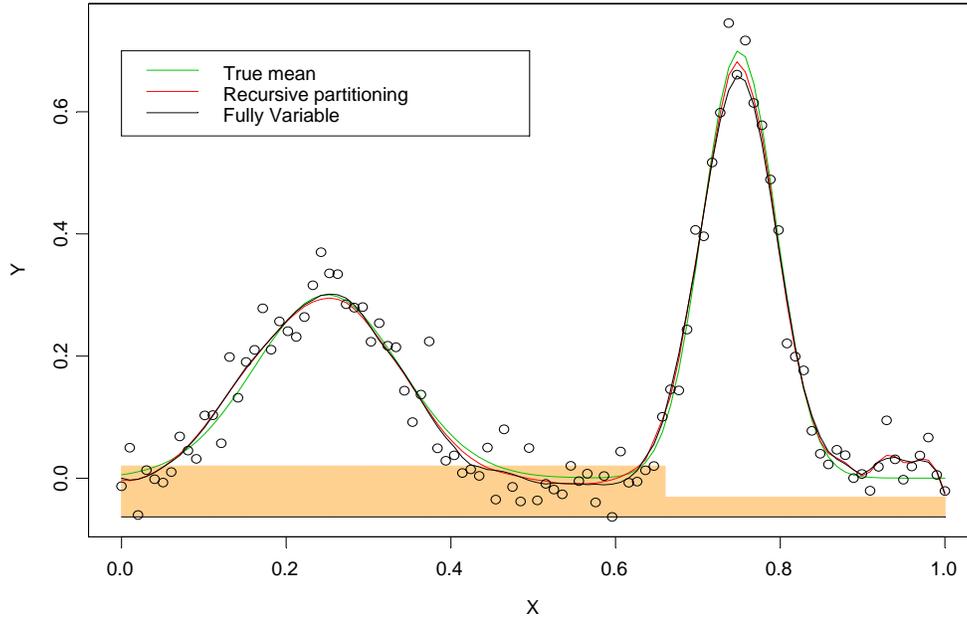


Figure 4.2. A typical two-interval partition for m_2 .

The mean and median partition sizes $RPVB$ generated for the six functions are listed in the last columns of Table 4.2. We will discuss them in detail by function.

The $RPVB$ method in most of the samples uses a global bandwidth for function m_1 , the quadratic function with constant second derivative. This is consistent even when we introduce larger error. In the case when $h > 0.03$, 88.61% of the samples did not even find an initial split that improves the AIC_C by 1%. Some of those that did were also pruned all the way back. Investigation of those samples that did use variable bandwidths reveals that they form clear random patterns that do appear to change in curvature and hence the choice of variable bandwidths is sensible.

A two-interval partition is appropriate for m_2 , one for each mode. A third interval for the transition area in the middle is also justifiable. This is exactly what $RPVB$ found. The 400 Monte Carlo runs yield mean partition size 2.41 and median 2. A typical fit is shown in Figure 4.2. Among the limited three-interval partitions, the largest bandwidth goes to the almost flat middle area between the two modes, which are acceptable.

The mean function m_3 is defined to be three equal-length pieces with a sine function in the middle $(1/3, 2/3]$. Therefore a three-interval partition is ideal for m_3 . The mean partition size out of the 400 simulation runs (when $h > 0.03$) is 3.29 and the median is exactly 3, which implies that $RPVB$ automatically detected the curvature changes and finds a partition of 3 in most of the cases. On the other hand, FVB fails in 99% of the samples to detect the sine wave. Typical $RPVB$ and FVB fits are shown in Figure 4.3.

The function m_4 is defined as a single normal density added in the middle of a straight line. So a symmetric three-interval partition like in the case of m_3 is the right partition except the normal density damps off beyond interval $[-.0125, 0.125]$, which is about $1/5$ of the range of support. This is consistent with what $RPVB$ achieved. The mean partition size is 3.24 and the median is 3. Typical

RPVB and *FVB* fits of m_4 are shown in Figure 4.4. *FVB* is successful in fitting the normal density but it undersmooths the straight line very often.

RPVB also consistently yields sensible partitions for function m_5 . This function simulates the *motorcycle impact data*. The sharp curve at the bottom to the left mimics the effect of the impact. The function then rises rapidly back to its peak and gradually levels off. Therefore partitions with three or more intervals with increasing bandwidths from left to right are expected. This matches what *RPVB* found. The mean and median partition size are 3.02 and 3 respectively. A typical *RPVB* fit of m_5 is shown in Figure 4.5. For a function with such large spatial variability, *RPVB* demonstrated superior spatial adaptation. It minimized oversmoothing the steep curve to the left with very small bandwidth that is almost equivalent to interpolation, while it minimized undersmoothing the flat right side with a large bandwidth that is almost equivalent to regular linear regression.

Finally, m_6 is the sum of a sine function and a normal density. A partition of three to five would be appropriate. This matches the simulation results, whose mean partition is 3.58 and median is 3. A typical *RPVB* fit is shown in Figure 4.6.

Based on the above discussion we close this section with the following remarks.

- *RPVB* performed satisfactorily in automatically finding the right partitions in all six simulated functions.
- The partitions are not restricted to equal-in – length ones, and do reflect the relationship or curvature changes in data. And hence,

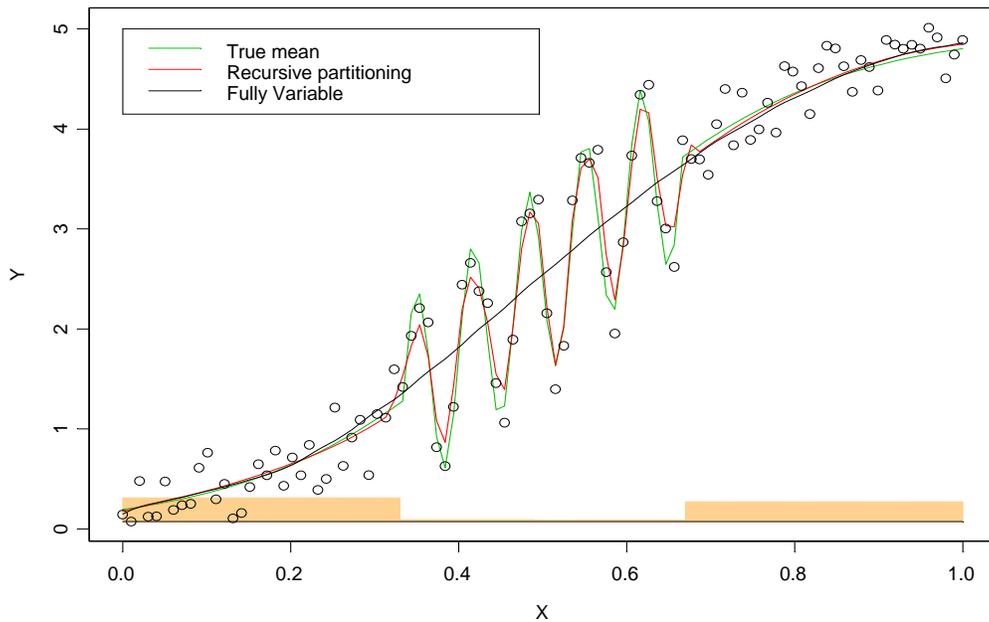


Figure 4.3. Typical *RPVB* and *FVB* fits for m_3 .

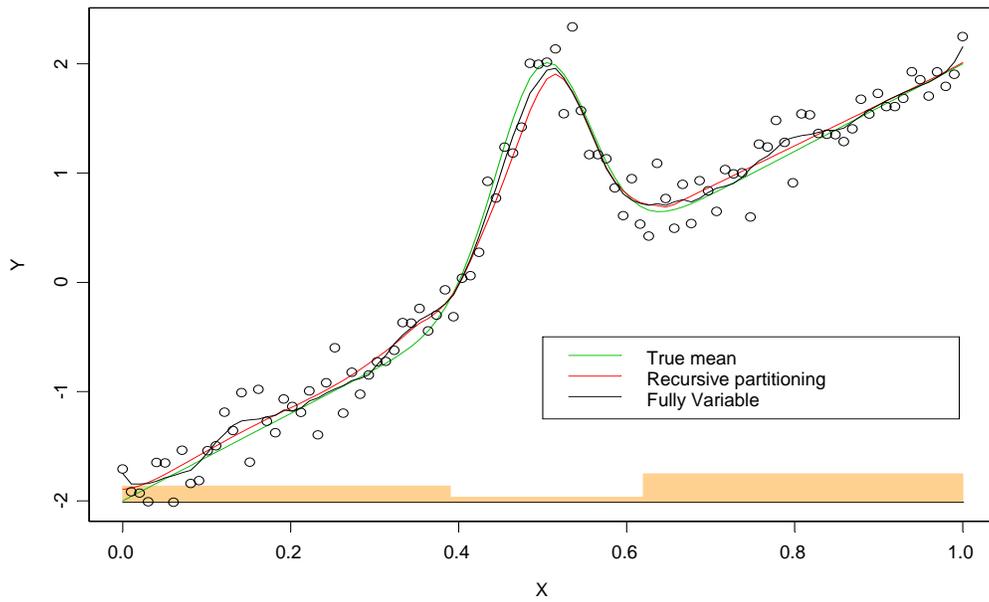


Figure 4.4. A typical RPVB fit for m_4 .

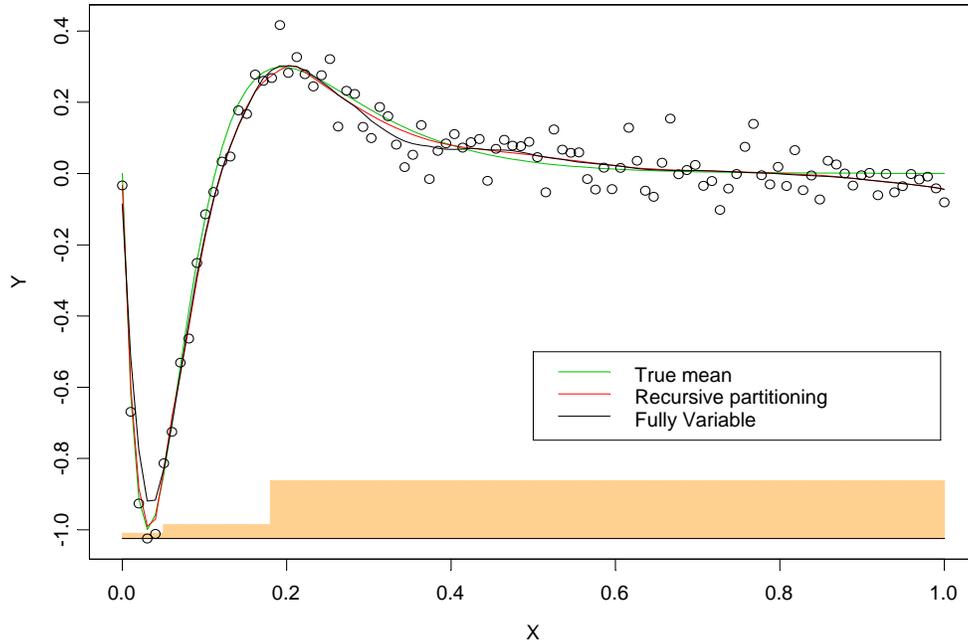


Figure 4.5. A typical RPVB fit for m_5 .

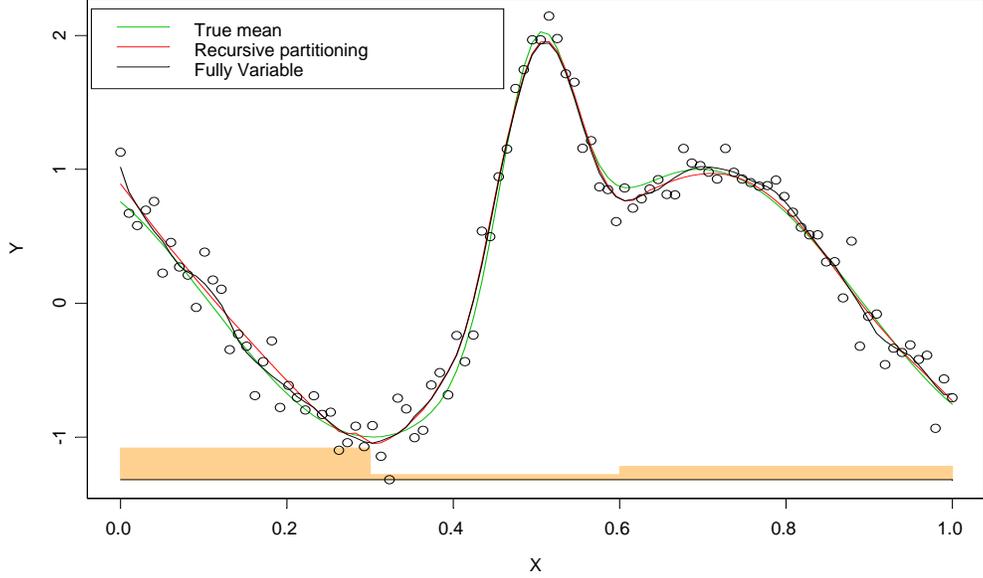


Figure 4.6. A typical RPVB fit of m_6 compared with FVB.

- *RPVB* reaches smaller partitions than *FVB*, which simply uses 5 equal-in-length intervals for all the functions with our simulation settings.
- *RPVB* gives appealing estimates of the piecewise-constant bandwidth corresponding to the partitions it yields.
- *RPVB* results in smoother estimated mean functions.
- Allowing the selection of smaller h 's helps reduce oversmoothing, but it is also more likely to pick up small random patterns and results in more partitions. The choice should be made based on anticipated application.

4.6. Local Comparison

In Section 4.4 we investigated the overall performance of *RPVB* and demonstrated that it compares favorably with *FVB* for all six simulated functions. In this section we investigate the performance of *RPVB* method at specific estimation points. We select one extreme point for each test function except m_1 , and investigate how close the estimates *RPVB* and *FVB* are to the true mean values using the local measure in Section 4.3. The locations for the five test functions are listed in Table 4.5. They are all local extrema in the functions and are selected as being difficult for any method.

Table 4.3. Local extrema selected for local comparison

Function	X_{mode}	Description.
m_2	0.57	Only minimum, change point.
m_3	0.36	First peak to the left. Close to change point.
m_4	0.50	Only maximum.
m_5	0.04	Minimum. Steep curve.

m_6	0.70	Second lower maximum. Close to inflection point.
-------	------	--

As defined in Section 4.3, we use the *Mean Absolute Difference (Ratio) (MADR)* to compare our *RPVB* with the *FVB*. A *MADR* less than 1, or 100% implies our proposed *RPVB* estimates are closer to the true mean than *FVB*. The smaller *MADR* is, the more superior *RPVB* is to *FVB*. Table 4.4 list the *MAD* and *MADR* from 400 Monte Carlo runs per function. The following conclusions can be made.

- *RPVB* proved to be superior to *FVB* for m_3 and m_5 . For m_3 this is because *FVB* failed 99% of the time in capturing the sine wave. In the case of m_5 it is because of *RPVB*'s adaptability to spatial variability in its partitions, which results in less oversmoothing when there are such extensive curvature changes.
- *MADR* is not significantly different from 1 for the other functions, which implies that *RPVB* is comparable to *FVB* locally for these functions. This is confirmed in the scatterplots of the sample absolute differences of these two methods.
- Allowing smaller bandwidths reduces oversmoothing significantly. In a full replication of the simulation with $h > 0.015$, *MADR* drops from 47.4% to 26.2% for m_3 and from 81.1% to 42.9% for m_5 . However, it is also more likely to pick up random patterns and make the estimates unstable. For example a *MADR* of 99% when $h > 0.03$ increases to 107% when $h > 0.015$ for m_6 .

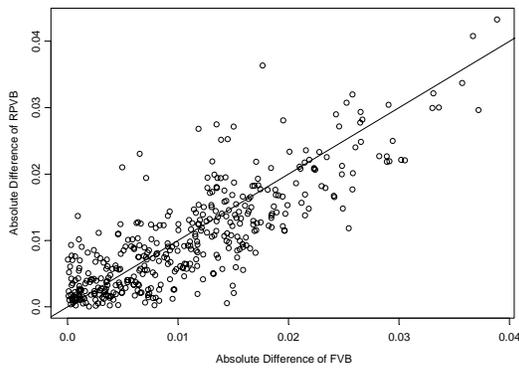
Table 4.4. *MAD* and *MADR* statistics for local comparison

Function	X_{mode}	<i>MAD</i> of <i>RPVB</i>	<i>MAD</i> of <i>FVB</i>	<i>MADR</i>	s.e. of <i>MADR</i>
m_2	0.57	0.0103	0.0109	0.946	0.3389
m_3	0.36	0.4675	0.9856	0.474	0.0055
m_4	0.50	0.1278	0.1278	1.000	0.2035
m_5	0.04	0.0834	0.1028	0.811	0.0126
m_6	0.70	0.0520	0.0525	0.990	0.1813

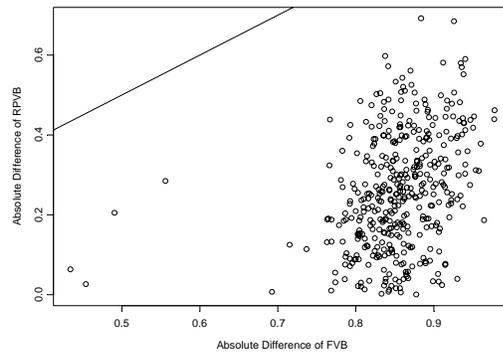
Now let us have a closer look behind the summary statistics. Figure 4.7 shows the joint distribution of Absolute Differences, $|m_2(x_0) - \hat{m}_2(x_0)|$, of *RPVB* and *FVB* for *the sample functions at selected estimation points* based on 400 Monte Carlo runs. In the scatterplots of absolute differences a point below the diagonal line implies that the estimate of *RPVB* at the given point x_0 is closer to the true mean (above or below) than that of *FVB* for that sample, otherwise *FVB* estimate is closer. In this sense we may call this the break-even line. If the majority of the samples are below this line, it implies that *RPVB* is favorable, and *vice versa*. Here we want our judgments to be based on the general trend rather than specific samples. Individual estimates could be far off the true mean because of random patterns.

The *MADR* ratio for m_2 at $x_0 = 0.57$ is 94.6%, which is not significant different from 1. In Figure 4.7 (a) however, the scatterplot does indicate that our *RPVB* performs slightly better than *FVB*, especially for the larger differences.

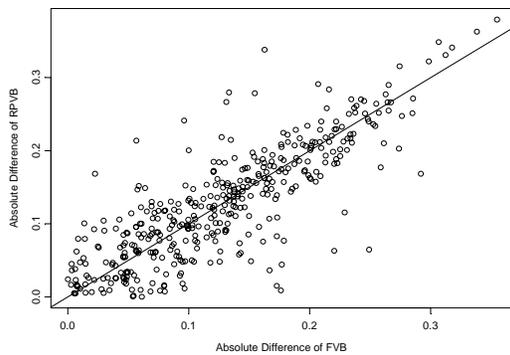
RPVB estimates at the given location $x_0 = 0.36$ for m_3 are much closer to the true mean than those of *FVB*. The *MADR*s is 47.4%. The distribution of the absolute differences at $x_0 = 0.36$ based on 400 runs is shown in Figure 4.7 (b). *RPVB* estimates are much better than *FVB* in



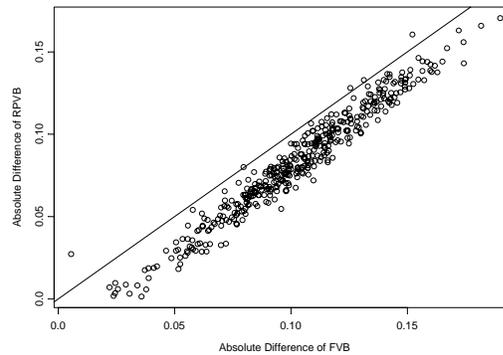
(a)



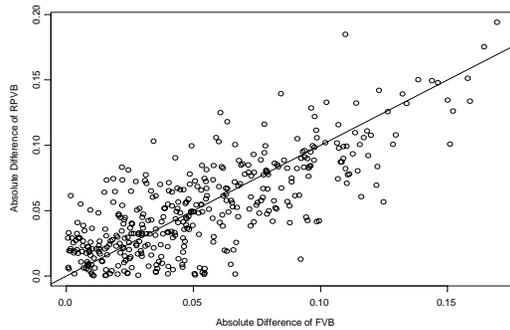
(b)



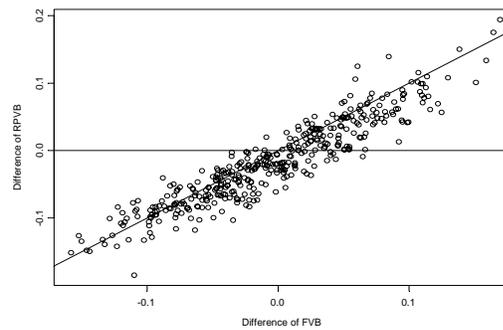
(c)



(d)



(e)



(f)

Figure 4.7. The scatterplot of Absolute Differences of (a) m_2 at $x_0 = 0.57$, (b) m_3 at $x_0 = 0.36$, (c) m_4 at $x_0 = 0.50$, (d) m_5 at $x_0 = 0.04$, (e) m_6 at $x_0 = 0.7$ and (f) signed differences of m_6 at $x_0 = 0.7$. All based on 400 Monte Carlo runs:

all 400 samples, including the four cases in which FVB succeeded in depicting the sine wave. RPVB successfully recognized the sine wave without undersmoothing the gently curving sides in each run of the 400 samples. This requires both appropriate partitions and variable bandwidth selections. Yet

all these are achieved without any *a priori* knowledge or even a hint, but rather totally driven by the data. This is an excellent demonstration of the data adaptive potential of our method.

Both RPVB and FVB did well for function m_4 because of its relatively small curvature change. Even a global bandwidth would yield a decent estimate. The joint distribution of absolute differences at $x_0 = 0.50$ based on 400 random samples is shown in Figure 4.7 (c).

RPVB estimates for m_5 at the given location are also much closer to the true mean than those of FVB. The MADR against FVB is 81.1%. This is consistent with the distribution shown in Figure 4.7 (d). RPVB almost always yields a closer estimate of $m_5(.04)$. On average the estimates are about 20% closer to the true mean when using RPVB than using FVB.

For m_6 , the location $x_0 = 0.7$ is very close to the sharp peak at $x = 0.51$ with an inflection point in between at $x = 0.65$. Some samples have erratic patterns around the inflection point. There are a lot of cases that can be interpreted as having quite different curvature. Therefore the estimates in this neighborhood are not very stable. Figure 4.7 (e) shows the joint distribution of the absolute differences of the 400 sample runs. The high correlation between the absolute differences yielded by RPVB and FVB reflect that these two methods perform similarly in this situation (MADR = 99%).

More insight can be gained from the signed differences, $m_6(x_0) - \hat{m}_6(x_0)$, which are shown in Figure 4.7(f), including $y = 0$ in addition to the diagonal to emphasize that these are not absolute differences. However, when more pairs fall below the break-even line this means that RPVB estimates tend to be larger than the FVB estimates. The samples with positive differences below the break-even line indicate that both estimates are below the true mean but RPVB estimates are closer. The samples on the negative side under the line mean that both estimated values are above the true mean and RPVB estimates are higher. The behavior of RPVB at this unstable point of m_6 actually demonstrates that it is responsive to curvature changes. Even though RPVB and FVB perform similarly at this specific location, globally RPVB still performs better, specifically about 7% more efficient than FVB. Moreover, RPVB estimates? ??

RPVB is one type of pattern recognition; it tries to find partitions by exploiting the relationships in the data, curvature changes in particular. When insufficient sampling or excessive noise obscures the true signal, it is hard for RPVB or any smoothing technique to produce reliable estimates. We assessed how these factors affect the performance of our proposed method with two different settings, samples size $n = 50$ and error standard deviations 15% R_m .

When $n = 50$. RPVB is preferred on overall efficiencies relative to FVB for m_4 , m_5 and m_6 , but only comparable for m_2 and m_3 . FVB is preferred to m_1 RPVB lost its edge for m_3 when $n = 50$ since it fails in most of the cases to detect the sine function in the middle and hence behaves similarly to FVB. This suggests that at this sampling rate, RPVB does not have enough evidence to detect the curvature change. The seeming inefficiency of RPVB for m_1 is because RPVB weights smoothness over bias. Local comparisons for $n = 50$, however, indicate that RPVB compares favorably for all test functions except m_4 .

When we increase the standard deviation of additive errors to $\sigma_e^2 = 15\% R_m$, the overall relative efficiencies are mixed. RPVB leads significantly for m_3 and m_5 , is comparable with FVB for m_1 , m_2 and m_6 , but loses significantly for m_4 . The same comparisons are observed in the local comparison. The lead in m_3 is explained by the fact that RPVB still recognizes the sine curve in the middle. But only for some samples in this case due to the large error. However, RPVB is able to raise the bandwidths on both sides to adapt to the gentle curves unless there are significant sinusoid patterns. It is observed that undersmoothing by FVB is typical at this error rate. The large spatial variability of m_5 is hard to conceal even when large error is introduced. Therefore it is not surprising

that RPVB is more successful with its superior data adaptivity. It consistently yields 3-interval partitions that are appropriate for m_5 . It is not surprising that RPVB is less successful for m_1 , m_2 , and m_6 . These functions have less curvature, which is more easily to be buried in the noise, harder for RPVB to recognize. Function m_4 is an extreme of this situation.

When we increase the error standard deviation to 20% R_{m_i} , the situation is somewhat different. In this case here the curvatures are almost completely concealed by the large additive error. This is reflected by the fact that the median partition sizes are 1 for of m_1 , m_3 , and m_4 and 2 for the rest. RPVB is preferred locally for all functions except for m_2 and m_4 , for which RPVB chooses global bandwidths and yields smoother estimates. Relative efficiencies indicate that RPVB is preferred for of m_1 , m_3 , and m_5 , while FVB is preferred for m_2 , m_4 and m_6 . However, only m_6 is significantly worse in both comparisons. This is because m_6 plus noise often looks like a big “S” with the second lower maximum being concealed, in which case RPVB picked a two-interval partition instead. This choice looks more natural for these data give that one does not know the true mean function. However, FVB is significantly undersmoothing all six functions. RPVB remains the preferred choice even though it also fails to recognize the sine wave, which is totally buried in the noise.

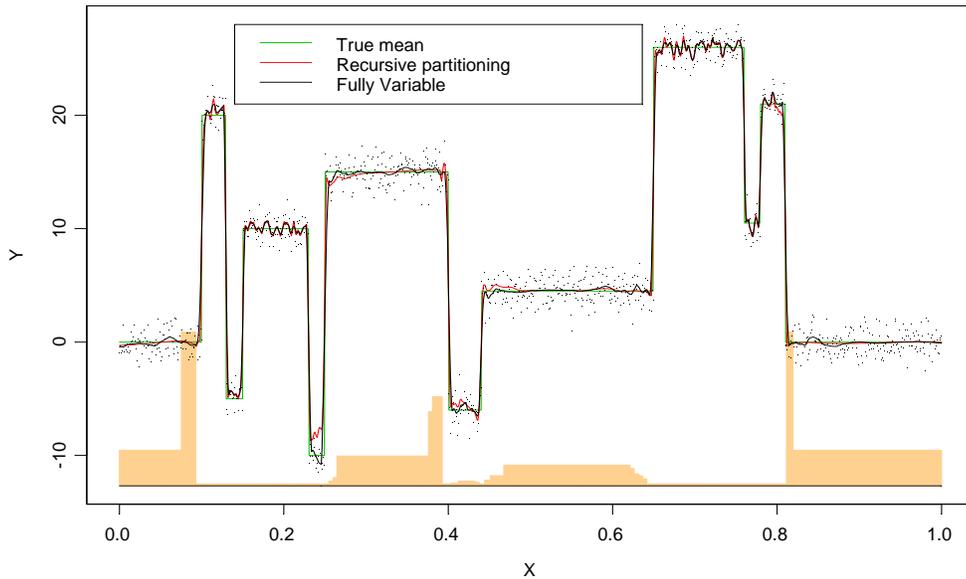


Figure 4.11. RPVB and FVB estimates of the Noisy Blocks function ($n = 1024$, $\sigma_e^2 = 1$).

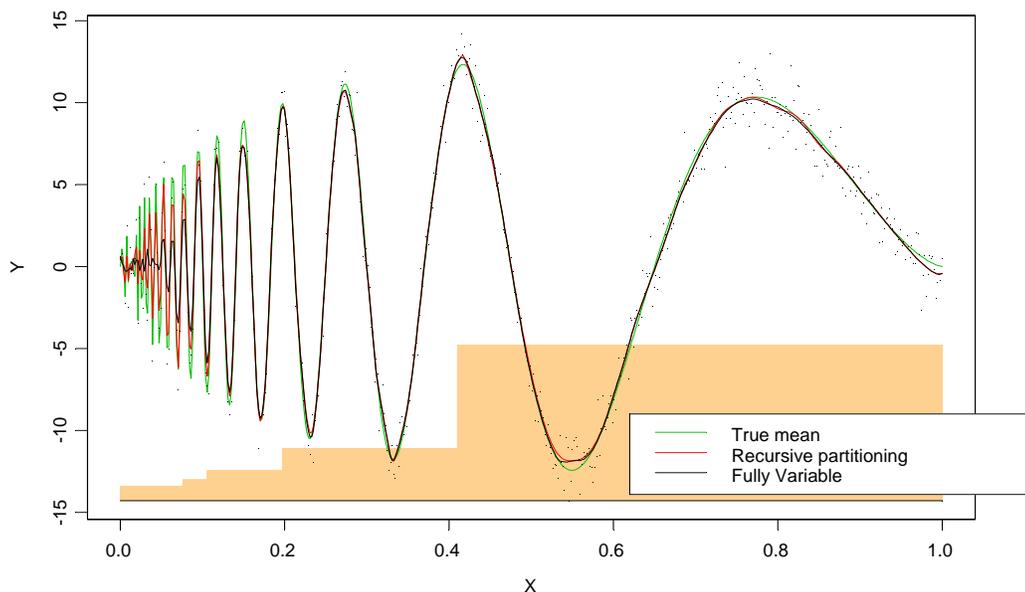


Figure 4.12. RPVB and FVB estimates of the Noisy Doppler function
($n = 512$, $\sigma_e^2 = 1$).

As mentioned in Section 2.1, Fan and Gijbels argue that local polynomial regression using their data-driven variable bandwidth has spatial adaptation properties that are similar to wavelets. They used two other test functions, *the Noisy Blocks* and *the Noisy Doppler*, defined in Donoho and Johnstone (1994), to compete with their wavelet shrinkage method. Figures 4.11 and 4.12 exhibit RPVB and FVB estimates of these two functions. The bandwidths are rescaled to be visible. RPVB and FVB estimates for the Noisy Blocks are comparable. RPVB outperforms FVB for the Noisy Doppler function with much less oversmoothing for the high frequency area between 0 and 0.15. The variable bandwidths match the trend of curvature changes of the target function. Our estimates are also comparable to wavelet shrinkage in Donoho and Johnstone (1994).

5. CONCLUSIONS

5.1. Summary

In this paper we developed a tree-based approach for local linear regression. This new methodology recursively partitions the range of the predictor variable based on the unknown structure of the relationship and simultaneously estimates the piecewise constant bandwidths. Our partitions do not have to be equal-in-length and are totally data driven as in CART. The recursive partitioning is based on a modified version of Akaike Information Criterion (AIC_C). We defined a partition-adjusted version of this criterion, called AIC_P , and used it to derive the optimal sized tree.

In Section 1 we briefly introduce the problem of local linear regression using variable bandwidth and surveyed the existing methods, mainly Fan & Gijbels (1995) and Pitblado (2000). In Section 2 we discuss the bandwidth selection problem and the AIC_C and AIC_P criteria. In Section 3 we give the details of the recursive partitioning algorithm and its implementation proposed here. The simulation results are presented in Section 4 with respect to global and local performance, as well as

the sizes of partitions on six test functions with direct comparisons to the fully variable bandwidth of Fan & Gijbels (1995).

Simulation results demonstrate that this new approach retains the adaptability to spatially inhomogeneous curves while it requires fewer smoothing parameters, and compares favorably to the FVB approach of Fan & Gijbels (1995). Even though it greatly enhanced the performance of RPVB for m_3 and m_5 to allow the choice of smaller ($h > 0.015$), it may be too sensitive in general. Hence focus on the results obtained with $h > 0.03$. In Table 5.1 we summarize both the overall and local comparisons of the two methods for the six functions in all simulation settings. We use “+” (“-“) to indicate that the corresponding statistic (RE or MADR) is significantly less (greater) than 1 at the 5% level. “0” indicates that we failed to reject the null hypothesis that the statistic is equal to 1.

Table 5.1. Summary of the global and local comparisons between RPVB and FVB
(+ : favorable, 0 : comparable, - : unfavorable)

		$n=100$ $\sigma_e^2=5\% R_m$	$n=50$ $\sigma_e^2=5\% R_m$	$n=100$ $\sigma_e^2=15\% R_m$	$n=100$ $\sigma_e^2=20\% R_m$
m_1	RE	0	-	0	+
	MADR	NA	NA	NA	NA
m_2	RE	+	0	0	-
	MADR	0	+	0	-
m_3	RE	+	0	+	+
	MADR	+	+	+	+
m_4	RE	+	+	-	-
	MADR	0	-	-	-
m_5	RE	+	+	+	+
	MADR	+	+	+	+
m_6	RE	+	+	0	-
	MADR	0	+	0	+

Among the 44 function-setting-measure combinations, RPVB outperforms FVB in 24 of the cases (55%). Two methods performed similarly in 11 cases (25%). FVB outperforms only in 9 cases (20%).

However, 7 out of the 9 factor combinations for which FVB wins were for functions m_4 (4), m_2 (2) and m_6 (1) with large additive error, where the curvature changes were concealed by the large error and RPVB choose smoothness over bias.

In summary, RPVB yields reasonable partitions based on curvature and sensible estimates of bandwidths. It consistently results in a smaller number of partitions while achieving similar MSE and smoother estimates than FVB. It finds a good balance between bias and smoothness. It demonstrated satisfactory adaptivity to data and the degree of adaptability could be tuned by the choices of C_0 and h_{\min} at the tree growing stage, and α_0 at the tree pruning stage.

REFERENCES

- Adams, B.E. (1998), “Scatter Plot Smoothing with Partially Variable Bandwidths,” unpublished dissertation, Southern Methodist University, Department of Statistical Science.
- Allen, D.M. (1974). “The relationship between variable and data augmentation and a method of prediction,” *Technometrics*, 16, 125-127.

- Breiman, L., Friedman, J.H., Olshen, R.A., and Stone, C.J., (1984), *Classification and Regression Trees*, London: Chapman & Hall.
- Donoho, D.L. and Johnstone, I.M. (1994), "Ideal spatial adaptation by wavelet shrinkage," *Biometrika*, 81, 425-455.
- Fan, J. (2000). "Prospects of nonparametric modeling", *Journal of American Statistical Association*, 95, 1296-1300.
- Fan, J. and Gijbels, I. (1992), "Variable Bandwidth and Local Linear Regression Smoothers," *The Annals of Statistics*, 20, 2008-2036.
- Fan, J. and Gijbels, I. (1995), "Data-driven Bandwidth Selection in Local Polynomial Fitting: Variable Bandwidth and Spatial Adaptation," *J. R. Statist. Soc. B*, 57, 371-394.
- Fan, J. and Gijbels, I. (1996), *Local Polynomial Modeling and Its Applications*, London: Chapman & Hall.
- Hall, P. and Johnstone, I. (1992). "Empirical functionals and efficient smoothing parameter selection (with discussion)." *Journal of the Royal Statistical Society, Ser. B*, 54, 475-530.
- Hastie, T., Tibshirani, R., and Friedman, J. (2001), *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, New York: Springer-Verlag.
- Hurvich, C. M. and Tsai, C. -L. (1989), "Regression and Time Series Model Selection in Small Samples," *Biometrika*, 76, 297-307.
- Hurvich, C.M., Simonoff, J.S., and Tsai, C.-L. (1998), "Smoothing parameter selection in Nonparametric Regression Using an Improved Akaike Information Criterion", *Journal of the Royal Statistical Society, Ser. B*, 60, 271-293.
- Jones, M.C. (1986), "Variable Kernel Density Estimates and Variable Kernel Density Estimates," *Australian Journal of Statistics*, 32, 361-371.
- Jones, M.C., Marron, J.S., and Sheather, S. J. (1996), "Progress in Data-Based Bandwidth Selection for Kernel Density Estimation," Mimeo Series 2088. Department of Statistics, University of North Carolina, Chapel Hill.
- Loader, C.R. (1999), "Bandwidth Selection: Classical or Plug-in?" *The Annals of Statistics*, 27, 415-438.
- Marron, J.S. and Padgett, W.J. (1987), "Asymptotically optimal bandwidth selection from randomly right-censored samples," *The Annals of Statistics*, 15, 1520-1535.
- Pitblado, J.S. (2000), "Estimating Partially Variable Bandwidths in Local Linear Regression Using an Information Criterion," unpublished dissertation, Southern Methodist University, Department of Statistical Science.
- Press, W.H., et al. (1992), *Numerical Recipes in C, 2nd ed.*, New York: Cambridge.
- Rudemo, M. (1982). Empirical choice of histograms and kernel density estimators. *Scand. J. Statist.*, 9, 65-78.
- Ruppert, D. (1997). "Empirical-bias bandwidths for local polynomial nonparametric regression and density estimation," *Journal of the American Statistical Association.*, 92, 1049-1062.
- Stone, M. (1974), "Cross-validatory choice and assessment of statistical predictions (with discussion)", *Journal of the Royal Statistical Society, Ser. B*, 36, 11-147.
- Wahba, G. (1977). "A survey of some smoothing problems and the method of generalized cross-validation for solving them". In *Applications of Statistics* (P.R. Krisnaiah, ed.), 507-523. North Holland, Amsterdam.