Engineering Applications of Artificial Intelligence 72 (2018) 124–135

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Random Barzilai–Borwein step size for mini-batch algorithms

Zhuang Yang [a], Cheng Wang [a,*], Zhemin Zhang [a], Jonathan Li [a,b]

[a] Fujian Key Laboratory of Sensing and Computing for Smart Cities, School of Information Science and Engineering, Xiamen University, Xiamen, FJ 361005, China
[b] Department of Geography and Environmental Management, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada

ARTICLE INFO

Keywords:
Stochastic gradient descent
Mini batches
Barzilai–Borwein method
Variance reduction
Convex optimization

ABSTRACT

Mini-batch algorithms, a well-studied, highly popular approach in stochastic optimization methods, are used by practitioners because of their ability to accelerate training through better use of parallel processing power and reduction of stochastic variance. However, mini-batch algorithms often employ either a diminishing step size or a tuning step size by hand, which, in practice, can be time consuming. In this paper, we propose using the improved Barzilai–Borwein (BB) method to automatically compute step sizes for the state of the art mini-batch algorithm (mini-batch semi-stochastic gradient descent (mS2GD) method), which leads to a new algorithm: mS2GD-RBB. We theoretically prove that mS2GD-RBB converges with a linear convergence rate for strongly convex objective functions. To further validate the efficacy and scalability of the improved BB method, we introduce it into another modern mini-batch algorithm, Accelerated Mini-Batch Prox SVRG (Acc-Prox-SVRG) method. In a machine learning context, numerical experiments on three benchmark data sets indicate that the proposed methods outperform some advanced stochastic optimization methods.

1. Introduction

In recent years, the variety and volume of data have grown rapidly. Masses of data have led to increased interest in scalable optimization. One of the most popular and practical methods, dating back to the 1951 seminal work of Robbins and Monro (1951), is the stochastic gradient descent (SGD) method. The SGD method has significant theoretical and empirical advantages in machine learning (Bekkerman et al., 2011; Wang and Han, 2015), as well as in compressed sensing (Carpentier and Munos, 2012; Xu and Minin, 2015), wireless sensor networks (Lavanya and Udgata, 2011; Manjarres et al., 2013), matrix factorization (Gemulla et al., 2011; Luo et al., 2012), and large scale natural language processing (Gimpel et al., 2010).

In machine learning, the traditional SGD method (Zhang, 2004; Shamir and Zhang, 2013) uses a single random example in each iteration. The information obtained by computing the gradient of the empirical risk function associated with this example is used to update the predictor. This leads to a more fine-grained iterative process with low computational cost per iteration, but concurrently introduces considerable stochastic noise. The most obvious manifestation is that the stochastic estimate of the gradient has a non-vanishing variance.

Typically, there have been two approaches to deal with the issue of stochastic noise. (1) Use a decreasing step size (a.k.a learning rate) (Luo, 1991; Solodov, 1998; Zhang, 2004; Nemirovski et al., 2008; Shamir and Zhang, 2013). However, a diminishing step size, often leading to slow convergence near the eventual limit, demands exhaustive experimentation to determine how rapidly the step size must decrease in order to prevent scenarios where the step size becomes too small when the iterations are far from the eventual limit. (2) Use a mini-batching technique (Shalev-Shwartz et al., 2007; Dekel et al., 2012; Cotter et al., 2011; Konečnỳ et al., 2016). However, this technique leads to the unwelcome side-effect of requiring more computations. As these two cases indicate, traditional methods manage to decrease the variance in the stochastic estimate, but that decrease comes at a cost.

Does a mini-batching strategy allow the stochastic optimization methods to use a non-decreasing step size? Actually, mini-batch algorithms often employ either a diminishing step size, or a tuning step size by hand, which, in practice, can be time consuming. For instance, under certain assumptions, some researchers (Duchi and Singer, 2009; Nesterov, 2009; Xiao, 2010; Dekel et al., 2012; Lan, 2012; Byrd et al., 2016) employ a diminishing step size in their proposed mini-batch methods. Berahas et al. (2016) show that the Multi-Batch L-BFGS method, with a constant step size, converges to within a neighborhood of the optimal solution. They also point out that, according to the schedule proposed by Robbins and Monro (1951), by using a step size sequence, $\{\eta_k\}$ to zero, the Multi-Batch L-BFGS method converges to the optimal solution. Li et al. (2014) introduce a technique based on an approximate optimization of a conservatively regularized objective function within

* Corresponding author.
E-mail address: cwang@xmu.edu.cn (C. Wang).

https://doi.org/10.1016/j.engappai.2018.03.017
Received 2 November 2017; Received in revised form 27 January 2018; Accepted 22 March 2018
Available online 12 April 2018

each mini-batch and establish convergence on a decreasing step size for the proposed method. In addition, under certain assumptions, they argue that the step size can develop into a constant step size. Ghadimi et al. (2016) propose a randomized stochastic projected gradient (RSPG) algorithm and analyze its convergence when it employs a non-increasing step size or a non-decreasing step size. Recently, Konečnỳ et al. (2016) proposed the mini-batch semi-stochastic gradient descent (mS2GD) method, which uses a tuning constant step size.

As Roux et al. (2012) indicate, one vital issue regarding stochastic algorithms, that has not been fully addressed in the literature, is how to choose an appropriate step size while running the algorithms. In the classical deterministic method, step size is often obtained by employing line search techniques. However, line search is computationally prohibitive in stochastic gradient methods, because it uses randomly chosen gradient samples and does not allow for a strict sequence of decisions that collapse the search space. Hence, a decreasing or best-tuned fixed step size is often employed in stochastic optimization methods.

Inspired by recent works (Sopyła and Drozda, 2015; Tan et al., 2016; Bordes et al., 2009; Byrd et al., 2016), instead of using a diminishing step size or a tuning step size by hand in the mini-batch algorithms, we equip the state of the art mini-batch algorithm, mS2GD , with the ability to automatically compute step size by using the improved Barzilai–Borwein (BB) method. Sopyła and Drozda (2015) incorporated the BB method into the classic SGD algorithm for training the linear SVM in its primal form. In Sopyła and Drozda (2015), the proposed methods use a random sample to compute step size. However, such methods perform worse than the existing methods. Moreover, in Sopyła and Drozda (2015), theoretical justifications are not established. Tan et al. (2016) proposed using the BB method to compute step size for SGD and its variants: the stochastic variance reduced gradient (SVRG) method, which leads to two algorithms: SGD-BB and SVRG-BB. Each step size in SGD-BB and SVRG-BB is computed using the full gradient of objective functions after a succession of stochastic iterations. SVRG-BB and SGD-BB show promise because, while running, they automatically generate the best step sizes. Indeed, the key idea behind the BB method is motivated by the quasi-Newton property in deterministic optimization. Bordes et al. (2009) and Byrd et al. (2016) used batch samples to approximate quasi-Newton property in stochastic optimization and indicated that their proposed methods show great promise for solving the problems that arise in machine learning.

In our proposed method, to compute step size, the improved BB method uses partial samples, randomly chosen from full samples. Compared with SGD-BB and SVRG-BB, which update each step size after a large number of stochastic steps, our proposed method updates the step size in each stochastic iteration faster and performs well in practice.

The following are some recent works that discuss the choice of step size in stochastic optimization methods: Cotter et al. (2011) specify a novel, accelerated gradient strategy for mini-batch algorithms, where step size, η_k, is scaled polynomially in iteration, k. Schmidt et al. (2015) incorporate the standard backtracking line search into SAG to obtain step size. Mahsereci and Hennig (2015) suggest performing line search to obtain step size for a univariate optimization objective in the Gaussian process.

The primary contributions of this paper are as follows:

- We equip the state of the art mini-batch algorithm, mS2GD, which already has a fast rate, with the ability to automatically compute step size by using the improved BB method, thereby, obtaining a new method: mS2GD-RBB. We prove that our mS2GD-RBB method converges linearly for strongly convex objective functions.
- To further validate the efficacy and scalability of the improved BB method, we introduce it into another modern mini-batch algorithm, the Accelerated Mini-Batch Prox SVRG (Acc-Prox-SVRG) method, which leads to another new mini-batch algorithm: Acc-Prox-SVRG-RBB.
- We conduct experiments, using the proposed methods, to solve logistic regression in three benchmark data sets. Experimental results show that our proposed method obtains a rapidly updated step size sequence in each stochastic stage and achieves better performance than the variants of some advanced SGD and batch algorithms.

The remainder of this paper is organized as follows: Section 2 gives the problem statement and background. Section 3 introduces the details of our proposed method. Section 4 analyzes the convergence of our proposed method. Section 5 presents our numerical results. Section 6 further discusses the efficacy and scalability of the improved BB method. Section 7 concludes the paper.

2. Problem statement and background

Many problems of interest are often formulated as the following optimization problem:

$$\min_{w\in\mathbb{R}^d} F(w) = \frac{1}{n}\sum_{i=1}^{n} f_i(w). \tag{1}$$

Throughout this paper, we focus on such problems where both each f_i and $F(w)$ have Lipschitz continuous derivatives, and, also, are strongly convex. The canonical example is least squares, and in that case, $f_i(w) = \frac{1}{2}(x_i^T w - y_i)^2 + \frac{\lambda}{2}\|w\|_2^2$, where λ is a regularization parameter. Another widespread example is logistic regression, described by the choice $f_i(w) = \log(1+\exp[-y_i x_i^T w]) + \frac{\lambda}{2}\|w\|_2^2$.

To solve the above optimization, the standard mini-batch SGD (Byrd et al., 2012; Dekel et al., 2012) uses the following stochastic update rule: at each iteration k, mini-batch $S_1 \subset \{1,\dots,n\}$ of size b_1 is picked at random and let

$$w_{k+1} = w_k - \eta_k \nabla F_{S_1}(w_k), \tag{2}$$

where $\eta_k > 0$ is the step size in the kth iteration, and

$$\nabla F_{S_1}(w_k) = \frac{1}{b_1}\sum_{i\in S_1} \nabla f_i(w_k), \tag{3}$$

where ∇f_i is the gradient of the ith component function at w_k. If we set mini-batch size $b_1 = 1$, the iteration scheme Eq. (2) degrades into the common SGD method (Bottou, 2010) that employs a single sample per iteration, i.e., $w_{k+1} = w_k - \eta_k \nabla f_i(w_k)$.

3. The algorithm

In this section, we introduce the random BB step size, followed by the introduction of mS2GD, and then describe our mS2GD-RBB method, which equips mS2GD with the random BB step size.

3.1. Random Barzilai–Borwein step size

The BB method, proposed by Barzilai and Borwein in Barzilai and Borwein (1988), has been proven to be an efficient gradient method for solving nonlinear optimization problems. In the BB method, some quasi-Newton properties are used (Zheng and Zheng, 2016). Suppose we want to solve the unconstrained minimization problem

$$\min_{w\in\mathbb{R}^n} f(w), \tag{4}$$

where f is differentiable. A typical iteration of the quasi-Newton methods (Dennis and More, 1974) for solving Eq. (4) is:

$$w_{k+1} = w_k - H_k^{-1}\nabla f(w_k), \tag{5}$$

where H_k is an approximation of the Hessian matrix of f at the current iteration, w_k. The most important feature of H_k is that it must satisfy the so-called secant equation (Biglari and Solimanpur, 2013; Dai, 2013): $H_k s_k = y_k$, where $s_k = w_k - w_{k-1}$ and $y_k = \nabla f(w_k) - \nabla f(w_{k-1})$. Now approximate Hessian matrix H_k by $H_k = (1/\eta_k)I$ with $\eta_k > 0$ and

substitute $H_k = (1/\eta_k)I$ into the secant equation. Alternatively, the step size, η_k, can be found, such that the residual of the secant equation, i.e., $\|(1/\eta_k)s_k - y_k\|_2^2$ is minimized, which leads to the following choice of η_k:

$$\eta_k = \|s_k\|_2^2/(s_k^T y_k). \tag{6}$$

Here η_k denotes the BB step size.

Recently, Bordes et al. (2009) and Byrd et al. (2016) used batch samples to approximate the Hessian matrix, H_k. Therefore, in our work, instead of using a full gradient to compute step size in Eq. (6), we employ random samples to compute an estimated step size in the stochastic stage for the form of Eq. (1) in the following iteration scheme:

$$\eta_k = \frac{1}{b_2} \cdot \frac{(w_k - w_{k-1})^T(w_k - w_{k-1})}{(w_k - w_{k-1})^T(\nabla F_{S_2}(w_k) - \nabla F_{S_2}(w_{k-1}))}, \tag{7}$$

where w_k is obtained in each stochastic iteration, mini-batch $S_2 \subset \{1, \dots, n\}$ of size b_2, $\nabla F_{S_2}(w_k)$ and $\nabla F_{S_2}(w_{k-1})$ are similarly defined as Eq. (3), but they choose b_2 samples updated themselves per iteration, i.e., $\nabla F_{S_2}(w_k) = \frac{1}{b_2}\sum_{i\in S_2}\nabla f_i(w_k)$. In this paper, Eq. (7) is regarded as a random BB (RBB) update step and η_k is called random BB step size.

We determine the difference between the RBB and the BB methods by comparing Eqs. (6) and (7). In computing the random BB step size, the batch samples were employed, which reduces computational cost, but also retains the quasi-Newton property. Also, as indicted Eq. (7), to ensure the convergence of our proposed method, the random BB step size must be divided by batch sample sizes, b_2.

For more detailed information, such as convergence analysis and variants of the BB method, please see Raydan (1993), Molina and Raydan (1996), Birgin et al. (2000), Zhou et al. (2006), Dai and Fletcher (2005), Xie and Chen (2011), Dai (2013), Biglari and Solimanpur (2013), Nosratipour et al. (2017) and the references therein.

3.2. mS2GD

Konečnỳ et al. (2016) proposed the mS2GD method for solving the non-smooth case of Eq. (1). The mS2GD method is shown to reach a pre-defined accuracy with less overall work than a method without mini-batching; mS2GD performs a deterministic step (evaluating the gradient of the objective function at the starting point), followed by a large number of stochastic steps.

In the stochastic procedure of mS2GD, the stochastic estimate of $\nabla F(w_{k-1})$ is of the form

$$G_k = \nabla F_{S_1}(w_{k-1}) - \nabla F_{S_1}(\widetilde{w}) + \nabla F(\widetilde{w}), \tag{8}$$

where $\nabla F_{S_1}(w_{k-1}), \nabla F_{S_1}(\widetilde{w})$ is similarly defined as Eq. (3), $\nabla F(\widetilde{w}) = \frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\widetilde{w})$, and $\widetilde{w}$ is an "old" reference point for which the gradient $\nabla F(\widetilde{w})$ has already been previously evaluated in deterministic step. Then the iteration scheme in Eq. (2) can be implemented as:

$$w_{k+1} = w_k - \eta_k G_{k+1}. \tag{9}$$

3.3. mS2GD with random Barzilai–Borwein step size

In this section, we describe our mS2GD-RBB method (Algorithm 1) for solving Eq. (1). The difference between mS2GD and mS2GD-RBB is that, in the latter, the random BB update step is used to compute the step size, η_k, instead of using a prefixed η in mS2GD. In addition, the difference between SVRG-BB and mS2GD-RBB is that the former evaluates step size, which, after completing m stochastic iterations in the inner loop, employs the full gradient of function $F(w)$ in an outer loop. However, our mS2GD-RBB method immediately updates step size in the inner iteration using batch samples.

Remark. The initial step size, η_0, is taken in mS2GD-RBB. However, numerical results on three benchmark data sets show that mS2GD-RBB is insensitive to the choice of η_0. If we always set $\eta_k = \eta$ instead of using the random BB step size, then mS2GD-RBB is reduced to the original mS2GD method.

Algorithm 1 mS2GD-RBB

Parameters: update frequency m, samples sizes b_1 and b_2, initial point $\widetilde{w}_0$, initial step size η_0.
for $s = 1, 2, \dots,$ **do**
$\widetilde{w} = \widetilde{w}_{s-1}$
$\varphi = \nabla F(\widetilde{w})$
$w_0 = \widetilde{w}$
for $k = 1$ **to** m **do**
Randomly pick subset $S_1 \subset \{1, \dots, n\}$ of size b_1, compute a stochastic estimate of $\nabla F(w_{k-1})$

$$G_k = \nabla F_{S_1}(w_{k-1}) - \nabla F_{S_1}(\widetilde{w}) + \varphi \tag{10}$$

$w_k = w_{k-1} - \eta_{k-1}G_k$
Randomly pick subset $S_2 \subset \{1, 2, \dots, n\}$ of size b_2, compute a random BB step size:
$\eta_k = \frac{1}{b_2} \cdot \|w_k - w_{k-1}\|_2^2/((w_k - w_{k-1})^T(\nabla F_{S_2}(w_k) - \nabla F_{S_2}(w_{k-1})))$
end for
$\widetilde{w}_s = w_m$
end for

4. Convergence analysis

In this section, we prove the convergence of mS2GD-RBB (Algorithm 1) for solving Eq. (1) with the strongly convex objective function, $F(w)$. Our analysis is conducted based on the following assumptions and lemmas:

Assumption 1. Each gradient of convex function, $f_i(w)$, in Eq. (1) is differentiable and Lipschitz continuous with positive constant, L, which means that for all w and v in $\mathbb{R}^d$, we have

$$\| \nabla f_i(w) - \nabla f_i(v)\|_2 \le L \| w - v\|_2. \tag{11}$$

Assumption 2. $F(w)$ is μ-strongly convex, i.e. there exists $\mu > 0$ such that for all $w, v \in \mathbb{R}^d$,

$$(\nabla F(w) - \nabla F(v))^T(w - v) \ge \mu \| w - v \|_2^2, \tag{12}$$

or equivalently

$$F(w) \ge F(v) + \nabla F(v)^T(w - v) + \frac{\mu}{2} \| w - v \|_2^2 . \tag{13}$$

Lemma 1. *If $F(w)$: $\mathbb{R}^d \to \mathbb{R}$ is convex and its gradient is Lipschitz continuous, then for all $w, v \in \mathbb{R}^d$*

$$F(w) \ge F(v) + \nabla F(w)^T(w - v) + \frac{1}{2L} \| \nabla F(w) - \nabla F(v) \|_2^2 . \tag{14}$$

Both assumptions and lemma are from Nesterov (2004). Assumptions 1 and 2 are usually satisfied by most objective functions in machine learning models, such as the logistic regression and linear regression with L_2-norm regularization. For more detail information on above assumptions and lemma, such as proof, we refer you to see Nesterov (2004).

We first provide the following lemma, which establishes the boundary of the modified stochastic gradient, G_k.

Lemma 2. *Consider $F(w)$ as defined in Eq. (1). Suppose Assumptions 1 and 2 hold. Let $w_* = \mathrm{argmin}_w F(w)$. If G_k is defined as Eq. (10) in Algorithm 1, then*

$$\mathbb{E} \| G_k \|_2^2 \le \frac{4L}{b_1}\left[F(w_{k-1}) - F(w_*) + F(\widetilde{w}) - F(w_*)\right] + \frac{2}{b_1} \| \nabla F(w_{k-1}) \|_2^2 . \tag{15}$$

Table 1
Details of the data sets in our experiments.

Dataset	Training size	Testing size	Feature	λ
a8a	22,696	9,865	123	10^{-2}
w8a	49,749	14,951	300	10^{-2}
ijcnn1	49,990	91,701	22	10^{-4}

Proof. The proof is available in the Appendix. □

The following theorem establishes the linear convergence of mS2GD-RBB (Algorithm 1):

Theorem 1. *Suppose Assumptions 1, 2 and Lemmas 1, 2 hold. Let* $w_* = \mathrm{argmin}_w F(w)$ *and choose* $S_1, S_2 \subset \{1, \ldots, n\}$ *of size* b_1 *and* b_2 *at random, respectively. Assume that* $b_2 > 4L/\mu b_1$ *and that* m, b_1 *and* b_2 *are chosen so that*

$$\alpha = \frac{\mu b_1 b_2^2}{m(\mu b_1 b_2 - 4L)} + \frac{2L}{\mu b_1 b_2 - 4L} < 1, \tag{16}$$

then mS2GD-RBB has linear convergence in expectation with rate α*:*

$$\mathbb{E}[F(\widetilde{w}_s)] - F(w_*) \le \alpha^s [F(\widetilde{w}_0) - F(w_*)].$$

Proof. The proof is available in the Appendix. □

We have the following remarks regarding the above result:

- For any fixed b_1, by properly adjusting parameters m and b_2 we can force α to be arbitrarily small. Actually, the second term can be made arbitrary small by choosing an appropriate b_2.
- Theorem 1 implies that setting m be of the same order as L/μ, where the ratio L/μ, regarded as a condition number of $F(w)$ and often recorded as κ, is sufficient to have geometric convergence. From (16), we have

  $$\alpha = \frac{b_2}{m(1 - 4L/\mu b_1 b_2)} + \frac{2L}{\mu b_1 b_2 - 4L} < 1. \tag{17}$$

  From (17), we have $m = O(b_2)$. While $b_2 > 4\kappa/b_1$, then we obtain $m = O(\kappa/b_1)$.
- To satisfy $\mathbb{E}[F(\widetilde{w}_s)] - F(w_*) \le \epsilon$, the number of stages, s, must satisfy

  $$s \ge \log \frac{F(\widetilde{w}_0) - F(w_*)}{\epsilon} \Big/ \log\left(\frac{1}{\alpha}\right).$$

  Hence, the complexity of mS2GD-RBB is $O(n + \kappa/b_1)\log(1/\epsilon)$, which matches the results of the modern stochastic gradient methods, such as SAG (Roux et al., 2012), SDCA (Shalev-Shwartz and Zhang, 2013) and SVRG (Johnson and Zhang, 2013).

5. Numerical experiments

In this section, we discuss the numerical experiments that were conducted to illustrate the efficacy of our algorithm: mS2GD-RBB (Algorithm 1). In particular, we applied our mS2GD-RBB method to solve a standard testing problem in machine learning: Logistic Regression (LR) with L_2-norm regularization:

$$\min_{w \in \mathbb{R}^d} \quad F(w) := \frac{1}{n}\sum_{i=1}^{n} \log(1 + \exp(-y_i x_i^T w)) + \frac{\lambda}{2} \| w \|_2^2, \tag{18}$$

where $x_i \in \mathbb{R}^d$ and $y_i \in \{+1, -1\}$ are the feature vector and class label of the ith sample, respectively, and $\lambda > 0$ is a regularization parameter.

We tested mS2GD-RBB with three standard real data sets, which were download from the LIBSVM website.[1] Detailed information of the data sets are given in Table 1.

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

5.1. Comparison with mS2GD

In this section, we compare our mS2GD-RBB method and mS2GD with fixed step size for solving Eq. (18). We used the best-tuned step size for mS2GD. For mS2GD-RBB, as with mS2GD, we used the same batch size, b_1, to update the solution sequence, $\{w_k\}$, on each data set. In addition, we tested all the cases of different mini-batch sizes, b_1, appearing in Konečnỳ et al. (2016) on different data sets. Therefore, the cases of batch size, b_2, for updating the step size sequence, $\{\eta_k\}$, are easily seen.

The comparison results of mS2GD-RBB and mS2GD are shown in Fig. 1 through Fig. 3. In all figures,unless otherwise stated, the x-axis represents the number of effective passes over the data, where each effective pass evaluates n component gradients. Each full gradient computation counts as one effective pass and appears as a small flat segment of length 1 on the curves. For mS2GD-RBB, each iteration access $b_1 + b_2$ data points. For mS2GD, each iteration access b_1 data points. In Figs. 1(a), 1(b), 2(a), 2(b), 3(a), 3(b), the y-axis denotes the sub-optimality, $F(w_k) - F(w_*)$. w_* is obtained by running mS2GD with the best-tuned step size until it converges. In Figs. 1(c), 1(d), 2(c), 2(d), 3(c), 3(d), the y-axis denotes the test error rate. Moreover, the dashed lines correspond to mS2GD with different fixed step sizes. The solid lines stand for mS2GD-RBB with different batch sizes b_1 and b_2. The various values of the parameters are given in the legends of the sub-figures.

The plots in Figs. 1(a), 1(b), 2(a), 2(b), 3(a), 3(b) show that, mS2GD-RBB can always achieve the same level of sub-optimality as mS2GD with best-tuned step size, and even can achieve better performance than mS2GD with best-tuned step size. As seen from Figs. 1(c), 1(d), 2(c), 2(d), 3(c), 3(d), our mS2GD-RBB method achieves the same test error rate as mS2GD with the best-tuned step size. Also, as seen in Figs. 1, 2, and 3, when fixed batch size, b_1, is used in updating solution sequence $\{w_k\}$, it is unnecessary to choose a large batch size, b_2, when updating step size sequence. However, if the batch size, b_2, is too small, the algorithm diverges.

In Figs. 2(a) and 2(b), it seems that mS2GD-RBB cannot achieve better performance than mS2GD when we varied b_1 in the set {4, 16} and b_2 in the set {20, 30, 40, 50}. Actually, we can make mS2GD-RBB achieve better performance than mS2GD with an appropriate b_2 for $b_1 = 4$ or $b_1 = 16$, respectively. However, our motivation is to find an easy approach to compute step size for mini-batch algorithms. It is known that the performance of mini-batch algorithms can vary significantly based on the choice of the step size sequence, but in general, little guidance is provided about good choices.

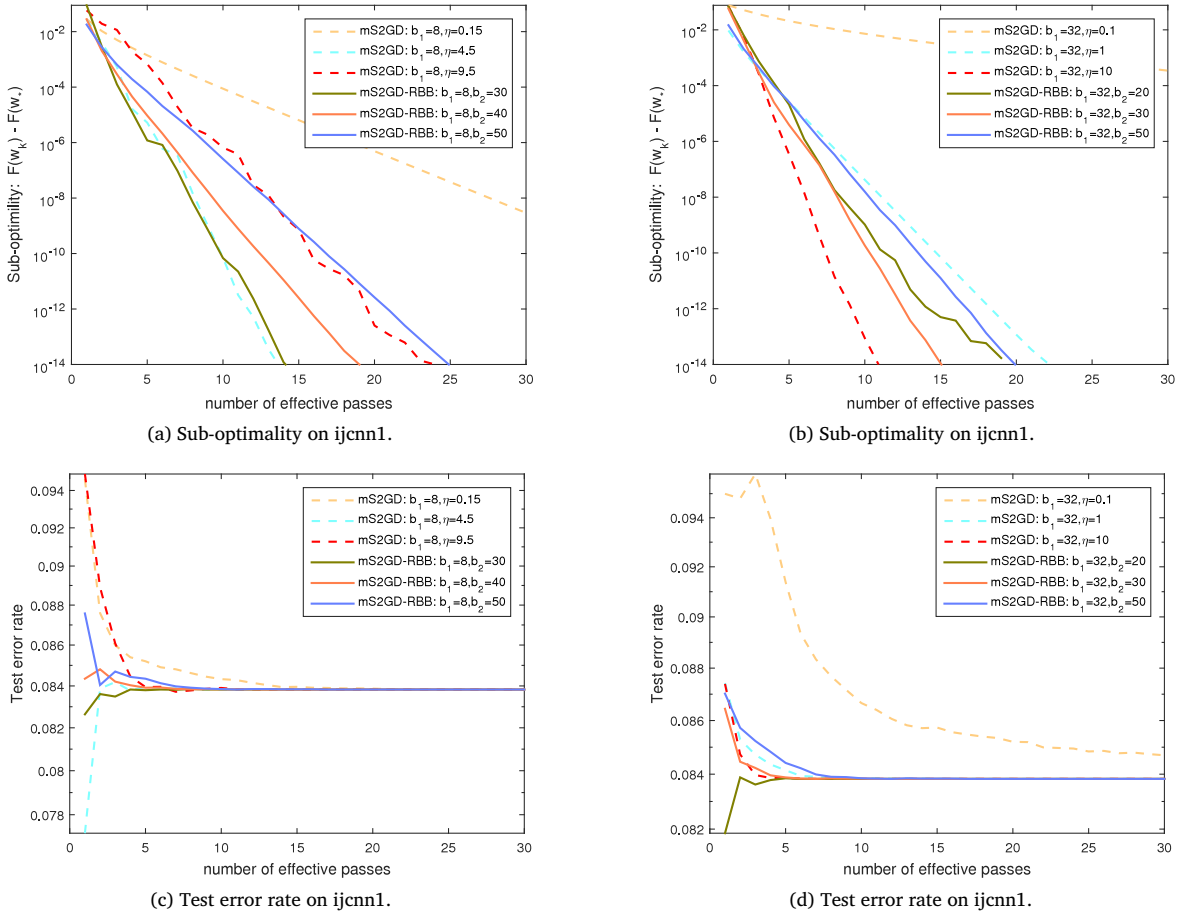
As pointed out in Section 3.3, the mS2GD-RBB method is insensitive to the initial step size, η_0. To show this case, we chose three different initial step sizes, η_0 ($\eta_0 = 0.1, 1, 10$) for mS2GD-RBB on three data sets and plotted in Fig. 4. In addition, we set $b_1 = 16$ and $b_2 = 40$ for all data sets.

Fig. 4 shows that the performance of mS2GD-RBB is not affected by the initial step size η_0.

5.2. Comparison with SVRG-BB

To further show the efficacy of our proposed method with random BB step size in the stochastic procedure, we compare our proposed method with SVRG-BB (Tan et al., 2016), which updates the new step size after each stochastic procedure. As suggested in Tan et al. (2016), for SVRG-BB, we set $m = 2n$ and use three different initial step sizes ($\eta_0 = 0.1, 1, 10$) on ijcnn1, a8a and w8a, respectively.

The comparison results of mS2GD-RBB and SVRG are shown in Fig. 5. In this section, the dotted lines represent SVRG-BB with different initial step sizes. The solid lines stand for mS2GD-RBB with different batch size b_1 and b_2. Note that in Figs. 5(g), 5(h), 5(i), the x-axis denotes the number of step sizes. Figs. 5(a), 5(b), 5(c) show that mS2GD-RBB achieves better sub-optimality than the SVRG-BB method with three different initial step sizes. In addition, we plot the step sizes of SVRG-BB



(a) Sub-optimality on ijcnn1.

(b) Sub-optimality on ijcnn1.

(c) Test error rate on ijcnn1.

(d) Test error rate on ijcnn1.

Fig. 1. Comparison of mS2GD-RBB and mS2GD with fixed step size on ijcnn1. Sub-optimality: $F(w_k) - F(w_*)$ (top) and test error rate (bottom). The dashed lines correspond to mS2GD with different fixed step sizes η given in the legend. The solid lines stand for mS2GD-RBB with different mini-batch sizes b_1 and b_2.

and mS2GD-RBB in Figs. 5(d), 5(e), 5(f), 5(g), 5(h), 5(i) on ijcnn1, a8a and w8a, respectively. For the sake of brevity, we just show one case of step size for mS2GD-RBB with different mini-batch sizes b_1 and b_2 on each data set in Figs. 5(g), 5(h), 5(i). Fig. 5(d), 5(e) and 5(f) show that SVRG-BB converges to a fixed step size after several effective passes. The SVRG-BB method regards the fixed step size as the best step size and uses this step size in subsequent iterations. Compared with the results shown in Figs. 5(d), 5(e) and 5(f), the results in Figs. 5(g), 5(h) and 5(i) indicate that mS2GD-RBB achieves a set of dynamic step sizes by using the random BB update step and such dynamic step sizes are more timely and effective than the step sizes which obtained by SVRG-BB.

5.3. Comparison with variants of the BB algorithm

In this section, we compare our mS2GD-RBB method with two variants of the BB algorithm, proposed by Zhou et al. (2006) and Biglari and Solimanpur (2013). For solving (4), Zhou et al. (2006) proposed an adaptive BB (ABB) method in which the step size is given by

$$\eta_k^{ABB} = \begin{cases} \alpha_k^{BB_2}, & \alpha_k^{BB_2}/\alpha_k^{BB_1} < \varsigma, \\ \alpha_k^{BB_1}, & otherwise \end{cases} \tag{19}$$

where $\alpha_k^{BB_2} = \frac{s_k^T y_k}{\|y_k\|_2^2}$, $\alpha_k^{BB_1} = \frac{\|s_k\|_2^2}{s_k^T y_k}$ and $\varsigma \in (0,1)$. Based on a fourth order conic model and the modified quasi Newton equation (Biglari et al., 2011), Biglari and Solimanpur (2013) obtained some new step sizes for BB-like methods, for instance,

$$\eta_k^{SBB_4} = \frac{s_{k-1}^T \bar{y}_{k-1}}{\bar{y}_{k-1}^T \bar{y}_{k-1}}, \tag{20}$$

where $\bar{y}_{k-1} = y_{k-1} + \frac{4(f(w_{k-1}) - f(w_k)) + 2(\nabla f(w_k) + \nabla f(w_{k-1}))^T s_{k-1}}{s_{k-1}^T y_{k-1}} y_{k-1}$, and the numerical results in Biglari and Solimanpur (2013) show that these BB-like methods are very efficient. For convenience, we denote the gradient methods corresponding to the step sizes (19) and (20) by ABB and SBB, respectively. For ABB, we set $\varsigma = 0.5$. In addition, for mS2GD-RBB, we set $b_1 = 32$, $b_2 = 20$ and $b_1 = 64$, $b_2 = 20$ on each data set.

Fig. 6 shows that our mS2GD-RBB method achieves better performance than variants of the BB algorithm.

5.4. Comparison results with other methods

In this part, we implemented the following algorithms to conduct a numerical comparison:

(1) **B-SGDcon**: Batch stochastic gradient descent method with a constant step size which gave the best performance in hindsight.

(2) **B-SGD +**: Batch stochastic gradient descent method with variable step size $\eta = \eta_0/k$, where k is the number of effective passes, and η_0 is some initial constant step size.

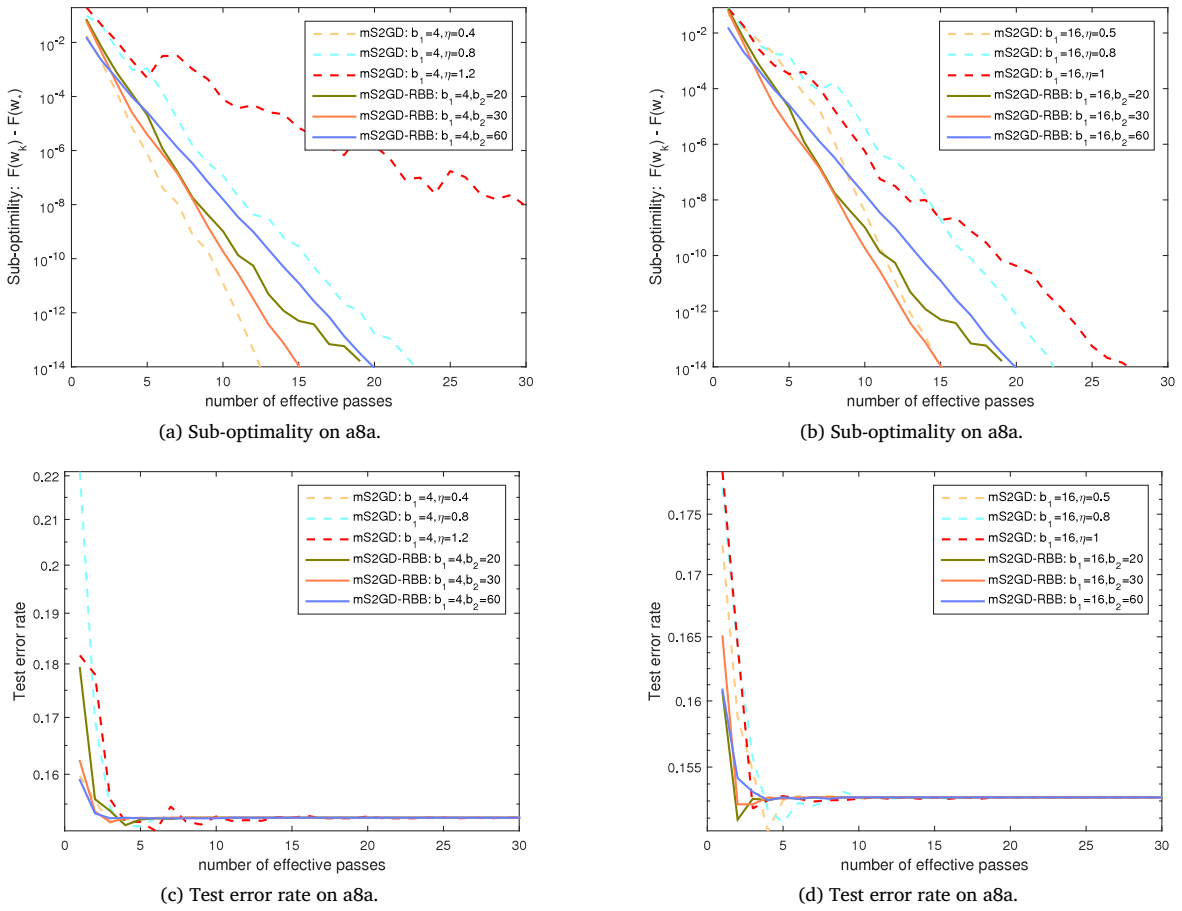
(3) **FISTA**: Fast iterative shrinkage-thresholding algorithm proposed in Beck and Teboulle (2009).

(4) **SAG-LS**: Stochastic average gradient algorithm with line search proposed in Schmidt et al. (2015).

(5) **SAG-BB**: Stochastic average gradient algorithm with BB step size proposed in the supplementary materials of Tan et al. (2016).

(6) **SVRG**: Stochastic variance reduction gradient method proposed in Johnson and Zhang (2013). We used a constant step size.

(7) **SDCA**: Stochastic descent coordinate ascent method proposed in Shalev-Shwartz and Zhang (2013).



(a) Sub-optimality on a8a.

(b) Sub-optimality on a8a.

(c) Test error rate on a8a.

(d) Test error rate on a8a.

Fig. 2. Comparison of mS2GD-RBB and mS2GD with fixed step size on a8a.

(8) **Acc-Prox-SVRG**: Accelerated stochastic proximal gradient descent method proposed in Nitanda (2014). We set $\eta = 1$, $m = \delta b$ ($\delta = 10$), and $\beta_k = \frac{b-2}{b+2}$ ($b = 100$), as suggested in Nitanda (2014).

In all cases, unless otherwise stated, we have used the best constant step sizes in hindsight.

Fig. 7 shows that our mS2GD-RBB method always achieves better performance than the competing methods on the three data sets described above. For mS2GD-RBB, we set $b_1 = 8$ and $b_2 = 40$.

6. Discussion

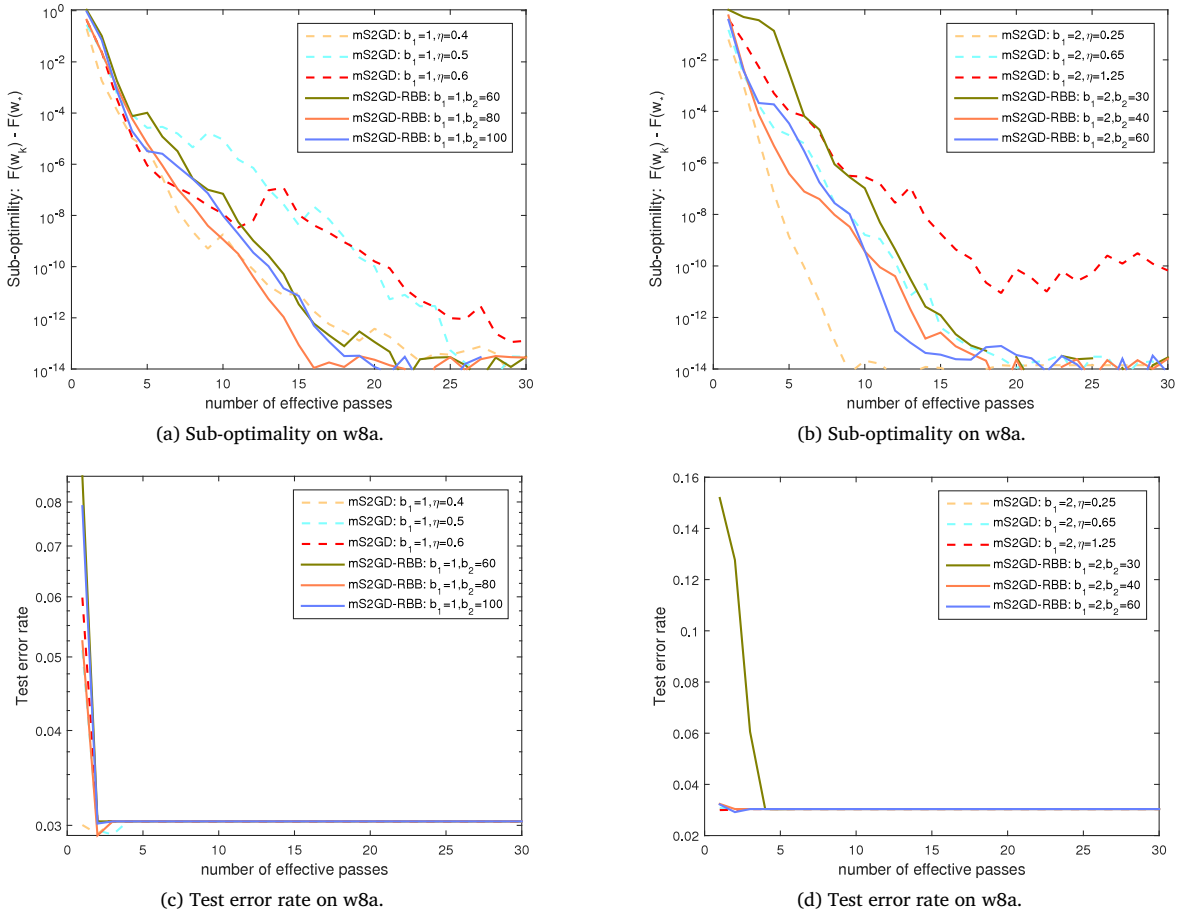
It has been suggested that, to show the efficacy and scalability, we incorporate the random BB step size into other approaches. In this section, we incorporate the random BB step size into another modern mini-batch algorithm, Acc-Prox-SVRG (which was used in the last section), thereby obtaining another new algorithm: Acc-Prox-SVRG-RBB. The Acc-Prox-SVRG method incorporates two acceleration techniques, Nesterov's acceleration method and an variance reduction techniques, in the mini-batch setting for solving (1). With the appropriate mini-batch size, the Acc-Prox-SVRG method can achieve lower overall complexity than both the accelerated proximal gradient descent (APG) and proximal stochastic variance gradient (Prox-SVRG) methods. (For more detail of Acc-Prox-SVRG, such as an analysis of the complexity and convergence, please see Nitanda, 2014.) Here, because of its magnitude, we do not discuss the convergence of Acc-Prox-SVRG-RBB. We show just the results for the comparison between Acc-Prox-SVRG and Acc-Prox-SVRG-RBB to further validate the efficacy and scalability of the random BB step size. Note that the only difference between Acc-Prox-SVRG and Acc-Prox-SVRG-RBB is that, in the latter, the random BB update step is used to compute the step size, η_k, instead of using a prefixed η in Acc-Prox-SVRG.

We ran Acc-Prox-SVRG using the values of η from the range {0.01, 0.05, 0.1, 0.5, 1.0, 5.0, 10.0} and chose the three best results as shown in Figs. 8 and 9. In addition, we varied mini-batch size, b_1, in the set {100, 500}. In Acc-Prox-SVRG-RBB, we varied b_1 in the set {100, 500} and b_2 in the set {200, 300, 400, 500, 800}. The dashed lines stand for Acc-Prox-SVRG with different fixed step sizes. The solid lines correspond to Acc-Prox-SVRG-RBB with different batch sizes b_1 and b_2.

Figs. 8 and 9 show that Acc-Prox-SVRG-RBB achieves the same level of sub-optimality as Acc-Prox-SVRG with the best-tuned step size, and even achieves better performance than Acc-Prox-SVRG. The results further validate the efficacy and scalability of the random BB step size.

7. Conclusion

In this paper, we proposed to use the random BB update step to automatically compute step size for mini-batch algorithms. We first introduced it into the state of the art min-batch algorithm, mS2GD, which already enjoys a fast rate, thereby obtaining a new method, mS2GD-RBB. We proved that mS2GD-RBB converges linearly in expectation for strongly convex objective functions. The results obtained on standard data sets indicate that a rapidly updated step size sequence can be obtained by running mS2GD-RBB. Comparative experiments in logistic regression show that mS2GD-RBB converges more rapidly than some advanced stochastic optimization methods. Finally, the performance of Acc-Prox-SVRG-RBB, which we equipped with another modern mini-batch algorithm (Acc-Prox-SVRG) possessing the ability to automatically compute step size by using the random BB update step, further validates the efficacy and scalability of the random BB update step.



(a) Sub-optimality on w8a.

(b) Sub-optimality on w8a.

(c) Test error rate on w8a.

(d) Test error rate on w8a.

Fig. 3. Comparison of mS2GD-RBB and mS2GD with fixed step size on w8a.

Acknowledgment

This work was supported by grants from the Natural Science Foundation of China under Grant 61371144 and U1605254.

Appendix

A.1. Proof of Lemma 2

Before proving Lemma 2, we first give the following lemma from Johnson and Zhang (2013).

Lemma 3. *Considering* $G_j^1 = \nabla f_j(w_{k-1}) - \nabla f_j(\widetilde{w}) + \nabla F(\widetilde{w})$, *conditioned on* w_{k-1}, *we have the* $E[G_j^1] = \nabla F(w_{k-1})$ *and*

$$\mathbb{E} \parallel G_j^1 \parallel_2^2 \leq 4L[F(w_{k-1}) - F(w_*) + F(\widetilde{w}) - F(w_*)]. \tag{21}$$

Now we prove Lemma 2.

Proof. In view of the definition of G_j^1 in Lemma 3 and G_k in Algorithm 1, then implies that

$$G_k = \frac{1}{b_1} \sum_{j \in S_1} G_j^1. \tag{22}$$

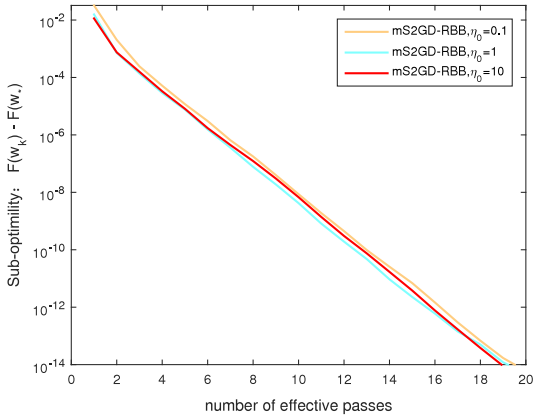
From Eq. (22), we have

$$\parallel G_k \parallel_2^2 = \frac{1}{b_1^2} \parallel \sum_{j \in S_1} G_j^1 \parallel_2^2 . \tag{23}$$

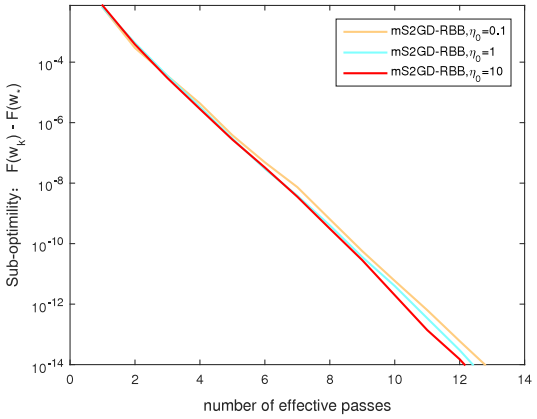
Taking expectation on the above equality, we obtain

$$\begin{aligned}
\mathbb{E}\left[\|G_k\|_2^2\right] &= \frac{1}{b_1^2}\mathbb{E}\left[\| \sum_{j\in S_1} G_j^1\|_2^2\right] \\
&= \frac{1}{b_1^2}\mathbb{E}\left[\| \sum_{j\in S_1'} G_j^1\|_2^2 + 2\langle \sum_{j\in S_1'} G_j^1, G_{j\in S_1 - S_1'}^1\rangle + \|G_{j\in S_1 - S_1'}^1\|_2^2\right] \\
&= \frac{1}{b_1^2}\left[\mathbb{E}\left[\| \sum_{j\in S_1'} G_j^1\|_2^2\right] + 2\langle \nabla F(w_{k-1}), \nabla F(w_{k-1})\rangle \right. \\
&\quad \left. + \mathbb{E}\left[\|G_{j\in S_1 - S_1'}^1\|_2^2\right]\right] \\
&= \cdots = \frac{1}{b_1^2}\left[\sum_{j\in S_1}\mathbb{E}\left[\|G_j^1\|_2^2\right] + 2(b_1 - 1)\|\nabla F(w_{k-1})\|_2^2\right] \\
&\leq \frac{1}{b_1^2}\left[\sum_{j\in S_1}\mathbb{E}\left[\|G_j^1\|_2^2\right] + 2b_1\|\nabla F(w_{k-1})\|_2^2\right] \\
&\leq \frac{4L}{b_1}\left[F(w_{k-1}) - F(w_*) + F(\widetilde{w}) - F(w_*)\right] \\
&\quad + \frac{2}{b_1}\|\nabla F(w_{k-1})\|_2^2
\end{aligned} \tag{24}$$

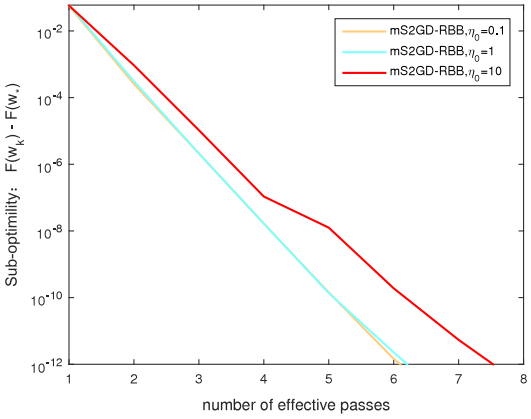
where $S_1' \subset S_1$ and $card(S_1\text{-}S_1') = 1$. (The notation "card" denotes the number of members of the set.) The last equality uses Lemma 3. □



(a) Sub-optimality on ijcnn1.



(b) Sub-optimality on a8a.



(c) Sub-optimality on w8a.

Fig. 4. Different initial step sizes for mS2GD-RBB with $b_1 = 16$ and $b_2 = 40$ on ijcnn1 (left), a8a (middle) and w8a (right).

A.2. Proof of Theorem 1

The idea of proving mS2GD-RBB comes from Johnson and Zhang (2013) and Tan et al. (2016).

Proof. Using the strong convexity of $f_i(w)$, we obtain the following upper boundary for the random BB step size computed by Algorithm 1.

$$\begin{aligned}\eta_k &= \frac{1}{b_2}\cdot\frac{\|w_k - w_{k-1}\|_2^2}{(w_k - w_{k-1})^T(\nabla F_{S_2}(w_k) - \nabla F_{S_2}(w_{k-1}))}\\ &\le \frac{1}{b_2}\cdot\frac{\|w_k - w_{k-1}\|_2^2}{\mu\|w_k - w_{k-1}\|_2^2} = \frac{1}{\mu b_2}.\end{aligned}$$

Note that

$$\begin{aligned}\mathbb{E}\left[\|w_k - w_*\|_2^2\right] &= \mathbb{E}\left[\|w_{k-1} - \eta_{k-1}G_k - w_*\|_2^2\right]\\ &= \|w_{k-1} - w_*\|_2^2 - 2\eta_{k-1}\mathbb{E}[(w_{k-1} - w_*)^T G_k]\\ &\quad + \eta_{k-1}^2\mathbb{E}\left[\|G_k\|_2^2\right]\\ &\le \|w_{k-1} - w_*\|_2^2 - 2\eta_{k-1}(w_{k-1} - w_*)^T\nabla F(w_{k-1})\\ &\quad + \frac{4L\eta_{k-1}^2}{b_1}[F(w_{k-1})\\ &\quad - F(w_*) + F(\widetilde{w}) - F(w_*)] + \frac{2\eta_{k-1}^2}{b_1}\|\nabla F(w_{k-1})\|_2^2\\ &\le \|w_{k-1} - w_*\|_2^2 - 2\eta_{k-1}[F(w_{k-1}) - F(w_*)]\\ &\quad + \frac{4L\eta_{k-1}^2}{b_1}[F(w_{k-1}) - F(w_*)\\ &\quad + F(\widetilde{w}) - F(w_*)] + \frac{2\eta_{k-1}^2}{b_1}\|\nabla F(w_{k-1})\|_2^2\\ &\le \|w_{k-1} - w_*\|_2^2 - 2\eta_{k-1}[F(w_{k-1}) - F(w_*)]\\ &\quad + \frac{4L\eta_{k-1}^2}{b_1}[F(w_{k-1}) - F(w_*)\\ &\quad + F(\widetilde{w}) - F(w_*)] + \frac{4L\eta_{k-1}^2}{b_1}[F(w_{k-1}) - F(w_*)]\\ &= \|w_{k-1} - w_*\|_2^2 - 2\eta_{k-1}\left(1 - \frac{4L\eta_{k-1}}{b_1}\right)\\ &\quad \times [F(w_{k-1}) - F(w_*)] + \frac{4L\eta_{k-1}^2}{b_1}\cdot[F(\widetilde{w}) - F(w_*)],\end{aligned}$$

where in the first inequality we use the boundary of the modified stochastic gradient (Lemma 2) and $\mathbb{E}[G_k] = \nabla F(w_{k-1})$, in the second inequality we use the convexity of $F(w)$, which implies that $-(w_{k-1} - w_*)^T\nabla F(w_{k-1}) \le F(w_*) - F(w_{k-1})$, and in the last inequality we use Lemma 1.

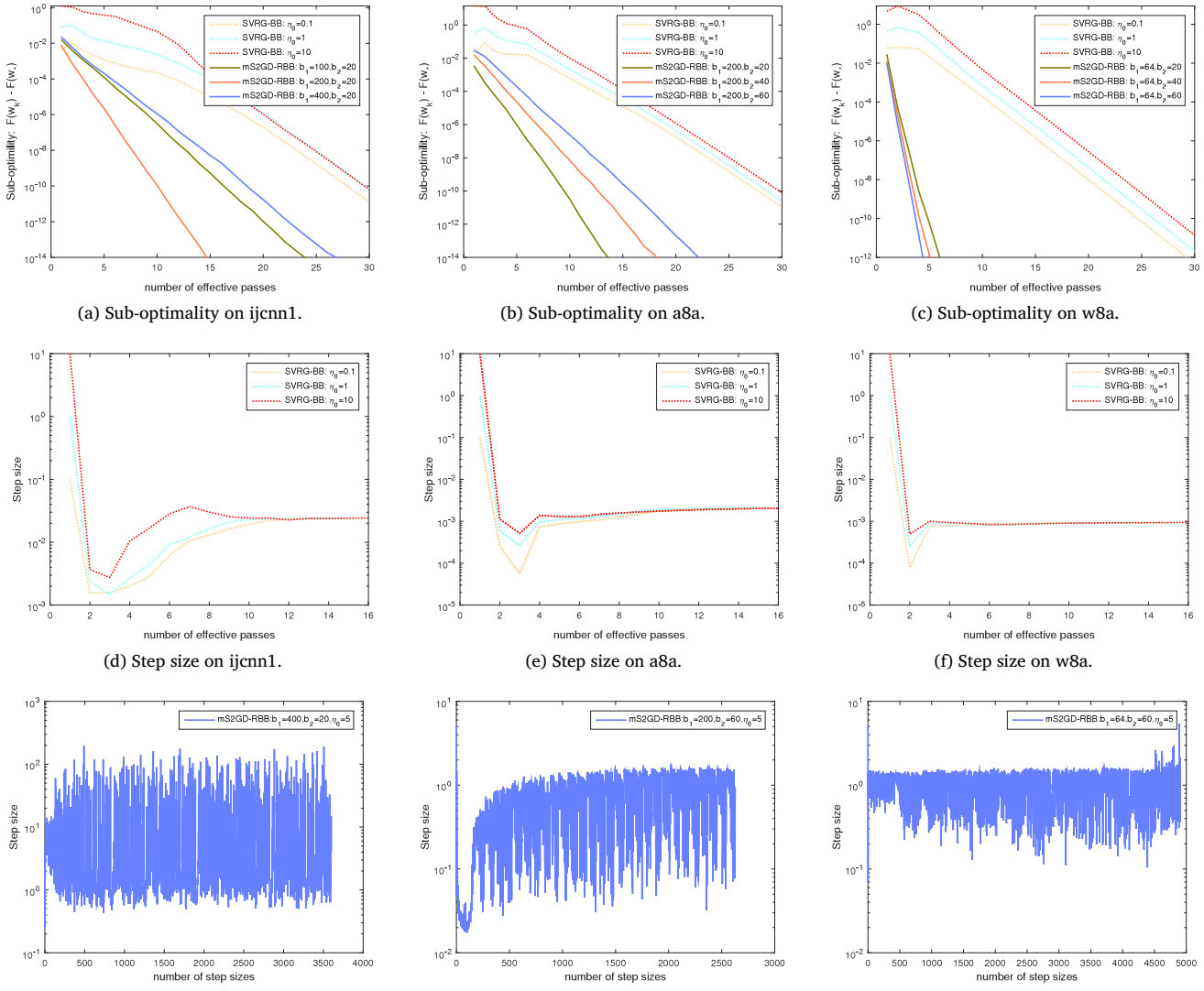
According to the boundary of the random BB step size, we ascertain that

$$\begin{aligned}\mathbb{E}\|w_k - w_*\|_2^2 &\le \|w_{k-1} - w_*\|_2^2\\ &\quad - \frac{2}{\mu b_2}\left(1 - \frac{4L}{\mu b_1 b_2}\right)[F(w_{k-1}) - F(w_*)]\\ &\quad + \frac{4L}{\mu^2 b_1 b_2^2}[F(\widetilde{w}) - F(w_*)].\end{aligned}$$

Now, by the definition of $\widetilde{w}_{s-1}$ in Algorithm 1 we have that

$$\mathbb{E}[F(\widetilde{w}_s)] = \frac{1}{m}\sum_{k=1}^{m}\mathbb{E}[F(w_k)].$$

By summing the previous inequality over $k = 1, \ldots, m$, taking expectation with all the history, and using $\widetilde{w}_s = w_m$ in Algorithm 1,

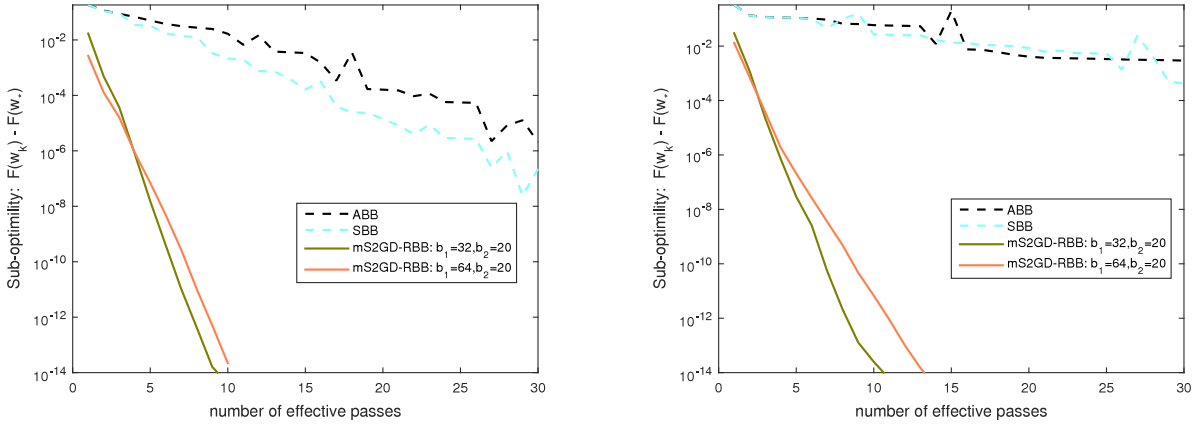


(a) Sub-optimality on ijcnn1. (b) Sub-optimality on a8a. (c) Sub-optimality on w8a.

(d) Step size on ijcnn1. (e) Step size on a8a. (f) Step size on w8a.

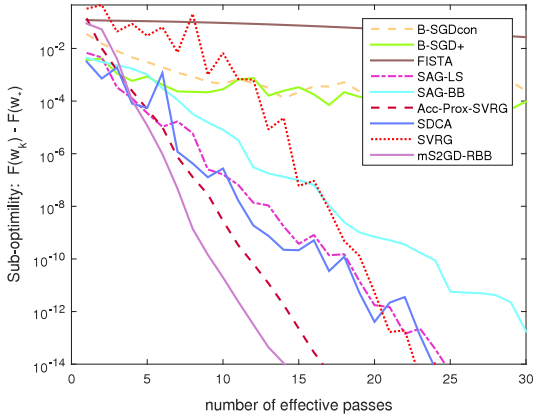
(g) Step size on ijcnn1. (h) Step size on a8a. (i) Step size on w8a.

Fig. 5. Comparison of mS2GD-RBB and SVRG-BB with fixed step size on ijcnn1 (left), a8a (middle) and w8a (right).

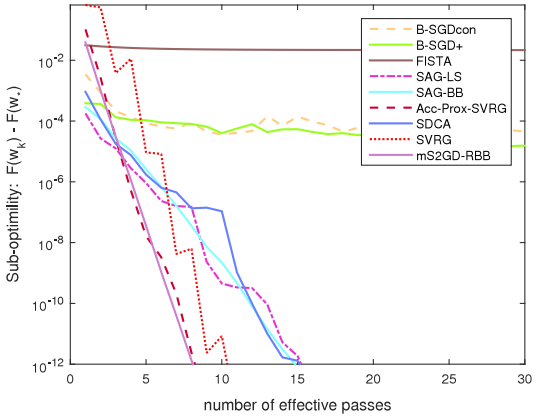


(a) Sub-optimality on a8a. (b) Sub-optimality on ijcnn1.

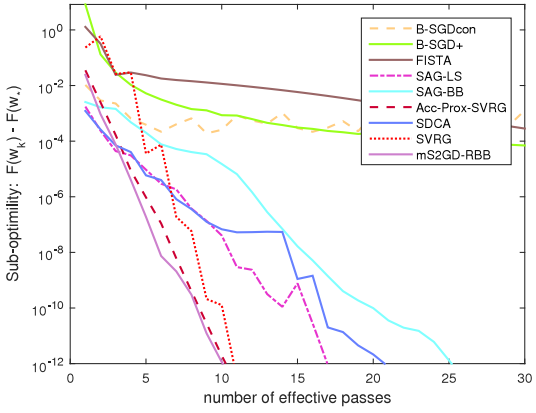
Fig. 6. Comparison of mS2GD-RBB and the variants of BB algorithms on a8a (right) and ijcnn1 (left).



(a) Sub-optimality on ijcnn1.

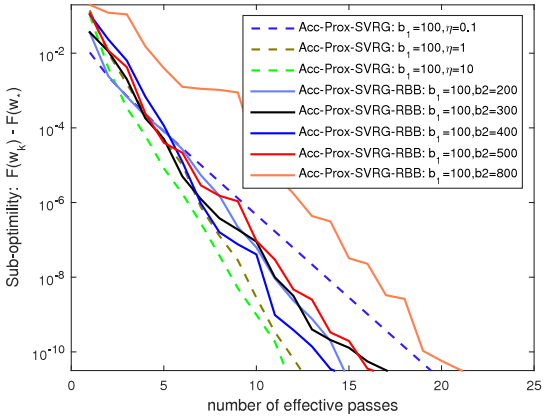


(b) Sub-optimality on w8a.

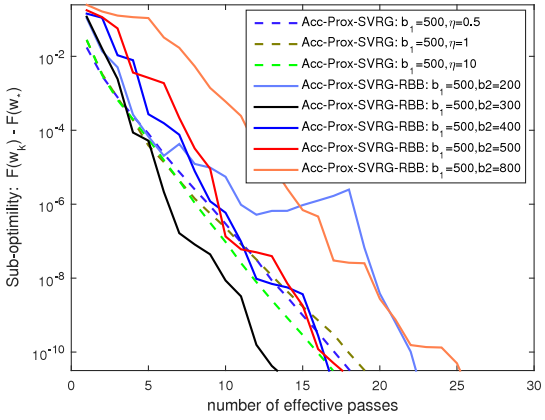


(c) Sub-optimality on a8a.

Fig. 7. Comparison of different methods on ijcnn1 (left), w8a (middle) and a8a (right).



(a) Sub-optimality on ijcnn1.



(b) Sub-optimality on ijcnn1.

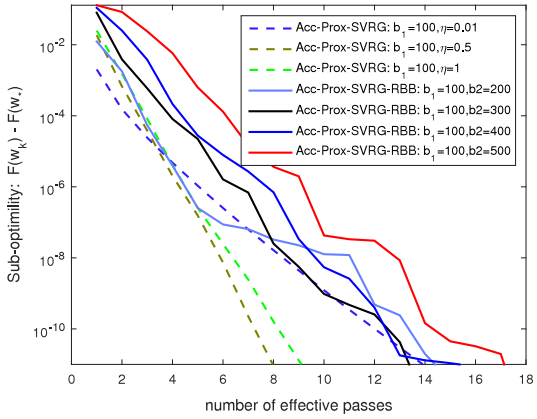
Fig. 8. Comparison of Acc-Prox-SVRG and Acc-Prox-SVRG-RBB on ijcnn1.

we obtain

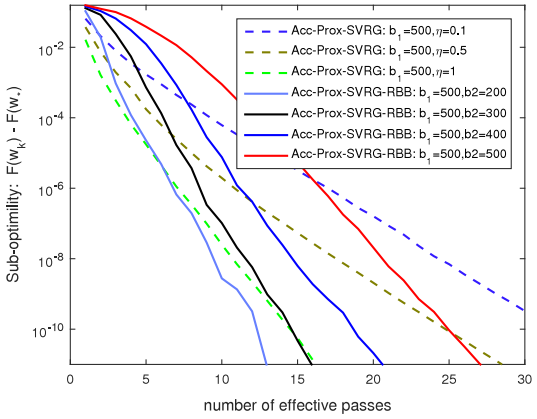
$$\begin{aligned}
&\mathbb{E}\|w_m - w_*\|_2^2 + \frac{2m}{\mu b_2}\left(1 - \frac{4L}{\mu b_1 b_2}\right)\mathbb{E}[F(\widetilde{w}_s) - F(w_*)] \\
&\le \mathbb{E}\|w_0 - w_*\|_2^2 + \frac{4mL}{\mu^2 b_1 b_2^2}\mathbb{E}[F(\widetilde{w}) - F(w_*)] \\
&= \mathbb{E}\|\widetilde{w} - w_*\|_2^2 + \frac{4mL}{\mu^2 b_1 b_2^2}\mathbb{E}[F(\widetilde{w}) - F(w_*)] \\
&\le \frac{2}{\mu}\mathbb{E}[F(\widetilde{w}) - F(w_*)] + \frac{4mL}{\mu^2 b_1 b_2^2}\mathbb{E}[F(\widetilde{w}) - F(w_*)] \\
&= \left(\frac{2}{\mu} + \frac{4mL}{\mu^2 b_1 b_2^2}\right)\mathbb{E}[F(\widetilde{w}) - F(w_*)].
\end{aligned}$$

The second inequality uses the strong convexity property Eq. (13). We thus obtain

$$\begin{aligned}
\mathbb{E}[F(\widetilde{w}_s) - F(w_*)] \le &\left[\frac{\mu b_1 b_2^2}{m(\mu b_1 b_2 - 4L)} + \frac{2L}{\mu b_1 b_2 - 4L}\right] \\
&\times \mathbb{E}[F(\widetilde{w}_{s-1}) - F(w_*)].
\end{aligned}$$



(a) Sub-optimality on a8a.



(b) Sub-optimality on a8a.

Fig. 9. Comparison of Acc-Prox-SVRG and Acc-Prox-SVRG-RBB on a8a.

This implies that $\mathbb{E}[F(\widetilde{w}_s) - F(w_*)] \leq \alpha^s \mathbb{E}[F(\widetilde{w}_0) - F(w_*)]$. The desired bound follows.

References

Barzilai, J., Borwein, J.M., 1988. Two-point step size gradient methods. IMA J. Numer. Anal. 8 (1), 141–148.

Beck, A., Teboulle, M., 2009. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J. Imag. Sci. 2 (1), 183–202.

Bekkerman, Ron, Mikhail, Bilenko, John, Langford, 2011. Scaling up Machine Learning: Parallel and Distributed Approaches. Cambridge University Press.

Berahas, A.S., Nocedal, J., Takac, M., 2016. A Multi-Batch L-BFGS Method for Machine Learning. In: Advance in Neural Information Processing Systems. pp. 1055–1063.

Biglari, F., Hassan, M.A., Leong, W.J., 2011. New quasi-Newton methods via higher order tensor models. J. Comput. Appl. Math. 235 (8), 2412–2422.

Biglari, F., Solimanpur, M., 2013. Scaling on the spectral gradient method. J. Optim. Theory Appl. 158 (2), 626–635.

Birgin, E.G., Martínez, J.M., Raydan, M., 2000. Nonmonotone spectral projected gradient methods on convex sets. SIAM J. Optim. 10 (4), 1196–1211.

Bordes, A., Bottou, L., Gallinari, P., 2009. SGD-QN: Careful quasi-newton stochastic gradient descent. J. Mach. Learn. Res. 10 (Jul), 1737–1754.

Bottou, L., 2010. Large-scale machine learning with stochastic gradient descent. In: Proceedings of COMPSTAT'2010. Springer, pp. 177–186.

Byrd, R.H., Chin, G.M., Nocedal, J., Wu, Y., 2012. Sample size selection in optimization methods for machine learning. Math. Program. 134 (1), 127–155.

Byrd, R.H., Hansen, S.L., Nocedal, J., Singer, Y., 2016. A stochastic quasi-newton method for large-scale optimization. SIAM J. Optim. 26 (2), 1008–1031.

Carpentier, A., Munos, R., 2012. Bandit theory meets compressed sensing for high dimensional stochastic linear bandit. In: International Conference on Artificial Intelligence and Statistics, pp. 190–198.

Cotter, A., Shamir, O., Srebro, N., Sridharan, K., 2011. Better mini-batch algorithms via accelerated gradient methods. In: Advances in Neural Information Processing Systems. pp. 1647–1655.

Dai, Y.H., 2013. A new analysis on the Barzilai-Borwein gradient method. J. Oper. Res. Soc. China 1 (2), 187–198.

Dai, Y.H., Fletcher, R., 2005. On the asymptotic behaviour of some new gradient methods. Math. Program. 103 (3), 541–559.

Dekel, O., Gilad-Bachrach, R., Shamir, O., Xiao, L., 2012. Optimal distributed online prediction using mini-batches. J. Mach. Learn. Res. 13 (Jan), 165–202.

Dennis, J.E., More, J.J., 1974. Quasi-newton methods, motivation and theory. SIAM Rev. 19 (1), 46–89.

Duchi, J.C., Singer, Y., 2009. Efficient online and batch learning using forward backward splitting. J. Mach. Learn. Res. 10, 2899–2934.

Gemulla, R., Nijkamp, E., Haas, P.J., Sismanis, Y., 2011. Large-scale matrix factorization with distributed stochastic gradient descent. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 69–77.

Ghadimi, S., Lan, G., Zhang, H., 2016. Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. Math. Program. 155, 267–305.

Gimpel, K., Das, D., Smith, N.A., 2010. Distributed asynchronous online learning for natural language processing. In: Proceedings of the Fourteenth Conference on Computational Natural Language Learning, pp. 213–222.

Johnson, R., Zhang, T., 2013. Accelerating stochastic gradient descent using predictive variance reduction. In: Advances in Neural Information Processing Systems. pp. 315–323.

Konečnỳ, J., Liu, J., Richtárik, P., Takáč, M., 2016. Mini-batch semi-stochastic gradient descent in the proximal setting. IEEE J. Sel. Top. Sign. Proces. 10 (2), 242–255.

Lan, G., 2012. An optimal method for stochastic composite optimization. Math. Program. 133, 365–397.

Lavanya, D., Udgata, S.K., 2011. Swarm Intelligence Based Localization in Wireless Sensor Networks. Springer Berlin Heidelberg, pp. 317–328.

Li, M., Zhang, T., Chen, Y., Smola, A.J., 2014. Efficient mini-batch training for stochastic optimization. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, pp. 661–670.

Luo, Z., 1991. On the convergence of the LMS algorithm with adaptive learning rate for linear feedforward networks. Neural Comput. 3 (2), 226–245.

Luo, X., Liu, H., Gou, G., Xia, Y., Zhu, Q., 2012. A parallel matrix factorization based recommender by alternating stochastic gradient decent. Eng. Appl. Artif. Intell. 25 (7), 1403–1412.

Mahsereci, M., Hennig, P., 2015. Probabilistic line searches for stochastic optimization. In: Advances in Neural Information Processing Systems. pp. 181–189.

Manjarres, D., Ser, J.D., Gillopez, S., Vecchio, M., Landatorres, I., Salcedosanz, S., Lopez-valcarce, R., 2013. On the design of a novel two-objective harmony search approach for distance- and connectivity-based localization in wireless sensor networks. Eng. Appl. Artif. Intell. 26 (2), 669–676.

Molina, B., Raydan, M., 1996. Preconditioned Barzilai-Borwein method for the numerical solution of partial differential equations. Numer. Algorithms 13 (1), 45–60.

Nemirovski, A., Juditsky, A., Lan, G., Shapiro, A., 2008. Robust stochastic approximation approach to stochastic programming. SIAM J. Optim. 19 (4), 1574–1609.

Nesterov, Y., 2004. Introductory Lectures on Convex Optimization : Basic Course. Kluwer Academic.

Nesterov, Y., 2009. Primal-dual subgradient methods for convex problems. Math. Program. 120 (1), 221–259.

Nitanda, A., 2014. Stochastic proximal gradient descent with acceleration techniques. In: Advances in Neural Information Processing Systems. pp. 1574–1582.

Nosratipour, H., Fard, O.S., Borzabadi, A.H., 2017. An adaptive nonmonotone global Barzilai–Borwein gradient method for unconstrained optimization. Optimization 1–15.

Raydan, M., 1993. On the Barzilai and Borwein choice of steplength for the gradient method. IMA J. Numer. Anal. 13 (3), 321–326.

Robbins, H., Monro, S., 1951. A stochastic approximation method. Ann. Math. Statist. 400–407.

Roux, N.L., Schmidt, M., Bach, F.R., 2012. A stochastic gradient method with an exponential convergence rate for finite training sets. In: Advances in Neural Information Processing Systems. pp. 2663–2671.

Schmidt, M., Babanezhad, R., Ahmed, M.O., Defazio, A., Clifton, A., Sarkar, A., 2015. Non-uniform stochastic average gradient method for training conditional random fields. In: International Conference on Artificial Intelligence and Statistics, pp. 819–828.

Shalev-Shwartz, S., Singer, Y., Srebro, N., 2007. Pegasos: Primal estimated sub-gradient solver for SVM. In: International Conference on Machine Learning, pp. 807–814.

Shalev-Shwartz, S., Zhang, T., 2013. Stochastic dual coordinate ascent methods for regularized loss minimization. J. Mach. Learn. Res. 14 (Feb), 567–599.

Shamir, O., Zhang, T., 2013. Stochastic gradient descent for non-smooth optimization: convergence results and optimal averaging schemes. In: International Conference on Machine Learning, pp. 71–79.

Solodov, M.V., 1998. Incremental gradient algorithms with stepsizes bounded away from zero. Comput. Optim. Appl. 11 (1), 23–35.

Sopyła, K., Drozda, P., 2015. Stochastic gradient descent with Barzilai–Borwein update step for SVM. Inform. Sci. 316, 218–233.

Tan, C., Ma, S., Dai, Y.-H., Qian, Y., 2016. Barzilai-Borwein step size for stochastic gradient descent. In: Advances in Neural Information Processing Systems. pp. 685–693.

Wang, X., Han, M., 2015. Improved extreme learning machine for multivariate time series online sequential prediction. Eng. Appl. Artif. Intell. 40, 28–36.
Xiao, L., 2010. Dual averaging methods for regularized stochastic learning and online optimization. J. Mach. Learn. Res. 11, 2543–2596.
Xie, Z., Chen, S., 2011. SCIHTBB: Sparsity constrained iterative hard thresholding with barzilai-borwein step size. Neurocomputing 74 (17), 3663–3676.
Xu, J., Minin, V.N., 2015. Efficient transition probability computation for continuous-time branching processes via compressed sensing. In: Uncertainty in Artificial Intelligence, Vol. 2015. pp. 952–961.
Zhang, T., 2004. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In: International Conference on Machine Learning. ACM, p. 116.
Zheng, Y., Zheng, B., 2016. A new modified Barzilai–Borwein gradient method for the quadratic minimization problem. J. Optim. Theory Appl. 1–8.
Zhou, B., Gao, L., Dai, Y., 2006. Gradient methods with adaptive step-sizes. Comput. Optim. Appl. 35 (1), 69–86.