J Glob Optim (2009) 44:273–282
DOI 10.1007/s10898-008-9323-9

Choosing the best set of variables in regression analysis using integer programming

Hiroshi Konno · Rei Yamamoto

Received: 1 March 2007 / Accepted: 15 June 2008 / Published online: 8 July 2008

Abstract This paper is concerned with an algorithm for selecting the best set of s variables out of $k(> s)$ candidate variables in a multiple linear regression model. We employ absolute deviation as the measure of deviation and solve the resulting optimization problem by using 0-1 integer programming methodologies. In addition, we will propose a heuristic algorithm to obtain a close to optimal set of variables in terms of squared deviation. Computational results show that this method is practical and reliable for determining the best set of variables.

Keywords Linear regression · Least absolute deviation · Variable selection · Cardinality constraint · 0-1 integer programming

1 Introduction

Variable selection is of primary importance in regression analysis [3,9]. Let Y be a random variable to be explained by a set of k candidate variables $X_i, i = 1, 2, \ldots, k$ using the linear expression:

$$Y = \alpha_0 + \alpha_1 X_1 + \alpha_2 X_2 + \cdots + \alpha_k X_k + \varepsilon, \tag{1}$$

where $\alpha_0, \alpha_1, \ldots, \alpha_k$ are parameters to be estimated and ε is a residual random variable.

If there exist many candidate variables, we have to choose a certain portion of these variables which achieve the required quality of fitting. AIC [1] is one of the very popular methods for this purpose. To apply this criterion, however, we need to have prior information about the statistical properties of residual variables. Also, it need not always lead to the best set of variables. Many other criteria have been proposed in the past [3], but they all inherit advantages and disadvantages of AIC.

H. Konno · R. Yamamoto (✉)
Department of Industrial and Systems Engineering, Chuo University, 2-13-27, Kasuga, Bunkyo-ku, Tokyo 112-8551, Japan
e-mail: rei@kc.chuo-u.ac.jp

R. Yamamoto
Mitsubishi UFJ Trust Investment Technology Institute Co., Ltd, 2-5-6, Shiba, Minato-ku, Tokyo 105-0014, Japan

In this paper, we will consider those problems for which AIC is not applicable due to the lack of information about the statistical structure of the residual variable. Such problems often appear in finance, bio-informatics and elsewhere.

The problem to be considered in this paper is:

> Given a positive integer s, find the set of variables $x_{i_1}, x_{i_2}, \ldots, x_{i_s}$ such that the total amount of residual error is minimal.

This is a difficult combinatorial optimization problem for which no exact and efficient algorithm has been proposed in the past. To find an optimal combination, we need to use an enumeration approach. The number of possible explanatory variables k in the traditional field is not very large, say less than 20 or 30. However, much larger problems are under consideration. For example, k is over 100 in failure discriminant analysis [7,8] and it is sometimes larger than 1,000 in bio-informatics [11]. When $k = 100$ and $s = 20$ as in the case of failure discriminant analysis [8], the number of possible combinations is ${}_{100}C_{20} \sim 10^{21}$, so that total enumeration is completely out of reach. Therefore, people use some sort of heuristic approach [6,9,10].

One commonly used method is to sequentially introduce s "important" variables one at a time. When the residual error is small enough, then we are done. Otherwise we eliminate a certain variable from the model and add a new one in its place and continue until the fitting is satisfactory enough. This procedure usually leads to a good solution, but it may not generate the best combination. The purpose of this paper is to propose an exact and efficient method to solve the problem stated above.

In Sect. 2, we first formulate this problem as a quadratic mixed 0-1 integer programming problem [13]. However, there exists no efficient algorithm for solving this problem to date. Therefore we will propose an alternative representation of the problem by replacing squared deviation by absolute deviation as a measure of variation. Least absolute deviation estimation [2] is less popular among practitioners, but it has some nice properties. Also, the use of absolute deviation leads to a 0-1 mixed linear integer programming problem which can be solved by the state-of-the-art mathematical programming methodology [5,13].

Further, we will propose a two-step method for calculating an optimal solution of the quadratic mixed 0-1 integer programming formulation. Though this procedure need not always generate an optimal solution, it usually generates very good solutions as demonstrated by a series of numerical experiments to be presented in Sect. 3. Finally, in Sect. 4 we will discuss the future direction of research.

2 Least absolute deviation fitting problem and a two-step algorithm

Given T sets of data $(y_t, x_{1t}, x_{2t}, \ldots, x_{kt})$, $t = 1, 2, \ldots, T$, let us define

$$f(\alpha_0, \alpha_1, \ldots, \alpha_k) = \sum_{t=1}^{T} \left\{ y_t - \left(\alpha_0 + \sum_{j=1}^{k} \alpha_j x_{jt} \right) \right\}^2. \tag{2}$$

Then the problem posed in the Introduction can be formulated as the following constrained minimization problem:

$$\begin{array}{l} \text{minimize} \quad f(\alpha_0, \alpha_1, \ldots, \alpha_k) \\ \text{subject to:} \quad \text{at most } s \text{ components of } (\alpha_1, \alpha_2, \ldots, \alpha_k) \text{ are non-zero.} \end{array} \tag{3}$$

By introducing zero-one integer variables z_j, $j = 1, 2, \ldots, k$, this problem can be formulated as a quadratic 0-1 integer programming problem:

$$P_k(s)\left|\begin{array}{l}\text{minimize } f(\alpha_0, \alpha_1, \ldots, \alpha_k)\\ \text{subject to } \displaystyle\sum_{j=1}^{k} z_j = s\\ \quad \underline{\alpha}_j z_j \le \alpha_j \le \overline{\alpha}_j z_j,\ j = 1, 2, \ldots, k\\ \quad z_j \in \{0,\ 1\},\ j = 1, 2, \ldots, k,\end{array}\right. \tag{4}$$

where $\overline{\alpha}_j$ and $\underline{\alpha}_j$ are respectively, the largest and smallest possible value of α_j. If $z_j = 0$ in (4) then $\alpha_j = 0$, so that at most s components of α_j can be non-zero, as required. Algorithmic research for solving quadratic 0-1 integer programming problems is now under way [13]. However, there exists no efficient and exact algorithm to date.

The key idea of this paper is to replace squared deviation $f(\alpha_0, \alpha_1, \ldots, \alpha_k)$ by absolute deviation:

$$g(\alpha_0, \alpha_1, \ldots, \alpha_k) = \sum_{t=1}^{T} \left| y_t - \left(\alpha_0 + \sum_{j=1}^{k} \alpha_j x_{jt} \right) \right|, \tag{5}$$

and consider the following problem:

$$Q_k(s)\left|\begin{array}{l}\text{minimize } g(\alpha_0, \alpha_1, \ldots, \alpha_k)\\ \text{subject to } \displaystyle\sum_{j=1}^{k} z_j = s\\ \quad \underline{\alpha}_j z_j \le \alpha_j \le \overline{\alpha}_j z_j,\ j = 1, 2, \ldots, k\\ \quad z_j \in \{0,\ 1\},\ j = 1, 2, \ldots, k.\end{array}\right. \tag{6}$$

It is well known that the least absolute deviation estimator is more robust than the least square estimator [2]. Also, optimal solutions of $P_k(s)$ and $Q_k(s)$ are similar in many situations, since both problems minimize measures of residual variation. For detailed discussion about least absolute deviation estimation, readers are referred to [2].

It is well known that $Q_k(s)$ can be rewritten as a 0-1 linear integer programming problem below:

$$\left|\begin{array}{l}\text{minimize } \displaystyle\sum_{t=1}^{T}(u_t + v_t)\\ \text{subject to } u_t - v_t = y_t - \left(\alpha_0 + \displaystyle\sum_{j=1}^{k} \alpha_j x_{jt} \right),\ t = 1, 2, \ldots, T\\ \quad u_t \ge 0,\ v_t \ge 0,\ t = 1, 2, \ldots, T\\ \quad \displaystyle\sum_{j=1}^{k} z_j = s\\ \quad \underline{\alpha}_j z_j \le \alpha_j \le \overline{\alpha}_j z_j,\ j = 1, 2, \ldots, k\\ \quad z_j \in \{0,\ 1\},\ j = 1, 2, \ldots, k.\end{array}\right. \tag{7}$$

Problem (7) has an optimal solution since it is feasible and the objective function is bounded below.

Theorem 1 Let $(\alpha_0^*, \alpha_1^*, \ldots, \alpha_k^*, u_1^*, u_2^*, \ldots, u_T^*, v_1^*, v_2^*, \ldots, v_T^*, z_1^*, z_2^*, \ldots, z_k^*)$ be an optimal solution of (7). Then $(\alpha_0^*, \alpha_1^*, \ldots, \alpha_k^*, z_1^*, z_2^*, \ldots, z_k^*)$ is an optimal solution of (6).

Proof See Chapter 14 of Ref. [4]. □

When k, s, T are not very large, problem (7) can be solved by the state-of-the-art software such as CPLEX10.1 [5].

Let us now propose a two-step algorithm for solving $P_k(s)$. The first step is to solve the least absolute deviation estimation problem $Q_k(s+r)$:

$$Q_k(s+r)\left|\begin{array}{l}\text{minimize } g(\alpha_0,\alpha_1,\ldots,\alpha_k)\\ \text{subject to } \displaystyle\sum_{j=1}^{k} z_j = s+r\\ \qquad \underline{\alpha}_j z_j \le \alpha_j \le \overline{\alpha}_j z_j,\ j=1,2,\ldots,k\\ \qquad z_j \in \{0,\ 1\},\ j=1,2,\ldots,k,\end{array}\right. \tag{8}$$

where $r \le k-s$ is some positive integer. Let $(\alpha_0^*, \alpha_1^*, \ldots, \alpha_k^*)$ be an optimal solution of (8), and assume without loss of generality that $z_j^* = 0$ for $j > s+r$.

Let $(\hat{\alpha_0}, \hat{\alpha_1}, \ldots, \hat{\alpha_k})$ be an optimal solution of $P_k(s)$. Then it is likely that $\hat{\alpha_j} = 0$ for almost all $j > s+r$ for large enough r since absolute deviation and squared deviation are similar measures of variation.

To attempt to recover an optimal solution of $P_k(s)$, we solve

$$P_{s+r}(s)\left|\begin{array}{l}\text{minimize } f(\alpha_0,\alpha_1,\ldots,\alpha_k)\\ \text{subject to } \displaystyle\sum_{j=1}^{s+r} z_j = s\\ \qquad \underline{\alpha}_j z_j \le \alpha_j \le \overline{\alpha}_j z_j,\ j=1,2,\ldots,k\\ \qquad z_j \in \{0,\ 1\},\ j=1,2,\ldots,s+r.\end{array}\right. \tag{9}$$

If r is not large, this problem can be solved by solving ${}_{s+r}C_s$ least square subproblems associated with all possible s out of $s+r$ combinations.

3 Result of computational experiments

3.1 Test for randomly generated data

We conducted numerical experiments using several types of randomly generated data sets. For explanatory variables X_j, we assumed that all variables are *i.i.d.*. For Y variables we generated five sets of data.

Data Set 1 (DS1) $Y \sim N(0,1)$ and independent of each $X_j \sim N(0,1)$.
Data Set 2 (DS2) $Y \sim N(2,1)$ correlated with each $X_j \sim N(2,1)$ with correlation coefficient 0.1.[1]
Data Set 3 (DS3) $Y \sim N(2,1)$ correlated with each $X_j \sim N(2,1)$ with correlation coefficient 0.3.
Data Set 4 (DS4) Y is an inverse logit transformation of the random variable $\hat{Y} \sim N(0,1)$, independent of each $X_j \sim N(0,1)$.
Data Set 5 (DS5) $Y = 0$ or 1 with equal probability, independent of each $X_j \sim N(0,1)$.

[1] Let $U \sim N(0,1)$, $V \sim N(0,1)$. We generated Y, X with correlation coefficient ρ as $X = U+2$, $Y = \rho U + \sqrt{1-\rho^2}V + 2$.

We solved the problem (7) by CPLEX10.1 [5] on a personal computer Xeon (3.73 GHz). Table 1 shows the number of possible combinations ${}_kC_s$. We see from this that total enumeration is impractical for large scale problems.

First, let us explain the method for choosing lower and upper bounds $\underline{\alpha}_j$ and $\overline{\alpha}_j$ on variables α_j, $j = 1, 2, \ldots, k$. If we choose the interval $[\underline{\alpha}_j, \overline{\alpha}_j]$ large enough, then we would not miss an optimal solution. We conducted preliminary experiments using a variety of test data to find that $(\underline{\alpha}_j, \overline{\alpha}_j) = (-10, 10)$ is more than enough for all data belonging to Data Sets 1–5.

However, this scheme results in large computation time as presented in Table 2. The number p of the 3rd column denotes the number of problems solved within 1,500 CPU seconds out of 5 test problems. On the other hand, problems with $(\underline{\alpha}_j, \overline{\alpha}_j) = (0, 10)$ can be solved an order of magnitude faster.

This observation leads us to the following scheme:

$$(\underline{\alpha}_j, \overline{\alpha}_j) = \begin{cases} (0, 10), & \text{if } \operatorname{cov}(X_j, Y) > 0, \\ (-10, 0), & \text{otherwise.} \end{cases} \tag{10}$$

by noting the fact that α_j associated with X_j having positive (negative) correlation with Y is expected to have non-negative (non-positive) optimal value.

Table 1 Values of $\log_{10k} C_s$

s	k			
	20	50	100	200
5	4.2	6.3	7.9	9.4
10	5.3	10.0	13.2	16.4
20	0.0	13.7	20.7	27.2
30		13.7	25.5	35.6
40		10.0	28.1	42.3
50		0.0	29.0	47.7

Table 2 CPU time for solving problems with different choices of $(\underline{\alpha}_j, \overline{\alpha}_j)$

k	s	p	CPU (s)
(a) $(\underline{\alpha}_j, \overline{\alpha}_j) = (-10, 10)$			
20	5	5	1.08
	10	5	1.09
50	5	5	174.43
	10	2	1011.21
	20	0	–
(b) $(\underline{\alpha}_j, \overline{\alpha}_j) = (0, 10)$			
20	5	5	0.29
	10	5	0.13
50	5	5	4.91
	10	5	3.66
	20	5	0.85

Table 3 CPU time for solving $Q_k(s)$ ($T = 200$)

k	s	DS1	DS2	DS3	DS4	DS5
20	5	0.35	0.80	9.43	0.26	0.19
	10	0.20	0.55	8.56	0.18	0.13
50	5	6.73	325.16	–	5.00	9.68
	10	14.60	–	–	3.70	23.36
	20	1.41	–	–	0.84	3.44
100	5	299.05	–	–	306.61	541.30

Table 4 Average gap between upper bound and lower bound (%) ($k = 100$, DS4)

s	T		
	200	500	1,000
5	0.00	0.43	1.16
10	3.90	1.34	0.98
20	3.11	1.09	0.77
30	0.74	0.17	0.23

We compared this scheme with an alternative scheme $(\underline{\alpha}_j, \overline{\alpha}_j) = (-10, 10)$ to find that the two schemes result in the same solutions with a very few exceptions.

Table 3 shows the CPU time for several combinations of (k, s) and 5 different types of data sets. The number in the table shows the average CPU seconds of 5 test problems. Blanks in this table imply that the corresponding problems could not be solved to optimality within 1,500 CPU seconds. We see that problems are more difficult when Y and X_j's are correlated.

Table 4 shows the quality of incumbent solutions (the best solution obtained after the elapse of 1,500 CPU seconds) for problems which were not solved to optimality within 1,500 CPU seconds. We see that very good solutions have been generated.

In Table 5, we look into the details of incumbent solutions, where relative error implies the relative difference between the incumbent objective value and optimal value. We see that a reasonably good solution has been generated within 200 CPU seconds though more than 800 s are required to generate an optimal solution.

Figure 1 shows the magnitude of $\bar{R}^2$, the (degrees of freedom) adjusted R^2, a well used measure of fit of solutions obtained by using the two-step algorithm and S-plus [10,12] where

$$\bar{R}^2 = 1 - \frac{T-1}{T-s-1}\left\{1 - \frac{\sum_{t=1}^{T}(\hat{y}_t - \bar{\hat{y}})^2}{\sum_{t=1}^{T}(y_t - \bar{y})^2}\right\}, \tag{11}$$

$$\hat{y}_t = \alpha_0^* + \sum_{j=1}^{k} \alpha_j^* x_{jt}, \tag{12}$$

$$\bar{\hat{y}} = \sum_{t=1}^{T} \hat{y}_t / T. \tag{13}$$

Table 5 Quality of the incumbent solution ($k = 100$, $s = 10$, $T = 200$, DS4)

CPU (s)	Objective value	Relative error (%)	Gap (%)	$\bar{R}^2$
5	31.36	0.98	8.21	0.096
10	31.22	0.52	7.28	0.116
50	31.11	0.17	4.20	0.117
100	31.11	0.17	3.53	0.117
200	31.06	0.00	2.47	0.117
500	31.06	0.00	1.31	0.117
832	31.06	0.00	0.01	0.117

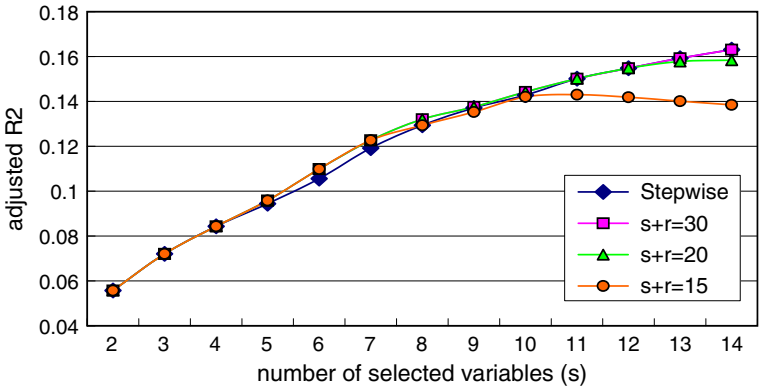

Fig. 1 Comparison of $\bar{R}^2$ ($T = 200$, $k = 50$, DS1)

Larger $\bar{R}^2$'s are more desirable. We see that the two-step algorithm generates more or less the same result if we choose r large enough. Also the two step algorithm sometimes generates better solutions, when $s + r \geq 20$.

3.2 Test for real data associated with failure discriminant analysis

We next compare the two-step algorithm and S-plus using real data associated with failure discriminant analysis where $y_t = 1$ or 0 depending upon whether the tth company failed or not and x_{jt} is the jth financial attribute of the tth company. We prepared four data sets with $(T, k) = (200, 50)$, (200,70), (1000,50), (1000,70) randomly chosen from 6,556 corporate data among which 40% failed.

Figure 2 shows the distribution of correlation coefficients between pairs of 50 financial attributes. We see that the majority of financial attributes are lowly correlated with correlation coefficient between -0.3 and 0.3, but there are a non-negligible number of highly correlated pairs.

Figures 3 and 4 show the quality of fitting. We see that the two-step algorithm generates a significantly better solution. In fact, when $s + r$ is over 15 and $T = 1{,}000$, the two-step algorithm outperforms S-plus.

Table 6 shows the CPU time for the two-step algorithm. We see that computation time is much smaller than the random case. Finally, Fig. 5 shows the magnitude of the absolute deviation calculated by solving $Q_k(s)$ and the sequential method similar to the one used in

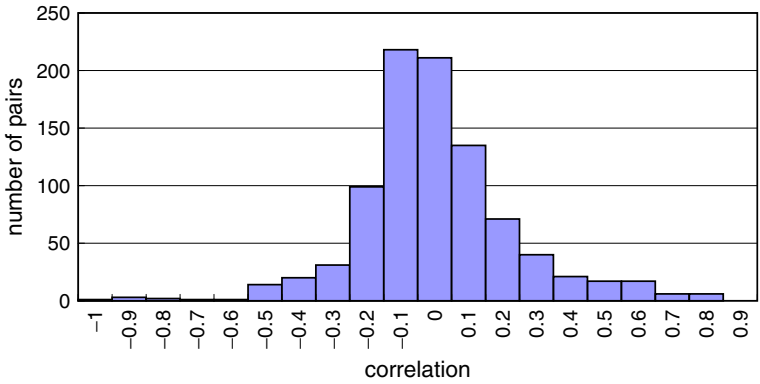

Fig. 2 Histogram of correlation coefficients ($k = 50$)

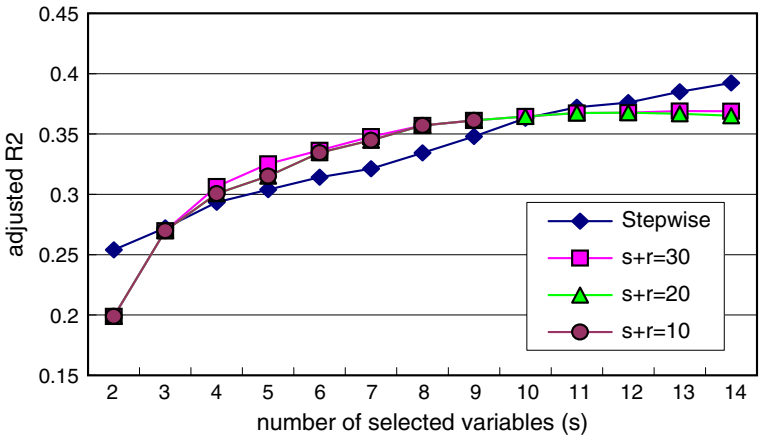

Fig. 3 Comparison of $\bar{R}^2$ ($T = 200, k = 50$)

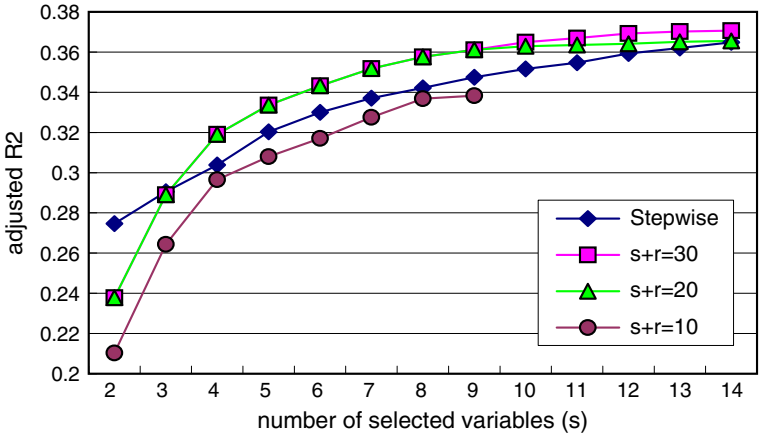

Fig. 4 Comparison of $\bar{R}^2$ ($T = 1{,}000, k = 70$)

Table 6 CPU time (s)

k	$s+r$	1st	2nd
(a) $T = 200$			
50	10	0.65	0.55
	20	0.65	0.86
	30	0.65	1.33
	S-plus	–	6.22
70	10	135.52	0.61
	20	24.78	1.69
	30	1.08	17.94
	S-plus	–	9.65
(b) $T = 1{,}000$			
50	10	10.17	1.36
	20	10.31	3.01
	30	0.53	3.61
	S-plus	–	11.54
70	10	1500.00	1.34
	20	1500.00	2.98
	30	23.23	13.00
	S-plus	–	18.63

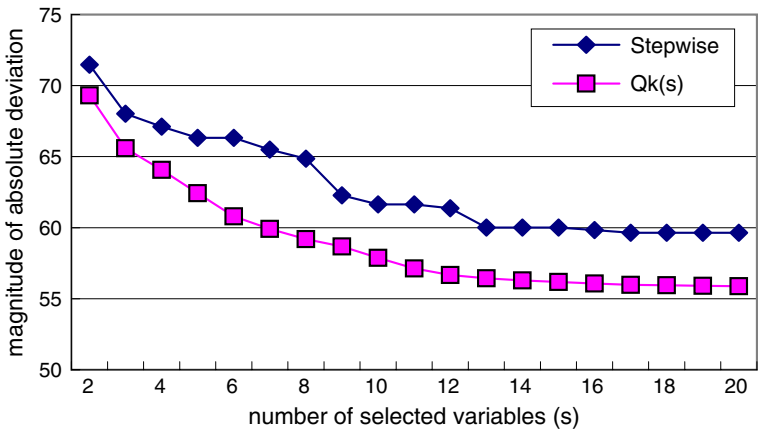

Fig. 5 Comparison of the magnitude of absolute deviation ($T = 200$, $k = 50$)

S-plus for least squares fitting. We see that the solution of $Q_k(s)$ is significantly better than that obtained by the sequential method.

4 Conclusions and future research directions

We showed in this paper that the problem posed in the Introduction can be solved within a practical amount of computation time if k is less than, say 100 by employing absolute deviation as the measure of variation.

The selected set of variables may be used in the subsequent regression analysis. Also we may be able to recover the best set of variables in terms of squared deviation by using two-step algorithm explained in Sect. 2. The quality of solutions obtained by this method is usually very close to that of S-plus and sometimes superior.

The largest problem solved in this paper is $(T, k) = (1000, 100)$, but problems with $(T, k) = (2000, 200)$ would be solved without too much difficulty. For larger problems with k over 200, we are now developing a good heuristic algorithm.

Also, we are trying to develop an efficient method for solving problem (6) without imposing the condition (10). Further, we are trying to solve problem (8) with additional constraint on the choice of variables. For example it may be better to avoid the inclusion of highly correlated variables in the model. These extensions will be reported in a forthcoming paper.

Acknowledgements The research of the first author has been partly supported by the Grant-in-Aid for Scientific Research B18310109 of MEXT of the Government of Japan.

References

1. Akaike, H.: A new look at the statistical model identification. IEEE Trans. Automat Control **19**, 716–723 (1974)
2. Bloomfield, P., Steiger, W.L.: Least Absolute Deviations: Theory, Applications, and Algorithms. Birkhäuser, Boston (1983)
3. Burnham, K., Anderson, D.: Model Selection and Multimodel Inference: A Practical Information Theoretic Approach, 2nd edn. Springer, Berlin (2002)
4. Chvatál, V.: Linear Programming. Freeman and Co., New York (1983)
5. CPLEX10.1 User's Manual, ILOG (2006)
6. Furnival, G.M., Wilson, R.W. Jr.: Regressions by leaps and bounds. Technometrics **16**, 499–511 (1974)
7. Galindo, J., Tamayo, P.: Credit risk assessment using statistical and machine learning: basic methodology and risk modeling applications. Comput. Econ. **15**, 107–143 (2000)
8. Konno, H., Kawadai, N., Wu, D.: Estimation of failure probability using semi-definite logit model. Comput. Manage. Sci. **1**, 59–73 (2003)
9. Miller, A.J.: Subset Selection in Regression. Chapman and Hall, London (1990)
10. Osborne, M.R.: On the computation of stepwise regressions. Australia Comput. J. **8**, 61–68 (1976)
11. Pardalos, P., Boginski, V., Vazacopoulos, A.: Data Mining in Biomedicine. Springer, Berlin (2007)
12. S-PLUS 6 for Windows Guide to Statistics, vol. 1. Insightful Corporation (2001)
13. Wolsey, L.A.: Integer Programming. Wiley, New York (1998)