A Hybrid Stochastic Optimization Framework for Stochastic Composite Nonconvex Optimization

Quoc Tran-Dinh, Nhan H. Pham, Dzung T. Phan, and Lam M. Nguyen

July 10, 2019

Abstract

In this paper, we introduce a new approach to develop stochastic optimization algorithms for solving stochastic composite and possibly nonconvex optimization problems. The main idea is to combine two stochastic estimators to form a new hybrid one. We first introduce our hybrid estimator and then investigate its fundamental properties to form a foundation theory for algorithmic development. Next, we apply our theory to develop several variants of stochastic gradient methods to solve both expectation and finite-sum composite optimization problems. Our first algorithm can be viewed as a variant of proximal stochastic gradient methods with a single-loop, but can achieve \( O(\sigma^3\varepsilon^{-1} + \sigma\varepsilon^{-3}) \) complexity bound that is significantly better than the \( O(\sigma^2\varepsilon^{-4}) \)-complexity in state-of-the-art stochastic gradient methods, where \( \sigma \) is the variance and \( \varepsilon \) is a desired accuracy. Then, we consider two different variants of our method: adaptive step-size and double-loop schemes that have the same theoretical guarantees as in our first algorithm. We also study two mini-batch variants and develop two hybrid SARAH-SVRG algorithms to solve the finite-sum problems. In all cases, we achieve the best-known complexity bounds under standard assumptions. We test our methods on several numerical examples with real datasets and compare them with state-of-the-arts. Our numerical experiments show that the new methods are comparable and, in many cases, outperform their competitors.

1 Introduction

In this paper, we consider the following stochastic composite and possibly nonconvex optimization problem, which is widely studied in the literature:

\[
\min_{x \in \mathbb{R}^p} \left\{ F(x) := f(x) + \psi(x) \equiv \mathbb{E}_{\xi \sim \Omega} [f_\xi(x)] + \psi(x) \right\},
\]

where \( f_\xi(\cdot) : \mathbb{R}^p \times \Omega \rightarrow \mathbb{R} \) is a stochastic function such that for each \( x \in \mathbb{R}^p \), \( f_\xi(x) \) is a random variable in a given probability space \( (\Omega, \mathcal{F}, P) \), while for each realization \( \xi \in \Omega \), \( f_\xi(\cdot) \) is smooth on \( \mathbb{R}^p \); \( f(x) := \mathbb{E}_{\xi \sim \Omega} [f_\xi(x)] = \int_\Omega f_\xi(x) dP(\xi) \) is the expectation of the random function \( f_\xi(x) \) over \( \xi \) on \( \Omega \); and \( \psi : \mathbb{R}^p \rightarrow \mathbb{R} \cup \{+\infty\} \) is a proper, closed, and convex function.

In addition to (1), we also consider the following composite finite-sum problem:

\[
\min_{x \in \mathbb{R}^p} \left\{ F(x) := f(x) + \psi(x) \equiv \frac{1}{n} \sum_{i=1}^n f_i(x) + \psi(x) \right\},
\]

where \( f_i : \mathbb{R}^p \rightarrow \mathbb{R} \) for \( i = 1, \ldots, n \) are all smooth functions. Problem (2) can be considered as special case of (1) where \( \Omega := \{1, 2, \ldots, n\} \) and \( \mathcal{F} \) is a uniform distribution on \( \Omega \). If \( n \) is extremely large such that evaluating the full gradient \( \nabla f(x) \) and the function value \( f(x) \) is expensive, then, as usual, we refer to this setting as online models.

If the regularizer \( \psi \) is absent, then we obtain a smooth problem which has been widely studied in the literature. As another special case, if \( \psi \) is the indicator of a nonempty, closed, and convex set \( X \), i.e. \( \psi(\cdot) := \delta_X(\cdot) \), then (1) also covers constrained nonconvex optimization problems.
1.1 Our goals, approach, and contribution

Our goals: Our objective is to develop a new approach to approximate a stationary point of (1) and its finite-sum setting (2) under standard assumptions used in existing methods. In this paper, we only focus on stochastic gradient descent-type (SGD) variants. We are also interested in both oracle complexity bounds and implementation aspects. The ultimate goal is to design simple algorithms that are easy to implement and require less parameter tuning effort.

Our approach: Our approach relies on a so-called “hybrid” idea which merges two existing stochastic estimators through a convex combination to design a “hybrid” offspring that inherits the advantages of its underlying estimators. We will focus on the hybrid estimators formed from the SARAH (a recursive stochastic estimator) introduced in [50] and any given unbiased estimator such as SGD [62], SVRG [32], or SAGA [17]. For the sake of presentation, we only focus on either SGD or SVRG estimator in this paper. We emphasize that our method is fundamentally different from momentum or exponential moving average-type methods such as in [15, 34] where we use two independent estimators instead of a combination of the past and the current estimators.

While our hybrid estimators are biased, fortunately, they provide some useful properties to develop new algorithms. One important feature is the variance reduced property which often allows us to derive a large step-size or a constant step-size in stochastic methods. Whereas a majority of stochastic algorithms rely on unbiased estimators such as SGD, SVRG, and SAGA, interestingly, recent evidence has shown that biased estimators such as SARAH, biased SAGA, or biased SVRG estimators also provide comparable and even better algorithms in terms of oracle complexity bounds as well as empirical performance, see, e.g. [19, 22, 53, 56, 68].

Our approach, on the one hand, can be extended to study second-order methods such as cubic regularization and subsampled schemes as in [8, 21, 63, 71, 75]. The main idea is to exploit hybrid estimators to approximate both gradient and Hessian of the objective function similar to [8, 71, 75]. On the other hand, it can be applied to approximate a second-order stationary point of (1) and (2). The idea is to integrate our methods with a negative curvature search such as Oja’s algorithm [55] or Neon2 [6], or to employ perturbed/noise gradient techniques such as [23, 26, 38] in order to approximate a second-order stationary point. However, to avoid overloading this paper, we leave these extensions for our future work.

Our contribution: To this end, the contribution of this paper can be summarized as follows:

(a) We first introduce a “hybrid” approach to merge two existing stochastic estimators in order to form a new one. Such a new estimator can be viewed as a convex combination of a biased estimator and an unbiased one to inherit the advantages of its underlying estimators. Although we only focus on a convex combination between SARAH [50] and either SGD [62] or SVRG [32] estimator, our approach can be extended to cover other possibilities. Given such new hybrid estimators, we develop several fundamental properties that can be useful for developing new stochastic optimization algorithms.

(b) Next, we employ our new hybrid SARAH-SGD estimator to develop a novel stochastic proximal gradient algorithm, Algorithm 1 to solve (1). This algorithm can achieve $O(\sigma^3\varepsilon^{-1} + \sigma\varepsilon^{-3})$-oracle complexity bound. To the best of our knowledge, this is the first variant of SGD that achieves such an oracle complexity bound without using double loop or check-points as in SVRG or SARAH, or requiring an $n \times p$-table to store gradient components as in SAGA-type methods.

(c) Then, we derive two different variants of Algorithm 1: adaptive step-size and double-loop schemes. Both variants have the same complexity as of Algorithm 1. We also propose a mini-batch variant of Algorithm 1 and provide a trade-off analysis between mini-batch sizes and the choice of step-sizes to obtain better practical performance.

(d) Finally, we design a hybrid SARAH-SVRG estimator and use it to develop new stochastic variants for solving the composite finite-sum problem (2). These variants also achieve the best-known complexity bounds while having new properties compared to existing methods.

Let us emphasize the following additional points of our contribution. Firstly, the new algo-
Algorithm 1 is rather different from existing SGD methods. It first forms a mini-batch stochastic gradient estimator at a given initial point to provide a good approximation to the initial gradient of $f$. Then, it performs a single loop to update the iterate sequence which consists of two steps: proximal-gradient step and averaging step, where our hybrid estimator is used.

Secondly, our methods work with both single-sample and mini-batches, and achieve the best-known complexity bounds in both cases. This is different from some existing methods such as SVRG-type and SpiderBoost that only achieve the best complexity under certain choices of parameters. Our methods are also flexible to choose different mini-batch sizes for the hybrid components to achieve different complexity bounds and to adjust the performance. For instance, in Algorithm 1, we can choose single sample in the SARAH estimator while using a mini-batch in the SGD estimator that leads to different trade-off on the choice of the weight.

Finally, our theoretical results on hybrid estimators are also self-contained and independent. As we have mentioned, they can be used to develop other stochastic algorithms such as second-order methods or perturbed SGD schemes. We believe that they can also be used in other problems such as composition and constrained optimization [16, 44, 67].

1.2 Related work

Problem (1) and its sample averaging setting (2) have been widely studied in the literature for both convex and nonconvex models, see, e.g. [9, 10, 17, 21, 22, 46, 62, 64]. However, due to applications in deep learning, large-scale nonconvex optimization problems have attracted huge attention in recent years [30, 37]. Numerical methods for solving these problems heavily rely on two approaches: deterministic and stochastic approaches, ranging from first-order to second-order methods. Notable first-order methods include stochastic gradient descent-type, conditional gradient descent [59], and primal-dual schemes [13]. In contrast, advanced second-order methods consist of quasi-Newton, trust-region, sketching Newton, subsampled Newton, and cubic regularized Newton-based methods, see, e.g. [11, 48, 57, 63].

In terms of stochastic first-order algorithms, there has been a tremendously increasing trend in stochastic gradient descent methods and their variants in the last fifteen years. SGD-based algorithms can be classified into two categories: non-variance reduction and variance reduction schemes. The classical SGD method was studied in early work of Robbins & Monro [62], but its convergence rate was then investigated in [46] under new robust variants. Ghadimi & Lan extended SGD to nonconvex settings and analyzed its complexity in [27]. Other extensions of SGD can be found in the literature, including [4, 16, 20, 25, 28, 31, 34, 35, 45, 51, 58].

Alternatively, variance reduction-based methods have been intensively studied in recent years for both convex and nonconvex settings. Apart from mini-batch and importance sampling schemes [29, 73], the following methods are the most notable. The first class of algorithms is based on SAG estimator [64], including SAGA-variants [17]. The second one is SVRG [32] and its variants such as Katyusha [3], MiG [77], and many others [29, 60]. The third class relies on SARAH [50] such as SPIDER [22], SpiderBoost [68], ProxSARAH [56], and momentum variants [75]. Other approaches such as Catalyst [41] and SDCA [65] have also been proposed.

In terms of theory, many researchers have focussed on theoretical aspects of existing algorithms. For example, [27] appeared as one of the first works studying convergence rates of stochastic gradient descent-type methods for nonconvex and non-composite finite-sum problems. They later extended it to the composite setting in [29]. The authors of [65] also investigated the gradient dominant case, and [33] considered both finite-sum and composite finite-sum problems under different assumptions. Whereas many researchers have been trying to improve complexity upper bounds of stochastic first-order methods using different techniques [5, 6, 7, 22], other works have attempted to construct examples to establish lower-bound complexity barriers. The upper oracle complexity bounds have been substantially improved among these works and some results have matched the lower bound complexity in both convex and nonconvex settings [5, 1, 22, 29, 69, 50, 56, 60, 68, 76]. We refer to Table 1 for some notable examples of stochastic gradient-type methods for solving (1) and (2) and their non-composite settings.
Let us compare these methods in detail as follows: other algorithms such as Natasha [4] or Natasha1.5 [5] even have three loops. While numerical stochastic algorithms for solving the non-composite setting, i.e. $\psi = 0$, are well-developed and have received considerable attention [5, 6, 7, 22, 10, 52, 53, 54, 60, 76], methods for composite setting remain limited [60, 68]. In this paper, we will develop a novel hybrid approach to design stochastic optimization algorithms for solving the composite problems [1] and [2]. Our approach is rather different from existing ones and we call it a “hybrid” approach.

### 1.3 Comparison

Let us compare our algorithms and existing methods in the following aspects:

**Single-loop vs. multiple-loop:** As mentioned, we aim at developing practical methods that are easy to implement. One of the major differences between our methods and existing state-of-the-art methods is the algorithmic style: single-loop vs. multiple-loop style. As discussed in several works, including [36], single-loop methods have some advantages over double-loop methods, including tuning parameters. The single-loop style consists of SGD, SAGA, and their variants [17, 18, 27, 63, 62, 64], while the double-loop style comprises SVRG, SARAH, and their variants [32, 50]. Other algorithms such as Natasha [4] or Natasha1.5 [5] even have three loops. Let us compare these methods in detail as follows:

- SGD and SAGA-type methods have single-loop, but SAGA-type algorithms use an $n \times p$ matrix to maintain $n$ individual gradients which can be very large if $n$ and $p$ are large. In

### Algorithms Expectation Finite-sum Composite Type

| Algorithms | Expectation | Finite-sum | Composite | Type |
|------------|-------------|------------|-----------|------|
| GD [19]   | NA          | $O\left(n\varepsilon^{-4}\right)$ | NA        | Single |
| SGD [22]  | $O\left(n^2\varepsilon^{-2}\right)$ | NA        | NA        | Single |
| SAGA [60] | NA          | $O\left(n + n^2/3\varepsilon^{-2}\right)$ | NA        | Single* |
| SVRG [60] | NA          | $O\left(n + n^2/3\varepsilon^{-2}\right)$ | NA        | Single |
| SVRG+ [39] | $O\left(n + n^2/3\varepsilon^{-2}\right)$ | NA        | NA        | Single |
| SCSG [40] | $O\left(n^2\varepsilon^{-2} + \varepsilon^{-10/3}\right)$ | $O\left(n + n^2/3\varepsilon^{-2}\right)$ | X         | Double |
| SNVRG [76] | $O\left(n + n^2/3\varepsilon^{-2}\right)$ | $O\left(n + n^2/3\varepsilon^{-2}\right)$ | X         | Double |
| SPIDER [22] | $O\left(n^2\varepsilon^{-2} + \varepsilon^{-3}\right)$ | $O\left(n + n^2/3\varepsilon^{-2}\right)$ | X         | Double |
| SpiderBoost [65] | $O\left(n^2\varepsilon^{-2} + \varepsilon^{-3}\right)$ | $O\left(n + n^2/3\varepsilon^{-2}\right)$ | NA        | Double |
| ProxSARAH [50] | $O\left(n + n^2/3\varepsilon^{-2}\right)$ | $O\left(n + n^2/3\varepsilon^{-2}\right)$ | NA        | Double |
| HybridSGD (This paper) | $O\left(n^2\varepsilon^{-2} + \varepsilon^{-3}\right)$ | $O\left(n + n^2/3\varepsilon^{-2}\right)$ | NA        | Single |

Table 1: A comparison of stochastic first-order oracle complexity bounds and the type of algorithms for nonsmooth nonconvex optimization (both non-composite and composite case). Here, $n$ is the number of data points and $\sigma$ is the variance in Assumption 2.1, and “single/double” means that the algorithm uses single-loop or double-loop, respectively. All the complexity bounds here must depend on the Lipschitz constant $L$ in Assumption 2.2 and $F(x_0) - F^*$, the difference between the initial objective value $F(x_0)$ and the lower-bound $F^*$ in Assumption 2.4. We assume that $L = O(1)$ and ignore these quantities in the complexity bounds. Note that SAGA is a single-loop method, but it requires a matrix of size $n \times p$ to store stochastic gradients ($\star$).
addition, SAGA has not yet been applied to solve (1). Our first algorithm, Algorithm 1 has single-loop as SGD and SAGA, and does not require heavy memory storage. However, to apply to (2), it still requires either an additional assumption or a check-point compared to SAGA. But if it solves (1), then it has the same assumptions as in SGD. In terms of complexity, Algorithm 1 is much better than SGD. To the best of our knowledge, Algorithm 1 is the first single-loop SGD variant that achieves the best-known complexity. Another related work is [15], which uses momentum approach, but requires additional bounded gradient assumption to achieve similar complexity as Algorithm 1.

- Algorithm 2 has double-loop as SVRG and SARAH-type methods. While the double-loop in SVRG, SARAH, and their variants are required to achieve convergence, it is optional in Algorithm 2. Note that double-loop or multiple-loop methods require to tune more parameters such as epoch lengths and possibly the mini-batch size of the snapshot points. Although Algorithm 3 is loopless, it can be viewed as a double-loop variant. This algorithm has different complexity bound than existing methods.

**Single-sample and mini-batch:** Our methods work with both single-sample and mini-batch, and in both cases, they achieve the best-known complexity bounds. This is different from some existing methods such as SVRG or SARAH-based methods [22, 56] where the best complexity is only obtained if one chooses the best parameter configuration.

**Complexity bounds:** Algorithm 1 and its variants all achieve the best-known complexity bounds as in [22, 56] for solving (1). In early work such as Natasha [1] and Natasha1.5 [5] which are based on the SVRG estimator, the best complexity is often $O\left(\sigma^2\epsilon^{-2} + \sigma\epsilon^{-10/3}\right)$ for solving (1) and $O\left(n + n^{2/3}\epsilon^{-2}\right)$ for solving (2). By combining with additional sophisticated tricks, these complexity bounds are slightly improved. For instance, Natasha [1] or Natasha1.5 [5] can achieve $O\left(n + \frac{n^{2/3}}{\epsilon^2}\right)$ in the finite-sum case, and $O\left(\frac{1}{\epsilon^2} + \frac{\sigma^{1/3}}{\epsilon^{2/3}}\right)$ in the expectation case, but they require three loops with several parameter adjustment which are difficult to tune in practice. SNVRG [76] exploits a dynamic epoch length as used in [40] to improve its complexity bounds. Again, this method also requires complicated parameter selection procedure. To achieve better complexity bounds, SARAH-based methods have been studied in [22, 53, 56, 68]. Their complexity meets the lower-bound one in the finite-sum case as indicated in [22, 56].

### 1.4 Paper organization

The rest of this paper is organized as follows. Section 2 discusses the main assumptions of our problems (1) and (2), and their optimality conditions. Section 3 develops new hybrid stochastic estimators and investigates their properties. We consider both single-sample and mini-batch cases. Section 4 studies a new class of hybrid gradient algorithms to solve both (1) and (2). We develop three different variants of hybrid algorithms and analyze their convergence and complexity estimates. Section 5 extends our algorithms to mini-batch cases. Section 6 is devoted to investigating hybrid SARAH-SVRG methods to solve the finite-sum problem (2). Section 7 gives several numerical examples and compares our methods with existing state-of-the-arts. For the sake of presentation, all technical proofs are provided in the appendix.

### 2 Basic assumptions and optimality condition

**Notation and basic concepts:** We work with the Euclidean spaces, $\mathbb{R}^p$ and $\mathbb{R}^n$ equipped with standard inner product $\langle \cdot, \cdot \rangle$ and norm $\| \cdot \|$. For any function $f$, $\text{dom}(f) := \{ x \in \mathbb{R}^p \mid f(x) < +\infty \}$ denotes the effective domain of $f$. If $f$ is continuously differentiable, then $\nabla f$ denotes its gradient. If, in addition, $f$ is twice continuously differentiable, then $\nabla^2 f$ denotes its Hessian.

For a stochastic function $f_\xi$, defined on a probability space $(\Omega, \mathcal{F})$, we use $\mathbb{E}_\xi \left[ f_\xi \right] := \mathbb{E}_{\xi \sim \Omega} \left[ f_\xi \right]$ to denote the expectation of $f_\xi$ w.r.t. $\xi$ on $\Omega$. We also overload the notation $\mathbb{E}_{\xi \sim \Omega} \left[ \cdot \right]$ to express the expectation w.r.t. a realization $\xi_\ell$ in both single-sample and mini-batch cases. Given a finite set $S_m := \{ s_1, \cdots, s_m \}$, we denote $s \sim \mathbb{U}(S_m)$ if $P(s = s_i) = p_i$, for $p_i > 0$ and $\sum_{i=1}^m p_i = 1$. If $p_i = \frac{1}{m}$ for $i = 1, \cdots, m$, then we write $s \sim \mathbb{U}(S_m)$ by dropping the probability distribution $\mathbf{p}$. 

Given a random mapping $G : \mathbb{R}^p \times \Omega \to \mathbb{R}^q$ depending on a random vector $\xi \in \Omega$, we say that $G$ is $L$-average Lipschitz continuous if $\mathbb{E}_\xi \left[ \|G(x) - G(y)\|^2 \right] \leq L^2 \|x - y\|^2$ for all $x, y$, where $L \in (0, +\infty)$ is called the Lipschitz constant of $G$. If $G$ is a deterministic function, then this condition becomes $\|G(x) - G(y)\| \leq L \|x - y\|$ which states that $G$ is $L$-Lipschitz continuous. In particular, if this condition holds for $G = \nabla f$, then we say that $f$ is $L$-smooth.

For a proper, closed, and convex function $\psi : \mathbb{R}^p \to \mathbb{R} \cup \{+\infty\}$, $\partial \psi(x) := \{w \in \mathbb{R}^p \mid \psi(y) \geq \psi(x) + \langle w, y - x \rangle, \ \forall y \in \text{dom}(\psi)\}$ denotes its subdifferential at $x$, and $\text{prox}_\psi(x) := \arg\min \{\psi(x) + \frac{\eta}{2}\|u - x\|^2 \mid u \in \mathbb{R}^p\}$ denotes its proximal operator. If $\psi$ is the indicator of a nonempty, closed, and convex set $\mathcal{X}$, then $\text{prox}_\psi(x)$ reduces to the projection $\text{proj}_\mathcal{X}$ onto $\mathcal{X}$. We say that $\psi$ is $\nu$-weakly convex if $\psi(y) \geq \psi(x) + \langle \nabla \psi(x), y - x \rangle - \frac{\nu}{2}\|y - x\|^2$ for all $x, y \in \text{dom}(\psi)$ and $\nabla \psi(x) \in \partial \psi(x)$, where $\nu \geq 0$ is a given constant. Clearly, a weakly convex function is not necessarily convex. However, any $L$-smooth function is $L$-weakly convex. If $\psi$ is $\nu$-weakly convex, then $\psi(\cdot) + \frac{\eta}{2}\|\cdot\|^2$ is $(\eta - \nu)$-strongly convex if $\eta > \nu$. Therefore, the proximal operator $\text{prox}_{\eta \psi}$ is well-defined and single-valued if $\eta > \nu$. Note that $\text{prox}_{\eta \psi}$ is non-expansive, i.e. $\|\text{prox}_{\eta \psi}(x) - \text{prox}_{\eta \psi}(y)\| \leq \|x - y\|$ for all $x, y \in \text{dom}(\psi)$.

If $f$ is a matrix, then $\|f\|$ is the spectral norm of $f$ and the inner product of two matrices $x$ and $y$ is defined as $(x, y) := \text{trace}(x^\top y)$. Also, $\mathbb{N}_+$ stands for the set of positive integer numbers, and $[n] := \{1, 2, \cdots, n\}$. Given $a \in \mathbb{R}$, $\lfloor a \rfloor$ denotes the maximum integer number that is less than or equal to $a$. We also use $\mathcal{O}(\cdot)$ to express complexity bounds of algorithms.

### 2.1 Fundamental assumptions

Our algorithms developed in the sequel rely on the following fundamental assumptions:

**Assumption 2.1.** Both problems (1) and (2) satisfy the following conditions:

(a) **(Convexity of the regularizer)** $\psi : \mathbb{R}^p \to \mathbb{R} \cup \{+\infty\}$ is a proper, closed, and convex function. The domain $\text{dom}(F) := \text{dom}(f) \cap \text{dom}(g)$ is nonempty.

(b) **(Boundedness from below)** There exists a finite lower bound

$$F^* := \inf_{x \in \mathbb{R}^p} \left\{ F(x) := f(x) + \psi(x) \right\} > -\infty.$$  

(3)

This assumption is fundamental and required for any algorithm. Here, since $\psi$ is proper, closed, and convex, its proximal operator $\text{prox}_{\eta \psi}(\cdot)$ is well-defined, single-valued, and non-expansive. We assume that this proximal operator can be computed exactly.

**Assumption 2.2 (L-average smoothness).** The expectation function $f(\cdot)$ in (1) is $L$-smooth on $\text{dom}(F)$, i.e. there exists $L \in (0, +\infty)$ such that

$$\mathbb{E}_\xi \left[ \|\nabla f_\xi(x) - \nabla f_\xi(y)\|^2 \right] \leq L^2 \|x - y\|^2, \ \forall x, y \in \text{dom}(F).$$  

(4)

In the finite sum setting (2), the $L$-smoothness condition (4) can be expressed as the $L$-average smoothness of all $f_i$ with the moduli $L_i$ as:

$$\frac{1}{n} \sum_{i=1}^n \|\nabla f_i(x) - \nabla f_i(y)\|^2 \leq L_i^2 \|x - y\|^2, \ \forall x, y \in \text{dom}(F).$$  

(5)

**Assumption 2.3 (Bounded variance).** There exists $\sigma \in (0, \infty)$ such that

$$\mathbb{E}_\xi \left[ \|\nabla f_\xi(x) - \nabla f(x)\|^2 \right] \leq \sigma^2, \ \forall x \in \text{dom}(F).$$  

(6)

The bounded variance condition for (2) becomes

$$\mathbb{E}_\xi \left[ \|\nabla f_\xi(x) - \nabla f(x)\|^2 \right] \leq \frac{1}{n} \sum_{i=1}^n \|\nabla f_i(x) - \nabla f(x)\|^2 \leq \sigma^2, \ \forall x \in \text{dom}(F).$$  

(7)
Assumptions 2.2 and 2.3 are very standard in stochastic optimization methods and required for any stochastic gradient-based methods for solving (1). The \( L \)-average smoothness in 2.3 is in general weaker than the individual smoothness of each \( f_i \). Note that we do not require the Lipschitz continuity of \( f \) or \( \psi \) as in some recent work, e.g. [15].

We also consider problem (2) under the following assumption, which cover the case \( \psi = \delta_\mathcal{X} \), the indicator of a nonempty, closed, convex, and bounded set \( \mathcal{X} \). This assumption will be used to develop algorithms for solving (2) using hybrid SVRG estimators.

**Assumption 2.4.** The domain \( \mathcal{X} := \text{dom}(\psi) \) of \( \psi \) is bounded, i.e.:

\[
\mathcal{D}_\mathcal{X} := \sup_{x,y} \{ \|x - y\| \mid x, y \in \text{dom}(\psi) \} \in (0, +\infty).
\]

### 2.2 First-order optimality condition

The optimality condition of (1) can be written as

\[
0 \in \nabla f(x^*) + \partial \psi(x^*) \equiv \mathbb{E}_\xi [\nabla f_\xi(x^*) + \partial \psi(x^*)].
\]

Any point \( x^* \) satisfying (8) is called a stationary point of (1). The same definition applies to (2).

Note that (8) can be written equivalently to

\[
\mathcal{G}_\eta(x^*) := \frac{1}{\eta} (x^* - \text{prox}_{\eta \psi}(x^* - \eta \nabla f(x^*))) = 0.
\]

Here, \( \mathcal{G}_\eta \) is called the gradient mapping of \( F \) in (1) for any \( \eta > 0 \). It is obvious that if \( \psi = 0 \), then \( \mathcal{G}_\eta(x) = \nabla f(x) \), the gradient of \( f \) for any \( \eta > 0 \). Our goal is to seek an \( \varepsilon \)-stationary point \( \mathcal{T}_\varepsilon \) of (1) or (2) defined as follows:

**Definition 2.1.** Given a desired accuracy \( \varepsilon > 0 \), a point \( \mathcal{T}_\varepsilon \in \text{dom}(F) \) is said to be an \( \varepsilon \)-stationary point of (1) or (2) if

\[
\mathbb{E} \left[ \|\mathcal{G}_\eta(\mathcal{T}_\varepsilon)\|^2 \right] \leq \varepsilon^2.
\]

Here, the expectation is taken over all the randomness rendered from both \( \xi \) and the algorithm.

Let us clarify why \( \mathcal{T}_\varepsilon \) is an approximate stationary point of (1). Indeed, if \( x_+ := \text{prox}_{\eta \psi}(x - \eta \nabla f(x)) \), then \( \mathbb{E} \left[ \|x_+ - x\|^2 \right] \leq \varepsilon^2 \) means that \( \mathbb{E} \left[ \|x_+ - x\|^2 \right] \leq \eta^2 \varepsilon^2 \). On the other hand, \( x_+ = \text{prox}_{\eta \psi}(x - \eta \nabla f(x)) \) is equivalent to \( \frac{1}{\eta}(x - x_+) \in \nabla f(x) + \partial \psi(x+) \). Therefore, \( \|\nabla f(x_+) + \nabla \psi(x_+)) \| \leq \|\nabla f(x_+) - \nabla f(x)\| + \frac{1}{\eta} \|x_+ - x\| \) for some \( \nabla \psi(x_+) \in \partial \psi(x_+) \). Using the \( L \)-average smoothness of \( f \), we have \( \mathbb{E} \left[ \|\nabla f(x_+) + \nabla \psi(x_+))\|^2 \right] \leq 2 \left( L^2 + \frac{1}{\eta^2} \right) \mathbb{E} \left[ \|x_+ - x\|^2 \right] \leq 2 \left( L^2 + \eta^2 \right) \varepsilon^2 \). This condition shows that \( x_+ \) is an approximate stationary point of (1).

In practice, we often replace the condition (10) by \( \min_{0 \leq t \leq T} \mathbb{E} \left[ \|\mathcal{G}_\eta(x_t)\|^2 \right] \leq \varepsilon^2 \) which can avoid storing the iterate sequence \( x_t \).

### 3 Hybrid stochastic estimators

In this section, we propose new stochastic estimators for a generic function \( G \) that can cover function values, gradient, and Hessian of any expectation function \( f \) in (1).

#### 3.1 The construction of hybrid stochastic estimators

Given a function \( G(x) := \mathbb{E}_\xi [G_\xi(x)] \), where \( G_\xi \) is a (vector) stochastic function from \( \mathbb{R}^p \times \Omega \to \mathbb{R}^q \).

We define the following stochastic estimator of \( G \). As concrete examples, \( G \) can be the gradient mapping \( \nabla f \) or the Hessian mapping \( \nabla^2 f \) of \( f \) in problem (1) or (2).

**Definition 3.1.** Let \( u_t \) be an unbiased stochastic estimator of \( G(x_t) \) formed by a realization \( \zeta_t \) of \( \xi \), i.e. \( \mathbb{E}_{\xi}(u_t) = G(x_t) \) at a given \( x_t \). The following quantity:

\[
v_t := \beta_{t-1} v_{t-1} + \beta_{t-1} (G_\xi(x_t) - G_\xi(x_{t-1})) + (1 - \beta_{t-1}) u_t,
\]

is called a hybrid stochastic estimator of \( G \) at \( x_t \), where \( \zeta_t \) and \( \zeta_{t-1} \) are two independent realizations of \( \xi \) on \( \Omega \) and \( \beta_{t-1} \in [0, 1] \) is a given weight.
Clearly, if \( \beta_t = 0 \), then we obtain a simple unbiased stochastic estimator. If \( \beta_t = 1 \), then we obtain the SARAH-type estimator as studied in [50] but for general function \( G \). We are interested in the case \( \beta_t \in (0, 1) \), which can be referred to as a hybrid recursive stochastic estimator.

We can rewrite \( v_t \) as

\[
v_t := \beta_{t-1} G_{\xi_t}(x_t) + (1 - \beta_{t-1}) u_t + \beta_{t-1} (v_{t-1} - G_{\xi_t}(x_{t-1})).
\]

The first two terms are two stochastic estimators evaluated at \( x_t \), while the third term is the difference \( u_t := v_{t-1} - G_{\xi_t}(x_{t-1}) \) of the previous estimator and a stochastic estimator at the previous iterate. Here, since \( \beta_{t-1} \in (0, 1) \), the main idea is to exploit more recent information than the old one.

In fact, if \( G = \nabla f \), then the hybrid estimator \( v_t \) covers many other estimators, including SGD, SVRG, and SARAH. We consider three concrete examples of the unbiased estimator \( u_t \) of \( G(x_t) \) as follows:

- **The classical stochastic estimator**: \( u_t := G_{\xi_t}(x_t) \).
- **The SVRG estimator**: \( u_t := u_t^{\text{svrg}} = \mathcal{G}(\bar{x}) + G_{\xi_t}(x_t) - G_{\xi_t}(\bar{x}) \), where \( \mathcal{G}(\bar{x}) \) is a given unbiased snapshot evaluated at a given point \( \bar{x} \).
- **The SAGA estimator**: \( u_t := u_t^{\text{saga}} = G_{\xi_t}(y_{t+1}^i) - G_{\xi_t}(y_t^i) + \frac{1}{t} \sum_{i=1}^t G_i(y_t^i) \), where \( y_{t+1}^i = x_t \) if \( i = j_t \) and \( y_t^i = y_t^j \) if \( i \neq j_t \).

While both the classical stochastic and SVRG estimators work for both expectation and finite-sum settings, the SAGA estimator currently works for the finite-sum setting [50]. Note that it is also possible to consider mini-batch and important sampling settings for our hybrid estimators.

### 3.2 Properties of hybrid stochastic estimators

Let us first define

\[
\mathcal{F}_t := \sigma(\xi_0, \xi_1, \cdots, \xi_{t-1}, \xi_t)
\]

the \( \sigma \)-field generated by the history of realizations \( \{\xi_0, \xi_1, \cdots, \xi_{t-1}, \xi_t\} \) of \( \xi \) up to the iteration \( t \). We first prove in Appendix 1.1. the following property of the hybrid stochastic estimator \( v_t \).

**Lemma 3.1.** Let \( v_t \) be defined by (11). Then

\[
E(\xi_t, \xi_t) [v_t] = G(x_t) + \beta_{t-1} (v_{t-1} - G(x_{t-1})).
\]

If \( \beta_{t-1} \neq 0 \), then \( v_t \) is a biased estimator of \( G(x_t) \). Moreover, we have

\[
E(\xi_t, \xi_t) [||v_t - G(x_t)||^2] = \beta_{t-1}^2 ||v_{t-1} - G(x_{t-1})||^2 - \beta_{t-1}^2 ||G(x_t) - G(x_{t-1})||^2 + \beta_{t-1}^2 E(\xi_t, \xi_t) [||G_{\xi_t}(x_t) - G_{\xi_t}(x_{t-1})||^2] + (1 - \beta_{t-1})^2 E(\xi_t, \xi_t) [||u_t - G(x_t)||^2].
\]

**Remark 3.1.** From (11), we can see that \( v_t \) remains a biased estimator as long as \( \beta_{t-1} \in (0, 1) \). Its biased term is

\[
\text{Bias}[v_t | \mathcal{F}_t] = E(\xi_t, \xi_t) [v_t - G(x_t) | \mathcal{F}_t] = \beta_{t-1} ||v_{t-1} - G(x_{t-1})|| \leq ||v_{t-1} - G(x_{t-1})||.
\]

Clearly, the biased term of the estimator \( v_t \) is smaller than the one in the SARAH estimator \( v_t^{\text{sarah}} := v_t^{\text{sarah}} + G_{\xi_t}(x_t) - G_{\xi_t}(x_{t-1}) \) in [50] which is \( \text{Bias}[v_t^{\text{sarah}} | \mathcal{F}_t] = ||v_t^{\text{sarah}} - G(x_t)|| \).

While the variance \( E[||u_t^{\text{svrg}} - G(x_t)||^2] \) of \( u_t^{\text{svrg}} \) can only be bounded by a constant \( \sigma^2 \), the variance \( E[||u_t^{\text{svrg}} - G(x_t)||^2] \) of \( u_t^{\text{svrg}} \) can be reduced by gradually changing the snapshot \( \bar{x} \). The following lemma shows this property, whose proof can be found, e.g. in [61].

**Lemma 3.2.** Assume that \( u_t^{\text{svrg}} := G(\bar{x}) + G_{\xi_t}(x_t) - G_{\xi_t}(\bar{x}) \) is an SVRG estimator of \( G(x_t) := E(\xi_t, \xi_t) [G_{\xi_t}(x_t)] \). Then the following estimate holds:

\[
E[||u_t^{\text{svrg}} - G(x_t)||^2 | \mathcal{F}_t] = E[||G_{\xi_t}(x_t) - G_{\xi_t}(\bar{x})||^2 | \mathcal{F}_t] - ||G(\bar{x}) - G(x_t)||^2.
\]
If \(G_\xi\) is L-average Lipschitz continuous, i.e. \(\mathbb{E}_\xi \left[ \|G_\xi(x) - G_\xi(\tilde{x})\|^2 \right] \leq L^2\|x - \tilde{x}\|^2\) for all \(x, \tilde{x} \in \text{dom}(G)\), then we have

\[
\mathbb{E} \left[ \|u_t^{\text{MVT}} - G(x_t)\|^2 \mid F_t \right] \leq L^2\|x_t - \tilde{x}\|^2 - \|G(\tilde{x}) - G(x_t)\|^2 \leq L^2\|x_t - \tilde{x}\|^2. \tag{16}
\]

The following lemma bounds the variance \(\Delta_t := v_t - \nabla f(x_t)\) of \(v_t\) defined in (11). Its proof is given in Appendix 2.1.

**Lemma 3.3.** Assume that \(G_\xi\) is L-average Lipschitz continuous and \(u_t := G_\xi(x_t)\) is a classical stochastic estimator of \(G\). Then, we have the following upper bound:

\[
\mathbb{E} \left[ \|v_t - G(x_t)\|^2 \right] \leq \omega_1 \mathbb{E} \left[ \|v_0 - G(x_0)\|^2 \right] + L^2 \sum_{i=0}^{t-1} \omega_{i,t} \mathbb{E} \left[ \|x_{i+1} - x_i\|^2 \right] + S_t, \tag{17}
\]

where the expectation is taken over all the randomness \(F_{t+1} := \sigma(\xi_0, \xi_1, \cdots, \xi_t, \xi_{t+1})\), and

\[
\begin{align*}
\omega_t &:= \prod_{i=1}^t \beta_i^2 - 1, \\
\omega_{i,t} &:= \prod_{j=i+1}^t \beta_j^2 - 1, \quad i = 0, \cdots, t, \\
S_t &:= \sum_{i=0}^{t-1} \left( \prod_{j=i+2}^t \beta_j^2 \right)(1 - \beta_i^2) \mathbb{E} \left[ \|u_{i+1} - G(x_{i+1})\|^2 \right].
\end{align*} \tag{18}
\]

### 3.3 Mini-batch hybrid stochastic estimators

We can also consider a mini-batch hybrid recursive stochastic estimator \(\hat{v}_t\) of \(G(x_t)\) defined as:

\[
\hat{v}_t := \beta_{t-1} \hat{v}_{t-1} + \frac{\beta_{t-1}}{b_t} \sum_{i \in B_t} (G_\xi(x_t) - G_\xi(x_{t-1})) + (1 - \beta_{t-1}) u_t, \tag{19}
\]

where \(\beta_{t-1} \in [0, 1]\) and \(B_t\) is a mini-batch of size \(b_t\) and independent of \(u_t\).

Note that \(u_t\) can also be a mini-batch unbiased estimator of \(G(x_t)\). For example, \(u_t := \frac{1}{b_t} \sum_{j \in B_t} G_\xi(x_t)\) is a mini-batch unbiased stochastic estimator.

For \(\hat{v}_t\) defined by (19), we have the following property, whose proof is in Appendix 1.3.

**Lemma 3.4.** Let \(\hat{v}_t\) be the mini-batch stochastic estimator of \(G(x_t)\) defined by (19), where \(u_t\) is also a mini-batch unbiased stochastic estimator of \(G(x_t)\) with \(\mathbb{E}_{B_t} [u_t] = G(x_t)\) such that \(B_t\) is independent of \(B_t\). Then, the following estimate holds:

\[
\mathbb{E}_{(B_t, \hat{B}_t)} \left[ \|\hat{v}_t - G(x_t)\|^2 \right] = \beta_{t-1}^2 \|\hat{v}_{t-1} - G(x_{t-1})\|^2 - \rho \beta_{t-1}^2 \|G(x_t) - G(x_{t-1})\|^2 \\
+ \rho \beta_{t-1}^2 \mathbb{E}_\xi \left[ \|G_\xi(x_t) - G_\xi(x_{t-1})\|^2 \right] \\
+ (1 - \beta_{t-1})^2 \mathbb{E}_{\hat{B}_t} \left[ \|u_t - G(x_t)\|^2 \right], \tag{20}
\]

where \(\rho := \frac{n - b_t}{(n-1)b_t}\) if \(n\) is finite (i.e. \(G(x) := \frac{1}{n} \sum_{i=1}^n G_i(x)\)), and \(\rho := \frac{1}{b_t}\), otherwise (i.e. \(G(x) := \mathbb{E}_\xi [G_\xi(x)]\)).

Similar to Lemma 4.4, we can bound the variance \(\mathbb{E} \left[ \|\hat{v}_t - G(x_t)\|^2 \right]\) of the mini-batch hybrid estimator \(\hat{v}_t\) from (19) in the following lemma, whose proof is in Appendix 1.4. For simplicity of presentation, we choose \(b_t = b \in \mathbb{N}_+\) and \(b_t = b \in \mathbb{N}_+\), respectively for all \(t \geq 0\). Then, we have the following upper bound on the variance \(\mathbb{E} \left[ \|\hat{v}_t - G(x_t)\|^2 \right]\):

\[
\mathbb{E} \left[ \|\hat{v}_t - G(x_t)\|^2 \right] \leq \omega_1 \mathbb{E} \left[ \|v_0 - G(x_0)\|^2 \right] + \rho L^2 \sum_{i=0}^{t-1} \omega_{i,t} \mathbb{E} \left[ \|x_{i+1} - x_i\|^2 \right] + \rho S_t, \tag{21}
\]
where the expectation is taking over all the randomness $F_{t+1} := \sigma(B_0, \tilde{B}_0, \cdots, B_t, \tilde{B}_t)$, and $\omega_t, \omega_{t,t}$, and $S_t$ are defined in (18). Here, $\rho := \frac{n-b}{b(n-1)}$ and $\hat{\rho} := \frac{n-b}{b(n-1)}$ if $|\Omega| = n$ is finite and $\rho := \frac{1}{b}$ and $\hat{\rho} := \frac{1}{\hat{b}}$, otherwise.

The theoretical results developed in Section [3] are self-contained. They can be specified to develop stochastic optimization methods for solving (1) and (2). In the next sections, we only exploit these properties for $G = \nabla f$ to develop stochastic gradient-type methods.

4 Hybrid SARAH-SGD Algorithms

In this section, we utilize our hybrid stochastic estimator above with $G_\xi(x) := \nabla f_\xi(x)$ to develop new stochastic gradient algorithms for solving (1) and its finite-sum setting (2).

4.1 The single-loop algorithm

Our first algorithm is a single-loop stochastic proximal-gradient scheme for solving (1). This algorithm is described in detail in Algorithm 1.

Algorithm 1 (Hybrid stochastic gradient descent (Hybrid-SGD) algorithm)

1: Initialization: An initial point $x^0 \in \text{dom}(F)$.
2: Input the parameters $b \in \mathbb{N}_+$, $\beta_t \in (0, 1)$, $\gamma_t \in (0, 1]$, and $\eta > 0$ (will be specified later).
3: Generate an unbiased estimator $v_0 := \frac{1}{b} \sum_{i \in B} \nabla f_{\xi_i}(x_0)$ at $x_0$ using a mini-batch $\tilde{B}$.
4: Update $\hat{x}_1 := \text{prox}_{\eta \psi}(x_0 - \eta v_0)$ and $x_1 := (1 - \gamma_0)x_0 + \gamma_0 \hat{x}_1$.
5: For $t := 1, \cdots, m$ do
6: Generate a proper sample pair $(\xi_t, \xi_t)$ independently (single sample or mini-batch).
7: Evaluate $v_t := \beta_{t-1}v_{t-1} + \beta_{t-1}(\nabla f_{\xi_t}(x_t) - \nabla f_{\xi_t}(x_{t-1})) + (1 - \beta_{t-1})\nabla f_{\xi_t}(x_t)$.
8: Update $\tilde{x}_{t+1} := \text{prox}_{\eta \psi}(x_t - \eta v_t)$ and $x_{t+1} := (1 - \gamma_t)x_t + \gamma_t \tilde{x}_{t+1}$.
9: EndFor
10: Choose $\pi_m$ from $\{x_0, x_1, \cdots, x_m\}$ (at random or deterministic, specified later).

Algorithm 1 is different from existing SGD methods at the following points:

- Firstly, it starts with a relatively large mini-batch $\tilde{B}$ to compute an initial estimate for the initial gradient $\nabla f(x_0)$. This is quite different from existing methods where they often use single-sample, mini-batch, or increasing mini-batch sizes for the whole algorithms (e.g. [24]), and do not separate into two stages as in Algorithm 1.
  - Stage 1: Step 3 and Step 4
  - Stage 2: Step 5 to Step 8

The idea behind this difference is to find a good stochastic approximation $v_0$ for $\nabla f(x_0)$ to move on.

- Secondly, Algorithm 1 adopts the idea of ProxSARAH in [50] with two steps in $\tilde{x}_t$ and $x_t$ to handle the composite forms. This is different from existing methods as well as methods for noncomposite problems where two step-sizes $\gamma_t$ and $\eta_t$ are used. While the first step on $\tilde{x}_t$ is a standard proximal-gradient step, the second one on $x_t$ is an averaging step. If $\psi = 0$, i.e., in the non-composite problems, then Steps 4 and 8 reduce to $x_{t+1} := x_t - \eta_t v_t$, where $\hat{\eta}_t := \gamma_t \eta_t$.

Therefore, the product $\gamma_t \eta_t$ can be viewed as a combined step-size of Algorithm 1. Note that by using $\mathcal{G}_\eta(x)$ to approximate the gradient mapping $G_\eta$ defined by (9), we can rewrite the main-step of Algorithm 1 as

\begin{align*}
x_{t+1} := x_t - \hat{\eta}_t \tilde{v}_\eta(x_t), \quad \text{where} \quad \tilde{v}_\eta(x_t) := \frac{1}{\eta_t} (x_t - \text{prox}_{\eta \psi}(x_t - \eta_t v_t)) \quad \text{and} \quad \hat{\eta}_t := \gamma_t \eta_t.
\end{align*}
• Thirdly, another main difference between Algorithm [1] and existing methods is at Step 7 where we use our hybrid gradient estimator \( v_t \). In addition, we will show in the sequel that by using different step-sizes, Algorithm [1] leads to different variants.

Note that Algorithm [1] has only one-loop as standard SGD or SAGA. Hitherto, SAGA has been developed to solve the finite-sum setting [3], and there has existed no variant for solving [1] yet. Algorithm [1] can solve both (1) and (2). Moreover, it does not use an n \( \times \) p-table to store stochastic gradient components as in SAGA so that it almost has the same memory requirement as in SGD. However, each iteration, it requires three stochastic gradient evaluations instead of one as in SGD for the single-sample case. Therefore, its per-iteration cost can be viewed as a mini-batch SGD scheme of size 3.

4.2 One-iteration analysis

We first prove the following two lemmas to provide key estimates for convergence analysis of Algorithm [1]. The proof of these lemmas can be found in Appendices 2.1 and 2.2, respectively.

**Lemma 4.1.** Assume that Assumptions 2.1, 2.2, and 2.3 hold. Let \( \{(x_t, \tilde{x}_t)\} \) be the sequence generated by Algorithm 1 and the gradient mapping \( G_{n_t} \) of [1] be defined by (9). Then

\[
\begin{align*}
\mathbb{E}[F(x_{t+1})] & \leq \mathbb{E}[F(x_t)] - \frac{\eta_t}{2} \mathbb{E}[[G_{n_t}(x_t)]^2] + \frac{\eta_t}{2} \mathbb{E}[[\nabla f(x_t) - v_t]^2] \\
& \quad - \frac{\alpha_t}{2} \mathbb{E}[[\tilde{x}_{t+1} - x_t]^2] - \frac{1}{2} \mathbb{E}[\tilde{\sigma}_t^2].
\end{align*}
\]

(22)

where \( \{c_t\}, \{r_t\}, \) and \( \{q_t\} \) are any given positive sequences, \( \tilde{\sigma}_t^2 := \frac{\eta_t}{2}[[\nabla f(x_t) - v_t - c_t(\tilde{x}_{t+1} - x_t)]^2 \geq 0, S_t \) is defined in [15], and \( \theta_t \) and \( \kappa_t \) are given by

\[
\theta_t := \frac{\gamma_t}{c_t} + (1 + r_t)q_t\tau_t^2 \quad \text{and} \quad \kappa_t := \frac{2\gamma_t}{\eta_t} - L\gamma_t^2 - \gamma_t c_t - q_t \left(1 + \frac{1}{r_t}\right).
\]

(23)

**Lemma 4.2.** Assume that Assumptions 2.1, 2.2, and 2.3 hold. Let \( \{(x_t, \tilde{x}_t)\} \) be the sequence generated by Algorithm 7 and the gradient mapping \( G_{n_t} \) of [1] be defined by (9). Let \( V \) be a Lyapunov function defined by

\[
V(x_t) := \mathbb{E}[F(x_t)] + \frac{\alpha_t}{2} \mathbb{E}[[v_t - \nabla f(x_t)]^2],
\]

(24)

for a given \( \alpha_t > 0 \). Assume that

\[
\alpha_t - \beta_t^2\alpha_{t+1} - \theta_t \geq 0 \quad \text{and} \quad \kappa_t - \alpha_{t+1}\beta_t^2\gamma_t^2L^2 \geq 0.
\]

(25)

Then, the following estimate holds

\[
V(x_{t+1}) \leq V(x_t) - \frac{\eta_t\beta_t}{2} \mathbb{E}[[G_{n_t}(x_t)]^2] + \frac{1}{2}\alpha_{t+1}(1 - \beta_t)^2\gamma_t^2L^2.
\]

(26)

where \( \gamma_t^2 := \mathbb{E}_{G_t}[[\nabla f_G(x_t) - \nabla f(x_t)]^2] \). As a consequence, for any \( m \geq 0 \), we also have

\[
\sum_{t=0}^{m} \frac{\eta_t\beta_t}{2} \mathbb{E}[[G_{n_t}(x_t)]^2] \leq F(x_0) - F^* + \frac{\alpha_0}{2} \mathbb{E}[[v_0 - \nabla f(x_0)]^2] + \frac{1}{2}\sum_{t=0}^{m} \alpha_{t+1}(1 - \beta_t)^2\gamma_t^2L^2.
\]

(27)

Note that if \( \beta_t = 1 \) for all \( t \geq 0 \), then our hybrid estimator \( v_t \) reduces to the SARAH estimator [50]. In this case, the estimate (26) becomes \( V(x_{t+1}) \leq V(x_t) - \frac{\eta_t\beta_t}{2} \mathbb{E}[[G_{n_t}(x_t)]^2] \), which shows a monotonicity of \( \{V(x_t)\} \). This estimate can be used to analyze the convergence of double-loop SARAH-based algorithms in [53] [66].
4.3 Convergence analysis of Algorithm \([1]\)  

**Constant step-size:** We first investigate the convergence of Algorithm \([1]\) for constant step-sizes. The following theorem shows the convergence of Algorithm \([1]\) and its complexity bound, whose proof is given in Appendix \([2,3]\).

**Theorem 4.1.** Assume that Assumptions 2.1, 2.2, and 2.3 hold. Let \(\{x_t\}_{t=0}^m\) be the sequence generated by Algorithm \([2]\) to solve \([1]\) using the initial mini-batch size \(b := c_1^2 \sigma^{8/3}(m + 1)^{1/3}\) at Step \(3\) to evaluate \(v_0\), and the following constant weight \(\beta_t\) and step-sizes \(\gamma_t\) and \(\eta_t\):

\[
\begin{align*}
\beta_t &= \beta := 1 - \frac{1}{b^{\frac{1}{2}}(m+1)^{1/2}}, \\
\gamma_t &= \gamma := \frac{3}{\sqrt{Lc_1}L(\sigma^{2/3}(m+1)^{1/3})}, \\
\eta_t &= \eta := \frac{2}{4+L},
\end{align*}
\]

where \(c_1 \geq \frac{1}{\sigma^{1/3}(m+1)^{1/3}}\) is a given constant. Then the following statements hold:

(a) The parameters \(\beta, \gamma, \) and \(\eta\) satisfy \(\beta \in (0, 1), \gamma \in (0, 1], \) and \(\frac{2}{4+L} \leq \eta \leq \frac{1}{2}.

(b) Let \(\mathcal{F}_m \sim U \{(x_t)_{t=0}^m\}\) be the output of Algorithm \([2]\). Then, we have

\[
E \left[\|G_0(\mathcal{F}_m)\|^2\right] \leq \left[\frac{\sqrt{17c_1L}}{6} [F(x_0) - F^*] + \frac{17}{9c_1} (L+4)^2 \sigma^{2/3} \right] \leq \frac{\|F_0\|_2^2}{\epsilon^2}.
\]

(c) Let \(\Delta_0 := \frac{\sqrt{17c_1L}(L+4)}{6} [F(x_0) - F^*] + \frac{17(L+4)^2}{9c_1}\) be a given constant. Then, for any \(\epsilon > 0\), the number of iterations \(m\) to obtain \(\mathcal{F}_m\) such that \(E \left[\|G_0(\mathcal{F}_m)\|^2\right] \leq \epsilon^2\) is at most

\[
m := \left\lfloor \frac{\Delta_0^{1/2} \sigma}{\epsilon^3} \right\rfloor = \mathcal{O} \left( \frac{\sigma}{\epsilon^3} \right).
\]

This is also the total number of proximal operations \(\text{prox}_{\eta \psi}\). In addition, the total number of stochastic gradient evaluations \(\nabla f_\xi(x_t)\) is at most

\[
\mathcal{T}_m := \left[ c_1^2 \Delta_0^{1/2} \frac{\sigma^3}{\epsilon^3} + 3 \Delta_0^{3/2} \frac{\sigma^3}{\epsilon^3} \right] = \mathcal{O} \left( \frac{\sigma^3}{\epsilon^3} \right).
\]

**Adaptive step-size:** Theorem \([4.1]\) states the convergence and complexity estimate of Algorithm \([1]\) with constant step-sizes. However, when the number of iterations \(m\) is relatively small, we can develop an adaptive rule to update the step-size \(\gamma_t\) as follows:

- Let us first fix \(\beta := 1 - \frac{1}{b^{\frac{1}{2}}(m+1)^{1/2}} \in (0, 1)\) as in Theorem \([4.1]\).
- Next, we also fix \(\eta_t := \eta \in (0, \frac{2}{3})\) and define \(\delta := \frac{2}{3} - 3 > 0\).
- Then, we can update \(\gamma_t\) adaptively as

\[
\gamma_m := \frac{\delta}{L} \quad \text{and} \quad \gamma_t := \frac{\delta}{L + L^2} \frac{\beta^2 \gamma_{t+1} + \beta^3 \gamma_{t+2} + \cdots + \beta^2(m-1) \gamma_m}{\gamma_m}.
\]

for \(t = 0, \ldots, m-1\). Applying Lemma \([A.1]\), it is obvious to show that \(0 < \gamma_0 < \gamma_1 < \cdots < \gamma_m\). Interestingly, our step-size is updated in an increasing manner instead of diminishing as in existing SGD-type methods. Moreover, given \(m\), we can pre-compute the sequence of these step-sizes \(\{\gamma_t\}_{t=0}^m\) in advance within \(\mathcal{O}(m)\) basic operations. Therefore, it does not significantly incur the computational cost of our method.

The following theorem states the convergence of Algorithm \([1]\) under the adaptive update \([30]\), whose proof can be found in Appendix \([2,3]\).

**Theorem 4.2.** Assume that Assumptions 2.1, 2.2, and 2.3 hold. Let \(\{x_t\}_{t=0}^m\) be the sequence generated by Algorithm \([2]\) to solve \([1]\) using the parameters \(\beta, \eta\), and step-size \(\gamma_t\) defined by \([30]\). Then, the following statements hold:
(a) If we define $\Sigma_m := \sum_{t=0}^{m} \gamma_t$, then $\Sigma_m$ is bounded from below as $\Sigma_m \geq \frac{\delta(m+1)^{3/4}}{2^{5/4}L^{1/2} + 2\beta}$.
(b) Let $x_m \sim U_p \left(\{x_t\}_{t=0}^{m}\right)$ with $p_t \equiv \mathbb{P}(x_t = x) = \frac{m}{t}$. Then, the following estimate holds:

$$
\mathbb{E} \left[ \|G_t(x_m)\|^2 \right] \leq \frac{4L\tilde{\beta}_{1/4} \sqrt{1 + 2\delta}}{\eta^2(m + 1)^{3/4}} \left[ F(x_0) - F^* \right] + \frac{2(1 + 2\eta^2)\sigma^2}{\eta^2 \tilde{\beta} \sqrt{2}(m + 1)^{1/2}}.
$$

(c) If we choose $\tilde{b} := c_2^2 \sigma^{8/3} (m + 1)^{1/3}$ for any $c_1 \geq \frac{1}{m + 1 + 1/\epsilon^2}$, then, for any $\epsilon > 0$, to guarantee $\mathbb{E} \left[ \|G_t(x_m)\|^2 \right] \leq \epsilon^2$, we need at most $m$ iterations as

$$
m := \left\lfloor \frac{3^{1/2} \sigma^3}{\epsilon^3} \right\rfloor = \mathcal{O} \left( \frac{\sigma}{\epsilon^3} \right),
$$

where $\Delta_0 := \frac{4L\sqrt{1 + 2\delta}}{\eta^2(m + 1)^{3/4}} \left[ F(x_0) - F^* \right] + \frac{2(1 + 2\eta^2)\sigma^2}{\eta^2 \tilde{\beta} \sqrt{2}(m + 1)^{1/2}}$. This is also the total number of proximal operations $\text{prox}_{\eta^2}$.

The number of stochastic gradient evaluations $\nabla f_{t_{k}}(x_t)$ is at most

$$
T_m := \frac{c_2^2 \Delta^2 \sigma^3}{\epsilon^3} + \frac{3 \Delta_0 \sigma^3}{\epsilon^3} = \mathcal{O} \left( \frac{\sigma^3}{\epsilon^3} + \frac{\sigma}{\epsilon^3} \right).
$$

While the proof of Theorem 4.1 relies on the Lyapunov function $V$ defined by (24) that has an asymptotically monotone property, the proof of Theorem 4.2 is completely different by adopting the techniques in [56] and does not use any Lyapunov function.

**Remark 4.1 (No initial mini-batch).** If we choose the initial mini-batch size $\tilde{b} := 1$ (i.e. single sample) to compute $v_0$ at Step 3 of Algorithm 1 then (31) reduces to

$$
\mathbb{E} \left[ \|G_t(x_m)\|^2 \right] \leq \frac{4L\sqrt{1 + 2\delta}}{\eta^2(m + 1)^{3/4}} \left[ F(x_0) - F^* \right] + \frac{(1 + 2\eta^2)\sigma^2}{\eta^2 \tilde{\beta} \sqrt{2}(m + 1)^{1/2}}.
$$

In this case, the oracle complexity of Algorithm 1 reduces to $\mathcal{O} \left( \frac{\sigma^2}{\epsilon^3} \right)$ as in classical proximal SGD methods, see, e.g. [27]. Therefore, the choice of mini-batch $\tilde{B}$, initial estimator $v_0$ is crucial in Algorithm 1 to achieve better complexity bounds than SGD.

**Remark 4.2 (The effect of $m$ on $\gamma_t$).** Due to the update (30), we have $\gamma_m \geq \gamma_{m-1} \geq \cdots \geq \gamma_0 > 0$. Clearly, if $m$ is large, $\{\gamma_t\}$ is getting smaller and smaller as $t$ is decreasing, which leads to a slow convergence. This suggests that we should restart Algorithm 1 after a relatively small number of iterations $m$ to avoid small step-sizes $\{\gamma_t\}$. This algorithmic variant becomes more efficient if we combine it with a double loop as described in Algorithm 2 in Subsection 4.4.

### 4.4 Double-Loop Hybrid Stochastic Gradient Descent Algorithm

Similar to SVRG or SARAH variants, we can develop double-loop variants for our methods. However, unlike SVRG and SARAH-based methods where their double-loop is mandatory, we use an outer loop as a restarting strategy in order to restart Algorithm 1 at each stage. Without the outer loop, Algorithm 1 still has convergence guarantee as shown in Theorems 4.1 and 4.2. The complete double-loop algorithm is described in Algorithm 2.

To analyze Algorithm 2, we use $x_t^{(s)}$ to represent the iterate of Algorithm 1 at the $t$-th inner iteration within each stage $s$. As we can see, Algorithm 2 calls Algorithm 1 as a subroutine at each iteration, called stage $s$ and exports the output $x_t^{(s)} := x_t^{(s)}$ as the last iterate of Algorithm 1 instead of taking it randomly from $\{x_t^{(s)}\}_{t=0}^m$. Here, we assume that we fix the step-size $\eta_t = \eta \in (0, \frac{1}{2})$, fix the mini-batch $\tilde{b}_t = \tilde{b} \in \mathbb{N}_+$, and choose $\beta := 1 - \frac{1}{\delta (m+1)^{3/4}} \in (0, 1)$ for simplicity of our analysis.

Now, we can derive the convergence of Algorithm 2 in the following theorem whose proof is deferred to Appendix 2.5.
Algorithm 2 (Double-loop Hybrid-SGD algorithm)

1: Initialization: An initial point $x(0)$ and parameters $b$, $m$, $\beta_t$, and $\eta_t$ (will be specified).
2: OuterLoop: For $s := 1, 2, \ldots, S$ do
3: Run Algorithm 1 with an initial point $x(s) := \pi(s-1)$.
4: Set $\pi(s) := x_{m+1}$ as the last iterate of Algorithm 1.
5: EndFor

Theorem 4.3. Let $\{x_t(s)\}_{t=0}^{\infty}$ be the sequence generated by Algorithm 2 to solve (1) using $\eta \in (0, \frac{2}{L})$ and $b_0 = \overline{b} \in \mathbb{R}_+$. Moreover, if we choose $\beta := 1 - \frac{1}{\eta^2(m+1)^3}$, then, the following estimate holds:

$$\frac{1}{S\tau_m} \sum_{s=1}^{S} \sum_{t=0}^{m} \mathbb{E} \left[ \|G_t(x_t(s))\|^2 \right] \leq \frac{4L\hat{b}^{1/4} \sqrt{1 + 2\delta}}{\eta^2 \sigma(\sigma + 1)^{3/4}} \left[ F(\pi(0)) - F^* \right] + \frac{(1 + 2\eta)^2 \sigma^2}{\eta^2 b L^{1/2} (m + 1)^{1/2}}. \tag{34}$$


Let us choose $\mathcal{P}_T \sim \mathcal{U}_p \left( \{x_t(s)\}_{t=0}^{\infty} \right)$, $\overline{b} := \frac{\sigma^2}{\epsilon^4}$, and $m + 1 := \frac{\sigma^2}{\epsilon^4}$ for some constants $c_1 > 0$ and $c_2 > 0$ and $c_1c_2 > 4(1+2\delta)^2$. Then, for any $\epsilon > 0$, to guarantee $\mathbb{E} \left[ \|G_t(\mathcal{P}_T)\|^2 \right] \leq \epsilon^2$, we need at most $S$ outer iterations with

$$S := \left\lfloor \frac{\Delta_0}{\epsilon^2} \right\rfloor, \quad \text{where } \Delta_0 := \frac{4Lc_1^{1/4} \sqrt{1 + 2\delta} \left[ F(\pi(0)) - F^* \right]}{\eta^2 c_2^{1/2} \left( 1 - \frac{2(1+2\delta)}{\eta^2 \sigma^2 (2\epsilon^2 - 1) \epsilon^4} \right)}.$$ \tag{35}

Consequently, the total number $T_T$ of stochastic gradient evaluations $\nabla f_x(x_t(s))$ does not exceed

$$T_T := \frac{(c_1 + 3c_2) \Delta_0 \sigma}{\epsilon^4} = \mathcal{O} \left( \frac{\sigma}{\epsilon^2} \right). \tag{36}$$

The total number of proximal operations $\text{prox}_{\psi}$ is at most $T_{\text{prox}} := \frac{\sigma \Delta_0 \sigma}{\epsilon^4} = \mathcal{O} \left( \frac{\sigma}{\epsilon^2} \right)$.

Note that the complexity bound (36) only holds if $\frac{(c_1 + 3c_2) \Delta_0 \sigma}{\epsilon^4} > \frac{c_1 \sigma^2}{\epsilon^4}$. Otherwise, the total number of stochastic gradient evaluations is at most $\mathcal{O} \left( \max \left\{ \frac{\sigma}{\epsilon^2}, \frac{\sigma^2}{\epsilon^4} \right\} \right)$, where some constants independent of $\sigma$ and $\epsilon$ are hidden. Practically, if $\beta$ is very close to 1, one can remove the unbiased SGD term to save one stochastic gradient evaluation. In this case, our estimator reduces to SARAH but using different step-size. We observed empirically that when $\beta \approx 0.999$, the performance of our methods is not affected.

Remark 4.3 (Linear convergence under gradient dominant condition). If the composite function $F$ satisfies a $\tau$-gradient dominant condition $F(x) - F^* \leq \frac{\tau}{2} \|G_t(x)\|^2$ for any $x \in \text{dom}(F)$ and $\eta > 0$, where $\tau > 0$ (see, e.g. [65]), then, we can modify Algorithm 1 by setting $\pi(s) := x_m$, where $x_m := \{x_t(s)\}_{t=0}^{\infty}$, to obtain

$$\mathbb{E} \left[ F(\pi(s)) - F^* \right] \leq \frac{\tau}{2} \mathbb{E} \left[ \|G_t(\pi(s))\|^2 \right] \leq \frac{2\tau L \hat{b}^{1/4} (1 + 2\delta)^{1/2}}{\eta^2 \delta (m + 1)^{3/4}} \mathbb{E} \left[ F(\pi(s)) - F^* \right] + \frac{\tau (1 + 2\eta)^2 \sigma^2}{\eta^2 b L^{1/2} (m + 1)^{1/2}}.$$ 

Now, for any $\epsilon > 0$, let us choose $m + 1 := \frac{4\sqrt{2L} \tau^{3/2} (1 + 2\eta)^{1/4} (1 + 2\delta)^{1/2}}{\eta^2 \sigma} \frac{\sigma}{\epsilon^4}$ and $b := \frac{\delta \tau^{1/2} (1 + 2\eta)^{1/2}}{\sqrt{2L} \tau^{1/2} (1 + 2\delta)^{1/2}} \frac{\sigma}{\epsilon^4}$. Then, by denoting $\Delta_s := \mathbb{E} \left[ F(\pi(s)) - F^* \right]$, the last inequality leads to $\Delta_s \leq \frac{\tau}{2} \Delta_{s-1} + \frac{\tau}{2}$. This implies $\Delta_s - \epsilon \leq \frac{1}{2} (\Delta_{s-1} - \epsilon)$. By induction, we have $\Delta_s \leq \frac{1}{2} (\Delta_0 - \epsilon) + \epsilon$. We conclude that the sequence $\{\mathbb{E} \left[ F(\pi(s)) - F^* \right]\}$ converges linearly to an $\epsilon$-ball around zero.

Note that if $\psi = 0$, then the gradient dominant condition above reduces to the standard one $f(x) - f(x^*) \leq \frac{\tau}{2} \|\nabla f(x)\|^2$ for any $x \in \text{dom}(f)$, which is widely used in the literature.
4.5 Applications to finite-sum and non-composite settings

The finite-sum case: We can apply both Algorithm 1 and Algorithm 2 to solve the finite-sum problem \( (2) \). We can use a mini-batch \( \tilde{B}_t \) of the size \( b \in [n] \) to approximate \( v_0 \). However, we make the following changes in Algorithm 1 to solve \( (2) \):

- We compute \( v_0 := \nabla f(x_0) = \frac{1}{n} \sum_{i=1}^n \nabla f_i(x_0) \), the full gradient of \( f \) at \( x_0 \).
- We evaluate \( v_t := \beta v_{t-1} + \tilde{\beta} (\nabla f_t(x_t) - \nabla f_t(x_{t-1})) + (1 - \tilde{\beta}) \nabla f_t(x_t) \), where \( i_t, j_t \in [n] \) are two independent random indices generated from a uniform distribution on \([n]\).

Since we set \( \tilde{b} = n \), we need to change the weight \( \tilde{\beta} \) in Theorem 4.2 and Theorem 4.3 to

\[
\tilde{\beta} := 1 - \frac{c_1}{(m+1)^{2/3}}
\]

for some \( 0 < c_1 \leq (m+1)^{2/3} \). With this choice of \( \tilde{b} \) and \( \tilde{\beta} \), the conclusions of Theorem 4.2 and Theorem 4.3 remain true. But the number of stochastic gradient evaluations is at most \( T_m := O \left( n + \frac{1}{\tilde{\beta}} \right) \). To avoid overloading the paper, we skip the detailed analysis here.

In terms of assumptions, apart from Assumptions 2.1 and 2.2, we still require Assumption 2.3 (i.e., (7)) to hold for \( (2) \). Hence, Algorithm 1 can solve \( (2) \), but it requires stronger assumptions (Assumptions 2.1, 2.2, and 2.3) than ProxSVRG [60], SpiderBoost [68], and ProxSARAH [56]. However, as a compensation, Algorithm 1 uses a single-loop.

The non-composite settings: If \( \psi = 0 \), then we obtain a non-composite setting of \( (1) \) and \( (2) \). In this case, Algorithms 1 and 2 reduce to the ones in our preprint [66]. The analysis in Theorems 4.1, 4.2, and 4.3 can be modified accordingly to cover non-composite settings of \( (1) \):

- Step 4 of Algorithm 1 becomes \( x_t = x_0 - \bar{\eta}_t v_0 \), where \( \bar{\eta}_t \) is a new step-size.
- Step 8 of Algorithm 1 reduces to \( x_{t+1} = x_t - \tilde{\eta}_t v_t \), where \( \tilde{\eta}_t \) is a new step-size.

The step-size \( \tilde{\eta}_t := \frac{2}{L(1 + 4\alpha_m)} \) which combines both \( \eta_t \) and \( \gamma_t \) in Theorem 4.1, where

\[
\beta := 1 - \frac{c_1}{(m+1)^{2/3}} \quad \text{and} \quad \alpha_m := \frac{\beta^2 (1 - \beta^{2\alpha_m})}{1 - \beta}
\]

for \( 0 < c_1 \leq \hat{b}^{1/2}(m+1)^{1/2} \).

For clarity of exposition, we skip the analysis of this variant here.

5 Extensions to mini-batch variants

We consider the mini-batch variants of Algorithm 1 and Algorithm 2 for solving \( (1) \). More precisely, the mini-batch SARAH-SGD estimator \( \hat{v}_t \) for \( \nabla f(x_t) \) is defined as

\[
\hat{v}_t := \beta \hat{v}_{t-1} + \tilde{\beta} \sum_{i \in B_t} \left( \nabla f_i(x_t) - \nabla f_i(x_{t-1}) \right) + \frac{1 - \beta}{b} \sum_{i \in B_t} \nabla f_i(x_t), \tag{37}
\]

where \( B_t \) is a mini-batch of size \( b \) and \( \tilde{B}_t \) is a mini-batch of size \( \hat{b} \) and independent of \( B_t \). Here, we fix \( \hat{b} \in (0, 1) \) and the mini-batch sizes \( b \in \mathbb{N}_+ \) and \( \hat{b} \in \mathbb{N}_+ \) for all \( t \geq 0 \). Note that the estimator \( \hat{v}_t \) is an instance of \( [19] \) when \( G = \nabla f \). For the sake of presentation, we only consider the constant step-size variant as a consequence of Theorem 4.1. We state our first result in the following theorem whose proof can be found in Appendix 3.1.

Theorem 5.1. Assume that Assumptions 2.1, 2.2, and 2.3 hold. Let \( \{x_t\}_{t=0}^m \) be the sequence generated by Algorithm 7 to solve \( (1) \) using the mini-batch update for \( \hat{v}_t \) as in (37) at Step 2 instead of \( v_t \), and the following parameter configuration:

\[
\begin{aligned}
\tilde{b} &:= c_1^2 \sigma^{8/3}[b(m+1)]^{1/3} \\
\beta_t &:= \beta_t \equiv 1 - \frac{\sqrt{5}}{\sqrt{b(m+1)}} \\
\gamma_t &:= \gamma_t = \frac{3c_1 b^{1/3}(m+1)^{1/2}}{\sqrt{14Lb^{1/4}(m+1)^{1/4}}} \\
\eta_t &:= \eta_t = \frac{2}{4 + L},
\end{aligned}
\tag{38}
\]

where \( 1 \leq \tilde{b} \leq \hat{b}(m+1) \) and \( 0 < c_1 \leq \frac{\sqrt{27}L}{30\tilde{b}} \) are given. Then the following statements hold:

(a) The parameters \( \beta, \gamma, \) and \( \eta \) satisfy \( \beta \in [0, 1), \gamma \in (0, 1], \) and \( \frac{2}{4 + L} \leq \eta \leq \frac{1}{2} \).
(b) Let \( \mathcal{F}_m \sim \mathcal{U}(\{x_t\}_{t=0}^n) \) be the output of Algorithm 2. Then, we have

\[
\mathbb{E} \left[ \|G_y(\mathcal{F}_m)\|^2 \right] \leq \frac{2\sqrt{2}L_0^{1/4}}{3c_1^2\eta b^4 L b^{1/2}(m + 1)^{3/4}} [F(x_0) - F^*] + \frac{34\sigma^2}{9\eta b^{1/2} b^{1/2}(m + 1)^{1/2}}. \tag{39}
\]

(c) Let us choose \( b = \tilde{b} \in \mathbb{N}_+ \) and \( \tilde{b} = c_2^2\sigma^{8/3}(b(m + 1))^{1/3} \) for some \( c_2 > 0 \). Then, for any \( \varepsilon > 0 \), the number of iterations \( m \) to obtain \( \mathcal{F}_m \) such that \( \mathbb{E} \left[ \|G_y(\mathcal{F}_m)\|^2 \right] \leq \varepsilon^2 \) is at most

\[
m := \left\lfloor \frac{\Delta_0^{3/2} \sigma}{\varepsilon^3} \right\rfloor = O \left( \frac{\sigma}{\varepsilon^3} \right),
\]

where \( \Delta_0 := \frac{2\sqrt{2}L_0^{1/4}}{3c_1^2\eta b^4 L b^{1/2}(m + 1)^{3/4}} [F(x_0) - F^*] + \frac{34\sigma^2}{9\eta b^{1/2} b^{1/2}(m + 1)^{1/2}} \) is a given constant. This is also the total number \( T_{\text{prox}} \) of proximal operations \( \text{prox}_{\eta y} \). The total number of stochastic gradient evaluations \( \nabla f_t(x_t) \) is at most

\[
T_m := \left\lfloor \frac{c_2\Delta_0^{3/2} \sigma^3}{\varepsilon^3} + \frac{3\Delta_0^{3/2} \sigma}{\varepsilon^3} \right\rfloor = O \left( \frac{\sigma^3}{\varepsilon^3} + \frac{\sigma}{\varepsilon^3} \right). \tag{40}
\]

Theorem 5.1 states that using the mini-batch estimator \( \hat{v}_l \) from (37), the total number of stochastic gradient evaluations \( T_m \) in Algorithm 2 remains the same as in Theorem 4.1. However, the total number of proximal operations \( T_{\text{prox}} \) reduces to \( O \left( \frac{\sigma}{\varepsilon^3} \right) \).

We can also modify Algorithm 2 to obtain a mini-batch variant. The following theorem shows the convergence of this mini-batch variant whose proof can be found in Appendix 3.2.

**Theorem 5.2.** Let \( \{x_t^{(s)}\}_{t=0}^{s=1 \ldots m} \) be the sequence generated by Algorithm 2 to solve (37) using the mini-batch update for \( \hat{v}_l \) at Step 3 instead of \( v_t \), \( \eta \in (0, \frac{1}{3}) \), and

\[
\gamma_m := \frac{\delta}{L} \quad \text{and} \quad \gamma_t := \frac{b\delta}{bL + L^2 [2\beta^2 \gamma_t + 3\beta \gamma_{t+1} + \cdots + \beta^2 (m-t) \gamma_m]}, \tag{41}
\]

where \( \delta := \frac{2}{\eta} - 3 > 0 \).

If we choose the output of Algorithm 3 as \( \mathcal{F}_m \sim \mathcal{U}_p \left( \{x_t^{(s)}\}_{t=0}^{s=1 \ldots m} \right) \) such that \( p_t := \mathbb{P} \left( \mathcal{F}_T = x_t^{(s)} \right) = \frac{m_t}{\sum m_t} \), then the following estimate holds

\[
\mathbb{E} \left[ \|G_y(\mathcal{F}_T)\|^2 \right] \leq \frac{2L \left[ b\delta b^{1/2}(m + 1)^{1/2} + 8b^{3/2} \right]}{9\eta^2 \rho^2 L b^{1/2} b^{1/2}(m + 1)^{1/2}} [F(\mathcal{F}^{(0)}) - F^*] + \frac{34\sigma^2}{9\eta b^{1/2} b^{1/2}(m + 1)^{1/2}}. \tag{42}
\]

If we further choose \( \tilde{b} := c_2^2\sigma^{8/3}(m + 1)^{1/3} \) and \( \tilde{b} = \tilde{b} \in \mathbb{N}_+ \) such that \( 1 \leq \tilde{b} \leq \delta^2/3 \sigma^{2/3} \sigma^{8/9}(m + 1)^{1/3} \) for any \( c_1 > 0 \), then, for any \( \varepsilon > 0 \), the total number of iterations \( T \) to achieve \( \mathbb{E} \left[ \|G_y(\mathcal{F}_T)\|^2 \right] \leq \varepsilon^2 \) is at most

\[
T := (m + 1)S = \left\lfloor \frac{\Delta_0^{3/2} \sigma}{\varepsilon^3} \right\rfloor = O \left( \frac{\sigma}{\varepsilon^3} \right), \tag{43}
\]

where \( \Delta_0 := \frac{8\sqrt{2}L_0^{1/4}}{3c_1^2\eta b^4 L b^{1/2}(m + 1)^{3/4}} [F(\mathcal{F}^{(0)}) - F^*] + \frac{34\sigma^2}{9\eta b^{1/2} b^{1/2}(m + 1)^{1/2}} \) is a constant. This is also the total number of proximal operations \( \text{prox}_{\eta y} \). The total number of stochastic gradient evaluations \( \nabla f_t(x_t) \) is at most

\[
T_T := \left\lfloor \frac{c_2\Delta_0^{3/2} \sigma^3}{\varepsilon^3} + 3\Delta_0^{3/2} \sigma^3 \right\rfloor = O \left( \frac{\sigma^3 S^{2/3}}{\varepsilon} + \frac{\sigma}{\varepsilon^3} \right) \quad \text{for any} \quad S \geq 1. \tag{44}
\]

**Remark 5.1 (Mini-batch and step-size trade-off).** From (38) of Theorem 5.1, we can see that \( \gamma = \frac{34\sigma^2}{9\eta b^{1/2} b^{1/2}(m + 1)^{1/2}} \). Clearly, if we use a large mini-batch size \( b \) for \( \hat{v}_l \), then we obtain a large value of the step-size \( \gamma \). Assume that \( \gamma \approx 1 \), which is equivalent to \( \frac{34\sigma^2}{9\eta b^{1/2} b^{1/2}(m + 1)^{1/2}} \approx 1 \).
Moreover, from Theorem 5.1(c), we also have \( b(m + 1) = \frac{\Delta_0^{3/2}}{3\epsilon} \). Combining both conditions, we can roughly set \( b \approx \frac{1 + \Delta_0^{3/2}}{3\sqrt{\epsilon}L}\frac{\sigma^3}{\sqrt{L}} \) and \( m + 1 \approx \frac{1 + \Delta_0^{3/2}}{3\sqrt{\epsilon}L} \frac{\sigma^3}{\sqrt{L}} \). Empirical evidence in Section 7 will show that large step-size \( \gamma \) leads to a better performance.

For the mini-batch double-loop variant stated in Theorem 5.2, we use the update (40) of \( \gamma_t \) hints that if \( m \) is small then the sequence of step-sizes \( \{ \gamma_t \}_{t=0}^{m} \) is large. From (42), we have \( m + 1 = \frac{\Delta_0^{3/2}}{3\epsilon} \). Therefore, to obtain small \( m \), we need to choose \( S \) large. However, as a compensation, the first term in the total number of stochastic gradient evaluations \( T_T = O \left( \frac{\sigma^2\epsilon^{2/3}}{\epsilon} + \frac{\sigma}{\epsilon} \right) \) may increase. As an example, we can choose \( S \) such that \( \frac{\sigma^2\epsilon^{2/3}}{\epsilon} = \frac{\sigma}{\epsilon} \) which leads to \( S = O \left( \frac{1}{\epsilon^{2/3}} \right) \).

**Remark 5.2 (Practical termination condition).** In Algorithm 1 Algorithm 2 Algorithm 3 and their variants, we need to choose \( \tau_m \) or \( \tau_T \) randomly among the iterate sequence generated up to the iteration \( m \) or \( T := S(m + 1) \), respectively. This requires to save the sequence of iterates. To avoid saving this sequence, we can choose the best-so-far iterate \( x_{t(s)} \) based on the following guarantee:

\[
\min_{0 \leq t \leq m} \mathbb{E} \left[ \| G_0(x_t) \|^2 \right] \leq \epsilon^2 \quad \text{or} \quad \min_{0 \leq t \leq m, 1 \leq s \leq S} \mathbb{E} \left[ \| G_0(x_{t(s)}) \|^2 \right] \leq \epsilon^2.
\]

However, in practice, we often take the last iterate \( x_m \) or \( \bar{x}^{(S)} \) as the output of the algorithm which unfortunately does not have a theoretical guarantee in this paper.

### 6 Hybrid SARAH-SVRG Algorithms

**Motivation:** As indicated in Subsection 4.5 Algorithm 1 using hybrid SARAH-SGD estimators can be applied to solve the finite-sum problem (2), but it requires Assumption 2.3 to hold. In this section, we employ SVRG estimator instead of SGD to develop algorithms for solving (2) without Assumption 2.3.

#### 6.1 Hybrid SARAH-SVRG algorithm for the bounded domain case

We first consider the case \( \mathcal{X} := \text{dom}(\psi) \) is bounded, i.e. Assumption 2.4 holds. As a concrete example, if \( \psi(x) = \delta_{\mathcal{X}}(x) \), the indicator of a nonempty, closed, convex, and bounded set \( \mathcal{X} \) in \( \mathbb{R}^p \). In this case, we can modify Algorithm 1 at the following steps to solve (2).

- Evaluate \( v_0 := \nabla f(x_0) = \frac{1}{m} \sum_{i=0}^{m} \nabla f(x_0) \).
- Replace \( v_t \) at Step 7 by

\[
v_t := \beta v_{t-1} + \beta (\nabla f_{i_t}(x_t) - \nabla f_{i_t}(x_{t-1})) + (1 - \beta t_{-1}) u_{t,svrg}^{\psi},
\]

where

\[
u_{t,svrg} := v_0 + \nabla f_{j_t}(x_t) - \nabla f_{j_t}(x_0).
\]

Here \( i_t, j_t \in [n] \) are two independent indices generated uniformly randomly in \( [n] \).

Based on this modification, we prove in Appendix 4.1 the following theorem.

**Theorem 6.1.** Assume that Assumptions 2.1 2.2 2.3 and 2.4 hold for \( f \) and \( \psi \) in the finite-sum problem (2). Let \( \{x_t\}_{t=0}^{m} \) be the sequence generated by Algorithm 1 to solve (2) using the initial mini-batch size \( b := n \) at Step 3. The SVRG estimator for \( v_t \) as in (13), the constant weight \( \beta_t = \beta := 1 - \frac{c_1}{(m+1)^{2/3}} \), \( n_t := \eta \in (0, \frac{3}{4}) \), and \( \gamma_t \) is given by (50), for some \( 0 < c_1 \leq (m+1)^{2/3} \).

(a) Let \( \mathcal{T}_m \sim \mathcal{U}_p \{ \{x_t\}_{t=0}^{m} \} \) be the output of Algorithm 2. Then, after \( m \) iterations, we have

\[
\mathbb{E} \left[ \| G_0(\mathcal{T}_m) \|^2 \right] \leq \frac{4L\sqrt{1 + 2\delta}}{\delta^2} \left( \frac{F(x_0) - F^*}{\sqrt{m} \max(1, m+1)^{2/3}} \right) + \frac{17c_1L^2\mathcal{D}^2_X}{9\eta^2 \max(1, m+1)^{2/3}}.
\]

(b) Consequently, if \( \mathcal{D}^2_X \leq O(1) \), then, by defining \( \Delta_0 := \frac{4L\sqrt{1 + 2\delta}}{\delta^2} \left( \frac{F(x_0) - F^*}{\sqrt{m} \max(1, m+1)^{2/3}} \right) + \frac{17c_1L^2\mathcal{D}^2_X}{9\eta^2 \max(1, m+1)^{2/3}} \),

for any \( \epsilon > 0 \), the number of iterations \( m \) to obtain \( \mathcal{T}_m \) such that \( \mathbb{E} \left[ \| G_0(\mathcal{T}_m) \|^2 \right] \leq \epsilon^2 \) is at
most m := \left\lfloor \frac{\Delta^{3/2}}{\bar{c}^2} \right\rfloor = O \left( \frac{1}{\bar{c}^3} \right). \] This is also the total number of proximal operations \( \text{prox}_{q\theta} \).

In addition, the total number of stochastic gradient evaluations \( \nabla f_i(x_t) \) is at most

\[ T_m := n + \left\lfloor \frac{4\Delta_0^{3/2}}{\bar{c}^3} \right\rfloor = O \left( n + \frac{1}{\bar{c}^3} \right). \]

**Remark 6.1** (Best-known complexity). From Theorem 6.1 we can see that if \( \varepsilon > \frac{1}{\sqrt{n}} \) (e.g., when \( n \) is big), then the complexity bound \( O \left( n + \frac{1}{\bar{c}^3} \right) \) is better than the best-known one \( O \left( n + \frac{\Delta_0^{3/2}}{\bar{c}^2} \right) \) in [22, 53, 56, 68]. As a compensation, Assumption 2.4 is needed. However, Algorithm 1 has one loop compared to the double-loop methods in [22, 53, 56, 68].

**Remark 6.2** (Memory vs. computation). Note that if we store \( \nabla f_i(x_0) \) in (43), then we do not need to re-evaluate these values again to compute \( u_t^{\text{svrg}} \) in (43), and can save some computational time. However, we have to pay memory cost of storing \( \nabla f_i(x_0) \) for \( i = 1, \cdots, n \).

### 6.2 Loopless Hybrid SARAH-SVRG variant

The variant of Algorithm 1 stated in Theorem 6.1 relies on Assumption 2.4. To remove this assumption, we make the following modification of Algorithm 1. Instead of fixing the snapshot point \( \tilde{x}_t \) of the SVRG estimator \( u_t^{\text{svrg}} \) at Step 5 of Algorithm 1 as in Algorithm 1, we update it regularly with a positive probability \( p_0 \in (0, 1) \), which will be determined later.

The detailed algorithm is presented in Algorithm 3, where we use a constant weight \( \beta \in (0, 1) \), and constant step-sizes \( \gamma \in (0, 1] \) and \( \eta > 0 \) for simplicity of our analysis.

**Algorithm 3** (Loopless Hybrid SARAH-SVRG Algorithm for solving (2))

1: **Initialization:** An initial point \( x^0 \) and parameters \( \bar{b} \in \mathbb{N}_+, \beta, \gamma \in (0, 1), \) and \( \eta > 0 \).
2: Evaluate the full gradient \( v_0 := \nabla f(x_0) \) at \( x_0 \). Set \( \bar{x}_0 := x_0 \).
3: Update \( \hat{x}_1 := \text{prox}_{q\theta} (x_0 - \eta v_0) \) and \( x_1 := (1 - \gamma)x_0 + \gamma \hat{x}_1 \).
4: For \( t := 1, \cdots, m \) do
5: Generate a proper sample pair \((i_t, j_t)\) independently (single sample or mini-batch).
6: Evaluate an SVRG estimator \( u_t^{\text{svrg}} := \nabla f(\bar{x}_t) + \nabla f_{j_t}(x_t) - \nabla f_{j_t}(\bar{x}_t) \).
7: Evaluate a hybrid estimator \( v_t := \beta v_{t-1} + \beta (\nabla f_{i_t}(x_t) - \nabla f_{i_t}(x_{t-1})) + (1 - \beta) u_t^{\text{svrg}} \).
8: Update \( \hat{x}_{t+1} := \text{prox}_{q\theta} (x_t - \eta v_t) \) and \( x_{t+1} := (1 - \gamma)x_t + \gamma \hat{x}_{t+1} \).
9: Update the snapshot point \( \bar{x}_t \) as

\[ \bar{x}_{t+1} := \begin{cases} x_{t+1} & \text{with probability } p_0 \\ \hat{x}_t & \text{with probability } 1 - p_0. \end{cases} \]

10: EndFor
11: Choose \( \bar{x}_m \) uniformly randomly from \( \{x_0, x_1, \cdots, x_m\} \).

We can view Algorithm 3 as a double-loop variant with a random epoch length determined by the probability \( p_0 \). However, we only update the snapshot point \( \bar{x}_t \) for the SVRG term but do not restart the SARAH term in our estimator \( v_t \).

The following theorem states the convergence of Algorithm 3 in the single-sample case. Its proof is provided in Appendix 4.2.

**Theorem 6.2.** Assume that Assumptions 2.1 and 2.2 hold for (2). Let \( \{x_t\}_{t=0}^m \) be the sequence...
generated by Algorithm 3 for solving (2) using $\tilde{x}_0 := x_0$ and the following configuration:

$$
\begin{aligned}
\beta & := 1 - \frac{2\eta}{\mathcal{L}} \\
\gamma & := \frac{3\eta c_1^{1/2}c_2^{3/2}}{\sqrt{\mathcal{L}(2c_1^2 + c_2^3)}} \sqrt{\frac{1}{n}} \\
\eta & := \frac{2}{4 + \mathcal{L}} \\
\rho_0 & := \frac{c}{n^{2/3}}.
\end{aligned}
$$

(45)

where $c_1$ and $c_2$ satisfy $0 < c_1 \leq n$, $0 < c_2 < n^{2/3}$, and $9c_1c_2^3 \leq 17\mathcal{L}(2c_1^2 + c_2^3)$. Then, for $\tau_m \sim U([x_t]_t=0)$, we obtain the following bound:

$$
\mathbb{E} \left[ \|G_\eta(\tau_m)\|^2 \right] \leq \frac{2\sqrt{17}\mathcal{L}(2c_1^2 + c_2^3)^{1/2} \sqrt{n}}{3\eta c_1^{1/2}c_2^{3/2}(m + 1)} [F(x_0) - F^*].
$$

(46)

Consequently, for any $\varepsilon > 0$, the number of iterations $m$ for Algorithm 3 to reach $\mathbb{E} \left[ \|G_\eta(\tau_m)\|^2 \right] \leq \varepsilon^2$ does not exceed

$$
m + 1 := \left[ \Delta_0 \frac{\sqrt{n}}{\varepsilon^2} \right] = O \left( \frac{\sqrt{n}}{\varepsilon^2} \right), \quad \text{where} \quad \Delta_0 := \frac{2\sqrt{17}\mathcal{L}(2c_1^2 + c_2^3)^{1/2}}{3\eta c_1^{1/2}c_2^{3/2}} [F(x_0) - F^*].
$$

This is also the total number of proximal operations $\text{prox}_{\varphi_\eta}$. The total number $\mathcal{T}_m$ of stochastic gradient evaluations $\nabla f_i(x_t)$ does not exceed

$$
\mathcal{T}_m := \frac{4\Delta_0 \sqrt{n}}{\varepsilon^2} + c_2 n^{5/6} \Delta_0 \frac{\varepsilon}{\varepsilon^2} = O \left( \frac{\sqrt{n}}{\varepsilon^2} + \frac{n^{5/6}}{\varepsilon^2} \right).
$$

Theorem 6.2 provides a different complexity bound compared to existing methods such as \cite{39, 50, 60, 68}. Unfortunately, the stochastic gradient complexity bound of Theorem 6.2 seems to be worse than the best-known $O \left( n + \frac{\sqrt{n}}{\varepsilon^2} \right)$ one achieved by \cite{50, 68}. We believe that this happens due to an artifact of our analysis, and not because of the limit of Algorithm 3.

7 Numerical experiments

In this section, we provide three examples to illustrate the performance of our algorithms and compare them with several existing state-of-the-art methods. We use different configurations of parameters to investigate the empirical advantages and disadvantages of our methods.

7.1 Implementation details and configuration

Algorithms and competitors: We implement the following variants of our algorithms:

- Algorithm 1 with constant stepsizes stated in Theorem 4.1. We denote it by HybridSGD-SL. The parameters are set as suggested by Theorem 4.1. For the mini-batch variants stated in Theorem 5.1, we fix $\gamma := 0.95$, and choose mini-batch sizes as suggested in Remark 5.1. We also test different initial mini-batches $b \in \{n^{1/3}, n^{1/2}, n^{2/3}, 0.5n, n\}$.
- Algorithm 2 with constant step-sizes as stated in Theorem 4.3. We skip the adaptive variant of Algorithm 1 stated in Theorem 4.2 since it is already integrated in HybridSGD-DL. We set $\eta$ and $\beta$ from Theorem 4.3 with $\gamma := 0.95$. We also run this algorithm with different initial mini-batches as $b \in \{n^{1/2}, n^{2/3}, n^{3/4}, 0.5n, n\}$ and with an epoch length $m := \frac{n}{2}$. We select the best variant among these choices for each dataset and denote it by HybridSGD-DL.
- Algorithm 3 with SVRG estimator as stated in Theorem 6.1. We use this algorithm to only solve constrained problems of the form (2) and denote it by HybridSVRG-SL. In this algorithm, the parameters $\eta$ and $\beta$ are set as in Theorem 6.1 while $\gamma$ is fixed at $\gamma := 0.95$.
- Algorithm 3 with constant step-sizes as stated in Theorem 6.2. We denote this algorithm by HybridSVRG-DL. We set $\gamma := 0.95$, the probability $p_0 := \frac{1}{n}$ and follow Theorem 6.2 to set $\beta$ and $\eta$. 

19
For these algorithms, we also implement their mini-batch variants. We will normalize datasets so that the average-Lipschitz constant $L$ in our experiment is $L = L_\ell$, the Lipschitz constant of the outer loss function $\ell$ specified in the sequel.

For comparison, we also implement the following algorithms from the literature:

- The proximal stochastic gradient methods, e.g. from [20] with constant and diminishing step-sizes $\eta := \eta_0 > 0$ and $\eta := \frac{\eta_0}{1 + \sqrt{n}}$, respectively, where $\eta_0 > 0$ and $\eta' \geq 0$ that will be tuned in each experiment. We denote the SGD variant with constant stepsize by \texttt{ProxSGD1} using $\eta' = 0$, and the SGD scheme with diminishing step-size by \texttt{ProxSGD2} using $\eta'' > 0$. Without further specification, we will set $\hat{\eta} := 1.0$ and $\eta_0 := 0.01$ when the minibatch size $b = 1$, whereas $\eta_0 := 0.1$ when $b > 1$ which allows us to obtain consistent performance.

- We also implement the proximal SpiderBoost methods in [68], which works well in several examples, see [56]. Note that this algorithm can be viewed as an instance of ProxSARAH in [59], where we skip comparing with other variants here. We denote it by \texttt{ProxSpiderBoost}.

In this algorithm, we choose the constant step-size $\eta := \frac{1}{2L}$ and choose the optimal mini-batch size $b := O(\sqrt{n})$ and epoch length $m := O(\sqrt{n})$ as suggested by the authors.

- Another algorithm is the proximal SVRG scheme in [39, 60] and we denote it by \texttt{ProxSVRG}.

For the single-sample case, the step-size $\eta$ is set to $\eta := \frac{1}{2L}$ as suggested by [60]. For the mini-batch case, we choose $\eta := \frac{1}{bL}$ and the mini-batch size $b := O(\sqrt{n})^{2/3}$ as in [39, 60].

All the algorithms are implemented in Python running on a single node of a Linux server (called Longleaf) with configuration: 3.40GHz Intel processors, 30M cache, and 256GB RAM. For the last experiment, we implement these algorithms in TensorFlow (https://www.tensorflow.org) running on a GPU system. Since each algorithm use different value of stepsize $\eta$, we pick a fixed value $\eta = 0.5$ to compute the norm of gradient mapping $\|G_i(x^{(i)}_*)\|$ for visualization and report in all methods. We run the first and second examples up to $20 \sim 40$ epochs, respectively whereas we increase it up to 60 epochs in the last example.

Datasets: Several datasets used in this paper are from [14], which are available online at https://www.csie.ntu.edu.tw/~cjlin/libsvm/. We use 6 datasets as follows:

- **Small and medium datasets**: Three different well-known datasets: \texttt{w8a} ($n = 49,749$, $p = 300$), \texttt{rcv1.binary} ($n = 20242$, $p = 47236$), and \texttt{real-sim} ($n = 72309$, $p = 20958$).

- **Large datasets**: We also test these algorithms on larger datasets: \texttt{url_combined} ($n = 2,396,130; p = 3,231,961$), \texttt{epsilon} ($n = 400,000; p = 2,000$), and \texttt{news20.binary} ($n = 19,996; p = 1,355,191$).

Another well-known dataset is \texttt{mnist} downloaded from [20].

### 7.2 Nonnegative principal component analysis

The first example is a non-negative principal component analysis (NN-PCA) model studied in [60], which can be described as follows:

\[
    f^* := \min_{x \in \mathbb{R}^p} \left\{ f(x) := -\frac{1}{2n} \sum_{i=1}^{n} x^\top(z_i z_i^\top) x \mid \|x\| \leq 1, x \geq 0 \right\}. \quad (47)
\]

Here, \{\(z_i\)\}_{i=1}^{n} in \(\mathbb{R}^p\) is a given set of samples. By defining \(f_i(x) := -\frac{1}{2} x^\top(z_i z_i^\top) x\) for \(i = 1, \ldots, n\), and \(\psi(x) := \delta_{\mathcal{X}}(x)\), the indicator of \(\mathcal{X} := \{x \in \mathbb{R}^p \mid \|x\| \leq 1, x \geq 0\}\), we can formulate (47) into (2). Moreover, since \(z_i\) is normalized, the Lipschitz constant of \(\nabla f_i\) is \(L = 1\) for \(i = 1, \ldots, n\).

**Configuration**: In this experiment, we set the learning rate of \texttt{ProxSGD1} as $\eta_0 := 0.05$ which is smaller that its diminishing variant \texttt{ProxSGD2} where $\eta_0 := 0.1$. The parameters of other algorithms are set as stated in Subsection 7.1. We use the mini-batch size $b := 50$ for \texttt{w8a}, \texttt{rcv1-binary}, \texttt{real-sim}, and \texttt{news20.binary}, $b := 300$ for \texttt{epsilon}, and $b := 500$ for \texttt{url_combined}. For \texttt{ProxSpiderBoost} and \texttt{ProxSVRG}, we set the mini-batch sizes as stated in Subsection 7.1.

**Small and medium datasets**: Our first experiment is to run multiple single-loop variants of Algorithm 1 constant step-sizes (HybridSGD-SL) and HybridSVRG variants (HybridSVRG-SL)
to select the best one. We also compare them with ProxSGD1 and ProxSGD2. The results are reported in Figure 1 for three different datasets: w8a, rcv1-binary, and real-sim.

**Figure 1:** The training loss and gradient mapping norms of (47): Single-loop with single-sample.

As we can observe from Figure 1 that our HybridSGD variant works relatively well and outperforms ProxSGD both in constant and diminishing step-sizes. However, it is then slow down or saturated at a certain value of the loss function due to the effect of the SGD term $u_t$ in our estimator $v_t$. Our HybridSVRG-SL variant works better in this example which shows that the SVRG term really reduces the variant and helps to get lower training loss as well as smaller gradient mapping norms. Note that the performance of ProxSGD variants depends on the step-size. Here, we have tried to pick the one that works well for all three datasets in different settings.

Now, if we test our algorithms and ProxSGD schemes with mini-batches on the same datasets, then we obtain the results as reported in Figure 2. Here, we choose the same mini-batch sizes for our methods and ProxSGD variants as described above.

**Figure 2:** The training loss and gradient mapping norms of (47): Single-loop with mini-batch.
Again, we observe a similar performance as in the single-sample case. The HybridSVRG-SL variant still works well and outperforms other methods. Our HybridSGD-SL scheme is slightly better than ProxSGDs. Note that (47) satisfies Assumption 2.4. Therefore, the convergence of HybridSVRG-SL is guaranteed by Theorem 6.1.

Next, we test the double-loop variants in both single-sample and mini-batch cases. Our results for the single-sample case are revealed in Figure 3. Clearly, due to small step-sizes in the single-sample case, ProxSVRG [39, 60] performs quite poorly. HybridSVRG-SL with double-loop still outperforms all other methods. Our HybridSGD is again saturated after a few epochs, perhaps due to the unbiased SGD term and the choice of the weight $\beta$ that is far from $1$.

If we test these algorithms with mini-batch, then the results are given in Figure 4. Here, we also add ProxSpiderBoost since it uses a large step-size $\eta = \frac{1}{2L}$ and mini-batch. It also works well in many cases as observed in [56].
Figure 4 shows that our HybridSVRG-SL is comparable with the best method, ProxSpiderBoost, in the mini-batch case. ProxSVRG works better in the mini-batch case since it uses larger step-size, but still slower than HybridSVRG and ProxSpiderBoost in this test.

**Large datasets:** Finally, we test the mini-batch variants on three larger datasets: url_combined, epsilon, and news20.binary. Figure 5 shows the convergence of four single-loop algorithms on these three datasets.

Figure 5: The training loss and gradient norms of (47): Single-loop with mini-batch.

We still see that our hybrid variants still work better than ProxSGD schemes in all three datasets. Again, the HybridSVRG variant highly outperforms other candidates in this example.

### 7.3 Binary classification with nonconvex models

In this example, we consider the following binary classification involving a nonconvex objective function and a convex regularizer:

\[
\min_{x \in \mathbb{R}^p} \left\{ F(x) := \frac{1}{n} \sum_{i=1}^{n} \ell(a_i^T x, b_i) + \psi(x) \right\},
\]

where \(\{(a_i, b_i)\}_{i=1}^{n} \subset \mathbb{R}^p \times \{-1, 1\}^n\) is a given training dataset, \(\psi\) is a given convex regularizer, and \(\ell(\cdot, \cdot)\) is a given smooth and nonconvex loss function. By setting \(f_i(w) := \ell(a_i^T w, b_i)\) and choosing a convex regularizer \(\psi\), we obtain the form (2) that fulfills Assumptions 2.1 and 2.2.

We consider the following settings for the choice of \(\ell\) and \(\psi\), where three first models were studied in [72], and the last one has been used in [43]:

- **Normalized sigmoid loss:** \(\ell_1(s, \tau) := 1 - \tanh(\tau s)\) for a given \(\omega > 0\) and \(\psi(x) := \lambda \|x\|_1\).
  Here, \(\ell_1(\cdot, \cdot)\) is \(L\)-smooth with respect to \(s\), where \(L \approx 0.7698\).
- **Nonconvex loss in 2-layer neural networks:** \(\ell_2(s, \tau) := \left(1 - \frac{1}{1 + \exp(-\tau s)}\right)^2\) and \(\psi(x) := \lambda \|x\|_1\).
  This function is also \(L\)-smooth with \(L = 0.15405\).
- **Logistic difference loss:** \(\ell_3(s, \tau) := \ln(1 + \exp(-\tau s)) - \ln(1 + \exp(-\tau s - 1))\) and \(\psi(x) := \lambda \|x\|_1\).
  This function is \(L\)-smooth with \(L = 0.092372\).
- **Lorenz loss [43]:** \(\ell_4(s, \tau) := \ln(1 + (\tau s - 1)^2)\) if \(\tau s \leq 1\), and \(\ell_4(s, \tau) = 0\), otherwise, and \(\psi(x) = \lambda \|x\|_1\).
  This function is \(L\)-smooth with \(L = 4\).

We set the regularization parameter \(\lambda := \frac{1}{n}\) in all the tests which gives us relatively sparse solutions. We test the above algorithms on different scenarios ranging from small to large datasets.
Single-loop schemes with single-sample: Our first experiment for (48) is on single-loop variants with single-sample. To be consistent, we use the same three datasets as in the first example. The result of our first test is shown in Figure 6 for four algorithms on the training loss and the gradient mapping norms.

![Figure 6](image)

**Figure 6:** The training loss and gradient mapping norms of (48) with the nonconvex training loss $\ell_1$: Single-loop with single-sample.

For the loss $\ell_1$ and 20 epochs, both $\text{ProxSGD1}$ and $\text{ProxSGD2}$ give better training loss values for the $\text{w8a}$ dataset, but have worse gradient mapping norms than our single-loop variants. Our methods (two variants) work better than these SGD variants on the two remaining datasets: $\text{rcv1\_train\_binary}$ and $\text{real\_sim}$.

For the other losses: $\ell_2$, $\ell_3$, and $\ell_4$, the results are shown in Figure 7 for the training loss values and in Figure 8 for the norm of gradient mappings.

As we can observe from Figures 7 and 8 that, for the losses $\ell_2$ and $\ell_4$, our hybrid schemes perform worse than the $\text{ProxSGD}$ variants on the $\text{w8a}$ dataset, but they work better than $\text{ProxSGD}$ on the other two datasets. Or methods are comparable with the $\text{ProxSGD}$ variants on the loss $\ell_4$.

**Single-loop with mini-batches with the loss $\ell_2$ on small datasets:** We also test these algorithms with mini-batches, but this time we use the loss $\ell_2$. The results are shown in Figure 9. We test two different single-loop variants with constant step-size, and compare them with the mini-batch variants of $\text{ProxSGD}$.

Figure 9 shows that our methods work relatively well on three datasets with 40 epochs. They significantly outperform $\text{ProxSGD2}$ in both the constant and diminishing step-sizes.

**Mini-batch on large datasets:** For the three larger datasets, we only test the mini-batch variants, and the results are shown in Figure 10.

Figure 10 shows that our single-loop hybrid schemes greatly outperform both the $\text{ProxSGD}$ variants with the loss $\ell_1$ on three given datasets. If we test the loss $\ell_2$, then the results are given in Figure 11. Such results are similar to the $\ell_1$ loss, meaning that our methods work much better than the two $\text{ProxSGD}$ variants.

To confirm the performance of our hybrid schemes in the mini-batch case, we further test them and compare with two proximal SGD methods on the losses $\ell_3$ and $\ell_4$. The results are shown in Figures 12 and 13, respectively.

Clearly, Figures 12 and 13 still show that our single-loop hybrid methods work better than $\text{ProxSGD}$ on all the three large datasets.
Figure 7: The training loss of (48) with three losses: $\ell_2$ (first row), $\ell_3$ (middle row), and $\ell_4$ (last row): Single-loop with single sample.

Figure 8: The norm of gradient mappings of (48) with three losses: $\ell_2$ (first row), $\ell_3$ (middle row), and $\ell_4$ (last row): Single-loop with single sample.
Figure 9: The training loss and gradient norms of (48) with the loss \( \ell_2 \): Single-loop with mini-batch.

Figure 10: The training loss and gradient norms of (48) with the \( \ell_1 \)-loss: Single-loop with mini-batch.

Figure 11: The training loss and gradient norms of (48) with the \( \ell_2 \)-loss: Single-loop with mini-batch.
Figure 12: The training loss and gradient norms of (48) with the $\ell_3$-loss: Single-loop with mini-batch.

Figure 13: The training loss and gradient norms of (48) using the loss $\ell_4$: Single-loop with mini-batch.

Figure 14: The training loss and gradient norms of (48) using the loss $\ell_4$: Double-loop with mini-batch.
Finally, we test the double-loop variants on three large datasets using the loss $\ell_i$ with mini-batch. The results of the four algorithms are shown in Figure 14. Clearly, we can see that Algorithm 2, HybridSGD-DL, works much better than both ProxSVRG and ProxSpiderBoost. While ProxSpiderBoost works well in this example, ProxSVRG performs quite poorly.

7.4 Feedforward neural network training problem

In the last example, we consider the following composite nonconvex optimization problem obtained from a fully connected feedforward neural network training task:

$$\min_{x \in \mathbb{R}^p} \left\{ F(x) := \frac{1}{n} \sum_{i=1}^n \ell(F_L(x, a_i), b_i) + \psi(x) \right\}, \quad (49)$$

where we concatenate all the weight matrices and bias vectors of the neural network in one vector of variable $x$, $\{(a_i, b_i)\}_{i=1}^n$ is a training dataset, $F_L(\cdot)$ is a composition between all linear transforms and activation functions as $F_L(x, a) := \sigma_L(W_L \sigma_{L-1}(W_{L-1} \cdots \sigma_0(W_0a + \mu_0) \cdots) + \mu_L)$, where $W_i$ is a weight matrix, $\mu_i$ is a bias vector, $\sigma_i$ is an activation function, $L$ is the number of layers, $\ell(\cdot)$ is a cross-entropy loss, and $\psi(x) := \lambda \|x\|_1$ is the $\ell_1$-norm regularizer for some $\lambda > 0$ to obtain sparse weights. By defining $f_i(x) := \ell(F_L(x, a_i), b_i)$ for $i = 1, \cdots, n$, we can formulate (49) into the composite finite-sum setting (2).

We implement a mini-batch variants of Algorithm 1 and Algorithm 2, and two other mini-batch methods: ProxSVRG and ProxSpiderBoost in Tensorflow and use a well-known dataset mnist to evaluate their performance.

In the first experiment, we use an one-hidden-layer fully connected neural network: $784 \times 128 \times 10$, while in the second test, we increase the number of neurons in the hidden layer to obtain another fully connected neural network: $784 \times 800 \times 10$. The activation function $\sigma_i$ of the hidden layer is ReLU, and in the output layer is a soft-max one.

Our experiment configuration is as follows. We choose $\lambda := \frac{1}{n}$ to obtain sparse weights. We set $\gamma = 0.95$ for all methods and tune $\eta$ to obtain the best results. Here, we obtain $\eta = 0.3$ for HybridSGD-SL and HybridSGD-DL. We also tune $\eta$ in ProxSpiderBoost and ProxSVRG to obtain the best results. We finally get $\eta = 0.12$ for ProxSpiderBoost and $\eta = 0.2$ for ProxSVRG. We set $b := 100$ for our algorithms, $b := \lfloor \sqrt{n} \rfloor$ for ProxSpiderBoost, and $b := \lceil n^{2/3} \rceil$ for ProxSVRG and set the epoch length $m := \lfloor \frac{n}{2} \rfloor$. The performance of the four algorithms on the first network example is reported in Figure 15.

![Figure 15: The performance of 4 algorithms on the mnist dataset for solving (49). A fully connected 784 × 128 × 10 neural net.](image)

As we can see from Figure 15 that both the hybrid single- and double-loop algorithms work relatively well in this example, and outperform two other methods. Our methods achieve better training loss values, the norms of gradient mapping, and the test accuracy than both ProxSpiderBoost and ProxSVRG.

If we run these algorithms on the second neural network, then the results are given in Figure 16. We can observe the same behavior of these four algorithms as in Figure 15.
8 Conclusion

We have introduced a new hybrid approach to design stochastic estimators for smooth functions that covers both gradient and Hessian mappings of the objective function in stochastic optimization problems. We have proposed a hybrid biased and unbiased estimator that combines the SARAH estimator and one unbiased estimator such as SGD, SVRG, or SAGA. We have developed some key properties on the bound of variance of the new estimator that can be useful for algorithmic development. Then, we have applied our approach to develop a class of proximal stochastic gradient methods for solving stochastic composite nonconvex optimization problems. Our single-loop method significantly achieves better complexity than existing SGD schemes, while our double variant matches the best-known complexity bound in the literature. We have also studied several other variants including adaptive step-sizes and mini-batches. We have intensively tested our algorithms on three examples and compared them with some state-of-the-art methods. Our empirical evidence has shown that the new methods work relatively well and are comparable with these algorithms in a majority of experiments. We believe that our theory and approach can be extended to study second-order stationary points using negative curvature search procedures or cubic regularization methods. In addition, we also believe that our approach can potentially be extended to study other classes of stochastic optimization problems such as composition models and nonconvex constrained settings.

A Appendix: Properties of hybrid stochastic estimators

This appendix provides the full proof of our theoretical results in Section 3. However, we also need the following lemma in the sequel. Hence, we prove it here.

Lemma A.1. Given $L > 0$, $\delta > 0$, $\epsilon > 0$, and $\omega \in (0, 1)$, let \( \{\gamma_t\}_{t=0}^m \) be the sequence updated by

\[
\gamma_m := \frac{\delta}{L} \quad \text{and} \quad \gamma_t := \frac{\delta}{L + \epsilon L^2 \omega \gamma_{t+1} + \omega^2 \gamma_{t+2} + \cdots + \omega^{(m-t)} \gamma_m},
\]

for $t = 0, \cdots, m-1$. Then

\[
0 < \gamma_0 < \gamma_1 < \cdots < \gamma_m = \frac{\delta}{L} \quad \text{and} \quad \Sigma_m := \sum_{t=0}^m \gamma_t \geq \frac{\delta (m+1) \sqrt{1-\omega}}{L \left[ \sqrt{1-\omega} + \sqrt{1-\omega} + 4\delta \omega \right]}. \tag{51}
\]

Proof. First, from (50) it is obvious to show that $0 < \gamma_0 < \cdots < \gamma_m = \frac{\delta}{L}$. At the same time, since $\omega \in (0, 1)$, we have $1 \geq \omega \geq \omega^2 \geq \cdots \geq \omega^m$. By Chebyshev’s sum inequality, we have

\[
(m-t)(\omega^{t+1} \gamma_{t+1} + \omega^{t+2} \gamma_{t+2} + \cdots + \omega^{m-t} \gamma_m) \leq (\sum_{j=t+1}^m \gamma_j) (\omega + \omega^2 + \cdots + \omega^{m-t}) \leq \frac{L}{\omega} (\sum_{j=t+1}^m \gamma_j). \tag{52}
\]
From the update (50), we also have
\[
\begin{align*}
\epsilon L^2 \gamma_0 (\omega \gamma_1 + \omega^2 \gamma_2 + \cdots + \omega^m \gamma_m) &= \delta - L \gamma_0 \\
\epsilon L^2 \gamma_1 (\omega \gamma_2 + \omega^2 \gamma_3 + \cdots + \omega^{m-1} \gamma_m) &= \delta - L \gamma_1 \\
\vdots & \\
\epsilon L^2 \gamma_{m-1} \omega \gamma_m &= \delta - L \gamma_{m-1} \\
0 &= \delta - L \gamma_m.
\end{align*}
\]
(53)

Substituting (52) into (53), we get
\[
\begin{align*}
\omega t L^2 \gamma_0 (\gamma_0 + \gamma_1 + \cdots + \gamma_m) &\geq \delta m - m L \gamma_0 + \omega t L^2 \gamma_0^2 \\
\omega t L^2 \gamma_1 (\gamma_0 + \gamma_1 + \cdots + \gamma_m) &\geq \delta (m - 1) - (m - 1) L \gamma_1 + \omega t L^2 (\gamma_1 \gamma_0 + \gamma_1^2) \\
\vdots & \\
\omega t L^2 \gamma_{m-1} (\gamma_0 + \gamma_1 + \cdots + \gamma_m) &\geq \delta - L \gamma_{m-1} + \omega t L^2 (\gamma_{m-1} \gamma_0 + \cdots + \gamma_2^2) \\
\omega t L^2 \gamma_m (\gamma_0 + \gamma_1 + \cdots + \gamma_m) &\geq \delta - L \gamma_m + \omega t L^2 (\gamma_m \gamma_0 + \cdots + \gamma_2^2).
\end{align*}
\]

Let us define \( \Sigma_m := \sum_{i=0}^{m} \gamma_i \) and \( S_m := \sum_{i=0}^{m} \gamma_i^2 \). Summing up both sides of the above inequalities, we get
\[
\omega t L^2 \left( \frac{\sum_{i=0}^{2} \gamma_i^2}{1 - \omega} \right) \geq \frac{\delta (m^2 + m + 2)}{2} - L (m \gamma_0 + (m - 1) \gamma_1 + \cdots + \gamma_{m-1} + \gamma_m) + \frac{\omega t L^2}{2(1 - \omega)} (S_m + \Sigma_m^2).
\]

Using again Chebyshev's sum inequality, we have
\[
m \gamma_0 + (m - 1) \gamma_1 + \cdots + \gamma_{m-1} + \gamma_m \leq m^2 + m + 2 \frac{\sum_{i=0}^{m} \gamma_i}{2(m + 1)} = \frac{m^2 + m + 2}{2(m + 1)} \Sigma_m.
\]

Note that \((m + 1) S_m \geq \Sigma_m^2\) by Cauchy-Schwarz's inequality, which shows that \( S_m + \Sigma_m^2 \geq \frac{(m + 1)}{(m + 2)} (m + 2) \). Combining three last inequalities, we obtain the following quadratic inequation in \( \Sigma_m > 0\):
\[
\frac{m \omega t L^2}{(1 - \omega)} \Sigma_m^2 + L (m^2 + m + 2) \Sigma_m - \delta (m + 1)(m^2 + m + 2) \geq 0.
\]

Solving this inequation with respect to \( \Sigma_m > 0 \), we obtain
\[
\Sigma_m \geq \frac{(1 - \omega) \left( \sqrt{24(m + 1) - \pi^2} + \frac{4m^3 + 11m^2 + 11m + 2}{L \sqrt{1 - \omega}} \right) - (m^2 + m + 2)}{2(m + 1)}
\]
\[
\geq \frac{24(m + 1) - \pi^2 + \frac{24(m + 1) - \pi^2}{L \sqrt{1 - \omega}}}{L \sqrt{1 - \omega} + \sqrt{1 - \omega} \sqrt{\pi^2}} \quad \text{since} \quad \frac{m(m + 1)(m + 2)}{m^2 + m + 2} < 1.
\]

This proves (51).

1.1 The proof of Lemma 3.1: Variance estimate

By taking the expectation of both sides in (11) and using the fact that \( \xi_t \) and \( \zeta_t \) are independent, we can easily obtain (13).

To prove (14), let us first denote \( \delta_t := v_t - G(x_t), \hat{\delta}_t := u_t - G(x_t), \Delta_t := G(x_t) - G(x_{t-1}), \) and \( \hat{\Delta}_t := G(x_t) - G(x_{t-1}) \). Clearly, \( E_{\xi_t} [\Delta_{\xi_t}] = \Delta_t \) and \( E_{\xi_t} [\hat{\Delta}_t] = 0 \). Next, we write
\[
\begin{align*}
\delta_t &= v_t - G(x_t) = \beta_{t-1}(v_t - G(x_{t-1})) + \beta_{t-1}(\Delta_{\xi_t} - \Delta_t) + (1 - \beta_{t-1})[u_t - G(x_t)] \\
&= \beta_{t-1} \delta_{t-1} + \beta_{t-1}(\Delta_{\xi_t} - \Delta_t) + (1 - \beta_{t-1}) \hat{\delta}_t.
\end{align*}
\]
In this case, we have
\[
\|\delta_t\|^2 = \beta_{t-1}^2\|\delta_{t-1}\|^2 + \beta_t^2\|\Delta_{t-1}\|^2 + (1 - \beta_{t-1})^2\|\delta_t\|^2 \\
+ 2\beta_{t-1}\langle \delta_{t-1}, \Delta_{t-1} \rangle + 2\beta_{t-1}(1 - \beta_{t-1})\langle \delta_{t-1}, \delta_t \rangle + 2\beta_{t-1}(1 - \beta_{t-1})\langle \Delta_{t-1}, \Delta_t, \delta_t \rangle.
\]
Taking expectation w.r.t. \(\xi_t\) conditioned on \(\zeta_t\), and noting that \(E_{\xi_t}[\Delta_{t-1}] = \Delta_t\), we obtain
\[
E_{\xi_t}[\|\delta_t\|^2] = \beta_{t-1}^2\|\delta_{t-1}\|^2 + \beta_t^2E_{\xi_t}[\|\Delta_{t-1}\|^2] + (1 - \beta_{t-1})^2\|\delta_t\|^2 \\
+ 2\beta_{t-1}(1 - \beta_{t-1})\langle \delta_{t-1}, \delta_t \rangle.
\]
Taking the expectation over \(\zeta_t\), and noting that \(E_{(\xi_t, \zeta_t)}[\cdot] = E_{\xi_t}[E_{\zeta_t}[\cdot]]\), \(E_{\xi_t}[\|\Delta_{t-1}\|^2] = E_{\xi_t}[\|\Delta_{t}\|^2 - \|\Delta_t\|^2]\), and \(E_{\zeta_t}[\delta_t] = 0\), we obtain
\[
E_{(\xi_t, \zeta_t)}[\|\delta_t\|^2] = \beta_{t-1}^2\|\delta_{t-1}\|^2 + \beta_t^2E_{\xi_t}[\|\Delta_{t}\|^2] - \|\Delta_t\|^2 + (1 - \beta_{t-1})^2E_{\xi_t}[\|\delta_t\|^2],
\]
which is exactly (14) by substituting back the definitions of \(\delta_t, \Delta_t, \Delta_{t-1}\), and \(\delta_t\) defined above. □

1.2 The proof of Lemma 4.1: Upper bound of variance

We first upper bound (14) by using \(\sigma_t^2 := E_{\xi_t}[\|u_t - G(x_t)\|^2]\) and then taking the full expectation over \(F_{t+1} := \sigma(\xi_0, \xi_t, \ldots, \xi_t)\) as
\[
E[\|v_t - G(x_t)\|^2] \leq \beta_{t-1}^2E[\|v_{t-1} - G(x_{t-1})\|^2] + \beta_t^2E[\|G_{\xi_t}(x_t) - G_{\xi_t}(x_{t-1})\|^2] \\
+ (1 - \beta_{t-1})^2\sigma_t^2 \\
\leq \beta_t^2E[\|v_{t-1} - G(x_{t-1})\|^2] + \beta_{t-1}^2L^2E[\|x_t - x_{t-1}\|^2] + (1 - \beta_{t-1})^2\sigma_t^2.
\]
If we define \(a_t^2 := E[\|v_t - G(x_t)\|^2]\) and \(b_{t-1}^2 := E[\|x_t - x_{t-1}\|^2]\), then the above inequality can be rewritten as
\[
a_t^2 \leq \beta_{t-1}^2a_{t-1}^2 + L^2\beta_{t-1}^2b_{t-1}^2 + (1 - \beta_{t-1})^2\sigma_t^2.
\]
By induction, the last inequality implies
\[
a_t^2 \leq \beta_{t-1}^2a_{t-1}^2 + L^2\beta_{t-1}^2b_{t-1}^2 + (1 - \beta_{t-1})^2\sigma_t^2 \\
\leq \beta_{t-1}^2(\beta_{t-2}^2a_{t-2}^2 + L^2\beta_{t-2}^2b_{t-2}^2 + (1 - \beta_{t-2})^2\sigma_t^2) \\
+ L^2\beta_{t-1}^2(\beta_{t-2}^2b_{t-2}^2 + 2\beta_{t-2}^2b_{t-2}^2 + (1 - \beta_{t-2})^2\sigma_t^2) \\
= \beta_{t-1}^2(\beta_{t-2}^2a_{t-2}^2 + L^2(\beta_{t-2}^2b_{t-2}^2 + (1 - \beta_{t-2})^2\sigma_t^2)) \\
+ L^2\beta_{t-2}^2b_{t-2}^2 + (1 - \beta_{t-2})^2\sigma_t^2 \\
\ldots \\
\leq (\beta_{t-1}^2 \cdots \beta_0^2)a_0^2 + L^2(\beta_{t-1}^2 \cdots \beta_0^2)b_0^2 + L^2(\beta_{t-2}^2 \cdots \beta_0^2)b_1^2 + \cdots + L^2b_{t-1}^2b_{t-1}^2 \\
+ [(1 - \beta_{t-1})^2\sigma_t^2 + \beta_{t-2}^2(1 - \beta_{t-2})^2\sigma_t^2] + \beta_{t-1}^2b_{t-2}^2(1 - \beta_{t-3})^2\sigma_t^2 + \cdots + \beta_{t-1}^2b_{t-2}^2(1 - \beta_0)^2\sigma_t^2.
\]
Here, we use a convention that \(\prod_{i=t+1}^{\infty} \beta_{i-1} := 1\). As a result, the last expression can be written in the following compact form:
\[
a_t^2 \leq \left(\prod_{i=1}^t \beta_{i-1}^2\right)a_0^2 + L^2\sum_{i=0}^{t-1} \left(\prod_{j=i+1}^t \beta_{j-1}^2\right)b_i^2 + \sum_{i=0}^{t-1} \left(\prod_{j=i+1}^t \beta_{j-1}^2\right)(1 - \beta_i)^2\sigma_{i+1}^2. \quad (54)
\]
Define \(\omega_t := \prod_{i=1}^t \beta_{i-1}^2\), \(\omega_{i,t} := \prod_{j=i+1}^t \beta_{j-1}^2\), and \(S_t := \sum_{i=0}^{t-1} s_i = \sum_{i=0}^{t-1} (\prod_{j=i+1}^t \beta_{j-1}^2)(1 - \beta_i)^2\sigma_{i+1}^2\) with \(s_i := (1 - \beta_i)^2\sigma_{i+1}^2(\prod_{j=i+1}^t \beta_{j-1}^2)\). Then, we can rewrite (54) as
\[
a_t^2 \leq \omega_t a_0^2 + L^2 \sum_{i=0}^{t-1} \omega_{i,t} b_i^2 + S_t,
\]
which is exactly (17). □
1.3 The proof of Lemma 3.4: Variance estimate with mini-batch
Let $\Delta_{B_i} := \frac{1}{n_i} \sum_{x \in B_i} (G_{(x_i)} - G_{(x_{i-1})})$, $\Delta_t := G(x_t) - G(x_{t-1})$, $\delta_t := \hat{v}_t - G(x_t)$, and $\delta u_t := u_t - G(x_t)$. Clearly, we have
\[
E_{B_i}[\Delta_{B_i}] = \Delta_t \quad \text{and} \quad E_{B_i}[\delta u_t] = 0.
\]
Moreover, we can rewrite $\hat{v}_t$ as
\[
\hat{v}_t = \beta_{t-1}\hat{v}_{t-1} + \beta_{t-1}\Delta_{B_i} + (1 - \beta_{t-1})\delta u_t - \beta_{t-1}\Delta_t.
\]
Therefore, using these two expressions, we can derive
\[
E_{(B_i, \hat{v}_t)}[\|\hat{v}_t\|^2] = \beta_{t-1}^2\|\hat{v}_{t-1}\|^2 + \beta_{t-1}^2E_{B_i}[\|\Delta_{B_i}\|^2] + (1 - \beta_{t-1})^2E_{B_i}[\|\delta u_t\|^2] + \beta_{t-1}^2\|\Delta_t\|^2
+ 2\beta_{t-1}\langle \hat{v}_{t-1}, E_{B_i}[\Delta_{B_i}] \rangle + 2\beta_{t-1}(1 - \beta_{t-1})\langle \hat{v}_{t-1}, E_{B_i}[\delta u_t] \rangle - 2\beta_{t-1}^2(\hat{v}_{t-1}, \Delta_{B_i})
+ 2\beta_{t-1}(1 - \beta_{t-1})E_{(B_i, \hat{v}_t)}[\|\Delta_{B_i}\|^2] - 2\beta_{t-1}^2E_{B_i}[\|\Delta_{B_i}\|, \Delta_t]
- 2\beta_{t-1}(1 - \beta_{t-1})E_{B_i}[\|\Delta_{B_i}\|, \delta u_t], \Delta_t)
= \beta_{t-1}^2\|\hat{v}_{t-1}\|^2 + \beta_{t-1}^2E_{B_i}[\|\Delta_{B_i}\|^2] + (1 - \beta_{t-1})^2E_{B_i}[\|\delta u_t\|^2] - \beta_{t-1}^2\|\Delta_t\|^2.
\]
Similar to the proof of [56] Lemma 2, for the finite-sum case (i.e. $|\Omega| = n$), we can show that
\[
E_{B_i}[\|\Delta_{B_i}\|^2] = \frac{n(b_i - 1)}{(n - 1)b_i} \|\Delta_t\|^2 + \frac{(n - b_i)}{(n - 1)b_i} E_{\xi}[\|G_{\xi}(x_i) - G_{\xi}(x_{i-1})\|^2].
\]
For the expectation case, we have
\[
E_{B_i}[\|\Delta_{B_i}\|^2] = \left(1 - \frac{1}{b_i}\right) \|\Delta_t\|^2 + \frac{1}{b_i} E_{\xi}[\|G_{\xi}(x_i) - G_{\xi}(x_{i-1})\|^2].
\]
Using the definition of $\rho$ in Lemma 3.5, we can unify these two expressions as
\[
E_{B_i}[\|\Delta_{B_i}\|^2] = (1 - \rho) \|\Delta_t\|^2 + \rho E_{\xi}[\|G_{\xi}(x_i) - G_{\xi}(x_{i-1})\|^2].
\]
Substituting the last expression into the previous one, we obtain [20].

1.4 The proof of Lemma 3.5: Upper bound of mini-batch variance
From Lemma 3.4 taking the expectation with respect to $F_i := \sigma(B_0, B_0, \ldots, B_i, \hat{v}_t)$, we have
\[
E \left[ \|\hat{v}_t - G(x_t)\|^2 \right] \leq \beta_{t-1}^2E \left[ \|\hat{v}_{t-1} - G(x_{t-1})\|^2 \right]
+ \rho L^2\beta_{t-1}^2E \left[ \|x_t - x_{t-1}\|^2 \right] + (1 - \beta_{t-1})^2E_{B_i} \left[ \|u_t - G(x_t)\|^2 \right].
\]
In addition, from [56] Lemma 2, we have $E_{B_i} \left[ \|u_t - G(x_t)\|^2 \right] \leq \hat{\rho}E_{\xi} \left[ \|G_{\xi}(x_t) - G_{\xi}(x_{t-1})\|^2 \right] = \hat{\rho}\sigma_{\xi}^2$, where $\sigma_{\xi}^2 := E_{\xi} \left[ \|G_{\xi}(x_i) - G_{\xi}(x_{i-1})\|^2 \right]$. Let $a_t := E \left[ \|\hat{v}_t - G(x_t)\|^2 \right]$ and $b_t := E \left[ \|x_{t+1} - x_t\|^2 \right]$. Then, the above estimate can be upper bounded as follows:
\[
a_t^2 \leq \beta_{t-1}^2a_{t-1}^2 + \rho L^2\beta_{t-1}^2b_{t-1}^2 + \hat{\rho}(1 - \beta_{t-1})^2\sigma_{\xi}^2.
\]
By following inductive step as in the proof of Lemma 3.1, we obtain from the last inequality:
\[
a_t^2 \leq (\beta_{t-1}^2 \cdots \beta_0^2) a_0^2 + \rho L^2 (\beta_{t-1}^2 \cdots \beta_0^2) b_0^2 + \cdots + \rho L^2 \beta_{t-1}^2 b_{t-1}^2
+ \hat{\rho} [(\beta_{t-1}^2 \cdots \beta_0^2) (1 - \beta_0)^2\sigma_{\xi}^2 + \cdots + (1 - \beta_{t-1})^2\sigma_{\xi}^2].
\]
Using the definition of $\omega_t$, $\omega_{t,t}$, and $S_t$ from [18], the previous inequality becomes
\[
a_t^2 \leq \omega_t a_0^2 + \rho L^2 \sum_{i=0}^{t-1} \omega_{t,i}b_i^2 + \hat{\rho}S_t,
\]
which is the same as [21].

B The proof of technical results in Section 4

We provide the full proof of technical results in Section 4.

2.1 The proof of Lemma 4.1: Key estimate

From the update \( x_{t+1} := (1 - \gamma_t) x_t + \gamma_t \hat{x}_{t+1} \) at Step 8 of Algorithm 1, we have \( x_{t+1} - x_t = \gamma_t (\hat{x}_{t+1} - x_t) \). From the \( L \)-average smoothness condition in Assumption 2.2, one can write

\[
\begin{aligned}
f(x_{t+1}) &\le f(x_t) + \langle \nabla f(x_t), x_{t+1} - x_t \rangle + \frac{L}{2} \| x_{t+1} - x_t \|^2 \\
&= f(x_t) + \gamma_t \langle \nabla f(x_t), \hat{x}_{t+1} - x_t \rangle + \frac{L \gamma_t^2}{2} \| \hat{x}_{t+1} - x_t \|^2.
\end{aligned}
\]

Using convexity of \( \psi \), we can show that

\[
\psi(x_{t+1}) \le (1 - \gamma_t) \psi(x_t) + \gamma_t \psi(\hat{x}_{t+1}) \le \psi(x_t) + \gamma_t \langle \nabla \psi(\hat{x}_{t+1}), \hat{x}_{t+1} - x_t \rangle,
\]

where \( \nabla \psi(\hat{x}_{t+1}) \in \partial \psi(\hat{x}_{t+1}) \) is any subgradient of \( \psi \) at \( \hat{x}_{t+1} \).

Utilizing the optimality condition of \( \hat{x}_{t+1} = \text{prox}_{\eta t \psi}(x_t - \eta_t v_t) \), we can show that \( \nabla \psi(\hat{x}_{t+1}) = -v_t - \frac{1}{\eta_t} (\hat{x}_{t+1} - x_t) \) for some \( \nabla \psi(\hat{x}_{t+1}) \in \partial \psi(\hat{x}_{t+1}) \). Substituting this relation into (56), we get

\[
\psi(x_{t+1}) \le \psi(x_t) - \gamma_t \langle v_t, \hat{x}_{t+1} - x_t \rangle - \frac{\gamma_t}{\eta_t} \| \hat{x}_{t+1} - x_t \|^2.
\]

Combining (55) and (57), and using \( F(x) := f(x) + \psi(x) \) from (4), we obtain

\[
F(x_{t+1}) \le F(x_t) + \gamma_t \langle \nabla f(x_t), v_t, \hat{x}_{t+1} - x_t \rangle - \left( \frac{\gamma_t}{\eta_t} - \frac{L \gamma_t^2}{2} \right) \| \hat{x}_{t+1} - x_t \|^2.
\]

For any \( c_t > 0 \), we can always write

\[
\langle \nabla f(x_t) - v_t, \hat{x}_{t+1} - x_t \rangle = \frac{1}{2 c_t} \| \nabla f(x_t) - v_t \|^2 + \frac{c_t}{2} \| \hat{x}_{t+1} - x_t \|^2 - \frac{1}{2 c_t} \| \nabla f(x_t) - v_t - c_t (\hat{x}_{t+1} - x_t) \|^2.
\]

Utilizing this expression, we can rewrite as

\[
F(x_{t+1}) \le F(x_t) + \frac{\gamma_t}{2c_t} \| \nabla f(x_t) - v_t \|^2 - \left( \frac{\gamma_t}{\eta_t} - \frac{L \gamma_t^2}{2} - \frac{\gamma_t c_t}{2} \right) \| \hat{x}_{t+1} - x_t \|^2 - \frac{\tilde{\sigma}^2}{2}.
\]

where \( \tilde{\sigma}^2 := \frac{\gamma_t}{2c_t} \| \nabla f(x_t) - v_t - c_t (\hat{x}_{t+1} - x_t) \|^2 \ge 0 \).

Taking expectation both sides of this inequality over the entire history \( \mathcal{F}_{t+1} \), we obtain

\[
\mathbb{E}[F(x_{t+1})] \le \mathbb{E}[F(x_t)] + \frac{L \gamma_t}{2 c_t} \mathbb{E}[\| \nabla f(x_t) - v_t \|^2] - \left( \frac{\gamma_t}{\eta_t} - \frac{L \gamma_t^2}{2} - \frac{\gamma_t c_t}{2} \right) \mathbb{E}[\| \hat{x}_{t+1} - x_t \|^2] - \frac{1}{2} \mathbb{E}[\tilde{\sigma}^2].
\]

Next, from the definition of gradient mapping \( G_\eta(x) := \frac{1}{\eta} (x - \text{prox}_{\eta \psi}(x - \eta \nabla f(x))) \) in (39), we can see that

\[
\eta_t \| G_\eta(x_t) \| = \| x_t - \text{prox}_{\eta \psi}(x_t - \eta_t \nabla f(x_t)) \|.
\]

Using this expression, the triangle inequality, and the nonexpansive property \( \| \text{prox}_{\eta \psi}(z) - \text{prox}_{\eta \psi}(w) \| \le \| z - w \| \) of \( \text{prox}_{\eta \psi} \), we can derive that

\[
\eta_t \| G_\eta(x_t) \| \le \| \hat{x}_{t+1} - x_t \| + \| \text{prox}_{\eta \psi}(x_t - \eta_t \nabla f(x_t)) - \hat{x}_{t+1} \|
= \| \hat{x}_{t+1} - x_t \| + \| \text{prox}_{\eta \psi}(x_t - \eta_t \nabla f(x_t)) - \text{prox}_{\eta \psi}(x_t - \eta v_t) \|
\le \| \hat{x}_{t+1} - x_t \| + \eta_t \| \nabla f(x_t) - v_t \|.
\]

Now, for any \( r_t > 0 \), the last estimate leads to

\[
\eta_t^2 \mathbb{E}[\| G_\eta(x_t) \|^2] \le \left( 1 + \frac{1}{r_t} \right) \mathbb{E}[\| \hat{x}_{t+1} - x_t \|^2] + (1 + r_t) \eta_t^2 \mathbb{E}[\| \nabla f(x_t) - v_t \|^2].
\]
Multiplying this inequality by $\frac{\gamma t}{c_t} > 0$ and adding the result to (60), we finally get
\[
\mathbb{E}[F(x_{t+1})] \leq \mathbb{E}[F(x_t)] - \frac{2 \eta^2 t}{2} \mathbb{E} [\|G_{\eta_t}(x_t)\|^2] \\
+ \frac{1}{2} \left[ \frac{\gamma t}{c_t} + (1 + r_t) q_t \eta^2_t \right] \mathbb{E} [\|\nabla f(x_t) - v_t\|^2] \\
- \frac{1}{2} \left[ \frac{2 \eta^2 t}{\lambda t} - L \gamma_t^2 - \gamma_t c_t - q_t \left( 1 + \frac{1}{r_t} \right) \right] \mathbb{E} [\|\bar{x}_{t+1} - x_t\|^2] - \frac{1}{2} \mathbb{E} [\tilde{\sigma}_t^2].
\]
Using the definition of $\theta_t$ and $\kappa_t$ from (23), i.e.:
\[
\theta_t := \frac{\gamma t}{c_t} + (1 + r_t) q_t \eta^2_t \quad \text{and} \quad \kappa_t := \frac{2 \gamma_t}{\eta_t} - L \gamma_t^2 - \gamma_t c_t - q_t \left( 1 + \frac{1}{r_t} \right),
\]
can simplify this estimate as follows:
\[
\mathbb{E}[F(x_{t+1})] \leq \mathbb{E}[F(x_t)] - \frac{\eta^2 t^2}{2} \mathbb{E} [\|G_{\eta_t}(x_t)\|^2] + \frac{\eta^2}{2} \mathbb{E} [\|\nabla f(x_t) - v_t\|^2] \\
- \frac{\gamma t}{2} \mathbb{E} [\|x_{t+1} - x_t\|^2] - \frac{1}{2} \mathbb{E} [\tilde{\sigma}_t^2].
\]
This is exactly (22). \qed

2.2 The proof of Lemma 4.2: Key estimate of Lyapunov function
From (14), by taking the full expectation on the history $F_{t+1}$ and using the $L$-average smoothness of $f$, we can show that
\[
\mathbb{E}[\|v_{t+1} - \nabla f(x_{t+1})\|^2] \leq \beta_t^2 \mathbb{E}[\|v_t - \nabla f(x_t)\|^2] + \beta_t^2 L^2 \mathbb{E}[\|x_{t+1} - x_t\|^2] + (1 - \beta_t)^2 \sigma_{t+1}^2 \\
= \beta_t^2 \mathbb{E}[\|v_t - \nabla f(x_t)\|^2] + \beta_t^2 \gamma_t^2 L^2 \mathbb{E}[\|x_{t+1} - x_t\|^2] + (1 - \beta_t)^2 \sigma_{t+1}^2,
\]
where $\sigma_{t+1}^2 := \mathbb{E}[\|\nabla f_{\eta_t}(x_t) - \nabla f(x_t)\|^2]$. Let $V$ be the Lyapunov function defined by (24). Then, by multiplying (61) by $\frac{\eta^2 t}{2} > 0$, adding the result to (60), and then using this Lyapunov function we can show that
\[
V(x_{t+1}) \leq V(x_t) - \frac{2 \eta^2 t}{2} \mathbb{E}[\|G_{\eta_t}(x_t)\|^2] - \frac{1}{2}(\alpha_t - \beta_t^2 \alpha_{t+1} - \theta_t) \mathbb{E}[\|v_t - \nabla f(x_t)\|^2] \\
- \frac{1}{2}(\kappa_t - \alpha_t + \beta_t^2 \gamma_t^2 L^2) \mathbb{E}[\|x_{t+1} - x_t\|^2] + \frac{1}{2}(1 - \beta_t)^2 \alpha_{t+1} \sigma_{t+1}^2 - \frac{1}{2} \mathbb{E}[\tilde{\sigma}_t^2].
\]
Let us choose $\gamma_t$, $\eta_t$, and other parameters such that the conditions (25) hold, i.e.:
\[
\alpha_t - \beta_t^2 \alpha_{t+1} - \theta_t \geq 0 \quad \text{and} \quad \kappa_t - \alpha_{t+1} \beta_t^2 \gamma_t^2 L^2 \geq 0.
\]
In this case, (62) can be simplified as follows:
\[
V(x_{t+1}) \leq V(x_t) - \frac{\eta^2 t}{2} \mathbb{E}[\|G_{\eta_t}(x_t)\|^2] + \frac{1}{2} \alpha_{t+1} (1 - \beta_t)^2 \sigma_{t+1}^2.
\]
This proves (26).
Finally, summing up this inequality from $t := 0$ to $t := m$, we obtain
\[
\sum_{t=0}^{m} \frac{\eta^2 t}{2} \mathbb{E}[\|G_{\eta_t}(x_t)\|^2] \leq V(x_0) - V(x_{m+1}) + \frac{1}{2} \sum_{t=0}^{m} \alpha_{t+1} (1 - \beta_t)^2 \sigma_{t+1}^2.
\]
Note that $V(x_{m+1}) := \mathbb{E}[F(x_{m+1})] + \frac{\eta m}{2} \mathbb{E}[\|v_{m+1} - \nabla f(x_{m+1})\|^2] \geq \mathbb{E}[F(x_{m+1})] \geq F^*$ by Assumption 2.1 and $V(x_0) = F(x_0) + \frac{\eta}{2} \mathbb{E}[\|v_0 - \nabla f(x_0)\|^2]$. Using these estimates into (65), we obtain (27). \qed
2.3 The proof of Theorem 4.1: Constant step-sizes

(a) Let us first choose \( c_1 := 1, r_1 := 1, \) and \( q_t := \gamma_t \) in Lemma 4.1. We also fix \( \beta_t := \beta \in (0, 1), \) \( \eta_t := \eta > 0, \) and \( \gamma_t := \gamma \in (0, 1). \) In this case, we have

\[
\theta_t = \theta = (1 + 2\eta^2)\gamma \quad \text{and} \quad \kappa_t = \kappa = \gamma \left( \frac{2}{\eta} - L\gamma - 3 \right).
\]

First, to guarantee \( \kappa > 0, \) we need to choose \( \eta < \frac{2}{3 + L\gamma} \leq \frac{2}{3}. \) Hence, \( 1 + 2\eta^2 < 1 + \frac{8}{9} = \frac{17}{9}. \)

Next, let us choose \( \tilde{b} := \frac{1}{2}c_1^\alpha \sigma^{8/3}(m + 1)^{1/3} \) and \( \alpha_t = \alpha := c_2(m + 1)^{1/3} \) in (24) for some constant \( c_1 \geq \frac{1}{\sigma^{8/3}(m + 1)^{2/3}} \) and \( c_2 > 0, \) respectively. We also choose \( \beta := 1 - \frac{1}{\tilde{b}^{1/2}(m + 1)^{1/2}} = 1 - \frac{1}{c_1\sigma^{4/3}(m + 1)^{2/3}} \in (0, 1]. \)

Let us simplify the conditions (25) as

\[
(1 + 2\eta^2)\gamma \leq (1 - \beta^2)\alpha \quad \text{and} \quad \frac{2}{\eta} - L\gamma - 3 \geq \alpha_\gamma \beta^2 L^2.
\]

Since \( 1 + 2\eta^2 < \frac{17}{9}, \) the first condition holds if we choose \( \gamma \leq \tilde{\gamma} := \frac{9\alpha(1 - \beta^2)}{17}. \) With these choices of \( \tilde{b}, \alpha, \) and \( \beta, \) we can show that

\[
1 - \beta^2 = \frac{2}{c_1\sigma^{4/3}(m + 1)^{2/3}} - \frac{1}{c_1\sigma^{8/3}(m + 1)^{2/3}} \geq \frac{1}{c_1\sigma^{4/3}(m + 1)^{2/3}}.
\]

Consequently, we have

\[
\tilde{\gamma} = \frac{9\alpha(1 - \beta^2)}{17} \geq c_2(m + 1)^{1/3}, \quad \frac{9}{17c_1\sigma^{4/3}(m + 1)^{2/3}} = \frac{c_2}{17c_1\sigma^{4/3}(m + 1)^{1/3}}.
\]

Therefore, the condition \( \gamma \leq \tilde{\gamma} \) holds if we choose

\[
0 < \gamma \leq \frac{9c_2}{17c_1\sigma^{4/3}(m + 1)^{1/3}}. \tag{67}
\]

Now, we also choose \( \gamma \) such that \( \alpha_\gamma \beta^2 L^2 \leq 1 \) which leads to \( \gamma \leq \frac{1}{L\gamma^2}. \) Under this condition of \( \gamma, \) the second condition of (66) holds if we choose \( \eta := \frac{2}{\tilde{b}^{1/2}(m + 1)^{1/2}} \) as in (28). Clearly, since \( 0 < \gamma \leq 1, \) we have \( \frac{2}{\tilde{b}^{1/2}(m + 1)^{1/2}} \leq \eta \leq \frac{2}{\tilde{b}^{1/2}(m + 1)^{1/2}}. \)

Note that since \( \frac{1}{L\gamma^2} \geq \frac{1}{L\gamma^2} \geq \frac{1}{L\gamma^2} \geq \frac{1}{L\gamma^2} \geq \frac{1}{L\gamma^2} \geq \frac{1}{L\gamma^2}, \) the second condition of (66) holds if we choose

\[
0 < \gamma \leq \frac{1}{L^2c_2(m + 1)^{1/3}}. \tag{68}
\]

If choose \( c_2 := \sqrt{17c_1\sigma^{2/3}L}, \) then \( \frac{9c_2}{17c_1\sigma^{2/3}L} = \frac{1}{c_2L^2}. \) On the other hand, since \( c_1 \geq \frac{1}{\sigma^{2/3}(m + 1)^{2/3}} \), we have \( \frac{3}{\sqrt{17c_1\sigma^{2/3}(m + 1)^{2/3}}} \leq \frac{3}{\sqrt{17}} \) and \( \frac{1}{c_2} \leq \frac{1}{L}\). Therefore, if update

\[
\gamma := \sqrt{17c_1\sigma^{2/3}(m + 1)^{1/3}},
\]

as shown in (28), then \( \gamma \in (0, 1] \) and satisfies both (67) and (68). Therefore, (66) holds.

(b) From (27), we have

\[
\frac{1}{m+1} \sum_{t=0}^{m} \mathbb{E} \left[ \| G_t(x_t) \|^2 \right] \leq \frac{2}{\gamma(m+1)} \mathbb{E} \left[ \| F(x_0) - F^* \| \right] + \frac{\alpha}{\gamma^2(m+1)} \mathbb{E} \left[ \| v_t - \nabla f(x_t) \|^2 \right] + \frac{2\alpha(1 - \beta^2)\sigma^2}{\gamma^2}. \tag{69}
\]
Using the update rule (28), we can lower bound
\[ \gamma^2 \geq \frac{4\gamma}{(4 + L)^2} \geq \frac{12}{\sqrt{17c_1} L(4 + L)^2 \sigma^2(3(1 + 1)/3)}. \] (70)

In addition, we have \( \alpha = \frac{\sqrt{17c_1} \sigma^2/3}{4L} (m + 1)^{1/3} \) which implies that
\[ \frac{2\alpha}{\gamma^2} \geq \frac{17c_1 (4 + L)^2 \sigma^4/3 (m + 1)^{2/3}}{18}. \] (71)

Since \( v_0 \) is computed by Step 3, we also have
\[ \mathbb{E} \left[ \|v_0 - \nabla f(x_0)\|^2 \right] \leq \frac{\sigma^2}{b} = \frac{1}{c \sigma^2/3 (m + 1)^{1/3}}, \] (72)

by Lemma 2. Substituting (70), (71), and (72) into (69), and noting that \( \mathcal{P}_m = \cup \{x_t^m \}_{t=0} \), we can derive
\[ \mathbb{E} \left[ \|\mathcal{G}_0(\mathcal{P}_m)\|^2 \right] = \frac{1}{m+1} \sum_{t=0}^{m} \mathbb{E} \left[ \|\mathcal{G}_0(x_t)\|^2 \right] \leq \frac{\sigma^2}{b} = \frac{1}{c \sigma^2/3 (m + 1)^{1/3}}. \]

This proves (29).

(c) Finally, for any \( \varepsilon > 0 \), the number of iterations \( m \) to achieve \( \mathbb{E} \left[ \|\mathcal{G}_0(\mathcal{P}_m)\|^2 \right] \leq \varepsilon^2 \) can be estimated by letting:
\[ \frac{\Delta_0 \sigma^2/3}{(m + 1)^{2/3}} = \frac{\sqrt{17c_1} L(4 + L)^2 \sigma^2/3}{6(m + 1)^{2/3}} [F(x_0) - F^*] + \frac{17c_1 (4 + L)^2 \sigma^4/3}{9c_1 (m + 1)^{2/3}} \leq \varepsilon^2, \]

where \( \Delta_0 := \frac{\sqrt{17c_1} L(4 + L)^2}{6} [F(x_0) - F^*] + \frac{17c_1 (4 + L)^2 \sigma^4/3}{9c_1}. \) This implies that \( m + 1 \geq \frac{\Delta_0 \sigma^2/3}{\varepsilon^2} \). Therefore, we can choose \( m := \left[ \frac{\Delta_0 \sigma^2/3}{\varepsilon^2} \right] = \mathcal{O} \left( \frac{\Delta_0 \sigma^2}{\varepsilon^2} \right) \). This is also the number of proximal operations \( \text{prox}_{\eta \psi} \).

The total number of stochastic gradient evaluations \( \nabla f_L(x_t) \) is
\[ T_m := \tilde{b} + 3(m + 1) \leq c \sigma^2/3 (m + 1)^{1/3} + \frac{3\Delta_0 \sigma^2}{\varepsilon^2} = \frac{c_1 \Delta_0 \sigma^2}{\varepsilon^2} + \frac{3\sigma \Delta_0 \sigma^2}{\varepsilon^3} \]

Hence, we can choose \( T_m := \left[ \frac{c_1 \Delta_0 \sigma^2}{\varepsilon^2} + \frac{3\sigma \Delta_0 \sigma^2}{\varepsilon^3} \right] = \mathcal{O} \left( \frac{\sigma^2}{\varepsilon^2} + \frac{\sigma \sigma^2}{\varepsilon^3} \right) \), which proves (c).

2.4 The proof of Theorem 4.2: Adaptive step-sizes
Let \( \{x_t, \hat{x}_t\} \) be generated by Algorithm 1. Let us also choose \( c_t = r_t = 1 \) and \( q_t := \gamma_t \) and fix \( \gamma_t := \eta \in (0, \frac{4}{3}) \) in Lemma 4.3. In this case, we have
\[ \theta_t := (1 + 2\eta^2)\gamma_t \quad \text{and} \quad \kappa_t := \left( \frac{2}{\eta} - L\gamma_t - 3 \right) \gamma_t. \]
Using these parameters into (22) and then summing up the result from \( t = 0 \) to \( t = m \) and using (17) from Lemma 4.1, we obtain

\[
\mathbb{E}[F(x_{m+1})] \leq \mathbb{E}[F(x_0)] + \frac{L^2}{2} \sum_{t=0}^{m} \theta_t \sum_{i=0}^{t-1} \gamma_i^2 \omega_{i,t} \mathbb{E} \left[ \| \hat{x}_{i+1} - x_i \|^2 \right] \\
- \frac{1}{2} \sum_{t=0}^{m} \gamma_t \mathbb{E} \left[ \| \hat{x}_{t+1} - x_t \|^2 \right] - \sum_{t=0}^{m} \gamma_t \eta_t^2 \mathbb{E} \left[ \| G_t(x_t) \|^2 \right] \\
- \sum_{t=0}^{m} \mathbb{E} [\hat{\sigma}_t] + \frac{1}{2} \sum_{t=0}^{m} \theta_t \omega_t \sigma^2 + \frac{1}{2} \sum_{t=0}^{m} \theta_t S_t,
\]

where \( \sigma^2 := \mathbb{E} \left[ \| v_0 - \nabla f(x_0) \|^2 \right] \geq 0, \tilde{\sigma}^2 := \frac{\gamma_t}{2} \| \nabla f(x_t) - v_t - (\hat{x}_{t+1} - x_t) \|^2 \geq 0, \) and \( \omega_{i,t}, \omega_t, \) and \( S_t \) are defined in Lemma 4.1.

By ignoring the nonnegative term \( \mathbb{E}[\tilde{\sigma}_t^2] \), we can further estimate the last inequality as follows:

\[
\mathbb{E}[F(x_{m+1})] \leq \mathbb{E}[F(x_0)] - \frac{\eta^2}{2} \sum_{t=0}^{m} \gamma_t \mathbb{E} \left[ \| G_t(x_t) \|^2 \right] \\
+ \frac{(1+2\eta^2)\sigma}{2} \sum_{t=0}^{m} \omega_t \gamma_t + \frac{(1+2\eta^2)}{2} \sum_{t=0}^{m} \gamma_t S_t + \frac{T_m}{2},
\]

where \( T_m \) is defined as follows:

\[
T_m := L^2(1+2\eta)^2 \sum_{t=0}^{m} \gamma_t \sum_{i=0}^{t-1} \omega_{i,t} \gamma_i^2 \mathbb{E} \left[ \| \hat{x}_{i+1} - x_i \|^2 \right] - \sum_{t=0}^{m} \gamma_t \left( \frac{2}{\eta} - 3 - L\gamma_t \right) \mathbb{E} \left[ \| \hat{x}_{i+1} - x_i \|^2 \right].
\]

Now, with the choice of \( \beta_t = \beta := 1 - \frac{1}{\sqrt{t+1}} \in [0,1) \), we can easily show that \( \omega_t = \beta^{2t}, \omega_{i,t} = \beta^{2(t-i)}, \) and \( s_t = (1-\beta)^2 \frac{1-\beta^{2t}}{1-\beta^2} < \frac{1-\beta}{1+\beta} \) due to Lemma 4.1.

Let \( u_t^2 := \mathbb{E}[\| \hat{x}_{t+1} - x_t \|^2] \). To bound the quantity \( T_m \) defined by (74), we note that

\[
\sum_{t=1}^{m} \gamma_t \sum_{i=0}^{t-1} \beta^{2(t-i)} \gamma_i^2 u_i^2 = \beta^2 \gamma_0^2 [\gamma_1 + \beta^2 \gamma_2 + \cdots + \beta^{2(m-1)} \gamma_{m-1}] u_0^2 \\
+ \beta^2 \gamma_1^2 [\gamma_2 + \beta^2 \gamma_3 + \cdots + \beta^{2(m-2)} \gamma_{m-2}] u_1^2 + \cdots \\
+ \beta^2 \gamma_{m-2}^2 [\gamma_{m-1} + \beta^2 \gamma_m] u_{m-2}^2 + \beta^2 \gamma_{m-1}^2 u_{m-1}^2.
\]

Using this expression and \( \delta := \frac{2}{\eta} - 3 \), we can write \( T_m \) from (74) as

\[
T_m = \gamma_0 \left[ L^2 \beta^2 \gamma_0 [\gamma_1 + \beta^2 \gamma_2 + \cdots + \beta^{2(m-1)} \gamma_{m-1}] - (\delta - L\gamma_0) \right] u_0^2 \\
+ \gamma_1 \left[ L^2 \beta^2 \gamma_1 [\gamma_2 + \beta^2 \gamma_3 + \cdots + \beta^{2(m-2)} \gamma_{m-2}] - (\delta - L\gamma_1) \right] + \cdots \\
+ \gamma_{m-1} \left[ L^2 \beta^2 \gamma_{m-1} \gamma_m - (1 - L\gamma_{m-1}) \right] u_{m-1}^2 - \gamma_m (\delta - L\gamma_m) u_m^2.
\]

To guarantee \( T_m \leq 0 \), from the last expression of \( T_m \), we can impose the following condition:

\[
\begin{cases}
L^2 \beta^2 \gamma_0 [\gamma_1 + \beta^2 \gamma_2 + \cdots + \beta^{2(m-1)} \gamma_{m-1}] - (\delta - L\gamma_0) = 0 \\
L^2 \beta^2 \gamma_1 [\gamma_2 + \beta^2 \gamma_3 + \cdots + \beta^{2(m-2)} \gamma_{m-2}] - (\delta - L\gamma_1) = 0 \\
\vdots \\
L^2 \beta^2 \gamma_{m-1} \gamma_m - (\delta - L\gamma_{m-1}) = 0 \\
-(1 - L\gamma_m) = 0.
\end{cases}
\]

(75)
(a) It is obvious to show that the condition (75) leads to the following update of \( \gamma_t \):

\[
\gamma_m := \frac{\delta}{L} \quad \text{and} \quad \gamma_t := \frac{\delta}{L + L^2 [\beta^2 \gamma_{t+1} + \beta^4 \gamma_{t+2} + \ldots + \beta^{2(m-1)} \gamma_m]}, \quad t = 0, \ldots, m-1,
\]

which is exactly (30).

Next, note that since \( \beta \in [0, 1) \), we have \( 1 - \beta^2 \geq 1 - \beta = \frac{1}{\sqrt{b(m+1)}} \) which implies \( \sqrt{1 - \beta^2} \geq \frac{1}{\sqrt{b(m+1)}} \). Using \( \sqrt{1 - \omega} = \sqrt{1 - \beta^2} \geq \frac{1}{\sqrt{b(m+1)}} \), \( \epsilon = 1 \), and \( \sqrt{1 - \omega + \sqrt{1 - \omega + 4\delta \omega}} \leq 2\sqrt{1 + 2\beta} \) into (51) of Lemma A.1, we can show that \( \Sigma_m := \sum_{t=0}^{m} \gamma_t \geq \frac{\delta (m+1)^{3/4}}{2Lb^{3/4}\sqrt{1+2\beta}} \) as in the first statement (a) of Theorem 4.2.

(b) Since \( \omega_t = \beta^{t+1} \), by Chebyshev’s sum inequality, we have

\[
\sum_{t=0}^{m} \omega_t \gamma_t = \sum_{t=0}^{m} \beta^{2t} \gamma_t \leq \frac{\Sigma_m}{(m+1)} (1 + \beta^2 + \cdots + \beta^{2m}) \leq \frac{\Sigma_m}{(m+1)(1 - \beta^2)}.
\]

Utilizing this estimate, \( \sigma^2 := \mathbb{E} [\|v_0 - \nabla f(x_0)\|^2] \leq \frac{\sigma^2}{2} \), and \( S_t \leq \sigma^2 s_t \leq \frac{(1 - \beta)\sigma^2}{1 + \beta} \) into (73), and noting that \( T_m \leq 0 \), we can further upper bound it as

\[
\frac{\eta^2}{2} \sum_{t=0}^{m} \gamma_t \mathbb{E} [\|G_t(x_t)\|^2] \leq \frac{2}{\eta^2 \Sigma_m} [F(x_0) - F^\ast] + \frac{(1 + 2\eta^2)\sigma^2 \Sigma_m}{2(1 - \beta^2)b(m+1)} + \frac{(1 + 2\eta^2)(1 - \beta)\sigma^2 \Sigma_m}{2(1 + \beta)}.
\]

By Assumption 2.1, we have \( \mathbb{E} [F(x_{m+1})] \geq F^\ast \). Using this inequality into the last estimate and then multiplying the result by \( \frac{2}{\eta \Sigma_m} \) we obtain

\[
\frac{1}{\Sigma_m} \sum_{t=0}^{m} \gamma_t \mathbb{E} [\|G_t(x_t)\|^2] \leq 2 \mathbb{E} [\|G_t(x_t)\|^2] + \frac{(1 + 2\eta^2)\sigma^2 \Sigma_m}{2(1 - \beta^2)b(m+1)} + \frac{(1 + 2\eta^2)(1 - \beta)\sigma^2 \Sigma_m}{2(1 + \beta)}.
\]

Since \( \frac{1}{b(m+1)(1 - \beta)} + (1 - \beta) = \frac{2}{b^{3/2}(m+1)^{3/2}} \) with the choice \( \beta = 1 - \frac{1}{b^{3/2}(m+1)^{3/2}} \) and \( \Sigma_m \geq \frac{\delta (m+1)^{3/4}}{2Lb^{3/4}\sqrt{1+2\beta}} \), we can again upper bound (76) as

\[
\frac{1}{\Sigma_m} \sum_{t=0}^{m} \gamma_t \mathbb{E} [\|G_t(x_t)\|^2] \leq \frac{4Lb^{3/4}\sqrt{1 + 2\beta}}{\delta \eta^2(m+1)^{3/2}} [F(x_0) - F^\ast] + \frac{2(1 + 2\eta^2)\sigma^2}{\eta^2 b^{3/2}(m+1)^{1/2}}.
\]

However, due to the choice of \( \mathcal{F}_m \sim \mathcal{U}_p (\{x_i\}_{i=0}^{m}) \), we have \( \mathbb{E} [\|G_t(\mathcal{F}_m)\|^2] = \frac{1}{\Sigma_m} \sum_{t=0}^{m} \gamma_t \mathbb{E} [\|G_t(x_t)\|^2] \), the last estimate implies (31).

(c) Let us choose \( b := c_1 \sigma^{3/2} (m + 1)^{1/3} \) for some constant \( c_1 \geq \frac{1}{\sigma^{3/2}(m+1)^{3/2}} \). Then, \( \beta = 1 - \frac{1}{b^{3/2}(m+1)^{3/2}} \in (0, 1) \). Moreover, (31) becomes

\[
\mathbb{E} [\|G_t(\mathcal{F}_m)\|^2] \leq \frac{2\sigma^{2/3}}{(m+1)^{2/3}} \left[ \frac{2L\sqrt{c_1(1 + 2\beta)}}{\delta \eta^2} [F(x_0) - F^\ast] + \frac{(1 + 2\eta^2)}{c_1 \eta^2} \right].
\]

Let \( \Delta_0 := \frac{4L\sqrt{c_1(1 + 2\beta)}}{\delta \eta^2} [F(x_0) - F^\ast] + \frac{2(1 + 2\eta^2)}{c_1 \eta^2} \) be a constant. For any \( \varepsilon > 0 \), to guarantee \( \mathbb{E} [\|G_t(\mathcal{F}_m)\|^2] \leq \varepsilon^2 \), from (77) we need to set

\[
\frac{\Delta_0 \sigma^{2/3}}{(m+1)^{2/3}} \leq \varepsilon^2.
\]

This leads to \( m + 1 \geq \frac{\Delta_0^{3/2} \sigma}{\varepsilon^2} \). Therefore, we can choose \( m := \left\lfloor \frac{\Delta_0^{3/2} \sigma}{\varepsilon^2} \right\rfloor \) as shown in (32). This is also the total number of proximal operations \( \text{prox}_{\eta \Phi} \).
Finally, we estimate the total number $T_m$ of stochastic gradient evaluations $\nabla f_k(x_t)$ as

$$T_m = \tilde{b} + 3(m + 1) = c_1^2 \sigma S^{5/3}(m + 1)^{1/3} + \frac{3 \Delta_0^{3/2} \sigma}{\varepsilon^3}$$

$$= \frac{c_1^2 \Delta_0^{3/2} \sigma^3}{\varepsilon} + \frac{3 \Delta_0^{3/2} \sigma}{\varepsilon^3} = O \left( \frac{\sigma}{\varepsilon^3} + \frac{\sigma^3}{\varepsilon} \right),$$

which proves (33).

2.5 The proof of Theorem 4.3: Double-loop variant

Since we use the adaptive variant of Algorithm 1 as stated in Theorem 4.2 for the inner loop of Algorithm 2 from [76], we can see that at each stage $s$, the following estimate holds

$$\frac{1}{S \Sigma_m} \sum_{t=0}^{m} \gamma_t \mathbb{E} \left[ \|G_{\eta}(x^{(s)}_t)\|^2 \right] \leq \frac{2}{\eta^2 \Sigma_m} \mathbb{E} \left[ F(x^{(s)}_0) - F(x^{(s)}_{m+1}) \right] + \frac{2(1 + 2\eta^2)\sigma^2}{\eta^2 b^{1/2}(m + 1)^{1/2}} ,$$

(78)

where we use the superscript “$^{(s)}$” to indicate the stage $s$ in Algorithm 2. Summing up this inequality from $s = 1$ to $s = S$, and then multiplying the result by $\frac{1}{S}$ and using $\mathbb{E} \left[ F(x^{(S)}_{m+1}) \right] \geq F^* > -\infty$, and $\mathbb{E} \left[ \|G_{\eta}(\pi_T)\|^2 \right] = \frac{1}{S \Sigma_m} \sum_{s=1}^{S} \sum_{t=0}^{m} \mathbb{E} \left[ \|G_{\eta}(x^{(s)}_t)\|^2 \right]$, we get

$$\mathbb{E} \left[ \|G_{\eta}(\pi_T)\|^2 \right] = \frac{1}{S \Sigma_m} \sum_{s=1}^{S} \sum_{t=0}^{m} \mathbb{E} \left[ \|G_{\eta}(x^{(s)}_t)\|^2 \right]$$

$$\leq \frac{2}{\eta^2 S \Sigma_m} \left[ F(\pi^{(0)}) - F^* \right] + \frac{2(1 + 2\eta^2)\sigma^2}{\eta^2 b^{1/2}(m + 1)^{1/2}}$$

(79)

$$\leq \frac{4L \tilde{b}^{1/4} \sqrt{1 + 2\delta} \Delta F_0}{\delta \eta^2 S (m + 1)^{3/4}} + \frac{2(1 + 2\eta^2)\sigma^2}{\eta^2 b^{1/2}(m + 1)^{1/2}},$$

Here, we use the fact that $\Sigma_m \geq \frac{2^{5(m + 1)^{3/4}}}{2L \tilde{b}^{1/4} \sqrt{1 + 2\delta}}$ from Theorem 4.2(a) in the last inequality.

For any $\varepsilon > 0$, if we choose $\tilde{b} := \frac{c_1^2 \sigma^2}{\eta^2}$ and $m + 1 := \frac{c_1^2 \sigma^2}{\eta^2}$, then, from (79), to guarantee $\mathbb{E} \left[ \|G_{\eta}(\pi_T)\|^2 \right] \leq \varepsilon^2$, we require

$$\frac{4L \tilde{b}^{1/4} \sqrt{1 + 2\delta} \Delta F_0}{\delta \eta^2 S (m + 1)^{3/4}} + \frac{2(1 + 2\eta^2)\sigma^2}{\eta^2 b^{1/2}(m + 1)^{1/2}} \leq \frac{4L c_1^{1/4} \sqrt{1 + 2\delta} \Delta F_0 \cdot \sigma}{\delta \eta^2 c_2^{3/4} \cdot \varepsilon}$$

$$\leq \frac{4L c_1^{1/4} \sqrt{1 + 2\delta} \Delta F_0}{\delta \eta^2 c_2^{3/4} \left( 1 - \frac{2(1 + 2\eta^2)}{\eta^2 \sqrt{c_1 c_2}} \right) \cdot \sigma \varepsilon},$$

Hence, we use $\Delta F_0 := F(\pi^{(0)}) - F^*$. Consequently, the total number of stochastic gradient evaluations $\nabla f_k(x_t)$ is at most

$$T_T := (b + 3(m + 1)) S = (c_1 + 3c_2) \frac{\sigma^2}{\varepsilon^2} \cdot \frac{4L c_1^{1/4} \sqrt{1 + 2\delta} \left[ F(\pi^{(0)}) - F^* \right]}{\delta \eta^2 c_2^{3/4} \sigma \left( 1 - \frac{2(1 + 2\eta^2)}{\eta^2 \sqrt{c_1 c_2}} \right) \varepsilon}$$

$$= \frac{4L(c_1 + 3c_2) c_1^{1/4} \sqrt{1 + 2\delta} \left[ F(\pi^{(0)}) - F^* \right]}{\delta \eta^2 c_2^{3/4} \left( 1 - \frac{2(1 + 2\eta^2)}{\eta^2 \sqrt{c_1 c_2}} \right) \cdot \sigma \varepsilon} = O \left( \frac{\sigma}{\varepsilon^3} \right).$$
Since we choose \( \tilde{b} := \varepsilon_2 \sigma^2 \), the final complexity is \( O \left( \max \left\{ \sigma, \varepsilon_1 \right\} \right) \), where other constants independent of \( \sigma \) and \( \varepsilon \) are hidden. The total number of proximal operators \( \text{prox}_{\rho g} \) is at most

\[
T_{\text{prox}} := S(m+1) = \frac{4Lc_1^{1/4} \sqrt{1 + 28 \left( F(x^{(0)}) - F^* \right) \varepsilon^2}}{\delta \eta^2 c_2^{1/4} \left( 1 - \frac{2(1+2\gamma^2)}{\eta \sqrt{\varepsilon_1}} \right) \cdot \sigma \varepsilon^2} = \frac{4L(c_1c_2)^{1/4} \sqrt{1 + 28 \left( F(x^{(0)}) - F^* \right) \varepsilon^2}}{\delta \eta^2 \left( 1 - \frac{2(1+2\gamma^2)}{\eta \sqrt{\varepsilon_1}} \right)} \varepsilon^3
\]

The proof is completed. \( \square \)

C The proof of technical results in Section 5

This appendix presents the full proof of the results in Section 5 for the mini-batch case.

3.1 The proof of Theorem 5.1: The single-loop variant

Using (20) from Lemma 3.4 with \( G := \nabla f \) and taking full expectation and using a constant weight \( \beta_t := \beta \in (0, 1) \) and \( b_t := b \in \mathbb{N}_+ \), we have

\[
\mathbb{E} \left[ ||\hat{v}_{t+1} - \nabla f(x_{t+1})||^2 \right] \leq \beta^2 \mathbb{E} \left[ ||\hat{v}_t - \nabla f(x_t)||^2 \right] + \rho \beta^2 \mathbb{E} \left[ ||\nabla f(x_{t+1}) - \nabla f(x_t)||^2 \right] + (1 - \beta)^2 \mathbb{E} \left[ ||u_{t+1} - \nabla f(x_{t+1})||^2 \right],
\]

where \( \rho := \frac{1}{b} \) since we solve (1).

However, we have \( \mathbb{E} \left[ ||\nabla f(x_{t+1}) - \nabla f(x_t)||^2 \right] \leq L^2 \mathbb{E} \left[ ||x_{t+1} - x_t||^2 \right] \leq L^2 \gamma_t^2 \mathbb{E} \left[ ||\tilde{x}_{t+1} - x_t||^2 \right] \) by Assumption 2.2 and \( \mathbb{E} \left[ ||u_{t+1} - \nabla f(x_{t+1})||^2 \right] \leq \frac{\sigma^2}{b} \) by Assumption 2.3 and [53] Lemma 2, the last estimate leads to

\[
\mathbb{E} \left[ ||\hat{v}_{t+1} - \nabla f(x_{t+1})||^2 \right] \leq \beta^2 \mathbb{E} \left[ ||\hat{v}_t - \nabla f(x_t)||^2 \right] + \frac{\beta^2 \gamma_t^2 L^2}{b} \mathbb{E} \left[ ||\tilde{x}_{t+1} - x_t||^2 \right] + \frac{(1 - \beta)^2 \sigma^2}{b}. \tag{80}
\]

Next, let us choose \( \eta_t = \eta > 0 \), \( c_t = r_t = 1 \), and \( q_t = \gamma_t = \gamma > 0 \) in Lemma 4.1. Then, we have \( \theta_t = \theta = (1 + 2\eta^2) \gamma > 0 \) and \( \kappa_t = \kappa = \left( \frac{2}{\eta} - 3 - L\gamma \right) \gamma > 0 \). To guarantee \( \kappa > 0 \), we need to choose \( \eta < \frac{1}{2} \). Using these values into (22), we obtain the following estimate:

\[
\mathbb{E} \left[ F(x_{t+1}) \right] \leq \mathbb{E} \left[ F(x_t) \right] - \frac{\gamma \eta^2}{2} \mathbb{E} \left[ ||G_t(x_t)||^2 \right] + \frac{\theta}{2} \mathbb{E} \left[ ||\nabla f(x_t) - \hat{v}_t||^2 \right] - \frac{\kappa}{2} \mathbb{E} \left[ ||\tilde{x}_{t+1} - x_t||^2 \right]. \tag{81}
\]

Multiplying (80) by \( \frac{\alpha}{2} \) for some \( \alpha > 0 \), and adding the result to (81), we obtain

\[
\mathbb{E} \left[ F(x_{t+1}) \right] + \frac{\alpha}{2} \mathbb{E} \left[ ||\hat{v}_{t+1} - \nabla f(x_{t+1})||^2 \right] \leq \mathbb{E} \left[ F(x_t) \right] + \frac{(\alpha \beta^2 + \theta)}{2} \mathbb{E} \left[ ||\hat{v}_t - \nabla f(x_t)||^2 \right] - \frac{\gamma \eta^2}{2} \mathbb{E} \left[ ||G_t(x_t)||^2 \right] + \frac{\alpha(1 - \beta)^2 \sigma^2}{2b} - \frac{1}{2} \left( \kappa - 2 \beta^2 \gamma^2 L^2 \right) \mathbb{E} \left[ ||x_t - x_{t-1}||^2 \right].
\]

Using the Lyapunov function \( V \) defined by (24), the last estimate leads to

\[
V(x_{t+1}) \leq V(x_t) - \frac{\gamma \eta^2}{2} \mathbb{E} \left[ ||G_t(x_t)||^2 \right] + \frac{\alpha(1 - \beta)^2 \sigma^2}{2b} - \frac{1}{2} \left( \kappa - 2 \beta^2 \gamma^2 L^2 \right) \mathbb{E} \left[ ||x_t - x_{t-1}||^2 \right] + \frac{\theta - \alpha(1 - \beta^2)}{2} \mathbb{E} \left[ ||\hat{v}_t - \nabla f(x_t)||^2 \right].
\]

If we impose the following conditions

\[
\kappa = \left( \frac{2}{\eta} - 3 - L\gamma \right) \gamma \geq \frac{\alpha \beta^2 \gamma^2 L^2}{b} \quad \text{and} \quad \theta = (1 + 2\eta^2) \gamma \leq \alpha(1 - \beta^2), \tag{82}
\]
then we obtain
\[ V(x_{t+1}) \leq V(x_t) - \frac{\gamma \eta^2}{2} \mathbb{E} \left[ \| G(x_t) \|^2 \right] + \frac{(1 - \beta)^2 \sigma^2}{2b}. \]  

(83)

The conditions (82) can be simplified as
\[ \frac{2}{\eta} - 3 - L\gamma \geq \frac{\alpha \gamma \beta^2 L^2}{b} \quad \text{and} \quad (1 + 2\eta^2)\gamma \leq \alpha (1 - \beta^2). \]  

(84)

Moreover, by induction and \( V(x_{m+1}) \geq F^* \), we can further derive from (83) that
\[ \frac{1}{m+1} \sum_{t=0}^{m} \mathbb{E} \left[ \| G(x_t) \|^2 \right] \leq \frac{2}{\eta^2 \gamma (m+1)} [V(x_0) - F^*] + \frac{(1 - \beta)^2 \sigma^2}{\eta^2 \gamma b}. \]

Now, since \( x_m \sim U \{ x_t \}_{t=0}^m \) and \( V(x_0) := F(x_0) + \frac{\alpha \sigma^2}{2} \mathbb{E} \left[ \| \tilde{v}_0 - \nabla f(x_0) \|^2 \right] \leq F(x_0) + \frac{\alpha \sigma^2}{\eta^2} \), the last estimate implies
\[ \mathbb{E} \left[ \| G_{\omega}(x_m) \|^2 \right] \leq \frac{2}{\eta^2 \gamma (m+1)} [F(x_0) - F^*] + \frac{(1 - \beta)^2 \sigma^2}{\eta^2 \gamma b}. \]

(85)

To minimize the right-hand side of (85), we choose \( \beta := 1 - \frac{\hat{b}^{1/2}}{b^{1/2}(m+1)^{1/2}} \). Clearly, if \( 1 \leq \hat{b} \leq b(m+1) \), then \( \beta \in [0, 1] \).

Note that since \( \beta \in [0, 1] \), we have \( 1 - \beta^2 \geq 1 - \beta = \frac{\hat{b}^{1/2}}{b^{1/2}(m+1)^{1/2}} \). On the other hand, since \( \eta \in (0, \frac{2}{3}) \), we also have \( 1 + 2\eta^2 \leq \frac{17}{9} \). Therefore, the second condition of (84) holds if we choose \( 0 < \gamma \leq \frac{9\alpha \hat{b}^{1/2}}{17b^{1/2}(m+1)^{1/2}} \). Alternatively, if we choose \( \eta := \frac{4}{2+L} \), then \( \frac{\hat{b}}{\eta} = 3 - L\gamma = 1 \). The first condition of (84) holds if we choose \( 0 < \gamma \leq \frac{b}{2L\alpha} \). Combining all the conditions on \( \gamma \), we get
\[ 0 < \gamma \leq \min \left\{ 1, \frac{b}{L^2 \alpha}, \frac{9\alpha \hat{b}^{1/2}}{17b^{1/2}(m+1)^{1/2}} \right\}. \]

If we choose \( \alpha := \frac{\sqrt{17} \hat{b}^{1/4} \hat{b}^{1/2}(m+1)^{1/4}}{3L b^{1/4}} \), then \( \frac{b^{1/2}}{L^2 \alpha} = \frac{9\alpha \hat{b}^{1/2}}{17b^{1/2}(m+1)^{1/2}} \). Now, let us update \( \gamma \) as
\[ \gamma := \frac{3c_1 \hat{b}^{1/4} \hat{b}^{1/2}}{\sqrt{17} L \hat{b}^{1/4}(m+1)^{1/4}}. \]

Since \( 1 \leq \hat{b} \leq b(m+1) \), we have \( \gamma \leq \frac{3c_1 \hat{b}^{1/4}}{\sqrt{17} L} \). If we choose \( 0 < c_1 \leq \frac{\sqrt{17} L}{3b^{1/4}} \), then \( \gamma \in (0, 1] \). Consequently, we obtain (85).

The choice of \( \alpha \) and \( \gamma \) also implies that
\[ \frac{\alpha}{\gamma} = \frac{17b^{1/2}(m+1)^{1/2}}{9\hat{b}^{1/2}} \quad \text{and} \quad \frac{1}{\gamma} = \frac{\sqrt{17} L b^{1/4}(m+1)^{1/4}}{3c_1 \hat{b}^{1/4} b^{1/2}}. \]

Using these expressions into (85), we finally get
\[ \mathbb{E} \left[ \| G_{\omega}(x_m) \|^2 \right] \leq \frac{2\sqrt{17} L \hat{b}^{1/4}}{3c_1 \eta^2 b^{1/4} \hat{b}^{1/2}(m+1)^{1/4}} [F(x_0) - F^*] + \frac{34\alpha^2}{9\gamma^2 b^{1/2} b^{1/2}(m+1)^{1/2}}, \]

which proves (85).

Finally, let us choose \( b = \hat{b} \in N_+ \) and \( \hat{b} := c_2 \sigma^{8/3} [b(m+1)]^{1/3} \) for some \( c_2 > 0 \). Then (85) reduces to
\[ \mathbb{E} \left[ \| G_{\omega}(x_m) \|^2 \right] \leq \left[ \frac{2\sqrt{17} c_2 L}{3c_1 \eta^2} [F(x_0) - F^*] + \frac{34}{9\gamma^2} \right] \frac{\sigma^{2/3}}{[b(m+1)]^{2/3}}. \]
For any $\varepsilon > 0$, to guarantee $\mathbb{E} \left[ \|G_{\eta}(\overline{T}_m)\|^2 \right] \leq \varepsilon^2$, we need to impose $\Delta_0^{2/3} = \varepsilon^2$, where $\Delta_0 := \frac{2\mathcal{N}_{\eta}(1)}{b(m+1) \sigma^2} \|F(x_0) - F^*\| + \frac{3\eta^2}{b^2}$. This implies $b(m+1) = \frac{\Delta_0^{2/3}}{\varepsilon^2}$, which also leads to $m + 1 = \frac{\Delta_0^{2/3}}{\varepsilon^2}$. Therefore, the maximum number of iterations is at most $m := \left\lfloor \frac{\Delta_0^{2/3}}{\varepsilon^2} \right\rfloor$. This is also the number of proximal operations $\text{prox}_{\eta \hat{\rho}}$. The number of stochastic gradient evaluations $\nabla f(x_t)$ is at most $\overline{T}_m := \tilde{b} + 3(m+1)b = \frac{\alpha_0^{1/2} \alpha^3}{\varepsilon} + \frac{\alpha_0^{1/2} \alpha^3}{\varepsilon^2} = O \left( \frac{\alpha^3}{\varepsilon} + \frac{\alpha}{\varepsilon^2} \right)$.

### 3.2 The proof of Theorem 5.2: The mini-batch double-loop variant

Similar to the proof of Theorem 4.2, summing up (22) from $t = 0$ to $t = m$ and using $\rho = \frac{1}{b}$ and $\hat{\rho} = \frac{1}{b}$ from Lemma 3.5 we obtain

\begin{equation}
\mathbb{E} \left[ F(x_{m+1}) \right] \leq \mathbb{E} \left[ F(x_0) \right] + \frac{L^2}{2b} \sum_{i=0}^{m} \theta_i \sum_{t=0}^{t-1} \gamma_t^2 \omega_t \mathbb{E} \left[ \|x_{i+1}^t - x_t^s\|^2 \right] - \frac{1}{2} \sum_{t=0}^{m} \theta_t \mathbb{E} \left[ \|G_{\eta}(x_t^s)\|^2 \right] + \frac{1}{2b} \sum_{t=0}^{m} \theta_t S_t,
\end{equation}

where $\gamma_t$, $\eta_t$, $\theta_t$, $\omega_t$, $\chi_t$, and $S_t$ are defined in Lemmas 4.1 and 4.1.

Let us fix $\epsilon_t = r_1 = 1$, $q_t = \gamma_t \in (0,1)$, and $\beta_t = \beta \in [0,1]$. Then $\theta_t = (1 + 2\gamma^2)\gamma_t$ and $\gamma_t = \gamma_t \left( \frac{2}{\eta} - 3 - L\gamma_t \right)$. Moreover, $\omega_t = \beta^3 t$, $\omega_t = \beta^2 (t-1)$, and $s_t = (1 - \beta^2) \left( \frac{1 - \beta^2}{1 + \beta^2} \right) < \frac{1}{1 + \beta^2}$ due to Lemma 4.1 and $\mathbb{E} \left[ \|\tilde{v}_0^s - \nabla f(x_0^s)\|^2 \right] \leq \frac{\sigma^2}{b}$.

Using this configuration and noting that $x^{(s)} = x_{m+1}^s$ and $x^{(s-1)} = x_0^s$, following the same argument as (76), (86) reduces to

\begin{equation}
\mathbb{E} \left[ F(x^{(s)}) \right] \leq \mathbb{E} \left[ F(x^{(s-1)}) \right] - \frac{\eta^2}{2} \sum_{t=0}^{m} \gamma_t \mathbb{E} \left[ \|G_{\eta}(x_t^s)\|^2 \right] + \left( \frac{(1 + 2\gamma^2)\sigma^2}{2b(m+1)(1 - \beta^2)} + \frac{(1 + 2\gamma^2)(1 - \beta)\sigma^2}{2b(1 + \beta)} \right) \Sigma_m + \frac{T_m}{2},
\end{equation}

where $T_m$ is defined as follows:

\begin{equation}
\tilde{T}_m := \frac{L^2 + 2\sigma^2}{b} \sum_{t=0}^{m} \gamma_t \sum_{i=0}^{t-1} \beta^{2(t-i)} \gamma_i^2 \mathbb{E} \left[ \|x_{i+1}^t - x_t^s\|^2 \right] - \sum_{t=0}^{m} \gamma_t \left( \frac{2}{\eta} - 3 - L\gamma_t \right) \mathbb{E} \left[ \|x_{i+1}^t - x_t^s\|^2 \right].
\end{equation}

Similar to the proof of (30), if we choose $\eta \in (0, \frac{2}{\beta})$, set $\delta := \frac{2}{\beta} - 3 > 0$, and update $\gamma$ as in (40):

\begin{equation}
\gamma_m := \frac{\delta}{L} \quad \text{and} \quad \gamma_t := bL + \frac{b\delta}{L} \beta^2 \gamma_{t+1}^s + \beta^4 \gamma_{t+2} + \cdots + \beta^2 (m-t) \gamma_m,
\end{equation}

then $T_m \leq 0$. Moreover, (87) can be simplified as

\begin{equation}
\mathbb{E} \left[ F(x^{(s)}) \right] \leq \mathbb{E} \left[ F(x^{(s-1)}) \right] - \frac{\eta^2}{2} \sum_{t=0}^{m} \gamma_t \mathbb{E} \left[ \|G_{\eta}(x_t^s)\|^2 \right] + \frac{17\sigma^2}{18} \left[ \frac{1}{b(m+1)(1 - \beta)} + \frac{(1 - \beta)}{b} \right] \Sigma_m.
\end{equation}
Summing up this inequality from $s = 1$ to $s = S$ and noting that $F(\pi(S)) \geq F^*$, we obtain
\[
\frac{1}{S\Sigma_m} \sum_{s=1}^{S} \sum_{t=0}^{m} \gamma_t \mathbb{E} \left[ \|G_t(x_t^{(s)})\|^2 \right] \leq \frac{2}{\eta^2 S \Sigma_m} \left[ F(\pi(0)) - F^* \right] + \frac{17\sigma^2}{9\eta^2 S} \frac{1}{b(m+1)(1-\beta)} + \frac{(1-\beta)}{b} \tag{89}
\]

Let us first choose $\beta := 1 - \frac{\delta^{1/2}}{b(1/2m+1/2)}$. Then, from the update rule (40) of $\gamma_t$, we apply Lemma [A.1] with $\omega = \beta^2$ and $\epsilon = \frac{\beta}{b}$ to obtain
\[
\Sigma_m \geq \frac{\delta(m+1)\sqrt{1-\beta^2}}{L \left[ \sqrt{1-\beta^2 + \sqrt{1-\beta^2 + \frac{6\delta^2}{b}}} \right]} \frac{\delta(m+1)\theta^{1/4}b^{1/2}}{L \sqrt{8\delta b^{1/2}(m+1)^{1/2} + 8bb^{1/2}}}
\]

Using this bound into (89) and noting that $\tau_T \sim U_p \left( \{x_t^{(s)}\}_{t=0}^{S} \right)$, we can upper bound it as
\[
\mathbb{E} \left[ \|G_\pi(\tau_T)\|^2 \right] \leq \frac{2L}{\delta \eta^2 S(m+1)b^{1/4}b^{1/2}} \left[ F(\pi(0)) - F^* \right] + \frac{34\sigma^2}{9\eta^2 Sb^{1/2}b^{1/2}(m+1)^{1/2}} \tag{90}
\]
which is (41).

Now, let us choose $b = \hat{b} \in \mathbb{N}_+$ and assume that $b^{3/2} = \hat{b}b^{1/2} \leq \delta b^{1/2}(m+1)^{1/2}$, which leads to $b \leq \delta^2/\beta^2 b^{1/2}(m+1)^{1/2} = \delta^2/\beta^2 c_1 b^{1/2}(m+1)^{1/2}(m+1)^{1/2}$. In this case, the right-hand side of (90) can be upper bounded as
\[
\mathcal{R}_T := \frac{2L}{\delta \eta^2 S(m+1)b^{1/4}b^{1/2}} \left[ F(\pi(0)) - F^* \right] + \frac{34\sigma^2}{9\eta^2 Sb^{1/2}b^{1/2}(m+1)^{1/2}}
\]

Let us choose $\hat{b} := c_1^2 \sigma^3 (b(m+1))^{1/3}$ for some $c_1 > 0$. Then, we can bound $\mathcal{R}_T$ as
\[
\mathcal{R}_T \leq \frac{8L\sigma^3 \sqrt{c_1}}{\delta \eta^2 S(m+1)} \left[ F(\pi(0)) - F^* \right] + \frac{34}{9c_1 \sigma^2}
\]

Let $\Delta_0 := \frac{8L\sigma^3 \sqrt{c_1}}{\delta \eta^2 S(m+1)} \left[ F(\pi(0)) - F^* \right] + \frac{34 c_1 \sigma^2}{9 \sigma^2}$. Then, for any $\varepsilon > 0$, to achieve $\mathbb{E} \left[ \|G_\pi(\tau_T)\|^2 \right] \leq \varepsilon^2$, we impose $\frac{\Delta_0 \sigma^2/\varepsilon}{\delta \eta^2 S(m+1)} = \varepsilon^2$, which implies that the total number of iterations
\[
T := (m+1)S = \frac{\Delta_0 \sigma^2/\varepsilon}{b e^3} = O \left( \frac{\sigma}{b e^3} \right).
\]

This is also the total number of proximal operations $\text{prox}_{\eta b}$. The total number of stochastic gradient evaluations $\nabla f_i(x_t)$ is at most $T_T := \hat{b} \hat{b}^2 + \hat{b} \hat{b}^2 (m+1) S = c_1^2 \sigma^3 S [b(m+1)]^{1/3} + \frac{3\Delta_0^2 \sigma^2}{\varepsilon^2} = \frac{c_1^2 \sigma^3 \Delta_0^{1/2}}{\varepsilon} \cdot S^{2/3} + \frac{3\Delta_0^2 \sigma^2}{\varepsilon^2}$ for any $S \geq 1$.

\section{D Proof of technical results in Section 6}
We give the full proof of the results in Section 6 for solving the finite-sum problem (2).

\subsection{4.1 The proof of Theorem 6.1: Bounded domain}
(a) If we apply Algorithm 1 to solve (2) with $b = n$, then $\mathbb{E} \left[ \|x_0 - \nabla f(x_0)\|^2 \right] = 0$. Moreover, under Assumption 2.4 and [16], we have $\mathbb{E} \left[ \|w_{tv}\|^2 \right] \leq L^2 \|x_t - x_0\|^2 \leq L^2 D_X^2$. Consequently, we can bound $S_t \leq s_t L^2 D_X^2 \leq \frac{(1-\beta)}{1+\beta} L^2 D_X^2$ in (73) and then use the fact that $\mathcal{T}_m \leq 0$ from the proof of Theorem 4.2 to obtain
\[
\eta_t^2 \sum_{t=0}^{m} \gamma_t \mathbb{E} \left[ \|G_t(x_t)\|^2 \right] \leq F(x_0) - \mathbb{E} \left[ F(x_{m+1}) \right] + \frac{(1+2\eta_t^2)(1-\beta)}{2(1+\beta)} \cdot \Sigma_m L^2 D_X^2.
\]
By Lemma \ref{lem:lyapunov}, we have $\Sigma_m \geq \frac{\delta(m+1)\sqrt{1-\beta^2}}{2Lm} \geq \frac{\delta(m+1)\sqrt{1-\beta^2}}{2L(m+1)^{2/3}}$ as shown in the proof of Theorem \ref{thm:main}. In addition, $\mathbb{E} [F(x_{m+1})] \geq F^*$ from Assumption \ref{assump:3}. Substituting these estimates into the last inequality, and multiplying the result by $\frac{1}{\eta^2\Sigma_m}$, we obtain

$$1 \sum_{i=0}^{m} \gamma_i \mathbb{E} \left[ \|G_n(x_i)\|^2 \right] \leq \frac{2}{\eta^2 \Sigma_m} \left[ F(x_0) - F^* \right] + \frac{(1+2\eta^2)}{\eta^2(1+\beta)} (1 - \beta) L^2 D_X^2.$$  

We can minimize the right-hand size of this estimate over $\beta \in [0,1]$ to obtain $\beta := 1 - \frac{c_1}{(m+1)^{2/3}}$ for some $0 < c_1 \leq (m+1)^{2/3}$. Using this choice of $\beta$ and $1 + 2 \eta^2 \leq \frac{12}{7}$, the last estimate leads to

$$1 \sum_{i=0}^{m} \gamma_i \mathbb{E} \left[ \|G_n(x_i)\|^2 \right] \leq \frac{4L \sqrt{1 + 2\eta \beta m} + 17c_1 L^2 D_X^2}{\delta \eta^2 \sqrt{c_1 (m+1)^{2/3}}} + \frac{17c_1 L^2 D_X^2}{9 \eta^2 (m+1)^{2/3}}.$$  

Finally, due to the choice of $\tau_m \sim \mathcal{U}_p \{x_i \}_{i=0}^{m}$, we obtain \eqref{eq:approx_1} from the last estimate.

(b) If $D_X^2 \leq O(1)$, then, by defining $\Delta_0 := \frac{4L \sqrt{1 + 2\eta \beta m} + 17c_1 L^2 D_X^2}{\delta \eta^2 \sqrt{c_1 (m+1)^{2/3}}}$, for any $\epsilon > 0$, to guarantee $\mathbb{E} \left[ \|G_n(x_m)\|^2 \right] \leq \epsilon^2$, we need to impose $\frac{\Delta_0}{\delta \eta^2 (m+1)^{2/3}} = \epsilon^2$, which leads to $m+1 = \frac{\sqrt{\Delta}}{\epsilon^2}$. Therefore, we can choose the number of iterations $m$ as

$$m := \left\lceil \frac{\sqrt{\Delta}}{\epsilon^2} \right\rceil = O \left( \frac{1}{\epsilon^2} \right).$$

This is the same as the total number of proximal operations $\text{prox}_{\eta^2 \nu}$. The total number of stochastic gradient evaluations $\nabla f_i(x_i)$ is at most $\mathcal{T}_m = n + 4m = n + \left\lceil \frac{4\Delta^{1/2}}{\epsilon^2} \right\rceil = O \left( n + \frac{1}{\epsilon^2} \right).$ □

### 4.2 The proof of Theorem \ref{thm:main} Loopless SARAH-SVRG variant

We first prove a key estimate in Lemma \ref{lem:lyapunov}. Then, we prove Theorem \ref{thm:main}.

**Lemma D.1.** Assume that Assumptions \ref{assump:2} and \ref{assump:3} hold for \eqref{eq:objective}. Let $\{x_i\}_{i=0}^{m}$ be the sequence generated by Algorithm \ref{alg:main}. Let us define the following Lyapunov function:

$$\hat{V}(x_i) := \mathbb{E} [F(x_i)] + \frac{\alpha_i}{2} \mathbb{E} \left[ \|v_i - \nabla f(x_i)\|^2 \right] + \frac{\lambda_i}{2} \mathbb{E} \left[ \|x_i - \bar{x}_i\|^2 \right],$$

where $\alpha_i$ and $\lambda_i$ are two nonnegative parameters. Let $\beta_t$, $\theta_t$, and $\kappa_t$ be defined in Lemma \ref{lem:lyapunov} and $\nu_t > 0$ and $p_0 \in (0,1)$ be two constants such that

$$\begin{cases} 
\alpha_t \geq \alpha_{t+1} \beta_t^2 + \theta_t, \\
\lambda_t \geq (1 + \nu_t)(1 - p_0) \left[ \alpha_{t+1}(1 - \beta_t) L^2 + \lambda_{t+1} \right], \\
\kappa_t \geq \alpha_{t+1} \beta_t^2 \gamma_t^2 L^2 + (1 + \frac{1}{\eta_t})(1 - p_0) \gamma_t^2 \left[ \alpha_{t+1}(1 - \beta_t) L^2 + \lambda_{t+1} \right].
\end{cases}$$

Then, we have

$$\hat{V}(x_{t+1}) \leq \hat{V}(x_t) - \frac{4\sigma_t^2}{2} \mathbb{E} \left[ \|G_n(x_t)\|^2 \right].$$

**Proof.** From \eqref{eq:lyapunov}, by taking full expectation on the history $\mathcal{F}_{t+1}$, using the $L$-average smoothness of $f$ from Assumption \ref{assump:3}, and $(x_{t+1} - x_t = \gamma_t \tilde{x}_{t+1} - x_t)$, we can show that

$$\mathbb{E} \left[ \|v_{t+1} - \nabla f(x_{t+1})\|^2 \right] \leq \beta_t^2 \mathbb{E} \left[ \|v_t - \nabla f(x_t)\|^2 \right] + \beta_t^2 \alpha_{t+1} \lambda_{t+1} + \lambda_{t+1},$$

where $\hat{V}(x_{t+1}) := \mathbb{E} \left[ \|v_{t+1} - \nabla f(x_{t+1})\|^2 \right]$. Moreover, by \eqref{eq:lyapunov}, we have

$$\hat{V}(x_{t+1}) \leq \hat{V}(x_t) - \frac{4\sigma_t^2}{2} \mathbb{E} \left[ \|G_n(x_t)\|^2 \right].$$

44
Combining these estimates we get
\[
\mathbb{E} \left[ \|v_{t+1} - \nabla f(x_{t+1})\|^2 \right] \leq \beta_t^2 \mathbb{E} \left[ \|v_t - \nabla f(x_t)\|^2 \right] + \beta_t^2 \gamma_t^2 L^2 \mathbb{E} \left[ \|\tilde{x}_{t+1} - x_t\|^2 \right] + (1 - \beta_t)^2 L^2 \mathbb{E} \left[ \|x_{t+1} - \tilde{x}_{t+1}\|^2 \right].
\] (94)

On the other hand, from Step 9 of Algorithm 3, for any \(i_t > 0\), we have
\[
\mathbb{E} \left[ \|x_{t+1} - \tilde{x}_{t+1}\|^2 \right] = (1 - p_0) \mathbb{E} \left[ \|x_{t+1} - \tilde{x}_t\|^2 \right] + p_0 \mathbb{E} \left[ \|x_{t+1} - x_{t+1}\|^2 \right]
\leq (1 - p_0)(1 + i_t) \mathbb{E} \left[ \|x_t - \tilde{x}_t\|^2 \right] + (1 - p_0) \left( 1 + \frac{\alpha}{\gamma} \right) \gamma_t^2 \mathbb{E} \left[ \|\tilde{x}_{t+1} - x_t\|^2 \right].
\]

Combining this estimate and (91), we can derive that
\[
D_{t+1} := \alpha_{t+1} \mathbb{E} \left[ \|v_{t+1} - \nabla f(x_{t+1})\|^2 \right] + \lambda_{t+1} \mathbb{E} \left[ \|x_{t+1} - \tilde{x}_{t+1}\|^2 \right]
\leq \alpha_{t+1} \beta_t^2 \mathbb{E} \left[ \|v_t - \nabla f(x_t)\|^2 \right] + (1 - p_0)(1 + i_t) \left[ \alpha_{t+1}(1 - \beta_t)^2 L^2 + \lambda_{t+1} \right] \mathbb{E} \left[ \|x_t - \tilde{x}_t\|^2 \right]
\]
\[
+ \left[ \alpha_{t+1} \beta_t^2 \gamma_t^2 L^2 + \left( 1 + \frac{\alpha}{\gamma} \right) \gamma_t^2 \left[ \alpha_{t+1}(1 - \beta_t)^2 L^2 + \lambda_{t+1} \right] \right] \mathbb{E} \left[ \|\tilde{x}_{t+1} - x_t\|^2 \right].
\] (95)

Next, from (22), we have
\[
\mathbb{E} \left[ F(x_{t+1}) \right] \leq \mathbb{E} \left[ F(x_t) \right] - \frac{\theta_t \eta_t^2}{2} \mathbb{E} \left[ \|G_{\eta_t}(x_t)\|^2 \right] + \frac{\theta_t}{2} \mathbb{E} \left[ \|\nabla f(x_t) - v_t\|^2 \right] + \frac{\kappa_t}{2} \mathbb{E} \left[ \|\tilde{x}_{t+1} - x_t\|^2 \right].
\] (96)

Combining (95) and (96), and then using the Lyapunov function \(\hat{V}\) from (91), we can show that
\[
\hat{V}(x_{t+1}) \leq \hat{V}(x_t) - \frac{c_1 \eta_t^2}{2} \mathbb{E} \left[ \|G_{\eta_t}(x_t)\|^2 \right]
\]
\[
+ \frac{1}{2} \left[ \alpha_{t+1} \beta_t^2 + \theta_t - \alpha \right] \mathbb{E} \left[ \|v_t - \nabla f(x_t)\|^2 \right]
\]
\[
+ \frac{1}{2} \left[ \left( 1 + i_t \right) \left( 1 - p_0 \right) \left[ \alpha_{t+1}(1 - \beta_t)^2 L^2 + \lambda_{t+1} \right] - \lambda_t \right] \mathbb{E} \left[ \|x_t - \tilde{x}_t\|^2 \right]
\]
\[
- \frac{1}{2} \left[ \kappa_t - \alpha_{t+1} \beta_t^2 \gamma_t^2 L^2 - \left( 1 + \frac{\alpha}{\gamma} \right) \gamma_t^2 \left[ \alpha_{t+1}(1 - \beta_t)^2 L^2 + \lambda_{t+1} \right] \right] \mathbb{E} \left[ \|\tilde{x}_{t+1} - x_t\|^2 \right].
\]

Clearly, if (92) holds, then the last estimate implies (93).

**The proof of Theorem 6.2.** Let us first fix \(c_i := 1\), \(r_i := 1\), \(q_i := \gamma_t = \gamma \in (0, 1]\) and \(i_t := p_0 \in (0, 1]\). We also fix \(\beta_i := \beta \in [0, 1]\), \(\eta_i := \eta > 0\), \(\alpha_i := \alpha > 0\), and \(\lambda_i := \lambda > 0\). In this case, we have
\[
\theta_t = \theta = \left( 1 + 2\eta^2 \right) \gamma \quad \text{and} \quad \kappa_t = \kappa = \gamma \left( \frac{2}{n} - L\gamma - 3 \right).
\]

To guarantee \(\kappa > 0\), we need to choose \(\eta \in (0, \frac{2}{n})\).

Next, we choose \(\beta := 1 - \frac{c_1}{n}\) and \(\alpha := c\sqrt{n}\) for \(0 \leq c_1 \leq n\) and \(c > 0\). Since \(\beta \in [0, 1]\), we have \(1 - \beta^2 \geq 1 - \beta = \frac{c^2}{n}\). In addition, since \(\eta \in (0, \frac{2}{n})\), we also have \(1 + 2\eta^2 \leq \frac{9n}{4}\). Therefore, the first condition of (92) holds if we impose
\[
\alpha(1 - \beta^2) \geq \frac{c_1 c}{\sqrt{n}} \geq \frac{17\gamma}{9} \geq (1 + 2\eta^2) \gamma = \theta.
\]

This condition shows that if we choose
\[
0 < \gamma \leq \frac{9c_1 c}{17\sqrt{n}}.
\] (97)

then first condition of (92) holds.

Next, since \(i_t := p_0 \in (0, 1]\), the second condition of (92) becomes
\[
p_0^2 \lambda \geq \alpha(1 - p_0)(1 - \beta)^2 L^2 = \frac{c_1^2 c(1 - p_0^2) L^2}{n^{eta/2}}.
\]
If we choose \( \lambda := \frac{c_2 t L^2}{p_0^2 n^3} \), then this condition holds.

Similarly, the third condition of (92) becomes
\[
\frac{2}{\eta} - L \gamma - 3 \geq \alpha \beta^2 L^2 + \gamma \left[ \lambda + \frac{c_2 t L^2}{n^3/2} \right] = \alpha \lambda^{1/2} \beta^2 L^2 \gamma + \frac{c_2 t L^2 \gamma (1 + p_0^2)}{p_0^2 n^{3/2}}.
\]
Assume that \( \eta := \frac{2}{4 + t \alpha \gamma} \). Then, we have \( \frac{2}{\eta} - L \gamma - 3 = 1 \). Therefore, we can choose \( \gamma \) and \( p_0 \) such that
\[
\left[ n^{1/2} + \frac{2c_1^2}{p_0^2 n^{3/2}} \right] \hat{c} L^2 \gamma \leq 1.
\]
This condition leads to
\[
\gamma \leq \frac{p_0^2 n^{3/2}}{\hat{c} L^2 (2c_1^2 + c_2^2) \sqrt{n}}.
\]
Let us choose \( p_0 := \frac{1}{\eta \gamma} \) for given \( 0 < c_2 < n^{2/3} \). Then, the last condition leads to
\[
0 < \gamma \leq \frac{c_2^3}{\hat{c} L^2 (2c_1^2 + c_2^2) \sqrt{n}}.
\]
From (97) and (98), let us choose \( \hat{c} > 0 \) such that \( \frac{c_2^3}{\hat{c} L^2 (2c_1^2 + c_2^2)} = \frac{9 \gamma \hat{c}}{17} \). This shows that \( \hat{c} = \frac{\sqrt{17} \gamma^{2/3}}{3L \sqrt{c_1^2 (2c_1^2 + c_2^2)^{1/2}}} \). Therefore, we can choose
\[
\gamma := \frac{3c_1^{1/2} c_2^{3/2}}{\sqrt{17} \lambda L^2 (2c_1^2 + c_2^2)^{1/2} \sqrt{n}}.
\]
Clearly, since \( m \geq 0 \), if \( 9c_1 c_2^2 \leq 17 L^2 (2c_1^2 + c_2^2) \), then \( \gamma \in (0, 1] \).

With the above choices of parameters, we obtain the update in (45). Moreover, we also obtain from (93) that \( \frac{\gamma \eta^2}{2} \mathbb{E} ||G_\eta(x_i)||^2 \leq \hat{V}(x_i) - \hat{V}(x_{i+1}) \). By induction, we have
\[
\frac{\gamma \eta^2}{2} \sum_{t=0}^m \mathbb{E} ||G_\eta(x_t)||^2 \leq \hat{V}(x_0) - \hat{V}(x_m+1) \leq F(x_0) - F^* + \frac{\alpha}{2} \mathbb{E} \left[ ||v_0 - \nabla f(x_0)||^2 \right] + \frac{\lambda}{2} \mathbb{E} \left[ ||x_0 - \hat{x}_0||^2 \right].
\]
The last inequality holds since \( \hat{V}(x_{m+1}) \geq F^* \). However, since \( v_0 = \nabla f(x_0) \) and \( x_0 = \hat{x}_0 \) due to Algorithm 3, we can further simplify the last estimate as
\[
\frac{1}{m + 1} \sum_{t=0}^m \mathbb{E} ||G_\eta(x_t)||^2 \leq \frac{2}{\eta \gamma (m + 1)} \left[ F(x_0) - F^* \right].
\]
Using \( \gamma \) from (99), this inequality leads to
\[
\mathbb{E} \left[ ||G_\eta(\hat{x}_m)||^2 \right] = \frac{1}{m + 1} \sum_{t=0}^m \mathbb{E} ||G_\eta(x_t)||^2 \leq \frac{2 \sqrt{17} L (2c_1^2 + c_2^2)^{1/2} \sqrt{n}}{3 \eta^2 c_1^{1/2} c_2^{3/2} (m + 1)} \left[ F(x_0) - F^* \right].
\]
This leads to (46).

Finally, for any \( \epsilon > 0 \), to achieve \( \mathbb{E} \left[ ||G_\eta(\hat{x}_m)||^2 \right] \leq \epsilon^2 \), we need \( m \) iterations as
\[
m + 1 := \left[ \frac{\Delta_0 \sqrt{n}}{\epsilon^2} \right] = \mathcal{O} \left( \frac{\sqrt{n}}{\epsilon^2} \right), \quad \text{where} \quad \Delta_0 := \frac{2 \sqrt{17} L (2c_1^2 + c_2^2)^{1/2}}{3 \eta^2 c_1^{1/2} c_2^{3/2}} \left[ F(x_0) - F^* \right].
\]
The overall average number of stochastic gradient evaluations \( \nabla f_i(x_1) \) is at most
\[
T_m := (2 + p_0 n + 2(1 - p_0))(m + 1) = 4(m + 1) + p_0 n - 2(m + 1)
\]
\[
\leq 4 \Delta_0 \sqrt{n} + 2 \eta n^{3/6} \Delta_0 = \mathcal{O} \left( \frac{\sqrt{n}}{\epsilon^2} + \frac{n^{3/6}}{\epsilon^2} \right).
\]
Therefore, we can take \( T_m := 4 \Delta_0 \sqrt{n} + 2 \eta n^{3/6} \Delta_0 = \mathcal{O} \left( \frac{\sqrt{n}}{\epsilon^2} + \frac{n^{3/6}}{\epsilon^2} \right) \) as its upper bound.
References

1. A. Agarwal, P. L. Bartlett, P. Ravikumar, and M. J. Wainwright. Information-theoretic lower bounds on the oracle complexity of stochastic convex optimization. IEEE Transactions on Information Theory, 99:1–1, 2010.

2. A. Agarwal and L. Bottou. A lower bound for the optimization of finite sums. In International Conference on Machine Learning, pages 78–86, 2015.

3. Z. Allen-Zhu. Katyusha: The first direct acceleration of stochastic gradient methods. Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing (STOC), pages 1200–1205, June 2017. Montreal, Canada.

4. Z. Allen-Zhu. Natasha: Faster non-convex stochastic optimization via strongly non-convex parameter. In Proceedings of the 34th International Conference on Machine Learning, volume 70, pages 89–97. JMLR. org, 2017.

5. Z. Allen-Zhu. Natasha 2: Faster non-convex optimization than SGD. In Advances in neural information processing systems, pages 2675–2686, 2018.

6. Z. Allen-Zhu and Y. Li. NEON2: Finding local minima via first-order oracles. In Advances in Neural Information Processing Systems, pages 3720–3730, 2018.

7. Zeyuan Allen-Zhu and Yang Yuan. Improved SVRG for Non-Strongly-Convex or Sum-of-Non-Convex Objectives. In ICML, pages 1080–1089, 2016.

8. R. Bollapragada, R. Byrd, and J. Nocedal. Exact and Inexact Subsampled Newton Methods for Optimization. IMA Journal of Numerical Analysis, Online First:1–37, 2016.

9. L. Bottou. Online learning and stochastic approximations. In David Saad, editor, Online Learning in Neural Networks, pages 9–42. Cambridge University Press, New York, NY, USA, 1998.

10. L. Bottou. Large-scale machine learning with stochastic gradient descent. In Proceedings of COMPSTAT’2010, pages 177–186. Springer, 2010.

11. Richard H Byrd, SL Hansen, Jorge Nocedal, and Yoram Singer. A stochastic quasi-Newton method for large-scale optimization. SIAM J. Optim., 26(2):1008–1031, 2016.

12. Y. Carmon, J. Duchi, O. Hinder, and A. Sidford. Lower bounds for finding stationary points i. Math. Program., Online First:1–50, 2017.

13. A. Chambolle, M. J. Ehrhardt, P. Richtárik, and C.-B. Schönlieb. Stochastic primal-dual hybrid gradient algorithm with arbitrary sampling and imaging applications. SIAM J. Optim., 28(4):2783–2808, 2018.

14. C.-C. Chang and C.-J. Lin. LIBSVM: A library for Support Vector Machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1–27:27, 2011.

15. A. Cutkosky and F. Orabona. Momentum-based variance reduction in non-convex SGD. arxiv:1905.10018, 2019.

16. D. Davis and B. Grimmer. Proximally guided stochastic subgradient method for nonsmooth, nonconvex problems. arXiv preprint arXiv:1707.03505, 2017.

17. A. Defazio, F. Bach, and S. Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In Advances in Neural Information Processing Systems (NIPS), pages 1646–1654, 2014.
18. A. Defazio, T. Caetano, and J. Domke. Finito: A faster, permutable incremental gradient method for big data problems. In International Conference on Machine Learning, pages 1125–1133, 2014.

19. D. Driggs, J. Liang, and C.-B. Schönlieb. On the bias-variance tradeoff in stochastic gradient methods. arXiv preprint arXiv:1906.01133, 2019.

20. John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. J. Mach. Learn. Res., 12:2121–2159, 2011.

21. Murat A Erdogdu and Andrea Montanari. Convergence rates of sub-sampled Newton methods. In Advances in Neural Information Processing Systems, pages 3052–3060, 2015.

22. C. Fang, C. J. Li, Z. Lin, and T. Zhang. SPIDER: Near-optimal non-convex optimization via stochastic path integrated differential estimator. In Advances in Neural Information Processing Systems, pages 689–699, 2018.

23. C. Fang, Z. Lin, and T. Zhang. Sharp analysis for nonconvex SGD escaping from saddle points. arXiv preprint arXiv:1902.00247, 2019.

24. D. Foster, A. Sekhari, O. Shamir, N. Srebro, K. Sridharan, and B. Woodworth. The complexity of making the gradient small in stochastic convex optimization. arXiv preprint arXiv:1902.04686, 2019.

25. R. Ge, F. Huang, C. Jin, and Y. Yuan. Escaping from saddle points - online stochastic gradient for tensor decomposition. In Conference on Learning Theory, pages 797–842, 2015.

26. R. Ge, Z. Li, W. Wang, and X. Wang. Stabilized SVRG: Simple variance reduction for nonconvex optimization. arXiv preprint arXiv:1905.00529, 2019.

27. S. Ghadimi and G. Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. SIAM J. Optim., 23(4):2341–2368, 2013.

28. S. Ghadimi and G. Lan. Accelerated gradient methods for nonconvex nonlinear and stochastic programming. Math. Program., 156(1-2):59–99, 2016.

29. S. Ghadimi, G. Lan, and H. Zhang. Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. Math. Program., 155(1-2):267–305, 2016.

30. I. Goodfellow, Y. Bengio, and A. Courville. Deep learning, volume 1. MIT press Cambridge, 2016.

31. A. Jofré and P. Thompson. On variance reduction for stochastic smooth convex optimization with multiplicative noise. Math. Program., 174(1-2):253–292, 2019.

32. R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In Advances in Neural Information Processing Systems (NIPS), pages 315–323, 2013.

33. H. Karimi, J. Nutini, and M. Schmidt. Linear convergence of gradient and proximal-gradient methods under the Polyak-Łojasiewicz condition. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pages 795–811. Springer, 2016.

34. D. P. Kingma and J. Ba. ADAM: A Method for Stochastic Optimization. Proceedings of the 3rd International Conference on Learning Representations (ICLR), abs/1412.6980, 2014.

35. Jakub Konečný, Jie Liu, Peter Richtárik, and Martin Takáč. Mini-batch semi-stochastic gradient descent in the proximal setting. IEEE Journal of Selected Topics in Signal Processing, 10:242–255, 2016.
36. D. Kovalev, S. Horvath, and P. Richtarik. Don’t jump through hoops and remove those loops: SVRG and Katyusha are better without the outer loop. arXiv preprint arXiv:1901.08689, 2019.

37. Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. Nature, 521(7553):436–444, 2015.

38. Z. Li. SSRGD: Simple stochastic recursive gradient descent for escaping saddle points. arXiv preprint arXiv:1904.09265, 2019.

39. Z. Li and J. Li. A simple proximal stochastic gradient method for nonsmooth nonconvex optimization. In Advances in Neural Information Processing Systems, pages 5564–5574, 2018.

40. L. Lihua, C. Ju, J. Chen, and M. Jordan. Non-convex finite-sum optimization via SCSG methods. In Advances in Neural Information Processing Systems, pages 2348–2358, 2017.

41. H. Lin, J. Mairal, and Z. Harchaoui. A universal catalyst for first-order optimization. In Advances in Neural Information Processing Systems, pages 3384–3392, 2015.

42. J. Mairal. Incremental majorization-minimization optimization with application to large-scale machine learning. SIAM J. Optim., 25(2):829–855, 2015.

43. M. Metel and A. Takeda. Simple stochastic gradient methods for non-smooth non-convex regularized optimization. In International Conference on Machine Learning, pages 4537–4545, 2019.

44. A. Mokhtari, A. Ozdaglar, and A. Jadbabaie. Escaping saddle points in constrained optimization. In Advances in Neural Information Processing Systems, pages 3629–3639, 2018.

45. Eric Moulines and Francis R Bach. Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In Advances in Neural Information Processing Systems, pages 451–459, 2011.

46. A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. Robust stochastic approximation approach to stochastic programming. SIAM J. Optim., 19(4):1574–1609, 2009.

47. A. Nemirovskii and D. Yudin. Problem Complexity and Method Efficiency in Optimization. Wiley Interscience, 1983.

48. Y. Nesterov and B.T. Polyak. Cubic regularization of Newton method and its global performance. Math. Program., 108(1):177–205, 2006.

49. Yuriii Nesterov. Introductory lectures on convex optimization: A basic course. Applied optimization. Kluwer Academic Publ., Boston, Dordrecht, London, 2004.

50. L. M. Nguyen, J. Liu, K. Scheinberg, and M. Takáč. SARAH: A novel method for machine learning problems using stochastic recursive gradient. In The 34th International Conference on Machine Learning (ICML), 2017.

51. L. M. Nguyen, N. H. Nguyen, D. T. Phan, J. R. Kalagnanam, and K. Scheinberg. When does stochastic gradient algorithm work well? arXiv:1801.06159, 2018.

52. L. M. Nguyen, K. Scheinberg, and M. Takac. Inexact SARAH Algorithm for Stochastic Optimization. arXiv preprint arXiv:1811.10105, 2018.

53. L. M. Nguyen, M. van Dijk, D. T. Phan, P. H. Nguyen, T.-W. Weng, and J. R. Kalagnanam. Optimal finite-sum smooth non-convex optimization with SARAH. arXiv preprint arXiv:1901.07648, 2019.
54. Lam M. Nguyen, Jie Liu, Katya Scheinberg, and Martin Takác. Stochastic recursive gradient algorithm for nonconvex optimization. CoRR, abs/1705.07261, 2017.

55. E. Oja. Simplified neuron model as a principal component analyzer. Journal of mathematical biology, 15(3):267–273, 1982.

56. H. N. Pham, M. L. Nguyen, T. D. Phan, and Q. Tran-Dinh. ProxSARAH: An efficient algorithmic framework for stochastic composite nonconvex optimization. Preprint: arXiv:1902.05679, 2019.

57. M. Pilanci and M.J. Wainwright. Newton sketch: A linear-time optimization algorithm with linear-quadratic convergence. SIAM J. Optim., 27(1):205–245, 2017.

58. Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. SIAM Journal on Control and Optimization, 30(4):838–855, 1992.

59. S. J. Reddi, S. Sra, B. Póczos, and A. Smola. Stochastic Frank-Wolfe methods for nonconvex optimization. In 2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton), pages 1244–1251. IEEE, 2016.

60. S. J. Reddi, S. Sra, B. Póczos, and A. J. Smola. Proximal stochastic methods for nonsmooth nonconvex finite-sum optimization. In Advances in Neural Information Processing Systems, pages 1145–1153, 2016.

61. Sashank J. Reddi, Ahmed Hefny, Suvrit Sra, Barnabás Póczos, and Alexander J. Smola. Stochastic variance reduction for nonconvex optimization. In ICML, pages 314–323, 2016.

62. Herbert Robbins and Sutton Monro. A stochastic approximation method. The Annals of Mathematical Statistics, 22(3):400–407, 1951.

63. F. Roosta-Khorasani and M. W. Mahoney. Sub-sampled Newton methods I: Globally convergent algorithms. Arxiv preprint:1601.04737, 2016.

64. M. Schmidt, N. Le Roux, and F. Bach. Minimizing finite sums with the stochastic average gradient. Math. Program., 162(1-2):83–112, 2017.

65. Shai Shalev-Shwartz and Tong Zhang. Stochastic dual coordinate ascent methods for regularized loss. Journal of Machine Learning Research, 14(1):567–599, 2013.

66. Q. Tran-Dinh, H. N. Pham, T. D. Phan, and M. L. Nguyen. Hybrid stochastic gradient descent algorithms for stochastic nonconvex optimization. Preprint: arXiv:1905.05920, 2019.

67. M. Wang, E. Fang, and L. Liu. Stochastic compositional gradient descent: algorithms for minimizing compositions of expected-value functions. Math. Program., 161(1-2):419–449, 2017.

68. Z. Wang, K. Ji, Y. Zhou, Y. Liang, and V. Tarokh. SpiderBoost: A class of faster variance-reduced algorithms for nonconvex optimization. arXiv preprint arXiv:1810.10690, 2018.

69. Z. Wang, Y. Zhou, Y. Liang, and G. Lan. Stochastic variance-reduced cubic regularization for nonconvex optimization. arXiv preprint arXiv:1802.07372, 2018.

70. B. E. Woodworth and N. Srebro. Tight complexity bounds for optimizing composite objectives. In Advances in neural information processing systems (NIPS), pages 3639–3647, 2016.

71. J. Zhang, L. Xiao, and S. Zhang. Adaptive stochastic variance reduction for subsampled Newton method with cubic regularization. Preprint: arXiv:1811.11637, 2018.
72. L. Zhao, M. Mammadov, and J. Yearwood. From convex to nonconvex: a loss function analysis for binary classification. In IEEE International Conference on Data Mining Workshops (ICDMW), pages 1281–1288. IEEE, 2010.

73. P. Zhao and T. Zhang. Stochastic optimization with importance sampling for regularized loss minimization. In International Conference on Machine Learning, pages 1–9, 2015.

74. D. Zhou and Q. Gu. Lower bounds for smooth nonconvex finite-sum optimization. International Conference on Machine Learning (ICML), 2019.

75. D. Zhou and Q. Gu. Stochastic recursive variance-reduced cubic regularization methods. Preprint: arXiv:1901.11518, 2019.

76. D. Zhou, P. Xu, and Q. Gu. Stochastic nested variance reduction for nonconvex optimization. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, pages 3925–3936. Curran Associates Inc., 2018.

77. K. Zhou, F. Shang, and J. Cheng. A simple stochastic variance reduced algorithm with fast convergence rates. In International Conference on Machine Learning, pages 5975–5984, 2018.

78. Y. Zhou, Z. Wang, K. Ji, Y. Liang, and V. Tarokh. Momentum schemes with stochastic variance reduction for nonconvex composite optimization. arXiv preprint arXiv:1902.02715, 2019.