Branch-and-Bound Algorithms for Computing the Best-Subset Regression Models

Cristian GATU and Erricos John KONTOGHIORGHES

An efficient branch-and-bound algorithm for computing the best-subset regression models is proposed. The algorithm avoids the computation of the whole regression tree that generates all possible subset models. It is formally shown that if the branch-and-bound test holds, then the current subtree together with its right-hand side subtrees are cut. This reduces significantly the computational burden of the proposed algorithm when compared to an existing leaps-and-bounds method which generates two trees. Specifically, the proposed algorithm, which is based on orthogonal transformations, outperforms by $O(n^3)$ the leaps-and-bounds strategy. The criteria used in identifying the best subsets are based on monotone functions of the residual sum of squares (RSS) such as R^2, adjusted R^2, mean square error of prediction, and C_p. Strategies and heuristics that improve the computational performance of the proposed algorithm are investigated. A computationally efficient heuristic version of the branch-and-bound strategy which decides to cut subtrees using a tolerance parameter is proposed. The heuristic algorithm derives models close to the best ones. However, it is shown analytically that the relative error of the RSS, and consequently the corresponding statistic, of the computed subsets is smaller than the value of the tolerance parameter which lies between zero and one. Computational results and experiments on random and real data are presented and analyzed.

Key Words: Least squares; QR decomposition; Subset regression.

1. INTRODUCTION

An important topic in statistical modeling is the subset-selection regression or, equivalently, finding the best regression equation (Hastie, Tibshirani, and Friedman 2001). One method to select the coefficients to be included in the regression is to search over all possible submodels and to choose the one which optimizes a given selection criterion (Miller 2002;

Cristian Gatu is affiliated with the Department of Computer Science, University of Neuchâtel, Emile-Argand 11, CH-2007 Neuchâtel, Switzerland (E-mail: cristian.gatu@unine.ch). Erricos John Kontoghiorghes is affiliated with the Department of Public and Business Administration, University of Cyprus, Cyprus, and the School of Computer Science and Information Systems, Birkbeck College, University of London, UK (E-mail: erricos@ucy.ac.cy).

Journal of Computational and Graphical Statistics, Volume 15, Number 1, Pages 139–156
DOI: 10.1198/106186006X100290

Seber 1977). Stepwise forward and backward procedures based on adding or deleting coefficients one at a time with respect to some criterion can also be employed. However, these methods search fewer combinations of submodels and rarely select the best one (Hocking 1976; Seber 1977). The recently proposed nonnegative garrote, least absolute shrinkage and selection operator (LASSO), and smoothly clipped absolute deviation (SCAD) methods combine the subset selection and ridge regression. These methods set to zero some coefficients and shrink the values of the remaining ones (Breiman 1995; Fan and Li 2001; Tibshirani 1996).

A common approach to subset selection is the computation of the best-subset regression models (Edwards and Havránek 1987; Furnival and Wilson 1974; Hocking 1983; Hocking and Leslie 1967; Miller 2002). Consider the standard regression model

$$y = A\beta + \varepsilon, \tag{1.1}$$

where $y \in \mathbb{R}^m$ is the dependent variable vector, $A \in \mathbb{R}^{m\times n}$ is the exogenous data matrix of full column rank, $\beta \in \mathbb{R}^n$ is the coefficient vector, and $\varepsilon \in \mathbb{R}^m$ is the noise vector. It is assumed that ε has zero mean and variance-covariance matrix $\sigma^2 I_m$. One approach to search for the best models is the straight-forward method of generating all $2^n - 1$ possible submodels (Hocking 1976; LaMotte and Hocking 1970; Miller 1984). As n increases the number of models to be computed increases exponentially. Efficient strategies for moving from one model to another with minimum computational cost have been previously proposed (Clarke 1981; Smith and Bremner 1989). Further improvements are possible by exploiting the structural properties of the problem and using parallel strategies (Gatu and Kontoghiorghes 2003).

Still, for a large number of variables the consideration of all models will be computational infeasible. In such cases procedures that compute the best submodels without investigating all possible subsets need to be considered. Existing methods based on a leaps-and-bounds strategy derive the best models for each number of variables by generating two trees (Furnival and Wilson 1974). The first one, the bounds tree, provides half of the models (2^{n-1}) that do not include the last variable. A Gaussian elimination step is used to obtain a child from its parent node (Goodnight 1979). The construction of this tree aims to provide good bounds in early stages of the execution of the algorithm. The second tree, called the regressions tree, provides the other half of the models $(2^{n-1} - 1)$ that include the last variable. The models are generated by moving from one node to another. A matrix inverse is computed each time a model is derived. The problem of finding the best-subset model of given size d, where $1 \leq d \leq n$, has been previously considered. The use of the QR decomposition (QRD) instead of inverse matrices in this context has also been discussed (Miller 1992, 2002). In a particular optimization context, branch-and-bound algorithms for choosing a subset of d features from a given larger set of size n have been also investigated (Narendra and Fukunaga 1977; Ridout 1988; Roberts 1984). These strategies are designed for the case when the selection criterion is a quadratic monotone function. The size of the subset to be selected, that is, d, is known and thus, these algorithms search over $\mathrm{C}_n^d = n!/(d!(n-d)!)$ subsets.

A branch-and-bound algorithm (hereafter abbreviated to BBA) that generates the best

regression models corresponding to each number of variables is proposed (Lawler and Wood 1966; Lee 2002). The algorithm avoids computing all-possible-subset models by exploiting the properties of a regression tree that consists of 2^{n-1} nodes (Gatu and Kontoghiorghes 2003). Each of these nodes provides a number of regression models. The computational tool used by the BBA is the QRD. Specifically, it employs Givens rotations to compute the QRD of a triangular matrix after deleting a column (Golub and Van Loan 1996; Kontoghiorghes 2000a). Theoretical measures of complexity are derived in order to compare the proposed BBA against existing methods. Several strategies and heuristics that improve the computational performance of the BBA are presented. The algorithms are implemented using FORTRAN, BLAS, and LAPACK on a Sun-Fire-280R of two SPARC-v9 processors with a clock speed of 750MHz and 4Gb of RAM.

Section 2 briefly discusses the use of the QRD to compute the least-squares estimates and residual sum of squares of the regression model. The regression tree that generates all possible models is introduced. Section 3 presents the BBA and various strategies for improving the efficiency and the applicability of the algorithm. Various heuristics that allow the investigation of large-scale models are also considered. Section 4 analyzes experimental results. Conclusions and notions for future work are presented and discussed in the final section.

2. COMPUTING ALL POSSIBLE SUBMODELS

The QRD of the exogenous matrix A in (1.1) is given by

$$Q^T A = \begin{pmatrix} R \\ 0 \end{pmatrix}_{m-n}^{n}, \tag{2.1}$$

where $Q \in \mathbb{R}^{m\times m}$ is orthogonal and $R \in \mathbb{R}^{n\times n}$ is upper triangular and nonsingular. Let

$$Q^T y = \tilde{y} = \begin{pmatrix} \tilde{y}_1 \\ \tilde{y}_2 \end{pmatrix}_{m-n}^{n}. \tag{2.2}$$

The least-squares estimator of β is given by

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}} \|y - A\beta\|^2 = \underset{\beta}{\operatorname{argmin}} \|Q^T(y - A\beta)\|^2 = R^{-1}\tilde{y}_1,$$

where $\|\cdot\|$ denotes the Euclidean norm. The residual sum of squares (RSS) is computed by

$$\text{RSS} = \|y - A\hat{\beta}\|^2 = \|Q^T(y - A\hat{\beta})\|^2 = \|\tilde{y}_2\|^2.$$

Let S denote a selection matrix, which comprises d columns of the $n \times n$ identity matrix I_n. Consider the modified regression model

$$y = A_{(S)}\beta_{(S)} + \varepsilon, \tag{2.3}$$

where $A_{(S)} = AS \in \mathbb{R}^{m\times d}$ and $\beta_{(S)} = S^T\beta \in \mathbb{R}^d$. The least-squares estimator of $\beta_{(S)}$, that is, $\hat{\beta}_{(S)}$, is obtained from the solution of

$$\underset{\beta_{(S)}}{\operatorname{argmin}} \|y - A_{(S)}\beta_{(S)}\|^2 = \underset{\beta_{(S)}}{\operatorname{argmin}} \|Q^T(y - AS\beta_{(S)})\|^2 = \operatorname{argmin} \beta_{(S)} \|\tilde{y}_1 - RS\beta_{(S)}\|^2.$$

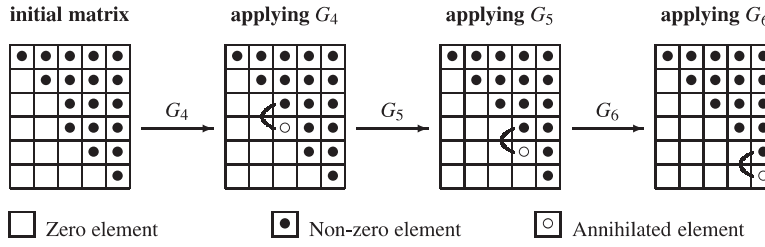


Figure 1. Re-triangularization of an $n \times n$ upper triangular matrix after deleting the ith column using Givens rotations, where $n = 6$ and $i = 3$.

Now, computing the QRD of RS

$$Q_{(S)}^T RS = \begin{pmatrix} R_{(S)} \\ 0 \end{pmatrix}_{n-d}^{d} \quad \text{and} \quad Q_{(S)}^T \tilde{y}_1 = \begin{pmatrix} \hat{y}_1 \\ \hat{y}_2 \end{pmatrix}_{n-d}^{d}, \tag{2.4}$$

the least-squares estimator and the residual sum of squares of the modified model (2.3) are given by $\hat{\beta}_{(S)} = R_{(S)}^{-1}\hat{y}_1$ and $\text{RSS}_{(S)} = \text{RSS} + \hat{y}_2^T\hat{y}_2$, respectively. The factorization (2.4) is equivalent to the re-triangularization of R in (2.1) after deleting columns (Kontoghiorghes 1995, 1999, 2000a; 2000b; Kontoghiorghes and Clarke 1993b, 1993a). Notice that if S consists of the first d $(d = 1, \ldots, n)$ columns of I_n, then $Q_{(S)}^T = I_n$ and $R_{(S)}$ is the leading $d \times d$ submatrix of RS. Thus, given R the submodels comprising the variables $[1], [1, 2], \ldots, [1, 2, \ldots, n]$ are obtained from the corresponding $1 \times 1, 2 \times 2, \ldots, n \times n$ leading triangular submatrices of R.

Now, let $V = [v_1, v_2, \ldots, v_d]$ denote the $d = |V|$ indices of the selected columns (variables) included in the submatrix $R_{(S)}$ (model). The submodels corresponding to the subsets $[v_1]$, $[v_1, v_2]$, …, $[v_1, v_2, \ldots, v_d]$ are immediately available. The problem is equivalent to computing all $2^n - 1$ subsets of $V = [v_1, v_2, \ldots, v_n]$, where $v_i = i$ $(i = 1, \ldots, n)$. In solving this problem, a function called Drop will be used. This function re-triangularizes a triangular matrix after deleting a column. That is,

$$\text{Drop}(V, i) = [v_1, \ldots, v_{i-1}, v_{i+1}, \ldots, v_n], \quad \text{where} \quad i = 1, \ldots, n-1.$$

The re-triangularization performed by Drop is obtained by applying a sequence of Givens rotations using $(n-i)(n-i+3)/2$ floating point operations (flops) (Golub and Van Loan 1996; Kontoghiorghes 2000a). The application of the Givens rotations on the modified response vector y is not discussed but is included in the complexity analysis. The rotations are between adjacent planes. Figure 1 shows the re-triangularization of a 6×6 matrix after deleting the third column. A rotation G_j $(j = 4, 5, 6)$ annihilates the element at position $(j, j-1)$. The rows affected by the rotations are indicated by an arc.

A formal and detailed description of the regression tree that generates all subset models was given by Gatu and Kontoghiorghes (2003). Here the basic concepts using a more convenient notation are introduced. A node of the regression tree is a tuple (V, k), where V is the set of indices and k $(k = 0, \ldots, |V| - 1)$ indicates that the children of this node will include the first k variables. If $V = [v_1, v_2, \ldots, v_n]$, then the regression tree $T(V, k)$

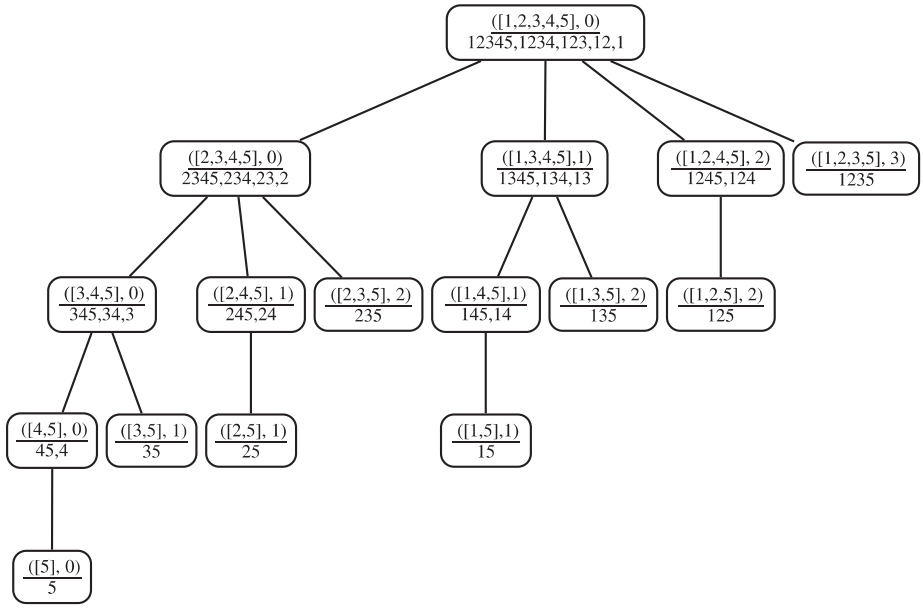


Figure 2. The regression tree $T(V, k)$, where $V = [1, 2, 3, 4, 5]$ and $k = 0$.

is an $(n-1)$–tree having as root the node (V, k), where $k = 0, \ldots, n-1$. The children are defined by the tuples $(\text{Drop}(V, i), i-1)$ for $i = k+1, \ldots, n-1$. Formally this can be expressed recursively as

$$T(V,k) = \begin{cases} (V,k) & \text{if} \quad k = n-1, \\ \big((V,k), T(\text{Drop}(V,k+1),k), \ldots, T(\text{Drop}(V,n-1),n-2)\big) & \text{if} \quad k < n-2. \end{cases}$$

The number of nodes in the subtree $T(\text{Drop}(V, i), i-1)$ is given by $\delta_i = 2^{n-i-1}$ and $\delta_i = 2\delta_{i+1}$, where $i = 1, \ldots, n-1$ (Gatu and Kontoghiorghes 2003).

Computing all possible subset regressions of a model having n variables is equivalent to generating $T(V, 0)$, where $V = [1, 2, \ldots, n]$. Generally, the complexity—in terms of flops—of generating $T(V, k)$ is given by

$$18 \cdot 2^{|V|-k} - 3(|V| - k + 2)(|V| - k + 3). \tag{2.5}$$

Figure 2 shows $T(V, k)$ together with the submodels generated from each node, where $V = [1, 2, 3, 4, 5]$ and $k = 0$. A submodel is denoted by a sequence of numerals that correspond to the indices of variables.

3. THE BRANCH-AND-BOUND ALGORITHM (BBA)

Most of the commonly used statistics associated with model selection, such as R^2, adjusted R^2, mean square error of prediction (MSEP), and Mallow's C_p, are monotone

functions of the RSS (Allen 1971; Hocking 1972; Mallows 1995; Miller 1984). In this context, the problem of finding the best submodels is reduced to one of finding the submodels of size d with the minimum RSS, where $d = 1, \ldots, n$. Let V_1 and V_2 be two sets of independent variables and $\text{RSS}(V_i)$ denote the RSS of the model comprising the variables in V_i $(i = 1, 2)$. The selection methods which find the best-subset models without computing all regressions are based on the fundamental property

$$\text{RSS}(V_1) \geq \text{RSS}(V_2), \quad \text{where} \quad V_1 \subseteq V_2. \tag{3.1}$$

That is, deleting variables from a regression increases the RSS of the modified model. Furthermore, if f is a monotone function of the RSS for subsets with the same number of variables and $\text{RSS}(V_1) \geq \text{RSS}(V_2)$, then

$$f(\text{RSS}(V_1)) \geq f(\text{RSS}(V_2)), \quad \text{where} \quad |V_1| = |V_2|. \tag{3.2}$$

Therefore, it follows from (3.1) and (3.2) that any monotone function of RSS (e.g., R^2, adjusted R^2, MSEP, and C_p) can be replaced by the RSS as the selection criteria for the branch-and-bound process (Furnival and Wilson 1974; Roberts 1984).

The number of evaluated subsets when searching for the best models can be restricted by using (3.1). Let $V = [1, 2, \ldots, n]$ and $r_j^{(g)}$ $(j = 1, \ldots, n$ and $g = 1, \ldots, 2^{n-1})$ denote the current minimum value of the RSS for the models with j variables after g nodes of the regression tree $T(V, 0)$ have been derived. Notice that the generating order of the children nodes is not important. The root node $(V, 0)$ provides the values $r_1^{(1)}, r_2^{(1)}, \ldots, r_n^{(1)}$ from the RSS of the available subsets $[1], [1, 2], \ldots, [1, 2, \ldots, n]$, respectively. Once the gth node $([v_1, \ldots, v_{i-1}, v_{i+1}, \ldots, v_d], i - 1)$ has been derived, the subsets $[v_1, \ldots, v_{i-1}, v_{i+1}]$, …, $[v_1, \ldots, v_{i-1}, v_{i+1}, \ldots, v_d]$ become available and the $r_j^{(g)}$ $(j = i, \ldots, d - 1)$ are updated. For $j = 1, \ldots, i-1$ and $j = d, \ldots, n$, the minimum values of the RSS are not modified and $r_j^{(g)} = r_j^{(g-1)}$. After the whole regression tree $T(V, 0)$ has been generated, the minimum RSS corresponding to the best regression models for each number of variables are given by $r_1^{(2^{n-1})}, \ldots, r_n^{(2^{n-1})}$.

Lemma 1. *$r_j^{(g)} \geq r_{j+1}^{(g)}$, where $g = 1, \ldots, 2^{n-1}$ and $j = 1, \ldots, n - 1$.*

Proof: The proof is by induction. For $g = 1$ only the root node $([1, 2, \ldots, n], 0)$ is derived. The initial values are $r_j^{(1)} = \text{RSS}([1, 2, \ldots, j])$. From (3.1) and $[1, \ldots, j] \subset [1, 2, \ldots, j, j+1]$ it follows that $r_j^{(1)} \geq r_{j+1}^{(1)}$ for $j = 1, \ldots, n-1$. The inductive hypothesis is that Lemma 1 holds for $1 \leq g < 2^{n-1}$. It is required to show that $r_j^{(g+1)} \geq r_{j+1}^{(g+1)}$ $(j = 1, \ldots, n - 1)$ after the $(g + 1)$th node of the regression tree $T([1, 2, \ldots, n], 0)$ has been generated.

Consider the $(g + 1)$th node $(\text{Drop}(V, i), i - 1)$ which has been derived from (V, k), where $0 \leq k < |V|$ and $k < i < |V|$. The $(g + 1)$th node provides the subsets

$$W_j = [v_1, \ldots, v_{i-1}, v_{i+1}, \ldots, v_{j+1}], \quad \text{where} \quad j = i, \ldots, |V| - 1.$$

The affected values are

$$r_j^{(g+1)} = \min(r_j^{(g)}, \text{RSS}(W_j)) \quad \text{for} \quad j = i, \ldots, |V| - 1.$$

The $r_j^{(g+1)}$, when modified, becomes

$$r_j^{(g+1)} = \begin{cases} \text{RSS}(W_j) \geq \text{RSS}(W_{j+1}) \geq \min(r_{j+1}^{(g)}, \text{RSS}(W_{j+1})) = r_{j+1}^{(g+1)} & \text{if } i \leq j < |V|-1, \\ \text{RSS}(W_{|V|-1}) \geq \text{RSS}(V) \geq r_{j+1}^{(g)} = r_{j+1}^{(g+1)} & \text{if } j = |V|-1. \end{cases}$$

Otherwise $r_j^{(g+1)} = r_j^{(g)}$ and from the inductive hypothesis it follows that $r_j^{(g+1)} = r_j^{(g)} \geq r_{j+1}^{(g)} \geq r_{j+1}^{(g+1)}$. This completes the proof. □

Lemma 2. *$r_{|W|}^{(g)} \leq RSS(W)$, where W is any set obtained from a node of $T(V, i-1)$, $r_i^{(g)} \leq RSS(V)$ and $1 \leq i < |V|$.*

Proof: Let $W = [w_1, \ldots, w_{|W|}]$ denote any set obtained from a node of $T(V, i-1)$. From the definition of $T(V, i-1)$ it follows that $W \subseteq V$ and contains the first $i-1$ variables of V. Thus, W can be written as $[v_1, \ldots, v_{i-1}, w_i, \ldots, w_{|W|}]$, where $i \leq |W|$. From Lemma 1 and (3.1) it follows, respectively, that $r_{|W|}^{(g)} \leq r_i^{(g)}$ and $\text{RSS}(V) \leq \text{RSS}(W)$. Since $r_i^{(g)} \leq \text{RSS}(V)$ the latter implies $r_{|W|}^{(g)} \leq \text{RSS}(W)$. □

Corollary 1. *$(1-\tau) r_{|W|}^{(g)} \leq RSS(W)$, where W is any set obtained from a node of $T(V, i-1)$, $0 \leq \tau < 1$, $(1-\tau) r_i^{(g)} \leq RSS(V)$ and $1 \leq i < |V|$.*

Proof: From the proof of Lemma 2 it results that $r_{|W|}^{(g)} \leq r_i^{(g)}$ and since $(1-\tau) r_i^{(g)} \leq \text{RSS}(V)$ it follows that $(1-\tau) r_{|W|}^{(g)} \leq \text{RSS}(V) \leq \text{RSS}(W)$ which completes the proof. □

Now, let (V, k) be one of the g generated nodes of the regression tree $T([1, 2, \ldots, n], 0)$, where $0 \leq k < |V| \leq n$ and $1 \leq g \leq 2^{n-1}$. Consider also that the children nodes $(\text{Drop}(V, k+1), k), \ldots, (\text{Drop}(V, i-1), i-2)$ have been generated, where $k+1 \leq i < |V|$. The remaining subtrees $T(\text{Drop}(V, \rho), \rho-1)$ need to be derived for $\rho = i, \ldots, |V|-1$. The bound of the node (V, k) is denoted by $b_V \equiv \text{RSS}(V)$. If $r_i^{(g)} \leq b_V$ then, from Lemma 2 it follows that $r_{|W|}^{(g)} \leq \text{RSS}(W)$, where W is any subset obtained from a node of $T(V, i-1)$. The subtrees of $T(V, i-1)$ are given by $T(\text{Drop}(V, \rho), \rho-1)$, where $\rho = i, \ldots, |V|-1$. This implies that the remaining subtrees of (V, k) cannot improve the values of $r_j^{(g)}$, where $j = 1, \ldots, n$. This suggests that prior to the computation of $(\text{Drop}(V, i), i-1)$ the $r_i^{(g)}$ is compared with the bound b_V, where $i = k+1, \ldots, |V|-1$. Specifically, if $r_i^{(g)} \leq b_V$, then all the subtrees $T(\text{Drop}(V, \rho), \rho-1)$ are eliminated for $\rho = i, \ldots, |V|-1$. Otherwise, $(\text{Drop}(V, i), i-1)$ is computed and $r_j^{(g+1)}$ $(j = i, \ldots, |V|-1)$ updated. This procedure is repeated for the next child, that is, $(\text{Drop}(V, i+1), i)$. Figure 3 illustrates this method, where $V = [3, 4, 5]$, $k = 0$, $i = 1$, $g = 13$, $(r_1^{(g)}, \ldots, r_5^{(g)}) \equiv (668, 615, 597, 592, 548)$ and $b_V = 720$. The RSS of each model is shown in brackets. Because $r_1^{(g)} \leq b_V$, it follows from Lemma 2 that the node comprising only the last variable, that is, $([5], 0)$, will have bigger RSS than $r_1^{(g)}$. This implies that all the children of $([3, 4, 5], 0)$, that is, $([4, 5], 0)$, $([3, 5], 1)$ and $([5], 0)$ need not to be computed.

The processing order of the nodes from $(\text{Drop}(V, k+1), k)$ to $(\text{Drop}(V, |V|-1), |V|-2)$ is the most efficient. If the subtree $T(\text{Drop}(V, i), i-1)$ is cut, then all the remaining subtrees, that is, $T(\text{Drop}(V, \rho), \rho-1)$ for $\rho = i, \ldots, |V|-1$ could also be cut. However, other processing orders will derive subtrees that become obsolete at a latter stage. The branch-and-bound procedure that generates the best-subset regression models is summarized by

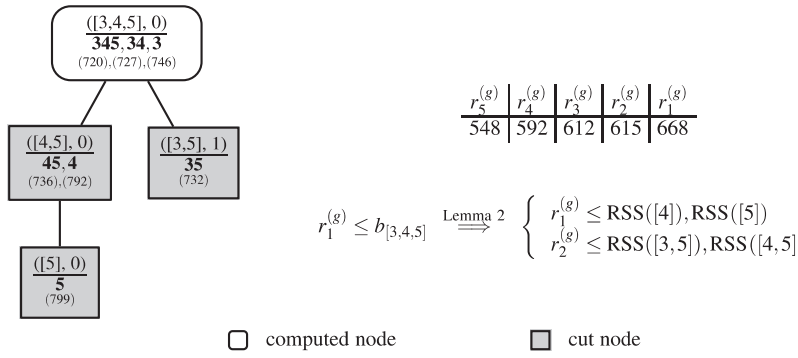


Figure 3. The cutting step in the subtree $T(V,k)$, where $V = [3, 4, 5]$ and $k = 0$.

Algorithm 1. Notice that the algorithm uses the recursive procedure ProcessSubtree, which processes the subtree with the root node (V,k). Figure 4 and Table 1 show, respectively, the regression tree and execution steps of Algorithm 1 for a five-variable model. The shadowed nodes (entries in cases of Table 1) are not computed, that is, cut.

Algorithm 1. Branch-and-bound procedure for finding the best-subset regression models.

1. Compute the QRD of $A \in \mathbb{R}^{m\times n}$: $Q^T A = \binom{R}{0}_{m-n}^{n}$ and $Q^T y = \tilde{y}$
2. Let $V = [1, 2, \ldots, n]$, $k = 0$ and $r_j = \text{RSS}([1, 2, \ldots, j]) \equiv \sum_{i=j+1}^{m} \tilde{y}_i^2$, where $j = 1, \ldots, n$
3. **call** ProcessSubtree(V, k)

4. **def** ProcessSubtree(V, k) = **do**
5. **for** $i = k + 1, \ldots, |V| - 1$ **do**
6. **if** $(r_i > \text{RSS}(V))$ **then**
7. $V^{(i)} \leftarrow \text{Drop}(V, i)$
8. $r_j = \min(r_j, \text{RSS}([v_1^{(i)}, v_2^{(i)}, \ldots v_j^{(i)}]))$, where $j = i, \ldots, |V| - 1$
9. **else**
10. Cut the remaining subtrees and go to Step 13
11. **end if**
12. **end for**
13. **call** ProcessSubtree$(V^{(j)}, j - 1)$, where $j = k + 1, \ldots, \min(i, |V| - 2)$
14. **end def**

The computational efficiency of the BBA improves when more nodes are cut. That is, if bigger subtrees are bounded with bigger values (Furnival and Wilson 1974). This can be achieved by preordering the variables in the initial set so that the node $([1, 2, \ldots, n], 0)$ satisfies

$$\text{RSS}(\text{Drop}(V, 1)) \ge \text{RSS}(\text{Drop}(V, 2)) \ge \cdots \ge \text{RSS}(\text{Drop}(V, n - 1)) \ge \text{RSS}(V). \quad (3.3)$$

That is, the most and less significant variables appear in the first and last position of V, respectively. The BBA that uses preordering (3.3) is denoted by BBA–1. Another approach,

Table 1. Execution Steps of the BBA for a Five-Variable Model

								min RSS for # of variables				
Step	*Node*	*Models (RSS)*					*Bound*	*5*	*4*	*3*	*2*	*1*
0.	([1,2,3,4,5], 0)	12345 (548)	1234 (592)	123 (612)	12 (615)	1 (668)	—	548	592	612	615	668
1.	([2,3,4,5], 0)		2345 (660)	234 (664)	23 (673)	2 (702)	548	548	592	612	615	668
2.	([1,3,4,5], 1)		1345 (605)	134 (612)	13 (641)		548	548	592	612	615	668
3.	([1,2,4,5], 2)		1245 (596)	124 (615)			548	548	592	612	615	668
4.	([1,2,3,5], 3)		1235 (592)				548	548	592	612	615	668
5.	([1,2,5], 2)			125 (597)			596	548	592	**597**	615	668
6.	([1,4,5], 1)			145 (618)	14 (648)		605	548	592	597	615	668
7.	([1, 3, 5], 2)			135 (618)			605	548	592	597	615	668
8.	([1,5], 1)				15 (618)		618	548	592	597	615	668
9.	([3,4,5], 0)			345 (720)	34 (727)	3 (746)	660	548	592	597	615	668
10.	([2,4,5], 1)			245 (667)	24 (685)		660	548	592	597	615	668
11.	([2,3,5], 2)			235 (666)			660	548	592	597	615	668
12.	([2,5], 1)				25 (675)		667	548	592	597	615	668
13.	([4,5], 0)				45 (736)	4 (792)	720	548	592	597	615	668
14.	([3,5], 1)				35 (732)		720	548	592	597	615	668
15.	([5], 0)					5 (799)	736	548	592	597	615	668

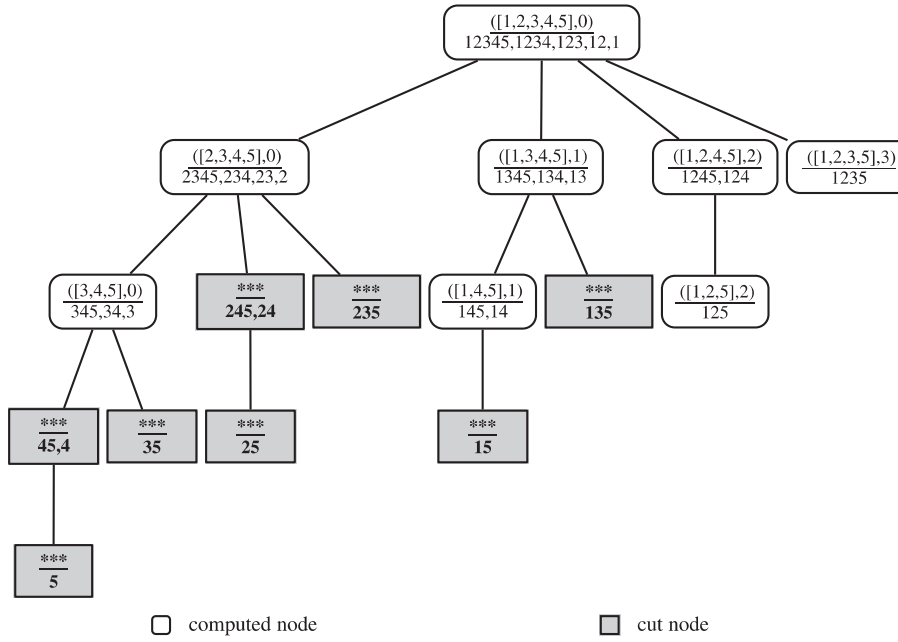


Figure 4. The regression tree $T(V, k)$ generated by the BBA, where $V = [1, \ldots, n]$ and $k = 0$.

BBA–2, which might cut bigger subtrees at early stages, processes first the node with the minimum bound. Thus, if $(V_1, k_1), \ldots, (V_d, k_d)$ denote the d $(0 < d \leq 2^{n-1})$ possible nodes that can be processed at some stage of the BBA, then the node that corresponds to $\min(\text{RSS}(V_1), \ldots, \text{RSS}(V_d))$ is processed first.

The BBA can be trivially modified to compute a list rather than a single best submodel for each number of variables. In this case, the current minimum values r_j used by the BBA are replaced by r_{jp} $(p = 1, \ldots, \gamma)$, where γ is the number of the best submodels to be computed comprising j variables, for $j = 1, \ldots, n$. The branch-and-bound cutting test is performed against the weakest value in the list of the best submodels, that is, $\max(r_{j1}, \ldots, r_{j\gamma})$.

For models with a large number of variables, the BBA might become computationally infeasible. In such cases various heuristics that provide solutions close to the optimum one should be applied (Winker 2001). One possibility is to cut subtrees (Step 6 of Algorithm 1) using a tolerance parameter. That is, when $(1 - \tau)r_i^{(g)} \leq \text{RSS}(V)$, where τ is the tolerance with $0 \leq \tau < 1$. Clearly this heuristic algorithm, HBBA, is equivalent to the BBA when $\tau = 0$. Generally, the closer τ is to zero, the greater the chance of obtaining a solution close to the optimum. Specifically, let W^* be a set obtained from the eliminated subtrees of $T(V, i-1)$ so that $\text{RSS}(W^*)$ is the optimal value of RSS for models with $|W^*|$ variables, that is,

$$\text{RSS}(W^*) = \min\{\text{RSS}(W); \quad W \subseteq V \text{ and } |W| = |W^*|\}.$$

The relative error of the RSS corresponding to models with $|W^*|$ variables provided by

HBBA is defined by

$$\zeta = \left(r_{|W^*|}^{(2^{n-1})} - \text{RSS}(W^*) \right) / \text{RSS}(W^*). \tag{3.4}$$

From Corollary 1 it follows that $r_{|W^*|}^{(g)} - \text{RSS}(W^*) \leq \tau r_{|W^*|}^{(g)}$, and since $r_{|W^*|}^{(2^{n-1})} \leq r_{|W^*|}^{(g)}$ it follows that

$$\zeta \leq \tau r_{|W^*|}^{(g)} / \text{RSS}(W^*) \leq \tau. \tag{3.5}$$

That is, the relative error ζ of the solution provided by the heuristic HBBA does not exceed the value of the tolerance parameter τ.

4. COMPUTATIONAL RESULTS

The number of nodes generated by the BBA depends on the model. If all the nodes of the regression tree need to be computed, then the BBA has the same complexity as that of the dropping-columns algorithm which generates all submodels. That is, an upper bound for the number of flops required by the BBA is given by (2.5) for $|V| = n$ and $k = 0$. This will be denoted by $C_{\text{BBA}}(n)$. The leaps-and-bounds algorithm (hereafter LBA) proposed by Furnival and Wilson (1974) generates two trees each having 2^{n-1} nodes. At each step of the LBA an inverse matrix is computed on the regressions tree and a pivoting (Gaussian elimination) is applied on the bounds tree. Both operations generate a new node. The flops required to generate a new node of size n in the regressions tree and to perform the pivoting on the bounds tree are given by $(20n^3 + 3n^2 + n)/6$ and $2n^2 - n$, respectively. Thus, the complexity of deriving a new node of size n, both on the regressions and the bounds trees, is given by

$$C_{\text{StepLBA}}(n) = 5(4n^3 + 3n^2 - n)/6. \tag{4.1}$$

The regressions tree generated by the LBA for an n-variable initial model has n levels. The nodes at level i ($i = 0, 1, \ldots n-1$) comprise $n - i$ variables and are obtained after deleting a combination of i variables from the original model. The number of possibilities of deleting these i variables from the model is given by $\text{C}_{n-1}^{i} = (n-1)!/(i!(n-1-i)!)$. Notice that the last variable is never deleted. Thus, the complexity of the LBA when generating all nodes is given by

$$\begin{aligned} C_{\text{LBA}}(n) &= \sum_{i=1}^{n-1} (\text{C}_{n-1}^{i} C_{\text{StepLBA}}(n-i)) \\ &= 5\left(2^n(2n^3 + 15n^2 + 13n - 6) - 8n(4n-1)(n+1)\right)/48. \end{aligned} \tag{4.2}$$

The ratio between the upper bound complexities of the LBA and BBA is

$$C_{\text{LBA}}(n)/C_{\text{BBA}}(n) \approx 0.0058(2n^3 + 15n^2 + 13n - 6) \equiv O(n^3).$$

Table 2. Execution Times in Seconds of the LBA and BBA

	Generating all models			*Cutting subtrees*			
	Time			*LBA*		*BBA*	
# of var	*LBA*	*BBA*	*LBA / BBA*	*Time*	*Nodes*	*Time*	*Nodes*
15	2.17	0.15	14	0.17	1958	0.02	969
16	4.55	0.28	16	0.15	1556	0.03	760
17	9.49	0.54	18	0.58	5932	0.08	2966
18	19.93	1.07	19	0.94	8870	0.13	4383
19	43.64	2.17	20	2.55	23316	0.29	11655
20	88.73	4.48	20	11.35	108806	1.07	54403
21	184.51	8.65	21	7.77	64348	0.80	32174
22	387.50	17.02	23	7.13	49994	0.72	24988
23	811.99	34.16	24	8.18	57060	0.81	28511
24	1739.70	68.20	26	62.76	454332	5.74	227159
25	3617.78	136.00	27	53.66	358008	5.60	178997
26	7610.29	270.20	28	73.93	443718	6.61	221854
27	15793.16	541.40	29	53.13	325488	4.60	162669
28	32381.89	1079.74	30	451.00	2649534	42.18	1324643
29	68327.62	2151.78	32	244.12	1264478	27.21	632231
30	142094.05	4317.85	33	221.25	1018050	22.30	509014

Table 2 shows the execution times (seconds) of the LBA and standard BBA used on models with different number of variables. The data were randomly generated from a uniform distribution and the predictors being uncorrelated with the response values. These data are used to evaluate the computational performance (execution time) of the different procedures. Table 2 also shows the time required to compute all the nodes of the regression tree. In this case, the BBA is equivalent to the dropping-columns algorithm in Gatu and Kontoghiorghes (2003). The models generated by the two algorithms are the same. The number of nodes processed by the algorithms are also shown. It can be observed that the BBA outperforms the LBA in terms of computational performance. The two algorithms have also been compared on different 20-variable initial models using random data. The BBA derived the best models 4 to 30 times faster than the LBA.

Table 3. Execution Times in Seconds for Various Versions of the BBA

	Exhaustive methods			*Heuristics with $\tau = 0.1$*		*Heuristics with $\tau = 0.25$*	
# of var	*BBA*	*BBA–1*	*BBA–2*	*HBBA*	*HBBA–1*	*HBBA*	*HBBA–1*
15	0.02	0.01	0.02	0.01	0.01	^{B}0.009	0.003
20	1.07	0.05	0.78	0.48	0.02	0.16	0.009
25	5.60	0.32	3.79	1.78	0.05	^{B} 0.20	0.010
30	22.30	1.09	18.23	3.46	0.04	1.46	0.005
35	171.64	3.01	117.12	42.47	0.32	4.77	0.040
40	10049.32	45.09	6644.69	168.17	1.31	^{B}12.27	0.030
41	3197.91	63.22	3163.20	80.94	1.12	^{B} 0.89	0.070
42	28176.72	76.09	20128.03	4949.46	3.15	255.20	0.090
43	31567.22	289.52	21293.12	1353.42	4.50	^{B}115.70	0.093
44	3806.57	89.07	2443.04	266.99	2.78	^{B} 11.93	0.086
45	47342.35	149.80	29566.28	2105.87	1.74	^{B}17.25	0.042

NOTE: ^{B} The execution time of HBBA is better than the one of the exhaustive BBA–1.

Table 4. Execution Times in Milliseconds of the HBBA

	Experimental data				*Random data*			
	OZONE		*POLLUTE*		*15 variables*		*30 variables*	
τ	*Time*	*Nodes*	*Time*	*Nodes*	*Time*	*Nodes*	*Time*	*Nodes*
0.0	(1)2.39	*143	(1)20.00	*710	(1)24.55	*969	(1)22590	*509014
0.01	2.28	*130	19.99	*608	22.94	*788	17540	371049
0.025	2.03	105	16.38	*523	21.28	645	12300	240875
0.05	1.85	90	14.99	438	17.14	430	6860	119364
0.1	1.38	60	11.83	335	14.37	267	3490	53234
0.25	0.69	21	6.80	134	8.10	99	1450	18993

(1) Equivalent to the execution time of the standard BBA. * The optimal models are found.

Table 3 shows the significant improvement in speed that can be achieved using various versions of the BBA. Recall that BBA–1 and BBA–2 are the standard BBA when preordering of variables in the root node and choosing first the node with minimum bound have been used, respectively. This improvement in speed is also shown for the case of the heuristic HBBA when a preordering of the variables is performed. This method is denoted by HBBA–1. The tolerances $\tau = 0.1$ and $\tau = 0.25$ are used in the experiments. Clearly, the BBA–1 is the most computationally efficient when an exhaustive search is performed to obtain the best models. The HBBA–1 brings a significant improvement in terms of computational performance when heuristics are used.

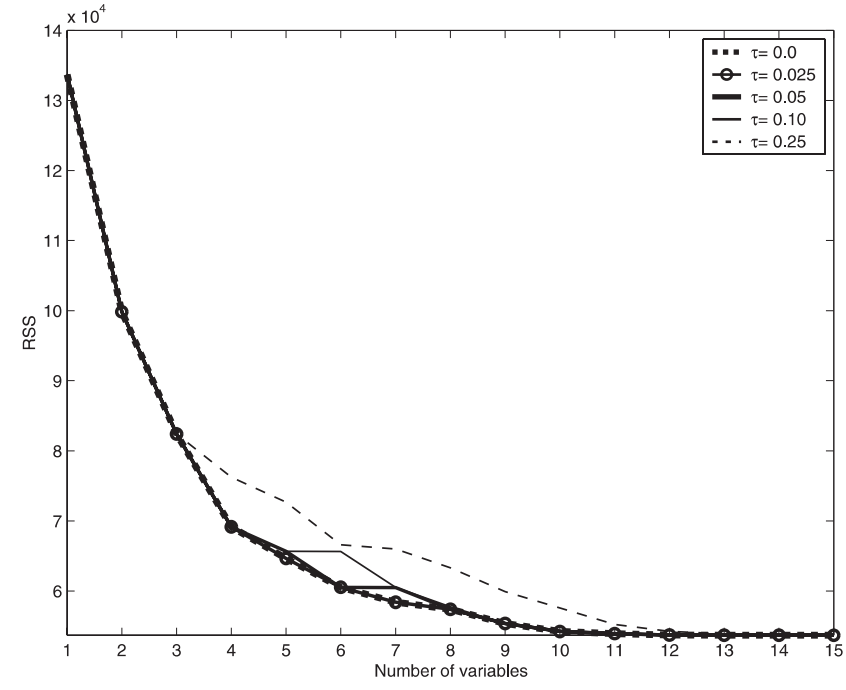
Tables 4 and 5 present the performance of the heuristics HBBA and HBBA–1, respectively, on experimental and random data for different values of tolerance τ. The experimental datasets Ozone (Hastie and Tibshirani 1990) and Pollute (Miller 2002) are used. The first dataset consists of daily measurements of ozone concentration and eight meteorological quantities for 330 days in 1976 in the Los Angeles basin. The latter dataset consists of 15 variables and is an age-adjusted mortality rate per 100,000 population. Notice that for $\tau = 0.0$ the HBBA and HBBA–1 perform an exhaustive search and are equivalent to the standard BBA and BBA–1, respectively. Figures 5 and 6 show the RSS and relative errors (3.4) of the models obtained by HBBA and HBBA–1 for the Pollute dataset.

From the experimental results the following observations can be made:

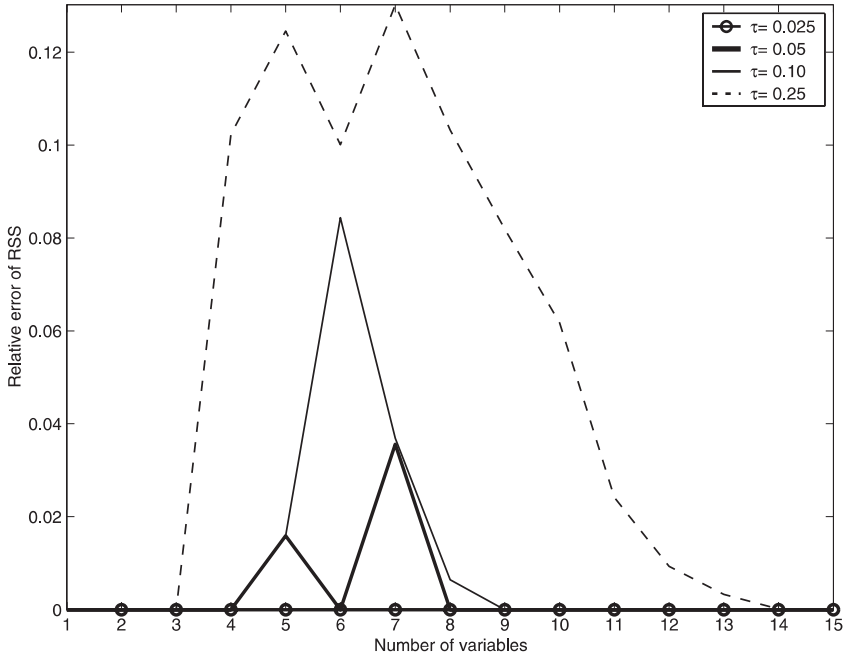
Table 5. Execution Times in Milliseconds of the HBBA–1

	Experimental data				*Random data*			
	OZONE		*POLLUTE*		*15 variables*		*30 variables*	
τ	*Time*	*Nodes*	*Time*	*Nodes*	*Time*	*Nodes*	*Time*	*Nodes*
0.0	(1)0.39	*13	(1)12.95	*381	(1)12.85	*237	(1)1068.81	*17229
0.01	0.26	*6	10.71	*302	12.22	205	704.22	10074
0.025	0.24	*5	9.42	*248	11.10	176	392.68	4794
0.05	0.21	*4	8.48	197	8.94	117	154.76	1430
0.1	0.17	*3	5.55	111	6.56	62	35.67	204
0.25	0.06	*2	2.68	36	2.48	16	4.80	18

(1) Equivalent to the execution time of the BBA–1. * The optimal models are found.

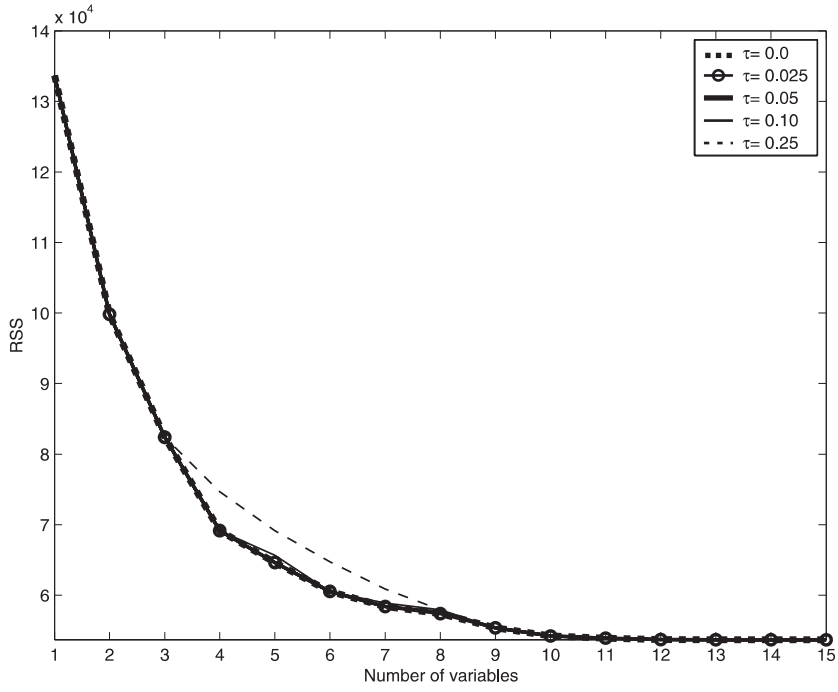


(a) The absolute values of the RSS

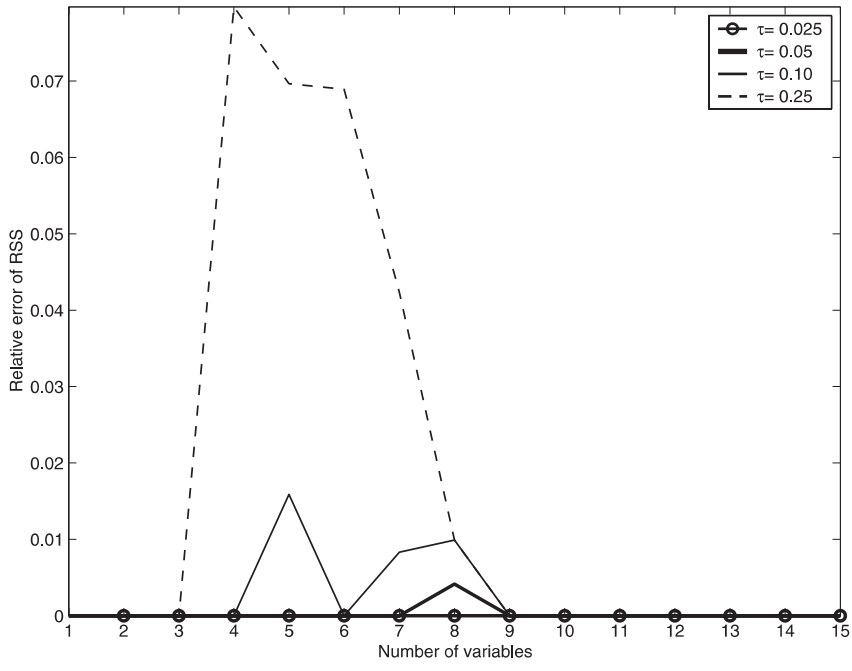


(b) The relative errors of the RSS

Figure 5. The results of the HBBA on the POLLUTE dataset with tolerance $\tau = 0.0, 0.025, 0.05, 0.1,$ and 0.25.



(a) The absolute values of the RSS



(b) The relative errors of the RSS

Figure 6. The results of the HBBA–1 on the POLLUTE dataset with tolerance $\tau = 0.0, 0.025, 0.05, 0.1,$ and 0.25.

- The BBA with preordering of the variables in the root node based on their individual significant value (BBA–1) is faster than the heuristic version of the BBA (HBBA) when the tolerance used is not very large.
- The heuristic version of the BBA–1 (HBBA–1) is significantly faster than the HBBA. This allows the HBBA–1 to use smaller tolerances and thus, to derive models close to the optimum ones.
- Furthermore, the HBBA–1 found models closer to the optimal one compared to the HBBA, when the same tolerance is used.
- The initial preordering of variables generates at early stages submodels which include the most significant variables, and thus smaller RSS, that is, bounds. This, in turn, results in the investigation of fewer nodes and consequently in a considerable reduction of the computational time.
- As the number of variables in the initial model increases, the tolerance of the heuristic algorithm needs to be reduced in order to obtain correct models.
- The relative error (3.4) and (3.5) can be used effectively when considering the tolerance of the heuristic algorithms. That is, the value of the tolerance can be manipulated, knowing that the relative error of the heuristic solution is less than the chosen tolerance.

5. CONCLUSIONS

A branch-and-bound algorithm (BBA) has been developed to compute the best-subset regression models corresponding to each number of variables. The algorithm exploits the properties of the regression tree that generates all submodels and avoids its computation. Specifically, it has been proven (Lemmas 1 and 2) that an entire subtree can be cut when the branch-and-bound test holds. The main computational tool is the QR decomposition and the residual sum of squares is used for selecting the models. The ratio between the upper bound complexities of the existing leaps-and-bounds algorithm (LBA) and the BBA is shown to be $O(n^3)$. Computational experiments confirm the theoretical results.

To increase the performance of the BBA, that is, to cut more nodes, two strategies are proposed. The first, BBA–1, preorders the variables prior to the execution of the algorithm so that bigger subtrees are bounded with bigger values. The second, BBA–2, selects the node at each step with the smallest bound from all possible nodes that can be processed. The experimental results show that the BBA–1 performs best when an exhaustive search is performed.

A heuristic version of the BBA (HBBA) that uses a tolerance parameter when deciding to cut a subtree is proposed. In this case, a subtree is also cut when the branch-and-bound test fails, but the relative improvement obtained by computing the subtree is smaller than the tolerance parameter. This HBBA significantly improves the computational performance of the BBA, especially when the variables are preordered. The HBBA might not provide the optimal solution. It is shown in (3.5) that the relative error of the computed solution is

smaller than the tolerance parameter used. The performance of the algorithms on various datasets has been investigated. For smaller tolerance parameters, the heuristic algorithm found the best models and reduced significantly the computation cost of the BBA which performs an exhaustive search.

It is expected that multicollinearity will have an effect in the computational performance of the BBA. Generally, more nonzero coefficients in the true model implies more cut branches of the regression tree and consequently smaller execution time. Thus, the impact of the data on the efficiency of the algorithm merits investigation.

The branch-and-bound algorithms and their heuristic versions can be extended and adapted to other subset model-selection problems. The algorithms need to be modified so that a restriction on the size of the selected models can be imposed. The dynamic calculation of the tolerance within the execution process of the HBBA, an extensive study of these methods to various models and their comparison with other (e.g., forward, backward, and greedy) selection methods should be investigated. Currently, the parallelization of the BBA on a cluster of workstations and the adaptation of the BBA to subset selection strategies for vector autoregressive and logistic regression models are being considered (Gatu and Kontoghiorghes 2005, in press; Zellner, Keller, and Zellner 2004).

ACKNOWLEDGMENTS

The authors are grateful to David A. Belsley, Alan J. Miller, the Editor and the four anonymous referees for their valuable comments and suggestions. This work is in part suported by the Swiss National Foundation Grants 101412-105978 and 200020-100116/1. Additional software can be found at http://iiun.uine.ch/matrix/.

[Received December 2003. Revised March 2005.]

REFERENCES

Allen, D. M. (1971), "Mean Square Error of Prediction as a Criterion for Selecting Variables," *Technometrics*, 13, 469–475.

Breiman, L. (1995), "Better Subset Regression Using Nonnegative GArrote," *Technometrics*, 37, 373–383.

Clarke, M. R. B. (1981), "Algorithm AS163. A Givens Algorithm for Moving from one Linear Model to Another Without Going Back to the Data," *Applied Statistics*, 30, 198–203.

Edwards, D., and Havránek, T. (1987), "A Fast Model Selection Procedure for Large Families of Models," *Journal of the American Statistical Association*, 82, 205–213.

Fan, J., and Li, R. Li (2001), "Variable Selection via Nonconcave Penalized Likelihood and its Oracle Properties," *Journal of the American Statistical Association*, 96, 1348–1360.

Furnival, G. M., and Wilson, R. W. (1974), "Regression by Leaps and Bounds," *Technometrics*, 16, 499–511.

Gatu, C., and Kontoghiorghes, E. J. (2003), "Parallel Algorithms for Computing all Possible Subset Regression Models using the QR Decomposition," *Parallel Computing*, 29, 505–521.

——— (2005), "Efficient Strategies for Deriving the Subset VAR Models," *Computational Management Science*, 2, 253–278.

——— (in press), "Estimating all Possible SUR Models With Permuted Exogenous Data Matrices Derived from a VAR Process," *Journal of Economic Dynamics and Control*.

Golub, G. H., and Van Loan, C. F. (1996), *Matrix Computations* (3rd ed.), Baltimore, MD: Johns Hopkins University Press

Goodnight, J. H. (1979), "A Tutorial on the SWEEP Operator," *The American Statistician*, 33, 116–135.

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

Hastie, T. J., Tibshirani, R. J., and Friedman, J. (2001), *The Elements of Statistical Learning. Data Mining, Inference, and Prediction*, New York: Springer-Verlag.

Hocking, R. R. (1972), "Criteria for Selection of a Subset Regression: Which One Should be Used?," *Technometrics*, 14, 967–970.

——— (1976), "The Analysis and Selection of Variables in Linear Regression," *Biometrics*, 32, 1–49.

——— (1983), "Developments in Linear Regression Methodology: 1959–1982," *Technometrics*, 25, 219–230.

Hocking, R. R., and Leslie, R. N. (1967), "Selection of the Best Subset in Regression Analysis," *Technometrics*, 9, 531–540.

Kontoghiorghes, E. J. (1995), "New Parallel Strategies for Block Updating the QR Decomposition," *Parallel Algorithms and Applications*, 5 229–239.

——— (1999), "Parallel Strategies for Computing the Orthogonal Factorizations used in the Estimation of Econometric Models," *Algorithmica*, 25, 58–74.

——— (2000a), *Parallel Algorithms for Linear Models: Numerical Methods and Estimation Problems*, Boston, MA: Kluwer.

——— (2000b), "Parallel Strategies for Rank–k Updating of the QR Decomposition," *SIAM Journal on Matrix Analysis and Applications*, 22, 714–725.

Kontoghiorghes, E. J., and Clarke, M. R. B. (1993a), "Solving the Updated and Downdated Ordinary Linear Model on Massively Parallel SIMD Systems," *Parallel Algorithms and Applications*, 2, 243–252.

——— (1993b), "Parallel Reorthogonalization of the QR Decomposition After Deleting Columns," *Parallel Computing*, 6, 703–707.

LaMotte, L. R., and Hocking, R. R. (1970), "Computational Efficiency in the Selection of Regression Variables," *Technometrics*, 12, 83–93.

Lawler, E. L., and Wood, D. E. (1966), "Branch-and-Bound Methods: A Survey," *Operations Research*, 14, 699–719.

Lee, E. K. (2002), "Branch-and-Bound Methods," in *Handbook of Applied Optimization*, eds. P. M. Pardalos and M. G. C. Resende, Cambridge, MA: Oxford University Press, pp. 53–65.

Mallows, C. L. (1995), "More Comments on C_P," *Technometrics*, 37, 362–372.

Miller, A. J. (1984), "Selection of Subsets of Regression Variables," *Journal of the Royal Statistical Society*, 147, 389–425.

——— (1992), "Algorithm AS 274: Least Squares Routines to Supplement those of Gentleman," *Applied Statistics*, 41, 458–478.

——— (2002), *Subset Selection in Regression*, New York: Chapman and Hall. Related software can be found online at http://users.bigpond.net.au/amiller/.

Narendra, P. M., and Fukunaga, K. (1977), "A Branch and Bound Algorithm for Feature Subset Selection," *IEEE Transactions on Computers*, C-26, 9, 917–922.

Ridout, M. S. (1988), "Algorithm AS 233: An Improved Branch and Bound Algorithm for Feature Subset Selection," *Applied Statistics*, 37, 139–147.

Roberts, S. J. (1984), "Algorithm AS 199: A Branch and Bound Algorithm for Determining the Optimal Feature Subset of Given Size," *Applied Statistics*, 33, 236–241.

Seber, G. A. F. (1977), *Linear Regression Analysis*, New York: Wiley.

Smith, D. M., and Bremner, J. M. (1989), "All Possible Subset Regressions using the QR Decomposition," *Computational Statistics and Data Analysis*, 7, 217–235.

Tibshirani, R. (1996), "Regression Shrinkage and Selection via the Lasso," *Journal of the Royal Statistical Society*, Series B, 58, 267–288.

Winker, P. (2001), *Optimization Heuristics in Econometrics: Applications of Threshold Accepting*, New York: Wiley.

Zellner, D., Keller, F., and Zellner, G. E. (2004), "Variable Selection in Logistic Regression Models," *Communications in Statistics-Simulation and Computation*, 3, 787–805.