OPT2020: 12th Annual Workshop on Optimization for Machine Learning

Adaptivity of Stochastic Gradient Methods for Nonconvex Optimization

Sameul Horváth[*] SAMUEL.HORVATH@KAUST.EDU.SA
KAUST, Thuwal, KSA

Lihua Lei[*] LIHUALEI@STANFORD.EDU
Stanford University, Stanford, USA

Peter Richtárik PETER.RICHTARIK@KAUST.EDU.SA
KAUST, Thuwal, KSA

Michael I. Jordan JORDAN@STAT.BERKELEY.EDU
University of California, Berkeley, USA

Abstract

Adaptivity is an important yet under-studied property in modern optimization theory. The gap between the state-of-the-art theory and the current practice is striking in that algorithms with desirable theoretical guarantees typically involve drastically different settings of hyperparameters, such as step-size schemes and batch sizes, in different regimes. Despite the appealing theoretical results, such divisive strategies provide little, if any, insight to practitioners to select algorithms that work broadly without tweaking the hyperparameters. In this work, blending the "geometrization" technique introduced by [27] and the SARAH algorithm of [39], we propose the Geometrized SARAH algorithm for non-convex finite-sum and stochastic optimization. Our algorithm is proved to achieve adaptivity to both the magnitude of the target accuracy and the Polyak-Łojasiewicz (PL) constant, if present. In addition, it achieves the best-available convergence rate for non-PL objectives simultaneously while outperforming existing algorithms for PL objectives.

1. Introduction

We study smooth nonconvex problems of the form

$$\min_{x\in\mathbb{R}^d} \left\{ f(x) \overset{\text{def}}{=} \mathbb{E} f_\xi(x) \right\} , \tag{1}$$

where the randomness comes from the selection of data points and is represented by the index ξ. If the number of indices n is finite, then we talk about *empirical risk minimization* and $\mathbb{E} f_\xi(x)$ can be written in the finite-sum form, $(1/n)\sum_{i=1}^n f_i(x)$. If n is not finite or if it is infeasible to process the entire dataset, we are in the *online learning* setting, where one obtains independent samples of ξ at each step. We assume that an optimal solution $x^\star$ of (1) exists and its value is finite: $f(x^\star) > -\infty$.

The many faces of stochastic gradient descent. We start with a brief review of relevant aspects of gradient-based optimization algorithms. Since the number of functions n can be large or even infinite, algorithms that process subsamples are essential. The canonical example is Stochastic Gradient Descent (SGD) [16, 35, 36], in which updates are based on single data points or small batches of points. The terrain around the basic SGD method has been thoroughly explored in recent years, resulting in theoretical and practical enhancements such as Nesterov acceleration [3], Polyak momentum [41, 53], adaptive step sizes [10, 21, 33, 48], distributed optimization [2, 32, 52], importance sampling [44, 60], higher-order optimization [24, 55], and several other useful techniques.

[*] Equal contribution.

Table 1: Complexity to reach an $\mathbb{E}\|\nabla f(x)\|^2 \le \epsilon^2$ with $L, \sigma^2, \Delta_f = O(1)$.

Method	Complexity	Required knowledge
SVRG (non-cvx) [47]	$\mathcal{O}\left(n + \frac{n^{2/3}}{\epsilon^2}\right)$	L
SCSG (non-cvx) [29]	$\tilde{\mathcal{O}}\left(\frac{1}{\epsilon^{10/3}} \wedge \frac{n^{2/3}}{\epsilon^2}\right)$	L
SNVRG (non-cvx) [61]	$\tilde{\mathcal{O}}\left(\frac{1}{\epsilon^3} \wedge \frac{\sqrt{n}}{\epsilon^2}\right)$ $\tilde{\mathcal{O}}\left(n + \frac{\sqrt{n}}{\epsilon^2}\right)$	L, σ^2, ϵ L
SARAH (non-cvx) [12, 40, 57]	$\mathcal{O}\left(n + \frac{\sqrt{n}}{\epsilon^2}\right)$	L
Q-Geom-SARAH (Theorem 8)	$\tilde{\mathcal{O}}\left(\left\{n^{3/2} + \frac{\sqrt{n}}{\mu}\right\} \wedge \frac{1}{\epsilon^3} \wedge \frac{\sqrt{n}}{\epsilon^2}\right)$	L
E-Geom-SARAH (Theorem 9)	$\tilde{\mathcal{O}}\left(\left(\frac{1}{\mu\wedge\epsilon}\right)^{2(1+\delta)} \wedge \left\{n + \frac{\sqrt{n}}{\mu}\right\} \wedge \frac{1}{\epsilon^4} \wedge \frac{\sqrt{n}}{\epsilon^2}\right)$	L
Non-adaptive Geom-SARAH (Theorem 13)	$\mathcal{O}\left(\left\{\frac{1}{\epsilon^{4/3}(\mu\wedge\epsilon)^{2/3}} \wedge n\right\} + \frac{1}{\mu}\left\{\frac{1}{\epsilon^{4/3}(\mu\wedge\epsilon)^{2/3}} \wedge n\right\}^{1/2}\right)$	$L, \sigma^2, \epsilon, \mu$

A particularly productive approach to enhancing SGD has been to make use of *variance reduction*, in which the classical stochastic gradient direction is modified in various ways so as to drive the variance of the gradient estimator towards zero. This significantly improves the convergence rate and may also enhance the quality of the output solution. The first variance-reduction method was SAG [49], closely followed by many more, for instance, [7, 8, 12, 15, 17, 19, 19, 20, 22, 23, 25, 29, 39, 44, 45, 51, 57, 61].

The dilemma of parameter tuning. Formally, each iteration of vanilla and variance-reduced SGD methods can be written in the generic form $x^+ = x - \eta g$, where $x \in \mathbb{R}^d$ is the current iterate, $\eta > 0$ is a step size and $g \in \mathbb{R}^d$ is a stochastic estimator of the true gradient $\nabla f(x)$.

A major drawback of many such methods is their dependence on parameters that are unlikely to be known in a real-world machine-learning setting. For instance, they may require the knowledge of a uniform bound on the variance or second moment of the stochastic estimators of the gradient which is simply not available, and might not even hold in practice. Moreover, some algorithms perform well in either low precision or high precision regimes and in order to make them perform well in all regimes, they require knowledge of extra parameters, such as target accuracy, which may be difficult to tune. Another related issue is the lack of adaptivity of many SGD variants to different modelling regimes. For example, in order to obtain good theoretical and experimental behavior for non-convex f, one needs to run a custom variant of the algorithm if the function is known to satisfy some extra assumptions such as the Polyak-Łojasiewicz (PL) inequality. As a consequence, practitioners are often forced to spend valuable time and resources tuning various parameters and hyper-parameters of their methods, which poses serious issues in implementation and practical deployment.

The search for adaptive methods. The above considerations motivate us to impose some algorithm design restrictions so as to resolve the aforementioned issues. First of all, good algorithms should be *adaptive* in the sense that they should perform comparably to methods with tuned parameters without an a-priori knowledge of the optimal parameter settings. In particular, in the non-convex regime, we might wish to design an algorithm that does not invoke nor need any bound on the variance of the stochastic gradient, or any predefined target accuracy in its implementation. In addition, we should desire algorithms which perform well if the Polyak-Lojasiewicz PL constant (or strong convexity parameter) μ happens to be large and yet are able to converge even if $\mu = 0$; all automatically, without the need for the method to be altered by the practitioner.

There have been several works on this topic, originating from works studying asymptotic rate for SGD with stepsize $\mathcal{O}(t^{-\alpha})$ for $\alpha \in (1/2, 1)$ [42, 43, 50] up to the most recent paper [28] which focuses on convex optimization, e.g. [5, 6, 9, 13, 18, 26, 30, 34, 56, 58, 59].

Table 2: Complexity to reach $\mathbb{E} f(x) - f(x^\star) \leq \epsilon^2$ with $L, \sigma^2, \Delta_f = O(1)$.

Method	Complexity	Required Knowledge
SVRG (PL) [31]	$\tilde{\mathcal{O}}\left(n + \frac{n^{2/3}}{\mu}\right)$	L
SCSG (PL) [29]	$\tilde{\mathcal{O}}\left(\left(\frac{1}{\mu\epsilon^2} \wedge n\right) + \frac{1}{\mu}\left(\frac{1}{\mu\epsilon^2} \wedge n\right)^{2/3}\right)$ $\tilde{\mathcal{O}}\left(n + \frac{n^{2/3}}{\mu}\right)$	$L, \sigma^2, \epsilon, \mu$ L
SNVRG (PL)[61]	$\tilde{\mathcal{O}}\left(\left(\frac{1}{\mu\epsilon^2} \wedge n\right) + \frac{1}{\mu}\left(\frac{1}{\mu\epsilon^2} \wedge n\right)^{1/2}\right)$ $\tilde{\mathcal{O}}\left(n + \frac{\sqrt{n}}{\mu}\right)$	$L, \sigma^2, \epsilon, \mu$ L
SARAH (PL) [40, 57]	$\tilde{\mathcal{O}}\left(n + \frac{1}{\mu^2}\right)$	L
Q-Geom-SARAH (Theorem 8)	$\tilde{\mathcal{O}}\left(\left(\frac{1}{\mu^2 \wedge \mu\epsilon^2}\right)^{3(1+\delta)/2} \wedge \left\{n^{3/2} + \frac{\sqrt{n}}{\mu}\right\}\right)$	L
E-Geom-SARAH (Theorem 9)	$\tilde{\mathcal{O}}\left(\left(\frac{1}{\mu^2 \wedge \mu\epsilon^2}\right)^{1+\delta} \wedge \left\{n + \frac{\sqrt{n}}{\mu}\right\}\right)$	L

This line of research has shown that algorithms with better complexity can be designed in a finite-sum setting with some levels of adaptivity, generally using the previously mentioned technique–variance reduction. Unfortunately, while these algorithms show some signs of adaptivity, e.g., they do not require the knowledge of μ, they usually fail to adapt to more than one regimes at once: strongly-convex vs convex loss functions, non-convex vs gradient-dominated regime and low vs high precision. To the best of our knowledge, the only paper that tackles multiple such issues is the work of [28]. However, even this work does not provide full adaptivity as it focuses on the convex setting. We are not aware of any work which manages to provide a fully adaptive algorithm in the non-convex setting.

Contributions. In this work we present a new method—the *geometrized stochastic recursive gradient* (Geom-SARAH) algorithm—that exhibits adaptivity to the PL constant, target accuracy and to the variance of stochastic gradients. Geom-SARAH is a double-loop procedure similar to the SVRG or SARAH algorithms. Crucially, our algorithm does not require the computation of the full gradient in the outer loop as performed by other methods, but makes use of stochastic estimates of gradients in both the outer loop and the inner loop. In addition, by exploiting a randomization technique "geometrization" that allows certain terms to telescope across the outer loop and the inner loop, we obtain a significantly simpler analysis. As a byproduct, this allows us to obtain adaptivity, and our rates either match the known lower bounds [12] or achieve the same rates as existing state-of-the-art specialized methods, perhaps up to a logarithmic factor; see Table 1 and 2 for the comparison of two versions of Geom-SARAH with existing methods. On a side note, we develop a non-adaptive version of Geom-SARAH (the last row of Table 1) that strictly outperforms existing methods in PL settings. Interestingly, when $\epsilon \sim \mu$, our complexity even beats the best available rate for strongly convex functions [4]. We would like to point out that our notion of adaptivity is different from the one pursued by algorithms such as AdaGrad [10] or Adam [21, 48], where they focus on the geometry of the loss surface. In our case, we focus on adaptivity to different parameters and regimes.

2. Preliminaries

Basic notation and definitions. We use $\|\cdot\|$ to denote standard Euclidean norm, we write either $\min\{a, b\}$ (resp. $\max\{a, b\}$) or $a \wedge b$ (resp. $a \vee b$) to denote minimum and maximum, and we use standard big $\mathcal{O}$ notation to leave out constants[1]. We adopt the computational cost model of the IFO framework introduced by [1] in

[1] As implicitly assumed in all other works, we use $\mathcal{O}(\log x)$ and $\mathcal{O}(1/x)$ as abbreviations of $\mathcal{O}((\log x) \vee 1)$ and $\mathcal{O}((1/x) \vee 1)$. For instance, the term $\mathcal{O}(1/\epsilon)$ should be interpreted as $\mathcal{O}((1/\epsilon) \vee 1)$ and the term $\mathcal{O}(\log n)$ should be interpreted as $\mathcal{O}((\log n) \vee 1)$.

which upon query x, the IFO oracle samples i and out outputs the pair $(f_i(x), \nabla f_i(x))$. A single such query incurs a unit cost.

Assumption 1 *The stochastic gradient of f is L-Lipschitz in expectation. That is,*

$$\mathbb{E}\|\nabla f_\xi(x) - \nabla f_\xi(y)\|^2 \leq L^2 \|x - y\|^2, \quad \forall x, y \in \mathbb{R}^d. \tag{2}$$

Assumption 2 *The stochastic gradient of f has uniformly bounded variance. That is, there exists $\sigma^2 > 0$ such that*

$$\mathbb{E}\|\nabla f_\xi(x) - \nabla f(x)\|^2 \leq \sigma^2, \quad \forall x \in \mathbb{R}^d. \tag{3}$$

Assumption 3 *f satisfies the PL condition*[2] *with parameter $\mu \geq 0$. That is,*

$$\|\nabla f(x)\|^2 \geq 2\mu(f(x) - f(x^\star)), \quad \forall x \in \mathbb{R}^d, \textit{ where } x^\star = \arg\min f(x). \tag{4}$$

We denote $\Delta_f \overset{\text{def}}{=} f(\tilde{x}_0) - f(x^\star)$ to be functional distance to optimal solution. For non-convex objectives, our goal is to output an ϵ-approximate first-order stationary point.

Definition 1 *We say that $x \in \mathbb{R}^d$ is an ϵ-approximate first-order stationary point of* (1) *if $\|\nabla f(x)\|^2 \leq \epsilon^2$.*

For a gradient dominated function, the quantity of the interest is the functional distance from an optimum, characterized in the following definition.

Definition 2 *We say that $x \in \mathbb{R}^d$ is an ϵ-accurate solution of* (1) *if $f(x) - f(x^\star) \leq \epsilon^2$.*

Accuracy independence and almost universality. We review two fundamental definitions introduced by [28] that serve as a building block for desirable “parameter-free" optimization algorithms. We refer to the first property as ϵ-independence.

Definition 3 *An algorithm is **ϵ-independent** if it guarantees convergence at all accuracies $\epsilon > 0$.*

This is a crucial property as the desired target accuracy is usually not known a-priori. Moreover, an ϵ-independent algorithm can provide convergence to any precision without the need for a manual adjustment of the algorithm or its parameters. To illustrate this, we consider `Spider` [12] and `Spiderboost` [57] algorithms. Both of these enjoy the same complexity $\mathcal{O}(n + \sqrt{n}/\epsilon^2)$ for non-convex smooth functions, but the stepsize for `Spider` is ϵ-dependent, making it impractical as this value is often hard to tune.

The second property is inspired by the notion of *universality* [37], requiring for an algorithm to not rely on any a-priori knowledge of smoothness or any other parameter such as the bound on variance.

Definition 4 *An algorithm is **almost universal** if it only requires the knowledge of the smoothness L.*

There are several algorithms that satisfy both properties for smooth non-convex optimization, including `SAGA`, `SVRG` [47], `Spiderboost` [57], `SARAH` [39], and `SARAH-SGD` [54]. Unfortunately, these algorithms are not able to provide a good result in both low and high precision regimes, and in order to perform well, they require the knowledge of extra parameters. *This is not the case for our algorithm which is both almost universal and ϵ-independent.* Moreover, our method is adaptive to the PL constant μ, and to low and high precision regimes.

[2] Functions satisfying this condition are sometimes also called *gradient dominated*.

Geometric distribution. Finally, we introduce an important technical tool behind the design of our algorithm, the *geometric distribution*, denoted by $N \sim \mathrm{Geom}(\gamma)$. Recall that $\mathrm{Prob}(N = k) = \gamma^k(1 - \gamma), \forall k = 1, 2, \ldots$, where an elementary calculation shows that $\mathrm{E}_{\mathrm{Geom}(\gamma)}[N] = {}^{\gamma}/_{1-\gamma}$.

We use the geometric distribution mainly due to its following property, which helps us to significantly simplify the analysis of our algorithm.

Lemma 5 *Let $N \sim \mathrm{Geom}(\gamma)$. Then for any sequence $D_0, D_1, \ldots$ with $\mathbb{E}|D_N| < \infty$,*

$$\mathbb{E}D_N - D_{N+1} = ({}^{1}/_{\mathbb{E}N})(D_0 - \mathbb{E}D_N). \tag{5}$$

Remark 6 *The requirement $\mathbb{E}|D_N| < \infty$ is essential. A useful sufficient condition is $|D_k| = O(\mathrm{Poly}(k))$ because a geometric random variable has finite moments of any order.*

3. Algorithm

Algorithm 1 Geom-SARAH

```
Input: stepsizes $\{\eta_j\}$, big-batch sizes $\{B_j\}$, expected inner-loop queries $\{m_j\}$, mini-batch sizes $\{b_j\}$,
initializer $\tilde{x}_0$, tail-randomized fraction $\delta$
for $j = 1, \ldots \lceil (1+\delta)T \rceil$ do
  $x_0^{(j)} = \tilde{x}_{j-1}$
  Sample $J_j$, $|J_j| = B_j$
  $v_0^{(j)} = {}^{1}/_{B_j} \sum_{i \in J_j} \nabla f_i(x_0^{(j)})$
  Sample $N_j \sim$ Geom($\gamma_j$) s.t. $\mathbb{E}N_j = {}^{m_j}/_{b_j}$
  for $k = 0, \ldots, N_j - 1$ do
    $x_{k+1}^{(j)} = x_k^{(j)} - \eta_j v_k^{(j)}$
    Sample $I_k^{(j)}$, $|I_k^{(j)}| = b_j$
    $v_{k+1}^{(j)} = ({}^{1}/_{b_j}) \sum_{i \in I_k^{(j)}} (\nabla f_i(x_{k+1}^{(j)}) - \nabla f_i(x_k^{(j)})) + v_k^{(j)}$
  end for
end for
Generate $\mathcal{R}(T)$ supported on $\{T, \ldots, \lceil (1+\delta)T \rceil\}$ with $\mathrm{Prob}(\mathcal{R}(T) = j) = {}^{\eta_j m_j}/_{\sum_{j=T}^{\lceil (1+\delta)T \rceil} \eta_j m_j}$
Output: $\tilde{x}_{\mathcal{R}(T)}$
```

The algorithm that we propose can be seen as a combination of the structure of `SCSG` methods [27, 29] and the `SARAH` biased gradient estimator $v_{k+1}^{(j)} = ({}^{1}/_{b_j}) \sum_{i \in I_k^{(j)}} \left(\nabla f_i(x_{k+1}^{(j)}) - \nabla f_i(x_k^{(j)})\right) + v_k^{(j)}$ due to its recent success in the non-convex setting. Our algorithm consists of several epochs. In each epoch, we start with an initial point $x_0^{(j)}$ from which the gradient estimator is computed using B_j sampled indices, which is not necessarily the full gradient as in the case of classic `SARAH` or `SVRG` algorithm. After this step, we incorporate geometrization of the inner-loop, where the epoch length is sampled from a geometric distribution with predefined mean m_j and in each step of the inner-loop, the `SARAH` gradient estimator with batch size b_j is used to update the current solution estimate. At the end of each epoch, the last point is taken as the initial estimate for consecutive epoch. The output of our algorithm is then a random iterate $\tilde{x}_{\mathcal{R}(T)}$, where the index $\mathcal{R}(T)$ is sampled such that $\mathrm{Prob}(\mathcal{R}(T) = j) = {}^{\eta_j m_j}/_{\sum_{j=T+1}^{\lceil (1+\delta)T \rceil} \eta_j m_j}$ for $j = T, \ldots, \lceil (1+\delta)T \rceil$. Note that $\mathcal{R}(T) = T$ when $\delta = 0$. This procedure can be seen tail-randomized iterate which as an analogue of tail-averaging in the convex-case [46]. For functions f with finite support (finite n), the sampling procedure in Algorithm 1 is sampling without replacement. For the infinite support, this is just B_j or b_j i.i.d. samples, respectively. The pseudo-code is shown in Algorithm 1.

Define $T_g(\epsilon)$ and $T_f(\epsilon)$ as the iteration complexity to find an ϵ-approximate first-order stationary point and an ϵ-approximate solution, respectively:

$$T_g(\epsilon) \overset{\text{def}}{=} \min\{T \,:\, \mathbb{E}\left\|\nabla f(\tilde{x}_{\mathcal{R}(T)})\right\|^2 \le \epsilon^2, \forall T' \ge T\}, \tag{6}$$

$$T_f(\epsilon) \overset{\text{def}}{=} \min\{T \,:\, \mathbb{E}(f(\tilde{x}_{\mathcal{R}(T)}) - f(x^\star)) \le \epsilon^2, \forall T' \ge T\}, \tag{7}$$

where $\tilde{x}_{\mathcal{R}(T)}$ is output of given algorithm.

The query complexity to find an ϵ-approximate first-order stationary point and an ϵ-approximate solution are defined as $\mathrm{Comp}_g(\epsilon)$ and $\mathrm{Comp}_f(\epsilon)$, respectively. It is easy to see that

$$\mathbb{E}\mathrm{Comp}_g(\epsilon) = \sum_{j=1}^{\lceil(1+\delta)T_g(\epsilon)\rceil}(2m_j + B_j), \quad \mathbb{E}\mathrm{Comp}_f(\epsilon) = \sum_{j=1}^{\lceil(1+\delta)T_f(\epsilon)\rceil}(2m_j + B_j).$$

4. Convergence Analysis

We conduct the analysis of our method in the way, where we first look at the progress of inner cycle for which we establish bounds on the norm of the gradient, which is subsequently used to prove convergence of the full algorithm. We assume f to be L-smooth and satisfy PL condition with $\mu \ge 0$.

4.1. One Epoch Analysis

We start from a one-epoch analysis that connects consecutive iterates. It lays the foundation for complexity analysis. The analysis is similar to [11] and presented in Appendix C.

Theorem 7 *Assume that $2\eta_j L \le \min\{1, b_j/\sqrt{m_j}\}$, then under assumptions 1 and 2,*

$$\mathbb{E}\|\nabla f(\tilde{x}_j)\|^2 \le \frac{2b_j}{\eta_j m_j}\mathbb{E}(f(\tilde{x}_{j-1}) - f(\tilde{x}_j)) + \frac{\sigma^2 I(B_j < n)}{B_j}.$$

4.2. Complexity Analysis

We consider two versions of our algorithm–`Q-Geom-SARAH` and `E-Geom-SARAH`. These two version differs only in the way how we select the big batch size B_j for our algorithm. For `Q-Geom-SARAH`, we select quadratic growth of B_j and `E-Geom-SARAH`, this is selected to be exponential. The convergence guarantees follow with all proofs relegated to Appendix.

Theorem 8 (`Q-Geom-SARAH`) *Set the hyperparameters as*

$$\eta_j = \frac{b_j}{2L\sqrt{m_j}}, \quad b_j \le \sqrt{m_j}, \quad m_j = B_j = j^2 \wedge n, \quad \delta = 1.$$

Then

$$\mathbb{E}\mathrm{Comp}_g(\epsilon) = \tilde{\mathcal{O}}\left(\left\{\frac{L^3}{\mu^3} + \frac{\sigma^3}{\epsilon^3}\right\} \wedge \left\{n^{3/2} + \frac{\sqrt{n}L}{\mu}\right\} \wedge \frac{(L\Delta_f)^{3/2} + \sigma^3}{\epsilon^3} \wedge \frac{\sqrt{n}(L\Delta_f + \sigma^2)}{\epsilon^2}\right),$$

$$\mathbb{E}\mathrm{Comp}_f(\epsilon) = \tilde{\mathcal{O}}\left(\left\{\frac{L^3}{\mu^3} + \frac{\sigma^3}{\mu^{3/2}\epsilon^3}\right\} \wedge \left\{n^{3/2} + \frac{\sqrt{n}L}{\mu}\right\}\right).$$

where $\tilde{O}$ only hides universal constantsand logarithmic terms.

In Appendix B.1, we state the detailed complexity bound in Theorem 10 without hiding any logarithmic terms. Theorem 8 shows an unusually strong adaptivity in that the last two terms match the state-of-the-art complexity [12] for general smooth non-convex optimization while it may be further improved when PL constant is large without any tweaks.

There is a gap between the complexity of `Q-Geom-SARAH` and the best achievable rate by non-adaptive algorithms in the PL case. This motivates us to consider another variant of `Geom-SARAH` that performs better for PL objectives while still have guarantees for general smooth nonconvex objectives.

Theorem 9 (`E-Geom-SARAH`) *Fix any $\alpha > 1$ and $\delta \in (0, 1]$. Set the hyperparameters as*

$$\eta_j = \frac{b_j}{2L\sqrt{m_j}}, \quad b_j \le \sqrt{m_j}, \text{ where } m_j = \alpha^{2j} \wedge n, \quad B_j = \lceil \alpha^{2j} \wedge n \rceil.$$

Then

$$\mathbb{E}\mathrm{Comp}_g(\epsilon) = \tilde{\mathcal{O}}\left(\left\{ \frac{L^{2(1+\delta)}}{\mu^{2(1+\delta)}} + \left(\frac{\sigma}{\epsilon}\right)^{2(1+\delta)} \right\} \wedge \left\{ n + \frac{\sqrt{n}L}{\mu} \right\} \wedge \frac{(L\Delta_f)^2 + \sigma^4}{\epsilon^4} \wedge \frac{\sqrt{n}(L\Delta_f + \sigma^2)}{\epsilon^2} \right),$$

$$\mathbb{E}\mathrm{Comp}_f(\epsilon) = \tilde{\mathcal{O}}\left(\left\{ \frac{L^{2(1+\delta)}}{\mu^{2(1+\delta)}} + \left(\frac{\sigma^2}{\mu\epsilon^2}\right)^{1+\delta} \right\} \wedge \left\{ n + \frac{\sqrt{n}L}{\mu} \right\} \right).$$

where $\tilde{\mathcal{O}}$ hides sub-polynomial terms defined in Appendix B.1, and constants that depend on α.

In Appendix B.1, we state the detailed complexity bound in Theorem 12 without hiding any sub-polynomial terms. Note that in order to provide convergence result for all the cases we need δ to be arbitrarily small positive constant, thus one might almost ignore factor $1 + \delta$ in the complexity results. Recall that $\delta = 0$ implies $\mathcal{R}(T) = T$ meaning that the output of an algorithm is the last iterate, which is common setting, e.g. for `Spiderboost` or `SARAH`, under assumption $\mu > 0$.

References

[1] Alekh Agarwal and Leon Bottou. A lower bound for the optimization of finite sums. *Proceedings of the 32nd International Conference on Machine Learning*, 2015.

[2] Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. QSGD: Communication-efficient SGD via gradient quantization and encoding. In *Advances in Neural Information Processing Systems*, pages 1709–1720, 2017.

[3] Zeyuan Allen-Zhu. Katyusha: The first direct acceleration of stochastic gradient methods. *The Journal of Machine Learning Research*, 18(1):8194–8244, 2017.

[4] Zeyuan Allen-Zhu. How to make the gradients small stochastically: Even faster convex and nonconvex sgd. In *Advances in Neural Information Processing Systems*, pages 1157–1167, 2018.

[5] Francis Bach and Eric Moulines. Non-strongly-convex smooth stochastic approximation with convergence rate $o(1/n)$. In *Advances in Neural Information Processing Systems*, pages 773–781, 2013.

[6] Zaiyi Chen, Yi Xu, Enhong Chen, and Tianbao Yang. SADAGRAD: Strongly adaptive stochastic gradient methods. In *International Conference on Machine Learning*, pages 912–920, 2018.

[7] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems*, pages 1646–1654, 2014.

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

[9] Aymeric Dieuleveut, Nicolas Flammarion, and Francis Bach. Harder, better, faster, stronger convergence rates for least-squares regression. *The Journal of Machine Learning Research*, 18(1):3520–3570, 2017.

[10] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

[11] Melih Elibol, Lihua Lei, and Michael I Jordan. Variance reduction with sparse gradients. *ICLR - International Conference on Learning Representations*, 2020.

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

[13] Nicolas Flammarion and Francis Bach. From averaging to acceleration, there is only a step-size. In *COLT - Conference on Learning Theory*, pages 658–695, 2015.

[14] Eduard Gorbunov, Filip Hanzely, and Peter Richtárik. A unified theory of SGD: Variance Reduction, Sampling, Quantization and Coordinate Descent. In *The 23rd International Conference on Artificial Intelligence and Statistics*, 2020.

[15] Robert Mansel Gower, Peter Richtárik, and Francis Bach. Stochastic quasi-gradient methods: variance reduction via Jacobian sketching. *arXiv:1805.02632*, 2018.

[16] Robert Mansel Gower, Nicolas Loizou, Xun Qian, Alibek Sailanbayev, Egor Shulgin, and Peter Richtárik. SGD: General analysis and improved rates. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5200–5209, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

[17] Filip Hanzely and Peter Richtárik. One method to rule them all: variance reduction for data, parameters and many new methods. *arXiv preprint arXiv:1905.11266*, 2019.

[18] Elad Hazan and Sham Kakade. Revisiting the polyak step size. *arXiv preprint arXiv:1905.00313*, 2019.

[19] Thomas Hofmann, Aurelien Lucchi, Simon Lacoste-Julien, and Brian McWilliams. Variance reduced stochastic gradient descent with neighbors. In *Advances in Neural Information Processing Systems*, pages 2305–2313, 2015.

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

[21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR - International Conference on Learning Representations*, 2015.

[22] Jakub Konečný and Peter Richtárik. Semi-stochastic gradient descent methods. *arXiv preprint arXiv:1312.1666*, 2013.

[23] Jakub Konečný, Zheng Qu, and Peter Richtárik. S2CD: Semi-stochastic coordinate descent. *Optimization Methods and Software*, 32(5):993–1005, 2017.

[24] Dmitry Kovalev, Konstantin Mishchenko, and Peter Richtárik. Stochastic newton and cubic newton methods with simple local linear-quadratic rates. *arXiv preprint arXiv:1912.01597*, 2019.

[25] Dmitry Kovalev, Samuel Horváth, and Peter Richtárik. Don't jump through hoops and remove those loops: Svrg and katyusha are better without the outer loop. *ALT-The 31st International Conference on Algorithmic Learning Theory*, 2020.

[26] Guanghui Lan, Zhize Li, and Yi Zhou. A unified variance-reduced accelerated gradient method for convex optimization. *Advances in Neural Information Processing Systems*, 2019.

[27] Lihua Lei and Michael I Jordan. Less than a single pass: Stochastically controlled stochastic gradient method. *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, 2017.

[28] Lihua Lei and Michael I Jordan. On the adaptivity of stochastic gradient-based optimization. *SIAM Journal on Optimization (SIOPT)*, 2019.

[29] Lihua Lei, Cheng Ju, Jianbo Chen, and Michael I Jordan. Non-convex finite-sum optimization via scsg methods. In *Advances in Neural Information Processing Systems*, pages 2348–2358, 2017.

[30] Yehuda Kfir Levy, Alp Yurtsever, and Volkan Cevher. Online adaptive methods, universality and acceleration. In *Advances in Neural Information Processing Systems*, pages 6500–6509, 2018.

[31] Zhize Li and Jian Li. A simple proximal stochastic gradient method for nonsmooth nonconvex optimization. In *Advances in Neural Information Processing Systems*, pages 5564–5574, 2018.

[32] Chenxin Ma, Jakub Konečný, Martin Jaggi, Virginia Smith, Michael I Jordan, Peter Richtárik, and Martin Takáč. Distributed optimization with arbitrary local solvers. *Optimization Methods and Software*, 32(4):813–848, 2017.

[33] Yura Malitsky and Konstantin Mishchenko. Adaptive gradient descent without descent. *arXiv preprint arXiv:1910.09529*, 2019.

[34] 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.

[35] Arkadi Nemirovski, Anatoli Juditsky, Guanghui Lan, and Alexander Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19(4):1574–1609, 2009.

[36] Arkadii Semenovich Nemirovsky and David Borisovich Yudin. *Problem complexity and method efficiency in optimization.* 1983.

[37] Yu Nesterov. Universal gradient methods for convex optimization problems. *Mathematical Programming*, 152(1-2):381–404, 2015.

[38] Yurii Nesterov. *Lectures on convex optimization*, volume 137. Springer, 2018.

[39] Lam M Nguyen, Jie Liu, Katya Scheinberg, and Martin Takáč. SARAH: A novel method for machine learning problems using stochastic recursive gradient. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2613–2621. JMLR. org, 2017.

[40] Lam M Nguyen, Marten van Dijk, Dzung T Phan, Phuong Ha Nguyen, Tsui-Wei Weng, and Jayant R Kalagnanam. Finite-sum smooth optimization with SARAH. *def*, 1:1, 2019.

[41] Boris Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

[42] Boris T Polyak. New stochastic approximation type procedures. *Automat. i Telemekh*, 7(98-107):2, 1990.

[43] Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization*, 30(4):838–855, 1992.

[44] Zheng Qu, Peter Richtárik, and Tong Zhang. Quartz: Randomized dual coordinate ascent with arbitrary sampling. In *Advances in Neural Information Processing Systems*, pages 865–873, 2015.

[45] Anant Raj and Sebastian U Stich. k-SVRG: Variance Reduction for Large Scale Optimization. *arXiv preprint arXiv:1805.00982*, 2018.

[46] Alexander Rakhlin, Ohad Shamir, and Karthik Sridharan. Making gradient descent optimal for strongly convex stochastic optimization. *Proceedings of the 29 th International Conference on Machine Learning*, 2012.

[47] Sashank J Reddi, Ahmed Hefny, Suvrit Sra, Barnabas Poczos, and Alex Smola. Stochastic variance reduction for nonconvex optimization. In *International conference on machine learning*, pages 314–323, 2016.

[48] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of Adam and beyond. *ICLR - International Conference on Learning Representations*, 2018.

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

[50] David Ruppert. Efficient estimators from a slowly convergent robbins-monro procedure. *School of Oper. Res. and Ind. Eng., Cornell Univ., Ithaca, NY, Tech. Rep*, 781, 1988.

[51] Shai Shalev-Shwartz and Tong Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *Journal of Machine Learning Research*, 14(Feb):567–599, 2013.

[52] Sebastian U Stich. Local SGD converges fast and communicates little. *ICLR - International Conference on Learning Representations*, 2019.

[53] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International Conference on Machine Learning*, pages 1139–1147, 2013.

[54] Quoc Tran-Dinh, Nhan H Pham, Dzung T Phan, and Lam M Nguyen. Hybrid stochastic gradient descent algorithms for stochastic nonconvex optimization. *arXiv preprint arXiv:1905.05920*, 2019.

[55] Nilesh Tripuraneni, Mitchell Stern, Chi Jin, Jeffrey Regier, and Michael I Jordan. Stochastic cubic regularization for fast nonconvex optimization. In *Advances in Neural Information Processing Systems*, pages 2899–2908, 2018.

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

[57] Zhe Wang, Kaiyi Ji, Yi Zhou, Yingbin Liang, and Vahid Tarokh. Spiderboost: A class of faster variance-reduced algorithms for nonconvex optimization. *Advances in Neural Information Processing Systems*, 2019.

[58] Yi Xu, Qihang Lin, and Tianbao Yang. Adaptive SVRG methods under error bound conditions with unknown growth parameter. In *Advances in Neural Information Processing Systems*, pages 3279–3289, 2017.

[59] Yi Xu, Qihang Lin, and Tianbao Yang. Accelerate stochastic subgradient method by leveraging local growth condition. *Analysis and Applications*, 17(5):773–818, 2019.

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

[61] Dongruo Zhou, Pan Xu, and Quanquan 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.

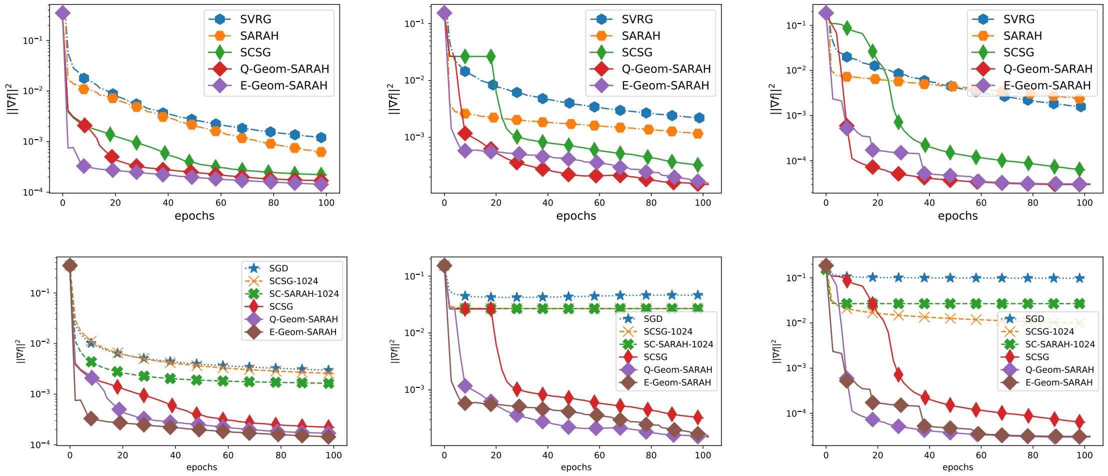

Figure 1: Comparison of convergence with respect to norm of the gradient for different high (top row) low precision (bottom row) VR methods. Datasets: `mushrooms` (left) `w8a` (middle) `ijcnn1` (right).

Appendix A. Experiments

To support our theoretical result, we conclude several experiments using logistic regression with non-convex penalty $\Psi_\lambda(x) = \lambda/2 \sum_{j=1}^{d} x_j^2/1+x_j^2$. The objective that we minimize is of the form

$$1/n \sum_{i=1}^{n} \log\left(1 + e^{-y_i w_i^\top x}\right) + \Psi_\lambda(x),$$

where w_i's are the features, y_i's the labels and $\lambda > 0$ is a regularization parameter. This fits to our framework with $L_{f_i} = \|a_i\|^2/4 + \lambda$. We compare our adaptive methods against state-of-the-art methods in this framework– `SARAH` [40], `SVRG` [47], `Spiderboost` [57], adaptive and fixed version of `SCSG` [29] with big batch sizes $B = cj^{3/2} \wedge n$ for some constant c. We use all the methods with their theoretical parameters. We use `SARAH` and `Spiderboost` with constant step size $1/2L$, which implies batch size to be $b = \sqrt{n}$. In this scenario, `Spiderboost` and `SARAH` are the same algorithm and we refer to both as `SARAH`. The same step size is also used for `SVRG` which requires batch size $b = n^{2/3}$. The same applies to `SCSG` and our methods and we adjust parameter accordingly, e.g. this applies that for our methods we set $b_j = \sqrt{m_j}$. For `E-Geom-SARAH`, we chose $\alpha = 2$. We also include `SGD` methods with the same step size for comparison. All the experiments are run with $\lambda = 0.1$. We use three dataset from LibSVM[3]: *mushrooms* ($n = 8,124, p = 112$), *w8a* ($n = 49,749, p = 300$), and *ijcnn1* ($n = 49,990, p = 22$).

We run two sets of experiments– low and high precision. Firstly, we compare our adaptive methods with the ones that can guarantee convergence to arbitrary precision ϵ – `SARAH`, `SVRG` and adaptive `SCSG`. Secondly, we conclude the experiment where we compare our adaptive methods against ones that should provide better convergence in low precision regimes– `SARAH` and `SVRG` with big batch size $B = 1024$, adaptive `SCSG` and `SGD` with batch size equal to 32. For all the experiments, we display functional value and norm of the gradient with respect to number of epochs (IFO calls divided by n). For all Figures 2 and 1,

[3] available on https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

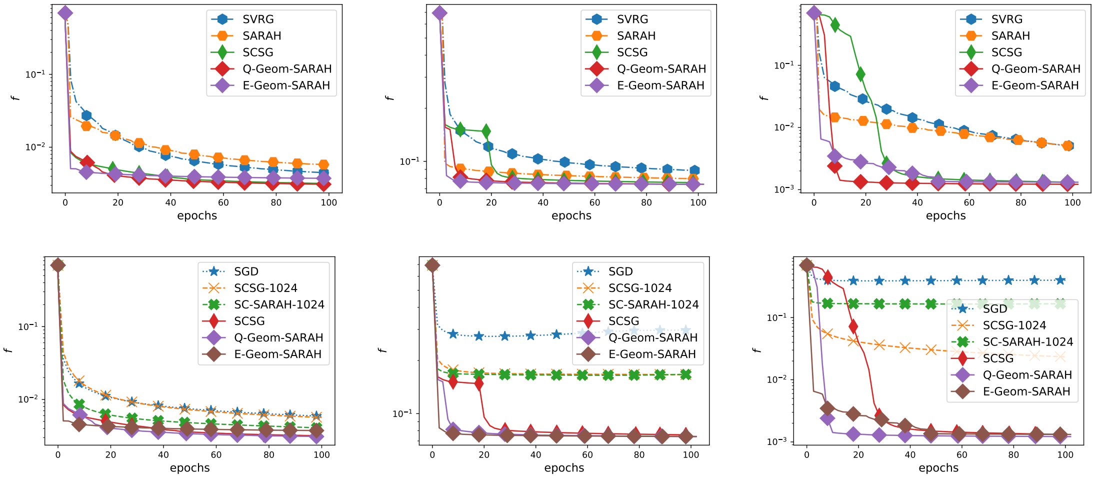

Figure 2: Comparison of convergence with respect to functional value for different high (top row) low precision (bottom row) VR methods. Datasets: `mushrooms` (left) `w8a` (middle) `ijcnn1` (right).

we can see that our adaptive method perfoms the best in all the regimes and the only method that reaches comparable performance is `SCSG`.

Appendix B. Detailed Theoretical Results

B.1. Theorems with all terms included

We state more detailed complexity bounds for `Q-Geom-SARAH` and `E-Geom-SARAH` by revealing the logarithmic and sub-polynomial terms. Throughout this subsection, we define Δ as $\Delta_f + \sigma^2/L$.

Theorem 10 (`Q-Geom-SARAH`) *Set the hyperparameters as*

$$\eta_j = \frac{b_j}{2L\sqrt{m_j}}, \quad b_j \le \sqrt{m_j}, \quad m_j = B_j = j^2 \wedge n, \quad \delta = 1.$$

Then

$$\begin{aligned}\mathbb{E}\mathrm{Comp}_g(\epsilon) = \mathcal{O}\Bigg(&\left\{ \frac{L^3}{\mu^3} + \frac{\sigma^3}{\epsilon^3} + \log^3\left(\frac{\mu\Delta}{\epsilon^2}\right)\right\} \wedge \left\{ n^{3/2} + \left(n + \frac{\sqrt{n}L}{\mu}\right)\log\left(\frac{L\Delta}{\sqrt{n}\epsilon^2}\right)\right\} \\ &\wedge \left\{ \frac{(L\Delta)^{3/2}}{\epsilon^3} + \frac{\sigma^3}{\epsilon^3}\log^3\left(\frac{\sigma}{\epsilon}\right)\right\} \wedge \left\{ \frac{\sqrt{n}L\Delta}{\epsilon^2} + \frac{\sqrt{n}\sigma^2}{\epsilon^2}\log^3 n\right\}\Bigg),\\ \mathbb{E}\mathrm{Comp}_f(\epsilon) = \mathcal{O}\Bigg(&\left\{ \frac{L^3}{\mu^3} + \frac{\sigma^3}{\mu^{3/2}\epsilon^3} + \log^3\left(\frac{\Delta}{\epsilon^2}\right)\right\} \wedge \left\{ n^{3/2} + \left(n + \frac{\sqrt{n}L}{\mu}\right)\log\left(\frac{\Delta}{\epsilon^2}\right)\right\}\Bigg),\end{aligned}$$

where $\mathcal{O}$ only hides universal constants.

Remark 11 *Theorem 10 continues to hold if $\eta_j L = \theta b_j/\sqrt{m_j}$ for any $0 < \theta_j < 1/2$ and $m_j, B_j \in [a_1 j^2, a_2 j^2]$ for some $0 < a_1 < a_2 < \infty$ for sufficiently large j.*

Let es denote the exponential square-root, i.e. $\mathrm{es}(x) = \exp\{\sqrt{x}\}$. It is easy to see that $\log x = \mathcal{O}(\mathrm{es}(\log x))$ and $\mathrm{es}(\log x) = \mathcal{O}\left(x^a\right)$ for any $a > 0$.

Theorem 12 (`E-Geom-SARAH`) *Fix any $\alpha > 1$ and $\delta \in (0, 1]$. Set the hyperparameters as*

$$\eta_j = \frac{b_j}{2L\sqrt{m_j}}, \quad b_j \leq \sqrt{m_j}, \text{ where } m_j = \alpha^{2j} \wedge n, \quad B_j = \lceil \alpha^{2j} \wedge n \rceil.$$

Then

$$\begin{aligned}\mathbb{E}\mathrm{Comp}_g(\epsilon) = \mathcal{O}\Bigg(&\left\{ \frac{L^{2(1+\delta)}}{\mu^{2(1+\delta)}} \mathrm{es}\left(2\log_\alpha\left\{\frac{\mu\Delta}{\epsilon^2}\right\}\right) + \left(\frac{\sigma}{\epsilon}\right)^{2(1+\delta)} \right\} \log^2 n \\ &\wedge \left\{ n\log\left(\frac{L}{\mu}\right) + \left(n + \frac{\sqrt{n}L}{\mu}\right)\log\left(\frac{L\Delta}{\sqrt{n}\epsilon^2}\right)\right\} \wedge \frac{(L\Delta)^2}{\delta^2\epsilon^4} \wedge \frac{\sqrt{n}L\Delta\log n}{\delta\epsilon^2}\Bigg), \\ \mathbb{E}\mathrm{Comp}_f(\epsilon) = \mathcal{O}\Bigg(&\left\{ \frac{L^{2(1+\delta)}}{\mu^{2(1+\delta)}} \mathrm{es}\left(2\log_\alpha\left\{\frac{\Delta}{\epsilon^2}\right\}\right) + \left(\frac{\sigma^2}{\mu\epsilon^2}\right)^{1+\delta} \right\} \log^2 n \\ &\wedge \left\{ n\log\left(\frac{L}{\mu}\right) + \left(n + \frac{\sqrt{n}L}{\mu}\right)\log\left(\frac{\Delta}{\epsilon^2}\right)\right\}\Bigg),\end{aligned}$$

where $\mathcal{O}$ only hides universal constants and constants that depend on α.

B.2. Better rates for non-adaptive `Geom-SARAH`

In this section, we provide the versions of our algorithms, which are neither almost universal nor ϵ-independent, but they either reach the known lower bounds or best achievable results known in literature. We include this result for two reasons. Firstly, we want to show there is a small gap between results in Section 4.2 and the best results, which might be obtained. We conjecture that this gap is inevitable. Secondly, our complexity result for the functional gap beats the best known complexity result known in literature which is $\mathcal{O}\left(\log^3 B\{B + \sqrt{B}/\mu\}\log(1/\epsilon)\right)$, where $B = \mathcal{O}\left(\{\sigma^2/\mu\epsilon^2\} \wedge n\right)$ [61], where our complexity result does not involve $\log^3 B$ factor. Finally, we obtain very interesting result for the norm of the gradient, which we discuss later in this section. The proofs are relegated into Appendix C.

Theorem 13 (Non-adaptive) *Set the hyperparameters as*

$$\eta_j = \frac{b_j}{2L\sqrt{m_j}}, \quad b_j \leq \sqrt{m_j}, \quad B_j = m_j = B.$$

1. *If $B = \left(\frac{\sigma^2}{4\mu\epsilon^2} \wedge n\right)$ and $\delta = 0$ then*

$$\mathbb{E}\mathrm{Comp}_f(\epsilon) = \mathcal{O}\left(\left(B + \frac{\sqrt{B}L}{\mu}\right)\log\left(\frac{\Delta_f}{\epsilon^2}\right)\right)$$

.

2. *If $B = \left(\left\{\frac{8\sigma^2}{\epsilon^2} + \frac{8\sigma^{4/3}L^{2/3}}{\epsilon^{4/3}\mu^{2/3}}\right\} \wedge n\right)$ and $\delta = 0$ then*

$$\mathbb{E}\mathrm{Comp}_g(\epsilon) = \mathcal{O}\left(\left(B + \frac{\sqrt{B}L}{\mu}\right)\log\frac{L\Delta_f}{\sqrt{B}\epsilon^2}\right)$$

.

Looking into these result, there is one important thing to note. While these methods reach state-of-the-art performance for PL objectives, they provide no guarantees for the case $\mu = 0$.

For the ease of presentation we assume $\sigma^2, \Delta_f, L = \mathcal{O}(1)$. For Q-Geom-SARAH, we can see that in term of $\tilde{\mathcal{O}}$ notation, we match the best reachable rate in case $\mu = 0$. For the case $\mu > 0$, we see slight degradation in performance for both high and low precision regimes. For E-Geom-SARAH, we can see a bit different results. There is a $1/\epsilon$ degradation comparing to the low precision case and exact match for high precision case with $\mu = 0$. For the case $\mu > 0$, E-Geom-SARAH matches the best achievable rate for high precision and also for in low precision regime in the case when rate is dominated by factor $1/\epsilon^2$. Comparison to other methods together with the dependence on parameters can be found in Tables 1 and 2.

One interesting fact to note is that in the second case of Theorem 13, if $\mu \sim \epsilon$ and $L, \Delta_f, \sigma^2 = \mathcal{O}(1)$, $B \sim 1/\epsilon^2$ and $\mathbb{E}\mathrm{Comp}_g(\epsilon) = \mathcal{O}\left(1/\epsilon^2 \log\left(1/\epsilon\right)\right)$. This is even logarithmically better than the rate $\mathcal{O}(1/\epsilon^2 \log^3(1/\mu))$ obtained by [4] for strongly-convex functions. Note that a strongly convex function with modulus μ is always μ-PL. We plan to further investigate this strong result in the future.

Appendix C. Proofs

C.1. Proof of Lemma 5

By definition,

$$
\begin{aligned}
E(D_N - D_{N+1}) &= \sum_{n\geq 0}(D_k - D_{k+1})\cdot\gamma^k(1-\gamma) \\
&= (1-\gamma)(D_0 - \sum_{k\geq 1} D_k(\gamma^{k-1} - \gamma^k)) = (1-\gamma)\left(\frac{1}{\gamma}D_0 - \sum_{k\geq 0} D_k(\gamma^{k-1}-\gamma^k)\right) \\
&= (1-\gamma)\left(\frac{1}{\gamma}D_0 - \frac{1}{\gamma}\sum_{k\geq 0} D_k\gamma^k(1-\gamma)\right) = \frac{1-\gamma}{\gamma}(D_0 - \mathbb{E}D_N),
\end{aligned}
$$

where the last equality is implied by the condition that $\mathbb{E}|D_N| < \infty$.

In order to use Lemma 5, one needs to show $\mathbb{E}|D_N| < \infty$. We start with the following lemma as the basis to apply geometrization. The proof is distracting and relegated to the end of this section.

Lemma 14 *Assume that $\eta_j L \leq 1$. Then $\mathbb{E}|D^{(s)}_{N_j}| < \infty$ for $s = 1, 2, 3$, where*

$$
D^{(1)}_k = \mathbb{E}_j\left\|\nu^{(j)}_k - \nabla f(x^{(j)}_k)\right\|^2, \quad D^{(2)}_k = \mathbb{E}_j f(x^{(j)}_k), \quad D^{(3)}_k = \mathbb{E}_j\left\|\nabla f(x^{(j)}_k)\right\|^2,
$$

and $\mathbb{E}_j$ denotes the expectation over the randomness in j-th outer loop.

Based on Lemma 14, we prove two lemmas, which helps us to establish the sequence that is used to prove convergence. Throughout the rest of the section we assume that assumption 1 and 2 hold.

Lemma 15 *For any j,*

$$
\mathbb{E}_j\left\|\nu^{(j)}_{N_j} - \nabla f(\tilde{x}_j)\right\|^2 \leq \frac{m_j\eta_j^2 L^2}{b_j^2}\mathbb{E}_j\left\|\nu^{(j)}_{N_j}\right\|^2 + \frac{\sigma^2 I(B_j \leq n)}{B_j},
$$

where $\mathbb{E}_j$ denotes the expectation over the randomness in j-th outer loop.

Proof Let $\mathbb{E}_{j,k}$ and $\mathrm{Var}_{j,k}$ denote the expectation and variance operator over the randomness of $\mathcal{I}_k^{(j)}$. Since $\mathcal{I}_k^{(j)}$ is independent of $x_k^{(j)}$,

$$\mathbb{E}_{j,k}\nu_{k+1}^{(j)} = \nu_k^{(j)} + (\nabla f(x_{k+1}^{(j)}) - \nabla f(x_k^{(j)})).$$

Thus,

$$\nu_{k+1}^{(j)} - \nabla f(x_{k+1}^{(j)}) = \nu_k^{(j)} - \nabla f(x_k^{(j)}) + \left(\nu_{k+1}^{(j)} - \nu_k^{(j)} - \mathbb{E}_{j,k}(\nu_{k+1}^{(j)} - \nu_k^{(j)})\right).$$

Since $\mathcal{I}_k^{(j)}$ is independent of $(\nu_k^{(j)}, x_k^{(j)})$,

$$\mathrm{Cov}_{j,k}\left(\nu_k^{(j)} - \nabla f(x_k^{(j)}), \nu_{k+1}^{(j)} - \nu_k^{(j)}\right) = 0.$$

As a result,

$$\mathbb{E}_{j,k}\left\|\nu_{k+1}^{(j)} - \nabla f(x_{k+1}^{(j)})\right\|^2 = \left\|\nu_k^{(j)} - \nabla f(x_k^{(j)})\right\|^2 + \mathrm{Var}_{j,k}(\nu_{k+1}^{(j)} - \nu_k^{(j)}). \tag{8}$$

By Lemma 22,

$$\begin{aligned}
\mathrm{Var}_{j,k}(\nu_{k+1}^{(j)} - \nu_k^{(j)}) &= \mathrm{Var}\left(\frac{1}{b_j}\sum_{i\in\mathcal{I}_k^{(j)}}(\nabla f_i(x_{k+1}^{(j)}) - \nabla f_i(x_k^{(j)}))\right) \\
&\le \frac{1}{b_j}\frac{1}{n}\sum_{i=1}^{n}\left\|\nabla f_i(x_{k+1}^{(j)}) - \nabla f_i(x_k^{(j)}) - (\nabla f(x_{k+1}^{(j)}) - \nabla f(x_k^{(j)}))\right\|^2 \qquad (9)\\
&\le \frac{1}{b_j}\frac{1}{n}\sum_{i=1}^{n}\left\|\nabla f_i(x_{k+1}^{(j)}) - \nabla f_i(x_k^{(j)})\right\|^2.
\end{aligned}$$

Finally by assumption 1,

$$\frac{1}{n}\sum_{i=1}^{n}\left\|\nabla f_i(x_{k+1}^{(j)}) - \nabla f_i(x_k^{(j)})\right\|^2 \le L^2\left\|x_{k+1}^{(j)} - x_k^{(j)}\right\|^2 = \eta_j^2 L^2\left\|\nu_k^{(j)}\right\|^2.$$

By (8),

$$\mathbb{E}_{j,k}\left\|\nu_{k+1}^{(j)} - \nabla f(x_{k+1}^{(j)})\right\|^2 = \left\|\nu_k^{(j)} - \nabla f(x_k^{(j)})\right\|^2 + \frac{\eta_j^2 L^2}{b_j}\left\|\nu_k^{(j)}\right\|^2.$$

Let $k = N_j$ and take expectation over all randomness in $\mathbb{E}_j$. By Lemma 14, we can apply Lemma 5 on $D_k = \mathbb{E}_j\left\|\nu_k^{(j)} - \nabla f(x_k^{(j)})\right\|^2$. Then we have

$$\begin{aligned}
0 &\le \mathbb{E}_j\left(\|\nu_{N_j}^{(j)} - \nabla f(x_{N_j}^{(j)})\|^2 - \|\nu_{N_j+1}^{(j)} - \nabla f(x_{N_j+1}^{(j)})\|^2\right) + \frac{\eta_j^2 L^2}{b_j}\mathbb{E}_j\|\nu_{N_j}^{(j)}\|^2 \\
&= \frac{b_j}{m_j}\mathbb{E}_j\left(\|\nu_0^{(j)} - \nabla f(x_0^{(j)})\|^2 - \|\nu_{N_j}^{(j)} - \nabla f(x_{N_j}^{(j)})\|^2\right) + \frac{\eta_j^2 L^2}{b_j}\mathbb{E}_j\|\nu_{N_j}^{(j)}\|^2.
\end{aligned}$$

Finally, by Lemma 22,

$$\mathbb{E}_j\|\nu_0^{(j)} - \nabla f(x_0^{(j)})\|^2 \le \frac{\sigma^2 I(B_j < n)}{B_j}. \tag{10}$$

The proof is then completed. ■

Lemma 16 *For any j,*

$$\mathbb{E}_j \left\|\nabla f(\tilde{x}_j)\right\|^2 \leq \frac{2b_j}{\eta_j m_j}\mathbb{E}_j(f(\tilde{x}_{j-1}) - f(\tilde{x}_j)) + \mathbb{E}_j \left\|\nu_{N_j}^{(j)} - \nabla f(\tilde{x}_j)\right\|^2 - (1-\eta_j L)\mathbb{E}_j \left\|\nu_{N_j}^{(j)}\right\|^2,$$

where $\mathbb{E}_j$ denotes the expectation over the randomness in j-th outer loop.

Proof By assumption (1),

$$\begin{aligned} f(x_{k+1}^{(j)}) &\leq f(x_k^{(j)}) + \left\langle \nabla f(x_k^{(j)}), x_{k+1}^{(j)} - x_k^{(j)} \right\rangle + \frac{L}{2}\left\|x_k^{(j)} - x_{k+1}^{(j)}\right\|^2 \\ &= f(x_k^{(j)}) - \eta\left\langle \nabla f(x_k^{(j)}), \nu_k^{(j)} \right\rangle + \frac{\eta_j^2 L}{2}\left\|\nu_k^{(j)}\right\|^2 \\ &= f(x_k^{(j)}) + \frac{\eta_j}{2}\left\|\nu_k^{(j)} - \nabla f(x_k^{(j)})\right\|^2 - \frac{\eta_j}{2}\left\|\nabla f(x_k^{(j)})\right\|^2 - \frac{\eta_j}{2}\left\|\nu_k^{(j)}\right\|^2 + \frac{\eta_j^2 L}{2}\left\|\nu_k^{(j)}\right\|^2. \quad (11) \end{aligned}$$

Let $j = N_j$ and take expectation over all randomness in $\mathbb{E}_j$. By Lemma 14, we can apply Lemma 5 with $D_k = \mathbb{E}_j f(x_k^{(j)})$ and $D_k = \mathbb{E}_j \left\|\nabla f(x_k^{(j)})\right\|^2$. Thus,

$$\begin{aligned} 0 &\leq \mathbb{E}_j\left(f(x_{N_j}^{(j)}) - f(x_{N_j+1}^{(j)})\right) + \frac{\eta_j}{2}\mathbb{E}_j\left\|\nu_{N_j}^{(j)} - \nabla f(x_{N_j}^{(j)})\right\|^2 - \frac{\eta_j}{2}\mathbb{E}_j\left\|\nabla f(x_{N_j}^{(j)})\right\|^2 - \frac{\eta_j}{2}(1-\eta_j L)\mathbb{E}_j\left\|\nu_{N_j}^{(j)}\right\|^2 \\ &= \frac{b_j}{m_j}\mathbb{E}_j\left(f(x_0^{(j)}) - f(x_{N_j}^{(j)})\right) + \frac{\eta_j}{2}\mathbb{E}_j\left\|\nu_{N_j}^{(j)} - \nabla f(x_{N_j}^{(j)})\right\|^2 - \frac{\eta_j}{2}\mathbb{E}_j\left\|\nabla f(x_{N_j}^{(j)})\right\|^2 - \frac{\eta_j}{2}(1-\eta_j L)\mathbb{E}_j\left\|\nu_{N_j}^{(j)}\right\|^2 \\ &= \frac{b_j}{m_j}\mathbb{E}_j\left(f(\tilde{x}_{j-1}) - f(\tilde{x}_j)\right) + \frac{\eta_j}{2}\mathbb{E}_j\left\|\nu_{N_j}^{(j)} - \nabla f(x_{N_j}^{(j)})\right\|^2 - \frac{\eta_j}{2}\mathbb{E}_j\left\|\nabla f(\tilde{x}_j)\right\|^2 - \frac{\eta_j}{2}(1-\eta_j L)\mathbb{E}_j\left\|\nu_{N_j}^{(j)}\right\|^2. \end{aligned}$$

The proof is then completed. ■

Theorem 7 is then proved by combining Lemma 15 and Lemma 16.

Proof [Proof of Theorem 7] By Lemma 15 and Lemma 16,

$$\mathbb{E}\|\nabla f(\tilde{x}_j)\|^2 \leq \frac{2b_j}{\eta_j m_j}\mathbb{E}(f(\tilde{x}_{j-1}) - f(\tilde{x}_j)) + \frac{\sigma^2 I(B_j < n)}{B_j} - \left(1 - \eta_j L - \frac{m_j \eta_j^2 L^2}{b_j^2}\right)\mathbb{E}\|\nu_{N_j}^{(j)}\|^2.$$

Under condition $2\eta_j L \leq \min\left\{1, \frac{b_j}{\sqrt{m_j}}\right\}$,

$$1 - \eta_j L - \frac{m_j(\eta_j L)^2}{b_j^2} \geq 1 - \frac{1}{2} - \frac{1}{4} \geq 0,$$

which concludes the proof. ■

Proof [Proof of Lemma 14] By (9) and assumption 2,

$$\begin{aligned} \mathrm{Var}_{j,k}(\nu_{k+1}^{(j)} - \nu_k^{(j)}) &\leq \frac{2}{b_j n}\left(\sum_{i=1}^n\left\|\nabla f_i(x_{k+1}^{(j)}) - \nabla f(x_{k+1}^{(j)})\right\|^2 + \sum_{i=1}^n\left\|\nabla f_i(x_k^{(j)}) - \nabla f(x_k^{(j)})\right\|^2\right) \\ &\leq \frac{4\sigma^2}{b_j} \leq 4\sigma^2. \end{aligned}$$

By (8) and taking expectation over all randomness in epoch j,

$$\mathbb{E}_j \left\|\nu_{k+1}^{(j)} - \nabla f(x_{k+1}^{(j)})\right\|^2 \le \mathbb{E}_j \left\|\nu_k^{(j)} - \nabla f(x_k^{(j)})\right\|^2 + 4\sigma^2.$$

Then

$$\mathbb{E}_j \left\|\nu_k^{(j)} - \nabla f(x_k^{(j)})\right\|^2 \le \left\|\nu_0^{(j)} - \nabla f(x_0^{(j)})\right\|^2 + 4k\sigma^2 \le (4k+1)\sigma^2 = \mathrm{poly}(k), \tag{12}$$

where the last inequality uses (10). By remark 6, we obtain that $\mathbb{E}|D_{N_j}^{(1)}| < \infty$.

On the other hand, by (11), since $\eta_j L \le 1$,

$$f(x_{k+1}^{(j)}) + \frac{\eta_j}{2}\left\|\nabla f(x_k^{(j)})\right\|^2 \le f(x_k^{(j)}) + \frac{\eta_j}{2}\left\|\nu_k^{(j)} - \nabla f(x_k^{(j)})\right\|^2 \le f(x_k^{(j)}) + (2k+1)\eta_j\sigma^2,$$

where the last inequality uses (12). Let

$$M_k^{(j)} = f(x_{k+1}^{(j)}) - f(x^\star) + \frac{\eta_j}{2}\left\|\nabla f(x_k^{(j)})\right\|^2.$$

Then

$$M_k^{(j)} \le M_{k-1}^{(j)} + (2k+1)\eta_j\sigma^2.$$

Applying the above inequality recursively, we have

$$M_k^{(j)} \le M_0^{(j)} + (k^2 + 2k)\eta_j\sigma^2 = \mathrm{poly}(k).$$

As a result,

$$0 \le f(x_{k+1}^{(j)}) - f(x^\star) \le M_{k-1}^{(j)} = \mathrm{poly}(k), \quad 0 \le \left\|\nabla f(x_k^{(j)})\right\|^2 \le \frac{1}{\eta_j} M_k^{(j)} = \mathrm{poly}(k).$$

By remark 6, we obtain that

$$\mathbb{E}|f(x_{N_j}^{(j)}) - f(x^\star)| < \infty, \quad \mathbb{E}|D_{N_j}^{(3)}| = \mathbb{E}\left\|\nabla f(x_{N_j}^{(j)})\right\|^2 < \infty.$$

Since $f(x^\star) > -\infty$, $\mathbb{E}|D_{N_j}^{(2)}| < \infty$. ■

C.2. Preparation for Complexity Analysis

Although Theorem 8 and 9 consider the tail-randomized iterate, we start by studying two conventional output – the randomized iterate and the last iterate. Throughout this subsection we let

$$\lambda_j = \eta_j m_j / b_j.$$

The first lemma states a bound for expected gradient norm of the randomized iterate.

Lemma 17 *Given any positive integer T, let $\mathcal{R}$ be a random variable supported on $\{1, \ldots, T\}$ with*

$$\mathbb{P}(\mathcal{R} = j) \propto \lambda_j$$

Then

$$\mathbb{E}\left\|\nabla f(x_{\mathcal{R}})\right\|^2 \le \frac{2\mathbb{E}\left(f(\tilde{x}_0) - f(x^\star)\right) + \sigma^2 \sum_{j=1}^T \lambda_j I(B_j < n)/B_j}{\sum_{j=1}^T \lambda_j}$$

Proof By Theorem 7,

$$\lambda_j \mathbb{E}\left\|\nabla f(\tilde{x}_j)\right\|^2 \leq 2\left(\mathbb{E}f(\tilde{x}_{j-1}) - \mathbb{E}f(\tilde{x}_j)\right) + \frac{\sigma^2 \lambda_j I(B_j < n)}{B_j}.$$

By definition,

$$\begin{aligned}
\mathbb{E}\left\|\nabla f(x_{\mathcal{R}})\right\|^2 &= \frac{\sum_{j=1}^T \mathbb{E}\left\|\nabla f(\tilde{x}_j)\right\|^2 \lambda_j}{\sum_{j=1}^T \lambda_j} \\
&\leq \frac{2\sum_{j=1}^T \left(\mathbb{E}f(\tilde{x}_{j-1}) - \mathbb{E}f(\tilde{x}_j)\right) + \sigma^2 \sum_{j=1}^T \lambda_j I(B_j < n)/B_j}{\sum_{j=1}^T \lambda_j} \\
&= \frac{2\left(\mathbb{E}f(\tilde{x}_0) - \mathbb{E}f(\tilde{x}_T)\right) + \sigma^2 \sum_{j=1}^T \lambda_j I(B_j < n)/B_j}{\sum_{j=1}^T \lambda_j}.
\end{aligned}$$

The proof is then completed by the fact that $f(\tilde{x}_T) \geq f(x^\star)$. ■

The next lemma provides contraction results for expected gradient norm and function value suboptimality of the last iterate.

Lemma 18 *Define the following Lyapunov function*

$$\mathcal{L}_j = \mathbb{E}\left(\lambda_j \left\|\nabla f(\tilde{x}_j)\right\|^2 + 2(f(\tilde{x}_j) - f(x^\star))\right).$$

Then under the assumption 3 with μ possibly being zero,

$$\mathcal{L}_j \leq \frac{1}{\mu\lambda_{j-1} + 1}\mathcal{L}_{j-1} + \frac{\sigma^2 \lambda_j I(B_j < n)}{B_j}, \tag{13}$$

and

$$\mathbb{E}\left(f(\tilde{x}_j) - f(x^\star)\right) \leq \frac{1}{\mu\lambda_j + 1}\mathbb{E}\left(f(\tilde{x}_{j-1}) - f(x^\star)\right) + \frac{\lambda_j}{\mu\lambda_j + 1}\frac{\sigma^2 I(B_j < n)}{2B_j}. \tag{14}$$

Proof When $\mu = 0$, the lemma is a direct consequence of Theorem 7. Assume $\mu > 0$ throughout the rest of the proof. Let

$$\chi_j = \frac{\mu\lambda_j}{\mu\lambda_j + 1}.$$

Then by assumption 3,

$$\begin{aligned}
\mathbb{E}(f(\tilde{x}_j) - f(x^\star)) &= (1 - \chi_j)\mathbb{E}(f(\tilde{x}_j) - f(x^\star)) + \chi_j \mathbb{E}(f(\tilde{x}_j) - f(x^\star)) \\
&\leq (1 - \chi_j)\mathbb{E}(f(\tilde{x}_j) - f(x^\star)) + \frac{\chi_j}{2\mu}\mathbb{E}\left\|\nabla f(\tilde{x}_j)\right\|^2 \\
&= \frac{1}{2(\mu\lambda_j + 1)}\left(\lambda_j \mathbb{E}\left\|\nabla f(\tilde{x}_j)\right\|^2 + 2\mathbb{E}(f(\tilde{x}_j) - f(x^\star))\right) \\
&= \frac{1}{2(\mu\lambda_j + 1)}\mathcal{L}_j.
\end{aligned}$$

By Theorem 7,

$$\begin{aligned}\mathcal{L}_j &\le 2\mathbb{E}(f(\tilde{x}_{j-1}) - f(x^\star)) + \frac{\sigma^2\lambda_j I(B_j < n)}{B_j}\\ &\le \frac{1}{\mu\lambda_{j-1}+1}\mathcal{L}_{j-1} + \frac{\sigma^2\lambda_j I(B_j < n)}{B_j}.\end{aligned}$$

On the other hand, by Theorem 5,

$$2\mu\mathbb{E}(f(\tilde{x}_j) - f(x^\star)) \le \mathbb{E}\|\nabla f(\tilde{x}_j)\|^2 \le \frac{2}{\lambda_j}\mathbb{E}(f(\tilde{x}_{j-1}) - f(\tilde{x}_j)) + \frac{\sigma^2 I(B_j < n)}{B_j}.$$

Rearranging terms concludes the proof. ■

The third lemma shows that $\mathcal{L}_j$ and $\mathbb{E}(f(\tilde{x}_j) - f(x^\star))$ are uniformly bounded.

Lemma 19 *For any $j > 0$,*

$$\mathcal{L}_j \le \Delta_j \overset{def}{=} 2\Delta_f + \left(\sum_{t\le j: B_t < n}\frac{\lambda_t}{B_t}\right)\sigma^2.$$

Proof By (13), since $\mu \ge 0$,

$$\mathcal{L}_j \le \mathcal{L}_{j-1} + \frac{\sigma^2\lambda_j I(B_j < n)}{B_j}.$$

Moreover, by Theorem 7,

$$\mathcal{L}_1 \le 2\mathbb{E}(f(\tilde{x}_0) - f(x^\star)) + \frac{\lambda_1\sigma^2 I(B_1 < n)}{B_1}.$$

Telescoping the above inequalities yields the bound for $\mathcal{L}_j$. ■

The last lemma states refined bounds for $\mathbb{E}\,\|\nabla f(\tilde{x}_j)\|^2$ and $\mathbb{E}(f(\tilde{x}_j) - f(x^\star))$ based on Lemma 18 and Lemma 19.

Lemma 20 *Fix any constant $c \in (0, 1)$. Suppose B_j can be written as*

$$B_j = \lceil \tilde{B}_j \wedge n\rceil,$$

for some strictly increasing sequence $\tilde{B}_j$. Assume that λ_j is non-decreasing and

$$\frac{\tilde{B}_{j-1}\lambda_j}{\tilde{B}_j\lambda_{j-1}} \ge \sqrt{c}.$$

Let

$$T_\mu(c) = \min\{j : \lambda_j > 1/\mu c\}, \quad T_n = \min\{j : \tilde{B}_j \ge n\},$$

where $T_\mu(c) = \infty$ if no such j exists, e.g. for $\mu = 0$. Then for any $j > T_\mu(c)$,

$$\mathbb{E}\,\|\nabla f(\tilde{x}_j)\|^2 \le \min\left\{\left(\prod_{t=T_\mu(c)}^{j-1}\frac{1}{\mu\lambda_t}\right)\frac{\Delta_{T_\mu(c)}}{\lambda_j} + \frac{\sigma^2 I(j > T_\mu(c))}{(1-\sqrt{c})\tilde{B}_j}, \left(\frac{1}{\mu\lambda_{T_n}+1}\right)^{(j-T_n)_+}\frac{\Delta_{T_n}}{\lambda_j}\right\},$$

and

$$\mathbb{E}\left(f(\tilde{x}_j) - f(x^\star)\right) \le \min\left\{\left(\prod_{t=T_\mu(c)+1}^{j}\frac{1}{\mu\lambda_t}\right)\Delta_{T_\mu(c)} + \frac{\sigma^2 I(j > T_\mu(c))}{2(1-\sqrt{c})\mu\tilde{B}_j}, \left(\frac{1}{\mu\lambda_{T_n}+1}\right)^{(j-T_n)_+}\Delta_{T_n}\right\},$$

where $\prod_{t=a}^{b} c_t = 1$ *if* $a > b$.

Proof We first prove the bounds involving $T_\mu(c)$. Assume $T_\mu(c) < \infty$. Then for $j > T_\mu(c)$,

$$\frac{1}{\mu\lambda_j + 1} \le \frac{1}{\mu\lambda_{j-1}+1} < \frac{1}{\mu\lambda_{j-1}} < c. \tag{15}$$

By (13), (14) and the condition that $\lambda_j \ge \lambda_{j-1}$, we have

$$\mathcal{L}_j \le \frac{1}{\mu\lambda_{j-1}}\mathcal{L}_{j-1} + \frac{\sigma^2\lambda_j I(B_j < n)}{B_j} \le \frac{1}{\mu\lambda_{j-1}}\mathcal{L}_{j-1} + \frac{\sigma^2\lambda_j}{\tilde{B}_j},$$

and

$$\begin{aligned}\mathbb{E}\left(f(\tilde{x}_j) - f(x^\star)\right) &\le \frac{1}{\mu\lambda_j}\mathbb{E}\left(f(\tilde{x}_{j-1}) - f(x^\star)\right) + \frac{\sigma^2 I(B_j < n)}{2\mu B_j}\\ &\le \frac{1}{\mu\lambda_j}\mathbb{E}\left(f(\tilde{x}_{j-1}) - f(x^\star)\right) + \frac{\sigma^2}{2\mu\tilde{B}_j}.\end{aligned}$$

Applying the above inequalities recursively and using Lemma 19 and (15), we obtain that

$$\begin{aligned}\lambda_j\mathbb{E}\left\|\nabla f(\tilde{x}_j)\right\|^2 \le \mathcal{L}_j &\le \left(\prod_{t=T_\mu(c)}^{j-1}\frac{1}{\mu\lambda_t}\right)\mathcal{L}_{T_\mu(c)} + \sigma^2\sum_{t=T_\mu(c)+1}^{j}\frac{c^{j-t}\lambda_t}{\tilde{B}_t}\\ &\overset{(i)}{\le} \left(\prod_{t=T_\mu(c)}^{j-1}\frac{1}{\mu\lambda_t}\right)\Delta_{T_\mu(c)} + \sigma^2\sum_{t=T_\mu(c)+1}^{j}\frac{(\sqrt{c})^{j-t}\lambda_j}{\tilde{B}_j}\\ &= \left(\prod_{t=T_\mu(c)}^{j-1}\frac{1}{\mu\lambda_t}\right)\Delta_{T_\mu(c)} + \frac{\sigma^2\lambda_j}{(1-\sqrt{c})\tilde{B}_j}\end{aligned}$$

where (i) uses the condition that $\tilde{B}_{j-1}\lambda_j/\tilde{B}_j\lambda_{j-1} \ge \sqrt{c}$ and thus $B_t \ge B_j(\sqrt{c})^{(j-t)}$. Similarly,

$$\mathbb{E}\left(f(\tilde{x}_j) - f(x^\star)\right) \le \left(\prod_{t=T_\mu(c)+1}^{j}\frac{1}{\mu\lambda_t}\right)\Delta_{T_\mu(c)} + \frac{\sigma^2}{2(1-\sqrt{c})\mu\tilde{B}_j}.$$

Next, we prove the bounds involving T_n. Similar to the previous step, the case with $j \le T_n$ can be easily proved. When $j > T_n$, $B_j = n$ and thus

$$\mathcal{L}_j \le \left(\frac{1}{\mu\lambda_{T_n}+1}\right)\mathcal{L}_{j-1}, \quad \mathbb{E}\left(f(\tilde{x}_j) - f(x^\star)\right) \le \left(\frac{1}{\mu\lambda_{T_n}+1}\right)\mathbb{E}\left(f(\tilde{x}_{j-1}) - f(x^\star)\right).$$

This implies the bounds involving T_n. ■

Combining Lemma 17 and Lemma 20, we obtain the convergence rate of the randomized iterate.

Theorem 21 *Given any positive integer T, let $\mathcal{R}$ be a random variable supported on $\{T, \dots, \lceil (1+\delta)T \rceil\}$ with*

$$\mathbb{P}(\mathcal{R} = j) \propto \lambda_j.$$

Then under the settings of Lemma 20,

$$\mathbb{E}\left\|\nabla f(\tilde{x}_{\mathcal{R}})\right\|^2 \le \min\left\{\left(\prod_{t=T_\mu(c)}^{T-1} \frac{1}{\mu\lambda_t}\right)\frac{\Delta_{T_\mu(c)}}{\lambda_T} + \frac{\sigma^2 I(T > T_\mu(c))}{(1-\sqrt{c})\tilde{B}_T}, \left(\frac{1}{\mu\lambda_{T_n}+1}\right)^{(T-T_n)_+}\frac{\Delta_{T_n}}{\lambda_T}, \right.$$
$$\left.\frac{2\Delta_T + \sigma^2\sum_{j=T}^{\lceil (1+\delta)T\rceil}\lambda_j I(B_j < n)/B_j}{\sum_{j=T}^{\lceil (1+\delta)T\rceil}\lambda_j}\right\},$$

and

$$\mathbb{E}\left(f(\tilde{x}_{\mathcal{R}}) - f(x^\star)\right) \le \min\left\{\left(\prod_{t=T_\mu(c)+1}^{T}\frac{1}{\mu\lambda_t}\right)\Delta_{T_\mu(c)} + \frac{\sigma^2 I(T > T_\mu(c))}{2(1-\sqrt{c})\mu\tilde{B}_T}, \left(\frac{1}{\mu\lambda_{T_n}+1}\right)^{(T-T_n)_+}\Delta_{T_n}\right\},$$

where we set $\prod_{t=a}^{b} c_t = 1$ if $a > b$.

Proof By Lemma 20, for any $j \in [T, \lceil (1+\delta)T\rceil]$,

$$\begin{aligned}\mathbb{E}\left\|\nabla f(\tilde{x}_j)\right\|^2 &\le \min\left\{\left(\prod_{t=T_\mu(c)}^{j-1}\frac{1}{\mu\lambda_t}\right)\frac{\Delta_{T_\mu(c)}}{\lambda_j} + \frac{\sigma^2 I(j > T_\mu(c))}{(1-\sqrt{c})\tilde{B}_j}, \left(\frac{1}{\mu\lambda_{T_n}+1}\right)^{(j-T_n)_+}\frac{\Delta_{T_n}}{\lambda_j}\right\}\\ &\le \min\left\{\left(\prod_{t=T_\mu(c)}^{j-1}\frac{1}{\mu\lambda_t}\right)\frac{\Delta_{T_\mu(c)}}{\lambda_T} + \frac{\sigma^2 I(T > T_\mu(c))}{(1-\sqrt{c})\tilde{B}_j}, \left(\frac{1}{\mu\lambda_{T_n}+1}\right)^{(j-T_n)_+}\frac{\Delta_{T_n}}{\lambda_T}\right\}.\end{aligned}$$

As a result,

$$\begin{aligned}\mathbb{E}\left\|\nabla f(\tilde{x}_{\mathcal{R}})\right\|^2 &= \frac{\sum_{j=T+1}^{\lceil (1+\delta)T\rceil}\lambda_j\mathbb{E}\left\|\nabla f(\tilde{x}_j)\right\|^2}{\sum_{j=T+1}^{\lceil (1+\delta)T\rceil}\lambda_j}\\ &\le \min\left\{\left(\prod_{t=T_\mu(c)}^{T-1}\frac{1}{\mu\lambda_t}\right)\frac{\Delta_{T_\mu(c)}}{\lambda_T} + \frac{\sigma^2 I(T > T_\mu(c))}{(1-\sqrt{c})\tilde{B}_j}, \left(\frac{1}{\mu\lambda_{T_n}+1}\right)^{(T-T_n)_+}\frac{\Delta_{T_n}}{\lambda_T}\right\}.\end{aligned}$$

Similarly we can prove the bound for $\mathbb{E}(f(\tilde{x}_j) - f(x^\star))$. To prove the third bound for $\mathbb{E}\left\|\nabla f(\tilde{x}_{\mathcal{R}})\right\|^2$, we first notice that $\tilde{x}_{\mathcal{R}}$ can be regarded as the randomized iterate with $\tilde{x}_T$ being the initializer. By Lemma 17,

$$\mathbb{E}\left\|\nabla f(\tilde{x}_{\mathcal{R}})\right\|^2 \le \frac{2\mathbb{E}\left(f(\tilde{x}_T) - f(x^\star)\right) + \sigma^2\sum_{j=T+1}^{\lceil (1+\delta)T\rceil}\lambda_j I(B_j < n)/B_j}{\sum_{j=T+1}^{\lceil (1+\delta)T\rceil}\lambda_j}.$$

By Lemma 19,

$$\mathbb{E}\left(f(\tilde{x}_T) - f(x^\star)\right) \le \Delta_T,$$

which concludes the proof. The bound for $\mathbb{E}(f(\tilde{x}_{\mathcal{R}}) - f(x^\star))$ can be proved similarly. ■

C.3. Complexity Analysis: Proof of Theorem 10

Under this setting,

$$2\lambda_j L = \frac{2\eta_j m_j}{b_j} = \sqrt{m_j} = jI(j < \sqrt{n}) + \sqrt{n}I(j \geq \sqrt{n}).$$

Let $c = 1/8$. It is easy to verify that $\tilde{B}_{j-1}\lambda_j/\tilde{B}_j\lambda_{j-1} \geq 1/2 > \sqrt{c}$. Moreover, by Lemma 24,

$$L \sum_{t\leq j: B_t < n} \frac{\lambda_t}{B_t} = \sum_{t < \sqrt{n}\wedge j} \frac{1}{t} \leq 1 + \log(\sqrt{n} \wedge j).$$

Recalling the definition of Δ in Lemma 19,

$$\Delta_j \leq 2\Delta_f + \left(\sum_{t\leq j: B_t < n} \frac{\lambda_t}{B_t}\right)\sigma^2 \leq 2\Delta_f + \frac{\sigma^2}{L} + \frac{2\sigma^2}{L}\log(n \wedge j). \tag{16}$$

Now we treat each of the three terms in the bound of $\mathbb{E}\|\nabla f(\tilde{x}_j)\|^2$ in Theorem 21 separately.

(**First term.**) Write T_μ for $T_\mu(c) = T_\mu(1/8)$. By definition,

$$T_\mu = \min\left\{j : \lambda_j \geq \frac{1}{\mu}\right\} = \left\{\begin{array}{ll} \lceil 2L/\mu \rceil & (\lceil 2L/\mu \rceil \leq \sqrt{n}) \\ \infty & (\text{otherwise}) \end{array}\right.$$

Let

$$T_{g1}(\epsilon) = T_\mu + \frac{\log(2\mu\Delta_{T_\mu}/\epsilon^2)}{\log 8} + \frac{2\sigma}{\epsilon}.$$

When $T_{g1}(\epsilon) = \infty$, it is obvious that $T_g(\epsilon) \leq T_{g1}(\epsilon)$. When $T_{g1}(\epsilon) < \infty$, for any $T \geq T_{g1}(\epsilon)$,

$$\left(\prod_{t=T_\mu}^{T-1} \frac{1}{\mu\lambda_t}\right) \frac{\Delta_{T_\mu}}{\lambda_T} \leq c^{T-T_\mu} \frac{\Delta_{T_\mu}}{\lambda_T} \leq \left(\frac{1}{8}\right)^{\frac{\log(2\mu\Delta_{T_\mu}/\epsilon^2)}{\log 8}} \frac{\Delta_{T_\mu}}{\lambda_{T_\mu}} = \frac{\epsilon^2}{2\mu\lambda_{T_\mu}} \leq \frac{\epsilon^2}{2}.$$

Note that $\tilde{B}_j = j^2$ in this case,

$$\frac{\sigma^2}{(1-\sqrt{c})\tilde{B}_T} \leq \frac{2\sigma^2}{T^2} \leq \frac{\epsilon^2}{2}.$$

Recalling the definition (6) of $T_g(\epsilon)$, we obtain that

$$T_g(\epsilon) \leq T_{g1}(\epsilon).$$

(**Second term.**) By definition,

$$T_n = \min\{j : B_j = n\} = \lceil \sqrt{n} \rceil, \quad \lambda_{T_n} = T_n/2L.$$

Let

$$T_{g2}(\epsilon) = T_n + \left(1 + \frac{2L}{\mu\sqrt{n}}\right)\log\left(\frac{2L\Delta_{T_n}}{\sqrt{n}\epsilon^2}\right).$$

By Lemma 24,

$$T_{g2}(\epsilon) - T_n \geq \frac{\log(2L\Delta_{T_n}/\sqrt{n}\epsilon^2)}{\log(1+\mu\sqrt{n}/2L)}.$$

When $T \geq T_{g2}(\epsilon)$,

$$\left(\frac{1}{\mu\lambda_{T_n}+1}\right)^{(T-T_n)_+}\frac{\Delta_{T_n}}{\lambda_T} \leq \frac{\sqrt{n}\epsilon^2}{2L\Delta_{T_n}}\frac{\Delta_{T_n}}{\lambda_{T_n}} \leq \epsilon^2.$$

Therefore, we have

$$T_g(\epsilon) \leq T_{g2}(\epsilon).$$

(**Third term.**) Note that

$$2L\sum_{j=T}^{2T}\lambda_j I(B_j < n)/B_j = \sum_{j=T}^{2T} I(j < \sqrt{n})/j \leq \sum_{j=T}^{2T} I(j<\sqrt{n})/T \leq \frac{T+1}{T} \leq 2.$$

and

$$2L\sum_{j=T}^{2T}\lambda_j = \sum_{j=T}^{2T}\left(jI(j<\sqrt{n}) + \sqrt{n}I(j\geq\sqrt{n})\right) \geq \sum_{j=T}^{2T}(T\wedge\sqrt{n}) \geq T^2\wedge\sqrt{n}T.$$

Let

$$\tilde{\Delta} = L\Delta_f + \sigma^2.$$

By Theorem 21 and (16),

$$\mathbb{E}\left\|\nabla f(\tilde{x}_j)\right\|^2 \leq 8\left(\frac{\tilde{\Delta}}{T^2\wedge\sqrt{n}T} + \frac{\sigma^2\log(T\wedge\sqrt{n})}{T^2\wedge\sqrt{n}T}\right).$$

Let

$$T_{g3}(\epsilon) = \frac{4\sqrt{\tilde{\Delta}}}{\epsilon} + \frac{16\tilde{\Delta}}{\sqrt{n}\epsilon^2}, \quad T_{g4}(\epsilon) = 2 + \frac{4\sigma}{\epsilon}\sqrt{2\log\left(\frac{4\sigma}{\epsilon}\vee 1\right)} + \frac{16\sigma^2\log\sqrt{n}}{\sqrt{n}\epsilon^2}.$$

If $T \geq T_{g3}(\epsilon)$,

$$\frac{\tilde{\Delta}}{T^2\wedge\sqrt{n}T} \leq \max\left\{\frac{\tilde{\Delta}}{T^2}, \frac{\tilde{\Delta}}{\sqrt{n}T}\right\} \leq \frac{\epsilon^2}{16}.$$

If $T \geq T_{g4}(\epsilon)$, by Lemma 25 with $a = \sqrt{n}$ and $x = \epsilon/4\sigma$,

$$\frac{\log(T\wedge\sqrt{n})}{T^2\wedge\sqrt{n}T} \leq \frac{\epsilon^2}{16\sigma^2}.$$

Therefore,

$$T_g(\epsilon) \leq T_{g3}(\epsilon)\vee T_{g4}(\epsilon).$$

Putting three pieces together, we conclude that

$$T_g(\epsilon) \leq T_{g1}(\epsilon)\wedge T_{g2}(\epsilon)\wedge(T_{g3}(\epsilon)\vee T_{g4}(\epsilon)).$$

In this case, the expected computational complexity is

$$\begin{aligned}\mathbb{E}\mathrm{Comp}_g(\epsilon) = \sum_{j=1}^{2T_g(\epsilon)} (2m_j + B_j) = 3\sum_{j=1}^{2T_g(\epsilon)} (j^2 \wedge n) \\ \leq 3\min\left\{\sum_{j=1}^{2T_g(\epsilon)} j^2, nT_g(\epsilon)\right\} = \mathcal{O}\left(T_g^3(\epsilon) \wedge nT_g(\epsilon)\right).\end{aligned}$$

Dealing with $T_{g1}(\epsilon)$ and $T_{g2}(\epsilon)$. First we prove that

$$\begin{aligned}&\left(T_{g1}^3(\epsilon) \wedge nT_{g1}(\epsilon)\right) \wedge \left(T_{g2}^3(\epsilon) \wedge nT_{g2}(\epsilon)\right) \\ =&\mathcal{O}\left(\left\{\frac{L^3}{\mu^3} + \frac{\sigma^3}{\epsilon^3} + \log^3\left(\frac{\mu\Delta}{\epsilon^2}\right)\right\} \wedge \left\{n^{3/2} + \left(n + \frac{\sqrt{n}L}{\mu}\right)\log\left(\frac{L\Delta}{\sqrt{n}\epsilon^2}\right)\right\}\right).\end{aligned} \tag{17}$$

We distinguish two cases.

- If $T_\mu \leq T_n$, since $T_{g2}(\epsilon) > T_n$ and $T_n^3 \leq nT_n$then

  $$\left(T_{g1}^3(\epsilon) \wedge nT_{g1}(\epsilon)\right) \wedge \left(T_{g2}^3(\epsilon) \wedge nT_{g2}(\epsilon)\right) = T_{g1}^3(\epsilon),$$

  which proves (17).

- If $T_\mu > T_n$, then

  $$\left(T_{g1}^3(\epsilon) \wedge nT_{g1}(\epsilon)\right) \wedge \left(T_{g2}^3(\epsilon) \wedge nT_{g2}(\epsilon)\right) \leq nT_{g2}(\epsilon).$$

  It is left to prove

  $$n^{3/2} + \left(n + \frac{\sqrt{n}L}{\mu}\right)\log\left(\frac{L\Delta}{\sqrt{n}\epsilon^2}\right) = \mathcal{O}\left(\frac{L^3}{\mu^3} + \frac{\sigma^3}{\epsilon^3} + \log^3\left(\frac{\mu\Delta}{\epsilon^2}\right)\right).$$

  Since $T_\mu > \sqrt{n}/2$, we have $\sqrt{n} = \mathcal{O}\left(\frac{L}{\mu}\right)$. This entails that

  $$n^{3/2} = \mathcal{O}\left(\frac{L^3}{\mu^3}\right), \quad \text{and} \quad n + \frac{\sqrt{n}L}{\mu} = \mathcal{O}\left(\frac{L^2}{\mu^2}\right).$$

  As a result,

  $$\begin{aligned}&\left(n + \frac{\sqrt{n}L}{\mu}\right)\log\left(\frac{L\Delta}{\sqrt{n}\epsilon^2}\right) = \mathcal{O}\left(\frac{L^2}{\mu^2}\left\{\log\left(\frac{\mu\Delta}{\epsilon^2}\right) + \log\left(\frac{L}{\sqrt{n}\mu}\right)\right\}\right) \\ =&\mathcal{O}\left(\frac{L^2}{\mu^2}\left\{\log\left(\frac{\mu\Delta}{\epsilon^2}\right) + \log\left(\frac{L}{\mu}\right)\right\}\right).\end{aligned}$$

  (17) is then proved by the fact that

  $$\frac{L^2}{\mu^2}\log\left(\frac{\mu\Delta}{\epsilon^2}\right) \leq \frac{L^3}{\mu^3} + \log^3\left(\frac{\mu\Delta}{\epsilon^2}\right), \quad \frac{L^2}{\mu^2}\log\left(\frac{L}{\mu}\right) = \mathcal{O}\left(\frac{L^3}{\mu^3}\right).$$

Dealing with $T_{g3}(\epsilon)$. We prove that

$$T_{g3}^3(\epsilon) \wedge nT_{g3}(\epsilon) = \mathcal{O}\left(\frac{\tilde{\Delta}^{3/2}}{\epsilon^3} \wedge \frac{\sqrt{n}\tilde{\Delta}}{\epsilon^2}\right). \tag{18}$$

We distinguish two cases.

- If $\tilde{\Delta} \leq n\epsilon^2$, then
$$\frac{\tilde{\Delta}}{\sqrt{n}\epsilon^2} \leq \frac{\sqrt{\tilde{\Delta}}}{\epsilon} \leq \sqrt{n}, \quad \text{and} \quad \frac{\tilde{\Delta}^{3/2}}{\epsilon^3} \leq \frac{\sqrt{n}\tilde{\Delta}}{\epsilon^2}.$$
As a result,
$$T_{g3}(\epsilon) = \mathcal{O}\left(\frac{\sqrt{\tilde{\Delta}}}{\epsilon}\right).$$
Thus,
$$T_{g3}^3(\epsilon) \wedge nT_{g3}(\epsilon) = \mathcal{O}\left(T_{g3}^3(\epsilon)\right) = \mathcal{O}\left(\frac{\tilde{\Delta}^{3/2}}{\epsilon^3}\right) = \mathcal{O}\left(\frac{\tilde{\Delta}^{3/2}}{\epsilon^3} \wedge \frac{\sqrt{n}\tilde{\Delta}}{\epsilon^2}\right).$$

- If $\tilde{\Delta} \geq n\epsilon^2$, then
$$\frac{\tilde{\Delta}}{\sqrt{n}\epsilon^2} \geq \frac{\sqrt{\tilde{\Delta}}}{\epsilon} \geq \sqrt{n}, \quad \text{and} \quad \frac{\tilde{\Delta}^{3/2}}{\epsilon^3} \geq \frac{\sqrt{n}\tilde{\Delta}}{\epsilon^2}.$$
As a result,
$$T_{g3}(\epsilon) = \mathcal{O}\left(\frac{\tilde{\Delta}}{\sqrt{n}\epsilon^2}\right).$$
Therefore,
$$T_{g3}^3(\epsilon) \wedge nT_{g3}(\epsilon) = \mathcal{O}\left(nT_{g3}(\epsilon)\right) = \mathcal{O}\left(\frac{\sqrt{n}\tilde{\Delta}}{\epsilon^2}\right) = \mathcal{O}\left(\frac{\tilde{\Delta}^{3/2}}{\epsilon^3} \wedge \frac{\sqrt{n}\tilde{\Delta}}{\epsilon^2}\right).$$

(18) is then proved by putting two pieces together..

Dealing with $T_{g4}(\epsilon)$. Note that

$$T_{g4}(\epsilon) = \mathcal{O}\left(\frac{\sigma}{\epsilon}\sqrt{\log\left(\frac{\sigma}{\epsilon}\right)} + \frac{\sigma^2 \log\sqrt{n}}{\sqrt{n}\epsilon^2}\right) = \mathcal{O}\left(\frac{\sigma}{\epsilon}\log\left(\frac{\sigma}{\epsilon}\right) + \frac{\sigma^2 \log\sqrt{n}}{\sqrt{n}\epsilon^2}\right).$$

We prove that

$$T_{g4}^3(\epsilon) \wedge nT_{g4}(\epsilon) = \mathcal{O}\left(\left\{\frac{\sigma^3}{\epsilon^3} \wedge \frac{\sqrt{n}\sigma^2}{\epsilon^2}\right\}\log^3\left(\frac{\sigma}{\epsilon} \wedge \sqrt{n}\right)\right). \tag{19}$$

We distinguish two cases.

- If $\sigma^2 \leq n\epsilon^2$, since the mapping $y \mapsto \log y/y$ is decreasing on $[3,\infty)$ (see the proof of Lemma 25),

$$\frac{\sigma/\epsilon}{\log(\sigma/\epsilon)} = \mathcal{O}\left(\frac{\sqrt{n}}{\log\sqrt{n}}\right) \Longrightarrow \frac{\sigma^2\log\sqrt{n}}{\sqrt{n}\epsilon^2} = \mathcal{O}\left(\frac{\sigma}{\epsilon}\log\left(\frac{\sigma}{\epsilon}\right)\right).$$

  As a result,

$$T_{g4}(\epsilon) = \mathcal{O}\left(\frac{\sigma}{\epsilon}\log\left(\frac{\sigma}{\epsilon}\right)\right) = \mathcal{O}\left(\frac{\sigma}{\epsilon}\log\left(\frac{\sigma}{\epsilon}\wedge\sqrt{n}\right)\right).$$

  Thus,

$$T_{g4}^3(\epsilon)\wedge nT_{g4}(\epsilon) = \mathcal{O}\left(T_{g4}^3(\epsilon)\right) = \mathcal{O}\left(\frac{\sigma^3}{\epsilon^3}\log^3\left(\frac{\sigma}{\epsilon}\wedge\sqrt{n}\right)\right) = \mathcal{O}\left(\left\{\frac{\sigma^3}{\epsilon^3}\wedge\frac{\sqrt{n}\sigma^2}{\epsilon^2}\right\}\log^3\left(\frac{\sigma}{\epsilon}\wedge\sqrt{n}\right)\right).$$

- If $\sigma^2 \geq n\epsilon^2$, similar to the above case,

$$\frac{\sqrt{n}}{\log\sqrt{n}} = \mathcal{O}\left(\frac{\sigma/\epsilon}{\log(\sigma/\epsilon)}\right) \Longrightarrow \frac{\sigma}{\epsilon}\log\left(\frac{\sigma}{\epsilon}\right) = \mathcal{O}\left(\frac{\sigma^2\log\sqrt{n}}{\sqrt{n}\epsilon^2}\right).$$

  As a result,

$$T_{g4}(\epsilon) = \mathcal{O}\left(\frac{\sigma^2\log\sqrt{n}}{\sqrt{n}\epsilon^2}\right) = \mathcal{O}\left(\frac{\sigma^2}{\sqrt{n}\epsilon^2}\log\left(\frac{\sigma}{\epsilon}\wedge\sqrt{n}\right)\right).$$

  Thus,

$$T_{g4}^3(\epsilon)\wedge nT_{g4}(\epsilon) = \mathcal{O}\left(nT_{g4}(\epsilon)\right) = \mathcal{O}\left(\frac{\sqrt{n}\sigma^2}{\epsilon^2}\log\left(\frac{\sigma}{\epsilon}\wedge\sqrt{n}\right)\right) = \mathcal{O}\left(\left\{\frac{\sigma^3}{\epsilon^3}\wedge\frac{\sqrt{n}\sigma^2}{\epsilon^2}\right\}\log^3\left(\frac{\sigma}{\epsilon}\wedge\sqrt{n}\right)\right).$$

(19) is then proved by putting two pieces together..

Dealing with $T_{g3}(\epsilon)\vee T_{g4}(\epsilon)$. Note that

$$(T_{g3}(\epsilon)\vee T_{g4}(\epsilon))^3\wedge n(T_{g3}(\epsilon)\vee T_{g4}(\epsilon)) = \mathcal{O}\left(T_{g3}^3(\epsilon)\wedge nT_{g3}(\epsilon) + T_{g4}^3(\epsilon)\wedge nT_{g4}(\epsilon)\right).$$

Using the fact that $a\wedge b + c\wedge d \leq (a+c)\wedge(b+d)$ and by (18) and (19), we have

$$\begin{aligned}&T_{g3}^3(\epsilon)\wedge nT_{g3}(\epsilon) + T_{g4}^3(\epsilon)\wedge nT_{g4}(\epsilon)\\ =&\mathcal{O}\left(\left\{\frac{\tilde{\Delta}^{3/2}}{\epsilon^3} + \frac{\sigma^3}{\epsilon^3}\log^3\left(\frac{\sigma}{\epsilon}\right)\right\}\wedge\left\{\frac{\sqrt{n}\tilde{\Delta}}{\epsilon^2} + \frac{\sqrt{n}\sigma^2}{\epsilon^2}\log^3 n\right\}\right).\end{aligned}$$

Summary Putting (17) and (18) together and using the fact that $\tilde{\Delta} = \mathcal{O}(L\Delta)$, we prove the bound for $\mathbb{E}\mathrm{Comp}_g(\epsilon)$. As for $\mathbb{E}\mathrm{Comp}_f(\epsilon)$, by Theorem 21, we can directly apply (17) by replacing Δ/λ_T by Δ and σ^2 with σ^2/μ.

C.4. Complexity Analysis: Proof of Theorem 12

Under this setting,

$$2\lambda_j L = \frac{2\eta_j m_j}{b_j} = \sqrt{m_j} = \alpha^j I(j < \log_\alpha n) + \sqrt{n}I(j \geq \log_\alpha n).$$

Let $c = 1/4\alpha^4$. Then

$$\frac{\tilde{B}_{j-1}\lambda_j}{\tilde{B}_j\lambda_{j-1}} \geq \frac{1}{\alpha^2} > \sqrt{c}.$$

On the other hand,

$$L\sum_{t:B_t<n}\frac{\lambda_t}{B_t} = \frac{1}{2}\sum_{t<\sqrt{n}}\alpha^{-t} \leq \frac{1}{2(1-\alpha^{-1})} = \frac{\alpha}{2(\alpha-1)}.$$

Recalling the definition of Δ_j in Lemma 19,

$$\Delta_j \leq 2\Delta_f + \left(\sum_{t:B_t<n}\frac{\lambda_t}{B_t}\right)\sigma^2 \leq 2\Delta_f + \frac{\alpha}{2(\alpha-1)}\frac{\sigma^2}{L} \triangleq \Delta'. \tag{20}$$

As in the proof of Theorem 8, we treat each of the three terms in the bound of $\mathbb{E}\left\|\nabla f(\tilde{x}_j)\right\|^2$ in Theorem 21 separately.

(**First term.**) Write T_μ for $T_\mu(c) = T_\mu(1/4\alpha^4)$. By definition,

$$T_\mu = \min\left\{j : \lambda_j > \frac{4\alpha^2}{\mu}\right\} = \left\{\begin{array}{ll} \lceil\log_\alpha(8L/\mu)\rceil + 4 & \left(\lceil 8L\alpha^4/\mu\rceil \leq \sqrt{n}\right) \\ \infty & \text{(otherwise)} \end{array}\right. ,$$

and

$$T_n = \min\{j : B_j = n\} = \lceil(\log_\alpha n)/2\rceil.$$

Let

$$A(\epsilon) = \max\left\{T_\mu + \sqrt{2\log_\alpha\left(\frac{2\mu\Delta'}{\epsilon^2}\right)}, \log_\alpha\left(\frac{2\sigma}{\epsilon}\right)\right\},$$

and

$$T_{g1}(\epsilon) = A(\epsilon)I(A(\epsilon) \leq T_n) + \infty I(A(\epsilon) > T_n).$$

When $T_{g1}(\epsilon) = \infty$, it is obvious that $T_g(\epsilon) \leq T_{g1}(\epsilon) = \infty$. When $T_{g1}(\epsilon) < \infty$, i.e. $T_\mu \leq A(\epsilon) \leq T_n$, for any $T \geq T_{g1}(\epsilon)$,

$$\left(\prod_{t=T_\mu}^{T-1}\frac{1}{\mu\lambda_t}\right)\frac{\Delta'}{\lambda_T} = \left(\prod_{t=T_\mu}^{T}\frac{1}{\mu\lambda_t}\right)(\mu\Delta') \leq \exp\left\{-\sum_{t=T_\mu}^{T\wedge A(\epsilon)}\log(\mu\lambda_t)\right\}(\mu\Delta').$$

For any $t \in [T_\mu, A(\epsilon)]$, since $t \leq T_n$, we have $\lambda_t = \alpha^t$ and thus

$$\log(\mu\lambda_t) = \log(\mu\lambda_{T_\mu}) + (\log\alpha)(t - T_\mu) \geq (\log\alpha)(t - T_\mu).$$

Then

$$\sum_{t=T_\mu}^{T\wedge A(\epsilon)}\log(\mu\lambda_t) \geq \frac{\log\alpha}{2}(\lfloor A(\epsilon)\rfloor - T_\mu)^2 \geq \log\left(\frac{2\mu\Delta'}{\epsilon^2}\right).$$

This implies that

$$\left(\prod_{t=T_\mu}^{T-1}\frac{1}{\mu\lambda_t}\right)\frac{\Delta'}{\lambda_T} \leq \frac{\epsilon^2}{2}.$$

On the other hand, note that $\tilde{B}_j = \alpha^{2j}$ in this case, when $T > T_{g1}(\epsilon)$,

$$T_{g1}(\epsilon) \geq \frac{\log(2\sigma/\epsilon)}{\log \alpha} \Longrightarrow \frac{\sigma^2}{(1-\sqrt{c})\tilde{B}_T} \leq \frac{\epsilon^2}{4(1-\sqrt{c})} \leq \frac{\epsilon^2}{2},$$

where the last inequality follows from the fact that

$$4(1-\sqrt{c}) = 4\left(1 - \frac{1}{2\alpha^2}\right) \geq 2.$$

Putting pieces together we have

$$T_g(\epsilon) \leq T_{g1}(\epsilon).$$

(**Second term.**) Let

$$T_{g2}(\epsilon) = T_n + \left(1 + \frac{2L}{\mu\sqrt{n}}\right)\log\left(\frac{2L\Delta'}{\sqrt{n}\epsilon^2}\right).$$

Using the same argument as in the proof of Theorem 10,

$$T_g(\epsilon) \leq T_{g2}(\epsilon).$$

(**Third term.**) Note that

$$2L\sum_{j=T}^{\lceil(1+\delta)T\rceil} \lambda_j I(B_j < n)/B_j = \sum_{j=T}^{\lceil(1+\delta)T\rceil} I(j < \sqrt{n})\alpha^{-j} \leq \sum_{j=1}^{\infty}\alpha^{-j} = \frac{1}{\alpha - 1}.$$

and

$$\begin{aligned} 2L\sum_{j=T}^{\lceil(1+\delta)T\rceil} \lambda_j &= \sum_{j=T}^{\lceil(1+\delta)T\rceil} \left(\alpha^j I(j < T_n) + \sqrt{n} I(j \geq T_n)\right) \\ &\geq \sum_{j=T+1}^{\lceil(1+\delta)T\rceil} (\alpha^j \wedge \sqrt{n}) \geq \alpha^{\lceil(1+\delta)T\rceil} \wedge \delta\sqrt{n}T. \end{aligned}$$

Let

$$\tilde{\Delta} = 2L\Delta' + \sigma^2/(\alpha - 1).$$

By Theorem 21,

$$\mathbb{E}\left\|\nabla f(\tilde{x}_j)\right\|^2 \leq \frac{2L\Delta' + \sigma^2/(\alpha-1)}{\alpha^{\lceil(1+\delta)T\rceil} \wedge \delta\sqrt{n}T} = \frac{\tilde{\Delta}}{\alpha^{\lceil(1+\delta)T\rceil} \wedge \delta\sqrt{n}T}.$$

Let

$$T_{g3}(\epsilon) = \max\left\{\frac{1}{1+\delta}\log_\alpha\left(\frac{\tilde{\Delta}}{\epsilon^2}\right), \frac{\tilde{\Delta}}{\delta\sqrt{n}\epsilon^2}\right\}.$$

Then

$$T_g(\epsilon) \leq T_{g3}(\epsilon).$$

Putting three pieces together, we conclude that

$$T_g(\epsilon) \leq T_{g1}(\epsilon) \wedge T_{g2}(\epsilon) \wedge T_{g3}(\epsilon).$$

In this case, the expected computational complexity is

$$\begin{aligned}\mathbb{E}\text{Comp}_g(\epsilon) &= \sum_{j=1}^{\lceil (1+\delta)T_g(\epsilon)\rceil} (2m_j + B_j) = 3 \sum_{j=1}^{\lceil (1+\delta)T_g(\epsilon)\rceil} (\alpha^{2j} \wedge n) \\ &\leq 3 \min\left\{ \sum_{j=1}^{\lceil (1+\delta)T_g(\epsilon)\rceil} \alpha^{2j}, nT_g(\epsilon) \right\} = \mathcal{O}\left(\alpha^{2(1+\delta)T_g(\epsilon)} \wedge nT_g(\epsilon) \right).\end{aligned}$$

Dealing with $T_{g1}(\epsilon)$ and $T_{g2}(\epsilon)$. First we prove that

$$\begin{aligned}&\left(\alpha^{2(1+\delta)T_{g1}(\epsilon)} \wedge nT_{g1}(\epsilon)\right) \wedge \left(\alpha^{2(1+\delta)T_{g2}(\epsilon)} \wedge nT_{g2}(\epsilon)\right) \\ =&\mathcal{O}\left(\left\{ \frac{L^{2(1+\delta)}}{\mu^{2(1+\delta)}}\text{es}\left(2\log_\alpha\left\{\frac{\mu\Delta'}{\epsilon^2}\right\}\right) + \frac{\sigma^{2(1+\delta)}}{\epsilon^{2(1+\delta)}} \right\} \log^2 n \wedge \left\{ n\log\left(\frac{L}{\mu}\right) + \left(n + \frac{\sqrt{n}L}{\mu}\right)\log\left(\frac{L\Delta'}{\sqrt{n}\epsilon^2}\right) \right\} \right).\end{aligned} \tag{21}$$

We distinguish two cases.

- If $T_{g1}(\epsilon) \leq T_n/(1+\delta)$, since $T_{g2}(\epsilon) > T_n$,

  $$\begin{aligned}&\left(\alpha^{2(1+\delta)T_{g1}(\epsilon)} \wedge nT_{g1}(\epsilon)\right) \wedge \left(\alpha^{2(1+\delta)T_{g2}(\epsilon)} \wedge nT_{g2}(\epsilon)\right) = \alpha^{2(1+\delta)T_{g1}(\epsilon)} \\ &= \mathcal{O}\left(\frac{L^{2(1+\delta)}}{\mu^{2(1+\delta)}}\text{es}\left(2\log_\alpha\left\{\frac{\mu\Delta'}{\epsilon^2}\right\}\right) + \frac{\sigma^{2(1+\delta)}}{\epsilon^{2(1+\delta)}} \right) \\ &= \mathcal{O}\left(\left\{ \frac{L^{2(1+\delta)}}{\mu^{2(1+\delta)}}\text{es}\left(2\log_\alpha\left\{\frac{\mu\Delta'}{\epsilon^2}\right\}\right) + \frac{\sigma^{2(1+\delta)}}{\epsilon^{2(1+\delta)}} \right\} \log^2 n \right),\end{aligned}$$

  which proves (21).

- If $T_{g1}(\epsilon) > T_n/(1+\delta)$, then

  $$\begin{aligned}&\left(\alpha^{2(1+\delta)T_{g1}(\epsilon)} \wedge nT_{g1}(\epsilon)\right) \wedge \left(\alpha^{2(1+\delta)T_{g2}(\epsilon)} \wedge nT_{g2}(\epsilon)\right) \leq nT_{g2}(\epsilon) \\ &= \mathcal{O}\left(n\log\left(\frac{L}{\mu}\right) + \left(n + \frac{\sqrt{n}L}{\mu}\right)\log\left(\frac{L\Delta'}{\sqrt{n}\epsilon^2}\right) \right).\end{aligned}$$

  It is left to prove that

  $$\begin{aligned}&n\log\left(\frac{L}{\mu}\right) + \left(n + \frac{\sqrt{n}L}{\mu}\right)\log\left(\frac{L\Delta'}{\sqrt{n}\epsilon^2}\right) \\ =&\mathcal{O}\left(\left\{ \frac{L^{2(1+\delta)}}{\mu^{2(1+\delta)}}\text{es}\left(2\log_\alpha\left\{\frac{\mu\Delta'}{\epsilon^2}\right\}\right) + \frac{\sigma^{2(1+\delta)}}{\epsilon^{2(1+\delta)}} \right\} \log^2 n \right).\end{aligned} \tag{22}$$

  We consider the following two cases.

- If $L/\mu > \sqrt{n}$,

  $$\begin{aligned} &n \log\left(\frac{L}{\mu}\right) + \left(n + \frac{\sqrt{n}L}{\mu}\right)\log\left(\frac{L\Delta'}{\sqrt{n}\epsilon^2}\right) \leq \frac{\sqrt{n}L}{\mu}\log\left(\frac{L}{\mu}\right) + \frac{2\sqrt{n}L}{\mu}\log\left(\frac{L\Delta'}{\sqrt{n}\epsilon^2}\right) \\ &= \mathcal{O}\left(\frac{\sqrt{n}L}{\mu}\log\left\{\frac{L^2\Delta'}{\mu\sqrt{n}\epsilon^2}\right\}\right) = \mathcal{O}\left(\frac{\sqrt{n}L}{\mu}\log\left(\frac{\mu\Delta'}{\epsilon^2}\right) + \frac{\sqrt{n}L}{\mu}\log\left(\frac{1}{\sqrt{n L^2}\mu^2}\right)\right) \end{aligned}$$

  The first term can be bounded by

  $$\frac{\sqrt{n}L}{\mu}\log\left(\frac{\mu\Delta'}{\epsilon^2}\right) \leq \frac{L^2}{\mu^2}\log\left(\frac{\mu\Delta'}{\epsilon^2}\right) = \mathcal{O}\left(\frac{L^{2(1+\delta)}}{\mu^{2(1+\delta)}}\text{es}\left(2\log_\alpha\left\{\frac{\mu\Delta'}{\epsilon^2}\right\}\right)\log^2 n\right).$$

  To bound the second term, we consider two cases.

  ∗ If $L/\mu > n$,

  $$\begin{aligned} \frac{\sqrt{n}L}{\mu}\log\left(\frac{L^2}{\sqrt{n}\mu^2}\right) &\leq \frac{2L^{3/2}}{\mu^{3/2}}\log\left(\frac{L}{\mu}\right) = \mathcal{O}\left(\frac{L^2}{\mu^2}\right) \\ &= \mathcal{O}\left(\frac{L^{2(1+\delta)}}{\mu^{2(1+\delta)}}\text{es}\left(2\log_\alpha\left\{\frac{\mu\Delta'}{\epsilon^2}\right\}\right)\log^2 n\right). \end{aligned}$$

  ∗ If $\sqrt{n} < L/\mu < n$,

  $$\frac{\sqrt{n}L}{\mu}\log\left(\frac{L^2}{\sqrt{n}\mu^2}\right) \leq \frac{2L\sqrt{n}\log n}{\mu} \leq \frac{2L^2\log n}{\mu^2} = \mathcal{O}\left(\frac{L^{2(1+\delta)}}{\mu^{2(1+\delta)}}\text{es}\left(2\log_\alpha\left\{\frac{\mu\Delta'}{\epsilon^2}\right\}\right)\log^2 n\right).$$

  (22) is then proved by putting pieces together.

- If $L/\mu \leq \sqrt{n}$.

  $$\begin{aligned} &n\log\left(\frac{L}{\mu}\right) + \left(n + \frac{\sqrt{n}L}{\mu}\right)\log\left(\frac{L\Delta'}{\sqrt{n}\epsilon^2}\right) \leq n\log n + 2n\log\left(\frac{L\Delta'}{\sqrt{n}\epsilon^2}\right) \\ &= n\log n + 2n\log\left(\frac{L}{\mu\sqrt{n}}\right) + 2n\log\left(\frac{\mu\Delta'}{\epsilon^2}\right) = \mathcal{O}\left(n\log n + n\log_\alpha\left(\frac{\mu\Delta'}{\epsilon^2}\right)\right) \end{aligned}$$

  Since $T_{g1}(\epsilon) > T_n/(1+\delta)$,

  $$n \leq \alpha^{2(1+\delta)T_{g1}(\epsilon)} = \mathcal{O}\left(\frac{L^{2(1+\delta)}}{\mu^{2(1+\delta)}}\text{es}\left(2\log_\alpha\left\{\frac{\mu\Delta'}{\epsilon^2}\right\}\right) + \frac{\sigma^{2(1+\delta)}}{\epsilon^{2(1+\delta)}}\right). \tag{23}$$

  It is left to prove that

  $$\frac{n}{\log^2 n}\log_\alpha\left(\frac{\mu\Delta'}{\epsilon^2}\right) = \mathcal{O}\left(\frac{L^{2(1+\delta)}}{\mu^{2(1+\delta)}}\text{es}\left(2\log_\alpha\left\{\frac{\mu\Delta'}{\epsilon^2}\right\}\right) + \frac{\sigma^{2(1+\delta)}}{\epsilon^{2(1+\delta)}}\right). \tag{24}$$

  We distinguish two cases.

  ∗ If $\log_\alpha(\mu\Delta'/\epsilon^2) \leq 2\log^2 n$, (24) is proved by (23).

- If $\log_\alpha(\mu\Delta'/\epsilon^2) > 2\log^2 n$,

$$\text{es}\left(2\log_\alpha\left\{\frac{\mu\Delta'}{\epsilon^2}\right\}\right) \geq \text{es}\left(\log_\alpha\left\{\frac{\mu\Delta'}{\epsilon^2}\right\}/2\right)^2 \geq n\cdot\text{es}\left(\log_\alpha\left\{\frac{\mu\Delta'}{\epsilon^2}\right\}/2\right).$$

Note that

$$\log_\alpha\left(\frac{\mu\Delta'}{\epsilon^2}\right) = \mathcal{O}\left(\text{es}\left(\log_\alpha\left\{\frac{\mu\Delta'}{\epsilon^2}\right\}/2\right)\right).$$

Therefore,

$$\frac{n}{\log^2 n}\log_\alpha\left\{\frac{\mu\Delta'}{\epsilon^2}\right\} = \mathcal{O}\left(\text{es}\left(2\log_\alpha\left\{\frac{\mu\Delta'}{\epsilon^2}\right\}\right)\right),$$

which proves (24).

Therefore, (22) is proved.

Dealing with $T_{g3}(\epsilon)$. If $\delta = 0$, the bound is infinite and thus trivial. Assume $\delta > 0$. We prove that

$$\alpha^{2(1+\delta)T_{g3}(\epsilon)} \wedge nT_{g3}(\epsilon) = \mathcal{O}\left(\frac{\tilde{\Delta}^2}{\delta^2\epsilon^4} \wedge \frac{\sqrt{n}\tilde{\Delta}\log n}{\delta\epsilon^2}\right). \tag{25}$$

Let

$$h(y) = \frac{1}{1+\delta}\log_\alpha(y) - \frac{y}{\delta\sqrt{n}}.$$

It is easy to see that

$$h'(y) = \frac{1}{y(1+\delta)\log\alpha} - \frac{1}{\delta\sqrt{n}}.$$

Thus $h(y)$ is decreasing on $[0, y^*]$ and increasing on $[y^*, \infty)$ where

$$y^* = \frac{\delta\sqrt{n}}{(1+\delta)\sqrt{\alpha}}.$$

Now we distinguish two cases.

- If $h(y^*) \leq 0$, then $h(y) \leq 0$ for all $y > 0$ and thus $h(\tilde{\Delta}/\epsilon^2) \leq 0$. As a result,

$$h\left(\frac{\tilde{\Delta}}{\epsilon^2}\right) \leq 0 \Longrightarrow T_{g3}(\epsilon) \leq \frac{\tilde{\Delta}}{\delta\sqrt{n}\epsilon^2}.$$

If $\tilde{\Delta}/\delta\epsilon^2 \leq \sqrt{n}$,

$$T_{g3}(\epsilon) = \mathcal{O}(1) \Longrightarrow \alpha^{2(1+\delta)T_{g3}(\epsilon)} \wedge nT_{g3}(\epsilon) = \mathcal{O}(1),$$

and hence (25) is proved by recalling the footnote in page 3. Otherwise, note that

$$\alpha^{2(1+\delta)T_{g3}(\epsilon)} \wedge nT_{g3}(\epsilon) = \mathcal{O}(nT_{g3}(\epsilon)) = \mathcal{O}\left(\frac{\sqrt{n}\tilde{\Delta}}{\delta\epsilon^2}\right).$$

Since $\tilde{\Delta}/\delta\epsilon^2 > \sqrt{n}$,

$$\frac{\sqrt{n}\tilde{\Delta}}{\delta\epsilon^2} = \mathcal{O}\left(\frac{\tilde{\Delta}^2}{\delta^2\epsilon^4}\right).$$

Therefore, (25) is proved.

- If $h(y^*) > 0$, noting that $h(0) = h(\infty) = -\infty$, there must exist $0 < y_1^* < y^* < y_2^* < \infty$ such that $h(y_1^*) = h(y_2^*) = 0$ and $h(y) \geq 0$ iff $y \in [y_1^*, y_2^*]$. First we prove that

$$y_1^* = \mathcal{O}\left(1\right), \quad y_2^* = \mathcal{O}(\delta\sqrt{n}\log n). \tag{26}$$

As for y_1^*, if $y^* \leq 4$, then $y_1^* \leq y^* = \mathcal{O}(1)$. If $y^* > 4$, let

$$y = 1 + \frac{4}{y^*}.$$

Now we prove $y_1^* \leq y$. It is sufficient to prove $h(y) \geq 0$ and $y \leq y^*$. In fact, a simple algebra shows that

$$h(y^*) \geq 0 \Longrightarrow y^* \geq e \Longrightarrow y \leq 4 \leq y^*.$$

On the other hand, by Lemma 24

$$\log y \geq \frac{4/y^*}{1 + 4/y^*} \geq \frac{2}{y^*}.$$

Recalling that $y^* = \delta\sqrt{n}/(1+\delta)\log\alpha$,

$$h(y) \geq \frac{2}{(1+\delta)(\log\alpha)y^*} - \frac{1}{\delta\sqrt{n}}\left(1 + \frac{4}{y^*}\right) = \frac{1}{\delta\sqrt{n}}\left(2 - 1 - \frac{4}{y^*}\right) \geq 0.$$

Therefore, $y_1^* = \mathcal{O}(1)$.

As for y_2^*, let $C > 0$ be any constant, then for sufficiently large C,

$$(C+1)\delta\sqrt{n}\log_\alpha n \geq y^* = \frac{\delta\sqrt{n}}{(1+\delta)\log\alpha}.$$

On the other hand,

$$h((C+1)\delta\sqrt{n}\log_\alpha(n)) = \log_\alpha(C\log_\alpha n) - C\log_\alpha(\delta\sqrt{n}).$$

Then for sufficiently large C,

$$h((C+1)\delta\sqrt{n}\log_\alpha(n)) \leq 0.$$

Recalling that $h(y)$ is decreasing on $[y^*, \infty)$ and $h(y_2^*) = 0$, (26) must hold. Based on (26), (25) can be equivalently formulated as

$$\alpha^{2(1+\delta)T_{g3}(\epsilon)} \wedge nT_{g3}(\epsilon) = \mathcal{O}\left(\frac{\tilde{\Delta}^2}{\delta^2\epsilon^2}\left\{\frac{\tilde{\Delta}^2}{\epsilon^2} \wedge y_2^*\right\}\right). \tag{27}$$

Now we consider three cases.

  - If $\tilde{\Delta}/\epsilon^2 \geq y_2^*$,

$$h\left(\frac{\tilde{\Delta}}{\epsilon^2}\right) \leq 0 \Longrightarrow T_{g3}(\epsilon) = \frac{\tilde{\Delta}}{\delta\sqrt{n}\epsilon^2}.$$

Then

$$\alpha^{2(1+\delta)T_{g3}(\epsilon)} \wedge nT_{g3}(\epsilon) = \mathcal{O}(nT_{g3}(\epsilon)) = \mathcal{O}\left(\frac{\sqrt{n}\tilde{\Delta}}{\delta\epsilon^2}\right) = \mathcal{O}\left(\frac{\tilde{\Delta}}{\delta^2\epsilon^2}y_2^*\right)$$

where the last equality uses the fact that

$$y_2^* \geq y^* = \frac{\delta\sqrt{n}}{(1+\delta)\log\alpha}.$$

This proves (27).

- If $\tilde{\Delta}/\epsilon^2 \leq y_1^*$,

  $$h\left(\frac{\tilde{\Delta}}{\epsilon^2}\right) \leq 0 \Longrightarrow T_{g3}(\epsilon) = \frac{\tilde{\Delta}}{\delta\sqrt{n}\epsilon^2}.$$

  By (26),

  $$T_{g3}(\epsilon) = \mathcal{O}(1) \Longrightarrow \alpha^{2(1+\delta)T_{g3}(\epsilon)} \wedge nT_{g3}(\epsilon) = \mathcal{O}(1),$$

  and hence (25) is proved by recalling the footnote in page 3.
- If $\tilde{\Delta}/\epsilon^2 \in [y_1^*, y_2^*]$,

  $$h\left(\frac{\tilde{\Delta}}{\epsilon^2}\right) \geq 0 \Longrightarrow T_{g3}(\epsilon) = \frac{1}{1+\delta}\log_\alpha\left(\frac{\tilde{\Delta}}{\epsilon^2}\right).$$

  Then

  $$\alpha^{2(1+\delta)T_{g3}(\epsilon)} \wedge nT_{g3}(\epsilon) = \mathcal{O}\left(\alpha^{2(1+\delta)T_{g3}(\epsilon)}\right) = \mathcal{O}\left(\frac{\tilde{\Delta}^2}{\epsilon^4}\right) = \mathcal{O}\left(\frac{\tilde{\Delta}}{\epsilon^2}\left\{\frac{\tilde{\Delta}}{\epsilon^2} \wedge y_2^*\right\}\right),$$

  which proves (27) since $\delta = O(1)$.

Summary Putting (21) and (25) together and using the fact that $\Delta', \tilde{\Delta} = \mathcal{O}(L\Delta)$, we prove the bound for $\mathbb{E}\mathrm{Comp}_g(\epsilon)$. As for $\mathbb{E}\mathrm{Comp}_f(\epsilon)$, by Theorem 21, we can directly apply (21) by replacing Δ/λ_T by Δ and σ^2 with σ^2/μ.

C.5. Complexity analysis: Proof of Theorem 13

For the first claim, we set $\delta = 0$ thus $\mathcal{R}(T) = T$. Applying (13) recursively with the fact $\sum_{i=0}^{T} 1/(1+x)^i \leq (1+x)/x$ for $x > 0$, we obtain

$$\mathbb{E}\left(f(\tilde{x}_{\mathcal{R}(T)}) - f(x^\star)\right) \leq \frac{1}{(\mu\lambda+1)^T}\mathbb{E}\left(f(\tilde{x}_0) - f(x^\star)\right) + \frac{\sigma^2 I(B<n)}{2\mu B},$$

where $\lambda = \sqrt{B}/2L$. Setting $B = \left(n \wedge \frac{\sigma^2}{4\mu\epsilon^2}\right)$, the second term is less than $\epsilon^2/2$. For $T \geq \left(1 + \frac{2L}{\mu\sqrt{B}}\right)\log\frac{2\Delta_f}{\epsilon^2}$ also the first term is less han $\epsilon^2/2$ which follows from Lemma 24. As the cost of each epoch is $2B$ this result implies that the total complexity is

$$\mathcal{O}\left(\left(B + \frac{\sqrt{B}L}{\mu}\right)\log\frac{\Delta_f}{\epsilon^2}\right).$$

For the second claim, we use (13) recursively together with Theorem 7 and with the fact $\sum_{i=0}^{\infty} 1/(1+x)^i \leq (1+x)/x$ for $x > 0$, we obtain

$$\mathcal{L}_T \leq \frac{\mathcal{L}_1}{(\mu\lambda+1)^{T-1}} + \frac{\sigma^2 \lambda I(B<n)}{B} \frac{1+\lambda\mu}{\lambda\mu}.$$

Further by Theorem 7,

$$\mathcal{L}_1 \leq 2\Delta_f + \frac{\sigma^2 \lambda I(B<n)}{B}.$$

Using definition of $\mathcal{L}_j$ and $\delta = 0$, we get

$$\begin{aligned}
\mathbb{E}\|\nabla f(\tilde{x}_{\mathcal{R}(T)})\|^2 &\leq \frac{2\Delta_f}{\lambda(\mu\lambda+1)^{T-1}} + \frac{\sigma^2 I(B<n)}{B}\left(2 + \frac{1}{\lambda\mu}\right) \\
&\leq \frac{2\Delta_f}{\lambda(\mu\lambda+1)^{T-1}} + \frac{\sigma^2 I(B<n)}{B}\left(2 + \frac{2L}{\sqrt{B}\mu}\right)
\end{aligned}$$

The choice of B to be $\left(\left\{\frac{8\sigma^2}{\epsilon^2} + \frac{8\sigma^{4/3}L^{2/3}}{\epsilon^{4/3}\mu^{2/3}}\right\} \wedge n\right)$ guarantees that the second term is less than $\epsilon^2/2$. By the same reasoning as for the second claim, we obtain following complexity

$$\mathcal{O}\left(\left(B + \frac{\sqrt{B}L}{\mu}\right) \log \frac{L\Delta_f}{\sqrt{B}\epsilon^2}\right).$$

Appendix D. Miscellaneous

Lemma 22 *Let $z_1, \ldots, z_M \in \mathbb{R}^d$ be an arbitrary population and $\mathcal{J}$ be a uniform random subset of $[M]$ with size m. Then*

$$\mathrm{Var}\left(\frac{1}{m}\sum_{j\in\mathcal{J}} z_j\right) \leq \frac{I(m<M)}{m} \cdot \frac{1}{M}\sum_{j=1}^{M} \|z_j\|_2^2.$$

Lemma 23 *For any positive integer n,*

$$\sum_{t=1}^{n} \frac{1}{t} \leq 1 + \log n.$$

Proof Since $x \mapsto 1/x$ is decreasing,

$$\sum_{t=1}^{n} \frac{1}{t} = 1 + \sum_{t=2}^{n} \frac{1}{t} \leq 1 + \int_1^n \frac{dx}{x} = 1 + \log n.$$

■

Lemma 24 *For any $x > 0$,*

$$\frac{1}{\log(1+x)} \leq 1 + \frac{1}{x}.$$

Proof Let $g(x) = (1+x)\log(1+x) - x$. Then

$$g'(x) = 1 + \log(1+x) - 1 = \log(1+x) \geq 0.$$

Thus g is increasing on $[0, \infty)$. As a result, $g(x) \geq g(0) = 0$. ■

Lemma 25 *For any $x > 0$ and $a > 0$, let*

$$y(x,a) = 2 + \sqrt{\frac{1}{x^2}\log\left(\frac{1}{x^2} \vee 1\right)} + \frac{\log a}{ax^2}.$$

Then for any $y \geq y(x,a)$,

$$\frac{\log(y \wedge a)}{y(y \wedge a)} \leq x^2.$$

Proof Let $h(y) = (\log y)/y$ and $H(y) = (\log(y \wedge a))/(y(y \wedge a))$. Then

$$h'(y) = \frac{1 - \log y}{y^2}.$$

Thus $h(y)$ is decreasing on $[0, e]$ and increasing on $[e, \infty)$. As a result,

$$H(y) = \max\left\{\frac{\log y}{y^2}, \frac{\log a}{ya}\right\}.$$

If $x^2 > 1/2e$, since $y \geq y(x,a) \geq 2$ and $h(y)$ attains its minimum at $y = e$ with $h(e) = 1/e$,

$$H(y(x,a)) = h(y(x,a))/y(x,a) \leq h(e)/2 \leq x^2.$$

If $x^2 \leq 1/2e$. First we note that

$$\frac{\log a}{y(x,a)a} \leq x^2,$$

On the other hand,

$$y(x,a) \geq \sqrt{\frac{1}{x^2}\log\left(\frac{1}{x^2}\right)} \geq \sqrt{2e\log(2e)} \geq e.$$

Since $h(y)$ is decreasing on $[e, \infty)$

$$\frac{\log y(x,a)}{y(x,a)^2} \leq \frac{\log\left(\frac{1}{x}\sqrt{\log\left(\frac{1}{x^2}\right)}\right)}{\frac{1}{x^2}\log\left(\frac{1}{x^2}\right)} \leq \frac{\log\left(\frac{1}{x}\sqrt{\frac{1}{x^2} - 1}\right)}{\frac{1}{x^2}\log\left(\frac{1}{x^2}\right)} \leq x^2.$$

■