Accelerated Sparse Linear Regression via Random Projection

Weizhong Zhang*, Lijun Zhang†, Rong Jin‡, Deng Cai*, Xiaofei He*
*State Key Lab of CAD&CG, College of Computer Science, Zhejiang University, Hangzhou, China
†National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China
‡Alibaba Group, Seattle, USA
{zhangweizhongzju, dengcai, xiaofeihe}@gmail.com, zhanglj@lamda.nju.edu.cn, jinrong.jr@alibaba-inc.com

Abstract

In this paper, we present an accelerated numerical method based on random projection for sparse linear regression. Previous studies have shown that under appropriate conditions, gradient-based methods enjoy a geometric convergence rate when applied to this problem. However, the time complexity of evaluating the gradient is as large as $\mathcal{O}(nd)$, where n is the number of data points and d is the dimensionality, making those methods inefficient for large-scale and high-dimensional dataset. To address this limitation, we first utilize random projection to find a rank-k approximator for the data matrix, and reduce the cost of gradient evaluation to $\mathcal{O}(nk + dk)$, a significant improvement when k is much smaller than d and n. Then, we solve the sparse linear regression problem via a proximal gradient method with a homotopy strategy to generate sparse intermediate solutions. Theoretical analysis shows that our method also achieves a global geometric convergence rate, and moreover the sparsity of all the intermediate solutions are well-bounded over the iterations. Finally, we conduct experiments to demonstrate the efficiency of the proposed method.

Introduction

In this study, we focus on developing an efficient algorithm based on random projection for sparse linear regression (Huang, Ma, and Zhang 2008), which is closely related to sparse recovery (Becker, Bobin, and Candès 2011) and compressed sensing(Candes and Tao 2006). Suppose we observe the data matrix $X = (\mathbf{x}_1, \ldots, \mathbf{x}_n) \in \mathbb{R}^{d\times n}$ and the corresponding response vector $\mathbf{y} = (y_1, \ldots, y_n)^T \in \mathbb{R}^n$, where each $\mathbf{x}_i \in \mathbb{R}^d$ is a vector representation for the i-th observation and y_i is its response value. The goal of sparse linear regression is to find a sparse linear prediction function $f(\mathbf{x}) = \mathbf{x}^T\boldsymbol{\beta}$ such that each y_i can be well approximated by $f(\mathbf{x}_i)$, where $\boldsymbol{\beta} \in \mathbb{R}^d$ is a sparse vector composed of the model coefficients. In this work, we are interested in estimating $\boldsymbol{\beta}$ in a special case when the dimension d of $\mathbf{x}_i$ is very high, which can be much larger than the sample size, i.e. $d > n$, thus the problem here is under-determined.

One common approach is to learn $\boldsymbol{\beta}$ by solving a ℓ_1-regularized Least-Square (ℓ_1-LS) problem, known as *Lasso* in statistics (Tibshirani 1996),

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^d} \frac{1}{2n}||\mathbf{y} - X^T\boldsymbol{\beta}||_2^2 + \lambda||\boldsymbol{\beta}||_1 \quad (1)$$

where λ is the regularization parameter that controls the sparsity of the model. Here $||\cdot||_2$ denotes the standard ℓ_2 norm for a vector, and $||\boldsymbol{\beta}||_1 = \sum_i^d |\beta_i|$ is the ℓ_1 norm. ℓ_1-LS problem has been studied extensively due to its important applications in machine learning, signal processing and statistics. Plenty of algorithms (Figueiredo, Nowak, and Wright 2007; Kim et al. 2007; Nesterov 2013; Xiao and Zhang 2013) have been developed for efficiently solving the optimization problem (1). For instance, in (Xiao and Zhang 2013), the authors propose a proximal gradient homotopy method to solve this problem by solving a sequence of optimization problems in form (1) with the regularization parameter λ decreasing along the sequence. They prove that under some common assumptions, they can obtain a global geometric convergence rate and the intermediate solutions over the iterations are always sparse. However, despite the provable fast convergence rates, most of these algorithms need to compute $XX^T\boldsymbol{\beta}_t$ for gradient evaluation at each iteration, where $\boldsymbol{\beta}_t$ is the intermediate solution at the t-th iteration. It costs $\mathcal{O}(nd)$ flops for generic dense matrix X, leading to a high computational cost when both n and d are large.

In this study, we adopt the key idea in randomized matrix algorithms (Mahoney 2011; Zhang et al. 2013; 2014) to develop a novel efficient algorithm for the sparse linear regression problem, due to their high efficiency in data intensive problems. Various of random matrix algorithms (Drineas et al. 2004; Frieze, Kannan, and Vempala 2004; Sarlos 2006; Drineas, Kannan, and Mahoney 2006; Drineas, Mahoney, and Muthukrishnan 2008; Mahoney and Drineas 2009; Drineas et al. 2011; Lu et al. 2013) have been developed in these years to accelerate the corresponding algorithms in large-scale data analysis. They typically construct a randomized sketch of the input matrix X to downsize the scale of the original problem. And then they solve the downsized problem to obtain a approximate solution. Based on the way they construct the sketch they can be categorized into two categories roughly, i.e., random sampling based methods and random projection based methods. In random sampling based methods (Drineas, Mahoney, and

Muthukrishnan 2008; Mahoney and Drineas 2009; Drineas et al. 2011), the sketch consists of some columns sampled from the columns of X according to some specific probability distribution. While in random projection based methods, the columns in the sketch are some linear combinations of the columns from X. In our method, we adopt the random projection method instead of the random sampling method, since our problem is under-determined and sampling would make it worse. To the best of our knowledge, it is the first attempt to exploit randomization for fast solving sparse linear regression problems.

To be precise, we develop an efficient two-phase algorithm based on random projection for sparse linear regression. In phase one, we approximate X with a low rank matrix $\widehat{X} = (QW)^T$ by using random projection, where $Q \in \mathbb{R}^{n\times k}$ and $W \in \mathbb{R}^{k\times d}$. Thus $XX^T\boldsymbol{\beta}_t$ can be approximated by $W^TQ^TQW\boldsymbol{\beta}_t$ and in this way the computing cost can be reduced from $\mathcal{O}(nd)$ to $\mathcal{O}(dk+nk)$. When $k \ll \min\{n,d\}$, this improvement in efficiency would be significant. If the data matrix X is low rank or approximately low rank, which is common in real applications, the matrix $\widehat{X}$ would be a good approximator. In phase two, based on the approximate matrix $\widehat{X}$, we update the vector $\boldsymbol{\beta}_t$ in each iteration by employing a proximal gradient method using a homotopy strategy, which is similar to the method in (Xiao and Zhang 2013).

After giving the rough sketch of our method, we would like to point out the main contributions of this work below:

- We reduce the computing complexity of each iteration from $\mathcal{O}(nd)$ to $\mathcal{O}(dk+nk)$, meanwhile the global geometric convergence rate is preserved.
- Compared to the recent well known method (Xiao and Zhang 2013), we need to run the composite gradient mapping only once for each regularization parameter.
- The sparsity of the intermediate solutions in our approach is well-bounded. The number of the nonzero entries in $\boldsymbol{\beta}_t$ is no more than two times of that in $\boldsymbol{\beta}_*$.

Related Work

In this section, we briefly review the recent work on the optimization methods for ℓ_1-LS problem, the random matrix algorithms for least square problem and low-rank matrix approximation.

Previous Algorithms for ℓ_1-LS Problem

Traditional methods (Daubechies, Defrise, and De Mol 2003; Nesterov 2013; Hale, Yin, and Zhang 2008; Wright, Nowak, and Figueiredo 2009) developed in the past decades for problem (1) are always based on composite gradient mapping (Nesterov 2004) and we call them proximal gradient methods. Given the current solution $\boldsymbol{\beta}_t$, they update the solution by using the following rule

$$\boldsymbol{\beta}_{t+1} = \arg\min_{\boldsymbol{\beta}}\{f(\boldsymbol{\beta}_t) + \nabla f(\boldsymbol{\beta}_t)^T(\boldsymbol{\beta}-\boldsymbol{\beta}_t) + \frac{L_t}{2}||\boldsymbol{\beta}-\boldsymbol{\beta}_t||_2^2 + \lambda||\boldsymbol{\beta}||_1\}, \quad (2)$$

where $f(\boldsymbol{\beta}_t) = \frac{1}{2n}||X^T\boldsymbol{\beta}_t - \mathbf{y}||_2^2$ and L_t is a parameter chosen at each iteration. The major computational efforts of them are used to compute the gradient $\nabla f(\boldsymbol{\beta}_t) = \frac{1}{n}X(X^T\boldsymbol{\beta}_t - \mathbf{y})$, which costs $\mathcal{O}(nd)$ flops for generic dense matrix X.

Since in the under-determined case the object function is not strongly convex, the convergence rate of the traditional proximal gradient methods can only be $\mathcal{O}(1/T)$, T is the iteration number. Some accelerated algorithms (Bioucas-Dias and Figueiredo 2007; Wright, Nowak, and Figueiredo 2009; Wen et al. 2010; Becker, Bobin, and Candès 2011) are then proposed to improve the convergence rate to $\mathcal{O}(1/T^2)$ by generating several concurrent sequences of iterates.

Recently, a proximal gradient homotopy method (Xiao and Zhang 2013) is proposed to improve the efficiency of the existing methods further. Its key idea is to solve a sequence of ℓ_1-LS problem over the stages with the values of the regularization parameter λ decreasing exponentially as the stage goes on. The main benefit they get from the homotopy strategy is that all the solutions along the homotopy path are sparse, which makes the objective function strongly convex along this path. Thus, the convergence rate can be improved to be geometric. The limitation of this method is that the times it updates the solution in each intermediate problem is unknown and always more than once. (Zhang et al. 2015) adopt this idea to compressive sensing and in their approach they update the solution only once for each regularization parameter. However, different from our problem, the sensing matrix X in compressive sensing is pre-designed, like sub-Gaussian random matrix. Thus, its theoretical analysis may not be always true in our case. More importantly, the computational complexity of each iteration is still $\mathcal{O}(nd)$ in all these accelerated algorithms, which would be prohibitive when both n and d are too large.

Random Algorithms for Least Square Problem and Low-rank Matrix Approximation

Below we briefly introduce some random matrix algorithms for Least Square problem and Low-rank Approximation problem since they are closely related to the key idea of our method and they are at the center of the recent developments.

To accelerate the existing algorithms for least square problem

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^d} \frac{1}{2n}||\mathbf{y} - X^T\boldsymbol{\beta}||_2, \quad (3)$$

where $X \in \mathbb{R}^{d\times n}, \mathbf{y} \in \mathbb{R}^n$ and $\boldsymbol{\beta} \in \mathbb{R}^d$, the researchers (Drineas et al. 2011; Sarlos 2006) typically find an approximate solution by solving the following approximate problem

$$\boldsymbol{\beta}_{opt} = \arg\min_{\boldsymbol{\beta}\in\mathbb{R}^d} \frac{1}{2n}||Z(\mathbf{y} - X^T\boldsymbol{\beta})||_2, \quad (4)$$

where $Z \in \mathbb{R}^{l\times n}$ is the data dependent matrix, the parameter l is usually much smaller than n. Since $l \ll n$, problem (4) has much smaller size compared to problem (3). For random sampling based algorithms (Drineas et al. 2011), Z is a data dependent random sampling matrix constructed from an importance sampling distribution to capture the non-uniformity structure of X, which is defined by

the statistical leverage scores (Mahoney and Drineas 2009). And for random projection based algorithms (Drineas et al. 2011), it is a data-independent random projection for constructing the sketch and its elements are the coefficients of the linear combinations. Recently, they show that by using random algorithm, they can find an approximate solution $\tilde{\boldsymbol{\beta}}_{opt}$ in $\mathcal{O}(dn\log(d/\epsilon) + \frac{d^3\log^2 n}{\epsilon})$ time with the error $||\tilde{\boldsymbol{\beta}}_{opt} - \boldsymbol{\beta}_{opt}||_2 \leq \sqrt{\epsilon}\big(\kappa(X)\sqrt{\gamma^{-2}-1}\big)||\boldsymbol{\beta}_{opt}||_2$, here $\kappa(X)$ is the conditional number of X, γ is a parameter defining the amount of the mass of $\mathbf{y}$ inside the column space of X^T. Compared to the cost $\mathcal{O}(nd^2)$ of the traditional methods for the least square problem (3), such as Cholesky decomposition, QR decomposition and Singular Value Decomposition (SVD), they are indeed much more efficient. Different from our problem, the least square problem they consider is overconstrained, i.e., $n > d$. In addition, they do not care about the sparsity of the solution in their problem.

In Low-rank Matrix Approximation problems, given a low rank $d \times n$ matrix X with $\text{rank}(X) = r \ll \min\{d, n\}$, the goal is to find a good approximation $\widehat{X}$ to X of rank k with $k \leq r$. As we know, we can obtain the best approximation $\widehat{X}$ from SVD with respect to the Frobenius norms. To be precise, if we have the SVD of X, say $X = U\Sigma V^T$, where $U = (U_k, U_{k,\perp}) = [u_1, u_2, ..., u_r] \in \mathbb{R}^{d\times r}$ and $V = (V_k, V_{k,\perp}) = [v_1, v_2, ..., v_r] \in \mathbb{R}^{n\times r}$ are two column orthonormal matrices and $\Sigma = \text{diag}(\Sigma_k, \Sigma_{k,\perp}) = \text{diag}(\sigma_1, ..., \sigma_r) \in \mathbb{R}^{r\times r}$, $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r > 0$, then $\widehat{X} = U_k\Sigma_k V_k^T$ is the best rank-k approximation to X. Since the computing complexity of SVD is typically $\mathcal{O}(d^2n + n^3)$, which is prohibitive in high dimensional and data intensive applications, some random algorithms (Drineas et al. 2004; Frieze, Kannan, and Vempala 2004; Drineas, Kannan, and Mahoney 2006) are developed to accelerate this procedure. These methods usually use random sampling (Drineas, Mahoney, and Muthukrishnan 2008; Mahoney and Drineas 2009) or random projection to identify a subspace C_k that captures most of the actions of the matrix X, then they project X to C_k to obtain the desired low-rank approximation $\widehat{X}$. The main steps of these methods are as follows:

- Generate the random sampling or random projection matrix $Z \in \mathbb{R}^{n\times(k+p)}$ with $p \geq 0$,
- Compute the sketch C of X: $C = XZ$,
- Obtain the rank-k approximate matrix $\widehat{X}$ by projecting X onto C_k: $\widehat{X} = P_{C_k}X$, where C_k is the best rank-k approximation to the matrix C, P is the projection operator.

In these random algorithms, the relative error bound $||X - \widehat{X}|| \leq (1+\epsilon)||X - P_{U_k}X||$ is often used to evaluate the precision of the random algorithms, here $||\cdot||$ is the Frobenius norm or spectral of the matrix. Their computing complexity can be reduced to $\mathcal{O}(dnk)$ or even $\mathcal{O}(dn\log(k))$ when Z is a structured matrix, which is a quite significant improvement in practice. In addition, compared to the traditional methods, they are much more robust and can be reorganized to exploit parallel computing architectures.

Preliminary and Notation

In this section, we present the mathematical setups and a few notions, which are necessary for the theoretical analysis of our methods and also widely used in sparse linear regression, lasso and compressed sensing.

Firstly, we assume that there exists a sparse linear prediction function $f_*(\mathbf{x}) = \mathbf{x}^T\boldsymbol{\beta}_*$, where $\boldsymbol{\beta}_*$ is a sparse vector with $||\boldsymbol{\beta}_*||_0 = s \ll d$, such that each y_i can be well approximated by $f_*(\mathbf{x}_i)$. To this end, we define ε as the mean squared regression error made by $f_*(\cdot)$ over the training examples, i.e.

$$\varepsilon^2 := \frac{1}{n}\sum_{i=1}^{n}(f_*(\mathbf{x}_i) - y_i)^2.$$

Since $f_*(\cdot)$ is assumed to be an accurate prediction function, we assume that ε is small. Our goal is to efficiently recover the sparse vector $\boldsymbol{\beta}_*$ and consequentially the sparse linear prediction function $f_*(\cdot)$.

Then, following the work (Koltchinskii 2011), we introduce a restricted eigenvalue condition and a restricted correlation (Bickel, Ritov, and Tsybakov 2009) for the feature matrix X.

Definition 1 *(Restricted Eigenvalue) Given an integer $s > 0$, we say that a matrix X satisfies the restricted eigenvalue condition at sparsity level s if there exist positive constants γ_ℓ and γ_u such that*

$$\begin{aligned}\gamma_\ell &:= \inf\left\{\frac{1}{\sqrt{n}}||X^T\mathbf{u}||_2 : ||\mathbf{u}||_2 = 1, ||\mathbf{u}||_0 \leq s\right\}\\ \gamma_u &:= \sup\left\{\frac{1}{\sqrt{n}}||X^T\mathbf{u}||_2 : ||\mathbf{u}||_2 = 1, ||\mathbf{u}||_0 \leq 2s\right\}.\end{aligned}$$

Definition 2 *(Restricted Correlation) We denote $\mathcal{S}(\boldsymbol{\beta})$ as the support set of $\boldsymbol{\beta}$, which is composed of the nonzero components of $\boldsymbol{\beta}$. Given an integer $s > 0$, we define the restricted correlation of the data matrix X as*

$$\rho := \max\Big\{\frac{|\boldsymbol{\beta}^T XX^T\boldsymbol{\beta}'|}{|X^T\boldsymbol{\beta}|_2|X^T\boldsymbol{\beta}'|_2} : ||\boldsymbol{\beta}||_0 \leq 2s, \\ ||\boldsymbol{\beta}'||_0 \leq s \text{ and } \mathcal{S}(\boldsymbol{\beta}) \cap \mathcal{S}(\boldsymbol{\beta}') = \emptyset\Big\}.$$

At last, we assume our data matrix X in the sequence satisfies the restricted eigenvalue and restricted correlation conditions with parameters γ_ℓ, γ_u and ρ respectively.

Accelerated Sparse Linear Regression via Random Projection

As described in the introduction section, our method is composed of two steps. It computes the approximation $\widehat{X}$ for the data matrix X at first. And then, using the homotopy strategy, it recovers the optimal pattern $\boldsymbol{\beta}_*$ by solving a sequence of subproblems in form (5). In this section, we will discuss the key ideas and the detail steps (Algorithm 1) of our approach below.

In the first step, we utilize the randomized matrix algorithms to get a low-rank approximation for X efficiently. As

Algorithm 1 Accelerated Sparse Linear Regression via Random Projection (SLRviaRP)

Input: the data matrix $X, \mathbf{y}, \lambda_0, \lambda_{\min}, \gamma, k, \eta$
// Compute the approximation $\widehat{X}$:
Sample a $d \times k$ random matrix Z with $Z_{ij} \sim \mathcal{N}(0,1)$
Compute the QR decomposition of $X^T Z$, i.e., $X^T Z = QR$
Approximate X by $\widehat{X} = (XQ)Q^T = W^T Q^T$
// Recover the sparse vector $\boldsymbol{\beta}$:
Initialize $\boldsymbol{\beta}_0 = \mathbf{0}$
for $t = 0$ to $T-1$ **do**
  Update:

$$\boldsymbol{\beta}_{t+1} = \min_{\boldsymbol{\beta}\in\mathbb{R}^d} \Big\{ -\frac{1}{n}(\boldsymbol{\beta}-\boldsymbol{\beta}_t)^T \widehat{X}(\mathbf{y} - \widehat{X}^T\boldsymbol{\beta}_t) + \lambda_t|\boldsymbol{\beta}|_1 + \frac{\gamma}{2}|\boldsymbol{\beta}-\boldsymbol{\beta}_t|_2^2 \Big\}$$

  where $\lambda_t = \max(\lambda_{\min}, \lambda_0\eta^t)$.
end for
Return: $\boldsymbol{\beta}_T$

our problem is highly under-determined since $n < d$, random sampling will make it worse. Thus, we adopt random projection instead of random sampling based method to approximate X with a low rank matrix $\widehat{X}$ (see Algorithm 4.1 in (Halko, Martinsson, and Tropp 2011)). We first compute a Gaussian random matrix $Z \in \mathbb{R}^{d\times k}$, where $k \ll d$ and each entry $Z_{i,j}$ is sampled independently from an Gaussian distribution $\mathcal{N}(0,1)$. We then compute the QR decomposition of $X^T Z$, i.e. $X^T Z = QR$, where $Q \in \mathbb{R}^{n\times k}$ is an orthogonormal matrix of rank k and $R \in \mathbb{R}^{k\times k}$ is an upper triangle matrix. Actually, Q is composed of some orthogonal basis of X^T. We finally approximate X by projecting X^T to the space spanned by the columns of Q, i.e., $\widehat{X} = XQQ^T = W^T Q^T$, where $W = (XQ)^T \in \mathbb{R}^{k\times d}$. At last, we should note here that the QR decomposition for $X^T Z$ can be computed in $2nk^2$ floating-point calculations, which is acceptable when k is relatively small. And in many real applications, the data matrix X is always a low-rank or approximately low-rank matrix, so $\widehat{X}$ is still an accurate approximator for X even when we set k to be a small integer. We will certify this phenomenon in Theorem 1.

In the second step, given the approximate low rank matrix $\widehat{X}$ returned by the first step, we recover the sparse linear model $\boldsymbol{\beta}_*$ by a simple first order method with a homotopy strategy. At iteration t, given the current solution $\boldsymbol{\beta}_t$, we update it by solving the following optimization problem

$$\boldsymbol{\beta}_{t+1} = \min_{\boldsymbol{\beta}\in\mathbb{R}^d} \{ -\frac{1}{n}(\boldsymbol{\beta}-\boldsymbol{\beta}_t)^T \widehat{X}(\mathbf{y} - \widehat{X}^T\boldsymbol{\beta}_t) + \lambda_t|\boldsymbol{\beta}|_1 + \frac{\gamma}{2}|\boldsymbol{\beta}-\boldsymbol{\beta}_t|_2^2 \} \quad (5)$$

where γ is a constant whose value will be discussed later, and $\lambda_t = \lambda_0\eta^t$ is an adaptive regularization parameter that usually declines over the iterations, where $\eta \in (0,1)$ controls the shrinkage speed of λ_t.

We will show that with appropriate choice of the parameters, we can accurately recover the true sparse solution $\boldsymbol{\beta}_*$. And furthermore, the sparsity of the intermediate solutions is well bounded, which leads to an additional improvement on the computation besides the low rank matrix $\widehat{X}$.

Main Results

In this subsection, we present the theoretical results about the convergence rate and the sparsity bound for our method. For the space limitation, we postpone the detailed proofs to the supplementary materials.

Firstly, we need the following theorem from (Halko, Martinsson, and Tropp 2011) to bound the difference between the data matrix X and its approximator $\widehat{X}$.

Theorem 1 *(Corollary 10.9 (Halko, Martinsson, and Tropp 2011)) Assume $p = k - k_0 \geq 4$, then it holds with failure probability no more than $3e^{-p}$ that*

$$||X - \widehat{X}|| \leq 17\sqrt{1 + \frac{k_0}{p+1}}\sigma_{k_0+1} + \frac{8\sqrt{k}}{p+1}\Big(\sum_{j>k_0}\sigma_j^2\Big)^{1/2},$$

where $||\cdot||$ stands for the spectral norm of matrix, $\sigma_1 \geq \sigma_2 \geq \ldots$ are the singular values of matrix X.

The theorem above shows that if X is low-rank or approximate low-rank that is most of the σ_is are 0 or close to 0, $\widehat{X}$ would be a good approximator for X even we choose a small integer k.

For the sake of convenience in the theoretical analysis below, we denote δ as $\delta = ||\widehat{X} - X||/\sqrt{n}$. From Theorem 1, we know, in the cases where the data matrix X has a low-rank or approximate row-rank structure, which is very common in real applications, δ would always be a small number, i.e., $\delta = o(\frac{1}{\sqrt{n}})$.

The following theorem demonstrates the sparsity of $\boldsymbol{\beta}_t$ over the iterations.

Theorem 2 *Assume $|\mathcal{S}(\boldsymbol{\beta}_t) \setminus \mathcal{S}(\boldsymbol{\beta}_*)| \leq s$, and $\lambda_t \geq \frac{2}{\sqrt{s}}\max\big((\gamma_u+\delta)(\varepsilon+\delta||\boldsymbol{\beta}_*||_2), \Lambda||\boldsymbol{\beta}_* - \boldsymbol{\beta}_t||_2\big)$, where $\Lambda = \big[(2\gamma_u+\delta)\delta + \rho\gamma_u^2 + \max(|\gamma-\gamma_u^2|, |\gamma-\gamma_\ell^2|)\big]$. Then we can have*

$$|\mathcal{S}(\boldsymbol{\beta}_{t+1}) \setminus \mathcal{S}(\boldsymbol{\beta}_*)| \leq s$$

In Theorem 2, the assumption $|\mathcal{S}(\boldsymbol{\beta}_t) \setminus \mathcal{S}(\boldsymbol{\beta}_*)| \leq s$ when $t = 0$ can be satisfied by initializing $\boldsymbol{\beta}_0$ as $\mathbf{0}$. Then by induction and choosing appropriate λ_t over the iterations, we can have $|\mathcal{S}(\boldsymbol{\beta}_{t+1}) \setminus \mathcal{S}(\boldsymbol{\beta}_*)| \leq s$ for all $t \in \mathbb{N}$. Thus the number of non-zero elements in the intermediate solution $\boldsymbol{\beta}_t$ is no more than two times of that in $\boldsymbol{\beta}_*$ and the sparsity of $\boldsymbol{\beta}_t$ is consequentially well-bounded.

At last, we present the convergence rate of our method in the theorem below.

Theorem 3 *Suppose $||\boldsymbol{\beta}_t - \boldsymbol{\beta}_*||_2 \leq \max(e_0, \theta_t)$ and $\eta = \frac{4\sqrt{s}}{\gamma}\Lambda < 1$, we let*

$$\lambda_t = 2\max\big((\gamma_u+\delta)(\varepsilon+\delta||\boldsymbol{\beta}_*||_2), \Lambda\max(e_0,\theta_t)\big),$$

where $e_0 = 4\sqrt{s}(\gamma_u+\delta)(\varepsilon+\delta||\boldsymbol{\beta}_*||_2)/\gamma$ and Λ is inherited from the theorem above. Then we have

$$||\boldsymbol{\beta}_{t+1} - \boldsymbol{\beta}_*||_2 \leq \max\big(e_0, \theta_{t+1}\big) \text{ with } \theta_{t+1} = \eta\theta_t.$$

Theorem 3 shows that by appropriately choosing the value of λ_t according to this theorem, we can get $||\boldsymbol{\beta}_t - \boldsymbol{\beta}_*||_2 \leq \max(e_0, \theta_t) = \max(e_0, \eta^t \theta_0)$. Furthermore, from the definition of Λ, it is clear that we can make $0 < \eta < 1$ if appropriate γ is chosen. Thus a global geometric convergence rate is obtained until we get $||\boldsymbol{\beta}_{t+1} - \boldsymbol{\beta}_*||_2 \leq e_0$. From the definition of e_0, we can see that since both δ and ϵ are small numbers, we can recover $\boldsymbol{\beta}_*$ accurately.

At last, we notice the fact that λ_t can be rewritten as $\lambda_t = 2\max\left(\max\left((\gamma_u + \delta)(\varepsilon + \delta||\boldsymbol{\beta}_*||_2), \Lambda e_0\right), \Lambda\theta_t\right)$. Hence we can let $\lambda_t = \max(\lambda_{\min}, \lambda_0 \eta^t)$ with $\lambda_{\min} = 2\max\left((\gamma_u + \delta)(\varepsilon + \delta||\boldsymbol{\beta}_*||_2), \Lambda e_0\right)$, just as we set in the Algorithm 1.

At the end of this section, we would like to conclude some properties of our approach. 1) We only need $\mathcal{O}(dk + nk)$ flops at each iteration to update the $\boldsymbol{\beta}_t$ instead of $\mathcal{O}(dn)$ in traditional methods. It is a significant improvement when both d and n are much larger than k. 2) A global geometric convergence rate is achieved. 3) The sparsity of intermediate solution $\boldsymbol{\beta}_t$ is well bounded, that is the number of the nonzero entries in $\boldsymbol{\beta}_t$ is no more than two times of that in $\boldsymbol{\beta}_*$. 4) Compared to (Xiao and Zhang 2013), we only need to update $\boldsymbol{\beta}_t$ only once a for each λ_t.

Experiment Study

Having analysed the performance of the proposed method in the theoretical perspective, now we turn to the empirical study that how the proposed method behaves.

We follow the work (Xiao and Zhang 2013) and evaluate the performance of our method mainly on two aspects: (i) The speed of the learned vector $\boldsymbol{\beta}_t$ converging to the optimal solution $\boldsymbol{\beta}_*$. (ii) The sparsity of the learned vector $\boldsymbol{\beta}_T$. And two baseline algorithms below will be used in our study.

- ADG ((Nesterov 2013), Algorithm 4.9): Nesterov's accelerated dual gradient method.
- PGH (Xiao and Zhang 2013): a state-of-art algorithm yields a global geometric convergence rate.
- SLviaRP: the proposed method in our study.

Experiments on the Synthesized Datasets

Dataset We generate the data matrix X, the response vector $\mathbf{y}$ and the optimal solution $\boldsymbol{\beta}_*$ in the following way:

1) For the data matrix X, we generate it by adding a Gaussian noise e_X to a low rank matrix X_0, i.e. $X = X_0 + e_X$, where $e_X \in \mathbb{R}^{d\times N}, [e_X]_{ij} \sim N(0, \sigma^2_{e_X})$. For generating the matrix X_0, we first construct a random matrix $U \in \mathbb{R}^{d\times N}$, each U_{ij} is sampled from an uniform distribution $U_{[-1,1]}$. Then, we calculate a QR factorization for U, i.e. $U = QR, Q \in \mathbb{R}^{d\times N}, R \in \mathbb{R}^{N\times N}$. We extract the first k columns from Q to form a new matrix $\hat{Q}$. At last, we get the low rank matrix X_0 by projecting U onto $\hat{Q}$, i.e., $X_0 = \hat{Q}\hat{Q}^T U$.

2) For the sparse vector $\boldsymbol{\beta}_* \in \mathbb{R}^d$, we select s components from its d coordinates randomly and set them to 1. Then we set all the rest components to 0.

3) The response vector is sampled from the prediction model $\mathbf{y} = X^T\boldsymbol{\beta}_* + e_{\mathbf{y}}$, where $e_{\mathbf{y}} \in \mathbb{R}^N$ is a random noise vector sampled from a normal distribution $N(0, \sigma^2_{e_{\mathbf{y}}} I_{N\times N})$.

Parameter setting For the training data, we set $d = 10000, N = 5000, k = 500, s = 25$ and vary the noise level $e_X, e_{\mathbf{y}}$ in $[0.01, 0.02, ..., 0.05]$. In ADG, $\Psi(x) = 0.001||\boldsymbol{\beta}||_1, \gamma_\mu = \gamma_d = 1.5, \mu = 0$ and $L_0 = 0.001$. In PGH, we set $\lambda_{tgt} = 0.001, \epsilon = 10^{-6}, L_{\min} = 10^{-6}, \eta = 0.7, \delta = 0.2$ and $\gamma_{dec} = \gamma_{inc} = 2$. At last, for our method SLviaRP, we let $\gamma = \lambda_0 = 0.3, \eta = 0.94$ and $\lambda_{\min} = 0.002$.

Goal To recover the optimal sparse vector $\boldsymbol{\beta}_*$ from the training data X and $\mathbf{y}$ both accurately and efficiently. And also, the solutions we obtain should be as sparse as possible.

Evalation metrics To evaluate the property of the learning $\boldsymbol{\beta}_T$, we measure the error $||\boldsymbol{\beta}_t - \boldsymbol{\beta}_*||_2$ and its sparsity over the iterations and the running time (second). The sparsity here is computed as $density = \frac{1}{d}\sum_{i=1}^{d} I([\boldsymbol{\beta}_T]_i = 0)$. And we also measure the recovery of the support set of $\boldsymbol{\beta}_T$ by $\mathbf{SSR}(\boldsymbol{\beta}_T) = 2|\mathcal{S}(\boldsymbol{\beta}_T) \cap \mathcal{S}(\boldsymbol{\beta}_*)|/|\mathcal{S}(\boldsymbol{\beta}_T)| + |\mathcal{S}(\boldsymbol{\beta}_*)|$. We run each experiment 100 times, and report the averaged results.

Experimental results Figures 1 and 2 show the performances of different algorithms when the noise level $e_X = e_Y = 0.01$. From the left panels of both Figure 1 and 2, we observe that our method converges to the optimal solution $\boldsymbol{\beta}_*$ as fast as the recent algorithm PGH, thus their convergence rates are comparable. The right panels give us a deep impression that we can improve run-time efficiency significantly due to the lower computing complexity at each iteration. In addition, it shows that ADG is not good at dealing with such under-determined problem, since its error $||\boldsymbol{\beta}_t - \boldsymbol{\beta}_*||_2$ decreases slowly over the iterations. It may result from the fact that its solution path is not sparse (Table 1) over the iterations (Xiao and Zhang 2013).

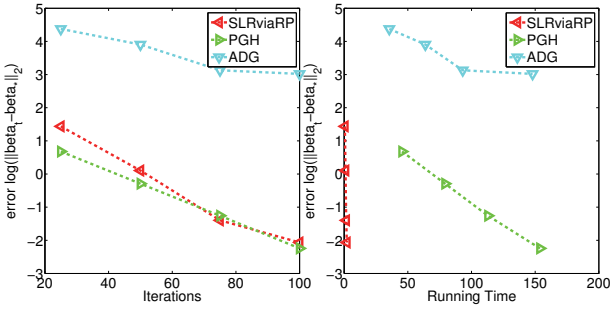

Figure 1: The error $\log(||\boldsymbol{\beta}_t - \boldsymbol{\beta}_*||_2)$ over the iterations (left) and running time (right) with $e_X = e_Y = 0.05$.

Table 1 summarizes the evaluation results for final solutions output from different algorithms after 100 iterations under different noise levels e_X and e_Y. For PGH and SLRviaRP, we observe that they have similar performances at convergence and sparsity preserving, since the values of the metrics, like the residual, the error, density and SSR are all comparable. More importantly, it is confirmed by the time cost of each method that our method is much more efficient than others (roughly 70 times faster in this case).

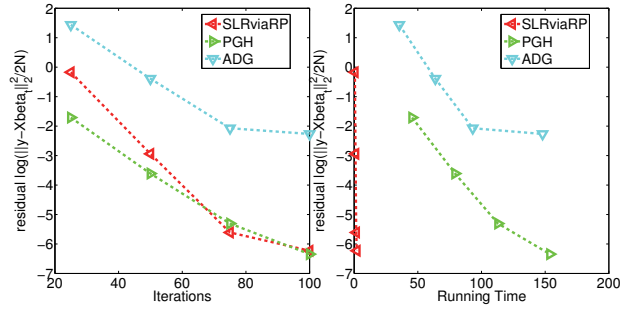

Figure 2: The residual $\log(||\mathbf{y} - X\boldsymbol{\beta}_t||_2^2/2N)$ over the iterations (left) and running time (right) with $e_X = e_Y = 0.05$.

Figure 3 shows the performance of our method under different shrinkage rates $\eta \in [0.91, 0.92, ..., 0.95]$. It reflects the insensitivity of our method to the parameter.

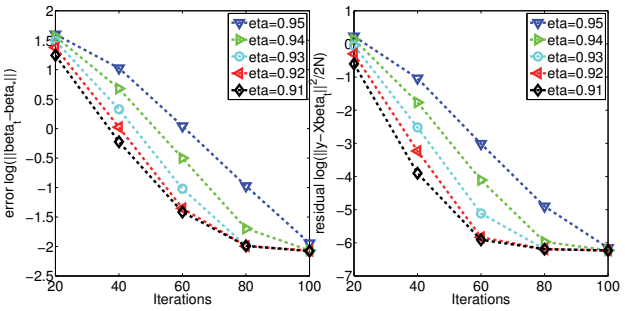

Figure 3: The error $\log(||\boldsymbol{\beta}_t - \boldsymbol{\beta}_*||_2)$ (left) and the residual $\log(||\mathbf{y} - X\boldsymbol{\beta}_t||_2^2/2N)$ (right) of our method over the iterations with different shrinkage rate η and $e_X = e_Y = 0.05$.

Experiments on Real-word Dataset

Experiment design To demonstrate the efficiency of our methods further, we conduct a regression experiment on the well-known dataset MNIST, comprised of the gray images of scanned handwritten digits. It has roughly 60000 training samples and 10000 testing samples. The dimension of each sample is 28×28. We randomly sample 10,000 examples from the training set and get a data matrix $X \in \mathbb{R}^{10,000\times 784}$. At last, we obtain the response vector $\mathbf{y} \in \mathbb{R}^{784}$ by sampling an image from the testing set. Our goal is to find a sparse vector $\boldsymbol{\beta}$ which can approximate $\mathbf{y}$ with $X^T\boldsymbol{\beta}$ accurately.

Parameter setting In ADG, $\Psi(x) = 0.0005||\boldsymbol{\beta}||_1, \gamma_\mu = \gamma_d = 1.2, \mu = 0$ and $L_0 = 0.001$. In PGH, we set $\lambda_{tgt} = 0.005, \epsilon = 0.001, L_{\min} = 0.005, \eta = 0.7, \delta = 0.7$ and $\gamma_{dec} = \gamma_{inc} = 2$. For our method SLviaRP, we let $k = 100, \gamma = 10, \lambda_0 = 0.2, \eta = 0.97$ and $\lambda_{\min} = 0.005$. At last, we run 100 trials and report the averaged performance of each method after 1000 iterations.

Evalation metrics Since the optimal solution $\boldsymbol{\beta}_*$ is unknown, we use the metrics like residual, sparsity and running time to evaluate the performance of our method. And summarize the numerical results in Table 2.

Table 1: Results with $d = 10000, N = 5000, k = 500$.

$e_X = 0.01, e_Y = 0.01$					
Method	Res	Error	density	SSR	time
ADG	0.026	24.566	0.7363	0.007	138.2
PGH	0.001	0.107	0.0025	0.996	134.7
SLRviaRP	**0.001**	**0.111**	**0.0025**	**0.995**	**2.4**
$e_X = 0.02, e_Y = 0.02$					
Method	Res	Error	density	SSR	time
ADG	0.066	22.132	0.7400	0.007	143.7
PGH	0.001	0.108	0.0025	0.997	147.3
SLRviaRP	**0.001**	**0.115**	**0.0025**	**0.994**	**2.3**
$e_X = 0.03, e_Y = 0.03$					
Method	Res	Error	density	SSR	time
ADG	0.091	20.718	0.7524	0.007	148.9
PGH	0.001	0.107	0.0025	0.993	157.5
SLRviaRP	**0.001**	**0.119**	**0.0026**	**0.989**	**1.9**
$e_X = 0.04, e_Y = 0.04$					
Method	Res	Error	density	SSR	time
ADG	0.098	20.13	0.7700	0.006	146.2
PGH	0.001	0.106	0.0025	0.991	149.1
SLRviaRP	**0.001**	**0.124**	**0.0027**	**0.971**	**1.9**
$e_X = 0.05, e_Y = 0.05$					
Method	Res	Error	density	SSR	time
ADG	0.104	20.478	0.7942	0.006	148
PGH	0.002	0.106	0.0026	0.988	153.8
SLRviaRP	**0.002**	**0.127**	**0.0027**	**0.97**	**2.1**

Experimental Result According to Table 2, it is obvious that our method has great superiority in efficiency over the baseline algorithms. And compared to PGH, our method has comparable sparse learning ability. The relatively big residual and the bad sparsity of ADG show that it is not good at dealing with such under-determined problem, which consists with the result in the last experiment.

Table 2: Numerical results on MNIST.

Method	Res	density	time
ADG	0.022	0.9982	48.3
PGH	0.004	0.0046	63.8
SLRviaRP	**0.005**	**0.0036**	**0.7**

Conclusion

In this paper, we propose a novel sparse linear regression method based on random projection. In our method, we reduce the complexity for evaluating the gradient at each iteration from $\mathcal{O}(nd)$ to $\mathcal{O}(nk+dk)$ by using random projection. And then,we adopt a proximal gradient method to solve the sparse linear regression problem with a homotopy strategy. We verify, both theoretically and experimentally, that our method achieves a geometric convergence and can significantly improve the time-efficiency of the existing methods for sparse linear regression. In addition, the sparsity of our intermediate solutions is well-bounded.

Acknowledgments

This work was supported in part by National Basic Research Program of China (973 Program) under Grant 2013CB336500, National Natural Science Foundation of China under Grants (61125203, 61233011) and National Program for Special Support of Top-Notch Young Professionals.

References

Becker, S.; Bobin, J.; and Candès, E. J. 2011. Nesta: a fast and accurate first-order method for sparse recovery. *SIAM Journal on Imaging Sciences* 4(1):1–39.

Bickel, P. J.; Ritov, Y.; and Tsybakov, A. B. 2009. Simultaneous analysis of lasso and dantzig selector. *The Annals of Statistics* 1705–1732.

Bioucas-Dias, J. M., and Figueiredo, M. A. 2007. A new twist: two-step iterative shrinkage/thresholding algorithms for image restoration. *Image Processing, IEEE Transactions on* 16(12):2992–3004.

Candes, E. J., and Tao, T. 2006. Near-optimal signal recovery from random projections: Universal encoding strategies? *Information Theory, IEEE Transactions on* 52(12):5406–5425.

Daubechies, I.; Defrise, M.; and De Mol, C. 2003. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *arXiv preprint math/0307152*.

Drineas, P.; Frieze, A.; Kannan, R.; Vempala, S.; and Vinay, V. 2004. Clustering large graphs via the singular value decomposition. *Machine learning* 56(1-3):9–33.

Drineas, P.; Mahoney, M. W.; Muthukrishnan, S.; and Sarlós, T. 2011. Faster least squares approximation. *Numerische Mathematik* 117(2):219–249.

Drineas, P.; Kannan, R.; and Mahoney, M. W. 2006. Fast monte carlo algorithms for matrices ii: Computing a low-rank approximation to a matrix. *SIAM Journal on Computing* 36(1):158–183.

Drineas, P.; Mahoney, M. W.; and Muthukrishnan, S. 2008. Relative-error cur matrix decompositions. *SIAM Journal on Matrix Analysis and Applications* 30(2):844–881.

Figueiredo, M. A.; Nowak, R. D.; and Wright, S. J. 2007. Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems. *Selected Topics in Signal Processing, IEEE Journal of* 1(4):586–597.

Frieze, A.; Kannan, R.; and Vempala, S. 2004. Fast monte-carlo algorithms for finding low-rank approximations. *Journal of the ACM (JACM)* 51(6):1025–1041.

Hale, E. T.; Yin, W.; and Zhang, Y. 2008. Fixed-point continuation for l1-minimization methodology and convergence. *SIAM Journal on Optimization* 19(3):1107–1130.

Halko, N.; Martinsson, P.-G.; and Tropp, J. A. 2011. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review* 53(2):217–288.

Huang, J.; Ma, S.; and Zhang, C.-H. 2008. Adaptive lasso for sparse high-dimensional regression models. *Statistica Sinica* 18(4):1603.

Kim, S.-J.; Koh, K.; Lustig, M.; Boyd, S.; and Gorinevsky, D. 2007. An interior-point method for large-scale l 1-regularized least squares. *Selected Topics in Signal Processing, IEEE Journal of* 1(4):606–617.

Koltchinskii, V. 2011. *Oracle Inequalities in Empirical Risk Minimization and Sparse Recovery Problems: Ecole dEté de Probabilités de Saint-Flour XXXVIII-2008*, volume 2033. Springer.

Lu, Y.; Dhillon, P.; Foster, D. P.; and Ungar, L. 2013. Faster ridge regression via the subsampled randomized hadamard transform. In *Advances in Neural Information Processing Systems*, 369–377.

Mahoney, M. W., and Drineas, P. 2009. Cur matrix decompositions for improved data analysis. *Proceedings of the National Academy of Sciences* 106(3):697–702.

Mahoney, M. W. 2011. Randomized algorithms for matrices and data. *Foundations and Trends® in Machine Learning* 3(2):123–224.

Nesterov, Y. 2004. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer.

Nesterov, Y. 2013. Gradient methods for minimizing composite functions. *Mathematical Programming* 140(1):125–161.

Sarlos, T. 2006. Improved approximation algorithms for large matrices via random projections. In *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*, 143–152. IEEE.

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

Wen, Z.; Yin, W.; Goldfarb, D.; and Zhang, Y. 2010. A fast algorithm for sparse reconstruction based on shrinkage, subspace optimization, and continuation. *SIAM Journal on Scientific Computing* 32(4):1832–1857.

Wright, S. J.; Nowak, R. D.; and Figueiredo, M. A. 2009. Sparse reconstruction by separable approximation. *Signal Processing, IEEE Transactions on* 57(7):2479–2493.

Xiao, L., and Zhang, T. 2013. A proximal-gradient homotopy method for the sparse least-squares problem. *SIAM Journal on Optimization* 23(2):1062–1091.

Zhang, L.; Mahdavi, M.; Jin, R.; Yang, T.; and Zhu, S. 2013. Recovering the optimal solution by dual random projection. In *Proceedings of the 26th Annual Conference on Learning Theory*, 135–157.

Zhang, L.; Mahdavi, M.; Jin, R.; Yang, T.; and Zhu, S. 2014. Random projections for classification: A recovery approach. *Information Theory, IEEE Transactions on* 60(11):7300–7316.

Zhang, L.; Yang, T.; Jin, R.; and Zhou, Z.-H. 2015. A simple homotopy algorithm for compressive sensing. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, 1116–1124.