PAGE: A Simple and Optimal Probabilistic Gradient Estimator for Nonconvex Optimization

Zhize Li  
Hongyan Bao  
Xiangliang Zhang  
Peter Richtárik

King Abdullah University of Science and Technology (KAUST)

Abstract

In this paper, we propose a novel stochastic gradient estimator—ProbAbilistic Gradient Estimator (PAGE)—for nonconvex optimization. PAGE is easy to implement as it is designed via a small adjustment to vanilla SGD: in each iteration, PAGE uses the vanilla minibatch SGD update with probability $p$ or reuses the previous gradient with a small adjustment, at a much lower computational cost, with probability $1 - p$. We give a simple formula for the optimal choice of $p$. We prove tight lower bounds for nonconvex problems, which are of independent interest. Moreover, we prove matching upper bounds both in the finite-sum and online regimes, which establish that PAGE is an optimal method. Besides, we show that for nonconvex functions satisfying the Polyak-Łojasiewicz (PL) condition, PAGE can automatically switch to a faster linear convergence rate. Finally, we conduct several deep learning experiments (e.g., LeNet, VGG, ResNet) on real datasets in PyTorch, and the results demonstrate that PAGE not only converges much faster than SGD in training but also achieves the higher test accuracy, validating our theoretical results and confirming the practical superiority of PAGE.

1. Introduction

Nonconvex optimization is ubiquitous across many domains of machine learning, including robust regression, low rank matrix recovery, sparse recovery and supervised learning [13]. Driven by the applied success of deep neural networks [21], and the critical place nonconvex optimization plays in training them, research in nonconvex optimization has been undergoing a renaissance [6, 8, 9, 25, 28, 47].

1.1. The problem

Motivated by this development, we consider the general optimization problem

$$
\min_{x \in \mathbb{R}^d} f(x),
$$

where $f : \mathbb{R}^d \to \mathbb{R}$ is a differentiable and possibly nonconvex function. We are interested in functions having the finite-sum form

$$
f(x) := \frac{1}{n} \sum_{i=1}^{n} f_i(x),
$$

© Z. Li, H. Bao, X. Zhang & P. Richtárik.
where the functions  are also differentiable and possibly nonconvex. Form (2) captures the standard empirical risk minimization problems in machine learning [41]. Moreover, if the number of data samples  is very large or even infinite, e.g., in the online/streaming case, then  usually is modeled via the online form

\[
    f(x) := \mathbb{E}_{\zeta \sim D}[ F(x, \zeta) ],
\]

which we also consider in this work. For notational convenience, we adopt the notation of the finite-sum form (2) in the descriptions and algorithms in the rest of this paper. However, our results apply to the online form (3) as well by letting \( f_i(x) := F(x, \zeta_i) \) and treating  as a very large number or even infinite.

1.2. Gradient complexity

To measure the efficiency of algorithms for solving the nonconvex optimization problem (1), it is standard to bound the number of stochastic gradient computations needed to find a solution of suitable characteristics. In this paper we use the standard term gradient complexity to describe such bounds. In particular, our goal will be to find a (possibly random) point \( \hat{x} \in \mathbb{R}^d \) such that

\[
    \mathbb{E} \| \nabla f(\hat{x}) \| \leq \epsilon,
\]

where the expectation is with respect to the randomness inherent in the algorithm. We use the term \( \epsilon \)-approximate solution to refer to such a point \( \hat{x} \).

Two of the most classical gradient complexity results for solving problem (1) are those for gradient descent (GD) and stochastic gradient descent (SGD). In particular, the gradient complexity of GD is \( O\left( \frac{n}{\epsilon^2} \right) \) in this nonconvex regime, and assuming that the stochastic gradient satisfies a (uniform) bounded variance assumption (Assumption 1), the gradient complexity of SGD is \( O\left( \frac{1}{\epsilon^4} \right) \). Note that although SGD has a worse dependence on \( \epsilon \), it typically only needs to compute a constant minibatch of stochastic gradients in each iteration instead of the full batch (i.e.,  stochastic gradients) used in GD. Hence, SGD is better than GD if the number of data samples \( n \) is very large or the error tolerance \( \epsilon \) is not very small.

There has been extensive research in designing gradient-type methods with an improved dependence on \( n \) and/or \( \epsilon \) [8, 9, 32, 34]. In particular, the SVRG method of Johnson and Zhang [14], the SAGA method of Defazio et al. [5] and the SARAH method of Nguyen et al. [35] are representatives of what is by now a large class of variance-reduced methods, which have played a particularly important role in this effort. However, the analyses in these papers focused on the convex regime. Furthermore, several accelerated (momentum) methods have been designed as well [1, 18–20, 27, 30, 31, 33], with or without variance reduction. There are also some lower bounds given by [44, 45].

Coming back to problem (1) in the nonconvex regime studied in this paper, interesting recent development starts with the work of Reddi et al. [40], and Allen-Zhu and Hazan [2], who have concurrently shown that if \( f \) has the finite-sum form (2), a suitably designed minibatch version of SVRG enjoys the gradient complexity \( O( n + n^{2/3} / \epsilon^2 ) \), which is an improvement on the \( O( n / \epsilon^2 ) \) gradient complexity of GD. Subsequently, other variants of SVRG were shown to posses the same improved rate, including those developed by [7, 11, 23, 26, 39]. More recently, Fang et al. [6] proposed the SPIDER method, and Zhou et al. [47] proposed the SNVRG method, both of which improve the gradient complexity further to \( O( n + \sqrt{n} / \epsilon^2 ) \). Further variants of the SARAH method (e.g., [12, 24, 25, 37, 43]) which also achieve the same \( O( n + \sqrt{n} / \epsilon^2 ) \) gradient complexity have been developed. Particularly, Horváth et al. [12] proposed the Geom-SARAH to automatically adapt
different parameter settings via dynamic batch size and epoch length. Also there are some lower bounds given by [3, 6, 46]. See Table 1 for an overview of results.

2. Our Contributions

As we show in through this work, despite enormous effort by the community to design efficient methods for solving (1) in the nonconvex regime, there is still a considerable gap in our understanding. First, while optimal methods for (1) in the finite-sum regime exist (e.g., SPIDER [6], SpiderBoost [43], SARAH [37], SSRGD [25]), the known lower bound \( \Omega(\sqrt{n}/\epsilon^2) \) [6] used to establish their optimality works only for \( n \leq O(1/\epsilon^4) \), i.e., in the small data regime (see Table 1). Moreover, these methods are unnecessarily complicated, often with a double loop structure, and reliance on several hyperparameters. Besides, there is also no tight lower bound to show the optimality of optimal methods in the online regime.

Table 1: Gradient complexity for finding \( \hat{x} \) satisfying \( \mathbb{E}\left\| \nabla f(\hat{x}) \right\| \leq \epsilon \) in nonconvex problems

<table>
<thead>
<tr>
<th>Problem</th>
<th>Assumption</th>
<th>Algorithm or Lower Bound</th>
<th>Gradient complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finite-sum (2)</td>
<td>Asp. 2</td>
<td>GD [34]</td>
<td>( O(\frac{n}{\epsilon^2}) )</td>
</tr>
<tr>
<td>Finite-sum (2)</td>
<td>Asp. 2</td>
<td>SVRG [2, 40], SCSG [23], SVRG+ [26]</td>
<td>( O(n + \frac{n^{2/3}}{\epsilon^2}) )</td>
</tr>
<tr>
<td>Finite-sum (2)</td>
<td>Asp. 2</td>
<td>SNVGR [47], Geom-SARAH [12]</td>
<td>( \tilde{O}(n + \frac{\sqrt{n}}{\epsilon^2}) )</td>
</tr>
<tr>
<td>Finite-sum (2)</td>
<td>Asp. 2</td>
<td>SPIDER [6], SpiderBoost [43], SARAH [37], SSRGD [25]</td>
<td>( O(n + \frac{\sqrt{n}}{\epsilon^2}) )</td>
</tr>
<tr>
<td>Finite-sum (2)</td>
<td>Asp. 2</td>
<td>PAGE (this paper)</td>
<td>( O(n + \frac{\sqrt{n}}{\epsilon^2}) )</td>
</tr>
<tr>
<td>Finite-sum (2)</td>
<td>Asp. 2</td>
<td>Lower bound [6]</td>
<td>( \Omega(\frac{\sqrt{n}}{\epsilon^2}) ) if ( n \leq O(\frac{1}{\epsilon^4}) )</td>
</tr>
<tr>
<td>Finite-sum (2)</td>
<td>Asp. 2</td>
<td>Lower bound (this paper)</td>
<td>( \Omega(n + \frac{\sqrt{n}}{\epsilon^2}) )</td>
</tr>
<tr>
<td>Finite-sum (2)</td>
<td>Asp. 2 and 3</td>
<td>PAGE (this paper)</td>
<td>( O((n + \sqrt{n}k) \log \frac{1}{\epsilon}) )</td>
</tr>
<tr>
<td>Online (3)(^2)</td>
<td>Asp. 1 and 2</td>
<td>SGD [9, 15, 28]</td>
<td>( O(\frac{\sigma^2}{\epsilon^2}) )</td>
</tr>
<tr>
<td>Online (3)</td>
<td>Asp. 1 and 2</td>
<td>SCSG [23], SVRG+ [26]</td>
<td>( O(b + \frac{\sqrt{n}}{\epsilon^2}) )</td>
</tr>
<tr>
<td>Online (3)</td>
<td>Asp. 1 and 2</td>
<td>SNVGR [47], Geom-SARAH [12]</td>
<td>( \tilde{O}(b + \frac{\sqrt{n}}{\epsilon^2}) )</td>
</tr>
<tr>
<td>Online (3)</td>
<td>Asp. 1 and 2</td>
<td>SPIDER [6], SpiderBoost [43], SARAH [37], SSRGD [25]</td>
<td>( O(b + \frac{\sqrt{n}}{\epsilon^2}) )</td>
</tr>
<tr>
<td>Online (3)</td>
<td>Asp. 1 and 2</td>
<td>PAGE (this paper)</td>
<td>( O(b + \frac{\sqrt{n}}{\epsilon^2}) )</td>
</tr>
<tr>
<td>Online (3)</td>
<td>Asp. 1 and 2</td>
<td>Lower bound (this paper)</td>
<td>( \Omega(b + \frac{\sqrt{n}}{\epsilon^2}) )</td>
</tr>
<tr>
<td>Online (3)</td>
<td>Asp. 1, 2 and 3</td>
<td>PAGE (this paper)</td>
<td>( O \left( (b + \sqrt{n}k) \log \frac{1}{\epsilon} \right) )</td>
</tr>
</tbody>
</table>

1. Note that PAGE can automatically switch to a faster linear convergence \( O(\cdot \log \frac{1}{\epsilon}) \) instead of sublinear \( O(\frac{1}{\epsilon^2}) \) by exploiting the local structure of the objective function via the PL condition (Assumption 3).
2. Note that we refer the online problem (3) as the finite-sum problem (2) with large or infinite \( n \) as discussed in the introduction Section 1.1. In this online case, the full gradient may not be available (e.g., if \( n \) is infinite), thus the bounded variance of stochastic gradient Assumption 1 is needed in this case.
3. In the online case, \( b := \min\left\{ \frac{\sigma^2}{\epsilon^2}, n \right\} \), and \( \sigma \) is defined in Assumption 1. If \( n \) is very large, i.e., \( b := \min\left\{ \frac{\sigma^2}{\epsilon^2}, n \right\} = \frac{\sigma^2}{\epsilon^2} \), then \( O(b + \frac{\sqrt{n}}{\epsilon^2}) = O(\frac{\sigma^2}{\epsilon^2} + \frac{\sqrt{n}}{\epsilon^2}) \) is better than the rate \( O(\frac{\sigma^2}{\epsilon^2}) \) of SGD by a factor of \( \frac{1}{\epsilon^2} \) or \( \frac{\sqrt{n}}{\epsilon^2} \).
In this paper, we resolve the above issues by designing a novel ProbAbilistic Gradient Estimator (PAGE) described in Algorithm 1 for achieving optimal convergence results in nonconvex optimization. Moreover, PAGE is very simple and easy to implement. In each iteration, PAGE uses minibatch SGD update with probability \( p_t \), or reuses the previous gradient with a small adjustment (at a low computational cost) with probability \( 1 - p_t \) (see Line 4 of Algorithm 1). We would like to highlight the following points:

- We prove that PAGE achieves the optimal rates for both nonconvex finite-sum problem (2) and online problem (3) (see Corollaries 2 and 4). We also provide tight lower bounds for these two problems to close the gap and show the optimality of PAGE (see Theorem 2 and Corollary 5). Our lower bounds are inspired and based by recent work [3, 6]. See Table 1 for a detailed comparison with previous work.

- Moreover, we show that PAGE can automatically switch to a faster linear convergence \( O(\frac{\cdot \log \frac{1}{\epsilon}}{\epsilon}) \) by exploiting the local structure of the objective function, via the PL condition (Assumption 3), although the objective function \( f \) is globally nonconvex. See the middle and the last row of Table 1. For example, PAGE automatically switches from the sublinear rate \( O(n + \sqrt{n}/\epsilon^2) \) to the faster linear rate \( O((n + \sqrt{n}c) \log \frac{1}{\epsilon}) \) for nonconvex finite-sum problem (2) (see the Remark after Corollary 6).

- PAGE is easy to implement via a small adjustment to vanilla minibatch SGD, and takes a lower computational cost than SGD (i.e., \( p = 1 \) in PAGE) since \( b' < b \). We conduct several deep learning experiments (e.g., LeNet, VGG, ResNet) on real datasets in PyTorch showing that PAGE indeed not only converges much faster than SGD in training but also achieves higher test accuracy. This validates our theoretical results and confirms the practical superiority of PAGE.

### 2.1. The PAGE gradient estimator

In this section, we describe PAGE, an SGD variant employing a new, simple and optimal gradient estimator (see Algorithm 1). In particular, PAGE was inspired by algorithmic design elements coming from methods such as SARAH [35], SPIDER [6], SSRGD [25] (usage of a recursive estimator), and L-SVRG [16] and SAGD [4] (probabilistic switching between two estimators to avoid a double loop structure). PAGE with constant probability \( p \) can be reduced to an equivalent form of the double loop algorithm with geometric distribution Geom-SARAH [12], but our single-loop PAGE is more flexible and also leads to simpler and better analysis.

**Algorithm 1** ProbAbilistic Gradient Estimator (PAGE)

<table>
<thead>
<tr>
<th>Input:</th>
<th>initial point ( x^0 ), stepsize ( \eta ), minibatch size ( b ), ( b' &lt; b ), probability ( {p_t} \in (0, 1] )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1:</td>
<td>( g^0 = \frac{1}{b} \sum_{i \in I} \nabla f_i(x^0) ) // ( I ) denotes random minibatch samples with (</td>
</tr>
<tr>
<td>2:</td>
<td>for ( t = 0, 1, 2, \ldots ) do</td>
</tr>
<tr>
<td>3:</td>
<td>( x^{t+1} = x^t - \eta g^t )</td>
</tr>
<tr>
<td>4:</td>
<td>( g^{t+1} = \begin{cases} \frac{1}{b} \sum_{i \in I} \nabla f_i(x^{t+1}) &amp; \text{with probability } p_t \ g^t + \frac{1}{b'} \sum_{i \in I'} (\nabla f_i(x^{t+1}) - \nabla f_i(x^t)) &amp; \text{with probability } 1 - p_t \end{cases} )</td>
</tr>
<tr>
<td>5:</td>
<td>end for</td>
</tr>
<tr>
<td>Output:</td>
<td>( \hat{x}<em>T ) chosen uniformly from ( {x^t}</em>{t \in [T]} )</td>
</tr>
</tbody>
</table>

In iteration \( t \), the gradient estimator \( g^{t+1} \) of PAGE is defined in Line 4 of Algorithm 1, which indicates that PAGE uses the vanilla minibatch SGD update with probability \( p_t \), and reuses the

---

4. All theorems and corollaries and their proofs can be found in the full version of this paper [29].
previous gradient \( g^t \) with a small adjustment (which lowers the computational cost since \( b' \ll b \)) with probability \( 1 - p_t \). In particular, the \( p_t \equiv 1 \) case reduces to vanilla minibatch SGD, and to GD if we further set the minibatch size to \( b = n \). We give a simple formula for the optimal choice of \( p_t \), i.e., \( p_t \equiv \frac{b'}{b + b'} \) is enough for PAGE to obtain the optimal convergence rates. More details can be found in the convergence theorems and corollaries in the full version of this paper [29].

3. Assumptions

**Assumption 1 (Bounded variance)** The stochastic gradient has bounded variance if

\[
\exists \sigma > 0, \text{ such that } \mathbb{E}_i[\|\nabla f_i(x) - \nabla f(x)\|^2] \leq \sigma^2, \quad \forall x \in \mathbb{R}^d. \tag{4}
\]

**Assumption 2 (Average L-smoothness)** A function \( f : \mathbb{R}^d \to \mathbb{R} \) is average \( L \)-smooth if

\[
\exists L > 0, \text{ such that } \mathbb{E}_i[\|\nabla f_i(x) - \nabla f_i(y)\|^2] \leq L^2\|x - y\|^2, \quad \forall x, y \in \mathbb{R}^d. \tag{5}
\]

Moreover, we also prove faster linear convergence rates for nonconvex functions under the Polyak-Łojasiewicz (PL) condition [38].

**Assumption 3 (PL condition)** A function \( f : \mathbb{R}^d \to \mathbb{R} \) satisfies PL condition if

\[
\exists \mu > 0, \text{ such that } \|\nabla f(x)\|^2 \geq 2\mu(f(x) - f^*), \quad \forall x \in \mathbb{R}^d. \tag{6}
\]

4. Experiments

We conduct several deep learning experiments for multi-class image classification. Concretely, we compare our PAGE algorithm with vanilla SGD by running standard LeNet [22], VGG [42] and ResNet [10] models on MNIST [22] and CIFAR-10 [17] datasets. We implement the algorithms in PyTorch [36] and run the experiments on several NVIDIA Tesla V100 GPUs.

According to the update form in PAGE (see Line 4 of Algorithm 1), PAGE enjoys a lower computational cost than vanilla minibatch SGD (i.e., \( p = 1 \) in PAGE) since \( b' < b \). Thus, in the experiments we want to show how the performance of PAGE compares with vanilla minibatch SGD under different minibatch sizes \( b \). Note that we do not tune the parameters for PAGE, i.e., we set \( b' = \sqrt{b} \) and \( p = \frac{b'}{b + b'} = \frac{\sqrt{b}}{b + \sqrt{b}} \) according to our theoretical results (see e.g., Corollary 2 and 4).

Concretely, in Figure 1, we choose standard minibatch \( b = 64 \) and \( b = 256 \) for both PAGE and vanilla minibatch SGD for MNIST experiments. In Figure 2, we choose \( b = 256 \) and \( b = 512 \) for CIFAR-10 experiments. The first row of Figures 1 and 2 denotes the training loss with respect to the gradient computations, and the second row denotes the test accuracy with respect to the gradient computations. Both Figures 1 and 2 demonstrate that PAGE not only converges much faster than SGD in training but also achieves the higher test accuracy (which is typically very important in practice, e.g., lead to a better model). Moreover, the performance gap between PAGE and SGD is larger when the minibatch size \( b \) is larger (i.e, gap between solid lines in Figures 1a, 1b, 2a, 2b), which is consistent with the update form of PAGE, i.e, it reuses the previous gradient with a small adjustment (lower computational cost \( b' = \sqrt{b} \) instead of \( b \)) with probability \( 1 - p_t \). The experimental results validate our theoretical results and confirm the practical superiority of PAGE.
Figure 1: LeNet and ResNet18 on MNIST dataset

(1a) Different minibatch size $b$  
(1b) Different minibatch size $b$  
(1c) Different neural networks

(2a) Different minibatch size $b$  
(2b) Different minibatch size $b$  
(2c) Different neural networks

Figure 2: VGG16 and ResNet18 on CIFAR-10 dataset

References


