On Data Reduction of Big Data
a designana strategy

Min Yang

University of Illinois at Chicago

Banff International Research Station
August 10, 2017

Collaborate with Haiying Wang, John Stufken, Qianshun Cheng, and Xin Wang

Outline

Statistical analysis challenge for Big Data

- Extraordinary size of data

- Limited computation resource

NSF TRIPODS

- Transdisciplinary Reserach in Principle of Data Science (TRIPODS)
- Call for proposal for institute of data science
- Address fundamental research and training in the theoretical foundations of data science
- Focuses on the theoretical foundations of data sciences
  - Core algorithmic
  - Mathematical
  - Statistical principles

NSF TRIPODS: Research focus

- Combinatorial inference on complex structures
- Tradeoffs Between Computational Costs and Statistical Efficiency
- Randomized numerical linear algebra
- Representation theory and noncommutative harmonic analysis
- Topological data analysis (TDA) & homological algebra
- Machine learning including deep learning

Some possible strategies

- Divide and conquer
- Bags of little bootstrap
- Mean log-likelihood
- Subdata

Subdata strategy

- “data reduction is perhaps the most critical component in retrieving information in big data" (Yildirim et al., 2014)
- Selecting a subdata suitable for limited computing resource
- Simple random sampling
- Leveraging algorithm
- Information-Based Optimal Subdata Selecion (IBOSS)

Models considered

- Linear Model
    - LASSO
- Logistic Model

Outline

Big Data Linear Regression

Our focus is on reducing the size of n.

Under the assumption of a linear regression model, write

$$y_i = \beta_0 + \mathbf{z}_i^T \boldsymbol{\beta}_1 + \epsilon_i, \quad i = 1, ..., n,$$

or in matrix form

$$\mathbf{y} = \beta_0 \mathbf{1} + \mathbf{Z}\boldsymbol{\beta}_1 + \boldsymbol{\epsilon} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}.$$

Here $\mathbf{Z}$ is $n \times p$, $\mathbf{X} = [\mathbf{1} \;\; \mathbf{Z}]$, y_i's are uncorrelated given $\mathbf{Z}$, response and covariates are continuous, and $\boldsymbol{\epsilon} \sim (\mathbf{0}, \sigma^2 \mathbf{I})$.

Computational Cost

Inference under linear regression has a computational complexity of $O(np^2)$, which can be problematic for large n.

Wisely chosen subdata can be used so that useful conclusions can be obtained with limited computational resources.

If subdata size is k, the overall computational complexity is $O(kp^2 + ??)$, where ?? depends on the computational complexity of the algorithm for selecting the subdata.

Existing approaches have focused on (random) subsampling-based methods, such as uniform sampling (UNIF) and leveraged sampling (LEV).

Limitation of random sampling approach

The commonly used subsampling-based estimators are bounded from below in the Loewner ordering by finite quantities that are at the order of $1/k$. These quantities do not go to 0 as the full data sample size n goes to ∞.

UNIF:

$$V\left\{\tilde{\beta}_L^{\mathrm{UNIF}}\middle|\mathbf{Z}, I_\Delta(\eta_L)=1\right\} \geq \frac{\sigma^2 P\{I_\Delta(\eta_L)=1|\mathbf{Z}\}}{k}\left(\frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i\mathbf{x}_i^T\right)^{-1}; \tag{1}$$

LEV:

$$V\left\{\tilde{\beta}_L^{\mathrm{LEV}}\middle|\mathbf{Z}, I_\Delta(\eta_L)=1\right\} \geq \frac{(p+1)\sigma^2 P_\eta}{k}\left(E[\mathbf{x}\mathbf{x}^T\{E(\mathbf{x}\mathbf{x}^T)\}^{-1}\mathbf{x}\mathbf{x}^T]\right)^{-1}, \tag{2}$$

IBOSS Approach

Our approach (Wang, Yang, Stufken 2016): Information-Based Optimal Subdata Selection (IBOSS)

Rather than sampling, we select subdata judiciously so as to maximize the Fisher information matrix, in some sense, for the parameters in the assumed model

For linear regression, under normality and taking $\sigma^2 = 1$ for simplicity, the information matrix for β with subdata is

$$\mathbf{M}(\boldsymbol{\delta}) = \sum_{i=1}^{n} \delta_i x_i x_i^T = \mathbf{X}^T \boldsymbol{\Delta} \mathbf{X},$$

with δ_i an "inclusion" indicator, $\boldsymbol{\delta} = (\delta_1, ..., \delta_n)$ and $\boldsymbol{\Delta} = diag(\boldsymbol{\delta})$

What is a good choice for $\boldsymbol{\delta}$?

Optimal Design of Experiments

As in optimal design of experiments (DOE), we could maximize a function of the information matrix

D-optimality: Find δ, subject to $\sum_i \delta_i = k$, that maximizes $\det(\mathbf{M}(\delta))$

A difference with DOE is that we already have data, and must make a choice for the z_{ij}'s that is consistent with the data

Another challenge is size: we need a computationally efficient algorithm to find, approximately, an optimal δ

D-optimal design under approximate design theory

Theorem (D-optimality)

For subdata of size k represented by δ,

$$|\mathbf{M}(\delta)| \le \frac{k^{p+1}}{4^p} \prod_{j=1}^{p} (z_{(n)j} - z_{(1)j})^2, \tag{3}$$

where $z_{(n)j} = \max\{z_{1j}, z_{2j}, ..., z_{nj}\}$ and $z_{(1)j} = \min\{z_{1j}, z_{2j}, ..., z_{nj}\}$ are the nth and first order statistics of $z_{1j}, z_{2j}, ..., z_{nj}$. If the subdata consists of the 2^p points $(a_1, \ldots, a_p)^T$ where $a_j = z_{(n)j}$ or $z_{(1)j}$, $j = 1, 2, ..., p$, each occurring equally often, then equality holds in (3).

Algorithm for *D*-optimality

To maximize $\det(\mathbf{M}(\delta))$, we need to include points with large and small covariate values

For a fixed subdata size k, using a partition-based selection algorithm, for $j = 1, ..., p$, select the $k/(2p)$ largest and smallest z_{ij}-values, and include these points in the subdata

Estimate β by $\hat{\beta}^D = (\mathbf{X}^T\mathbf{\Delta}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{\Delta}\mathbf{y}$

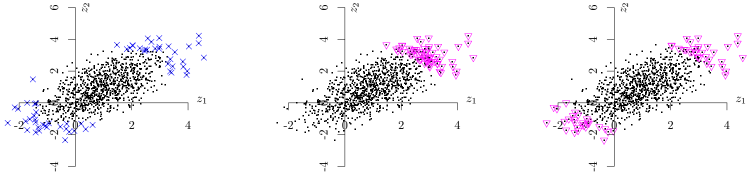
Computational complexity for selection $O(np)$

Overall computational complexity $O(kp^2 + np)$, or $O(np)$ if $n > kp$

Can do this one covariate at a time (no duplication) or in parallel (possibly less than k points due to duplication)

Toy Example of Subdata Selected by IBOSS

For a full data of size $n = 1000$ and $p = 2$, and subdata size $k = 60$.



(a) *D*-optimality **(b)** naive *T*-optimality **(c)** *cT*-optimality

Figure: Shapes of IBOSS subdata based on different optimality criteria.

Other optimality criteria require a different algorithm

Theoretical Results

IBOSS can be used no matter what the distribution of the covariates is ...

... but, for D-optimality, the performance depends on this

Let $\mathbf{z}_1, ..., \mathbf{z}_n$ be iid, and consider 3 scenarios:

1. $\mathbf{z}_i \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$
2. $\mathbf{z}_i \sim Lognormal(\boldsymbol{\mu}, \boldsymbol{\Sigma})$
3. $\mathbf{z}_i \sim t_\nu(\boldsymbol{\mu}, \boldsymbol{\Sigma})$

For all scenarios, $Var(\hat{\beta}_0^D|\mathbf{Z})$ is proportional to $1/k$ when $n \to \infty$

But the story is different for $Var(\hat{\beta}_1^D|\mathbf{Z})$ ...

Theoretical Results

Elements of $Var(\hat{\beta}_1^D|\mathbf{Z})$ converge to 0 when $n \to \infty$ in all cases (even though the subdata size k is fixed)

For scenario 1 (normal), elements converge to 0 as $1/\log(n)$

For scenario 2 (lognormal), the element in position (j_1, j_2) converges to 0 as $\exp(-(\sigma_{j_1} + \sigma_{j_2})\sqrt{2\log(n)})$

For scenario 3 (t-distribution), elements converge to 0 as $n^{-2/\nu}$

Similar results typically do not hold for subsampling methods

Simulation setup

$p = 50$, $\beta = \mathbf{1}_{51\times1}$, $\epsilon_i \sim N(0, \sigma^2)$ with $\sigma^2 = 9$, $\boldsymbol{\Sigma} = (.5^{I(i\neq j)})$.

$\mathbf{z}_i$'s are generated from the following distributions.

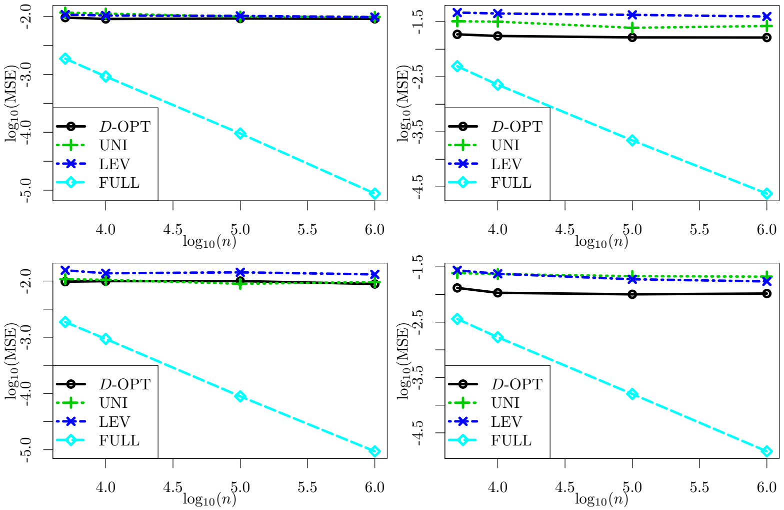
1. **Normal**, $N(\mathbf{0}, \boldsymbol{\Sigma})$;
2. **Lognormal**, $\exp\{N(\mathbf{0}, \boldsymbol{\Sigma})\}$;
3. $t_2(\mathbf{0}, \boldsymbol{\Sigma})$;
4. **Mixture** of $N(\mathbf{1}, \boldsymbol{\Sigma})$, $t_2(\mathbf{1}, \boldsymbol{\Sigma})$, $t_3(\mathbf{1}, \boldsymbol{\Sigma})$, Unif$[\mathbf{0}, \mathbf{2}]$ and $\exp\{N(\mathbf{0}, \boldsymbol{\Sigma})\}$ with equal proportions.

Each simulation was repeated $S = 1000$ times.

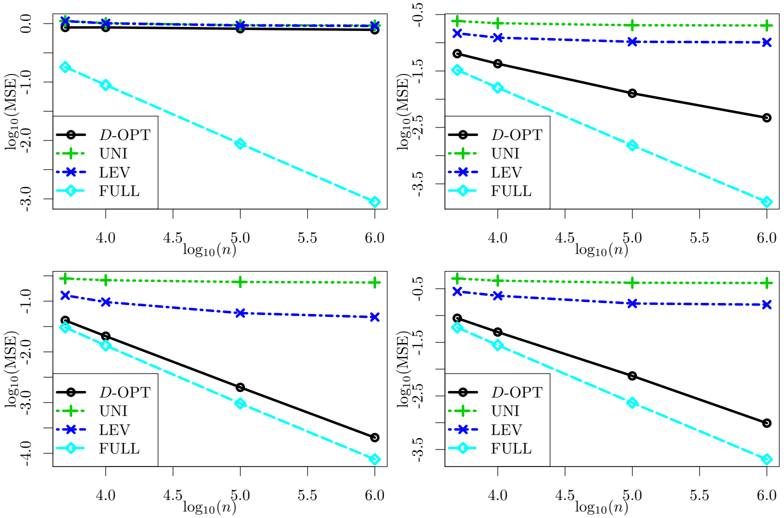
Empirical mean squared errors (MSE) are compared.

Light blue = full data; black = IBOSS with D-optimality; red = IBOSS with T-optimality; green = uniform sampling; blue = leveraged sampling

MSE of the intercept estimator with $k = 1000$



MSE of the slope estimator with $k = 1000$



CPU times for different n and p

Table: CPU times (seconds) for different n with $p = 500$

n	D-opt	T-opt	UNI	LEV	FULL
5×10^3	1.19	0.35	0.33	0.88	1.44
5×10^4	1.36	0.50	0.29	2.20	13.39
5×10^5	8.89	2.64	0.31	21.23	132.04

Table: CPU times (seconds) for different p with $n = 5 \times 10^5$

p	D-opt	T-opt	UNI	LEV	FULL
10	0.19	0.05	0.00	1.94	0.21
100	1.74	0.42	0.02	4.66	6.55
500	9.30	2.53	0.31	21.94	132.47

A Small Example

Data from US Department of Agriculture from Continuing Survey of Food Intakes by Individuals (CSFII)

Calorie intake as dependent variable, average intake levels of fat, protein, and carbohydrate, as well as BMI and age as covariates

So $p = 5$; with $n = 1,827$ this is a small example that also enables comparison to full data analysis

Conclusions from IBOSS with D-optimality and $k = 10p = 50$ are compared to those from full data

A Small Example

Table: Estimation results for the CSFII data. For the D-OPT IBOSS method, the subdata size is $k = 10p = 50$.

Parameter	D-OPT		FULL	
	Estimate	Std. Error	Estimate	Std. Error
Intercept	33.545	46.833	45.489	11.883
Age	-0.496	1.015	-0.200	0.234
BMI	-0.153	0.343	-0.521	0.224
Fat	8.459	0.405	9.302	0.115
Protein	5.080	0.386	4.254	0.127
Carb	3.761	0.106	3.710	0.035

A Small Example

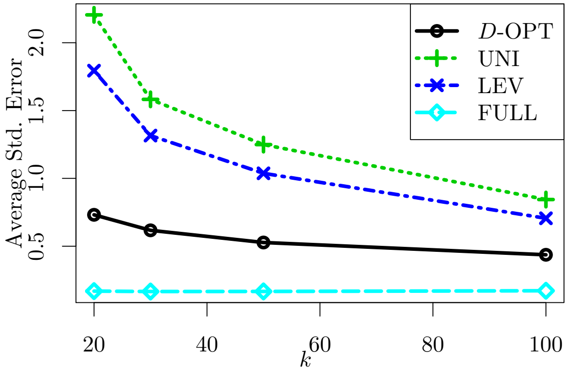


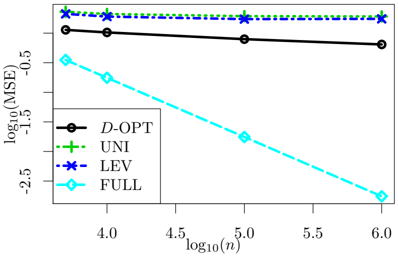
Figure: Average standard errors for estimating slope parameters for the CSFII data, using $k = 4p,\ 6p,\ 10p$ and $20p$. Standard errors are computed from thousand bootstrap samples.

Interaction model

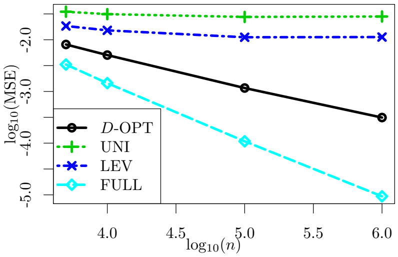
Suppose the true model is

$$y_i = \beta_0 + \sum_{j=1}^{10} z_{ij}\beta_j + \sum_{j_1 \neq j_2}^{10} z_{ij_1} z_{ij_2}\beta_{j_1 j_2} + \varepsilon_i, \qquad i = 1, ..., n, \qquad (4)$$

Only the main effects are used in selecting subdata.



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



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

Figure: The subdata size k is fixed at $k = 1000$ and the full data size n changes

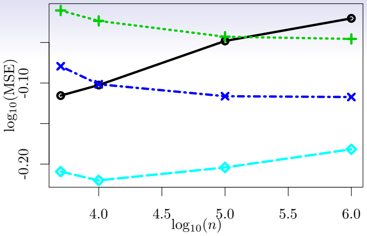
Nonlinear model

The true relationships between the response and the covariates are nonlinear.

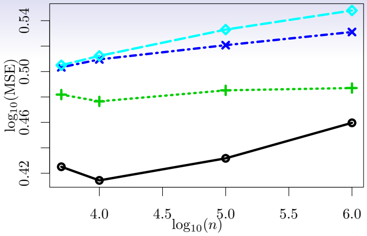
$$y_i = \beta_0 + \sum_{j=1}^{p-1} z_{ij}\beta_j + \frac{3e^{z_{ip}^{(t)}}}{1 + e^{z_{ip}^{(t)}}} + \varepsilon_i, \quad i = 1, ..., n, \tag{WM1}$$

$$y_i = \beta_0 + \sum_{j=1}^{p-1} z_{ij}\beta_j + 30 \log\left(1 + e^{z_{ip}^{(t)}}\right) + \varepsilon_i, \quad i = 1, ..., n, \tag{WM2}$$

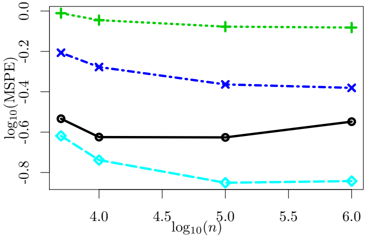
where $z_{ip}^{(t)} = z_{ip} I(z_{ip} \leq 100) + 100 I(z_{ip} > 100)$.



(a) Slope parameter

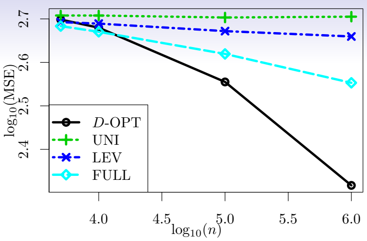


(b) Intercept parameter

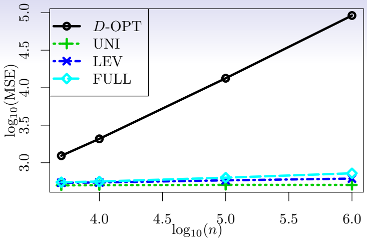


(c) Prediction

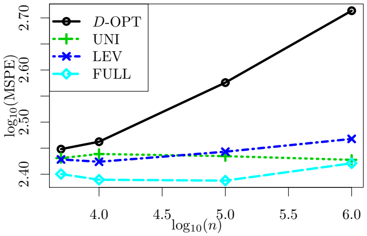
Figure: For model (WM1): the subdata size k is fixed at $k = 1000$ and the full data size n changes



(a) Slope parameter



(b) Intercept parameter



(c) Prediction

Figure: For model (WM2): the subdata size k is fixed at $k = 1000$ and the full data size n changes

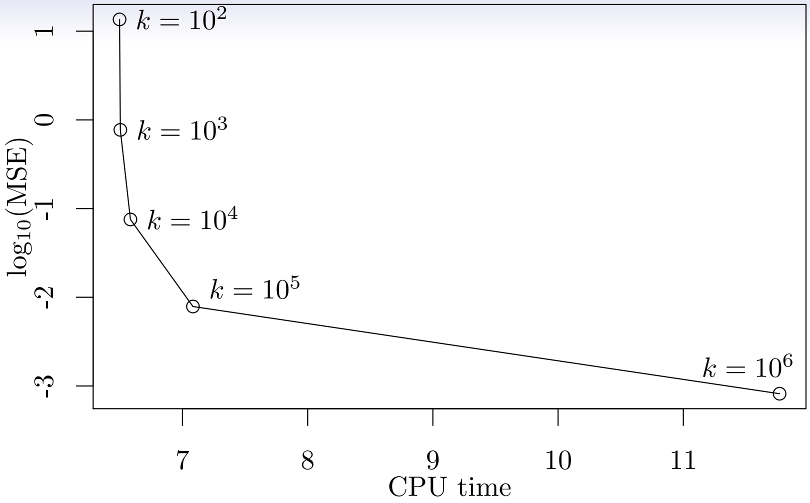
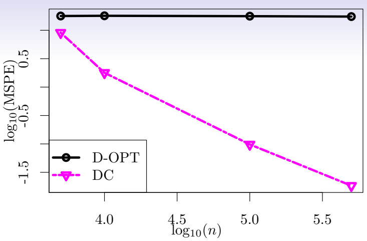


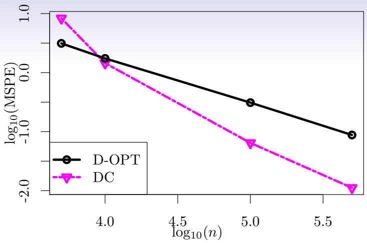
Figure: Average CPU times and MSEs for different subdata sample size k when the covariates are from a multivariate normal distribution. The full data size is set to $n = 5 \times 10^6$ with a dimension $p = 50$.

Comparison with the divide-and-conquer method

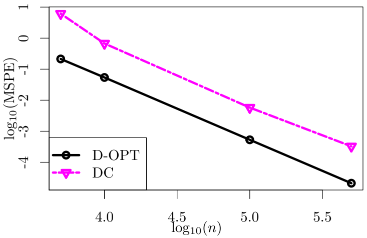
Battery et al. (2017) propose to divide the full data into S subdata sets and the ordinary least squares estimate, say $\hat{\boldsymbol{\beta}}_s$, is calculated for each subdata. The DC estimate is the average of $\hat{\boldsymbol{\beta}}_s$'s, i.e., $\bar{\boldsymbol{\beta}} = S^{-1} \sum_{s=1}^{S} \hat{\boldsymbol{\beta}}_s$.



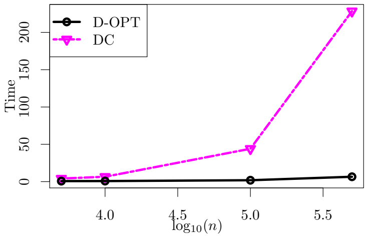
(a) Case 1: $\mathbf{z}_i$'s are normal.



(b) Case 4: $\mathbf{z}_i$'s are a mixture.



(c) Case 6: $\mathbf{z}_i$'s are t_1.



(d) CPU times.

Figure: $k = 1000$ and $p = 500$

Extension to large *p*

- Variable selection

- Penalized likelihood estimators
  - LASSO(least absolute shrinkage and selection operator)
  - LARS algorithm (Least-angles Regression)
  - SCAD (Smoothly Clipped Absolute Deviation)
  - MCP (Minimax Concave Penalty)

Model Setup

- For linear model $E(Y) = X\beta$, LASSO estimator is

$$\hat{\beta}_{LASSO} = \arg\min_{\beta}\{||Y - X\beta||_2^2/N + \lambda||\beta||_1\}$$

- λ is usually selected by minimizing cross-validation error
- It takes about 200 seconds for $n = 1000$ and $p = 2000$ (Chen and Xie, 2014)
- To develop subset selection method (Xin Wang's ongoing work)

Rationale

Knight and Fu (2000) shows that

$$\sqrt{n}(\hat{\beta}^n_{LASSO} - \beta) \rightarrow_d \arg\min(V(U)),$$

where

$$V(U) = -2U^T W + U^T CU + \lambda_0 U^T sign(\beta)$$

and $W \sim N(0, \sigma^2 C)$ and $C = \lim_{n\rightarrow\infty} \frac{1}{n} \sum_{i=1}^n X_i X_i^T$.

Rationale

$$\sqrt{n}(\hat{\beta}^n_{LASSO} - \beta) \rightarrow_d C^{-1}\left(-\frac{\lambda_0}{2} sign(\beta) - W\right).$$

Under certain condition,

$$||\sqrt{n}(\hat{\beta}^n_{LASSO} - \beta)||^2 \rightarrow \frac{\lambda_0^2}{4} sign(\beta)^T C^{-2} sign(\beta) + Trace(C^{-1}).$$

D-optimal Motivated Algorithm – Top Correlated Only

- Use Pearson's correlation coefficient to filter variables for the algorithm – **only include columns having largest $|\rho|$ with Y.**

- Use IBOSS approach for the selected variables to select a subdata

Theoretical Computation Time Comparison

If consider LARS algorithm for lasso regression when $n > p$,

- LASSO regression with all the N observations will take $O(np^2)$
- LASSO regression with D-optimality motivated algorithm will take $O(np + kp^2)$
- LASSO regression with top correlated only D-optimal algorithm will take $O(ns + kp^2)$, s is the number of columns considered by the algorithm.

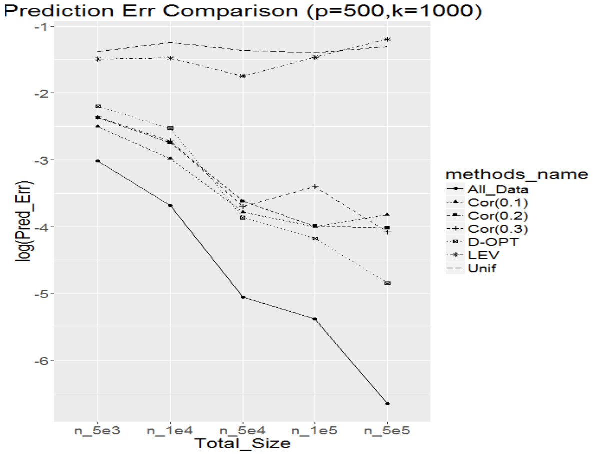
Simulation Example 1

Table: Lasso: Time Cost ($n = 5 \times 10^5, k = 10^3$) – $X_{ij} \sim t(2)$

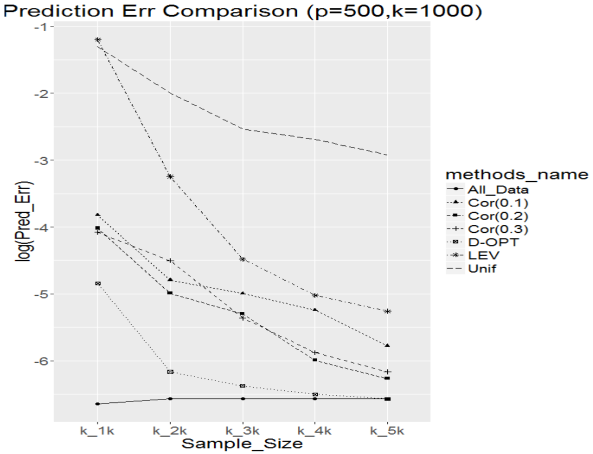
p	Full	Cor(0.1)	UNIF
50	17.9819	0.8195	0.1242
100	45.0246	1.3368	0.2051
500	433.2812	6.0242	3.2958

- -$\epsilon_i \sim N(0,1), \beta_j \approx 2\sqrt{log(p)/k}/3$
- -averaged over 100 runs
- -Cor(0.1) represent top 10% correlated variables are considered
- -Random is simple random sample approach
- -Full is LASSO with all data

Simulation Results with Fixed *k* and Varied *n*



Simulation Results With Fixed *n* and Varied *k*



Outline

Logistic Regression Model

Logistic regression

- $Prob(Y_i = 1|X_i) = P(X_i) = \frac{e^{X_i^T \beta}}{1+e^{X_i^T \beta}}$
- $\beta = (\beta_0, \cdots, \beta_m)^T$
- X_i is the covariate vector for response Y_i, $i = 1, \cdots, n$.

Computation challenge for Big Data under logistic regression models

- Computation cost is $O(\delta n p^2)$
- China Mobile data example

Levraging type algorithm for logistic regreesion

- Leveraging algorithms: Subsampling algorithms by assigning sampling probability weight to each dataline.
- Wang et al. (2017) proposed the OSMAC algorithms using leveraging methods to handle the subsampling work for logistic model.
    - The probability weight is calculated based on the criteria of A-optimality from optimal design theory.
    - Among all of OSMAC algorithms , mVc algorithms is the most convenient one.
    - Probability weight depends on the response variable.

Asymptotic property of the LEV type algorithms

Theorem (Wang et.al(2017))

Let n denote total data size and r denote subsample size. The leveraging sampling probability for each dataline is set as pi_i. *Let* F_N *the set of (Y,X). Then under certain conditions specified in Wang et.al(2016), as n,* $r \to \infty$, *conditional on* F_N,

$$V^{-\frac{1}{2}}(\hat{\beta}_{sub} - \hat{\beta}) \to N(0, I).$$

Where $\hat{\beta}_{sub}$ *is the weighted maximum likelihood estimation for subdata,* $\hat{\beta}$ *is the maximum likelihood estimation for full data and* $V = M_X^{-1} V_c M_X^{-1}$. *In this formula,* $M_X = \frac{1}{n}\sum_{i=1}^{n} w_i(\hat{\beta}) x_i x_i^T$, $V_c = \frac{1}{rn^2}\sum_{i=1}^{n} \frac{(y_i - p_i(\hat{\beta}))^2 x_i x_i^T}{\pi_i}$ *and* $w_i(\beta) = p_i(\beta)(1 - p_i(\beta))$.

Asymptotic property of the LEV type algorithms II

Theorem

For all leveraging type algorithms, if the conditions in Wang et.al (2017) is satisfied and the consistency holds, then the information matrix $I = V^{-1}$ can be simplified as

$$I \leq r(\sum_{i=1}^{n} \pi_i x_i x_i^T)$$

Remarks:

- For simple random sampling, which means $\pi_i = \frac{1}{n}$ for $i = 1, \cdots, n$, $I \leq rM$.
- For mVc algorithm (one of the efficient LEV algorithms in Wang et al(2016)), $\pi_i^{mVc} = \frac{|y_i - p_i(\hat{\beta})|||x_i||}{\sum_{j=1}^{n} |y_j - p_j(\hat{\beta})|||x_j||}$, $I \leq r(\frac{M+o(1)}{a})$, where a is positive constant and $M = E(xx^T)$

Properties of the LEV type algorithms

- Easy to understand and to implement the algorithm.
- Subsampling efficiency is good under certain simulation and real scenarios.
- The probability weight assigned to each data line, especially for the logistic regression, may depends on response variable Y.
- When the subsample size r is fixed, then the fisher information of the popular LEV algorithm is bounded even when $n \to \infty$.

Motivation

- Easy computation and implementation
- Facilitate scientific discoveries from big data with limited computing resources.
- How to preserve the majority information contained in the full data?

Information matrix for logistic regression

- $$I_\xi(\beta) = n \sum_{i=1}^{k} \omega_i X_i \Psi(c_i)(X_i)^T \qquad (5)$$

  where $c_i = \beta_0 + \beta_1 x_{i1} + \ldots + \beta_m x_{im}$, and $\Psi(c) = [P'(c)]^2/[P(c)(1-P(c))]$.

- D-optimality: Maximize the determinant of the information matrix.

Local D-optimality and New Subsampling Algorithm

Theorem (Yang, Zhang and Huang, 2011)

Under logistic model mentioned above, a design ξ^* is D-optimal design of parameter β if
$\xi^* = \{(C^*_{l1}, 1/2^m)\&(C^*_{l2}, 1/2^m),\ l = 1, \cdots, 2^{m-1}\}$, *where* $C^*_{l1} = (1, a_{l,1}, \cdots, a_{l,m-1}, c^*)$ *and* $C^*_{l2} = (1, a_{l,1}, \cdots, a_{l,m-1}, -c^*)$
.

- c^* minimize function $f(c)$, where $f(c) = c^{-2}(\Psi(c))^{-m-1}$ and $\Psi(c) = \frac{[P'(x)]^2}{P(x)(1-P(x))}$
- $a_{l,i}$ is the boundary of the design space in the i-th dimension, $i = 1, \cdots, m-1$

New Subsampling Algorithm

Inspired by the local D-optimality of logistic model, the following subsampling algorithm is proposed
Algorithm:

1. Given data set $\{(Y_i, X_i^T),\ i = 1, \cdots, n\}$, first do random sampling and pick r_o sub-samples, fitting the data and get estimate $\hat{\beta} = (\hat{\beta}_0, \cdots, \hat{\beta_m})$
2. Compute $c_i = X_i^T \hat{\beta}$, pick B = $\{i \mid min\{|c_i - c^*|, |c_i + c^*|\} \leq \delta\}$
3. Inside $\{(Y_i, X_i^T),\ i \in B\}$, pick $[\frac{r_1}{2(m-1)}]$ data lines with largest value and $[\frac{r_1}{2(m-1)}]$ data lines with smallest value on the k-th dimension, $k = 1, \cdots, m-1$; combined the picked datalines as the newly constructed new sub-sample

Advantage of New Algorithm

- Easy implementations
- Subsampling procedure is independent of Y.
- Low computational cost

Some asymptotic property of the new algorithm

- One of the advantage of the new algorithm is that the fisher information for the picked subsample might goes to infinity for a fixed subsample size, as long as $n \to \infty$.

- Due to the complexity of the computation, we only investigate into the two dimension case, where $X = (X_1, X_2)^T$, $\beta = (\beta_0, \beta_1, \beta_2)$ and $c = X^T\beta$.

- Define $f_1 = F^{-1}(1 - \frac{1}{n})$ and $f_2 = F_2^{-1}(1 - \frac{1}{n})$, where F_1 is the cdf of the first dimension X_1 conditional on $|c - c^*| < \delta$, and F_2 is the cdf of the first dimension X_1 conditional on $|c + c^*| < \delta$

Some asymptotic property of the new algorithm II

Theorem

Consider we have two dimension case, where $c = \beta_0 + x_1\beta_1 + x_2\beta_2$, assume covariate vector X follows multivariate normal distribution and the new subsampling algorithm is implemented, then the fisher information I of the picked data can be written as

$$I^{IBOSS} = \sum_{i \text{ is picked}} \Phi_i(\beta) X_i X_i^T \geq a \sum_{i \text{ is picked}} X_i X_i^T = a \begin{pmatrix} k & l_{12} \\ l_{12}^T & l_{22} \end{pmatrix}$$

where $l_{12} = \left([\frac{r}{2}]f_1 - [\frac{r}{2}]f_2 + o(1) \quad [\frac{r}{2}]f_2 - [\frac{r}{2}]f_1 + O(1) \right)$,

$$l_{22} = \begin{pmatrix} [\frac{r}{2}]((f_1)^2 + (f_2)^2) + o(F) & -[\frac{r}{2}](\frac{(f_1)^2}{\beta_2} + \frac{(f_2)^2}{\beta_2}) + O(F) \\ -[\frac{r}{2}](\frac{(f_1)^2}{\beta_2} + \frac{(f_2)^2}{\beta_2}) + O(F) & [\frac{r}{2}](\frac{(f_1)^2}{\beta_2^2} + \frac{(f_2)^2}{\beta_2^2}) + o(F) \end{pmatrix},$$

and $F = max(f_1, f_2) \to \infty$ *as* $n \to \infty$.

Comparison of Computational Cost

- Computaional cost of different approach with fixed sample size r=1000 and dimension p=7.

Table: Compuational Cost of Different Approach with Different Data Size

	SRS	LEV	New Algorithm	Full Data
$N = 500000$	0	0.02	0.03	0.40
$N = 1000000$	0	0.06	0.09	0.79
$N = 5000000$	0	0.47	0.36	4.09

Comparison of Computational Cost

- Computaional cost of different approach with fixed sample size r=1000 and data size n=500000.

Table: Compuational Cost of Different Approach with Different Dimension

	SRS	LEV	New Algorithm	Full Data
$p = 10$	0.001	0.035	0.04	0.397
$p = 20$	0.004	0.180	0.104	1.378
$p = 100$	0.02	0.372	0.157	3.261

Simulations Settings

$n = 10000$, $r_0 = 200$,β is 1 X 7 vector and true value $\beta_0 = (0.5, \cdots, 0.5)$, variance-covariance structure Σ, $\Sigma_{ij} = 0.5$ for $i \neq j$ and $\Sigma_{ij} = 1$ for $i = j$

- NzNormal: $N(\mu, \Sigma)$ with $\mu = (0, \ldots, 0)$
- MzNormal: $N(\mu, \Sigma)$ with $\mu = (1, \ldots, 1)$
- Mixed Normal: $\frac{1}{2}N(\mu, \Sigma) + \frac{1}{2}N(-\mu, \Sigma)$ with $\mu = (1, \ldots, 1)$
- T3

Simulation Results Small Size

Figure: MzNormal

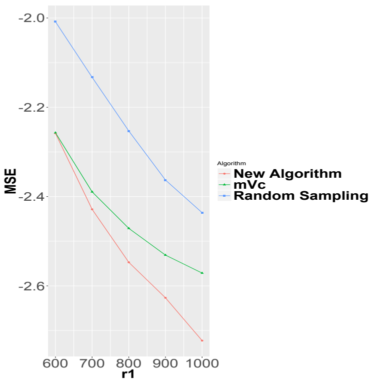
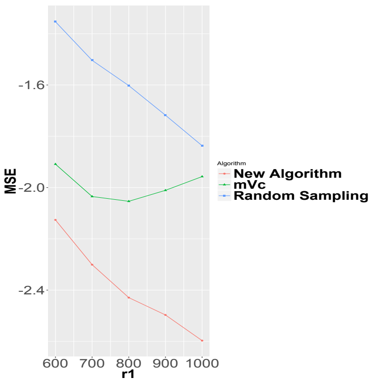


Figure: NzNormal



Simulation Results Small Size

Figure: MixNormal

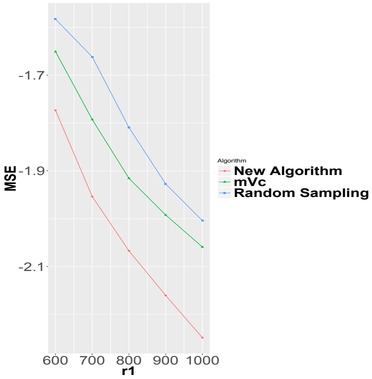
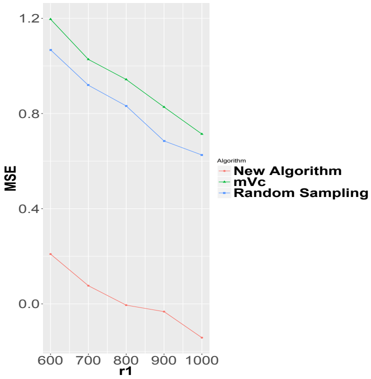


Figure: T3



Simulations Settings Large Size

$n = 500000$, $r_0 = 1000$,β is 1 X 7 vector and true value $\beta_0 = (0.5, \cdots, 0.5)$, variance-covariance structure Σ, $\Sigma_{ij} = 0.5$ for $i \neq j$ and $\Sigma_{ij} = 1$ for $i = j$

- Mixed Normal
- T3

Simulation Results Large Size

Figure: MixNormal

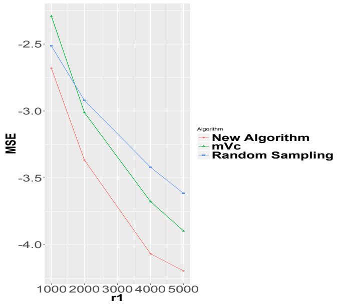
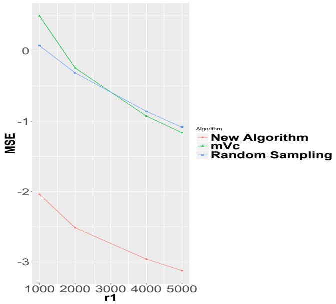


Figure: T3



Irregular X space

- When X is taking value in an irregular space, the performance of IBOSS maybe limited under this scenario.

- Alternative approach - Directional Derivative Approach

Directional Derivative Approach: Detail

- Step 1: Numerically find the optimal design ξ under the given range of X values for all dimensions.
- Step 2: Using D-optimality, compute $m_i = tr(I_\xi^{-1}(I_{X_i} - I_\xi))$ for each dataline X_i.
- Pick out (X_i, Y_i) with the highest r $m_i's$ as the subsample.

Directional Derivative Approach: Settings

We now testing the performance of Directional Derivative Approach under bounded case. In the simulation for this part, we adjust the distribution center and scale to make X on all dimension falls in range around $[0, 6]$.

- T distribution case: Generate data from $\frac{\textbf{T}\text{-dist}(\textbf{df}=\textbf{3})}{\textbf{5}} + \textbf{3}$.
- Exponent distribution case: Generate data from $\textbf{Exp}(\textbf{1})$
- Mixed Normal distribution: Generate data from $0.5N(2, \Sigma) + 0.5N(3, \Sigma)$
- Normal distribution I: Generate data from $N(2, \Sigma)$.
- Normal distribution II: Generate data from $N(3, \Sigma)$.

where Σ is diagonal matrix with all entry as 2. All other settings is similar to the former scenarios.

Directional Derivative Approach: Results

Table: Performace of Directional Derivative Approach

Distribution	N	MSE_{FULL}	MSE_{LEV}	MSE_{SRS}	MSE_{DD}
T	200000	$2.62\text{x}10^{-3}$	$3.00\text{x}10^{-2}$	$7.13\text{x}10^{-2}$	9.17x10
Mixed	200000	$4.09\text{x}10^{-4}$	$5.29\text{x}10^{-3}$	$1.03\text{x}10^{-2}$	2.77x10
Normal I	200000	$1.08\text{x}10^{-4}$	$1.99\text{x}10^{-3}$	$2.69\text{x}10^{-3}$	1.00x10
Normal II	200000	$3.16\text{x}10^{-4}$	$8.67\text{x}10^{-3}$	$7.92\text{x}10^{-3}$	1.48x10
T	400000	$1.34\text{x}10^{-3}$	$1.37\text{x}10^{-2}$	$6.98\text{x}10^{-2}$	6.38x10
Mixed	400000	$2.15\text{x}10^{-4}$	$2.59\text{x}10^{-3}$	$9.91\text{x}10^{-3}$	2.41x10
Normal I	400000	$5.30\text{x}10^{-5}$	$1.14\text{x}10^{-3}$	$2.61\text{x}10^{-3}$	1.00x10
Normal II	400000	$1.64\text{x}10^{-4}$	$2.61\text{x}10^{-3}$	$7.92\text{x}10^{-3}$	1.32x10
Exponent	500000	$2.25\text{x}10^{-5}$	$1.24\text{x}10^{-3}$	$1.51\text{x}10^{-3}$	2.30x10

Comparison of Computational Cost with Directional Derivative

- Computaional cost of different approach with fixed sample size r=5000 and dimension p=3.

Table: Compuational Cost of Different Approach with Different Data Size

	SRS	LEV	Directional Derivative	Full Dat
$N = 200000$	0.001	0.002	0.001	0.016
$N = 500000$	0.001	0.009	0.006	0.058
$N = 1000000$	0.003	0.013	0.016	0.101

Main references

Wang, H., Yang, M., and Stufken, J. (2017). Information-based optimal subdata selection for big data linear regression. Under revision.

Cheng, Q., Yang, M., and Wang, H. (2017). Information-based optimal subdata selection for big data logistic regression. In preparation

Wang, X. and Yang, M. (2017). Information-Based Optimal Subdata Selection for Penalized Regression. In preparation