Automatic Smoothing for Discontinuous Regression Functions

Thomas C. M. Lee*

Colorado State University

September 20, 2000; Revised: February 28, 2001, November 5, 2001

Abstract

This article proposes an automatic smoothing method for recovering discontinuous regression functions. The method models the target regression function with a series of disconnected cubic regression splines which partition the function's domain. In this way discontinuity points can be incorporated in a fitted curve simply as the boundary points between adjacent splines. Three objective criteria are constructed and compared for choosing the number and placement of these discontinuity points as well as the amount of smoothing. These criteria are derived from three fundamentally different model selection methods: AIC, GCV and the MDL principle. Practical optimization of these criteria is done by genetic algorithms. Simulation results show that the proposed method is superior to many existing smoothing methods when the target function is non–smooth. The method is further made robust by using a Gaussian mixture approach to model outliers.

KEY WORDS: Akaike's information criterion, Discontinuity–Preserving, Generalized

*Postal: Department of Statistics, Colorado State University, Fort Collins, CO 80523-1877, USA; Email: `tlee@stat.colostate.edu`; Phone: (970) 491 2185; Fax: (970) 491 7895.

Cross–Validation, Genetic algorithms, Minimum Description Length, Regression Spline, Robust curve estimation

Running title: Automatic Smoothing for Discontinuous Functions

1 Introduction

This article considers the problem of nonparametric curve estimation (part of the material has been presented in Lee (1999)). Popular approaches to this problem include kernel/local polynomial regression, smoothing spline methods, regression spline smoothing and wavelet techniques.

We follow the regression spline approach. Our main contribution is the proposal of a new regression spline–based smoothing procedure capable of recovering curves with discontinuities. This procedure models an unknown curve by a series of disconnected cubic regression splines that partition the curve's domain. In this way boundary points between adjacent splines of a fitted curve serve as discontinuity points. The main problem with estimating an unknown curve by this strategy is the choice of the number and placement of (i) the discontinuity points between splines, and (ii) the knots within each individual spline. This can be posed as a model selection problem, and three objective criteria for choosing a "best" fitting curve model are discussed and compared. The criteria are constructed using different model selection principles: Akaike's information criterion (AIC; e.g., see Burnham and Anderson (1998)), generalized cross–validation (GCV; e.g., see Wahba (1990)) and the minimum description length principle (MDL; e.g., see Rissanen (1989) and Lee (2001)). Finally, the procedure is made robust by using a Gaussian mixture approach to model outliers.

Finding the best fitting curve defined by any of the three model selection criteria often involves solving a hard, large scale minimization problem. We use genetic algorithms for solving such problems. Simulation results suggest that the use of genetic algorithms is very promising in the present context.

1.1 Previous Work

Many regression spline based smoothing procedures have been proposed in the literature. These include Friedman and Silverman (1989), Friedman (1991), Smith and Kohn (1996), Koo (1997), Denison, Mallick and Smith (1998) and Lee (2000). In particular the procedures of Koo (1997) and Denison, Mallick and Smith (1998) are also capable of preserving discontinuity points in the regression functions. The common strategy of these authors has been to handle discontinuity points by introducing additional interior knots so that additional "intercepts" can be added between adjacent segments of a spline. However, none of these procedures has considered the issue of simultaneously handling discontinuities and outliers.

Another strategy for preserving discontinuity points is to first apply three different smoothers, left, right, and central, to smooth the data, then compare the three resulting smoothed curves; see McDonald and Owen (1986) and Hall and Titterington (1992). When estimating the function value at location x, a central smoother uses information from both sides of x, while a left or right smoother only uses information from the left or right of x, respectively.

We also remark that a problem that is related to the present context is change–point analysis; an excellent reference list is provided by Wang (1995). However, as noted in

Koo (1997), since change–point analysis and the smoothing of discontinuous functions have different aims, we do not discuss the problem of change–point detection further.

Lastly we highlight the contributions beyond those in Lee (2000). The current work (i) investigates, in addition to MDL, the use of AIC and GCV for selecting a "best" fitting curve; (ii) handles discontinuity points; (iii) considers robust smoothing; and (iv) uses a different (and better) optimization method, genetic algorithms, for obtaining "best" fitting curves.

The rest of this article is organized as follows. Section 2 presents the curve model and poses the problem of smoothing discontinuous curves as a model selection problem. Section 3 discusses three different solutions to this model selection problem, and in Section 4 a genetic algorithm is developed to numerically compute these solutions. Section 5 shows how to make the proposed smoothing procedures robust to outliers using a Gaussian mixture approach. Section 6 reports simulation results. Section 7 offers concluding remarks and technical details are deferred to the appendices. Finally, additional simulation results and further details regarding the implementation of the genetic algorithm are given in the separate document Lee (2002), obtainable over the internet.

2 Curve Model: Disconnected Regression Splines

Suppose n pairs of noisy measurements (x_i, y_i) are observed, with

$$y_i = f(x_i) + e_i, \quad x_1 < \ldots < x_n, \quad e_i \sim \text{ independent } N(0, \sigma^2), \quad i = 1, \ldots, n.$$

The aim is to estimate f. It is anticipated that f may possess a few discontinuity points but is otherwise smooth. In Section 5 the assumption of Gaussian e_i's will be relaxed so as to allow outliers.

Suppose it is known that there are $B-1$ discontinuity points in f and that these discontinuity points are located at $b_1, b_2, \ldots, b_{B-1}$. For convenience let $b_0 = x_1$ and $b_B = x_n$, and assume $b_0 < b_1 < \ldots < b_B$. Then one way to estimate f is to fit a separate cubic regression spline to each of the B disjoint segments $[b_{j-1}, b_j)$, $j = 1, \ldots, B$. If I_E is the indicator function for the event E, then such a disconnected regression spline model for f can be expressed as

$$f(x) = f_1(x) I_{\{b_0 \le x < b_1\}} + f_2(x) I_{\{b_1 \le x < b_2\}} + \ldots + f_B(x) I_{\{b_{B-1} \le x \le b_B\}}, \tag{1}$$

where each of the f_j's is a cubic regression spline having m_j knots located at

$k_{j1}, k_{j2}, \ldots, k_{jm_j}$. Furthermore,

$$f_j(x) = \alpha_{j0} + \alpha_{j1} x + \alpha_{j2} x^2 + \alpha_{j3} x^3 + \sum_{r=1}^{m_j} \beta_{jr} (x - k_{jr})_+^3, \quad b_{j-1} \le x < b_j, \quad j = 1, \ldots, B, \tag{2}$$

where $\boldsymbol{\alpha}_j = \{\alpha_{j0}, \ldots, \alpha_{j3}\}$ and $\boldsymbol{\beta}_j = \{\beta_{j1}, \ldots, \beta_{jm_j}\}$ are model parameters, $j = 1, \ldots, B$, with $(a)_+ = \max(0, a)$. To simplify notation, let $\boldsymbol{b} = \{b_1, \ldots, b_{B-1}\}$, $\boldsymbol{m} = \{m_1, \ldots, m_B\}$ and $\boldsymbol{k}_j = \{k_{j1}, \ldots, k_{jm_j}\}$ for $j = 1, \ldots, B$. For computational convenience it is assumed that $\{b_j, k_{jr}; \text{for all } j, r\}$ is a subset of $\{x_1, \ldots, x_n\}$ and that $b_0 < k_{11} < \ldots < k_{1m_1} < b_1 < \ldots < b_{j-1} < k_{j1} < \ldots < k_{jm_j} < b_j < \ldots < b_B$. Of course, in most situations the number and locations of the discontinuity points and knots are not known and need to be estimated.

If f is modelled by (1) and (2), then an estimate $\hat{f}$ of f can be obtained by first estimating $\boldsymbol{\theta} = \left\{B, \boldsymbol{b}, \boldsymbol{m}, \{\boldsymbol{k}_j, \boldsymbol{\alpha}_j, \boldsymbol{\beta}_j\}_{j=1}^{B}\right\}$ and then plugging the resulting estimate $\hat{\boldsymbol{\theta}} = \left\{\hat{B}, \hat{\boldsymbol{b}}, \hat{\boldsymbol{m}}, \{\hat{\boldsymbol{k}}_j, \hat{\boldsymbol{\alpha}}_j, \hat{\boldsymbol{\beta}}_j\}_{j=1}^{\hat{B}}\right\}$ into (1) and (2). Hence using the disconnected regression spline approach, our original curve estimation problem can be posed as a model selection problem with each candidate model specified by a $\hat{\boldsymbol{\theta}}$. The goal, then, is to select a "best" $\hat{\boldsymbol{\theta}}$. Notice that different $\hat{\boldsymbol{\theta}}$'s may have different dimensions, but once $\hat{B}$, $\hat{\boldsymbol{b}}$, $\hat{\boldsymbol{m}}$ and $\{\hat{\boldsymbol{k}}_j\}_{j=1}^{\hat{B}}$ are specified, natural maximum likelihood estimates of $\{\hat{\boldsymbol{\alpha}}_j, \hat{\boldsymbol{\beta}}_j\}_{j=1}^{\hat{B}}$ can be computed by least-squares regression.

3 Three Model Selection Methods

This section presents three different methods for selecting a "best" fitting model $\hat{\boldsymbol{\theta}}$: the MDL principle, GCV and AIC.

3.1 Minimum Description Length Principle

The MDL principle *defines* the best fitting model as the one that produces the shortest code length of the data; see Rissanen (1989) and references given therein. In this context the code length of an object is treated as the amount of memory space that is required to store the object. Of course comparing code lengths is neither the only nor the best approach for defining a best fitting model, but it is a sensible one. This is because a common feature of a good encoding (or compression) scheme and a good statistical model is the ability to capture the regularities, or patterns, present in the data. An MDL principle tutorial targeted at a statistical audience can be found in Lee (2001).

When applying the MDL principle, it is common to split the code length for a set of data into two parts: (i) a fitted model plus (ii) the data "conditioned on" the fitted model; i.e., the residuals. For the present case, the data are $\boldsymbol{y} = (y_1, \ldots, y_n)^T$ and a fitted model can be simply specified by $\hat{\boldsymbol{\theta}}$. We write the residual vector as $\hat{\boldsymbol{e}} = (\hat{e}_1, \ldots, \hat{e}_n)^T$, where $\hat{e}_i = y_i - \hat{f}(x_i)$ for $i = 1, \ldots, n$. In other words, one splits $\boldsymbol{y}$ into $\hat{\boldsymbol{\theta}}$ plus $\hat{\boldsymbol{e}}$.

If $L(z)$ denotes the code length of object z, we have $L(\boldsymbol{y}) = L(\hat{\boldsymbol{\theta}}) + L(\hat{\boldsymbol{e}}|\hat{\boldsymbol{\theta}})$. Note that in this expression it is stressed that $\hat{\boldsymbol{e}}$ is conditional on $\hat{\boldsymbol{\theta}}$. Now the task is to find an expression for $L(\boldsymbol{y})$ so that the best MDL $\hat{\boldsymbol{\theta}}$ can be defined and obtained as its minimizer. It is shown in Appendix A that $L(\boldsymbol{y})$ can be well approximated by

$$\mathrm{MDL}(\hat{f}) = L(\boldsymbol{y}) = \log \hat{B} + \sum_{j=1}^{\hat{B}} \log \hat{m}_j + \sum_{j=1}^{\hat{B}} \left(3 + \frac{\hat{m}_j}{2}\right) \log \hat{l}_j + \frac{n}{2} \log \left[\frac{1}{n} \sum_{i=1}^{n} \left\{y_i - \hat{f}(x_i)\right\}^2\right], \quad (3)$$

where $\hat{l}_j$ is the number of x_i's in the jth fitted disconnected spline $\hat{f}_j(x)$. We propose to select the minimizer of $\mathrm{MDL}(\hat{f})$ as our MDL–based curve estimate.

3.2 Modified GCV of Friedman and Silverman

One interpretation of GCV is that it attempts to construct an asymptotically unbiased estimator for the risk $E[\int\{f(x) - \hat{f}(x)\}^2 dx]$ and then selects the best fitting model as the one that minimizes this risk estimator (e.g., see Wahba (1990, Chapter 4)). In Friedman and Silverman (1989), the following GCV criterion is applied to choose a best curve estimate for an adaptive knot–locating piecewise linear fitting procedure:

$$\mathrm{GCV}(\hat{f}) = \frac{1}{n} \sum_{i=1}^{n} \left\{y_i - \hat{f}(x_i)\right\}^2 \Big/ \left\{1 - \frac{d(K)}{n}\right\}^2. \quad (4)$$

Here $d(K)$ is an increasing function of the number of the knots K. In order to penalize the additional flexibility inherited by the free choice of knot locations, the authors suggested using $d(K) = 3K + 1$ instead of the conventional GCV choice $d(K) = K + 1$. That is, they proposed taking the penalty for each additional *free parameter* as 3 degrees of freedom instead of 1.

We apply this "3 degrees of freedom" rule to construct a GCV–based selection criterion for our fitting procedure. In addition to the knots (there are $\sum \hat{m}_j$ of them), we also count all those α's in the second to last piecewise disconnected splines as *free parameters* (there are $4(\hat{B}-1)$ of them). Thus we choose the $\hat{f}$ (or $\hat{\boldsymbol{\theta}}$) to minimize the GCV($\hat{f}$) with $d(K)$ given by $3\{4(\hat{B}-1)+\sum \hat{m}_j\}+1$.

3.3 Modified AIC of Koo

With AIC the best fitting model is chosen as the one that minimizes an estimator of the Kullback–Leibler (KL) distance measure between a fitted model and the "true" model (e.g., see Burnham and Anderson (1998)). If p is the number of parameters that need to be estimated in a fitted model, then under some mild regularity conditions one can show that such a KL distance estimator is $-2 \times$ "maximized log likelihood" $+ 2p$. For our Gaussian–noise curve estimation problem this distance estimator amounts to

$$\text{AIC}(\hat{f}) = \left[n \log \sum_{i=1}^{n} \left\{ y_i - \hat{f}(x_i) \right\}^2 + \gamma p \right]_{\gamma=2}. \tag{5}$$

Koo (1997) studied the use of AIC($\hat{f}$) as a selection criterion for the fitting of linear regression splines. He indicated that AIC($\hat{f}$) with $\gamma = 2$ is *not* a good selection criterion; based on empirical experience, he suggested using $\gamma = \log n$.

We follow Koo's advice to reach our AIC–based model selection criterion: select the $\hat{f}$ that minimizes AIC($\hat{f}$) with $\gamma = \log n$. Note that for our regression model, $p = 4\hat{B} + \sum \hat{m}_j$.

3.4 Which Criterion to Use?

Of the three criteria discussed above, MDL($\hat{f}$) is an attractive choice for the following reasons. First, the "3 degrees of freedom" rule of GCV($\hat{f}$) and the choice of $\gamma = \log n$ in AIC($\hat{f}$) seem somewhat arbitrary. Second, MDL($\hat{f}$) can be extended in a straightforward and consistent manner to handle the presence of outliers, see Section 5. Furthermore, simulation results to be reported in Section 6 suggest that the three criteria perform roughly the same.

4 Optimization by Genetic Algorithms

When the number of data points is large, finding the best estimate according to any of the above criteria involves solving a hard, large scale minimization problem. We recommend genetic algorithms for this; for a general introduction, see Davis (1991) for example.

4.1 General Description

Genetic algorithms for solving optimization problems can be briefly described as follows. An initial set, or population, of possible solutions to an optimization problem is obtained and represented in vector form. These vectors are often called *chromosomes* and are free to "evolve" in the following way. Parent chromosomes are randomly chosen from the initial population and chromosomes having lower (higher) values of the objective criterion to be minimized (maximized) would have a higher chance of being chosen; offspring are produced by applying a *crossover* or a *mutation* operation to the chosen parents; once a sufficient number of such second generation offspring are produced, third generation offspring are further produced from these second generation offspring; this process continues for a number of generations. If one believes in Darwin's Natural Selection, the expectation is that objective criterion values of the offspring will gradually improve over generations and approach the optimal value.

In a crossover operation, one child chromosome is produced from "mixing" two parent chromosomes. The aim is to allow the possibility that the child receives different best parts from its parents. A typical "mixing" strategy is that every child gene location has an equal chance of receiving either the corresponding father gene or the corresponding mother gene. This crossover operation is the distinct feature that makes genetic algorithms different from other optimization methods. For possible variants of the crossover operation, consult Davis (1991).

In a mutation operation one child chromosome is produced from one parent chromosome. The child is essentially the same as its parent except for a small number of genes where randomness is introduced to alter the types of these genes. Such a mutation operation prevents the algorithm being trapped in local optima.

4.2 Chromosome Representation

The performance of a genetic algorithm certainly depends on how a possible solution is represented as a chromosome. In traditional applications, solutions are often represented as binary vectors; that is, chromosomes with two types of genes. However we use a "three–gene–types" representation.

First recall that for our curve fitting problem, a possible solution $\hat{\boldsymbol{\theta}}$ can be uniquely specified by $\hat{B}$, $\hat{\boldsymbol{b}}$, $\hat{\boldsymbol{m}}$ and $\{\hat{\boldsymbol{k}}_j\}_{j=1}^{\hat{B}}$. Once these are specified, the corresponding maximum likelihood estimates for $\{\hat{\boldsymbol{\alpha}}_j, \hat{\boldsymbol{\beta}}_j\}_{j=1}^{\hat{B}}$ can be uniquely calculated. Thus for our problem a chromosome only needs to carry information about $\hat{B}$, $\hat{\boldsymbol{b}}$, $\hat{\boldsymbol{m}}$ and $\{\hat{\boldsymbol{k}}_j\}_{j=1}^{\hat{B}}$. A simple example will be used to illustrate this representation scheme. Suppose $n = 20$, $\hat{B} = 2$, $\hat{\boldsymbol{b}} = \{12\}$, $\hat{\boldsymbol{m}} = \{2, 1\}$, $\hat{\boldsymbol{k}} = \{\hat{\boldsymbol{k}}_1, \hat{\boldsymbol{k}}_2\}$, $\hat{\boldsymbol{k}}_1 = \{6, 10\}$ and $\hat{\boldsymbol{k}}_2 = \{17\}$. That is, the curve estimate is composed of two disconnected splines separated at x_{12}, and there are two and one knots in the first and the second spline, respectively. These knots are located at x_6, x_{10} and x_{17}. If we use "$\wr$" to denote a discontinuity gene, "$\diamond$"to denote a knot gene and "$\cdot$" to denote a normal gene, then the chromosome for this example is composed of $n = 20$ genes arranged as: $\cdot\cdot\cdot\cdot\cdot\diamond\cdot\cdot\cdot\diamond\cdot\wr\cdot\cdot\cdot\cdot\diamond\cdot\cdot\cdot$.

Empirical evidence suggests that this representation scheme is extremely effective for the purpose of using genetic algorithms to minimize any of the three selection criteria discussed previously. It is most likely due to the fact that the location information of the discontinuity points and the knots of a $\hat{\boldsymbol{\theta}}$ are explicitly represented. Further details regarding the implementation of our genetic algorithm are given in Section B of the supporting document Lee (2002).

4.3 Previous Use of Genetic Algorithms for Curve Fitting

After the completion of this work in 1999, the author was made aware of the recent work of Pittman (1999) and Pittman and Murthy (2000), in which genetic algorithms are also applied to the problem of regression spline fitting. However, the works are quite different. A major difference is that, in the current work, the basic units for constructing a chromosome are the design points x's, while the previous authors used the knots k_{jr}'s as the basic units. Consequently the definitions for the crossover and mutation operations are necessary different. An advantage of using the design points as the basic units is that it can be extended naturally to handle outliers; see Section 5.

In Liang and Wong (2000) a new Markov chain Monte Carlo algorithm, termed Evolutionary Monte Carlo (EMC), is proposed. This algorithm incorporates many attractive features of both genetic algorithms and simulated annealing. In that paper the authors also demonstrate how the EMC algorithm can be applied to change point detection problems. A planned research is to apply this EMC algorithm to the present setting and see if it improves the current genetic algorithm.

5 Robust Fitting

This section extends the proposed MDL–based smoothing procedure by using Gaussian mixtures to model outliers. It assumes that the noise e_i's are independent with density function

$$(1-w)p(e;0,\sigma^2)+wp(e;0,c^2\sigma^2), \quad 0 \le w < 0.5, \quad c > 1, \tag{6}$$

where $p(\cdot;\mu,\sigma^2)$ is the Gaussian density with mean μ and variance σ^2. The percentage of outliers w is assumed to be small and will be estimated, while the variance inflation factor c is fixed *a priori*; see below for more comments on this. Model (6) is often known as the inflated–variance model for outliers; see Titterington, Smith and Makov (1985, Ch. 4) and references given therein.

Suppose that $\hat{B}$, $\hat{\boldsymbol{b}}$, $\hat{\boldsymbol{m}}$ and $\{\hat{\boldsymbol{k}}_j\}_{j=1}^{\hat{B}}$ are specified and that an initial guess of which y_i's are contaminated by outliers is made. Such initial guesses are made randomly by the genetic algorithms to form the first generation of chromosomes. Now, for a particular guess, if there are $\hat{n}_{\text{OUT}}$ of these "suspected outliers", then w can be estimated by $\hat{w}=\hat{n}_{\text{OUT}}/n$, and estimates of α_{ir}'s and β_{ir}'s (hence $\hat{f}$ and $\hat{\boldsymbol{e}}$) can be obtained by performing weighted regressions, with small weights (e.g., c^{-1}) attached to those "outlying y_i's". Recall that for the MDL principle the goal is to find an expression for $L(\boldsymbol{y})$ and use its minimizer as the final estimate. It is outlined in Appendix B that, when using (6) to model outliers, the corresponding $L(\boldsymbol{y})$ can be approximated by

$$\begin{aligned}\mathrm{RMDL}(\hat{f}) &= \log \hat{B} + \sum_{j=1}^{\hat{B}} \log \hat{m}_j + \sum_{j=1}^{\hat{B}} \left(3 + \frac{\hat{m}_j}{2}\right) \log \hat{l}_j + \\ &\quad \log \left[\left\{ \frac{1-\hat{w}}{\sqrt{2\pi\hat{\sigma}^2}} \exp \frac{-1}{2\hat{\sigma}^2} + \frac{\hat{w}}{\sqrt{2\pi c^2 \hat{\sigma}^2}} \exp \frac{-1}{2c^2\hat{\sigma}^2} \right\} \sum_{i=1}^{n} \left\{ y_i - \hat{f}(x_i) \right\}^2 \right]. \qquad (7)\end{aligned}$$

Here $\hat{\sigma}^2 = \sum\{y_i - \hat{f}(x_i)\}^2/(n - \hat{n}_{\mathrm{OUT}})$, where the sum does not include outlying y_i's. Our robust version of the MDL–based smoothing procedure is to define the best estimate as the minimizer of $\mathrm{RMDL}(\hat{f})$. Note that when $\hat{w} = 0$, $\mathrm{RMDL}(\hat{f})$ reduces to $\mathrm{MDL}(\hat{f})$ up to an additive constant.

For the minimization of $\mathrm{RMDL}(\hat{f})$, in addition to $\hat{\boldsymbol{\theta}}$, the number and locations of the suspected outliers are also arguments. Such a minimization problem appears difficult, but the genetic algorithm approach discussed in the previous section can be extended to incorporate the outliers: simply introduce an additional type of genes, outlier genes, into the algorithm.

We have the following remarks about the variance inflation factor c. In theory one does not have to fix it *a priori*, as one can estimate $\sigma^2_{\mathrm{OUT}} = c^2\sigma^2$ by a normalized outlier residual sum of squares. However, as the number of outliers is usually small, a stable estimate of σ^2_{OUT} is hard to obtain. Such instability would almost certainly be carried over to $\mathrm{RMDL}(\hat{f})$, and hence it was decided to choose c *a priori*. As the main focus of this section is to perform robust fitting and not outlier detection, the penalty for misclassifying a y_i as an outlier is not usually severe. Since a small value of c would allow more y_i's to be classified as outliers, as long as the choice of c is not too small, one would not expect the quality of $\hat{f}$ to be severely affected. In our work we have chosen $c = 7$.

6 Simulation Results

This section reports results of four numerical experiments. These experiments were designed for assessing and comparing the practical performances of various aspects of the proposed approach with some popular approaches found in the literature.

6.1 Smooth Curves

In the first experiment all three test functions are smooth. They have been used by other authors and are listed as Test Functions 1 to 3 in Table 1. Altogether eight smoothing procedures were tested. For easy referencing, we shall label these procedures in the sans serif font. Upper case letters are reserved for author initials, otherwise lower case letters are used. The eight procedures are:

1. mdl: the proposed approach with $\mathrm{MDL}(\hat{f})$ as the target,
2. aic: the proposed approach with $\mathrm{AIC}(\hat{f})$ as the target,
3. gcv: the proposed approach with $\mathrm{GCV}(\hat{f})$ as the target,
4. rmdl: the proposed robust procedure with $\mathrm{RMDL}(\hat{f})$ as the target,
5. DMS: the Bayesian curve fitting procedure of Denison, Mallick and Smith (1998) with $\lambda = 1$ — codes downloaded from `http://www.ma.ic.ac.uk/~dgtd`,
6. SK: the Bayesian regression spline smoothing procedure of Smith and Kohn (1996) with modal estimate — codes downloaded from `http://www.agsm.unsw.edu.au/~mikes`,

7. RSW: local linear regression with the direct bandwidth plug–in choice of Ruppert, Sheather and Wand (1995) — codes downloaded from `http://biosun1.harvard.edu/~mwand` (see also Wand (1998)),

8. HST: nearest neighbour local polynomial estimator LOESS (Cleveland and Devlin (1988)) with the $\mathrm{AIC_C}$ choice of smoothing parameter proposed by Hurvich, Simonoff and Tsai (1998) — codes downloaded from `http://www.stern.nyu.edu/~jsimonof`.

Unless stated otherwise, for procedures DMS, SK, RSW and HST, default values supplied by the downloaded codes were used for all the parameters and/or priors that are required to be pre–selected.

Test Function	Formula
1	$(4x-2)+2\exp\{-16(4x-2)^2\},\quad 0\le x\le 1$
2	$\sin^3(2\pi x^3),\quad 0\le x\le 1$
3	$\sin(5\pi x),\quad 0\le x\le 1$
4	$2x - I_{\{0.5\le x\}},\quad 0\le x\le 1$
5	$4x^2(3-4x)I_{\{x\le 0.5\}}+\{\frac{4}{3}x(4x^2-10x+7)-\frac{3}{2}\}I_{\{0.5<x\le 0.75\}}+$
	$\frac{16}{3}x(x-1)^2 I_{\{0.75<x\}},\quad 0\le x\le 1$
6	$2-2\lvert x-0.26\rvert^{1/5}I_{\{x\le 0.26\}}-2\lvert x-0.26\rvert^{3/5}I_{\{x>0.26\}}+I_{\{x\ge 0.78\}},\quad 0\le x\le 1$

Table 1: Test functions.

Signal–to–noise ratios (snrs) are defined as $\mathrm{snr}=\{\mathrm{var}(f)/\sigma^2\}^{\frac{1}{2}}$, as in Donoho and Johnstone (1995). Three snr levels were used: 2, 4 and 6. For each combination of test function and snr, 50 sets of noisy observations were simulated with $U[0,1]$ as the design density for x_i. Only one n was used: $n = 200$ (n and snr are interchangeable when both of them are not too small). Only results corresponding to the case when snr=4 are reported here, results from the other two snrs are similar.

For each simulated data set, the eight smoothing procedures listed above were applied to estimate the test function. The numerical measure used to evaluate the quality of an estimated curve was $\mathrm{MSE}(\hat{f}) = \sum\{f(x_i) - \hat{f}(x_i)\}^2$. For each test function, boxplots of the log of the $\mathrm{MSE}(\hat{f})$ values of all $\hat{f}$ are plotted in Fig. 1.

To visually evaluate and compare the performances of the eight smoothing procedures, the following was done. For Test Function 2, the 50 $\hat{f}$'s obtained by mdl were ranked according to their values of $\mathrm{MSE}(\hat{f})$. The 25th best $\hat{f}$, together with the corresponding simulated noisy data, are plotted in Fig. 2. Curve estimates obtained by applying the remaining seven smoothing procedures to this same simulated noisy data set are also plotted in Fig. 2. The same procedure was then repeated for the remaining two test functions, and the results are displayed in a similar manner in Figs. 1 and 2 of Lee (2002).

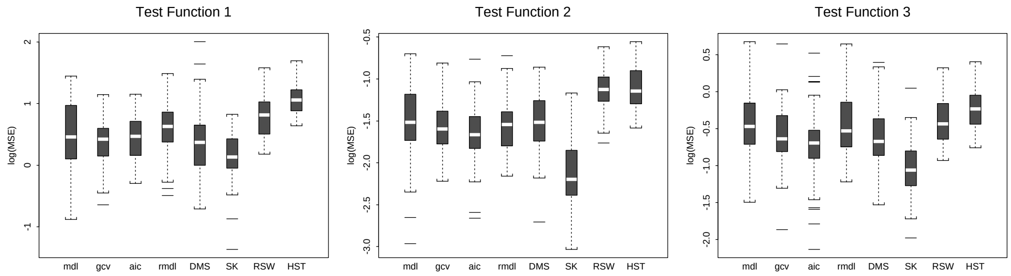

Figure 1: Boxplots of $\log\{\mathrm{MSE}(\hat{f})\}$ values of various smoothing procedures for Test Functions 1 to 3.

Some empirical conclusions can be drawn. When the curves are smooth, SK seems to be the best procedure. Also, mdl, gcv, aic and rmdl performed roughly the same, and are at least as good as DMS, RSW and HST.

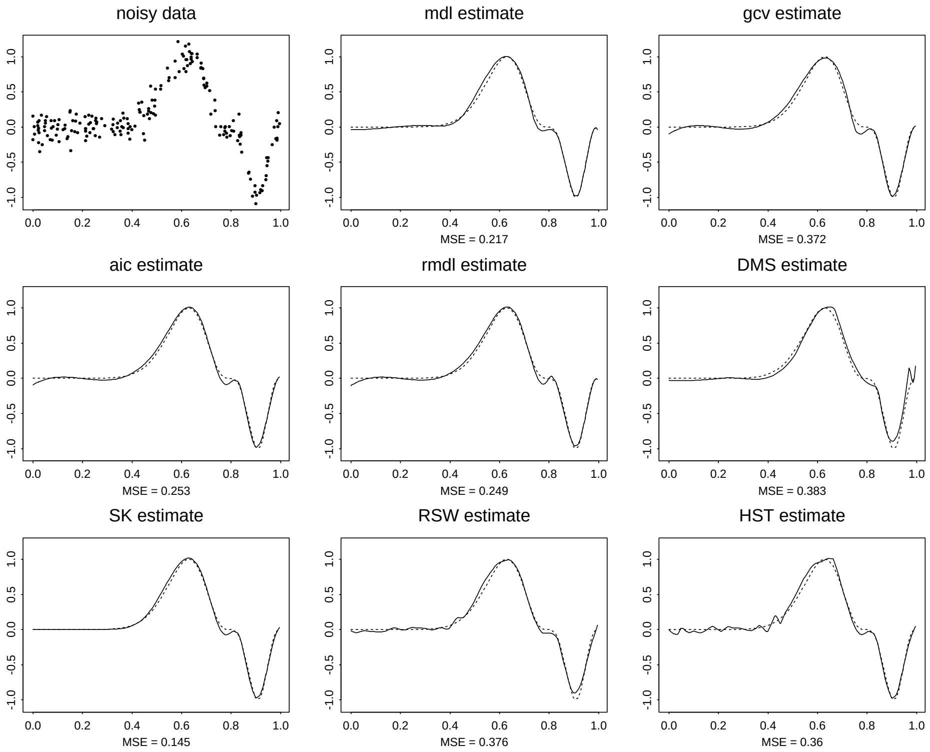

Figure 2: Test Function 2. Solid lines: estimates; broken lines: true curves.

The following numerical values are provided for the purposes of speed comparison. These values are based on a Sun Ultra–60 machine. The computational times required for the non–robust procedures mdl, aic and gcv were very similar and dependent on the structure of the target curve: they ranged from 6 to 30 seconds. The robust procedure rmdl, for most cases, was about 3 to 5 seconds slower than its non–robust counterpart. DMS typically took 2 to 3 seconds, while SK, RSW and HST took less than 1 second. Obviously the four proposed procedures are more computationally expensive, but acceptably so in view of their good performance.

6.2 Discontinuous Curves

The above experiment was repeated with three discontinuous test functions. These test functions have been used by other authors, and are listed in Table 1 as Test Functions 4 to 6. Observe that SK, RSW and HST are *not* expected to perform well, as these procedures were *not* designed for recovering non–smooth curves.

Boxplots analogous to those in Fig. 1 are given in Fig. 3, while plots of various function estimates for Test Function 5 are given in Fig. 4. Similar function estimate plots for Test Functions 4 and 6 are provided respectively in Figs. 3 and 4 of Lee (2002). From these plots one concludes that, for the cases of discontinuous curves, mdl, gcv, aic and rmdl give similar, and overall best, results.

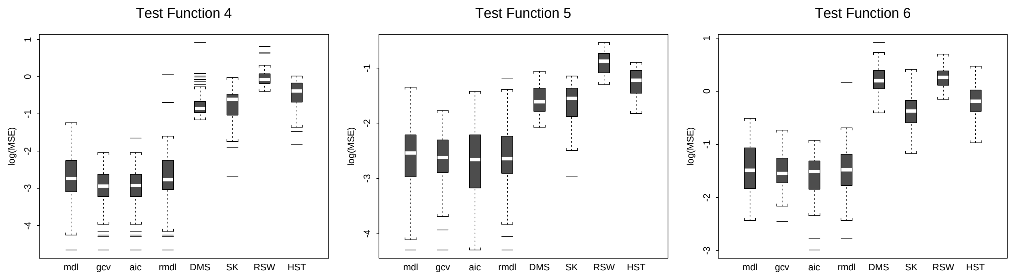

Figure 3: Boxplots of $\log\{\mathrm{MSE}(\hat{f})\}$ values of various smoothing procedures for Test Functions 4 to 6.

6.3 Highly Spatial Inhomogeneous Curves

The experiment described in Section 6.1 was repeated again but with the following changes: the test functions were the four highly spatial inhomogeneous curves advocated by Donoho and Johnstone (1995); the SureShrink procedure (sure) of Donoho and Johnstone (1995) and

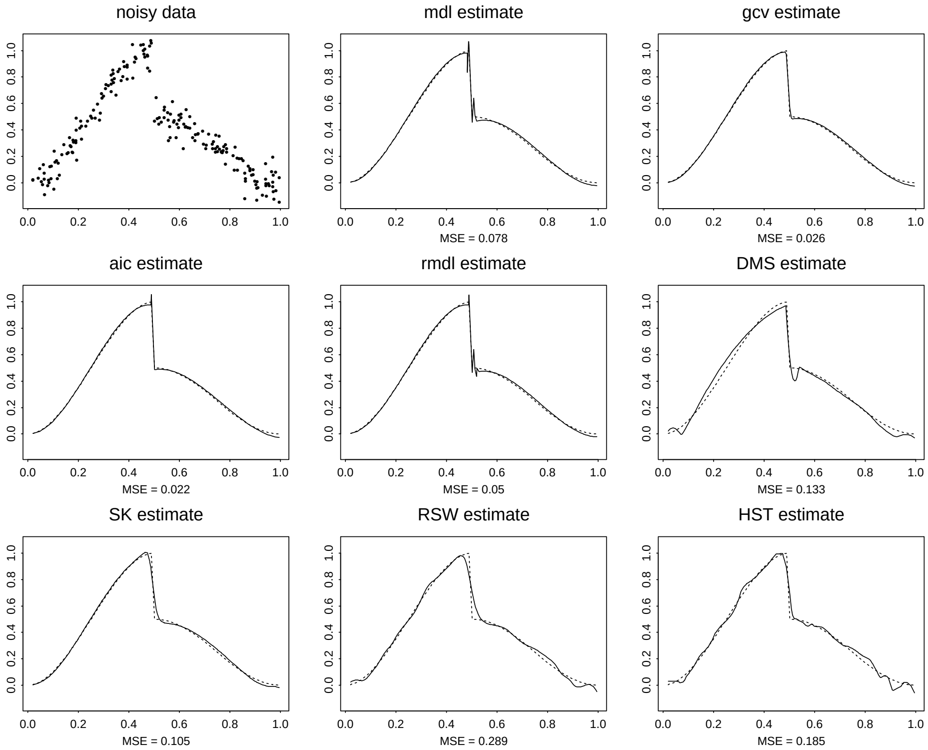

Figure 4: Test Function 5. Solid lines: estimates; broken lines: true curves.

the BayesShrink procedure (Bayes) of Abramovich, Sapatinas and Silverman (1998) were added; the parameter λ in DMS was changed from 1 to 5; the snrs used were 5, 7 and 9, but only those results associated with snr=7 are reported; the design points x_i's were regularly–spaced and $n = 512$. Here all wavelet computations are performed using the `wavethresh` package of Nason and Silverman (1994). *S–Plus* codes for sure and Bayes were adopted from Luo and Wahba (1997) and provided by Dr. Fanis Sapatinas respectively. As in the previous subsection, it is expected that SK, RSW and HST would not perform well.

Plots analogous to those in Section 6.1 are provided; see Figs. 5 and 6, and Figs. 5 to 7 of Lee (2002). These plots suggest that mdl has the smallest MSE, and that the performance of rmdl depends heavily on the structures of the underlying curves. The latter is not surprising, as the highly oscillating structures in *Doppler* and *Bumps* could be easily mistaken as outliers by rmdl. A more elaborate discussion on this issue is in Section 7.

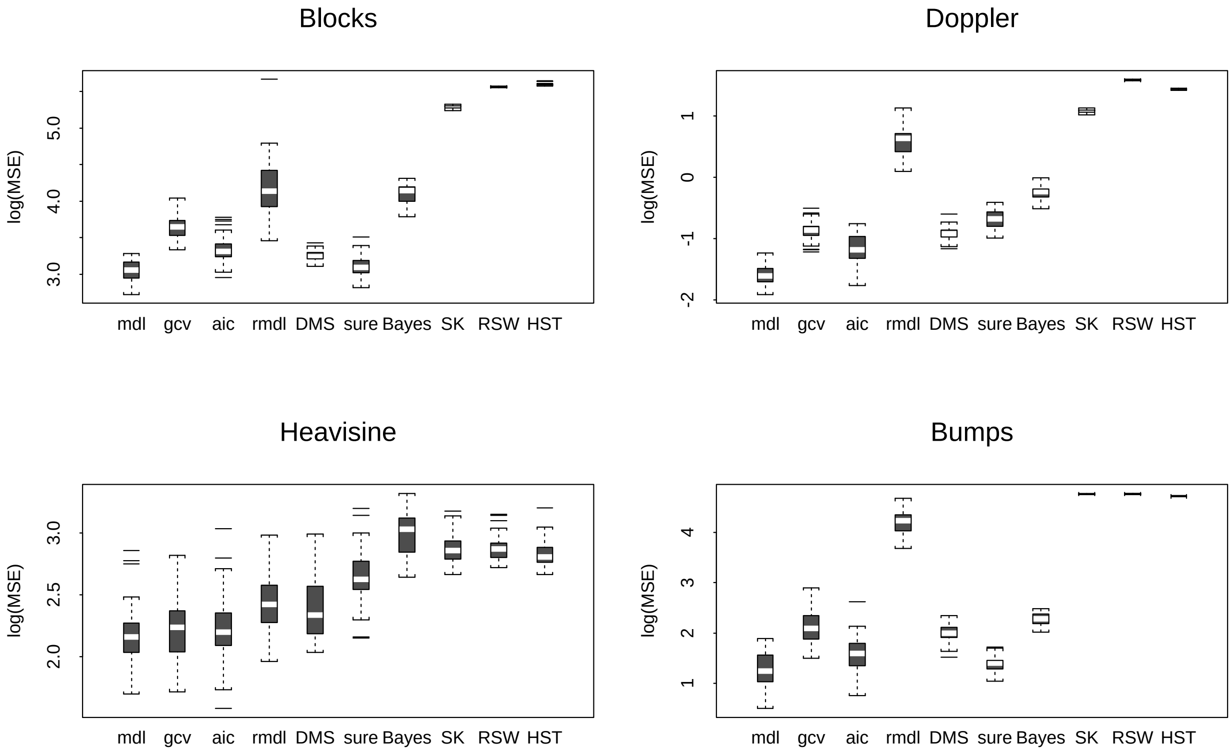

Figure 5: Boxplots of $\log\{\mathrm{MSE}(\hat{f})\}$ values of various smoothing procedures.

6.4 Data with Outliers

The main focus of this last experiment was to assess the performance of rmdl when outliers are present. One hundred data sets were generated from Test Function 1, with snr=4, $n = 200$ and $U[0,1]$ as the design density. Then for each simulated data set n_{OUT} outliers were introduced, where n_{OUT} is discrete uniform on $[0,\ldots,5]$. The size of each outlier was generated from $(5+U[0,15])\sigma$, with equal probability of being positive or negative. Finally four smoothing procedures, mdl, rmdl, SK and the robust version rSK of SK, were applied to smooth the data sets. The whole process was repeated with Test Functions 2 to 6.

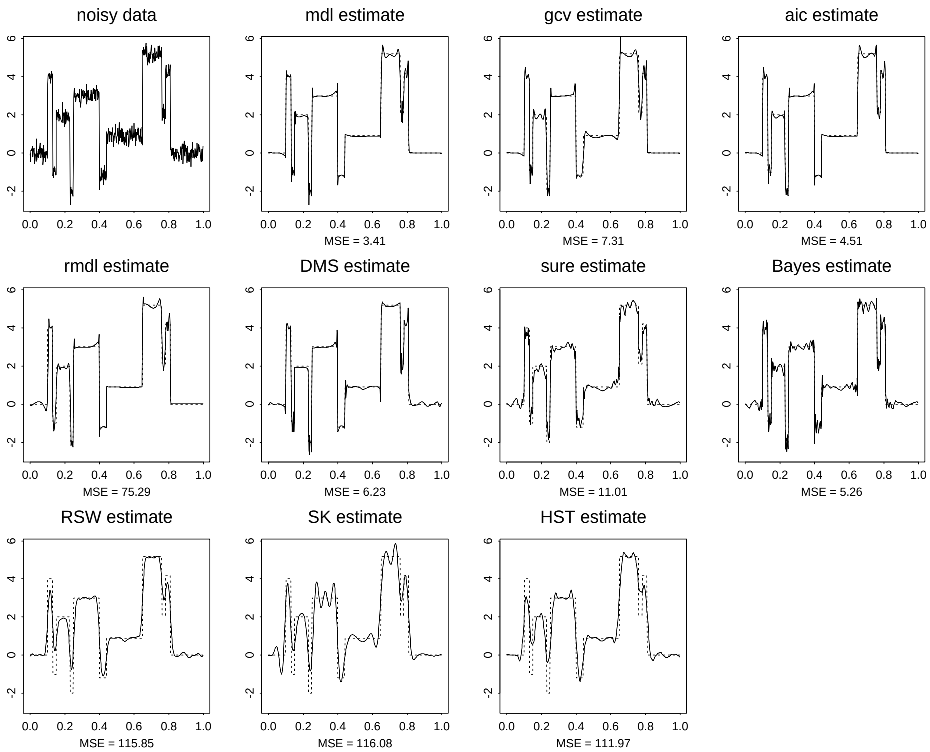

Figure 6: Test Functions *Blocks*. Solid lines: estimates; broken lines: true curves.

Boxplots for log(MSE) are given in Fig. 7, and noisy data and various curve estimates corresponding to the "50th smallest MSE rmdl estimates" are given in Fig. 8, and Figs. 8 and 9 of Lee (2002). From these plots one can conclude that, when the data are contaminated by outliers, rmdl outperforms mdl. Also, when the target curves are smooth rSK is superior to rmdl, and the converse is true if the target curves are non–smooth.

7 Concluding Remarks

7.1 Summary

In this article three automatic smoothing procedures, mdl, gcv and aic, for recovering discontinuous curves are proposed. Simulation results show that, when the target function is

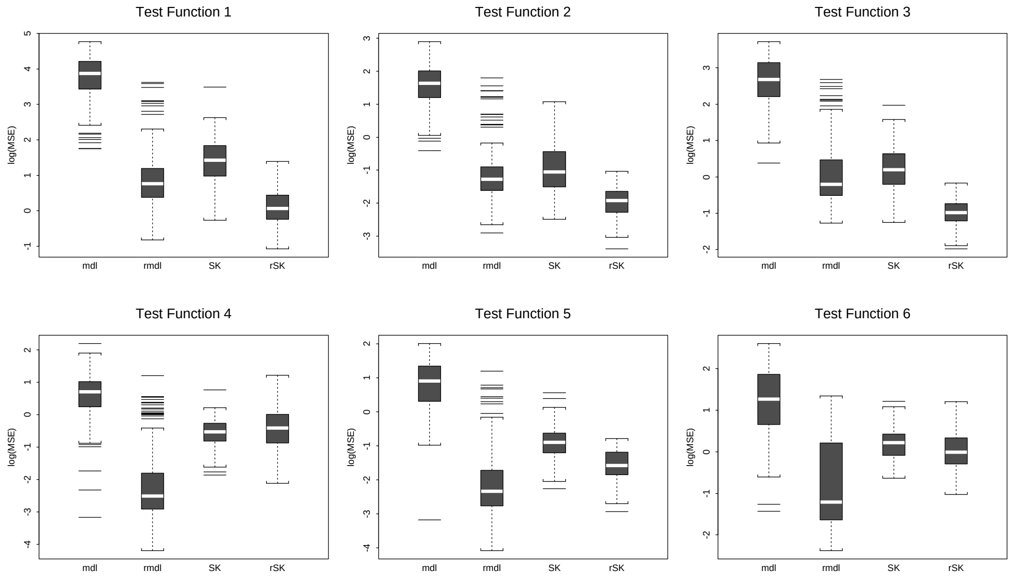

Figure 7: Boxplots of $\log\{\text{MSE}(\hat{f})\}$ values of various smoothing procedures when data were contaminated by outliers.

non–smooth, these proposed procedures are superior to many existing smoothing methods, including Bayesian approaches, local polynomial smoothing and wavelet techniques. The procedure mdl is further made robust by using a Gaussian mixture approach to model outliers, and the resulting robust procedure, rmdl, performs very well, with or without outliers, for regression functions that do not contain many rapid–changing structures.

7.2 Use mdl or rmdl?

Given a noisy data set, should one use mdl, or rmdl? A definite answer is hard to give, as the presence of both outliers and rapid–changing structures in the regression function can produce nearly indistinguishable noisy observations (especially when the number of data points is small). However, given the simulation results, some guidelines can be offered. We first summarize the simulation results: when there are no outliers, mdl and rmdl gave

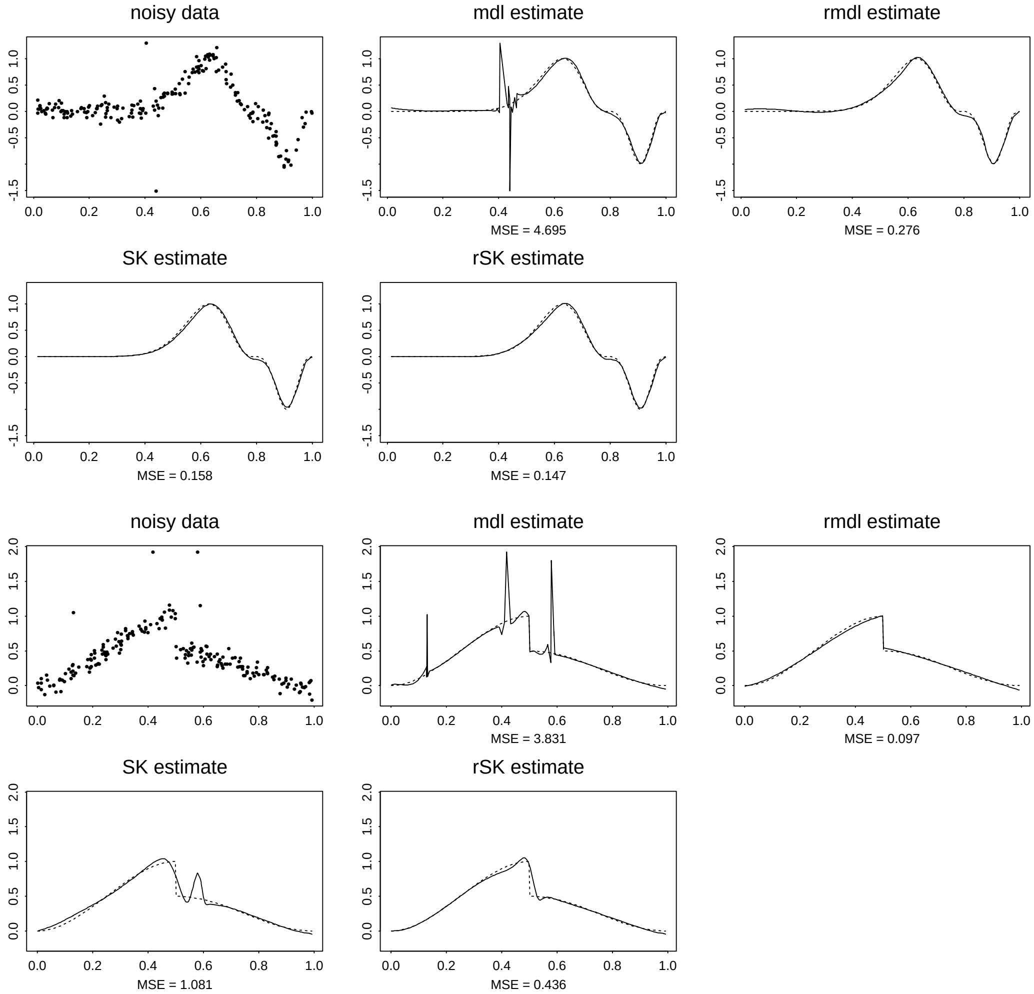

Figure 8: Various estimates of Test Functions 2 (top two rows) and 5 (bottom two rows) when data were contaminated by outliers. Solid lines: estimates; broken lines: true curves.

very similar results for functions that do not contain many rapid–changing structures; when there are no outliers, mdl gave better results then rmdl for functions that contain many rapid–changing structures; when there are outliers, rmdl gave better results then mdl for functions that do not contain many rapid–changing structures.

Therefore, if it is reasonable to assume that there are no outliers (regardless of the existence of any rapid–changing structures), use mdl. If it is reasonable to belief that the regression function does not contain rapid–changing structures (regardless of the presence of any outliers), use rmdl. Otherwise, one may want to apply both procedures to the same data set and visually compare the two estimated curves. It is very likely that two different estimated curves can reveal different hidden structures of the unknown regression function. This idea of inspecting more than one estimated curve is in the same spirit as the SiZer approach of Chaudhuri and Marron (1999) for performing (non–robust) nonparametric regression. In SiZer one first obtains many estimated curves by applying different amount of smoothing to the same data set, and then performs tests of significance for the existence of structures in the regression function.

Acknowledgement

The author would like to thank Mary Morrissey and Kenneth Wilder for several interesting discussions, and Xiao-Li Meng for his constant encouragement during the course of getting this work published. The author is also grateful to an associate editor and a referee for helpful comments.

A Derivation of MDL($\hat{f}$)

This appendix outlines the derivation of MDL$(\hat{f}) = L(\hat{\boldsymbol{y}})$. It follows, but also extends, a similar derivation in Lee (2000). Our first step is to decompose $L(\hat{\boldsymbol{\theta}})$ and obtain

$$\begin{aligned}
L(\hat{\boldsymbol{y}}) &= L(\hat{\boldsymbol{\theta}}) + L(\hat{\boldsymbol{e}}|\hat{\boldsymbol{\theta}}) \\
&= L\left(\hat{B}, \hat{\boldsymbol{b}}, \hat{\boldsymbol{m}}, \{\hat{\boldsymbol{k}}_j, \hat{\boldsymbol{\alpha}}_j, \hat{\boldsymbol{\beta}}_j\}_{j=1}^{\hat{B}}\right) + L(\hat{\boldsymbol{e}}|\hat{\boldsymbol{\theta}}) \\
&= L(\hat{B}) + L(\hat{\boldsymbol{b}}|\hat{B}) + L(\hat{\boldsymbol{m}}|\hat{B}, \hat{\boldsymbol{b}}) + L\left(\{\hat{\boldsymbol{k}}_j\}_{j=1}^{\hat{B}}|\hat{B}, \hat{\boldsymbol{b}}, \hat{\boldsymbol{m}}\right) +
\end{aligned}$$

$$L\left(\{\hat{\boldsymbol{\alpha}}_j, \hat{\boldsymbol{\beta}}_j\}_{j=1}^{\hat{B}} | \hat{B}, \hat{\boldsymbol{b}}, \hat{\boldsymbol{m}}, \{\hat{\boldsymbol{k}}_j\}_{j=1}^{\hat{B}}\right) + L(\hat{\boldsymbol{e}}|\hat{\boldsymbol{\theta}}). \tag{8}$$

Derivation of $L(\hat{B})$ and $L(\hat{\boldsymbol{b}}|\hat{B})$. Since the code length for encoding an integer N is approximately $\log_2 N$, $L(\hat{B}) = \log_2 \hat{B}$. Now as $\hat{\boldsymbol{b}} = \{\hat{b}_1, \ldots, \hat{b}_{\hat{B}-1}\}$ is restricted to be a subset of $\{x_1, \ldots, x_n\}$, $\hat{\boldsymbol{b}}$ can be specified by the indices of those x_i's where a discontinuity point is located. Such a set of indices can be compactly specified by their successive differences. To simplify notation, let $\hat{l}_j$ be the number of x_i's that satisfy $\hat{b}_{j-1} \le x_i < \hat{b}_j$, $j = 1, \ldots, \hat{B}$. That is, $\hat{l}_j$ is the jth successive "index difference". By noting that these $\hat{l}_j$'s are integers, we have

$$L(\hat{B}) + L(\hat{\boldsymbol{b}}|\hat{B}) = L(\hat{B}) + L(\hat{l}_1, \ldots, \hat{l}_{\hat{B}}|\hat{B}) = \log_2 \hat{B} + \sum_{j=1}^{\hat{B}} \log_2 \hat{l}_j. \tag{9}$$

Derivation of $L(\hat{\boldsymbol{m}}|\hat{B}, \hat{\boldsymbol{b}})$ and $L\left(\{\hat{\boldsymbol{k}}_j\}_{j=1}^{\hat{B}} | \hat{B}, \hat{\boldsymbol{b}}, \hat{\boldsymbol{m}}\right)$. Recall that the jth element $\hat{m}_j$ of $\hat{\boldsymbol{m}} = \{\hat{m}_1, \ldots, \hat{m}_{\hat{B}}\}$ is the estimated number of knots of the jth fitted regression spline, and that these $\hat{m}_j$ knots are located at $\hat{\boldsymbol{k}}_j = \{\hat{k}_{j1}, \ldots, \hat{k}_{j\hat{m}_j}\}$. Thus the encoding for each pair of $(\hat{m}_j, \hat{\boldsymbol{k}}_j)$ can be done in a similar fashion to the encoding of $(\hat{B}, \hat{\boldsymbol{b}})$ described above. Analogous to $\hat{l}_j$, define $\hat{d}_{jr}$ as the "index difference" between $\hat{k}_{j,r-1}$ and $\hat{k}_{jr}$. Using similar arguments as before, it can be shown that the code length for $\hat{m}_j$ and $\hat{\boldsymbol{k}}_j$ is $\log_2 \hat{m}_j + \log_2 \hat{d}_{j1} + \ldots + \log_2 \hat{d}_{j\hat{m}_j}$. Hence

$$L(\hat{\boldsymbol{m}}|\hat{B}, \hat{\boldsymbol{b}}) + L\left(\{\hat{\boldsymbol{k}}_j\}_{j=1}^{\hat{B}} | \hat{B}, \hat{\boldsymbol{b}}, \hat{\boldsymbol{m}}\right) = \sum_{j=1}^{\hat{B}} \log_2 \hat{m}_j + \sum_{j=1}^{\hat{B}} \sum_{r=1}^{\hat{m}_j} \log_2 \hat{d}_{jr}. \tag{10}$$

Derivation of $L\left(\{\hat{\boldsymbol{\alpha}}_j, \hat{\boldsymbol{\beta}}_j\}_{j=1}^{\hat{B}} | \hat{B}, \hat{\boldsymbol{b}}, \hat{\boldsymbol{m}}, \{\hat{\boldsymbol{k}}_j\}_{j=1}^{\hat{B}}\right)$. Once $\hat{B}$, $\hat{\boldsymbol{b}}$, $\hat{\boldsymbol{m}}$ and $\{\hat{\boldsymbol{k}}_j\}_{j=1}^{\hat{B}}$ are specified, (conditional) maximum likelihood estimates of $\{\hat{\boldsymbol{\alpha}}_j, \hat{\boldsymbol{\beta}}_j\}_{j=1}^{\hat{B}}$ can be uniquely computed by least–squares regression. Rissanen (1989, Ch. 3) demonstrated that if a (conditional) maximum likelihood estimate is estimated from N data points, then it can be effectively encoded with $\frac{1}{2}\log_2 N$ bits. It is obvious to see that, for a given j, the corresponding $\hat{\alpha}_{jr}$'s and $\hat{\beta}_{jr}$'s are estimated from $\hat{l}_j$ data points. There are four $\hat{\alpha}_{jr}$'s and $\hat{m}_j$ $\hat{\beta}_{jr}$'s, so the code length for $\{\hat{\boldsymbol{\alpha}}_j, \hat{\boldsymbol{\beta}}_j\}$ is $\frac{1}{2}(4+\hat{m}_j)\log_2 \hat{l}_j$, and hence

$$L\left(\{\hat{\boldsymbol{\alpha}}_j, \hat{\boldsymbol{\beta}}_j\}_{j=1}^{\hat{B}} | \hat{B}, \hat{\boldsymbol{b}}, \hat{\boldsymbol{m}}, \{\hat{\boldsymbol{k}}_j\}_{j=1}^{\hat{B}}\right) = \sum_{j=1}^{\hat{B}} \frac{4+\hat{m}_j}{2} \log_2 \hat{l}_j. \tag{11}$$

Derivation of $L(\hat{\boldsymbol{e}}|\hat{\boldsymbol{\theta}})$. Based on Shannon's classical results in information theory (e.g., Shannon and Weaver (1949)), Rissanen (1989, Ch. 3) showed that the code length of $\hat{\boldsymbol{e}}$ is given by the negative of the log of the likelihood of $\hat{\boldsymbol{e}}$ conditioned on $\hat{\boldsymbol{\theta}}$. For the present problem, it simplifies to

$$L(\hat{\boldsymbol{e}}|\hat{\boldsymbol{\theta}}) = \frac{n}{2} \log\left[\frac{1}{n}\sum_{i=1}^{n}\left\{y_i - \hat{f}(x_i)\right\}^2\right] + C, \tag{12}$$

where C is a constant term. Now by changing $\log_2$ to log, combining (8) to (12) and ignoring the negligible terms $\log_2 \hat{d}_{jr}$'s and C, one obtains MDL($\hat{f}$).

B Derivation of RMDL($\hat{f}$)

The derivation of RMDL($\hat{f}$) is very similar to the derivation of MDL($\hat{f}$). To simplify notation we continue to denote a candidate model for MDL($\hat{f}$) as $\hat{\boldsymbol{\theta}}$, but use $\hat{\boldsymbol{\theta}}_{\text{OUT}}$ for a RMDL($\hat{f}$) candidate model. The goal is to derive an approximation for $L(\boldsymbol{y}) = L(\hat{\boldsymbol{\theta}}_{\text{OUT}}) + L(\hat{\boldsymbol{e}}|\hat{\boldsymbol{\theta}}_{\text{OUT}})$. Since $\hat{\boldsymbol{\theta}}_{\text{OUT}}$ and $\hat{\boldsymbol{\theta}}$ only differ by $\hat{w} = \hat{n}_{\text{OUT}}/n$, $L(\hat{\boldsymbol{\theta}}_{\text{OUT}})$ is just $L(\hat{\boldsymbol{\theta}})$ plus the extra code length for $\hat{n}_{\text{OUT}}$. This extra code length is given by $\log_2 \hat{n}_{\text{OUT}}$, but it can be ignored as $\hat{n}_{\text{OUT}}$ is small. Our numerical investigations also confirm this. So $L(\hat{\boldsymbol{\theta}}_{\text{OUT}})$ is given by the first three terms of (7). Now, similar to $L(\hat{\boldsymbol{e}}|\hat{\boldsymbol{\theta}})$, $L(\hat{\boldsymbol{e}}|\hat{\boldsymbol{\theta}}_{\text{OUT}})$ is given by the negative of the log of the likelihood of $\hat{\boldsymbol{e}}$ conditioned on $\hat{\boldsymbol{\theta}}_{\text{OUT}}$, which simplifies to the last term of (7).

References

Abramovich, F., Sapatinas, T. and Silverman, B. W. (1998). Wavelet thresholding via a Bayesian approach. *J. Royal Statist. Soc. Ser. B* **60**, 725–749.

Burnham, K. P. and Anderson, D. R. (1998). *Model Selection and Inference: A Practical Information-Theoretic Approach.* Springer-Verlag New York Inc.

Chaudhuri, P. and Marron, J. S. (1999). SiZer for exploration of structures in curves. *J. Amer. Statist. Assoc.* **94**, 807–823.

Cleveland, W. S. and Devlin, S. J. (1988). Locally weighted regression: an approach to regression analysis by local fitting. *J. Amer. Statist. Assoc.* **83**, 596–610.

Davis, L. (1991). *Handbook of Genetic Algorithms.* Van Nostrand Reinhold, New York.

Denison, D. G. T., Mallick, B. K. and Smith, A. F. M. (1998). Automatic Bayesian curve fitting. *J. Royal Statist. Soc. Ser. B* **60**, 333–350.

Donoho, D. L. and Johnstone, I. M. (1995). Adapting to unknown smoothness via wavelet shrinkage. *J. Amer. Statist. Assoc.* **90**, 1200–1224.

Friedman, J. H. (1991). Multivariate adaptive regression splines (with discussion). *Ann. Statist.* **19**, 1–141.

Friedman, J. H. and Silverman, B. W. (1989). Flexible parsimonious smoothing and additive modeling (with discussion). *Technometrics* **31**, 3–21.

Hall, P. and Titterington, D. M. (1992). Edge–preserving and peak–preserving smoothing. *Technometrics* **34**, 429–440.

Hurvich, C. M., Simonoff, J. S. and Tsai, C.-L. (1998). Smoothing parameter selection in nonparametric regression using an improved Akaike information criterion. *J. Royal Statist. Soc. Ser. B* **60**, 271–293.

Koo, J.-Y. (1997). Spline estimation of discontinuous regression functions. *J. Comp. Graph. Statist.* **6**, 266–284.

Lee, T. C. M. (1999). Robust fitting of discontinuous regression functions. *in* Proceedings of the Interface 31. pp. 476–481.

Lee, T. C. M. (2000). Regression spline smoothing using the minimum description length principle. *Statist. Probab. Letters* **48**, 71–82.

Lee, T. C. M. (2001). An introduction to coding theory and the two–part minimum description length principle. *International Statistical Review* **69**, 169–183.

Lee, T. C. M. (2002). Automatic smoothing for discontinuous regression functions: Supporting document. Download: `http://www.stat.colostate.edu/~tlee/PSfiles/support.ps.gz`.

Liang, F. and Wong, W. H. (2000). Evolutionary Monte Carlo: Applications to C_p model sampling and change point problem. *Statist. Sinica* **10**, 317–342.

Luo, Z. and Wahba, G. (1997). Hybrid adaptive splines. *J. Amer. Statist. Assoc.* **92**, 107–116.

McDonald, J. A. and Owen, A. B. (1986). Smoothing with split linear fits. *Technometrics* **28**, 195–208.

Nason, G. P. and Silverman, B. W. (1994). The discrete wavelet transform. *J. Comp. Graph. Statist.* **3**, 163–191.

Pittman, J. (1999). Adaptive splines and genetic algorithms. *J. Comp. Graph. Statist.* To appear.

Pittman, J. and Murthy, C. (2000). Fitting optimal piecewise linear functions using genetic algorithms. *IEEE Trans. Patt. Anal. Machine Intell.* **22**, 701–718.

Rissanen, J. (1989). *Stochastic Complexity in Statistical Inquiry.* World Scientific, Singapore.

Ruppert, D., Sheather, S. J. and Wand, M. P. (1995). An effective bandwidth selector for local least squares regression. *J. Amer. Statist. Assoc.* **90**, 1257–1270.

Shannon, C. E. and Weaver, W. (1949). *The Mathematical Theory of Communication.* University of Illinois Press, Urbana and Chicago.

Smith, M. and Kohn, R. (1996). Nonparametric regression using Bayesian variable selection. *J. Econometrics* **75**, 317–344.

Titterington, D. M., Smith, A. F. M. and Makov, U. E. (1985). *Statistical Analysis of Finite Mixture Distributions.* John Wiley & Sons Ltd. Chichester.

Wahba, G. (1990). *Spline models for observational data. CBMS-NSF, Regional Conference Series in Applied Mathematics*. SIAM. Philadelphia.

Wand, M. P. (1998). *KernSmooth 2.22 Reference Manual.*

Wang, Y. (1995). Jump and sharp cusp detection by wavelets. *Biometrika* **82**, 385–397.

Automatic Smoothing for Discontinuous Regression Functions: Supporting Document

This supporting document provides additional material for the manuscript "Automatic Smoothing for Discontinuous Regression Functions". It contains two sections. Section A provides additional plots for Section 6 (Simulation Results) of the manuscript, while Section B lists the major steps of the genetic algorithms used in the manuscript.

A Graphical Outputs

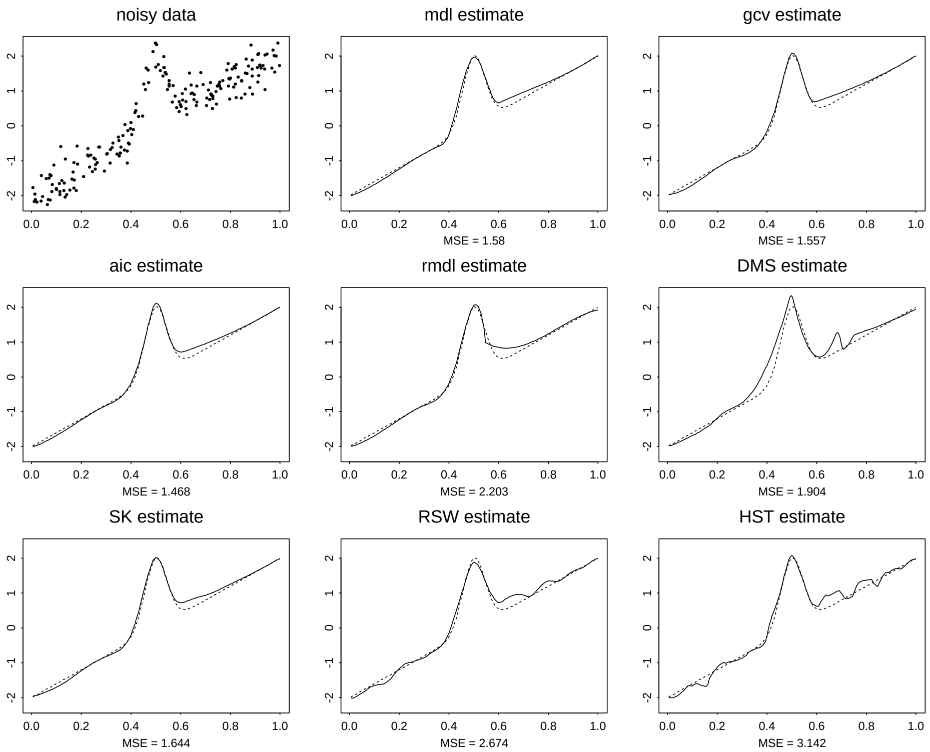

Figure 1: Test Function 1. Solid lines: estimates; broken lines: true curves.

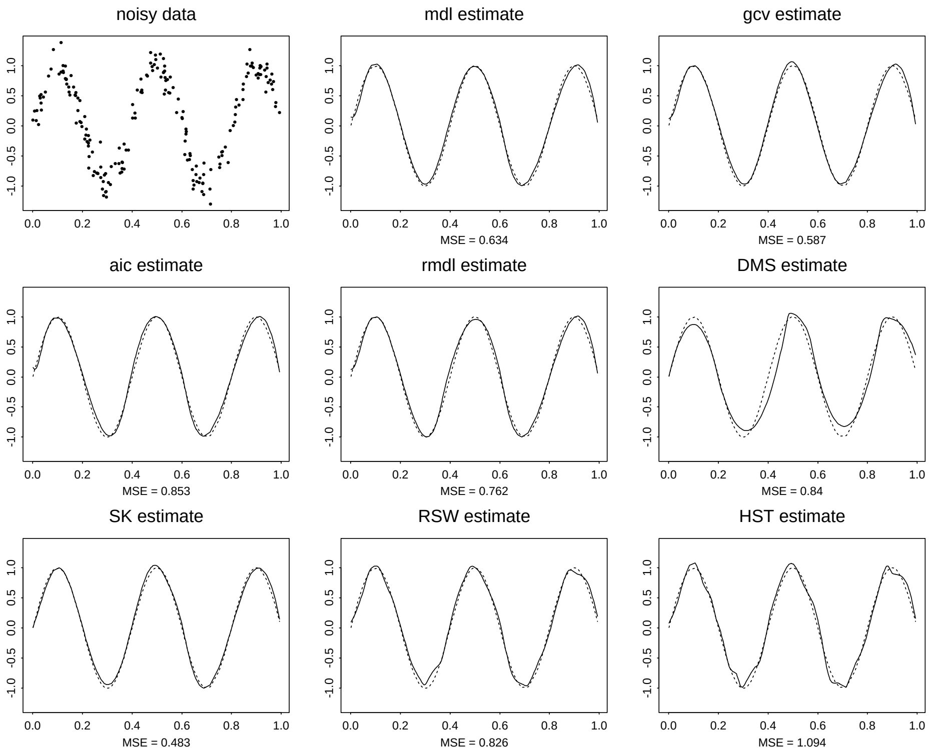

Figure 2: Test Function 3. Solid lines: estimates; broken lines: true curves.

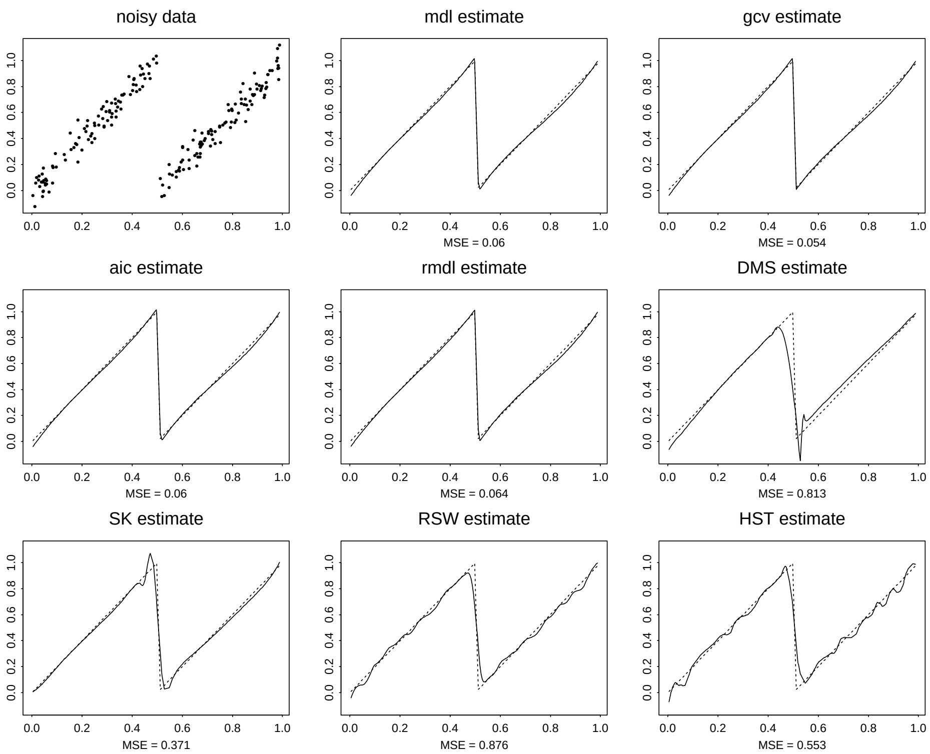

Figure 3: Test Function 4. Solid lines: estimates; broken lines: true curves.

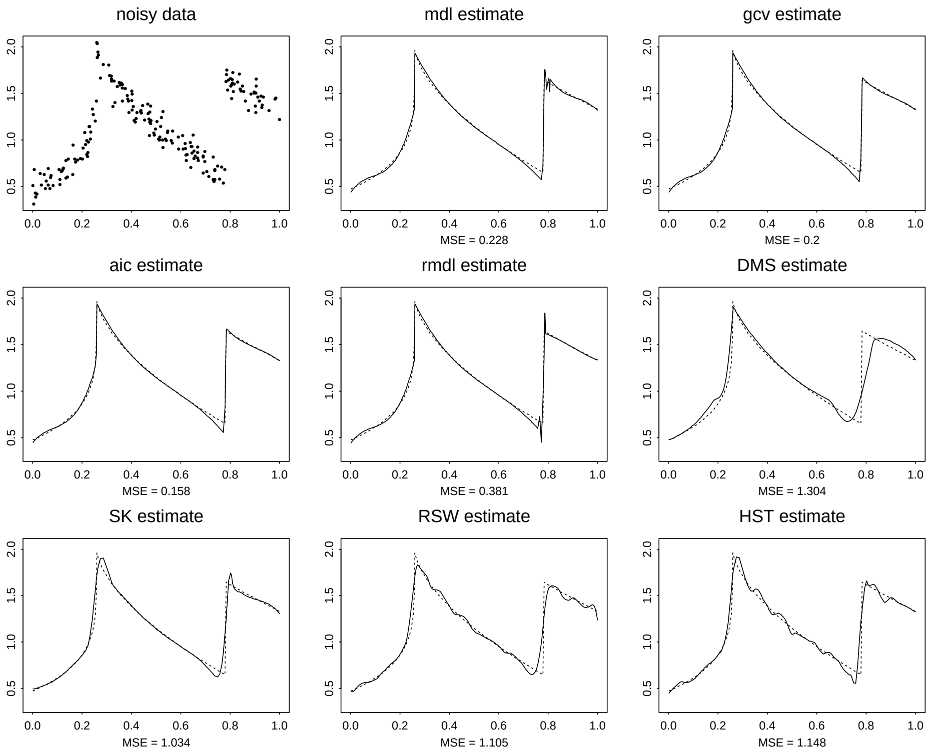

Figure 4: Test Function 6. Solid lines: estimates; broken lines: true curves.

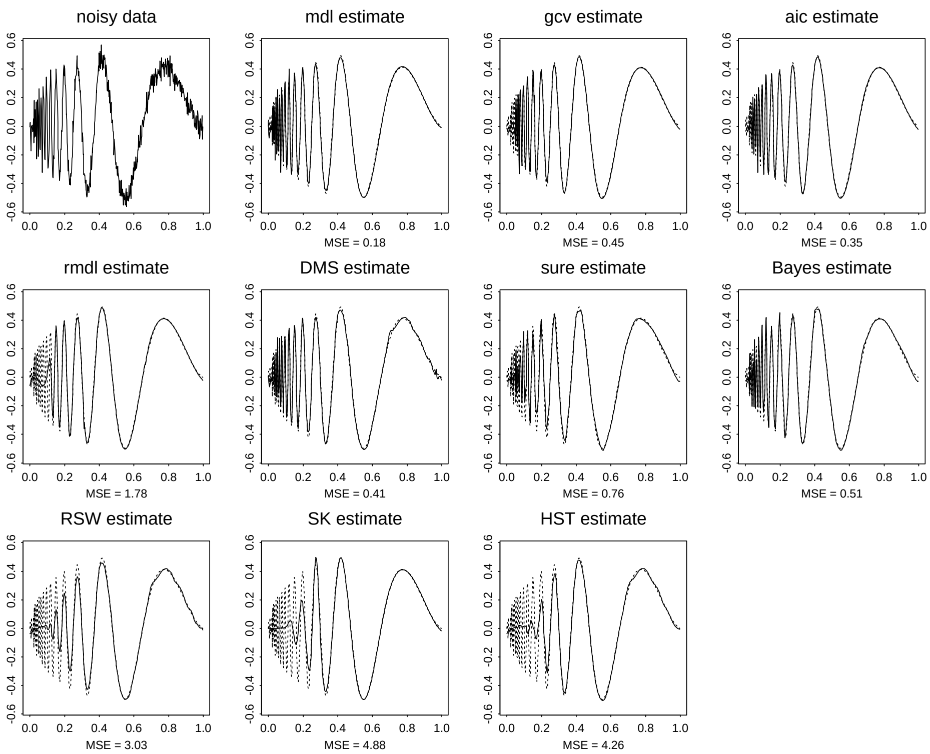

Figure 5: Test Function *Doppler*. Solid lines: estimates; broken lines: true curves.

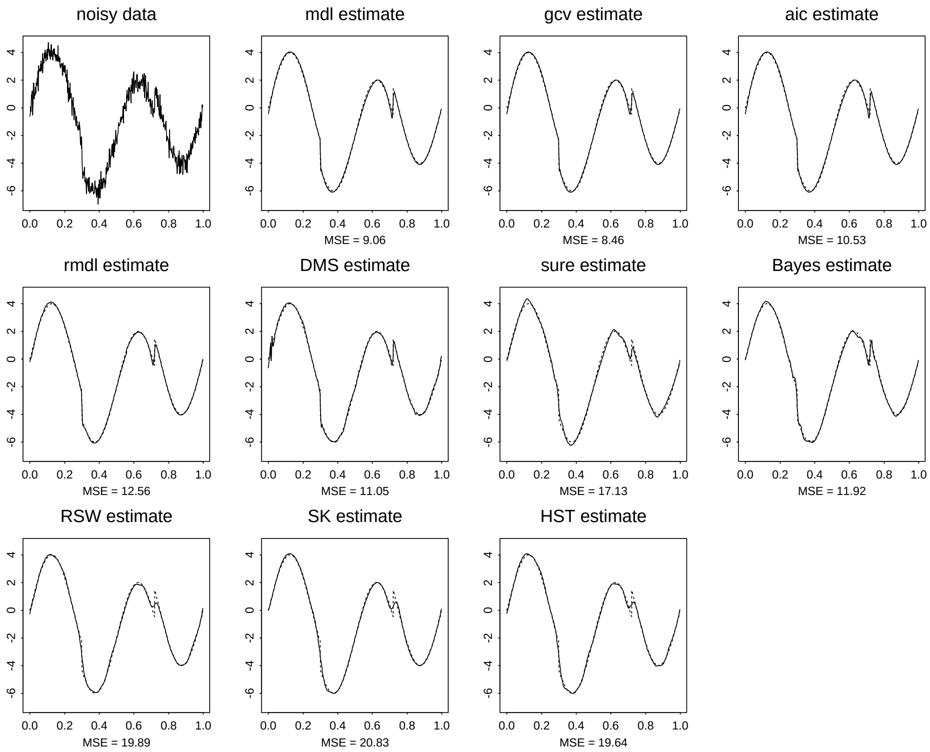

Figure 6: Test Function *Heavisine*. Solid lines: estimates; broken lines: true curves.

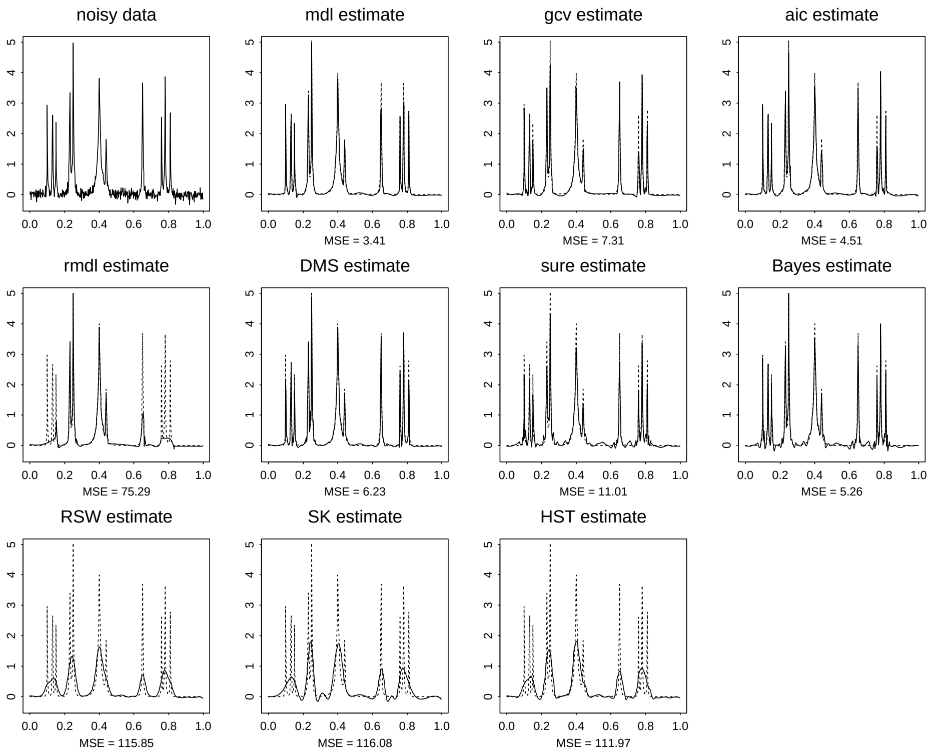

Figure 7: Test Function *Bumps*. Solid lines: estimates; broken lines: true curves.

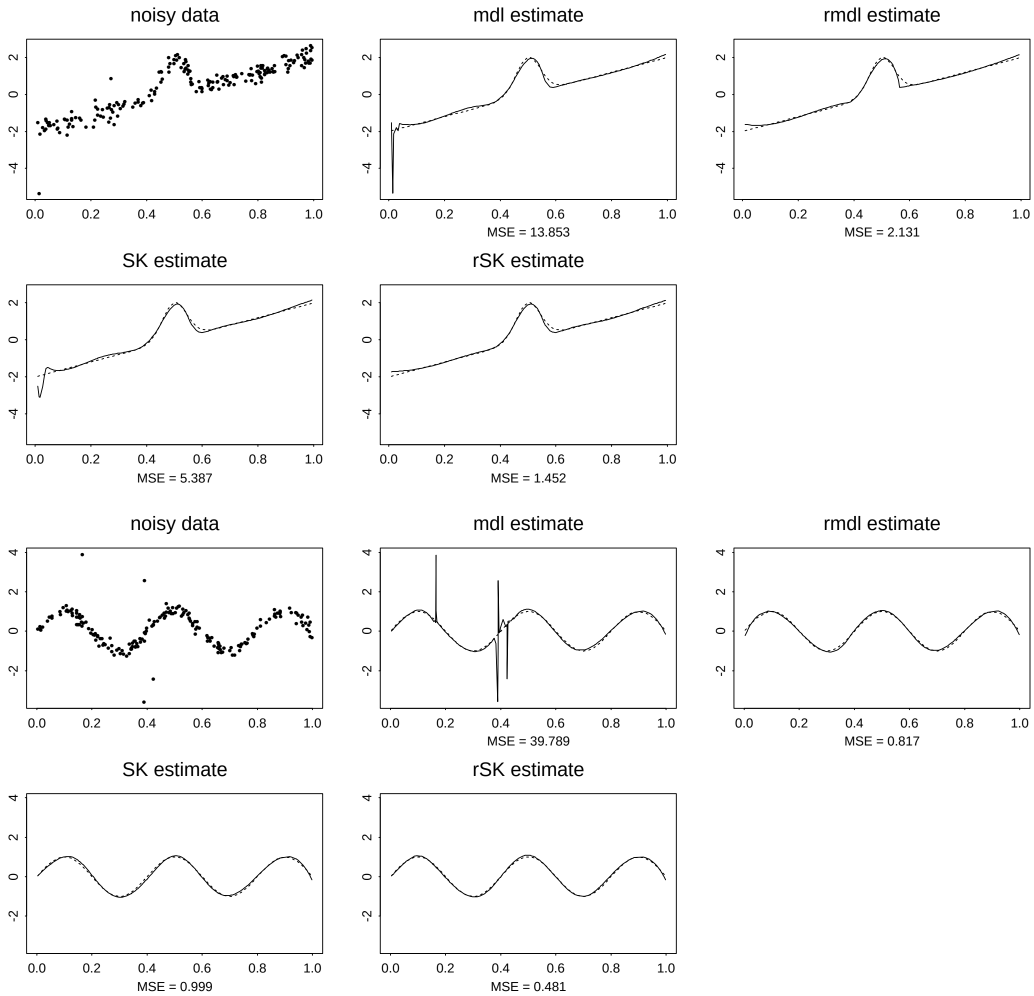

Figure 8: Various estimates of Test Functions 1 (top two rows) and 3 (bottom two rows) when data were contaminated by outliers. Solid lines: estimates; broken lines: true curves.

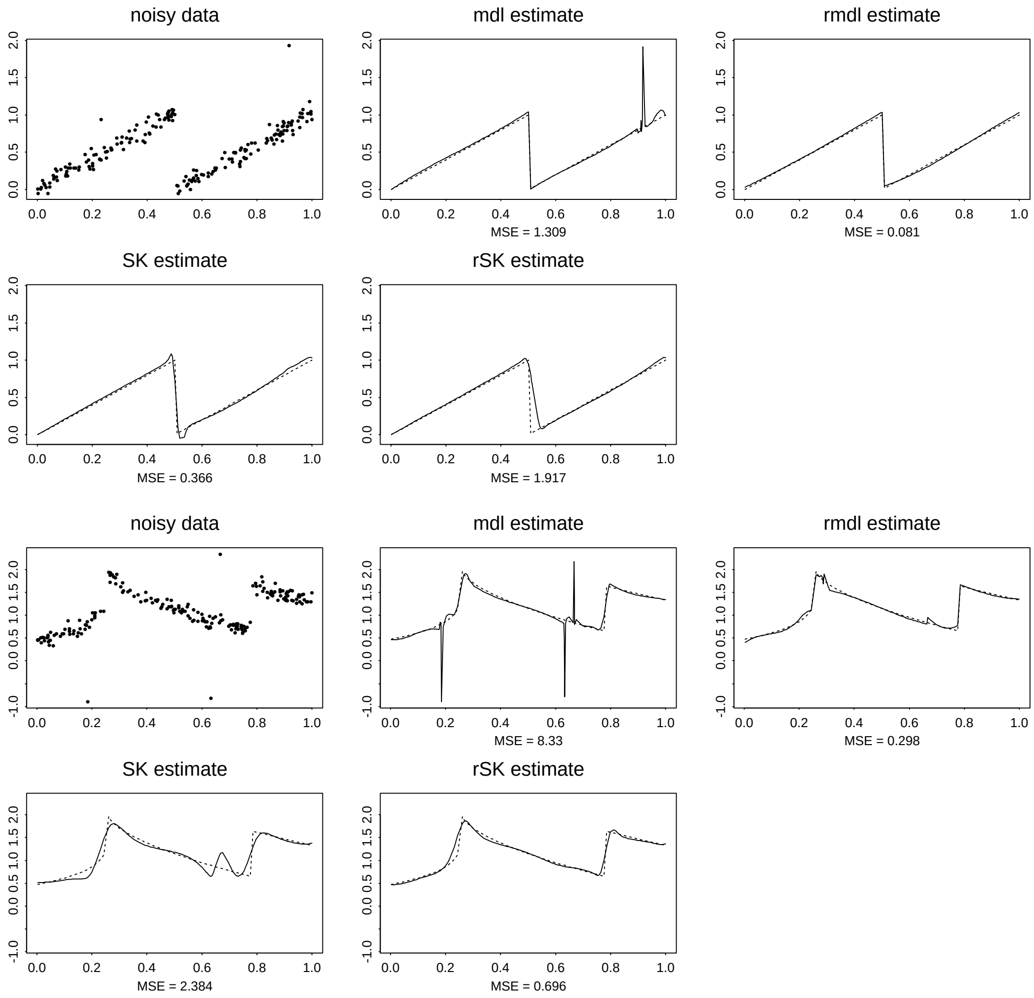

Figure 9: Various estimates of Test Functions 4 (top two rows) and 6 (bottom two rows) when data were contaminated by outliers. Solid lines: estimates; broken lines: true curves.

B Implementation of Genetic Algorithms

This section lists the major steps in the implementation of the genetic algorithm that was used to minimize $\mathrm{RMDL}(\hat{f})$ in the numerical experiments. Genetic algorithms for minimizing $\mathrm{MDL}(\hat{f})$, $\mathrm{GCV}(\hat{f})$ and $\mathrm{AIC}(\hat{f})$ are similar and hence omitted. The steps are as follows.

1. Randomly generate an initial population of chromosomes of size S.

2. Compute the $\mathrm{RMDL}(\hat{f})$ value for each of the S chromosomes.

3. Sort the $\mathrm{RMDL}(\hat{f})$ values in descending order. Assign rank 1 to the chromosome with the largest $\mathrm{RMDL}(\hat{f})$ value, rank 2 to the one with the second largest $\mathrm{RMDL}(\hat{f})$ value, and so on. Denote the rank of chromosome i as r_i.

4. With probability P_{CROSS}, perform a crossover operation. Otherwise, perform a mutation operation.

   - *Crossover* — one child is produced from two parents: select two parent chromosomes from the initial population. The probability that chromosome i will be selected is $r_i/(r_1 + \ldots + r_S)$. Then for each gene location of the child chromosome, with equal probability, assign it either the corresponding father gene or the corresponding mother gene.

   - *Mutation* — one child is produced from one parent: select one parent using the same probability law as in crossover. For each child gene location, assign it either a knot gene with probability P_{K}, a discontinuity gene with probability P_{D}, an outlier gene with probability P_{O}, a normal gene with probability P_{N}, or the corresponding father gene with probability $1 - P_{\text{K}} - P_{\text{D}} - P_{\text{O}} - P_{\text{N}}$.

5. Repeat Step 4 until S child chromosomes are produced. That is, until a whole new generation is obtained.

6. Repeat Steps 2 to 5 using the new generation as the initial population. Then repeat the whole process until the RMDL($\hat{f}$) value of the best chromosome does not change for N_{SAME} generations.

7. The best chromosome of the youngest generation is taken as the minimizer of RMDL($\hat{f}$).

In our implementation $S = 300$, $P_{\text{CROSS}} = 0.9$, $P_{\text{K}} = 0.1$, $P_{\text{D}} = 0.02$, $P_{\text{O}} = 0.3$, $P_{\text{N}} = 0.1$ and $N_{\text{SAME}} = 20$.