Noname manuscript No.
(will be inserted by the editor)

Finite-Sum Smooth Optimization with SARAH

Lam M. Nguyen* · **Marten van Dijk** · **Dzung T. Phan** · **Phuong Ha Nguyen** · **Tsui-Wei Weng** · **Jayant R. Kalagnanam**

Received: date / Accepted: date

Abstract We introduce NC-SARAH for non-convex optimization as a *practical* modified version of the original SARAH algorithm that was developed for convex optimization. NC-SARAH is the first to achieve two crucial performance properties at the same time – allowing flexible minibatch sizes and large step sizes to achieve fast convergence in practice as verified by experiments. NC-SARAH has a close to optimal asymptotic convergence rate equal to existing prior variants of SARAH called SPIDER and SpiderBoost that either use an order of magnitude smaller step size or a fixed minibatch size. For convex optimization, we propose SARAH++ with sublinear convergence for general convex and linear convergence for strongly convex problems; and we provide a practical version for which numerical experiments on various datasets show an improved performance.

Lam M. Nguyen, Dzung T. Phan, Jayant R. Kalagnanam
IBM Research, Thomas J. Watson Research Center, Yorktown Heights, NY, USA
E-mail: LamNguyen.MLTD@ibm.com, phandu@us.ibm.com, jayant@us.ibm.com

Marten van Dijk
CWI, Computer Security Group, Amsterdam, The Netherlands
E-mail: mevd@cwi.nl

Phuong Ha Nguyen
eBay Inc., San Jose, CA, USA
E-mail: phuongha.ntu@gmail.com

Tsui-Wei Weng
University of California San Diego, La Jolla, CA, USA
E-mail: lweng@ucsd.edu

* Corresponding author

1 Introduction

We are interested in solving the *finite-sum* minimization problem

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

where each f_i, $i\in[n]\overset{\text{def}}{=}\{1,\ldots,n\}$, has a Lipschitz continuous gradient. Throughout the paper, we consider the case where F has a finite lower bound F^*. We would like to attain an ϵ-*accurate solution* satisfying $\mathbb{E}[\|\nabla F(\tilde{w})\|^2]\leq\epsilon$ for the outputted approximation $\tilde{w}$.

Problems of form (1) cover a wide range of convex and non-convex problems in machine learning applications including but not limited to logistic regression, neural networks, multi-kernel learning, etc. In many of these applications, the number of component functions n is very large, which makes the classical Gradient Descent (GD) method less efficient since it requires to compute a full gradient many times. Stochastic Gradient Descent (SGD), originally proposed by [24], has been widely used to solve (1) thanks to its scalability and efficiency in dealing with large-scale problems. SGD and its variants have gained a lot of attention in the machine learning community (see e.g. [7,10,4,17,20]). In recent years, a large number of improved variants of stochastic gradient algorithms called variance reduction methods have been proposed to obtain better computational cost compared to GD, in particular, SAG/SAGA [26,6], SDCA [27], MISO [15], SVRG/S2GD [9,11], SARAH [18], etc. These methods were first analyzed for strongly convex problems of form (1). Due to recent interest in deep neural networks, *non-convex* problems of form (1) have been studied and analyzed by considering a number of different approaches including many variants of variance reduction techniques (see e.g. [23,12,1,2,8], etc.)

SARAH is the variance reduction algorithm which was originally proposed in [18] in the *convex case*. In this paper, we introduce a modification to SARAH in Algorithm 1, called NC-SARAH, for the non-convex case. SARAH's as well as NC-SARAH's iterations are divided into an outer loop where a full gradient is computed and an inner loop where only one stochastic gradient is computed. We use upper index (s) to indicate the s-th outer loop and lower index t to indicate the t-th iteration in the inner loop. The key update rule, which is called the SARAH update [18] for the inner loop, is

$$v_t^{(s)}=\nabla f_{i_t}(w_t^{(s)})-\nabla f_{i_t}(w_{t-1}^{(s)})+v_{t-1}^{(s)}, \tag{2}$$

where i_t is chosen uniformly at random in $[n]$. The computed $v_t^{(s)}$ is used to update $w_{t+1}^{(s)}=w_t^{(s)}-\eta v_t^{(s)}$. In NC-SARAH, after m iterations in the inner loop, the outer loop remembers the last computed $w_{m+1}^{(s)}$ and starts its loop anew – first with a full gradient computation before again entering the inner loop with updates (2). Instead of remembering $\tilde{w}_s=w_{m+1}^{(s)}$ for the next outer loop, the original SARAH algorithm in [18] uses $\tilde{w}_s=w_t^{(s)}$ with t chosen uniformly at random from $\{0,1,\ldots,m\}$; the authors of [18] chose to do this in order to being able to analyze the convergence rate for a single outer loop.

Algorithm 1 NC-SARAH

Parameters: the learning rate $\eta > 0$, the inner loop size m, and the outer loop size S
Initialize: $\tilde{w}_0$
Iterate:
for $s = 1, 2, \ldots, S$ **do**
  $w_0^{(s)} = \tilde{w}_{s-1}$
  $v_0^{(s)} = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(w_0^{(s)})$
  $w_1^{(s)} = w_0^{(s)} - \eta v_0^{(s)}$
  Iterate:
  for $t = 1, \ldots, m$ **do**
    Sample i_t uniformly at random from $[n]$
    $v_t^{(s)} = \nabla f_{i_t}(w_t^{(s)}) - \nabla f_{i_t}(w_{t-1}^{(s)}) + v_{t-1}^{(s)}$
    $w_{t+1}^{(s)} = w_t^{(s)} - \eta v_t^{(s)}$
  end for
  Set $\tilde{w}_s = w_{m+1}^{(s)}$
end for

We notice that in [19] SARAH was extended to deal with *mini-batch* updates by, instead of choosing a single sample i_t in (2), we choose b samples uniformly at random from $[n]$ for updating v_t in the inner loop. This gives a SARAH update rule for mini-batches:

$$v_t^{(s)} = \frac{1}{b}\sum_{i \in I_t} [\nabla f_i(w_t^{(s)}) - \nabla f_i(w_{t-1}^{(s)})] + v_{t-1}^{(s)}, \tag{3}$$

where we choose a mini-batch $I_t \subseteq [n]$ of size b uniformly at random at each iteration of the inner loop. NC-SARAH in Algorithm 1 is for the single batch case and if we replace the update rule in the inner loop by (3) we get NC-SARAH for the mini-batch case. SARAH in [19] for mini-batches was analyzed for the non-convex case for only a single outer loop giving a total complexity of $\mathcal{O}(n + \frac{L^2}{\epsilon^2})$, where L is the Lipschitz constant of the gradients. With our modification to $\tilde{w}_s$ in NC-SARAH we are able to provide a "multiple outer loop" analysis for the non-convex case for single batches and mini-batches.

SPIDER [8], a recent variant of SARAH for the non-convex case, is the first work that achieves the best known total[1] complexity of $\mathcal{O}\left(n + L\sqrt{n}/\epsilon\right)$ for the non-convex case. Its complexity matches the lower-bound worst case complexity of $\mathcal{O}\left(\sqrt{n}/\epsilon\right)$ in [8] up to a constant factor when $n \leq \mathcal{O}(\epsilon^{-2})$. Another variant of SARAH [28] provides an improved version of SPIDER called SpiderBoost which allows a larger learning rate but restricting on the choice of the mini-batch size. Both SPIDER and SpiderBoost use the SARAH update rule (2) as originally proposed in [18] and use the mini-batch version of the update rule (3) in [19]. SPIDER and SpiderBoost do not divide into an outer loop and inner loop like SARAH, although SPIDER and SpiderBoost do similarly perform a full gradient update after a certain fixed number of iterations.

The drawback of SPIDER is the utilization of a small learning rate which depends on ϵ, but it offers flexibility in the range of mini-batch sizes for the inner loop $[1, \sqrt{n}]$.

[1] Measured as the total number of gradient computations needed to achieve an ϵ-accurate solution.

SpiderBoost has a larger stepsize independent on ϵ, which gives a big practical improvement for solving real applications. However, one needs to fix the mini-batch size of $\sqrt{n}$ for SpiderBoost, which limits the design space to such mini-batch sizes. Besides achieving the state-of-the-art asymptotic total complexity $\mathcal{O}\left(n + L\sqrt{n}/\epsilon\right)$ like SPIDER and SpiderBoost, our proposed variant of NC-SARAH mitigates at the same time both the learning rate limitation of SPIDER and the mini-batch limitation of SpiderBoost. In fact our learning rate is higher than those of SPIDER as well as SpiderBoost, and our mini-batch size for the inner loop can be freely selected from $[1, \sqrt{n}]$ (see Section 3 for more detail).

Contributions: We summarize our key contributions as follows.

1. **Smooth Non-Convex**.
   - We provide a new convergence analysis for a new variant of the SARAH algorithm (NC-SARAH) for non-convex problems. We show that NC-SARAH achieves the state-of-the-art total complexity[2] for finding a first-order stationary point in the non-convex case based on **only** the average smooth assumption; see Theorem 1 and Corollary 2 for the single batch case and Theorem 2 and Corollary 3 for the mini-batch case. We notice that our convergence analysis framework is simple and intuitive (Lemma 2 and Theorem 1).
   - We rigorously show that, given a fixed mini-batch size for the inner loop and given a fixed number of inner loop iterations, NC-SARAH can adopt an *order of magnitude larger learning rate* compared to SPIDER (Corollary 5) and a *strictly larger learning rate* compared to SpiderBoost (Corollary 6). Numerical experiments show how NC-SARAH outperforms both SPIDER and SpiderBoost (Section 5.1).
   - NC-SARAH allows *a range of mini-batch sizes* for the inner loop similar to SPIDER (Section 3). In this sense NC-SARAH adopts the advantage of SPIDER and, unlike SpiderBoost, does not need to give up on the flexibility of choosing mini-batch sizes in order to achieve practical large learning rates. Numerical experiments show that a flexible mini-batch size improves performance – a mini-batch size of about $n^{0.1} - n^{0.2}$ rather than the fixed mini-batch size of $n^{0.5}$ in SpiderBoost leads to best performance in our case study (Section 5.1).
2. **Smooth Convex**. In order to complete the picture, we study SARAH+ [18] which was designed as a variant of SARAH for convex optimization. SARAH+ provides a stopping criteria for the inner loop and shows the efficiency over SARAH. SARAH+ suggests to empirically choose parameter without theoretical guarantee. We propose a novel variant of SARAH+ called *SARAH++*. Here, we study the *iteration complexity* measured by the total number of iterations (which counts one full gradient computation as adding one iteration to the complexity) – and leave an analysis of the total complexity as an open problem. For SARAH++, we show a sublinear convergence rate in the general convex case (Theorem 3) and a linear convergence rate in the strongly convex case (Theorem 4). SARAH itself may already lead to good convergence and there may no need to introduce

[2] State-of-the-art complexity matches the lower-bound worst case complexity of $\Omega\left(n + \sqrt{n}/\epsilon\right)$ in [13].

SARAH++; in numerical experiments we show the advantage of SARAH++ over SARAH. We further propose a practical version called *SARAH Adaptive* which improves the performance of SARAH and SARAH++ for convex problems – numerical experiments on various data sets show good overall performance.

3. **Generalized Gradient Descent**. For the convergence analysis of NC-SARAH for the non-convex case and SARAH++ for the convex case, we show that the analysis generalizes the total complexity of Gradient Descent (Remarks 1, 2, and 3), i.e., the analysis reproduces known total complexity results of GD. Up to the best of our knowledge, this is the first variance reduction method having this property.

1.1 Related Work

Table 1: Comparison of results on the total complexity for smooth non-convex optimization

Method	Total Complexity	Additional assumption
GD [16]	$\mathcal{O}\left(\frac{n}{\epsilon}\right)$	None
SVRG [23]	$\mathcal{O}\left(n+\frac{n^{2/3}}{\epsilon}\right)$	None
SCSG [12]	$\mathcal{O}\left(\left(\frac{\sigma}{\epsilon}\wedge n\right)+\frac{1}{\epsilon}\left(\frac{\sigma}{\epsilon}\wedge n\right)^{2/3}\right)$	Bounded variance
	$\mathcal{O}\left(n+\frac{n^{2/3}}{\epsilon}\right)$	None ($\sigma\to\infty$)
SNVRG [29]	$\mathcal{O}\left(\log^3\left(\frac{\sigma}{\epsilon}\wedge n\right)\left[\left(\frac{\sigma}{\epsilon}\wedge n\right)+\frac{1}{\epsilon}\left(\frac{\sigma}{\epsilon}\wedge n\right)^{1/2}\right]\right)$	Bounded variance
	$\mathcal{O}\left(\log^3(n)\left(n+\frac{\sqrt{n}}{\epsilon}\right)\right)$	None ($\sigma\to\infty$)
SPIDER [8]	$\mathcal{O}\left(n+\frac{\sqrt{n}}{\epsilon}\right)$	None
SpiderBoost [28]	$\mathcal{O}\left(n+\frac{\sqrt{n}}{\epsilon}\right)$	None
NC-SARAH (this paper)	$\mathcal{O}\left(n+\frac{\sqrt{n}}{\epsilon}\right)$	**None**

Table 1[3] shows the comparison of results on the total complexity for smooth non-convex optimization. (a) Each of the complexities in Table 1 also depends on the Lipschitz constant L, however, since we consider smooth optimization, it is custom to assume/design $L=\mathcal{O}(1)$ and we therefore ignore the dependency on L in the complexity results. (b) Although many algorithms have appeared during the past few years, we only compare algorithms having a convergence result which only supposes the smooth assumption. (c) Among algorithms with convergence results that only suppose the smooth assumption, Table 1 only mentions recent state-of-the-art results. (d) Although the bounded variance assumption $\mathbb{E}[\|\nabla f_i(w)-\nabla F(w)\|^2]\leq\sigma^2$ is acceptable in many existing literature, this additional assumption limits the applicability of these convergence results since it adds dependence on σ which can be arbitrarily large. For fair comparison with convergence analysis without the bounded variance assumption, σ must be set to go to infinity – and this is what is mentioned in Table 1. As an example, from Table 1 we observe that SCSG has an advantage over SVRG only if $\sigma=\mathcal{O}(1)$ but, theoretically, by removing the bounded variance assumption, it has the same total complexity as SVRG if $\sigma\to\infty$.

[3] $a\wedge b$ is defined as $\min\{a,b\}$ and $a\vee b$ is defined as $\max\{a,b\}$

Table 2: Comparison properties among SPIDER, SpiderBoost, and NC-SARAH

Method	Complexity	Mini-batch size b and Number of inner loop iterations m	Learning Rate
SPIDER [8]	$\mathcal{O}\left(n + \frac{\sqrt{n}}{\epsilon}\right)$	$b = n^{1/2-\gamma}$ and $m = n^{1/2+\gamma}$ $\gamma \in [0, 1/2]$	Dependent on ϵ
SpiderBoost [28]	$\mathcal{O}\left(n + \frac{\sqrt{n}}{\epsilon}\right)$	$b = n^{1/2}$ and $m = n^{1/2}$	Independent on ϵ
NC-SARAH (this paper)	$\mathcal{O}\left(n + \frac{\sqrt{n}}{\epsilon}\right)$	$b = n^{1/2-\gamma}$ and $m = n^{1/2+\gamma}$ $\gamma \in [0, 1/2]$	**Independent on ϵ**

From Table 1, we observe that NC-SARAH, SPIDER and SpiderBoost achieve the total complexity of $\mathcal{O}\left(n + \sqrt{n}/\epsilon\right)$ and dominate the complexity of all other algorithms. Indeed, its complexity matches the lower-bound worst case complexity of $\Omega\left(n + \sqrt{n}/\epsilon\right)$ in [13]. We note that SPIDER and SpiderBoost can easily be rewritten by using an inner loop and outer loop algorithm description similar to SARAH (see also Algorithm 1). For consistency, we will use the term "inner loop" to indicate where the SARAH update rule is used in these three algorithms. The advantages of NC-SARAH over SPIDER and SpiderBoost, respectively, are shown in Table 2. Both SPIDER and NC-SARAH allow a mini-batch size for the update rule in the inner loop in $b \in [1, \sqrt{n}]$.[4] SpiderBoost is restricted in choosing a mini-batch size $b = \sqrt{n}$ for the inner loop while NC-SARAH like SPIDER has more choices. Our experimental results confirm that the choice of $b = \sqrt{n}$ and $m = \sqrt{n}$ of SpiderBoost is not the best choice (see Section 5.1). NC-SARAH outperforms SPIDER in that it can choose a much larger learning rate for the same mini-batch size b and number of inner loop iterations m. This is because the choice of NC-SARAH's learning rate does not depend on ϵ while SPIDER does; the smaller learning rate of SPIDER makes it converge slowly to small ϵ-accurate solution. Even though the learning rate of SpiderBoost also does not depend on ϵ, we show that for the same mini-batch size $b = \sqrt{n}$ a number of inner loop iterations $m = \sqrt{n}$ NC-SARAH can still choose a larger learning rate.

The more general settings have been considered in the existing literature (e.g., ProxSARAH [22]). However, by taking the benefit of our special case, we are able to show some advantages of our NC-SARAH as follows. We can quantify and show that NC-SARAH has clear advantages over SPIDER and SpiderBoost in both theory (see Section 3) and practice (see Section 5.1). It is currently unclear how ProxSARAH could show these results without taking the benefit of the special setting. More importantly, in the convergence analysis of NC-SARAH for the non-convex case, we show that our analysis generalizes the total complexity of Gradient Descent (Remark 1). Although GD is a special case of SVRG and SARAH when there is no inner loop, we are not aware of any work that can show that the complexity of variance reduction methods reduces to GD. Up to the best of our knowledge, this problem has been a standing open question since 2012. Using the proof techniques in this paper, we are now able to answer this question affirmatively. Our contribution has provided a rigorous relation between GD and variance reduction methods. Therefore, we believe, our

[4] According to our analysis, NC-SARAH also has the option to choose a mini-batch size $b \in (\sqrt{n}, n]$, but for such a choice we cannot attain a total complexity of $\mathcal{O}\left(\frac{\sqrt{n}}{\epsilon} \vee n\right)$.

technical results manage to build a bridge and connect the existing knowledge from GD - a well known method to the new variance reduction technique.

For convex problems, we derive the convergence results of our proposed method SARAH++ in terms of the iteration complexity. It is not capable to compare our complexity results with the SVRG-type algorithms (see e.g. [3,23,14]) since they use the complexity based on the total number of component gradient evaluations. In Section 5.2, our experiments show the improved performance of SARAH++ over SARAH in the convex cases. Moreover, we have also proposed a new variant called SARAH Adaptive which further improves the performance as show in Section 5.3. Therefore, we believe that the total complexity for SARAH++ and the convergence analysis for SARAH Adaptive are desired and potential for future research.

1.2 Paper Organization

The rest of the paper is organized as follows. Section 2 gives the convergence analysis of NC-SARAH in the non-convex case for both single batch and mini-batch cases. Section 3 shows the advantages of NC-SARAH over SPIDER and Spider-Boost in detail. In Section 4, we provide the convergence analysis of SARAH++ in the convex case and its iteration complexity. Numerical experiments are given in Section 5 to show the good performance of NC-SARAH over SPIDER and Spider-Boost (Section 5.1) and SARAH++ and SARAH Adaptive over the original SARAH (Sections 5.2 and 5.3). We conclude the paper and discuss future work in Section 6.

2 Non-Convex Case: Convergence Analysis of NC-SARAH

We will analyze NC-SARAH for smooth non-convex optimization, i.e., we study (1) with the following *average smooth* assumption (see e.g. [8]).

Assumption 1 (average-L-smooth) *The objective function F is L-average-smooth, i.e., there exists a constant $L > 0$ such that, $\forall w, w' \in \mathbb{R}^d$,*

$$\frac{1}{n}\sum_{i=1}^{n} \|\nabla f_i(w) - \nabla f_i(w')\|^2 \leq L^2 \|w - w'\|^2. \tag{4}$$

We notice that, the above assumption is weaker than the assumption on L-smoothness of each f_i, $i = 1, \ldots, n$. Throughout this paper for non-convex results, we only consider Assumption 1 and no additional assumptions are needed. We stress that our convergence analysis only relies on the above average smooth assumption without bounded variance assumption (as required in [12,29]). We note that Assumption 1 implies that F is L-smooth, that is, there exists a constant $L > 0$ such that, $\forall w, w' \in \mathbb{R}^d$, $\|\nabla F(w) - \nabla F(w')\| \leq L\|w - w'\|$. By Theorem 2.1.5 in [16], we obtain

$$F(w) \leq F(w') + \nabla F(w')^T (w - w') + \frac{L}{2}\|w - w'\|^2. \tag{5}$$

2.1 Single batch case

We start analyzing NC-SARAH (Algorithm 1) for the case where we choose a single sample i_t uniformly at random from $[n]$ in the inner loop. We first provide the following key lemmas.

Lemma 1 (Lemma 2 in [18] (or in [19])) *Consider $v_t^{(s)}$ defined by* (2) *(or* (3)*) in SARAH (Algorithm 1) for any $s \geq 1$. Then for any $t \geq 1$,*

$$\mathbb{E}[\|\nabla F(w_t^{(s)}) - v_t^{(s)}\|^2] = \sum_{j=1}^{t} \mathbb{E}[\|v_j^{(s)} - v_{j-1}^{(s)}\|^2] - \sum_{j=1}^{t} \mathbb{E}[\|\nabla F(w_j^{(s)}) - \nabla F(w_{j-1}^{(s)})\|^2]. \tag{6}$$

Lemma 2 *Suppose that Assumption 1 holds. Consider a single outer loop iteration in NC-SARAH (Algorithm 1) with $\eta \leq \frac{2}{L(\sqrt{1+4m}+1)}$. Then, for any $s \geq 1$, we have*

$$\mathbb{E}[F(w_{m+1}^{(s)})] \leq \mathbb{E}[F(w_0^{(s)})] - \frac{\eta}{2} \sum_{t=0}^{m} \mathbb{E}[\|\nabla F(w_t^{(s)})\|^2]. \tag{7}$$

Proof We use some parts of the proof in [19]. By Assumption 1 and $w_{t+1}^{(s)} = w_t^{(s)} - \eta v_t^{(s)}$, for any $s \geq 1$, we have

$$\begin{aligned}
\mathbb{E}[F(w_{t+1}^{(s)})] &\overset{(5)}{\leq} \mathbb{E}[F(w_t^{(s)})] - \eta \mathbb{E}[\nabla F(w_t^{(s)})^T v_t^{(s)}] + \frac{L\eta^2}{2} \mathbb{E}[\|v_t^{(s)}\|^2] \\
&= \mathbb{E}[F(w_t^{(s)})] - \frac{\eta}{2} \mathbb{E}[\|\nabla F(w_t^{(s)})\|^2] + \frac{\eta}{2} \mathbb{E}[\|\nabla F(w_t^{(s)}) - v_t^{(s)}\|^2] \\
&\quad - \left(\frac{\eta}{2} - \frac{L\eta^2}{2}\right) \mathbb{E}[\|v_t^{(s)}\|^2],
\end{aligned} \tag{8}$$

where the last equality follows from the fact $a^T b = \frac{1}{2}\left[\|a\|^2 + \|b\|^2 - \|a-b\|^2\right]$, for any $a, b \in \mathbb{R}^d$. By summing over $t = 0, \ldots, m$, we have

$$\begin{aligned}
\mathbb{E}[F(w_{m+1}^{(s)})] \leq \mathbb{E}[F(w_0^{(s)})] &- \frac{\eta}{2} \sum_{t=0}^{m} \mathbb{E}[\|\nabla F(w_t^{(s)})\|^2] \\
&+ \frac{\eta}{2} \left(\sum_{t=0}^{m} \mathbb{E}[\|\nabla F(w_t^{(s)}) - v_t^{(s)}\|^2] - (1 - L\eta) \sum_{t=0}^{m} \mathbb{E}[\|v_t^{(s)}\|^2] \right).
\end{aligned} \tag{9}$$

Now, we would like to determine η such that the expression in (9)

$$\sum_{t=0}^{m} \mathbb{E}[\|\nabla F(w_t^{(s)}) - v_t^{(s)}\|^2] - (1 - L\eta) \sum_{t=0}^{m} \mathbb{E}[\|v_t^{(s)}\|^2] \leq 0.$$

Let $\mathcal{F}_j = \sigma(w_0, w_1, \ldots, w_j)$ be the σ-algebra generated by $w_0, w_1, \ldots, w_j$. Note that $\mathcal{F}_j$ also contains all information of $v_0, \ldots, v_{j-1}$. We have

$$\begin{aligned}\mathbb{E}[\|v_j^{(s)} - v_{j-1}^{(s)}\|^2|\mathcal{F}_j] &\overset{(2)}{=} \mathbb{E}[\|\nabla f_{i_j}(w_j^{(s)}) - \nabla f_{i_j}(w_{j-1}^{(s)})\|^2|\mathcal{F}_j] \\ &\overset{(4)}{\leq} L^2\|w_j^{(s)} - w_{j-1}^{(s)}\|^2 = L^2\eta^2\|v_{j-1}^{(s)}\|^2,\ j \geq 1.\end{aligned}$$

Taking the expectations to both sides yields

$$\mathbb{E}[\|v_j^{(s)} - v_{j-1}^{(s)}\|^2] \leq L^2\eta^2\mathbb{E}[\|v_{j-1}^{(s)}\|^2]. \tag{10}$$

Hence, by Lemma 1, we have

$$\mathbb{E}[\|\nabla F(w_t^{(s)}) - v_t^{(s)}\|^2] \leq \sum_{j=1}^{t}\mathbb{E}[\|v_j^{(s)} - v_{j-1}^{(s)}\|^2] \overset{(10)}{\leq} L^2\eta^2\sum_{j=1}^{t}\mathbb{E}[\|v_{j-1}^{(s)}\|^2].$$

Note that $\|\nabla F(w_0^{(s)}) - v_0^{(s)}\|^2 = 0$. By summing over $t = 0, \ldots, m$ ($m \geq 1$), we have

$$\sum_{t=0}^{m}\mathbb{E}\|\nabla F(w_t^{(s)}) - v_t^{(s)}\|^2 \leq L^2\eta^2\Big[m\mathbb{E}\|v_0^{(s)}\|^2 + (m-1)\mathbb{E}\|v_1^{(s)}\|^2 + \cdots + \mathbb{E}\|v_{m-1}^{(s)}\|^2\Big].$$

By choosing $\eta \leq \frac{2}{L\left(\sqrt{1+4m}+1\right)}$, we have

$$\begin{aligned}&\sum_{t=0}^{m}\mathbb{E}[\|\nabla F(w_t^{(s)}) - v_t^{(s)}\|^2] - (1-L\eta)\sum_{t=0}^{m}\mathbb{E}[\|v_t^{(s)}\|^2] \\ &\leq L^2\eta^2\Big[m\mathbb{E}\|v_0^{(s)}\|^2 + (m-1)\mathbb{E}\|v_1^{(s)}\|^2 + \cdots + \mathbb{E}\|v_{m-1}^{(s)}\|^2\Big] \\ &\quad - (1-L\eta)\Big[\mathbb{E}\|v_0^{(s)}\|^2 + \mathbb{E}\|v_1^{(s)}\|^2 + \cdots + \mathbb{E}\|v_m^{(s)}\|^2\Big] \\ &\leq \Big[L^2\eta^2 m - (1-L\eta)\Big]\sum_{t=1}^{m}\mathbb{E}[\|v_{t-1}^{(s)}\|^2] \leq 0,\end{aligned} \tag{11}$$

since $\eta = \frac{2}{L\left(\sqrt{1+4m}+1\right)}$ is a root of equation $L^2\eta^2 m - (1-L\eta) = 0$. Therefore, with $\eta \leq \frac{2}{L(\sqrt{1+4m}+1)}$, we have

$$\mathbb{E}[F(w_{m+1}^{(s)})] \leq \mathbb{E}[F(w_0^{(s)})] - \frac{\eta}{2}\sum_{t=0}^{m}\mathbb{E}[\|\nabla F(w_t^{(s)})\|^2].$$

This completes the proof. □

The above result is for a single outer loop iteration of NC-SARAH, which includes a full gradient step together with the inner loop. Since the outer loop iteration concludes with $\tilde{w}_s = w_{m+1}^{(s)}$, and $\tilde{w}_{s-1} = w_0^{(s)}$, we have $\mathbb{E}[F(\tilde{w}_s)] \leq \mathbb{E}[F(\tilde{w}_{s-1})] - \frac{\eta}{2}\sum_{t=0}^{m}\mathbb{E}[\|\nabla F(w_t^{(s)})\|^2]$. Summing over $1 \leq s \leq S$ gives

$$\mathbb{E}[F(\tilde{w}_S)] \leq \mathbb{E}[F(\tilde{w}_0)] - \frac{\eta}{2}\sum_{s=1}^{S}\sum_{t=0}^{m}\mathbb{E}[\|\nabla F(w_t^{(s)})\|^2]. \tag{12}$$

This proves our main result:

Theorem 1 *Suppose that Assumption 1 holds. Consider NC-SARAH (Algorithm 1) with* $\eta \leq \frac{2}{L(\sqrt{1+4m}+1)}$. *Then, for any given* $\tilde{w}_0$, *we have*

$$\frac{1}{(m+1)S}\sum_{s=1}^{S}\sum_{t=0}^{m}\mathbb{E}[\|\nabla F(w_t^{(s)})\|^2] \leq \frac{2}{\eta[(m+1)S]}[F(\tilde{w}_0) - F^*],$$

where F^* *is any lower bound of* F, *and* $w_t^{(s)}$ *is the solution at the* t*-th iteration in the* s*-th outer loop.*

The proof easily follows from (12) since F^* is a lower bound of F (that is, $\mathbb{E}[F(\tilde{w}_S)] \geq F^*$). We note that the term $\frac{1}{(m+1)S}\sum_{s=1}^{S}\sum_{t=0}^{m}\mathbb{E}[\|\nabla F(w_t^{(s)})\|^2]$ is simply the average of the expectation of the squared norms of the gradients of all the iteration results generated by NC-SARAH. For non-convex problems, our goal is to achieve

$$\frac{1}{(m+1)S}\sum_{s=1}^{S}\sum_{t=0}^{m}\mathbb{E}[\|\nabla F(w_t^{(s)})\|^2] \leq \epsilon. \tag{13}$$

We note that, for simplicity, if $\bar{w}_s$ is chosen uniformly at random from all the iterations generated by NC-SARAH, we are able to have accuracy $\mathbb{E}[\|\nabla F(\bar{w}_s)\|^2] \leq \epsilon$. From Theorem 1 with $\eta = \mathcal{O}(1/\sqrt{m+1})$ we infer that (13) can be realized for $S = \mathcal{O}(\frac{1}{\epsilon\sqrt{m+1}} \vee 1)$. The total complexity of NC-SARAH is equal to $S(n+2m)$ which proves the following result.

Corollary 1 *Suppose that Assumption 1 holds. Let us consider NC-SARAH (Algorithm 1) with* $\eta = \frac{2}{L(\sqrt{1+4m}+1)}$ *where* m *is the inner loop size. Then, in order to achieve an* ϵ*-accurate solution, the total complexity is* $\mathcal{O}\left(\left[\left(\frac{n+2m}{\sqrt{m+1}}\right)\frac{1}{\epsilon}\right] \vee [n+2m]\right)$.

Proof To achieve $\frac{1}{(m+1)S}\sum_{s=1}^{S}\sum_{t=0}^{m}\mathbb{E}[\|\nabla F(w_t^{(s)})\|^2] \leq \epsilon$, it is sufficient to prove

$$\frac{2}{\eta[(m+1)S]}[F(\tilde{w}_0) - F^*] \leq \epsilon. \tag{14}$$

Notice that $\eta\sqrt{m+1} = \frac{2}{L}\frac{\sqrt{m+1}}{\sqrt{1+4m}+1} \leq \frac{2}{L}$. Hence, in order to achieve (14), we need

$$S \geq \frac{2}{\eta[(m+1)\epsilon]}[F(\tilde{w}_0) - F^*] \geq \frac{L[F(\tilde{w}_0) - F^*]}{(\sqrt{m+1})}\frac{1}{\epsilon}.$$

Together with the requirement $S \geq 1$, we can choose $S = \mathcal{O}\left(\left[\frac{1}{\sqrt{m+1}} \cdot \frac{1}{\epsilon}\right] \vee 1\right)$. Therefore, the total complexity to achieve ϵ-accurate solution is

$$(n+2m)S = \mathcal{O}\left(\left[\left(\frac{n+2m}{\sqrt{m+1}}\right)\frac{1}{\epsilon}\right] \vee [n+2m]\right).$$

This completes the proof. □

The total complexity can be minimized over the inner loop size m. By choosing $m = n$, we achieve the minimal total complexity:

Corollary 2 *Suppose that Assumption 1 holds. Let us consider NC-SARAH (Algorithm 1) with* $\eta = \frac{2}{L(\sqrt{1+4m}+1)}$ *where* m *is the inner loop size and chosen equal to* $m = n$*. Then, in order to achieve an* ϵ*-accurate solution, the total complexity is* $\mathcal{O}\left(\frac{\sqrt{n}}{\epsilon} \vee n\right)$.

Remark 1 *The total complexity in Corollary 1 covers all choices for the inner loop size* m*. For example, in the case of* $m = 0$*, NC-SARAH recovers the Gradient Descent (GD) algorithm which has total complexity* $\mathcal{O}\left(\frac{n}{\epsilon}\right)$*. Theorem 1 for* $m = 0$ *also recovers the requirement on the learning rate for GD, which is* $\eta \leq \frac{1}{L}$.

The above results explain the relationship between NC-SARAH and GD and explain the advantages of the inner loop and outer loop of NC-SARAH. NC-SARAH becomes more beneficial in ML applications where n is large.

2.2 Mini-batch case

The above results can be extended to the *mini-batch* case where instead of (2) the update rule (3) is used as explained in the introduction.

Theorem 2 *Suppose that Assumption 1 holds. Consider NC-SARAH (Algorithm 1) by replacing the update of* v_t *in the inner loop by* (3) *with*

$$\eta \leq \frac{2}{L\left(\sqrt{1 + \frac{4m}{b}\left(\frac{n-b}{n-1}\right)} + 1\right)}. \tag{15}$$

Then, for any given $\tilde{w}_0$*, we have*

$$\frac{1}{(m+1)S}\sum_{s=1}^{S}\sum_{t=0}^{m}\mathbb{E}[\|\nabla F(w_t^{(s)})\|^2] \leq \frac{2}{\eta[(m+1)S]}[F(\tilde{w}_0) - F^*],$$

where F^* *is any lower bound of* F*, and* $w_t^{(s)}$ *is the solution at the* t*-th iteration in the* s*-th outer loop.*

Proof Consider $v_t^{(s)}$ defined by (3) in NC-SARAH (Algorithm 1) for any $s \geq 1$. Then by Lemma 2, for any $t \geq 1$,

$$\mathbb{E}[\|\nabla F(w_t^{(s)}) - v_t^{(s)}\|^2] = \sum_{j=1}^{t}\mathbb{E}[\|v_j^{(s)} - v_{j-1}^{(s)}\|^2] - \sum_{j=1}^{t}\mathbb{E}[\|\nabla F(w_j^{(s)}) - \nabla F(w_{j-1}^{(s)})\|^2]. \tag{16}$$

Following the proof of Lemma 2, we would like to determine η such that the expression in (9)

$$\sum_{t=0}^{m}\mathbb{E}[\|\nabla F(w_t^{(s)}) - v_t^{(s)}\|^2] - (1 - L\eta)\sum_{t=0}^{m}\mathbb{E}[\|v_t^{(s)}\|^2] \leq 0.$$

Let

$$\xi_t = \nabla f_t(w_j^{(s)}) - \nabla f_t(w_{j-1}^{(s)}). \tag{17}$$

Let $\mathcal{F}_j = \sigma(w_0^{(s)}, I_1, I_2, \ldots, I_{j-1})$ be the σ-algebra generated by $w_0^{(s)}, I_1, I_2, \ldots, I_{j-1}$; $\mathcal{F}_0 = \mathcal{F}_1 = \sigma(w_0^{(s)})$. Note that $\mathcal{F}_j$ also contains all the information of $w_0^{(s)}, \ldots, w_j^{(s)}$ as well as $v_0^{(s)}, \ldots, v_{j-1}^{(s)}$. We have

$$\begin{aligned}
&\mathbb{E}[\|v_j^{(s)} - v_{j-1}^{(s)}\|^2|\mathcal{F}_j] - \|\nabla F(w_j^{(s)}) - \nabla F(w_{j-1}^{(s)})\|^2 \\
&\overset{(3)}{=} \mathbb{E}\Big[\Big\|\frac{1}{b}\sum_{i\in I_j}[\nabla f_i(w_j^{(s)}) - \nabla f_i(w_{j-1}^{(s)})]\Big\|^2\Big|\mathcal{F}_j\Big] - \Big\|\frac{1}{n}\sum_{i=1}^{n}[\nabla f_i(w_j^{(s)}) - \nabla f_i(w_{j-1}^{(s)})]\Big\|^2 \\
&\overset{(17)}{=} \mathbb{E}\Big[\Big\|\frac{1}{b}\sum_{i\in I_j}\xi_i\Big\|^2\Big|\mathcal{F}_j\Big] - \Big\|\frac{1}{n}\sum_{i=1}^{n}\xi_i\Big\|^2 = \frac{1}{b^2}\mathbb{E}\Big[\sum_{i\in I_j}\sum_{k\in I_j}\xi_i^T\xi_k\Big|\mathcal{F}_j\Big] - \frac{1}{n^2}\sum_{i=1}^{n}\sum_{k=1}^{n}\xi_i^T\xi_k \\
&= \frac{1}{b^2}\mathbb{E}\Big[\sum_{i\neq k\in I_j}\xi_i^T\xi_k + \sum_{i\in I_j}\xi_i^T\xi_i\Big|\mathcal{F}_j\Big] - \frac{1}{n^2}\sum_{i=1}^{n}\sum_{k=1}^{n}\xi_i^T\xi_k \\
&= \frac{1}{b^2}\Big[\frac{b}{n}\frac{(b-1)}{(n-1)}\sum_{i\neq k}\xi_i^T\xi_k + \frac{b}{n}\sum_{i=1}^{n}\xi_i^T\xi_i\Big] - \frac{1}{n^2}\sum_{i=1}^{n}\sum_{k=1}^{n}\xi_i^T\xi_k \\
&= \frac{1}{b^2}\Big[\frac{b}{n}\frac{(b-1)}{(n-1)}\sum_{i=1}^{n}\sum_{k=1}^{n}\xi_i^T\xi_k + \Big(\frac{b}{n} - \frac{b}{n}\frac{(b-1)}{(n-1)}\Big)\sum_{i=1}^{n}\xi_i^T\xi_i\Big] - \frac{1}{n^2}\sum_{i=1}^{n}\sum_{k=1}^{n}\xi_i^T\xi_k \\
&= \frac{1}{bn}\Big[\Big(\frac{(b-1)}{(n-1)} - \frac{b}{n}\Big)\sum_{i=1}^{n}\sum_{k=1}^{n}\xi_i^T\xi_k + \frac{(n-b)}{(n-1)}\sum_{i=1}^{n}\xi_i^T\xi_i\Big] \\
&= \frac{1}{bn}\Big(\frac{n-b}{n-1}\Big)\Big[-\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{n}\xi_i^T\xi_k + \sum_{i=1}^{n}\xi_i^T\xi_i\Big] \\
&= \frac{1}{bn}\Big(\frac{n-b}{n-1}\Big)\Big[-n\Big\|\frac{1}{n}\sum_{i=1}^{n}\xi_i\Big\|^2 + \sum_{i=1}^{n}\|\xi_i\|^2\Big] \\
&\leq \frac{1}{b}\Big(\frac{n-b}{n-1}\Big)\frac{1}{n}\sum_{i=1}^{n}\|\xi_i\|^2 \\
&\overset{(17)}{=} \frac{1}{b}\Big(\frac{n-b}{n-1}\Big)\frac{1}{n}\sum_{i=1}^{n}\|\nabla f_i(w_j^{(s)}) - \nabla f_i(w_{j-1}^{(s)})\|^2 \overset{(4)}{\leq} \frac{1}{b}\Big(\frac{n-b}{n-1}\Big)L^2\eta^2\|v_{j-1}^{(s)}\|^2.
\end{aligned}$$

Hence, by taking expectation, we have

$$\mathbb{E}[\|v_j^{(s)} - v_{j-1}^{(s)}\|^2] - \mathbb{E}[\|\nabla F(w_j^{(s)}) - \nabla F(w_{j-1}^{(s)})\|^2] \leq \frac{1}{b}\Big(\frac{n-b}{n-1}\Big)L^2\eta^2\mathbb{E}[\|v_{j-1}^{(s)}\|^2].$$

By (16), for $t \geq 1$,

$$\mathbb{E}[\|\nabla F(w_t^{(s)}) - v_t^{(s)}\|^2] = \sum_{j=1}^{t}\mathbb{E}[\|v_j^{(s)} - v_{j-1}^{(s)}\|^2] - \sum_{j=1}^{t}\mathbb{E}[\|\nabla F(w_j^{(s)}) - \nabla F(w_{j-1}^{(s)})\|^2]$$

$$\leq \frac{1}{b}\left(\frac{n-b}{n-1}\right) L^2\eta^2 \sum_{j=1}^{t} \mathbb{E}[\|v_{j-1}^{(s)}\|^2].$$

Note that $\|\nabla F(w_0^{(s)}) - v_0^{(s)}\|^2 = 0$. By summing over $t = 0, \ldots, m$ ($m \geq 1$), we have

$$\sum_{t=0}^{m} \mathbb{E}\|\nabla F(w_t^{(s)}) - v_t^{(s)}\|^2$$
$$\leq \frac{1}{b}\left(\frac{n-b}{n-1}\right) L^2\eta^2 \Big[m\mathbb{E}\|v_0^{(s)}\|^2 + (m-1)\mathbb{E}\|v_1^{(s)}\|^2 + \cdots + \mathbb{E}\|v_{m-1}^{(s)}\|^2\Big].$$

By choosing $\eta \leq \frac{2}{L\left(\sqrt{1+\frac{4m}{b}\left(\frac{n-b}{n-1}\right)}+1\right)}$, we have

$$\begin{aligned}
&\sum_{t=0}^{m} \mathbb{E}[\|\nabla F(w_t^{(s)}) - v_t^{(s)}\|^2] - (1-L\eta)\sum_{t=0}^{m} \mathbb{E}[\|v_t^{(s)}\|^2] \\
&\leq \frac{1}{b}\left(\frac{n-b}{n-1}\right) L^2\eta^2 \Big[m\mathbb{E}\|v_0^{(s)}\|^2 + (m-1)\mathbb{E}\|v_1^{(s)}\|^2 + \cdots + \mathbb{E}\|v_{m-1}^{(s)}\|^2\Big] \\
&- (1-L\eta)\Big[\mathbb{E}\|v_0^{(s)}\|^2 + \mathbb{E}\|v_1^{(s)}\|^2 + \cdots + \mathbb{E}\|v_m^{(s)}\|^2\Big] \\
&\leq \Big[\frac{1}{b}\left(\frac{n-b}{n-1}\right) L^2\eta^2 m - (1-L\eta)\Big] \sum_{t=1}^{m} \mathbb{E}[\|v_{t-1}^{(s)}\|^2] \leq 0, \qquad (18)
\end{aligned}$$

since $\eta = \frac{2}{L\left(\sqrt{1+\frac{4m}{b}\left(\frac{n-b}{n-1}\right)}+1\right)}$ is a root of equation $\frac{1}{b}\left(\frac{n-b}{n-1}\right) L^2\eta^2 m - (1 - L\eta) = 0$. Therefore, with $\eta \leq \frac{2}{L\left(\sqrt{1+\frac{4m}{b}\left(\frac{n-b}{n-1}\right)}+1\right)}$, we have

$$\mathbb{E}[F(w_{m+1}^{(s)})] \leq \mathbb{E}[F(w_0^{(s)})] - \frac{\eta}{2}\sum_{t=0}^{m} \mathbb{E}[\|\nabla F(w_t^{(s)})\|^2].$$

Following the same derivation of Theorem 1, for any given $\tilde{w}_0$, we could achieve the desired result. □

We can again derive similar corollaries as was done for Theorem 1.

Corollary 3 *For the conditions in Theorem 2 with equality for η in (15), in order to achieve an ϵ-accurate solution the total complexity is*

$$\mathcal{O}\left(\left[\left(\frac{n+2bm}{m+1}\right)\left(\sqrt{1+\frac{4m}{b}\left(\frac{n-b}{n-1}\right)}\right)\frac{1}{\epsilon}\right] \vee [n+2bm]\right).$$

Proof By Theorem 2, let $\eta = \frac{2}{L\left(\sqrt{1+\frac{4m}{b}\left(\frac{n-b}{n-1}\right)}+1\right)}$. Hence, we have

$$\begin{aligned}\frac{1}{(m+1)S}\sum_{s=1}^{S}\sum_{t=0}^{m}\mathbb{E}[\|\nabla F(w_t^{(s)})\|^2] &\leq \frac{2}{\eta[(m+1)S]}[F(\tilde{w}_0)-F^*] \\ &= \frac{\left(\sqrt{1+\frac{4m}{b}\left(\frac{n-b}{n-1}\right)}+1\right)}{(m+1)S}L[F(\tilde{w}_0)-F^*] \\ &\leq \frac{\left(\sqrt{1+\frac{4m}{b}\left(\frac{n-b}{n-1}\right)}\right)}{(m+1)S}2L[F(\tilde{w}_0)-F^*] = \epsilon.\end{aligned}$$

In order to achieve the ϵ-accurate solution, we need

$$S = \frac{\left(\sqrt{1+\frac{4m}{b}\left(\frac{n-b}{n-1}\right)}\right)}{(m+1)\epsilon}2L[F(\tilde{w}_0)-F^*] = \mathcal{O}\left(\frac{\left(\sqrt{1+\frac{4m}{b}\left(\frac{n-b}{n-1}\right)}\right)}{(m+1)\epsilon}\vee 1\right),$$

since $S \geq 1$. Therefore, the total complexity is

$$(n+b\cdot 2m)S = \mathcal{O}\left(\left[\left(\frac{n+2bm}{m+1}\right)\left(\sqrt{1+\frac{4m}{b}\left(\frac{n-b}{n-1}\right)}\right)\frac{1}{\epsilon}\right]\vee[n+2bm]\right).$$

This completes the proof. □

Corollary 4 *For the conditions in Corollary 3 with $b = n^{1/2-\gamma}$ and $m = n^{1/2+\gamma}$, where $0 \leq \gamma \leq 1/2$, in order to achieve an ϵ-accurate solution the total complexity is $\mathcal{O}\left(\frac{\sqrt{n}}{\epsilon}\vee n\right)$.*

Proof Let $b = n^{\alpha}$ and $m = n^{\beta}$ where $0 \leq \alpha, \beta \leq 1$, we have

$$\begin{aligned}\left(\frac{n+2bm}{m+1}\right)\sqrt{1+\frac{4m}{b}\left(\frac{n-b}{n-1}\right)} &= \left(\frac{n+2n^{\alpha+\beta}}{n^{\beta}+1}\right)\sqrt{1+4n^{\beta-\alpha}\left(\frac{n-n^{\alpha}}{n-1}\right)} \\ &\leq \frac{n+2n^{\alpha+\beta}}{n^{\beta}}2\sqrt{1+n^{\beta-\alpha}}.\end{aligned}$$

If $\beta \geq \alpha$, we have

$$\begin{aligned}\frac{n+2n^{\alpha+\beta}}{n^{\beta}}2\sqrt{1+n^{\beta-\alpha}} &\leq 2\sqrt{2}\left(\frac{n+2n^{\alpha+\beta}}{n^{\beta}}\right)n^{(\beta-\alpha)/2} \\ &= 2\sqrt{2}(n^{1-\alpha/2-\beta/2}+2n^{\alpha/2+\beta/2}).\end{aligned}$$

In order to minimize the order of n, we need to choose $1-\alpha/2-\beta/2 = \alpha/2+\beta/2$, which is equivalent to $\alpha+\beta = 1$ with $\beta \geq \alpha$. The best option is to choose $\alpha+\beta = 1$ with $\beta \geq 1/2$ and $0 \leq \alpha \leq 1/2$ in order to achieve $\mathcal{O}(n^{1/2})$.

If $\beta \leq \alpha$, we have $\frac{n+2n^{\alpha+\beta}}{n^{\beta}} 2\sqrt{1+n^{\beta-\alpha}} \leq 2\sqrt{2}(n^{1-\beta}+2n^{\alpha})$. In order to minimize the order of n, we need to choose $1-\beta=\alpha$, which is equivalent to $\alpha+\beta=1$ with $\beta \leq \alpha$. The best option is to choose $\beta=1/2$ and $\alpha=1/2$ in order to achieve $\mathcal{O}(n^{1/2})$.

Therefore, with $b=n^{\alpha}$ and $m=n^{\beta}$ where $\alpha+\beta=1$ with $\beta \geq 1/2$ and $0 \leq \alpha \leq 1/2$, we have $\left(\frac{n+2bm}{m+1}\right)\sqrt{1+\frac{4m}{b}\left(\frac{n-b}{n-1}\right)}=\mathcal{O}(n^{1/2})$.

By Corollary 3 with $bm=n^{\alpha+\beta}=n$, it implies the total complexity

$$(n+b\cdot 2m)S=\mathcal{O}\left(\frac{\sqrt{n}}{\epsilon}\vee n\right).$$

Hence, by setting $\alpha=1/2-\gamma$ and $\beta=1/2+\gamma$ with $0 \leq \gamma \leq 1/2$, we obtain the corollary. □

Remark 2 *The choice of η in Theorem 2 is more general than in Theorem 1. For $b=n$ and $m=m_0$, for some non-negative integer m_0, it recovers the convergence rate of Gradient Descent with learning rate $\eta \leq \frac{1}{L}$ and total complexity $\mathcal{O}\left(\frac{n}{\epsilon}\right)$ (see Corollary 3).*

3 Comparison of NC-SARAH, SPIDER, and SpiderBoost

As shown in the previous section, like SPIDER [8] and SpiderBoost [28], also NC-SARAH enjoys the same asymptotic total complexity of $\mathcal{O}\left(n+\sqrt{n}/\epsilon\right)$.

In this section, we show practical advantages of NC-SARAH over SPIDER and SpiderBoost. By using our notation of b and m, the three algorithms have the following properties:

– **SPIDER**: For $0 \leq \gamma \leq 1/2$,

$$b=n^{1/2-\gamma}\ ,\ m=n^{1/2+\gamma}\ ,\ \eta_t^{(s)}=\min\left\{\frac{\epsilon}{Ln^{\gamma}\|v_t^{(s)}\|},\frac{1}{2Ln^{\gamma}}\right\}, \tag{19}$$

where $v_t^{(s)}$ is the SARAH update (3), and $\eta_t^{(s)}$ denotes the learning rate of the t-th iteration in the inner loop of the s-th outer loop.

– **SpiderBoost**:

$$b=n^{1/2}\ ,m=n^{1/2}\ ,\ \eta=\frac{1}{2L}. \tag{20}$$

– **NC-SARAH**: For $0 \leq \gamma \leq 1/2$,

$$b=n^{1/2-\gamma}\ ,\ m=n^{1/2+\gamma}\ ,\ \eta=\frac{2}{L\left(\sqrt{1+4n^{2\gamma}\left(\frac{n-n^{1/2-\gamma}}{n-1}\right)}+1\right)}. \tag{21}$$

The following subsections analyze NC-SARAH in comparison to SPIDER and SpiderBoost, respectively.

3.1 NC-SARAH vs SPIDER

NC-SARAH and SPIDER have the same flexibility of choosing mini-batch size $b \in [1, \sqrt{n}]$. However, the learning rate of SPIDER can be quite small compared to the learning rate of NC-SARAH because SPIDER's learning rate scales linearly with ϵ for learning rates $\leq \frac{1}{2Ln^{\gamma}}$ and ϵ will be small especially when we want a small ϵ-accurate solution.

Corollary 5 *For the same mini-batch size b for the inner loop and the same number of inner loop iterations m, the learning rate choice of NC-SARAH is strictly larger than that of SPIDER when $n > 1$.*

Proof We recall the SPIDER's setting:

$$b = n^{1/2-\gamma} \ , \ m = n^{1/2+\gamma} \ , \ \eta_t^{(s)} = \min\left\{ \frac{\epsilon}{Ln^{\gamma}\|v_t^{(s)}\|}, \frac{1}{2Ln^{\gamma}} \right\},$$

where $v_t^{(s)}$ is the SARAH update (3), and $\eta_t^{(s)}$ is the learning rate of the s-outer loop and t-th iteration in the inner loop; and the NC-SARAH's setting:

$$b = n^{1/2-\gamma} \ , \ m = n^{1/2+\gamma} \ , \ \eta = \frac{2}{L\left(\sqrt{1 + 4n^{2\gamma}\left(\frac{n-n^{1/2-\gamma}}{n-1}\right)} + 1\right)},$$

where $0 \leq \gamma \leq 1/2$.

Since $\eta_t^{(s)} = \min\left\{ \frac{\epsilon}{Ln^{\gamma}\|v_t^{(s)}\|}, \frac{1}{2Ln^{\gamma}} \right\} \leq \frac{1}{2Ln^{\gamma}}$. In order to achieve the desired result, it is sufficient to show that, for $0 \leq \gamma \leq 1/2$,

$$\frac{2}{L\left(\sqrt{1 + 4n^{2\gamma}\left(\frac{n-n^{1/2-\gamma}}{n-1}\right)} + 1\right)} > \frac{1}{2Ln^{\gamma}}. \tag{22}$$

This is equivalent to showing

$$\begin{aligned} 4n^{\gamma} &> \sqrt{1 + 4n^{2\gamma}\left(\frac{n-n^{1/2-\gamma}}{n-1}\right)} + 1 \\ 16n^{2\gamma} - 8n^{\gamma} + 1 &> 1 + 4n^{2\gamma}\left(\frac{n-n^{1/2-\gamma}}{n-1}\right) \\ 4 - \frac{2}{n^{\gamma}} &> \frac{n-n^{1/2-\gamma}}{n-1} = 1 - \frac{n^{1/2-\gamma}-1}{n-1}. \end{aligned}$$

The last inequality clearly holds since $4 - \frac{2}{n^{\gamma}} \geq 2$. Therefore, we obtain (22). □

3.2 NC-SARAH vs SpiderBoost

It is clear that, compared to SpiderBoost with only $b = \sqrt{n}$, NC-SARAH has more flexibility of choosing mini-batch size $b = n^{1/2-\gamma}$ for some $0 \leq \gamma \leq 1/2$. In order to compare the learning rate of NC-SARAH and SpiderBoost for the same mini-batch size, we let $\gamma = 0$ in NC-SARAH's parameter setting; we obtain

$$b = \sqrt{n}\ ,\ m = \sqrt{n}\ ,\ \eta = \frac{2}{L\left(\sqrt{1+4\left(\frac{n-n^{1/2}}{n-1}\right)}+1\right)}.$$

We have the following corollary.

Corollary 6 *For the same mini-batch size b for the inner loop and the same number of inner loop iterations m, the learning rate choice of NC-SARAH is at least a factor $4/(\sqrt{5}+1) \approx 1.236$ larger than that of SpiderBoost when $n > 1$.*

Proof We need to choose $\gamma = 0$ for NC-SARAH in order to have the same option as SpiderBoost, i.e., $b = \sqrt{n}$ and $m = \sqrt{m}$. Hence, we have the learning rate of NC-SARAH

$$\frac{2}{L\left(\sqrt{1+4\left(\frac{n-n^{1/2}}{n-1}\right)}+1\right)}.$$

We notice that $\frac{n-n^{1/2}}{n-1} < 1$ for $n > 1$. Hence, we have

$$\frac{2}{L\left(\sqrt{1+4\left(\frac{n-n^{1/2}}{n-1}\right)}+1\right)} > \frac{2}{L(\sqrt{5}+1)} = \frac{4}{(\sqrt{5}+1)} \cdot \frac{1}{2L}.$$

This completes the proof. □

In Theorem 2 we can choose a smaller learning rate than the right-hand side of (15) in NC-SARAH. In this sense, the above corollary shows that SpiderBoost is a special case of NC-SARAH. The following subsection confirms numerically that the choice of $b = \sqrt{n}$ and $m = \sqrt{n}$ in SpiderBoost is not the best choice.

4 Convex Case: Convergence Analysis of SARAH++

In this section, we propose a new variant of SARAH+ (Algorithm 2) [18], called SARAH++ (Algorithm 3), for convex problems of form (1).

Different from SARAH, SARAH+ provides a stopping criteria for the inner loop; as soon as

$$\|v_{t-1}^{(s)}\|^2 \leq \gamma \|v_0^{(s)}\|^2,$$

the inner loop terminates. This idea originates from the property of SARAH that, for each outer loop iteration s, $\mathbb{E}[\|v_t^{(s)}\|^2] \to 0$ as $t \to \infty$ in the strongly convex

case (Theorems 1a and 1b in [18]). Therefore, it does not make any sense to update with tiny steps when $\|v_t^{(s)}\|^2$ is small. (We note that SVRG [9] does not have this property.) SARAH+ suggests to empirically choose parameter $\gamma = 1/8$ [18] without theoretical guarantee.

Algorithm 2 SARAH+ [18]

Parameters: the learning rate $\eta > 0$, $0 < \gamma \leq 1$, the maximum inner loop size m, and the outer loop size S
Initialize: $\tilde{w}_0$
Iterate:
for $s = 1, 2, \ldots, S$ **do**
  $w_0^{(s)} = \tilde{w}_{s-1}$
  $v_0^{(s)} = \frac{1}{n}\sum_{i=1}^n \nabla f_i(w_0^{(s)})$
  $w_1^{(s)} = w_0^{(s)} - \eta v_0^{(s)}$
  $t = 1$
  while $\|v_{t-1}^{(s)}\|^2 > \gamma \|v_0^{(s)}\|^2$ **and** $t \leq m$ **do**
    Sample i_t uniformly at random from $[n]$
    $v_t^{(s)} = \nabla f_{i_t}(w_t^{(s)}) - \nabla f_{i_t}(w_{t-1}^{(s)}) + v_{t-1}^{(s)}$
    $w_{t+1}^{(s)} = w_t^{(s)} - \eta v_t^{(s)}$
    $t \leftarrow t + 1$
  end while
  Set $\tilde{w}_s = w_t^{(s)}$
end for

Here, we modify SARAH+ (Algorithm 2) into SARAH++ (Algorithm 3) by choosing the stopping criteria for the inner loop as

$$\|v_{t-1}^{(s)}\|^2 < \gamma \|v_0^{(s)}\|^2 \text{ where } \gamma \geq L\eta$$

and by introducing a stopping criteria for the outer loop.

Before analyzing and explaining SARAH++ in detail, we introduce the following assumptions used in this section.

Assumption 2 (L-smooth) *Each $f_i : \mathbb{R}^d \to \mathbb{R}$, $i \in [n]$, is L-smooth, i.e., there exists a constant $L > 0$ such that, $\forall w, w' \in \mathbb{R}^d$,*

$$\|\nabla f_i(w) - \nabla f_i(w')\| \leq L\|w - w'\|. \tag{23}$$

Assumption 3 (μ-strongly convex) *The objective function $F : \mathbb{R}^d \to \mathbb{R}$, is μ-strongly convex, i.e., there exists a constant $\mu > 0$ such that $\forall w, w' \in \mathbb{R}^d$,*

$$F(w) \geq F(w') + \nabla F(w')^T(w - w') + \frac{\mu}{2}\|w - w'\|^2.$$

Under Assumption 3, let us define the (unique) optimal solution of (1) as w_*. Then strong convexity of F implies that

$$2\mu[F(w) - F(w_*)] \leq \|\nabla F(w)\|^2, \ \forall w \in \mathbb{R}^d. \tag{24}$$

We note here, for future use, that for strongly convex functions of the form (1), arising in machine learning applications, the condition number is defined as $\kappa \overset{\text{def}}{=} L/\mu$.

Assumption 3 covers a wide range of problems, e.g. l_2-regularized empirical risk minimization problems with convex losses.

We separately assume the special case of strong convexity of all f_i's with $\mu = 0$, called the general convexity assumption, which we will use for convergence analysis.

Assumption 4 *Each function $f_i : \mathbb{R}^d \to \mathbb{R}$, $i \in [n]$, is convex, i.e.,*

$$f_i(w) \geq f_i(w') + \nabla f_i(w')^T (w - w').$$

The following existing result is used in the proof of our main result in this section.

Lemma 3 (Lemma 3 in [18]) *Suppose that Assumptions 2 and 4 hold. Consider $v_t^{(s)}$ defined as* (2) *in SARAH (Algorithm 1) with $\eta < 2/L$ for any $s \geq 1$. Then we have that for any $t \geq 0$,*

$$\mathbb{E}[\|\nabla F(w_t^{(s)}) - v_t^{(s)}\|^2] \leq \frac{\eta L}{2 - \eta L} \Big[\mathbb{E}[\|v_0^{(s)}\|^2] - \mathbb{E}[\|v_t^{(s)}\|^2] \Big]. \tag{25}$$

SARAH++ is motivated by the following lemma.

Lemma 4 *Suppose that Assumptions 2 and 4 hold. Consider a single outer loop iteration in SARAH (Algorithm 1) with $\eta \leq \frac{1}{L}$. Then, for $t \geq 0$ and any $s \geq 1$, we have*

$$\begin{aligned} \mathbb{E}[F(w_{t+1}^{(s)}) - F(w_*)] \leq \mathbb{E}[F(w_t^{(s)}) - F(w_*)] - \frac{\eta}{2} \mathbb{E}[\|\nabla F(w_t^{(s)})\|^2] \\ + \frac{\eta}{2} \left(L\eta \mathbb{E}[\|v_0^{(s)}\|^2] - \mathbb{E}[\|v_t^{(s)}\|^2] \right), \end{aligned} \tag{26}$$

where w_* is any optimal solution of F.

Proof By using (8) and adding $-F(w_*)$ for both sides, where $w_* = \arg\min_w F(w)$, we have

$$\begin{aligned} &\mathbb{E}[F(w_{t+1}^{(s)}) - F(w_*)] \\ &\leq \mathbb{E}[F(w_t^{(s)}) - F(w_*)] - \frac{\eta}{2} \mathbb{E}[\|\nabla F(w_t^{(s)})\|^2] + \frac{\eta}{2} \mathbb{E}[\|\nabla F(w_t^{(s)}) - v_t^{(s)}\|^2] \\ &\quad - \left(\frac{\eta}{2} - \frac{L\eta^2}{2} \right) \mathbb{E}[\|v_t^{(s)}\|^2] \\ &\overset{(25)}{\leq} \mathbb{E}[F(w_t^{(s)}) - F(w_*)] - \frac{\eta}{2} \mathbb{E}[\|\nabla F(w_t^{(s)})\|^2] \\ &\quad + \frac{\eta}{2} \frac{\eta L}{(2 - \eta L)} \left(\mathbb{E}[\|v_0^{(s)}\|^2] - \mathbb{E}[\|v_t^{(s)}\|^2] \right) - \left(\frac{\eta}{2} - \frac{L\eta^2}{2} \right) \mathbb{E}[\|v_t^{(s)}\|^2] \\ &= \mathbb{E}[F(w_t^{(s)}) - F(w_*)] - \frac{\eta}{2} \mathbb{E}[\|\nabla F(w_t^{(s)})\|^2] \\ &\quad + \frac{\eta}{2} \left(\frac{\eta L}{(2 - \eta L)} \left(\mathbb{E}[\|v_0^{(s)}\|^2] - \mathbb{E}[\|v_t^{(s)}\|^2] \right) - (1 - L\eta) \mathbb{E}[\|v_t^{(s)}\|^2] \right) \\ &\overset{\eta \leq \frac{1}{L}}{\leq} \mathbb{E}[F(w_t^{(s)}) - F(w_*)] - \frac{\eta}{2} \mathbb{E}[\|\nabla F(w_t^{(s)})\|^2] \end{aligned}$$

Algorithm 3 SARAH++

Parameters: The controlled factor $0 < \gamma \leq 1$, the learning rate $0 < \eta \leq \frac{\gamma}{L}$, the total iteration $T > 0$, and the maximum inner loop size $m \leq T$.
Initialize: $\tilde{w}_0$
$G = 0$
Iterate:
$s = 0$
while $G < T$ **do**
  $s \leftarrow s + 1$
  $w_0^{(s)} = \tilde{w}_{s-1}$
  $v_0^{(s)} = \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(w_0^{(s)})$
  $t = 0$
  while $\|v_t^{(s)}\|^2 \geq \gamma \|v_0^{(s)}\|^2$ **and** $t \leq m$ **do**
    $w_{t+1}^{(s)} = w_t^{(s)} - \eta v_t^{(s)}$
    $t \leftarrow t + 1$
    if $m \neq 0$ **then**
      Sample i_t uniformly at random from $[n]$
      $v_t^{(s)} = \nabla f_{i_t}(w_t^{(s)}) - \nabla f_{i_t}(w_{t-1}^{(s)}) + v_{t-1}^{(s)}$
    end if
  end while
  $T_s = t$
  $\tilde{w}_s = w_{T_s}^{(s)}$
  $G \leftarrow G + T_s$
end while
$S = s$
Set $\hat{w} = \tilde{w}_S$

$$+ \frac{\eta}{2} \left(\eta L \left(\mathbb{E}[\|v_0^{(s)}\|^2] - \mathbb{E}[\|v_t^{(s)}\|^2] \right) - (1 - L\eta) \mathbb{E}[\|v_t^{(s)}\|^2] \right)$$
$$\leq \mathbb{E}[F(w_t^{(s)}) - F(w_*)] - \frac{\eta}{2} \mathbb{E}[\|\nabla F(w_t^{(s)})\|^2] + \frac{\eta}{2} \left(L\eta \mathbb{E}[\|v_0^{(s)}\|^2] - \mathbb{E}[\|v_t^{(s)}\|^2] \right).$$

This completes the proof. □

Clearly, if

$$L\eta \mathbb{E}[\|v_0^{(s)}\|^2] - \mathbb{E}[\|v_t^{(s)}\|^2] \leq \gamma \mathbb{E}[\|v_0^{(s)}\|^2] - \mathbb{E}[\|v_t^{(s)}\|^2] \leq 0,$$

where $\eta \leq \frac{\gamma}{L}$, inequality (26) implies

$$\mathbb{E}[F(w_{t+1}^{(s)}) - F(w_*)] \leq \mathbb{E}[F(w_t^{(s)}) - F(w_*)] - \frac{\eta}{2} \mathbb{E}[\|\nabla F(w_t^{(s)})\|^2].$$

For this reason, we choose the stopping criteria for the inner loop in SARAH++ as $\|v_t^{(s)}\|^2 < \gamma \|v_0^{(s)}\|^2$ with $\gamma \geq L\eta$. Unlike SARAH+, for analyzing the convergence rate γ can be as small as $L\eta$.

The above discussion leads to SARAH++ (Algorithm 3). In order to analyze its convergence for convex problems, we define random variable T_s as the stopping time of the inner loop in the s-th outer iteration:

$$T_s = \min \left\{ \min_{t \geq 0} \left\{ t : \|v_t^{(s)}\|^2 < \gamma \|v_0^{(s)}\|^2 \right\}, m + 1 \right\}, \ s = 1, 2, \ldots$$

Note that T_s is at least 1 since at $t = 0$, the condition $\|v_0^{(s)}\|^2 \geq \gamma \|v_0^{(s)}\|^2$ always holds (and $m \geq 0$).

Let random variable S be the stopping time of the outer iterations as a function of an algorithm parameter $T > 0$:

$$S = \min_{\hat{S}} \left\{ \hat{S} : \sum_{s=1}^{\hat{S}} T_s \geq T \right\}.$$

Notice that SARAH++ maintains a running sum $G = \sum_{j=1}^{s} T_i$ against which parameter T is compared in the stopping criteria of the outer loop.

For the general convex case which supposes Assumption 4 in addition to smoothness we have the next theorem.

Theorem 3 (Smooth general convex) *Suppose that Assumptions 2 and 4 hold. Consider SARAH++ (Algorithm 3) with $\eta \leq \frac{\gamma}{L}$, $0 < \gamma \leq 1$. Then,*

$$\mathbb{E}\left[\frac{1}{T_1 + \cdots + T_S} \sum_{s=1}^{S} \sum_{t=0}^{T_s - 1} \|\nabla F(w_t^{(s)})\|^2 \right] \leq \frac{2}{T\eta} [F(\tilde{w}_0) - F(w_*)].$$

Proof We recall the following definitions. T_s is the stopping time (a random variable) of the s-th outer iteration such that

$$T_s = \min \left\{ \min_{t \geq 0} \left\{ t : \|v_t^{(s)}\|^2 < \gamma \|v_0^{(s)}\|^2 \right\}, m + 1 \right\} \ , \ s = 1, 2, \ldots$$

and S is the stopping time of the outer iterations (a random variable) and such that for some $T > 0$

$$S = \min_{\hat{S}} \left\{ \hat{S} : \sum_{s=1}^{\hat{S}} T_s \geq T \right\}.$$

Note that $T_s \geq 1$ is the first time such that $\|v_{T_s}^{(s)}\|^2 < \gamma \|v_0^{(s)}\|^2$. Hence, for a given T_s, we have $\|v_t^{(s)}\|^2 \geq \gamma \|v_0^{(s)}\|^2$, for $0 \leq t \leq T_s - 1$, and

$$\begin{aligned}
\mathbb{E}[F(w_{T_s}^{(s)}) - F(w_*)] &\leq \mathbb{E}[F(w_{T_s-1}^{(s)}) - F(w_*)] - \frac{\eta}{2} \mathbb{E}[\|\nabla F(w_{T_s-1}^{(s)})\|^2] \\
&\quad + \frac{\eta}{2} \left(L\eta \mathbb{E}[\|v_0^{(s)}\|^2] - \mathbb{E}[\|v_{T_s-1}^{(s)}\|^2] \right) \\
&\overset{\eta \leq \frac{\gamma}{L}}{\leq} \mathbb{E}[F(w_{T_s-1}^{(s)}) - F(w_*)] - \frac{\eta}{2} \mathbb{E}[\|\nabla F(w_{T_s-1}^{(s)})\|^2] \\
&\quad + \frac{\eta}{2} \left(\gamma \mathbb{E}[\|v_0^{(s)}\|^2] - \mathbb{E}[\|v_{T_s-1}^{(s)}\|^2] \right) \\
&\leq \mathbb{E}[F(w_{T_s-1}^{(s)}) - F(w_*)] - \frac{\eta}{2} \mathbb{E}[\|\nabla F(w_{T_s-1}^{(s)})\|^2] \\
&\leq \mathbb{E}[F(w_0^{(s)}) - F(w_*)] - \frac{\eta}{2} \sum_{t=0}^{T_s-1} \mathbb{E}[\|\nabla F(w_t^{(s)})\|^2].
\end{aligned}$$

Since $\tilde{w}_s = w_{T_s}^{(s)}$ and $\tilde{w}_{s-1} = w_0^{(s)}$, for given $T_1, \ldots, T_S$, we have

$$\begin{aligned}\mathbb{E}[F(\tilde{w}_S) - F(w_*)] &\leq \mathbb{E}[F(\tilde{w}_{S-1}) - F(w_*)] - \frac{\eta}{2}\sum_{t=0}^{T_S-1}\mathbb{E}[\|\nabla F(w_t^{(s)})\|^2] \\ &\leq \mathbb{E}[F(\tilde{w}_0) - F(w_*)] - \frac{\eta}{2}\sum_{s=1}^{S}\sum_{t=0}^{T_s-1}\mathbb{E}[\|\nabla F(w_t^{(s)})\|^2].\end{aligned}$$

Since $F(\tilde{w}_S) \geq F(w_*)$, bringing the second term of the RHS to the LHS. For any given $\tilde{w}_0$, we have

$$\frac{\eta}{2}\sum_{s=1}^{S}\sum_{t=0}^{T_s-1}\mathbb{E}[\|\nabla F(w_t^{(s)})\|^2 | T_1, \ldots, T_S] \leq [F(\tilde{w}_0) - F(w_*)].$$

Hence,

$$\begin{aligned}&\frac{1}{T_1 + \cdots + T_S}\sum_{s=1}^{S}\sum_{t=0}^{T_s-1}\mathbb{E}[\|\nabla F(w_t^{(s)})\|^2 | T_1, \ldots, T_S] \\ &\quad\leq \frac{1}{T_1 + \cdots + T_S}\frac{2}{\eta}[F(\tilde{w}_0) - F(w_*)] \\ &\quad\leq \frac{2}{\eta T}[F(\tilde{w}_0) - F(w_*)],\end{aligned}$$

where the last inequality follows since $\sum_{s=1}^{S} T_s \geq T$. Hence, by taking the expectation to both sides, we could achieve the desired result. □

The theorem leads to the next corollary about iteration complexity, i.e., we bound T which is the total number of iterations performed by the inner loop across all outer loop iterations. This is different from the total complexity since T does not separately count the n gradient evaluations when the full gradient is computed in the outer loop.

Corollary 7 (Smooth general convex) *For the conditions in Theorem 3 with $\eta = \mathcal{O}(\frac{1}{L})$, we achieve an ϵ-accurate solution after $\mathcal{O}(\frac{1}{\epsilon})$ inner loop iterations.*

Proof The proof is trivial since we want $\frac{2}{\eta T}[F(\tilde{w}_0) - F(w_*)] = \epsilon$, which requires $T = \frac{2[F(\tilde{w}_0) - F(w_*)]}{\eta} \cdot \frac{1}{\epsilon} = \mathcal{O}(\frac{1}{\epsilon})$ iterations, where we could choose $\eta = \mathcal{O}(\frac{1}{L})$. □

By supposing Assumption 3 in addition to the smoothness and general convexity assumptions, we can prove a linear convergence rate. For strongly convex objective functions we have the following result.

Theorem 4 (Smooth strongly convex) *Suppose that Assumptions 2, 3 and 4 hold. Consider SARAH++ (Algorithm 3) with $\eta \leq \frac{\gamma}{L}$, $0 < \gamma \leq 1$. Then, for the final output $\hat{w}$ of SARAH++, we have*

$$\mathbb{E}[F(\hat{w}) - F(w_*)] \leq (1 - \mu\eta)^T[F(\tilde{w}_0) - F(w_*)]. \tag{27}$$

Proof Following the beginning part of the proof of Theorem 3, we have, for a given T_s,

$$\begin{aligned}\mathbb{E}[F(w_{T_s}^{(s)}) - F(w_*)] &\leq \mathbb{E}[F(w_{T_s-1}^{(s)}) - F(w_*)] - \frac{\eta}{2}\mathbb{E}[\|\nabla F(w_{T_s-1}^{(s)})\|^2] \\ &\overset{(24)}{\leq} (1-\mu\eta)\mathbb{E}[F(w_{T_s-1}^{(s)}) - F(w_*)] \\ &\leq (1-\mu\eta)^{T_s}\mathbb{E}[F(w_0^{(s)}) - F(w_*)]\end{aligned}$$

Since $\tilde{w}_s = w_{T_s}^{(s)}$ and $\tilde{w}_{s-1} = w_0^{(s)}$, for given $T_1, \ldots, T_S$, we have

$$\begin{aligned}\mathbb{E}[F(\hat{w}) - F(w_*)|T_1, \ldots, T_S] &= \mathbb{E}[F(\tilde{w}_S) - F(w_*)|T_1, \ldots, T_S] \\ &\leq (1-\mu\eta)^{T_1+\cdots+T_S}[F(\tilde{w}_0) - F(w_*)] \\ &\leq (1-\mu\eta)^{T}[F(\tilde{w}_0) - F(w_*)],\end{aligned}$$

where the last inequality follows since $\sum_{s=1}^{S} T_s \geq T$. Hence, by taking the expectation to both sides, we could have $\mathbb{E}[F(\hat{w}) - F(w_*)] \leq (1-\mu\eta)^T[F(\tilde{w}_0) - F(w_*)]$. □

This leads to the following iteration complexity.

Corollary 8 (Smooth strongly convex) *For the conditions in Theorem 4 with $\eta = \mathcal{O}(\frac{1}{L})$, we achieve $\mathbb{E}[F(\hat{w}) - F(w_*)] \leq \epsilon$ after $\mathcal{O}(\kappa \log(\frac{1}{\epsilon}))$ total iterations, where $\kappa = L/\mu$ is the condition number.*

Proof We want $(1-\mu\eta)^T[F(\tilde{w}_0) - F(w_*)] = \epsilon$. Hence,

$$T = -\frac{1}{\log(1-\mu\eta)} \log\left(\frac{[F(\tilde{w}_0) - F(w_*)]}{\epsilon}\right).$$

Note that: $-\frac{1}{x} - 1 \leq -\frac{1}{\log(1+x)} \leq -\frac{1}{x}$, $-1 < x < 0$. We can have

$$\left(\frac{1}{\mu\eta} - 1\right) \log\left(\frac{[F(\tilde{w}_0) - F(w_*)]}{\epsilon}\right) \leq T \leq \frac{1}{\mu\eta} \log\left(\frac{[F(\tilde{w}_0) - F(w_*)]}{\epsilon}\right).$$

By choosing $\eta = \mathcal{O}(\frac{1}{L})$, we have $T = \mathcal{O}(\kappa \log(\frac{1}{\epsilon}))$. □

Remark 3 *The proofs of the above results hold for any $m \leq T$. If we choose $m = 0$, then SARAH++ reduces to the Gradient Descent algorithm since the inner "while" loop stops right after updating $w_1^{(s)} = w_0^{(s)} - \eta v_0^{(s)}$. In this case, Corollaries 7 and 8 recover the rate of convergence and complexity of GD.*

In this section, we showed that *SARAH++ has a guarantee of theoretical convergence* (see Theorems 3 and 4) while SARAH+ does not have such a guarantee.

An interesting open question we would like to discuss here is the total complexity of SARAH++. Although we have shown the convergence results of SARAH++ in terms of the iteration complexity, the total complexity which is computed as the total number of evaluations of the component gradient functions still remains an open

question. It is clear that the total complexity must depend on the learning rate η (or γ) – the factor that decides when to stop the inner iterations.

We note that T can be "closely" understood as the total number of updates $w_{t+1}^{(s)}$ of the algorithm. The total complexity is equal to $\sum_{i=1}^{S}(n + 2(T_i - 1))$. For the special case $T_i = 1$, $i = 1, \ldots, S$, the algorithm recovers the GD algorithm with $T = \sum_{i=1}^{S} T_s = S$. Since each full gradient takes n gradient evaluations, the total complexity for this case is equal to $nS = \mathcal{O}(\frac{n}{\epsilon})$ (in the general convex case) and $nS = \mathcal{O}(n\kappa \log(\frac{1}{\epsilon}))$ (in the strongly convex case).

However, it is non-trivial to derive the total complexity of SARAH++ since it should depend on the learning rate η. We leave this question as an open direction for future research.

5 Numerical Experiments

In this section, we provide numerical experiments to show the advantages of NC-SARAH over SPIDER and SpiderBoost on the binary classification problem with non-convex loss function. We also show the advantage of SARAH++ over SARAH on the logistic regression problems with convex loss function. We further propose a practical version called *SARAH Adaptive* which improves the performance of SARAH and SARAH++ for convex problems – numerical experiments on various data sets show good overall performance.

5.1 Non-Convex Case: NC-SARAH

In this subsection, we numerically verify the advantages of NC-SARAH over SPIDER and SpiderBoost. We consider the binary classification problem with non-convex loss function in [28] as follows

$$\min_{w\in\mathbb{R}^d}\left\{F(w) = \frac{1}{n}\sum_{i=1}^{n}\Big[f_i(w) := \log(1+\exp(-y_i x_i^T w)) + \frac{\lambda}{2}\sum_{j=1}^{d}\frac{w_i^2}{1+w_i^2}\Big]\right\}, \tag{28}$$

where $\{x_i, y_i\}_{i=1}^n$ is the training data with $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, +1\}$, $i \in [n]$.

We conducted experiments to demonstrate the advantage in performance of NC-SARAH over SPIDER and SpiderBoost on the popular classification data sets *covtype* ($n = 406,708$ training data; estimated $L \simeq 1.90$), *ijcnn1* ($n = 91,701$ training data; estimated $L \simeq 1.77$), and *w8a* ($n = 49,749$ training data, estimated $L \simeq 7.05$) from LIBSVM [5]. Since we only care about the non-convexity of each f_i, we can simply choose $\lambda = 0.01$.

Figure 1 shows comparisons of the values of $F(w)$, $\|\nabla F(w)\|^2$, and Test Accuracy among NC-SARAH, SPIDER with $\epsilon = 0.1$, SPIDER with $\epsilon = 0.01$, and SpiderBoost. In order to fit Spiderboost's mini-batch size of $b = \sqrt{n}$ we choose $\gamma = 0$ in NC-SARAH and SPIDER. In this scenario, SPIDER with $\epsilon = 0.1$ performs similarly to SpiderBoost. We observe that NC-SARAH has better performance than

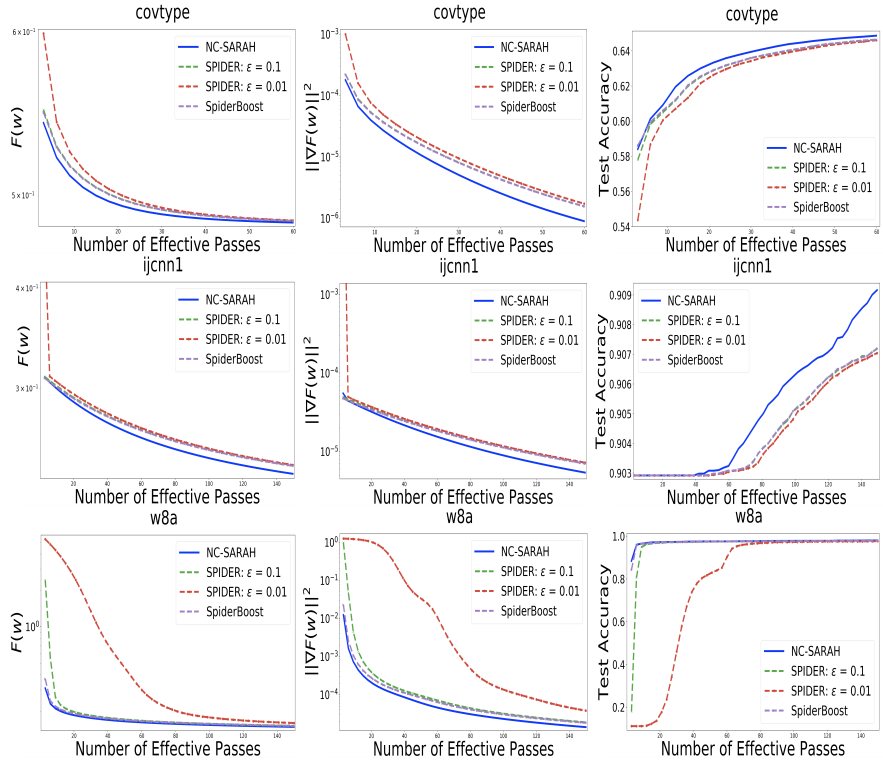

Fig. 1: Comparisons of $F(w)$, $\|\nabla F(w)\|^2$, and Test Accuracy among NC-SARAH, SPIDER with $\epsilon = 0.1$, SPIDER with $\epsilon = 0.01$, and SpiderBoost on *covtype*, *ijcnn1* and *w8a* datasets

both SPIDER and SpiderBoost since NC-SARAH is able to adopt a larger learning rate than those used in SPIDER and SpiderBoost (Corollaries 5 and 6) as shown in Figure 1. We experimented with 10 runs and reported the average results with the same initial point w_0 for all the algorithms.

SpiderBoost only allows a mini-batch size of $b = \sqrt{n}$ while NC-SARAH allows $b = n^{1/2-\gamma}$ with $m = n^{1/2+\gamma}$ for $\gamma \in [0, 0.5]$. Figure 2 shows the sensitivity of γ for NC-SARAH. We observe that the choice of $b = \sqrt{n}$ and $m = \sqrt{n}$ (or equivalently $\gamma = 0$) is not a good choice; for *covtype* $b \in [n^{0.1}, n^{0.2}]$ (or equivalently $\gamma = 0.3, 0.4$) leads to the best performance. This demonstrates that allowing a flexible range of mini-batch sizes beyond $b = \sqrt{n}$ is beneficial.

5.2 Convex Case: SARAH++

The authors in [18] provide experiments showing good overall performance of SARAH over other algorithms such as SGD [24], SAG [25], SVRG [9], etc. For this reason, we provide experiments comparing SARAH++ directly with SARAH. We notice that SARAH (with multiple outer loops) like SARAH++ has theoretical guarantees with sublinear convergence for general convex and linear convergence for strongly convex problems as proved in [18]. Because of these theoretical guarantees (which SARAH+

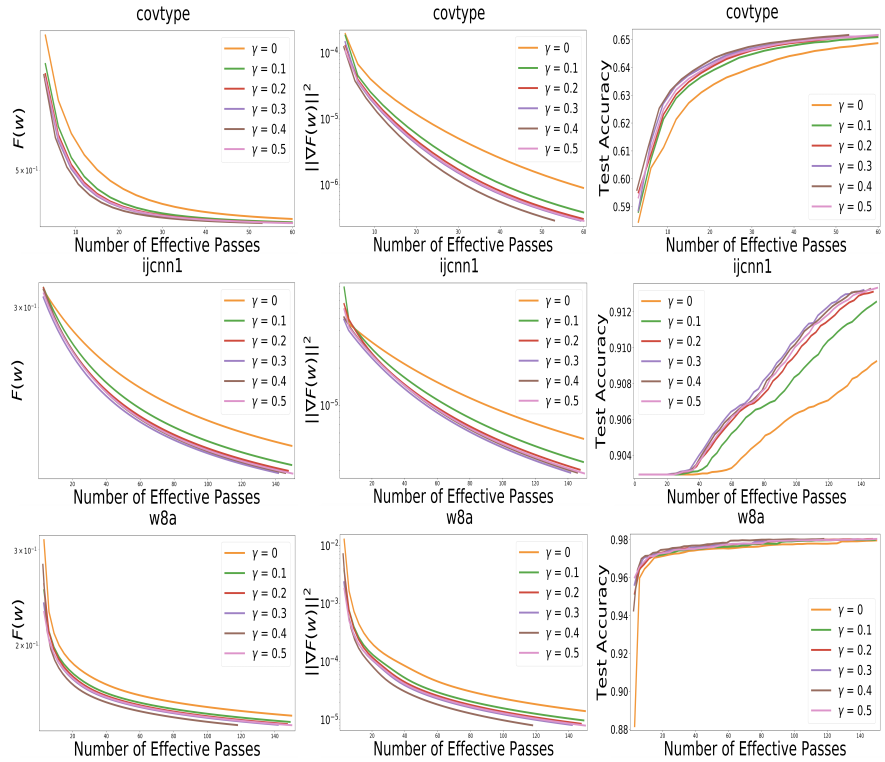

Fig. 2: Comparisons of $F(w)$, $\|\nabla F(w)\|^2$, and Test Accuracy for NC-SARAH with different values of $\gamma = \{0, 0.1, 0.2, 0.3, 0.4, 0.5\}$ on *covtype*, *ijcnn1* and *w8a* datasets

does not have), SARAH itself may already perform well for convex problems and the question is whether SARAH++ offers an advantage.

We consider ℓ_2-regularized logistic regression problems with

$$f_i(w) = \log(1 + \exp(-y_i\langle x_i, w\rangle)) + \tfrac{\lambda}{2}\|w\|^2, \tag{29}$$

where $\{x_i, y_i\}_{i=1}^n$ is the training data and the regularization parameter λ is set to $1/n$, a widely-used value in literature [25, 18]. The condition number is equal to $\kappa = L/\mu = n$. We conducted experiments to demonstrate the advantage in performance of SARAH++ over SARAH for convex problems on popular data sets including *covtype*, *ijcnn1*, *w8a* (introduced in Section 5.1), and *phishing* ($n = 7,738$ training data, estimated $L \simeq 7.49$) from LIBSVM.

Figure 3 shows comparisons between SARAH++ and SARAH for different values of learning rate η. We depicted the value of $\log[F(w) - F(w_*)]$ (i.e. $F(w) - F(w_*)$ in log scale) for the y-axis and "number of effective passes" (or number of epochs, where an epoch is the equivalent of n component gradient evaluations or one full gradient computation) for the x-axis. For SARAH, we choose the outer loop size $S = 10$ and tune the inner loop size $m = \{0.5n, n, 2n, 3n, 4n\}$ to achieve the best performance. The optimal solution w_* of the strongly convex problem in (29) is found by using Gradient Descent with stopping criterion $\|\nabla F(w)\|^2 \leq 10^{-15}$. We

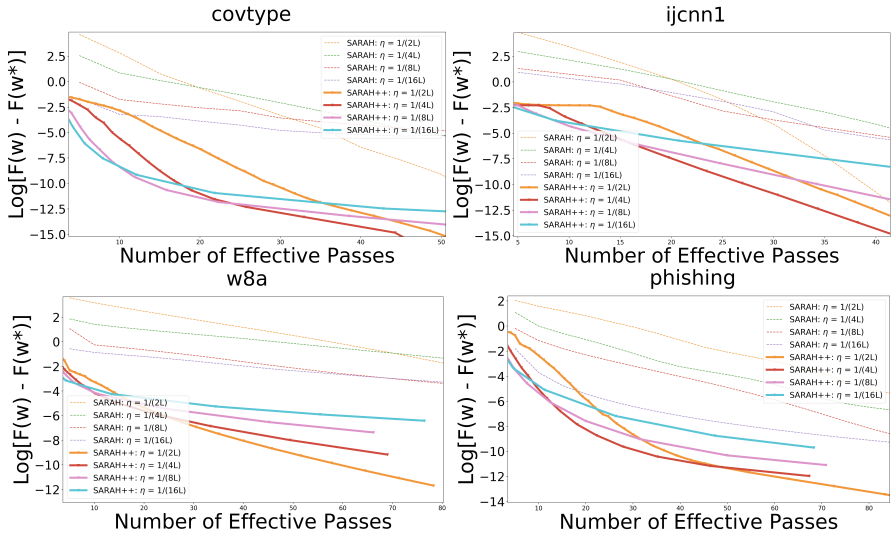

Fig. 3: Comparisons of $\log[F(w) - F(w_*)]$ between SARAH++ and SARAH with different learning rates on *covtype*, *ijcnn1*, *w8a*, and *phishing* datasets

observe that, SARAH++ achieves improved overall performance compared to regular SARAH as shown in Figure 3. From the experiments we see that the stopping criteria $\|v_t^{(s)}\|^2 < \gamma\|v_0^{(s)}\|^2$ ($\gamma = L\eta$) of SARAH++ is indeed important. The stopping criteria helps the inner loop to prevent updating tiny redundant steps.

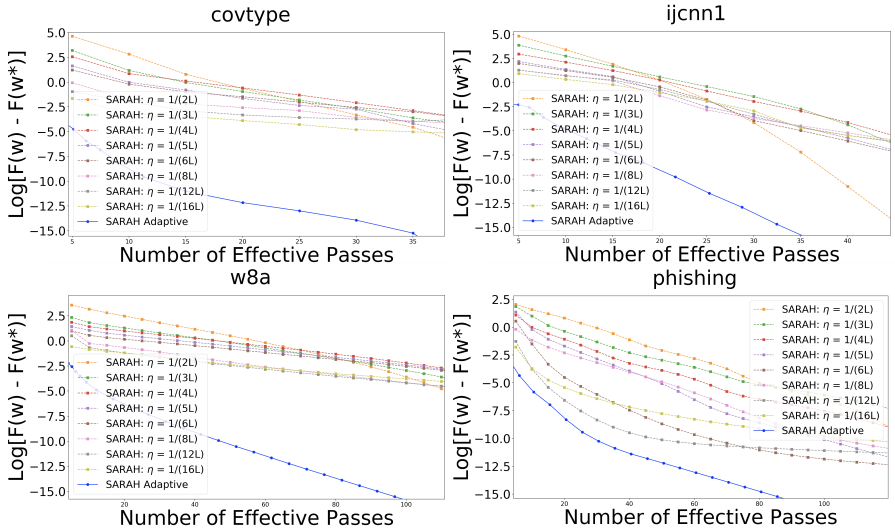

Fig. 4: Comparisons of $\log[F(w) - F(w_*)]$ between SARAH Adaptive and SARAH with different learning rates on *covtype*, *ijcnn1*, *w8a*, and *phishing* datasets

5.3 SARAH Adaptive: A New Practical Variant

We now propose a practical adaptive method which aims to improve performance. Although we do not have any theoretical result for this adaptive method, numerical experiments are very promising and they heuristically show the improved performance on different data sets.

Algorithm 4 SARAH Adaptive

Parameters: The maximum inner loop size m, the outer loop size S, the factor $0 < \gamma \leq 1$.
Initialize: $\tilde{w}_0$
Iterate:
for $s = 1, 2, \ldots, S$ **do**
  $w_0^{(s)} = \tilde{w}_{s-1}$
  $v_0^{(s)} = \frac{1}{n}\sum_{i=1}^n \nabla f_i(w_0^{(s)})$
  $t = 0$
  while $\|v_t^{(s)}\|^2 \geq \gamma \|v_0^{(s)}\|^2$ **and** $t \leq m$ **do**
    $\eta_t^{(s)} = \frac{1}{L} \cdot \frac{\|v_t^{(s)}\|^2}{\|v_0^{(s)}\|^2}$ **(adaptive)**
    $w_{t+1}^{(s)} = w_t^{(s)} - \eta_t^{(s)} v_t^{(s)}$
    $t \leftarrow t + 1$
    if $m \neq 0$ **then**
      Sample i_t uniformly at random from $[n]$
      $v_t^{(s)} = \nabla f_{i_t}(w_t^{(s)}) - \nabla f_{i_t}(w_{t-1}^{(s)}) + v_{t-1}^{(s)}$
    end if
  end while
  Set $\tilde{w}_s = w_t^{(s)}$
end for

The motivation of this algorithm comes from the intuition of Lemma 4 for convex optimization. For a single outer loop with $\eta \leq \frac{1}{L}$, (26) holds for SARAH (Algorithm 1). Hence, for any s, we intentionally choose $\eta = \eta_t^{(s)} = \frac{\|v_t^{(s)}\|^2}{L\|v_0^{(s)}\|^2}$ such that $L\eta\mathbb{E}[\|v_0^{(s)}\|^2] - \mathbb{E}[\|v_t^{(s)}\|^2] = 0$. Since $\|v_t^{(s)}\|^2 \leq \|v_0^{(s)}\|^2$, $t \geq 0$, in [18] for convex problems, we have $\eta_t^{(s)} \leq \frac{1}{L}$, $t \geq 0$. We also stop the inner loop when $\|v_t^{(s)}\|^2 < \gamma\|v_0^{(s)}\|^2$ for some $0 < \gamma \leq 1$. SARAH Adaptive is given in detail in Algorithm 4 without convergence analysis.

We have conducted numerical experiments on the same datasets and problems as introduced in the previous subsection. Figures 4 and 5 show the comparison between SARAH Adaptive and SARAH and SARAH++ for different values of η. We observe that SARAH Adaptive has an improved performance over SARAH and SARAH++ (without tuning learning rate). In Figure 6 we present the numerical performance of SARAH Adaptive for different values of $\gamma = \left\{\frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \frac{1}{6}, \frac{1}{8}, \frac{1}{10}, \frac{1}{12}, \frac{1}{16}\right\}$.

6 Conclusion and Future Research

In this paper, we address almost important open problems for the original SARAH algorithm, i.e., SARAH for convex and non-convex optimization problems. For non-convex optimization, we propose NC-SARAH, which achieves the state-of-the-art

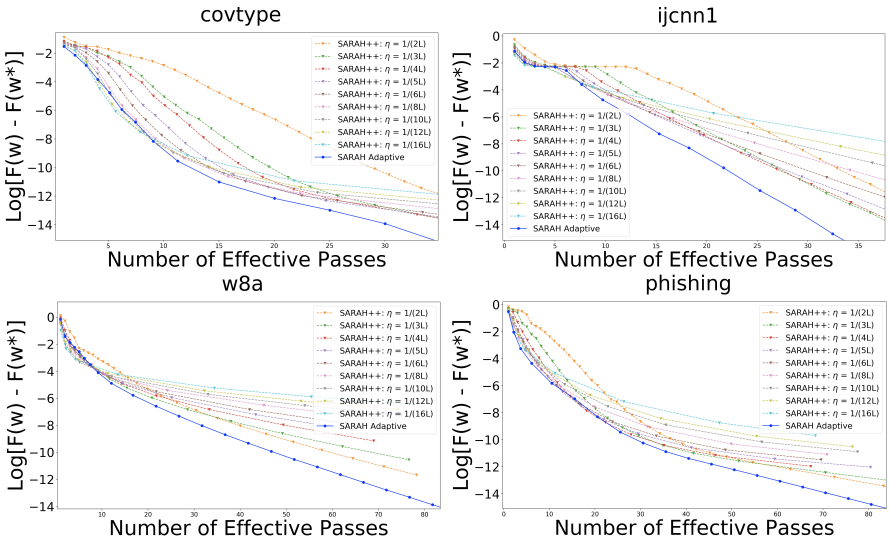

Fig. 5: Comparisons of $\log[F(w) - F(w_*)]$ between SARAH Adaptive and SARAH++ with different learning rates on *covtype*, *ijcnn1*, *w8a*, and *phishing* datasets

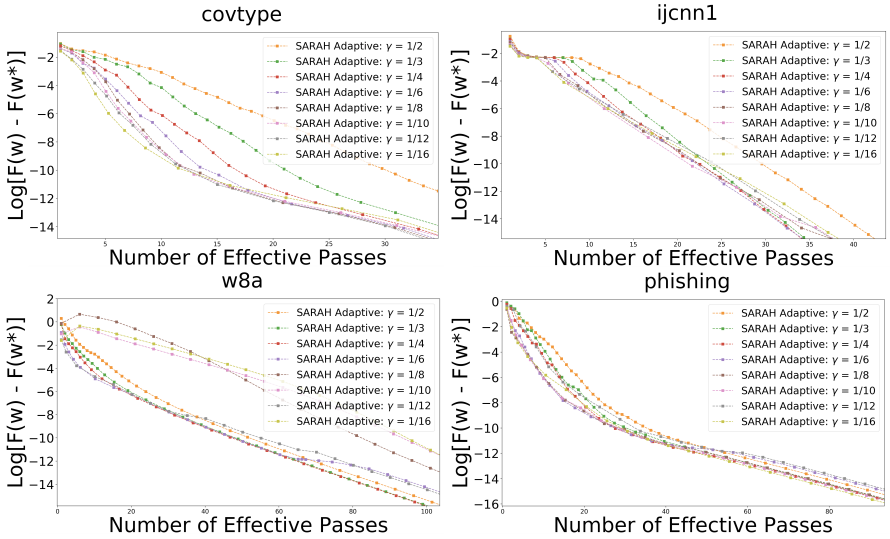

Fig. 6: Comparisons of $\log[F(w) - F(w_*)]$ with different value of γ for SARAH Adaptive on *covtype*, *ijcnn1*, *w8a*, and *phishing* datasets

asymptotic total complexity for finding a first-order stationary point based on only the average smooth assumption as SPIDER and SpiderBoost. The total complexity matches the lower-bound worst case complexity. Indeed, NC-SARAH has advantages over SPIDER and SpiderBoost in both theory and practice. That is, NC-SARAH allows larger learning rates as well as a range of mini-batch sizes. Numerical experiments show how NC-SARAH outperforms SPIDER and SpiderBoost. In theory, our proof is significantly simpler and more intuitive. Moreover, we showed the promis-

ing numerical results for SARAH++ and SARAH Adaptive in the convex case. The total complexity of SARAH++ and the convergence analysis of SARAH Adaptive could be potential for future research. More general optimization problems (e.g. in [21,22]) can be considered in the convex cases. In addition, we show that SARAH (NC-SARAH and SARAH++) can be reduced to Gradient Descent - an open problem since 2012 - which may motivate new research directions.

Data Availability Statement

The authors confirm that all data used in this paper are publicly available in `https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/` [5].

References

1. Allen-Zhu, Z.: Natasha: Faster non-convex stochastic optimization via strongly non-convex parameter. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70, pp. 89–97. JMLR. org (2017)
2. Allen-Zhu, Z.: Natasha 2: Faster non-convex optimization than sgd. In: Advances in Neural Information Processing Systems, pp. 2675–2686 (2018)
3. Allen-Zhu, Z., Yuan, Y.: Improved SVRG for Non-Strongly-Convex or Sum-of-Non-Convex Objectives. In: ICML, pp. 1080–1089 (2016)
4. Bottou, L., Curtis, F.E., Nocedal, J.: Optimization Methods for Large-Scale Machine Learning. SIAM Rev. **60**(2), 223–311 (2018)
5. Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology **2**, 27:1–27:27 (2011)
6. Defazio, A., Bach, F., Lacoste-Julien, S.: Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In: Advances in Neural Information Processing Systems, pp. 1646–1654 (2014)
7. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. Journal of Machine Learning Research **12**, 2121–2159 (2011)
8. Fang, C., Li, C.J., Lin, Z., Zhang, T.: Spider: Near-optimal non-convex optimization via stochastic path-integrated differential estimator. In: Advances in Neural Information Processing Systems, pp. 689–699 (2018)
9. Johnson, R., Zhang, T.: Accelerating stochastic gradient descent using predictive variance reduction. In: Advances in Neural Information Processing Systems, pp. 315–323 (2013)
10. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. CoRR **abs/1412.6980** (2014)
11. Konečnỳ, J., Richtárik, P.: Semi-stochastic gradient descent methods. Frontiers in Applied Mathematics and Statistics **3**, 9 (2017)
12. Lei, L., Ju, C., Chen, J., Jordan, M.I.: Non-convex finite-sum optimization via SCSG methods. In: I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (eds.) Advances in Neural Information Processing Systems 30, pp. 2348–2358. Curran Associates, Inc. (2017)
13. Li, Z., Bao, H., Zhang, X., Richtárik, P.: Page: A simple and optimal probabilistic gradient estimator for nonconvex optimization. In: International Conference on Machine Learning, pp. 6286–6295. PMLR (2021)
14. Liu, Y., Feng, F., Yin, W.: Acceleration of svrg and katyusha x by inexact preconditioning. In: International Conference on Machine Learning, pp. 4003–4012. PMLR (2019)
15. Mairal, J.: Optimization with first-order surrogate functions. In: International Conference on Machine Learning, pp. 783–791 (2013)
16. Nesterov, Y.: Introductory lectures on convex optimization : a basic course. Applied optimization. Kluwer Academic Publ., Boston, Dordrecht, London (2004)
17. Nguyen, L., Nguyen, P.H., van Dijk, M., Richtarik, P., Scheinberg, K., Takac, M.: SGD and Hogwild! convergence without the bounded gradients assumption. In: Proceedings of the 35th International Conference on Machine Learning-Volume 80, pp. 3747–3755 (2018)

18. Nguyen, L.M., Liu, J., Scheinberg, K., Takáč, M.: SARAH: A novel method for machine learning problems using stochastic recursive gradient. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70, pp. 2613–2621. JMLR. org (2017)
19. Nguyen, L.M., Liu, J., Scheinberg, K., Takác, M.: Stochastic recursive gradient algorithm for nonconvex optimization. CoRR **abs/1705.07261** (2017)
20. Nguyen, L.M., Nguyen, P.H., Richtárik, P., Scheinberg, K., Takáč, M., van Dijk, M.: New convergence aspects of stochastic gradient algorithms. Journal of Machine Learning Research **20**(176), 1–49 (2019). URL `http://jmlr.org/papers/v20/18-759.html`
21. Nguyen, L.M., Scheinberg, K., Takac, M.: Inexact sarah algorithm for stochastic optimization. Optimization Methods and Software **0**(0), 1–22 (2020). DOI 10.1080/10556788.2020.1818081
22. Pham, N.H., Nguyen, L.M., Phan, D.T., Tran-Dinh, Q.: Proxsarah: An efficient algorithmic framework for stochastic composite nonconvex optimization. J. Mach. Learn. Res. **21**, 110–1 (2020)
23. Reddi, S.J., Hefny, A., Sra, S., Poczos, B., Smola, A.: Stochastic variance reduction for nonconvex optimization. In: International conference on machine learning, pp. 314–323 (2016)
24. Robbins, H., Monro, S.: A stochastic approximation method. The Annals of Mathematical Statistics **22**(3), 400–407 (1951)
25. Roux, N.L., Schmidt, M., Bach, F.R.: A stochastic gradient method with an exponential convergence _rate for finite training sets. In: Advances in Neural Information Processing Systems, pp. 2663–2671 (2012)
26. Schmidt, M., Le Roux, N., Bach, F.: Minimizing finite sums with the stochastic average gradient. Mathematical Programming pp. 1–30 (2016)
27. Shalev-Shwartz, S., Zhang, T.: Stochastic dual coordinate ascent methods for regularized loss. Journal of Machine Learning Research **14**(1), 567–599 (2013)
28. Wang, Z., Ji, K., Zhou, Y., Liang, Y., Tarokh, V.: Spiderboost: A class of faster variance-reduced algorithms for nonconvex optimization. Advances in Neural Information Processing Systems (2019)
29. Zhou, D., Xu, P., Gu, Q.: Stochastic nested variance reduced gradient descent for nonconvex optimization. In: Advances in Neural Information Processing Systems, pp. 3921–3932 (2018)