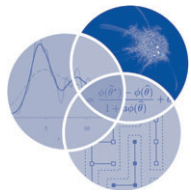
Focus Article

Leveraging for big data regression



Ping Ma* and Xiaoxiao Sun

Rapid advance in science and technology in the past decade brings an extraordinary amount of data, offering researchers an unprecedented opportunity to tackle complex research challenges. The opportunity, however, has not yet been fully utilized, because effective and efficient statistical tools for analyzing super-large dataset are still lacking. One major challenge is that the advance of computing resources still lags far behind the exponential growth of database. To facilitate scientific discoveries using current computing resources, one may use an emerging family of statistical methods, called leveraging. Leveraging methods are designed under a subsampling framework, in which one samples a small proportion of the data (subsample) from the full sample, and then performs intended computations for the full sample using the small subsample as a surrogate. The key of the success of the leveraging methods is to construct nonuniform sampling probabilities so that influential data points are sampled with high probabilities. These methods stand as the very unique development of their type in big data analytics and allow pervasive access to massive amounts of information without resorting to high performance computing and cloud computing.

How to cite this article:
WIREs Comput Stat 2014. doi: 10.1002/wics.1324

Keywords: big data; subsampling; estimation; least squares; leverage

INTRODUCTION

Rapid advance in science and technology in the past decade brings an extraordinary amount of data that were inaccessible just a decade ago, offering researchers an unprecedented opportunity to tackle much larger and more complex research challenges. The opportunity, however, has not yet been fully utilized, because effective and efficient statistical and computing tools for analyzing super-large dataset are still lacking. One major challenge is that the advance of computing technologies still lags far behind the exponential growth of database. The cutting edge computing technologies of today and foreseeable future are high performance computing (HPC) platforms, such as supercomputers and cloud computing. Researchers have proposed use of GPUs (graphics processing units),[1] and explorations of FPGA (field-programmable gate array)-based accelerators are ongoing in both academia and industry (for example, with TimeLogic and Convey Computers). However, none of these technologies by themselves can address the big data problem accurately and at scale. Supercomputers are precious resources and are not designed to be used routinely by everyone. The selection of the projects for running on supercomputer is very stringent. For example, the resource allocation of latest deployed Blue Water supercomputer at National Center for Super Computing Applications (https://bluewaters.ncsa.illinois.edu/) requires a rigorous National Science Foundation (NSF) review process and supports only a handful projects. While clouds have large storage capabilities, 'cloud computing is not a panacea: it poses problems for developers and users of cloud software, requires large data transfers over precious low-bandwidth Internet uplinks, raises new privacy and security issues, and is an inefficient solution'.[2] While cloud facilities like mapReduce do offer some advantages, the overall issues of data transfer and usability of today's clouds will indeed limit what we can achieve; the

*Correspondence to: pingma@uga.edu
Department of Statistics, University of Georgia, Athens, GA, USA

Conflict of interest: The authors have declared no conflicts of interest for this article.

tight coupling between computing and storage is absent. A recently conducted comparison[3] found that even high-end GPUs are sometimes outperformed by general-purpose multi-core implementations because of the time required to transfer data.

'One option is to invent algorithms that make better use of a fixed amount of computing power'.[2] In this article, we review an emerging family of statistical methods, called leveraging methods that are developed for achieving such a goal. Leveraging methods are designed under a subsampling framework, in which we sample a small proportion of the data (subsample) from the full sample, and then perform intended computations for the full sample using the small subsample as a surrogate. The key of the success of the leveraging methods relies on effectively constructing nonuniform sampling probabilities so that influential data points are to be sampled with high probabilities. Traditionally, 'subsampling' has been used to refer to 'm-out-of-n' bootstrap, whose primary motivation is to make approximate inference owing to the difficulty or intractability in deriving analytical expressions.[4,5] However, the general motivation of the leveraging method is different from the traditional subsampling. In particular, (1) leveraging methods are used to achieve feasible computation even if the simple analytic results are available, (2) leveraging methods enable the visualization of the data when visualization of the full sample is impossible, and (3) as we will see later, leveraging methods usually use unequal sampling probabilities for subsampling data, whereas subsampling almost exclusively uses equal sampling probabilities. Leveraging methods stand as the very unique development in big data analytics and allow pervasive access to massive amounts of information without resorting to HPC and cloud computing.

LINEAR REGRESSION MODEL AND LEVERAGING

In this article, we focus on the linear regression setting. The first effective leveraging method in regression is developed in Refs 6 and 7 The method uses the sample leverage scores to construct a nonuniform sampling probability. Thus, the resulting sample tends to concentrate more on important or influential data points for the ordinary least squares (OLS).[6,7] The estimate based on such a subsample has been shown to provide an excellent approximation to the OLS based on full data (when p is small and n is large). Ma et al.[8,9] studied the statistical properties of the basic leveraging method in Refs 6 and 7 and proposed two novel methods to improve the basic leveraging method.

Linear Model and Least Squares Problem

Given n observations, $\left(x_i^{\mathrm{T}}, y_i\right)$, where $i = 1, \ldots, n$, we consider a linear model,

$$y = X\beta + \epsilon, \tag{1}$$

where $y = (y_1, \ldots, y_n)^{\mathrm{T}}$ is the $n \times 1$ response vector, $X = (x_1, \ldots, x_n)^{\mathrm{T}}$ is the $n \times p$ predictor or design matrix, β is a $p \times 1$ coefficient vector, and ϵ is zero mean random noise with constant variance. The unknown coefficient β can be estimated via the OLS, which is to solve

$$\widehat{\beta}_{\mathrm{ols}} = \arg\min_{\beta} ||y - X\beta||^2, \tag{2}$$

where $||\cdot||$ represents the Euclidean norm. When matrix X is of full rank, the OLS estimate $\widehat{\beta}_{\mathrm{ols}}$ of β, minimizer of (2), has the following expression,

$$\widehat{\beta}_{\mathrm{ols}} = \left(X^{\mathrm{T}}X\right)^{-1} X^{\mathrm{T}}y. \tag{3}$$

When matrix X is not of full rank, the inverse of $X^{\mathrm{T}}X$ in expression (3) is replaced by generalized inverse. The predicted response vector is

$$\widehat{y} = Hy, \tag{4}$$

where the hat matrix $H = X(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}$. The ith diagonal element of the hat matrix H, $h_{ii} = x_i^{\mathrm{T}}\left(X^{\mathrm{T}}X\right)^{-1}x_i$, is called the leverage score of the ith observation. It is well known that as h_{ii} approaches to 1, the predicted response of the ith observation $\widehat{y}_i$ gets close to y_i. Thus the leverage score has been regarded as the most important influence index indicting how influential the ith observation to the least squares estimator.

Leveraging

When sample size n is super large, the computation of the OLS estimator using the full sample may become extremely expensive. For example, if $p = \sqrt{n}$, the computation of the OLS estimator is $O(n^2)$, which may be infeasible for super large n. Alternatively, one may opt to use leveraging methods, which are presented below.

There are two types of leveraging methods, weighted leveraging methods and unweighted leveraging methods. Let us look at the weighted leveraging methods first.

Algorithm 1. Weighted Leveraging

- **Step 1. Taking a random subsample of size r from the data** Constructing sampling probability $\pi = \{\pi_1, \ldots, \pi_n\}$ for the data points. Draw a random subsample (with replacement) of size $r \ll n$, denoted as (X^*, y^*), from the full sample according to the probability π. Record the corresponding sampling probability matrix $\Phi^* = \text{diag}\left\{\pi_k^*\right\}$.
- **Step 2. Solving a weighted least squares (WLS) problem using the subsample.** Obtain least squares estimate using the subsample. Estimate β via solving a WLS problem on the subsample to get estimate $\widetilde{\beta}$, i.e., solve

$$\arg\min_{\beta} ||\Phi^{*-1/2} y^* - \Phi^{*-1/2} X^* \beta||^2. \tag{5}$$

In the weighted leveraging methods (and unweighted leveraging methods to be reviewed below), the key component is to construct the sampling probability π. An easy choice is $\pi_i = \frac{1}{n}$ for each data point, i.e., uniform sampling probability. The resulting leveraging estimator is denoted as $\widetilde{\beta}_{\text{UNIF}}$. The leveraging algorithm with uniform subsampling probability is very simple to implement but performs poorly in many cases, see Figure 1. The first nonuniform weighted leveraging algorithm was developed by Drineas et al..[6,7] In their weighted leveraging algorithm, they construct sampling probability π_i using the (normalized) sample leverage score as $\pi_i = h_{ii}/\sum_i h_{ii}$. The resulting leveraging estimator is denoted as $\widetilde{\beta}_{\text{BLEV}}$, which yields impressive results: A preconditioned version of this leveraging method 'beats LAPACK's direct dense least squares solver by a large margin on essentially any dense tall matrix'.[10]

An important feature of the weighted leveraging method is, in step 2, to use sampling probability as weight in solving WLS based on the subsample. The weight is analogous to the weight in classic Hansen–Hurwitz estimator.[11] Recall in classic sampling methods, given data $(y_1, \ldots, y_n)$ with sampling probability $(\pi_1, \ldots, \pi_n)$, taking a random sample of size r, denoted by $\left(y_1^*, \ldots, y_r^*\right)$, the Hansen–Hurwitz estimator is $r^{-1}\sum_{i=1}^r y_i^*/\pi_i^*$, where π_i^* is the corresponding sampling probability of y_i^*. It is well known that $r^{-1}\sum_{i=1}^r y_i^*/\pi_i^*$ is an unbiased estimate of $\sum_{i=1}^n y_i$. It was shown in Refs 8 and 9 that weighted leveraging estimator is an approximately unbiased estimator of $\widehat{\beta}_{\text{ols}}$.

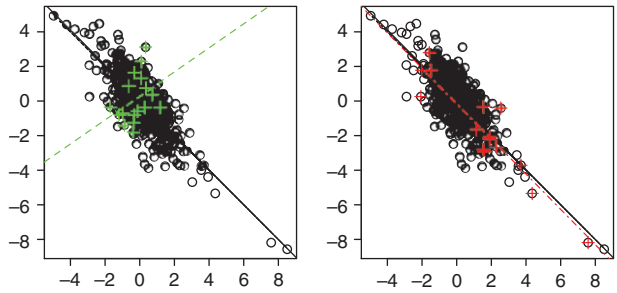


FIGURE 1 | A random sample of $n = 500$ was generated from $y_i = -x_i + \epsilon_i$ where x_i is $t(6)$-distributed, $\epsilon_i \sim N(0, 1)$. The true regression function is in the dotted black line, the data in black circles, and the OLS estimator using the full sample in the black solid line. (a) The uniform leveraging estimator is in the green dashed line. The uniform leveraging subsample is superimposed as green crosses. (b) The weighted leveraging estimator is in the red-dot dashed line. The points in the weighted leveraging subsample are superimposed as red crosses.

It was also noticed in Ref 8 that the variance of the weighted leveraging estimator may be inflated by the data with tiny sampling probability. Subsequently, Ma et al.[8,9] introduced a shrinkage leveraging method. In the shrinkage leveraging method, sampling probability was shrunk to uniform sampling probability, i.e., $\pi_i = \lambda \frac{h_{ii}}{\sum h_{ii}} + (1-\lambda)\frac{1}{n}$, where tuning parameter λ is a constant in $[0, 1]$. The resulting estimator is denoted as $\widetilde{\beta}_{\text{SLEV}}$. It was demonstrated in Refs 8 and 9 that $\widetilde{\beta}_{\text{SLEV}}$ has smaller means squared error than $\widetilde{\beta}_{\text{BLEV}}$ when α is chosen between 0.8 and 0.95 in their simulation studies.

It is worth noting that the first leveraging method is designed primarily to approximate full sample OLS estimate.[6,7] Ma et al.[8] substantially expand the scope of the leveraging methods by applying them to infer the true value of coefficient β. Moreover, if inference of the true value of coefficient β is the major concern, one may use an unweighted leveraging method.[8]

Algorithm 2. Unweighted Leveraging

- **Step 1. Taking a random sample of size r from the data.** This step is the same as step 1 in Algorithm 1.
- **Step 2. Solving an OLS problem using the subsample.** Obtain least squares estimate using the subsample. Estimate β by solving an OLS problem (**instead of WLS problem**) on the sample to get estimate $\widetilde{\beta}^u$, i.e., solve

$$\arg\min_{\beta} ||y^* - X^* \beta||^2. \tag{6}$$

Although the $\widetilde{\beta}^{u}$ is an approximately biased estimator of the full sample OLS estimator $\widehat{\beta}_{\text{ols}}$, Ma et al.[8] show that $\widetilde{\beta}^{u}$ is an unbiased estimator of the true value of coefficient β. Simulation studies also indicate that the unweighted leveraging estimator $\widetilde{\beta}^{u}$ has smaller variance than the corresponding weighted leveraging estimator $\widetilde{\beta}$.

Analytic Framework for Leveraging Estimators

We use the mean squared error (MSE) to compare the performance of the leveraging methods. In particular, MSE can be decomposed in to bias and variance components. We start with deriving the bias and variance of the unweighted leveraging estimator $\widetilde{\beta}$ in Algorithm 1. It is instructive to note that when studying the statistical properties of leveraging estimators there are two layers of randomness (from sampling and subsampling).

From Algorithm 1, we can easily see

$$\widetilde{\beta} = \left(X^{*\mathrm{T}}\Phi^{*-1}X^{*}\right)^{-1} X^{*\mathrm{T}} W^{*} y^{*}.$$

This expression is difficult to analyze as it has both random subsample (X^{*}, y^{*}) and their corresponding sampling probability Φ^{*}. Instead, we rewrite the weighted leveraging estimator in terms of the original data (X, y),

$$\widetilde{\beta} = \left(X^{\mathrm{T}} W X\right)^{-1} X^{\mathrm{T}} W \boldsymbol{y},$$

where $W = \mathrm{diag}(w_1, w_2, \ldots, w_n)$ is an $n \times n$ diagonal random matrix, $w_i = \frac{k_i}{r\pi_i}$, and k_i is the number of times that the ith data point in the random subsample (Recall that step 1 in Algorithm 1 is to take subsample with replacement).

Clearly, the estimator $\widetilde{\beta}$ can be regarded as a function of the random weight vector $w = (w_1, w_2, \ldots, w_n)^{T}$, denoted as $\widetilde{\beta}(w)$. As we take random subsample with replacement, it is easy to see that w has a scaled multinomial distribution,

$$\Pr\left\{w_1 = \frac{k_1}{r\pi_1}, w_2 = \frac{k_2}{r\pi_2}, \ldots, w_n = \frac{k_n}{r\pi_n}\right\} = \frac{r!}{k_1!k_2!, \ldots, k_n!}\pi_1^{k_1}\pi_2^{k_2}\cdots\pi_n^{k_n},$$

with mean $\mathrm{E}w = \mathbf{1}$. By setting w_0, the vector around which we perform our Taylor series expansion, to be the all-ones vector, i.e., $w_0 = \mathbf{1}$, then $\widetilde{\beta}(w)$ can be expanded around the full sample OLS estimate $\widehat{\beta}_{\text{ols}}$, i.e., $\widetilde{\beta}(\mathbf{1}) = \widehat{\beta}_{\text{ols}}$. From this, we have the following lemma.

LEMMA 1. *Let $\widetilde{\beta}$ be the weighted leveraging estimator. Then, a Taylor expansion of $\widetilde{\beta}$ around the point $w_0 = \mathbf{1}$* yields

$$\widetilde{\beta} = \widehat{\beta}_{\text{ols}} + \left(X^{\mathrm{T}}X\right)^{-1} X^{\mathrm{T}}\mathrm{diag}\left\{\widehat{e}\right\}(w - \mathbf{1}) + R_{W}, \quad (7)$$

where $\widehat{e} = y - X\widehat{\beta}_{\text{ols}}$ is the OLS residual vector, and $R_{W} = o_p(\|w - w_0\|)$ is the Taylor expansion remainder.

Note that the detailed proofs of all lemma and theorems can be found in Ref 8.

Given Lemma 1, we can establish the following lemma, which provides expressions for the conditional and unconditional expectations and variances for the weighted leveraging estimator.

LEMMA 2. *The conditional expectation and conditional variance for the weighted leveraging estimator are given by:*

$$E\left\{\widetilde{\beta}|y\right\} = \widehat{\beta}_{\text{ols}} + E\left\{R_{W}\right\}; \quad (8)$$

$$\mathrm{Var}\left\{\widetilde{\beta}|y\right\} = \left(X^{\mathrm{T}}X\right)^{-1} X^{\mathrm{T}}\mathrm{diag}\left\{\widehat{e}\right\}\mathrm{diag}\left\{\frac{1}{r\boldsymbol{\pi}}\right\} \mathrm{diag}\left\{\widehat{e}\right\} X\left(X^{\mathrm{T}}X\right)^{-1} + \mathrm{Var}\left\{R_{W}\right\}, \quad (9)$$

where $\mathbf{W}$ specifies the probability distribution used in the sampling and rescaling steps. The unconditional expectation and unconditional variance for the weighted leveraging estimator are given by:

$$E\left\{\widetilde{\beta}\right\} = \beta_0; \quad (10)$$

$$\mathrm{Var}\left\{\widetilde{\beta}\right\} = \sigma^2\left(X^{\mathrm{T}}X\right)^{-1} + \frac{\sigma^2}{r}\left(X^{\mathrm{T}}X\right)^{-1} X^{\mathrm{T}} \mathrm{diag}\left\{\frac{\left(1 - h_{ii}\right)^2}{\pi_i}\right\} X\left(X^{\mathrm{T}}X\right)^{-1} + \mathrm{Var}\left\{R_{W}\right\}. \quad (11)$$

Equation (8) states that, when the $\mathrm{E}\{R_{W}\}$ term is negligible, i.e., when the linear approximation is valid, then, conditioning on the observed data y, the estimate $\widetilde{\beta}$ is approximately unbiased, relative to the full sample OLS estimate $\widehat{\beta}_{\text{ols}}$; and Eq. (10) states that the estimate $\widetilde{\beta}$ is unbiased, relative to the true value β_0 of the parameter vector β. That is, given a particular dataset (X, y), the conditional expectation result of Eq. (8) states that the leveraging estimators can approximate well $\widehat{\beta}_{\text{ols}}$; and, as a statistical inference procedure for arbitrary datasets, the unconditional expectation

result of Eq. (10) states that the leveraging estimators can infer well β_0.

Both the conditional variance of Eq. (9) and the (second term of the) unconditional variance of Eq. (11) are inversely proportional to the subsample size r; and both contain a sandwich-type expression, the middle of which depends on how the leverage scores interact with the sampling probabilities. Moreover, the first term of the unconditional variance, $\sigma^2(X^{\mathrm{T}}X)^{-1}$, equals the variance of the OLS estimator; this implies, e.g., that the unconditional variance of Eq. (11) is larger than the variance of the OLS estimator, which is consistent with the Gauss-Markov theorem.

Now consider the unweighted leveraging estimator, which is different from the weighted leveraging estimator, in that no weight is used for least squares based on the subsample. Thus, we shall derive the bias and variance of the unweighted leveraging estimator $\widetilde{\beta}^u$. To do so, we first use a Taylor series expansion to get the following lemma.

LEMMA 3. *Let $\widetilde{\beta}^u$ be the unweighted leveraging estimator. Then, a Taylor expansion of $\widetilde{\beta}^u$ around the point $w_0 = r\pi$ yields*

$$\widetilde{\beta}^u = \widehat{\beta}_{\mathrm{wls}} + \left(X^{\mathrm{T}}W_0X\right)^{-1}X^{\mathrm{T}}\mathrm{diag}\left\{\widehat{\mathrm{e}}_w\right\}(\boldsymbol{w} - r\pi) + R^u, \tag{12}$$

where $\widehat{\beta}_{\mathrm{wls}} = \left(X^TW_0X\right)^{-1}XW_0y$ is the full sample WLS estimator, $\widehat{\mathrm{e}}_w = \boldsymbol{y} - X\widehat{\beta}_{\mathrm{wls}}$ is the LS residual vector, $W_0 = diag\{r\pi\}$, and R^{u} is the Taylor expansion remainder.

This lemma is analogous to Lemma 1. In particular, here we expand around the point $w_0 = r\pi$ since $\mathrm{E}\{w\} = r\pi$ when no reweighting takes place.

Given this Taylor expansion lemma, we can now establish the following lemma for the mean and variance of unweighted leveraging estimator, both conditioned and unconditioned on the data y.

LEMMA 4. *The conditional expectation and conditional variance for the unweighted leveraging estimator are given by:*

$$E\left\{\widetilde{\beta}^u|y\right\} = \widehat{\beta}_{\mathrm{wls}} + E\left\{R^u\right\};$$

$$\mathrm{Var}\left\{\widetilde{\beta}^u|\boldsymbol{y}\right\} = \left(X^{\mathrm{T}}W_0X\right)^{-1}X^{\mathrm{T}}\mathrm{diag}\left\{\widehat{e}_w\right\}W_0$$
$$\mathrm{diag}\left\{\widehat{e}_w\right\}X\left(X^{\mathrm{T}}W_0X\right)^{-1} + \mathrm{Var}\left\{R^u\right\},$$

where $\mathrm{W}_0 = diag\{\mathrm{r}\pi\}$, and where $\widehat{\beta}_{\mathrm{wls}} = \left(X^{\mathrm{T}}W_0X\right)^{-1}XW_0y$ is the full sample WLS estimator. The unconditional expectation and unconditional variance for the unweighted leveraging estimator are given by,

$$E\left\{\widetilde{\beta}^u\right\} = \beta_0;$$

$$\mathrm{Var}\left\{\widetilde{\beta}^u\right\} = \sigma^2\left(X^{\mathrm{T}}W_0X\right)^{-1}X^{\mathrm{T}}W_0^2X\left(X^{\mathrm{T}}W_0X\right)^{-1}$$
$$+\sigma^2\left(X^{\mathrm{T}}W_0X\right)^{-1}X^{\mathrm{T}}\mathrm{diag}\left\{I - P_{X,W_0}\right\}W_0$$
$$\mathrm{diag}\left\{I - P_{X,W_0}\right\}X\left(X^{\mathrm{T}}W_0X\right)^{-1} + \mathrm{Var}\left\{R^u\right\} \tag{13}$$

where $P_{X,W_0} = X\left(X^{\mathrm{T}}W_0X\right)^{-1}X^{\mathrm{T}}W_0$.

These two expectation results in this lemma state: (1), when $\mathrm{E}\{R^u\}$ is negligible, then, conditioning on the observed data y, the estimator $\widetilde{\beta}^u$ is approximately unbiased, relative to the full sample WLS estimator $\widehat{\beta}_{\mathrm{wls}}$ and (2) the estimator $\widetilde{\beta}^u$ is unbiased, relative to the true value β_0 of the parameter vector β.

As expected, when the leverage scores are all the same, the variance in Eq. (13) is the same as the variance of uniform random sampling. This is expected since, when reweighting with respect to the uniform distribution, one does not change the problem being solved, and thus the solutions to the weighted and unweighted OLS problems are identical. More generally, the variance is not inflated by very small leverage scores, as it is with weighted leveraging estimator. For example, the conditional variance expression is also a sandwich-type expression, the center of which is $W_0 = \mathrm{diag}\{rh_{ii}/n\}$, which is not inflated by very small leverage scores.

Computation

The running time of the leveraging methods depends on both the time to construct the probability π, and the time to solve the least squares based on the subsamples. Solving least squares based on subsample of size r requires $O(rp^2)$ time. On the other hand, when constructing π involves leverage score h_{ii}, the running time is dominated by the computation of those scores. The leverage score h_{ii} can be computed using singular value decomposition (SVD) of X. The SVD of X can be written as

$$X = U\Lambda V^{\mathrm{T}},$$

where U is an $n \times p$ orthogonal matrix whose columns contain the left singular vectors of X, V is an $p \times p$ orthogonal matrix whose columns contain the right singular vectors of X, $p \times p$ matrix $\Lambda = \mathrm{diag}\{\lambda_i\}$, and λ_i, $i = 1, \ldots, p$ are singular values of X. Hat matrix H can alternatively be expressed as $H = UU^{\mathrm{T}}$. Then,

TABLE 1 | Mean (Standard Deviation) of Sample MSEs over 100 Replicates for different subsample size.

Algorithms	200	500	1000	2000
Uniform	8.46e-5(2.44e-5)	1.32e-5(8.16e-7)	9.89e-6(2.38e-7)	8.80e-6(9.20e-8)
Weighted	7.72e-5(2.64e-5)	1.24e-5(6.21e-7)	9.61e-6(2.47e-7)	8.80e-6(1.60e-7)
Unweighted	7.03e-5(2.36e-5)	1.18e-5(5.14e-7)	9.28e-6(1.99e-7)	8.51e-6(6.88e-8)
Shrinkage	7.02e-5(1.88e-5)	1.24e-5(6.50e-7)	9.67e-6(2.80e-7)	8.79e-6(1.49e-7)

MSE, mean squared error.

the leverage score of the ith observation can be expressed as

$$h_{ii} = ||\boldsymbol{u}_i||^2, \tag{14}$$

where u_i is the ith row of U. The exact computation of h_{ii}, for $i = 1, \ldots, n$, in Eq. (14) requires $O(np^2)$ time.[12]

The fast randomized algorithm from Ref 13 can be used to compute approximations to the leverage scores of X in $o(np^2)$ time. The algorithm of Ref 13 takes an arbitrary $n \times p$ matrix X as input.

- Generate an $r_1 \times n$ random matrix Π_1 and a $p \times r_2$ random matrix Π_2.
- Let R be the R matrix from a QR decomposition of $\Pi_1 X$.
- Compute and return the leverage scores of the matrix $XR^{-1}\Pi_2$.

For appropriate choices of r_1 and r_2, if one chooses Π_1 to be a Hadamard-based random projection matrix, then this algorithm runs in $o(np^2)$ time, and it returns $1 \pm \epsilon$ approximations to all the leverage scores of X.[13] In addition, with a high-quality implementation of the Hadamard-based random projection, this algorithm runs faster than traditional deterministic algorithms based on LAPACK.[10,14] Ma et al.[8] used two variants of this fast algorithm of Ref 13: the binary fast algorithm, where (up to normalization) each element of Π_1 and Π_2 is generated i.i.d. from $\{-1, 1\}$ with equal sampling probabilities and the Gaussian Fast algorithm, where each element of Π_1 is generated i.i.d. from a Gaussian distribution with mean zero and variance $1/n$ and each element of Π_2 is generated i.i.d. from a Gaussian distribution with mean zero and variance $1/p$. Ma et al.[8] showed that these two algorithms perform very well.

Real Example

Human Connectome Project (HPC) aims to delineate human brain connectivity and function in the healthy adult human brain and to provide these data as public resource for biomedical research. In HPC, functional magnetic resonance imaging (fMRI) data were collected as follows. Scanners acquire the slices of three-dimensional (3D) functional image, resulting in brain oxygenation-level dependent (BOLD) signal samples at different layers of the brain while subjects perform certain task. In this example, fMRI BOLD intensities released in 2013[15] were used. The BOLD signals at approximately 0.2 million voxels in each subject's brain were taken at 176 consecutive time points during a task named emotion. Most of conventional methods can only analyze single subject imaging data due to the daunting computational cost. We stacked on five subjects' BOLD signal together, resulting in a data matrix of $1,004,738 \times 176$, in which each row is a voxel and each column is a time point. We use the BOLD signal at the last time point as response y, the rest 175 BOLD measurements with additional intercept as the predictor X, which is $1,004,738 \times 176$. We fit the linear regression model of (1) to the data.

We applied four leveraging methods to fit the linear regression model. In particular, uniform leveraging, weighted leveraging, unweighted leveraging, and shrinkage leveraging with $\lambda = 0.9$ were compared with subsample size ranging from $r = 200$ to 2000. The sample MSE $= (y - \hat{y})^{\mathrm{T}} (y - \hat{y}) / n$ was calculated. We repeated applying the leveraging methods for one hundred times at each subsample size. The mean and standard deviations of MSEs were reported in Table 1. From Table 1, one can clearly see that the unweighted leveraging outperforms other methods in terms of sample MSE. The leveraging methods are significantly better than uniform leveraging especially when the subsample size is small.

As for computing time, the OLS takes 70.550 CPU seconds to complete in full sample whereas leveraging methods only need about 28 CPU seconds to complete. Particularly, in step 1 of weighted leveraging method, SVD takes 22.277 CPU seconds, 5.848 seconds for calculating leverages, and 0.008 seconds for sampling. Step 2 takes additional 0.012 CPU seconds.

CONCLUSION

In the article, we reviewed the emerging family of leveraging methods in the context of linear regression. For linear regression, there are many methods available for computing the exact least squares estimator of the full sample using divide-and-conquer approaches. However, none of them are designed to visualize the big data. Without looking at the data, exploratory data analysis is impossible. In contrast, step 1 in leveraging algorithms provides a random 'sketch' of the data. Thus visualization of the subsample obtained in step 1 of leveraging methods enables a surrogate visualization of the full data. While outliers are likely to be the high leverage data points and thus likely selected into the subsample in leveraging methods, we suggest that model diagnostic may be performed after leveraging estimates are calculated. Subsequently, outliers can be identified and removed. Finally, the leveraging estimates may be calculated in the subsample without outliers.

ACKNOWLEDGMENTS

We thank Tianming Liu Lab for providing the fMRI data. This research was partially supported by National Science Foundation grants DMS-1055815, 12228288, and 1222718.

REFERENCES

1. Suchard MA, Wang Q, Chan C, Frelinger J, Cron A, West M. Understanding GPU programming for statistical computation: studies in massively parallel massive mixtures. *J Comput Graph Stat* 2010, 19: 419–438.
2. Schatz MC, Langmead B, Salzberg SL. Cloud computing and theDNA data race. *Nat Biotechnol* 2010, 28:691.
3. Pratas F, Trancoso P, Sousa L, Stamatakis A, Shi G, Kindratenko V. Fine-grain parallelism using multi-core, cell/BE, and GPU systems. *Parallel Comput* 2012, 38:365–390.
4. Efron B. Bootstrap methods: another look at the jackknife. *Ann Stat* 1979, 7:1–26.
5. Wu CFJ. Jackknife, bootstrap and other resampling methods in regression analysis. *Ann Stat* 1986, 14:1261–1295.
6. Drineas P, Mahoney MW, Muthukrishnan S. Sampling algorithms for l_2 regression and applications. In *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms*, Miami, Florida, 2006, 1127–1136.
7. Drineas P, Mahoney MW, Muthukrishnan S, Sarlós T. Faster least squares approximation. *Numer Math* 2010, 117:219–249.
8. Ma P, Mahoney M, Yu B. A statistical perspective on algorithmic leveraging. ArXiv e-prints, June 2013.
9. Ma P, Mahoney M, Yu B. A statistical perspective on algorithmic leveraging. In *Proceedings of the 31st International Conference on Machine Learning*, Beijing, China, 2014, 91–99.
10. Avron H, Maymounkov P, Toledo S. Blendenpik: Supercharging LAPACK's least-squares solver. *SIAM J Sci Comput* 2010, 32:1217–1236.
11. Hansen M, Hurwitz W. On the theory of sampling from a finite population. *Ann Math Stat* 1943, 14:333–362.
12. Golub G, Van Loan C. *Matrix Computations*. Baltimore: Johns Hopkins University Press; 1996.
13. Drineas P, Magdon-Ismail M, Mahoney MW, Woodruff DP. Fast approximation of matrix coherence and statistical leverage. *J Mach Learn Res* 2012, 13:3475–3506.
14. Gittens A, Mahoney MW. Revisiting the Nyström method for improved large-scale machine learning. Technical Report, Preprint: arXiv:1303.1849, 2013.
15. Van Essen DC, Smith SM, Barch DM, Behrens TE, Yacoub E, Ugurbil K. The WU-Minn Human Connectome Project: an overview. *Neuroimage* 2013, 80:62–79.