Journal of Machine Learning Research 24 (2023) 1-29 Submitted 10/22; Revised 5/23; Published 8/23

Improved Powered Stochastic Optimization Algorithms for Large-Scale Machine Learning

Zhuang Yang ZHUANGYANG@SUDA.EDU.CN
School of Computer Science and Technology
Soochow University
Suzhou, 215006, China

Editor: Prateek Jain

Abstract

Stochastic optimization, especially stochastic gradient descent (SGD), is now the workhorse for the vast majority of problems in machine learning. Various strategies, e.g., control variates, adaptive learning rate, momentum technique, etc., have been developed to improve canonical SGD that is of a low convergence rate and the poor generalization in practice. Most of these strategies improve SGD that can be attributed to control the updating direction (e.g., gradient descent or gradient ascent direction), or manipulate the learning rate. Along these two lines, this work first develops and analyzes a novel type of improved powered stochastic gradient descent algorithms from the perspectives of variance reduction, where the updating direction was determined by the Powerball function. Additionally, to bridge the gap between powered stochastic optimization (PSO) and the learning rate, which is now still an open problem for PSO, we propose an adaptive mechanism of updating the learning rate that resorts the Barzilai-Borwein (BB) like scheme, not only for the proposed algorithm, but also for classical PSO algorithms. The theoretical properties of the resulting algorithms for non-convex optimization problems are technically analyzed. Empirical tests using various benchmark data sets indicate the efficiency and robustness of our proposed algorithms.

Keywords: Powerball function, stochastic optimization, variance reduction, adaptive learning rate, non-convex optimization

1. Introduction

Large numbers of applications in machine learning (Le Thi et al., 2022), computer vision (Xu et al., 2021) and numerical optimization (Fehrman et al., 2020) are often implemented to be a finite-sum structure of the form

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

where n denotes the total number of the instances, w defines the parameter to learn and $f_i(\cdot) : \mathbb{R}^d \to \mathbb{R}$ represents a loss function corresponding to the i-th instance with d dimensions. In this work, we are mainly interested in each $f_i(w)$ and $f(w)$ that are L-smooth but may be non-convex. More concretely and vividly, the cases Problem (1) referred into logistic regression, eigenvector computation, multi-kernel learning, neural networks, to name a few.

Hitherto, modern optimization approaches for solving Problem (1) tend to seek the help from stochastic optimization, where deterministic optimization algorithms meet the challenge, or are even unavailable when the instances n are super huge and the computational cost is significantly higher.

Commonly, vanilla stochastic gradient descent (SGD) requires for only one instance (or mini-batch samples), thus leading to a low computational burden per-iteration as opposed to deterministic gradient algorithms, which call for accessing full data sets at each update step. Nevertheless, the advantage in SGD-like scheme does not come for free. As confirmed by many references (Bottou, 2012; Pillaud-Vivien et al., 2018; Yang et al., 2018a), the high variance in SGD, arising from the sampling strategy, leads to a slow convergence rate and a poor generalization in practice. Even tackling the case that the empirical loss function is strongly convex and smooth, standard SGD merely converges sub-linearly (Rakhlin et al., 2012). Another side effect of the variance is that stochastic optimization algorithms are usually sensitive to several crucial hyperparameters.

Recently, the theoretical and empirical behaviors of SGD have witnessed great modification by variance-reduced stochastic optimization algorithms. Representative instances include the stochastic variance reduced gradient (SVRG) method (Johnson and Zhang, 2013), Finito (Defazio et al., 2014), the stochastic recursive gradient algorithm (SARAH) (Nguyen et al., 2017), the stochastic average gradient (SAG) method (Roux et al., 2012), the stochastic path-integrated differential estimator (SPIDER) (Fang et al., 2018), the accelerated mini-batch Prox-SVRG (Acc-Prox-SVRG) method (Nitanda, 2014), the stochastically controlled stochastic gradient (SCSG) method (Lei et al., 2017), to mention a few. The main difference among these variance-reduced algorithms, or between variance-reduced algorithms and conventional stochastic optimization methods, relies on the manner of how to utilize the historical gradient information.

But beyond above cases, there are some other popular techniques of improving stochastic optimization, covering but not limited to the introduction of curvature information (Mokhtari and Ribeiro, 2020; Bordes et al., 2009), the rule of adaptive learning rates (Tieleman and Hinton, 2017; Kingma and Ba, 2015), the strategy of importance sampling (Zhao and Zhang, 2015; Al-Qaq et al., 1995), and the momentum technique (Loizou and Richtárik, 2020; Wang et al., 2019). More recently, several works start to consider using the Powerball function to modify stochastic optimization through the perspective of alternating the updating directions (Zhou et al., 2020; Yuan et al., 2019). Different from the existing accelerating techniques for stochastic optimization, the idea behind the powered stochastic optimization (PSO) method is inspired by viewing the optimization algorithm as discretizations of ordinary differential equations. As a special case of PSO, SIGNSGD, developed in (Bernstein et al., 2018), not only lowered the per iteration burden of communicating gradients, but also yielded a more rapid empirical convergence rate than standard SGD. There seems to be a direct between gradient descent-based optimization algorithms and PSO algorithms. More concretely, PSO algorithms can be viewed as a kind of a steepest descent approach with respect to the p-norm, where $p = 1 + (1/\gamma)$ ($\gamma \in [0, 1)$).

1.1 Our Contributions

The theoretical and experimental properties of the Powerball function in quasi-Newton, vanilla SGD and SGD with momentum, etc., have been reported in (Yuan et al., 2019; Zhou et al., 2020; Shi et al., 2021; Cui et al., 2022). These studies showed the significant promise of the resulting algorithm in machine learning and engineering problems. Nevertheless, several issues are still needed to be addressed for PSO. First of all, the application of the Powerball function in stochastic optimization algorithms still makes the resulting approach suffer from the high variance, inheriting from the stochastic optimization method, because of adopting a similar algorithm framework to the existing stochastic optimization algorithms. Second, it is still an open problem that how to determine the

learning rate for PSO algorithms. Yuan et al. (2019) provided a hint that the Powerball method can employ a standard backtrack line search to compute the learning rate under deterministic optimization background.

As a consequence, this paper develops and analyzes a novel type of stochastic optimization algorithms from the perspective of utilizing the Powerball function by tackling the problems in existing PSO methods. More specifically, to make our ideas easy to follow, we make a summarization about our key contributions below.

- To mitigate the negative effect of the sampling procedure into powered stochastic optimization, we introduce the stochastic variance reduction gradient estimator into the classical Powerball SGD (pbSGD) method, leading to the new algorithm: namely PB-SVRGE. The convergence behavior for PB-SVRGE is carefully studied in both theory and practice under non-convex optimization background. We confirm a faster convergence rate of the resulting algorithm via comparing with several baseline algorithms.

- To bridge the gap between PSO and the learning rate, we adopt the idea of Barzilai-Borwein (BB) like schemes to obtain an adaptive learning rate sequence for PSO algorithms. In particular, we first study the performance of PB-SVRGE with this kind of adaptive learning rates. To further verify the efficacy of such the adaptive learning rate, we applied it into more general PSO algorithms. Moreover, the convergence properties of adaptive PSO algorithms for non-convex cases were analyzed as well.

- We conduct a set of experiments on benchmark data sets to show the performance of the proposed algorithms and investigate their parameter sensitivity. We also show the superior performance of the proposed algorithms by comparing them with modern stochastic optimization algorithms.

1.2 Notation

Throughout this work, we take $\mathbb{R}^d$ to define the set of d-dimensional real number vectors. For vectors, the mark $w \cdot v$ is used to denote their inner product. Additionally, we use [n] to define the set $\{1,2,\ldots,\mathrm{n}\}$. The symbol $\|\cdot\|$ and $\|\cdot\|_p$ are chosen to represent the Euclidean norm and p-norm on $\mathbb{R}^d$ respectively. Moreover, Denote by $\nabla f(w)$ the gradient of the loss function $f(w)$. We adopt $\mathbb{E}[\cdot]$ to represent the expectation with respect to the underlying probability space. We use $(w)_i$ to represent the i-th element of vector $w \in \mathbb{R}^d$.

1.3 Organize of the Work

The following structures of this work are deployed as follows: Section 2 discusses some related works. Section 3 presents the idea of PSO algorithms with the variance reduction technique and establishes the theoretical results of the proposed algorithm under the non-convex assumption. Section 4 equips the proposed algorithm and more general PSO algorithms with ability to automatically compute the learning rate and rigorously analyzes the convergence behavior of adaptive PSO algorithms. Section 5 investigates the parameter sensitivity of the proposed algorithms and conducts empirical comparisons over some modern optimization methods. Section 6 provides a conclusion for this work.

2. Related Work

More recently, to speed up the convergence of distributed optimization approaches in the non-convex case with only zeroth-order information available, Zhang and Bailey (2022) developed a zeroth-order distributed primal-dual stochastic coordinate method, which was equipped with the Powerball function. Shi et al. (2021) put forward Powerball AdaBelief to enhance the performance of Takagi-Sugeno-Kang (TSK) fuzzy systems, where the Powerball method amplified the small gradients, and AdaBelief dynamically tuned the learning rate. In addition, Zhang et al. (2019) came up with an interior point Powerball approach to accelerate the optimal power flow (OPF) solution process.

As emphasized above, the theoretical and empirical performances of PSO algorithms inherited the drawbacks from stochastic optimization algorithms. It seemed that the accelerating techniques, used in vanilla SGD, could be applied into PSO algorithms to improve their performance. The learning rate, a crucial hyperparameter, has a heated debate on the best method of handling it for stochastic optimization algorithms. Some studies tried to introduce the line search technique into stochastic optimization, see, e.g., (Byrd et al., 2012; Vaswani et al., 2019). In addition, some references showed that the learning rate warmup heuristic achieved significant success in stabilizing training, improving generalization and speeding up convergence for first-order adaptive stochastic optimization approaches like ADAM (Kingma and Ba, 2015), RMSProp (Tieleman and Hinton, 2012), AdaGrad (Duchi et al., 2011), etc. Recently, Liu et al. (2019) identified a problem of adaptive learning rates that its variance was problematically large in the early procedure, and presumed warmup works as a variance-reduced technique. Similar to BB-type methods, the stochastic Polyak step-size, another adaptive learning rate strategy, was also widely adopted for reducing the difficulty in selecting the hyperparameters and accelerating the convergence rate of stochastic optimization algorithms (Loizou et al., 2021; Prazeres and Oberman, 2021).

3. Powered Stochastic Optimization with Variance Reduction

Since this work will delve into the theoretical and numerical characteristics of SVRGE in PSO algorithms, we offer some discussions about the updating scheme of the original SVRG algorithm, originally developed in (Johnson and Zhang, 2013). Comparing with the conventional SGD algorithm that worked with the update scheme:

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

for solving Problem (1), the update scheme of SVRG usually took the form:

$$w_{k+1} = w_k - \eta_k (\nabla f_i(w_k) - \nabla f_i(\tilde{w}) + \nabla f(\tilde{w})), \tag{3}$$

where $\tilde{w}$ was a snapshot point and $\eta_k > 0$ was the learning rate. In general, there were two loops in the SVRG-like approach, where the iterative rule (3) located in the inner loop. The outer loop was used to store the historical information, then transferred it into the inner loop.

The iterative scheme (3), utilizing historical information (i.e., $\nabla f_i(\tilde{w})$ and $\nabla f(\tilde{w})$), made the stochastic optimization algorithms enjoy a lower variance than classical SGD, and resulted in a faster convergence rate and a better generalization in practice.

In contrast, when dealing with Problem (1), the classical iterative scheme of the PSO algorithm was:

$$w_{k+1} = w_k - \eta_k \sigma_\gamma(\nabla f_i(w_k)), \tag{4}$$

where the power coefficient $\gamma \in [0,1)$, $\sigma_\gamma(\nabla f_i(w_k)) = \text{sign}(\nabla f_i(w_k))|\nabla f_i(w_k)|^\gamma$. Here $\text{sign}(x)$ backed to the sign of x, or 0 if $x = 0$. Also, it was noted that the operations $\text{sign}(w)$ and $\sigma_\gamma(w)$ was applied to each element if the parameter $y \in \mathbb{R}^d$ was a vector. For example, $\sigma_\gamma(y) = (\sigma_\gamma((y)_1), \sigma_\gamma((y)_2), \ldots, \sigma_\gamma((y)_d))$.

Comparing the iterative schemes (2) and (4), we observe that the classical PSO algorithm was similar to vanilla SGD, but the Powerball function. Actually, the Powerball approach can be seen as the steepest gradient descent method with respect to the p-norm. Similarly, Newton-like algorithms can also be regarded as the steepest descent algorithm with respect to the ellipsoid norm, but attained a faster convergence rate (Yuan and Sun, 1997).

Naturally, it was asked whether SVRGE created a positive effect on PSO algorithms? In order to respond to this question, we first proposed the PB-SVRGE algorithm (Algorithm 1) and analyzed its theoretical and empirical properties in the upcoming sections.

Algorithm 1 PB-SVRGE

Input: $\widetilde{w}^0 = w_K^0 = w^0$, epoch length K, outer loop size $\mathfrak{S} = T/K$, mini-batch size b, learning rate $\{\eta_j\}_{j=0}^{K-1}$, power coefficient γ.
for $s = 0, \ldots, \mathfrak{S}-1$ **do**
  $w_0^{s+1} = w_K^s$
  $g^{s+1} = \nabla f(\tilde{w}^s)$
  for $k = 0, \ldots, K-1$ **do**
    Randomly choose $S \subseteq [n]$ with $|S| = b$, compute a new stochastic estimate $\nabla f(w_k^s)$;
    $v_k^{s+1} = \nabla f_S(w_k^{s+1}) - \nabla f_S(\tilde{w}^s) + g^{s+1}$
    $w_{k+1}^{s+1} = w_k^{s+1} - \eta_k \sigma(v_k^{s+1})$
  end for
  Set $\widetilde{w}^{s+1} = w_K^{s+1}$
end for

Remark 1: For PB-SVRGE, the following remarks were necessary to be demonstrated:

(i) PR-SVRGE utilized the similar algorithm framework as SVRG-like algorithms. The main difference between these two approaches is that the introduction of the Powerball function in PB-SVRGE. Specifically, we can view the SVRG-like algorithm as an extension of PB-SVRGE when taking $\gamma = 1$. Moreover, we considered PSO algorithms in the mini-batching setting, where $v_k^{s+1} = \frac{1}{b}\sum_{i\in S}\nabla f_i(w_k^{s+1}) - \frac{1}{b}\sum_{i\in S}\nabla f_S(\tilde{w}^s) + \nabla f(\tilde{w}^s)$ ($S \subseteq [n]$ with b batch samples). Here, $\nabla f_S(w_k^{s+1}) = \frac{1}{b}\sum_{i\in S}\nabla f_i(w_k^{s+1})$ and $\nabla f_S(\tilde{w}^s) = \frac{1}{b}\sum_{i\in S}\nabla f_i(\tilde{w}^s)$.

(ii) SIGNSGD, developed in (Bernstein et al., 2018), which used the iterative rule $w_{k+1} = w_k - \eta_k \text{sign}(\nabla f_i(w_k))$ for addressing Problem (1), seemed to have some certain relationships with PB-SVRGE. Surprisingly, there only existed a little work to discuss the performance of SIGNSGD with the variance-reduced technique, see (Jin and Sidford, 2019). When setting $\gamma = 0$, PB-SVRGE can be regarded as a case of SIGNSGD, using the variance-reduced technique. We will consider the numerical behavior of this case in our experiments.

3.1 Convergence Analysis of PB-SVRGE

This work considers the theoretical properties of PB-SVRGE on non-convex optimization problems. To proceed the analysis, we make the following standard smoothness assumptions.

Assumption 1. *Each cost function $f_i(w)$ in Problem* (1) *is differential and has a L-Lipschitz gradient, that is,*

$$\forall w, v \in \mathbb{R}^d, \quad \|\nabla f_i(w) - \nabla f_i(v)\| \leq L \parallel w - v \parallel . \tag{5}$$

The conclusion in Assumption 1 indicates that the average function $f(w)$, presented in Problem (1), has a L-Lipschitz gradient as well.

Clearly, from the smooth property of the cost function, we have the following result:

$$f(w) \leq f(v) + \nabla f(v)^T (w - v) \ + \frac{L}{2} \parallel w - v \parallel^2 . \tag{6}$$

Assumption 2. *The stochastic gradient oracle is an independent and unbiased estimator of the gradient and satisfies*

$$\begin{aligned} &\mathbb{E}_{\xi_i}[\nabla f(w, \xi_i)] = \nabla f(w), \ \ \forall w \in \mathbb{R}^d, \\ &\mathbb{E}_{\xi_i}[\|\nabla f(w, \xi_i) - \nabla f(w)\|^2] \leq \hat{\sigma}^2, \ \ \forall w \in \mathbb{R}^d, \end{aligned} \tag{7}$$

where $\nabla f_i(w) = \nabla f(w, \xi_i)$ and ξ_i denotes a random variable.

Two assumptions mentioned above were very common in the most existing studies for finding a first-order stationary point of non-convex or convex loss functions (Ghadimi and Lan, 2013; Wei et al., 2021; Zhou et al., 2018). Observe that, by (5), the gradient of the loss function does not change arbitrary rapid, and by (7), the variance of the random variable $\mathbb{E}[\|\nabla f_i(w) - \nabla f(w)\|^2]$ has an upper bound. Notice that, when taking b samples in vanilla SGD at each iterative step, we have $\mathbb{E}[\|\nabla f_S(w) - \nabla f(w)\|^2] \leq \frac{\hat{\sigma}^2}{b}$.

The following result elucidates some convergence properties of the PB-SVRGE method (Algorithm 1).

Theorem 1. *Set $w^* = \arg\min_w f(w)$, and choose $S \subseteq [n]$ with $|S| = b$. Let T denote the number of total iterations, then, under Assumption 1 and Assumption 2, for any $T \geq 1$, PB-SVRGE (Algorithm 1) can lead to*

$$\mathbb{E}\left[\frac{1}{T}\sum_{s=0}^{\mathfrak{S}-1}\sum_{k=0}^{K-1}\|\nabla f_S(w_k^{s+1})\|_{1+\gamma}^2\right] \leq \frac{4L\|\mathbf{1}\|_p}{T(1-\theta)}\mathbb{E}[f(\tilde{w}^0) - f(w^*)] + \frac{8\|\mathbf{1}\|_p\hat{\sigma}^2}{b\theta(1-\theta)}. \tag{8}$$

Proof According to the results in the inequality (6) and $w_{k+1}^{s+1} = w_k^{s+1} - \eta_k \sigma_\gamma(v_k^{s+1})$, we easily inferred

$$\begin{aligned} f(w_{k+1}^{s+1}) &\leq f(w_k^{s+1}) + \nabla f(w_k^{s+1}) \cdot (w_{k+1}^{s+1} - w_k^{s+1}) + \frac{L}{2}\|w_{k+1}^{s+1} - w_k^{s+1}\|^2 \\ &= f(w_k^{s+1}) - \eta_k \nabla f(w_k^{s+1}) \cdot \sigma_\gamma(v_k^{s+1}) + \frac{L\eta_k^2}{2}\|\sigma_\gamma(v_k^{s+1})\|^2 \\ &= f(w_k^{s+1}) - \eta_k v_k^{s+1} \cdot \sigma_\gamma(v_k^{s+1}) + \frac{L\eta_k^2}{2}\|\sigma_\gamma(v_k^{s+1})\|^2 + \eta_k(v_k^{s+1} \\ &\quad - \nabla f(w_k^{s+1})) \cdot \sigma_\gamma(v_k^{s+1}). \end{aligned} \tag{9}$$

Setting $\eta_k = \frac{v_k^{s+1}\cdot\sigma_\gamma(v_k^{s+1})}{L\|\sigma_\gamma(v_k^{s+1})\|^2}$, we further obtained

$$f(w_{k+1}^{s+1}) \le f(w_k^{s+1}) - \frac{(v_k^{s+1}\cdot\sigma_\gamma(v_k^{s+1}))^2}{2L\|\sigma_\gamma(v_k^{s+1})\|^2} + \eta_k(v_k^{s+1} - \nabla f(w_k^{s+1}))\cdot\sigma_\gamma(v_k^{s+1}). \tag{10}$$

For the last term in the right hand side of (10), we have

$$\begin{aligned}\eta_k(v_k^{s+1} - \nabla f(w_k^{s+1}))\cdot\sigma_\gamma(v_k^{s+1}) &= \frac{v_k^{s+1}\cdot\sigma_\gamma(v_k^{s+1})}{L\|\sigma_\gamma(v_k^{s+1})\|^2}(v_k^{s+1} - \nabla f(w_k^{s+1}))\cdot\sigma_\gamma(v_k^{s+1}) \\ &\le \frac{|v_k^{s+1}\cdot\sigma_\gamma(v_k^{s+1})|}{L\|\sigma_\gamma(v_k^{s+1})\|^2}\|v_k^{s+1} - \nabla f(w_k^{s+1})\|\|\sigma_\gamma(v_k^{s+1})\| \\ &= \frac{|v_k^{s+1}\cdot\sigma_\gamma(v_k^{s+1})|}{L\|\sigma_\gamma(v_k^{s+1})\|}\|v_k^{s+1} - \nabla f(w_k^{s+1})\|, \end{aligned} \tag{11}$$

where the first inequality can be satisfied due to the Cauchy-Schwartz inequality.

Further, via utilizing the fact $2ab \le \theta a^2 + \frac{1}{\theta}b^2$, where θ denotes any positive real number, and a and b are also two real numbers, we have the following result:

$$\eta_k(v_k^{s+1} - \nabla f(w_k^{s+1}))\cdot\sigma_\gamma(v_k^{s+1}) \le \frac{1}{2L}\left(\theta\frac{(v_k^{s+1}\cdot\sigma_\gamma(v_k^{s+1}))^2}{\|\sigma_\gamma(v_k^{s+1})\|^2} + \frac{1}{\theta}\|v_k^{s+1} - \nabla f(w_k^{s+1})\|^2\right). \tag{12}$$

Combining the inequalities (10) and (12), we obtain

$$f(w_{k+1}^{s+1}) \le f(w_k^{s+1}) - \frac{1-\theta}{2L}\frac{(v_k^{s+1}\cdot\sigma_\gamma(v_k^{s+1}))^2}{\|\sigma_\gamma(v_k^{s+1})\|^2} + \frac{1}{2L\theta}\|v_k^{s+1} - \nabla f(w_k^{s+1})\|^2. \tag{13}$$

Based on the Hölder's inequality for $\gamma \in (0,1)$ with $p = \frac{1+\gamma}{1-\gamma}$ and $q = \frac{1+\gamma}{2\gamma}$, we have the fact

$$\begin{aligned}\|\sigma_\gamma(v_k^{s+1})\|^2 &= \sum_{i=1}^{d}|(v_k^{s+1})_i|^{2\gamma} \\ &\le \left(\sum_{i=1}^{d}1^p\right)^{\frac{1}{p}}\left(\sum_{i=1}^{d}\left(|(v_k^{s+1})_i|^{2\gamma}\right)^q\right)^{\frac{1}{q}} \\ &= \|\mathbf{1}\|_p\left(\sum_{i=1}^{d}|(v_k^{s+1})_i|^{1+\gamma}\right)^{\frac{2\gamma}{1+\gamma}}. \end{aligned} \tag{14}$$

Therefore, we have

$$\begin{aligned}\frac{(v_k^{s+1}\cdot\sigma_\gamma(v_k^{s+1}))^2}{\|\sigma_\gamma(v_k^{s+1})\|^2} &\ge \frac{\left(\sum_{i=1}^{d}|(v_k^{s+1})_i|^{1+\gamma}\right)^2}{\|\mathbf{1}\|_p\left(\sum_{i=1}^{d}|v_k^{s+1}|^{1+\gamma}\right)^{\frac{2\gamma}{1+\gamma}}} \\ &= \frac{\|v_k^{s+1}\|_{1+\gamma}^2}{\|\mathbf{1}\|_p}. \end{aligned} \tag{15}$$

The following results can be derived by utilizing the inequalities (13) and (15) simultaneously:

$$f(w_{k+1}^{s+1}) \leq f(w_k^{s+1}) - \frac{1-\theta}{2L}\frac{\|v_k^{s+1}\|_{1+\gamma}^2}{\|\mathbf{1}\|_p} + \frac{1}{2L\theta}\|v_k^{s+1} - \nabla f(w_k^{s+1})\|^2. \tag{16}$$

Now, we consider the second term and the third term of the right hand side of (16), respectively. First, considering $\|v_k^{s+1}\|_{1+\gamma}^2$, defining in PB-SVRGE (Algorithm 1), we have

$$\begin{aligned}\|v_k^{s+1}\|_{1+\gamma}^2 &= \|\nabla f_S(w_k^{s+1}) - \nabla f_S(\tilde{w}^s) + \nabla f(\tilde{w}^s)\|_{1+r}^2 \\ &\geq \frac{1}{2}\|\nabla f_S(w_k^{s+1})\|_{1+\gamma}^2 - \|\nabla f_S(\tilde{w}^s) - \nabla f(\tilde{w}^s)\|_{1+r}^2,\end{aligned} \tag{17}$$

where the first inequality holds due to the fact $\|a\|_{1+\gamma}^2 \geq \frac{1}{2}\|b\|_{1+\gamma}^2 - \|b-a\|_{1+\gamma}^2$.

Second, considering $\|v_k^{s+1} - \nabla f(w_k^{s+1})\|^2$, we further obtain

$$\begin{aligned}\|v_k^{s+1} - \nabla f(w_k^{s+1})\|^2 &= \|\nabla f_S(w_k^{s+1}) - \nabla f_S(\tilde{w}^s) + \nabla f(\tilde{w}^s) - \nabla f(w_k^{s+1})\|^2 \\ &\leq 2\|\nabla f_S(w_k^{s+1}) - \nabla f(w_k^{s+1})\|^2 + 2\|\nabla f_S(\tilde{w}^s) - \nabla f(\tilde{w}^s)\|^2,\end{aligned} \tag{18}$$

where the first inequality can be satisfied since the fact $\|a+b\|^2 \leq 2\|a\|^2 + 2\|b\|^2$ holds.

Combining (16), (17) and (18), we conclude that

$$\begin{aligned}f(w_{k+1}^{s+1}) \leq f(w_k^{s+1}) &- \frac{1-\theta}{4L\|\mathbf{1}\|_p}\|\nabla f_S(w_k^{s+1})\|_{1+\gamma}^2 + \frac{1-\theta}{2L\|\mathbf{1}\|_p}\|\nabla f_S(\tilde{w}^s) - \nabla f(\tilde{w}^s)\|_{1+r}^2 \\ &+ \frac{1}{L\theta}\left[\|\nabla f_S(w_k^{s+1}) - \nabla f(w_k^{s+1})\|^2 + \|\nabla f_S(\tilde{w}^s) - \nabla f(\tilde{w}^s)\|^2\right].\end{aligned} \tag{19}$$

To hold the inequality (19), it is appropriate to hold the following condition, that is

$$\begin{aligned}f(w_{k+1}^{s+1}) \leq f(w_k^{s+1}) &- \frac{1-\theta}{4L\|\mathbf{1}\|_p}\|\nabla f_S(w_k^{s+1})\|_{1+\gamma}^2 + \frac{1}{L\theta}[\|\nabla f_S(w_k^{s+1}) - \nabla f(w_k^{s+1})\|^2 \\ &+ \|\nabla f_S(\tilde{w}^s) - \nabla f(\tilde{w}^s)\|^2].\end{aligned} \tag{20}$$

Taking expectation on both sides of the inequality (20), we have

$$\mathbb{E}[f(w_{k+1}^{s+1}) - f(w_k^{s+1})] \leq \mathbb{E}\left[-\frac{1-\theta}{4L\|\mathbf{1}\|_p}\|\nabla f_S(w_k^{s+1})\|_{1+\gamma}^2\right] + \frac{2\hat{\sigma}^2}{Lb\theta}, \tag{21}$$

where in this inequality, we took the fact in Assumption 2.

By telescoping sum (21) over $k = 0, \ldots, K-1$, we have

$$\mathbb{E}[f(w_K^{s+1}) - f(w_0^{s+1})] \leq \mathbb{E}\left[-\frac{1-\theta}{4L\|\mathbf{1}\|_p}\sum_{k=0}^{K-1}\|\nabla f_S(w_k^{s+1})\|_{1+\gamma}^2\right] + \frac{2K\hat{\sigma}^2}{Lb\theta}. \tag{22}$$

In addition, according to $\tilde{w}^{s+1} = w_K^{s+1}$ and $w_0^{s+1} = w_K^s$, defined in Algorithm 1, we further obtain

$$\mathbb{E}[f(\tilde{w}^{s+1}) - f(\tilde{w}^s)] \leq \mathbb{E}\left[-\frac{1-\theta}{4L\|\mathbf{1}\|_p}\sum_{k=0}^{K-1}\|\nabla f_S(w_k^{s+1})\|_{1+\gamma}^2\right] + \frac{2K\hat{\sigma}^2}{Lb\theta}. \tag{23}$$

By telescoping (23) over $s = 0, \ldots, \mathfrak{S}-1$, we have

$$\mathbb{E}\left[\sum_{s=0}^{\mathfrak{S}-1}\sum_{k=0}^{K-1}\|\nabla f_S(w_k^{s+1})\|_{1+\gamma}^2\right] \leq \frac{4L\|\mathbf{1}\|_p}{1-\theta}\mathbb{E}[f(\tilde{w}^0)-f(w^*)] + \frac{8K\mathfrak{S}\|\mathbf{1}\|_p\hat{\sigma}^2}{b\theta(1-\theta)}$$
$$= \frac{4L\|\mathbf{1}\|_p}{1-\theta}\mathbb{E}[f(\tilde{w}^0)-f(w^*)] + \frac{8T\|\mathbf{1}\|_p\hat{\sigma}^2}{b\theta(1-\theta)}, \tag{24}$$

where the first inequality holds because of $w^* = \arg\min_w f(w)$ and the first equality holds due to $\mathfrak{S} = T/K$, defined in Algorithm 1.

By dividing T on both sides of (24), the desired results in Theorem 1 was derived. ■

As seen from Theorem 1, to satisfy $\mathbb{E}\left[\frac{1}{T}\sum_{s=0}^{\mathfrak{S}-1}\sum_{k=0}^{K-1}\|\nabla f_S(w_k^{s+1})\|_{1+\gamma}^2\right] \leq \varepsilon$, we only call for that the condition $\frac{4L\|\mathbf{1}\|_p}{T(1-\theta)}\mathbb{E}[f(\tilde{w}^0)-f(\tilde{w}^*)] + \frac{8\|\mathbf{1}\|_p\sigma^2}{b\theta(1-\theta)} \leq \varepsilon$ holds. More specifically, when adopting $b = O(T)$, to obtain an ε-accurate solution, PB-SVRGE (Algorithm 1) requires $T = O\left(\frac{4L\theta\|\mathbf{1}\|_pC+8\|\mathbf{1}\|_p\sigma^2}{\theta(1-\theta)\varepsilon}\right)$ iterations, where we set $C = \mathbb{E}[f(\tilde{w}^0)-f(w^*)]$. As pointed out by the work (Zhou et al., 2020), the classical powered stochastic gradient descent algorithms, pbSGD and pbSGDM, converged with rate $O\left(\frac{1}{\sqrt{T}}\right)$. In contrast, PB-SVRGE (Algorithm 1) converged with rate $O\left(\frac{1}{\sqrt{(1+2b)T}}\right)$, which is faster than the conventional powered stochastic gradient descent algorithms.

In particular, each epoch of PB-SVRGE (Algorithm 1) requires $n+2bK$ component gradient computations. Further, since the iteration numbers T has been some multiple of K and consider $K = o(n)$, we have that the overall complexity of PB-SVRGE (Algorithm 1) is $O\left(n+\frac{L^2\|\mathbf{1}\|_p^2}{\varepsilon^2}\right)$. To help understanding such a theoretical result well, we provided the computational complexity of several advanced stochastic optimization algorithms for solving non-convex cases. Lei et al. (2017) showed that the complexity of SCSG to achieve a stationary point was $O\left(n+\frac{n^{2/3}}{\varepsilon^2}\right)$ in the non-convex assumption. Fang et al. (2018) proved that SPIDER attained a complexity of $O\left(n+\frac{n^{1/2}}{\varepsilon^2}\right)$ for obtaining an ε-accurate first-order stationary point under the non-convex setting. Based on SARAH, Pham et al. (2020) proposed the proximal SARAH (ProxSARAH) approach and showed that its complexity of attaining an ε-accurate solution was also $O\left(n+\frac{n^{1/2}}{\varepsilon^2}\right)$ for non-convex cases. On the other hand, Reddi et al. (2016) analyzed that the complexity of SVRG and its mini-batch version for non-convex optimization was $O\left(n+\frac{n^{2/3}}{\varepsilon}\right)$ and $O\left(\min\left\{\frac{1}{\varepsilon^2}, \frac{n^{2/3}}{\varepsilon}\right\}\right)$, respectively. As seen from above-mentioned theoretical results, we can safely conclude that PB-SVRGE (Algorithm 1) matched advanced stochastic optimization algorithms.

4. Powered Stochastic Optimization with Adaptive Learning Rates

As pointed out by many references (Liu et al., 2018; Zhang et al., 2019; Yuan et al., 2019), PSO algorithms improved the original optimization algorithm through alternating the updating directions with a nonlinear mapping. Another preferred strategy of perfecting optimization algorithms was manipulating the learning rate per iterative step, covering a line search technique, an adaptive learning rate and a heuristic learning rate (Grippo et al., 1986; Yang, 2021a; Liu et al., 2019; Ip and Kahn, 2008). Notice that the accelerating mechanism of PSO was orthogonal to most of the

existing accelerating techniques for stochastic optimization algorithms. This section considers improving stochastic optimization by two perspectives of the learning rate and the updating directions simultaneously. So far, how to determine the learning rate for PSO was still an open problem.

So as to bridge the gap between PSO and the learning rate, this work explored how the random stabilized Barzilai-Borwein (RSBB) like technique, originally proposed by Yang (2021b), influenced PSO algorithms empirically. We first applied the RSBB-type approach into PB-SVRGE (Algorithm 1). Then, we discussed the behavior of more general PSO algorithms, PbSGD (shown in Zhou et al., 2020), with adaptive learning rates.

We here briefly discussed the RSBB method. In RSBB, it also employed the similar updating rule to SGD in (2), i.e., $w_{k+1} = w_k - \eta_k \nabla f_i(w_k)$, but the learning rate

$$\eta_k = \frac{\zeta}{b_H} \cdot \frac{\|s_k\|^2}{s_k \cdot y_k + \tau \|s_k\|^2}, \tag{25}$$

where $s_k = w_k - w_{k-1}$ and $y_k = \nabla f_{S_H}(w_k) - \nabla f_{S_H}(w_{k-1})$ for $k \geq 1$. Note that, here, $S_H \subseteq [n]$ with b_H samples (i.e., $|S_H| = b_H$) and $\nabla f_{S_H}(w_k) = \frac{1}{b_H} \sum_{i \in S_H} \nabla f_i(w_k)$. In addition, ζ and τ were two positive parameters. Specifically, ζ was employed to control the convergence speed. The term $\tau \|s_k\|^2$ was used to avoid the denominator of the original BB step size being zero. For the BB-type technique, it enjoyed a high-order information, resorting employing a simple but a low computational cost way to approximate the Hessian matrix unlike Newton or quasi-Newton type methods.

Next, we first equipped PR-SVRGE (Algorithm 1) with such adaptive learning rates, leading to a novel PSO algorithm: PB-SVRGE-RSBB (Algorithm 2). Following, we explored the performance of pbSGD with such an adaptive learning rate, leading to another new PSO algorithm: PB-SGD-RSBB (Algorithm 3).

Algorithm 2 PB-SVRGE-RSBB

Input: $\widetilde{w}^0 = w_K^0 = w^0$, epoch length K, outer loop size $\mathfrak{S} = T/K$, mini-batch size b, initial learning rate η_0, power coefficient γ, positive parameters ζ and τ.
for $s = 0, \ldots, \mathfrak{S} - 1$ **do**
  $w_0^{s+1} = w_K^s$
  $g^{s+1} = \nabla f(\tilde{w}^s)$
  for $k = 0, \ldots, K-1$ **do**
    Randomly choose $S \subseteq [n]$ with $|S| = b$, compute a new stochastic estimate $\nabla f(w_k^{s+1})$;
    $v_k^{s+1} = \nabla f_S(w_k^{s+1}) - \nabla f_S(\tilde{w}^s) + g^{s+1}$
    $w_{k+1}^{s+1} = w_k^{s+1} - \eta_k \sigma_\gamma(v_k^{s+1})$
    Randomly choose $S_H \subseteq [n]$ with $|S_H| = b_H$ and compute the learning rate
    $\eta_{k+1} = \frac{\zeta}{b_H} \cdot \frac{\|w_{k+1}^{s+1} - w_k^{s+1}\|^2}{(w_{k+1}^{s+1} - w_k^{s+1}) \cdot (\sigma_\gamma(\nabla f_{S_H}(w_{k+1}^{s+1})) - \sigma_\gamma(\nabla f_{S_H}(w_k^{s+1}))) + \tau \|w_{k+1}^{s+1} - w_k^{s+1}\|^2}$
  end for
  Set $\widetilde{w}^{s+1} = w_K^{s+1}$
end for

Remark 2: For PB-SVRGE-RBB (Algorithm 2) and PB-SGD-RBB (Algorithm 3), we offered the following remarks:

(1) The original BB method and its variants have been widely adopted not only in classical stochastic optimization algorithms (Sopyła and Drozda, 2015; De et al., 2017), but also in

Algorithm 3 PB-SGD-RSBB

Input: update frequency T; initial point $\widetilde{w}_0$; starting learning rate η_0; a constant power coefficient γ, positive parameters ζ and τ.
for $t = 0, \ldots, T-1$ **do**
Randomly choose $S \subseteq [n]$ with $|S| = b$ and update:
$w_{t+1} = w_t - \eta_t \sigma_\gamma(\nabla f_S(w_t))$
Randomly choose $S_H \subseteq [n]$ with $|S_H| = b_H$ and compute the learning rate
$\eta_t = \frac{\zeta}{b_H} \cdot \frac{\|w_t - w_{t-1}\|^2}{(w_t - w_{t-1})^T(\sigma_\gamma(\nabla f_{S_H}(w_t)) - \sigma_\gamma(\nabla f_{S_H}(w_{t-1}))) + \tau\|w_t^{s+1} - w_{t-1}^{s+1}\|^2}$
end for
Return: w_T

modern stochastic optimization algorithms (Yang et al., 2018a,b) during recent years. However, to the best of our knowledge, the research of selecting the learning rate for PSO algorithms was quite limited. Additionally, when computing the learning rate in our proposed algorithms, we also adopted the Powerball function that was totally different from the existing literature, which used the BB-like technique to obtain the learning rate.

(2) The idea of the BB-like method was coming from the quasi-Newton method through using a scalar matrix to approximate the curvature information. In this work, both the theoretical and numerical results showed the great promising, using the scalar matrix with the Powerball function to approximate the true Hessian matrix.

(3) Recent studies paid attention to the behavior of the Powerball function on conventional SGD, SGD with momentum, or the quasi-Newton method (Zhou et al., 2020; Yuan et al., 2019). In practice, we can easily incorporate such an adaptive learning rate into these different PSO algorithms. For conciseness, we do not show this case in this paper.

4.1 Convergence Analysis of PB-SVRGE-RSBB

The boundary of the learning rate η_k was first established in the following lemma.

Lemma 1. *From the definition of η_k in PB-SVRGE-RSBB (Algorithm 2), we have the following conclusion*

$$\eta_k \leq \frac{\zeta}{b_H \tau}. \tag{26}$$

Proof From the definition of η_k in PB-SVRGE-RSBB (Algorithm 2), we have

$$\begin{aligned} \eta_{k+1} &= \frac{\zeta}{b_H} \cdot \frac{\|w_{k+1}^{s+1} - w_k^{s+1}\|^2}{(w_{k+1}^{s+1} - w_k^{s+1}) \cdot (\sigma_\gamma(\nabla f_{S_H}(w_{k+1}^{s+1})) - \sigma_\gamma(\nabla f_{S_H}(w_k^{s+1}))) + \tau\|w_{k+1}^{s+1} - w_k^{s+1}\|^2} \\ &\leq \frac{\zeta}{b_H} \cdot \frac{\|w_{k+1}^{s+1} - w_k^{s+1}\|^2}{\tau\|w_{k+1}^{s+1} - w_k^{s+1}\|^2} = \frac{\zeta}{\tau b_H}. \end{aligned}$$

■

Note that the original BB technique (Barzilai and Borwein, 1988) that used the iterative scheme $\eta_k = \frac{\|s_k\|^2}{s_k^T y_k}$ to update the learning rate usually kept such the learning rate positive under deterministic optimization background, which meant that the inner product of the vectors s_k and y_k was positive. Due to the introduction of the mini-batching and variance reduced techniques under stochastic optimization setting, it, here, still almost kept the inner product of the vectors s_k and y_k positive. This is indeed the case and can be easily confirmed through numerical experiments.

The following theorem established the convergence results of PB-SVRGE-RSBB under non-convex assumptions.

Theorem 2. *Set $w^* = \arg\min_w f(w)$, and choose $S \subseteq [n]$ and $S_H \subseteq [n]$ with $|S| = b$ and $|S_H| = b_H$ respectively. Let T denote the number of total iterations, then, under Assumption 1, Assumption 2 and Lemma 1, for any $T \geq 1$, PB-SVRGE-RSBB (Algorithm 2) leads to*

$$\mathbb{E}\left[\frac{1}{T}\sum_{s=0}^{\mathfrak{S}-1}\sum_{k=0}^{K-1}\|\nabla f_S(w_k^{s+1})\|_{1+\gamma}^2\right] \leq \frac{4b_H^2 L\|\mathbf{1}\|_p}{(2b_H-\theta b_H-1)T}\mathbb{E}[f(\tilde{w}^0)-f(w^*)] + \frac{8\hat{\sigma}^2 b_H\|\mathbf{1}\|_p}{(2b_H-\theta b_H-1)b\theta}. \tag{27}$$

Proof According to the results in the inequality (6) and $w_{k+1}^{s+1} = w_k^{s+1} - \eta_k\sigma_\gamma(v_k^{s+1})$, we easily inferred

$$\begin{aligned}
f(w_{k+1}^{s+1}) &\leq f(w_k^{s+1}) + \nabla f(w_k^{s+1})\cdot(w_{k+1}^{s+1}-w_k^{s+1}) + \frac{L}{2}\|w_{k+1}^{s+1}-w_k^{s+1}\|^2 \\
&= f(w_k^{s+1}) - \eta_k\nabla f(w_k^{s+1})\cdot\sigma_\gamma(v_k^{s+1}) + \frac{L\eta_k^2}{2}\|\sigma_\gamma(v_k^{s+1})\|^2 \\
&\leq f(w_k^{s+1}) - \frac{\zeta}{b_H\tau}\nabla f(w_k^{s+1})\cdot\sigma_\gamma(v_k^{s+1}) + \frac{L\zeta^2}{2b_H^2\tau^2}\|\sigma_\gamma(v_k^{s+1})\|^2 \\
&= f(w_k^{s+1}) - \frac{\zeta}{b_H\tau}v_k^{s+1}\sigma_\gamma(v_k^{s+1}) + \frac{L\zeta^2}{2b_H^2\tau^2}\|\sigma_\gamma(v_k^{s+1})\|^2 + \frac{\zeta}{b_H\tau}(v_k^{s+1} \\
&\quad - \nabla f(w_k^{s+1}))\sigma_\gamma(v_k^{s+1}),
\end{aligned} \tag{28}$$

where the second inequality utilized the fact in Lemma 1.

When setting $\zeta = \frac{\tau v_k^{s+1}\cdot\sigma_\gamma(v_k^{s+1})}{L\|\sigma_\gamma(v_k^{s+1})\|^2}$, we have

$$\begin{aligned}
f(w_{k+1}^{s+1}) &\leq f(w_k^{s+1}) - \frac{(2b_H-1)(v_k^{s+1}\cdot\sigma_\gamma(v_k^{s+1}))^2}{2Lb_H^2\|\sigma_\gamma(v_k^{s+1})\|^2} + \frac{v_k^{s+1}\cdot\sigma_\gamma(v_k^{s+1})}{b_H L\|\sigma_\gamma(v_k^{s+1})\|^2}\cdot(v_k^{s+1} \\
&\quad - \nabla f(w_k^{s+1}))\cdot\sigma_\gamma(v_k^{s+1}) \\
&\leq f(w_k^{s+1}) - \frac{(2b_H-1)(v_k^{s+1}\cdot\sigma_\gamma(v_k^{s+1}))^2}{2Lb_H^2\|\sigma_\gamma(v_k^{s+1})\|^2} + \frac{|v_k^{s+1}\cdot\sigma_\gamma(v_k^{s+1})|}{b_H L\|\sigma_\gamma(v_k^{s+1})\|}\cdot\|v_k^{s+1} \\
&\quad - \nabla f(w_k^{s+1})\|,
\end{aligned} \tag{29}$$

where the last inequality took the Cauchy-Schwartz inequality.

Further, via the condition $2ab \leq \theta a^2 + \frac{1}{\theta} b^2$, we have

$$\begin{aligned} f(w_{k+1}^{s+1}) &\leq f(w_k^{s+1}) - \frac{(2b_H - 1)(v_k^{s+1} \cdot \sigma_\gamma(v_k^{s+1}))^2}{2Lb_H^2 \|\sigma_\gamma(v_k^{s+1})\|^2} + \frac{\theta(v_k^{s+1} \cdot \sigma_\gamma(v_k^{s+1}))^2}{2b_H L \|\sigma_\gamma(v_k^{s+1})\|^2} \\ &\quad + \frac{\|v_k^{s+1} - \nabla f(w_k^{s+1})\|^2}{2b_H L\theta} \\ &= f(w_k^{s+1}) - \frac{(2b_H - \theta b_H - 1)(v_k^{s+1} \cdot \sigma_\gamma(v_k^{s+1}))^2}{2Lb_H^2 \|\sigma_\gamma(v_k^{s+1})\|^2} + \frac{\|v_k^{s+1} - \nabla f(w_k^{s+1})\|^2}{2b_H L\theta}. \end{aligned} \tag{30}$$

According to the conclusion in the inequality (15), we derived:

$$f(w_{k+1}^{s+1}) \leq f(w_k^{s+1}) - \frac{(2b_H - \theta b_H - 1)}{2Lb_H^2} \cdot \frac{\|v_k^{s+1}\|_{1+\gamma}^2}{\|\mathbf{1}\|_p} + \frac{\|v_k^{s+1} - \nabla f(w_k^{s+1})\|^2}{2b_H L\theta}. \tag{31}$$

Combining (31), (17) and (18), the following conclusion was obtained

$$\begin{aligned} f(w_{k+1}^{s+1}) &\leq f(w_k^{s+1}) - \frac{(2b_H - \theta b_H - 1)}{4Lb_H^2 \|\mathbf{1}\|_p} \cdot \|\nabla f(w_k^{s+1})\|_{1+\gamma}^2 + \frac{(2b_H - \theta b_H - 1)}{2Lb_H^2 \|\mathbf{1}\|_p} \\ &\quad \cdot \|\nabla f_S(\tilde{w}_s) - \nabla f(\tilde{w}_s)\|_{1+\gamma}^2 + \frac{1}{b_H L\theta}[\|\nabla f_S(w_k^{s+1}) - \nabla f(w_k^{s+1})\|^2 \\ &\quad + \|\nabla f_S(\tilde{w}^s) - \nabla f(\tilde{w}^s)\|^2]. \end{aligned} \tag{32}$$

To make the inequality (19) be satisfied, we only keep the following condition

$$\begin{aligned} f(w_{k+1}^{s+1}) &\leq f(w_k^{s+1}) - \frac{(2b_H - \theta b_H - 1)}{4Lb_H^2 \|\mathbf{1}\|_p} \cdot \|\nabla f(w_k^{s+1})\|_{1+\gamma}^2 + \frac{1}{b_H L\theta}[\|\nabla f_S(w_k^{s+1}) - \nabla f(w_k^{s+1})\|^2 \\ &\quad + \|\nabla f_S(\tilde{w}^s) - \nabla f(\tilde{w}^s)\|^2]. \end{aligned} \tag{33}$$

Taking expectation on both sides of the inequality (33), we have

$$\mathbb{E}[f(w_{k+1}^{s+1}) - f(w_k^{s+1})] \leq \mathbb{E}\left[-\frac{(2b_H - \theta b_H - 1)}{4Lb_H^2 \|\mathbf{1}\|_p} \|\nabla f_S(w_k^{s+1})\|_{1+\gamma}^2\right] + \frac{2\hat{\sigma}^2}{Lb_H b\theta}, \tag{34}$$

where in this inequality, we took the fact in Assumption 2 simultaneously.

When summing (34) over $k = 0, \ldots, K-1$ recursively, we have

$$\mathbb{E}[f(w_K^{s+1}) - f(w_0^{s+1})] \leq \mathbb{E}\left[-\frac{(2b_H - \theta b_H - 1)}{4Lb_H^2 \|\mathbf{1}\|_p} \sum_{k=0}^{K-1} \|\nabla f_S(w_k^{s+1})\|_{1+\gamma}^2\right] + \frac{2K\hat{\sigma}^2}{Lb_H b\theta}. \tag{35}$$

In addition, according to $\tilde{w}^{s+1} = w_K^{s+1}$ and $w_0^{s+1} = w_K^s$, defined in Algorithm 2, we further obtain

$$\mathbb{E}[f(\tilde{w}^{s+1}) - f(\tilde{w}^s)] \leq \mathbb{E}\left[-\frac{(2b_H - \theta b_H - 1)}{4Lb_H^2 \|\mathbf{1}\|_p} \sum_{k=0}^{K-1} \|\nabla f_S(w_k^{s+1})\|_{1+\gamma}^2\right] + \frac{2K\hat{\sigma}^2}{Lb_H b\theta}. \tag{36}$$

In addition, via telescoping (36) over $s = 0, \ldots, \mathfrak{S}-1$, we derive

$$\mathbb{E}\left[\sum_{s=0}^{\mathfrak{S}-1}\sum_{k=0}^{K-1}\|\nabla f_S(w_k^{s+1})\|_{1+\gamma}^2\right] \leq \frac{4b_H^2 L\|\mathbf{1}\|_p}{2b_H-\theta b_H-1}\mathbb{E}[f(\tilde{w}^0)-f(w^*)] + \frac{8b_H K\mathfrak{S}\|\mathbf{1}\|_p\hat{\sigma}^2}{(2b_H-\theta b_H-1)b\theta}$$
$$= \frac{4b_H^2 L\|\mathbf{1}\|_p}{2b_H-\theta b_H-1}\mathbb{E}[f(\tilde{w}^0)-f(w^*)] + \frac{8\hat{\sigma}^2 b_H T\|\mathbf{1}\|_p}{(2b_H-\theta b_H-1)b\theta}, \quad (37)$$

where the first inequality holds because of $w^* = \arg\min_w f(w)$ and the first equality uses $\mathfrak{S} = T/K$, defined in Algorithm 2.

Finally, by dividing T on both sides of (37), the desired results in Theorem 2 was obtained. ■

Now, we discussed the computational complexity of PB-SVRGE-RSBB (Algorithm 2) for non-convex loss functions. PB-SVRGE-RSBB (Algorithm 2) requires $T = O\left(\frac{4b_H^2 L\theta C\|\mathbf{1}\|_p + 8\hat{\sigma}^2 b_H\|\mathbf{1}\|_p}{(2b_H-\theta b_H-1)\theta\varepsilon}\right)$ iterations to obtain an ε-accurate solution, where we set $b = O(T)$ and $C = \mathbb{E}[f(\tilde{w}^0)-f(w^*)]$. Additionally, observed by Algorithm 2, each stage of PB-SVRGE-RSBB requires $n+2(b+b_H)K$ component gradient evaluations. Further, consider $K = o(n)$, we obtain that the complexity of PB-SVRGE-RSBB (Algorithm 2) is $O\left(n + \frac{b_H^2 L^2\|\mathbf{1}\|_p^2 + \varepsilon b_H^2 L\|\mathbf{1}\|_p}{\varepsilon^2}\right)$. Moreover, we have that PB-SVRGE-RSBB (Algorithm 2) converged with rate $O\left(\frac{1}{\sqrt{1+2(b+b_H)T}}\right)$, which is faster than the conventional PSO algorithms as well.

4.2 Convergence Analysis for PB-SGD-RSBB

This part will provide the theoretical analysis of PB-SGD-RSBB (Algorithm 3) under non-convex assumptions.

Theorem 3. *Set $w^* = \arg\min_w f(w)$, and choose $S \subseteq [n]$ and $S_H \subseteq [n]$ with $|S| = b$ and $|S_H| = b_H$ respectively. Let T denote the number of total iterations, then, under Assumption 1, Assumption 2 and Lemma 1, for any $T \geq 1$, PB-SGD-RSBB (Algorithm 3) results in*

$$\mathbb{E}\left[\frac{1}{T}\sum_{t=0}^{T-1}\|\nabla f_S(w_k^{s+1})\|_{1+\gamma}^2\right] \leq \frac{2b_H^2 L\|\mathbf{1}\|_p}{(2b_H-\theta b_H-1)T}\mathbb{E}[f(\tilde{w}^0)-f(w^*)] + \frac{\hat{\sigma}^2 b_H\|\mathbf{1}\|_p}{(2b_H-\theta b_H-1)b\theta}. \quad (38)$$

Notice that, the proof of Theorem 3 can follow from the analysis in proving Theorem 2. In the following, we only sketched the key points in proving Theorem 3.

Proof According to the inequality (6) and Lemma 1, we have the following conclusion

$$f(w_{t+1}) \leq f(w_t) - \frac{\zeta}{b_H\tau}\nabla f_S(w_t)\cdot\sigma_\gamma(\nabla f_S(w_t)) + \frac{L\zeta^2}{2b_H^2\tau^2}\|\sigma_\gamma(\nabla f_S(w_t))\|^2 + \frac{\zeta}{b_H\tau}(\nabla f_S(w_t)$$
$$-\nabla f(w_t))\cdot\sigma_\gamma(\nabla f_S(w_t)). \quad (39)$$

Adopting $\zeta = \frac{\tau\nabla f_S(w_t)\cdot\sigma_\gamma(\nabla f_S(w_t))}{L\|\sigma_\gamma(\nabla f_S(w_t))\|^2}$, we obtain

$$f(w_{t+1}) \leq f(w_t) - \frac{2b_H-\theta b_H-1}{2b_H^2 L\|\mathbf{1}\|_p}\|\nabla f_S(w_t)\|_{1+\gamma}^2 + \frac{\hat{\sigma}^2}{2b_H Lb\theta}. \quad (40)$$

Summing the inequality (40) over $t = 0, 1, \ldots T-1$ recursively, we derive

$$\mathbb{E}\big[\sum_{t=0}^{T-1} \|\nabla f_S(w_t)\|_{1+\gamma}^2\big] \leq \frac{2b_H^2 L\|\mathbf{1}\|_p}{2b_H - \theta b_H - 1}\mathbb{E}[f(w_0) - f(w_*)] + \frac{b_H T\hat{\sigma}^2 L\|\mathbf{1}\|_p}{(2b_H - \theta b_H - 1)\theta b}. \tag{41}$$

Finally, by diving T on both sides of the inequality (41), we can obtain the desired results. ■

According to Theorem 3, we easily derived that the total complexity of PB-SGD-RSBB (Algorithm 3) was $O\left(\frac{b_H^2 L^2\|\mathbf{1}\|_p^2 + \varepsilon b_H^2 L\|\mathbf{1}\|_p}{\varepsilon^2}\right)$ for non-convex optimization problems, when employing $b = O(T)$. Additionally, we obtain that PB-SGD-RSBB (Algorithm 3) converged with rate $O\left(\frac{1}{\sqrt{b+b_H T}}\right)$, which recovers optimal $O\left(\frac{1}{\sqrt{bT}}\right)$ convergence speed of mini-batch stochastic optimization algorithms (Li et al., 2014).

5. Experiments

In order to evaluate the proposed algorithms, we conducted the experiments on six standard data sets, where the details of these data sets were listed in Table 1. [1] Specifically, the data sets (a8a, covtype, ijcnn1 and news20) can be downloaded from LIBSVM (Chang and Lin, 2011). In addition, we took the non-convex logistic regression as the loss function. Given a set of example pairs $\{a_i, b_i\}_{i=1}^n$, the goal was to find a solution of the following loss function:

$$\min_{w\in\mathbb{R}^d} \quad f(w) := \frac{1}{n}\sum_{i=1}^{n} \log(1 + \exp(-y_i x_i^T w)) + \lambda r(w), \tag{42}$$

where $r(w) = \sum_{i=1}^d \frac{w_i^2}{1+w_i^2}$ is a non-convex regularizer. In practice, the non-convex regularizer was widely adopted in machine learning and statistical learning such as gaining robustness and approximating sparsity. The model (42) has already been used in (Wang et al., 2020; Yang, 2021a; Nguyen et al., 2021). We studied the numerical behaviors of the proposed algorithms with the regularizer coefficient $\lambda = 10^{-1}$. Note that all experiments were conducted on an Intel(R) Core(TM) i7-10750H CPU @2.60GHz 2.59GHz with MATLAB 2019a.

Table 1: Summary of data sets

Data set	♯ examples	♯ features
a8a	22,696	123
covtype	581,012	54
CIFAR-10	60,000	1,024
ijcnn1	49,990	22
MNIST	60,000	784
news20.binary	19,996	1,355,191

1. For the CIFAR-10 data set, it can be downloaded from `http://www.cs.toronto.edu/~kriz/cifar.html`. For the MNIST data set, it can be accessed from `http://yann.lecun.com/exdb/mnist/`.

5.1 Properties of PB-SVRGE

We first demonstrate the numerical behavior of PB-SVRGE (Algorithm 1) on various standard data sets, provided in Table 1. In our experiments, we selected n stochastic gradient computations (a.k.a. a full gradient evaluation counts) as one number of effective passes. Without otherwise specified, in all figures, the horizontal axis denoted the number of effective passes and the vertical axis represented the objective gap: $f(\tilde{w}_s) - f(w^*)$. Such criterion was widely adopted to verify the efficacy of the algorithms, see, e.g., Johnson and Zhang (2013); Roux et al. (2012); Nitanda (2014); Yang (2022).

As seen from Algorithm 1, the performance of PB-SVRGE highly relies on the power coefficient γ, the mini-batch size b, the learning rate η_k. As a result, we will discuss the effect of these parameters on PB-SVRGE respectively.

Figure 1 plots the performance of PB-SVRGE when using different power coefficients, where the power coefficient γ was chosen from $[0,1]$. On all data sets, we set $b = 10$. In addition, on *a*8*a* and *ijcnn*1, we set $\eta = 0.01$. While on *covtype* and *news*20.*binary*, we set $\eta = 0.1$. As presented in Figure 1, PB-SVRGE achieved better performance on a slighter large power coefficient. It was noted that when taking $\gamma = 1$, PB-SVRGE was reduced to the original SVRG method. Therefore, Figure 1 demonstrated that PB-SVRGE matched or even outperformed the original SVRG method, which confirmed the effectiveness of the Powerball function in improving stochastic optimization. Additionally, when taking $\gamma = 0$, PB-SVRGE was viewed as a kind of a sign SVRG method. Obviously, Figure 1 showed that the sign-based variance-reduced algorithm performed worse on different data sets.

Figure 2 shows the numerical behavior of PB-SVRGE when we took four different mini-batch sizes. When conducting experiments on *a*8*a* and *ijcnn*1, we set $\eta = 0.01$ and $\gamma = 0.9$; In addition, for *covtype* and *news*20.*binary*, we set $\eta = 0.1$ and $\gamma = 0.9$. The cases of the mini-batch sizes can be found in the legend of Figure 2. It can be observed from Figure 2, PB-SVRGE achieved better performance with a small mini-batch size.

Figure 3 presented the properties of PB-SVRGE when we employed four different learning rates. For different data sets, we took $b = 10$ and $\gamma = 0.9$. Figure 3 pointed out that PB-SVRGE converged slowly with a small learning rate, while diverging with a large learning rate. In practice, it's a tedious work to choose an optimal learning rate from multiple learning rates. To address this issue, this work developed an adaptive strategy of updating the learning rate in Section 4.

5.2 Properties of PB-SVRGE-RSBB

It can be seen from Algorithm 2, the performance of PB-SVRGE-RSBB was highly influenced by the parameters ζ, τ, γ, b and b_H. Therefore, we will take the similar strategy of the case in discussing the numerical properties of PB-SVRGE to explore the properties of PB-SVRGE-RSBB step by step.

First of all, Figure 4 elucidated the performance of PB-SVRGE-RSBB with the parameter τ. In this case, we set $b = 10$, $b_H = 20$, $\zeta = 1$ and $\gamma = 0.9$ on different data sets when executing PB-SVRGE-RSBB (Algorithm 2). The parameter ζ was chosen from $\{0.01, 0.1, 1, 10\}$. Figure 4 demonstrated that the numerical behavior of PB-SVRGE-RSBB executed well on a small ζ.

Following, Figure 5 discussed the numerical performance of PB-SVRGE-RSBB (Algorithm 2) with the parameter ζ. For PB-SVRGE-RSBB, we set $b = 10$, $b_H = 20$ and $\gamma = 0.9$ on *a*8*a* and *ijcnn*1; $b = 10$, $b_H = 100$ and $\gamma = 0.9$ on *covtype* and *news*20. Figure 5 pointed out that a sightly

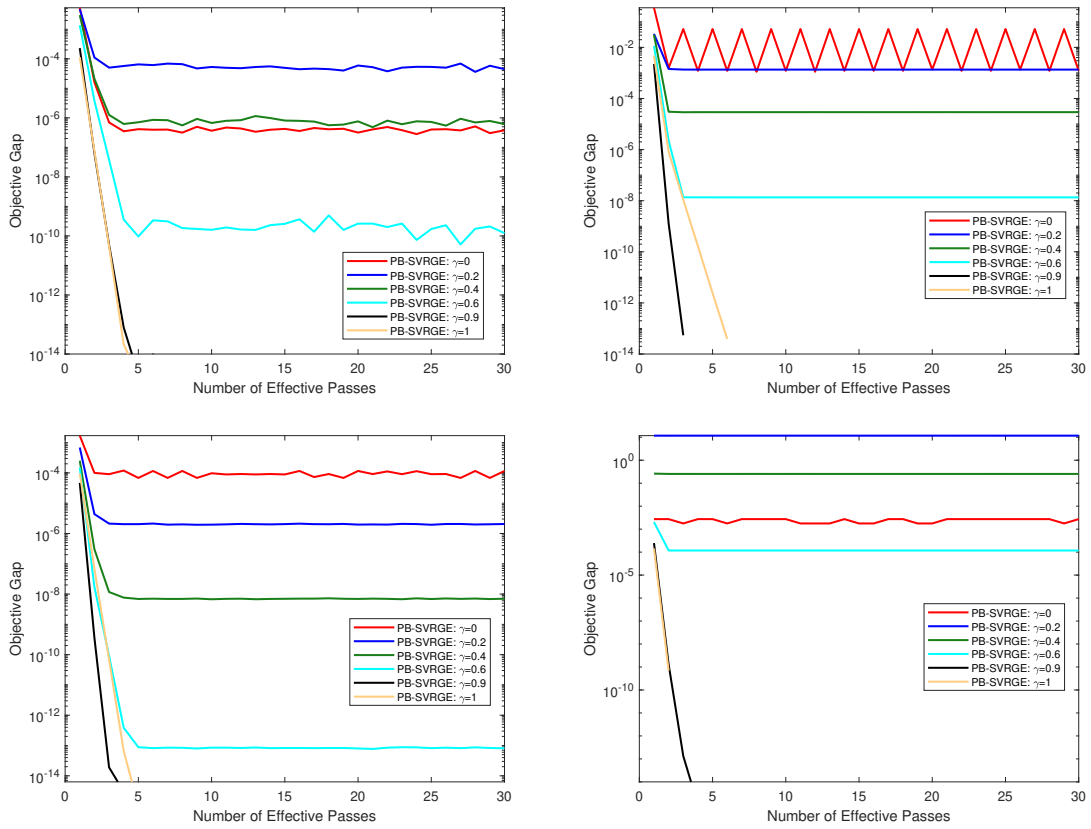

Figure 1: PB-SVRGE on *a8a* (top left), *covtype* (top right), *ijcnn*1 (bottom left), and *news*20.*binary* (bottom right): varying the power coefficient γ.

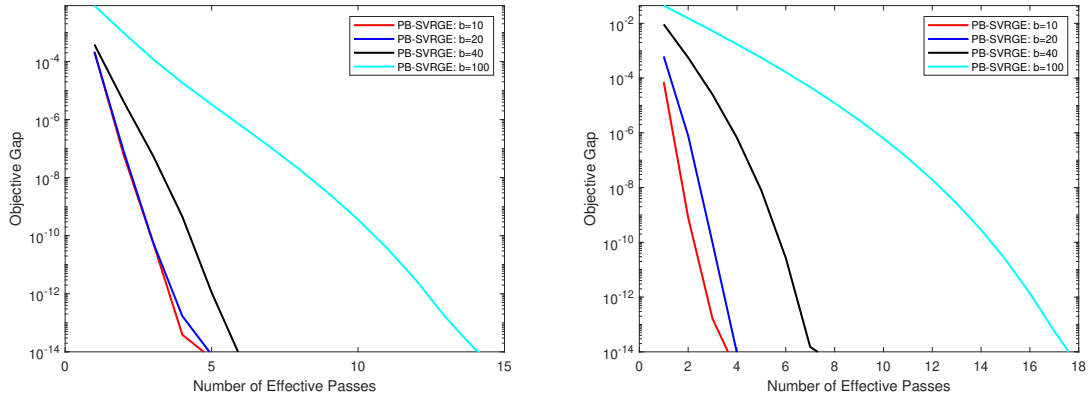

Figure 2: PB-SVRGE on *a8a* (left) and *ijcnn*1 (right): varying the mini-batch size b.

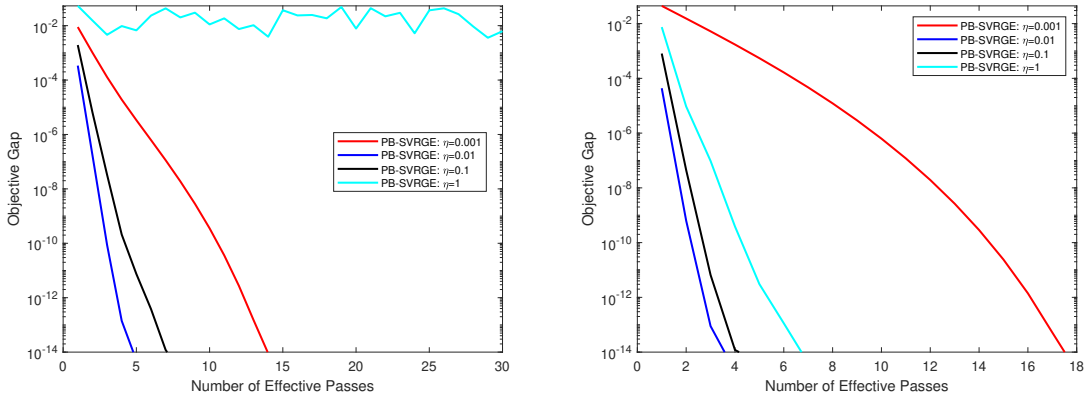

Figure 3: PB-SVRGE on *a8a* (left) and *ijcnn1* (right): varying the learning rate η.

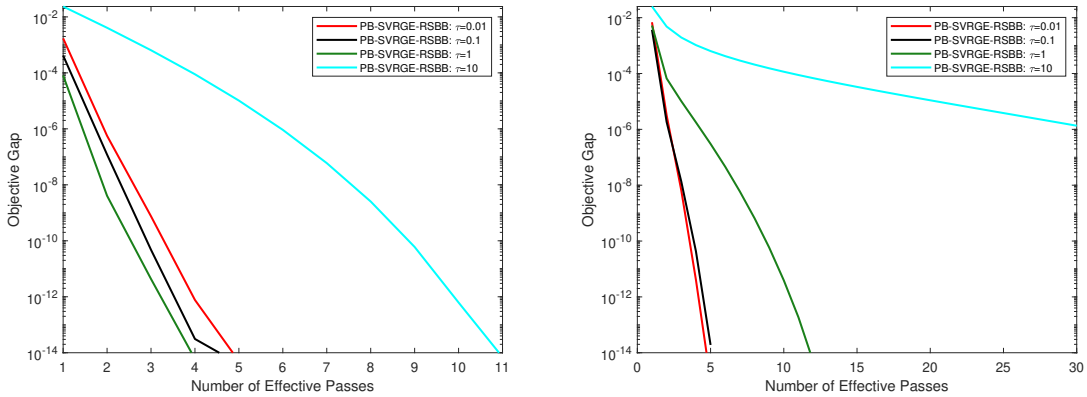

Figure 4: PB-SVRGE-RSBB on *ijcnn1* (left) and *covtype* (right): varying the parameter τ.

large ζ made PB-SVRGE-RSBB achieve better performance. It's noted that a larger ζ will lead to the divergence of the proposed algorithms.

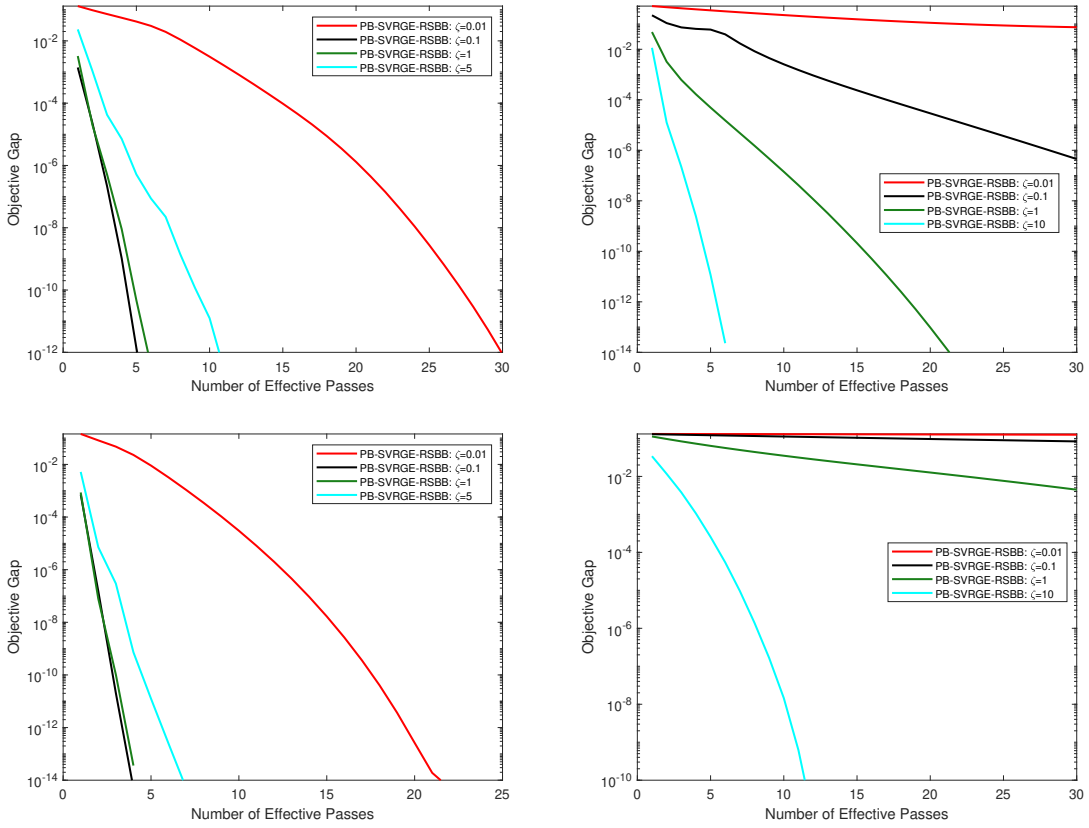

Figure 5: PB-SVRGE-RSBB on *a8a* (top left), *covtype* (top right), *ijcnn*1 (bottom left), and *news*20.*binary* (bottom right): varying the parameter ζ.

Figure 6 explored the properties of PB-SVRGE-RSBB with different power coefficients. For different data sets, we took $b = 10$ and $b_H = 20$. The details of the power coefficient were plotted in the legend of Figure 6. Figure 6 indicated that a bigger power coefficient made PB-SVRGE-RSBB converge speedily.

Further, Figure 7 and Figure 8 discussed the numerical behavior of PB-SVRGE-RSBB with the parameters b and b_H respectively. As seen from Figure 7, PB-SVRGE-RSBB was robust to the selection of the parameter b. In contrast, Figure 8 indicated PB-SVRGE-RSBB had better performance with a small parameter b_H.

5.3 Properties of PB-SGD-RSBB

PB-SGD-RSBB (Algorithm 3) has a analogical performance with PB-SVRGE-RSBB (Algorithm 2). As a consequence, to avoid redundancy, we only discussed the numerical behavior of PB-SGD-RSBB with the power coefficient γ alone.

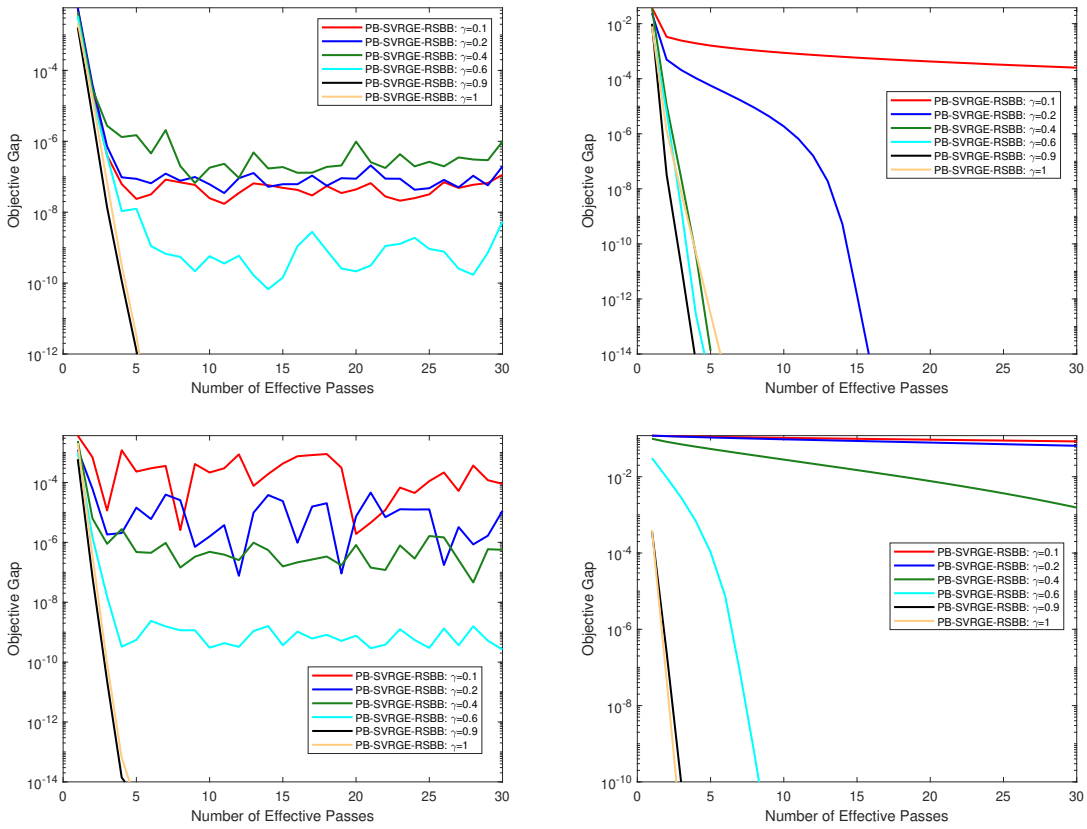

Figure 6: PB-SVRGE-RSBB on *a8a* (top left), *covtype* (top right), *ijcnn*1 (bottom left), and *news*20.*binary* (bottom right): varying the power coefficient γ.

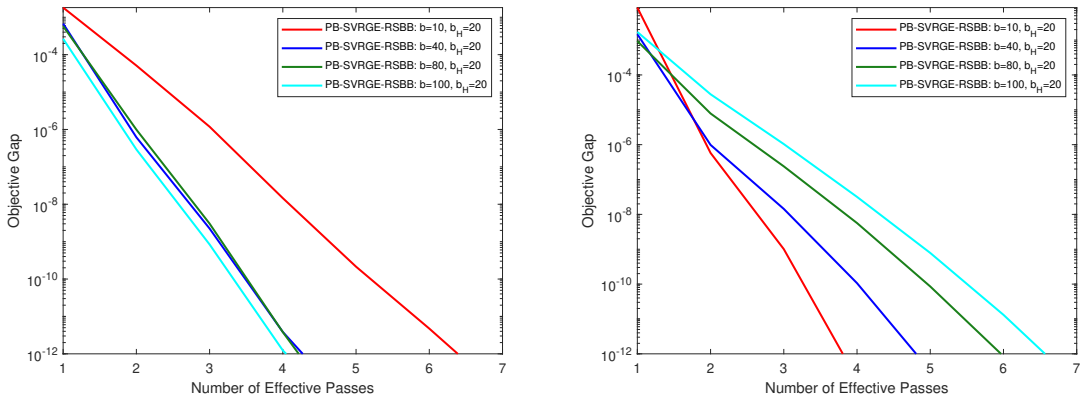

Figure 7: PB-SVRGE-RSBB on *a8a* (left) and *covtype* (right): varying the mini-batch size b.

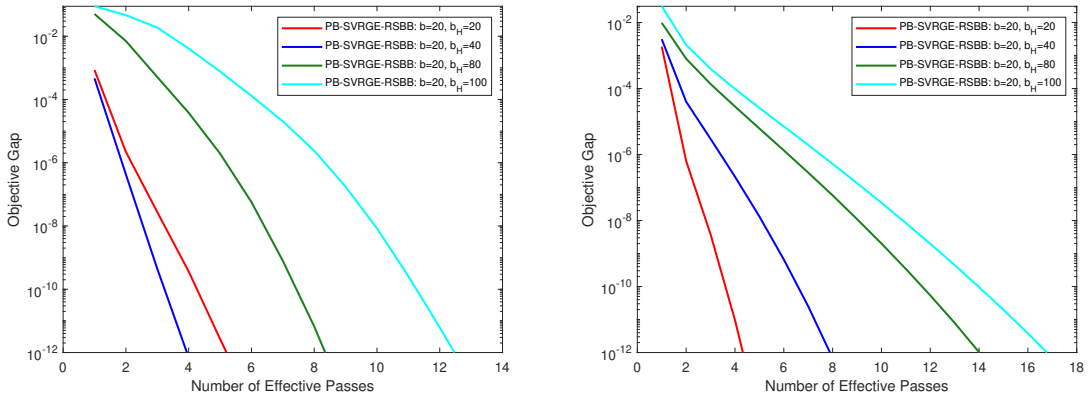

Figure 8: PB-SVRGE-RSBB on *a8a* (left) and *covtype* (right): varying the mini-batch size b_H.

Figure 9 showed that PB-SGD-RBB performed well under the case of a large power coefficient, which is parallel to the two previous algorithms.

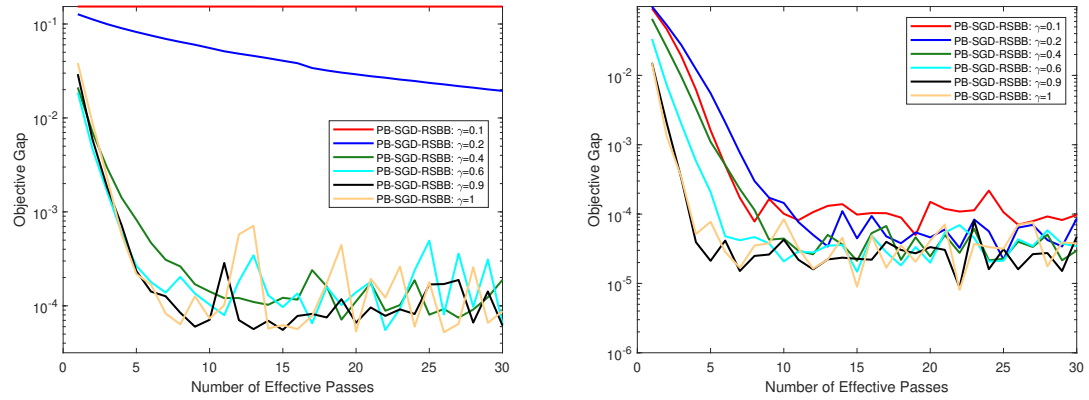

Figure 9: PB-SGD-RSBB on *a8a* (left) and *ijcnn1* (right): varying the power coefficient γ.

5.4 Comparison with Related Algorithms

So as to further demonstrate the efficacy of the proposed algorithms, the following algorithms were implemented to compare with PB-SGD-RSBB, PB-SVRGE and PB-SVRGE-RSBB:

- pbSGD: plain stochastic gradient descent with the Powerball function in Zhou et al. (2020). The pbSGD method worked with a constant learning rate. As suggested by Zhou et al. (2020), we chose the best learning rate from multiple learning rates. Moreover, we set the power coefficient $\gamma = 0.9$ for all data sets.

- pbSGDM: pbSGD with a momentum term (Polyak, 1964) in Zhou et al. (2020). The learning rate for pbSGDM was also chosen from multiple learning rates with best performance. We set the power coefficient $\gamma = 0.9$ and the momentum value $\beta = 0.9$ for various data sets.

- SGD-HD: vanilla stochastic gradient descent with the hyper-gradient descent approach in Baydin et al. (2018). We tested SGD-HD with parameters $\eta_0 = 10^{-3}$, $\beta = 10^{-4}$ on *a8a*, *ijcnn*1 and *news*20.*binary*, and $\eta_0 = 10^{-2}$, $\beta = 10^{-3}$ on *covtype*.
- SPIDER: stochastic path-integrated differential estimator in Fang et al. (2018). It is a kind of biased variance-reduced optimization algorithms, working with a constant learning rate. In experiments, SPIDER worked with $\eta = 0.1$ on *a8a* and $\eta = 0.2$ on other three data sets.
- CGVR: stochastic conjugate gradient with variance reduction in Jin et al. (2019). For CGVR, the line search technique was adopted to determine the learning rate.
- MSVRG-RSBB: stochastic variance reduction gradient algorithm with the random stabilized Barzilai-Borwein method in the mini-batching setting, shown in Yang (2021b). We tested MSVRG-RSBB with $b = 8$, $b_H = 8$, $\gamma = 1.4$, $\sigma = 0.25$ on *a8a*; $b = 8$, $b_H = 8$, $\gamma = 1$, $\sigma = 0.25$ on *covtype* and *ijcnn*1; $b = 8$, $b_H = 20$, $\gamma = 1$, $\sigma = 0.25$ on *news*20.*binary*.
- MB-SARAH-RSBB: stochastic recursive gradient algorithm with the random stabilized Barzilai-Borwein method in the mini-batching setting, appearing in Yang (2021b). We executed MB-SARAH-RSBB with $b = 4$, $b_H = 8$, $\gamma = 0.01$, $\sigma = 0.1$ on *a8a*, *covtype* and *ijcnn*1; $b = 4$, $b_H = 20$, $\gamma = 0.1$, $\sigma = 0.1$ on *news*20.*binary*.

Figure 10 showed the comparison of the three proposed algorithms (PB-SGD-RSBB, PB-SVRGE and PB-SVRGE-RSBB) with different approaches mentioned above on various data sets. We can clearly observe that PB-SVRGE and PB-SVRGE-RSBB were the two algorithms that had the best performance on all data sets, comparing with other advanced stochastic optimization algorithms. Especially, unlike MSVRG-RSBB and MB-SARAH-RSBB that were specifically designed for non-convex optimization problems and required commanding many hyperparameters at the same time in determining step size, our proposed algorithms were much easier to be implemented in practice.

5.5 Performance on Neural Networks

To further test the superior of the resulting algorithms, we train neural networks (NN). For clarity, this subsection also offers the comparison results between the resulting algorithms, PB-SGD-RSBB, PB-SVRGE and PB-SVRGE-RSBB, and other related algorithms. Specifically, similar to the work in (Johnson and Zhang, 2013), in our experiments, we train NNs with one fully-connected hidden layer of 100 nodes and 10 softmax output nodes.

The numerical results are offered in Figure 11. In Figure 11, the x-axis denotes the number of effective passes as well. In contrast, the y-axis corresponds to the accuracy of the algorithms on the test data sets.

Figure 11 demonstrates that PB-SVRGE (Algorithm 1) outperforms other modern stochastic optimization algorithms. In particular, the comparison results among PB-SVRGE (Algorithm 1), Acc-Prox-SVRG, and the original SVRG method imply the effectiveness of the Powerball function in improving stochastic optimization algorithms. Additionally, the comparison results among PB-SVRGE-RSBB (Algorithm 2), PB-SGD-RSBB (Algorithm 3), MSVRG-RSBB and pbSGD validates the efficacy of the update rule of RSBB in improving the powered stochastic optimization algorithms.

The numerical results on both non-convex logistic regression and neural networks show much promise of the resulting algorithms.

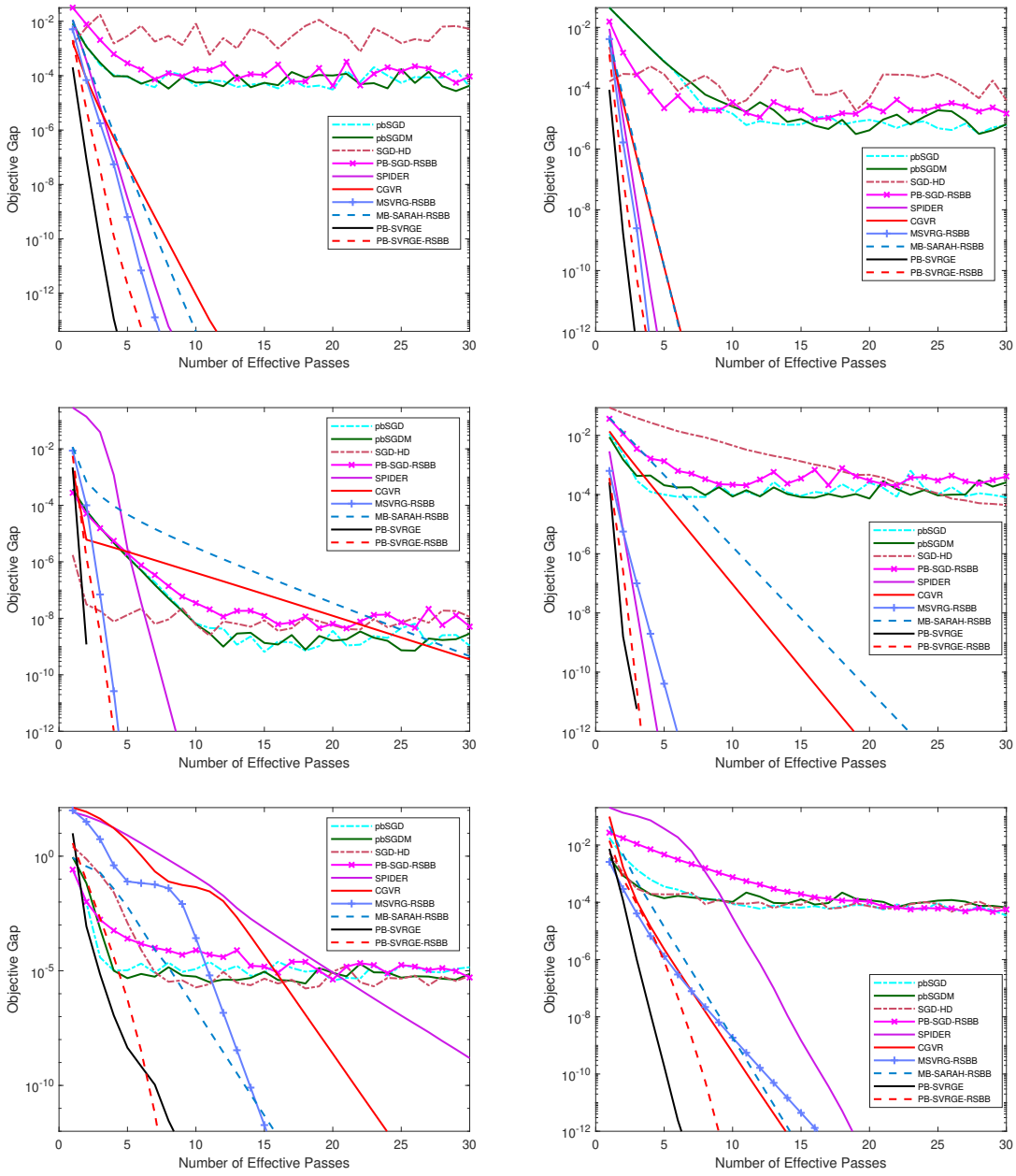

Figure 10: Comparison of different approaches on *a8a* (top left), *covtype* (top right), *ijcnn*1 (Middle left), *news*20.*binary* (Middle right), CIFAR-10 (bottom left), and MNIST (bottom right).

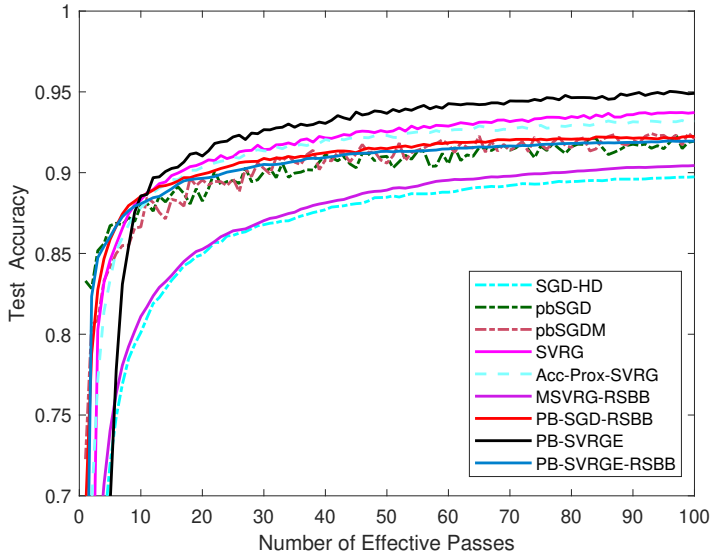

Figure 11: The numerical behavior of different approaches on MNIST data set for training NNs.

6. Conclusion

This paper explored the theoretical and empirical properties of PSO algorithms from the perspectives of variance reduction and adaptive learning rates. Concretely, we first proposed a variance-reduced PSO algorithm, PB-SVRGE (Algorithm 1), by incorporating SVRGE into the classical PSO algorithm. Moreover, we strictly analyzed the convergence behavior of PB-SVRGE under the non-convex assumption. Meanwhile, we showed that the computational complexity of PB-SVRGE (Algorithm 1) was superior than that of advanced stochastic optimization algorithms. Further, to bridge the gap between PSO and the learning rate, we utilized the idea of the BB-like technique to obtain an adaptive rule of updating learning rates for PSO algorithms. We first studied the performance of our PB-SVRGE algorithm with the BB-like technique, obtaining PB-SVRGE-RSBB (Algorithm 2). Then, we generalized such an adaptive learning rate to the classical PSO algorithm, obtaining PB-SGD-RSBB (Algorithm 3). Moreover, the theoretical behavior for PB-SGD-RSBB (Algorithm 3) and PB-SVRGE-RSBB (Algorithm 2) were also provided. We performed a serious of numerical experiments on various data sets to show the performance of the proposed algorithms and investigated their parameter sensitivity. We also displayed the superior behaviors of our proposed algorithms by comparing them with state-of-the-art stochastic optimization methods.

Acknowledgments

This work was supported by the China Postdoctoral Science Foundation under Grant 2019M663238. Also, this work was partially supported by Project Funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions.

References

Wael A Al-Qaq, Michael Devetsikiotis, and J-K Townsend. Stochastic gradient optimization of importance sampling for the efficient simulation of digital communication systems. *IEEE Transactions on Communications*, 43(12):2975–2985, 1995.

Jonathan Barzilai and Jonathan M Borwein. Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, 8(1):141–148, 1988.

Atilim Gunes Baydin, Robert Cornish, David Martinez Rubio, Mark W Schmidt, and Frank D Wood. Online learning rate adaptation with hypergradient descent. In *International Conference on Learning Representations*, 2018.

Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. signsgd: Compressed optimisation for non-convex problems. In *International Conference on Machine Learning*, pages 560–569. PMLR, 2018.

Antoine Bordes, Léon Bottou, and Patrick Gallinari. SGD-QN: Careful quasi-newton stochastic gradient descent. *Journal of Machine Learning Research*, 10(Jul):1737–1754, 2009.

Léon Bottou. Stochastic gradient descent tricks. In *Neural networks: Tricks of the trade*, pages 421–436. Springer, 2012.

Richard H Byrd, Gillian M Chin, Jorge Nocedal, and Yuchen Wu. Sample size selection in optimization methods for machine learning. *Mathematical programming*, 134(1):127–155, 2012.

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):1–27, 2011.

Yuqi Cui, Yifan Xu, Ruimin Peng, and Dongrui Wu. Layer normalization for tsk fuzzy system optimization in regression problems. *IEEE Transactions on Fuzzy Systems*, 2022.

Soham De, Abhay Yadav, David Jacobs, and Tom Goldstein. Automated inference with adaptive batches. In *International Conference on Artificial Intelligence and Statistics*, 2017.

Aaron Defazio, Justin Domke, et al. Finito: A faster, permutable incremental gradient method for big data problems. In *International Conference on Machine Learning*, pages 1125–1133. PMLR, 2014.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(7), 2011.

Cong Fang, Chris Junchi Li, Zhouchen Lin, and Tong Zhang. Spider: Near-optimal non-convex optimization via stochastic path-integrated differential estimator. In *Advances in Neural Information Processing Systems*, pages 689–699, 2018.

Benjamin Fehrman, Benjamin Gess, and Arnulf Jentzen. Convergence rates for the stochastic gradient descent method for non-convex objective functions. *Journal of Machine Learning Research*, 21:136, 2020.

Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

Luigi Grippo, Francesco Lampariello, and Stephano Lucidi. A nonmonotone line search technique for newtons method. *SIAM journal on Numerical Analysis*, 23(4):707–716, 1986.

Ezra Ip and Joseph M Kahn. Compensation of dispersion and nonlinear impairments using digital backpropagation. *Journal of Lightwave Technology*, 26(20):3416–3425, 2008.

XB Jin, XY Zhang, K Huang, and GG Geng. Stochastic conjugate gradient algorithm with variance reduction. *IEEE Transactions on Neural Networks and Learning Systems*, 30(5):1360, 2019.

Yujia Jin and Aaron Sidford. Principal component projection and regression in nearly linear time through asymmetric svrg. *Advances in Neural Information Processing Systems*, 32, 2019.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, pages 315–323, 2013.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representation (Poster)*, 2015.

Hoai An Le Thi, Hoang Phuc Hau Luu, Hoai Minh Le, and Tao Pham Dinh. Stochastic dca with variance reduction and applications in machine learning. *Journal of Machine Learning Research*, 23(206):1–44, 2022.

Lihua Lei, Cheng Ju, Jianbo Chen, and Michael I Jordan. Non-convex finite-sum optimization via SCSG methods. *Advances in Neural Information Processing Systems*, 30, 2017.

Mu Li, Tong Zhang, Yuqiang Chen, and Alexander J Smola. Efficient mini-batch training for stochastic optimization. In *ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 661–670, 2014.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. In *International Conference on Learning Representations*, 2019.

Sijia Liu, Pin-Yu Chen, Xiangyi Chen, and Mingyi Hong. signsgd via zeroth-order oracle. In *International Conference on Learning Representations*, 2018.

Nicolas Loizou and Peter Richtárik. Momentum and stochastic momentum for stochastic gradient, newton, proximal point and subspace descent methods. *Computational Optimization and Applications*, 77(3):653–710, 2020.

Nicolas Loizou, Sharan Vaswani, Issam Hadj Laradji, and Simon Lacoste-Julien. Stochastic polyak step-size for sgd: An adaptive learning rate for fast convergence. In *International Conference on Artificial Intelligence and Statistics*, pages 1306–1314. PMLR, 2021.

Aryan Mokhtari and Alejandro Ribeiro. Stochastic quasi-newton methods. *Proceedings of the IEEE*, 108(11):1906–1922, 2020.

Lam M Nguyen, Jie Liu, Katya Scheinberg, and Martin Takáč. SARAH: A novel method for machine learning problems using stochastic recursive gradient. In *International Conference on Machine Learning-Volume 70*, pages 2613–2621, 2017.

Lam M Nguyen, Quoc Tran-Dinh, Dzung T Phan, Phuong Ha Nguyen, and Marten van Dijk. A unified convergence analysis for shuffling-type gradient methods. *Journal of Machine Learning Research*, 22(207):1–44, 2021.

Atsushi Nitanda. Stochastic proximal gradient descent with acceleration techniques. In *Advances in Neural Information Processing Systems*, pages 1574–1582, 2014.

Nhan H Pham, Lam M Nguyen, Dzung T Phan, and Quoc Tran-Dinh. Proxsarah: An efficient algorithmic framework for stochastic composite nonconvex optimization. *Journal of Machine Learning Research*, 21(110):1–48, 2020.

Loucas Pillaud-Vivien, Alessandro Rudi, and Francis Bach. Exponential convergence of testing error for stochastic gradient methods. In *Conference on Learning Theory*, pages 250–296. PMLR, 2018.

Boris T Polyak. Some methods of speeding up the convergence of iteration methods. *Ussr computational mathematics and mathematical physics*, 4(5):1–17, 1964.

Mariana Prazeres and Adam M Oberman. Stochastic gradient descent with polyaks learning rate. *Journal of Scientific Computing*, 89(1):1–16, 2021.

Alexander Rakhlin, Ohad Shamir, and Karthik Sridharan. Making gradient descent optimal for strongly convex stochastic optimization. In *International Coference on International Conference on Machine Learning*, pages 1571–1578, 2012.

Sashank J Reddi, Ahmed Hefny, Suvrit Sra, Barnabas Poczos, and Alex Smola. Stochastic variance reduction for nonconvex optimization. In *International Conference on Machine Learning*, pages 314–323, 2016.

Nicolas L Roux, Mark Schmidt, and Francis R Bach. A stochastic gradient method with an exponential convergence rate for finite training sets. In *Advances in Neural Information Processing Systems*, pages 2663–2671, 2012.

Zhenhua Shi, Dongrui Wu, Chenfeng Guo, Changming Zhao, Yuqi Cui, and Fei-Yue Wang. FCM-RDpA: TSK fuzzy regression model construction using fuzzy c-means clustering, regularization, droprule, and powerball adabelief. *Information Sciences*, 574:490–504, 2021.

Krzysztof Sopyła and Paweł Drozda. Stochastic gradient descent with Barzilai–Borwein update step for SVM. *Information Sciences*, 316:218–233, 2015.

T Tieleman and G Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*, 2012.

T Tieleman and G Hinton. Divide the gradient by a running average of its recent magnitude. coursera: Neural networks for machine learning. *Technical Report.*, 2017.

Sharan Vaswani, Aaron Mishkin, Issam Laradji, Mark Schmidt, Gauthier Gidel, and Simon Lacoste-Julien. Painless stochastic gradient: Interpolation, line-search, and convergence rates. In *Advances in Neural Information Processing Systems*, pages 3727–3740, 2019.

Zhe Wang, Kaiyi Ji, Yi Zhou, Yingbin Liang, and Vahid Tarokh. Spiderboost and momentum: faster stochastic variance reduction algorithms. In *Advances in Neural Information Processing Systems*, pages 2406–2416, 2019.

Zhe Wang, Yi Zhou, Yingbin Liang, and Guanghui Lan. Cubic regularization with momentum for nonconvex optimization. In *Uncertainty in Artificial Intelligence*, pages 313–322. PMLR, 2020.

Fuchao Wei, Chenglong Bao, and Yang Liu. Stochastic anderson mixing for nonconvex stochastic optimization. In *Advances in Neural Information Processing Systems*, volume 34, pages 22995–23008, 2021.

Jie Xu, Wei Zhang, and Fei Wang. A (dp)ˆ 2sgd: Asynchronous decentralized parallel stochastic gradient descent with differential privacy. *IEEE transactions on pattern analysis and machine intelligence*, 2021.

Zhuang Yang. Fast automatic step size selection for zeroth-order nonconvex stochastic optimization. *Expert Systems with Applications*, 174:114749, 2021a.

Zhuang Yang. On the step size selection in variance-reduced algorithm for nonconvex optimization. *Expert Systems with Applications*, 169:114336, 2021b.

Zhuang Yang. Adaptive stochastic conjugate gradient for machine learning. *Expert Systems with Applications*, page 117719, 2022.

Zhuang Yang, Cheng Wang, Yu Zang, and Jonathan Li. Mini-batch algorithms with Barzilai–Borwein update step. *Neurocomputing*, 314:177–185, 2018a.

Zhuang Yang, Cheng Wang, Zhemin Zhang, and Jonathan Li. Random Barzilai–Borwein step size for mini-batch algorithms. *Engineering Applications of Artificial Intelligence*, 72:124–135, 2018b.

Yaxiang Yuan and Wenyu Sun. Optimization theory and methods, 1997.

Ye Yuan, Mu Li, Jun Liu, and Claire Tomlin. On the powerball method: Variants of descent methods for accelerated optimization. *IEEE Control Systems Letters*, 3(3):601–606, 2019.

Hai-Tao Zhang, Weigao Sun, Yuanzheng Li, Dongfei Fu, and Ye Yuan. A fast optimal power flow algorithm using powerball method. *IEEE Transactions on Industrial Informatics*, 16(11):6993–7003, 2019.

Shengjun Zhang and Colleen P Bailey. Accelerated zeroth-order algorithm for stochastic distributed non-convex optimization. In *American Control Conference (ACC)*, pages 4274–4279. IEEE, 2022.

Peilin Zhao and Tong Zhang. Stochastic optimization with importance sampling for regularized loss minimization. In *International Conference on Machine Learning*, pages 1–9. PMLR, 2015.

Beitong Zhou, Jun Liu, Weigao Sun, Ruijuan Chen, Claire J Tomlin, and Ye Yuan. pbSGD: Powered stochastic gradient descent methods for accelerated non-convex optimization. In *International Joint Conferences on Artificial Intelligence*, pages 3258–3266, 2020.

Dongruo Zhou, Pan Xu, and Quanquan Gu. Stochastic nested variance reduction for nonconvex optimization. In *Advances in Neural Information Processing Systems*, pages 3925–3936. Curran Associates Inc., 2018.