An Information Based Optimal Subdata Selection Algorithm for Big Data Linear Regression and a Suitable Variable Selection Algorithm

by

Yi Zheng

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved April 2017 by the
Graduate Supervisory Committee:

John Stufken, Chair
Mark Reiser
Robert McCulloch

ARIZONA STATE UNIVERSITY

May 2017

ABSTRACT

This article proposes a new information-based subdata selection (IBOSS) algorithm, Squared Scaled Distance Algorithm (SSDA). It is based on the invariance of the determinant of the information matrix under orthogonal transformations, especially rotations. Extensive simulation results show that the new IBOSS algorithm retains nice asymptotic properties of IBOSS and gives a larger determinant of the subdata information matrix. It has the same order of time complexity as the D-optimal IBOSS algorithm. However, it exploits the advantages of vectorized calculation avoiding for loops and is approximately 6 times as fast as the D-optimal IBOSS algorithm in R. The robustness of SSDA is studied from three aspects: nonorthogonality, including interaction terms and variable misspecification. A new accurate variable selection algorithm is proposed to help the implementation of IBOSS algorithms when a large number of variables are present with sparse important variables among them. Aggregating random subsample results, this variable selection algorithm is much more accurate than the LASSO method using full data. Since the time complexity is associated with the number of variables only, it is also very computationally efficient if the number of variables is fixed as n increases and not massively large. More importantly, using subsamples it solves the problem that full data cannot be stored in the memory when a data set is too large.

TABLE OF CONTENTS

List of Tables

List of Figures

Figure Page

Chapter 1

Introduction

1.1 Background

As data grows larger and larger due to advances of technologies, a new challenge is how to analyze these big data. For high-dimensional big data where $p \gg n$, multiple methods have been proposed and studied, such as the LASSO (Tibshirani (1996), Meinshausen and Yu (2009)), Dantzig selector (Candes and Tao (2007)), and sure independence screening (Fan and Lv (2008)), among others. The focus of this paper is on situations where data are massive in data size n but not massive in number of variables p ($n \gg p$) and linear regression is used as the model. One challenge is that when n is too large, the size of data could exceed computer memory. Direct analysis of these data using ordinary least-squares (OLS) is not applicable. Also, with time complexity $O(np^2)$ for OLS, the computing times are intimidating when n is too large. Limited by computational resources, taking a subsample from the full data and estimating parameters based on the subsample can be a solution.

Several subsampling methods have been proposed including leveraging sampling methods (Drineas *et al.* (2006), Drineas *et al.* (2011), Ma *et al.* (2014), Ma and Sun (2015), Ma *et al.* (2015)) and information-based optimal subdata selection (IBOSS) (Wang *et al.* (2017)). This paper is based on the ideas from IBOSS. Compared to other subsampling methods, IBOSS has several advantages. First of all, the time complexity of existing IBOSS algorithms is $O(np)$, which is considerably faster than other subsampling methods. Secondly, IBOSS algorithms are suitable for parallel computing, which could improve the computation time of IBOSS algorithms even more. Thirdly, the variances of parameter estimators from IBOSS subdata have asymptotic properties as full data size n increases even though the subdata size k is fixed.

1.2 IBOSS Framework

In Wang *et al.* (2017), they propose the IBOSS framework, which will be restated in this section for a better understanding of proposed algorithms in this paper.

Assume the full data is $(\mathbf{y}, \mathbf{Z})$, where $\mathbf{y} = (y_1, y_2, \ldots, y_n)^T$ is the vector of responses, $\mathbf{Z} = (\mathbf{z}_1^T, \mathbf{z}_2^T, \ldots, \mathbf{z}_n^T)^T$ is the covariate matrix with p by 1 vectors $\mathbf{z}_i = (z_{i1}, z_{i2}, \ldots, z_{ip})^T, i = 1, \ldots, n$. Also, the linear model is used:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} = \beta_0 \boldsymbol{j}_n + \mathbf{Z}\boldsymbol{\beta}_1 + \boldsymbol{\varepsilon}, \tag{1.1}$$

where $\boldsymbol{j}_n$ is a vector of ones with length n, $\mathbf{X} = (\boldsymbol{j}_n, \mathbf{Z})$ is the model matrix, $\boldsymbol{\beta} = (\beta_0, \boldsymbol{\beta}_1)^T = (\beta_0, \beta_1, \beta_2, \ldots, \beta_p)^T$ is a $p+1$ by 1 vector of the unknown parameters with β_0 as the intercept parameter and $\boldsymbol{\beta}_1$ as the p-dimensional vector of slope parameters and $\boldsymbol{\varepsilon} = (\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n)^T$ is the vector of error terms with $E(\boldsymbol{\varepsilon}) = \mathbf{0}$ and $cov(\boldsymbol{\varepsilon}) = \sigma^2 \boldsymbol{I}_n$

Under model 1.1, the OLS estimator of $\boldsymbol{\beta}$ is given by

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y},$$

which is also the best linear unbiased estimator (BLUE) of $\boldsymbol{\beta}$. The covariance matrix of this BLUE can be easily found as:

$$cov(\hat{\boldsymbol{\beta}}) = \sigma^2(\mathbf{X}^T\mathbf{X})^{-1}.$$

If ε is normally distributed, the observed fisher information matrix is the inverse of $cov(\hat{\boldsymbol{\beta}})$, which is:

$$\mathbf{M} = \frac{1}{\sigma^2}\mathbf{X}^T\mathbf{X} = \frac{1}{\sigma^2}\sum_{i=1}^{n}\mathbf{x}_i\mathbf{x}_i^T.$$

For simplicity we will call it information matrix for the rest of this paper.

In IBOSS algorithms, subdata are selected deterministically from the full data. Assume the subdata selected from full data above are $(\mathbf{y}^*, \mathbf{Z}^*)$, where $\mathbf{y}^* = (y_1^*, y_2^*, \ldots, y_k^*)^T$ is the vector of responses, $\mathbf{Z}^* = (\mathbf{z}_1^{*T}, \mathbf{z}_2^{*T}, \ldots, \mathbf{z}_k^{*T})^T$ is the covariate matrix for subdata with p by 1 vectors $\mathbf{z}_i^* = (z_{i1}^*, z_{i2}^*, \ldots, z_{ip}^*)^T, i = 1, \ldots, k$. Then $(\mathbf{y}^*, \mathbf{Z}^*) = f(\mathbf{Z})$ meaning that whether or not a point is selected depends on the covariate matrix $\mathbf{Z}$ only. The subdata also follows linear model:

$$\mathbf{y}^* = \mathbf{X}^*\boldsymbol{\beta} + \boldsymbol{\varepsilon}^* = \beta_0\boldsymbol{j}_k + \mathbf{Z}^*\boldsymbol{\beta}_1 + \boldsymbol{\varepsilon}^*, \tag{1.2}$$

where $\boldsymbol{j}_k$ is a vector of ones with length k, $\mathbf{X}^* = (\boldsymbol{j}_k, \mathbf{Z}^*)$ is the model matrix for subdata, $\boldsymbol{\beta}$ is the same vector of unknown parameters as in model 1.1 and $\boldsymbol{\varepsilon}^* = (\varepsilon_1^*, \varepsilon_2^*, \ldots, \varepsilon_k^*)^T$ is the vector of error terms with $E(\boldsymbol{\varepsilon}^*) = \mathbf{0}$ and $cov(\boldsymbol{\varepsilon}^*) = \sigma^2\boldsymbol{I}_k$

The covariance of OLS estimator (BLUE) and information matrix under model 1.2 are given by similar reasonings as model 1.1:

$$cov(\hat{\boldsymbol{\beta}}^*) = \sigma^2(\mathbf{X}^{*T}\mathbf{X}^*)^{-1} \tag{1.3}$$

$$\mathbf{M}_s = cov(\hat{\boldsymbol{\beta}}^*)^{-1} = \frac{1}{\sigma^2}\mathbf{X}^{*T}\mathbf{X}^* = \frac{1}{\sigma^2}\sum_{i=1}^{n}\mathbf{x}_i^*\mathbf{x}_i^{*T}. \tag{1.4}$$

Among all unbiased estimators of $\boldsymbol{\beta}$ based on the subdata, OLS estimator gives the smallest variance thus is the BLUE. However, as shown in equation 1.3, the covariance matrix of this BLUE depends on how the subdata is selected. In IBOSS algorithms, we aim to select subdata which optimize an objective function. In an optimization problem, the information matrix of subdata can be written as follows:

$$\mathbf{M}(\boldsymbol{\delta}) = \frac{1}{\sigma^2}\sum_{i=1}^{n}\delta_i\mathbf{x}_i\mathbf{x}_i^T,$$

where $\boldsymbol{\delta} = (\delta_1, \delta_2, \ldots, \delta_n)$, $\delta_i = 1$ if point i is selected in the subdata and $\delta_i = 0$ if point i is not selected in the subdata, $i = 1, 2, \ldots, n$, $\sum_{i=1}^{n} \delta_i = k$. Suppose ψ is an optimality criterion function. The optimization problem of selecting subdata becomes:

$$\boldsymbol{\delta}^{opt} = \arg\max_{\boldsymbol{\delta}} \psi\{\mathbf{M}(\boldsymbol{\delta})\}, \text{ subject to} \sum_{i=i}^{n} \delta_i = k.$$

1.3 *D*-Optimal IBOSS

Under linear regression model, the D-optimal IBOSS uses determinant as optimality criterion function and aims to select the subdata that maximize $|\mathbf{M}(\boldsymbol{\delta})|$. Although exact solution for this optimization problem is not feasible, inspired by the upper bound

$$|\mathbf{M}(\boldsymbol{\delta})| \leq \frac{k^{p+1}}{4^p} \prod_{j=1}^{p} (z_{(n)j} - z_{(1)j}), \tag{1.5}$$

Wang *et al.* (2017) develops the D-optimal IBOSS algorithm by selecting points with extreme covariate values, which gives a good approximation of $\max_{\boldsymbol{\delta}} |\mathbf{M}(\boldsymbol{\delta})|$. Suppose $r = k/2p$ is an integer and for each variable, it selects r points with the r largest covariate values and r points with the r smallest covariate values. Parameters are estimated by $\hat{\boldsymbol{\beta}}^D = (\mathbf{X}_D^T \mathbf{X}_D)^{-1} \mathbf{X}_D^T \mathbf{y}_D$. To get a better estimation of the intercept, $\hat{\beta}_0^D$ is calculated as follows:

$$\hat{\beta}_0^D = \bar{y} - \bar{\mathbf{z}}^T \hat{\boldsymbol{\beta}}_1^D. \tag{1.6}$$

D-optimal IBOSS algorithm enjoys many nice properties and the most important two of them are $O(np)$ time complexity and the asymptotic properties. As the full data size n grows large with fixed subdata size k, $|\mathbf{M_D}(\boldsymbol{\delta})|$ increases as the same order as the upper bound of $|\mathbf{M}(\boldsymbol{\delta})|$. Also the rates of convergence of variances of slope estimators are

$$V(\beta_j^D|\mathbf{Z}) \asymp_P 1/(z_{(n)j} - z_{(1)j})^2, j = 1, \ldots, p. \tag{1.7}$$

The estimator $\hat{\beta}_0^D$ has a similar convergence rate to that of slope estimators. This is because $\hat{\beta}_0^D - \beta_0 = (\hat{\beta}_0^{full} - \beta_0) + \bar{\mathbf{z}}^T(\hat{\boldsymbol{\beta}}_1^{full} - \boldsymbol{\beta}_1) - \bar{\mathbf{z}}^T(\hat{\boldsymbol{\beta}}_1^D - \boldsymbol{\beta}_1)$ and the last term is the dominating term if $\mathbf{E}(\mathbf{z}) \neq \mathbf{0}$. If $\mathbf{E}(\mathbf{z}) = \mathbf{0}$, the convergence may be faster.

1.4 Improvement of *D*-Optimal IBOSS

In this paper, a new IBOSS algorithm, Squared Scaled Distance Algorithm (SSDA), is developed based on D-optimal IBOSS algorithm and a suitable variable selection algorithm is proposed to deal with the variable selection problem before using IBOSS algorithms. In Chapter 2, Squared Scaled Distance Algorithm is presented together with the time complexity analysis, simulation study and robustness study. D-optimal IBOSS Algorithm is compared with SSDA in all these aspects. In Chapter 3, an accurate variable selection algorithm is proposed and compared with the lasso on full data. Conclusions are offered in Chapter 4.

Chapter 2

Squared Scaled Distance Algorithm

2.1 Motivation

The core of D-optimal IBOSS algorithm is selecting extreme values of variables to approximate the upper bound of $|\mathbf{M}(\boldsymbol{\delta})|$. Well performed as it is, it only considers one variable at a time. The squared scaled distance algorithm (SSDA) proposed in this chapter will consider all variables simultaneously and share the same nice properties as those of D-optimal IBOSS. It is motivated by the invariance of $|\mathbf{X}^T\mathbf{X}|$ under rotation which will be explained by Theorem 1.

Suppose we have a p by p rotation matrix $\mathbf{R}$ with property $\mathbf{R}^T = \mathbf{R}^{-1}$. Our original model matrix is an n by $(p+1)$ matrix: $\mathbf{X} = (\boldsymbol{j}, \mathbf{Z})$. After rotating each data , we can obtain a new model matrix $\mathbf{X}_1 = (\boldsymbol{j}_n, \mathbf{Z}\mathbf{R}^T)$

Theorem 1 (Invariance of the Determinants of Information Matrices Under Rotation)**.** *The determinant of the original information matrix is the same as that of the information matrix after rotation transformation. This is equivalent to* $|\mathbf{X}_1^T\mathbf{X}_1| = |\mathbf{X}^T\mathbf{X}|$*. The invariance also carries to* $|\mathbf{M}(\boldsymbol{\delta})|$ *meaning* $|\mathbf{M}(\boldsymbol{\delta})| = |\mathbf{M_1}(\boldsymbol{\delta})|$*, where* $|\mathbf{M_1}(\boldsymbol{\delta})|$ *is the information matrix of subdata under the rotated coordinates.*

Proof. Let's suppose we have an n by p covariate matrix

$$\mathbf{Z} = \begin{pmatrix} \mathbf{z}_1^T \\ \mathbf{z}_2^T \\ \vdots \\ \mathbf{z}_n^T \end{pmatrix}$$

and a p by p rotation matrix $\mathbf{R}$ with property $\mathbf{R}^T = \mathbf{R}^{-1}$. The model matrix is $\mathbf{X} = (\boldsymbol{j}_n, \mathbf{Z})$. After rotation, data entry $\mathbf{z}_i, i = 1, \ldots, n$ becomes $\mathbf{R}\mathbf{z}_i, i = 1, \ldots, n$ respectively. Therefore, the new covariate matrix is

$$\mathbf{Z}_R = \begin{pmatrix} (\mathbf{R}\mathbf{z}_1)^T \\ (\mathbf{R}\mathbf{z}_2)^T \\ \vdots \\ (\mathbf{R}\mathbf{z}_n)^T \end{pmatrix} = \mathbf{Z}\mathbf{R}^T$$

and the new model matrix is $\mathbf{X}_1 = (\boldsymbol{j}_n, \mathbf{Z}\mathbf{R}^T)$.

The information matrix under original coordinates and its determinant are:

$$\mathbf{X}^T\mathbf{X} = (\boldsymbol{j}_n, \mathbf{Z})^T(\boldsymbol{j}_n, \mathbf{Z}) = \begin{pmatrix} n & \boldsymbol{j}_n^T\mathbf{Z} \\ \mathbf{Z}^T\boldsymbol{j}_n & \mathbf{Z}^T\mathbf{Z} \end{pmatrix}$$

Since n is invertible, $|\mathbf{X}^T\mathbf{X}| = |n||\mathbf{Z}^T\mathbf{Z} - \mathbf{Z}^T\boldsymbol{j}_n\boldsymbol{j}_n^T\mathbf{Z}/n| = |\mathbf{Z}^T(n\boldsymbol{I}_n - \mathbf{J}_n)\mathbf{Z}|$, where $\mathbf{J}_n = \boldsymbol{j}_n\boldsymbol{j}_n^T$.

The information matrix under rotated coordinates and its determinant are:

$$\mathbf{X}_1^T\mathbf{X}_1 = (\boldsymbol{j}_n, \mathbf{Z}\mathbf{R}^T)^T(\boldsymbol{j}_n, \mathbf{Z}\mathbf{R}^T) = \begin{pmatrix} n & \boldsymbol{j}_n^T\mathbf{Z}\mathbf{R}^T \\ \mathbf{R}\mathbf{Z}^T\boldsymbol{j}_n & \mathbf{R}\mathbf{Z}^T\mathbf{Z}\mathbf{R}^T \end{pmatrix}$$

$|\mathbf{X}_1^T\mathbf{X}_1| = |n||\mathbf{R}\mathbf{Z}^T\mathbf{Z}\mathbf{R}^T - \mathbf{R}\mathbf{Z}^T\boldsymbol{j}_n\boldsymbol{j}_n^T\mathbf{Z}\mathbf{R}^T/n| = |\mathbf{R}\mathbf{Z}^T(n\boldsymbol{I}_n - \mathbf{J}_n)\mathbf{Z}\mathbf{R}^T| = |\mathbf{R}||\mathbf{Z}^T(n\boldsymbol{I}_n - \mathbf{J}_n)\mathbf{Z}||\mathbf{R}^T| = |\mathbf{Z}^T(n\boldsymbol{I}_n - \mathbf{J}_n)\mathbf{Z}| = |\mathbf{X}^T\mathbf{X}|$, where $|\mathbf{R}^T| = |\mathbf{R}^{-1}| = 1/|\mathbf{R}|$

Therefore, $|\mathbf{X}_1^T\mathbf{X}_1| = |\mathbf{X}^T\mathbf{X}|$. With similar reasoning we can prove $|\mathbf{M}(\boldsymbol{\delta})| = |\mathbf{M}_1(\boldsymbol{\delta})|$. □

Under this rotated coordinate system, we can perform D-optimal IBOSS algorithm to the full data, which is selecting those points with extreme values in each rotated coordinate. These extreme values will also make contributions to the approximation of the upper bound of $|\mathbf{M}(\boldsymbol{\delta})|$. Since there are infinite number of rotations, we can generalize D-optimal IBOSS under all these rotations to a method that selects points with the largest scaled distances to center. In this way, more points with influential effect on $|\mathbf{M}(\boldsymbol{\delta})|$ will be included in our subdata. Based on this motivation, we create a new IBOSS algorithm which will be introduced in Section 2.2.

Remark 1. *The invariance of the determinants of information matrices is not limited to rotation transformation. Actually, $|\mathbf{X}^T\mathbf{X}|$ is invariant to any orthogonal transformation. We emphasize on rotation because it is easier to be interpreted geometrically.*

2.2 Algorithm

Algorithm 1 (Squared Scaled Distance Algorithm)**.** *Suppose that the covariate matrix is an n by p matrix $\mathbf{Z}$ and we would like to select subdata with size k from the original data by performing the following steps:*

Step 1 *Find the center of original data. Here sample mean is used to represent the center. $\bar{\mathbf{z}} = (\bar{z}_1, \bar{z}_2, \ldots, \bar{z}_p)^T$, where $\bar{z}_i = \sum_{l=1}^{n} z_l/n,\ i = 1, \ldots, n$.*

Step 2 *Calculate the sample variances for each variable:*

$$S_j = \sqrt{\sum_{l=1}^{n}(z_{lj} - \bar{z}_j)^2/(n-1)},\ j = 1, \ldots, p$$

Step 3 *For each $\mathbf{z}_i,\ i = 1, \ldots, n$, calculate its squared scaled euclidean distance to the center:*

$$D_i^2 = \sum_{j=1}^{p}\left(\frac{z_{ij} - \bar{z}_j}{S_j}\right)^2,\ i = 1, \ldots, n$$

Step 4 *Use quick-select algorithm to select the k points with largest square scaled euclidean distances to the center as our subdata $\mathbf{Z}^*$.*

Step 5 *Calculate $\hat{\boldsymbol{\beta}}^* = (\mathbf{X}^{*T}\mathbf{X}^*)^{-1}\mathbf{X}^{*T}\mathbf{y}^*$, where $\mathbf{X}^* = (\boldsymbol{j}_k,\ \mathbf{Z}^*)$ and $\hat{\boldsymbol{\beta}}^* = (\hat{\beta}_0^*,\ \hat{\boldsymbol{\beta}}_1^*)$. Replace $\hat{\beta}_0^*$ with $\hat{\beta}_0^{**} = \bar{y} - \bar{\mathbf{z}}^T\hat{\boldsymbol{\beta}}_1^*$.*

Remark 2. *The reason we use scaled distances instead of unscaled ones is that we want to make each variable dimensionless and eliminate the scale effects on the distances. To reduce computation time, square roots are not taken to these squared scaled distances.*

Remark 3. *In real world problem we don't know the correlation between variables, therefore we assume they are uncorrelated to each other and use the squared scaled euclidean distance under orthogonal (Cartesian) coordinates. In section 2.5.1, we will discuss the robustness of SSDA when there are correlations among variables.*

Remark 4. *There are many choices for defining data center. Sample mean is used here because it is simple to calculate. Other alternatives such as various concepts of data depth are so time expensive that they have the same time complexity as using full data.*

Remark 5. *We don't use statistical distance $(\mathbf{x}-\bar{\mathbf{x}})^{\mathbf{T}}\hat{\boldsymbol{\Sigma}}^{\mathbf{-1}}(\mathbf{x}-\bar{\mathbf{x}})$ because the time complexity of estimating $\boldsymbol{\Sigma}^{\mathbf{-1}}$ is $O(np^2)$, which is no better than using the full data with OLS. Instead, sample variance of each variable is used to scale the euclidean distances. The time complexity is $O(np)$ for calculating all p sample variances.*

Remark 6. *Using $\hat{\beta}_0^{**}$ instead of $\hat{\beta}_0^*$ endows the estimation of the intercept with a good asymptotic property without increasing the order of time complexity of the algorithm.*

Remark 7. *More informative points are included in the subdata Using SSDA. Therefore we expect a larger information matrix of SSDA subdata than that of D-optimal*

IBOSS subdata.

2.3 Time Complexity and Advantages

The time complexity analysis of SSDA is as follows:

The time to calculate $\bar{\mathbf{z}}^T = (\bar{z}_1, \bar{z}_2, \ldots, \bar{z}_p)$ is $O(np)$. Calculating the squared scaled euclidean distances has the time complexity of $O(4np)$. The average time complexity of quick-select algorithm is $O(n)$. Time complexities to get $\hat{\boldsymbol{\beta}}^*$ and $\hat{\beta}_0^{**}$ are $O(kp^2+p^3)$ and $O(np)$ respectively. Thus the time complexity of SSDA is $O(6np+n+kp^2+p^3)$. When kp is less than or in the same order as n, The time complexity of algorithm 1 is $O(np)$.

As we can see, SSDA has the same level of time complexity as the D-optimal IBOSS does. Meanwhile it enjoys the following advantages:

a It exploits the advantages of vectorized calculation. Since all the calculations in SSDA are based on vector, it can be really fast in languages such as R and Python, which take advantages of vectorized calculation. On the other hand, the D-optimal IBOSS inevitably uses loops to search in each variable for extreme values, causing the running time much slower than SSDA in those languages.

b It suffers much less from the instability of quick-select algorithm. Just as the quick-sort algorithm, quick-select is also an unstable algorithm with the worst case time complexity of $O(n^2)$. Quick-select algorithm is used p times in the D-optimal IBOSS but only once in SSDA. Therefore, the distribution of computation time of D-optimal IBOSS has the same mean but a much heavier tail than that of SSDA. In an extreme case, if one of p quick-select processes has the worst time complexity $O(n^2)$, then time complexity of the whole D-optimal IBOSS algorithm will become $O(n^2)$. The worst case may happen to SSDA. But using quick-select algorithm only once, SSDA has a much smaller probability to be slow than the D-optimal IBOSS.

2.4 Simulation Study and Comparison to $\boldsymbol{D}$-optimal IBOSS

In this section, simulations focus on two aspects of SSDA: asymptotic property and computation time.

2.4.1 Simulation on Asymptotic Properties of Parameter Estimation, Prediction Error and $|\mathbf{M}(\delta)|$

Including more informative points in the subdata, SSDA should result in a larger determinant of information matrix compared to that of D-optimal IBOSS. Meanwhile, applying D-optimal IBOSS to rotated coordinate systems, we expect SSDA to preserve the same asymptotic properties of estimation, prediction error and $|\mathbf{M}(\delta)|$ as D-optimal IBOSS. We will find out these properties by simulations with following settings:

Data The full data sizes are $n = 5 \times 10^3, 1 \times 10^4, 1 \times 10^5, 1 \times 10^6$ with 50 variables ($p = 50$). The subdata size is fixed $k = 1000$.

The covariance structure used here is a mutually moderately correlated covariance structure with $\boldsymbol{\Sigma}_{ij} = 0.5^{I(i \neq j)}$, where $i, j = 1, \ldots, p$ and $I(i \neq j) = 1$ if $i \neq j$ and 0 otherwise. We generate the covariate matrices $\mathbf{Z}$'s according to the following cases: for each entry $\mathbf{z}_i,\ i = 1, \ldots, n$

Case 1 $\mathbf{z}_i \sim N(\mathbf{0}, \boldsymbol{\Sigma})$ has a multivariate normal distribution.

Case 2 $\mathbf{z}_i \sim LN(\mathbf{0}, \boldsymbol{\Sigma})$, has a multivariate lognormal distribution.

Case 3 $\mathbf{z}_i \sim t_2(\mathbf{0}, \boldsymbol{\Sigma})$, has a multivariate t distribution with degrees of freedom $\upsilon = 2$.

Case 4 $\mathbf{z}_i$ has a mixture distribution of five different distributions, $N(\mathbf{1}, \mathbf{\Sigma})$, $t_2(\mathbf{1}, \mathbf{\Sigma})$, $t_3(\mathbf{1}, \mathbf{\Sigma})$, $U[\mathbf{0}, \mathbf{2}]$, $LN(\mathbf{0}, \mathbf{\Sigma})$ with equal proportions of variables. Where $U[\mathbf{0}, \mathbf{2}]$ means its component are independent uniform distributions between 0 and 2.

A test data set with $n_{test} = 5 \times 10^3$ are created for calculating the prediction errors.

Model The following linear model is used: $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$, where $\mathbf{X} = (\boldsymbol{j}_n, \mathbf{Z})$, $\boldsymbol{\beta}$ is a 51 by 1 vector of ones and $\varepsilon_i \overset{i.i.d}{\sim} N(0, \sigma^2)$, $i = 1, \ldots, n$, $\sigma^2 = 9$.

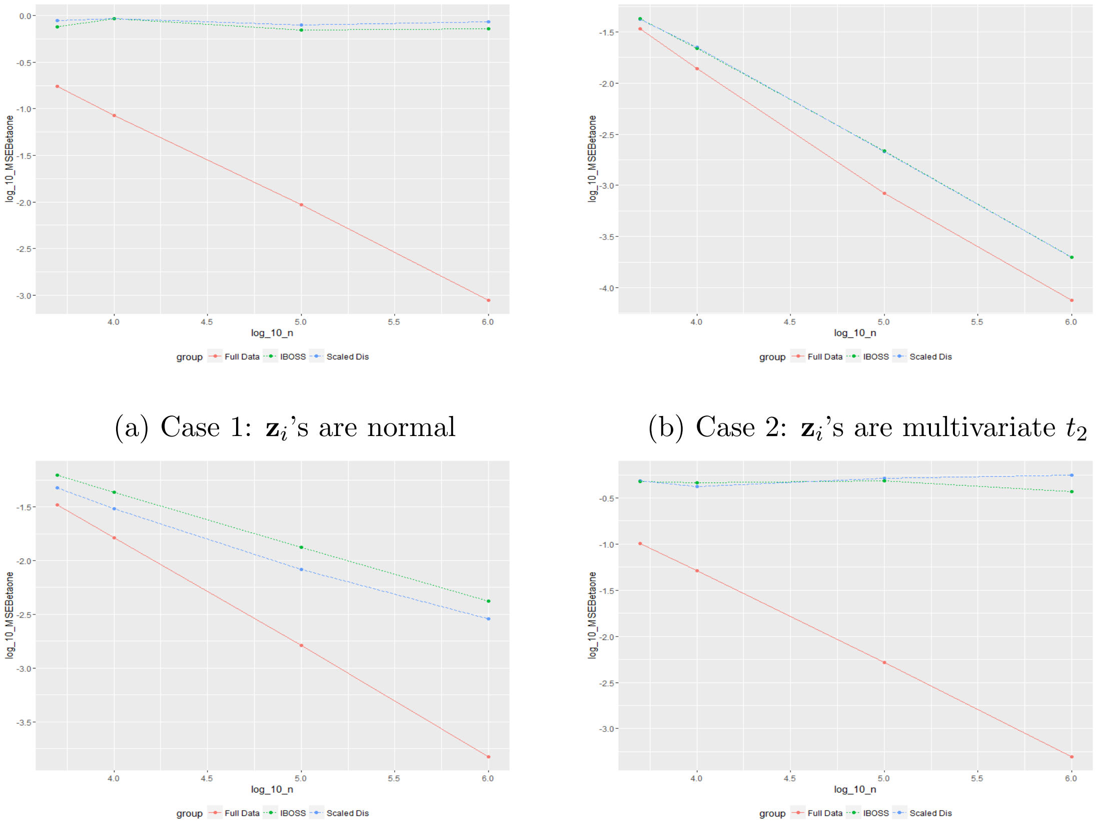
Simulation The simulation is repeated $S = 100$ times and the MSE's are calculated using $\text{MSE}_{\beta_0} = S^{-1} \sum_{s=1}^{S} (\hat{\beta}_0^{*(s)} - \beta_0)^2$ and $\text{MSE}_{\boldsymbol{\beta}_1} = S^{-1} \sum_{s=1}^{S} ||\hat{\boldsymbol{\beta}}_1^{*(s)} - \boldsymbol{\beta}_1||^2$. For prediction errors, we use mean squared prediction error (MSPE), $\text{MSPE} = E[\{E(y_{new}) - \hat{y}_{new}\}^2] = E[\{\mathbf{x}_{new}^T(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})\}^2]$. Also, $|\mathbf{M}(\boldsymbol{\delta})|$, determinants of the selected information matrices, are calculated. The means of MSPEs and $|\mathbf{M}(\boldsymbol{\delta})|$s over S simulations are calculated for plotting purpose. Three approaches including D-optimal IBOSS, SSDA and Full Data OLS are compared using the same full data set and response variable.

Graphics In each case, we plot $\log_{10}\text{MSE}_{\beta_0}$, $\log_{10}\text{MSE}_{\boldsymbol{\beta}_1}$, $\log_{10}\text{MSPE}$ and $\log_{10}|\mathbf{M}(\boldsymbol{\delta})|$ against $\log_{10}n$ with respect to the three approaches.

The simulation results are as follows:

Estimators

Figure 2.1 suggests that SSDA retains the asymptotic property for $\hat{\boldsymbol{\beta}}_1$ of D-optimal IBOSS. As shown in Figure 2.1(a), when $\mathbf{z}_i$'s are normally distributed, the decreases of MSEs for D-optimal IBOSS as well as those for SSDA are not as significant as that of MSEs for Full Data. This is because the rates of convergence of variances are only $1/log(n)$. In Figure 2.1(d), the rates of convergence of SSDA and D-optimal IBOSS are bounded by the rates of convergence of the slowest covariates, uniform distribution which dose not converge at all according to equation 1.7. Figure 2.1(b) and (c) show drastic decreases of MSEs as the full sample size n increases. When the distribution is multivariate t_2 , SSDA and D-optimal IBOSS have almost the same results. While in Figure 2.1(c), SSDA performs better than D-optimal IBOSS.



(a) Case 1: $\mathbf{z}_i$'s are normal

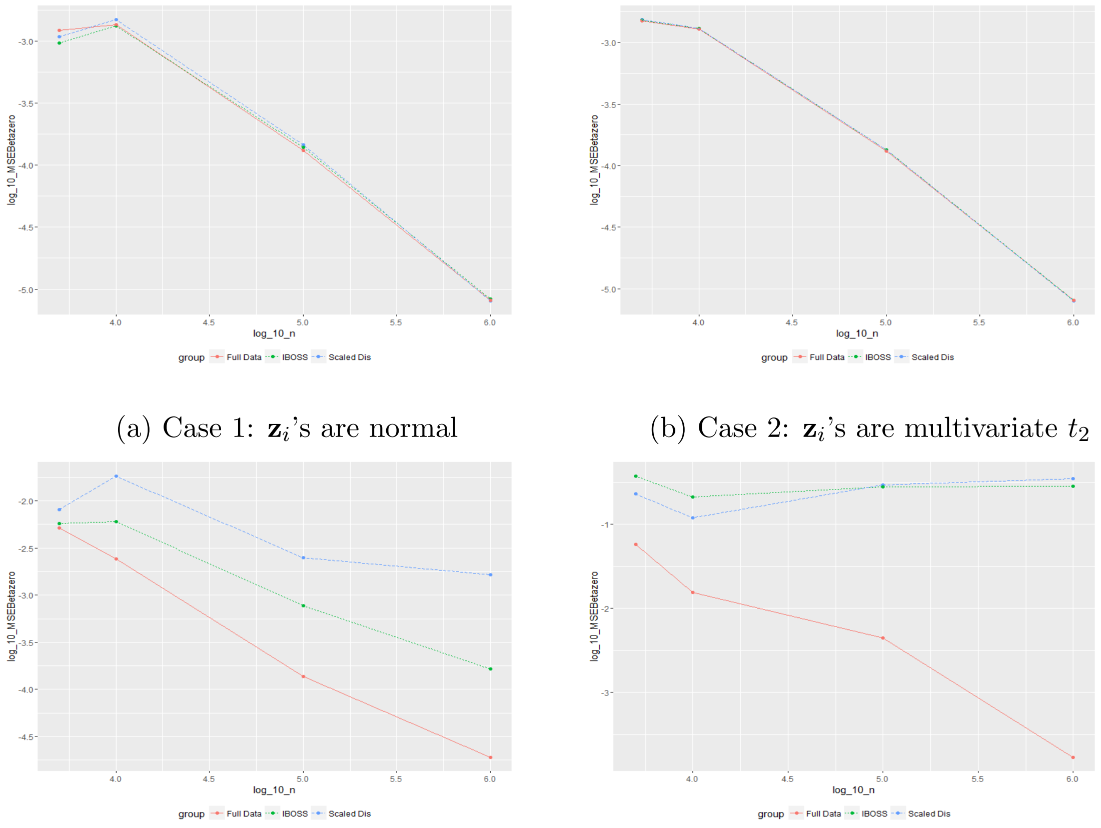
(b) Case 2: $\mathbf{z}_i$'s are multivariate t_2

(c) Case 3: $\mathbf{z}_i$'s are lognormal

(d) Case 4: $\mathbf{z}_i$'s are mixtures

Figure 2.1: Simulations of Asymptotic Property of $\hat{\boldsymbol{\beta}}_1^*$ with changing n and fixed k

The simulation results of the asymptotic property for $\hat{\beta}_0^*$ show similar conclusions except for Figure 2.2(c). As shown in Figure 2.2(a) and (b), Full data, SSDA and D-optimal IBOSS have almost the same behaviors. This is consistent with theoretical analysis when $E(\mathbf{z}_i) = 0,\ i = 1, \ldots, n$, the rate of convergence of $\hat{\beta}_0^*$ can be faster than that of $\hat{\boldsymbol{\beta}}_1^*$. In Figure 2.2(c) where $\mathbf{z}_i$'s are lognormal, there are obvious gaps among the three methods. The slower rates of D-optimal IBOSS and SSDA are because $E(\mathbf{z}_i) = 1,\ i = 1, \ldots, n$. In Figure 2.2(d), SSDA and D-optimal IBOSS behave similarly and the rates of convergence is bounded by the uniformly distributed covariates. The result of Full Data converges.



(a) Case 1: $\mathbf{z}_i$'s are normal

(b) Case 2: $\mathbf{z}_i$'s are multivariate t_2

(c) Case 3: $\mathbf{z}_i$'s are lognormal

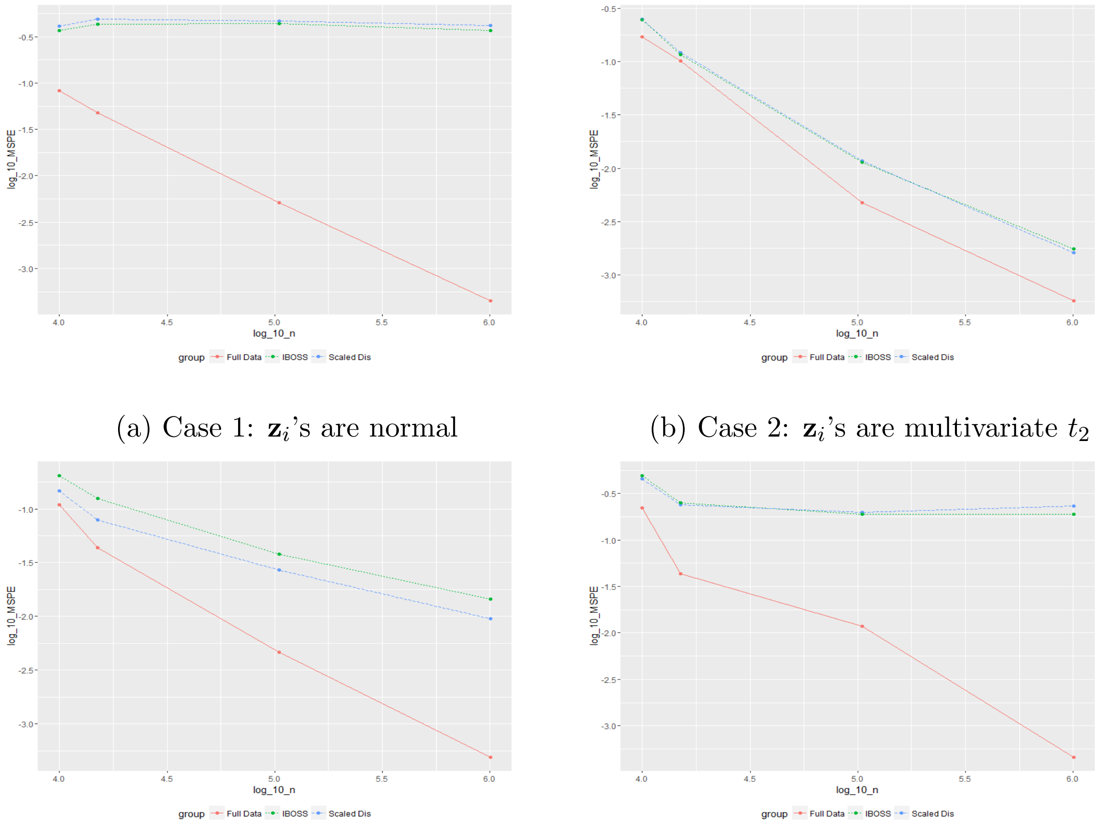
(d) Case 4: $\mathbf{z}_i$'s are mixtures

Figure 2.2: Simulations of Asymptotic Property of $\hat{\beta}_0^*$ with changing n and fixed k

Prediction Error

As can be seen from Figure 2.3, SSDA retains the the same asymptotic properties as D-optimal IBOSS. The relative behavior of SSDA in prediction compared to

D-optimal IBOSS and full data are similar to that of SSDA in slope parameter estimation. In Figure 2.3(a), (b) and (d), SSDA and D-optimal IBOSS have almost the same performance. Figure 2.3(c) suggests that under lognormal distribution, SSDA is a better way for subdata selection than D-optimal IBOSS. Note that the intercept should be estimated using $\hat{\beta}_0 = \bar{y} - \bar{\mathbf{z}}^T \hat{\boldsymbol{\beta}}_1^D$ for better prediction errors.



(a) Case 1: $\mathbf{z}_i$'s are normal

(b) Case 2: $\mathbf{z}_i$'s are multivariate t_2

(c) Case 3: $\mathbf{z}_i$'s are lognormal

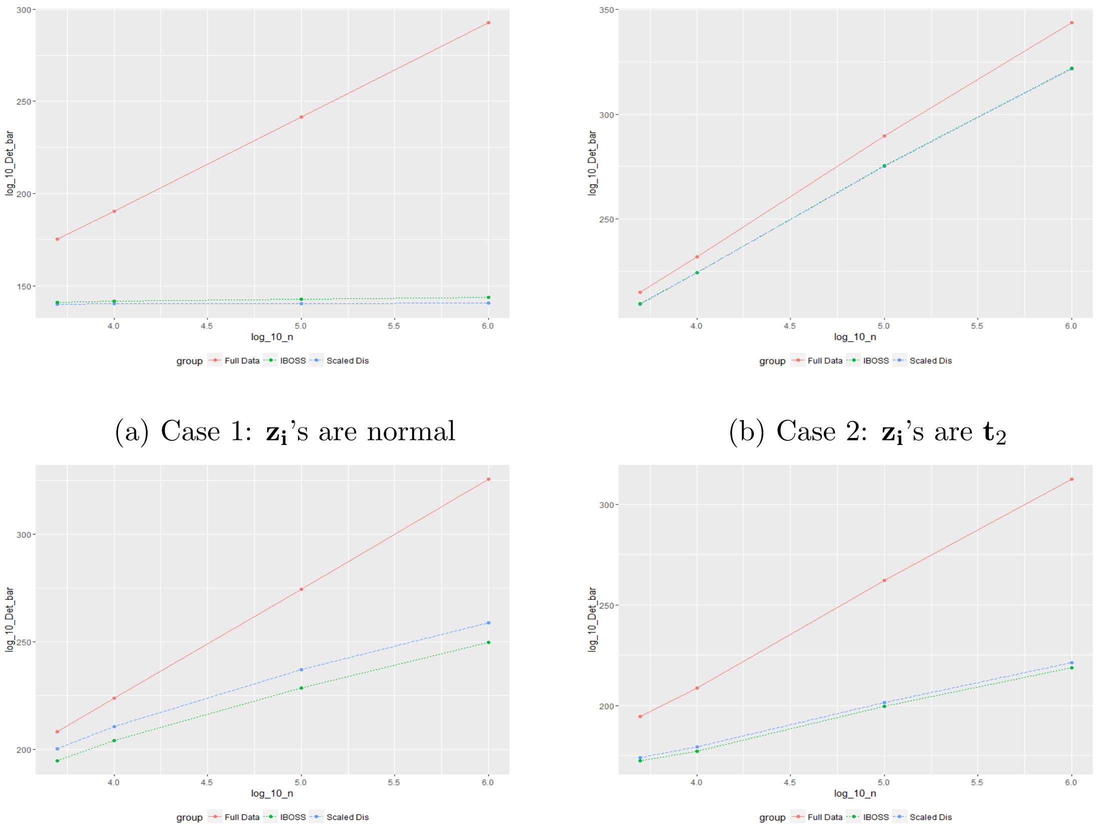
(d) Case 4: $\mathbf{z}_i$'s are mixtures

Figure 2.3: Simulations of Asymptotic Property of MSPEs with changing n and fixed k

Determinant of $\mathbf{M}(\boldsymbol{\delta})$

Rather than considering variables one by one as D-optimal IBOSS does, SSDA emphasizes on the combined effect of all variables. Therefore we expect SSDA gives a larger or at least almost equal determinant as D-optimal IBOSS does. Shown in

Figure 2.4 are the simulation results on $|\mathbf{M}(\boldsymbol{\delta})|$. In Figure 2.4(c) and (d), the determinants of SSDA are larger than those of D-optimal IBOSS, as we expected. In 2.4(b), the determinants are equal. Maybe it is because the t_2 distribution has such heavy tails that the extreme points selected by the two algorithms are almost the same. In Figure2.4(a), D-optimal IBOSS presents slightly larger determinants than SSDA. Both of the algorithms show asymptotic properties of $|\mathbf{M}(\boldsymbol{\delta})|$ as n increases.



(a) Case 1: $\mathbf{z_i}$'s are normal

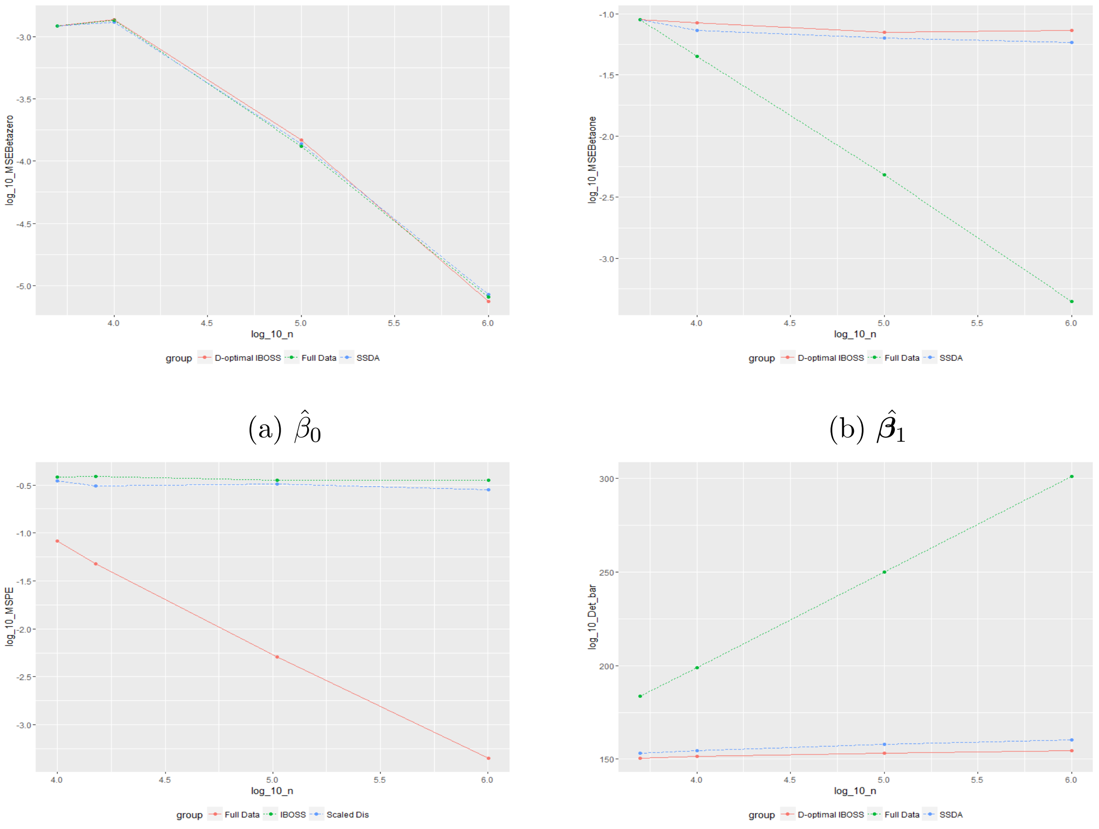
(b) Case 2: $\mathbf{z_i}$'s are $\mathbf{t}_2$

(c) Case 3: $\mathbf{z_i}$'s are lognormal

(d) Case 4: $\mathbf{z_i}$'s are mixtures

Figure 2.4: Simulations of Asymptotic Property of $|\mathbf{M}(\boldsymbol{\delta})|$

Remark 8. *When covariates are mutually moderately correlated as shown in the above settings, SSDA and D-optimal IBOSS perform similarly. This is because when selecting a point with extreme value for one variable, the values of other variables of this point tend to be extreme due to the correlations. And this point is a potential* *choice for the subdata selected by SSDA. As the correlation coefficients increase, SSDA and D-optimal IBOSS become more and more similar to each other. When covariate not correlated, we believe SSDA performs better than D-optimal IBOSS. To verify the above discussed ideas, we conduct similar simulations with two different covariance structures. he first one is mutually uncorrelated structure with $\boldsymbol{\Sigma} = \boldsymbol{I}_n$. The second one is mutually highly correlated structure with $\boldsymbol{\Sigma}_{ij} = 0.9^{I(i \neq j)}$, where $i, j = 1, \ldots, p$ and $I(i \neq j) = 1$ if $i \neq j$ and 0 otherwise. Only Case 1: normal distribution is used. Simulation results are shown in Figure 2.5 and Figure 2.6. The results support our discussions here.*



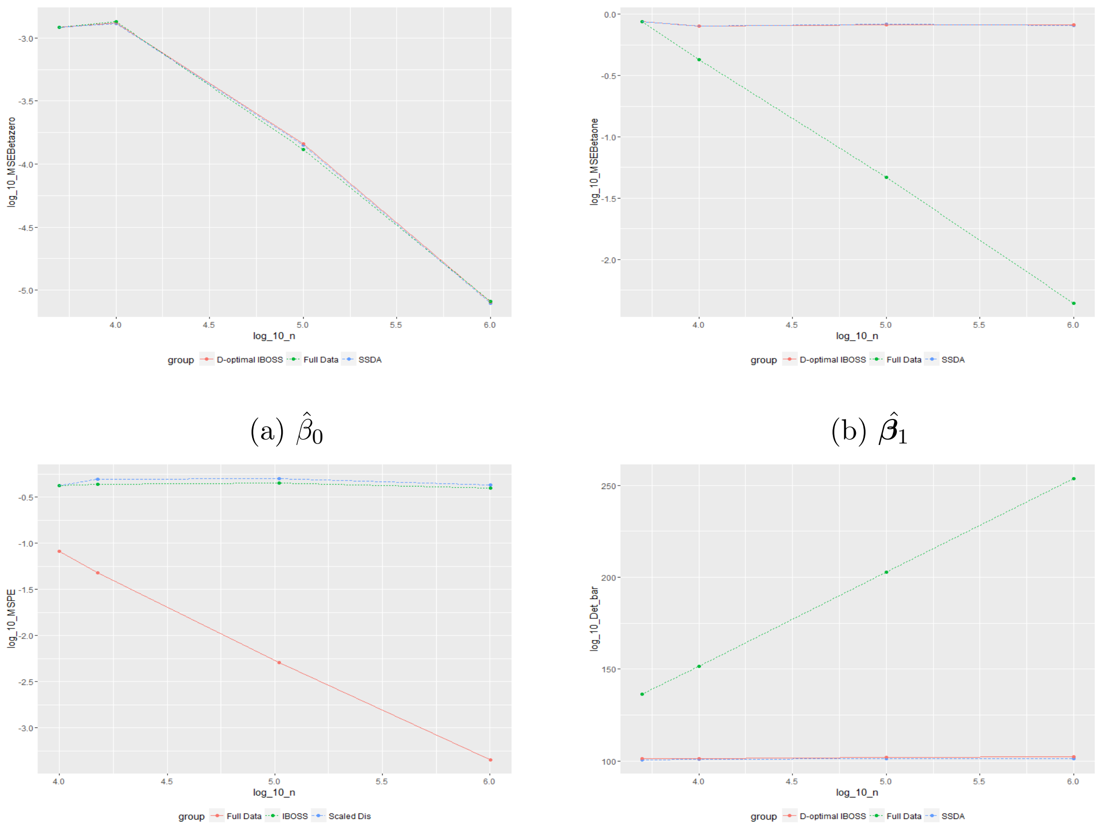
(a) $\hat{\beta}_0$

(b) $\hat{\boldsymbol{\beta}}_1$

(c) Prediction Errors

(d) Determinants

Figure 2.5: Simulations of Different Asymptotic Properties Under Uncorrelated Multivariate Normal Distribution



(a) $\hat{\beta}_0$

(b) $\hat{\boldsymbol{\beta}}_1$

(c) Prediction Errors

(d) Determinants

Figure 2.6: Simulations of Different Asymptotic Properties Under Mutually Highly Correlated Multivariate Normal Distribution

Remark 9. *As we have noticed in the simulations, when data are lognormally distributed SSDA performs noticeably better than D-optimal IBOSS except for the estimation of $\hat{\beta}_0$. The better performances are because lognormal distribution is a one tail distribution. Besides selecting points in the heavy tail, D-optimal IBOSS will also select those less influential points near zero. While SSDA will only select points with extreme values in the heavy tail. Therefore, SSDA includes twice as many influential points as D-optimal IBOSS. The worse performance of SSDA when estimating the intercept in lognormal distribution is due to the same reason. Only selecting points far from the origin point will make the estimation of intercept inaccurate. But the overall effect in prediction shows that SSDA is still a better choice than D-optimal*

IBOSS.

Conclusion From the simulations, we can see that the asymptotic behaviors of D-optimal IBOSS and SSDA are similar. In estimation of parameters, SSDA performs better than D-optimal IBOSS in estimating the slopes but worse in estimating intercept. When it comes to prediction and giving larger determinants of information matrices, SSDA is a better option. Correlation coefficients have effects on the relative performs on SSDA and D-optimal IBOSS.

2.4.2 Simulation on Computation Time

As we analyzed in Section 2.3, SSDA has the same time complexity as D-optimal IBOSS. However, it may perform much better than D-optimal IBOSS due to some features. The following simulations are conducted to help us understand how SSDA and D-optimal IBOSS perform in real computations.

The settings of the simulations are:

Case 1 The full data sizes are $n = 5 \times 10^3, 5 \times 10^4, 5 \times 10^5$ with fixed $p = 500, k = 1000$.

Case 2 The numbers of variables are $p = 10, 100, 500$ with fixed $n = 5 \times 10^5, k = 1000$.

All the simulations are conducted using R programming language on a desktop Windows 10 with an I5 laptop processor and 8GB memory.

At the beginning of simulation a random covariate matrix is generated. SSDA and D-optimal IBOSS method are wrapped in two functions. Using R function *microbenchmark*, we can apply each function to the covariate matrix 100 times and get the quantiles of CPU time (in milliseconds). The results are as follows:

The two tables on computation time show that SSDA is a significant improvement over D-optimal IBOSS method in computation efficiency, especially when $n \gg kp$.

n	Method	Min	LQ	Median	UQ	Max
5×10^3	SSDA	61.56	67.10	69.68	72.51	147.20
	D-OPT	258.07	264.59	268.23	271.95	350.27
5×10^4	SSDA	487.66	515.24	534.94	565.70	691.17
	D-OPT	1998.60	2034.31	2084.47	2188.47	2685.26
5×10^5	SSDA	5477.78	6278.06	7154.60	7406.64	9437.10
	D-OPT	31142.27	36794.27	40274.22	42449.30	50526.69

Table 2.1: Computation Comparison Case 1: n changes while p and k are fixed 500 and 1000 respectively

p	Method	Min	LQ	Median	UQ	Max
10	SSDA	97.98	103.36	107.06	137.02	270.64
	D-OPT	530.17	630.26	659.78	694.81	889.93
100	SSDA	922.59	1006.41	1096.28	1179.93	1504.53
	D-OPT	6066.83	6188.31	6260.32	6411.80	6946.96
500	SSDA	5477.78	6278.06	7154.60	7406.64	9437.10
	D-OPT	31142.27	36794.27	40274.22	42449.30	50526.69

Table 2.2: Computation Comparison Case 2: p changes while n and k are fixed 5×10^5 and 1000 respectively

The ratio r_t of D-optimal IBOSS median computation time over SSDA median computation time are approximately in the range from 3.85 to 6.16. In Table 2.1, as n grows from 5×10^3 to 5×10^5 and $kp = 5 \times 10^5$ stays the same, r_t grows from 3.85 to 5.63. Table 2.2 shows that with p increasing from 10 to 500, r_t decreases from 6.16 to 5.63. This is because as n/kp increases, the dominant part of time complexity $O(np+kp^2)$ changes from $O(kp^2)$ to $O(np)$. This is when the computation time shown in the tables truly reveals how well the two algorithms perform because the $O(kp^2)$ is the time complexity of OLS estimation process. Thus, we can draw conclusion that

SSDA is approximately 6 times faster than D-optimal IBOSS in this setting.

2.5 Robustness

When talking about the robustness of SSDA in this section, we focus on three issues, nonorthogonality, interaction terms and variable misspecification.

2.5.1 Nonorthogonality

One of the assumptions of SSDA is that the distances are calculated under the Cartesian Coordinate system, which means the variables are uncorrelated. In this subsection we will try to simulate under different covariance structures to see how these structures affect the performance of SSDA.

Remark 10. *The cosine of angle θ between two coordinates is related to their correlations with $\cos\theta = 1$ for 0 degree and $\cos\theta = 0$ for 90 degrees.*

As the covariances increase, the Euclidean distances we use become more and more misleading compared to the true Euclidean distances which take nonorthogonality (correlations) into consideration. This results in selecting those points with less contributions to maximizing the information matrix and leaving out those that really matter.

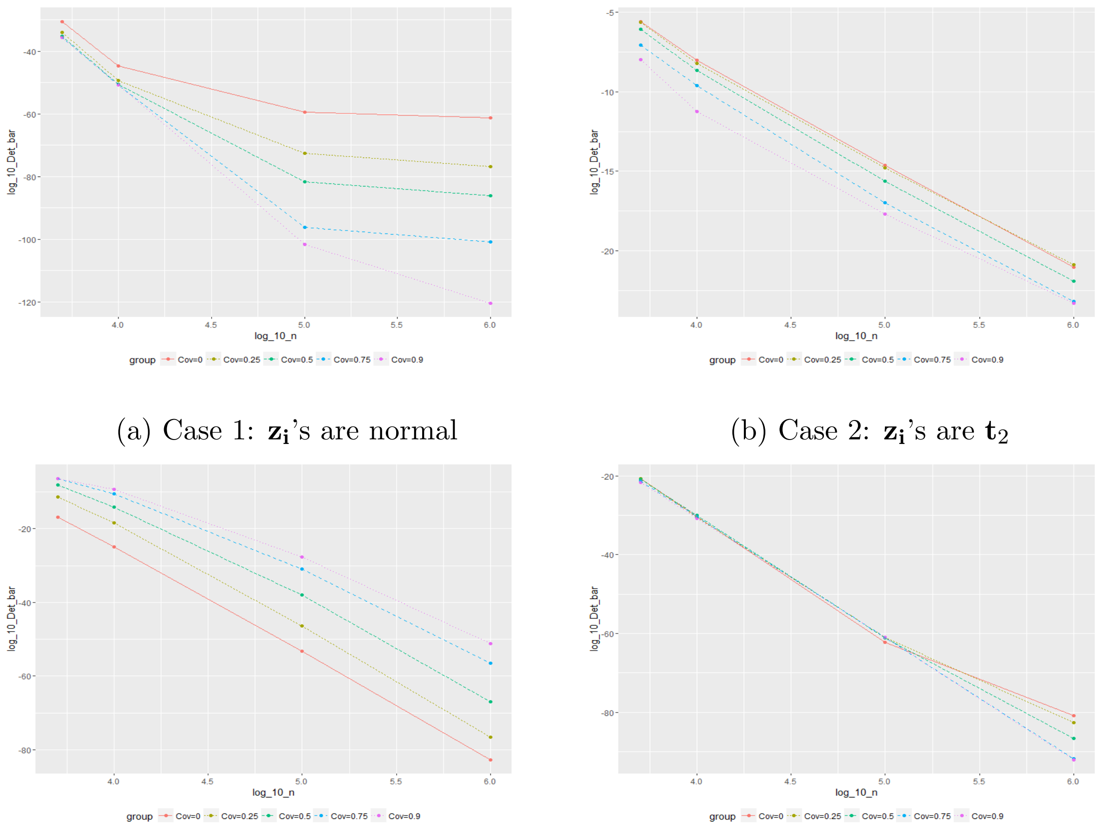
The settings of this simulation are the same as that of Section 2.4.1 except for the following:

The distributions now have means $\mathbf{1}$ instead of $\mathbf{0}$ and the covariance structures are matrices with the same diagonal elements 1 but different off-diagonal elements 0, 0.25, 0.5, 0.75, 0.9 respectively. Determinant ratios, $|\mathbf{M}(\boldsymbol{\delta})|/\min\{|\mathbf{X}^T\mathbf{X}|, \mathbf{upperbound}\}$, are calculated in each simulation, where **upperbound** refers to the upper bound in equation 1.5 and $\min\{|\mathbf{X}^T\mathbf{X}|, \mathbf{upperbound}\}$ provides an upper bound for the largest value $|\mathbf{M}(\boldsymbol{\delta})|$ can reach. The median of these ratios for a specific combination of n and covariance structure are used. The ratio here represents how well the subdata approximate full data with respect to maximizing the determinant of the information matrix.

Remark 11. *Medians of the determinant ratios are used here because the ratios are very small values. Thus, outliers will have big influence on the results. To capture the essential characteristic of the data with the robustness to outliers, medians are used instead of means.*

Plots of determinant ratios with different covariance structures but the same distributions are presented in Figure 2.7.

All of the four plots in Figure 2.7 show that the determinant ratio decreases as full data size increases. This means $\min\{|\mathbf{X}^T\mathbf{X}|, \mathbf{upperbound}\}$ converges faster than $|\mathbf{M}(\boldsymbol{\delta})|$ with respect to full data size n. But the story is slightly different for the relationship between determinant ratio and covariance structures. Figure 2.7(a) and (b) show the same pattern: as the covariances among variables increase, the determinant ratio decreases. This is exactly what we expected. The less correlated variables are the less deviation from our assumptions and thus the more effectively SSDA performs. However, Figure 2.7(c) shows an opposite trend of the determinant ratio. When the covariance increases, the determinant ratio increases. This may be because the covariate values of lognormal distribution are all positive. Let's denote the region where all covariate values are positive as region 1. Since all the variables are positively correlated, the distance to center of region 1 points are not heavily affected by nonorthogonality. The results for the mixed distribution can be viewed as a balance of the distributions. The plot shows the same patterns as in Figure 2.7(c) but the differences among the lines are really slight. We can notice that when n



(a) Case 1: $\mathbf{z_i}$'s are normal

(b) Case 2: $\mathbf{z_i}$'s are $\mathbf{t}_2$

(c) Case 3: $\mathbf{z_i}$'s are lognormal

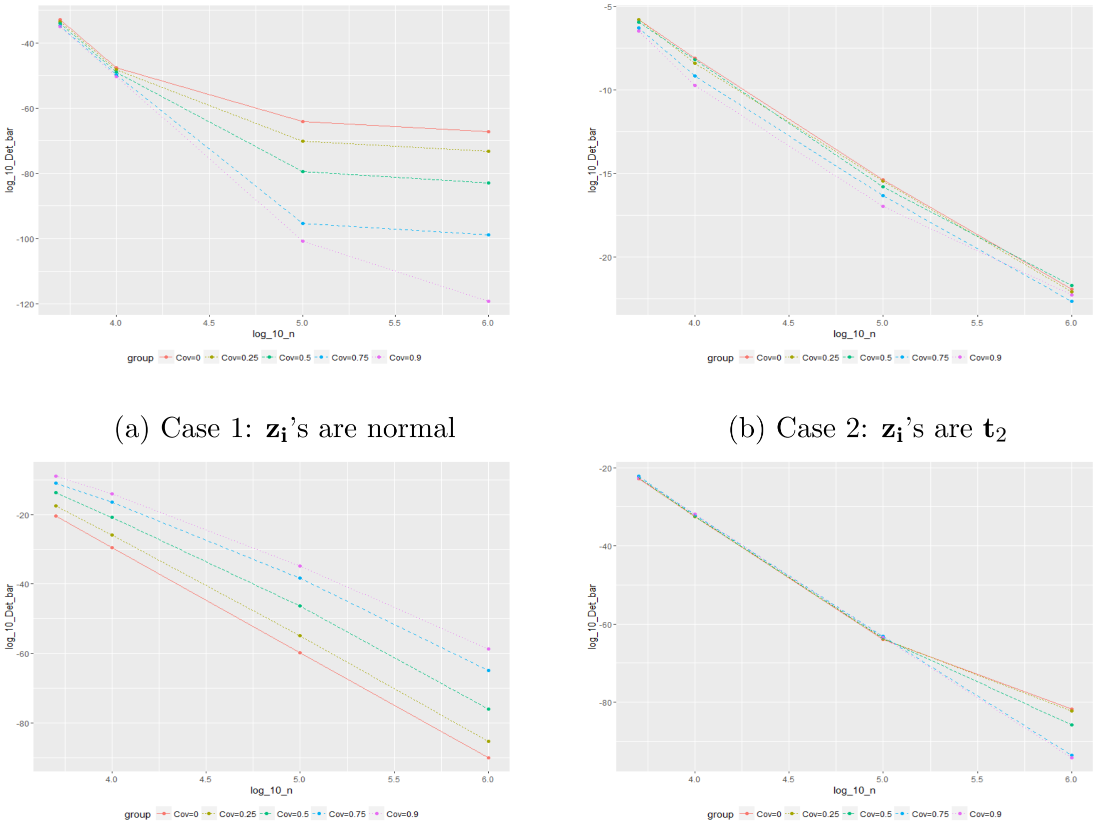
(d) Case 4: $\mathbf{z_i}$'s are mixtures

Figure 2.7: Simulations of Determinant Ratios Using SSDA Under Different Covariance Structures and Different Distributions

reaches 1×10^6, there is a reversal of the pattern. This may be because at this point, $|\mathbf{X}^T\mathbf{X}| > \mathbf{upperbound}$ in some covariance structures while $|\mathbf{X}^T\mathbf{X}| < \mathbf{upperbound}$ in the other ones.

Conclusions: From Figure 2.7, we can see that nonorthogonality does influence the performance of SSDA and the patterns are in the order of covariance. In real world cases where the covariance structures are much more complicated than what we have in this section, relationship between the determinant of an information matrix and corresponding covariance structure is a combined effect of the covariance structure and the distribution of covariate matrix.

Remark 12. *As is shown in Figure 2.8, D-optimal IBOSS is also influenced by nonorthogonality with similar patterns as SSDA. Therefore, SSDA is an excellent alternative to D-optimal IBOSS despite its lack of robustness in nonorthogonality.*



(a) Case 1: $\mathbf{z_i}$'s are normal

(b) Case 2: $\mathbf{z_i}$'s are $\mathbf{t}_2$

(c) Case 3: $\mathbf{z_i}$'s are lognormal

(d) Case 4: $\mathbf{z_i}$'s are mixtures

Figure 2.8: Simulations of Determinant Ratios Using D-optimal IBOSS Under Different Covariance Structures and Different Distributions

2.5.2 Interaction Terms

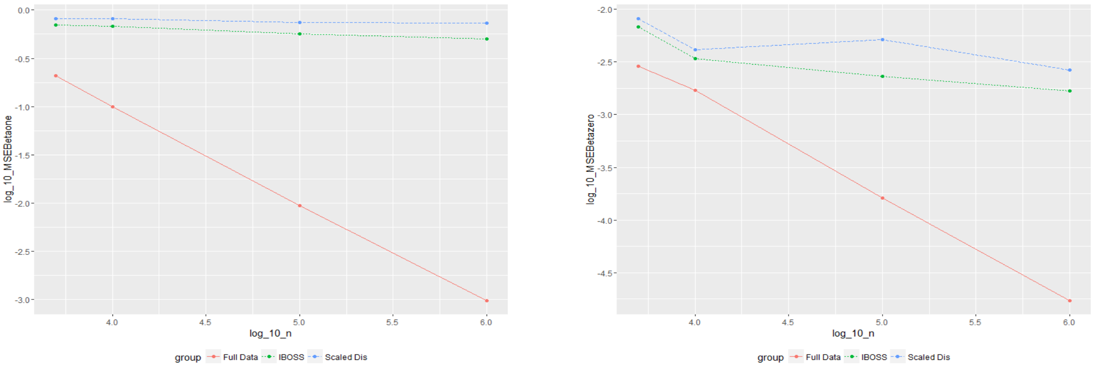
In this section, two models with interaction terms will be considered and interaction terms are not used in subdata selection but are used in parameter estimation.

Model 1 This model is the one discussed by Wang *et al.* (2017). The number of variables $p = 20$ and each $\mathbf{z}_i \sim N(\mathbf{0}, \mathbf{\Sigma})$, $i = 1, \ldots, n$ has a multivariate normal distribution. $\mathbf{\Sigma}$ is the covariance matrix with 1 as diagonal elements and 0.5 as off-diagonal elements. The model matrix contains interaction and quadratic terms with $\mathbf{x_i} = (\mathbf{z}^{\mathrm{T}}, z_1\mathbf{z}^{\mathrm{T}}, z_2z_{11}, z_2z_{12}, \ldots, z_2z_{20})^{\mathrm{T}},\ i = 1, \ldots, n$. Other settings are the same as those in Section 2.4.1.

Model 2 This model contains all the main effects and pairwise interaction terms. The number of variables $p = 10$ and other settings are the same as in Section 2.4.1. Case 1, 2, 3 and 4 are studied respectively.

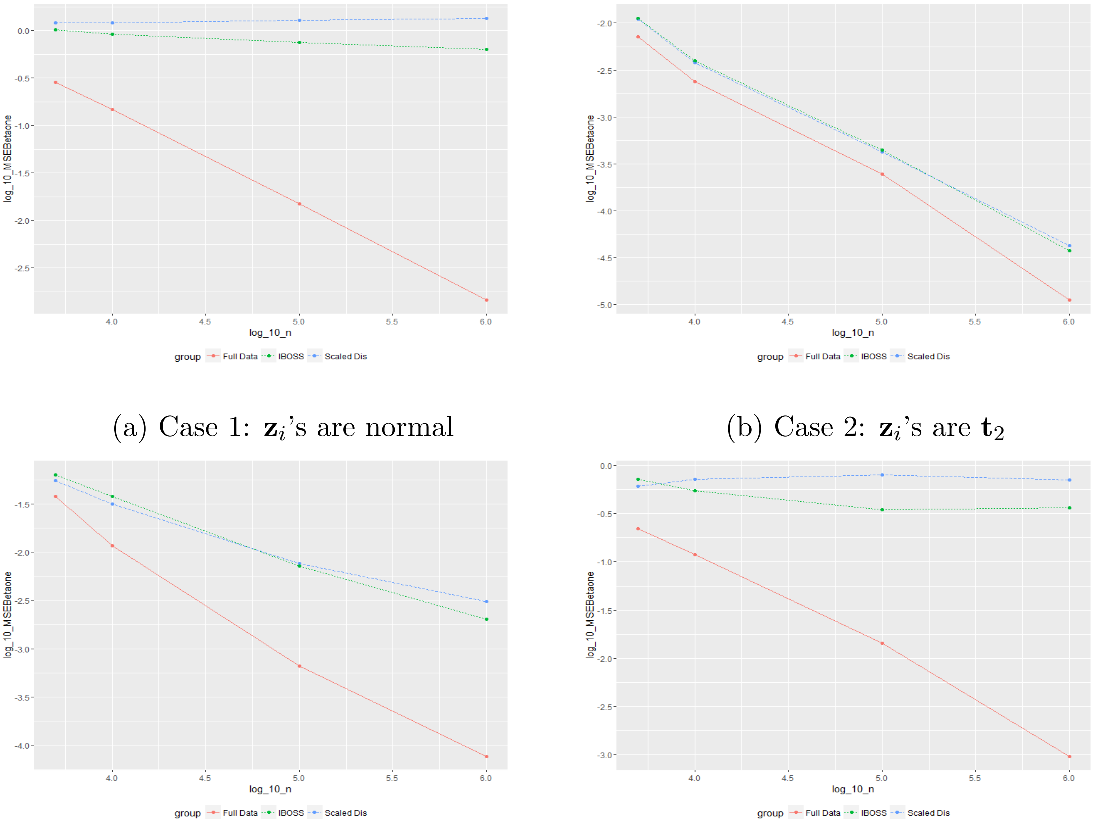
The results from Figure 2.9 are in accordance with theoretical results. In Figure 2.9(a), both of the rates of convergence of $\hat{\boldsymbol{\beta}}_1^*$'s are pretty slow and D-optimal IBOSS gives slightly better estimations than SSDA does. This is similar as the result from Figure 2.1(a), where interaction terms are not in the model. Figure 2.9(b) presents different behaviors of $\hat{\beta}_0^*$ compared to Figure 2.2(a). In Figure 2.2, both of the methods converge at the same rates as full data while in Figure 2.9(b) they converge at much slower rates. Weird as it seems, it actually comply with the theory. Let's suppose that Z is the covariate matrix and Z_1 is the model matrix without intercept. Since $E(z_iz_j) = E(z_i)E(z_j) + Cov(z_i, z_j) = Cov(z_i, z_j) \neq 0$, $E(\bar{\mathbf{z}}_1) \neq 0$. Therefore, the rates of convergence of $\hat{\beta}_0^*$ are dominated by the rates of $\hat{\boldsymbol{\beta}}_1^*$. So SSDA is robust under model 1.

From Figure 2.10 we can see that under normal and mixed distributions, SSDA does not converge while D-optimal IBOSS converges in a very slow rate. Under t_2 and lognormal distributions, D-optimal IBOSS and SSDA converges at similar rates as in Figure 2.1. Therefore, when estimating the slopes, SSDA is not robust under Model 2 with normal or mixed distributions but it is robust under heavy-tailed distributions such t_2 and lognormal. As a conclusion, it is comparable method to D-optimal IBOSS under Model 2.



(a) The asymptotic property of $\hat{\boldsymbol{\beta}}_1^*$ (b) The asymptotic property of $\hat{\beta}_0^*$

Figure 2.9: Simulations of $\hat{\boldsymbol{\beta}}_1^*$ and $\hat{\beta}_0^*$ under Model 1

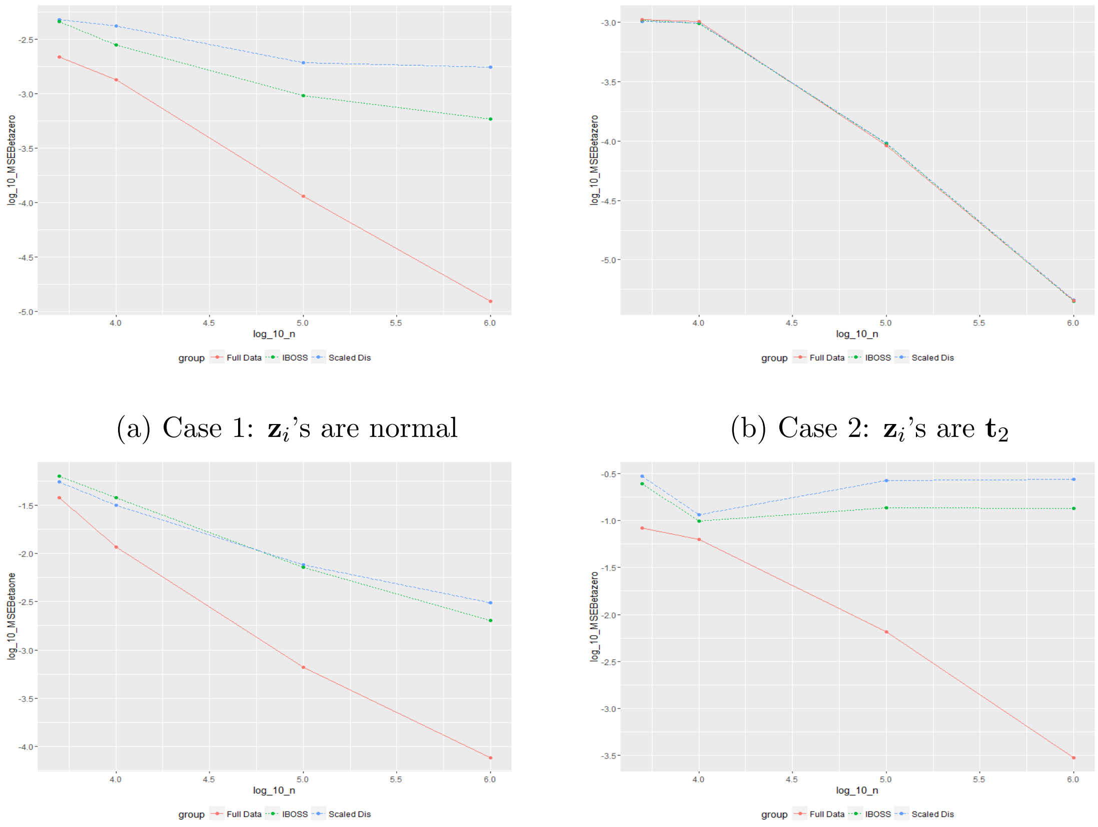


(a) Case 1: $\mathbf{z}_i$'s are normal (b) Case 2: $\mathbf{z}_i$'s are $\mathbf{t}_2$

(c) Case 3: $\mathbf{z}_i$'s are lognormal (d) Case 4: $\mathbf{z}_i$'s are mixtures

Figure 2.10: Simulations of $\hat{\boldsymbol{\beta}}_1^*$ under Model 2

As for $\hat{\beta}_0^*$, from Figure 2.11 we can see that D-optimal IBOSS and SSDA converge at similar rates as in Figure 2.2 except Figure 2.11(a). As analyzed previously, the result of Case 1 agrees with the theory. Also, in Figure 2.11(b), all of the three rates of convergence of $\hat{\beta}_0^*$ are the same even though $E(\bar{\mathbf{z}}_1) \neq 0$. This is because the two rates of convergence of $\hat{\boldsymbol{\beta}}_1^*$ using D-optimal IBOSS and SSDA are almost the same as that using full data. Above all, we can draw the conclusion that when estimating the intercept, SSDA is robust under Model 2.



(a) Case 1: $\mathbf{z}_i$'s are normal

(b) Case 2: $\mathbf{z}_i$'s are $\mathbf{t}_2$

(c) Case 3: $\mathbf{z}_i$'s are lognormal

(d) Case 4: $\mathbf{z}_i$'s are mixtures

Figure 2.11: Simulations of $\hat{\beta}_0^*$ under Model 2

2.5.3 Variable Misspecification

Both IBOSS and SSDA select subdata basing on variables. When the number of variables is large, true variables that is in the model and fake ones that is not in the model may be mixed in the full data. Variable selection is necessary. However, inaccurate variable selection algorithms will result in excluding the true variables or/and including the fake variables. In this section, we will study how these two situations: excluding the true variables and including the fake variables affect the behavior of IBOSS and SSDA.

Simulation Settings:

Situations 1. Excluding one the true variables but not including fake variables.

2. Including all the true variables and 200 other fake variables.

Data The full data sizes are $n = 5 \times 10^3, 1 \times 10^4, 1 \times 10^5, 1 \times 10^6$ with p variables ($p =$ 50 in Situation 1 and 250 in Situation 2). The subdata size is fixed at $k = 1000$. Suppose $\mathbf{\Sigma}$ is the covariance matrix with $\mathbf{\Sigma}_{ij} = 0.5^{I(i \neq j)}$, where $i, j = 1, \ldots, p$ and $I(i \neq j) = 1$ if $i \neq j$ and 0 otherwise. A test data set with $n_{test} = 5 \times 10^3$ are created for calculating the prediction errors. The covariate matrices $\mathbf{Z}$'s are generated according to the following cases: for each entry $\mathbf{z}_i,\ i = 1, \ldots, n$

Case 1 $\mathbf{z}_i \sim N(\mathbf{0}, \mathbf{\Sigma})$ has a multivariate normal distribution.

Case 2 $\mathbf{z}_i \sim LN(\mathbf{0}, \mathbf{\Sigma})$, has a multivariate lognormal distribution.

Case 3 $\mathbf{z}_i \sim t_2(\mathbf{0}, \mathbf{\Sigma})$, has a multivariate t distribution with degrees of freedom $\upsilon = 2$.

Model The Models are different for the two situations:

Situation 1: Since the 50 variables are equivalent to each other, without loss of generality we can choose the first variable to be excluded in variable selection. Then $\mathbf{Z}_{selected}$ is the other 49 columns of $\mathbf{Z}$. The following linear model is used: $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$, where $\mathbf{X} = (\boldsymbol{j}_n, \mathbf{Z})$, $\boldsymbol{\beta}$ is a 51 by 1 vector of ones and $\varepsilon_i \overset{i.i.d}{\sim} N(0, \sigma^2),\ i = 1, \ldots, n,\ \sigma^2 = 9$.

Situation 2: The first 50 variables are set to be true variables and the remaining ones are set to be fake. Then $\mathbf{Z}_{true}$ is the first 50 columns of $\mathbf{Z}$. The following linear model is used: $\mathbf{y} = \mathbf{X}_{true}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$, where $\mathbf{X}_{true} = (\boldsymbol{j}_n, \mathbf{Z}_{true})$, $\boldsymbol{\beta}$ is a 51 by 1 vector of ones and $\varepsilon_i \overset{i.i.d}{\sim} N(0, \sigma^2),\ i = 1, \ldots, n,\ \sigma^2 = 9$.

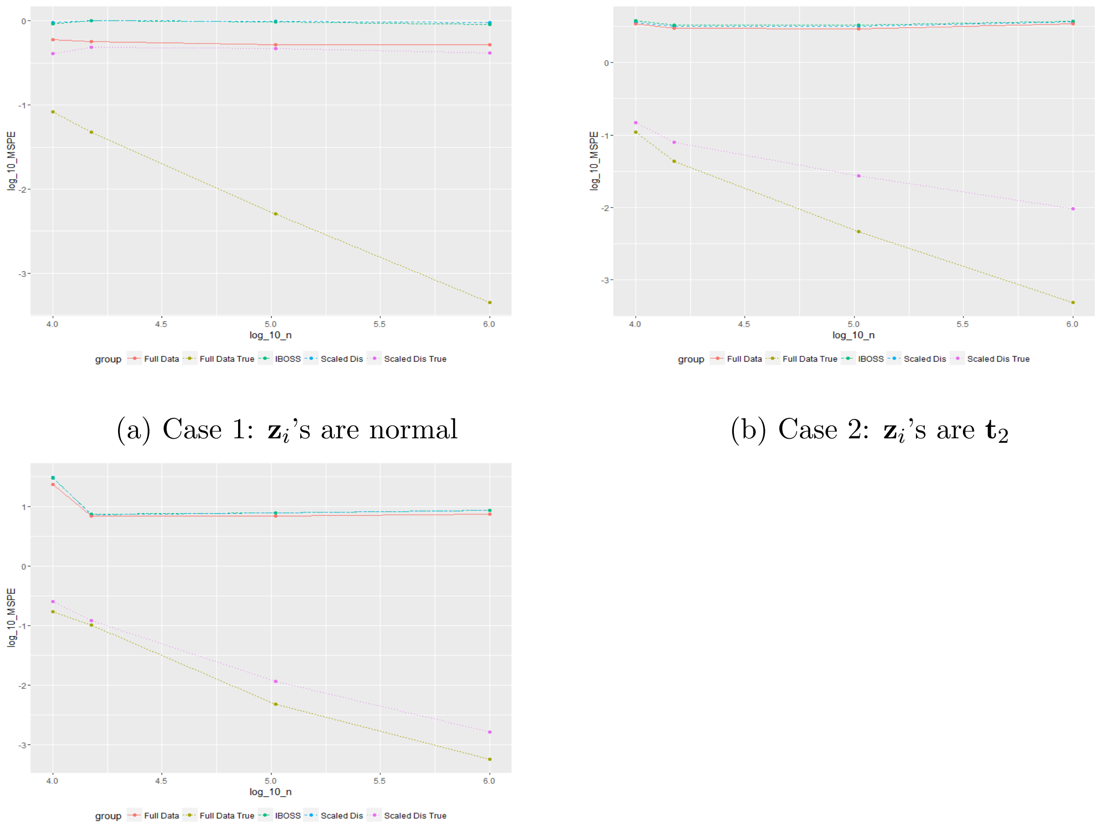
Simulation In each simulation, we use the selected data to calculate the parameter estimates. We have 49 parameter estimates in Situation 1 and 250 in Situation 2. Prediction errors MSPEs are then calculated using test sets. The simulation is repeated $S = 100$ times and the means of MSPEs over S simulations are calculated for plotting purpose. Five approaches including D-optimal IBOSS, SSDA, SSDA with true variables, Full Data OLS and Full Data OLS with true variables are compared using the same full data set and response values.

Graphics In each case, we plot $\log_{10}$MSPE and against $\log_{10}n$ with respect to the five approaches.

In Figure 2.12, we can clearly see that none of D-optimal IBOSS, SSDA and Full Data OLS are robust when a true variable is excluded from the model matrix. The Asymptotic properties are not preserved.

Figure 2.13 shows that both D-optimal IBOSS and SSDA are robust when fake variables is included in the model. Including fake variables, they converge at the same rate as SSDA with true variables. But the prediction accuracies of these two methods in Situation 2 are much worse than that of SSDA with true variables. When the tail of the distribution of covariate matrix is heavy enough, SSDA with true variables even out perform Full Data OLS in Situation 2. This can be seen from Figure 2.13(c) and part of 2.13(b).

As a conclusion, both D-optimal IBOSS and SSDA are robust when fake variables are included but not when true variables are excluded. Conducting an accurate variable selection algorithm before selecting subdata with SSDA or D-optimal IBOSS can improve the prediction errors significantly.

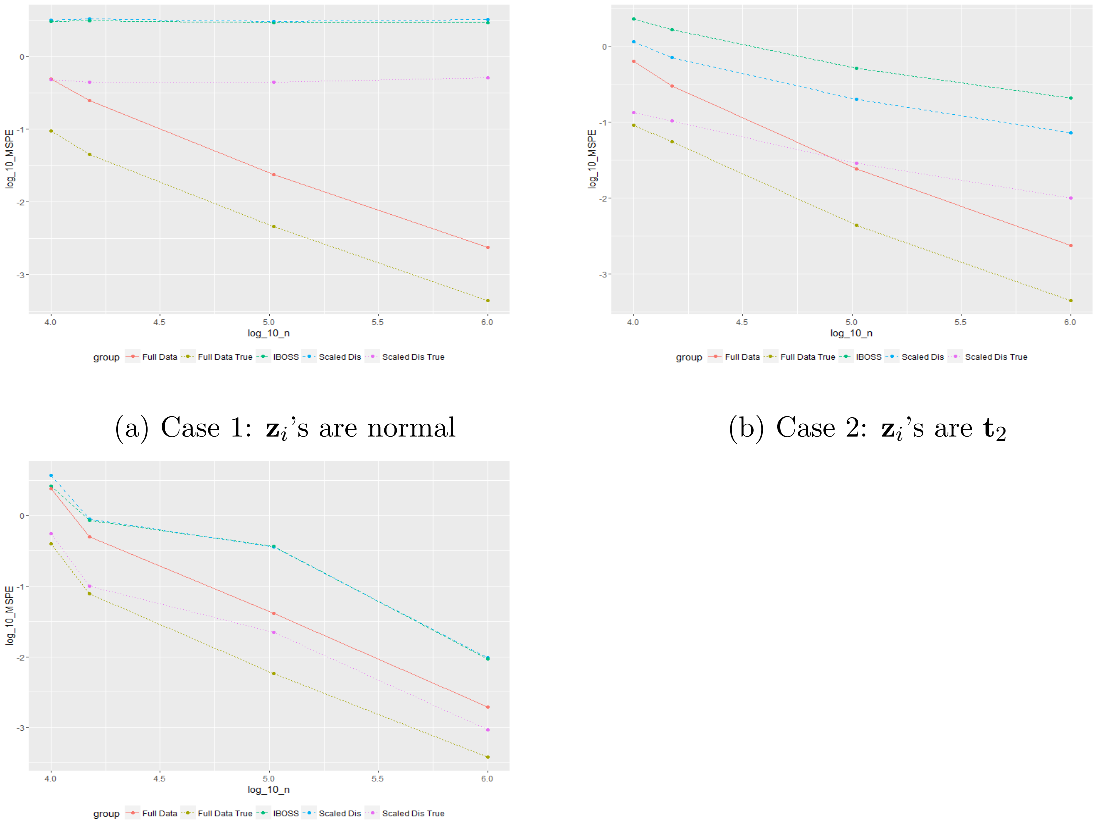


(a) Case 1: $\mathbf{z}_i$'s are normal

(b) Case 2: $\mathbf{z}_i$'s are $\mathbf{t}_2$

(c) Case 3: $\mathbf{z}_i$'s are lognormal

Figure 2.12: Simulations of MSPEs in Situation 1: Excluding True Variables



(a) Case 1: $\mathbf{z}_i$'s are normal

(b) Case 2: $\mathbf{z}_i$'s are $\mathbf{t}_2$

(c) Case 3: $\mathbf{z}_i$'s are lognormal

Figure 2.13: Simulations of MSPEs in Situation 2: Including Fake Variables

Chapter 3

A Suitable Variable Selection Method for Large Data Size

3.1 Motivation

From Section 2.5.3, we can see that both D-optimal IBOSS and Squared Scaled Distance Algorithms are sensitive to variable misspecification. Therefore, in situations where both the number of variables p and full data size n are large, an accurate variable selection algorithm must be implemented before subdata are selected using these two algorithms. Penalized regression models such as the LASSO have been extensively used in variable selection. However, when the data are too massive to be stored in the memory, fitting LASSO-type models is not feasible. Also, the LASSO does not perform well when multicollinearity is present between true variables and fake variables.

In this chapter, we will propose a variable selection algorithm when p is large but not massively and $n \gg p$. The motivation comes from random forest algorithms where small incomplete trees are built and the final results depend on the votes of all the trees. Our variable selection algorithm is based on the votes of multiple small random selected subsets to decide which variables are to be selected. A LASSO regression is conducted for each subset and we count the number of times each variable is selected. We believe that the true variables will be consistently selected and thus will have larger counts than fake variables do. Then the counts are clustered into two groups using k-means algorithm. The group of variables with higher average counts will be selected. We will show that our algorithm can break the multicollinearity among variables and select variables accurately even when they are highly correlated. Also, using less computational resources, our method performs far better than the LASSO using the full data (James *et al.* (2013)). Furthermore, this algorithm is feasible when the full data is too large to be stored in memory because we only use small subsamples.

3.2 Algorithm

Algorithm 2 (Variable Selection via the Votes of Random Subsamples)**.** *Suppose that the covariate matrix is an n by p matrix and we will randomly select g subsamples with the same subsample size $n_s > p$.*

Step 1 *Create a p by 1 count table with initial value $\mathbf{0}$. Each element of the count table corresponds to a variable from the covariate matrix.*

Step 2 *Randomly select a subsample of size n_s from the full data set with Replacement. The full data is treated as the population here.*

Step 3 *Perform the LASSO on the selected subsample. If a variable is selected by the LASSO, add 1 to the corresponding variable in the count table.*

Step 4 *Repeat Steps 2 and 3 g-1 times and get the counts.*

Step 5 *Cluster the counts into two groups using k-means algorithm.*

Step 6 *Select the group of variables with larger average counts.*

Remark 13. *The choice of g is important. From simulations we find that Algorithm 2 works well if g is of the same order as the number of true variables. But in practice we* *may not know the exact number of true variables. One way to solve this problem is to set g to a guess for number of true variables if we have historical data. Alternatively, we can set g to an initial value (can be really small) and conduct Algorithm 2 on the full data. Set g to the number of selected variables. Empirically, the g value from the second method will be of the same scale but larger than the number of true variables. Thus it will increase the computing time but will not do harm to the accuracy. Better ways to find optimal g values are to be found in future studies.*

Remark 14. *For efficiency purposes, the subsample size n_s is set to be just slightly larger than $p + 1$ so that the LASSO can be performed. Thus Algorithm 2 works best when $n \gg p$. When $n_s < p$, we may perform variable selection algorithms that are suitable for this situation instead. But it is outside the scope of this paper.*

Remark 15. *The time complexity of iterative convex optimization problems such as the LASSO is complicated and tricky to analyze. Therefore, we will only show that the computing time of Algorithm 2 depends on the number of variables p only. Taking a subsample of size n_s with replacement has a time complexity of $O(n_s)$. The time complexity of LASSO on each subsample is related to n_s and p and in our setting $n_s \approx p$. The number of subsample g is only related with the number of true variables which is a proportion of p. When the number of clusters is less than five, the upper bound for time complexity of k-means on one dimension p data points is $O(p)$ (Dasgupta (2003)). As a conclusion, the time complexity of Algorithm 2 depends on the number of variables p only. Since it does not depend on n, Algorithm 2 has a big advantage when p is large but not massive and massive data size n is the major challenge. Another advantage of Algorithm 2 is that parallel computing can be easily implemented because the analysis of each subsample is independent.*

3.3 Simulation Study and Comparison with the Lasso on Full data

The setting of simulations are as follows:

Data The full data size is fixed at $n = 1 \times 10^5$ with 50 true variables ($p_{true} = 50$) and the numbers of fake variables (p_{fake}) are 100, 500, 1000, representing dense, moderate and sparse situation respectively. $p = p_{true} + p_{fake}$ is the total number of variables. The number of subsamples is taken as $g = 50$. Three covariance structures are studied. The first one is mutually uncorrelated structure with $\mathbf{\Sigma}^{(1)} = \boldsymbol{I}$. The second one is highly mutually correlated structure with $\mathbf{\Sigma}^{(2)}_{ij} = 0.8^{I(i \neq j)}$, where $i, j = 1, \ldots, p$ and $I(i \neq j) = 1$ if $i \neq j$ and 0 otherwise. The third one is a structure with elements $\mathbf{\Sigma}^{(3)}_{ij} = 0.8^{|i-j|},\ i, j = 1, \ldots, p$. The n by p covariate matrices $\mathbf{Z}$ are generated according to the following two cases: for each entry $\mathbf{z}_i,\ i = 1, \ldots, n$

Case 1 $\mathbf{z}_i \sim N(\mathbf{0}, \mathbf{\Sigma})$ has a multivariate normal distribution.

Case 2 $\mathbf{z}_i \sim t_2(\mathbf{0}, \mathbf{\Sigma})$, has a multivariate t distribution with degrees of freedom $\upsilon = 2$.

Model The following linear model is used: $\mathbf{y} = \mathbf{X_{true}}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$, where $\mathbf{X_{true}} = (\mathbf{1}, \mathbf{Z_{true}})$, $\boldsymbol{\beta}$ is a 51 by 1 vector of ones and $\varepsilon_i \overset{i.i.d}{\sim} N(0, \sigma^2),\ i = 1, \ldots, n,\ \sigma^2 = 9$. The n by p_{true} matrix $\mathbf{Z_{true}}$ is a part of $\mathbf{Z}$ with each column representing a true variable. In covariance structure one and two, true variables are selected as the first p_{true} variables of $\mathbf{Z}$. In covariance structure three, we select the columns for $\mathbf{Z}_{true}$ as close to equally spaced as possible in $\mathbf{Z}$. In this way, the true variables are slightly correlated with each other but they are more highly correlated with some fake variables near them.

Simulation The simulation is repeated $S = 10$ times. In each simulation, A covariate matrix is generated for each possible p_{fake} value listed above. Algorithm 2 and Full Data Lasso Regression are conducted on it. Two misspecification errors are used to evaluate the performances. The first kind of error is not including true variables, denoted as E1. The second kind of error is including fake variables in the model, denoted as E2. The mean numbers of appearances of E1 and E2 are recorded. The results are presented in Table 3.1-3.3.

From Table 3.1-3.3, we show that Algorithm 2 is extremely accurate. The only mistake it makes is in Table 3.2 where the variables are highly correlated in a dense situation, $p_{true} = 33.33\%p$. And the mistake is minor, including 0.3 fake variable on average. In moderate and sparse situations ($p_{true} = 9.09\%p$ and 4.76% respectively), Algorithm 2 perfectly includes all true variables and excludes all fake ones in all combinations of the three covariance structures and two distributions. It performs significantly better than full data LASSO in all cases. To some degree, the full data LASSO performs excellently in including true variables but does a terrible job in excluding fake variables. According to Section 2.5.3, Algorithm 2 is a much better choice than full data LASSO if we want to combine variable selection with IBOSS algorithms.

Although computing time is not recorded, we have shown that the time complexity of Algorithm 2 depends on p only while full data LASSO depends on n and p. Thus Algorithm 2 is much more efficient than full data LASSO when n is massively large. Also, thanks to the using of subsamples, Algorithm 2 is applicable to the situation where data size n is too large for the data to be stored in memory while full data LASSO is not feasible in this situation.

As a conclusion, When n is massively large and p is large, Algorithm 2 is a much better choice than full data LASSO in accuracy, efficiency and practicability. These nice properties of Algorithm 2 make it an excellent variable selection method before using IBOSS algorithms.

Number	t_2				Normal			
of Fake	Algorithm 2		Full Lasso		Algorithm 2		Full Lasso	
Variables	E1	E2	E1	E2	E1	E2	E1	E2
100	0	0	0	3.5	0	0	0	49
500	0	0	0	22.4	0	0	0	103.2
1000	0	0	0	49.0	0	0	0	141.2

Table 3.1: Covariance Structure I

Number	t_2				Normal			
of Fake	Algorithm 2		Full Lasso		Algorithm 2		Full Lasso	
Variables	E1	E2	E1	E2	E1	E2	E1	E2
100	0	0.3	0.7	36.3	0	0	0	6.4
500	0	0	1.2	110.8	0	0	0	17
1000	0	0	2.3	161.7	0	0	0	25.1

Table 3.2: Covariance Structure II

Number	t_2				Normal			
of Fake	Algorithm 2		Full Lasso		Algorithm 2		Full Lasso	
Variables	E1	E2	E1	E2	E1	E2	E1	E2
100	0	0	0	14.6	0	0	0	44.5
500	0	0	0	31.6	0	0	0	98.9
1000	0	0	0	64.0	0	0	0	140.1

Table 3.3: Covariance Structure III

Chapter 4

Conclusions

In this paper, we have created a new IBOSS algorithm (SSDA) for linear regression. It considers all the variables simultaneously instead of one by one as in D-optimal IBOSS. Through extensive simulation studies, we have shown that parameter estimates from SSDA retains the same asymptotic properties as those from D-optimal IBOSS while it performs approximately six times as fast as D-optimal IBOSS. When it comes to the determinant of information matrix of subdata, SSDA performs better than D-optimal IBOSS. In robustness study, we discover that the nonorthogonality is an influential factor for both D-optimal IBOSS and SSDA. Both of the algorithms are robust when interaction terms are present in the model. Excluding true variables is a critical problem for both D-optimal IBOSS and SSDA. But they are robust when fake variables are included. As a conclusion, SSDA is a good alternative for D-optimal IBOSS.

Further studies are necessary for better understanding of SSDA as well as D-optimal IBOSS. For example, the asymptotic properties of SSDA should be proved theoretically. Theoretical explanations for the behaviors under different covariance structures (nonorthogonality) should also be made.

The variable selection algorithm we have developed in Chapter 3 is a promising tool not only in the scenario where it selects variables for IBOSS algorithms but also in all the suitable situations where n is massive and p is moderately large. Its consistent accuracy under different covariance structures is its advantage for broad applications. Also, only correlated with p, its computation time is efficient when massive full data size n is the main challenge for analyzing the data. Using subsamples makes Algorithm 2 suitable for parallel computing.

The version of the variable selection algorithm here is a basic one. Further improvement can be made. For example, how to determine the value of g in a more efficient way and how to further improve the time complexity of Algorithm 2 are important questions to be solved.

We hope our work here can intrigue interests in further researches in both IBOSS and variable selection algorithms.

BIBLIOGRAPHY

Candes, E. and T. Tao, “The dantzig selector: statistical estimation when p is much larger than n”, The Annals of Statistics pp. 2313–2351 (2007).

Dasgupta, S., “How fast is k-means?”, COLT Computational Learning Theory, **2777**, 735 (2003).

Drineas, P., M. Mahoney, S. Muthukrishnan and T. Sarlos, “Sampling algorithms for l_2 regression and applications”, Numerische Mathematik **117**, 219–249 (2011).

Drineas, P., M. W. Mahoney and S. Muthukrishnan, “Faster least squares approximation”, In Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm pp. 1127–1136 (2006).

Fan, J. and J. Lv, “Sure independence screening for ultrahigh dimensional feature space”, Journal of the Royal Statistical Society: Series B (Statistical Methodology) **70**, 5, 849–911 (2008).

James, G., D. Witten, T. Hastie and R. Tibshirani, *An Introduction to Statistical Learning* (Springer Science, 2013).

Ma, P., M. Mahoney. and B. Yu, “A statistical perspective on algorithmic leveraging”, In Proceedings of the 31st International Conference on Machine Learning (ICML-14) pp. 91–99 (2014).

Ma, P., M. Mahoney. and B. Yu, “A statistical perspective on algorithmic leveraging”, Journal of Machine Learning Research **16**, 861–911 (2015).

Ma, P. and X. Sun, “Leveraging for big data regression”, Wiley Interdisciplinary Reviews: Computational Statistics **7**, 1, 70–76 (2015).

Meinshausen, N. and B. Yu, “Lasso-type recovery of sparse representations for high-dimensional data”, The Annals of Statistics **37**, 1, 246–270 (2009).

Tibshirani, R., “Regression shrinkage and selection via the lasso”, Journal of the Royal Statistical Society. Series B (Methodological) pp. 267–288 (1996).

Wang, H., M. Yang and J. Stufken, “Information-based optimal subdata selection for big data linear regression”, Under Review for Journal of the American Statistical Association (2017).