이학석사 학위논문

다중선형회귀모델의 변수선택을 위한 개선된 타부서치 알고리즘

Subset Selection in a Multiple Linear Regression Model: An Improved Tabu Search

지도교수 배 재 국

공동지도교수 김 재 환

2016년 2월

한국해양대학교 대학원

데이터정보학과

김 정 태

본 논문을 김정태의 이학석사 학위논문으로 인준함.

위원장	김 재 환	(인)
위 원	배 재 국	(인)
위 원	장 길 웅	(인)

2015년 11월 24일

한국해양대학교 대학원

Contents

List of Tables

List of Figures

Subset Selection in a Multiple Linear Regression Model: An Improved Tabu Search

Kim, Jung Tae

Department of Data Information
Graduate School of Korea Maritime and Ocean University

Abstract

This thesis deals with the subset selection that is a vital combinatorial optimization problem in multivariate statistics. It is the selection of the optimal subset of variables in order to reliably construct a multiple linear regression model. Since this problem has NP-complete nature, the larger the size of the variables, the harder to find the optimal solution. In general, many metaheuristic methods have been developed to tackle the problem. In the subset selection problem, two typical metaheuristics, which are tabu search and hybrid GSA (genetic and simulated annealing algorithm), was proposed. However, they have some shortcomings, that is, the tabu search takes a lot of computing time due to many neighborhood moves and GSA's solution quality is less accurate. This paper proposes an improved tabu search algorithm to reduce moves of the neighborhood and adopt the appropriate move search strategy. To evaluate the performance of the proposed method, a comparative study is performed on both the literature data sets and simulation data sets. Computational results show that the proposed method outperforms the previous metaheuristics in terms of the computing time and solution quality.

KEY WORDS: metaheuristics, an improved tabu search, the subset selection problem

1. Introduction

An important subset selection, which is a vital combinatorial optimization problem in multivariate statistics, is considered in this paper. It is the selection of the optimal subset of variables in order to reliably construct a multiple linear regression model. 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 regression, machine learning, and time-series prediction to multi-class classification.

In this paper, we focus on the variable selection problem of multiple linear regression models. There is no doubt that the all-possible regression approach, called exact method, is the best because it examines every possible model given for the p independent variables. However, since this problem has NP-complete nature, the larger the size of the variables, the harder to find the optimal solution by all-possible regression approach. Practically, when the number of variable exceeds 40, it is very difficult to obtain the optimum by exact methods. Exact methods are based on BNB (branch-and-bound) algorithm. Furnival and Wilson [2] proposed the earliest BNB algorithm for this problem of multiple linear regression model. Many authors have 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]).

Popularly, heuristic methods are applied for the large size of the subset selection problem. They are usually classified into two categories. One is the simple heuristic method and the other is metaheuristics. The former consists of forward selection, backward elimination, and step-wise regression. Their computing time is fast, but solution quality is not good. Metaheuristic methods have been developed to provide better solution quality than simple heuristics. In variable selection problem, two typical metaheuristics, which are TS (tabu search) [9] and hybrid GSA (genetic and simulated annealing algorithm) [10], was proposed. They have some shortcomings, respectively. That is, the tabu search [9] takes a lot of computing time due to many neighborhood candidates and GSA's solution quality is less accurate.

To overcome their shortcomings, we propose an improved tabu search to reduce moves of the neighborhood and to adopt the effective search strategy for neighborhoods, that is, the first move strategy (see Section 4). To evaluate the performance of the proposed method, a comparative analysis is performed on both the literature data sets [9] and simulation data sets for larger size of variables. From computational results, we notice that the proposed method outperforms the previous metaheuristics in terms of the computing time and solution quality.

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 often called the subset selection (or variable selection) problem. That is, 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.

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

There are 2^{p-1} possible subset models. When p > 40, computational burdens to construct optimal subset model are increased exponentially.

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

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

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

3. Previous Metaheuristic Methods

3.1 TS(Tabu search)

TS algorithm, originally proposed by Glover [12, 13], is a metaheuristic method to expand its search beyond local optimality using adaptive memory. The adaptive memory is a mechanism based on the tabu list of prohibited moves. The tabu list is one of the mechanism to prevent cycling and guide the search towards unexplored region of the solution space. The TS generally adopts the penalty function to allow to explore the search towards the attractive infeasible region.

The TS has been successfully applied to many combinatorial optimization problems such as vehicle routing problems, travelling salesman problems, time tabling problems, and resource allocation problems, etc.

In the subset problem, Drezner [9] developed a tabu search for this problem. The procedure of TS is described as the following pseudocode in Fig. 1.

The neighborhood is generated by three moves, which are adding a variable, removing 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: $\{x_1, x_2\}$. The neighborhood of this subset consist of the following neighborhood in Table 1.

The stopping criterion of TS is the total number of 30 iterations without improving the best so far solution, and the size of the tabu list is less than equal to 10.

```
s = a random initial solution
best = s
tabulist = empty set
**While**(stopping criterion) **do**
    s = the best solution in N(s) which is not in tabulist
    **If**(f(best)<f(s)) **then**
        best=s
    **End If**
    update tabuList
**End While**
```

Fig. 1 The pseudocode of TS

Table 1 All neighborhood of $\{x_1, x_2\}$

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 :	$\{x_2, x_3\}$, $\{x_2, x_4\}$, $\{x_2, x_5\}$, $\{x_1, x_3\}$, $\{x_1, x_4\}$, $\{x_1, x_5\}$

3.2 Hybrid GSA(A hybird of genetic and simulated annealing algorithms)

```
Initial temperature t=100, probility of crossover=0.8 and probability of mutation=0.1.
For i=1 to 1000 do
    b= the best solution in population
    If((best)<f(b) then
        best=b
    End If
    new_population=null
    For j=1 to 100 by=2 do
        parent1=selection(population), parent2=selection(population)
        If(random()<0.8) then
            offspring1=crossover(p1,p2), offspring2=crossover(p2,p1)
        Else then
            offspring1=parent1, offspring2=parent2
        End If
            offspring1=Mutation(offspring1), offspring2=Mutation(offspring1)
            $\triangle E_1$=f(offspring1)-f(parent1)
        If($\triangle E_1$>0 or random()<$exp(\triangle E_2/t)$) then
            new_population[j]=offspring1
        Else then
            new_population[j]=parent1
        End If
            $\triangle E_2$=f(offspring2)-f(parent2)
        If($\triangle E_2$>0 or random()<$exp(\triangle E_2/t)$) then
            new_population[j+1]=offspring2
        Else then
            new_population[j+1]=parent2
        End If
    End For
    population=new_population
    t=0.9t
End For
```

Fig. 2 The pseudocode of GSA

Hasan [10] proposed a hybrid GSA for the subset problem, in which GA (genetic algorithm) [14] is combined with SA (simulated annealing) algorithm [15]. Lin *et.al* [16] suggested the original version of GSA for solving some NP-hard problems such as knapsack problem, travelling salesman problem, and set partitioning problem.

The pseudocode of GSA is shown in Fig. 2. 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 repaced with the chosen probability as shown in Fig. 2.

4. The Proposed Method

The previous metaheuristics have some shortcomings, that is, the tabu search [10] takes a lot of computing time due to many neighborhood moves and GSA's solution quality is less accurate. To overcome their shortcomings, we propose an improved tabu search to reduce moves of the neighborhood and to adopt the effective search strategy for neighborhoods, that is, the fisrt move strategy.

4.1 The neighborhood moves

The neighborhood of Drezner [9] is generated by three moves, which are adding a variable, removing variable, and swapping variables. In Table 1 of Section 3.1, 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 swapping neighborhood $\{x_2,\ x_3\}$ is optimum, it can be also obtained by adding $\{x_1,\ x_2\}$ to x_3 variable and removing a 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 can be overlapped with two moves of adding and removing. Actually, the number of neighborhood of swapping variables is totally $k(p-k)$. As the number of variable increases, the number of the neighborhood of swapping becomes more or less large. Therefore, we suggest that the neighborhood of swapping variables should be entirely excluded from the proposed tabu search to reduce the computing time. Practically, from our computational results, we noticed that the search engine with only two moves, adding or removing a variable, plays a sufficient role in improving the best solution, and the neighborhood strategy without swapping variables reduces computing time.

4.2 Search strategies of neighborhoods

Widmer & Hertz [18] and Tailard [19] proposed tabu search for the flow shop scheduling problem. In this paper, we consider the following two strategies, that is, the best move strategy and the first move strategy. The best move strategy is to examine the entire neighborhood and take the best move that is not tabu. The pseudocode of the best move is described in Fig. 3. In the meanwhile, the first move strategy is to examine the neighbours and take the first move which improves the current best solution. If there is no any first move that improves the current best one, the first move strategy becomes the same way as the best move strategy.

```
s=random initial solution
best=s
tabuList=empty set
While(stopping criterion) do
    neighbor=sort(N(s))
    For i=1 to size do
        If((neighbor[i] $\not\in$ tabuList)) then
            s=neighbor[i]
            stop
        End If
    End For
    update tabuList
    If(f(best)<f(s)) then
        best=s
    End If
End While
```

Fig. 3 The pseudocode of the best move strategy

As depicted in Fig. 4, the procedure of improving the best solution can be usually divided into two phases, *i.e.* Phase-I and Phase-II. In the Phase-I, the procedure of improving the best solution is rapidly progressed. After the solution reaches a local optimum through the tabu search, it is very difficult to improve the local optimum. Accordingly, the procedure of improving the local optimum is very slowly processed in the Phase-II.

In the improved tabu search, the first move strategy is adopted. In Phase-I, it plays a role in rapidly improving the current best solution. If the solution reaches a local optimum, then the first move strategy would become the same way as the best move strategy in Phase-II.

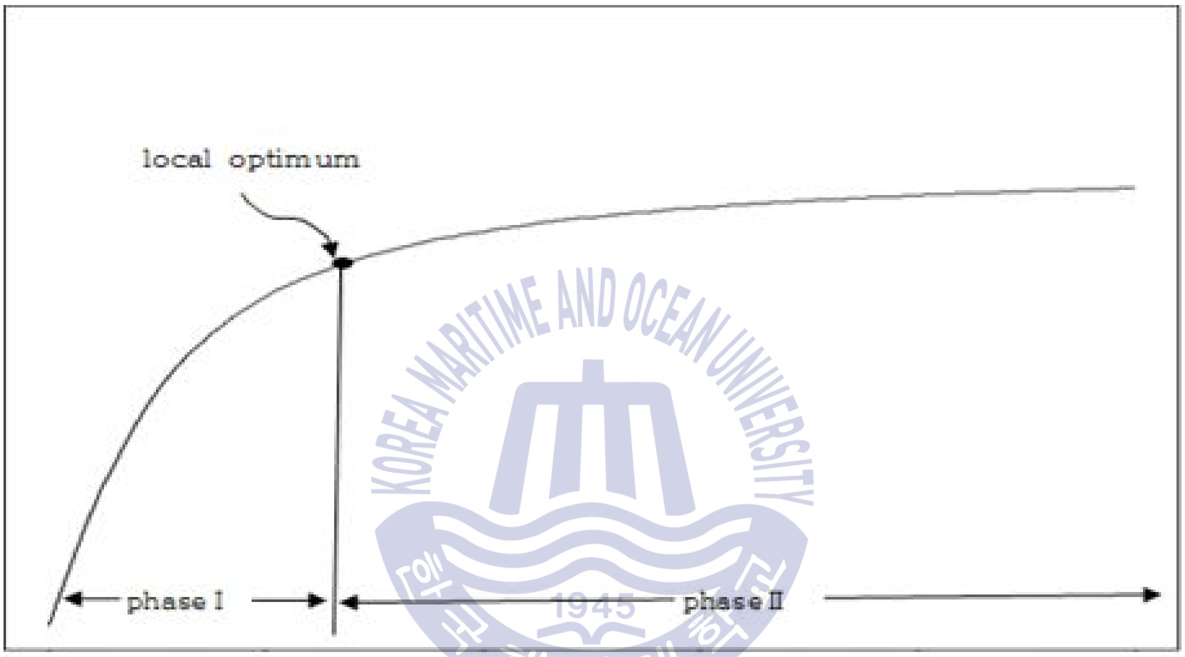

Fig. 4 The Phase-I and Phase-II

4.3 Tabu list

In Drezner [9], the tabu list contains a list of moves. In our tabu list, moves are entirely replaced by solutions. That is to say, we adopted the tabu list of solutions. In our computational experiments, we noticed that our tabu list of solutions is more efficient than that of moves for this problem.

4.4 Stopping criterion

Stopping criterion is the total number of 30 iterations without improving the best so far solution.

5. Computational Results

We performed two comparative analysis to compare the proposed method with the previous metaheuristics. At first, we experimented the data sets in the literature to evaluate their performance. These data sets [10] only consisted of the small size of variables (less than 23). Secondly, to additionally evaluate the performance of the proposed method for larger data sets, we randomly generated the data sets with up to 100 variables. That is, the number of variable varies from 40, 60, 80 to 100. In the Section 5.1, we did experiment on the data set in the literature. Also we performed computational experiments on the simulation data sets in the Section 5.2.

5.1 The benchmark problem

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

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

In table 2, p, n, Best, Freq. are defined as follows.

p: the total number of independent variables

n: the number of sample data

Best: the maximum of adjusted R^2 for 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 adjusted R^2. In the value of Freq., however, TS is not better than GSA and ITS.

5.2 The simulation data sets

we generated the simulation data sets as follows.

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

ii) Error terms are 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).

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

k: the number of independent variable that is included in the actual regression equation.

p: the total number of independent variables

Table 3 Experimental results of GSA, TS and ITS (p=40, n=200)

		GSA		TS		ITS	
E	λ	Best	Freq.	Best	Freq.	Best	Freq.
0.25	0.25	0.936571	10/10	0.936571	10/10	0.936571	10/10
	0.65	0.743673	9/10	0.743673	8/10	0.743673	10/10
0.5	0.25	0.936571	10/10	0.936571	10/10	0.936571	10/10
	0.65	0.705421	10/10	0.705421	6/10	0.705421	10/10
0.75	0.25	0.943397	10/10	0.943397	7/10	0.943397	10/10
	0.65	0.760162	10/10	0.760162	7/10	0.760162	10/10

Table 4 Experimental results of GSA, TS and ITS (p=60, n=300)

		GSA		TS		ITS	
E	λ	Best	Freq.	Best	Freq.	Best	Freq.
0.25	0.25	0.948695	0/10	0.948737	9/10	0.948737	10/10
	0.65	0.689304	0/10	0.690549	7/10	0.690549	10/10
0.5	0.25	0.944630	0/10	0.944781	10/10	0.944781	10/10
	0.65	0.713936	0/10	0.714963	2/10	0.714963	10/10
0.75	0.25	0.940132	1/10	0.940132	8/10	0.940132	10/10
	0.65	0.720683	0/10	0.720867	6/10	0.720867	10/10

Table 5 Experimental results of GSA, TS and ITS (p=80, n=400)

		GSA		TS		ITS	
E	λ	Best	Freq.	Best	Freq.	Best	Freq.
0.25	0.25	0.944685	0/10	0.945054	7/10	0.945054	10/10
	0.65	0.733602	0/10	0.736012	8/10	0.736012	10/10
0.5	0.25	0.942780	0/10	0.944055	7/10	0.944055	10/10
	0.65	0.696030	0/10	0.70374	10/10	0.70374	10/10
0.75	0.25	0.935835	0/10	0.939282	10/10	0.939282	10/10
	0.65	0.659400	0/10	0.674227	9/10	0.674227	10/10

Table 6 Experimental results of GSA, TS and ITS (p=100, n=500)

		GSA		TS		ITS	
E	λ	Best	Freq.	Best	Freq.	Best	Freq.
0.25	0.25	0.937582	0/10	0.938899	4/10	0.938899	10/10
	0.65	0.712726	0/10	0.719262	6/10	0.719262	10/10
0.5	0.25	0.942780	0/10	0.944055	7/10	0.944055	10/10
	0.65	0.696030	0/10	0.703740	10/10	0.703740	10/10
0.75	0.25	0.947111	0/10	0.947830	10/10	0.947830	10/10
	0.65	0.728375	0/10	0.732835	9/10	0.732835	10/10

Table 7 The computing time(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

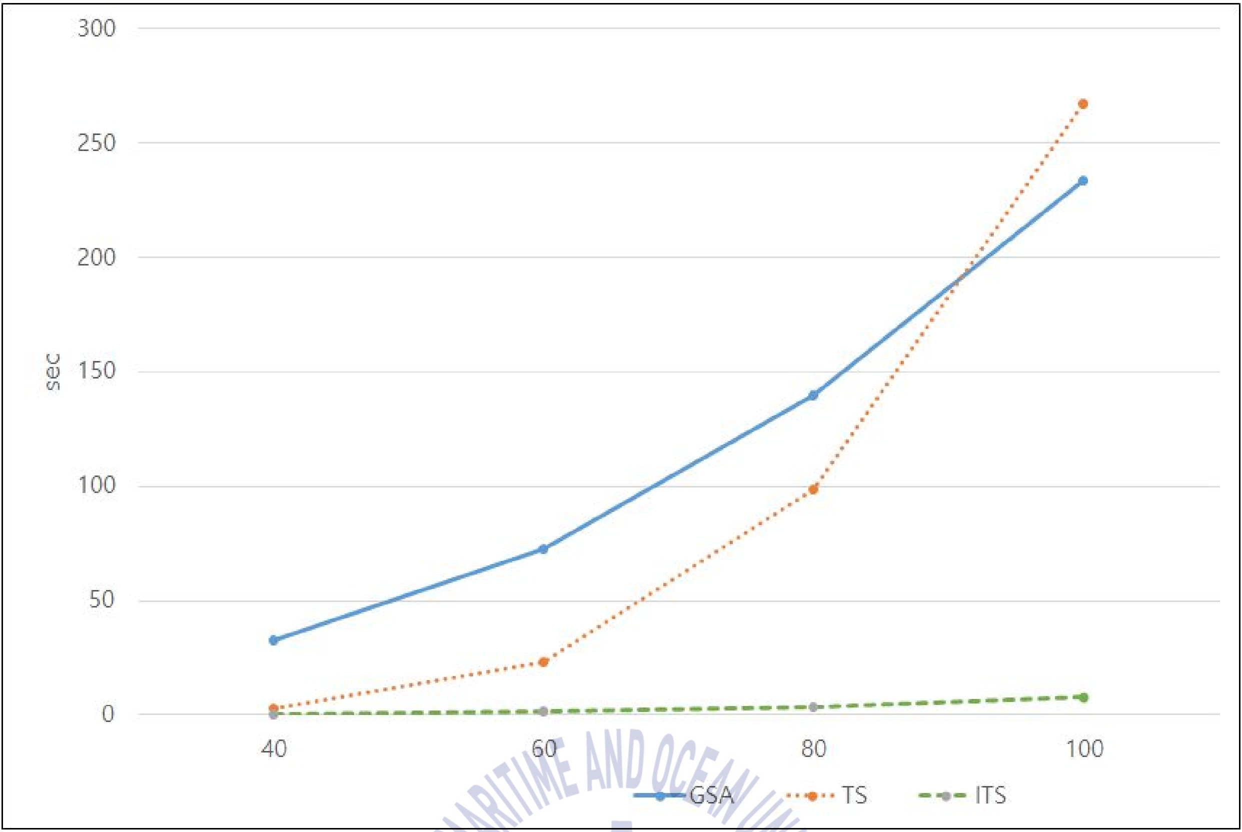

Fig. 5 The computing time of GSA, TS and ITS

From the computational results, we noticed that the proposed ITS outperforms the GSA and TS in terms of the computing time and solution quality. As shown in Fig. 5, ITS is the fastest among them. Specifically, when p is 100, GSA and TS take a considerable amount of computing time than the proposed ITS. As seen in Table 7, 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, exact algorithms such as branch-and-bound programming, obtain the global optimum, but their computational feasibility tend to diminish for $p > 40$. Not surprisingly, simple heuristic methods are usually used in SAS packages. They are forward selection, backward elimination, and step-wise regression. Their computing time is fast, but solution quality is not good.

In general, metaheuristic methods have been developed to provide better solution quality than simple heuristics. In the variable selection problem, two typical metaheuristics, which are tabu search [9] and hybrid GSA (genetic and simulated annealing algorithm) [10], was proposed. However, they have some shortcomings, that is, the tabu search [9] takes a lot of computing time due to many neighborhood moves and GSA's solution quality is less accurate.

To overcome their shortcomings, we proposed an improved tabu search to reduce moves of the neighborhood and to exploite the effective search strategy for the neighborhoods. To evaluate the performance of the proposed method, a comparative analysis was performed on both the literature data sets [10] and simulation data sets for larger size of variables. Computational results showed that the proposed method outperforms the previous metaheuristics in terms of the computing time and solution quality.

References

[1] Isabelle Guyon and Andre Elisseeff (2003), “An Introduction to Variable and Feature Selection”, Journal of machine Leaning Research, Vol. 3, pp. 1157-1182.

[2] Furnival, G. M. and Wilson, R.W. (1974), “Regression by Leaps and Bounds”, Technometrics, Vol. 16, No. 4, pp. 416-423.

[3] Duarte Silva, A.P.(2001), “Efficient Variable Screening for Multivariate Analysis”, Journal of Multivariate Analysis, Vol.76, pp. 35-62.

[4] Duarte Silva, A.P. (2002), “Discarding variables in a principal component analysis: algorithms for all-subsets comparisons”, Computational statistics, Vol.17 pp. 251-271.

[5] Gatu, C and Kontoghiorghes, E.J.(2006), “Branch-and-Bound Algorithms for Computing the Best-Subset Regression Models”, Journal of Computational and graphical Statistics, Vol.15 , pp.139-156

[6] Hofmann, M., Gatu, C and Kontoghiorghes, E.J.(2007), “Efficient algorithms for computing the best subset regression models for large-scale problems”, Computational statistics and data analysis, Vol. 52, pp. 16-29.

[7] Brusco, M.J., Steinley, D. and Cradit, J.D.(2009)“An exact algorithm for hierarchically well-formulated subsets in second-order polynomial regression”, Technometrics, Vol. 51, pp. 306-315.

[8] Pacheco, J., Casado, S. and Porras, S.(2013), “Exact methods for variable selection in principal component analysis: guide functions and pre-selection”,

Computational Statistics and data analysis, Vol. 57, pp. 95-111.

[9] Zvi Drezner and George A. Marcoulides(1999), “Tabu seach model selection in multiple regression analysis”, commun. statistics.-simula, Vol. 28, pp. 349-367

[10] H.Hasan orkcu (2013), “Subset selection in multiple linear regression models: A hybrid of genetic and simulated annealing algorithms”, Applied Mathematics and Computation, Vol. 219, pp. 11018-11028

[11] Draper, N. R, and smith, H. (1998), “Applied regression analysis(3th Edition)”, NewYork: Wiley

[12] Montgomery, D. G., and Peck, E. A.(1992), “Introduction to linear regression analysis(2nd Edition)”, NewYork: Wiley

[13] Glover, F. (1986), “Future Paths for Integer Programming and Links to Artificial Intelligence.” Computers and Operations Research, Vol. 13, pp. 533-549

[14] Glover, F. (1990), “Tabu Search: A Tutorial”, Kluwer Academic publishers

[15] Holland, J. H. (1975), “Adaptaion in Natural and Artificial Systems.”, University of Michigan Press

[16] Kirpatirck, S., Gelatt, C. D., and Vecchi, M. P. (1983), “Optimiztion by Simulated Annealing”, Science, Vol.220, 671-680

[17] F.T. Lin, F.T., Kao, C.Y. and Hsu, C.C (1993), “Applying the genetic approach to simulated annealing in solving some NP-hard problems”, IEEE Trans. Syst. Man Cybern, Vol.23, pp.1752-1767

[18] Widmer, M. and Hertz, A. (1989), “A new heuristic method for the flow shop sequencing problem”, European Journal of Operational Research., Vol.41, pp. 186-193

[19] Tailard, E. (1990), “Some efficient heuristic methods for the flow shop sequencing problem”, Department of Mathematics, Vol.47, pp.65-74

Appendix

A. The source code of Hybrid GSA

```
gsa<-function(size,popSize){

        mutation<-function(offspring){

                offspring<-ifelse(runif(size)<0.1,ifelse(offspring==1,0,1),offspring)

                return(offspring)

        }

        crossover<-function(p1,p2){

                n<-length(p1)

                point<-sample(2:n,1)

                offspring1<-c(p1[1:point-1],p2[(point):n])

                offspring2<-c(p2[1:point-1],p1[(point):n])

                return(rbind(offspring1,offspring2))

        }

        selection<-function(o,sumF){

                point<-runif(1,0,sumF)

                s<-0

                for(i  in  seq(o)){

                        s<-s+o[i]
        if(point<s)break
    }
    return(i)
}
replacement<-function(parent,offspring){
    diff<-offspring[size+1]-parent[size+1]
    if(diff>0 || (runif(1)<exp(diff/t))){
        return(offspring)
    }else{
        return(parent)
    }
}
pop<-matrix(-1,ncol=size+1,nrow=popSize)
for(i in 1:(size*popSize)){pop[i]<-sample(0:1,1)}
pop[,size+1]<-apply(pop[,1:size],1,adjr)
best<-rep(0,size+1)
t<-100
for(iter in 1:1000){
    b<-which.max(pop[,size+1])
    if(best[size+1]<pop[b,size+1]){best<-pop[b,]}
    sumF<-sum(pop[,size+1])
    pop1<-matrix(-1,ncol=size+1,nrow=popSize)
    for(i in seq(1,100,2)){
```

```
                    parent1<-pop[selection(pop[,size+1],sumF),]

                    parent2<-pop[selection(pop[,size+1],sumF),]

                    if(runif(1)>0.8){

                              pop1[i,]<-parent1

                              pop1[i+1,]<-parent2

                              next

                    }

                    offsprings<-crossover(parent1[1:size],parent2[1:size])

                    offspring1<-offsprings[1,]

                    offspring2<-offsprings[2,]

                    offspring1<-mutation(offspring1)

                    offspring2<-mutation(offspring2)

                    offspring1<-c(offspring1,adjr(offspring1))

                    offspring2<-c(offspring2,adjr(offspring2))

                    pop1[i,]<-replacement(parent1,offspring1)

                    pop1[i+1,]<-replacement(parent2,offspring2)

          }

     pop<-pop1

     t<-0.9*t

     }

     return(best)

}
```

B. The source code of TS

```
ts<-function(size,failsize,tabusize){

          ts.move<-function(s){

                    o<-setdiff(which(s==1),tabulist)

                    z<-setdiff(which(s==0),tabulist)

                    ol<-length(o)

                    zl<-length(z)

                    ns<-(ol+zl)+(ol*zl)

                    neigh<-matrix(rep(s,ns),nrow=ns,byrow=T)

                    p<-1

                    for(i  in  o){

                              neigh[p,i]<-0;p<-p+1

                    }

                    for(i  in  z){

                              neigh[p,i]<-1;p<-p+1

                    }

                    for(i  in  o){

                              for(j  in  z){

                                        neigh[p,i]<-0

                                        neigh[p,j]<-1

                                        p<-p+1

                              }
          }
          cost<-apply(neigh,1,adjr)
          w<-which.max(cost)
          return(c(neigh[w,],cost[w]))
}
current<-vector(length=size)
for(i  in  1:size){
          current[i]<-sample(0:1,1)
}
current<-c(current,adjr(current))
best<-current
tabulist<-NULL
fn<-0
while(fn<failsize){
          s<-ts.move(current[1:size])
          dc<-which(c(current[1:size]!=s[1:size]))
          current<-s
          tabulist<-c(tabulist,dc)
          if(length(tabulist)>tabusize){
                    if(length(tabulist)==tabusize+2){
                              tabulist<-tabulist[3:(tabusize+2)]
                    }else{
                              tabulist<-tabulist[2:(tabusize+1)]
                    }
                }
                if(best[size+1]<current[size+1]){
                    best<-current
                    fn<-0
                }else{
                    fn<-fn+1
                }
            }
            return(best)
        }
```

C. The source code of ITS

```
its<-function(size,failsize){
          its.move<-function(sol,i){
                    sol[i]<-ifelse(sol[i]==1,0,1)
                    return(c(sol,adjr(sol)))
          }
          tabu<-function(sol,tabuList){
                    l<-length(sol)
                    sl<-length(which(sol[-l]==1))
                    ob<-sol[l]
                    for(i in seq(tabuList)){
                              if(tabuList[[i]][1]==ob && tabuList[[i]][2]==sl)
                                        return(TRUE)
                    }
                    return(FALSE)
          }
          fn<-0
          tabuList<-matrix(-1,ncol=size+1,nrow=size)
          current<-vector(length=size)
          for(i in 1:size){
                    current[i]<-sample(0:1,1)
                    }
```

```
best<-c(current,adjr(current))
tabuList<-list()
while(fn<failsize){
        or<-sample(1:size,size)
        nb<-rep(0,size+1)
        for(i in or){
                neigh<-its.move(current[1:size],i)
                if(tabu(neigh,tabuList))
                        next
                if(neigh[size+1]>best[size+1]){
                        nb<-neigh
                        break
                }else if(neigh[size+1]>nb[size+1]){
                        nb<-neigh
                }
        }
        current<-nb
  tabuList[[length(tabuList)+1]]<-c(current[size+1],length(which(current[-(size+1)]==1)))
        if(length(tabuList)>10){
                tabuList[[1]]<-NULL
        }
        if(best[size+1]<current[size+1]){
                best<-current
        }else{
                fn<-fn+1
        }
}
return(best)
```

D. The source code of adjusted R^2

```
identity<-function(n)

{

        I<-matrix(0,n,n)

        for(i in 1:n)

                I[i,i]<-1

        return(I)

}

setSST<-function()

{

        J<-(1/nr)*matrix(1,nr,nr)

        sst<-t(Y)%*%(I-J)%*%Y

        return(sst)

}

adjr2<-function(th)

{

        if (sum(th) == 0)return(0)

        x<-cbind(1,X[,th==1])

        k<-ncol(x)

        xt<-t(x)

        nomal<-solve(xt%*%x)

        b<-t((nomal%*%xt)%*%Y)
        yy<-t(Y)%*%Y

        bxy<-(b%*%xt)%*%Y

        SSE<-(yy-bxy)

        r<-(SSE/SST)

        c<-(nr-1)/(nr-k)

        adjr1<-  1-(c*r)

        return(adjr1)

}
```

E. The source code of generating simulation data

```
generate<-function(p,n,r,a){
        d<-matrix(0,nrow=n,ncol=p)
        for(i in 1:p){d[,i]<-rnorm(n)}
        s<-sample(1:p,r*p)
        for(i in s){cat(i,",",sep="")}
        cat("\n")
        print(length(s))
        y<-apply(d[,s],1,sum)
        e<-rnorm(n,0,sd(y)*a)
        y<-y-e
        return(data.frame(y,d))
}
```

감사의 글

설렘과 꿈을 안고 정문을 들어섰던 게 엊그제 같은데, 어느 새 2년의 석사과정을 마치고 학위 논물을 제출하게 되었습니다. 지난 2년의 시간동안 저에게 도움을 주신 분들이 많습니다. 미흡하지만 학위 논문을 마치면서 그분들께 감사의 말을 전합니다.

먼저 학부와 대학원 생활을 하는 동안 누구보다 많은 도움을 주시고, 부족한 점을 언제나 세심하고 꼼꼼한 손길로 지적해주신 배재국 교수님께 진심으로 감사드립니다. 바쁘신 와중에도 제 연구들의 대부분을 공동으로 지도해 주시고 따뜻한 격려와 조언을 해주신 김재환 교수님께 감사드립니다. 또한 논문 심사를 맡아주신 장길웅 교수님과 자상한 가르침으로 학문의 길에서 인도해 주신 김익성 교수님, 홍정희 교수님, 박찬근 교수님, 그리고 손미정 교수님께도 감사드립니다. 그리고 항상 챙겨주시고 많은 도움을 주신 정지영 조교님께 감사의 말씀을 전합니다. 2년 동안 같은 연구실 생활을 한 연구실 식구들에게도 고맙다는 말을 전합니다.

마지막으로 한없이 부족한 저에게 인간됨의 진리를 가르쳐주신 아버지, 철없는 아들을 사랑과 헌신으로 키워주신 어머니께 가장 큰 감사의 마음 전합니다.