

# Regressions by Enhanced Leaps-and-Bounds via Additional Optimality Tests (LBOT)

Xuelei (Sherry) Ni and Xiaoming Huo <sup>a</sup>

*Georgia Institute of Technology*

January 25, 2006

## Abstract

In exhaustive subset selection in regressions, the leaps-and-bounds algorithm by Furnival and Wilson (1974) is the current state-of-the-art. It utilizes a branch and bound strategy. We improve it by introducing newly designed optimality tests, retaining the original general framework. Being compared with the original leaps-and-bounds algorithm, the proposed method further reduces the number of subsets that are needed to be considered in the exhaustive subset search. Simulations demonstrate the improvements in numerical performance. Our new description of the leaps-and-bounds algorithm, which is based on our newly designed *pair tree*, is independent of programming languages, and therefore is more accessible.

**AMS 2000 Subject Classification.** Primary 62J05, secondary 62F07.

**Key Words and Phrases.** subset selection, leaps and bounds, optimality tests, inverse tree, pair tree.

**Acknowledgements.** Part of this research is supported by NSF grant #0140587.

<sup>a</sup>: School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0205. Email: xni, xiaoming@isye.gatech.edu

## 1 Introduction

We study the variable selection problem in a generic regression model. Regression model can be expressed as follows:

$$y = \Phi x + \varepsilon,$$

where  $y \in \mathbb{R}^n$  is a response vector,  $n$  is the number of observations, matrix  $\Phi \in \mathbb{R}^{n \times (m+1)}$ ,  $\Phi = [\mathbf{1}_n/\sqrt{n}, \phi_1, \dots, \phi_m]$ , is the model matrix with a constant column  $\phi_0 = \mathbf{1}_n/\sqrt{n}$  and covariates

$\phi_i \in \mathbb{R}^n, 1 \leq i \leq m$ , and vector  $\varepsilon \in \mathbb{R}^n$  is a random vector. The model selection is to choose a subset of  $\{\phi_i : i = 1, 2, \dots, m\}$ , so that the regression model based on the selected subset is as effective in prediction as the model built on the full set of covariates. There is a huge related literature in statistics, e.g., model estimation theory, which is not the theme of this paper. We will concentrate on the leaps-and-bound (LB) algorithm (Furnival and Wilson, 1974), which is a widely used subset selection method based on all-subsets comparisons. Recent papers – Kadane and Lazar (2004) and George (2000) – give excellent surveys on *subset selection*.

In Furnival and Wilson (1974), the following problem is solved: for all integer  $k, 1 \leq k \leq m$ ,

$$\begin{aligned} \text{(FW)} \quad & \min_x \quad \|y - \Phi x\|_2^2, \\ & \text{subject to: } \|x\|_0 = k, \end{aligned}$$

where  $\|\cdot\|_2^2$  denotes the sum of squares of the elements in a vector (i.e., the square of the  $\ell_2$  vector norm), and  $\|\cdot\|_0$  is the number of nonzero entries in a vector (which is also called the  $\ell_0$  quasi-norm). We name the problem (FW) to recognize the contribution of the original proposers of LB. Solving (FW) gives a way to realize model selection. It is connected with many widely used model selection methods, which can be summarized as the following optimization problem:

$$\min_x \quad \|y - \Phi x\|_2^2 + \lambda_0 \cdot \|x\|_0, \tag{1.1}$$

where  $\lambda_0$  is an algorithmic parameter. For AIC and  $C_p$ , we have  $\lambda_0 = 2\hat{\sigma}^2$ , where  $\hat{\sigma}^2$  is an unbiased estimate of the common variance of the random errors. For BIC and MDL, we have  $\lambda_0 = \sigma^2 \log n$ . We refer to Huo and Ni (2005) for more relevant information. The foregoing paper also proves that problem (1.1) is NP-hard. Readers may compare the difference between (1.1) and (FW). Note that solutions to (FW) lead to a solution to (1.1), with a small amount of additional computation.

Since the initial introduction of LB, little effort has been reported to improve this algorithm. The essence of the LB method is a branch and bound procedure, which uses tests to reduce the number of subsets that should be considered in an exhaustive subset search. In this paper, in the same branch-and-bound framework, we derive new optimality tests. It is shown that the induced additional tests can further reduce the number of subsets that are required to be considered. Hence, it accelerates LB. The derived method is named *leaps-and-bounds via optimality tests* (LBOT).

We briefly describe the motivation for the new tests. The original LB algorithm utilizes the following optimality test. Let  $A$  and  $B$  denote two distinct subsets of covariates, and assume that  $A$  is a subset of  $B$ :  $A \subset B$ . Let  $\text{RSS}(A)$  (resp.,  $\text{RSS}(B)$ ) denote the residual sum of squares of

the regression model that is built on subset  $A$  (resp.,  $B$ ). We have  $\text{RSS}(A) \geq \text{RSS}(B)$ . In this paper, more powerful optimality tests will be derived. The key idea is to derive a more strict necessary condition for a subset to outperform an existing optimal subset. *We will not only use the residual sums of squares, which is utilized in the LB, but also consider the coefficients and residuals associated with the optimal subsets.* Details regarding the derivation of such a condition are presented in Section 4.2.

Simulations demonstrate the improvement in performance. They also indicate the situations in which LB and its enhanced one – LBOT – are likely to significantly reduce the number of subsets that are needed to be computed. We will argue in this paper that the number of subsets that are examined is a good indicator of the computational complexity, because of its implementational independence. Some heuristics that can possibly improve the performance of our algorithms are tested, and the results are presented.

This paper is organized as follows. In Section 2, some basic results regarding the fast computation of RSS's and matrix inverse are given. In Section 3, a specific version of the LB in Furnival and Wilson (1974) is reviewed, and will serve as a starting point of our algorithmic description. In Section 4, additional optimality tests are derived. In Section 5, the newly derived optimality tests are integrated with LB, and the new leaps-and-bounds method (i.e., LBOT) is established. In Section 6, simulations are provided to demonstrate the improvements of performance. Some discussions and the conclusion are provided in sections 7 and 8, respectively. A shorter version of this paper is presented as a conference paper (Ni and Huo, 2005).

## 2 Review of Basics

Some relevant computational details are presented here. The ideas can be found in the original paper (Furnival and Wilson, 1974). The purposes of re-presenting them are

- to demonstrate that from one subset to a new subset, by inserting or deleting a covariate, there is an efficient numerical approach for the computation regarding the submodels;
- to make a point that the number of subsets that are needed to be computed in an algorithm (e.g., LB or LBOT) is an indicator of the complexity of this algorithm.

### 2.1 Computing Regarding Submodels

From now on, for simplicity, we assume that the covariates  $\phi_i, i = 1, 2, \dots, m$ , are standardized, i.e., for  $1 \leq i \leq m$ ,  $\text{ave}(\phi_i) = 0$ , and  $\|\phi_i\|_2 = 1$ , where  $\text{ave}(\cdot)$  (resp.,  $\|\cdot\|_2$ ) denotes the average

(resp.,  $\ell_2$ -norm) of a vector. It is evident that the correlation matrix among the response and the covariates is

$$(y, \Phi)^T (y, \Phi) = \begin{pmatrix} y^T y & y^T \Phi \\ \Phi^T y & \Phi^T \Phi \end{pmatrix}.$$

Moreover, the diagonal entries of the matrix  $\Phi^T \Phi$  are all equal to 1.

A submodel is determined by a subset of the covariates. Suppose subset  $\Omega$  ( $\Omega \subset \{1, 2, \dots, m\}$ ) determines a submodel. The corresponding model matrix is made by the constant column vector and the columns having indices from  $\Omega$ . The model matrix is denoted by  $\Phi_\Omega$ :  $\Phi_\Omega = [\mathbf{1}_n/\sqrt{n}, \{\phi_i\}_{i \in \Omega}]$ . Note the first column corresponds to the intercept term, which is also included in submodels. Let  $\hat{\beta}(\Omega)$  denote the least square fit on this submodel, we have

$$\hat{\beta}(\Omega) = (\Phi_\Omega^T \Phi_\Omega)^{-1} \Phi_\Omega^T \cdot y. \quad (2.2)$$

Let  $\text{RSS}(\Omega)$  denote the residual sum of squares of the least square fit, we have

$$\text{RSS}(\Omega) = y^T y - (\Phi_\Omega^T \cdot y)^T (\Phi_\Omega^T \Phi_\Omega)^{-1} (\Phi_\Omega^T \cdot y). \quad (2.3)$$

Note  $\Phi_\Omega^T \cdot y$  is a subvector of  $\Phi^T \cdot y$ , which can be handily read from the correlation matrix.

## 2.2 Two Basic Linear Algebra Results

The following simple linear algebra results show that when adding or deleting one covariate, the resulting inverse matrix can be computed efficiently. The original LB paper also took advantage of these facts; however, their description is less direct.

**Lemma 2.1** *For a positive integer  $k$ , given symmetric matrix  $M \in \mathbb{R}^{k \times k}$  with its inverse  $M^{-1}$ , a vector  $v \in \mathbb{R}^k$ , and a constant  $c \in \mathbb{R}$ , we have*

$$\begin{pmatrix} M & v \\ v^T & c \end{pmatrix}^{-1} = \begin{pmatrix} M^{-1} + \tau M^{-1} v v^T M^{-1} & -\tau M^{-1} v \\ -\tau v^T M^{-1} & \tau \end{pmatrix},$$

where scalar  $\tau = (c - v^T M^{-1} v)^{-1}$ .

The following is an easy extension.

**Corollary 2.2** *Given the same notations as in Lemma 2.1, if we have*

$$\begin{pmatrix} M & v \\ v^T & c \end{pmatrix}^{-1} = \begin{pmatrix} B_{11} & B_{12} \\ B_{12}^T & B_{22} \end{pmatrix},$$

where  $B_{11} \in \mathbb{R}^{k \times k}$ ,  $B_{12} \in \mathbb{R}^{k \times 1}$ , and  $B_{22} \in \mathbb{R}$ , then the following holds:

$$M^{-1} = B_{11} - B_{12} B_{12}^T / B_{22}.$$

Let  $I(\Omega)$  denote the inverse matrix  $(\Phi_{\Omega}^T \Phi_{\Omega})^{-1}$ . For  $i \in \Omega$  and  $j \notin \Omega$ , from Lemma 2.1 (resp., Corollary 2.2), there is a fast way to compute  $I(\Omega \cup \{j\})$  (resp.,  $I(\Omega \setminus \{i\})$ ). The following is in the original LB paper (Furnival and Wilson, 1974). Readers can easily verify it.

**Lemma 2.3** *For above mentioned indices  $i$  and  $j$ , subset  $\Omega$ , and integer  $k = |\Omega|$ , which is the size of subset  $\Omega$ , it takes  $O(k^2)$  numerical operations (additions, subtractions, multiplications, and divisions) to generate the inverse matrices corresponding to adding/deleting one covariate to/from the subset  $\Omega$ .*

### 3 Subset Arrangement and the Leaps-and-Bounds Algorithm

The ingenious idea in the original LB paper (Furnival and Wilson, 1974) is to introduce a systematic way to scan through all the subsets, at the same time, ‘leaping’ over those evidently nonoptimal subsets. Here, we redescribe their scheme. The pairing structure and the pair tree in Section 3.2 are new, which is motivated by the description in Furnival and Wilson (1974). Because the original description requires knowledge in Fortran language and two trees, we believe our description is more understandable.

#### 3.1 Inverse Tree

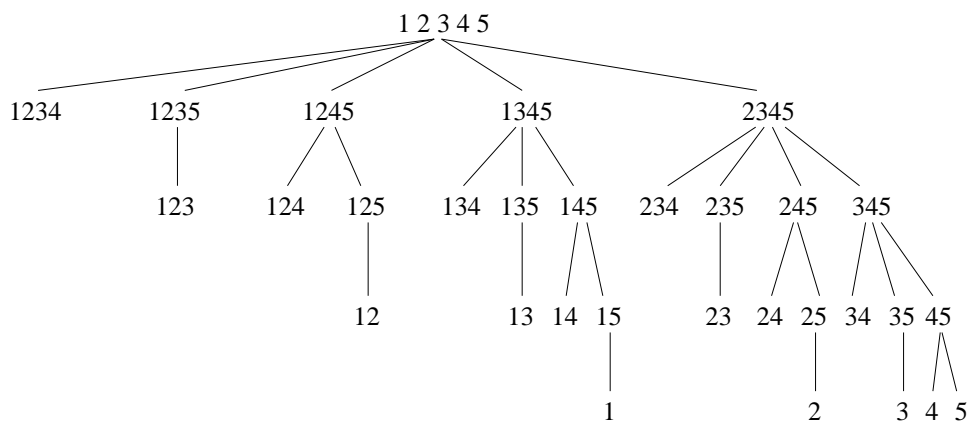


Figure 1: An inverse tree with  $m = 5$ . For simplicity, in figures, “1234” is equivalent with subset  $\{1234\}$  in the text.

Figure 1 gives an inverse tree with  $m = 5$  covariates. Its construction is in the original LB paper. For completeness, we briefly describe it in the following.

1. The root node is the full set  $\{1, 2, \dots, m\}$ .

2. Level 1 is made by  $m$  ordered children of the root node by removing one covariate at a time from the full set at the decreasing order:  $m, m - 1, \dots, 2, 1$ .
3. Consider a node associated with subset  $\{i_1 i_2 \dots i_k\}, k \geq 1, i_1 < i_2 < \dots < i_k$ . Assume it is the  $j$ th ( $j \geq 1$ ) child of its parent. At the next level, this node has  $j - 1$  children that are generated by deleting one covariate at a time from the set  $\{i_1 i_2 \dots i_k\}$  with the order  $i_k, i_{k-1}, \dots, i_{k+2-j}$ .
4. The tree stops growing when it reaches the subsets made by one covariate, or all terminal nodes are the first children.

Readers can easily verify the following facts, which are collectively presented in a theorem.

**Theorem 3.1** *The inverse tree has the following properties:*

- *The above constructed inverse tree contains all the  $2^m - 1$  subsets of  $\{1, 2, \dots, m\}$ .*
- *Each subset appears once and only once in this tree.*
- *The sizes of the subsets associated with the nodes at level  $k$  ( $1 \leq k \leq m - 1$ ) of this tree are equal to  $m - k$ .*
- *Each subset associated with a node in this tree, except the root node, is obtainable by removing one covariate from the subset associated with its parent node.*

The following observation will be utilized in a new description of the LB algorithm.

**Theorem 3.2** *In the inverse tree with  $m$  covariates, the subtree rooted at node  $\{2, 3, \dots, m\}$  has the identical structure with the subtree started at the original root node, after pruning the subtree rooted at node  $\{2, 3, \dots, m\}$  and ignoring the only terminal nodes at the bottom level:  $\{1\}$ . Moreover, if  $\Omega$  is a subset in the subtree rooted at node  $\{2, 3, \dots, m\}$ , the subset  $\Omega \cup \{1\}$  is associated with the node at the same position in the latter pruned tree.*

## 3.2 Pair Tree

The original description of the LB method is based on two trees: *regression* tree and *bound* tree. Being inspired by these two trees, we construct the following pairing scheme and a new tree for the pairs of subsets, so that the subsets searching and leaping can be realized based on one single structure. We believe this new scheme gives a more intuitive description. Figure 2 gives such a pair tree for the same case ( $m = 5$ ) as depicted in Figure 1.

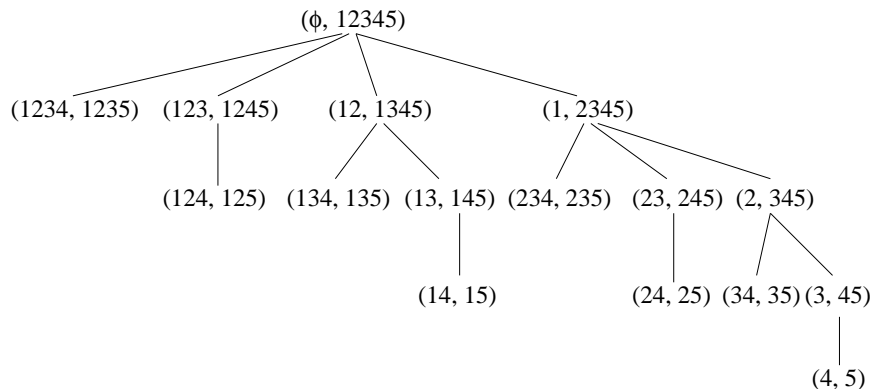


Figure 2: A pair tree with  $m = 5$ .

A pair tree can be constructed by using induction. Readers can use Theorem 3.2 to verify that the following construction is well defined. For  $m = 2$ , the pair tree (denoted as  $PT(2)$ ) is

$$\begin{array}{c}
 (\emptyset, \{12\}) \\
 \downarrow \\
 (\{1\}, \{2\})
 \end{array}$$

For  $m = 3$ , the corresponding pair tree is constructed by the following three steps:

- Transfer index  $i$  ( $i = 1, 2$ ) in tree  $PT(2)$  to  $i + 1$ . The generated tree is called  $T_1$ .
- Take a  $T_1$ , insert  $\{1\}$  in all the nonempty subsets. The new tree is called  $T_2$ .
- Take another  $T_1$ , convert  $\emptyset$  to  $\{1\}$ , and make it an additional subtree of  $T_2$  by making the root node of the modified  $T_1$  a new child of the root node of  $T_2$ . The new child is at the first level of  $T_2$ . The combined pair tree is  $PT(3)$ .

The following depicts  $PT(3)$ :

$$\begin{array}{ccc}
 & (\emptyset, \{123\}) & \\
 \swarrow & & \searrow \\
 (\{12\}, \{13\}) & & (\{1\}, \{23\}) \\
 & & \downarrow \\
 & & (\{2\}, \{3\})
 \end{array}$$

In general, given  $PT(m)$ ,  $PT(m + 1)$  is generated by three steps: (1) Transfer index  $i$  ( $i = 1, \dots, m$ ) in tree  $PT(m)$  to  $i + 1$ . The generated tree is called  $T_1$ . (2) Take a  $T_1$ , insert  $\{1\}$  in all

the nonempty subsets. The new tree is called  $T_2$ . (3) Take another  $T_1$ , convert  $\emptyset$  to  $\{1\}$ , make it an additional subtree of  $T_2$  by making the root node of the modified  $T_1$  a new child of the root node of  $T_2$ . The result pair tree is  $PT(m+1)$ . Readers can observe the strong parallelism to the previous description. In fact, it is a generalization.

We can easily verify the following.

**Theorem 3.3** *Consider a pair tree for  $m$  covariates (i.e.,  $PT(m)$ ).*

1. *In the aforementioned pair tree, each subset of  $\{1, 2, \dots, m\}$  appears once and only once.*
2. *For an intermediate node  $(\Omega_1, \Omega_2)$ , all the subsets in the descendant nodes are the subsets of  $\Omega_2$ . This indicates that the order in a pair can not be changed.*
3. *For integers  $1 \leq k_1, k_2 \leq m$ , suppose two subsets in a pair contain  $k_1$  and  $k_2$  covariates, respectively. The sizes of the subsets in the descendent nodes are at least  $\min(k_1, k_2)$ . Such a fact is utilized in the original LB algorithm.*
4. *Consider a node  $(\Omega_1, \Omega_2)$  and one of its children  $(\Omega'_1, \Omega'_2)$ . If  $(\Omega'_1, \Omega'_2)$  is the first child of  $(\Omega_1, \Omega_2)$ , then subset  $\Omega'_1$  (resp.,  $\Omega'_2$ ) is a subset of  $\Omega_2$  by removing the last (resp. the second last) covariate from  $\Omega_2$ . If  $(\Omega'_1, \Omega'_2)$  is not the first child, assuming  $(\Omega'_3, \Omega'_4)$  is the child of  $(\Omega_1, \Omega_2)$  and, in the pair tree, is immediately in the left hand side of  $(\Omega'_1, \Omega'_2)$ , then subset  $\Omega'_1$  is obtained by removing the last covariate from subset  $\Omega'_3$  and subset  $\Omega'_2$  is obtained by removing one covariate from  $\Omega_2$ . In summary, subsets of a particular node can be obtained by removing one covariate from a subset in its parent and possibly its left sibling. This relation ensures an efficient numerical approach to scan through all the nodes; more specifically, an efficient scan moves top down and left to right in the pair tree.*

### 3.3 Test in the Original Leaps-and-Bounds Algorithm

Now, after the analysis of the inverse tree and the pair tree, we are ready to give a new description of the LB method. We still use the case of 5 covariates as an example. Recall the contents of Section 2.2. The following inverse matrices can be computed, according to the scheme below:

$$\begin{aligned}
I(\{1\}) &\rightarrow I(\{12\}) \rightarrow I(\{123\}) \rightarrow I(\{1234\}) \rightarrow I(\{12345\}), \\
I(\{12345\}) &\rightarrow I(\{1235\}), \\
I(\{12345\}) &\rightarrow I(\{1245\}), \\
I(\{12345\}) &\rightarrow I(\{1345\}), \\
I(\{12345\}) &\rightarrow I(\{2345\}),
\end{aligned}$$

where each ‘ $\rightarrow$ ’ involves inserting/removing one covariate to/from the subset on the left hand side. The consequent residual sums of squares can be computed correspondingly by (2.3).

In the pair tree, we have the residual sums of squares of the subsets included in the root node and the nodes in the first level. We consider the remaining nodes. Whether or not to compute  $RSS(\{124\})$  and  $RSS(\{125\})$  depends on the values of  $RSS(\{1245\})$  and  $RSS(\{123\})$ . If  $RSS(\{123\}) \leq RSS(\{1245\})$ , because  $\{124\}$  and  $\{125\}$  are subsets of  $\{1245\}$ , we immediately have  $RSS(\{123\}) \leq RSS(\{124\})$  and  $RSS(\{123\}) \leq RSS(\{125\})$ . Hence, there is no need to compute for  $RSS(\{124\})$  and  $RSS(\{125\})$ . Otherwise, they should be computed.

Similarly, whether or not to compute  $RSS(\{134\})$  and  $RSS(\{135\})$  (or  $RSS(\{13\})$  and  $RSS(\{145\})$ ) depends on three values:  $RSS(\{12\})$ ,  $RSS(\{1345\})$ , and  $\min(RSS(\{123\}), RSS(\{124\}), RSS(\{125\}))$  (denoted as  $\underline{RSS}(3)$ ). Note that  $\underline{RSS}(3) \leq RSS(\{12\})$ . There are three cases for those three values:

- If  $RSS(\{12\}) \leq RSS(\{1345\})$ , then none of  $RSS(\{134\})$ ,  $RSS(\{135\})$ ,  $RSS(\{13\})$ , or  $RSS(\{145\})$  needs to be calculated.
- If  $\underline{RSS}(3) \leq RSS(\{1345\}) < RSS(\{12\})$ , then only  $RSS(\{13\})$  and  $RSS(\{145\})$  need to be calculated to update the minimum RSS with 2 covariates.
- If  $RSS(\{1345\}) < \underline{RSS}(3)$ , then all of the four RSS’s need to be calculated.

Repeating this step through the entire inverse tree gives the original LB algorithm.

In general, the original LB algorithm is equivalent to scanning through the pair tree according to the following scheme.

- Compute the residual sums of squares for all the subsets in the root node and the nodes in level 1 of the pair tree.
- Suppose  $(\Omega_1, \Omega_2)$  is an intermediate node in the pair tree, and  $RSS(\Omega_1)$  and  $RSS(\Omega_2)$  have been computed. In our construction, readers can verify that we have  $|\Omega_1| \leq |\Omega_2|$ , where  $|\cdot|$  is the size of a subset. For  $|\Omega_1| \leq k \leq |\Omega_2|$ , let  $\underline{RSS}(k)$  denote the minimum of the residual sum of squares of all the  $k$ -subsets that have been scanned up to this point. If  $RSS(\Omega_2) \geq \underline{RSS}(k)$ , for all  $|\Omega_1| \leq k \leq |\Omega_2|$ , then the computations for the descendants of node  $(\Omega_1, \Omega_2)$  can be ignored, because none of them can be an optimal solution to (FW). Otherwise, we should consider at least partial of the descendants of  $(\Omega_1, \Omega_2)$ .

Readers can easily verify the following result.

**Lemma 3.4** *In a top-down and left-to-right scheme to scan through the pair tree, the following inequality is true,*

$$\underline{RSS}(k+1) \leq \underline{RSS}(k),$$

for any  $k$  that is applicable.

Hence, in the original LB algorithm, we have the following cases:

- If

$$\underline{\text{RSS}}(|\Omega_1|) \leq \text{RSS}(\Omega_2),$$

then skip all the descendants of node  $(\Omega_1, \Omega_2)$ . Because none of the subset in a descendants of the node  $(\Omega_1, \Omega_2)$  can have a smaller residual sum of squares than the corresponding existing  $\underline{\text{RSS}}(k)$ 's.

- If

$$\underline{\text{RSS}}(|\Omega_2| - k) \leq \text{RSS}(\Omega_2) < \underline{\text{RSS}}(|\Omega_2| - k - 1)$$

for certain  $k$ , where  $1 \leq k \leq |\Omega_2| - |\Omega_1| - 1$ , then we can skip the first  $k$  children of node  $(\Omega_1, \Omega_2)$ .

- If

$$\text{RSS}(\Omega_2) < \underline{\text{RSS}}(|\Omega_2| - 1),$$

then none of the children of  $(\Omega_1, \Omega_2)$  can be skipped.

In summary, the optimality tests in LB completely depends on the values of residual sums of squares.

## 4 Additional Optimality Tests

To the best of our knowledge, little effort has been reported to bring new optimality test. In this section, additional tests are derived. The key intuition is to bring in the considerations of the coefficients and residuals in the up-to-date optimal solutions. In comparison, the original LB method only considers the values of residual sums of squares. Additional optimality tests, together with the original test, will reduce the number of subsets that are needed to be considered. Hence, it reduces the computational requirement.

### 4.1 New Tests

We now consider additional optimality tests for node  $(\Omega_1, \Omega_2)$  in the pair tree. The following notations will be used:

- Let  $\Omega_{(k)}$  be the  $k$ -subset associated with the minimum residual sum of squares  $\underline{\text{RSS}}(k)$ .

- Let  $\widehat{\beta}_{(k)} = \widehat{\beta}(\Omega_{(k)})$  denote the coefficients of the least square fit on the subset  $\Omega_{(k)}$ , whose computation is given in (2.2).

- Denote a residual vector

$$\varepsilon_{(k)} = y - \Phi_{\Omega_{(k)}} \widehat{\beta}_{(k)}.$$

- Recall  $(\Omega_1, \Omega_2)$  is a pair of subsets in the pair tree. Recall  $\phi_a$  and  $\phi_b$  are standardized covariates. A new quantity  $\mu$  is defined as follows:

$$\mu = \max_{\substack{a, b \in \Omega_2 \cup \Omega_{(k)} \\ a \neq b}} |\langle \phi_a, \phi_b \rangle|.$$

Quantity  $\mu$  is the maximum absolute value of the correlation within the subset  $\Omega_2 \cup \Omega_{(k)}$ .

- Define  $k_1(k) = \min(2k, |\Omega_2 \cup \Omega_{(k)}|)$ . Quantities  $\mu$  and  $k_1(k)$  are easily computable.
- For an arbitrary vector  $v$  and an arbitrary integer  $k_2$ , assuming that the dimension of  $v$  is no less than  $k_2$ , we define

$$\|v\|_{(k_2)} = \sqrt{\sum_{j=1}^{k_2} |v|_{(j)}^2},$$

where  $|v|_{(1)} \geq |v|_{(2)} \geq |v|_{(3)} \geq \dots$  are ordered absolute values of the entries of vector  $v$ .

- It is easy to observe that vector  $\Phi^T \varepsilon_{(k)}$  is an  $(m+1)$ -dimensional vector, which is handly computable. We define vector  $\Phi_{\Omega_2 \cup \Omega_{(k)}}^T \varepsilon_{(k)}$  as a subvector of  $\Phi^T \varepsilon_{(k)}$  by taking covariate indices in the subset  $\Omega_2 \cup \Omega_{(k)}$ .

The following theorem points out a new optimality test.

**Theorem 4.1 (Optimality Rule)** *For the previously defined  $\Omega_1, \Omega_2, \Omega_{(k)}, \varepsilon_{(k)}, k_1(k)$  (simplified as  $k_1$ ) and  $\mu$ . For  $|\Omega_1| \leq k \leq |\Omega_2|$ , define a set  $\Theta(k)$  of covariate indices such that  $i \in \Theta(k)$  if and only if  $i \in \Omega_{(k)}$  and*

$$|(\widehat{\beta}_{(k)})_i| > \frac{\|\Phi_{\Omega_2 \cup \Omega_{(k)}}^T \varepsilon_{(k)}\|_{\infty} + \|\Phi_{\Omega_2 \cup \Omega_{(k)}}^T \varepsilon_{(k)}\|_{(k_1)}}{1 - (k_1 - 1)\mu}, \quad (4.4)$$

where  $(\widehat{\beta}_{(k)})_i$  denotes the coefficient of the  $i$ th covariate in  $\widehat{\beta}_{(k)}$ . If a subset  $\Omega$  ( $\Omega \subset \Omega_2$  and  $|\Omega| = k$ ) achieves  $RSS(\Omega) \leq \underline{RSS}(k)$ , then we must have  $\Theta(k) \subset \Omega$ .

If a  $k$ -subset achieves a residual sum of squares that is less than  $\underline{RSS}(k)$ , then  $\Theta(k)$  is a subset of this  $k$ -subset. Hence, in the pair tree, any descendant that does not include  $\Theta(k)$  as a subset cannot achieve a residual sum of squares less than  $\underline{RSS}(k)$ . This fact can be used to screen out some descendants of the nodes in a pair tree.

## 4.2 Proof of Theorem 4.1

The key idea adopted in the proof is to find a sufficient condition for a subset, such that this subset can not achieve a smaller residual sum of square than the one that corresponds to the up-to-date optimal subset having the same size.

We use the same notations as in the previous subsection. Let  $\Omega$  be a subset of  $\Omega_2$ :  $\Omega \subset \Omega_2$ . Let  $\delta$  denote a vector that satisfies the following rules.

- $\delta$  only takes possibly nonzero values at position  $i$  when  $i \in \Omega \cup \Omega_{(k)}$ .
- Let  $\widehat{\beta}(\Omega)$  denote the coefficient of the least square fit when the subset is  $\Omega$ . Given previously defined  $\widehat{\beta}_{(k)}$ , the entries of  $\delta$  are given as follows:

$$\delta_i = \begin{cases} (\widehat{\beta}(\Omega))_i - (\widehat{\beta}_{(k)})_i, & \text{if } i \in \Omega \cap \Omega_{(k)}, \\ (\widehat{\beta}(\Omega))_i, & \text{if } i \in \Omega, \text{ however } i \notin \Omega_{(k)}, \\ -(\widehat{\beta}_{(k)})_i, & \text{if } i \in \Omega_{(k)}, \text{ however } i \notin \Omega, \\ 0, & \text{elsewhere,} \end{cases}$$

where  $(\cdot)_i$  denotes the value of the coefficient corresponding to the covariate  $i$  in the coefficient vector.

The following derives a necessary condition for  $\text{RSS}(\Omega) \leq \underline{\text{RSS}}(k) = \text{RSS}(\Omega_{(k)})$ . We start with the following inequality:

$$\|y - \Phi_{\Omega} \widehat{\beta}(\Omega)\|_2^2 \leq \|\varepsilon_{(k)}\|_2^2.$$

The above is equivalent to the following:

$$\|\varepsilon_{(k)} - \Phi_{\Omega \cup \Omega_{(k)}} \delta\|_2^2 \leq \|\varepsilon_{(k)}\|_2^2,$$

which is equivalent to the following inequality:

$$\|\Phi_{\Omega \cup \Omega_{(k)}} \delta\|_2^2 \leq 2\langle \Phi_{\Omega \cup \Omega_{(k)}}^T \varepsilon_{(k)}, \delta \rangle, \quad (4.5)$$

where  $\langle \cdot, \cdot \rangle$  denotes the inner product of two vectors.

In order to prove the theorem, we will need the following two lemmas.

**Lemma 4.2** *Recall  $|\Omega_{(k)}| = k$ . Given  $\Omega \subset \Omega_2$  and  $|\Omega| = k$ , we have*

$$|\Omega \cup \Omega_{(k)}| \leq \min(2k, |\Omega_2 \cup \Omega_{(k)}|) = k_1.$$

The proof of the above is simple, we leave it for the readers. The next result is critical in constructing the new optimality tests.

**Lemma 4.3** Recall  $k_1 = \min(2k, |\Omega_2 \cup \Omega_{(k)}|)$ . Given the previously defined quantities  $\mu$  and  $\delta$ , we have

$$\|\Phi_{\Omega \cup \Omega_{(k)}} \delta\|_2^2 \geq (1 - (k_1 - 1)\mu) \|\delta\|_2^2.$$

**Proof.** First of all, we have

$$\begin{aligned} \|\Phi_{\Omega \cup \Omega_{(k)}} \delta\|_2^2 &= \delta^T \Phi_{\Omega \cup \Omega_{(k)}}^T \Phi_{\Omega \cup \Omega_{(k)}} \delta \\ &\geq \sum_i \delta_i^2 - \mu \sum_{a \neq b, a, b \in \Omega \cup \Omega_{(k)}} |\delta_a| \cdot |\delta_b|. \end{aligned}$$

Applying the Cauchy's inequality, readers can easily verify the following:

$$\sum_{\substack{a \neq b, \\ a, b \in \Omega \cup \Omega_{(k)}}} |\delta_a| \cdot |\delta_b| \leq (k_1 - 1) \|\delta\|_2^2.$$

Combining the above two, we have proved the inequality in the lemma.  $\square$

Combining Lemma 4.3 and inequality (4.5), we have

$$(1 - (k_1 - 1)\mu) \|\delta\|_2^2 \leq 2 \langle \Phi_{\Omega \cup \Omega_{(k)}}^T \varepsilon_{(k)}, \delta \rangle,$$

which is equivalent to

$$\left\| (1 - (k_1 - 1)\mu) \delta - \Phi_{\Omega \cup \Omega_{(k)}}^T \varepsilon_{(k)} \right\|_2^2 \leq \|\Phi_{\Omega \cup \Omega_{(k)}}^T \varepsilon_{(k)}\|_2^2.$$

The above implies the following,

$$(1 - (k_1 - 1)\mu) \|\delta\|_\infty \leq \|\Phi_{\Omega \cup \Omega_{(k)}}^T \varepsilon_{(k)}\|_\infty + \|\Phi_{\Omega \cup \Omega_{(k)}}^T \varepsilon_{(k)}\|_2, \quad (4.6)$$

where  $\|\cdot\|_\infty$  denotes the maximum absolute value in a vector. Recalling  $\Omega \subset \Omega_2$ , the following is evident:

$$\|\Phi_{\Omega \cup \Omega_{(k)}}^T \varepsilon_{(k)}\|_\infty \leq \|\Phi_{\Omega_2 \cup \Omega_{(k)}}^T \varepsilon_{(k)}\|_\infty. \quad (4.7)$$

It is easy to verify that

$$\|\Phi_{\Omega \cup \Omega_{(k)}}^T \varepsilon_{(k)}\|_2 \leq \|\Phi_{\Omega_2 \cup \Omega_{(k)}}^T \varepsilon_{(k)}\|_{(k_1)}. \quad (4.8)$$

Combining (4.7) and (4.8), we have

$$(1 - (k_1 - 1)\mu) \|\delta\|_\infty \leq \|\Phi_{\Omega_2 \cup \Omega_{(k)}}^T \varepsilon_{(k)}\|_\infty + \|\Phi_{\Omega_2 \cup \Omega_{(k)}}^T \varepsilon_{(k)}\|_{(k_1)}. \quad (4.9)$$

Given (4.9), we are ready to prove the theorem. For  $i \in \Omega_{(k)}$ , suppose (4.4) holds. From (4.9), we have

$$|(\widehat{\beta}_{(k)})_i - (\widehat{\beta}(\Omega))_i| \leq \frac{\|\Phi_{\Omega_2 \cup \Omega_{(k)}}^T \varepsilon_{(k)}\|_\infty + \|\Phi_{\Omega_2 \cup \Omega_{(k)}}^T \varepsilon_{(k)}\|_{(k_1)}}{1 - (k_1 - 1)\mu}.$$

The above and (4.4) lead to  $|(\widehat{\beta}(\Omega))_i| > 0$ , which implies that  $i \in \Omega$ . Hence, we have  $\Theta(k) \subset \Omega$ . The theorem is proven.

## 5 Algorithm

In this section, the implementation strategies are described. The scanning described in Section 5.1 is equivalent to the method in the original LB (Furnival and Wilson, 1974). We believe our new description is more accessible. The integration of new optimality tests is trivial. Hence, it is only briefly described in Section 5.2.

### 5.1 A Scheme to Scan Through the Pair Tree $PT(m)$

We design an algorithm that reaches each node of  $PT(m)$  once and only once. We will use the following notations. For an arbitrary set  $\Omega$  with ordered elements, for integer  $j \geq 1$ ,  $r(\Omega, j)$  denotes a subset of  $\Omega$  by removing the  $j$ th last element of  $\Omega$ . For example, we have

$$r(\{12345\}, 1) = \{1234\}, \text{ and } r(\{12345\}, 5) = \{2345\}.$$

Based on the above, define

$$r^k(\Omega, j) = \underbrace{r(r(\cdots r}_{k \text{ times}}(\Omega, j) \cdots, j), j).$$

For example, we have  $r^2(\{12345\}, 1) = \{123\}$  and  $r^3(\{12345\}, 1) = \{12\}$ .

Given the structure of  $PT(m)$ , readers can verify that the following scheme reaches every node in  $PT(m)$  once and only once.

1. A *node list* is empty initially. Starting from the root node  $(\emptyset, \{1, 2, \dots, m\})$ , the following array,

$$\begin{array}{lll} r(\{1, 2, \dots, m\}, 1), & r(\{1, 2, \dots, m\}, 2), & 1, \\ r^2(\{1, 2, \dots, m\}, 1), & r(\{1, 2, \dots, m\}, 3), & 2, \\ r^3(\{1, 2, \dots, m\}, 1), & r(\{1, 2, \dots, m\}, 4), & 3, \\ \vdots & \vdots & \vdots \\ r^{m-1}(\{1, 2, \dots, m\}, 1), & r(\{1, 2, \dots, m\}, m), & m-1, \end{array}$$

is inserted into the node list. Note each row of the node list is made by two subsets and its order among the siblings.

2. Suppose the node list is not empty and the top row is  $(\Omega_1, \Omega_2, s)$ , where  $\Omega_1$  and  $\Omega_2$  are the subsets of  $\{1, \dots, m\}$ , and integer  $s \geq 1$ .
  - If  $s = 1$ , this row is removed from the node list (has been scanned).

- If  $s > 1$ , add the following array at the bottom of the node list:

$$\begin{array}{ccc} r(\Omega_2, 1), & r(\Omega_2, 2), & 1, \\ r^2(\Omega_2, 1), & r(\Omega_2, 3), & 2, \\ \vdots & \vdots & \vdots \\ r^{s-1}(\Omega_2, 1), & r(\Omega_2, s), & s - 1. \end{array}$$

Then, remove the top row  $(\Omega_1, \Omega_2, s)$  from the node list.

3. Step 2 is repeated until the node list is empty again.

## 5.2 Integrating the Optimality Tests

In the scheme that was described in the last subsection, it is straightforward to integrate the optimality test. We maintain the *optimality tests list*, whose rows are made by

$$\underline{\text{RSS}}(k), \Omega_{(k)}, \hat{\beta}_{(k)}, \varepsilon_{(k)}^T \Phi,$$

where  $0 \leq k \leq m$ . In the step 2 in the last subsection, if the pair of subsets in a row fail at least one optimality test (which includes both the original test in LB and the newly proposed test in Theorem 4.1), then this row is not inserted into the node list.

Note the original LB only uses the information in the first column of the optimality tests list, while the newly proposed tests (LBOT) use additional information.

# 6 Simulations

## 6.1 Synthetic Data

### 6.1.1 An Illustrative Example

In order to illustrate the efficiency of our method, we create a table – Table 1 – that includes all the pairs from  $PT(5)$ . The pairs that are used by the original LB are marked with ‘\*’, while those that are required by our enhanced leaps-and-bounds algorithm are marked with ‘Δ’. The underlying regression model is:

$$y = 167.5058 + 27.0171x_1 + 5.2054x_3 + 135.8065x_4 + -0.0431x_5 + \varepsilon,$$

which is generated by random. It is observed that for the case illustrated in Table 1, the enhanced leaps-and-bounds reduces the number of examined pairs nodes from 13 (which is for the original LB) to 9. Table 2 presents the minimum residual sums of squares for different subset size  $k$ , as well as the optimal  $k$ -subsets.

LBOT	LB	$\Omega_1$	RSS	$\Omega_2$	RSS
$\Delta$	*	$\emptyset$	20030.67	12345	1077.78
$\Delta$	*	1234	1077.81	1235	19667.34
$\Delta$	*	123	19667.86	1245	1104.57
$\Delta$	*	12	19691.99	1345	1077.80
$\Delta$	*	1	19733.12	2345	1731.38
	*	124	1104.59	125	19691.40
	*	134	1077.83	135	19708.79
$\Delta$	*	13	19709.37	145	1104.61
		234	1731.45	235	19964.93
$\Delta$	*	23	19965.92	245	1759.68
$\Delta$	*	2	19991.05	345	1731.45
	*	14	1104.62	15	19732.47
		24	1759.77	25	19989.98
		34	1731.51	35	20004.86
$\Delta$	*	3	20005.92	45	1759.76
	*	4	1759.85	5	20029.52

Table 1: Pair tree and residual sums of squares.

$k$	RSS	Optimal Subset
0	20030.67	$\emptyset$
1	1759.85	4
2	1104.62	14
3	1077.83	134
4	1077.80	1345
5	1077.78	12345

Table 2: Minimum residual sums of squares and corresponding optimal  $k$ -subsets.

### 6.1.2 LB versus LBOT in Random Experiments

To further compare LBOT with LB, some random experiments are performed. Recall the regression model,  $y = \Phi x + \varepsilon$ , where model matrix  $\Phi \in \mathbb{R}^{n \times (m+1)}$ . In the following experiments, we set  $n = 1000$  and  $m = 10$ . Each column  $\phi_i, i = 1, 2, \dots, m$ , is first generated from multivariate normal  $N(\vec{0}_{n \times 1}, I_n)$ , then followed by standardization. Coefficients are generated as  $x_i \sim N(0, \sigma^2), i = 0, 1, \dots, m$ , where  $\sigma = 100$ . Set  $\varepsilon_i \sim N(0, 1)$ , for  $i = 1, 2, \dots, n$ . We present the dot-plots of quantities –

(number of pairs used in LBOT, number of pairs used in LB)

– in Figure 3. Totally 1000 random simulations are performed. The dashed line is the diagonal. We can see that all points are below the diagonal: LBOT always requires less pairs to be examined. To illustrate the reduction of the number of pairs of subsets that are required to be examined, in Figure 4, we present the histograms of the ratios – the number of pairs used in LBOT over the number of pairs used in leaps and bounds. On average, LBOT requires 87.05% of the subsets that are required by LB.

In Table 3, for a range of the values of  $m$ , following the same random simulations described in the last paragraph, the average number of pairs that are examined based on 10 random experiments are reported. Again, we see a reduction in the number of required pairs. It is interesting to observe that the percentage of pairs that are examined in a pair tree reduces as the number of covariates ( $m$ ) increases; see the last column of Table 3.

$m$	# Pairs	LB	LBOT	LBOT/#Pairs
10	512	148.7	131.3	0.1282
12	2048	598.9	550.6	0.1344
14	8192	1714.3	1643.3	0.1003
16	32768	6226.2	6112.2	0.0933
18	262144	21034.0	20973.7	0.0800
20	1048576	42654.8	42278.6	0.0403

Table 3: For different values of  $m$ , the average numbers of pairs that are examined by both LB and LBOT among ten random experiments are presented. The second column includes the total number of pairs in the complete pair trees.

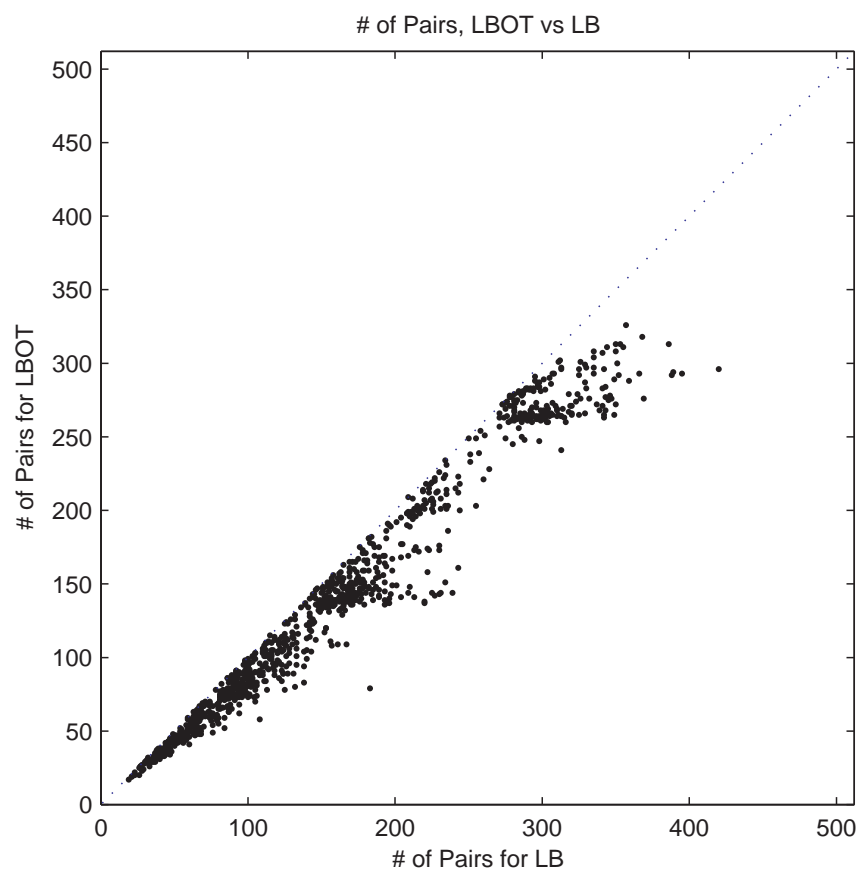


Figure 3: The number of pairs in LBOT versus the number of pairs used in LB. The number of covariates  $m = 10$ .

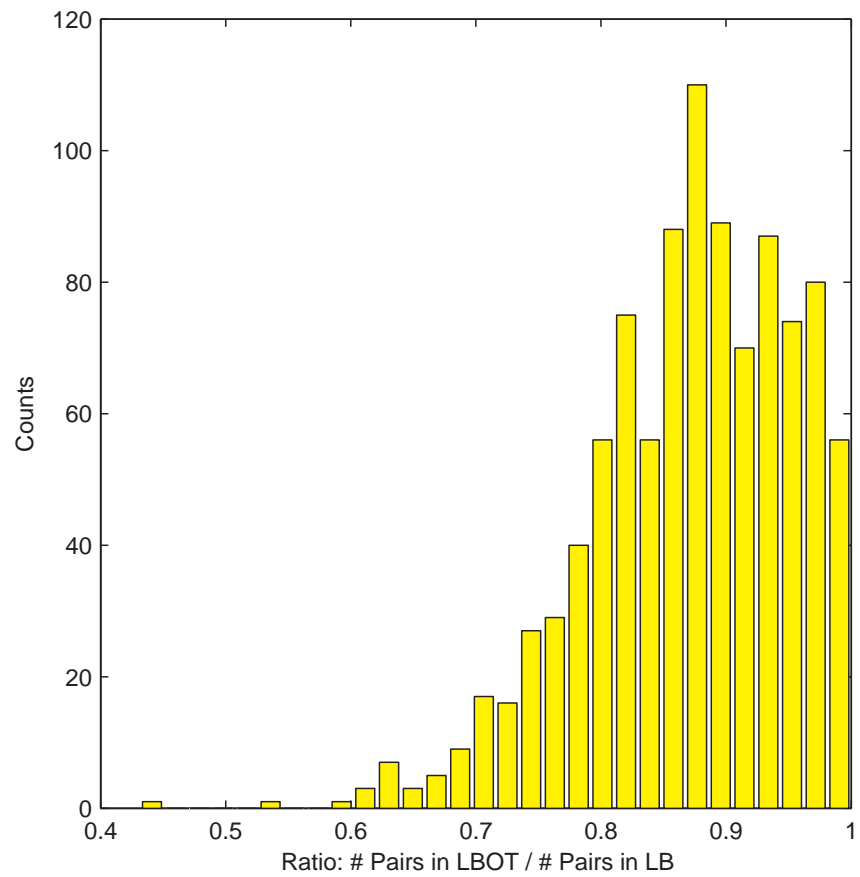


Figure 4: Histogram: when  $m = 10$ , the number of pairs in LBOT over the number of pairs used in LB.

## 6.2 Effect of Model

For simplicity, the experiment that leads to Figures 3 and 4 is denoted as Exp. A. Note in this experiment, the coefficients are sampled from  $N(0, 100^2)$ ; i.e., the absolute values of the coefficients are likely to be large. To see when LBOT can significantly reduce the number of pairs from LB, we repeat the Exp. A, but change the distribution from  $N(0, 100^2)$  to  $N(0, 1)$ . The new experiment is denoted as Exp. B. Figure 5 presents the histograms of the numbers of pairs used by LB, together with the ratios of the pairs between the two methods. Note Figure 5 (b) repeats Figure 4. It is redrawn and scaled for comparison. No dramatic difference can be seen from Figure 5 (a) and (c): the number of pairs that are required by LB does not change much. However, being compared with Figure 5 (b), histogram (d) is significantly skewed to the right: the additional optimality tests are less likely to reduce the computation in Exp. B, in which the coefficients of the underlying models are likely to be close to zero.

In summary, the additional optimality tests are likely to improve LB when the underlying true model has relatively large absolute values of coefficients (relative to the noise level).

## 6.3 Heuristic: Pre-Sorting

The pair tree is scanned top-down, left-to-the-right. If the optimal subsets appear earlier in the scanning scheme, LB and LBOT will have a better chance to reduce the computation. Based on this observation, we can reorder the covariates before the evocation of LB or LBOT. We carry out a random experiment (denoted as Exp. C), which is identical with Exp. A, except a pre-sorting of the covariates that imposes the following condition:

$$\langle y, \phi_1 \rangle \geq \langle y, \phi_2 \rangle \geq \dots \geq \langle y, \phi_m \rangle.$$

The histograms of the number of pairs that are used by LB in Exp. A and C are presented in Figure 6 (a) and (c), respectively. After the pre-sorting, we see a significant reduction in the numbers of pairs that are required. Figure 6 (b) is another rescaled version of Figure 4. It will be compared with subfigure (d). Based on Figure 6 (d), LBOT still reduces the number of pairs in a significant proportion of cases.

In summary, some preprocessing can help to improve the efficient of both LB and LBOT.

## 6.4 Real Data

Being compared with LB, LBOT always reduces the number of subsets that are required to be examined. The actual amount of reduction depends on the data, as one can observe in the

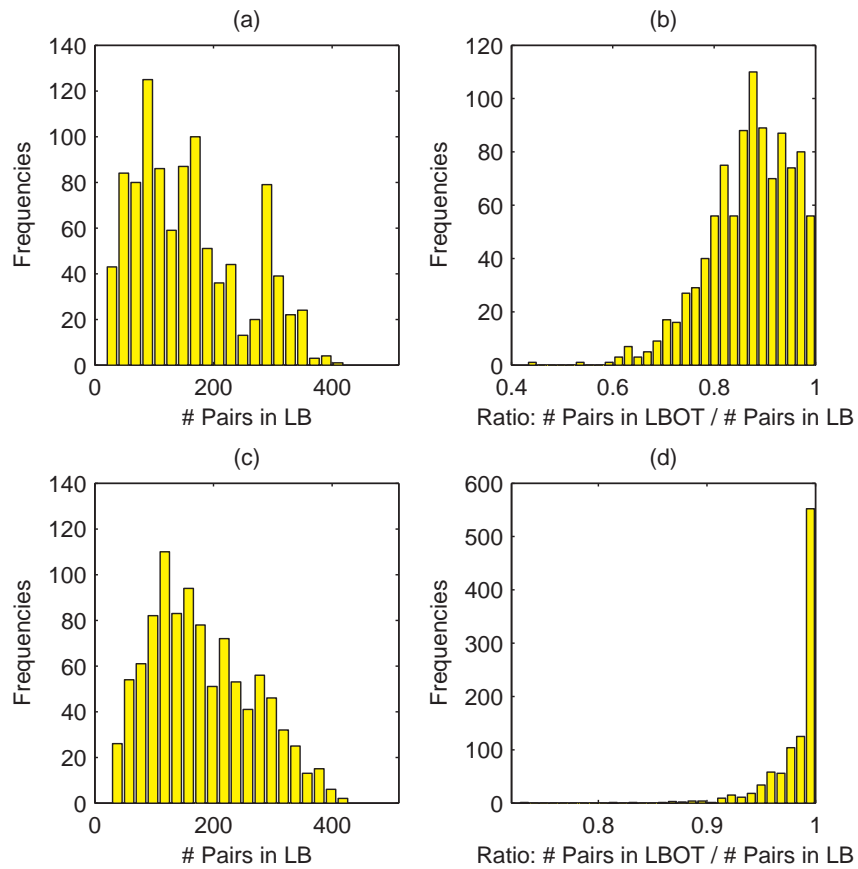


Figure 5: Histograms: (a) and (c), the number of pairs in LB; (b) and (d), ratios between the numbers of pairs in LB and the numbers of pairs in LBOT. (a) and (b) (resp., (c) and (d)) are for Exp. A (resp., Exp. B). See the context for details.

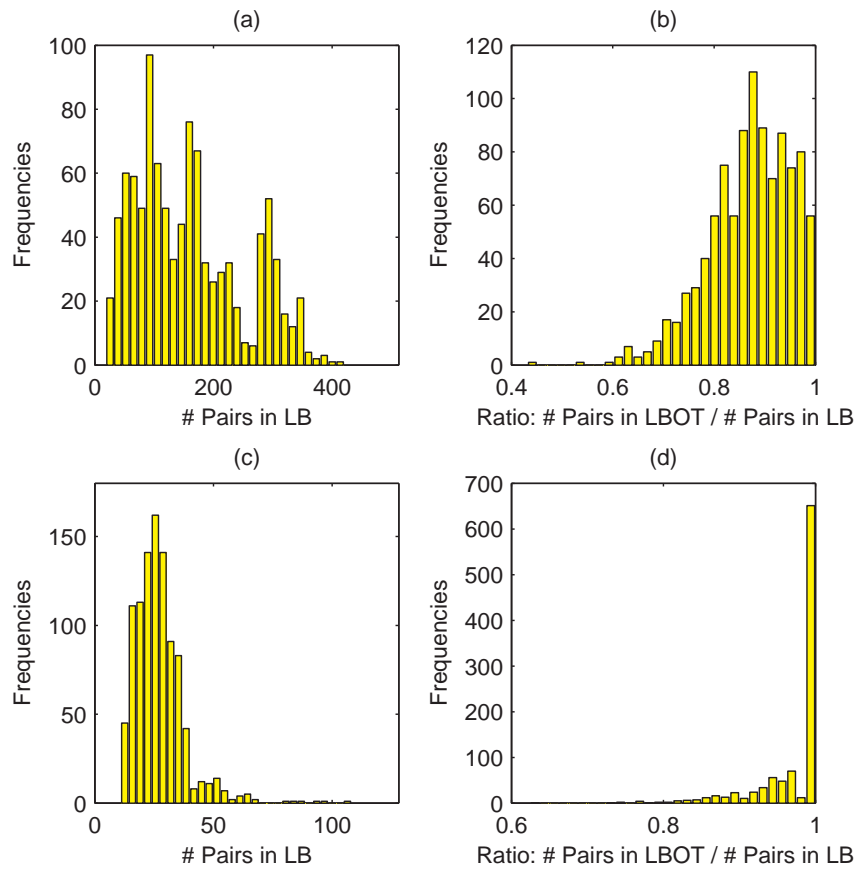


Figure 6: Histograms: (a) and (c), the number of pairs in LB; (b) and (d), ratios between the numbers of pairs in LB and the numbers of pairs in LBOT. Subfigures (a) and (b) (resp., (c) and (d)) are for Exp. A (resp., Exp. C). See the context for details.

previous experiments. For example, the reduction is more evident in Exp. A than in Exp. B. How much will LBOT reduce the number of subsets in real data? We experimented with two datasets: diabetes data (<http://www-stat.stanford.edu/~hastie/Papers/LARS/>) and housing data (<http://www.ics.uci.edu/~mlearn/databases/housing/>). They are chosen because they have been widely used in the regression literature. The diabetes data has 10 covariates and has been used in Efron et al. (2004) to illustrate a stepwise algorithm (LARS). The housing data has 13 covariates. In both cases, LBOT fails to reduce the number of subsets. In other words, it is equivalent to the case when the ratio is equal to 1 in Exp. A, B, or C. Note in the random experiments, comparing to the histograms in Figure 4, 5 (d), and 6 (d), the probability of having a ratio “1” is small. To our surprise, the ratio is “1” for both ‘real’ data sets that we experimented. As a future research topic, it will be interesting to derive reasonable condition(s), under which the additional optimality tests can *not* reduce the number of required pairs in LB (or LBOT).

For the diabetes data, Table 4 gives the optimal subsets. The covariates are standardized. No pre-sorting is adopted. Note that some covariates come and leave the optimal subsets, e.g., covariates 5 and 7, as the subset size increases. Such a phenomenon can not be caught by a pure forward or backward subset selection.

Subset Sizes $k$	RSS	Optimal Subsets									
0	2621009.12										
1	1719581.81	3									
2	1416694.01	3						9			
3	1362708.69	3	4					9			
4	1331431.40	3	4	5				9			
5	1287881.16	2	3	4			7	9			
6	1271494.00	2	3	4	5	6		9			
7	1267807.81	2	3	4	5	6		8	9		
8	1264714.58	2	3	4	5	6		8	9	10	
9	1264068.10	2	3	4	5	6	7	8	9	10	
10	1263985.79	1	2	3	4	5	6	7	8	9	10

Table 4: The optimal subsets and corresponding residual sums of squares, for the diabetes data.

For the housing data, the same table is produced in Table 5. Again, we see some covariates (e.g., 2 and 4) coming and leaving from the optimal subsets.

Subset Sizes $k$	RSS	Optimal Subsets										
0	42716.30											
1	19472.38	13										
2	15439.31	6 13										
3	13727.99	6 11 13										
4	13228.91	6 8 11 13										
5	12469.34	5 6 8 11 13										
6	12141.07	4 5 6 8 11 13										
7	11868.24	4 5 6 8 11 12 13										
8	11678.30	2	4 5 6 8 11 12 13									
9	11526.12	1	4 5 6 8 9 11 12 13									
10	11308.58	1 2	5 6 8 9 10 11 12 13									
11	11081.36	1 2	4 5 6 8 9 10 11 12 13									
12	11078.85	1 2 3	4 5 6 8 9 10 11 12 13									
13	11078.78	1 2 3	4 5 6 7 8 9 10 11 12 13									

Table 5: The optimal subsets and corresponding residual sums of squares for the Housing data.

## 7 Discussion

LB is a branch-and-bound (B&B) approach designed specifically for regression problems. B&B has been applied to many other problems, e.g., *feature subset selection* (FSS). Typical references are Narendra and Fukunaga (1977), Kohavi and John (1997), and Jain and Zongker (1997). The objective in an FSS problem is different from the objective in a regression problem. FSS mostly involves with a classification problem, instead of a regression problem. In FSS, researchers have proposed enhanced B&B by adapting various heuristics: Yu and Yuan (1993), Chen (2003), Somol et al. (2004), and Cao and Saha (2005). Similarly, leaps-and-bounds has been applied to variable selection in discriminant analysis (Silva, 2001). A careful comparison will review that the strategy of deriving the additional optimality test in LBOT is very distinct from the above works in FSS. On the other hand, it will be very interesting to explore the heuristics that have been developed in accelerating the B&B algorithms in FSS. They could further improve the performance of LBOT. Some serious structural works are required.

## 8 Conclusion

New optimality conditions are derived in the framework of leaps-and-bounds algorithm. The new tests guarantee to reduce the number of subsets that are required in a branch-and-bound exhaustive subset search. The reduction of computation are testified in random experiments. We improved a state-of-the-art method in comprehensive subset selections. The ideas behind the newly introduced tests are novel. The analysis technique that is used in deriving the new condition could be insightful in studying other regression-related problems.

## References

- Cao, Y. and P. Saha (2005, March). Improved branch and bound method for control structure screening. *Chemical Engineering Science* 60(6), 1555–1564.
- Chen, X. (2003, August). An improved branch and bound algorithm for feature selection. *Pattern Recognition Letters* 24(12), 1925–1933.
- Efron, B., T. Hastie, I. Johnstone, and R. Tibshirani (2004). Least angle regression. *Ann. Statist.* 32(2), 407–499.
- Furnival, G. and R. Wilson (1974). Regression by leaps and bounds. *Technometrics* 16(4), 499–511.
- George, E. I. (2000). The variable selection problem. *J. Amer. Statist. Assoc.* 95(452), 1304–1308.
- Huo, X. and X. S. Ni (2005, July). When do stepwise algorithms meet subset selection criteria? ISyE Statistics Technical Report, URL = <http://www.isye.gatech.edu/statistics/papers/>.
- Jain, A. and D. Zongker (1997, February). Feature selection: evaluation, application, and small sample performance. *IEEE Trans. On Pattern Analysis and Machine Intelligence* 19(2), 153–158.
- Kadane, J. B. and N. A. Lazar (2004, March). Methods and criteria for model selection. *Journal of The American Statistical Association* 99(465), 279–290.
- Kohavi, R. and G. H. John (1997, December). Wrappers for feature subset selection. *Artificial Intelligence* 97(1-2), 273–324.
- Narendra, P. M. and K. Fukunaga (1977, September). A branch and bound algorithm for feature subset selection. *IEEE Trans. Comput.* 26(9), 917–922.

- Ni, X. S. and X. Huo (2005, August). Enhanced leaps-and-bounds methods in subset selection with additional optimality tests. Submitted conference paper, available under request from authors.
- Silva, A. P. D. (2001, January). Efficient variable screening for multivariate analysis. *Journal of Multivariate Analysis* 76(1), 35–62.
- Somol, P., P. Pudil, and J. Kittler (2004, July). Fast branch & bound algorithms for optimal feature selection. *IEEE Trans. On Pattern Analysis and Machine Intelligence* 26(7), 900–912.
- Yu, B. and B. Yuan (1993, June). A more efficient branch and bound algorithm for feature selection. *Pattern Recognition* 26(6), 883–889.