Divide-and-Conquer Information-Based Optimal Subdata Selection Algorithm [*]

HaiYing Wang
Department of Statistics, University of Connecticut
haiying.wang@uconn.edu

Abstract

The information-based optimal subdata selection (IBOSS) is a computationally efficient method to select informative data points from large data sets through processing full data by columns. However, when the volume of a data set is too large to be processed in the available memory of a machine, it is infeasible to implement the IBOSS procedure. This paper develops a divide-and-conquer IBOSS approach to solving this problem, in which the full data set is divided into smaller partitions to be loaded into the memory and then subsets of data are selected from each partitions using the IBOSS algorithm. We derive both finite sample properties and asymptotic properties of the resulting estimator. Asymptotic results show that if the full data set is partitioned randomly and the number of partitions is not very large, then the resultant estimator has the same estimation efficiency as the original IBOSS estimator. We also carry out numerical experiments to evaluate the empirical performance of the proposed method.
Keywords: Big data, D-optimality, Information matrix, Linear regression, Subdata

1 Introduction

In big data analysis, a common challenge is that available computing facilities are inadequate to meet the computational needs. To overcome this challenge, there are two fundamental approaches: one is the divide-and-conquer approach in which a large data set is divided into smaller partitions and results are calculated from each partition and then combined; the

[*]This work was supported by NSF grant 1812013, a UCONN REP grant, and a GPU grant from NVIDIA Corporation.

other approach is to use a subset of the full data as a surrogate, and use calculation results from the subdata to approximate the full data calculation results.

The divide-and-conquer approach naturally fits into parallel and distributed computing systems, and it is effective to deal with the challenge that the data volume is too large to be loaded into the memory although it may not always reduce the computing time with a single processor. Most distributed computational platforms are expensive to access and they are not suitable for daily routine usages such as exploratory data analysis and pilot model prototyping. The divide-and-conquer method has attracted many researchers in machine learning and statistics, leading to key advancements in Lin and Xie (2011); Jordan (2012); Chen and Xie (2014); Shang and Cheng (2017); Battey *et al.* (2018), among others. In the streaming setting where data blocks are accessible sequentially for only one time, the online updating method has been developed (Schifano *et al.*, 2016; Xue *et al.*, 2018). In order to produce an estimator that preserves the same convergence rate as the full data estimator, a key requirement for the divide-and-conquer method is that the number of partitions cannot be too large (Shang and Cheng, 2017).

The approach of using a subset of the full data is an effective method to reduce the computational burden, and it is often the only way to extract useful information from massive data sets when available computing power is limited. An advantage of this approach is that once a subset of data is taken, thorough analysis can often be performed on a regular computer such as a laptop. This is important in statistics because data analysis is not just computing. Note that the primary goal of selecting an informative subset from the full data agrees with the basic motivation of optimal design of experiments in which one wants to get as much information as possible with a fixed design budget. The major challenge of subdata selection from big data is how to identify informative data points computationally fast in order to maintain the computational advantage in the whole procedure of data analysis. For this purpose, statistical leverage scores are often used to define subsampling probabilities in linear regression (Drineas *et al.*, 2012; Ma *et al.*, 2015). In the context of logistic regression, Wang *et al.* (2018) derived the optimal subsampling probabilities that minimize the asymptotic variance of the subsampling estimator under the A- and L-optimality criteria, and Wang (2018) further developed a more efficient estimation approach based on optimal subsamples. Yao and Wang (2019) extended this technique to include the softmax regression.

The aforementioned work uses random subsampling to take subsets of full data, and Wang *et al.* (2019) showed that an estimator obtained from this approach for linear regression has a variance that is bounded from below by a term that is proportional to the inverse of the subset sample size, i.e., the estimator will not converge to the true parameter if the subset sample size is fixed no matter how fast the full data sample size goes to infinity.

From the characterization of the optimal subdata under the D-optimality criterion, they further develop a computationally efficient IBOSS algorithm. The resulting estimator from this algorithm converges to the true parameter even if the subdata sample size is fixed as long as the full data sample size gets large and the supports of the covariate distributions are unbounded. The IBOSS algorithm selects data points by examining each column of the design matrix, so it is infeasible to apply if the data volume is too large to be loaded into the random-access memory (RAM). This paper combines the divide-and-conquer method and the IBOSS algorithm to solve this issue. The basic approach is to divide a large data set into smaller partitions so that each partition can be loaded into the RAM and the IBOSS algorithm is applied on the smaller partitions. In practice, big data are seldom stored in a single file; they are often stored in multiple data blocks. The proposed method fits this scenario naturally. We will investigate both finite sample properties and asymptotic properties of this procedure, and use numerical experiments to evaluate its empirical performance.

The rest of the paper is organized as the following. We present notations for the model setup and introduce the IBOSS method in Section 2. The main theoretical results on the divide-and-conquer IBOSS algorithm will be presented in Section 3. Section 4 evaluates the empirical performance of the divide-and-conquer IBOSS algorithm using numerical examples. Section 5 concludes the paper and all proofs are provided in the Appendix.

2 IBOSS framework and detailed algorithm

Assume that the full data $\mathcal{D}_N = \{(\mathbf{z}_1, y_1), ..., (\mathbf{z}_N, y_N)\}$ satisfy the following linear regression model

$$y_i = \beta_0 + \sum_{j=1}^{p} z_{ij}\beta_j + \varepsilon_i, \qquad i = 1, ..., N, \tag{1}$$

where y_i are the responses, $\mathbf{z}_i = (z_{i1}, ..., z_{ip})^{\mathrm{T}}$ are the covariates, β_0 and $\beta_1, ..., \beta_p$ are the intercept and slope parameters, respectively, and ε_i are model errors. For easy of presentation, we define some notations. Let $\boldsymbol{x}_i = (1, \mathbf{z}_i)$ and $\mathcal{Z} = (\mathbf{z}_1^{\mathrm{T}}, ..., \mathbf{z}_N^{\mathrm{T}})^{\mathrm{T}}$. Denote $\boldsymbol{\beta}_1 = (\beta_1, \beta_2, ..., \beta_p)^{\mathrm{T}}$ and $\boldsymbol{\beta} = (\beta_0, \boldsymbol{\beta}_1^{\mathrm{T}})^{\mathrm{T}}$. Here we assume that the error terms ε_i are uncorrelated and satisfy $\mathbb{E}(\varepsilon_i) = 0$ and $\mathbb{V}(\varepsilon_i) = \sigma^2$. Without loss of generality, we assume that $\sigma^2 = 1$ when discussing the selection of subdata sets.

To estimate the unknown $\boldsymbol{\beta}$, the best linear unbiased estimator (BLUE) is the least squares estimator. With the full data $\mathcal{D}_N$, it has an expression of

$$\hat{\boldsymbol{\beta}}_{\text{full}} = \left(\sum_{i=1}^{N} \boldsymbol{x}_i\boldsymbol{x}_i^{\mathrm{T}}\right)^{-1} \sum_{i=1}^{N} \boldsymbol{x}_i y_i,$$

with variance-covariance matrix

$$\mathbb{V}(\hat{\boldsymbol{\beta}}_{\text{full}}|\mathcal{Z}) = \boldsymbol{M}_{\text{full}}^{-1} = \left(\sum_{i=1}^{N} \boldsymbol{x}_i \boldsymbol{x}_i^{\mathrm{T}}\right)^{-1},$$

where $\boldsymbol{M}_{\text{full}}$ is the observed Fisher information matrix of $\boldsymbol{\beta}$ for the full data if ε_i are normally distributed.

When the sample size N of the full data set is too large, full data analysis may take too long to afford. To extract some useful information from the data in time with limited computing resources, a subset of the full data can be selected and thoroughly analyzed. For this purpose, Wang *et al.* (2019) proposed the IBOSS method. We summarize the motivation and procedure in the following.

Use $\boldsymbol{\delta} = \{\delta_1, \delta_2, ..., \delta_N\}$ to indicate a subset of data where $\delta_i = 1$ if $(\mathbf{z}_i, y_i)$ is included in the subset and $\delta_i = 0$ otherwise. Here $\boldsymbol{\delta}$ may depend on $\mathcal{Z}$ but it does not depend on $\boldsymbol{y}$, so the subdata selection rule is ancillary to the regression parameter, and the least squares estimator is still the BLUE based on the subset of the full data. With this notation, the information matrix for $\boldsymbol{\beta}$ based on a subset of the full data indexed by $\boldsymbol{\delta}$ is

$$\boldsymbol{M}_{\boldsymbol{\delta}} = \sum_{i=1}^{N} \delta_i \boldsymbol{x}_i \boldsymbol{x}_i^{\mathrm{T}}.$$

To extract the maximum amount of information from the full data with a fixed subset sample size k so that $|\boldsymbol{\delta}| = \sum_{i=1}^{N} \delta_i = k$, Wang *et al.* (2019) proposed to select the subset of data that maximizes $\boldsymbol{M}_{\boldsymbol{\delta}}$ with respect to $\boldsymbol{\delta}$ under the the D-optimality criterion in optimal design of experiments, namely, to find

$$\boldsymbol{\delta}_{\mathrm{D}}^{opt} = \arg\max_{\boldsymbol{\delta}:|\boldsymbol{\delta}|=k} \left|\sum_{i=1}^{N} \delta_i \boldsymbol{x}_i \boldsymbol{x}_i^{\mathrm{T}}\right|.$$

In general, there is no analytical solution to $\boldsymbol{\delta}_{\mathrm{D}}^{opt}$, and numerical search for an exact solution is computationally NP-hard. Wang *et al.* (2019) derived an upper bound for $|\boldsymbol{M}_{\boldsymbol{\delta}}|$, and then proposed an algorithm to approximate it. To be specific, they showed that for any $\boldsymbol{\delta}$ with $|\boldsymbol{\delta}| = k$,

$$|\boldsymbol{M}_{\boldsymbol{\delta}}| \leq \zeta_N := \frac{k^{p+1}}{4^p} \prod_{j=1}^{p} (z_{(N)j} - z_{(1)j})^2, \tag{2}$$

where $z_{(N)j}$ and $z_{(1)j}$ are the sample maximum and sample minimum, respectively, for the

j-th covariate. Although the upper bound ζ_N is typically unachievable, it indicates that the more informative data points are in the tail regions of the covariates. This motivated the IBOSS algorithm to approximate the upper bound, and here is a summary of the IBOSS procedure. Assume that $r = k/(2p)$ is an integer. For each covariate, using a partition-based selection algorithm (Martínez, 2004), select r data points with the smallest z_{ij} values and r data points with the largest z_{ij} values among the data points that are not yet included in the subsample. The data points obtained according to the p covariates are then combined to form a subdata of size k for statistical analysis.

The original IBOSS paper by Wang *et al.* (2019) did not present all practical details of implementing the IBOSS procedure. For example, it is mentioned that if some data points have been included in the subdata by some covariate, the IBOSS algorithm requires to exclude them from consideration when using other covariates to select data points, but the implementation details were not provided. This step may add significant CPU time if not implemented appropriately. Thus, for completeness, we present a detailed IBOSS algorithm taking into account necessary practical considerations in Algorithm 1.

3 Divide-and-conquer IBOSS algorithm

The IBOSS approach in Algorithm 1 needs to look at the data column by column. Although for most programming languages, such as R (R Core Team, 2018) and Julia (Bezanson *et al.*, 2017), data matrices are stored by columns in the memory, data files are often stored on hard drive by rows. Thus, if the data volume exceeds the size of the available memory, it is difficult to implement the IBOSS algorithm. To deal with the problem of limited memory, a natural approach is the divide-and-conquer procedure, in which the full data set is divided into partitions by rows and then each partition can be loaded into the memory and processed individually. To combine this procedure with the IBOSS procedure, we present the divide-and-conquer IBOSS method in Algorithm 2.

Remark 1. Algorithm 2 takes a subdata $\mathcal{D}_S^{(b)}$ from each partition $\mathcal{D}_N^{(b)}$ and then combines them to have a final subdata $\mathcal{D}_{\text{BS}}$. This is recommended if only one machine is used, and the combined subdata should be small enough so that thorough analysis can be performed on the machine. If a distributed computing system is used and data partitions are processed in parallel by multiple machines, we can calculate the least squares estimators and their estimated variance-covariance matrices on each subdata $\mathcal{D}_S^{(b)}$ in all machines and then aggregate these estimates using a linear combination with the inverses of the variance-covariance matrices as weights. The final estimators from these two approaches are identical (Lin and Xie, 2011; Schifano *et al.*, 2016). The latter procedure takes advantage of parallel computing

Algorithm 1 Original IBOSS algorithm

```
Input: full data $\mathcal{D}_N$, subdata size $k$
Output: subdata set $\mathcal{D}_S$
$(N, p) = \text{size}(\mathcal{Z})$    {get the full data size and covariate dimension}
$\boldsymbol{S} = \emptyset$    {initialize the index set of the subdata}
$r = \lceil k/(2p) \rceil$    {number of data points to take at each covariate tail}
for $j$ in $1, ..., p$ do
  $\boldsymbol{S} = \text{sort}(\boldsymbol{S})$    {sort elements in $\boldsymbol{S}$ in an increasing order}
  $\boldsymbol{u} = (u_1, ..., u_{N-|\boldsymbol{S}|})^{\mathrm{T}}$    {allocate $\boldsymbol{u}$ to store elements of $\mathbf{z}_j$, $z_{ij}, i \notin \boldsymbol{S}$}
  $t = 1$, $s = 1$    {$t$ is the counter for $\boldsymbol{u}$; $s$ is the counter for $\boldsymbol{S}$}
  for $i$ in $1, ..., N$ do
    if $i \neq \boldsymbol{S}[s]$ then
      $u_t = z_{ij}$, $t = t + 1$    {record $z_{ij}$ in $\boldsymbol{u}$ if $i \notin \boldsymbol{S}$}
    else if $s < |\boldsymbol{S}|$ then
      $s = s + 1$    {skip $i$ if it is already in $\boldsymbol{S}$}
    end if
  end for
  $l = u_{(r)}$, $q = u_{(N-r+1)}$    {using partition based quick selection}
  $s = 1$, $r_l = 1$, $r_q = 1$    {$r_l$, $r_q$ count numbers of data points in each tail}
  for $i$ in $1, ..., N$ do
    if $|\boldsymbol{S}| \geq k$ or $(r_l > r$ and $r_q > r)$ then
      break    {break when enough data points are taken}
    else if $s \leq |\boldsymbol{S}|$ and $i = \boldsymbol{S}[s]$ then
      $s = s + 1$    {skip $i$ if it is already in $\boldsymbol{S}$}
    else if $r_l \leq r$ and $z_{ij} \leq l$ then
      $\boldsymbol{S} = \boldsymbol{S} \cup i$, $r_l = r_l + 1$    {include one data point from the left tail}
    else if $r_q \leq r$ and $z_{ij} \geq q$ then
      $\boldsymbol{S} = \boldsymbol{S} \cup i$, $r_q = r_q + 1$    {include one data point from the right tail}
    end if
  end for
end for
$\boldsymbol{S}$=sort($\boldsymbol{S}$)
$\mathcal{D}_S = \emptyset$    {initialize the set of the subdata}
$t = 1$, $s = 1$
for $i$ in $1, ..., N$ do
  if $s \leq |\boldsymbol{S}|$ and $i = \boldsymbol{S}[s]$ then
    $\mathcal{D}_S = \mathcal{D}_S \cup (\mathbf{z}_i, y_i)$, $s = s + 1$    {take the subdata according to $\boldsymbol{S}$}
  end if
end for
return $\mathcal{D}_S$
```

Algorithm 2 Divide-and-conquer IBOSS algorithm

```
Input: full data $\mathcal{D}_N$, subdata size $k$, partition size $n_B$.
Output: subdata set $\mathcal{D}_{\text{BS}}$
if $r_B = k/(2pB) < 1$ then
    print "The number of data points from each covariate tail is smaller than one."
end if
$(\mathcal{D}_N^{(1)}, ..., \mathcal{D}_N^{(B)}) = \text{split}(\mathcal{D}_N, n_B)$    {Divide the full data into $B$ partitions of size $n_B$. This step can be skipped if the full data are stored in multiple small files.}
$\mathcal{D}_{\text{BS}} = \emptyset$
for $b$ in $1, ..., B$ do
    Run Algorithm 1 on $\mathcal{D}_N^{(b)}$ with $k_b = \lceil k/B \rceil$ to obtain $\mathcal{D}_S^{(b)}$
    $\mathcal{D}_{\text{BS}} = \mathcal{D}_{\text{BS}} \cup \mathcal{D}_S^{(b)}$
end for
return $\mathcal{D}_{\text{BS}}$
```

facilities, but it has to carry out additional calculations on each machine if further regression diagnostics are to be performed.

Remark 2. Using one machine to implement Algorithm 2, the step to divide the full data $\mathcal{D}_N$ into blocks $\mathcal{D}_N^{(1)}$, ..., $\mathcal{D}_N^{(B)}$ can be done by using the UNIX command `split`. This command is also available for Windows through Cygwin.

3.1 Theoretical properties

In this section, we investigate the performance of Algorithm 2 theoretically. With out loss of generality, we assume that $r_B = r/B = k/(2pB)$ and $n_B = N/B$ are both integers, where B is the number of partitions on the full data.

Let $z_{(i)j}$ $(i = 1, ..., N; j = 1, ..., p)$ be the i-th order statistic on the j-th covariate in the full data and $z_{b(i)j}$ $(b = 1, ..., B; i = 1, ..., n_B; j = 1, ..., p)$ be the i-th order statistics on the j-th covariate in the b-th block. Denote the covariate matrix for the subdata $\mathcal{D}_{\text{BS}}$ by $\mathcal{Z}^*_{\mathcal{D}_{\text{BS}}}$ and denote the sample correlation matrix of $\mathcal{Z}^*_{\mathcal{D}_S}$ by $\mathbf{R}_{\mathcal{D}_{\text{BS}}}$. Let $\mathcal{X}^*_{\mathcal{D}_{\text{BS}}} = (\mathbf{1}, \mathcal{Z}^*_{\mathcal{D}_{\text{BS}}})$.

For the divide-and-conquer IBOSS method, we first present a result showing the quality of using Algorithm 2 to approximate ζ_N, a typically unachievable upper bound of $|\boldsymbol{M}_{\boldsymbol{\delta}}|$ defined in (2).

Theorem 1. *The subdata $\mathcal{D}_{\text{BS}}$ selected using Algorithm 2 satisfies that*

$$\frac{|(\mathcal{X}^*_{\mathcal{D}_{\text{BS}}})^{\text{T}} \mathcal{X}^*_{\mathcal{D}_{\text{BS}}}|}{\zeta_N} \geq \max(C_R, C_E), \tag{3}$$

where

$$C_R = \frac{\lambda_{\min}^p(\boldsymbol{R}_{\mathcal{D}_{\mathrm{BS}}})}{(Bp)^p} \prod_{j=1}^{p} \left\{ \sum_{b=1}^{B} \left(\frac{z_{b(n_B - r_B + 1)j} - z_{b(r_B)j}}{z_{(N)j} - z_{(1)j}} \right)^2 \right\}, \tag{4}$$

$$C_E = \frac{\lambda_{\min}^p(\boldsymbol{R}_{\mathcal{D}_{\mathrm{BS}}})}{(Bp)^p} \times \prod_{j=1}^{p} \left(\frac{z_{(N - r_B + 1)j} - z_{(r_B)j}}{z_{(N)j} - z_{(1)j}} \right)^2, \tag{5}$$

and $\lambda_{\min}(\mathbf{R}_{\mathcal{D}_{\mathrm{BS}}})$ *is the smallest eigenvalue of* $\mathbf{R}_{\mathcal{D}_{\mathrm{BS}}}$.

Remark 3*.* The lower bound of the approximation ratio in (3) has two terms: C_R in (4) and C_E in (5). Heuristically, C_R is a bound corresponding to the scenario that the full data is divided randomly so that the covariate distributions for different partitions are the same. In this scenario C_R is a sharper bound compared with C_E, i.e., $C_R > C_E$. On the other hand, C_E is a bound corresponding to some extreme cases, e.g., ranges of covariate values for all blocks do not overlap. In this case, C_E may be larger than C_R.

Theorem 1 is aligned with Theorem 3 of Wang *et al.* (2019) for the original IBOSS algorithm, which shows that for the subdata $\mathcal{D}_S$ obtained from Algorithm 1, the following inequality holds.

$$\frac{|(\mathcal{X}_{\mathcal{D}_S}^*)^{\mathrm{T}} \mathcal{X}_{\mathcal{D}_S}^*|}{\zeta_N} \geq \frac{\lambda_{\min}^p(\mathbf{R}_{\mathcal{D}_S})}{p^p} \prod_{j=1}^{p} \left(\frac{z_{(N-r+1)j} - z_{(r)j}}{z_{(N)j} - z_{(1)j}} \right)^2, \tag{6}$$

where $\mathcal{X}_{\mathcal{D}_S}^* = (\mathbf{1}, \mathcal{Z}_{\mathcal{D}_S}^*)$, $\mathcal{Z}_{\mathcal{D}_S}^*$ is the covariate matrix for the subdata $\mathcal{D}_S$, and $\lambda_{\min}^p(\mathbf{R}_{\mathcal{D}_S})$ is the minimum eigenvalue of $\mathbf{R}_{\mathcal{D}_S}$, the correlation matrix of $\mathcal{Z}_{\mathcal{D}_S}^*$. Comparing (3) and (6), we see that there may be some information loss due to the divide-and-conquer procedure because $(z_{b(n_B - r_B + 1)j} - z_{b(r_B)j})^2$ and $(z_{(N - r_B + 1)j} - z_{(r_B)j})^2/B$ are typically smaller than $(z_{(N-r+1)j} - z_{(r)j})^2$. However, if B is not too large, $z_{b(n_B - r_B + 1)j} - z_{b(r_B)j}$ and $z_{(N-r+1)j} - z_{(r)j}$ are on the same order under reasonable assumptions, so the lower bounds of the approximation ratios are at the same order. Theorem 1 also indicates that $|(\mathcal{X}_{\mathcal{D}_{\mathrm{BS}}}^*)^{\mathrm{T}} \mathcal{X}_{\mathcal{D}_{\mathrm{BS}}}^*|$ may achieve the same order of the unachievable upper bound ζ_N under some reasonable assumptions if B and p do not go to infinity.

Now we investigate the properties of the resulting estimator form Algorithm 2. Let $\hat{\boldsymbol{\beta}}^{\mathcal{D}_{\mathrm{BS}}} = (\hat{\beta}_0^{\mathcal{D}_{\mathrm{BS}}}, \hat{\beta}_1^{\mathcal{D}_{\mathrm{BS}}}, ..., \hat{\beta}_p^{\mathcal{D}_{\mathrm{BS}}})^{\mathrm{T}}$ be the least squares estimator calculated from the divide-and-conquer IBOSS subdata, namely,

$$\hat{\boldsymbol{\beta}}^{\mathcal{D}_{\mathrm{BS}}} = \left(\sum_{b=1}^{B} \sum_{i=1}^{n_B} \boldsymbol{x}_{bi}^* \boldsymbol{x}_{bi}^{*\mathrm{T}} \right)^{-1} \left(\sum_{b=1}^{B} \sum_{i=1}^{n_B} \boldsymbol{x}_{bi}^* y_{bi}^* \right),$$

where $\boldsymbol{x}_{bi}^* = (1, \mathbf{z}_{bi}^*)^{\mathrm{T}}$ and $(\mathbf{z}_{bi}^*, y_{bi}^*)$ is the i-th observation in the subdata $\mathcal{D}_S^{(b)}$ from the b-th block. To investigate the theoretical properties of $\hat{\boldsymbol{\beta}}^{\mathcal{D}_{\mathrm{BS}}}$, we focus on the variances because $\hat{\boldsymbol{\beta}}^{\mathcal{D}_{\mathrm{BS}}}$ is unbiased for $\boldsymbol{\beta}$.

The following theorem provides finite sample bounds on variances of $\hat{\beta}_j^{\mathcal{D}_{\mathrm{BS}}}$, $j = 0, 1, ..., p$. No quantity is required to go to infinity here.

Theorem 2. *If the sample correlation matrix for covariates $\mathcal{Z}_{\mathcal{D}_{\mathrm{BS}}}^*$ in the subdata $\mathcal{D}_{\mathrm{BS}}$ is column full rank, then the following results hold for the estimator, $\hat{\boldsymbol{\beta}}^{\mathcal{D}_{\mathrm{BS}}}$, obtained from Algorithm 2:*

$$\mathbb{V}(\hat{\beta}_0^{\mathcal{D}_{\mathrm{BS}}} | \mathcal{Z}) \geq \frac{\sigma^2}{k}, \quad \textit{and} \tag{7}$$

$$\frac{4\sigma^2}{k(z_{(N)j} - z_{(1)j})^2} \leq \mathbb{V}(\hat{\beta}_j^{\mathcal{D}_{\mathrm{BS}}} | \mathcal{Z}) \leq \min(V_{aj}, V_{ej}), \tag{8}$$

with probability one for $j = 1, ..., p$, where

$$V_{aj} = \frac{4pB\sigma^2}{k\lambda_{\min}(\boldsymbol{R}_{\mathcal{D}_{\mathrm{BS}}}) \sum_{b=1}^{B} \left(z_{b(n_B - r_B + 1)j} - z_{b(r_B)j}\right)^2},$$
$$V_{ej} = \frac{4pB\sigma^2}{k\lambda_{\min}(\boldsymbol{R}_{\mathcal{D}_{\mathrm{BS}}})(z_{(N - r_B + 1)j} - z_{(r_B)j})^2},$$

and $\lambda_{\min}(\boldsymbol{R}_{\mathcal{D}_{\mathrm{BS}}})$ is the minimum eigenvalue of the sample correlation matrix of $\mathcal{Z}_{\mathcal{D}_{\mathrm{BS}}}^*$.

Theorem 2 shows that for the intercept, the variance of the estimator $\hat{\beta}_0^{\mathcal{D}_{\mathrm{BS}}}$ is bounded from below by σ^2/k, which is the same as the bound for the original IBOSS algorithm. It indicates that this variance cannot go to zero for a fixed k. However, for slope parameters, the variances of the estimators $\hat{\beta}_j^{\mathcal{D}_{\mathrm{BS}}}$ may converge to zero under some assumptions even k is fixed.

Theorem 2 is aligned with Theorem 4 of Wang *et al.* (2019), which shows that the variances of the slope estimators from the original IBOSS procedure are bounded from above by

$$V_{oj} = \frac{4p\sigma^2}{k\lambda_{\min}(\mathbf{R}_{\mathcal{D}_S})(z_{(N-r+1)j} - z_{(r)j})^2}, \quad j = 1, ..., p.$$

Comparing the upper bounds V_{aj} and V_{ej} with the upper bounds V_{oj} for the original IBOSS procedure, we also see that the divide-and-conquer IBOSS algorithm may subject to some information loss. However, the information loss can be ignored asymptotically if the full data is partitioned randomly and the number of partitions does not go to infinity too fast. We derive the following theorem showing that the orders of $\mathbb{V}(\hat{\beta}_j^{\mathcal{D}_{\mathrm{BS}}} | \mathcal{Z})$ can be the same as those of the variances for the original IBOSS estimators.

Theorem 3. *Assume that covariate distributions are in the domain of attraction of the generalized extreme value distribution, and* $\liminf_{N\to\infty}\lambda_{\min}(\boldsymbol{R}_{\mathcal{D}_{\mathrm{BS}}}) > 0$. *For large enough* N, *when the full data is divided randomly in Algorithm 2, the following results hold for the estimator,* $\hat{\beta}_j^{\mathcal{D}_{\mathrm{BS}}}$, $j = 1, ..., p$.

$$\mathbb{V}(\hat{\beta}_j^{\mathcal{D}_{\mathrm{BS}}}|\mathcal{Z}) = O_P\left\{\frac{p}{k(z_{(N)j} - z_{(1)j})^2}\right\}, \quad j = 1, ..., p,$$

if one of the following conditions holds: 1) r *and* B *are fixed; 2) the support of* F_j *is bounded and* $r/N \to 0$, *where* F_j *is the marginal distribution function of the* j*-th component of* $\mathbf{z}$; *3) the upper endpoint for the support of* F_j *is* ∞ *and the lower endpoint for the support of* F_j *is finite, and*

$$\frac{r}{N[1 - F_j\{(1-\epsilon)F_j^{-1}(1-N^{-1})\}]} \to 0 \quad \text{and} \quad \frac{F_j(1-1/N)}{F_j(1-B/N)} \to 1, \tag{9}$$

for all $\epsilon > 0$; *4) the upper endpoint for the support of* F_j *is finite and the lower endpoint for the support of* F_j *is* $-\infty$, *and*

$$\frac{r}{NF_j\{(1-\epsilon)F_j^{-1}(N^{-1})\}} \to 0, \quad \text{and} \quad \frac{F_j(1/N)}{F_j(B/N)} \to 1, \tag{10}$$

for all $\epsilon > 0$; *and 5) the upper endpoint and the lower endpoint for the support of* F_j *are* ∞ *and* $-\infty$, *respectively, and both* (9) *and* (10) *hold.*

The convergence rates of $\mathbb{V}(\hat{\beta}_j^{\mathcal{D}_{\mathrm{BS}}}|\mathcal{Z})$ described in Theorem 3 are the same as those given in Theorem 5 of Wang *et al.* (2019) for the original IBOSS estimators of slope parameters.

For case 1), if p is fixed, then the total subdata size k is also fixed. As long as the supports of covariate distributions are not bounded, $z_{(N-r+1)j} - z_{(r)j} \to \infty$ in probability as $n \to \infty$, and thus the variances $\mathbb{V}(\hat{\beta}_j^{\mathcal{D}_{\mathrm{BS}}}|\mathcal{Z})$ converge to zero in probability. This is not the case for random subsampling-based methods (Ma *et al.*, 2015; Wang *et al.*, 2018). In this case, the asymptotic expression for the variance-covariance matrix of $\hat{\boldsymbol{\beta}}^{\mathcal{D}_{\mathrm{BS}}}$ when $\mathbf{z}_i$ follow normal and lognormal distributions are identical to those in Theorem 6 of Wang *et al.* (2019). For case 2), $z_{(N-r+1)j} - z_{(r)j}$ goes to some finite constant, so it is necessary that $k \to \infty$ in order for $\mathbb{V}(\hat{\beta}_j^{\mathcal{D}_{\mathrm{BS}}}|\mathcal{Z})$ to converge to zero. For case 3), the condition in (9) impose constrains on r, B, N, and the tail behavior of the covariate distributions. For example, if

$$F_j(z) = 1 - \exp\{-z^{\gamma}h(z)\}, \tag{11}$$

where $\gamma > 0$ and $h(z)$ is a slowly varying function, then the first constrain in (9) is to require that $\log r/\log N \to 0$ (Hall, 1979). The distribution in (11) includes many commonly seen distributions, such as exponential distribution, gamma distribution, Gumbel distribution, Laplace distribution, and normal distribution. For these distributions, if $\log B/\log N \to 0$, then the second constrain in (9) also holds. For case 4), the condition in (10) is essentially the same as that in (9) if one takes transformation $\mathbf{z} = -\mathbf{z}$. For case 5, it is a combination of cases 3 and 4. Under the given conditions, $F_j^{-1}(1-N^{-1})/z_{(N)j} \to 1$ and $F_j^{-1}(N^{-1})/z_{(1)j} \to 1$, so the rates for $\mathbb{V}(\hat{\beta}_j^{\mathcal{D}_{\mathrm{BS}}}|\mathcal{Z})$ to converge to zero are

$$\frac{p}{k\{F_j^{-1}(1-N^{-1}) - F_j^{-1}(N^{-1})\}^2}, \quad j = 1, ..., p.$$

4 Numerical experiments

In this section, we use numerical experiments to evaluate the empirical performance of the divide-and-conquer IBOSS algorithm.

We generate data from model (1) with $\beta_0 = 1$ and $\boldsymbol{\beta}_1$ being a 50 dimensional vector of ones. We assume that ε_i are independent following $N(0, 1)$. The following five cases are considered to generate the covariate matrix $\mathcal{Z}$.

Case 1: **Normal**. We generate $\mathcal{Z}$ from a multivariate normal distribution, $N(\mathbf{0}, \mathbf{\Sigma})$, where the (i, j)-th element of $\mathbf{\Sigma}$ is 0.5 if $i \neq j$ and 1 otherwise.

Case 2: **LogNormal**. We generate $\mathcal{Z}$ from a multivariate lognormal distribution, namely, generate $\mathcal{V}$ from $N(\mathbf{0}, \mathbf{\Sigma})$ as defined in Case 1 and then set $\mathcal{Z} = e^{\mathcal{V}}$, where the exponentiation is element-wise.

Case 3: $\boldsymbol{T_2}$. We generate $\mathcal{Z}$ from a multivariate t distribution with two degrees of freedom $t_2(\mathbf{0}, \mathbf{\Sigma})$ with $\mathbf{\Sigma}$ being the same as in Case 1.

Case 4: **Mix with order**. We generate $\mathcal{Z}$ from five different distributions. Specifically, generate $\mathcal{Z}_1$ from $N(\mathbf{0}, \mathbf{\Sigma})$; generate $\mathcal{Z}_2$ from $t_2(\mathbf{0}, \mathbf{\Sigma})$; generate $\mathcal{Z}_3$ from $t_3(\mathbf{0}, \mathbf{\Sigma})$; generate $\mathcal{Z}_4$ with its elements independently following the same uniform distribution between zero and two; and generate $\mathcal{Z}_5$ from the multivariate lognormal distribution defined in Case 2. Here, $\mathcal{Z}_i$ $(i = 1, ..., 5)$ all contain $n/5$ rows, and $\mathbf{\Sigma}$ is the same as defined in Case 1. Set $\mathcal{Z} = (\mathcal{Z}_1^{\mathrm{T}}, \mathcal{Z}_2^{\mathrm{T}}, \mathcal{Z}_3^{\mathrm{T}}, \mathcal{Z}_4^{\mathrm{T}}, \mathcal{Z}_5^{\mathrm{T}})^{\mathrm{T}}$.

Case 5: **Mix random order**. We generate $\mathcal{Z}$ using the same procedure as in Case 4, and then we randomize the row orderings.

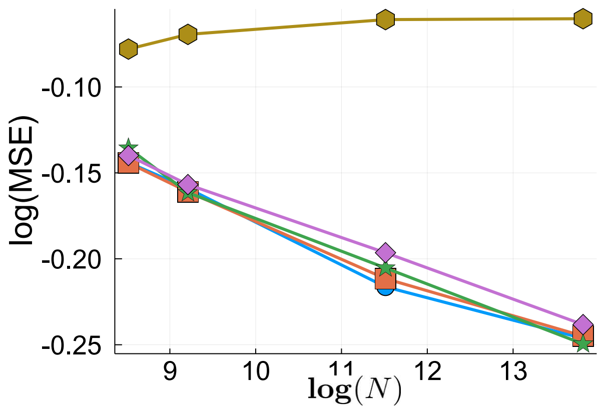
Here, Cases 1-4 are the same as those in Wang *et al.* (2019). Cases 4 and 5 would produce identical results for the full data analysis approach but produce different results for the divide-and-conquer IBOSS algorithm as different row orderings will affect the data partitions. The purpose of considering both Cases 4 and 5 is to investigate the effect of different data partitions on the the divide-and-conquer IBOSS algorithm. We divide the full data according to the sequence of row numbers in the numerical experiments.

We repeat the simulation for $T = 1000$ times to calculate the empirical mean squared error (MSE) as MSE=$T^{-1}\sum_{t=1}^{T}\|\boldsymbol{\beta}_1^{(t)} - \boldsymbol{\beta}_1\|^2$ for the slope parameter, where $\boldsymbol{\beta}_1^{(t)}$ is the estimate at the t-th repetition.

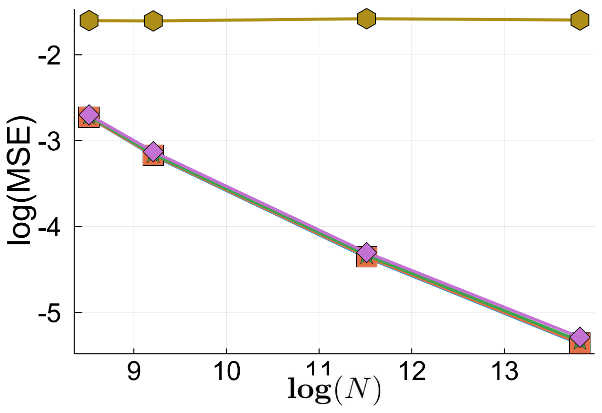
Figure 1 plots log of empirical MSEs against full data sample size N of $N = 5 \times 10^3, 10^4, 10^5$ and 10^6, with total subdata size $k = 1000$. Here the natural logarithm is taken to have better presentations. For the number of partitions B, we considered four values: $B = 1, 2, 5,$ and 10, and the corresponding values of r_B are $r_B = 10, 5, 2,$ and 1, respectively. Note that when $B = 1$, the method is the original IBOSS approach. For comparison, the uniform Poisson subsampling method is also implemented. From Figure 1, the empirical MSE decreases as N gets large for the divide-and-conquer IBOSS approach, while it stays constant for the random sampling approach. Different values of B have some effect on the performance of divide-and-conquer IBOSS approach, the effect is not very significant, especially for Cases 2, 3, and 5. Comparing results in Cases 4 and 5, we see that different row orderings affect the performance of the divide-and-conquer IBOSS algorithm, and the random row ordering (random partitions) is preferable.

We also consider the scenario that r_B is fixed while B changes. This is to mimic the scenario that one may have B machines to use while the computational capacity for each machine is limited. This is often the case for parallel distributed computing systems. We set $r_B = 5$ and choose B to be $B = 1$, 2, 5, 10, and 20, which gives the values of k as $k = 500$, 1000, 2500, 5000, and 10000, respectively. We fixed $N = 10^6$. Figure 2 presents the results. It is seen that both the divide-and-conquer method and the Poisson subsampling method improve as B increases because the total subdat size k increases. The divide-and-conquer method dominates the Poisson subsampling method and the difference in their performance depends on the covariate distribution.

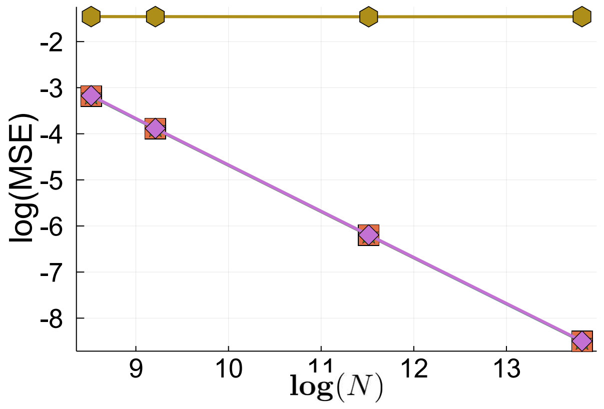
To further explore the effect of larger number of partitions B with a fixed $k = 2000$, we reduce the value of p to be $p = 2$ so that the maximum value of B is $B = 500$ corresponding to $r_B = 1$. We consider $B = 1$, 2, 5, 10, 20, 50, 100, 125, 250, and 500, which gives values of r_B as $r_B = 500$, 250, 100, 50, 25, 10, 5, 4, 2, and 1, respectively. Figure 3 gives the results. We see that the effect of B on the performance of the divide-and-conquer IBOSS algorithm is small compared with its significant improvement relative to the Poisson subsampling method.



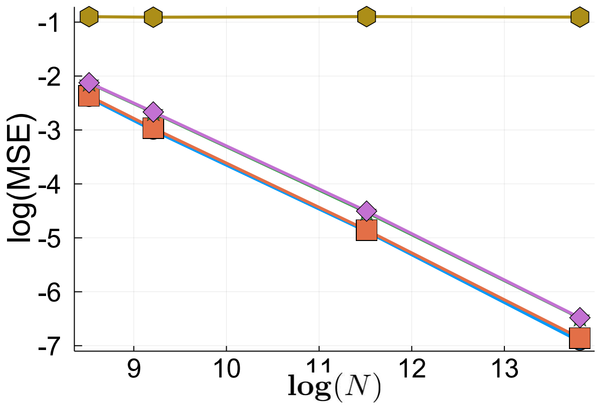
(a) Case 1: **Normal**



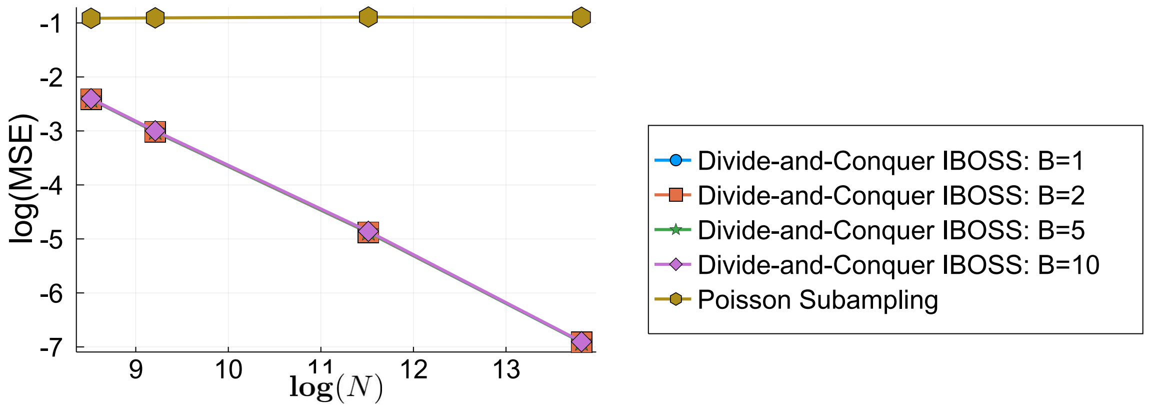
(b) Case 2: **LogNormal**



(c) Case 3: $\boldsymbol{T_2}$

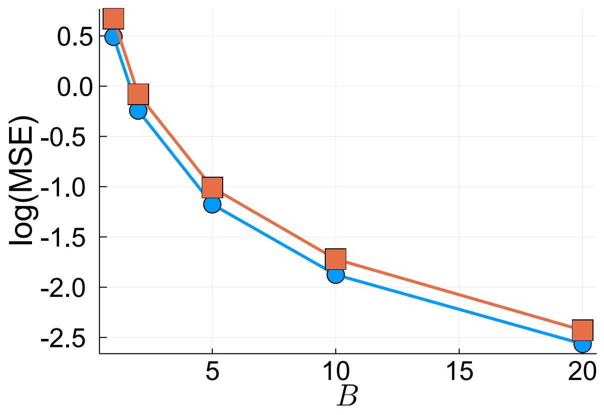


(d) Case 4: **Mix with order**

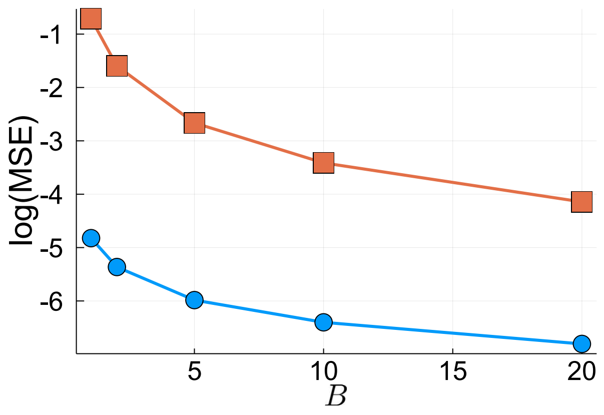


(e) Case 5: **Mix random order**

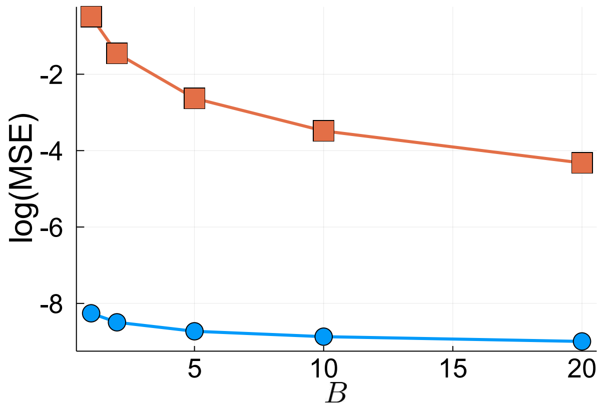
Figure 1: MSE of the slope parameter against full data sample size for the five cases of covariate distributions. The subdata size k is fixed at $k = 1000$.



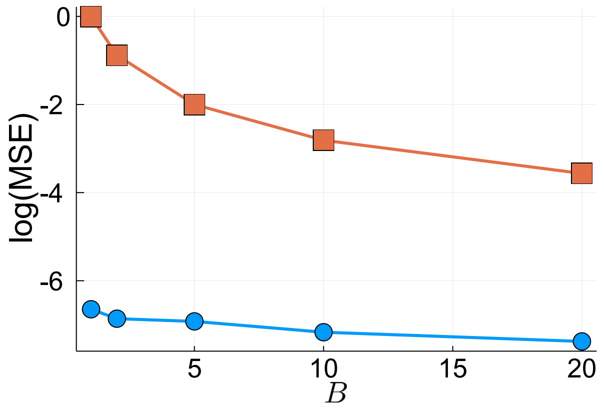
(a) Case 1: **Normal**



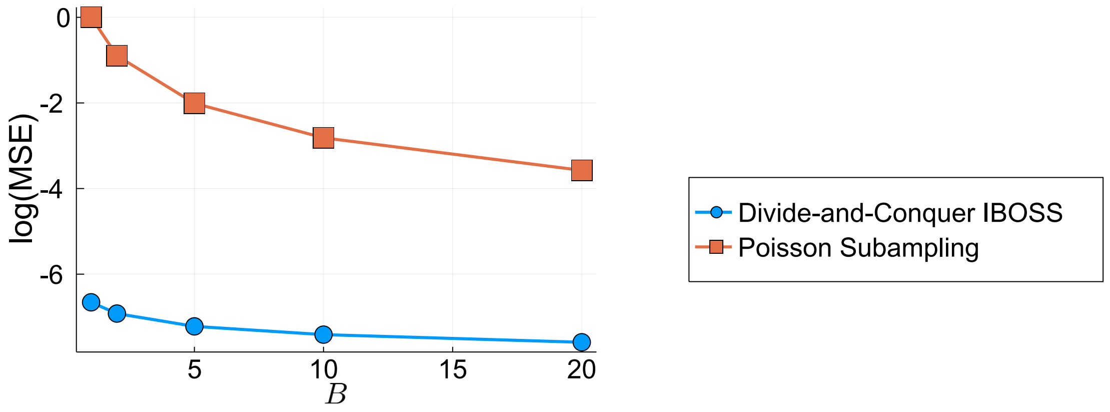
(b) Case 2: **LogNormal**



(c) Case 3: $\boldsymbol{T_2}$



(d) Case 4: **Mix with order**



(e) Case 5: **Mix random order**

Figure 2: MSE of the slope parameter against number of partitions B with a fixed r_B. The subdata size k changes as B changes.

Table 1: Computing times for different number of partitions B with $N = 10^7$, $p = 10^2$, and $k = 10^4$. "FULL" uses all the observations in each data block; "IBOSS" uses the IBOSS algorithm to select a subdata set of k/B observations; and "UNI uses Poisson subsampling to select a subdata set of k/B observations.

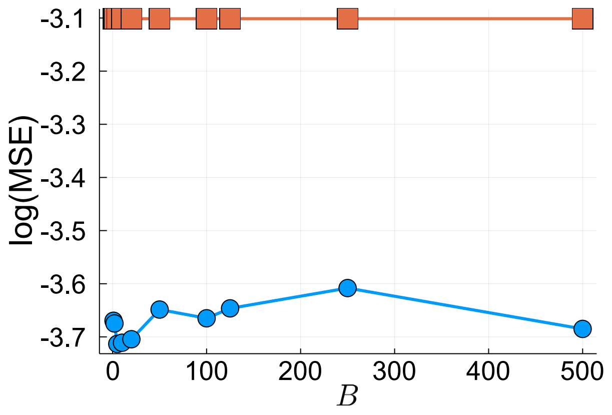
Methods	$B = 2$	$B = 5$	$B = 10$	$B = 25$	$B = 50$
FULL	98.80	95.76	90.82	86.41	85.28
IBOSS	29.74	24.90	20.79	18.81	18.42
UNI	6.81	5.67	4.32	6.75	14.08

Additionally, there is no clear increasing pattern for the empirical MSE as B gets large. This is different from the observations for regular cases in the existing literature where the estimation efficiency of the divide-and-conquer estimators often decreases as the number of partitions increases (e.g. Lin and Xie, 2011; Schifano *et al.*, 2016; Battey *et al.*, 2018).

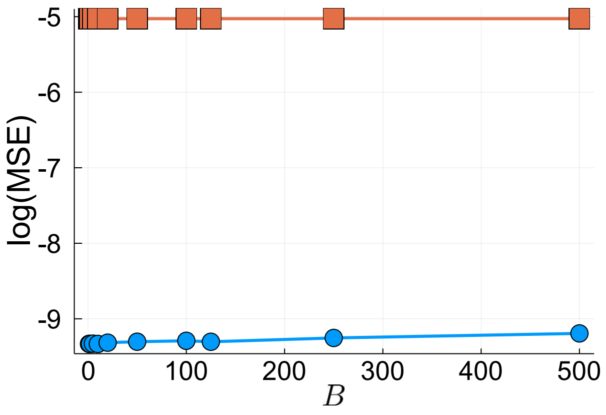
We investigate the computing times of the divide-and-conquer IBOSS algorithm for the case that data blocks are stored on hard drive. We set $N = 10^7$, $p = 100$, $k = 10^4$, and choose values of B to be $B = 2, 5, 10, 25$, and 50, giving values of $r_B = 25, 10, 5, 2$, and 1, respectively. For comparisons, we implement the full data divide-and-conquer approach in which all observations in each block of data are used to calculate least squares estimators and these estimators are then aggregated to form a full data estimator. We also implement the Poisson subsampling method on each data block to obtain subdata sets and then combine these subdata sets to calculate an estimate. Table 1 presents the results on computing times, where the time to load each data block into the memory is also counted. The calculation was performed on a computer with Intel Core i7 processors, a 64 GB RAM, and SSD hard drives, running Ubuntu 18.04 Linux system. Only one CPU was used for fair comparisons. It is seen that the divide-and-conquer IBOSS algorithm is faster than the full data approach, but is not as fast as the the uniform subsampling approach. For the divide-and-conquer IBOSS approach and the full data approach, as the number of blocks increases, the computing times are reduced. However, this is not the case for the uniform subsampling approach, for which the computing time decreases first but then increases as B increases. This is because as B gets large, the data loading time becomes the dominating term for the time required by the divide-and-conquer uniform Poisson subsampling method.

5 Concluding remarks

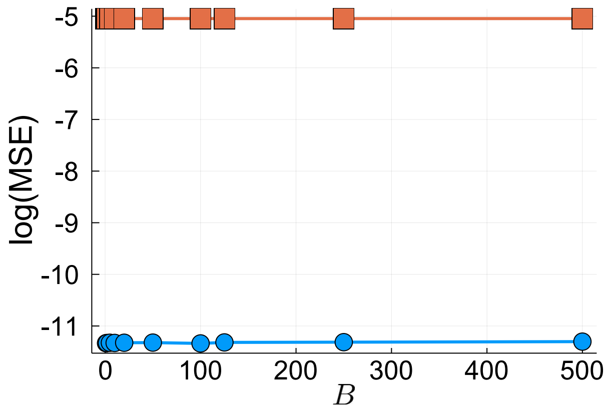
In this paper, we have developed a divide-and-conquer IBOSS method in the context of linear regression to solve the issue that the data volume is too large to apply the D-optimality motivated IBOSS method in memory. We have derived theoretical guarantees for the resul-



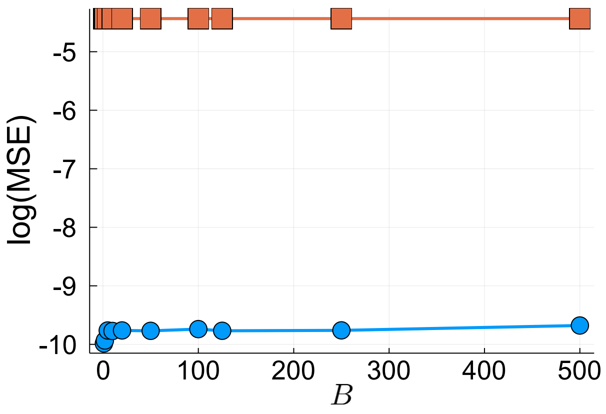
(a) Case 1: **Normal**



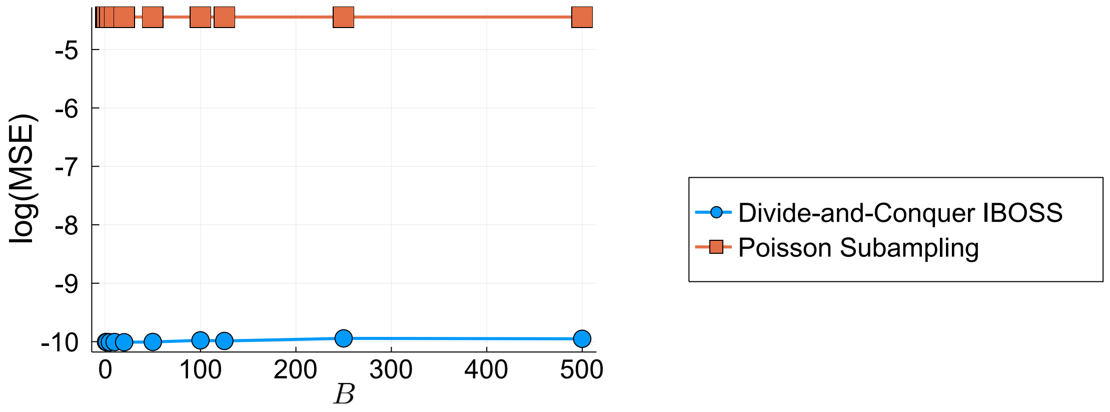
(b) Case 2: **LogNormal**



(c) Case 3: $\boldsymbol{T_2}$



(d) Case 4: **Mix with order**



(e) Case 5: **Mix random order**

Figure 3: MSE of the slope parameter against numbers of partitions B with a fixed $k = 2000$ for the five cases of covariate distributions. Here, r_B changes as B changes.

tant estimator and drawn comparisons with the original IBOSS approaches. We have also carried out numerical experiments to evaluate and demonstrate the proposed method.

We found that the estimation efficiency of the divide-and-conquer IBOSS algorithm may not be a monotonic function of the number of partitions as in other divide-and-conquer methods, which indicates the possibility of a nontrivial optimal number of partitions. This is an interesting and challenging question warrants for further investigations.

6 Proofs

6.1 Proof of Theorem 1

Proof. For $l \neq j$, let $z_j^{(i)l}$ be the concomitant of $z_{(i)l}$ for z_j, i.e., if $z_{(i)l} = z_{sl}$ then $z_j^{(i)l} = z_{sj}$, $i = 1, ..., N$. Let $\bar{z}^*_{j\mathcal{D}_{\mathrm{BS}}}$ and $\mathrm{var}(z^*_{j\mathcal{D}_{\mathrm{BS}}})$ be the sample mean and sample variance for covariate z_j of subdata $\mathcal{D}_{\mathrm{BS}}$. For the proof of Theorem 3 in Wang *et al.* (2019), we know that

$$|(\mathcal{X}^*_{\mathcal{D}_{\mathrm{BS}}})^{\mathrm{T}}\mathcal{X}^*_{\mathcal{D}_{\mathrm{BS}}}| \geq k(k-1)^p \lambda^p_{\min}(\boldsymbol{R}_{\mathcal{D}_{\mathrm{BS}}}) \prod_{j=1}^{p} \mathrm{var}(z^*_{j\mathcal{D}_{\mathrm{BS}}}). \tag{12}$$

Let $k_B = k/B$ be the number of data points to take from each partition, and z^*_{bij} be the i-th observation on the j-th covariate in the subdata $\mathcal{D}_S^{(b)}$ from the b-th partition. The sample variance for each j satisfies,

$$\begin{aligned}
&(k-1)\mathrm{var}(z^*_{j\mathcal{D}_{\mathrm{BS}}}) = \sum_{i=1}^{k} \left(z^*_{ij} - \bar{z}^*_{j\mathcal{D}_{\mathrm{BS}}}\right)^2 = \sum_{b=1}^{B}\sum_{i=1}^{k_B} \left(z^*_{bij} - \bar{z}^*_{j\mathcal{D}_{\mathrm{BS}}}\right)^2 \\
\geq & \sum_{b=1}^{B} \left(\sum_{i=1}^{r_B} + \sum_{i=n_B-r_B+1}^{n_B}\right) \left(z_{b(i)j} - \bar{z}^{**}_{bj}\right)^2 \\
= & \sum_{b=1}^{B} \left\{ \sum_{i=1}^{r_B} \left(z_{b(i)j} - \bar{z}^{*l}_{bj}\right)^2 + \sum_{i=n_B-r_B+1}^{n_B} \left(z_{b(i)j} - \bar{z}^{*u}_{bj}\right)^2 + \frac{r_B}{2}\left(\bar{z}^{*u}_{bj} - \bar{z}^{*l}_{bj}\right)^2 \right\} \\
\geq & \frac{r_B}{2} \sum_{b=1}^{B} \left(\bar{z}^{*u}_{j} - \bar{z}^{*l}_{j}\right)^2 \geq \frac{r_B}{2} \sum_{b=1}^{B} \left(z_{b(n_B-r_B+1)j} - z_{b(r_B)j}\right)^2,
\end{aligned}$$

where $\bar{z}^{**}_{bj} = \left(\sum_{i=1}^{r_B} + \sum_{i=n_B-r_B+1}^{n_B}\right) z_{b(i)j}/(2r_B)$, $\bar{z}^{*l}_{bj} = \sum_{i=1}^{r_B} z_{b(i)j}/r_B$,
and $\bar{z}^{*u}_{bj} = \sum_{i=n_B-r_B+1}^{n_B} z_{b(i)j}/(r_B)$. Thus,

$$\mathrm{var}(z^*_{j\mathcal{D}_{\mathrm{BS}}}) \geq \frac{r(z_{(N)j} - z_{(1)j})^2}{2B(k-1)} \sum_{b=1}^{B} \left(\frac{z_{b(n_B-r_B+1)j} - z_{b(r_B)j}}{z_{(N)j} - z_{(1)j}}\right)^2, \tag{13}$$

which, combined with (12), shows that

$$\begin{aligned}|(\mathcal{X}_{\mathcal{D}_{\mathrm{BS}}}^{*})^{\mathrm{T}}\mathcal{X}_{\mathcal{D}_{\mathrm{BS}}}^{*}| \geq & \frac{r^p}{2^p B^p} k \lambda_{\min}^p(\boldsymbol{R}_{\mathcal{D}_{\mathrm{BS}}}) \prod_{j=1}^{p} (z_{(N)j} - z_{(1)j})^2 \\ & \times \prod_{j=1}^{p} \sum_{b=1}^{B} \left(\frac{z_{b(n_B - r_B + 1)j} - z_{b(r_B)j}}{z_{(N)j} - z_{(1)j}} \right)^2 .\end{aligned}$$

This shows that

$$\frac{|(\mathcal{X}_{\mathcal{D}_{\mathrm{BS}}}^{*})^{\mathrm{T}}\mathcal{X}_{\mathcal{D}_{\mathrm{BS}}}^{*}|}{M_N} \geq \frac{\lambda_{\min}^p(\boldsymbol{R}_{\mathcal{D}_{\mathrm{BS}}})}{(Bp)^p} \prod_{j=1}^{p} \sum_{b=1}^{B} \left(\frac{z_{b(n_B - r_B + 1)j} - z_{b(r_B)j}}{z_{(N)j} - z_{(1)j}} \right)^2 . \tag{14}$$

Each numerator in the summation of the bound in (14) relay on the covariate range of each data partition. If the full data is not divided randomly, this may not produce a sharp bound and using the full data covariate ranges may produce a better bound. We use this idea to derive the bound C_E in the following. From Algorithm 2, for each $j = 1, ..., p$, the r_B data points with the smallest value of z_j and the r_B data points with the largest value of z_j must be included in $\mathcal{D}_{\mathrm{BS}}$. Thus, for each sample variance,

$$\begin{aligned}(k-1)\mathrm{var}(z_{j\mathcal{D}_{\mathrm{BS}}}^{*}) &\geq \left(\sum_{i=1}^{r_B} + \sum_{i=N-r_B+1}^{N} \right) \left(z_{(i)j} - \bar{z}_{j\mathcal{D}_{\mathrm{BS}}}^{*} \right)^2 \\ &\geq \left(\sum_{i=1}^{r_B} + \sum_{i=N-r_B+1}^{N} \right) \left(z_{(i)j} - \bar{z}_{j}^{**} \right)^2 \\ &\geq \frac{r_B}{2} \left(\bar{z}_j^{*u} - \bar{z}_j^{*l} \right)^2 \geq \frac{r_B}{2} \left(z_{(N-r_B+1)j} - z_{(r_B)j} \right)^2 .\end{aligned}$$

In this case, $\bar{z}_j^{**} = (\sum_{i=1}^{r_B} + \sum_{i=N-r_B+1}^{N}) z_{(i)j}/(2r_B)$, $\bar{z}_j^{*l} = \sum_{i=1}^{r} z_{(i)j}/(r_B)$,
and $\bar{z}_j^{*u} = \sum_{i=n-r+1}^{n} z_{(i)j}/(r_B)$. Thus,

$$\mathrm{var}(z_{j\mathcal{D}_{\mathrm{BS}}}^{*}) \geq \frac{r(z_{(N)j} - z_{(1)j})^2}{2(k-1)B} \left(\frac{z_{(N-r_B+1)j} - z_{(r_B)j}}{z_{(N)j} - z_{(1)j}} \right)^2 . \tag{15}$$

This, combined with (12), shows that

$$\frac{|(\mathcal{X}_{\mathcal{D}_{\mathrm{BS}}}^{*})^{\mathrm{T}}\mathcal{X}_{\mathcal{D}_{\mathrm{BS}}}^{*}|}{M_N} \geq \frac{\lambda_{\min}^p(\boldsymbol{R}_{\mathcal{D}_{\mathrm{BS}}})}{(Bp)^p} \times \prod_{j=1}^{p} \left(\frac{z_{(N-r_B+1)j} - z_{(r_B)j}}{z_{(N)j} - z_{(1)j}} \right)^2 . \tag{16}$$

The proof finishes if we combine (14) and (16). □

6.2 Proof of Theorem 2

Proof. The proof for (7) is similar to the proof of inequality (19) in Wang *et al.* (2019). For (8), from the proof of Theorem 4 in Wang *et al.* (2019), we know that

$$\mathrm{var}(z^*_{j\mathcal{D}_{\mathrm{BS}}}) \leq \frac{k}{4(k-1)} \left(z_{(N)j} - z_{(1)j}\right)^2, \tag{17}$$

and

$$\mathbb{V}(\hat{\beta}_j^{\mathcal{D}_{\mathrm{BS}}}|\mathcal{Z}) = \frac{\sigma^2}{k-1} \frac{(\boldsymbol{R}^{-1}_{\mathcal{D}_{\mathrm{BS}}})_{jj}}{\mathrm{var}(z^*_j)}. \tag{18}$$

From the (17), (18), and the fact that $(\boldsymbol{R}^{-1}_{\mathcal{D}_{\mathrm{BS}}})_{jj} \geq 1$, we have

$$\mathbb{V}(\hat{\beta}_j^{\mathcal{D}_{\mathrm{BS}}}|\mathcal{Z}) \geq \frac{4\sigma^2 (z_{(N)j} - z_{(1)j})^2}{k}. \tag{19}$$

From (13), (15) and the fact that $(\boldsymbol{R}^{-1}_{\mathcal{D}_{\mathrm{BS}}})_{jj} \leq \lambda^{-1}_{\min}(\boldsymbol{R}_{\mathcal{D}_{\mathrm{BS}}})$, we have

$$\mathbb{V}(\hat{\beta}_j^{\mathcal{D}_{\mathrm{BS}}}|\mathcal{Z}) \leq \frac{4pB\sigma^2}{k\lambda_{\min}(\boldsymbol{R}_{\mathcal{D}_{\mathrm{BS}}})(z_{(N-r_B+1)j} - z_{(r_B)j})^2}, \quad \text{and} \tag{20}$$

$$\mathbb{V}(\hat{\beta}_j^{\mathcal{D}_{\mathrm{BS}}}|\mathcal{Z}) \leq \frac{4pB\sigma^2}{k\lambda_{\min}(\boldsymbol{R}_{\mathcal{D}_{\mathrm{BS}}}) \sum_{b=1}^B \left(z_{b(n_B-r_B+1)j} - z_{b(r_B)j}\right)^2}. \tag{21}$$

The proof finishes by combining (19), (20), and (21). □

6.3 Proof of Theorem 3

Proof. For the first case that B and r are fixed, from (8) with V_{ej}, we only need to verify that

$$\frac{z_{(N)j} - z_{(1)j}}{z_{(N-r_B+1)j} - z_{(r_B)j}} = O_P(1), \tag{22}$$

which is true according to Theorems 2.8.1 and 2.8.2 in Galambos (1987).

For the second case, $z_{b(n_B-r_B+1)j} - z_{b(r_B)j}$ and $z_{(N)j} - z_{(1)j}$ converge to the same fixed constant in probability, and $z_{b(n_B-r_B+1)j} - z_{b(r_B)j}$ are bounded by this constant for all b. Thus,

$$\frac{1}{B}\sum_{b=1}^{B} (z_{b(n_B-r_B+1)j} - z_{b(r_B)j})^2$$

converges to the same finite constant. Thus, (22) can be easily verified.

For the third case, let $g_{N,j} = F_j^{-1}(1 - 1/N)$ and $g_{n_B,j} = F_j^{-1}(1 - 1/n_B)$. When (9) holds, from the proof of Theorem 5 in Wang *et al.* (2019), we have $z_{(N)j}/g_{N,j} = 1 + o_P(1)$ and $z_{(n_B-r_B+1)j}/g_{n_B,j} = 1 + o_P(1)$. Thus, noting that $z_{b(r)j}$ and $z_{(1)j}$ are bounded in probability when the lower endpoint for the support of F_j is finite, we have

$$\frac{z_{b(n_B-r_B+1)j} - z_{b(r)j}}{z_{(N)j} - z_{(1)j}} = 1 + o_P(1).$$

Note that $\left|\frac{z_{b(n_B-r_B+1)j}-z_{b(r)j}}{z_{(N)j}-z_{(1)j}}\right|$ are bounded by the same constant for all b, so

$$\frac{1}{B}\sum_{b=1}^{B}\frac{z_{b(n_B-r_B+1)j} - z_{b(r)j}}{z_{(N)j} - z_{(1)j}} = 1 + o_P(1).$$

Combining this and (8) with V_{aj}, the result follows.

For the forth case, the result follows from reversing the signs of the covariates in the proof for the third case.

The proof for the fifth case is obtained by combining the proof for the third case and the proof for the fourth case. □

References

Battey, H., Fan, J., Liu, H., Lu, J., and Zhu, Z. (2018). Distributed estimation and inference with statistical guarantees. *Annals of Statistics,* to appear.

Bezanson, J., Edelman, A., Karpinski, S., and Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM review* **59**, 1, 65–98.

Chen, X. and Xie, M.-g. (2014). A split-and-conquer approach for analysis of extraordinarily large data. *Statistica Sinica* **24**, 1655–1684.

Drineas, P., Magdon-Ismail, M., Mahoney, M., and Woodruff, D. (2012). Faster approximation of matrix coherence and statistical leverage. *Journal of Machine Learning Research* **13**, 3475–3506.

Galambos, J. (1987). *The asymptotic theory of extreme order statistics.* Florida: Robert E. Krieger.

Hall, P. (1979). On the relative stability of large order statistics. In *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 86, 467–475. Cambridge Univ Press.

Jordan, M. I. (2012). Divide-and-conquer and statistical inference for big data. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 4–4. ACM.

Lin, N. and Xie, R. (2011). Aggregated estimating equation estimation. *Statistics and Its Interface* **4**, 73–83.

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

Martínez, C. (2004). On partial sorting. Tech. rep., 10th Seminar on the Analysis of Algorithms.

R Core Team (2018). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Schifano, E. D., Wu, J., Wang, C., Yan, J., and Chen, M.-H. (2016). Online updating of statistical inference in the big data setting. *Technometrics* **58**, 3, 393–403.

Shang, Z. and Cheng, G. (2017). Computational limits of a distributed algorithm for smoothing spline. *The Journal of Machine Learning Research* **18**, 1, 3809–3845.

Wang, H. (2018). More efficient estimation for logistic regression with optimal subsample. *arXiv preprint arXiv:1802.02698* .

Wang, H., Yang, M., and Stufken, J. (2019). Information-based optimal subdata selection for big data linear regression. *Journal of the American Statistical Association* **114**, 525, 393–405.

Wang, H., Zhu, R., and Ma, P. (2018). Optimal subsampling for large sample logistic regression. *Journal of the American Statistical Association* **113**, 522, 829–844.

Xue, Y., Wang, H., Yan, J., and Schifano, E. D. (2018). An online updating approach for testing the proportional hazards assumption with streams of big survival data. *arXiv preprint arXiv:1809.01291* .

Yao, Y. and Wang, H. (2019). Optimal subsampling for softmax regression. *Statistical Papers* **60**, 2, 235–249.