Stochastic Second-Order Method for Large-Scale Nonconvex Sparse Learning Models

Hongchang Gao, Heng Huang
Department of Electrical and Computer Engineering, University of Pittsburgh, USA
hongchanggao@gmail.com, heng.huang@pitt.edu

Abstract

Sparse learning models have shown promising performance in the high dimensional machine learning applications. The main challenge of sparse learning models is how to optimize it efficiently. Most existing methods solve this problem by relaxing it as a convex problem, incurring large estimation bias. Thus, the sparse learning model with nonconvex constraint has attracted much attention due to its better performance. But it is difficult to optimize due to the non-convexity. In this paper, we propose a linearly convergent stochastic second-order method to optimize this nonconvex problem for large-scale datasets. The proposed method incorporates the second-order information to improve the convergence speed. Theoretical analysis shows that our proposed method enjoys linear convergence rate and guarantees to converge to the underlying true model parameter. Experimental results have verified the efficiency and correctness of our proposed method.

1 Introduction

Sparse learning models, which play an important role in high dimensional machine learning applications [Lee et al., 2006; Gao et al., 2015; 2017], have attracted much attention in the past decade. Specifically, it assumes that only a few number of model parameters are responsible for the response. Thus, a straightforward way is to enforce sparsity on the model parameter by the $\ell_0$-norm constraint, which restricts the number of non-zero entries in the model parameter. Due to the non-convexity of $\ell_0$-norm, most existing works [Tibshirani, 1996; Van de Geer, 2008; Yuan and Lin, 2006; Friedman et al., 2008; Banerjee et al., 2008] employ the relaxed $\ell_1$-norm regularization to enforce the sparsity of model parameters, since it is easy to solve due to the convexity of $\ell_1$-norm. For instance, the well-known Lasso [Tibshirani, 1996] solves a regression term to fit the data and an $\ell_1$-norm regularization term to pursue a sparse model parameter. However, such a convex relaxation usually degenerates the performance of the model. Thus, it is necessary and challenging to solve the $\ell_0$-norm constraint problem directly.

Formally, in this paper, we focus on the following sparsity-constrained optimization problem:

$$\min_{w} \mathcal{F}(w) = \frac{1}{n} \sum_{i=1}^{n} f_i(w)$$
$$s.t. \|w\|_0 \leq s,$$

where $\mathcal{F}(w)$ is a smooth and convex function, which measures how well the model fits the input space. $\|w\|_0 \leq s$ denotes the number of nonzero entries in $w$ is not more than $s$, controlling the sparsity level of the model parameter. This model is very common in machine learning area. A representative case is the sparse linear regression problem, which is shown as follows:

$$\min_{w} \mathcal{F}(w) = \frac{1}{n} \sum_{i=1}^{n} (y_i - x_i^T w)^2$$
$$s.t. \|w\|_0 \leq s,$$

where $y = [y_1, \cdots, y_n]^T \in \mathbb{R}^n$ is the response vector, $X = \{x_1, \cdots, x_n\} \in \mathbb{R}^{n \times d}$ is the design matrix, and $w \in \mathbb{R}^d$ is the model parameter.

The challenge to solve Eq. (1) is the nonconvex sparse constraint, which makes Eq. (1) as an NP-hard problem. In the past few decades, a large family of algorithms [Mallat and Zhang, 1993; Needell and Tropp, 2009; Tropp and Gilbert, 2007; Zhang, 2011] have been proposed to solve Eq. (1). Among them, there has been much progress towards the gradient-based method, such as gradient hard thresholding pursuit (GraHTP) [Yuan et al., 2014], iterative hard thresholding (IHT) [Blumensath and Davies, 2009], and so on. In particular, the gradient-based method updates the model parameter with the gradient descent method followed by Hard-Thresholding. However, with the development of large-scale data in recent years, these algorithms fail to handle large-scale datasets. The reason is that they need to compute the gradient with respect to all data points at each iteration, making it prohibitive for large-scale datasets. To address this problem, some researchers resort to stochastic algorithms to solve Eq. (1) in recent years, such as SGHT [Nguyen et al., 2014], SVR-GHT [Li et al., 2016], ASBCDHT [Chen and Gu, 2016], and so on. Unlike the full gradient descent method whose complexity of each iteration is $O(nd)$, stochastic methods have only $O(d)$ complexity in each iteration so that they are efficient for large-scale datasets.
Although the gradient-based method has achieved good performance when solving Eq. (1), yet it only considers the first-order information of the objective function, ignoring the second-order curvature information. As a result, its performance is far from satisfactory in some cases. For instance, when the condition number of the objective function in Eq. (1) is extremely large, the first-order method will converge very slowly. If incorporating the second-order curvature information of the objective function into the first-order gradient method, we can obtain a better searching direction in each iteration, making it converge fast. Thus, it is important to employ the second-order method to solve Eq. (1). In optimization community, there has been much progress towards the second-order method, such as [Byrd et al., 2016; Gower et al., 2016; Moritz et al., 2016; Zhao et al., 2017]. But most of them just focus on the convex problem. In [Yuan and Liu, 2014], a Newton greedy pursuit method was proposed to solve the nonconvex Eq. (1). However, it is not suitable for large-scale problems since it uses all data points in each iteration. Recently, some stochastic second-order methods have been proposed, such as [Moritz et al., 2016; Gower et al., 2016]. Although these algorithms enjoy a good converging property, the convergence analysis is based on the strongly convex condition, which is not applicable for the nonconvex Eq. (1). Thus, employing the second-order method to solve Eq. (1) is necessary and challenging.

To incorporate the second-order curvature information and address the scalability problem, we propose a stochastic L-BFGS method to solve the large-scale problem in Eq. (1). In particular, at each iteration, we first randomly sample a mini-batch of data points to evaluate the approximated inverse Hessian matrix and the gradient, updating the model parameter without considering the sparsity constraint and then performing Hard-Thresholding on the updated model parameter to get the sparse one. Additionally, due to the stochastic sampling, the introduced variance will slow down the convergence rate. To address this problem, we incorporate the variance reduction technique as [Moritz et al., 2016]. One of the most important contributions of this paper is that we prove the linear convergence rate of the second-order stochastic L-BFGS for solving the nonconvex Eq. (1). As far as we know, this is the first work which proves stochastic L-BFGS has a linear convergence rate for the nonconvex problem with the sparsity constraint. Furthermore, the output estimator from our proposed method is guaranteed to converge to the unknown true constraint. Furthermore, the output estimator from our proposed method is guaranteed to converge to the unknown true constraint. Furthermore, the output estimator from our proposed method is guaranteed to converge to the unknown true constraint. Furthermore, the output estimator from our proposed method is guaranteed to converge to the unknown true constraint. Furthermore, the output estimator from our proposed method is guaranteed to converge to the unknown true constraint. Furthermore, the output estimator from our proposed method is guaranteed to converge to the unknown true constraint.

Notation Throughout this paper, the matrix is represented by the uppercase letter, the vector is denoted by the bold lowercase letter, and the scalar is represented by the unbold lowercase letter. In particular, X = {x1, . . . , xn} ∈ Rd×d denotes the design matrix, y = [y1, . . . , yn]T ∈ Rn denotes the response vector, and w ∈ Rd denotes the model parameter. For the vector w ∈ Rd, we define ∥w∥1 = ∑i=1n |wi|, ∥w∥2 = ∑i=1n wi2, ∥w∥∞ = max1≤i≤n |wi|. Additionally, supp(w) denotes the index of nonzero elements in w, and supp(w, s) denotes the index of the top s elements of w in regard to magnitude. w(t) denotes the vector in the t-th iteration.

2 Stochastic L-BFGS for Large-Scale Nonconvex Sparse Learning Models

In this section, we will present the detail of our proposed method for large-scale nonconvex sparse learning models. The core idea is employing stochastic L-BFGS to solve Eq. (1). However, the naive stochastic L-BFGS converges slowly due to the introduced variance by random sampling. Inspired by [Moritz et al., 2016], we employ the variance reduction technique [Johnson and Zhang, 2013] to accelerate it. Meanwhile, unlike the traditional stochastic L-BFGS [Moritz et al., 2016] which is only applicable for strongly convex problems, our method can successfully handle the nonconvex problem with the sparsity constraint. The details of our proposed method are summarized in Algorithm 1.

In Algorithm 1, there are two nested loops. In the outer loop, we calculate the full gradient µ in Line 8 such that we can use it to reduce the variance of the stochastic gradient. In the inner loop, our algorithm combines the variance reduced gradient v(t) and the approximated inverse Hessian matrix H(t) to update the model parameter. More specifically, the gradient

\[ \mathbf{v}(t) = \nabla f_{B}(\mathbf{w}(t)) - \nabla f_{B}(\mathbf{w}) + \mu \]  \hspace{1cm} (3)

is an unbiased estimation to \( \nabla F(\mathbf{w}(t)) = \frac{1}{n} \sum_{i=1}^{n} \nabla f(\mathbf{w}(i)) \) where \( \nabla f_{B}(\mathbf{w}(t)) = \frac{1}{1} \sum_{i \in B} \nabla f(\mathbf{w}(i)) \). After that, we update the model parameter without considering the sparsity constraint as follows:

\[ \mathbf{w}(t+1) = \mathbf{w}(t) - \eta H(t)\mathbf{v}(t), \]  \hspace{1cm} (4)

where \( \eta \) is the step size, and \( \mathbf{w}(t+1) \) is the temporary model parameter. With this updating rule, the second-order curvature is incorporated by \( H(t) \). Thus, it will converge faster than the first order approach. In the following, the Hard-Thresholding is performed on \( \mathbf{w}(t+1) \) to obtain the solution satisfying the sparsity constraint as follows:

\[ \mathbf{w}(t+1) = \mathcal{H}(\mathbf{w}(t+1), s), \]  \hspace{1cm} (5)

where s is the sparsity level in Eq. (1), and the Hard-Thresholding operator \( \mathcal{H}(\cdot, s) \) is defined as follows:

\[ \mathcal{H}(w, s) = \begin{cases} w_i, & i \in \text{supp}(w, s), \\ 0, & \text{otherwise}, \end{cases} \]  \hspace{1cm} (6)

where supp(w, s) denotes the s largest non-zero values of the model parameter w.

In Line 11-19 of Algorithm 1, after every L iterations, we update the approximated inverse Hessian matrix \( H^{(r)} \) by the L-BFGS schema as follows:

\[ H^{(r)} = (I - \rho^{(r)}s^{(r)}y^{(r)}y^{(r)T})H^{(r)}_{r-1}(I - \rho^{(r)}s^{(r)}y^{(r)}y^{(r)T}) + \rho^{(r)}s^{(r)}s^{(r)T}, \]  \hspace{1cm} (7)

where \( r - M + 1 \leq i \leq r, M \) is the memory size, \( \rho^{(r)} = \frac{1}{s^{(r)T}y^{(r)}}, \) and \( H^{(r)}_{r-M} = \frac{s^{(r)T}y^{(r)}}{y^{(r)T}y^{(r)}}I \). Then, we set
\( H^{(r)} = H^{(r)} \). Note that unlike traditional L-BFGS method, we update \( y^{(r)} = \nabla^2 f_{B'}(\theta^{(r)}) s^{(r)} \) since it works better in the stochastic setting [Moritz et al., 2016], where \( \nabla^2 f_{B'}(\theta^{(r)}) = \frac{1}{|B'|} \sum_{i \in B'} \nabla^2 f_i(\theta^{(r)}) \).

**Practical Acceleration** In Algorithm 1, we need to compute the Hessian matrix \( \nabla^2 f_{B'}(\theta^{(r)}) \) and its inverse approximation \( H^{(r)} \). Both of them require \( O(d^2) \) storage, which is prohibitive for high dimensionality problems. Instead of constructing \( H^{(r)} \) explicitly, we employ the two-loop recursion method [Nocedal and Wright, 2006] to directly compute \( H^{(r)} y^{(t)} \) based on the correction pairs \( \{ s^{(i)}, y^{(i)} \}^{r}_{i=r-M+1} \).

For the Hessian matrix, we assume it can be represented as follows:

\[
\nabla^2 f_{B'}(\theta^{(r)}) = \frac{1}{|B'|} \sum_{i \in B'} A_i(\theta^{(r)}) A_i^T(\theta^{(r)}) .
\]

Actually, it is very common in many machine learning problems. For example, \( A_i(\theta^{(r)}) \) in Eq. (2) is \( x_i \) so that \( \nabla^2 f_{B'}(\theta^{(r)}) = \frac{1}{|B'|} \sum_{i \in B'} x_i x_i^T \). Based on this representation, instead of computing \( \nabla^2 f_{B'}(\theta^{(r)}) \) explicitly, we can directly compute \( y^{(r)} \) as follows:

\[
y^{(r)} = \frac{1}{|B'|} \sum_{i \in B'} A_i(\theta^{(r)}) [A_i^T(\theta^{(r)}) s^{(r)}] ,
\]

which will save much storage and computation since no explicit Hessian matrix needs to store.

**Algorithm 1** Stochastic L-BFGS Algorithm for Solving Eq. (1).

**Input:** \( X \in \mathbb{R}^{n \times d}, Y \in \mathbb{R}^n, s > 0 \).

**Output:** \( w \in \mathbb{R}^d \)

1: Initialize \( r = 0, H_0 = I \)
2: for \( k = 0, 1, 2, \ldots \) do
3: \( \bar{w} = \bar{w}^{(k-1)}, \bar{w}^{(0)} = \bar{w}^{(k-1)} \)
4: \( \bar{\mu} = \frac{1}{n} \sum_{i=1}^n \nabla f_i(\bar{w}) \)
5: for \( t = 0, 1, 2, \ldots, m - 1 \) do
6: Randomly sample a subset \( B \) from \( \{1, 2, \ldots, n\} \)
7: Compute gradient \( \nabla v^{(t)} = \nabla f_B(\bar{w}^{(t)}) - \nabla f_B(\bar{w}) + \bar{\mu} \)
8: Update \( \bar{w}^{(t+1)} = \bar{w}^{(t)} - H^{(r)} \nabla v^{(t)} \)
9: Hard-Thresholding \( w^{(t+1)} = H(\bar{w}^{(t+1)}, s) \)
10: if \( \text{mod}(t, L) = 0 \) then
11: \( r = r + 1 \)
12: \( \theta^{(r)} = \frac{1}{L} \sum_{j=-L}^{-1} w^{(j)} \)
13: Randomly sample a subset \( B' \) from \( \{1, 2, \ldots, n\} \)
14: Compute \( \nabla^2 f_{B'}(\theta^{(r)}) \)
15: Compute \( s^{(r)} = \theta^{(r)} - \theta^{(r-1)} \)
16: Compute \( y^{(r)} = \nabla^2 f_{B'}(\theta^{(r)}) s^{(r)} \)
17: Compute \( H^{(r)} \) with Eq. (7)
18: end if
19: end for
20: Set \( w^{(k)} \) as the randomly selected \( w^{(i)} \) from \( \{ w^{(0)}, \ldots, w^{(m-1)} \} \)
21: end for
22: return \( w \)

## 3 Convergence Analysis

In this section, we will present the convergence analysis about Algorithm 1. At first, we will introduce the following assumptions that our analysis depends on.

**Assumption 1.** (Restricted Strong Convexity) Function \( F \) satisfies restricted \( \lambda_2 \)-strong convexity condition at sparse level \( \bar{s} \). Formally, we have

\[
F(w) \geq F(w') + \nabla F(w')^T (w - w') + \frac{\lambda_2}{2} ||w - w'||_2^2 ,
\]

for all \( w, w' \) such that \( ||w - w'||_0 \leq s \) and \( \lambda_2 > 0 \).

**Assumption 2.** (Restricted Strong Smoothness) Function \( f_i \) satisfies restricted \( \bar{L}_2 \)-strong smoothness condition at sparse level \( \bar{s} \). Formally, we have

\[
f_i(w) \leq f_i(w') + \nabla f_i(w')^T (w - w') + \frac{\bar{L}_2}{2} ||w - w'||_2^2 ,
\]

for all \( w, w' \) such that \( ||w - w'||_0 \leq s \) and \( \bar{L}_2 > 0 \).

These two assumptions indicate that \( F(w) \) is strongly convex and \( f_i(w) \) is smooth in the sparse subspace. Additionally, based on these two assumptions, we can define the restricted condition number as \( \kappa_2 = \frac{\bar{L}_2}{\bar{L}} \).

**Assumption 3.** The gradient is bounded as follows:

\[
E(||\nabla f_i(w)||_2^2) \leq G^2.
\]

In the following, we present three lemmas for proving the main theorem.

**Lemma 1.** Suppose Assumption 1 and 2 satisfy with the sparsity level \( \bar{s} = 2s + s^* \). Then, for the sparse vector \( w^* \in \mathbb{R}^d \) such that \( ||w^*||_0 \leq s^* \), and the sparse vector \( w^{(t)} \in \mathbb{R}^d \) such that \( ||w^{(t)}||_0 \leq \bar{s} \), we have

\[
E(||v_{S}^{(t)}||_2^2) \leq 12\bar{L}_2 [F(w^{(t)}) - F(w^*) + F(\bar{w}) - F(w^*)] + \frac{3||\nabla SF(\bar{w})||_2^2}{\lambda_2},
\]

where \( S \supseteq (\text{supp}(w^*) \cup \text{supp}(w^{(t)})) \).

This lemma bounds the variance of the stochastic gradient \( v_{S}^{(t)} \). The proof can be found in Lemma 3.5 [Li et al., 2016]. Thus, we do not include it due to the space limitation.

**Lemma 2.** Given the sparse vector \( w^* \in \mathbb{R}^d \) such that \( ||w^*||_0 \leq s^* \), for \( s > s^* \) and any \( w \in \mathbb{R}^d \), we have

\[
||H(w, s) - w^*||_2^2 \leq (1 + \frac{2\sqrt{s^*}}{\sqrt{s - s^*}})||w, w^*||_2^2 .
\]

This lemma actually presents the projection error bound for the Hard-Thresholding operator. The proof can be referred to Lemma 3.3 of [Li et al., 2016].

**Lemma 3.** If Assumption 1 and 2 hold, the estimation of inverse Hessian matrix \( H^{(r)} \) (for all \( r \geq 1 \)) is bounded by

\[
\gamma I \preceq H^{(r)} \preceq \Gamma I ,
\]

where \( 0 < \gamma \leq \Gamma \).
This is a common condition for stochastic L-BFGS method, and the detailed proof can be found from Lemma 4 in [Moritz et al., 2016].

Based on these assumptions and lemmas, we turn to present the main result of our proposed method.

**Theorem 1.** Suppose Assumption 1 and 2 satisfy with the sparsity level \( s = 2s + s^* \). Denote \( w^* \) as the unknown true model parameter such that \( \|w^*\|_0 \leq s ^* \) and \( \tilde{S} = supp(H(\nabla F(w^*), 2s)) \cup supp(w^*) \). By choosing \( C_2 \leq \frac{2}{\sqrt{s - 1}} \) and \( s \geq 1 + 1/A^2s^* \) where \( A = \frac{C_2 - 3\eta^2(1 + \eta)^2}{\beta_m(1 + \eta)(\beta - 1)} \lambda^2(1 - 6\eta^2) (\beta_m - 1) < 1 \), we guarantee \( \beta_m(1 + \eta)(\beta - 1) \lambda^2(1 - 6\eta^2) (\beta_m - 1) < 1 \), where \( \beta = (1 + \frac{2\sqrt{\eta}}{\sqrt{s - 1}})(1 + \eta) \).

Then, for all \( k \geq 0 \), we have

\[
E[\mathcal{F}(\mathbf{w}(k)) - \mathcal{F}(\mathbf{w}^*)] \leq \theta^k E[\mathcal{F}(\mathbf{w}(0)) - \mathcal{F}(\mathbf{w}^*)] + \frac{2\eta(1 - 6\eta^2 L_s^2)\beta_m(\beta - 1) \lambda^2}{(1 + \eta)(\beta - 1)} E[\mathcal{F}(\mathbf{w}(k)) - \mathcal{F}(\mathbf{w}^*)]
\]

(16)

In the following, we present the detailed proof about Theorem 1.

**Proof.** Due to \( \mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta H^{(t)} v^{(t)} \), conditioning on \( \mathbf{w}^{(t)} \), we have

\[
E[\mathcal{F}(\mathbf{w}^{(t+1)}) - \mathcal{F}(\mathbf{w}^*)]\]

\[
= E[\|\mathbf{w}^{(t+1)} - \mathbf{w}^*\|_2^2]
\]

\[
= E[\|\mathbf{w}^{(t)} - \mathbf{w}^*\|_2^2 + 2\eta E[H^{(t)}] v^{(t)} \cdot \nabla S F(\mathbf{w}^{(t)})]
\]

\[
+ 2\eta E[H^{(t)}] v^{(t)} - \mathbf{w}^{(t)} \cdot \nabla S F(\mathbf{w}^{(t)})]
\]

\[
\leq (1 + \eta E[\|\mathbf{w}^{(t)} - \mathbf{w}^*\|_2^2 + \eta^2 G^2 E[\|\nabla S F(\mathbf{w}^{(t)})\|_2^2]
\]

\[
- 2\eta E[H^{(t)}] v^{(t)} \cdot \nabla S F(\mathbf{w}^{(t)})]
\]

\[
\leq (1 + \eta E[\|\mathbf{w}^{(t)} - \mathbf{w}^*\|_2^2 + 2\eta^2 G^2 E[\|\nabla S F(\mathbf{w}^{(t)})\|_2^2]
\]

\[
- 3\eta^2 G^2 E[\|\nabla S F(\mathbf{w}^{(t)})\|_2^2]
\]

where \( \mathcal{S} = supp(w^*) \cup supp(w^{(t)}) \cup supp(w^{(t+1)}) \).

According to Lemma 2, we have

\[
E[\|\mathbf{w}^{(t+1)} - \mathbf{w}^*\|_2^2]
\]

\[
\leq \alpha(1 + \eta) E[\|\mathbf{w}^{(t)} - \mathbf{w}^*\|_2^2 + 3\alpha^2 \eta^2 ||\nabla S F(\mathbf{w}^*)||_2^2
\]

\[
- 2\eta(1 - 6\eta^2 L_s) E[\|\nabla S F(\mathbf{w}^{(t)}) - \mathbf{w}^*\|_2^2]
\]

\[
+ 12\eta^2 L_s^2 E[\|\nabla S F(\mathbf{w}) - \mathbf{w}^*\|_2^2]
\]

\[
+ \eta^2 G^2
\]

where \( \alpha = \eta(1 + \eta)(\beta - 1) \lambda^2(1 - 6\eta^2) (\beta_m - 1) < 1 \).

Now, we need to verify that

\[
\theta = \frac{6\eta^2 L_s^2}{1 - 6\eta^2 L_s^2} < 1
\]

At first, assume \( \eta \leq \frac{C_2}{\beta_m(1 + \eta)(\beta - 1)} \lambda^2(1 - 6\eta^2) L_s^2 \), then

\[
\frac{6\eta^2 L_s^2}{1 - 6\eta^2 L_s^2} < 1
\]

Furthermore, assume \( s \geq (1 + 1/A^2)s^* \) where \( A = \frac{C_2 - 3\eta^2(1 + \eta)^2}{\beta_m(1 + \eta)(\beta - 1)} \lambda^2(1 - 6\eta^2) (\beta_m - 1) < 1 \).

Thus, we can conclude that \( \eta \leq \frac{C_2}{\beta_m(1 + \eta)(\beta - 1)} \lambda^2(1 - 6\eta^2) L_s^2 \), and \( \theta < 1 \).

Therefore, the above inequality holds true.
we get
\[ \beta_m > \frac{C_2}{C_2 - 3\kappa_s(1 + \eta)(\beta - 1)}. \] (24)
Hence, to guarantee \( \theta < 1 \), we should have
\[ m > \frac{6\kappa_s(1 + \eta)^2}{C_2 - 3\kappa_s(1 + \eta)\eta} \log B, \] (25)
where \( B = \frac{C_2}{C_2 - 3\kappa_s(1 + \eta)(\beta - 1)}. \) In this way, we have \( \theta < 1 \).
By recursively applying Eq. (20), we can get the desired result as follows:
\[
E[\mathbf{F}(\tilde{\mathbf{w}}^{(k)}) - \mathbf{F}(\mathbf{w}^*)] \leq \theta^k E[\mathbf{F}(\tilde{\mathbf{w}}^{(0)}) - \mathbf{F}(\mathbf{w}^*)] + 3\eta\Gamma^2 ||\nabla \mathbf{F}(\mathbf{w}^*)||_2^2 + \Gamma^2 G^2 + \frac{2(1 - 6\eta\Gamma^2 L_2)(1 - \theta)}{\lambda_3} \sqrt{3} ||\nabla \mathbf{F}(\mathbf{w}^*)||_\infty.
\] (26)
which completes the proof.

\[ \square \]

**Remark 1.** Theorem 1 indicates a linear convergence rate. In particular, to get a pre-defined accuracy \( \epsilon > 0 \) with respect to the function value gap, we need \( O(\log(1/\epsilon)) \) outer iterations. Additionally, to have linear convergence rate, \( m \) should be set sufficiently large as in Eq. (25).

In the following, we present the approximation error bound for the estimator obtained from Algorithm 1.

**Corollary 1.** With the same conditions as Theorem 1, for all \( k > 0 \), we have the error bound for the estimator \( \tilde{\mathbf{w}}^{(k)} \) as follows:
\[
E||\tilde{\mathbf{w}}^{(k)} - \mathbf{w}^*||_2 \leq \sqrt{\frac{2\theta^k [\mathbf{F}(\tilde{\mathbf{w}}^{(0)}) - \mathbf{F}(\mathbf{w}^*)]}{\lambda_3}} + \frac{\Gamma^2}{\lambda_3(1 - 6\eta\Gamma^2 L_2)(1 - \theta)} G + \left( \frac{2}{\lambda_3} + \frac{3\eta\Gamma^2}{\lambda_3(1 - 6\eta\Gamma^2 L_2)(1 - \theta)} \right) \sqrt{3} ||\nabla \mathbf{F}(\mathbf{w}^*)||_\infty.
\] (27)

**Proof.** Due to Assumption 1, we have
\[
\mathbf{F}(\mathbf{w}^*) \leq \mathbf{F}(\tilde{\mathbf{w}}^{(k)}) + \nabla \mathbf{F}(\mathbf{w}^*)^T(\mathbf{w}^* - \tilde{\mathbf{w}}^{(k)}) - \frac{\lambda_3}{2} ||\mathbf{w}^* - \tilde{\mathbf{w}}^{(k)}||_2^2.
\] (28)

Furthermore, denote
\[
\Sigma = \theta^k [\mathbf{F}(\tilde{\mathbf{w}}^{(0)}) - \mathbf{F}(\mathbf{w}^*)] + \frac{\Gamma^2 G^2 + 3\eta\Gamma^2 ||\nabla \mathbf{F}(\mathbf{w}^*)||_2^2}{2(1 - 6\eta\Gamma^2 L_2)(1 - \theta)},
\] (29)
then based on Eq. (16), we have
\[
E[\mathbf{F}(\tilde{\mathbf{w}}^{(k)}) - \Sigma] \leq E[\mathbf{F}(\tilde{\mathbf{w}}^{(k)}) + \nabla \mathbf{F}(\mathbf{w}^*)^T(\mathbf{w}^* - \tilde{\mathbf{w}}^{(k)}) - \frac{\lambda_3}{2} ||\mathbf{w}^* - \tilde{\mathbf{w}}^{(k)}||_2^2].
\] (30)
Furthermore, we have
\[
E[\nabla \mathbf{F}(\mathbf{w}^*)^T(\mathbf{w}^* - \tilde{\mathbf{w}}^{(k)})] \leq ||\nabla \mathbf{F}(\mathbf{w}^*)||_2 E[||\mathbf{w}^* - \tilde{\mathbf{w}}^{(k)}||_2]
\] (31)
\[
\leq \sqrt{3} ||\nabla \mathbf{F}(\mathbf{w}^*)||_\infty E[||\mathbf{w}^* - \tilde{\mathbf{w}}^{(k)}||_2] + \Sigma.
\]
Put Eq. (31) into Eq. (30), we have
\[
\frac{\lambda_3}{2} E[||\mathbf{w}^* - \tilde{\mathbf{w}}^{(k)}||_2]^2 \leq \sqrt{3} ||\nabla \mathbf{F}(\mathbf{w}^*)||_\infty E[||\mathbf{w}^* - \tilde{\mathbf{w}}^{(k)}||_2] + \Sigma.
\]

By solving this inequality with respect to \( E[||\mathbf{w}^* - \tilde{\mathbf{w}}^{(k)}||_2] \), we can obtain the desired result.

**Remark 2.** This error bound for the estimator consists of three terms. The first term corresponds to the optimization error, the second term is a constant, and the third term corresponds to the statistical error. After sufficient iterations, the first term will approach to zero. Therefore, our algorithm can always converge to the unknown true parameter \( \mathbf{w}^* \), up to the statistical error and a constant value.

4. **Experiments**

In this section, we will present the performance of our proposed method on both synthetic and real-world datasets.

Throughout the experiments, we compare it with two state-of-the-art stochastic methods. They are SGHT [Nguyen et al., 2014] and SVR-GHT [Li et al., 2016]. Specifically,

- **SGHT** [Nguyen et al., 2014]: It employs stochastic gradient to update the model parameter and then performs Hard-Thresholding on the obtained model parameter.
- **SVR-GHT** [Li et al., 2016]: This method adopts the variance reduced gradient to update model parameter, accelerating the converging speed.

All of these methods belong to the stochastic method so that they are suitable for large-scale problems. Additionally, we set \( L = 10, M = 10, |B| = 10, |B'| = 50 \). The step length of each method is chosen to achieve the best performance.

4.1. **Synthetic Data**

In this experiment, we focus on the sparse linear regression problem, just as shown in Eq. (2). For the synthetic data, each row of the design matrix \( \mathbf{X} \in \mathbb{R}^{n \times d} \) is independently generated from a multivariate Gaussian distribution \( \mathcal{N}(0, \Sigma) \) where \( \Sigma \in \mathbb{R}^{d \times d} \). For the sparse regression coefficient \( \mathbf{w}^* \), the nonzero entries are independently generated from a uniform distribution in \([-1, 1]\). The response vector is constructed by \( \mathbf{y} = \mathbf{Xw}^* + \mathbf{\epsilon} \), where the noise \( \mathbf{\epsilon} \) is generated from a Gaussian distribution \( \mathcal{N}(0, \sigma^2 \mathbf{I}) \), and we set \( \sigma^2 = 0.01 \). With these settings, we construct two synthetic datasets. Toy-1 is with \( n = 20000, d = 2000, s^* = 100, s = 200, \Sigma = \mathbf{I} \). Toy-2 is with \( n = 50000, d = 5000, s^* = 500, s = 1000 \), and diagonal entries of the covariance matrix \( \Sigma \) are set as 1, the other entries are set as 0.1.
In this paper, we propose a stochastic L-BFGS method for solving large-scale nonconvex sparse learning problems. By theoretical analysis, the proposed method has shown a linear convergence rate for this kind of nonconvex problems. Meanwhile, it can guarantee to converge the underlying true model parameters. The extensive experiments have verified the efficiency of the proposed method. Thus, it can be applied to the real-world nonconvex large-scale problems.

Acknowledgements
This work was partially supported by the following grants: NSF-IIS 1302675, NSF-IIS 1344152, NSF-DBI 1356628, NSF-IIS 1619308, NSF-IIS 1633753, NIH R01 AG049371.

1https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets
References


