Proceedings of Machine Learning Research vol 75:1–23, 2018

Taming Convergence for Asynchronous Stochastic Gradient Descent with Unbounded Delay in Non-Convex Learning

Xin Zhang XINZHANG@IASTATE.EDU
Department of Statistics, Iowa State University

Jia Liu JIALIU@IASTATE.EDU
Department of Computer Science, Iowa State University

Zhengyuan Zhu ZHUZ@IASTATE.EDU
Department of Statistics, Iowa State University

Abstract

Understanding the convergence performance of asynchronous stochastic gradient descent method (Async-SGD) has received increasing attention in recent years due to their foundational role in machine learning. To date, however, most of the existing works are restricted to either bounded gradient delays or convex settings. In this paper, we focus on Async-SGD and its variant Async-SGDI (which uses increasing batch size) for non-convex optimization problems with unbounded gradient delays. We prove $o(1/\sqrt{k})$ convergence rate for Async-SGD and $o(1/k)$ for Async-SGDI. Also, a unifying sufficient condition for Async-SGD's convergence is established, which includes two major gradient delay models in the literature as special cases and yields a new delay model not considered thus far.

Keywords: Asynchronous stochastic gradient descent, parallel distributed computing, convergence rate, unbounded delay.

1. Introduction

Fueled by large-scale machine learning and data analytics, recent years have witnessed an ever-increasing need for computing power. However, with the miniaturization of transistors nearing the limit at atomic scale, it is projected that the celebrated Moore's law (the doubling growth rate of CPU speed in every 18 months) will end in around 2025 (Poeter, 2015). Consequently, to sustain the rapid growth for machine learning technologies in the post-Moore's-Law era, the only viable solution is to exploit *parallelism* at and across different spatial scales. Indeed, the recent success of machine learning research and applications is due in a large part to the advances in multi-core CPU/GPU technologies (on the micro chip level) and networked cloud computing (on the macro data center level), which enable the developments of highly parallel and distributed algorithmic architectures. Such examples include parallel SVM (Zhang et al., 2005), scalable matrix factorization (Yu et al., 2012, 2014; Xu et al., 2013), distributed deep learning (Dean et al., 2012; Povey et al., 2014; Abadi et al., 2016; Li et al., 2014), to name just a few.

However, developing efficient and effective parallel algorithms is highly non-trivial. In the literature, most parallel machine learning algorithms are synchronous in nature, i.e., a set of processors performing certain computational tasks in a distributed fashion under a common clock. Although synchronous parallel algorithms are relatively simpler to design and analyze theoretically, their implementations in practice are usually problematic: First, in many computing systems, maintaining clock synchronization is expensive and incurs high complexity and system overhead. Second, synchronous parallel algorithms do not work well under heterogenous computing environments since all processors must wait for the slowest processor to finish in each iteration. Exacerbating the problem is the fact that, in many machine learning applications, it is often difficult to decompose a problem into subproblems with similar difficulty. This introduces yet another layer of heterogeneity in CPU/GPU processing time. Third, synchronous operations in parallel algorithms often induce periodic spikes in information exchanges and congestions in the systems, which further cause high communication latency and even information losses. Due to these limitations, it is not only desirable but also necessary to consider *asynchronous parallel algorithmic designs* in practice.

In an asynchronous parallel algorithm, rather than making updates simultaneously, each node computes its own solution in each iteration without waiting for other nodes in the system. Compared with their synchronous counterparts, asynchronous parallel algorithms are more resilient to heterogeneous computing environments and cause less network congestions and delay. As a result, asynchronous parallel algorithmic designs are more attractive in practice for solving large-scale machine learning problems. However, one of the most critical issues of asynchronous parallel algorithms is that the use of stale system state information is unavoidable due to the asynchronous updates. If not treated carefully, such delayed system information could destroy the convergence performance of their synchronous versions. This problem is particularly concerning for the *asynchronous stochastic gradient descent method* (Async-SGD), which is the fundamental building block of many distributed machine learning frameworks in use today (e.g., TensorFlow, MXNet, Caffe, etc.). Hence, understanding the convergence performance of Async-SGD is important and has received increasing attention in recent years (see, e.g., Recht et al. (2011); Lian et al. (2015); Zheng et al. (2017); Huo and Huang (2017) etc.). To date, however, most of the existing work in this area are restricted to the bounded gradient delay setting, whereas results on *unbounded* gradient delay remain quite limited (see Section 2 for more detailed discussions). Moreover, all existing convergence results with unbounded delay in the literature require convexity assumptions, which are irrelevant to the inherently non-convex nature of many challenging machine learning problems. In light of these limitations, our goal is in this paper is to fill this gap and achieve a deeper understanding of the convergence performance of Async-SGD in non-convex learning.

Toward this end, in this paper, we consider using Async-SGD to solve a non-convex optimization problem in the form of:

$$\min_{x\in\mathbb{R}^d} f(x) = \mathbb{E}[F(x;\xi)], \tag{1}$$

where ξ is an i.i.d. random sample drawn from the database, and $f(x)$ is a smooth non-convex function. The objective in (1) could be infinite-sum, which means the sample size in database is large. We note that Problem (1) is general enough to represent a wide range of machine learning problems in practice. Further, we do *not* assume any bounded delay of the outdated stochastic gradients during the execution of Async-SGD. As will be shown later, the unbounded assumption significantly complicates the convergence analysis of Async-SGD. Our main technical results and key contributions in this paper are summarized as follows:

- First, we show that by choosing step-sizes at the speed $O(1/(\sqrt{k}\log(k)))$, $\mathbb{E}\{\|\nabla f(x_k)\|_2\}$ converges to zero with rate $o(1/\sqrt{k})$, which is much stronger compared to the $O(1/\sqrt{k})$ convergence rate in existing works of this area (see, e.g., (Lian et al., 2015) and references therein). This is a surprising result because, to our knowledge, most existing work in the literature only yields Big-O bounds (e.g.,. $O(1/\sqrt{k})$). In other words, our result shows that unbounded gradient delay in

Async-SGD actually makes *no* difference in terms of convergence rate (in order sense) compared to the synchronous version.

- Second, by leveraging a supermartingale convergence theorem, we prove a generalized and more relaxed sufficient condition on the probability distribution of the gradient-updating delay that guarantees convergence. Our sufficient condition offers a *unifying* framework that includes two major gradient update delay models often assumed in the literature as special cases, namely: 1) bounded delays, and 2) unbounded i.i.d. delay (see more detailed discussions in Section 2). Further, our sufficient condition implies a new gradient delay model: *uniformly second-moment bounded delays*, which means the delay distributions across iterations could be non-i.i.d, unbounded, and even *heavy-tailed* (e.g., log-normal, T-distribution, Weibull, etc.). Interestingly, this new gradient delay model itself also generalizes the previous two delay models in the literature.
- Inspired by the idea of variance reduction methods for stochastic approximation (Bernstein et al., 2018), we consider a variant of Async-SGD with increasing batch size (Async-SGDI). We show that, if the batch size grows at rate $\omega(k)$, Async-SGDI achieves an $o(1/k)$ convergence rate result under a *fixed* step-size. In other words, as long as the batch size grows slightly faster than linear, a small constant step-size is sufficient to achieve an even faster Small-O convergence rate. Therefore, there is no need to be concerned with the use of vanishing step-size strategies, which could be problematic because of numerical instability in practice.

The rest of the paper is organized as follows. Related work is discussed in Section 2. We will present the system model of Async-SGD in Section 3. In Section 4, the convergence rate of Async-SGD with unbounded delay is derived. The conclusion is given in Section 5. Due to limited space, experiment results and proofs are shown in Supplementary.

2. Related work

To put our work in comparison perspectives, in this section, we first provide a quick overview on stochastic gradient descent method (SGD). We then focus on the recent advancements of Async-SGD.

1) SGD and variance reduction: The SGD algorithm traces its root to the seminal work by Robbins and Monro (1951) and Kiefer and Wolfowitz (1952), and has become a key component for solving many large-scale optimization problems. Due to its foundational importance, the convergence rates of SGD and its variants have been actively researched over the years. It is well known that the convergence rate of SGD is $O(1/\sqrt{k})$ for convex problems (see, e.g., Nemirovski et al. (2009)) and $O(1/k)$ for strongly convex problems (see, e.g., Moulines and Bach (2011)). To improve the convergence speed of SGD, stochastic variance reduction methods have been proposed. For example, the stochastic averaged gradient (SAG) method proposed in (Schmidt et al., 2017) converges at $O(1/k)$ speed for convex problems and converges linearly for strongly convex problems. The stochastic variance reduced gradient (SVRG) method proposed in (Johnson and Zhang, 2013) also enjoys similar sublinear convergence rate for convex problems and linear convergence rate for strongly convex problems.

2) SGD for non-convex problems: Due to the inherent non-convex nature in training deep neural networks, the convergence performance of SGD for non-convex optimization problems has

Work	Method	Sum	Convexity	Delay	Rate
Hannah and Yin (2016)	ARock	-	convex	unbounded	-
Sra et al. (2015)	Adadelay	infinite-sum	convex	unbounded	$O(1/k)$
Sun et al. (2017)	Async-BCD	-	strongly convex	bounded	$O(\rho^k)$
			convex		$o(1/k)$
			nonconvex		$o(1/\sqrt{k})$
Lian et al. (2015)	Async-SGD	finite-sum	nonconvex	bounded	$O(1/\sqrt{k})$
Huo and Huang (2017)	Async-SVRG	finite-sum	nonconvex	bounded	$O(1/k)$
Our work	Async-SGD	infinite-sum	nonconvex	unbounded	$o(1/\sqrt{k})$
	Async-SGDI		nonconvex	unbounded	$o(1/k)$

Table 1: Convergence comparisons for existing asynchronous methods($\rho \in (0,1)$ is a constant; "Sum" means whether the total size of sample is finite or not).

also become a focal research area recently. For example, Ghadimi and Lan (2013) proved that the ergodic convergence rate for nonconvex objection function with $\sigma-$bounded gradient is $O(1/\sqrt{k})$. Later, Reddi et al. (2016) extended SVRG to non-convex problem and proved that it has a sublinear convergence. We note that this convergence rate result is consistent with that of the convex case.

3) Asynchronous SGD for convex problems: As mentioned in Section 1, Async-SGD has become increasingly popular recently due to its simplicity in implementation and practical relevance in many machine learning frameworks. One of the earliest studies on Async-SGD is the algorithm termed HOGWILD! in (Recht et al., 2011). HOGWILD! is a lock-free asynchronous parallel implementation of SGD on the shared memory system with sublinear convergence rate for strongly convex smooth problems. At roughly the same time, Agarwal and Duchi (2011) studied the convergence performance of SGD-based optimization algorithms on distributed stochastic convex problems with asynchronous and yet delayed gradients. Interestingly, asymptotic convergence rate $O(1/\sqrt{k})$ is shown in their work, which is consistent with that of the non-delayed case. However, compactness of feasible domain and bounded gradient are assumed in this work. In (Reddi et al., 2015), asynchronous stochastic variance reduction (Async-SVR) methods were analyzed for convex objectives and bounded delay. Note that all aforementioned Async-SGD methods assumed bounded gradient delay. One of the first investigations on unbounded delay is due to Hannah and Yin (2016), where the convergence rate of ARock, an asynchronous coordinate decent method for solving convex optimization problems, is considered. It was shown that ARock converges weakly to a solution with probability 1 if the unbounded delayed gradients are independent and identically distributed (i.i.d.).

4) Asynchronous SGD for non-convex problems: Similar to their synchronous counterparts, Async-SGD for nonconvex optimization problems also starts to attract some attentions lately. For example, Lian et al. (2015) studied the convergence rate of Async-SGD for non-convex optimization problems with bounded delay, where they showed an $O(1/\sqrt{k})$ sublinear convergence rate. However, the best convergence rate they provided is highly dependent on the step-size selection strategy, which in turn depends on some *a priori* iteration threshold value K. Most recently in (Huo and Huang, 2017), an asynchronous mini-batch SVRG with bounded delay is proposed for solving non-convex optimization problems. They proved that the proposed method converges with an $O(1/k)$ convergence rate for non-convex optimization.

To conclude this section, we summarize the convergence performance guarantees in the prior literature and our results in Table 1 for clearer comparisons.

3. System model and the asynchronous stochastic gradient descent algorithms

In this section, we first describe the system model for Asynchronous parallel algorithms. Then, we present the standard Async-SGD algorithm and a variant of Async-SGD with increasing batch size, denoted as Asyn-SGDI.

Consider solving the optimization problem in (1) in a parallel computing architecture consisting of a parameter server and N workers (oftenly N is a fixed number), as shown in Figure 1. In practice, each worker could be a GPU on a chip-scale or a standalone server on a datacenter-scale. Under the Async-SGD algorithm, each worker independently retrieves the current values x_k from the parameter server and randomly select data samples from database and compute the stochastic gradient. Once the computation is finished, each worker immediately reports the computed stochastic gradient to the parameter server without waiting for other workers, and then start next computing cycle. On the other hand, upon collecting M gradients from workers, the parameter server updates its current parameter with these stochastic gradients. However, due to asynchronicity, the server could use stale gradient information to update the parameters, which will affect the convergence. We present Async-SGD in Algorithm 1.

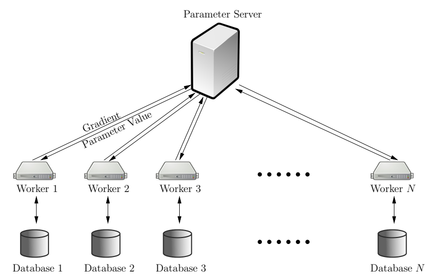


Figure 1: A parallel computing architecture for asynchronous gradient descent (Async-SGD).

Algorithm 1: Asynchronous Stochastic Gradient Descent (Async-SGD).

At the parameter server:

1. In the i-th update, wait till collecting M stochastic gradients $G(x_{i-\tau_{i,m}};\xi_{i,m})$ from the workers.
2. Update parameter $x_{i+1} = x_i - \gamma_i \sum_{m=1}^{M} G(x_{i-\tau_{i,m}};\xi_{i,m})$.

At each worker:

1. Retrieve the current value of parameter x from the parameter server.
2. Randomly select a sample ξ from the database.
3. Compute stochastic gradient $G(x;\xi)$ and report it to server.

Algorithm 2: Async-SGD with increasing batch size (Async-SGDI).

At the parameter server:

1. In the i-th update, wait till collecting $n_i M$ stochastic gradients $G(x_{i-\tau_{i,m}};\xi_{i,m})$ from workers.
2. Update parameter $x_{i+1} = x_i - \frac{\gamma_i}{n_i} \sum_{m=1}^{n_i M} G(x_{i-\tau_{i,m}};\xi_{i,m})$.

At each worker:

1. Retrieve the current value of parameter x from the parameter server.
2. Randomly select a sample ξ from the database.
3. Compute stochastic gradient $G(x;\xi)$ and report it to server.

In Algorithm 1, $G(x;\xi)$ denotes a stochastic gradient of $f(x)$ that is dependent on a random sample ξ; $\tau_{i,m}$ represents the delay for the m-th gradient in the mini-batch in i-th update seen by the parameter server. As shown in Algorithm 1, the parameter server updates the parameters regardless of the freshness of the collected gradients.

Further, inspired by variance reduction idea in (Bernstein et al., 2018), we consider a modified scheme for Async-SGD with the same system. Instead of a fixed number of gradients, the server collects an increasing number of gradients as the number of iterations increases to help reduce the variance of stochastic gradients. We outline this scheme in Algorithm 2. Apparently, compared to the basic Async-SGD, the only difference between the two algorithms is that the batch size $n_i M$ at the parameter server is increasing, where $\{n_i\}_{i=1}^{\infty}$ is an integer-valued increasing series.

4. Convergence analysis

In this section, we will conduct convergence analysis for the two Async-SGD algorithms described in Section 3. Similar to previous work on optimization for non-convex learning problems (see, e.g., Cartis et al. (2010); Gratton et al. (2008)), we use the expected ℓ_2 norm of the gradient, i.e., $\mathbb{E}\{\|\nabla f(x)\|^2\}$, as the convergence metric. For non-convex optimization problems, we show that Async-SGD converges to a stationary point with asymptotic convergence rate $o(1/\sqrt{k})$. For Async-SGDI, the asymptotic convergence rate is even faster at $o(1/k)$.

4.1. Assumptions

We first state the following assumptions for our analysis. The first three are commonly assumed in the literature for analyzing the convergence of SGD. The fourth assumption is a sufficient condition for the characteristics of gradient delays under which the convergence of Async-SGD is guaranteed.

Assumption 1 (Lower bounded objective function) *For the objective function f, there exists an optimal solution x^*, such that $\forall x \neq x^*$, we have $f(x) \geq f(x^*)$.*

Assumption 2 (Lipschitz continuous gradient) *There exists a constant $L > 0$ such that the objective function $f(\cdot)$ satisfies $\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$, $\forall x, y \in \mathbb{R}^d$.*

Assumption 3 (Unbiased gradients with bounded variance) *The stochastic gradient $G(x;\xi)$ satisfies: $\mathbb{E}(G(x;\xi)) = \nabla f(x)$, $\forall x, \xi$, and $\mathbb{E}(\|(G(x;\xi)) - \nabla f(x)\|^2) \leq \sigma^2$, $\forall x$.*

Assumption 4 (Restriction of gradient delay probabilities) *There exists a non-negative sequence $\{c_i\}_{i=1}^{\infty}$, such that*

$$c_{j+1} + \frac{\gamma_k M L^2}{2} \sum_{i=j}^{k} i\mathbb{P}(\tau_k = i) \leq c_j,\ \forall\, k, \tag{2}$$

where τ_k denotes the maximum delay in k-th iteration, i.e., $\tau_k = \max_m \tau_{k,m}$ and γ_k is the step-size.

The intuition behind Assumption 4 is as follows: The existence of sequence $\{c_i\}_{i=1}^{\infty}$ ensures that the probability of gradient-updating delay uniformly decays to zero. In other words, the average of gradient-updating delay tends to be small despite the support of the random delay is unbounded. We will further discuss the consequence of this assumption in Section 4.4.

4.2. Convergence Results for Async-SGD with Unbounded Delay

To establish the convergence results of Async-SGD with unbounded delay, we construct the following Lyapunov function:

$$\zeta^k = f(x_k) - f(x^*) + \sum_{j=1}^{k} c_j \|x_{k+1-j} - x_{k-j}\|^2. \tag{3}$$

In the Lyapunov function in (3), the first part $f(x_k) - f(x^*)$ measures the optimality error between current objective value and the optimal objective value. The second part $\sum_{j=1}^{\infty} c_j \|x_{k+1-j} - x_{k-j}\|^2$ in (3) is a weighted sum of the distances between past iterates, which can be viewed as the accumulative error due to asynchronous gradient updates. Here the weight sequence $\{c_i\}_{i=1}^{\infty}$ is as stated in Assumption 4. We note that a similar type of Lyapunov function was used in (Hannah and Yin, 2016), where they used $\|x_k - x^*\|^2$ the optimality error thanks to the non-expansiveness assumption therein. Similar to the discussions in previous work, because of asynchronicity, it is hard to directly show the contraction relationship $\mathbb{E}[f(x_k) - f(x^*)] \leq f(x_{k-1}) - f(x^*)$. However, we can prove the following inequality for the proposed Lyapunov function ζ^k, which will play a key role in our subsequent analysis.

Lemma 1 *Under Assumptions 1–4, if the step-size $\{\gamma_k\}_{k=1}^{\infty}$ satisfies that $\gamma_k \leq \frac{1}{2Mc_1 + ML}$, $\forall k$, then the following inequality holds:*

$$\mathbb{E}\{\zeta^{k+1} | \mathscr{F}^k\} + \frac{\gamma_k M}{2} \|\nabla f(x_k)\|^2 \leq \zeta^k + (c_1 \gamma_k^2 M + \frac{L\gamma_k^2 M}{2})\sigma^2. \tag{4}$$

where $\mathscr{F}^k$ represents the filtration of the history of iterates and delays, i.e., $\mathscr{F}^k = \sigma\langle x_0, x_1, \ldots, x_k;$ $\tau_1, \ldots, \tau_k\rangle$

Lemma 1 connects the total error ζ and the convergence criterion $\|\nabla f(\cdot)\|^2$. Intuitively, we can see that if the second term in the right hand side of (4) is summable, then $\|\nabla f(\cdot)\|^2$ should also be summable. Based on Lemma 1 and by applying the supermartingale convergence theorem in (Hannah and Yin, 2016; Combettes and Pesquet, 2015), we have following main convergence result for Async-SGD:

Theorem 1 *Under Assumptions 1–4, if the step-size sequence $\{\gamma_k\}_{k=1}^{\infty}$ satisfies: i) $\gamma_k \leq \frac{1}{2Mc_1 + ML}$, $\forall k$; ii) $\sum_{k=1}^{\infty} \gamma_k = \infty$; and iii) $\sum_{k=1}^{\infty} \gamma_k^2 < \infty$, where M is the fixed batch size, L is the Lipschitz constant in Assumption 2, and c_1 is the first element in the sequence $\{c_i\}_{i=1}^{\infty}$ in Assumption 4, then we have $\mathbb{E}\{\sum_{k=1}^{\infty} \gamma_k \|\nabla f(x_k)\|^2\} < \infty$ and $\mathbb{E}\{\|\nabla f(x_k)\|^2\} \to 0$.*

Due to space limitation, we relegate the proof details of Theorem 1 to Appendix 1. Next, we show that Theorem 1 implies that we can properly choose the step-size sequence $\{\gamma_k\}_{k=1}^{\infty}$ to obtain an $o(1/\sqrt{k})$ convergence rate for Async-SGD:

Proposition 1 *Consider the diminishing step-size sequence $\gamma_k = O(\frac{1}{k^{1/2}\log(k)})$ and $\gamma_k \leq \frac{1}{2Mc_1 + ML}$, $k = 1, 2, \ldots$. Then, the asymptotic convergence rate for Async-SGD is:*

$$\mathbb{E}\{\|\nabla f(x_k)\|^2\} = o(1/\sqrt{k}). \tag{5}$$

Proof The proof logic of the Little-O rates is based on contradiction:

Step 1): With the stated choice of step-size γ_k, we could show that γ_k is unsummable and $(\gamma_k)^2$ is summable, satisfying conditions ii) and iii) in Theorem 1, respectively. Thus, it follows that $\mathbb{E}\{\sum_{k=1}^{\infty} \gamma_k \|\nabla f(x_k)\|^2\} < \infty$.

Step 2): Now, suppose $\mathbb{E}\{\|\nabla f(x_k)\|^2\} = o(1/\sqrt{k})$ instead, then based on p-series properties, the stated choice of γ_k yields an $\mathbb{E}\{\sum_{k=1}^{\infty} \gamma_k \|\nabla f(x_k)\|^2\} = \infty$, contradicting to our conclusion in Step 1. Hence, $\mathbb{E}\{\|\nabla f(x_k)\|^2\} = o(1/\sqrt{k})$.

More detials are provided in Appendix 3. ■

Proposition 1 shows that as the number of iterations increases, the negative effect of outdated gradient information in Async-SGD will vanish asymptotically under the chosen step-sizes.

4.3. Async-SGD with increasing batch size

To analyze convergence performance of Async-SGD with increasing batch size (Async-SGDI), we extend Lemma 1 to obtain following inequality:

Lemma 2 *Under Assumptions 1–4, if the step-size sequence $\{\gamma_k\}_{k=1}^{\infty}$ satisfies $\gamma_k \leq \frac{1}{2Mc_1+ML}$, $\forall k$, then the following inequality holds for Async-SGDI:*

$$\mathbb{E}\{\zeta^{k+1}|\mathscr{F}^k\} + \frac{\gamma_k M}{2}\|\nabla f(x_k)\|^2 \leq \zeta^k + \Big(c_1\gamma_k^2 M + \frac{L\gamma_k^2 M}{2}\Big)\frac{\sigma^2}{n_k}, \tag{6}$$

where M denotes the initial batch size and $\{n_k\}$ is some integer-valued increasing sequence.

Then, by applying the supermartingale convergence theorem in a similar fashion, we have the key convergence result for Aysnc-SGDI:

Theorem 2 *Under Assumptions 1–4, let the batch size sequence be chosen as $\{M_k := n_k M\}$, where M is the initial batch size and the integer-valued sequence $\{n_k\}_{k=1}^{\infty}$ is increasing and satisfies $\sum_{k=1}^{\infty}\frac{1}{n_k} < \infty$. Also, suppose that the step-size $\{\gamma_k\}_{k=1}^{\infty}$ satisfies $\gamma_k \leq \frac{1}{2Mc_1+ML}$, $\forall k$. Then, we have $\mathbb{E}\{\sum_{k=1}^{\infty}\gamma_k\|\nabla f(x_k)\|^2\} < \infty$ and $\mathbb{E}\{\|\nabla f(x_k)\|^2\} \to 0$.*

Again, due to space limitation, we relegate the proof details of Theorem 2 to Appendix 2. With Theorem 2, we claim the following asymptotic convergence rate for Async-SGDI (proof detials in Appendix 3):

Proposition 2 *Let the sequence $\{n_k\}_{k=1}^{\infty}$ be chosen as $n_k = \omega(k)$. Then, with a fixed step-size satisfying $\gamma \leq \frac{1}{2Mc_1+ML}$, we have $\mathbb{E}\{\|\nabla f(x_k)\|^2\} = o(1/k)$.*

Proposition 2 implies that, by using an increasing batch size, Async-SGDI with a constant step-size converges at rate $o(1/k)$. However, as batch size increases, the runtime per iteration would also become longer. Hence, it is unclear whether the number of iteration is a good convergence performance metric. To resolve this ambiguity, we first note that we only need the batch size to grow at a rate $\omega(k)$. According to the small-omega definition, the batch size can grow linearly with an arbitrarily small slope, i.e., the batch size increases very slowly. Second, with constant step-size, the algorithm is much more stable numerically. We further provide some runtime simulation results in Appendix to demonstrate the efficiency of Async-SGDI.

4.4. Discussion

We can see from the proofs of Theorems 1 and 2 that the non-negative sequence $\{c_i\}_{i=1}^{\infty}$ in Assumption 4 is a mathematical construct that plays a key role in establishing the convergence of both Async-SGD algorithms. In this subsection, we will further discuss the intuition behind $\{c_i\}_{i=1}^{\infty}$ in Assumption 4 and show that such a non-negative sequence $\{c_i\}_{i=1}^{\infty}$ indeed exists. Further, we show that Assumption 4 is a general framework that unifies several known sufficient conditions in the literature for Async-SGD convergence as special cases.

To show the existence of $\{c_i\}_{i=1}^{\infty}$, we note that Assumption 4 establishes the relationship between the marginal probabilities of the random delays $\{\tau_i\}_{i=1}^{\infty}$ and the weights $\{c_i\}_{i=1}^{\infty}$ in the Lyapunov function in (3). It says that $\{c_i\}_{i=1}^{\infty}$ is a monotonic decreasing sequence and the difference between two adjacent elements satisfy:

$$c_j - c_{j+1} \geq \frac{\gamma_k M L^2}{2} \sum_{i=j}^{k} i\mathbb{P}(\tau_k = i) = O\Big(\sum_{i=j}^{k} i\mathbb{P}(\tau_k = i)\Big),$$

which depends on the term $\sum_{i=l}^{k} i\mathbb{P}(\tau_k = i)$ for all iterations k. Now, consider the upper bound $\gamma_k \leq \frac{1}{2Mc_1+ML}$ (cf. Theorems 1 and 2). For the step-size γ_1 to be non-zero, c_1 must be finite. Also, since $\{c_i\}_{i=1}^{\infty}$ is monotonic decreasing and bounded from below (since $c_i \geq 0$, $\forall i$), we have that $c_i > 0$, $i \geq 2$, also exist if c_1 exists.

In what follows, we show that Assumption 4 unifies two known delay models as special cases and implies a new delay model that has not been considered in the literature thus far. The first special case is the bounded delay model, which has been widely assumed and investigated in the literature (cf. Lian et al. (2015); Huo and Huang (2017) etc.). This model is reasonable as long as the gradient computation workload is finite for all workers. Assume that τ_k is bounded by a constant T. Then, since $P(\tau > T) = 0$, Eq. (2) could be written as:

$$c_j \geq \begin{cases} c_{j+1} + \frac{\gamma_k M L^2}{2} \sum_{i=j}^{k} i\mathbb{P}(\tau_k = i), & \text{if } j < T \\ c_{j+1}, & \text{if } j \geq T \end{cases} \tag{7}$$

It is easy to see that the weight sequence with $c_j = 0$ for $j > T$ and the subsequence $\{c_i\}_{i=1}^{T}$ chosen according to (7) satisfies (2). Hence, we have the following convergence result for the bounded delay model (see Appendix 4 for proof details):

Proposition 3 *(**Bounded Delay**) If the random delays in gradient updates $\{\tau_k\}_{k=1}^{\infty}$ are uniformly upper bounded by a constant $T > 0$, then $\{c_i\}_{i=1}^{\infty}$ exists. Hence, Async-SGD and Async-SGDI converge with rates $o(1/\sqrt{k})$ and $o(1/k)$, respectively.*

The second delay model is such that the sequence of random delays $\{\tau_k\}_{k=1}^{\infty}$ are i.i.d. and the underlying distribution has a finite second moment. This model is reasonable when the number of iterations is large and the system has reached the stationary state. For this case, we have the following result (see Appendix 4 for proof details):

Proposition 4 *(**I.I.D. Random Delay**) If the random delays τ_k, $\forall k$, are i.i.d. and the underlying distribution has a finite second moment, then $\{c_i\}_{i=1}^{\infty}$ exists. Hence, Async-SGD and Async-SGDI converge with rates $o(1/\sqrt{k})$ and $o(1/k)$, respectively.*

Next, we consider the case where the distributions for all random delay variables are different, but all distributions are uniformly upper bounded by a series that has a finite second moment. In this case, we have the following result (see Appendix 4 for proof details):

Proposition 5 ***(Uniformly Upper Bounded Probability Series)*** *Consider the probability series of random delays $\{\tau_k\}_{k=1}^{\infty}$. If there exists a series $\{a_i\}_{i=1}^{\infty}$ such that i) $\mathbb{P}(\tau_k = i) \le a_i$, $\forall k$, and ii) $\sum_{i=1}^{\infty} i^2 a_i < \infty$, then $\{c_i\}_{i=1}^{\infty}$ exists. Hence, Async-SGD and Async-SGDI converge with rates $o(1/\sqrt{k})$ and $o(1/k)$, respectively.*

It can be readily verified that Proposition 5 is a generalization of Propositions 3 and 4: For bounded delays, the series $\{a_i\}_{i=1}^{\infty}$ can be chosen as $a_i = 1$ for $i \le T$ and $a_i = 0$ otherwise. For i.i.d. random delay, we could just use the shared distribution as the series, i.e., $a_i = \mathbb{P}(\tau = i)$.

Remark 1 Note that not every realization of the algorithm and corresponding delays satisfy Assumption 4. For instance, if the marginal distribution of the delay τ_k is discrete uniform (i.e. $\mathbb{P}(\tau_k = i) = 1/k$), then by summing (2) from $j = 1$ to k, we have

$$c_1 \ge \frac{\gamma_k M L^2}{2} \sum_{j=1}^{k} \sum_{i=j}^{k} \frac{i}{k} = \frac{\gamma_k M L^2}{2} \frac{1}{k} \frac{k(k+1)(2k+1)}{6} = \frac{\gamma_k M L^2}{2} \frac{(k+1)(2k+1)}{6}.$$

Hence, as long as γ_k has higher order than $1/k^2$, c_1 is infinite and the algorithms would not converge.

Remark 2 Proposition 5 suggests that the algorithms would converge even when delays are *heavy-tailed distributed*. This includes discrete log-normal, discrete T-distribution, discrete Weibull, etc. Here, discrete log-normal means $\mathbb{P}(\tau = i) = \mathbb{P}(i \le x < i + 1)$, where x is random variable with log-normal. Similar distributions can be defined for discrete T-distribution and discrete Weibull.

5. Application examples

In this part, we will present several numerical experiments to further validate our theoretical results.

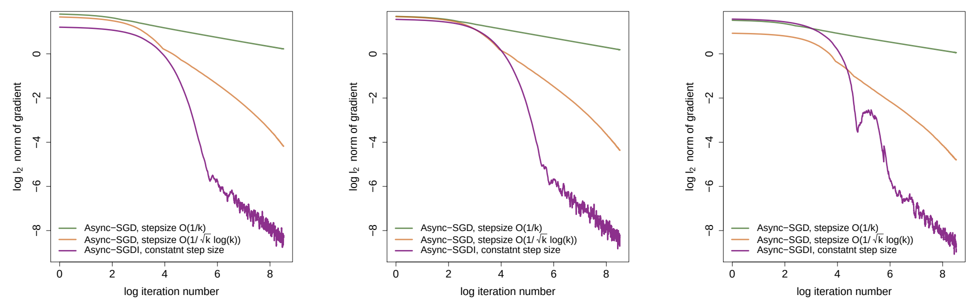
5.1. Low-rank matrix completion

First, we apply the Async-SGD algorithm in solving a low-rank matrix completion problem, where the goal of is to find the matrix X with the lowest rank that matches the expectation of observed symmetric matrices, $\mathbb{E}\{A\}$. This problem could be mathematically formulated as follows:

$$\begin{array}{rl} \text{Minimize} & \mathbb{E}\{\|A - YY^T\|_F^2\} \\ \text{subjet to} & Y \in \mathbb{R}^{n \times p} \end{array}$$

where $X = YY^T$. Using SGD to solve this problem has been investigated in many works (see, e.g., De Sa et al. (2014); Balzano et al. (2010) etc.).

In our experiment, we consider three random delay scenarios: 1) Delay is uniform at random with soupport being the interval [0,20]; 2) i.i.d delay with poisson distribution, Poisson(10); 3) Non i.i.d delay, which we call system delay, is simulated from a virtual system with 10 workers whose computation time t for a gradient follows a hierarchical distribution, $t \sim Exp(\lambda)$ and $\lambda \sim Gamma(2, 1)$. We let the central server update the parameter when it collets M gradients from



(a) Bounded delay, $\tau \leq 20$; (b) Poisson delay,$\tau \sim Poi(10)$; (c) System delay;

Figure 2: Simulation results for the convergence of Async-SGD and Async-SGDI with three kinds of delay on matrix completion problem.

the 10 workers. Note here that for the third delay model, we consider the working time follows a Gamma-Exponential distribution, which are often used for modeling working time. The delay is caused by the difference between the working times. In addition, for the three scenarios, the delay is 0 in the first iteration. And the delays of M gradients in each iteration are different. But in each iteration, the delays are generated following the same distribution.

Aysnc-SGD and Aysnc-SGDI are applied on our simulated data: the ground truth is a randomly generated rank-one matrix $\mathbb{E}(A)$ and the observed samples are $\mathbb{E}(A)+\epsilon$, where the random variable ϵ is drawn from $N(0,1)$. For Async-SGD, we consider two sets of step-sizes. The first one is chosen as $\{1\times 10^{-6}, \frac{1}{2}\times 10^{-6}, \frac{1}{3}\times 10^{-6}, \ldots\}$, decaying every 10 iterations, which can be viewed as $O(1/k)$. The second one is $\{1\times 10^{-6}, \frac{1}{2\log(2)}\times 10^{-6}, \frac{1}{3\log(3)}\times 10^{-6}, \ldots\}$, also decaying every 10 iterations. It satisfies the $O(1/(k^{1/2}\log(k)))$ step-size bound in Proposition 1. We choose the batch size M as 100. For Async-SGDI, we choose fixed step-size as 10^{-6} and increase the batch size as $\{100, 400, 900, \ldots\}$ every 100 iterations. We run both algorithms 5000 iterations and illustrate the convergence behaviors of these two schemes with the ℓ_2 norm of the gradients in Figure 2.

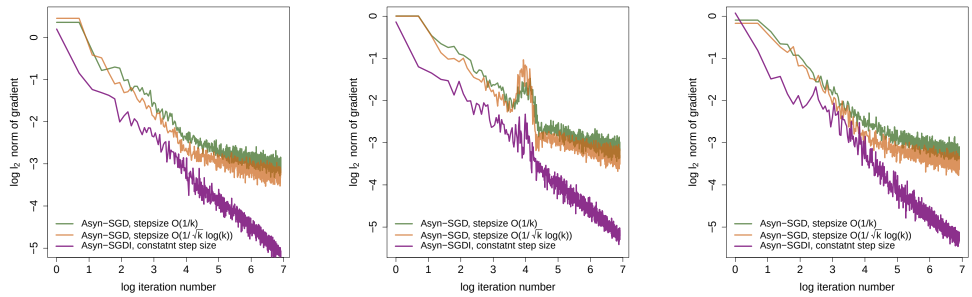
In Figure 2, both Async-SGD algorithms with the three different types of random gradient delay variables are convergent. We can see that the Async-SGDI algorithm has the fastest convergence speed. Async-SGD with $O(1/(k^{1/2}\log(k)))$ step-size is faster than that with $O(1/k)$ step-size. This result is consistent with our theoretical analysis.

5.2. Maximum likelihood estimation for multivariate normal covariance matrix

The second problem we experimented is the maximum likelihood estimation for the covariance matrix of a multivariate normal distribution. This problem can be formulated as:

$$\begin{aligned} \underset{\Sigma\in\mathbb{R}^{d\times d}}{\text{Minimize}} \quad & \ln|\Sigma| + \frac{1}{n}\sum_{i=1}^{n}(x_i-\mu)^T\Sigma^{-1}(x_i-\mu) \\ \text{subjet to} \quad & \Sigma \succ 0, \end{aligned}$$

where Σ is the covariance matrix to be estimated, μ is the mean vector and x_i are the samples. The gradient for this problem has been derived in Minka (2000). We randomly generate data from a



(a) Bounded delay, $\tau \le 50$; (b) Poisson delay, $\tau \sim Poi(30)$; (c) System delay;

Figure 3: Simulation results for the convergence of Async-SGD and Async-SGDI with three kinds of delay on MLE for MVN covariance matrix.

multivariate normal distribuion with mean $\mu^T = (0, 0, 0, 0, 0)$ and covariance matrix

$$\Sigma = \begin{bmatrix} 12.46 & 3.99 & 5.48 & 2.71 & 2.95 \\ 3.99 & 14.99 & 4.74 & 2.42 & 4.64 \\ 5.48 & 4.74 & 12.72 & 1.68 & 2.80 \\ 2.71 & 2.42 & 1.68 & 16.15 & 3.82 \\ 2.95 & 4.64 & 2.80 & 3.82 & 19.38 \end{bmatrix}.$$

Again, we apply Async-SGD and Async-SGDI on the simulated data with three different random gradient delay models as defined in Section 5.1: a) bounded by 50; b) Poisson(30); and c) System delay. For Async-SGD, we choose batch size M as 100 and consider two sets of step-sizes. The first step-size is chosen as $\{1\times10^{-3}, \frac{1}{2}\times10^{-3}, \frac{1}{3}\times10^{-3}, \ldots\}$, decaying every 50 iterations. And the second step-size is $\{1\times10^{-3}, \frac{1}{2\log(2)}\times10^{-3}, \frac{1}{3\log(3)}\times10^{-3}, \ldots\}$, also decaying every 50 iterations. For Async-SGDI, we choose step-size as 0.001 and increase the batch size as $10k^2$, with every $100k$ iterations. We illustrate the convergence results of these two Async-SGD schemes with the ℓ_2 norm of gradient in Figure 3. From Figure 3, we can observe similar results. The ℓ_2 norm of the gradients are decreasing as the number of iterations increases, regardless of the choice of random delay models. Among the three curves, the one for Async-SGDI converges the fastest and Async-SGD with $O(1/k)$ step-size is the slowest. These results confirms our theoretical analysis.

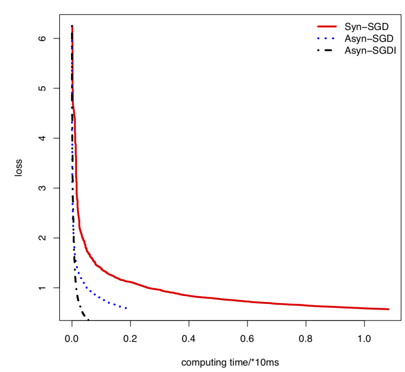


Figure 4: Runtime comparison for Sync-SGD, Async-SGD and Async-SGDI.

5.3. Runtime comparisons for Sync-SGD, Async-SGD and Async-SGDI

Here, we compare the runtime for the three algorithm: Sync-SGD, Async-SGD and Async-SGDI. We experiment these three algorithms using the covariance matrix estimation problem mentioned in

Section 5.2. The true parameter matrix Σ is

$$\Sigma = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.4 \end{bmatrix}.$$

We consider the same system delay model for Asynchronous algorithms, i.e., simulated on a virtual system with 10 workers whose computation time t for finding gradients follows a hierarchical distribution, $t \sim Exp(\lambda)$ and $\lambda \sim Gamma(2, 1)$. For Sync-SGD and Async-SGD, the batch size M is 100 and step-size is $\frac{0.1}{\sqrt{k}log(k+1)}$. For Async-SGDI, the batch size $n_k M$ is $100k^2$. The results are shown in Figure 4. We can see from Figure 4 that Sync-SGD converges slowest because it needs to let parameter sever wait gradients from all workers. As for Asynchronous algorithms, Async-SGDI is a little bit faster. This is consistent with our theoretical work.

6. Conclusion

In this paper, we analyzed the convergence of two asynchronous stochastic gradient descent methods, namely Async-SGD and Async-SGDI, for non-convex optimization problems. By constructing a Lyapunov function that combines optimality error and asynchronicity errors, we proved a convergence rate $o(1/\sqrt{k})$ for Async-SGD and a convergence rate $o(1/k)$ for Async-SGDI, respectively. We note that both convergence results are much stronger compared to previous work. Also, we developed a generalized and more relaxed sufficient condition on gradient update delay for Async-SGD's convergence. This condition provides a unifying framework that includes the major delay models in the existing works as special cases. Collectively, our results in this paper advance the understanding of the convergence performance of Async-SGD for non-convex learning with unbounded gradient update delay. Our future work may involve pursuing non-asymptotic convergence analysis with similar weak assumptions, as well as adaptive batch-size selection strategy for Async-SGDI to increase the computation speed.

References

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283, 2016.

Alekh Agarwal and John C Duchi. Distributed delayed stochastic optimization. In *Advances in Neural Information Processing Systems*, pages 873–881, 2011.

Laura Balzano, Robert Nowak, and Benjamin Recht. Online identification and tracking of subspaces from highly incomplete information. In *Communication, Control, and Computing (Allerton), 2010 48th Annual Allerton Conference on*, pages 704–711. IEEE, 2010.

Jeremy Bernstein, Kamyar Azizzadenesheli, Yu-Xiang Wang, and Anima Anandkumar. Convergence rate of sign stochastic gradient descent for non-convex functions, 2018. URL https://openreview.net/forum?id=HyxjwgbRZ.

Coralia Cartis, Nicholas IM Gould, and Ph L Toint. On the complexity of steepest descent, newton's and regularized newton's methods for nonconvex unconstrained optimization problems. *Siam journal on optimization*, 20(6):2833–2852, 2010.

Patrick L Combettes and Jean-Christophe Pesquet. Stochastic quasi-fejér block-coordinate fixed point iterations with random sweeping. *SIAM Journal on Optimization*, 25(2):1221–1248, 2015.

Christopher De Sa, Kunle Olukotun, and Christopher Ré. Global convergence of stochastic gradient descent for some non-convex matrix problems. *arXiv preprint arXiv:1411.1134*, 2014.

Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Andrew Senior, Paul Tucker, Ke Yang, Quoc V Le, et al. Large scale distributed deep networks. In *Advances in neural information processing systems*, pages 1223–1231, 2012.

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

Serge Gratton, Annick Sartenaer, and Philippe L Toint. Recursive trust-region methods for multiscale nonlinear optimization. *SIAM Journal on Optimization*, 19(1):414–444, 2008.

Robert Hannah and Wotao Yin. On unbounded delays in asynchronous parallel fixed-point algorithms. *arXiv preprint arXiv:1609.04746*, 2016.

Zhouyuan Huo and Heng Huang. Asynchronous mini-batch gradient descent with variance reduction for non-convex optimization. In *AAAI*, pages 2043–2049, 2017.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in neural information processing systems*, pages 315–323, 2013.

Jack Kiefer and Jacob Wolfowitz. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, pages 462–466, 1952.

Mu Li, David G Andersen, Jun Woo Park, Alexander J Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J Shekita, and Bor-Yiing Su. Scaling distributed machine learning with the parameter server. In *OSDI*, volume 1, page 3, 2014.

Xiangru Lian, Yijun Huang, Yuncheng Li, and Ji Liu. Asynchronous parallel stochastic gradient for nonconvex optimization. In *Advances in Neural Information Processing Systems*, pages 2737–2745, 2015.

Thomas P Minka. Old and new matrix algebra useful for statistics. *See www. stat. cmu. edu/minka/papers/matrix. html*, 2000.

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.

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.

Damon Poeter. Gordon Moore Predicts 10 More Years for Moore's Law, 2015. URL https://www.pcmag.com/article2/0,2817,2484098,00.asp.

Daniel Povey, Xiaohui Zhang, and Sanjeev Khudanpur. Parallel training of dnns with natural gradient and parameter averaging. *arXiv preprint arXiv:1410.7455*, 2014.

Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in neural information processing systems*, pages 693–701, 2011.

Sashank J Reddi, Ahmed Hefny, Suvrit Sra, Barnabas Poczos, and Alexander J Smola. On variance reduction in stochastic gradient descent and its asynchronous variants. In *Advances in Neural Information Processing Systems*, pages 2647–2655, 2015.

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.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162(1-2):83–112, 2017.

Suvrit Sra, Adams Wei Yu, Mu Li, and Alexander J Smola. Adadelay: Delay adaptive distributed stochastic convex optimization. *arXiv preprint arXiv:1508.05003*, 2015.

Tao Sun, Robert Hannah, and Wotao Yin. Asynchronous coordinate descent under more realistic assumptions. In *Advances in Neural Information Processing Systems*, pages 6183–6191, 2017.

Yangyang Xu, Ruru Hao, Wotao Yin, and Zhixun Su. Parallel matrix factorization for low-rank tensor completion. *arXiv preprint arXiv:1312.1254*, 2013.

Hsiang-Fu Yu, Cho-Jui Hsieh, Si Si, and Inderjit Dhillon. Scalable coordinate descent approaches to parallel matrix factorization for recommender systems. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, pages 765–774. IEEE, 2012.

Hsiang-Fu Yu, Cho-Jui Hsieh, Si Si, and Inderjit S Dhillon. Parallel matrix factorization for recommender systems. *Knowledge and Information Systems*, 41(3):793–819, 2014.

Jian-Pei Zhang, Zhong-Wei Li, and Jing Yang. A parallel svm training algorithm on large-scale classification problems. In *Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on*, volume 3, pages 1637–1641. IEEE, 2005.

Shuxin Zheng, Qi Meng, Taifeng Wang, Wei Chen, Nenghai Yu, Zhi-Ming Ma, and Tie-Yan Liu. Asynchronous stochastic gradient descent with delay compensation. In *International Conference on Machine Learning*, pages 4120–4129, 2017.

Appendix A. Proofs of Lemma 1 and Theorem 1

Lemma 1 can be proved as follows: Define $\mathscr{F}^k = \sigma\langle x_1, x_2, \dots, x_k; \tau_1, \dots, \tau_k\rangle$, where x_i is the i-th iterate, τ_i is the delay in i-th iteration.

$$
\begin{aligned}
&\mathbb{E}(f(x_{k+1}) - f(x_k)|\mathscr{F}^k) \overset{(a)}{\leq} \mathbb{E}(\langle \nabla f(x_k), x_{k+1} - x_k\rangle|\mathscr{F}^k) + \mathbb{E}(\frac{L}{2}\|x_{k+1} - x_k\|^2|\mathscr{F}^k)\\
&= -\gamma_k \mathbb{E}(\langle \nabla f(x_k), \sum_{m=1}^{M} G(x_{k-\tau_{k,m}}; \xi_{k,m})\rangle|\mathscr{F}^k) + \frac{\gamma_k^2 L}{2}\mathbb{E}(\|\sum_{m=1}^{M} G(x_{k-\tau_{k,m}}; \xi_{k,m})\|^2|\mathscr{F}^k)\\
&\overset{(b)}{=} -\gamma_k \sum_{m=1}^{M} \mathbb{E}(\langle \nabla f(x_k), \nabla f(x_{k-\tau_{k,m}})\rangle|\mathscr{F}^k) + \frac{\gamma_k^2 L}{2}\mathbb{E}(\|\sum_{m=1}^{M} G(x_{k-\tau_{k,m}}; \xi_{k,m})\|^2|\mathscr{F}^k)\\
&\overset{(c)}{\leq} -\frac{\gamma_k}{2}\sum_{m=1}^{M}\mathbb{E}(\|\nabla f(x_k)\|^2 + \|\nabla f(x_{k-\tau_{k,m}})\|^2 - \|\nabla f(x_k) - \nabla f(x_{k-\tau_{k,m}})\|^2|\mathscr{F}^k)\\
&+ \frac{\gamma_k^2 L}{2}(M\sigma^2 + M\sum_{m=1}^{M}\mathbb{E}(\|\nabla f(x_{k-\tau_{k,m}})\|^2|\mathscr{F}^k))\\
&= -\frac{\gamma_k M}{2}\|\nabla f(x_k)\|^2 + (\frac{\gamma_k^2 LM}{2} - \frac{\gamma_k}{2})\sum_{m=1}^{M}\mathbb{E}(\|\nabla f(x_{k-\tau_{k,m}})\|^2|\mathscr{F}^k)\\
&+ \frac{\gamma_k}{2}\sum_{m=1}^{M}\mathbb{E}(\|\nabla f(x_k) - \nabla f(x_{k-\tau_{k,m}})\|^2|\mathscr{F}^k) + \frac{\gamma_k^2 LM\sigma^2}{2}\\
&\overset{(d)}{\leq} -\frac{\gamma_k M}{2}\|\nabla f(x_k)\|^2 + (\frac{\gamma_k^2 LM}{2} - \frac{\gamma_k}{2})\sum_{m=1}^{M}\mathbb{E}(\|\nabla f(x_{k-\tau_{k,m}})\|^2|\mathscr{F}^k)\\
&+ \frac{\gamma_k L^2}{2}\sum_{m=1}^{M}\mathbb{E}(\|x_k - x_{k-\tau_{k,m}}\|^2|\mathscr{F}^k) + \frac{\gamma_k^2 LM\sigma^2}{2}\\
&\overset{(e)}{=} -\frac{\gamma_k M}{2}\|\nabla f(x_k)\|^2 + (\frac{\gamma_k^2 LM}{2} - \frac{\gamma_k}{2})\sum_{m=1}^{M}\mathbb{E}(\|\nabla f(x_{k-\tau_{k,m}})\|^2|\mathscr{F}^k)\\
&+ \frac{\gamma_k L^2}{2}\sum_{m=1}^{M}\sum_{j=1}^{k}((\sum_{i=j}^{k} i\mathbb{P}(\tau_{k,m} = i))\|x_{k+1-j} - x_{k-j}\|^2) + \frac{\gamma_k^2 LM\sigma^2}{2}.
\end{aligned}
\tag{8}
$$

The inequality (a) is due to Lipchitz condition for ∇f; the equality (b) is due to Assumption 4; the inequality (c) is similar to the part in the proof of Lian et al. (2015); the inequality (d) is because of Lipchitz condition; the equality (e) follows from the expectation of $\|x_k - x_{k-\tau_{k,m}}\|^2$:

$$
\begin{aligned}
&\mathbb{E}(\|x_k - x_{k-\tau_{k,m}}\|^2|\mathscr{F}^k) \overset{(a)}{=} \sum_{i=1}^{k}\mathbb{P}(\tau_{k,m} = i)\|\sum_{j=1}^{i} x_{k+1-j} - x_{k-j}\|^2\\
&\leq \sum_{i=1}^{k}\mathbb{P}(\tau_{k,m} = i)i\sum_{j=1}^{i}\|x_{k+1-j} - x_{k-j}\|^2
\end{aligned}
$$

$$= \sum_{j=1}^{k} \sum_{i=j}^{k} \mathbb{P}(\tau_{k,m} = i) i \|x_{k+1-j} - x_{k-j}\|^2,$$

where the equality (a) follows from telescoping. Consider the Lyapurov function:

$$\zeta^k = f(x_k) - f(x^*) + \sum_{j=1}^{\infty} c_j \|x_{k+1-j} - x_{k-j}\|^2, \tag{9}$$

for which we have:

$$\begin{aligned}
&\mathbb{E}(\zeta^{k+1}|\mathscr{F}^k) = \mathbb{E}(f(x_{k+1}) - f(x^*) + \sum_{j=1}^{k+1} c_j \|x_{k+2-j} - x_{k+1-j}\|^2 |\mathscr{F}^k) \\
&\quad = \mathbb{E}(f(x_{k+1}) - f(x^*) + c_1 \|x_{k+1} - x_k\|^2 + \sum_{j=1}^{k} c_{j+1} \|x_{k+1-j} - x_{k-j}\|^2 |\mathscr{F}^k) \\
&\quad \overset{(a)}{\leq} f(x_k) - \frac{\gamma_k M}{2} \|\nabla f(x_k)\|^2 + (\frac{\gamma_k^2 LM}{2} - \frac{\gamma_k}{2}) \sum_{m=1}^{M} \mathbb{E}(\|\nabla f(x_{k-\tau_{k,m}})\|^2 |\mathscr{F}^k) \\
&\quad + \frac{\gamma_k L^2}{2} \sum_{m=1}^{M} \sum_{j=1}^{k} ((\sum_{i=j}^{k} i \mathbb{P}(\tau_{k,m} = i)) \|x_{k+1-j} - x_{k-j}\|^2) + \frac{\gamma_k^2 LM\sigma^2}{2} \\
&\quad - f(x^*) + c_1 \mathbb{E}(\|x_{k+1} - x_k\|^2 |\mathscr{F}^k) + \sum_{j=1}^{k} c_{j+1} \|x_{k+1-j} - x_{k-j}\|^2 \\
&\quad = f(x_k) - f(x^*) + \sum_{j=1}^{k} c_{j+1} \|x_{k+1-j} - x_{k-j}\|^2 + c_1 \mathbb{E}(\|x_{k+1} - x_k\|^2 |\mathscr{F}^k) \\
&\quad - \frac{\gamma_k M}{2} \|\nabla f(x_k)\|^2 + (\frac{\gamma_k^2 LM}{2} - \frac{\gamma_k}{2}) \sum_{m=1}^{M} \mathbb{E}(\|\nabla f(x_{k-\tau_{k,m}})\|^2 |\mathscr{F}^k) \\
&\quad + \frac{\gamma_k L^2}{2} \sum_{m=1}^{M} \sum_{j=1}^{k} ((\sum_{i=j}^{k} i \mathbb{P}(\tau_{k,m} = i)) \|x_{k+1-j} - x_{k-j}\|^2) + \frac{\gamma_k^2 LM\sigma^2}{2} \\
&\quad = f(x_k) - f(x^*) + \sum_{j=1}^{k} c_{j+1} \|x_{k+1-j} - x_{k-j}\|^2 + c_1 \gamma_k^2 \mathbb{E}(\| \sum_{m=1}^{M} G(x_{k-\tau_{k,m}}; \xi_m) \|^2 |\mathscr{F}^k) \\
&\quad - \frac{\gamma_k M}{2} \|\nabla f(x_k)\|^2 + (\frac{\gamma_k^2 LM}{2} - \frac{\gamma_k}{2}) \sum_{m=1}^{M} \mathbb{E}(\|\nabla f(x_{k-\tau_{k,m}})\|^2 |\mathscr{F}^k) \\
&\quad + \frac{\gamma_k L^2}{2} \sum_{m=1}^{M} \sum_{j=1}^{k} ((\sum_{i=j}^{k} i \mathbb{P}(\tau_{k,m} = i)) \|x_{k+1-j} - x_{k-j}\|^2) + \frac{\gamma_k^2 LM\sigma^2}{2} \\
&\quad \leq f(x_k) - f(x^*) + \sum_{j=1}^{k} c_{j+1} \|x_{k+1-j} - x_{k-j}\|^2 + c_1 \gamma_k^2 (M\sigma^2 + M \sum_{m=1}^{M} \mathbb{E}(\|\nabla f(x_{k-\tau_{k,m}})\|^2))
\end{aligned}$$

$$
\begin{aligned}
&- \frac{\gamma_k M}{2}\|\nabla f(x_k)\|^2 + (\frac{\gamma_k^2 LM}{2} - \frac{\gamma_k}{2})\sum_{m=1}^{M}\mathbb{E}(\|\nabla f(x_{k-\tau_{k,m}})\|^2|\mathscr{F}^k) \\
&+ \frac{\gamma_k L^2}{2}\sum_{m=1}^{M}\sum_{j=1}^{k}((\sum_{i=j}^{k} i\mathbb{P}(\tau_{k,m}=i))\|x_{k+1-j}-x_{k-j}\|^2) + \frac{\gamma_k^2 LM\sigma^2}{2} \\
&= f(x_k) - f(x^*) + \sum_{j=1}^{k}(c_{j+1} + \frac{\gamma_k L^2}{2}\sum_{m=1}^{M}(\sum_{i=j}^{k} i\mathbb{P}(\tau_{k,m}=i))\|x_{k+1-j}-x_{k-j}\|^2 \\
&- \frac{\gamma_k M}{2}\|\nabla f(x_k)\|^2 + (c_1\gamma_k^2 M + \frac{L\gamma_k^2 M}{2})\sigma^2 \\
&+ (Mc_1\gamma_k^2 + \frac{ML\gamma_k^2}{2} - \frac{\gamma_k}{2})\sum_{m=1}^{M}\mathbb{E}(\|\nabla f(x_{k-\tau_{k,m}})\|^2|\mathscr{F}^k),
\end{aligned}
$$

where the inequality (a) follows from (8). Using the the assumption $\gamma_k \leq \frac{1}{2Mc_1+ML}$, we have:

$$
\begin{aligned}
\mathbb{E}(\zeta^{k+1}|\mathscr{F}^k) = f(x_k) - f(x^*) + \sum_{j=1}^{k}(c_{j+1} + \frac{\gamma_k ML^2}{2}\sum_{i=j}^{k} i\mathbb{P}(\tau_k=i))\|x_{k+1-j}-x_{k-j}\|^2 \\
- \frac{\gamma_k M}{2}\|\nabla f(x_k)\|^2 + (c_1\gamma_k^2 M + \frac{L\gamma_k^2 M}{2})\sigma^2,
\end{aligned}
$$

which implies that

$$
\begin{aligned}
&\mathbb{E}(\zeta^{k+1}|\mathscr{F}^k) + \frac{\gamma_k M}{2}||\nabla f(x_k)||^2 = f(x_k) - f(x^*) + \\
&\qquad \sum_{j=1}^{k}(c_{j+1} + \frac{\gamma_k ML^2}{2}\sum_{i=j}^{k} i\mathbb{P}(\tau_k=i))\|x_{k+1-j}-x_{k-j}\|^2 + (c_1\gamma_k^2 M + \frac{L\gamma_k^2 M}{2})\sigma^2
\end{aligned}
$$

Next, using Assumption 4, we have:

$$
\mathbb{E}(\zeta^{k+1}|\mathscr{F}^k) + \frac{\gamma_k M}{2}\|\nabla f(x_k)\|^2 \leq \zeta^k + (c_1\gamma_k^2 M + \frac{L\gamma_k^2 M}{2})\sigma^2.
$$

This completes the proof of Lemma 1.

To finish the proof of Theorem 1, we now invoke with the Lemma 1 and use the following supermartingale convergence theorem, which has been used in (Hannah and Yin, 2016; Combettes and Pesquet, 2015):

Theorem 3 (Hannah and Yin (2016); Combettes and Pesquet (2015)) *Let α^k, θ^k and η^k be positive sequences adapted to $\mathscr{F}^k$, and let η^k be summable with probably 1. If*

$$
\mathbb{E}[\alpha^{k+1}|\mathscr{F}^k] + \theta^k \leq \alpha^k + \eta^k,
$$

then with probability 1, α^k converages to a $[0,\infty)$-valued random variable, and $\sum_{k=1}^{\infty}\theta^k < \infty$.

Applying above theorem with $\alpha^k = \zeta^k$, $\theta^k = \frac{\gamma_k M}{2}\|\nabla f(x_k)\|^2$ and $\eta^k = (c_1\gamma_k^2 M + \frac{L\gamma_k^2 M}{2})\sigma^2$, we have $\sum_{k=1}^{\infty}\frac{\gamma_k M}{2}\|\nabla f(x_k)\|^2 < \infty$ with probability 1. Thus $\mathbb{E}\{\sum_{k=1}^{\infty}\frac{\gamma_k M}{2}\|\nabla f(x_k)\|^2\} < \infty$, which implies that $\mathbb{E}\{\|\nabla f(x_k)\|^2\} \to 0$. This completes the proof.

Appendix B. Proofs of Lemma 2 and Theorem 2

Similar to Lemma 1, Lemma 2 could be proved as follows:

$$
\begin{aligned}
&\mathbb{E}(f(x_{k+1}) - f(x_k)|\mathscr{F}^k) \leq \mathbb{E}(\langle \nabla f(x_k), x_{k+1} - x_k\rangle|\mathscr{F}^k) + \mathbb{E}(\frac{L}{2}\|x_{k+1} - x_k\|^2|\mathscr{F}^k) \\
&= -\gamma_k \mathbb{E}(\langle \nabla f(x_k), \frac{1}{n_k}\sum_{m=1}^{n_k M} G(x_{k-\tau_{k,m}}; \xi_{k,m})\rangle|\mathscr{F}^k) + \frac{\gamma_k^2 L}{2}\mathbb{E}(\|\frac{1}{n_k}\sum_{m=1}^{n_k M} G(x_{k-\tau_{k,m}}; \xi_{k,m})\|^2|\mathscr{F}^k) \\
&= -\gamma_k \frac{1}{n_k}\sum_{m=1}^{n_k M} \mathbb{E}(\langle \nabla f(x_k), \nabla f(x_{k-\tau_{k,m}})\rangle|\mathscr{F}^k) + \frac{\gamma_k^2 L}{2n_k}\mathbb{E}(\|\sum_{m=1}^{n_k M} G(x_{k-\tau_{k,m}}; \xi_{k,m})\|^2|\mathscr{F}^k) \\
&\leq -\frac{\gamma_k M}{2}\mathbb{E}(\|\nabla f(x_k)\|^2 + \|\nabla f(x_{k-\tau_k})\|^2 - \|\nabla f(x_k) - \nabla f(x_{k-\tau_k})\|^2|\mathscr{F}^k) \\
&+ \frac{\gamma_k^2 L}{2n_k^2}(n_k M\sigma^2 + (n_k M)^2\mathbb{E}(\|\nabla f(x_{k-\tau_k})\|^2|\mathscr{F}^k)) \\
&= -\frac{\gamma_k M}{2}\|\nabla f(x_k)\|^2 + (\frac{\gamma_k^2 L M^2}{2} - \frac{\gamma_k M}{2})\mathbb{E}(\|\nabla f(x_{k-\tau_k})\|^2|\mathscr{F}^k) \\
&+ \frac{\gamma_k M}{2}\mathbb{E}(\|\nabla f(x_k) - \nabla f(x_{k-\tau_k})\|^2|\mathscr{F}^k) + \frac{\gamma_k^2 L M\sigma^2}{2n_k} \\
&\leq -\frac{\gamma_k M}{2}\|\nabla f(x_k)\|^2 + (\frac{\gamma_k^2 L M^2}{2} - \frac{\gamma_k M}{2})\mathbb{E}(\|\nabla f(x_{k-\tau_k})\|^2|\mathscr{F}^k) \\
&+ \frac{\gamma_k L^2 M}{2}\mathbb{E}(\|x_k - x_{k-\tau_k}\|^2|\mathscr{F}^k) + \frac{\gamma_k^2 L M\sigma^2}{2n_k} \\
&= -\frac{\gamma_k M}{2}\|\nabla f(x_k)\|^2 + (\frac{\gamma_k^2 L M^2}{2} - \frac{\gamma_k M}{2})\mathbb{E}(\|\nabla f(x_{k-\tau_k})\|^2|\mathscr{F}^k) \\
&+ \frac{\gamma_k L^2 M}{2}\sum_{j=1}^{k}((\sum_{i=j}^{k} i\mathbb{P}(\tau_k = i))\|x_{k+1-j} - x_{k-j}\|^2) + \frac{\gamma_k^2 L M\sigma^2}{2n_k}
\end{aligned}
$$

Again, using the Lyapurov function, we have:

$$
\zeta^k = f(x_k) - f(x^*) + \sum_{j=1}^{\infty} c_j\|x_{k+1-j} - x_{k-j}\|^2, \tag{10}
$$

we have for ζ^{k+1}:

$$
\begin{aligned}
&\mathbb{E}(\zeta^{k+1}|\mathscr{F}^k) = \mathbb{E}(f(x_{k+1}) - f(x^*) + \sum_{j=1}^{k+1} c_j\|x_{k+2-j} - x_{k+1-j}\|^2|\mathscr{F}^k) \\
&= \mathbb{E}(f(x_{k+1}) - f(x^*) + c_1\|x_{k+1} - x_k\|^2 + \sum_{j=1}^{k} c_{j+1}\|x_{k+1-j} - x_{k-j}\|^2|\mathscr{F}^k) \\
&\leq f(x_k) - \frac{\gamma_k M}{2}\|\nabla f(x_k)\|^2 + (\frac{\gamma_k^2 L M^2}{2} - \frac{\gamma_k M}{2})\mathbb{E}(\|\nabla f(x_{k-\tau_k})\|^2|\mathscr{F}^k) \\
&+ \frac{\gamma_k L^2 M}{2}\sum_{j=1}^{k}((\sum_{i=j}^{k} i\mathbb{P}(\tau_k = i))\|x_{k+1-j} - x_{k-j}\|^2) + \frac{\gamma_k^2 L M\sigma^2}{2n_k}
\end{aligned}
$$

$$- f(x^*) + c_1\mathbb{E}(\|x_{k+1} - x_k\|^2|\mathscr{F}^k) + \sum_{j=1}^{k} c_{j+1}\|x_{k+1-j} - x_{k-j}\|^2$$

$$= f(x_k) - f(x^*) + \sum_{j=1}^{k} c_{j+1}\|x_{k+1-j} - x_{k-j}\|^2$$

$$+ c_1\mathbb{E}(\|x_{k+1} - x_k\|^2|\mathscr{F}^k) - \frac{\gamma_k M}{2}\|\nabla f(x_k)\|^2 + (\frac{\gamma_k^2 LM^2}{2} - \frac{\gamma_k M}{2})\mathbb{E}(\|\nabla f(x_{k-\tau_k})\|^2|\mathscr{F}^k)$$

$$+ \frac{\gamma_k L^2 M}{2}\sum_{j=1}^{k}((\sum_{i=j}^{k} i\mathbb{P}(\tau_k = i))\|x_{k+1-j} - x_{k-j}\|^2) + \frac{\gamma_k^2 LM\sigma^2}{2n_k}$$

$$= f(x_k) - f(x^*) + \sum_{j=1}^{k} c_{j+1}\|x_{k+1-j} - x_{k-j}\|^2 + c_1\gamma_k^2\mathbb{E}(\|\frac{1}{n_k}\sum_{m=1}^{n_k M} G(x_{k-\tau_{k,m}};\xi_m)\|^2|\mathscr{F}^k)$$

$$- \frac{\gamma_k M}{2}\|\nabla f(x_k)\|^2 + (\frac{\gamma_k^2 LM^2}{2} - \frac{\gamma_k M}{2})\mathbb{E}(\|\nabla f(x_{k-\tau_k})\|^2|\mathscr{F}^k)$$

$$+ \frac{\gamma_k L^2 M}{2}\sum_{j=1}^{k}((\sum_{i=j}^{k} i\mathbb{P}(\tau_k = i))\|x_{k+1-j} - x_{k-j}\|^2) + \frac{\gamma_k^2 LM\sigma^2}{2n_k}$$

$$\leq f(x_k) - f(x^*) + \sum_{j=1}^{k} c_{j+1}\|x_{k+1-j} - x_{k-j}\|^2 + c_1\frac{\gamma_k^2}{n_k^2}(n_k M\sigma^2 + (n_k M)^2\mathbb{E}(\|\nabla f(x_{k-\tau_{k,m}})\|^2))$$

$$- \frac{\gamma_k M}{2}\|\nabla f(x_k)\|^2 + (\frac{\gamma_k^2 LM^2}{2} - \frac{\gamma_k M}{2})\mathbb{E}(\|\nabla f(x_{k-\tau_{k,m}})\|^2|\mathscr{F}^k)$$

$$+ \frac{\gamma_k L^2 M}{2}\sum_{j=1}^{k}((\sum_{i=j}^{k} i\mathbb{P}(\tau_k = i))\|x_{k+1-j} - x_{k-j}\|^2) + \frac{\gamma_k^2 LM\sigma^2}{2n_k}$$

$$= f(x_k) - f(x^*) + \sum_{j=1}^{k}(c_{j+1} + \frac{\gamma_k L^2 M}{2}(\sum_{i=j}^{k} i\mathbb{P}(\tau_k = i)))\|x_{k+1-j} - x_{k-j}\|^2$$

$$- \frac{\gamma_k M}{2}\|\nabla f(x_k)\|^2 + (c_1\gamma_k^2 M + \frac{L\gamma_k^2 M}{2})\frac{\sigma^2}{n_k}$$

$$+ (M^2 c_1\gamma_k^2 + \frac{M^2 L\gamma_k^2}{2} - \frac{\gamma_k M}{2})\mathbb{E}(\|\nabla f(x_{k-\tau_k})\|^2|\mathscr{F}^k)$$

Using the assumptions for the step-size, we have:

$$\mathbb{E}(\zeta^{k+1}|\mathscr{F}^k) = f(x_k) - f(x^*) + \sum_{j=1}^{k}(c_{j+1} + \frac{\gamma_k ML^2}{2}\sum_{i=j}^{k} i\mathbb{P}(\tau_k = i))\|x_{k+1-j} - x_{k-j}\|^2$$

$$- \frac{\gamma_k M}{2}\|\nabla f(x_k)\|^2 + (c_1\gamma_k^2 M + \frac{L\gamma_k^2 M}{2})\frac{\sigma^2}{n_k},$$

which further implies that

$$\mathbb{E}(\zeta^{k+1}|\mathscr{F}^k) + \frac{\gamma_k M}{2}\|\nabla f(x_k)\|^2 = f(x_k) - f(x^*) + \\ \sum_{j=1}^{k}(c_{j+1} + \frac{\gamma_k M L^2}{2}\sum_{i=j}^{k} i\mathbb{P}(\tau_k = i))\|x_{k+1-j} - x_{k-j}\|^2 + (c_1\gamma_k^2 M + \frac{L\gamma_k^2 M}{2})\frac{\sigma^2}{n_k}$$

Again, by using Assumption 4, we have:

$$\mathbb{E}(\zeta^{k+1}|\mathscr{F}^k) + \frac{\gamma_k M}{2}\|\nabla f(x_k)\|^2 \le \zeta^k + (c_1\gamma_k^2 M + \frac{L\gamma_k^2 M}{2})\frac{\sigma^2}{n_k}$$

The rest of the proof again follows from the supermartingale convergence theorem, which yields $\mathbb{E}\{\sum_{k=1}^{\infty}\gamma_k\|\nabla f(x_k)\|^2\} < \infty$ and $\mathbb{E}\{\|\nabla f(x_k)\|^2\} \to 0$. This completes the proof.

Appendix C. Proofs for Propositions 1-2

C.1. Proof for Proposition 1

Proof
Firstly, we have following facts.

1. for the p-series $\{\frac{1}{k^p}\}_{k=1}^{\infty}$ and the log-series $\{\frac{1}{k\log^p(k)}\}_{k=2}^{\infty}$, they are both unsummable if $p \le 1$, while summable if $p > 1$;
2. $\{\frac{1}{\sqrt{k}\log(k)}\}_{k=2}^{\infty} = \Omega(1/k)$. Hence$\{\frac{1}{\sqrt{k}\log(k)}\}_{k=2}^{\infty}$ is unsummable;
3. the log-series $\{\frac{1}{k\log(k)}\}_{k=2}^{\infty}$ ($p = 1$) is unsummable, while $\{\frac{1}{k\log^2(k)}\}_{k=2}^{\infty}$ is summable ($p = 2$);

As a result, $\gamma_k = O(\frac{1}{k^{1/2}\log(k)})$ is unsummable and $\gamma_k^2 = O(\frac{1}{k\log^2(k)})$ is summable (from fact 1 and 2). These are condition ii) and iii) in Theorem 1. It then follows from Theorem 1 that $\mathbb{E}[\sum_{k=1}^{\infty}\gamma_k\|\nabla f(x_k)\|^2] < \infty$.

To show the $o(1/\sqrt{k})$ convergence rate, we note that if $\mathbb{E}\{\|\nabla f(x_k)\|^2\} = O(1/\sqrt{k})$, then $\mathbb{E}\{\gamma_k\|\nabla f(x_k)\|^2\} = O(\frac{1}{k\log(k)})$, which is unsummable (from fact 3) and contradict to our earlier conclusion that $\mathbb{E}\{\sum_{k=1}^{\infty}\gamma_k\|\nabla f(x_k)\|^2\} < \infty$. Therefore, $\mathbb{E}\{\|\nabla f(x_k)\|^2\}$ must be $o(1/\sqrt{k})$. ■

C.2. Proof for Propositions 2

Proof Note that the sequence $\{n_k\}_{k=1}^{\infty}$ with $\frac{1}{n_k} = o(\frac{1}{k+\epsilon})$ is a p-series with $p > 1$. Thus, $\{\frac{1}{n_k}\}_{k=1}^{\infty}$ is summable. It then follows from Theorem 2 that $\mathbb{E}\{\sum_{k=1}^{\infty}\gamma_k\|\nabla f(x_k)\|^2\} < \infty$. Since γ_k is a constant step-size, it then follows that $\mathbb{E}\{\sum_{k=1}^{\infty}\|\nabla f(x_k)\|^2\} < \infty$. By contradiction, suppose $\mathbb{E}\{\|\nabla f(x_k)\|^2\} = O(1/k)$, then $\mathbb{E}\{\sum_{k=1}^{\infty}\|\nabla f(x_k)\|^2\}$ is unsummable, a contradicting to Theorem 2. Therefore, we must have $\mathbb{E}(\|\nabla f(x_k)\|^2) = o(1/k)$. ■

Appendix D. Proofs for Propositions 3-5

D.1. Proof of Proposition 3

To prove Proposition 3, i.e., the bounded gradient update delay model, we note that we can define $\{c_i\}_{i=1}^{\infty}$ as follows:

$$
\begin{aligned}
&c_1 \geq c_2 + \frac{\gamma_k L^2}{2}\sum_{i=1}^{T} i \geq c_2 + \frac{\gamma_k L^2}{2}\sum_{i=1}^{T} i\mathbb{P}(\tau_k = i) \\
&c_2 \geq c_3 + \frac{\gamma_k L^2}{2}\sum_{i=2}^{T} i \geq c_3 + \frac{\gamma_k L^2}{2}\sum_{i=2}^{T} i\mathbb{P}(\tau_k = i) \\
&\dots \\
&c_{T-1} \geq c_T + \frac{\gamma_k L^2}{2} T \geq c_T + \frac{\gamma_k L^2}{2}\sum_{i=l}^{T} i\mathbb{P}(\tau_k = i)
\end{aligned}
$$

and $c_l = 0$ for $l \geq T$. Therefore $\{c_i\}_{i=1}^{\infty}$ exists. This completes the proof of Proposition 3.

D.2. Proof of Proposition 4

To prove Proposition 4, we note that since $\{\tau_k\}_{k=1}^{\infty}$ are i.i.d., we can use a random variable τ to denote the common delay distribution. Define $\{c_i\}_{i=1}^{\infty}$ as follows:

$$
\begin{aligned}
&c_1 \geq c_2 + \frac{\gamma_k L^2}{2}\sum_{i=1}^{\infty} i\mathbb{P}(\tau = i) \\
&c_2 \geq c_3 + \frac{\gamma_k L^2}{2}\sum_{i=2}^{\infty} i\mathbb{P}(\tau = i) \\
&\dots \\
&c_l \geq c_{l+1} + \frac{\gamma_k L^2}{2}\sum_{i=l}^{\infty} i\mathbb{P}(\tau = i) \\
&\dots
\end{aligned}
\tag{11}
$$

Similarly, we have $c_1 \geq \frac{\gamma_k L^2}{2}\mathbb{E}(\tau^2)$, $\forall k$. Thus $\{c_i\}_{i=1}^{\infty}$ could be generated by the inequalities in (11). This completes the proof of Proposition 4.

D.3. Proposition 5

To prove Proposition 5, i.e., uniformly upper bounded probability series, we can define the sequence $\{c_i\}_{i=1}^{\infty}$ as follows:

$$
\begin{aligned}
&c_1 \geq c_2 + \frac{\gamma L^2}{2}\sum_{i=1}^{\infty} ia_i \geq c_2 + \frac{\gamma_k L^2}{2}\sum_{i=1}^{\infty} i\mathbb{P}(\tau_k = i), \\
&c_2 \geq c_3 + \frac{\gamma L^2}{2}\sum_{i=2}^{\infty} ia_i \geq c_3 + \frac{\gamma_k L^2}{2}\sum_{i=2}^{\infty} i\mathbb{P}(\tau_k = i),
\end{aligned}
\tag{12}
$$

$$\begin{aligned}
&\dots \\
&c_l \geq c_{l+1} + \frac{\gamma L^2}{2}\sum_{i=l}^{\infty} i a_i \geq c_{l+1} + \frac{\gamma_k L^2}{2}\sum_{i=l}^{\infty} i\mathbb{P}(\tau_k = i), \\
&\dots
\end{aligned}$$

where $\gamma \geq \max \gamma_k$ is a constant. Then, to show that $\{c_i\}_{i=1}^{\infty}$, we only need to prove that c_1 is finite. To this end, summing all the inequalities in (12), we have:

$$\begin{aligned}
c_1 &= \frac{\gamma L^2}{2}\sum_{j=1}^{\infty}\sum_{i=j}^{\infty} i a_i = \frac{\gamma L^2}{2}\sum_{i=1}^{\infty}\sum_{j=1}^{i} i a_i \\
&= \frac{\gamma L^2}{2}\sum_{i=1}^{\infty} i^2 a_i < \infty
\end{aligned}$$

Therefore, we could generate $\{c_i\}_{i=1}^{\infty}$ by the inequalities in (12). This completes the proof of Proposition 5.