Statistica Sinica **12**(2002), 361-386

REGRESSION TREES WITH UNBIASED VARIABLE SELECTION AND INTERACTION DETECTION

Wei-Yin Loh

University of Wisconsin–Madison

Abstract: We propose an algorithm for regression tree construction called GUIDE. It is specifically designed to eliminate variable selection bias, a problem that can undermine the reliability of inferences from a tree structure. GUIDE controls bias by employing chi-square analysis of residuals and bootstrap calibration of significance probabilities. This approach allows fast computation speed, natural extension to data sets with categorical variables, and direct detection of local two-variable interactions. Previous algorithms are not unbiased and are insensitive to local interactions during split selection. The speed of GUIDE enables two further enhancements—complex modeling at the terminal nodes, such as polynomial or best simple linear models, and bagging. In an experiment with real data sets, the prediction mean square error of the piecewise constant GUIDE model is within ±20% of that of CART®. Piecewise linear GUIDE models are more accurate; with bagging they can outperform the spline-based MARS® method.

Key words and phrases: Bagging, bias correction, bootstrap, interaction detection, piecewise linear.

1. Introduction

A regression tree is a piecewise constant or piecewise linear estimate of a regression function, constructed by recursively partitioning the data and sample space. Its name derives from the practice of displaying the partitions as a decision tree, from which the roles of the predictor variables may be inferred. The AID algorithm (Morgan and Sonquist (1963), Fielding (1977)) is the first implementation of this idea. It searches over all axis-orthogonal partitions and yields a piecewise constant estimate. At each stage, the binary partition that minimizes the total sum of the squared errors (SSE) is selected. Splitting stops if the fractional decrease in total SSE is less than a pre-specified value γ or if the sample size is too small.

One weakness of AID is that it is hard to specify γ. Too small or too large a value leads to over- or under-fitting, respectively. Another weakness is that the greedy search approach induces a bias in variable selection (Doyle (1973)). Specifically, an ordered predictor with n distinct values gives rise to $n-1$ binary

splits of the data. Suppose X_1 and X_2 are two ordered predictors with n_1 and n_2 distinct values, respectively, with $n_1 \gg n_2$. All other things being equal, X_1 will have a higher chance to be selected than X_2. On the other hand, if X_1 is a categorical variable taking n distinct values, there are $2^{n-1}-1$ binary splits. Since this number grows exponentially with n, there is also a selection bias toward categorical variables that take many values. Obviously, the bias can lead to erroneous inferences from the tree structure.

The CART® (Breiman, Friedman, Olshen and Stone (1984)) algorithm avoids the difficulty of choosing γ by employing a backward-elimination strategy to determine the tree. It grows an overly large tree and then prunes away some branches, using a test sample or cross-validation (CV) to estimate the total SSE. It has the same problem with selection bias because it uses the greedy search approach of AID.

Other methods have been proposed for determining the final tree: Ciampi, Hogg, McKinney and Thiffault (1988) and Ciampi, Lou, Lin and Negassa (1991) combine non-adjacent partitions; Chaudhuri, Huang, Loh and Yao (1994) use a CV-based look-ahead procedure; Marshall (1995) finds non-hierarchical partitions; Chipman, George and McCulloch (1998) and Denison, Mallick and Smith (1998) employ Bayesian methods to search among trees; Li, Lue and Chen (2000) use a stopping rule based on statistical significance tests.

The FIRM (Kass (1975); Hawkins (1997)) method addresses the bias problem by using Bonferroni-adjusted significance tests to select predictors for splitting. Unlike AID and CART which yield binary splits, FIRM splits each node into as many as ten subnodes for an ordered predictor, and c subnodes for a c-category predictor. Hawkins ((1997), p.17) mentions that the Bonferroni adjustment can over-correct, resulting in a bias toward predictors that allow fewer splits.

As illustration, consider the baseball data from `Statlib` (`http://lib.stat.cmu.edu`) on 1987 salaries and twenty-two other variables for two hundred sixty-three professional baseball players (Table 1). Many regression tree models have been proposed for mean log-salary as a function of the other variables (Conlon and Meyer (1988); Chaudhuri, Huang, Loh and Yao (1994); Li, Lue and Chen (2000)). Figure 1 shows a piecewise constant CART model. It splits on seven predictors. Hoaglin and Velleman (1995) find that a continuous three-segment linear regression model of the form

$$\texttt{LogSal} = \beta_0+\beta_1\texttt{Runcr}/\texttt{Yrs}+\beta_2\sqrt{\texttt{Run86}}+\beta_3 \min[(\texttt{Yrs}-2)_+, 5]+\beta_4(\texttt{Yrs}-7)_+ \quad (1)$$

fits the data quite well. They criticize regression tree and other automated methods for generating "complex models that fit poorly and offered no insight" (Hoaglin and Velleman (1995), p.284).

Table 1. Predictor variables for baseball data. The response variable is natural log of 1987 salary in thousands of dollars.

`Bat86`	# times at bat in 1986	`Batcr`	# times at bat during career
`Hit86`	# hits in 1986	`Hitcr`	# hits during career
`Hr86`	# home runs in 1986	`Hrcr`	# home runs during career
`Run86`	# runs in 1986	`Runcr`	# runs during career
`Rb86`	# runs batted in in 1986	`Rbcr`	# runs batted in during career
`Wlk86`	# walks in 1986	`Wlkcr`	# walks during career
`Leag86`	league at end of 1986 (2 cat.)	`Leag87`	league at start of 1987 (2 cat.)
`Team86`	team at end of 1986 (24 cat.)	`Team87`	team at start of 1987 (24 cat.)
`Div86`	division at end of 1986 (2 cat.)	`Yrs`	# years in the major leagues
`Pos86`	position in 1986 (23 cat.)	`Puto86`	# put outs in 1986
`Asst86`	# assists in 1986	`Err86`	# errors in 1986

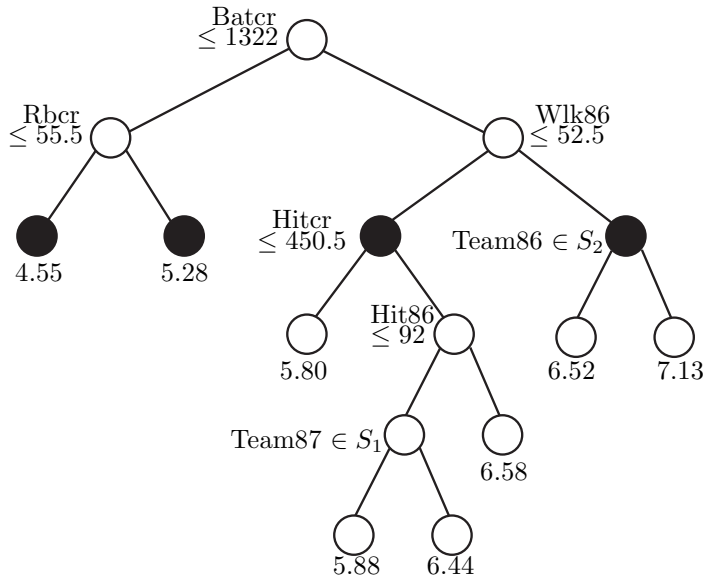

Figure 1. Piecewise constant 0-SE CART tree for baseball data. Terminal nodes for the 1-SE tree are colored black. At a split, a case goes to the left node if it satisfies the stated condition; otherwise it goes to the right. S_1 and S_2 denote sets of baseball teams. The number beneath each terminal node is the sample mean of log-salary.

Although none of the predictors in Figure 1 appear in (1), some of the splits make sense—high values of performance measures are associated with high salaries. The splits on `Team86` and `Team87` are harder to explain and could be due to selection bias. Nevertheless, the highly correlated nature of some of the

predictors suggest that it is possible to simplify the tree structure by fitting a linear model instead of a constant at each node.

An intrinsic difficulty in extending greedy search to piecewise multiple linear regression models is the substantial increase in computational complexity: for each split of a node, a linear model must be fitted to each of the subnodes. For example, the variable `Batcr` has 263 distinct values. Thus there are 262 ways to split the data into two subnodes and a search for the best split of the root node on `Batcr` alone would require the fitting of more than 500 linear models.

There are two ways to combat this problem: (i) retain the greedy search approach but fit a simple linear instead of a multiple linear regression model at each node (Alexander and Grimshaw (1996) use this method); (ii) fit piecewise multiple linear regression models but use statistical methods instead of greedy search to find the splits (SUPPORT, Chaudhuri et al. (1994), and PHDRT, Li et al. (2000)), are of this type). The main idea of SUPPORT is to apply the split selection techniques from the FACT (Loh and Vanichsetakul (1988)) classification tree algorithm to the regression problem. At each node, it uses the signs of the residuals to separate the observations into two classes and then uses two-sample t-tests for variable selection. Categorical predictors are not allowed. This idea is extended to piecewise generalized linear models and to regression models for censored data by Chaudhuri, Lo, Loh and Yang (1995) and Ahn and Loh (1994), respectively.

The models produced by PHDRT are even more general. While SUPPORT yields axis-orthogonal splits (also called univariate splits), PHDRT uses principal Hessian directions to find splits on linear combinations of predictors. The greater flexibility of the splits makes PHDRT potentially more accurate in terms of prediction error, although the trees are much harder to interpret.

We present a new algorithm called GUIDE (for *Generalized, Unbiased Interaction Detection and Estimation*) for building piecewise constant and piecewise linear regression models with univariate splits. It has four useful properties: (i) negligible selection bias; (ii) sensitivity to curvature and local pairwise interactions between regressor variables; (iii) inclusion of categorical predictor variables, including ordinal categorical variables; (iv) choice of three roles for each ordered predictor variable: split selection only, regression modeling only, or both.

The rest of the paper is organized as follows. Section 2 describes the GUIDE method for piecewise constant regression. Chi-square tests for curvature and interaction detection are presented and the problem of selection bias discussed. Section 3 extends GUIDE to piecewise linear regression. We define the different roles for ordered predictors and generalize the curvature and interaction tests. Section 4 introduces a bootstrap bias correction method and demonstrates its effectiveness in a simulation experiment. Section 5 examines the effect of pruning

on selection bias. Section 6 extends the approach to piecewise best simple linear models. Section 7 compares the prediction accuracy of GUIDE with CART and a spline-based method called MARS® (Friedman (1991)) on six real data sets. The benefit of bagging (Breiman (1996)) is also evaluated.

Unless stated otherwise, the trees presented here are pruned using the cost-complexity pruning method of CART with N-fold cross-validation, where N is the size of the training sample. That is, an overly large tree is constructed and then sequentially pruned back until only the root node is left. This yields a sequence of nested subtrees. The prediction mean square error (PMSE) of each subtree is estimated by N-fold cross-validation. The subtree with the smallest PMSE (p_0, say) is called the 0-SE tree. Letting s_0 be the estimated standard error of p_0, the 1-SE tree is the smallest subtree whose estimated PMSE is less than $p_0 + s_0$. The reader is referred to Breiman et al. (1984, Sec. 3.4) for further details on pruning and estimation of standard error.

2. Piecewise Constant Models

The SUPPORT algorithm fits a piecewise constant model as follows. At each node, a constant (namely, the sample Y-mean) is fitted and the residuals computed. Then the cases in the node are divided into two groups, with one group defined by the positive residuals and the other by the non-positive residuals. The variable with the smallest p-value among two-sample t and Levene (1960) tests for unequal means and variances, respectively, is chosen to split the node. The idea is to detect non-random patterns in the two groups of signed residuals.

There are two deficiencies in the SUPPORT method: exclusion of categorical predictors and inability to detect pairwise interactions. To solve the first problem, we use instead the Pearson chi-square test to detect associations between the signed residuals and groups of predictor values. If X is a c-category predictor, the test is applied to the $2 \times c$ table formed by the two groups of residuals as rows and the categories of X as columns. If X is a numerical-valued variable, its values can be grouped to form the columns of the table. We divide the range of X into four groups at the sample quartiles to yield a 2×4 table. There are other ways to define the groups for ordered variables, but there is probably none that is optimal for all situations. Our experience indicates that this choice provides sufficient detection power while keeping the chance of empty cells low.

Although the chi-square test is sensitive to curvature along the directions of the axes, it is ineffective against simple interaction models such as

$$Y = I(X_1 X_2 > 0) - I(X_1 X_2 \leq 0) + \epsilon,$$

where 0 is in the interior of the supports of X_1 and X_2 and ϵ is a noise variable. A test for pairwise interactions is needed here. To do this, we can partition the (X_1, X_2) space to form the columns of a new table. This yields the following procedure, which includes categorical variables.

Algorithm 1. Chi-square tests for constant fit.

1. Obtain the residuals from a constant model fitted to the Y data.
2. For each numerical-valued variable, divide the data into four groups at the sample quartiles; construct a 2×4 contingency table with the signs of the residuals (positive versus non-positive) as rows and the groups as columns; count the number of observations in each cell and compute the χ^2-statistic and its theoretical p-value from a χ^2_3 distribution. We refer to this as a *curvature* test.
3. Do the same for each categorical variable, using the categories of the variable to form the columns of the contingency table and omitting columns with zero column totals.
4. To detect interactions between a pair of numerical-valued variables (X_i, X_j), divide the (X_i, X_j)-space into four quadrants by splitting the range of each variable into two halves at the sample median; construct a 2×4 contingency table using the residual signs as rows and the quadrants as columns; compute the χ^2-statistic and p-value. Again, columns with zero column totals are omitted. We refer to this as an *interaction* test.
5. Do the same for each pair of categorical variables, using their value pairs to divide the sample space. For example, if X_i and X_j take c_i and c_j values, respectively, the χ^2-statistic and p-value are computed from a table with two rows and number of columns equal to $c_i c_j$ less the number of columns with zero totals.
6. For each pair of variables (X_i, X_j) where X_i is numerical-valued and X_j is categorical, divide the X_i-space into two at the sample median and the X_j-space into as many sets as the number of categories in its range (if X_j has c categories, this splits the (X_i, X_j)-space into $2c$ subsets); construct a $2 \times 2c$ contingency table with the signs of the residuals as rows and the subsets as columns; compute a χ^2-statistic and p-value for the table after omitting columns with zero totals.

If the smallest p-value is from a curvature test, it is natural to select the associated X variable to split the node. If the smallest p-value is from an interaction test, we need to select one of the two interacting variables. We could choose on the basis of the curvature p-values of the two variables but because the goal is to fit a constant model in each node, we base the choice on reduction in SSE.

Algorithm 2. Choice between interacting pair of X variables.

Suppose that a pair of variables is selected because their interaction test is the most significant among all the curvature and interaction tests.

1. If both variables are numerical-valued, the node is split in turn along the sample mean of each variable; for each split, the SSE for a constant model is obtained for each subnode; the variable yielding the split with the smaller total SSE is selected.
2. Otherwise if at least one variable is categorical, the one with the smaller curvature p-value is selected.

If a variable from a significant interaction is selected to split a node, one strategy could be to require the other variable in the pair to split the immediate children nodes. This has the advantage of highlighting the interaction in the tree structure. On the other hand, by letting all the variables compete for splits at every node, it may be possible to obtain a shorter tree. The latter strategy is adopted for this reason.

A simulation experiment was carried out to compare the variable selection bias of the methods, using three numerical-valued and two categorical variables. Three dependence structures among the X variables are considered: (i) the *independent* case, where the X's are mutually independent; (ii) a *weakly dependent* case, where some of the X_i's are not mutually independent; (iii) a *strongly dependent* case where the correlation between X_2 and X_3 is increased to 0.995. The marginal distributions are given in Table 2 and the joint distribution of the categorical variables X_4 and X_5 is in Table 3. The distribution of Y is independent standard normal in all cases.

Table 2. Distributions of X variables used in simulation models; C_5, C_{10}, U, T, W, and Z are mutually independent; C_k denotes a k-category variable taking values $\{1, 2, \ldots, k\}$ with equal probabilities; U is a uniform variate over the unit interval; T is a uniformly distributed variable on the set $\{\pm 1, \pm 3\}$; W is an exponential variable with mean 1; Z is a standard normal variable; $\lfloor . \rfloor$ is the greatest integer function.

	Independent	Weakly dependent	Strongly dependent
X_1	T	T	T
X_2	W	W	W
X_3	Z	$T + W + Z$	$W + 0.1Z$
X_4	C_5	$\lfloor UC_{10}/2 \rfloor + 1$	$\lfloor UC_{10}/2 \rfloor + 1$
X_5	C_{10}	C_{10}	C_{10}

Table 3. Joint distribution of categorical variables X_4 and X_5 under the weakly and strongly dependent simulation models.

	X_5									
X_4	1	2	3	4	5	6	7	8	9	10
1	1/10	1/10	2/30	1/20	2/50	1/30	2/70	1/40	2/90	1/50
2			1/30	1/20	2/50	1/30	2/70	1/40	2/90	1/50
3					1/50	1/30	2/70	1/40	2/90	1/50
4							1/70	1/40	2/90	1/50
5									1/90	1/50

Table 4. Estimated probabilities of variable selection for constant fit in the null case where Y is independent of the X's. Estimates are based on 1000 Monte Carlo iterations and 1000 samples in each iteration. A method is unbiased if it selects each variable with probability 0.2. All but two of the GUIDE estimates are within three standard errors of 0.2.

	Independent		Weakly dependent		Strongly dependent	
X_i	CART	GUIDE	CART	GUIDE	CART	GUIDE
X_1	.019	.176	.009	.183	.022	.201
X_2	.263	.193	.269	.175	.242	.173
X_3	.273	.204	.252	.178	.201	.154
X_4	.057	.200	.067	.228	.010	.242
X_5	.387	.227	.403	.236	.464	.230

Table 4 gives the estimated probability that each X_i is selected. To be unbiased, a method should select each variable with probability 0.2. The estimates in the table are based on 1000 simulation trials with 1000 samples used in each trial. The results for CART are given for comparison.

As expected, variables with more splits are more likely to be selected by CART. In the independent case, X_2 and X_3 each allow 999 splits and are preferred twelve times more often than X_1, which has only three splits. Similarly, X_5 with $2^9 - 1 = 511$ splits is preferred six times more often than X_4, with $2^4 - 1 = 15$ splits. It is interesting to note that X_5 has a higher selection probability than X_2 or X_3 even though it has fewer splits. Thus the bias toward categorical variables is not due to number of splits alone. In contrast, the GUIDE method is relatively unbiased. All but two of its values are within three standard errors of 0.2 (the two values occur in the strongly dependent model).

So far we have concentrated on the problem of variable selection. To complete the tree construction algorithm, we need to select the split points as well as determine the size of the tree. For the latter, we adopt the CART method of cost-complexity pruning with an independent test set or, in its absence, by cross-validation.

If the selected X is an ordered predictor, there are several ways to select c for a split of the form $X \leq c$. One is to use greedy search to find the value that minimizes the total SSE in the regression models fitted to the data subsets defined by the split. We will refer to this as the G (for 'greedy search') method. It can be computationally expensive, especially when applied to non-trivial piecewise models. To avoid the computations, SUPPORT chooses c to be the sample mean of X. This may seem inferior to the G method, but it ultimately depends on the form of the true regression surface. If the surface consists of two planes joining or breaking at some value of c along the X axis, the G method is better unless the break occurs at the sample mean. On the other hand, if the regression surface is nonlinear but smoothly varying, several splits would be required for an accurate piecewise linear approximation. In that case, the mean may be as good as any other split value.

The real disadvantage of using the sample mean for c is that the splits can yield highly unbalanced nodes if the data are skewed. To avoid this problem, we can use the sample median instead of the sample mean. This solution is called the M (for 'median') method in the sequel. GUIDE offers the G and M methods as well as a method in between that searches over a systematic sample of the order statistics. It will be shown in Table 11 in Section 7 that the M method is not always inferior to the G method.

If X is a categorical predictor, we need to find a split of the form $X \in A$, where A is a subset of the values taken by X. We accomplish this by viewing it as a classification problem. Label each observation in the node as class 1 if it is associated with a positive residual and as class 2 otherwise. Given a split determined by A, let L and R denote the data subsets in the left and right subnodes, respectively. We choose the set A for which the sum of the (binomial) variances in L and R is minimized. This solution is quickly found with an algorithm in Breiman et al. (1984, p.101).

The left panel of Figure 2 shows a piecewise constant GUIDE tree for the baseball data using the M method and the 1-SE rule. The first split is due to a significant curvature in `Yrs`. (Recall that `Yrs` was also chosen in (1) to segment the model.) The subsequent splits confirm that better past performance is generally rewarded by higher salaries. According to the tree, past performance is measured over the career for players with six or fewer years of experience. For those with seven or more years, past performance is mainly based on the previous year (`Bat86`).

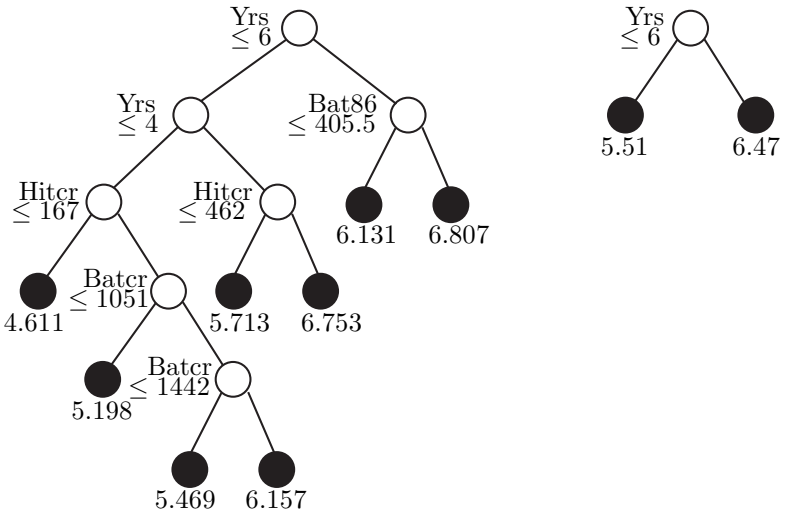

Figure 2. Piecewise constant (left) and piecewise linear (right) 1-SE GUIDE trees for baseball data with median split points. The piecewise constant 0-SE tree has 32 terminal nodes. The piecewise linear 0-SE and 1-SE trees are the same. The number beneath a node is the sample Y-mean.

The differences between this tree and the CART tree in Figure 1 are more apparent than real. For instance, their initial split variables (`Batcr` for CART and `Yrs` for GUIDE) have a correlation coefficient of 0.92. Therefore one variable is basically substituting for the other. The more years a player spends in the game, the longer his career and hence the more times he is at bat. Why then does CART choose `Batcr` instead of `Yrs`? We suspect the reason is selection bias, since there are 256 ways to split on `Batcr` but only 20 to split on `Yrs`.

3. Piecewise Linear Models

While no distinction between ordered and categorical variables is necessary for piecewise constant models, the situation is different for piecewise linear models. Previous piecewise linear regression tree algorithms have required categorical predictor variables to be transformed into 0-1 vectors before use. One problem with this approach is that a split on a categorical predictor X then takes the form $X \in A$, with A a singleton or the complement of a singleton. If the relationship between Y and X is a function of a set A that is not of this form, several splits on X will be required. This exaggerates the importance of the predictor and makes the tree more complicated than necessary.

Another problem with using 0-1 vectors is that an analysis of covariance (ANCOVA) model is fitted in each node. A predictor with c categorical values thus requires the estimation of $(c-1)$ coefficients, compared to one coefficient

for each ordered predictor. If c is large, this consumes a disproportionately large number of degrees of freedom. To avoid these problems, GUIDE employs categorical variables to split the nodes only; they are not used as regressors in the linear models. (GUIDE can be used to fit ANCOVA models but the user must first convert each categorical variable into 0-1 variables and treat them as ordered.)

For greater flexibility, GUIDE allows a numerical-valued predictor to play one of several roles. It can simultaneously compete for splits and act as a regressor, or it can be restricted to splitting only. In the latter situation, it can split the nodes but not enter into the regression equations. A useful consequence is that if all ordered variables are given this role, the algorithm produces a piecewise constant tree. Conversely, a numerical-valued predictor can act as a regressor in the linear models but not be allowed to compete for splits. To simplify the discussion, we use the following terminology:

`n`-variable: a numerical-valued predictor used to fit the regression models and to split the nodes;

`f`-variable: a numerical-valued predictor used to fit the regression models but not split the nodes;

`s`-variable: a numerical-valued predictor used to split the nodes but not fit the regression models;

`c`-variable: a categorical predictor used to split the nodes but not fit the regression models.

Thus a numerical-valued predictor can be employed as an `n`, `s`, or `f`-variable. The type to use depends on the application. For example, if it is desired to fit piecewise quadratic models, we can define the predictor X^2 as an `f`-variable. This will prevent X^2 from being selected for splitting. In other situations, where there is no obvious choice, a regression tree model can be fitted for each variable type and the final one selected according to cross-validation estimates of error and/or interpretability of the trees. The `s`-variable type can be used on an ordinal categorical variable after its values are given order-preserving numerical codes.

We can now extend Algorithm 1 to piecewise linear models.

Algorithm 3. Chi-square tests for linear fit.

1. Obtain the residuals from a linear model fitted to the `n`- and `f`-variables, leaving out the `s`- and `c`-variables.
2. For each `n`-variable, divide the data into four groups at the sample quartiles; construct a 2×4 contingency table with the signs of the residuals (positive versus non-positive) as rows and the groups as columns; count the number of observations in each cell and compute the χ^2-statistic and its theoretical p-value from a χ^2_3 distribution.

3. Do the same for each `s` and `c`-variable. For the latter, the categories of the variable form the columns of the table. Columns with zero column totals are omitted.
4. To detect interactions between each pair of `n`-variables (X_i, X_j), divide the (X_i, X_j)-space into four quadrants by splitting the range of each variable into two halves at the sample median; construct a 2×4 contingency table using the residual signs as rows and the quadrants as columns; compute the χ^2-statistic and p-value. Again, columns with zero column totals are omitted.
5. Do the same for each pair of `s`-variables.
6. Also do the same for each pair of `c`-variables, using their value pairs to divide the sample space. For example, if X_i and X_j take c_i and c_j unique values, respectively, the χ^2-statistic and p-value are computed from a table with 2 rows and number of columns equal to $c_i c_j$ less the number of zero columns.
7. Compute a χ^2-statistic and p-value for each pair (X_i, X_j) where X_i is an `n`-variable and X_j is a `c`-variable. If X_j has c categories, the table has 2 rows and number of columns equal to $2c$ less the number of zero columns.
8. Similarly, compute a χ^2-statistic and p-value for each pair where X_i is an `s`-variable and X_j is a `c`-variable.
9. Finally, do the same for each pair where X_i is an `s`-variable and X_j is an `n`-variable as in step 4.

Nine sets of chi-square tests are computed here: three sets to detect curvature in the `n`-, `s`-, and `c`-variables; another three sets to detect interactions between pairs of variables of the same type; three more sets to detect interactions between pairs of predictors of different types. If the smallest p-value comes from a curvature test, the associated variable is selected to split the node. For example, if it is an `n`-variable, we expect the curvature would be reflected in the different values its regression coefficient takes in the two subnodes. The next algorithm extends Algorithm 2.

Algorithm 4. Choice between interacting pair of variables.

Suppose that a pair of variables is selected because their interaction test is the most significant among the curvature and interaction tests.

1. If neither is an `n`-variable, choose the one with the smaller curvature p-value.
2. If both are `n`-variables, temporarily split the node along the sample mean of each variable; choose the variable whose split yields the smaller total SSE.
3. If exactly one is an `n`-variable, choose the other variable.

The main reason for preferring the non `n`-variable in Step 3 is that it simplifies the tree structure in some situations. For example, suppose that X_1 is an `n`-variable,

$X_2 \in \{a_1, a_2\}$ is a binary c-variable, and that the true model is

$$Y = \begin{cases} X_1 + \epsilon, & \text{if } X_2 = a_1 \\ -X_1 + \epsilon, & \text{if } X_2 = a_2. \end{cases}$$

Owing to the symmetry, the p-values from the curvature tests on X_1 and X_2 are likely to be of similar magnitude and be less significant than the p-value from the interaction test. The ideal split is on X_2, but there is no guarantee that this will happen if the choice is based on the p-values of the curvature tests.

After a variable is selected to split a node, the split point may be chosen by the G, M, or other method as described for piecewise constant models.

4. Bootstrap Bias Correction

Two simulation experiments were carried out to study the probabilities of variable selection in the case of piecewise linear regression when Y is independent of the X's. The distributions of the X's are the same as in Table 2. The only difference between the two experiments is that X_3 is an n-variable in the first and an s-variable in the second. Table 5 gives the results for the uncorrected method in the columns labeled "Unc." The large bias toward the c and s-variables is obvious.

Table 5. Estimated probabilities of variable selection for the uncorrected (denoted by "Unc.") and bias-corrected (denoted by "BC") chi-square methods for linear fit when Y is standard normal and independent of the X's. The distributions of the X's are given in Table 2. The results are based on 1000 Monte Carlo iterations and 1000 samples in each iteration. The bias-corrected method uses 50 bootstrap replications. A method is unbiased if it selects each variable with probability 0.2.

			Indep. X_i		Weak. dep. X_i		Strong. dep. X_i	
Expt.	X_i	Type	Unc.	BC	Unc.	BC	Unc.	BC
	X_1	n	0	.178	0	.191	0	.178
	X_2	n	0	.232	0	.206	0	.215
1	X_3	n	0	.200	0	.194	0	.197
	X_4	c	.469	.181	.519	.200	.532	.214
	X_5	c	.531	.209	.481	.209	.468	.196
	X_1	n	0	.202	0	.181	0	.197
	X_2	n	0	.217	0	.228	0	.214
2	X_3	s	.352	.203	.288	.134	.313	.121
	X_4	c	.307	.178	.360	.238	.360	.256
	X_5	c	.341	.200	.352	.219	.327	.212

The bias is not due solely to the preference for c and s-variables in Step 3 of Algorithm 4. It persists even when the selection is based on curvature p-value

as in Step 1. The reason is that the n-variables serve as regressors while the c and s-variables do not. As a result, the n-variables have zero sample correlation with the residuals. Therefore the distribution of their χ^2 values is stochastically much smaller than a chi-square distribution with the same degrees of freedom. The c and s-variables, on the other hand, are independent of the residuals and their χ^2 values follow the theoretical chi-square distributions more closely.

One way to correct the bias is to shrink the p-values of the chi-square tests involving the n-variables so that their chance of selection is increased. Finding the right shrinkage multiplier is tricky because some p-values may be near 0. To get around this problem, we convert the p-value from each chi-square test into a two-tailed z-value via the transformation $z = \Phi^{-1}(1 - p/2)$, where Φ is the standard normal distribution function, and then scale up the z-values associated with the n-variables by a constant factor denoted by r. Since the best value of r will likely depend on factors such as the number and mix of variables, their degree of association with one another, the sample size, and the configuration of the design points, we employ a bootstrap method to find r adaptively.

Algorithm 5. Bootstrap calibration for bias correction.

1. Let $Z = (Y, X_1, \ldots, X_k)$ denote the matrix of training samples, where $Y = (y_1, \ldots, y_n)'$ is a column vector of n responses and $X_i = (x_{1i}, \ldots, x_{ni})'$ is a column vector containing the corresponding observations on the ith predictor variable. Define the bootstrap sample as $Z^* = (Y^*, X_1, \ldots, X_k)$, where $Y^* = (y_1^*, \ldots, y_n^*)'$ and each y_j^* is a random draw with replacement from the set $\{y_1, \ldots, y_n\}$.
2. Fit a linear model to Z^* using only the n- and f-variables and obtain the residuals.
3. Compute the χ^2 p-values described in Algorithm 3 for this set of residuals and convert the p-values to two-tailed z-values.
4. Let z_n denote the largest z-value from a curvature test on an n-variable and let z_{nn} denote the largest z-value from an interaction test between two n-variables. Similarly, let z_s and z_{ss} denote the corresponding largest z-values for the s-variables, and z_c and z_{cc} the corresponding largest z-values for the c-variables. Finally, let z_{nc}, z_{ns}, and z_{sc} denote the largest z-values from the set of n-c, n-s, and s-c interaction tests. Given $r > 1$, select the n-variable if $r \max\{z_n, z_{nn}\} \geq \max\{z_s, z_c, z_{ss}, z_{cc}, z_{sc}, z_{nc}, z_{ns}\}$.
5. Repeat steps 1–4 B times over a grid of values of r. For each r, let $\pi(r)$ denote the proportion of times that an n-variable is selected.
6. Linearly interpolate, if necessary, to find the value of r such that $\pi(r)$ is equal to the proportion of n-variables among $(X_1, \ldots, X_k)$.

The results for the bias-corrected chi-square method are given in the columns labeled "BC" in Table 5. They are based on $B = 50$ and an equi-spaced grid of 40

values of r in the interval $[1, 5]$. (The selected value of r is almost always less than 2.) The bias correction is clearly effective—all entries lie within three simulation standard errors of 0.2 (the value for unbiased selection). Similar results were observed when different distributions, mix of variable types, and sample sizes were used.

In our examples and in the GUIDE program, bootstrap estimation of the scale factor r is carried out only at the root node. One argument for not bootstrapping at every node is that it reduces computational cost. Another is that the value of r is unlikely to change much in the first few levels of splits. At splits further down, estimation of r could become unstable because of the small sample sizes at the nodes. To assess the amount of computation incurred by bootstrapping, note that a one-time bias correction with B bootstrap iterations adds the equivalent of about B node computations. On the other hand, V-fold cross-validation requires $V + 1$ trees to be constructed and pruned. Letting M denote the average number of nodes in the trees, the total number of node computations is thus $B + M(V + 1)$ if bootstrapping is performed once, and $BM(V + 1)$ if it is carried out at every node. For the baseball data, $M \approx 30$ and $N = 263$. With N-fold cross-validation pruning, this yields $M(V + 1) \approx 30 \times 264 = 7920$. Since $B = 50$, the cost of a one-time correction increases the total cost by less than one percent. The results in the next and subsequent sections show that a one-time correction yields models with reasonably good fits in terms of SSE.

Table 6. Regression coefficients for the piecewise linear model in Figure 2

Predictor	Left node	Right node
Constant	4.16	6.17
Bat86	−2.50E−3	1.23E−3
Hit86	6.16E−3	2.24E−3
Hr86	3.23E−3	−1.00E−2
Run86	9.51E−3	−6.86E−4
Rb86	−1.20E−3	5.91E−4
Wlk86	−8.13E−4	6.97E−4
Yrs	1.14E−1	−9.39E−2
Batcr	4.18E−4	−2.53E−4
Hitcr	−1.13E−3	6.30E−4
Hrcr	−3.00E−3	−5.13E−4
Runcr	3.89E−4	7.04E−4
Rbcr	3.49E−3	9.29E−4
Wlkcr	1.84E−3	3.44E−4
Puto86	−1.09E−4	2.93E−4
Asst86	1.78E−4	−6.46E−4
Err86	−2.37E−3	6.79E−3

The piecewise linear GUIDE tree for the baseball data is shown on the right side of Figure 2. It splits only once, on `Yrs`. The same tree is obtained with the 0-SE or 1-SE rules. Compared with the piecewise constant 1-SE tree which has 8 terminal nodes, this tree is much shorter. Its simplicity is obtained by transferring some of the model complexity to the linear regression models whose coefficients are given in Table 6.

5. Effect of Pruning

The role of pruning has so far been ignored in our discussion of selection bias. In the null case where Y is independent of the X's, pruning may be expected to produce a tree with no splits. If that is the case, bias would be a problem only in non-null situations where the pruned trees are non-trivial. To study this problem, we performed a simulation experiment with the six regression models in Table 7. The first three models are simple functions of X_1 only. Therefore X_1 should be selected for splitting. The fourth model is additive in X_1 and X_5 with the latter acting as a `c`-variable. Since X_1 alters the mean of Y from -2.1 to 2.1 while X_5 changes it from 0.05 to 1.0, it is better to select X_1 than X_5. The fifth (Cross) model involves an interaction between X_1 and X_2. Clearly, a piecewise linear model should split on X_1. In the last (Steps) model, splitting on X_1 or X_3 would be correct.

Table 7. Distributions and models for simulation experiment on effect of pruning: C_k is a `c`-variable taking values in the set $\{1, 2, \ldots, k\}$ with equal probabilities; $N(0,1)$ and $E(0,1)$ denote the standard normal and exponential distributions, respectively; the predictors are mutually independent, with X_1, X_2, X_3 used as `n`-variables and X_4, X_5 as `c`-variables.

Distributions	Models	
$X_1 \sim U\{\pm 1, \pm 3\}$	Jump	$Y = 0.7I(X_1 > 0) + \epsilon$
$X_2 \sim E(0,1)$	Quadratic	$Y = 0.08X_1^2 + \epsilon$
$X_3 \sim N(0,1)$	Cubic	$Y = 0.02X_1^3 + \epsilon$
$X_4 \sim C_5$	Additive	$Y = 0.7I(X_1 > 0) + 0.05\sum_{i=1}^{20} iI(X_5 = i) + \epsilon$
$X_5 \sim C_{20}$	Cross	$Y = 0.5\text{sgn}(X_1)X_2 + \epsilon$
$\epsilon \sim N(0,1)$	Steps	$Y = 0.5\text{sgn}(X_1X_3) + \epsilon$

We simulated data from the models to compare the piecewise constant CART method with the piecewise constant and piecewise linear GUIDE methods with G split point selection. The predictors are mutually independent and the trees are pruned with ten-fold cross-validation and the 1-SE rule. Tables 8 and 9 give the estimated conditional and unconditional probabilities that the right variables are selected to split the root node, the condition being the event that the pruned tree is non-trivial. The results are based on 100 iterations and 500 samples per

iteration. Also reported are the estimated root mean square error (RMSE) of each method and the total CPU time on a 300MHz DEC Alpha workstation for 100 iterations. The RMSE is defined as the square root of $E\{\hat{f}(X) - f(X)\}^2$, where $\hat{f}(x)$ denotes the estimated regression function at x and the expectation is over an independent set of X values.

Table 8. Results of simulation experiment on effect of pruning for models without interactions. $|\tilde{T}|$ denotes the number of terminal nodes in a tree and A is the event that $\{|\tilde{T}| > 1\}$. $P(X_1)$ denotes the probability that X_1 is selected to split the root node. The GUIDE entries are obtained with the G split point selection method. Estimates are based on 100 iterations, with 500 samples per iteration. Simulation standard errors of RMSE are about 0.01.

True	Tree				Conditional on A			CPU
Model	Model	Method	$P(X_1)$	$P(A)$	$P(X_1)$	$E\|\tilde{T}\|$	RMSE	sec.
Jump	Constant	CART	1.00	0.77	1.00	2.00	0.12	1760
	Constant	GUIDE	1.00	0.52	1.00	2.00	0.20	1227
	Linear	GUIDE	0.70	0.00	-	-	0.18	1684
Quadratic	Constant	CART	0.89	0.33	1.00	2.91	0.25	3701
	Constant	GUIDE	0.96	0.34	1.00	2.85	0.25	2458
	Linear	GUIDE	0.98	0.20	1.00	2.00	0.30	3355
Cubic	Constant	CART	1.00	0.69	1.00	2.09	0.27	5615
	Constant	GUIDE	1.00	0.72	1.00	2.10	0.27	3472
	Linear	GUIDE	0.43	0.00	-	-	0.14	5044
Additive	Constant	CART	0.84	0.92	0.86	2.15	0.32	1867
	Constant	GUIDE	0.96	0.80	0.96	2.08	0.33	1268
	Linear	GUIDE	0.28	0.01	0.00	2.00	0.34	1684

Table 9. Results of simulation experiment on effect of pruning for models containing interactions. $|\tilde{T}|$ denotes the number of terminal nodes in a tree and A is the event that $\{|\tilde{T}| > 1\}$. $P(X_1)$ denotes the probability that X_1 is selected to split the root node. The GUIDE entries are obtained with the G split point selection method. Estimates are based on 100 iterations, with 500 samples per iteration. Simulation standard errors of RMSE are about 0.01.

True	Tree			Conditional on A				CPU
Model	Model	Method	$P(A)$	$P(X_1)$	$P(X_2)$	$E\|\tilde{T}\|$	RMSE	sec.
Cross	Constant	CART	1.00	1.00	0.00	3.15	0.41	1984
	Constant	GUIDE	1.00	1.00	0.00	3.35	0.40	1359
	Linear	GUIDE	0.90	0.98	0.01	2.16	0.18	1771
True	Tree			Conditional on A				CPU
Model	Model	Method	$P(A)$	$P(X_1)$	$P(X_3)$	$E\|\tilde{T}\|$	RMSE	sec.
Steps	Constant	CART	0.05	0.00	0.20	5.40	0.50	3906
	Constant	GUIDE	0.82	0.51	0.49	5.67	0.29	2761
	Linear	GUIDE	0.87	0.05	0.95	2.11	0.30	3519

We first compare CART with the piecewise constant GUIDE method. Table 8 shows that the non-trivial trees are very similar for the first three models—they always split on the right variable and have about the same average size. The selection bias of CART, however, is apparent in the Additive model, where it selects X_1 10% less often than GUIDE. In terms of RMSE, the two methods are also quite comparable, except at the Jump model, where GUIDE has a higher RMSE because it yields trivial trees twice as often as CART (48% versus 23%). Table 9 shows that the two methods are also comparable in the Cross model. Both split on X_1 all the time and they have similar sized trees and RMSEs.

The situation is quite different for the Steps model. Table 9 shows that CART gives a non-trivial tree only 5% of the time. The reason is that CART has a high probability of selecting uninformative variables for its splits here. Because these splits are noisy, they are eventually pruned away. In the rare instances where the CART tree is non-trivial, the first split is on a right variable (X_1 or X_3) only 20% of the time. As a result, the RMSE of CART is almost twice as large as that of GUIDE.

The piecewise linear GUIDE method almost never yields any splits for the four models in Table 8. Despite this, its RMSE is quite good, especially at the Cubic model. It does split (and on the right variables) in the Cross and Steps interaction models. Its RMSEs at these two models are good, especially at the Cross model. The results also show that the piecewise constant and piecewise linear GUIDE algorithms are faster than the piecewise constant CART algorithm over the six simulation models.

These simulations show that, in the absence of interactions, pruning can reduce the selection bias of the greedy search approach. On the other hand, when interactions are present, pruning does not help. We expect the same observations to hold in more complicated models.

6. Piecewise Best Simple Linear Models

A model that is in between a piecewise constant and a piecewise linear model is one in which a best simple linear model is fitted to each node. Such a model is useful when a piecewise constant model has too many nodes but a piecewise linear one has too few. In the baseball data, for example, the piecewise constant 1-SE tree has eight terminal nodes while the piecewise linear tree has two.

Alexander and Grimshaw (1996) implement this idea using greedy search to find the splits. But, possibly because of computational cost, they use a direct stopping rule instead of pruning to determine the tree size. Since our chi-square test approach is based on residuals, it can be applied in a straightforward manner to fit piecewise best simple linear models.

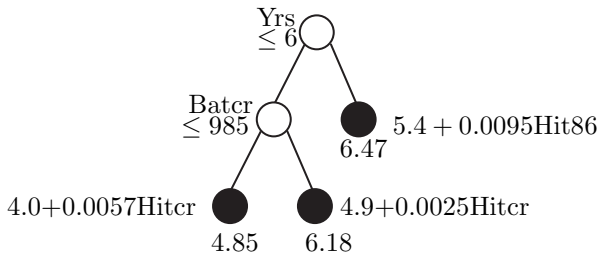

Figure 3. Best simple linear 1-SE GUIDE tree for baseball data using the M split point selection. A fitted equation is given beside each terminal node. The number beneath a node is the sample Y-mean. The 0-SE tree has nine terminal nodes.

Figure 3 shows the best simple linear GUIDE tree for the baseball data. It is shorter than the piecewise constant tree but has more structure than the piecewise linear tree. A unique advantage of a piecewise simple linear model is that the data and fitted line in each terminal node can be plotted and examined for lack of fit. For example, panels (b)–(d) of Figure 4 show that the data in

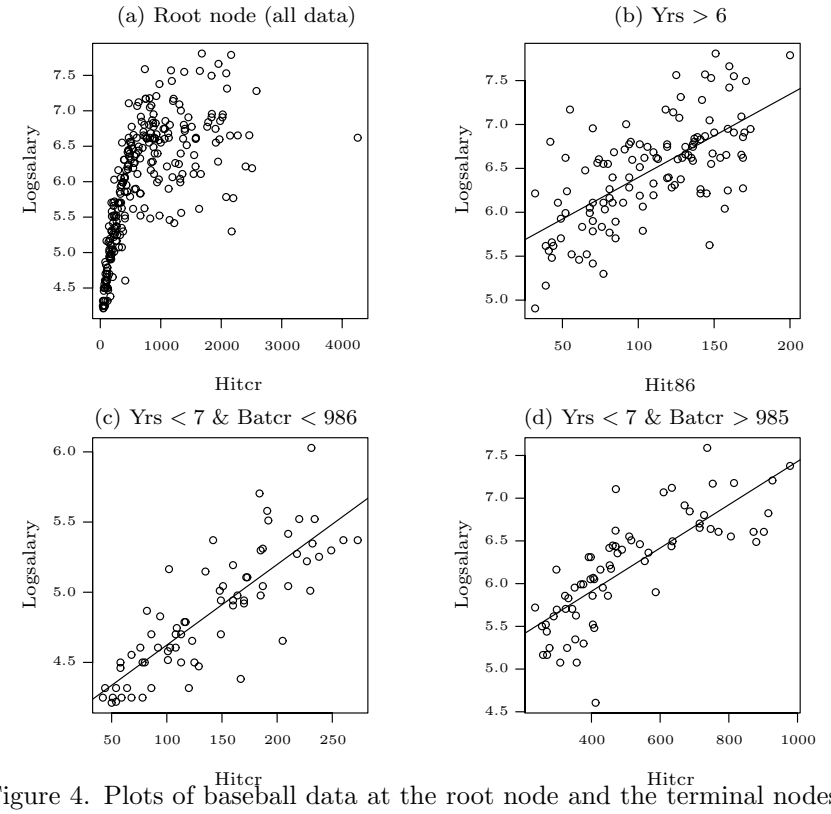

Figure 4. Plots of baseball data at the root node and the terminal nodes of the tree in Figure 3.

the terminal nodes of the tree fit their respective simple linear models well. This is not true in panel (a) which shows the relationship of `Logsalary` with `Hitcr` at the root node—here all kinds of problems with nonlinearity, heterogeneity and outliers are visible.

7. Prediction Accuracy for Real Data

To compare the prediction accuracy of the algorithms on real data, we test them on six data sets.

Baseball: This is the data set on salaries of baseball players introduced earlier.

Boston: The data contain information on median housing values and 13 related variables on 506 census tracts around Boston (Belsley, Kuh and Welsch (1980)). The response variable is the logarithm of the median housing value. Breiman et al. (1984) use the data to illustrate the CART method.

Mpg: This is a subset of the auto-mpg data at the University of California, Irvine, Repository of Machine Learning Databases (Merz and Murphy (1996)). The full data set was used in the 1983 American Statistical Association Exposition. We use the 392 cases with complete observations. There are 7 predictor variables; the dependent variable is mpg rating.

Mumps: This is taken from the `Statlib` archive. The data give the incidence of mumps in each of the 48 contiguous states of the U.S. (excluding the District of Columbia) from 1953 to 1989. There are 1523 observations on 4 variables. The dependent variable is the logarithm of the number of mumps cases reported per million population in each state. The predictor variables are year and the longitude and latitude of each state's center. Chaudhuri et al. (1994) use the data to illustrate the SUPPORT method.

TA: These data give the teaching evaluation scores of 324 teaching assistants at the University of Wisconsin, Madison, between 1993 and 2000. The problem is to predict the scores from two numerical and four categorical predictor variables: year, number of respondents, course (30 values), instructor (40 values), session (3 values), and whether or not the teaching assistant is a native English speaker (2 values).

Tecator: This is also taken from the `Statlib` archive. The data were collected by the Danish meat industry to calibrate a Tecator infrared spectrometer for the purpose of predicting the fat content of minced pork meat (Borggaard and Thodberg (1992)). The light transmitted through each of 215 meat samples was measured by the spectrometer at 100 different wavelengths.

Thodberg (1996) used the first ten principal components of the wavelengths to predict the actual fat content as measured by wet-chemistry methods. We employ the same predictors.

Table 10. Characteristics of real data sets.

		Predictors	
Data set	#cases	#n	#c
Baseball	263	16	6
Boston	506	13	0
Mpg	392	6	1
Mumps	1523	3	0
TA	324	2	4
Tecator	215	10	0

Table 10 summarizes the characteristics of each data set. We use ten-fold cross-validation to estimate the prediction mean square error (PMSE), defined as $E(Y - \hat{f}(X))^2$, where $\hat{f}$ is the regression function estimated from the training sample and (X, Y) is an independent observation. Each data set is randomly divided into ten roughly equal parts. One part is held out in turn and a regression estimate is constructed from the remaining nine parts. Ten-fold cross-validation is used to prune the trees. The held-out part is then used to estimate the PMSE. The average of the ten estimated PMSEs is reported in Table 11.

Table 11. Ten-fold cross-validation estimates of PMSE. The column labeled "Model" indicates whether the fitted model is piecewise constant, piecewise simple linear, piecewise multiple linear, or spline. For GUIDE, "G" and "M" stand for the greedy search and median split selection methods, respectively. All trees are pruned with the 0-SE rule. Bagged models are based on fifty bootstrap replicates.

Bagged	Model	Method	Baseball	Boston	Mpg	Mumps	TA	Tecator
No	Constant	CART	0.230	0.045	11.64	0.94	0.898	52.07
No	Constant	GUIDE(G)	0.192	0.041	9.87	1.10	0.749	57.02
No	Constant	GUIDE(M)	0.184	0.047	14.68	1.54	0.740	42.85
No	Simple	GUIDE(M)	0.178	0.040	9.54	1.25	0.778	43.79
No	Multiple	GUIDE(G)	0.133	0.028	9.67	0.87	0.786	7.45
No	Multiple	GUIDE(M)	0.133	0.079	10.24	1.02	0.774	5.53
Yes	Constant	CART	0.159	0.032	8.37	0.79	0.740	34.16
Yes	Constant	GUIDE(G)	0.140	0.027	8.46	0.81	0.763	27.47
Yes	Constant	GUIDE(M)	0.136	0.034	9.56	1.39	0.762	24.79
Yes	Simple	GUIDE(M)	0.133	0.027	7.20	1.05	0.742	16.99
Yes	Multiple	GUIDE(G)	0.126	0.028	10.22	0.76	0.755	4.94
Yes	Multiple	GUIDE(M)	0.123	0.024	7.44	0.86	0.762	4.06
No	Spline	MARS	0.156	0.030	7.89	1.40	0.911	6.78

Also reported are results from bagging the CART and GUIDE trees. Bagging (Breiman (1996)) is a technique for improving the prediction accuracy of regression trees. It creates an ensemble of trees by repeatedly drawing bootstrap samples from the training sample and constructing a tree from each sample. The predicted value of a new observation is obtained by averaging the predictions from the ensemble. The bagging results in Table 11 are obtained from ensembles of fifty bootstrap trees each. For comparison, we also include the results for MARS (Friedman (1991)), a nonparametric spline method that does not produce a tree structure.

Figure 5 gives a visual representation of the results. The length of each bar is equal to the PMSE of a method divided by that of CART without bagging. Among methods that do not employ bagging, we see the following.

1. The accuracy of the piecewise constant GUIDE models is quite similar to that of the CART models. In one-to-one comparisons, GUIDE(G) beats CART in four of six data sets and loses in two. GUIDE(M) is even against CART, and so is GUIDE(G) vs. GUIDE(M).

2. Piecewise linear models tend to possess better prediction accuracy than piecewise constant models, although the former are not uniformly superior. The performance of the piecewise multiple linear GUIDE models is especially impressive on the Tecator data set.

3. The two piecewise multiple linear GUIDE models hold up well against MARS. Each is more accurate than MARS four of six times.

4. The accuracy of the piecewise best simple linear GUIDE model is generally between that of the piecewise constant and the piecewise multiple linear models. Model simplicity is obtained at the cost of some loss in prediction accuracy.

Bagging usually reduces the PMSE, except in three instances: the piecewise constant GUIDE models in the TA data, and the multiple linear GUIDE(G) model in the Mpg data. The bagged piecewise constant CART model is more accurate than the corresponding GUIDE(M) model four of six times and is even against the bagged piecewise constant GUIDE(G) model. Bagging is particularly effective for the piecewise multiple linear GUIDE(M) method, which beats MARS uniformly across the data sets. It is recommended for sheer prediction accuracy. Of course, since the bagged model consists of fifty piecewise linear models, it is practically impossible to interpret.

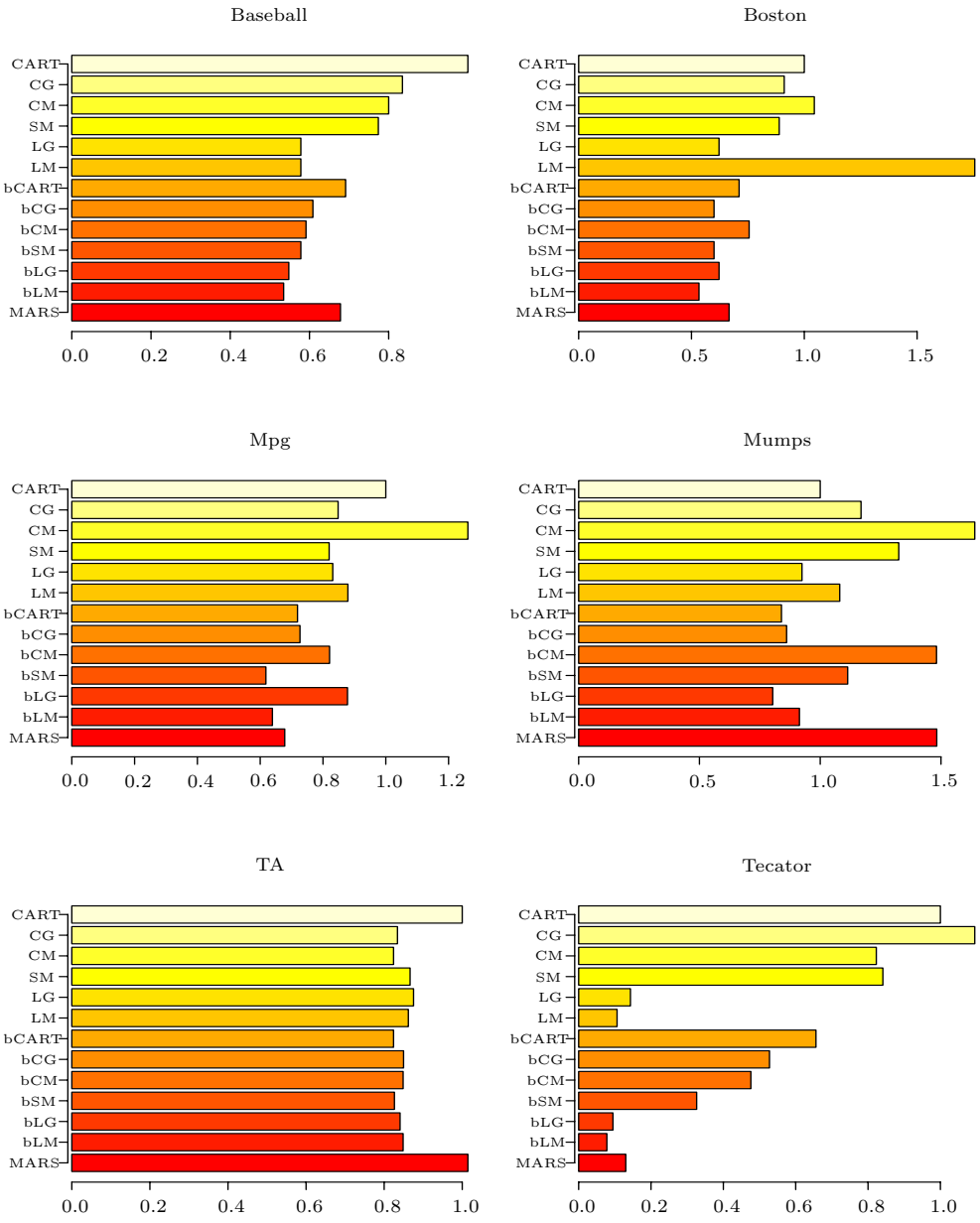

Figure 5. Barcharts of prediction mean square error relative to CART for the results in Table 11. `CG` and `CM` refer to the GUIDE piecewise constant tree methods with G and M split points, respectively, `SM` the piecewise simple linear method with M split point, and `LG` and `LM` the multiple linear methods with G and M split points. The `b` in a label indicates bagging. The bars are arranged in the same order as the rows of Table 11.

8. Conclusion

GUIDE is designed to solve three problems that can adversely affect the interpretability of a regression tree: bias in variable selection, insensitivity to local interactions, and complicated tree structures. Since the chief advantage of a regression tree over other models is the ease with which the model can be interpreted, it is important that the construction method be free of selection bias. GUIDE achieves this goal by employing lack-of-fit tests followed by bootstrap adjustment of the p-values. The latter is critical because the appropriate amount of adjustment is typically data dependent.

A welcome by-product of the bootstrap is that it permits greater freedom in split selection without the worry of selection bias. GUIDE exploits this by including tests for local interactions between pairs of variables. Splits that are sensitive to pairwise interactions can produce shorter trees.

Because interpretability of a tree structure decreases rapidly with increase in its complexity, a tree with a large number of splits can be harder to comprehend than a standard linear regression model. This problem is exacerbated by the traditional practice of fitting constants to each partition. One way to simultaneously reduce the complexity of a tree structure and increase prediction accuracy is to fit more complex models than constants. Unfortunately this is too computationally expensive to be practical with the standard greedy search approach. As a result, past attempts to fit piecewise linear models were forced to employ non-greedy search approaches. We showed that abandonment of the greedy search approach opens the way not only to computational savings but to the possibility of bias reduction.

It should be emphasized that although the greedy search approach of simultaneous optimization over the set of variables and their split points produces selection bias, this does not mean that greedy search cannot be employed to search for a split point *after* a split variable is found. The GUIDE results for the G split point selection show that there is nothing wrong with doing this. Finally, it is interesting to observe that in applications where model interpretation is unimportant, bagging a piecewise multiple linear GUIDE model can produce a highly accurate predictor.

Some of the ideas in this paper have been extended to quantile regression and classification problems in Chaudhuri and Loh (1998) and Kim and Loh (2001), respectively.

Acknowledgements

The author gratefully acknowledges the comments and suggestions of a referee and an associate editor. This material is based upon work supported by, or in

part by, the U. S. Army Research Laboratory and the U. S. Army Research Office under grant numbers DAAH04-94-G-0042 and DAAG55-98-1-0333, a grant from Pfizer Inc., and a University of Wisconsin Vilas Associateship. The GUIDE computer program may be obtained from http://www.stat.wisc.edu/~loh/. CART is a registered trademark of California Statistical Software, Inc. MARS is a trademark of JerIll, Inc.

References

Ahn, H. and Loh, W.-Y. (1994). Tree-structured proportional hazards regression modeling. *Biometrics* **50**, 471-485.

Alexander, W. P. and Grimshaw, S. D. (1996). Treed regression. *J. Comput. Graph. Statist.* **5**, 156-175.

Belsley, D. A. and Kuh, E. and Welsch, R. E. (1980). *Regression Diagnostics: Identifying Influential Data and Sources of Collinearity.* Wiley, New York.

Borggaard, C. and Thodberg, H. H. (1992). Optimal minimal neural interpretation of spectra. *Anal. Chemistry* **64**, 545-551.

Breiman, L. (1996). Bagging predictors. *Machine Learning* **24**, 123-140.

Breiman, L., Friedman, J. H., Olshen, R. A. and Stone, C. J. (1984). *Classification and Regression Trees.* Wadsworth, Belmont.

Chaudhuri, P. and Loh, W.-Y. (1998). Quantile regression trees. Technical Report 994, Department of Statistics, University of Wisconsin, Madison.

Chaudhuri, P., Huang, M.-C., Loh, W.-Y. and Yao, R. (1994). Piecewise-polynomial regression trees. *Statist. Sinica* **4**, 143-167.

Chaudhuri, P., Lo, W.-D., Loh, W.-Y. and Yang, C.-C. (1995). Generalized regression trees. *Statist. Sinica* **5**, 641-666.

Chipman, H., George, E. I. and McCulloch, R. E. (1998). Bayesian CART Model Search (with discussion). *J. Amer. Statist. Assoc.* **93**, 935-960.

Ciampi, A., Hogg, S. A., McKinney, S. and Thiffault, J. (1988). RECPAM: A computer program for recursive partition and amalgamation for censored survival data and other situations frequently occurring in biostatistics, I: Methods and program features. *Comput. Methods and Programs in Biomedicine* **26**, 239-256.

Ciampi, A., Lou, Z., Lin, Q. and Negassa, A. (1991). Recursive partition and amalgamation with the exponential family: theory and applications. *Appl. Stochastic Models Data Anal.* **7**, 121-137.

Conlon, M. and Meyer, J. (1988). Baseball performance and salaries. *ASA* 1988 *Proceedings of the Section on Statistical Graphics*, 98-103. American Statistical Association, Alexandria, VA.

Denison, D. G., Mallick, B. K. and Smith, A. F. M. (1998). A Bayesian CART algorithm. *Biometrika* **85**, 363-377.

Doyle, P. (1973). The use of automatic interaction detector and similar search procedures. *Oper. Res. Quarterly* **24**, 465-467.

Fielding, A. (1977). Binary segmentation: the automatic detector and related techniques for exploring data structure. In *The Analysis of Survey Data, Volume I, Exploring Data Structures* (Edited by C. A. O'Muircheartaigh and C. Payne). Wiley, New York.

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

Hawkins, D. M. (1997). FIRM: Formal inference-based recursive modeling. PC version, Release 2.1. Technical Resport 546, School of Statistics, University of Minnesota.

Hoaglin, D. C. and Velleman, P. F. (1995). A critical look at some analyses of major league baseball salaries. *Amer. Statist.* **49**, 277-285.

Kass, G. V. (1975). Significance Testing in Automatic Interaction Detection (A.I.D.). *Appl. Statist.* **24**, 178-189.

Kim, H. and Loh, W.-Y. (2001). Classification trees with unbiased multiway splits. *J. Amer. Statist. Assoc.* **96**, 589-604.

Levene, H. (1960). Robust tests for equality of variances. In *Contributions to Probability and Statistics* (Edited by I. Olkin, S. G. Ghurye, W. Hoeffding, W. G. Madow and H. B. Mann), 278-292. Stanford University Press, Palo Alto.

Li, K.-C., Lue, H.-H. and Chen, C.-H. (2000). Interactive tree-structured regression via principal Hessian directions. *J. Amer. Statist. Assoc.* **95**, 547-560.

Loh, W.-Y. and Vanichsetakul, N. (1988). Tree-structured classification via generalized discriminant analysis (with discussion). *J. Amer. Statist. Assoc.* **83**, 715-728.

Marshall, R. J. (1995). A program to implement a search method for identification of clinical subgroups. *Statist. in Medicine* **14**, 2645-2659.

Merz, C. J. and Murphy, P. M. (1996). UCI repository of machine learning databases. Department of Information and Computer Science, University of California, Irvine, CA. (http://www.ics.uci.edu/~mlearn/MLRepository.html).

Morgan, J. N. and Sonquist, J. A. (1963). Problems in the analysis of survey data, and a proposal. *J. Amer. Statist. Assoc.* **58**, 415-434.

Thodberg, H. H. (1996). A Review of Bayesian neural networks with an application to near infrared spectroscopy. *IEEE Transactions on Neural Networks* **7**, 56-72.

Department of Statistics, University of Wisconsin–Madison, 1210 West Dayton Street, Madison, WI 53706-1685, U.S.A.

E-mail: loh@stat.wisc.edu

(Received February 2000; accepted May 2001)