

title page

Adaptive Splines and Genetic Algorithms

Jennifer Pittman

National Institute of Statistical Sciences

P.O. Box 14006

Research Triangle Park, NC 27709

`pittman@niss.org`

Adaptive Splines and Genetic Algorithms

Jennifer PITTMAN

Most existing algorithms for fitting adaptive splines are based on nonlinear optimization and/or stepwise selection. Stepwise knot selection, although computationally fast, is necessarily suboptimal while determining the best model over the space of adaptive knot splines is a very poorly behaved nonlinear optimization problem. A possible alternative is to use a genetic algorithm to perform knot selection.

An adaptive modeling technique referred to as adaptive genetic splines (AGS) is introduced which combines the optimization power of a genetic algorithm with the flexibility of polynomial splines. Preliminary simulation results comparing the performance of AGS to those of existing methods such as HAS, SUREshrink and automatic Bayesian curve fitting are discussed. A real data example involving the application of these methods to a fMRI dataset is presented.

KEY WORDS: splines; genetic algorithms; generalized cross-validation; nonparametric regression

1 INTRODUCTION

One purpose of statistical data analysis is to determine a functional relationship between some input and output variables given a dataset of noisy observed values. In the univariate case, the dataset $\{(x_i, y_i) : i = 1, \dots, N\} \subset \mathcal{R}^2$, $a \leq x_1 < x_2 < \dots < x_N \leq b$, $a, b \in \mathcal{R}$, is assumed to be a number of realizations of some underlying process combined with random noise, i.e.,

$$\mathbf{y} = f(\mathbf{x}) + \epsilon$$

where the ϵ_i s are assumed to be independent and follow a distribution with mean zero. In this paper we consider the situation where little is known about the function f and hence a flexible yet powerful smoothing technique is desired for modeling purposes. An algorithm entitled Adaptive Genetic Splines (AGS) was developed which utilizes the optimization power of a genetic algorithm to determine the appropriate B-spline model for a given univariate dataset. We will outline the AGS algorithm and compare its performance with those of existing techniques on both simulated and real datasets.

Jennifer Pittman is a research fellow at the National Institute of Statistical Sciences (NISS) located in Research Triangle Park, NC. This research was partially supported by Army Research Office Assent Grant DAAG55-97-1-0219

© American Statistical Association
Journal of Computational and Graphical Statistics

Traditionally, adaptive smoothing methods could be classified into one of two groups. The first group consists of methods which use a locally adaptive smoothing parameter with smoothing splines or kernel techniques. An example is the variable bandwidth kernel method of Fan and Gijbels (1995). The second group consists of regression spline fitting algorithms where it is assumed that f can be approximated by a continuous piecewise polynomial (spline) satisfying certain continuity conditions on its lower order derivatives. The candidate models belong to the space of spline functions of order m with some knot sequence \mathbf{t} , denoted $\mathcal{S}_{m,\mathbf{t}}$, and in the fitting process the model knot sequence is chosen adaptively. Early attempts at adaptive selection include algorithms such as de Boor's (1978) which iteratively adds or deletes knots but does not attempt optimal knot placement, only a distribution for the knots which is in some sense optimal. There are also nonlinear optimization routines which attempt to optimize knot placement for a fixed, given number of knots; Schwetlick and Schütze (1995) have presented a nice extension of these works, based on generalized Gauss-Newton methods, which incorporates a knot removal strategy. A recent approach to free knot spline modeling which employs nonlinear optimization is the algorithm of Lindstrom (1999). Approaching the choice of knot locations as a parameter estimation problem, her algorithm seeks an estimate for the knot sequence which minimizes a measure of error times a penalty which increases as the knots coalesce. This aids in avoiding local optima and is intuitively appealing if the true, unknown function is assumed to be smooth.

Other applications of statistical variable selection techniques to adaptive spline modeling include Friedman's MARS (1991), which utilizes a stepwise knot addition/deletion strategy with linear splines. His generalized cross-validation (GCV)(Craven and Wahba 1979) model selection criterion includes a 'cost' term to adjust for the adaptive nature of the knot selection. Recent contributions to additive modeling have been made by Luo and Wahba [HAS](1997), whose algorithm performs forward knot selection via GCV with a 'cost' term as in MARS but replaces backward deletion by ridge regression, and Denison, Mallick and Smith [automatic Bayesian curve fitting](1998) whose method is discussed below.

Most of the algorithms mentioned are based on nonlinear optimization and/or stepwise selection. Although computationally fast and adaptive, stepwise knot selection is necessarily suboptimal while determining the best model over the space of adaptive splines is a very poorly behaved nonlinear optimization problem (Wahba 1988). The need for a more intelligent search scheme has driven recent work on Bayesian model selection and Markov chain Monte Carlo (McMC) methods for model identification (Hansen and Kooperberg 1999). Bayesian spline methods which combine both McMC and model selection include those from Smith and Kohn (1996), Denison et al. (1998), Shively, Kohn and Wood (1999)

and Biller (1999). In essence, a posterior distribution is created computationally which allows a simple Gibbs sampler to quickly search through a large number of knot locations in search of a posterior mode. These methods have proven to be quite successful although it should be noted that convergence can be slow if the McMC chain is started in a low density region and mixing can be slow when the density of interest may be highly correlated (Holmes and Mallick 1998). Also, in the case of a multi-modal density the sampler may effectively get stuck in a local mode. However, the performance of such methods shows that when properly designed, stochastic methods can find many more reasonable knot configurations than their greedy, deterministic competitors (Hansen and Kooperberg 1999).

As noted by Hansen and Kooperberg (1999), the use of McMC for selecting the posterior mode in a Bayesian framework is essentially simulated annealing. Simulated annealing methods, particularly Metropolis-type (Metropolis, Rosenbluth, Rosenbluth, Teller and Teller 1953), are stochastic optimization methods which produce a sequence that converges in probability to the set of global minima (maxima) of the loss function as the temperature converges to zero. Stone, Hansen, Kooperberg and Troung (1997) successfully used simulated annealing for logspline density estimation; Geyer and Thompson (1995) used a simulated annealing approach to aid in the mixing of McMC chains in correlated spaces.

The above research led us to consider the possibility of using another stochastic optimization technique, genetic algorithms (GAs), for knot selection. Genetic algorithms use a handful of basic Darwinian operations on the set of possible solutions (or possible population members) to find the 'fittest' solution (or individual). The underlying concept is to use properties of natural selection such as reproduction and genetic mutation to solve optimization problems. GAs have been used for optimization in similar contexts; for example, Manela, Thornhill and Campbell (1993) used a genetic algorithm to determine the appropriate spline order and roughness penalty for penalized least squares (LS) splines while Rogers [G/SPLINES](1991) incorporated a modified GA search into MARS. More recently, Holmes and Mallick (1998) used genetic-type operators for McMC updating in the context of knot selection in Bayesian spline models. The selection of non-linear basis functions by a GA has been employed in recent years with promising results by researchers in the earth sciences (Drijkonigen and White 1995) and in chemoinformatics (Cho, Cummins, Bentley, Andrews and Tropsha 1996; Rogers and Hopfinger 1994) as well as in the field of machine learning (Huang, Liu and Wechsler 1998; Wise 1994). Other works involving GAs and non-linear function approximation that are of note include Whitehead and Choate (1996) and Lankhorst and van der Laan (1994).

Genetic algorithms have not only been used successfully in numerous applications but have also been shown theoretically to converge, under certain design conditions, to the global optimum of the evaluation function of an optimization problem (Bhandari, Murthy and Pal 1996). Hence, given a variable selection criterion and a search space of possible knot locations, a genetic algorithm has the potential to find models whose quality of fit is optimal relative to models selected using alternative techniques.

Section 2 is a basic introduction to genetic algorithms followed by Section 3 in which the application of a genetic algorithm to the problem of interest is discussed. The proposed algorithm, Adaptive Genetic Splines (AGS), is described in Section 4 and simulation results comparing the performance of AGS to existing methods are presented. A real data example involving fMRI data is examined in Section 4. The final section, Section 5, contains a brief summary of our results followed by suggestions for future research.

2 GENETIC ALGORITHMS

Genetic algorithms (GAs) are a class of stochastic search methods originally developed by Holland (1975) which, in many cases, are able to provide a near optimal solution to an optimization problem. The basic idea is to maintain a population of possible solutions that evolves and improves over time through a process of competition and controlled variation. Their success comes from their *adaptability*, i.e., the ability to accumulate information about the search space through exploration and use this information to bias subsequent searches towards the exploitation of promising subspaces. Note that ‘hillclimbing’ algorithms do not allow for any such tradeoff between exploration and exploitation within the algorithm itself, thus limiting their own search capabilities. They use only local information and hence consider only modes in the search space, ignoring the possibility of other potentially optimal locations. In contrast, a GA provides a high degree of exploration through the diversity of the population of possible solutions and the use of various operators (described below). As the search proceeds a GA uses gradient information from the search space contained implicitly in the distribution of the population, along with less exploration are more exploitation, to converge to a solution.

Each solution in the domain space \mathcal{D} is encoded as a string or chromosome S . A string is built of elements from a finite alphabet $\mathcal{A} = \{\alpha_1, \alpha_2, \dots, \alpha_a\}$; the alphabet is determined by the encoding scheme. For example, suppose that we have a univariate dataset of size $N = 100$ to which we would like to fit a model from the domain space \mathcal{D} of all cubic splines with k interior knots whose locations are restricted to the design points. Then each possible solution may be represented by its interior knot locations and encoded as a string of length k of nondecreasing integer values selected

from the alphabet $\mathcal{A} = \{2, 3, \dots, N - 1\}$; the integer i would represent the i th largest design point. The length L of each string is determined by the number of parameters to be found and often the desired level of model quality or goodness-of-fit; the population size M is usually chosen to be approximately twice the string length. Each string S corresponds to a value x in \mathcal{D} and may be expressed as

$$S = (\gamma_1, \gamma_2, \dots, \gamma_L); \quad \gamma_i \in \mathcal{A}, \quad i = 1, \dots, L.$$

In each iteration t , a GA starts with a population $P^{(t)}$ of M strings, $P^{(t)} = \{S_1, \dots, S_M\}$. The strings in $P^{(t)}$ are randomly selected from the pool of all encoded solutions, e.g., in the above example each string in $P^{(t)}$ would be a nondecreasing sequence of integers randomly selected from $\{2, 3, \dots, N - 1\}$. Through operations based on natural selection, the GA performs a directed search for a solution by optimizing a specified function (e.g., a measure of model lack of fit) $fit(S_i)$, $S_i \in \mathcal{D}$.

2.1 The Basic Steps

Starting with a randomly generated population, at each iteration (or *generation*) a GA applies chosen operators to the strings in the given population which yields a new population of ‘fitter’ strings or solutions of higher average quality. The standard operators are selection, crossover and mutation.

- **Selection:** Selection gives members of the present population P^t with large fitness values an increased chance of being present in the next (intermediate) population $P_1^{(t)}$. M strings are selected with replacement from $P^{(t)}$ using a specific selection mechanism to form $P_1^{(t)}$. Most selection mechanisms use either *proportional* (Holland 1975) or *ordinal* (Miller and Goldberg 1995) selection. In proportional selection, the probability of selecting a string S_i from $P^{(t)}$ for inclusion in $P_1^{(t)}$ is $p_s(S_i) = fit(S_i) / \sum_i^M fit(S_i)$; selection is done with replacement. For example, if $P^{(t)}$ contains $M = 4$ strings which represent splines with RSS values of (0.1, 0.04, 0.025, 0.2), respectively, and $fit(S_i) = 1/RSS$, then the probabilities of selection for each string would be $(10/80, 25/80, 40/80, 5/80) = (0.125, 0.3125, 0.5, 0.0625)$. In ordinal selection, the strings are sorted according to their fitness values and $p_s(S_i)$ is based on the rank of S_i using a nonincreasing assignment function. The selection operator dominates early GA performance by increasing the concentration of solutions in regions with current fitness values above the population average (Li 1992).

- **Crossover:** Crossover or mating allows pairs of strings from $P_1^{(t)}$ to combine their better features to create improved strings for the next (intermediate) population $P_2^{(t)}$. It fosters population diversity and thus helps prevent premature convergence. All strings are paired at random in such a way that each string belongs to only one pair (hence there are $M/2$ pairs). Using the above example with $k = L = 4$ and sample size $N = 40$, let $S_1 = (\gamma_{1,1}, \dots, \gamma_{1,L}) = (2, 14, 26, 30)$ and $S_2 = (\gamma_{2,1}, \dots, \gamma_{2,L}) = (12, 24, 29, 35)$ be two strings from $P_1^{(t)}$ that have been selected for crossover. In *simple* crossover (Holland 1975), a position $i \in \{1, 2, \dots, L-1\}$ is randomly chosen and two new strings are built,

$$S'_1 = (\gamma_{1,1}, \dots, \gamma_{1,i}, \gamma_{2,i+1}, \dots, \gamma_{2,L}) \quad \text{and} \quad S'_2 = (\gamma_{2,1}, \dots, \gamma_{2,i}, \gamma_{1,i+1}, \dots, \gamma_{1,L}).$$

S'_1 and S'_2 are placed in $P_2^{(t)}$ and S_1 and S_2 are discarded. Hence if $i = 2$ then two new splines represented by $S'_1 = (2, 14, 29, 35)$ and $S'_2 = (12, 24, 26, 30)$ would be placed in $P_2^{(t)}$ and the splines $S_1 = (\gamma_{1,1}, \dots, \gamma_{1,L}) = (2, 14, 26, 30)$ and $S_2 = (\gamma_{2,1}, \dots, \gamma_{2,L}) = (12, 24, 29, 35)$ would be removed from the current population. Each of the $M/2$ pairs in $P_1^{(t)}$ undergoes crossover with probability p_c (see below); any pair not selected for crossover is placed directly into $P_2^{(t)}$.

- **Mutation:** Mutation gives the algorithm an opportunity to branch into previously unexplored regions of the domain space by arbitrarily altering one or more characters of a selected string. Mutation can also restore lost or unexplored genetic material into the population to prevent premature convergence and ensure that the probability of reaching any point in the search space is never zero. Each character of every string undergoes mutation with probability p_m . In the simplest case, for each character $\gamma_{i,j}$ in $P_2^{(t)}$, $i = 1, \dots, M$, $j = 1, \dots, L$, a random number rnd is generated from $[0, 1]$. If $rnd > p_m$, $\gamma_{i,j}$ is unchanged; otherwise, $\gamma_{i,j}$ is replaced by a randomly selected member of the set $\{\mathcal{A} - \gamma_{i,j}\}$. As a concrete example, consider S'_1 above, $p_m = 0.1$, and suppose that the values of rnd generated for each knot are $(0.08, 0.6, 0.2, 0.03)$. Then S'_1 may look like $(10, 14, 29, 31)$ after mutation, where the values 10 and 31 were selected from the sets $\{\{2, 3, \dots, 39\} - 10\}$ and $\{\{2, 3, \dots, 39\} - 31\}$, respectively, and the values 14, 29 were unchanged.

Both crossover and mutation are recombination operations which maintain a sufficiently broad yet refined search. The choice of p_c and p_m can be a complex nonlinear optimization problem and the appropriate values depend on the nature of the fitness function. Despite this, some guidelines are available. Crossover is designed to promote exploration and population diversity; as such, a value of p_c between 0.6 and 0.9 is often recommended (DeJong 1978; Herrera 1998)

although GA performance is not considered to be sensitive to the value of this parameter. Mutation should not destroy structures, or combinations of parameter values, more quickly than selection can reproduce them. Thus, for example, p_m can be proportional to $1/(L\sqrt{M})$ (Back 1993; Qi and Palmieri 1994), i.e., proportional to the dimension of the space yet also adaptive, decreasing exponentially as the number of iterations increases. In many applied contexts the choice of values for M , L , p_c and p_m (and the type of coding and operators) are determined by consulting the above guidelines as well as results from pilot studies (Bangalore, Shaffer and Small 1996) and/or considerations of available computing power (Broadhurst, Goodacre, Jones, Rowland and Kell 1997; Lankhorst and van der Laan 1994). For binary codings it has been found that the computational time of a GA is strictly proportional to the population size and the number of generations, while for a fixed accuracy level most problems exhibit a hyperbolic relationship between the number of generations and the population size (Chatterjee, Laudato and Lynch 1996). Thus, to some extent, an increase in population size can be exchanged for a decrease in the number of generations.

An additional selection strategy referred to as an *elitist* step (DeJong 1978), where in each generation the best individual from the current starting population $P^{(t)}$ is included in the subsequent starting population $P^{(t+1)}$, is often incorporated since the best chromosome may disappear due to crossover or mutation.

$P_2^{(t)}$ is now relabeled as $P^{(t+1)}$ and the cycle of operations is repeated starting with $P^{(t+1)}$; this process continues until some termination criterion is met, at which time the best string achieved is generally taken as the solution to the optimization problem. Popular stopping criteria include executing the process (1) for a fixed number of iterations, (2) until the best $r\%$ of strings are identical or the range of fitness values is adequately small, or (3) until the best fitness value does not show sufficient improvement over a fixed number of iterations. (Rogers and Hopfinger 1994; Broadhurst et al. 1997; Cho et al. 1996). In our experiments stopping rule (1) was implemented 5).

2.2 Why GAs?

Determining the proper adaptive spline for a given dataset can be viewed as a variable selection problem: given a large set of candidate knot locations and a criterion of fit, find the subset of knots of a certain size which yields the best model. Exhaustive enumeration is not an option due to the size of the domain space. For a fixed number of knots, there are usually multiple local optima in the sum of squared errors surface (Jupp 1978) so a traditional derivative-based or ‘hill-climbing’ approach may get stuck in a local minimum of the error surface depending upon the choice of the initial knot sequence. Unfortunately, good initial estimates of the knot locations are quite difficult to

construct (Lindstrom 1999). Alternatively one may consider the various stepwise and stagewise procedures popular in the statistical literature; however, these are necessarily suboptimal (Chatterjee et al. 1996).

As mentioned at the beginning of this section, the success of genetic algorithms comes from their *adaptability*. By performing a directed yet explorative/exploitive search over the model space, genetic algorithms can provide theoretical convergence to the global optimum without the need for very good initial estimates of the knot locations. However, a note of caution about GAs is in order. Although their convergence to the global optimum has been proven, the choice of the various algorithm parameters - population size, string length, crossover and mutation probabilities - does affect the rate of convergence. As noted previously, the nature of this dependence is problem-specific and only generally understood making the selection of parameter values to achieve the best performance a difficult task. GAs are very computer intensive and hence slower than most existing methods. Finally, different runs of the same GA on a single dataset can lead to different results (although the results are usually reasonable solutions). Hence the nature of the current problem may motivate the development of a GA-based modeling technique, but GAs may not be an appropriate optimization tool in other modeling contexts.

3 CURRENT PROBLEM

Given the mathematical framework presented in Section 1, the objective is to approximate the function f so that a criterion based on the residual sum of the squared errors (RSS) is minimized. Therefore we must define the model space and select a modeling criterion.

3.1 Model Space

Let $\{\tau_i\}_{i=1}^{k+2}$, a set of spline knots, be a strictly increasing sequence of points in $\{x_i\}$ with $\tau_1 = a$, $\tau_{k+2} = b$ and k a positive integer. Let $\{h_i\}_{i=1}^{k+1}$ be a sequence of polynomials of order m . Then the model space may be represented as the space of piecewise polynomials $\mathcal{G}_{m,\tau}$ where, for each $g \in \mathcal{G}_{m,\tau}$,

1. $g(x) = h_i(x)$ if $\tau_i < x < \tau_{i+1}$, $i = 1, \dots, k + 1$,
2. the first $(m - 1)$ derivatives of g at τ_i are continuous, $i = 1, \dots, k + 2$, and
3. $k_{\min} \leq k \leq k_{\max}$ with k_{\min} , k_{\max} known.

For the sake of computational efficiency and to avoid ill-conditioning, each element $g \in \mathcal{G}_{m,\tau}$ will be represented as a linear combination of normalized B-spline basis functions of order m . Thus $g \in \mathcal{G}_{m,\tau}$ is expressed as

$$g = \sum_{j=1}^n \alpha_j B_{j,m,t} \quad \alpha_j \in \mathcal{R}, \quad j = 1, \dots, n,$$

where $n = m + k$, \mathbf{t} is a knot sequence containing the same values as τ (with some boundary knots replicated) and $\{B_{j,m,t}\}_{j=1}^n$ represents the sequence of normalized B-splines of order m with respect to the knot sequence \mathbf{t} . $\mathcal{G}_{m,\tau}$ is now denoted as the spline space $\mathcal{S}_{m,\mathbf{t}}$. For more information on spline spaces, see de Boor (1978).

This choice of model space is motivated by the approximation properties of splines such as the following. Suppose that f is in the Sobolev space $\mathcal{W}_2^q[a, b] = \{f : f^{(0)}, \dots, f^{(q-1)} \text{ absolutely continuous on } [a, b], f^{(q)} \in \mathcal{L}^2[a, b]\}$ and the approximation g to f is defined as the minimizer of

$$\frac{1}{N} \sum_{i=1}^N (y_i - g(x_i))^2 + \alpha \int_a^b (g^{(q)}(x))^2 dx$$

over all $g \in \mathcal{W}_2^q[a, b]$, $\alpha > 0$, $\alpha \in \mathcal{R}$. The approximation g is a spline of order $m = 2q$ with simple knots at the predictor values $\{x_i\}$. Hence, as seen below, the spline models considered by the method presented here use a subset of the basis functions which may be used to construct traditional smoothing splines.

3.2 Model Criterion

The class of criteria from which the fitness function for our algorithm was chosen is the class of goodness-of-fit statistics based on generalized cross-validation (GCV)(Craven and Wahba 1979). These measures are founded upon the idea, borrowed from parametric linear regression, of minimizing the residual sum of squares adjusted for the complexity or amount of fitting being done by the model. Traditionally, the amount of fit associated with a given model is represented by the number of *degrees of freedom* it employs; a popular definition for the degrees of freedom of a spline g is $df = \text{tr}(\mathbf{S})$ where \mathbf{S} is the smoothing matrix of g (Hastie and Tibshirani 1990). Replacing \mathbf{S}_{ii} in the original cross-validation (CV) sum of squares (Stone 1974) by its average value yields the original GCV criterion,

$$GCV(g) = \{RSS_g/N\}/\{(1 - \text{tr}(\mathbf{S})/N)^2\},$$

where RSS_g is the residual sum of squares from the fit of the model g . The definition of degrees of freedom, $df = \text{tr}(\mathbf{S})$, appears in $GCV(g)$ and provides a measure of how much smoothing is being performed by g . If the degrees of freedom

is viewed as a measure of the amount of fit or the cost of the estimation process, then adaptively selected knots should

employ more degrees of freedom relative to nonadaptively selected knots. This, as well as the empirical observation that the flexibility of adaptive splines often leads to substantial overfitting (Ye 1998), motivated the adjusted GCV statistic,

$$GCV(g_N) = \{RSS_{g_N}/N\}/\{(1 - (N_1 + (N_2 * d))/N)^2\},$$

as used in MARS modeling (Friedman 1991), where N_2 of the N basis functions of which g is composed are adaptively selected, $N = N_1 + N_2$, and $d > 1$. Although a GA search does not have a stepwise component, GAs do employ an adaptive procedure for the selection of basis functions. Hence an adjusted GCV criterion will be utilized as our fitness function with $d = 3$ (for the purpose of comparison with existing methods); the actual spline fitting portion of the algorithm will be discussed in Section 4.

If adjusted GCV is to be used in this context, a value for d must be determined. The values suggested in the literature are based on arguments that involve the class of basis functions and the nature of the modeling procedure (Friedman 1991; Luo and Wahba 1997), arguments that cannot be directly applied to the proposed methodology. A possible solution is the generalized degrees of freedom (GDF) of Ye (1998), defined as a measure of the sum of the average sensitivities of each fitted value to its corresponding observed value. In brief, it is calculated using Monte Carlo methods where in each iteration random noise is added to the observed data and the modeling method is applied to the perturbed data set; after a fixed number of iterations, the sum of the slopes of regressing each set of fitted values on the corresponding perturbation values is calculated as the GDF. Although the GDF is perhaps too computationally intensive for standard use, it may be possible to calculate the GDF for our modeling approach on simulated data or in pilot studies and use the resulting information to calibrate the value of d in GCV.

3.3 Convergence

Bhandari et al. (1996) modeled a genetic algorithm with elitist step as a finite state Markov chain and proved, under various conditions, that such a GA will converge to the global optimal solution (i.e., the global optimum of the fitness function) if that solution is contained in the search space. An outline of their results is given below.

Assume a genetic algorithm with finite population size M , string length L and alphabet \mathcal{A} of cardinality a . Let P represent a collection of M strings, i.e., a possible population, and let \mathcal{P} represent the class of all possible populations. The fitness value of a population, $fit(P)$, is defined as $fit(P) = \max_{S \in P} fit(S)$ where the maximum is taken over all strings $S \in P$. \mathcal{P} can be partitioned into sets where each set contains those populations having the same fitness value.

In other words, let $\{F_1, \dots, F_s\}$ represent the set of possible fitness values, $s \leq a^L$, where $F_1 > F_2 > \dots > F_s$. Define

$$E_i = \{P : P \in \mathcal{P} \text{ and } \text{fit}(P) = F_i\} \quad \text{for } i = 1, \dots, s,$$

and let P_{ij} be the j th population of E_i , $j = 1, \dots, e_i$. Denote as $p_{ij.kl}$ the probability that the genetic operators, in one generation, result in a population $P_{kl} \in E_k$ from $P_{ij} \in E_i$ and let $p_{ij.k} = \sum_{l=1}^{e_k} p_{ij.kl}$, $j = 1, \dots, e_i$, $i, k = 1, \dots, s$.

We may then conclude the following.

Theorem 1 [Bhandari (1996)]. If $\min_{i,j} p_{ij.1} > 0$ then

1. $\bigcap_{i=1}^s E_i = \mathcal{P}$, and
2. $\lim_{n \rightarrow \infty} p_{ij.1}^{(n)} = 1 \quad \forall j = 1, \dots, e_i \text{ and } i = 1, \dots, s$ where $p_{ij.1}^{(n)}$ denotes the probability of reaching a population in E_1 in n generations from the starting population P_{ij} .

These results should hold for the specific genetic algorithm used in our modeling approach (see Section 4 below) since it is an elitist GA.

4 ADAPTIVE GENETIC SPLINES (AGS)

The adaptive genetic spline program (AGS) is designed to use a genetic algorithm to determine the model from the space $\mathcal{S}_{m,t}$ which optimizes a model selection criterion based on GCV. Given values within a given range $[k_{\min}, k_{\max}]$ for the number of interior knots k of the spline models under consideration, the genetic algorithm in AGS adaptively selects populations of candidate interior knot sequences from the set of design points $\{x_i\}$ as described in Section 2. We allow the knot sequences \mathbf{t} to be nondecreasing in this implementation although AGS could be redesigned to avoid duplicate knots if necessary. Each candidate knot sequence has a corresponding set of B-spline basis functions of a given order m , $m \in \{1, 2, \dots, 19\}$; the least squares coefficients $\{\alpha_j\}_{j=1}^n$ of these basis functions for the knot sequences in the generated populations are determined by the algorithm L2MAIN of de Boor (1978). At the end of each iteration the splines corresponding to all knot sequences in the current population are evaluated and the minimal RSS knot sequence or spline of all sequences yet considered is retained (*elitist* step); the result of the GA after a fixed number of iterations is the least squares spline model of size k with the smallest RSS. The execution of the genetic algorithm for each given value of k yields a series of models $\{g_k : k \in [k_{\min}, \dots, k_{\max}]\}$; the appropriate model from this group is chosen by minimizing the adjusted GCV score described in Section 3.2. The selection of m is based on knowledge of the data domain and the observation that given an adequately large k the GA will do local order selection. In other

words, by placing duplicate knots at an observation x_i the GA can decrease the number of continuous derivatives at that point and hence effectively lower the order of the spline model in that location. The value of k forces the trade-off between the ability to adaptively select the order m and the rising computational expense of the method as k increases.

Bit encoding was initially considered for AGS but the influence of string length on computational cost was a concern. In some cases the CPU time of a GA has been shown to be linear in the string length (Janikow and Michalewicz 1991), which here depends on both N and k . Since the use and competitive performance of nonbinary codings is supported by theoretical work (Miller and Goldberg 1995; Qi and Palmieri 1994) as well as experimental results (Davis 1991), it was decided that integer coding would be used in AGS. For example, the string (2 5 8 12 16 29) represents a model with 6 interior knots located at x_2 , x_5 , x_8 , x_{12} , x_{16} and x_{29} . This coding scheme leads to relatively shorter strings and is no farther from the domain of the problem than binary coding and hence no more difficult to understand. Since much of the GA literature is based on studies where binary coding was employed, however, the two codings should be compared in future research.

The genetic algorithm with integer coding (or ICGA) that has been implemented uses simple crossover and a simplified linear ranking selection with stochastic sampling with replacement (Miller and Goldberg 1995; see Section 2). Simulation results have shown that the crossover operators designed for real coded genetic algorithms (RCGAs) could lead to improved performance (Herrera et al. 1998). One such operator is BLX- α crossover; the BLX- α crossover of strings S_1 and S_2 described in Section 2.1, for example, would result in a string $S_{1.2} = (\gamma_{1.2,1}, \dots, \gamma_{1.2,i}, \dots, \gamma_{1.2,L})$ where each $\gamma_{1.2,i}$, $i = 1, \dots, L$, was randomly chosen from the interval $[\gamma_{\min} - I \cdot \alpha, \gamma_{\max} - I \cdot \alpha]$, $\gamma_{\min} = \min(\gamma_{1,i}, \gamma_{2,i})$, $\gamma_{\max} = \max(\gamma_{1,i}, \gamma_{2,i})$, $I = (\gamma_{\max} - \gamma_{\min})$. Standard linear ranking selection with stochastic universal sampling (Baker 1985) also looks promising. To describe stochastic universal sampling, imagine a roulette wheel with a space for each string in the population such that the size of each space is directly proportional to the fitness value of the corresponding string. Sampling then involves one spin of the wheel with M equally spaced markers, where M is the population size, and the final position of each marker indicates which string is to be selected. Stochastic universal sampling is one of the most efficient sampling techniques and may reduce the computational expense of AGS.

The works of various authors (Qi and Palmieri 1994; Herrera et al. 1998) support the use of an adaptive mutation scheme, e.g., varying the mutation size and probability during the GAs generations. In AGS a triangular mutation scheme (Murthy 1998) is utilized although nonuniform mutation (Michalewicz 1996) should also perform well. In triangular mutation, for each position or knot of each string in the population a triangular distribution at the current

knot value whose domain length is inversely proportional to the current iteration number. If the knot is chosen for mutation a new knot location is randomly selected from the distribution's domain; the probability of mutation decreases as the algorithm progresses. Note that both the probability and the average size of a mutation decreases as the number of iterations increases. [Similarly, a position selected for nonuniform mutation would be replaced by another randomly selected location such that initially the search space is explored uniformly and in later stages the probability of generating a new position closer to the original position than a random choice is increased].

The mutation probability p_m in AGS varies from 0.5 to $1/L$ (Bäck 1993). An elitist step is included to ensure that the current best solution is retained as the algorithm progresses.

4.1 Simulation studies

Simulated examples were used to examine the performance of AGS compared to the MARS, HAS, wavelet shrinkage and automatic Bayesian model fitting methods. The simulation studies performed here were modeled after those designed by Luo and Wahba (1997) for the HAS program; this was done primarily to facilitate comparison with existing methods. Table 1 gives information about the simulated datasets. Example 1 is taken from Donoho and Johnstone (1994), Example 2 was created by the author, Example 3 is from Schwetlick and Schütze (1995), Examples 4 and 5 are from Fan and Gijbels (1995) while Example 6, which is particularly non-smooth, can be found in Donoho and Johnstone (1994) and Fan and Gijbels (1995) as well as in Denison et al. (1998). To facilitate comparison with wavelet methods, the sample sizes were all powers of 2; Gaussian noise was used for all examples except Example 3 where the noise was uniform and the designs were equally spaced except for Example 2 where the design points were randomly placed.

Table 1. Simulated Examples

Example	f	σ	Sample size (N)	$SD(f)/\sigma$	Number of replicates
1	DJ(1994) Heavisine*2.2	1.0	2048	6.54	31
2	$(4\sqrt{x(1-x)})^{(2.1\pi/(x+0.25))}$	1.0	2048	6.48	31
3	$10(4x-2)/(1+(100*((4x-2)^2)))$	0.06	128	3.33	400
4	$\sin(2(4x-2))+2\exp(-16(4x-2)^2)$	0.3	256	2.80	400
5	$(4x-2)+2\exp(-16(4x-2)^2)$	0.4	256	3.16	400
6	DJ(1994) Bumps	1.0	2048	6.45	31

For wavelet shrinkage the SUREShrink method of Donoho and Johnstone (1995) was selected with a “primary resolution level” of 7 for Example 6 and 5 otherwise. Computations were performed with the *wavethresh* software of

Nason and Silverman (1994) in S-PLUS (Insightful 1996), with the S-PLUS commands provided by Luo and Wahba (1997). The family of wavelets was *DaubLeAsymm* with *filter number* 8. Continuous quadratic piecewise functions were fit by the Bayesian program with the parameter values set as recommended by the authors: 10,000 burn-in iterations, a poisson mean of 10 for Examples 1, 2 and 6 and a poisson mean of 2 otherwise.

The maximum number of basis functions in HAS and MARS was set at 150 for Examples 1 and 2 and at 60 for Examples 3-5. The remaining parameters in HAS and MARS were set at their default values. AGS fit cubic splines for each value of k within the given range: [20, 35] for Example 1, [20, 40] for Example 2, [5, 15] for Examples 3-5, and [50, 70] for Example 6. These choices for k_{\min} and k_{\max} are based on empirical observations and available CPU time; in practice the results of fitting a few preliminary models from a wider range of values could be used in the selection of these parameters. The IDF factor (i.e., the value d from Section 3.2) for AGS was set at 3 for comparison purposes.

In AGS the crossover probability was 0.8 and the same triangle mutation probability density (as described above) was used for all examples. The maximum number of generations was 500 for all examples except Example 6 where the maximum was 5000. The population size M was 50 for Examples 1 and 2, 40 for Examples 3-5 and 100 for Example 6; the choices of M and the number of generations were based on the values of k_{\min} and k_{\max} , the guidelines from the literature and the computational expense. The random number generator used in AGS was the Fortran subroutine RUNI from the package RV (Blue, Kahaner and Marsaglia 1983).

The median MSE and the difference between the 1st and 3rd quartiles of the MSE for each example and each method are given in Table 2; the median results for AGS, HAS and the Bayesian algorithm for Example 2 are shown in Figure 2. For Examples 1 and 3-6, the median results for all methods are shown in Figures 1 and 3-6.

Table 2. Median MSE (Difference between First and Third Quartiles of MSE)

Example	AGS	HAS	SUREShrink	MARS	Bayes
1	.0195 (.0065)	.0412 (.0085)	.0702 (.0397)	.1518 (.0134)	.0698 (.0113)
2	.0411 (.0243)	.0569 (.7784)			9.439 (3.448)
3	.0004 (.0002)	.0006 (.0003)	.0013 (.0003)	.0009 (.0009)	.0006 (.0003)
4	.0052 (.0035)	.0071 (.0059)	.0178 (.0038)	.0070 (.0038)	.0098 (.0039)
5	.0098 (.0062)	.0128 (.0114)	.0464 (.0162)	.0126 (.0068)	.0430 (.0161)
6	.4001 (.1739)	2.054 (.3410)	.8670 (1.403)	20.60 (.0700)	17.41 (.3500)

The performance of AGS compares quite favorably with the results of SUREShrink and the other adaptive modeling algorithms. On all examples AGS achieved the lowest median MSE, most likely due to the directed global selection of basis functions. For Examples 1, 4, 5 and 6 the results of HAS, SUREShrink and MARS reflect those shown in Luo and Wahba (1997). The selection of a lower “primary resolution level” for SUREShrink did not yield better performance.

MARS failed to give reasonable results for Example 2 perhaps due to its design for high dimensional problems and not for extremely non-smooth functions. Wavelet results are not shown for this example because the observations are not equally spaced.

The above results for AGS were achieved using the default value of $d = 3$ although subsequent experiments were run on each dataset with values of d ranging from 1 to 5. It was observed that the result of AGS was relatively insensitive to values of d between 2 and 4; values outside of this range lead to either undersmoothing ($d < 2$) or oversmoothing ($d > 4$).

Despite the relatively superior performance of AGS on the above examples several issues have yet to be resolved.

- Due to the global nature of the GA search, it is relatively slow and computationally expensive. In Table 3 the cost per replicate in CPU time for Example 4, Example 6 (Bumps) and Example 7 (described in the next section) on a HP-UX 10.2 system with 180 Mhz processor and 512 MB of RAM have been provided.

Table 3. Computational Expense in System Time (sec.) per Replicate

Example	Sample size (N)	AGS	HAS	SUREShrink	MARS	Bayes
4	256	25.0	0.4	0.13	0.2	20.0
6	2048	10.0	0.9	0.16	0.3	20.0
7	440	10.0	0.4	0.11	0.2	20.0

The expense of AGS on this system is less than the Bayesian approach on average but considerably larger than the expenses of the other three methods. This limits the number of model sizes which can be considered. Hopefully its efficiency can be improved by the use of ICGA/RCGA specific genetic operators (as mentioned above) and improved programming.

- The above disparity in computational expense raises the question, yet unanswered, of whether fairer comparisons could be made by restricting all of the methods to a fixed amount of computational resources or number of iterations instead of running all methods to ‘convergence’.
- Crossover and mutation can create models with decreasing knot sequences which then must be sorted. It is possible that this could be avoided and hence the size of the search space decreased by incorporating constraints into the GA model as in Michalewicz and Janikow (1991).

4.2 Real data application

Functional Magnetic Resonance Imaging (fMRI) is a technique used in the biosciences by which properties of the blood are exploited to create high-speed, high-resolution images of biological processes. In studies of brain processes, fMRI is often used to create a time series of high-resolution, 3-dimensional images of a subject's brain while some activity is being performed. Each 3-dimensional image is composed of a series of 2-dimensional images or 'slices' of the brain which can itself be treated as a time series in which a particular image i represents the activity or intensities of the pixels $\{(x, y)\}$ at time t . The intensity of a particular pixel (x, y) is a continuous random variable whose values across the 'slices' generate a 1-dimensional time series. In most experiments interest is focused on pixels in a particular region believed to be relevant to the task at hand; we will focus on one such pixel in the first 3-dimensional image of an experiment previously described and studied by Sanil (1998, Chapter 5). In this experiment the subject was asked to alternate two tasks over the duration of the experiment and 22 equally spaced images were taken during the performance of each task. The result was a series of 440 images in which images 1-22 were during task 1, 23-44 were during task 2, 45-66 were during task 1, etc.

The time series of length 440 is shown in Figure 7(a) and the same dataset divided into sections corresponding to the performance of each task is shown in Figure 7(b). We notice an overall downward trend and a large amount of noise. The experimental conditions suggest that there may be a periodic or sinusoidal component to the series, but we will assume no such knowledge here. The default parameter values described in Section 4.1 were used except in the Bayesian fitting (poisson mean = 10, number of replicates = 100) and in AGS (range of bases = [10, 30], maximum number of generations = 5000, population size $M = 100$, number of replicates = 100). Since 440 is not a power of 2, the wavelet method was applied to only the first 256 observations.

As Figures 7(c)-(g) clearly indicate, the amount of smoothing performed by the methods varies widely. The AGS and HAS models have maintained some of the bumpy character of the data; MARS and the Bayesian fit have done relatively more smoothing while the wavelet method has done relatively less smoothing. I find the HAS fit to be most satisfactory, followed by the AGS and wavelet fits, but this quality assessment is purely subjective. It is possible that a larger value of d would be appropriate for the AGS method in this context. Although this is a single example, it may provide the reader with some information regarding how AGS performs in the presence of noise that may not be *i.i.d.*.

5 SUMMARY

A modeling technique has been proposed for fitting adaptive splines. The basis functions are B-spline basis functions of a given order and the method can be used to fit adaptive spline models of various sizes which minimize an adjusted GCV

criterion. For each candidate model size, the search for the minimum RSS model is performed by a genetic algorithm which uses the least squares coefficients provided by de Boor (1978) and the model criterion in adaptively selecting the appropriate knot sequence.

Many interesting problems involve data sets with more than one predictor variable. We would like to explore the use of GAs for fitting multidimensional surfaces; currently the most feasible approach for extension to higher dimensional data appears to be through additive modeling, as was done by Rogers (1991) with the MARS algorithm. Another option is to use hyperplanes in our modeling, an extension of fitting linear splines. Each string would represent an individual solution to the problem, i.e., a k -piecewise hyperplane in dim dimensions, $dim > 1$. Further research is necessary before further comments can be made regarding the relative feasibility of these options.

The choice of inflation factor for the GCV criterion needs to be determined, as mentioned in Section 3.2, and criteria other than GCV should be considered. It is well known that least squares methods have problems with outliers so it is of interest to extend AGS to robust fitting. For example, the GCV criterion might be exchanged for one based on least median squares or an M -estimator as has been discussed in the context of GAs and splines (Karr and Weck 1995; Lenth 1977).

Apart from statistical issues, the issue of integer versus binary coding in AGS has been left unresolved as it has been in the computer science literature. Further experiments may show that an alternate coding is best for this particular application. It is also recognized that further experimentation with AGS is necessary, in particular (1) imposing limitations on time and the number of iterations, (2) involving other search methods and (3) providing error estimates at various sample sizes as in Zhou, Shen and Wolfe (1998).

REFERENCES

- Bäck, T. (1993), “Optimal mutation rates in genetic search”, in Forrest, S. (Ed.), *Proceedings of the Fifth International Conference on Genetic Algorithms*, San Mateo, CA: Morgan Kaufmann Publishers, pp. 2–8.
- Baker, J. (1985), “Adaptive selection methods for genetic algorithms”, in Grefenstette, J. (ed.), *Proceedings of an International Conference on Genetic Algorithms and their Applications*, Hillsdale, NJ.: L. Erlbaum Associates, pp. 101–111.
- Bangalore, A., Shaffer, R., and Small, G. (1996), “Genetic algorithm-based method for selecting wavelengths and model size for use with partial least-squares regression: Application to near-infrared spectroscopy”, *Analytical Chemistry*, 68, pp. 4200–4212.
- Bhandari, D., Murthy, C., and Pal, S. (1996), “Genetic algorithm with elitist model and its convergence”, *International Journal of Pattern Recognition and Artificial Intelligence*, 10, 6, pp. 731–747.
- Blue, J., Kahaner, D., and Marsaglia, G. (1983), RV package, National Bureau of Standards and Washington State University.
- Broadhurst, D., Goodacre, R., Jones, A., Rowland, J., and Kell, D. (1997), “Genetic algorithms as a method for variable selection in multiple linear regression and partial least squares regression, with applications to pyrolysis mass spectrometry”, *Analytica Chimica Acta*, 348, pp. 71–86.
- Biller, C. (1999), “Adaptive Bayesian regression splines in semiparametric generalized linear models”, *Journal of Computational and Graphical Statistics*, 9, 1, pp. 122–140.
- Chatterjee, S., Laudato, M., and Lynch, L. (1996), “Genetic algorithms and their statistical applications: an introduction”, *Computational Statistics and Data Analysis*, 22, 6, pp. 633–651.
- Cho, S., Cummins, D., Bentley, J., Andrews, C., and Tropsha, A. (1996), “‘Back’ to 2D QSAR: Application of genetic algorithms and partial least squares to variable selection of topological indices”, *Journal of Computer-Aided Molecular Design*, submitted.
- Craven, P., and Wahba, G. (1979), “Smoothing noisy data with spline functions”, *Numerische Mathematik*, 31, pp. 377–403.
- Davis, L. (ed.) (1991), *Handbook of genetic algorithms*, 1st Edition. New York: Van Nostrand Reinhold.
- De Boor, C. (1978), *A practical guide to splines*, 1st Edition. New York: Springer-Verlag.
- DeJong, K. (1975), *An Analysis of the behavior of a class of genetic adaptive systems*, PhD thesis, University of Michigan, Ann Arbor, MI.
- Denison, D., Mallick, B., and Smith, A. (1998), “Automatic Bayesian curve fitting”, *Journal of the Royal Statistical Society, Series B*, 60, pp. 333–350.
- Donoho, D., and Johnstone, I. (1994), “Ideal spatial adaptation by wavelet shrinkage”, *Biometrika*, 81, pp. 425–455.
- (1995), “Adapting to unknown smoothness via wavelet shrinkage”, *Journal of the American Statistical Association*, 90, pp. 1200–1224.
- Drijkoningen, G., and White, R. (1995), “Seismic velocity structure of oceanic crust by inversion using genetic algorithms”, *Geophysical Journal International*, 123, pp. 653–664.
- Fan, J., and Gijbels, I. (1995), “Data-driven bandwidth selection in local polynomial fitting: Variable bandwidth and spatial adaptation”, *Journal of the Royal Statistical Society, Ser. B.*, 57, pp. 371–394.
- Friedman, J. (1991), “Multivariate adaptive regression splines (with discussion)”, *Annals of Statistics*, 19, pp. 1–141.

- Geyer, C., and Thompson, E. (1995), "Annealing Markov chain Monte Carlo with applications to ancestral inference", *Journal of the American Statistical Association*, 90, pp. 909–920.
- Goldberg, D. (1991), "Real-coded genetic algorithms, virtual alphabets, and blocking", *Complex Systems*, 5, pp. 139–167.
- Green, P. (1995), "Reversible jump markov chain monte carlo computation and bayesian model determination", *Biometrika*, 82, 4, pp. 711–732.
- Hansen, M., and Kooperberg, C. (1999), "Spline adaptation in extended linear models", *Statistical Science*, tentatively accepted.
- Hastie, T., and Tibshirani, R. (1990), *Generalized Additive Models*. London: Chapman and Hall.
- Herrera, F., Lozano, M., and Verdegay, J. (1998), "Tackling real-coded genetic algorithms: Operators and tools for behavioral analysis", *Artificial Intelligence Review*, 12, pp. 265–319.
- Holland, J. (1975), *Adaptation in natural and artificial systems*. Ann Arbor, MI: The University of Michigan Press.
- Holmes, C., and Mallick, B. (1998), "Parallel Markov chain Monte Carlo sampling", unpublished manuscript.
- Huang, J., Liu, C., and Wechsler, H. (1998), "Evolutionary computation and face recognition", in Wechsler, H., Phillips, J. P., Bruce, V., Fogelman - Soulie, F., and Huang, T. (eds.), *Face Recognition : From Theory to Applications*. New York: Springer-Verlag.
- Inightful, Inc. (1996), *Splus 3.4* (Release 1), Seattle, WA.
- Janikow, C., and Michalewicz, Z. (1991), "An experimental comparison of binary and floating point representation in GA", in Belew, R. and Booker, L. (eds.), *Proc. of Fourth Int. Conf. on Genetic Algorithms*, San Mateo, CA: Morgan Kaufmann Publishers, pp. 31–36.
- Jupp, D. (1978), "Approximation to data by splines with free knots", *SIAM Journal of Numerical Analysis*, 15, pp. 328–343.
- Karr, C., and Weck, B. (1995), "Improved computer modelling using least median squares curve fitting and genetic algorithms", *Fluid/Practice Separation Journal*, 8, 2, pp. 117–124.
- Kokkonis, P., and Leute, V. (1996), "Least squares splines approximation applied to multicomponent diffusion data", *Computational Materials Science*, 6, 1, pp. 103–111.
- Lankhorst, M., and van der Laan, M. (1994), "Wavelet-based signal approximation with genetic algorithms". Technical Report CS-R 9409, Department of Mathematics and Computing Science, University of Groningen, The Netherlands.
- Lenth, R. (1977), "A robust method for multiple linear regression", *Communications in Statistics A*, 6, pp. 847–854.
- Li, T-H., Lucasius, C., and Katerman, G. (1992), "Optimization of calibration data with the dynamic genetic algorithm", *Analytica Chimica Acta*, 2768, pp. 123–134.
- Lindstrom, M. (1999), "Penalized estimation of free-knot splines", *Journal of Computational and Graphical Statistics*, 8, 2, pp. 333–352.
- Luo, Z., and Wahba, G. (1997), "Hybrid adaptive splines", *Journal of the American Statistical Association*, 92, pp. 107–115.
- Manela, M., Thornhill, N., and Campbell, J., "Fitting spline functions to noisy data using a GA", in S. Forrest (ed.), *Proceedings of the 5th International Conference on Genetic Algorithms*, San Mateo, CA: Morgan Kaufmann, 1993.
- Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller, E. (1953), "Equation of state calculations by fast computing machines", *Journal of Computational Physics*, 21, pp. 1087–1092.
- Meyer, M. (Ed.)(1994), *CMLIB*, Carnegie Mellon University, Pittsburgh, PA.

- Michalewicz, Z. (1996), *Genetic algorithms + data structures = evolution programs*. New York: Springer-Verlag.
- Michalewicz, Z., and Janikow, C. (1991), "Handling constraints in genetic algorithms", in Belew, R. and Booker, L.B. (eds.), *Proc. of Fourth Int. Conf. on Genetic Algorithms*, San Mateo, CA: Morgan Kaufmann Publishers, pp. 151–157.
- Miller, B., and Goldberg, D. (1995), "Genetic algorithms, selection schemes, and the varying effects of noise". Technical Report No. 95009, Univeristy of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory.
- Murthy, C. (1998), personal communication.
- Nason, G., and Silverman, B. (1994), "The discrete wavelet transform in S", *Journal of Computational and Graphical Statistics*, 3, pp. 163–191.
- Qi, X., and Palmieri, F. (1994), "Theoretical analysis of evolutionary algorithms with an infinite population size in continuous space Part I: Basic properties of selection and mutation", *IEEE Transactions on Neural Networks*, 5, 1, pp. 102–119.
- Rogers, D. (1991), "G/SPLINES: A hybrid of Friedman's Multivariate Adaptive Regression Splines (MARS) Algorithm with Holland's Genetic Algorithm", in Belew, R.K., and Hooker, L.B. (eds.), *Proceedings of the Fourth International Conference on Genetic Algorithms*, San Mateo, CA: Morgan Kaufmann.
- Rogers, D., and Hopfinger, A. (1994), "Application of genetic function approximation to quantitative structure-activity relationships and quantitative structure-property relationships", *Journal of Chemical Information and Computer Sciences*, 34, pp. 854–866.
- Sanil, A. (1998), *An approach for selection of nonparametric regression methods*, PhD thesis, Carnegie Mellon University, Pittsburgh, PA.
- Schwetlick, H., and Schütze, T. (1995), "Least squares approximation by splines with free knots", *BIT.Numerical Mathematics*, 35, pp. 361-384.
- Shively, T., Kohn, R., and Wood, S. (1999), "Variable selection and function estimation in additive nonparametric regression using a data-based prior (with discussion)", *Journal of the American Statistical Association*, 94, 447, pp. 777–807.
- Smith, M., and Kohn, R. (1996), "Nonparametric regression using bayesian variable selection", *Journal of Econometrics*, 75, pp. 317–344.
- Stone, C. (1974), "Cross-validatory choice and assessment of statistical predictions (with discussion)", *Journal of the Royal Statistical Society, Series B*, 36, pp. 111–147.
- Stone, C., Hansen, M., Kooperberg, C., and Troung, Y. (1997), "Polynomial splines and their tensor products in extended linear modeling (with discussion)", *Annals of Statistics*, 25, 4, pp. 1371–1470.
- Wahba, G. (1988), in Discussion to "Monotone regression splines in action", by J. Ramsay, *Statistical Science*, 3, pp. 425–462.
- Whitehead, B., and Choate, T. (1996), "Cooperative - Competitive genetic evolution of radial basis function centers and widths for time series prediction", *IEEE Transactions on Neural Networks*, 7, 4, pp. 869–880.
- Wise, B. (1994), "Alternatives to neural networks: genetic algorithms and non-linear biased regression", in Keller, P. (ed.), *Proceedings of the Neural Network Workshop for the Hanford Community*, Richland, WA: Pacific Northwest National Laboratory.
- Ye, J. (1998), "On measuring and correcting the effects of data mining and model selection", *Journal of the American Statistical Association*, 93, 441, pp. 120–131.
- Zhou, S., Shen, X., and Wolfe, D. (1998), "Local asymptotics for regression splines and confidence regions", *The Annals of Statistics*, 26, 5, pp. 1760–1782.

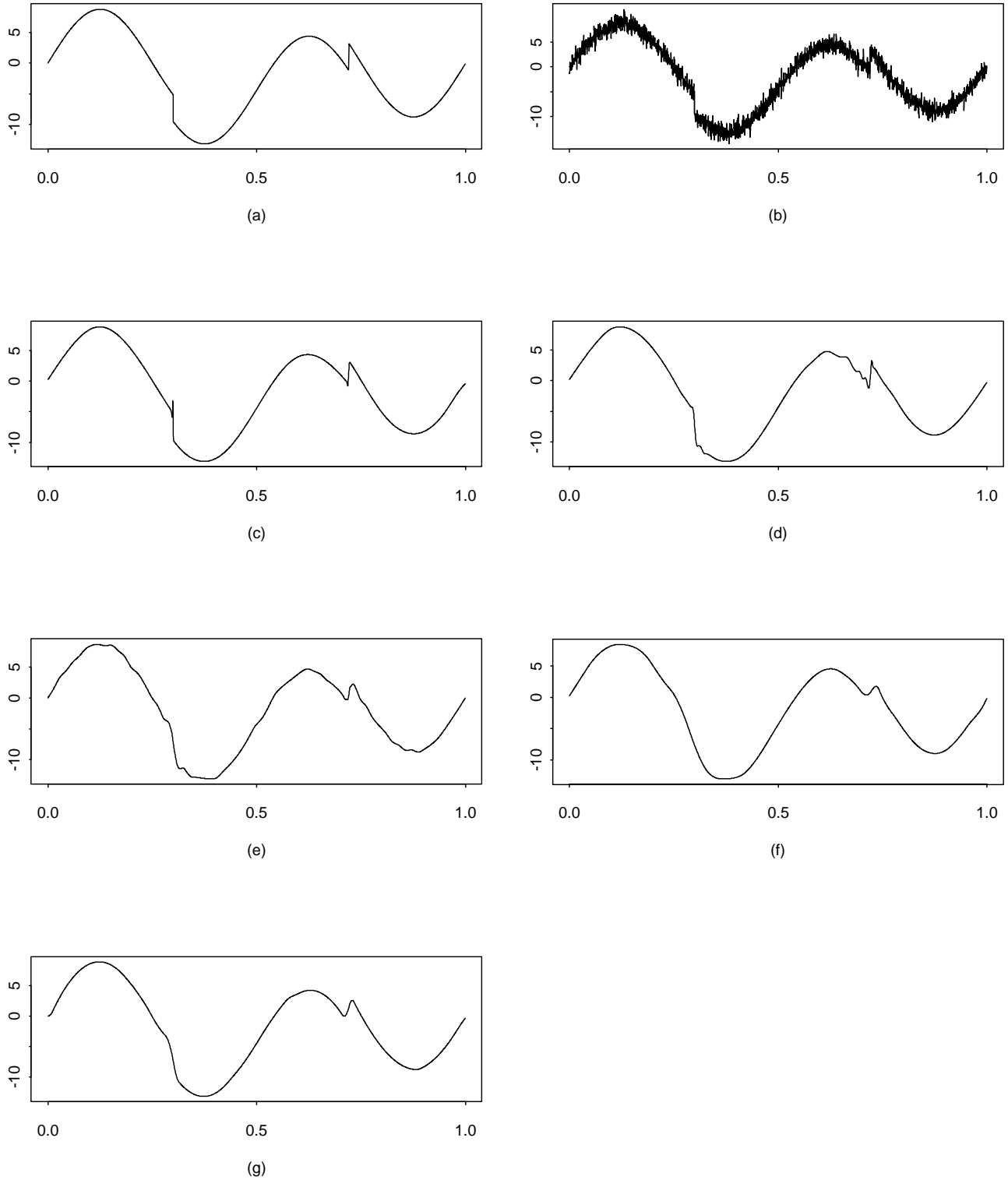


Figure 1. Example 1:(a) original function; (b) sample dataset of 2,048 observations from original function with added Gaussian ($\sigma=1.0$) noise; (c) AGS fit with median MSE (= .0195); (d) HAS fit with median MSE (= .0412); (e) SUREShrink fit with median MSE (= .0397); (f) MARS fit with median MSE (= .1518); (g) Bayes fit with median MSE (= .0698)

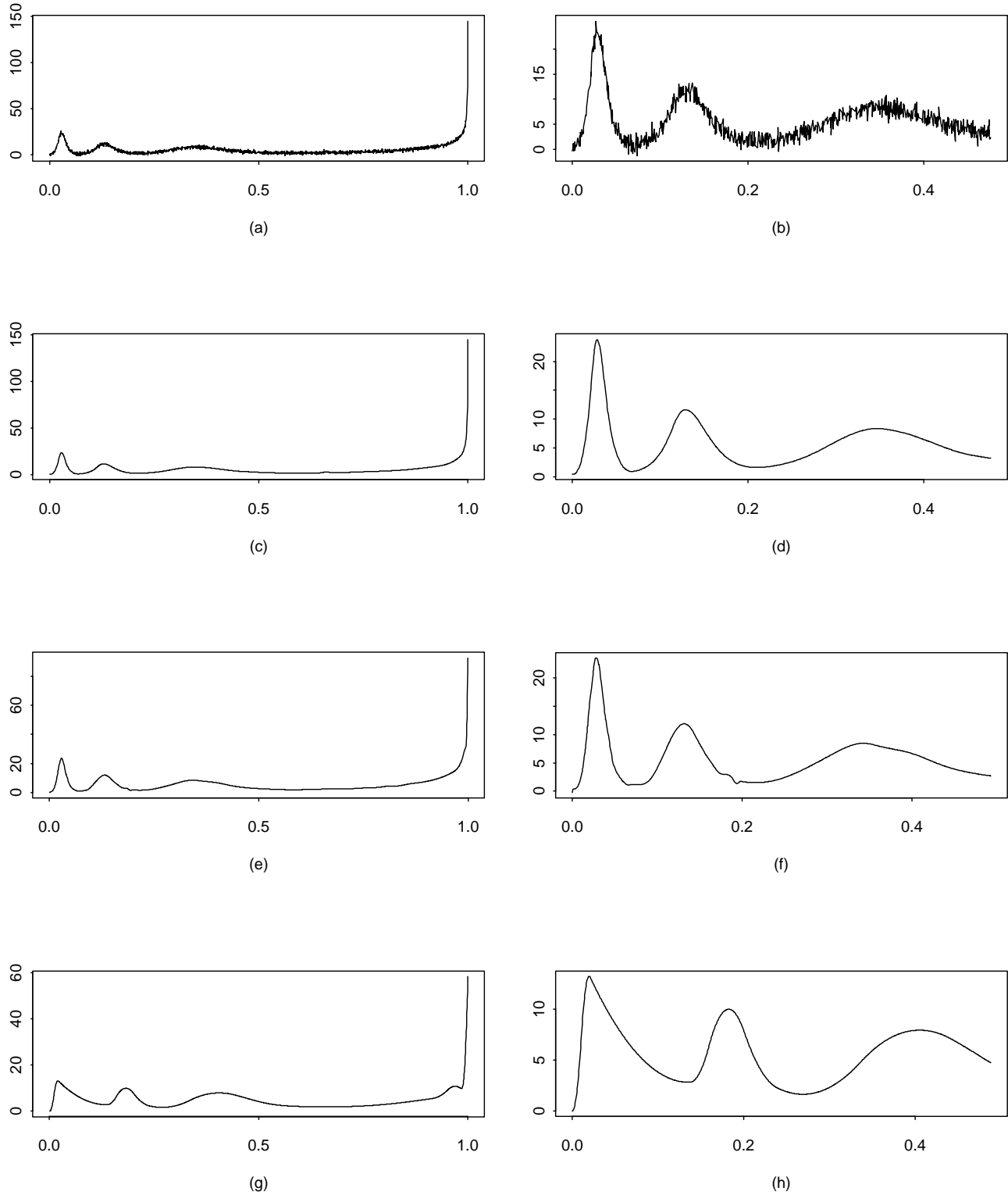


Figure 2. Example 2:(a) sample dataset of 2,048 observations from original function with added Uniform ($\sigma=1.0$) noise; (b) AGS fit with median MSE ($= .0411$); (c) HAS fit with median MSE ($= .0569$); (d) sample dataset on $[0, 0.5]$; (e) AGS fit with median MSE on $[0, 0.5]$; (f) HAS fit with median MSE on $[0, 0.5]$; (g) Bayes fit with median MSE ($= 9.439$); (h) Bayes fit with median MSE on $[0, 0.5]$

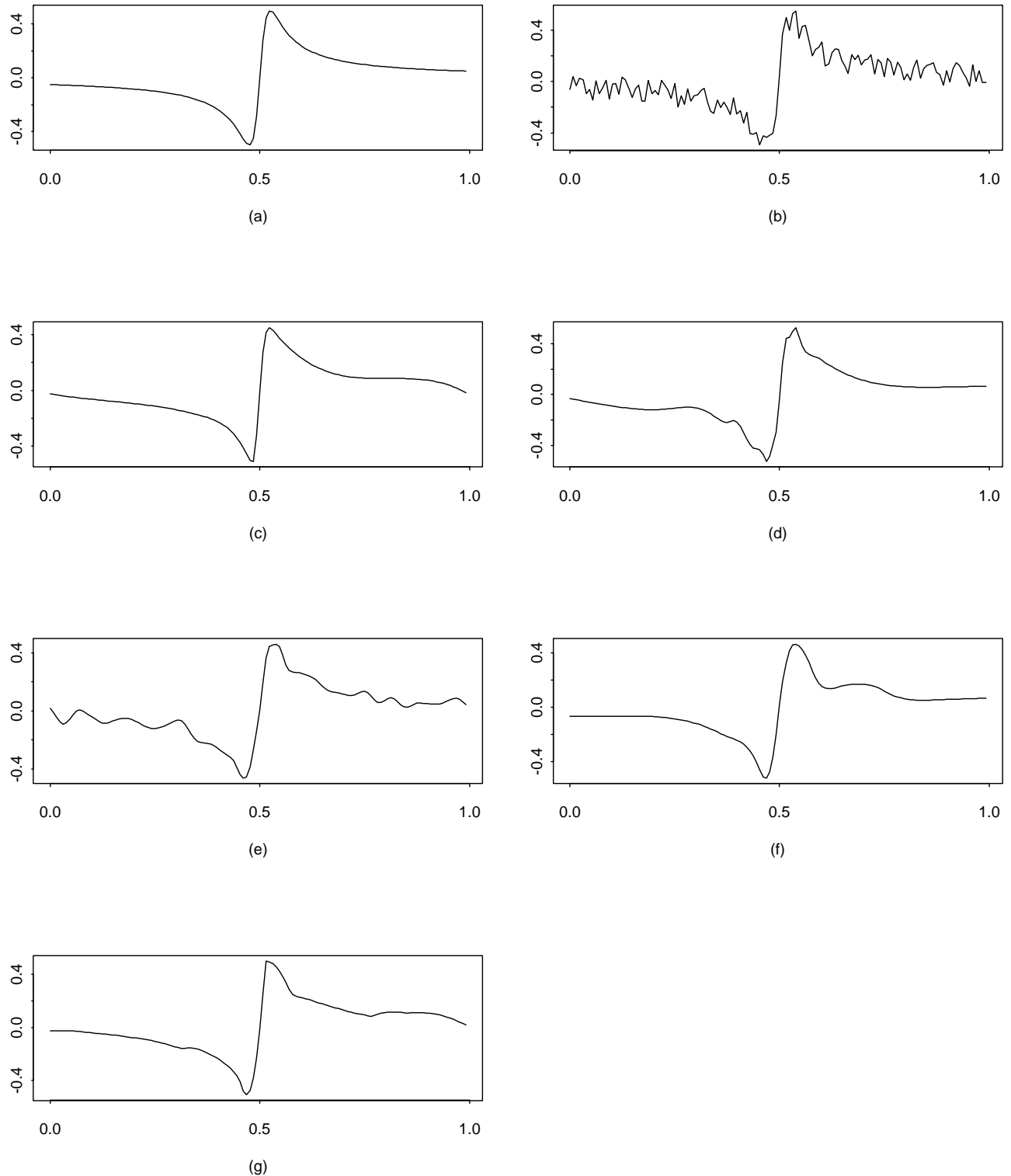


Figure 3. Example 3: (a) original function; (b) sample dataset of 400 observations from original function with added Gaussian ($\sigma=0.06$) noise; (c) AGS fit with median MSE ($= .0004$); (d) HAS fit with median MSE ($= .0006$); (e) SUREShrink fit with median MSE ($= .0013$); (f) MARS fit with median MSE ($= .0009$); (g) Bayes fit with median MSE ($= .0006$)

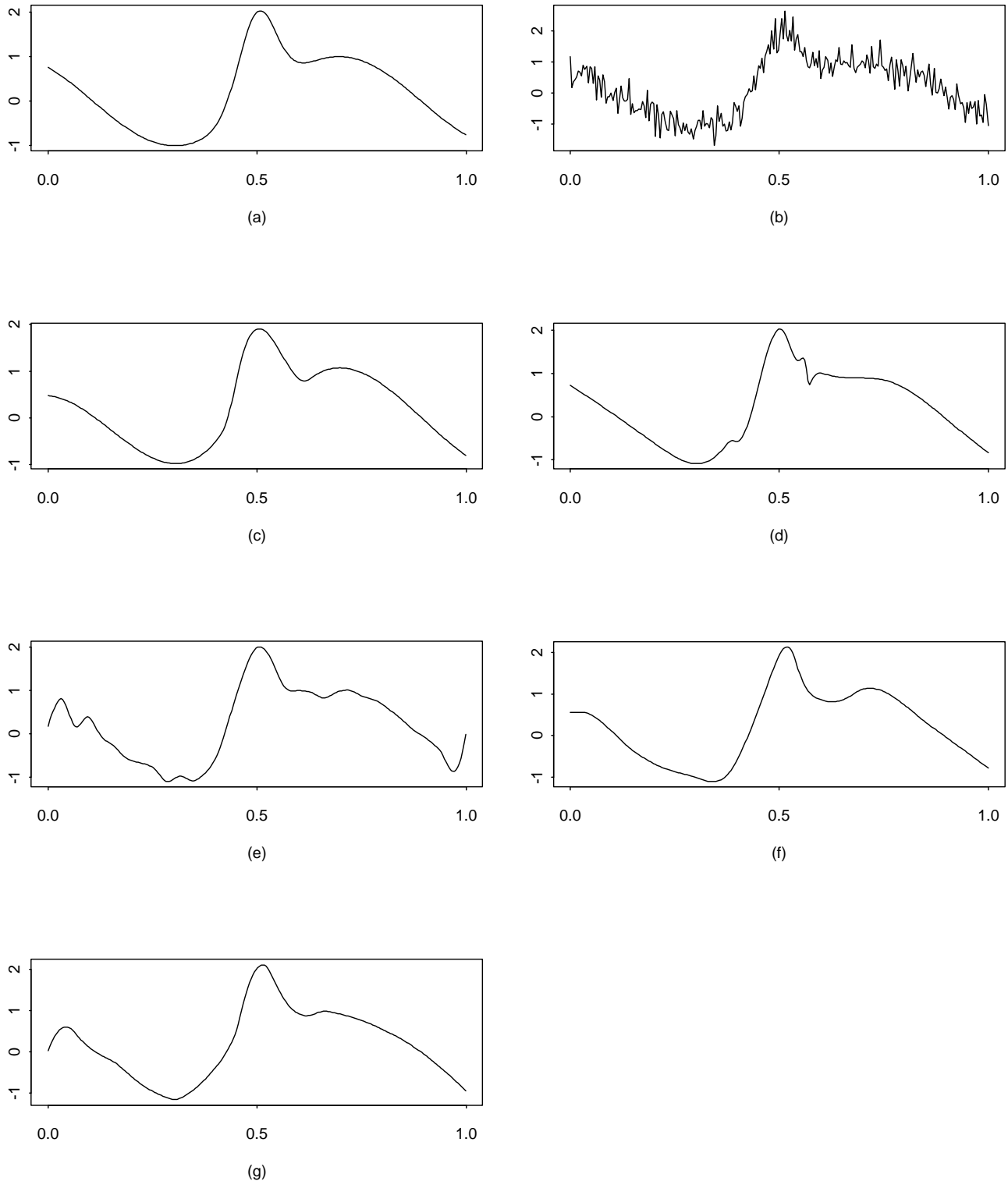


Figure 4. Example 4: (a) original function; (b) sample dataset of 400 observations from original function with added Gaussian ($\sigma=0.3$) noise; (c) AGS fit with median MSE (= .0052); (d) HAS fit with median MSE (= .0071); (e) SUREShrink fit with median MSE (= .0178); (f) MARS fit with median MSE (= .0070); (g) Bayes fit with median MSE (= .0098)

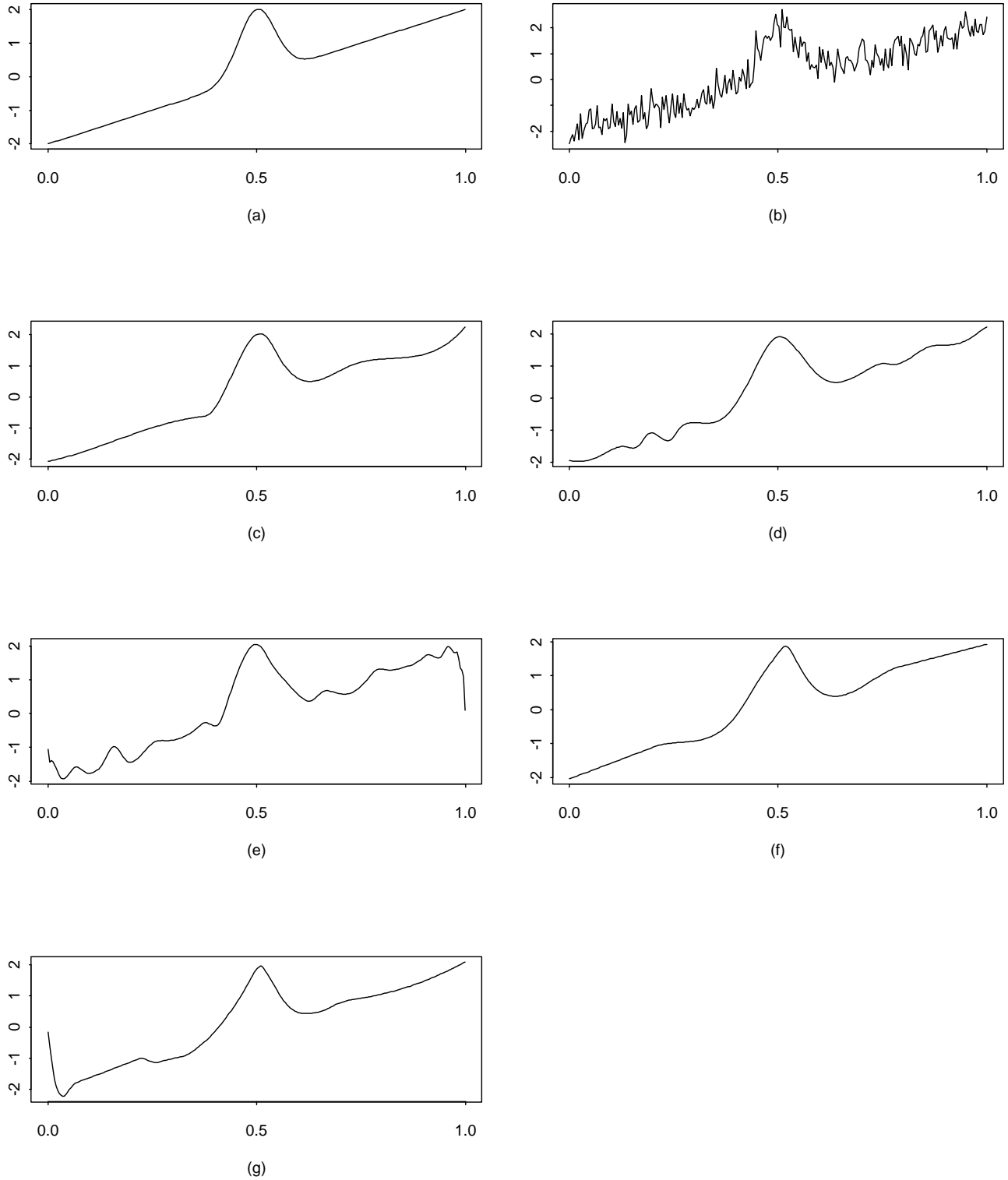


Figure 5. Example 5: (a) original function; (b) sample dataset of 400 observations from original function with added Gaussian ($\sigma=0.4$) noise; (c) AGS fit with median MSE ($= .0098$); (d) HAS fit with median MSE ($= .0128$); (e) SUREShrink fit with median MSE ($= .0464$); (f) MARS fit with median MSE ($= .0126$); (g) Bayes fit with median MSE ($= .0430$)

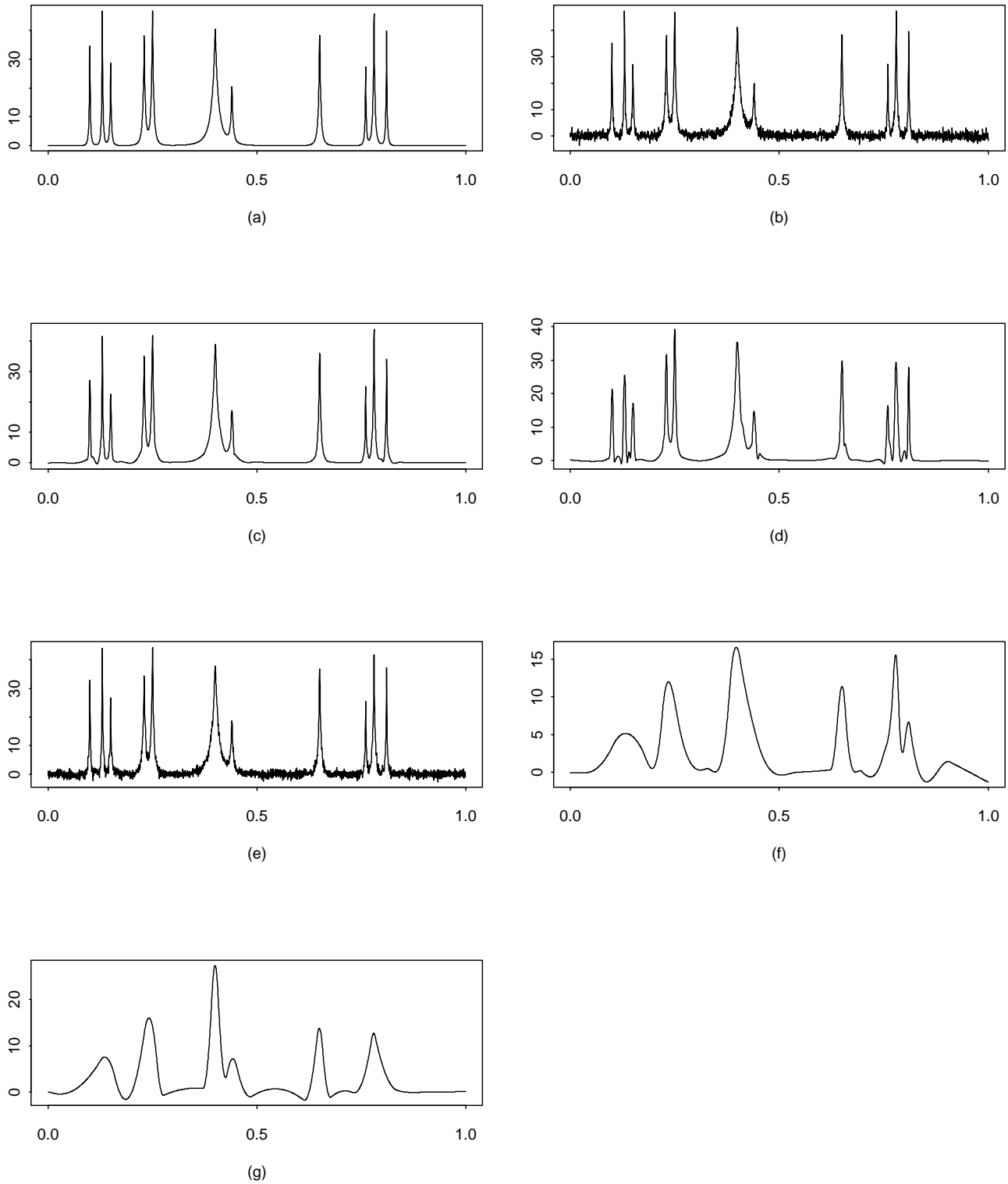


Figure 6. Example 6: (a) original function; (b) sample dataset of 31 observations from original function with added Gaussian ($\sigma=1.0$) noise; (c) AGS fit with median MSE ($= .4001$); (d) HAS fit with median MSE ($= 2.054$); (e) SUREShrink fit with median MSE ($= .8670$); (f) MARS fit with median MSE ($= 20.60$); (g) Bayes fit with median MSE ($= 17.41$)

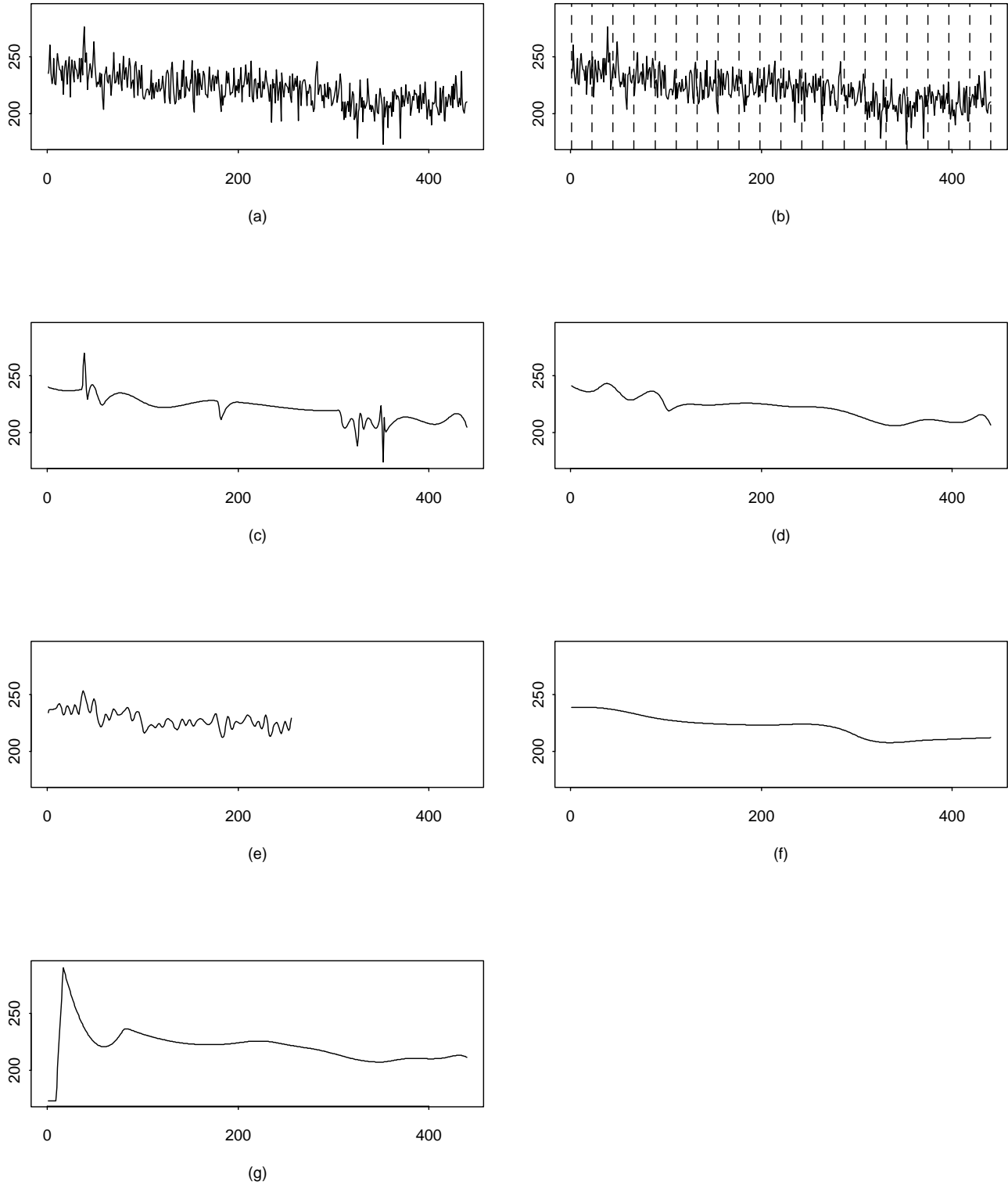


Figure 7. fMRI Example: (a) original data; (b) original data with task lines; (c) AGS fit with median MSE (= 93.56); (d) HAS fit with median MSE (= 107.9); (e) SUREShrink fit on first 256 observations with median MSE (= 91.31); (f) MARS fit with median MSE (= 187.8); (g) Bayes fit with median MSE (= 269.4)