Journal of the Korean Society of Marine Engineering, Vol. 40, No.2 pp. 138~145, 2016
J. Korean Soc. of Marine Engineering (JKOSME)
http://dx.doi.org/10.5916/jkosme.2016.40.2.138

ISSN 2234-7925 (Print)
ISSN 2234-8352 (Online)
Original Paper

Subset selection in multiple linear regression: An improved Tabu search

Jaegug Bae[1] · Jung-Tae Kim[2] · Jae-Hwan Kim[†]

(Received January 25, 2016 ; Revised February 1, 2016 ; Accepted February 1, 2016)

Abstract: This paper proposes an improved tabu search method for subset selection in multiple linear regression models. Variable selection is a vital combinatorial optimization problem in multivariate statistics. The selection of the optimal subset of variables is necessary in order to reliably construct a multiple linear regression model. Its applications widely range from machine learning, time-series prediction, and multi-class classification to noise detection. Since this problem has NP-complete nature, it becomes more difficult to find the optimal solution as the number of variables increases. Two typical metaheuristic methods have been developed to tackle the problem: the tabu search algorithm and hybrid genetic and simulated annealing algorithm. However, these two methods have shortcomings. The tabu search method requires a large amount of computing time, and the hybrid algorithm produces a less accurate solution. To overcome the shortcomings of these methods, we propose an improved tabu search algorithm to reduce moves of the neighborhood and to adopt an effective move search strategy. To evaluate the performance of the proposed method, comparative studies are performed on small literature data sets and on large simulation data sets. Computational results show that the proposed method outperforms two metaheuristic methods in terms of the computing time and solution quality.

Keywords: Metaheuristics, Improved tabu search, Subset selection problem

1. Introduction

An important subset selection, which is a vital combinatorial optimization problem in multivariate statistics, is considered in this paper. The objective of the problem is to provide faster and more cost-effective predictors for the purpose of improving the prediction performance **[1]**. Its applications widely range from machine learning, time-series prediction, and multi-class classification to noise detection.

In this paper, we focus on the variable selection problem of multiple linear regression models. It is the selection of the optimal subset of variables in order to reliably construct a multiple linear regression model. There is no doubt that the all-possible regression approach called exact method is the best because it examines every possible model for the p independent variables. However, since this problem has NP-complete nature, it becomes more difficult to find the optimal solution by all-possible regression approach. When the number of variable generally exceeds 40, it is no longer practical to obtain the optimum by exact methods. Exact methods are based on the branch-and-bound (BNB) algorithm. Furnival and Wilson **[2]** proposed the earliest BNB algorithm for this problem of multiple linear regression model. Many authors have further developed the efficient BNB algorithms (Duarte Silva **[3][4]**, Gatu and Kontoghiorghes **[5]**, Hofmann *et al.* **[6]**, Brusco *et al.* **[7]**, and Pacheco *et al.* **[8]**).

Heuristic methods are more frequently used for the large size of the subset selection problem. Heuristic methods are usually classified into two categories: simple heuristics and meta- heuristics. Simple heuristic methods involve forward selection, backward elimination, and stepwise regression. The computing time for simple heuristics is fast, but the solution quality is generally poor. Metaheuristic methods have been developed to provide better solution quality than simple heuristics. Two typical metaheuristic methods have been used previously to solve the optimal subset selection problem: tabu search (TS) **[9]** and hybrid genetic and simulated annealing algorithm (GSA) **[10]**. However, these two methods have shortcomings. The tabu search method requires a large amount of computing time, and the hybrid GSA method produces a less accurate solution.

† Corresponding Author (ORCID: http://orcid.org/0000-0002-8248-6325): Department of Data Information, Korea Maritime and Ocean University, 727, Taejong-ro, Yeongdo-gu, Busan 49112, Korea, E-mail: jhkim@kmou.ac.kr, Tel: 051-410-4374

1 Department of Data Information, Korea Maritime and Ocean University, E-mail: jgbae@kmou.ac.kr, Tel: 051-410-4377

2 Department of Data Information, Korea Maritime and Ocean University, E-mail: jt0998@gmail.com, Tel: 051-410-4884

To overcome their shortcomings, we propose an improved tabu search to reduce moves of the neighborhood and to adopt an effective search strategy for neighborhoods. To evaluate the performance of the proposed method, comparative studies are performed on small literature data sets and on large simulation data sets.

The remainder of this paper is organized as follows. The model of subset selection problem is introduced in Section 2. In Section 3, the previous two metaheuristic methods, which are TS **[9]** and hybrid GSA **[10]** are briefly described, and we propose an improved tabu search in Section 4. In Section 5, the results of the computational experiments on both benchmark problem **[10]** and simulation data sets are presented. Finally conclusions are offered in Section 6.

2. The Subset Selection Problem

Finding an appropriate subset of regressor variables for the model is usually called the subset selection (or variable selection) problem. It is the selection of the optimal subset of variables in order to reliably construct a multiple linear regression model.

Let p the number of independent variables in the full model and k the number of independent variables selected in the model. The subset selection model is as follows.

$$\mathrm{Y} = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k + \varepsilon \tag{1}$$

There are $2^{\mathrm{p}} - 1$ possible subset models. When $p > 40$, computational burdens to construct optimal subset model are increased exponentially due to the NP-complete nature of the problem.

For the subset selection problem, several measures with respect to the selection criteria have been proposed such as adjusted R^2, Mallow's Cp, and Akaike's AIC (see Draper and Smith **[11]** and Mogomery *et al.* **[12]**). In this paper, we focus on the selection criterion of adjusted R^2. The adjusted R^2 is given by

$$R_{adj}^2 = 1 - \frac{SSE_k/(n-k)}{SST/(n-1)} \tag{2}$$

where SSE_k is residual sum of squares for the k-variable model, SST is total sum of squares, and n is the number of observations.

3. The Previous Metaheuristic Methods

3.1 TS (Drezner **[9]**)

Tabu search, designed to escape from local optimum, is a metaheuristic algorithm for solving optimization problems. Motivated by Glover **[13]** as an optimization tool applicable to nonlinear covering problems, the TS algorithm was originally proposed by Glover **[14]**. The basic idea of the TS is to expand its search beyond local optimality using adaptive memory. The adaptive memory is a mechanism based on the tabu list of prohibited moves. Use of the tabu list is one way to prevent cycling and guide the search towards unexplored region of the solution space.

The tabu search algorithm has been successfully applied to telecommunications path assignment **[15]**, neural network pattern recognition **[16]**, machine learning **[17]**, just-in-time scheduling **[18]**, and electronic circuit design **[19]**.

Drezner **[9]** developed a tabu search for the subset selection problem. The TS procedure is described in the following pseudocode.

```
s = initial random solution
best = s
tabuList = Ø
While ( i >= 30 ) Do
        bestNeighbor = null
        For ( s' in neighborhood(s) ) Do
                If ( not tabuCodition(s,tabuList)) and
                        f(s') > f(bestNeighbor)) Then
                        bestNighbor = s'
                End If
        End For
        tabuList.push(featureDifferences(s,bestNighbor))
        While( tabulist.size > 10 )
                tabuList.removeFirst()
        End While
        s = bestNeighbor
        If ( f(s) > f(best) ) Then
                best = s
                i = 1
        Else
                i = i + 1
        End If
End While
```

Drezner **[9]** generated all neighborhoods by three moves, which are adding a variable, removing a variable, and swapping variables. For example, consider a set of p = 5 independent variables: full-set = $\{x_1, x_2, x_3, x_4, x_5\}$ and a subset of k= 2 variables: $\{x_1, x_2\}$. All neighborhoods of this subset consist of the following moves in **Table 1**.

Table 1: All neighborhoods of subset $\{x_1, x_2\}$

Adding a variable:	$\{x_1, x_2, x_3\}$, $\{x_1, x_2, x_4\}$, $\{x_1, x_2, x_5\}$
Removing a variable:	$\{x_1\}$, $\{x_2\}$
Swapping variables:	$\{x_2, x_3\}$, $\{x_2, x_4\}$, $\{x_2, x_5\}$, $\{x_1, x_3\}$, $\{x_1, x_4\}$, $\{x_1, x_5\}$

Drezner **[9]** also adapted the stopping criterion as the total number of 30 iterations without improving the best-so-far solution. The size of the tabu list was set to 10.

3.2 Hybrid GSA

```
population = initial random population
best = null
For ( i = 1 ; i <= 1000 ; i = i+1) Do
    b = the best solution in population
    If (f(best) < f(b) ) Then
        best = b
    End If
    new_population = null
    For (j = 1 ; j <= 100 ; j = j+2) Do
        parent1 = selection(population)
        parent2 = selection(population)
        If (random(0,1) < 0.8) Then
            children = crossover(parent1,parent2)
            child1 = children[1]
            child2 = children[2]
        Else
            child1 = parent1
            child2 = parent2
        End If
        child1 = mutation(child1)
        child2 = mutation(child2)
        △E1 = f(child1) – f(parent1)
    If (△E1 > 0 or random(0,1) < exp(△E1 / t) Then
            new_population[j] = child1
        Else
            new_population[j] = parent1
        End If
        △E2 = f(child2) – f(parent2)
    If (△E2 > 0 or random(0,1) < exp(△E2 / t) Then
            new_population[j+1] = child2
        Else
            new_population[j+1] = parent2
        End If
    End For
    population = new_population
    t = 0.9*t
End For
```

Lin *et. al*. **[20]** suggested the original version of GSA for solving some NP-hard problems such as knapsack problem, travelling salesman problem, and set partitioning problem. Hasan **[10]** proposed a hybrid GSA for the subset selection problem, in which a genetic algorithm (GA) **[21]** is combined with a simulated annealing (SA) algorithm **[22]**. The pseudocode for the hybrid GSA algorithm is as follows.

The SA operator is incorporated into the generation of children produced by the genetic operators. It is applied to decide which two of the parents and children remain. That is, if children are better than parents, then the parents is replaced by the children. If parents are better, they are replaced with the chosen probability as shown in the pseudocode of GSA.

Hasan **[10]** suggested that the number of the population, crossover rate, mutation rate, maximum number of iteration, the initial temperature and the cooling rate is chosen as 100, 0.8, 0.1, 1000, 100 and 0.9, respectively.

4. The Proposed Method

An improved tabu search (ITS) is proposed in this paper for solving the subset selection problem. The proposed method addresses shortcomings in two typical metaheuristic methods that have previously been developed. The tabu search method takes a large amount of computing time, due to many neighborhood moves, and the hybrid GSA method produces a less accurate solution. The proposed method is a fast tabu search that reduces moves of the neighborhood and adopts an effective search strategy for neighborhoods.

4.1 Neighborhood Moves

The neighborhood of Drezner **[9]** was generated by three moves, which are adding a variable, removing a variable, and swapping variables. In **Table 1** of the previous section, however, most of neighborhoods of swapping variables can be tentatively explored by the search engine with the neighborhood of only adding and removing a variable. For example, if the subset $\{x_2, x_3\}$ generated by swapping variables is optimum, it can be also obtained by adding $\{x_1, x_2\}$ to x_3 variable, and then removing x_1 variable from the resulting subset of $\{x_1, x_2, x_3\}$. That is, it can be transitively searched from $\{x_1, x_2\}$ to $\{x_1, x_2, x_3\}$ to $\{x_2, x_3\}$. This implies that the move of swapping may be replaced by two moves of adding and removing.

As the number of variable increases, the number of the neighborhood of swapping becomes larger. The number of neighborhood of swapping variables is totally k(p-k).

In order to reduce the computing time, we suggest that the neighborhood with swapping should be entirely excluded from the proposed method. Practically, from our computational results, we noticed that our search engine with only two moves, adding or removing a variable, plays a sufficient role in improving the current best solution. Consequently, the neighborhood strategy without swapping variables reduces a considerable amount of computing time.

4.2 Search Strategies for Neighborhoods

Widmer & Hertz **[23]** and Tailard **[24]** proposed tabu search for the flow shop scheduling problem. They also suggested two specific search strategies: the best move strategy and the first move strategy. The best move strategy examines all neighborhoods and selects the best move that is not on the prohibited tabu list. The first move strategy examines the neighborhoods and selects the first move that improves the current best solution. If there is no first move that improves the current best solution, the first move strategy eventually becomes equivalent to the best move strategy.

As depicted in **Figure 1**, the procedure of improving the best solution can be divided into two phases: Phase-I and Phase-II. In the Phase-I, the procedure for improving the best solution is rapidly progressed. After the solution reaches a local optimum through the tabu search, it is difficult to improve the local optimum. Accordingly, the procedure of improving the local optimum is slowly processed in the Phase-II

In the improved tabu search, the first move strategy is adopted.

In Phase-I, the first move strategy plays a role in rapidly improving the current best solution.

The pseudocode of the first move strategy is as follows.

```
s = initial random solution
best = s
tabuList = ∅
i = 1
While (i >= 30) Do
    bestNeighbor = null
    For ( s' in neighborhood(s) ) Do
        If ( s' ∉ tabulist and f(bestNeighbor) < f(s') )Then
            bestNeighbor = s'
            If ( f(best) < f(bestNeighbor)) Then
                Break
            End If
        End If
    End For
    s = bestNeighbor
    If ( f(s) > f(best)) Then
        best = s
        i = 1
    Else
        i = i + 1
    End If
        tabuList.push(s)
    If ( tabuList.size > 10 ) Then
        tabuList.removeFirst()
    End If
    i = i + 1
End While
```

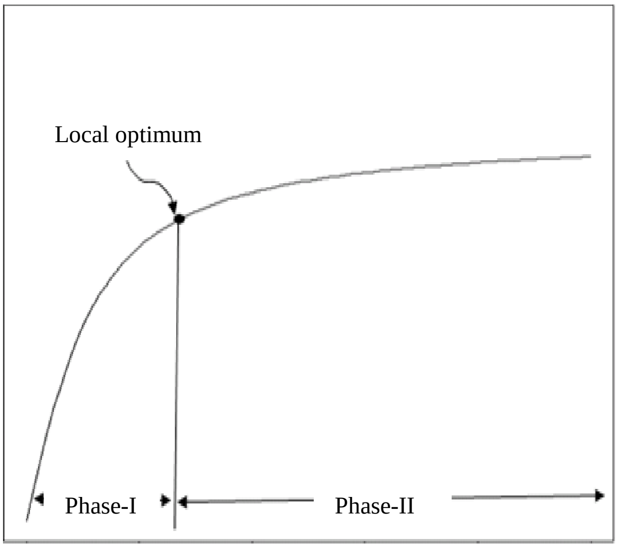

Figure 1: The Phase-I and Phase-II

4.3 Tabu List

In Drezner **[9]**, the tabu list contains a list of moves. In the proposed ITS method, the tabu list contains a list of solutions. In our experiments, we noticed that our tabu list of solutions is more efficient than that of moves for this problem.

4.4 Stopping Criterion

In our experiments, stopping criterion is the total number of 30 iterations without improving the best-so-far solution.

5. Computational Results

Comparative analytical tests were performed to compare the proposed method with the two previous metaheuristic methods. Experiments were initially performed using data sets obtained from the literature to evaluate their performance.

These data sets consisted of a small number of variables (less than 24). To further evaluate the performance of the proposed method for large number of variables, we randomly generated large data sets up to 100 variables. That is, the number of variable varies from 40, 60, 80 to 100. Results for the small literature data sets are reported in the Section 5.1. Computational results for the large simulation data sets are summarized in the Section 5.2. All computational experiments were conducted on an Intel i7 PC with 3.4 GHz CPU and 8 GB RAM, and all source codes were implemented with R language.

5.1 The Benchmark Problem

In **Table 2**, *p*, *n*, *Best*, and *Freq*. are defined as follows.

p : the total number of independent variables.
n : the number of sample data.
Best : the Maximum of R_{adj}^2 for 10 trials.
Mean : the Mean of R_{adj}^2 for 10 trials.
Freq : the number of *Best* found by each method for 10 trials.

As the experimental results in **Table 2** indicate, all methods find the optimal value of R^2_{adj}. In the value of *Freq.*, however, the performance of TS is more or less worse than that of the proposed ITS method.

5.2 Simulation Data Sets

To further test the performance of the ITS method, the simulation data sets were randomly generated as follows.

i) Independent variables were generated by normal distribution with a mean 0 and a standard deviation 1.

ii) Error terms were generated by normal distribution with a mean 0 and a standard deviation $\lambda\sigma$, where σ is standard deviation of actual regression equation, and λ is a constant.

iii) The number of sample data is five times the total number of independent variables.

The value of E is given by the following **Equation (3)**.

$$E = \frac{k}{p} \quad , \tag{3}$$

where *k* is the number of sample data and is five times the total number of independent variables, and *p* is the total number of independent variables.

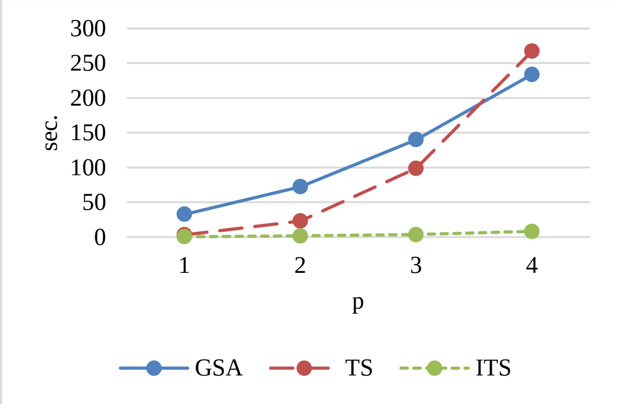

Figure 2: The Computing Time of GSA, TS and ITS

Table 3: The computing (sec.) of GSA , TS and ITS

p	GSA	TS	ITS
40	32.641	3.109	0.447
60	72.442	22.984	1.593
80	140.028	98.874	3.546
100	233.830	267.518	8.058

From the computational results shown in **Tables 4** to **7**, it is clear that the proposed ITS method outperforms the GSA and TS methods in terms of the computing time and solution quality. As shown in **Figure 2**, the ITS method is the fastest. Specifically, when the value of *p* is 100, the GSA and TS methods take a considerable amount of computing time. As seen in **Table 3**, the computing time (sec.) of GSA, TS, and ITS is 233.8, 267.5 and 8.058, respectively.

6. Conclusions

In the subset selection problem, all-possible regression or exact algorithms, such as branch-and-bound programming, obtain the global optimum, but their computational feasibility diminishes for a large number of variables (p > 40) due to long processing times. Simple heuristic methods, often used in statistical SAS programs, include forward selection, backward elimination, and stepwise regression methods. Their computing time is fast, but solution quality is generally poor.

In general, metaheuristic methods provide better solution quality than simple heuristics. In the subset selection problem, two typical metaheuristic methods, TS and hybrid GSA, have been used. However, these two methods have shortcomings. The tabu search method requires a large amount of computing time, due to many neighborhood moves, and the hybrid GSA method produces a less accurate solution.

Table 2: Experimental results of GSA, TS (Drezner [9]) and ITS (Improved TS)

Data Set	*p*	*n*	GSA			TS			ITS		
			Best	*Mean*	*Freq.*	*Best*	*Mean*	*Freq.*	*Best*	*Mean*	*Freq.*
Auto	11	65	0.543027	0.543027	10/10	0.543027	0.542852	8/10	0.543027	0.543027	10/10
Bankbill	15	71	0.994915	0.994915	10/10	0.994915	0.994898	7/10	0.994915	0.994915	10/10
Belle	7	27	0.649502	0.649502	10/10	0.649502	0.649502	10/10	0.649502	0.649502	10/10
Bodywomen	23	260	0.546150	0.546150	10/10	0.546150	0.546125	7/10	0.546150	0.546150	10/10
Horse	13	102	0.870527	0.870527	10/10	0.870527	0.870527	10/10	0.870527	0.870527	10/10
Papir	15	29	0.972387	0.972387	10/10	0.972387	0.972387	10/10	0.972387	0.972387	10/10
Pysical	10	22	0.964580	0.964580	10/10	0.964580	0.964580	10/10	0.964580	0.964580	10/10
US Crime	15	47	0.597745	0.597745	10/10	0.597745	0.897685	9/10	0.597745	0.597745	10/10

Table 4: Experimental results of GSA, TS and ITS (p=40, n=200)

E	*λ*	GSA			TS			ITS		
		Best	*Mean*	*Freq.*	*Best*	*Mean*	*Freq.*	*Best*	*Mean*	*Freq.*
0.25	0.25	0.936571	0.936571	10/10	0.936571	0.936571	10/10	0.936571	0.936571	10/10
	0.65	0.743673	0.743665	9/10	0.743673	0.743616	8/10	0.743673	0.743673	10/10
0.5	0.25	0.936571	0.936571	10/10	0.936571	0.936571	10/10	0.936571	0.936571	10/10
	0.65	0.705421	0.705421	10/10	0.705421	0.705360	6/10	0.705421	0.705421	10/10
0.75	0.25	0.943397	0.943397	10/10	0.943397	0.943353	7/10	0.943397	0.943397	10/10
	0.65	0.760162	0.760162	10/10	0.760162	0.759768	7/10	0.760162	0.760162	10/10

Table 5: Experimental results of GSA, TS and ITS (p=60, n=300)

E	*λ*	GSA			TS			ITS		
		Best	*Mean*	*Freq.*	*Best*	*Mean*	*Freq.*	*Best*	*Mean*	*Freq.*
0.25	0.25	0.948695	0.948466	0/10	0.948737	0.948737	9/10	0.948737	0.948737	10/10
	0.65	0.689304	0.690387	0/10	0.690549	0.690504	7/10	0.690549	0.690549	10/10
0.5	0.25	0.944630	0.944193	0/10	0.944781	0.944781	10/10	0.944781	0.944781	10/10
	0.65	0.713936	0.712843	0/10	0.714963	0.714915	2/10	0.714963	0.714963	10/10
0.75	0.25	0.940132	0.939687	1/10	0.940132	0.940116	8/10	0.940132	0.940132	10/10
	0.65	0.720683	0.718374	0/10	0.720867	0.720479	6/10	0.720867	0.720867	10/10

Table 6: Experimental results of GSA, TS and ITS (p=80, n=400)

E	*λ*	GSA			TS			ITS		
		Best	*Mean*	*Freq.*	*Best*	*Mean*	*Freq.*	*Best*	*Mean*	*Freq.*
0.25	0.25	0.944685	0.944299	0/10	0.945054	0.945049	7/10	0.945054	0.945054	10/10
	0.65	0.733602	0.732183	0/10	0.736012	0.735993	8/10	0.736012	0.736012	10/10
0.5	0.25	0.942780	0.941569	0/10	0.944055	0.944046	7/10	0.944055	0.944055	10/10
	0.65	0.696030	0.691928	0/10	0.70374	0.703740	10/10	0.703740	0.703740	10/10
0.75	0.25	0.935835	0.910728	0/10	0.939282	0.939282	10/10	0.939282	0.939282	10/10
	0.65	0.659400	0.655344	0/10	0.674227	0.674207	9/10	0.674227	0.674227	10/10

Table 7: Experimental results of GSA, TS and ITS (p=100, n=500)

E	*λ*	GSA			TS			ITS		
		Best	*Mean*	*Freq.*	*Best*	*Mean*	*Freq.*	*Best*	*Mean*	*Freq.*
0.25	0.25	0.937582	0.937260	0/10	0.938899	0.938883	4/10	0.938899	0.938899	10/10
	0.65	0.712726	0.711461	0/10	0.719262	0.719206	6/10	0.719262	0.719262	10/10
0.5	0.25	0.942780	0.941569	0/10	0.944055	0.944046	7/10	0.944055	0.944055	10/10
	0.65	0.696030	0.691928	0/10	0.703740	0.703740	10/10	0.703740	0.703740	10/10
0.75	0.25	0.947111	0.943321	0/10	0.947830	0.947830	10/10	0.947830	0.947830	10/10
	0.65	0.728375	0.722315	0/10	0.732835	0.732833	9/10	0.732835	0.732835	10/10

To overcome the shortcomings of these methods, an improved tabu search (ITS) was developed to reduce moves of the neighborhood and to adopt an effective search strategy for the neighborhoods. To evaluate the performance of the proposed ITS method, comparative analytical tests were performed on small literature data sets and on large simulation data sets. Computational results showed that the proposed method outperforms the previous metaheuristics in terms of the computing time and solution quality.

References

[1] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," Journal of Machine Leaning Research, vol. 3, pp. 1157-1182, 2003.

[2] G. M. Furnival and R.W. Wilson, "Regression by leaps and bounds," Technometrics, vol. 16, pp. 416-423, 1974.

[3] A. P. D. Silva, "Efficient variable screening for multivariate analysis," Journal of Multivariate Analysis, vol.76, pp. 35-62, 2001.

[4] A. P. Duarte-Silva, "Discarding variables in a principal component analysis: algorithms for all-subsets comparisons," Computational Statistics, vol. 17 pp. 251-271, 2002.

[5] C. Gatu and E. J. Kontoghiorghes, "Branch-and-bound algorithms for computing the best-subset regression models," Journal of Computational and Graphical Statistics, vol. 15, no. 1, pp. 139-156, 2006.

[6] M. Hofmann, C. Gatu, and E. J. Kontoghiorghes, "Efficient algorithms for computing the best subset regression models for large-scale problems," Computational Statistics and Data Analysis, vol. 52, no. 1, pp. 16-29, 2007.

[7] M. J. Brusco, D. Steinley, and J. D. Cradit, "An exact algorithm for hierarchically well-formulated subsets in second-order polynomial regression," Technometrics, vol. 51, no. 3, pp. 306-315, 2009.

[8] J. Pacheco, S. Casado, and S. Porras, "Exact methods for variable selection in principal component analysis: Guide functions and pre-selection," Computational Statistics and Data Analysis, vol. 57, no. 1, pp. 95-111, 2013.

[9] Z. Drezner and G. A. Marcoulides, "Tabu seach model selection in multiple regression analysis," Communications in Statistics – Simulation and Computation, vol. 28, no. 9, pp. 349-367, 1999.

[10] H. Hasan, "Subset selection in multiple linear regression models: A hybrid of genetic and simulated annealing algorithms," Applied Mathematics and Computation, vol. 219, no. 23, pp. 11018-11028, 2013.

[11] N. R. Draper and H. Smith, Applied Regression Analysis, 3th Edition, NewYork: Wiley, 1998.

[12] D. G. Montgomery and E. A Peck, Introduction to Linear Regression Analysis, 2nd Edition, NewYork: Wiley, 1992.

[13] F. Glover, "Heuristics for integer programming using surrogate constraints", Decision Sciences, vol. 8, no. 1, pp. 156-166, 1977.

[14] F. Glover, "Future paths for integer programming and links to artificial intelligence," Computers and Operations Research, vol. 13, no. 5, pp. 533-549, 1986.

[15] S. Oliveira and G. Stroud, "A parallel version of tabu search and the assignment problem," Heuristics for Combinatorial Optimization, vol. 4, pp. 1-24, 1989.

[16] D. D. Werra and A. Herz, "Tabu search techniques: a tutorial and an application to neural networks," OR Spektrum, vol. 11, pp. 131-141, 1989.

[17] M. Laguna, J. W. Barnes, and F. Glover, "Tabu search methods for a single machine scheduling problem", Journal of Intelligent Manufacturing, vol. 2, no. 2, pp. 63-74, 1991.

[18] M. Laguna and J. L. G. Velarde, "A search heuristic for just-in-time scheduling in parallel machines," Journal of Intelligent Manufacturing, vol. 2, no. 4, pp. 253-260, 1991.

[19] J. A. Bland and G. P. Dawson, "Tabu search and design optimization," Computer Aided Design, vol. 23, no. 3, pp. 195-202, 1991.

[20] F. T. Lin, C. Y. Kao, and C. C. Hsu, "Applying the genetic approach to simulated annealing in solving some NP-hard problems," IEEE Transactions on System Man Cybernetics, vol. 23, no. 6, pp. 1752-1767, 1993.

[21] J. H. Holland, "Adaptaion in natural and artificial systems," University of Michigan Press, 1975.

[22] S. Kirpatirck, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," Science, vol. 220, pp. 671-680, 1983.

[23] M. Widmer and A. Hertz, “A new heuristic method for the flow shop sequencing problem,” European Journal of Operational Research, vol. 41, no. 2, pp. 186-193, 1989.

[24] E. Tailard, “Some efficient heuristic methods for the flow shop sequencing problem,” European Journal of Operational Research, vol. 47, no. 1, pp. 65-74, 1990.