Convergence Rates of Accelerated Markov Gradient Descent with Applications in Reinforcement Learning

Thinh T. Doan  Lam M. Nguyen  Nhan H. Pham  Justin Romberg

February 10, 2020

Abstract

Motivated by broad applications in machine learning, we study the popular accelerated stochastic gradient descent (ASGD) algorithm for solving (possibly nonconvex) optimization problems. We characterize the finite-time performance of this method when the gradients are sampled from Markov processes, and hence dependent from time step to time step; in contrast, the analysis in existing work relies heavily on the stochastic gradients being independent. Our main contributions show that under certain (standard) assumptions on the underlying Markov chain generating the gradients, ASGD converges at the nearly the same rate with Markovian gradient samples as with independent gradient samples. The only difference is a logarithmic factor that accounts for the mixing time of the Markov chain.

One of the key motivations for this study are complicated control problems that can be modeled by a Markov decision process and solved using reinforcement learning. We apply the accelerated method to several challenging problems in the OpenAI Gym and Mujoco, and show that acceleration can significantly improve the performance of the classic REINFORCE algorithm.

1 Introduction

In the recent years, one of the most active areas of research related to machine learning is the development of optimization algorithms to train machine learning models (see e.g. [11, 33, 21, 16, 6, 18, 25]). Stochastic gradient descent (SGD), originally introduced in [28] under the name of stochastic approximation, has become the standard for solving optimization problems in machine learning; it can substantially reduce the cost of computing a step direction in supervised learning problems, and offers a framework for systematically handling uncertainty in reinforcement learning. The general problem statement is straightforward: we want to optimize an (unknown) objective function $f$ when queries for the gradient are noisy. At a point $x$, we observe a random vector $G(x, \xi)$ whose mean is the gradient (or in the subgradient) of $f$ at $x$. Through
judicious choice of step sizes, the “noise” induced by this randomness can be averaged out across iterations, and the algorithm converges to stationary point of $f$ \[12, 3, 24, 26\].

To further improve the performance of SGD, stochastic versions of Nesterov’s classic acceleration scheme \[23\] have been studied in different settings \[14, 35, 2, 22\]. In many of these cases, it has been observed that acceleration improves the performance of SGD both in theory \[19, 12, 31, 37, 7\] and in practice \[17\], with a notable application being the training of deep neural networks \[1\].

In this paper, we show that a particular version of accelerated SGD is still ergodic when the gradients of the underlying objective function are sampled from Markov process, and hence are biased and not independent across iterations. This model for the gradients has been considered previously in \[10, 30, 15, 27\], where different variants of stochastic mirror descent are considered. Moreover, it has also been observed that the SGD performs better when the gradients are sampled from Markov process as compared to i.i.d samples in both convex and nonconvex problems \[30\]. This paper shows that the benefits of acceleration extend to the Markovian setting in theory and in practice; we provide theoretical convergence rates that nearly match those in the i.i.d. setting, and show empirically that the algorithm is able to learn from significantly fewer samples on benchmark reinforcement learning problems.

**Main contributions.** We study accelerated stochastic gradient descent where the gradients are sampled from a Markov process. We show that, despite the gradients being biased and dependent across iterations, the convergence rate across many different types of objective functions (convex and smooth, strongly convex, nonconvex and smooth) is within a logarithmic factor of the comparable bounds for independent gradients. This logarithmic factor is naturally related to the mixing time of the underlying Markov process generating the stochastic gradients. To our knowledge, these are the first such bounds for accelerated stochastic gradient descent with Markovian sampling.

We also show that acceleration is extremely effective in practice by applying it to multiple problems in reinforcement learning. Compared with the popular Monte-Carlo policy gradient REINFORCE algorithm, the accelerated algorithm requires significantly fewer samples to learn a policy with comparable rewards, which aligns with our theoretical results.

## 2 Accelerated Markov Gradient Descent

We consider the (possibly nonconvex) optimization problem

$$\min_{x \in \mathcal{X}} f(x),$$

where $\mathcal{X} \subset \mathbb{R}^d$ is a closed convex set and $f : \mathcal{X} \to \mathbb{R}$ is given as

$$f(x) \triangleq \mathbb{E}_\pi [F(x; \xi)] = \int_\Xi F(x; \xi) d\pi(\xi).$$

Here $\Xi$ is a statistical sample space with probability distribution $\pi$ and $F(\cdot, \xi) : \mathcal{X} \to \mathbb{R}$ is a bounded below (possibly nonconvex) function associated with $\xi \in \Xi$. We are interested in the first-order stochastic optimization methods for solving problem \[1\]. Most of existing algorithms, such as the popular stochastic gradient descent (SGD), require a sequence of $\{\xi_k\}$ sampled i.i.d from the distribution $\pi$. Our focus is to consider the case where the samples $\xi$ are generated from an ergodic Markov process, whose stationary distribution is $\pi$. 

2
Algorithm 1 Accelerated Markov Gradient Descent

Initialize: Set arbitrarily \( x_0, \bar{x}_0 \in X \), step sizes \( \{\alpha_k, \beta_k, \gamma_k\} \), and an integer \( K \geq 1 \)

Iterations: For \( k = 1, \ldots, K \) do

\[
y_k = (1 - \beta_k)\bar{x}_{k-1} + \beta_k x_{k-1}
\]

\[
x_k = \arg \min_{x \in X} \left\{ \gamma_k \left[ (G(y_k, \xi_k), x - y_k) + \mu V(y_k, x) \right] + V(x_{k-1}, x) \right\}
\]

\[
\bar{x}_k = (1 - \alpha_k)\bar{x}_{k-1} + \alpha_k x_k
\]

Output: \( \bar{x}_k \)

In this paper, our focus is to study accelerated gradient methods for solving problem (1), which is originally proposed by Nesterov [23] and is studied later in different variants; see for example [19, 12, 14, 35] and the reference therein. In particular, we study an ergodic version of ASGD studied in [19, 20], where the gradients are sampled from a Markov process. We name this algorithm as accelerated Markov gradient descent formally stated in Algorithm 1. Our focus is to derive a finite-time performance of this method, which is unknown in the literature.

In Algorithm 1 \( V \) is the so-called Bregman distance associated with the 1-strongly convex function \( \psi \) defined as

\[
V(y, x) = \psi(x) - \psi(y) - \langle \psi(y), y - x \rangle.
\]

In addition, \( G(x; \xi) \in \partial F(x; \xi) \) is the subgradient of \( F(\cdot; \xi) \) evaluated at the point \( x \). As mentioned we consider the case where the random samples \( \xi_1, \cdots, \xi_k \) are drawn from a Markov ergodic stochastic process. Thus, to provide a finite-time analysis of this algorithm in the next section, we consider the following fairly standard technical assumptions about the Markov process, which are often assumed in the existing literature [10, 30, 29, 5, 8].

Assumption 1. The sequence \( \{\xi_k\} \) is a Markov chain with state space \( S \). In addition, the following limits exit

\[
\lim_{k \to \infty} \mathbb{E}[G(x; \xi_k)] = g(x) \in \partial f(x) \quad \forall x.
\]

Assumption 2. Given a positive constant \( \gamma \), we denote by \( \tau(\gamma) \) the mixing time of the Markov chain \( \{\xi_k\} \). We assume that for \( g(x) \in \partial f(x) \) and \( \forall \xi \in S \)

\[
\| \mathbb{E}[G(x; \xi_k)] - g(x) \| \xi_0 = \xi \| \leq \gamma, \quad \forall k \geq \tau(\gamma).
\]

In addition, the Markov chain \( \{\xi_k\} \) has a geometric mixing time, i.e., there exists a constant \( C \) such that

\[
\tau(\gamma) = C \log \left( \frac{1}{\gamma} \right).
\]
Table 1: Convergence rates of ASGD and its ergodic variant. Here, $M^2$ represents the variance of the noise and the abbreviations C, SC, NC, S, and NS denote for convex, strongly convex, nonconvex, smooth, and nonsmooth, respectively.

| OBJECTIVES | I.I.D | MARKOV |
|------------|-------|--------|
| C & S      | $O\left(\frac{1}{k^2} + \frac{M^2}{k}\right)$ | $O\left(\frac{1}{k^2} + \frac{M^2 \log(k)}{k}\right)$ |
| SC & NS    | $O\left(\frac{1}{k^2} + \frac{M^2}{k}\right)$ | $O\left(\frac{1}{k^2} + \frac{M^2 \log(k)}{k}\right)$ |
| NC & S     | $O\left(\frac{1}{k} + \frac{M^2}{\sqrt{k}}\right)$ | $O\left(\frac{1}{k} + \frac{M^2 \log(k)}{\sqrt{k}}\right)$ |

biased and dependent. However, under Assumption 2 their dependence and bias are very weak at samples spaced out at every $\tau$ step. This explains the $\log(k)$ factor in our result in Theorems 1–3 below. Indeed, our convergence results are the same as the ones of ASGD in both convex and nonconvex case, except for this additional $\log(k)$ factor. A complete comparison between these two settings is presented in Table 1, where we use the results of ASGD under i.i.d samples in [19, 12].

3 Convergence Analysis: Convex Case

We now present the finite-time analysis of Algorithm 1 when the function $f$ is convex. Similar to [30], we make the following assumption on the constraint set $\mathcal{X}$.

**Assumption 3.** The set $\mathcal{X}$ is compact.

The compactness of $\mathcal{X}$ is needed in our analysis to implicitly control the boundedness of the iterates. In addition, to reduce the notation burden we consider $V$ is the Euclidean distance, i.e., $\psi(x) = \frac{1}{2}\|x\|^2$ and $V(y, x) = \frac{1}{2}\|y - x\|^2$. Since $\mathcal{X}$ is compact, given $x_0$ there exist positive constants $D$ and $M$ such that $\forall \xi \in \Xi$

$$D = \max_{x \in \mathcal{X}} \|x - x_0\|^2, \quad \|G(x, \xi)\| \leq M, \forall x \in \mathcal{X}. \quad (9)$$

In addition, let $x^*$ be a solution of (1), i.e.,

$$x^* = \arg \min_{x \in \mathcal{X}} f(x).$$

We assume that $\{\alpha_k, \beta_k\}$ are chosen such that $\alpha_1 = 1$ and

$$\frac{\beta_k(1 - \alpha_k)}{\alpha_k(1 - \beta_k)} = \frac{1}{1 + \mu \gamma_k}, \quad 1 + \mu \gamma_k > L \alpha_k \gamma_k, \quad (10)$$

where $\mu$ and $L$ are some nonnegative constants defined later. In addition, let $\Gamma_k$ be defined as

$$\Gamma_k = \begin{cases} 1, & k \leq 1 \\ (1 - \alpha_k)\Gamma_{k-1} & k \geq 2. \end{cases} \quad (11)$$

Our main results in this section are established based on the following two key lemmas. For an ease of exposition we delay the proofs of these lemmas to Appendix. The first lemma is motivated by the result studied in [19].

---

1 We note that the results in this paper hold for a general Bregman distance defined in [6].
Lemma 1. Let $\alpha_k$ and $\gamma_k$ satisfy (10) and
\[
\frac{\alpha_k}{\gamma_k \Gamma_k} \leq \frac{\alpha_{k-1}(1 + \mu \gamma_{k-1})}{\gamma_{k-1} \Gamma_{k-1}},
\]
where $\Gamma_k$ is defined in (11). Then \{\bar{x}_k\} generated by Algorithm 1 satisfies for all $k \geq 1$
\[
f(\bar{x}_k) - f(x^*) \leq \Gamma_k \gamma_0 f(\bar{x}_0) + \alpha_0 (1 + \mu \gamma_0) D + \Gamma_k \sum_{t=1}^{k} \frac{4M^2 \gamma_t \alpha_t}{\Gamma_t (1 + \mu \gamma_t - L \gamma_t \alpha_t)}
+ \Gamma_k \sum_{t=1}^{k} \frac{\alpha_t}{\Gamma_t} \langle G(y_t; \xi_t) - \nabla f(y_t), z - \bar{x}_{t-1} \rangle,
\]
where $\bar{x}_{k-1}$ is defined as
\[
\bar{x}_{k-1} = \frac{1}{1 + \mu \gamma_k} x_{k-1} + \frac{\mu \gamma_k}{1 + \mu \gamma_k} y_k.
\]

The second lemma is to show the impact of biased gradients.

Lemma 2. Let the sequences \{x_k, y_k\} be generated by Algorithm 1 and $\bar{x}$ is defined in (14). Then we have
\[
\mathbb{E}[\langle G(y_k; \xi_k) - \nabla f(y_k), z - \bar{x}_{k-1} \rangle] \leq (2M^2 + 4\mu M D) \tau(\gamma_k) \gamma_k - \tau(\gamma_k)
+ (2D + 2M^2 + 8\mu M D) \gamma_k.
\]

Note that if the noise sequence $\xi_k$ is i.i.d, then the last term on the right-hand side has conditional expectation being zero. This implies that Lemma 2 is unnecessary. However, since the gradients are sampled from the Markov chain \{\xi_t\}, the samples are dependent and the gradient estimates are biased. We, therefore, have to utilize the geometric mixing time of the Markov chain to provide an upper bound for this cross-term, explaining the factor $\tau(\gamma_k)$ in (15). With these two lemmas, we now present the first main results of this paper, which are the rates of convergence of Algorithm 1 for solving convex and strongly convex problems.

3.1 Smooth Convex Functions

We now study the rates of Algorithm 1 when the function $f$ is only convex and has Lipschitz continuous gradient. In particular, we consider the following assumption.

Assumption 4. There exists a constant $L > 0$ s.t. \forall x, y
\[
\| \nabla f(x) - \nabla f(y) \| \leq L \| x - y \|,
\]
which also implies that
\[
f(y) - f(x) - \langle \nabla f(x), y - x \rangle \leq \frac{L}{2} \| y - x \|^2.
\]

In this case $\partial f(\cdot) = \nabla f(\cdot)$ and $\mu = 0$. Since $\mu = 0$ we have by (10) that $\beta_k = \alpha_k$. Our first result about the convergence rate of Algorithm 1 for optimizing a smooth convex function under Markovian samples is given below.
Theorem 1. Let Assumptions 1–4 hold. Suppose that the step sizes are chosen as
\[
\alpha_k = \frac{2}{k+1}, \quad \gamma_k = \frac{1}{2L\sqrt{k+1}}. \tag{18}
\]
Then we have for all \(k \geq 1\)
\[
f(\bar{x}_k) - f(x^*) \leq f(\bar{x}_0) + 4LD \cdot \frac{1}{2k} + 2(D + 3M^2) \frac{1}{\sqrt{k}} + \frac{2M^2(L + 1)\log(2L\sqrt{k})}{L} \frac{1}{\sqrt{k}}. \tag{19}
\]

Proof. First, using (18) and (11) we have \(\Gamma_0 = 1, \alpha_0 = 2,\) and \(\gamma_0 = 1/2L\). Note that the conditions in (18) satisfy (10) and (12) with \(\mu = 0\). Thus, using the quantities above, \(\mu = 0\), and (15) into (13) gives
\[
f(\bar{x}_k) - f(x^*) \leq \left[ f(\bar{x}_0) + 4LD \right] \Gamma_k + \Gamma_k \sum_{t=1}^{k} \frac{4M^2\gamma_t\alpha_t}{\Gamma_t(1 + \mu\gamma_t - L\gamma_t\alpha_t)}
+ 2M^2\Gamma_k \sum_{t=1}^{k} \frac{\alpha_t\gamma_t}{\Gamma_t} + 2(D + M^2)\Gamma_k \sum_{t=1}^{k} \frac{\alpha_t\gamma_t}{\Gamma_t}. \tag{20}
\]
We next analyze each term on the right-hand side of (20). Using (11) and (18) we immediately have
\[
\Gamma_k = \frac{2}{k(k+1)}, \quad \frac{1}{\gamma_k} - L\alpha_k \geq 2L\sqrt{k}, \tag{21}
\]
which yields
\[
\sum_{t=1}^{k} \frac{\alpha_t\gamma_t}{\Gamma_t} = \sum_{t=1}^{k} \frac{t}{2L\sqrt{t} + 1} \leq \sum_{t=1}^{\sqrt{t}} \frac{1}{2L} \leq \frac{k^{3/2}}{3L}. \tag{22}
\]
Next, we consider
\[
\sum_{t=1}^{k} \frac{\alpha_t\gamma_t}{\Gamma_t(1 - L\gamma_t\alpha_t)} \leq \sum_{t=1}^{k} \frac{1}{\gamma_t - L\alpha_t} \leq \sum_{t=1}^{k} \frac{1}{2L\sqrt{t}} \leq \frac{k^{3/2}}{3L}. \tag{23}
\]
By (8) we have \(\tau(\gamma_k) = \log(2L\sqrt{k})\) which implies that
\[
\frac{k\tau_k}{\sqrt{k} - \tau(\gamma_k)} \leq 2(L + 1)\tau(\gamma_k)\sqrt{k - \tau(\gamma_k)} \leq 2(L + 1)\tau(\gamma_k)\sqrt{k},
\]
where we use the fact that \(2Lk \geq \tau(\gamma_k)\) in the first inequality. Thus, by (18) and the previous relations we have
\[
\sum_{t=1}^{k} \frac{\alpha_t\tau(\gamma_t)\gamma_t - \tau(\gamma_t)}{\Gamma_t} = \sum_{t=1}^{k} \frac{\tau(\gamma_t)\gamma_t}{2L\sqrt{t} - \tau(\gamma_t)}
\leq \frac{L + 1}{L} \sum_{t=1}^{k} \tau(\gamma_t)\sqrt{t} = \frac{L + 1}{L} \sum_{t=1}^{k} \log(2L\sqrt{t})\sqrt{t}
\leq \frac{L + 1}{L} \sum_{t=1}^{k} \left( \log(2L)\sqrt{t} + \log(\sqrt{t})\sqrt{t} \right)
\leq \frac{(L + 1)\log(2L\sqrt{k})}{L} k^{3/2}, \tag{24}
\]
where in the last inequality we use the integral test to have
\[
\sum_{t=1}^{k} \log(2L) \sqrt{t} \leq \log(2L) k^{3/2}
\]
\[
\sum_{t=1}^{k} \log(\sqrt{t}) \sqrt{t} \leq \frac{k^{3/2} \log(k)}{3} \leq k^{3/2} \log(\sqrt{k}).
\]

Using (21)–(24) into (20) gives (19), i.e.,
\[
f(\bar{x}_k) - f(x^*) \leq f(\bar{x}_0) + 4LD \frac{2k}{k+1} + 4D \frac{M^2 k^{3/2}}{3Lk(k+1)} + 2(2M^2) \frac{2}{3k(k+1)} \leq f(\bar{x}_0) + 4LD \frac{2k}{k+1} + 2(2M^2) \frac{2}{3k(k+1)}. \]

### 3.2 Nonsmooth Strongly Convex Functions

We now provide the rates of Algorithm 1 when the function \( f \) is strongly convex but nonsmooth, that is, \( L = 0 \).

**Assumption 5.** There exists a constant \( \mu > 0 \) s.t. \( \forall x, y \) and \( g(x) \in \partial f(x) \) we have
\[
\frac{\mu}{2} \| y - x \|^2 \leq f(y) - f(x) - \langle g(x), y - x \rangle. \tag{25}
\]

The rate of Algorithm 1 in this case is stated as follows.

**Theorem 2.** Suppose that Assumptions 1–3 and 5 hold. Let \( \{\bar{x}_k\} \) be generated by Algorithm 1. In addition, we consider the step sizes chosen as
\[
\alpha_k = \frac{2}{k+1}, \quad \gamma_k = \frac{2}{\mu(k+1)}, \quad \beta_k = \frac{\alpha_k}{\alpha_k + (1 - \alpha_k)(1 + \mu \gamma_k)}. \tag{26}
\]

Then we have for all \( k \geq 1 \)
\[
f(\bar{x}_k) - f(x^*) \leq \frac{2f(\bar{x}_0) + 6\mu D}{k+1} + \frac{2D + 10M^2 + 8M\mu D}{\mu(k+1)} + \frac{(4M^2 + 8M\mu D)(2 + \mu) \log(\mu(k+1)^2)}{\mu k}. \tag{27}
\]

**Proof.** First, using (11) and (26) we have \( \Gamma_0 = 1, \alpha_0 = 2, \) and \( \gamma_0 = 2/\mu. \) Second, it is straightforward to
verify that (26) satisfies the conditions in (10) and (12). Thus, using (15) into (13) and since \( L = 0 \) we have

\[
\begin{align*}
& f(\bar{x}_k) - f(x^*) \\
\leq & [f(\bar{x}_0) + 3\mu D] + \Gamma_k k \sum_{t=1}^{k} \frac{4M^2\gamma_t \alpha_t}{\Gamma_t(1 + \mu \gamma_t)} \\
& + 2M(M + 2\mu D)\Gamma_k \sum_{t=1}^{k} \frac{\tau(\gamma_t) \alpha_t \gamma_t - \tau(\gamma_t)}{\Gamma_t} \\
& + 2(D + M^2 + 4\mu M D)\Gamma_k \sum_{t=1}^{k} \frac{\alpha_t \gamma_t}{\Gamma_t}.
\end{align*}
\]

(28)

We next analyze each summand on the right-hand side of (28). Using (26) and (21) (to have \( \Gamma_t = 2/(t(t + 1)) \)) yields

\[
\sum_{t=1}^{k} \frac{\gamma_t \alpha_t}{\Gamma_t(1 + \mu \gamma_t)} = \sum_{t=1}^{k} \frac{4t(t + 1)}{2\mu(t + 1)^2(1 + \frac{2}{t+1})} = \sum_{t=1}^{k} \frac{2t}{\mu(t + 3)} \leq \frac{2k}{\mu}.
\]

(29)

By (8) and (26) we have \( \tau(\gamma_k) = \log(\mu(k + 1)/2) \), which gives \( \mu(k + 1)/2 \geq \tau(\gamma_k) \). Thus, we obtain

\[
\gamma_{k-\gamma_k} = \frac{2}{\mu(k + 1) - \log(\mu(k + 1)/2)} \leq \frac{(2 + \mu)\tau(\gamma_t)}{\mu k}.
\]

Using the relation above, (18), and \( \Gamma_t = 2/(t(t + 1)) \) gives

\[
\sum_{t=1}^{k} \frac{\alpha_t \tau(\gamma_t) \gamma_t - \tau(\gamma_t)}{\Gamma_t} \leq \frac{2 + \mu}{\mu} \sum_{t=1}^{k} \tau(\gamma_t) \leq \frac{(2 + \mu)(k + 1) \log(\mu(k + 1)/2)}{\mu}.
\]

(30)

Finally, we consider

\[
\sum_{t=1}^{k} \frac{\alpha_t \gamma_t}{\Gamma_t} \leq \frac{2k}{\mu}.
\]

(31)

Using (29)–(31) into (28) together with \( \Gamma_k = 2/k(k + 1) \) immediately gives (27), i.e.,

\[
\begin{align*}
\frac{f(\bar{x}_k) - f(x^*)}{k(k + 1)} & \leq \frac{2f(\bar{x}_0) + 6\mu D}{k(k + 1)} + \frac{8M^2}{\mu(k + 1)} \\
& + \frac{4M(M + 2\mu D)(2 + \mu)[\log(\mu/2) + \log(k + 1)]}{\mu k} + \frac{2(D + M^2 + 4\mu M D)}{\mu(k + 1)} \\
& = \frac{2f(\bar{x}_0) + 6\mu D}{k(k + 1)} + \frac{2D + 10M^2 + 8\mu M D}{\mu k} + \frac{(4M^2 + 8\mu M D)(2 + \mu) \log(\mu(k + 1))/2}{\mu k}.
\end{align*}
\]

\[
\Box
\]

\section{Convergence Analysis: Nonconvex Case}

In this section, we consider problem (1) when the function \( f \) is nonconvex and \( \mathcal{X} = \mathbb{R}^d \). For solving this problem, we study a slight different version of accelerated Markov gradient descent, which is formally stated
Algorithm 2 Accelerated Markov Gradient Descent

Initialize: Set arbitrarily $x_0, \bar{x}_0 \in \mathcal{X}$, step sizes $\{\alpha_k, \beta_k, \gamma_k\}$, and an integer $K \geq 1$

Iterations: For $k = 1, \ldots, K$ do

\begin{align*}
y_k &= (1 - \alpha_k)\bar{x}_{k-1} + \alpha_k x_{k-1} \\
x_k &= x_{k-1} - \gamma_k G(y_k, \xi_k) \\
x_k &= y_k - \beta_k G(y_k, \xi_k)
\end{align*}

Output: $y_R$ randomly selected from the sequence $\{y_k\}_{k=1}^K$ with probability $p_k$ defined as

\begin{align*}
p_k &= \frac{\gamma_k (1 - L \gamma_k)}{\sum_{k=1}^K \gamma_k (1 - L \gamma_k)}.
\end{align*}

In Algorithm 2, this algorithm is proposed in [12] where the authors consider i.i.d samples. We adopt here to solve problem (1) when the gradient estimates are sampled from an ergodic Markov process. When the function $f$ is nonconvex, we first assume that it is smooth, i.e., Assumption 4 holds. We then require a bounded below assumption on $f$.

Assumption 6. $f^* = \inf_{x \in \mathbb{R}^d} f(x) > -\infty$.

In addition, we consider the following assumption about the boundedness of the gradient and its samples.

Assumption 7. There exists a constant $M > 0$ such that

\begin{align*}
\max\{\|\nabla f(x)\|, \|G(x, \xi)\|\} \leq M, \forall \xi \in \Xi, x \in \mathbb{R}^d.
\end{align*}

Under these assumptions, we first consider the following key lemma in deriving the rates of Algorithm 2. Its analysis is presented in Appendix for an ease of exposition.

Lemma 3. Suppose that Assumptions 1, 2, 4, and 7 hold. Let $\gamma_k$ be nonnegative and nonincreasing. Then we have

\begin{align*}
\mathbb{E}[f(x_k)] &\leq \mathbb{E}[f(x_{k-1})] - \gamma_k (1 - L \gamma_k) \mathbb{E} [\|\nabla f(y_k)\|^2] \\
&\quad + \frac{M^2 L \Gamma_k}{2} \sum_{t=1}^k \frac{(\gamma_t - \beta_t)^2}{\Gamma_t \alpha_t} + (4M^2 L + M) \gamma_k^2 + 2LM^2 \tau(\gamma_k) \gamma_k \gamma_{k-\tau(\gamma_k)} \gamma_k.
\end{align*}

To show our result for smooth nonconvex problems, we adopt the randomized stopping rule in [12], which is common used in nonconvex optimization. In particular, given a sequence $\{y_k\}$ generated by Algorithm 2, we study the convergence on $y_R$, a point randomly selected from this sequence (a.k.a. (35)). The convergence rate of Algorithm 2 in solving problem (1) is stated as follows.

Theorem 3. Suppose that Assumptions 1, 2, 4, 6 and 7 hold. Let $K > 0$ be an integer and $\gamma_k$ be nonincreasing. In addition, we consider

\begin{align*}
\alpha_k &= \frac{2}{k+1}, \quad \gamma_k \in [\beta_k, (1 + \alpha_k)\beta_k] \\
\beta_k &= \beta = \min \left\{ \frac{1}{4L}, \frac{1}{\sqrt{K}} \right\}.
\end{align*}

\[9\]
Then $y_R$ returned by Algorithm 2 satisfies

$$
\mathbb{E} \left[ \| \nabla f(y_R) \|^2 \right] \leq \frac{2(f(x_0) - f^*)}{K} \left( 4L + \sqrt{K} \right) + \frac{2(9LM^2 + 2M)}{\sqrt{K}} + \frac{8CLM^2 \log(K)}{\sqrt{K}}. 
$$

(39)

**Proof.** Summing up both sides of (37) over $k$ from 1 to $N$ and reorganizing yield

$$
\sum_{k=1}^K \gamma_k (1 - L\gamma_k) \mathbb{E} \left[ \| \nabla f(y_k) \|^2 \right] \leq \mathbb{E}[f(x_0)] - \mathbb{E}[f(x_K)] + \sum_{k=1}^K \frac{M^2L}{2} \Gamma_k \sum_{t=1}^k \frac{(\gamma_t - \beta_t)^2}{\Gamma_t \alpha_t}
$$

$$
+ (4LM^2 + M) \sum_{k=1}^K \gamma_k^2 + 2LM^2 \sum_{k=1}^K \tau(\gamma_k) \gamma_{k-\tau(\gamma_k)}\gamma_k
$$

$$
\leq (f(x_0) - f^*) + \frac{M^2L}{2} \sum_{k=1}^K \Gamma_k \sum_{t=1}^k \frac{(\gamma_t - \beta_t)^2}{\Gamma_t \alpha_t}
$$

$$
+ (4LM^2 + M) \sum_{k=1}^K \gamma_k^2 + 2LM^2 \sum_{k=1}^K \tau(\gamma_k) \gamma_{k-\tau(\gamma_k)}\gamma_k,
$$

(40)

where the last inequality follows since $\mathbb{E}[f(x_K)] \geq f^*$ and a given $x_0$.

We next analyze each summand on the right-hand side of (40). First, using (21) and (38) we consider

$$
\sum_{k=1}^K \frac{M^2L}{2} \Gamma_k \sum_{t=1}^k \frac{(\gamma_t - \beta_t)^2}{\Gamma_t \alpha_t}
$$

$$
= \sum_{t=1}^K \frac{(\gamma_t - \beta_t)^2}{\Gamma_t \alpha_t} \sum_{k=t}^K \Gamma_k
$$

$$
= \sum_{t=1}^K \frac{(\gamma_t - \beta_t)^2}{\Gamma_t \alpha_t} \sum_{k=t}^K \frac{2}{k(k+1)}
$$

$$
= \sum_{t=1}^K \frac{2(\gamma_t - \beta_t)^2}{\Gamma_t \alpha_t} \sum_{k=t}^K \left( \frac{1}{k} - \frac{1}{k+1} \right)
$$

$$
\leq \sum_{t=1}^K \frac{2(\gamma_t - \beta_t)^2}{\Gamma_t \alpha_t} \frac{1}{t} \leq 2 \sum_{t=1}^K \frac{\beta_t^2 \alpha_t^2}{t \Gamma_t \alpha_t} = 2\beta^2K.
$$

(41)

Second, using (38) and $\alpha_k \leq 1$ for $k \geq 1$ we have

$$
\sum_{k=1}^K \gamma_k^2 \leq 2\beta^2K.
$$

(42)

Third, using (8) we have

$$
\tau(\gamma_k) = C \log \left( \frac{1}{\gamma_k} \right) \leq \frac{C}{2} \log \left( k \right),
$$

which gives

$$
\sum_{k=1}^K \tau(\gamma_k) \gamma_{k-\tau(\gamma_k)}\gamma_k \leq \frac{C}{2} \sum_{k=1}^K 4\beta_k \beta_{k-\tau(\gamma_k)} \log(K) = 2C\beta^2K \log(K).
$$

(43)

---

Note that the same rate can be achieved for the quantity $\min_k \mathbb{E} \left[ \| \nabla f(y_k) \|^2 \right]$. 

---

10
Using (41)–(43) into (40) yields
\[
\sum_{k=1}^{K} \gamma_k (1 - L\gamma_k) \mathbb{E} \left[ \|\nabla f(y_k)\|^2 \right] \leq (f(x_0) - f^*) + M^2 L \beta^2 K
\]
\[
+ 2(4LM^2 + M)\beta^2 K + 4CLM^2 \beta^2 K \log(K)
\]
\[
= (f(x_0) - f^*) + (9LM^2 + 2M)\beta^2 K
\]
\[
+ 4CLM^2 \beta^2 K \log(K),
\]
which when dividing both sides by \(\sum_{k=1}^{K} \gamma_k (1 - L\gamma_k)\) gives
\[
\sum_{k=1}^{K} \gamma_k (1 - L\gamma_k) \mathbb{E} \left[ \|\nabla f(y_k)\|^2 \right] \sum_{k=1}^{K} \gamma_k (1 - L\gamma_k) \leq (f(x_0) - f^*) \sum_{k=1}^{K} \gamma_k (1 - L\gamma_k)
\]
\[
+ (9LM^2 + 2M)\beta^2 K \sum_{k=1}^{K} \gamma_k (1 - L\gamma_k) + 4CLM^2 \beta^2 K \log(K) \sum_{k=1}^{K} \gamma_k (1 - L\gamma_k),
\]
which by using (38) to have \(1 - L\gamma_k \geq 1/2\) and \(\sum_{k=1}^{K} \gamma_k (1 - L\gamma_k) \geq \sum_{k=1}^{K} \beta_k / 2 = K\beta / 2\) we obtain
\[
\sum_{k=1}^{K} \gamma_k (1 - L\gamma_k) \mathbb{E} \left[ \|\nabla f(y_k)\|^2 \right] \sum_{k=1}^{K} \gamma_k (1 - L\gamma_k) \leq \frac{2(f(x_0) - f^*)}{K\beta} + \frac{2(9LM^2 + 2M)\beta^2 K}{K\beta}
\]
\[
+ \frac{8CLM^2 \beta^2 K \log(K)}{K\beta}
\]
\[
\leq \frac{2(f(x_0) - f^*)}{K} \left( 4L + \sqrt{K} \right)
\]
\[
+ \frac{2(9LM^2 + 2M)}{\sqrt{K}} + \frac{8CLM^2 \log(K)}{\sqrt{K}}.
\]
Thus, by using (35) the preceding immediately gives (39). \(\square\)

5 Numerical Experiments

In this section, we apply the proposed accelerated Markov gradient method for solving a number of problems in reinforcement learning, where the samples are taken from Markov processes. In particular, we consider the usual setup of reinforcement learning where the environment is modeled by a Markov decision process \([32]\). Let \(\mathcal{S}\) and \(\mathcal{A}\) be the set of states and action. We denote by \(\pi_\theta(s, a) = Pr(a_t = a|s_t = s, \theta)\) the randomized policy parameterized by \(\theta\), where \(s \in \mathcal{S}\) and \(a \in \mathcal{A}\). The goal is to find \(\theta\) to maximize
\[
f(\pi_\theta) = \mathbb{E} \left[ \sum_{k=0}^{\infty} \gamma^k r_k | s_0, \pi_\theta \right],
\]
where \(\gamma\) is the discounted factor and \(r_k\) is the reward returned by the environment at time \(k\).

For solving this problem, we consider a popular method in reinforcement learning, namely, Monte-Carlo policy gradient (or REINFORCE), an equivalent version of SGD algorithm in reinforcement learning \([36]\).
Our goal is to compare the performance of this classic REINFORCE algorithm with its accelerated variant using our proposed approach.

For our numerical simulation, we consider five different control problems, namely, Acrobot, CartPole, Ant, Swimmer, and HalfCheetah, using the simulated environments from OpenAI Gym and Mujoco [4][34]. We utilize the implementation of REINFORCE from rllab library [9]. More details of these environments are given in Appendix or in [4]. The details of our experiments are given as follows.

**General setup:** For each environment, we randomly generate an initial policy represented by a neural network. For each algorithm, we run the algorithms 10 times with the same initial policy and record the performance measures. The performance of each algorithm is specified by the average rewards over number of episodes sampled. Here, an episode is defined as the set of state-action pairs collected from beginning until we arrive at the terminal state or reach the specified episode length. The plots consist of the mean with 90% confidence interval (shaded area) of the average rewards for each algorithm.

**Brief summary of algorithm update:** At every iteration, we collect a batch of episodes with different length depending on the environment. After that, we perform the algorithm update for the policy parameters. We record the performance measure by using the updated policy to collect 50 episodes and average the total rewards for all episodes.

We first compare the algorithms using discrete control tasks: Acrobot-v1 and CartPole-v0 environments. For these discrete tasks, we use a soft-max policy $\pi_\theta$ with parameter $\theta$ defined as

$$
\pi_\theta(a|s) = \frac{e^{\phi(s,a,\theta)}}{\sum_{k=1}^{|A|} e^{\phi(s,a_k,\theta)}}
$$

(44)

where $\phi(s,a,\theta)$ is represented by a neural network and $|A|$ is the total number of actions. For more details of the parameters and network architecture, the reader can refer to the Supplementary Material. Figure 1 presents the performance of two algorithms on these environments.

In both environments, the accelerated REINFORCE significantly outperforms its non-accelerated variant.

Next, we evaluate the performance of these algorithms on continuous control tasks in Mujoco. In these environments, we also incorporate a linear baseline to reduce the variance of the policy gradient estimator,
Figure 2: The performance of two algorithms on the Swimmer-v2 environment.

Figure 3: The performance of two algorithms on the HalfCheetah-v2 environment.

Figure 4: The performance of two algorithms on the Ant-v2 environment.
In these tasks, the actions are sampled from a deep Gaussian policy which can be written as

\[ \pi_{\theta}(a|s) = \mathcal{N}(\phi(s, a, \theta_\mu); \phi(s, a, \theta_\sigma)) \]  

(45)

where \( \phi(\cdot) \) is a neural network. Note that both the mean and variance of the Gaussian distribution is learned in this experiment. More details about the choice of parameters can be found in the Supplementary Material.

We evaluate these algorithms on three environments with increasing difficulty: Swimmer, HalfCheetah, and Ant. More details about these environments can be found in the Supplementary Material. Figure 2, 3, and 4 illustrate the results in those environments, respectively. In all figures, REINFORCE-Acc indeed shows its advantage over REINFORCE.

In summary, the accelerated policy gradient variant outperforms its non-accelerated one in all environments which confirm the advantage from theoretical analysis.

6 Conclusion

In this paper, we study a variant of ASGD for solving (possibly nonconvex) optimization problems, when the gradients are sampled from Markov process. We characterize the finite-time performance of this method when the gradients are sampled from Markov processes, which shows that ASGD converges at the nearly the same rate with Markovian gradient samples as with independent gradient samples. The only difference is a logarithmic factor that accounts for the mixing time of the Markov chain. We apply the accelerated method to several challenging problems in the OpenAI Gym and Mujoco, and show that acceleration can significantly improve the performance of the classic REINFORCE algorithm.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

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

[3] Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. Siam Review, 60(2):223–311, 2018.

[4] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym, 2016.
[5] C. Zaiwei Chen, S. Zhang, T. T. Doan, S. T. Maguluri, and J-P. Clarke. Performance of Q-learning with Linear Function Approximation: Stability and Finite-Time Analysis. available at: https://arxiv.org/abs/1905.11425, 2019.

[6] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In NIPS, pages 1646–1654, 2014.

[7] 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.

[8] T. T. Doan. Finite-time analysis and restarting scheme for linear two-time-scale stochastic approximation. available at: https://arxiv.org/abs/1912.10583, 2019.

[9] Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48, ICML16, page 13291338. JMLR.org, 2016.

[10] J. C. Duchi, A. Agarwal, M. Johansson, and M.I. Jordan. Ergodic mirror descent. SIAM Journal on Optimization, 22(4):1549–1578, 2012.

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

[12] S. Ghadimi and G. Lan. Stochastic first- and zeroth-order methods for nonconvex stochastic programming. SIAM Journal on Optimization, 23(4):2341–2368, 2013.

[13] Evan Greensmith, Peter L Bartlett, and Jonathan Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. Journal of Machine Learning Research, 5(Nov):1471–1530, 2004.

[14] Prateek Jain, Sham M Kakade, Rahul Kidambi, Praneeth Netrapalli, and Aaron Sidford. Accelerating stochastic gradient descent for least squares regression. arXiv preprint arXiv:1704.08227, 2017.

[15] B. Johansson, M. Rabi, and M. Johansson. A randomized incremental subgradient method for distributed optimization in networked systems. SIAM Journal on Optimization, 20(3):1157–1170, 2010.

[16] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In NIPS, pages 315–323, 2013.

[17] Hiroyuki Kasai. Sgdlibrary: A matlab library for stochastic optimization algorithms. The Journal of Machine Learning Research, 18(1):7942–7946, 2017.

[18] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. CoRR, abs/1412.6980, 2014.

[19] G. Lan. An optimal method for stochastic composite optimization. Mathematical Programming, 133(1):365–397, Jun 2012.

[20] G. Lan. Lectures on optimization methods for machine learning, 2019.

[21] Nicolas Le Roux, Mark Schmidt, and Francis Bach. A stochastic gradient method with an exponential convergence rate for finite training sets. In NIPS, pages 2663–2671, 2012.
[22] Chaoyue Liu and Mikhail Belkin. Accelerating sgd with momentum for over-parameterized learning. arXiv preprint arXiv:1810.13395, 2018.

[23] Yurii Nesterov. A method for unconstrained convex minimization problem with the rate of convergence $o(1/k^2)$. 1983.

[24] Lam Nguyen, Phuong Ha Nguyen, Marten van Dijk, Peter Richtarik, Katya Scheinberg, and Martin Takac. SGD and Hogwild! convergence without the bounded gradients assumption. In International Conference on Machine Learning, pages 3747–3755, 2018.

[25] Lam M. Nguyen, Jie Liu, Katya Scheinberg, and Martin Takac. SARAH: A novel method for machine learning problems using stochastic recursive gradient. In International Conference on Machine Learning, pages 2613–2621, 2018.

[26] Lam M. Nguyen, Phuong Ha Nguyen, Peter Richtárik, Katya Scheinberg, Martin Takáč, and Marten van Dijk. New convergence aspects of stochastic gradient algorithms. Journal of Machine Learning Research, 20(176):1–49, 2019.

[27] S.S. Ram, A. Nedi, and V. V. Veeravalli. Incremental stochastic subgradient algorithms for convex optimization. SIAM Journal on Optimization, 20(2):400–407, 1951.

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

[29] R. Srikant and L. Ying. Finite-time error bounds for linear stochastic approximation and TD learning. In COLT, 2019.

[30] T. Sun, Y. Sun, and W. Yin. On markov chain gradient descent. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, NIPS18, page 99189927, Red Hook, NY, USA, 2018. Curran Associates Inc.

[31] 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.

[32] R. S. Sutton and A. G. Barto. Introduction to Reinforcement Learning, 2nd Edition. MIT Press, 2018.

[33] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. Technical report, 2012.

[34] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 5026–5033. IEEE, 2012.

[35] Sharan Vaswani, Francis Bach, and Mark Schmidt. Fast and faster convergence of sgd for over-parameterized models and an accelerated perceptron. arXiv preprint arXiv:1810.07288, 2018.

[36] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine Learning, 8(3):229–256, May 1992.

[37] Kun Yuan, Bicheng Ying, and Ali H Sayed. On the influence of momentum acceleration on online learning. The Journal of Machine Learning Research, 17(1):6602–6667, 2016.
A Appendix

A.1 Proof of Lemma [1]

For convenience, we denote by $\Delta_k = G(y_k; \xi_k) - \nabla f(y_k)$, the difference between the gradient sample and its expected value w.r.t the stationary distribution $\pi$. The proof of this lemma is adopted from the results studied in [19]. We restate here with some minor modification for the purpose of our analysis.

**Proof.** Using the convexity of $f$, i.e.,

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle, \quad \forall x, y \in \mathbb{R}^d$$

and (5) we have for all $z \in \mathcal{X}$

$$f(z) + \langle \nabla f(z), \bar{x}_k - z \rangle = f(z) + \langle \nabla f(z), \alpha_k x_k + (1 - \alpha_k)\bar{x}_{k-1} - z \rangle$$

$$= (1 - \alpha_k) [f(z) + \langle \nabla f(z), \bar{x}_{k-1} - z \rangle] + \alpha_k [f(z) + \langle \nabla f(z), x_k - z \rangle]$$

$$\leq (1 - \alpha_k) f(\bar{x}_{k-1}) + \alpha_k [f(z) + \langle \nabla f(z), x_k - z \rangle].$$

By the preceding relation and (17) we have for all $z \in \mathcal{X}$

$$f(\bar{x}_k) \leq f(z) + \langle \nabla f(z), \bar{x}_k - z \rangle + \frac{L}{2} \| \bar{x}_k - z \|^2$$

$$\leq (1 - \alpha_k) f(\bar{x}_{k-1}) + \alpha_k [f(z) + \langle \nabla f(z), x_k - y_k \rangle] + \frac{L}{2} \| \bar{x}_k - y_k \|^2,$$

which by letting $z = y_k$ we obtain

$$f(\bar{x}_k) \leq (1 - \alpha_k) f(\bar{x}_{k-1}) + \alpha_k [f(y_k) + \langle \nabla f(y_k), x_k - y_k \rangle] + \frac{L}{2} \| \bar{x}_k - y_k \|^2. \tag{46}$$

By the update of $x_k$ in (4) and Lemma 3.5 in [20] we have for all $z \in \mathcal{X}$

$$\gamma_k \left[ \langle G(y_k; \xi_k), x_k - y_k \rangle + \mu V(y_k, x_k) \right] + V(x_{k-1}, x_k)$$

$$\leq \gamma_k \left[ \langle G(y_k; \xi_k), z - y_k \rangle + \mu V(y_k, z) \right] + V(x_{k-1}, z) - (1 + \mu \gamma_k) V(x_k, z),$$

which implies that

$$\langle \nabla f(y_k), x_k - y_k \rangle \leq \mu V(y_k, z) + \frac{1}{\gamma_k} V(x_{k-1}, z) - \frac{1 + \mu \gamma_k}{\gamma_k} V(x_k, z) - \mu V(y_k, x_k) - \frac{1}{\gamma_k} V(x_{k-1}, x_k)$$

$$+ \langle G(y_k; \xi_k), z - y_k \rangle - \langle G(y_k; \xi_k) - \nabla f(y_k), x_k - y_k \rangle$$

$$= \mu V(y_k, z) + \frac{1}{\gamma_k} V(x_{k-1}, z) - \frac{1 + \mu \gamma_k}{\gamma_k} V(x_k, z) - \mu V(y_k, x_k) - \frac{1}{\gamma_k} V(x_{k-1}, x_k)$$

$$+ \langle \nabla f(y_k), z - y_k \rangle + \langle G(y_k; \xi_k) - \nabla f(y_k), z - x_k \rangle.$$

Substituting the preceding equation into Eq. (46) yields

$$f(\bar{x}_k) \leq (1 - \alpha_k) f(\bar{x}_{k-1}) + \alpha_k [f(y_k) + \langle \nabla f(y_k), z - y_k \rangle + \mu V(y_k, z)] + \frac{L}{2} \| \bar{x}_k - y_k \|^2$$

$$+ \frac{\alpha_k}{\gamma_k} \left[ V(x_{k-1}, z) - (1 + \mu \gamma_k) V(x_k, z) - V(x_{k-1}, x_k) - \mu \gamma_k V(y_k, x_k) \right]$$

$$+ \alpha_k \langle G(y_k; \xi_k) - \nabla f(y_k), z - x_k \rangle. \tag{47}$$
We denote by $\bar{x}_{k-1}$
\[
\bar{x}_{k-1} = \frac{1}{1 + \mu \gamma_k} x_{k-1} + \frac{\mu \gamma_k}{1 + \mu \gamma_k} y_k.
\] (48)

And note that
\[
\frac{1}{1 + \mu \gamma_k} = \frac{\beta_k (1 - \alpha_k)}{\alpha_k (1 - \beta_k)} \quad \text{and} \quad \frac{\mu \gamma_k}{1 + \mu \gamma_k} = \frac{\alpha_k - \beta_k}{\alpha_k (1 - \beta_k)}.
\]

Thus, using Eqs. (3) and (5), and the preceding relations we then have
\[
\bar{x}_k - y_k = \alpha_k x_k + \frac{1 - \alpha_k}{1 - \beta_k} (y_k - \beta_k x_{k-1}) - y_k = \alpha_k \left[ x_k - \frac{\beta_k (1 - \alpha_k)}{\alpha_k (1 - \beta_k)} x_{k-1} - \frac{\alpha_k - \beta_k}{\alpha_k (1 - \beta_k)} y_k \right]
\] = \alpha_k (x_k - \bar{x}_{k-1}). \tag{49}

On the other hand, using the strong convexity of $V$ we have
\[
V(x_{k-1}, x_k) + \mu \gamma_k V(y_k, x_k) \geq \frac{1}{2} \|x_{k-1} - x_k\|^2 + \frac{\mu \gamma_k}{2} \|x_k - y_k\|^2
\]
\[
\geq \frac{1 + \mu \gamma_k}{2} \left\| \frac{1}{1 + \mu \gamma_k} (x_k - x_{k-1}) + \frac{\mu \gamma_k}{1 + \mu \gamma_k} (x_k - y_k) \right\|^2
\]
\[
= \frac{1 + \mu \gamma_k}{2} \left\| x_k - \frac{1}{1 + \mu \gamma_k} x_{k-1} - \frac{\mu \gamma_k}{1 + \mu \gamma_k} y_k \right\|^2
\]
\[
= \frac{1 + \mu \gamma_k}{2 \alpha_k^2} \|\alpha_k (x_k - \bar{x}_{k-1})\|^2. \tag{50}
\]

Moreover, recall that $\Delta_k = G(y_k; \xi_k) - \nabla f(y_k)$ we then have
\[
\langle G(y_k; \xi_k) - \nabla f(y_k), z - x_k \rangle = \langle G(y_k; \xi_k) - \nabla f(y_k), \bar{x}_{k-1} - x_k \rangle + \langle G(y_k; \xi_k) - \nabla f(y_k), z - \bar{x}_{k-1} \rangle
\]
\[
\leq \|\Delta_k\| \|x_k - \bar{x}_{k-1}\| + \langle \Delta_k, z - \bar{x}_{k-1} \rangle. \tag{51}
\]

Substituting Eqs. (49)–(51) into Eq. (47) yields
\[
f(\bar{x}_k) \leq (1 - \alpha_k) f(\bar{x}_{k-1}) + \alpha_k \left[ f(y_k) + \langle \nabla f(y_k), z - y_k \rangle + \mu V(y_k, z) \right]
\]
\[
+ \frac{\alpha_k}{\gamma_k} \left[ V(x_{k-1}, z) - (1 + \mu \gamma_k) V(x_k, z) \right] + \alpha_k \langle G(y_k; \xi_k) - \nabla f(y_k), z - \bar{x}_{k-1} \rangle
\]
\[
- \left( \frac{1 + \mu \gamma_k}{2 \gamma_k \alpha_k} - \frac{L}{2} \right) \|\alpha_k (x_k - \bar{x}_{k-1})\|^2 + \|\Delta_k\| \|\alpha_k (x_k - \bar{x}_{k-1})\|
\]
\[
\leq (1 - \alpha_k) f(\bar{x}_{k-1}) + \alpha_k \left[ f(y_k) + \langle \nabla f(y_k), z - y_k \rangle + \mu V(y_k, z) \right]
\]
\[
+ \frac{\alpha_k}{\gamma_k} \left[ V(x_{k-1}, z) - (1 + \mu \gamma_k) V(x_k, z) \right] + \alpha_k \langle G(y_k; \xi_k) - \nabla f(y_k), z - \bar{x}_{k-1} \rangle
\]
\[
+ \frac{\gamma_k \alpha_k \|\Delta_k\|^2}{1 + \mu \gamma_k - L \gamma_k \alpha_k}. \tag{52}
\]

Diving both sides of Eq. (52) by $\Gamma_k$ and using (11), we have
\[
\frac{1}{\Gamma_k} f(\bar{x}_k) \leq \frac{1 - \alpha_k}{\Gamma_k} f(\bar{x}_{k-1}) + \frac{\alpha_k}{\Gamma_k} \left[ f(y_k) + \langle \nabla f(y_k), z - y_k \rangle + \mu V(y_k, z) \right] + \frac{\gamma_k \alpha_k \|\Delta_k\|^2}{\Gamma_k (1 + \mu \gamma_k - L \gamma_k \alpha_k)}
\]
\[
+ \frac{\alpha_k}{\Gamma_k \gamma_k} \left[ V(x_{k-1}, z) - (1 + \mu \gamma_k) V(x_k, z) \right] + \frac{\alpha_k}{\Gamma_k} \langle G(y_k; \xi_k) - \nabla f(y_k), z - \bar{x}_{k-1} \rangle
\]
\[
\leq \frac{1}{\Gamma_{k-1}} f(\bar{x}_{k-1}) + \frac{\alpha_k}{\Gamma_k} f(z) + \frac{\gamma_k \alpha_k \|\Delta_k\|^2}{\Gamma_k (1 + \mu \gamma_k - L \gamma_k \alpha_k)}
\]
\[
+ \frac{\alpha_k}{\Gamma_k \gamma_k} \left[ V(x_{k-1}, z) - (1 + \mu \gamma_k) V(x_k, z) \right] + \frac{\alpha_k}{\Gamma_k} \langle G(y_k; \xi_k) - \nabla f(y_k), z - \bar{x}_{k-1} \rangle,
\]
\[18\]
where the last inequality due to the convexity of $f$. Summing up both sides of the preceding relation over $k$ from 1 to $K$ yields

$$
f(\bar{x}_K) \leq \frac{\Gamma_K}{\Gamma_0} f(\bar{x}_0) + \Gamma_K \sum_{k=1}^{K} \frac{\alpha_k}{\Gamma_k} f(z) + \Gamma_K \sum_{k=1}^{K} \frac{\gamma_k \alpha_k \|\Delta_k\|^2}{\Gamma_k(1 + \mu \gamma_k - L\gamma_k \alpha_k)}$

$$+ \Gamma_K \left[ \sum_{k=1}^{K} \frac{\alpha_k}{\Gamma_k} \left[ V(x_{k-1}, z) - (1 + \mu \gamma_k) V(x_k, z) \right] \right] \leq \frac{\Gamma_K}{\Gamma_0} f(\bar{x}_0) + f(z) + \Gamma_K \sum_{k=1}^{K} \frac{\gamma_k \alpha_k \|\Delta_k\|^2}{\Gamma_k(1 + \mu \gamma_k - L\gamma_k \alpha_k)}.$$

Thus, by letting $z = x^*$ in the preceding equation we obtain (13).

A.2 Proof of Lemma 2

We first consider the following lemma.

**Lemma 4.** The sequence $\{x_k\}$ generated by Algorithm 1 satisfies

$$\|x_k - x_{k-1}\| \leq (M + 2D\mu)\gamma_k.$$  

**Proof.** First, by the optimality condition of (4) and since $V(x, y) = \frac{1}{2}\|x - y\|^2$ we have

$$\langle \gamma_k G(y_k; \xi_k) + \mu \gamma_k (x_k - y_k), x_k - x_{k-1}, x_{k-1} - x_k \rangle \geq 0,$$

which by rearranging the equation and using (9) we have

$$\|x_k - x_{k-1}\|^2 \leq \langle \gamma_k G(y_k; \xi_k) + \mu \gamma_k (x_k - y_k), x_{k-1} - x_k \rangle \leq (M + 2D\mu)\gamma_k \|x_k - x_{k-1}\|,$$

Dividing both sides of the equation above by $x_k - x_{k-1}$ gives us Eq. (54).
Proof of Lemma 2. Consider

\[
\langle G(y_k; \xi_k) - \nabla f(y_k), z - x_{k-1} \rangle = \langle G(y_k; \xi_k) - \nabla f(y_k), z - x_{k-\tau(\gamma_k)} \rangle \\
+ \langle G(y_k; \xi_k) - \nabla f(y_k), x_{k-\tau(\gamma_k)} - x_k \rangle \\
+ \langle G(y_k; \xi_k) - \nabla f(y_k), x_k - x_{k-1} \rangle \\
+ \frac{\mu \gamma_k}{1 + \mu \gamma_k} \langle G(y_k; \xi_k) - \nabla f(y_k), x_k - x_{k-1} \rangle. \tag{55}
\]

Note that by (48) we have

\[
x_k - x_{k-1} = x_k - \frac{1}{1 + \mu \gamma_k} x_{k-1} - \frac{\mu \gamma_k}{1 + \mu \gamma_k} y_k = x_k - x_{k-1} + \frac{\mu \gamma_k}{1 + \mu \gamma_k} (x_{k-1} - y_k),
\]

which by substituting into Eq. (55) yields

\[
\langle G(y_k; \xi_k) - \nabla f(y_k), z - \tilde{x}_{k-1} \rangle = \langle G(y_k; \xi_k) - \nabla f(y_k), z - x_{k-\tau(\gamma_k)} \rangle \\
+ \langle G(y_k; \xi_k) - \nabla f(y_k), x_{k-\tau(\gamma_k)} - x_k \rangle \\
+ \langle G(y_k; \xi_k) - \nabla f(y_k), x_k - x_{k-1} \rangle \\
+ \frac{\mu \gamma_k}{1 + \mu \gamma_k} \langle G(y_k; \xi_k) - \nabla f(y_k), x_k - y_k \rangle. \tag{56}
\]

Next, we provide upper bounds for each term on the right-hand side of Eq. (56). First, let \( \tau(\gamma_k) \) be the mixing time of the underlying Markov chain associated with the step size \( \gamma_k \), defined in Assumption 2. In addition, we denote by \( F_k \) the filtration containing all the history generated by the algorithm up to time \( k \). Then we have,

\[
\mathbb{E}[\langle G(y_k; \xi_k) - \nabla f(y_k), z - x_{k-\tau(\gamma_k)} | F_{k-\tau(\gamma_k)} \rangle] = \mathbb{E}[\langle G(y_k; \xi_k) - \nabla f(y_k) | F_{k-\tau(\gamma_k)} \rangle, z - x_{k-\tau(\gamma_k)}] \\
\leq \| z - x_{k-\tau(\gamma_k)} \| \| \mathbb{E}[G(y_k; \xi_k) - \nabla f(y_k) | F_{k-\tau(\gamma_k)}] \| \leq 2D \gamma_k, \tag{57}
\]

where the last inequality is due to (9) and Assumption 2. Second, using Eqs. (54) and (9) we consider the second and third terms on the right-hand side of (56)

\[
\langle G(y_k; \xi_k) - \nabla f(y_k), x_{k-\tau(\gamma_k)} - x_k \rangle + \langle G(y_k; \xi_k) - \nabla f(y_k), x_k - x_{k-1} \rangle \\
\leq 2M \| x_{k-\tau(\gamma_k)} - x_k \| + 2M \| x_k - x_{k-1} \| \leq 2M \sum_{t=k+1-\tau(\gamma_k)}^{k} \| x_t - x_{t-1} \| + 2M \| x_k - x_{k-1} \| \leq 2M(M + 2\mu D) \sum_{t=k-1-\tau(\gamma_k)}^{k} \gamma_t + 2M(M + 2\mu D) \gamma_k \leq 2M(M + 2\mu D) \left[ \tau(\gamma_k) \gamma_k - (\gamma_k) + \gamma_k \right], \tag{58}
\]

where the last inequality is due to the fact that \( \gamma_k \) is nonincreasing. Finally, using (9) we consider the last term of Eq. (56)

\[
\frac{\mu \gamma_k}{1 + \mu \gamma_k} \langle G(y_k; \xi_k) - \nabla f(y_k), x_k - y_k \rangle \leq \frac{4 \mu M D \gamma_k}{1 + \mu \gamma_k}. \tag{59}
\]

Taking the expectation on both sides of (56) and using Eqs. (57)–(59) immediately gives Eq. (15). \( \square \)
A.3 Proof of Lemma

Proof. Since $f$ satisfies Assumption 4 by (17), (34), and (33) we have

$$f(x_k) \leq f(x_{k-1}) + \langle \nabla f(x_{k-1}), x_k - x_{k-1} \rangle + \frac{L}{2} \|x_k - x_{k-1}\|^2$$

$$= f(x_{k-1}) - \gamma_k \langle \nabla f(x_{k-1}), G(y_k, \xi_k) \rangle + \frac{L \gamma_k^2}{2} \|G(y_k, \xi_k)\|^2$$

$$= f(x_{k-1}) - \gamma_k \langle \nabla f(x_{k-1}), \nabla f(y_k) \rangle - \gamma_k \langle \nabla f(x_{k-1}), G(y_k, \xi_k) - \nabla f(y_k) \rangle + \frac{L \gamma_k^2}{2} \|G(y_k, \xi_k)\|^2$$

$$= f(x_{k-1}) - \gamma_k \|\nabla f(y_k)\|^2 - \gamma_k \langle \nabla f(x_{k-1}) - \nabla f(y_k), \nabla f(y_k) \rangle$$

$$- \gamma_k \langle \nabla f(x_{k-1}) - \nabla f(y_k), G(y_k, \xi_k) - \nabla f(y_k) \rangle + \frac{L \gamma_k^2}{2} \|G(y_k, \xi_k) - \nabla f(y_k)\|^2$$

$$\leq f(x_{k-1}) - \gamma_k \left(1 - \frac{L \gamma_k}{2}\right) \|\nabla f(y_k)\|^2 + L \gamma_k \|x_{k-1} - y_k\| \|\nabla f(y_k)\|$$

$$- \gamma_k \langle \nabla f(x_{k-1}) - \nabla f(y_k), G(y_k, \xi_k) - \nabla f(y_k) \rangle + 2M^2 L \gamma_k^2,$$

where the last inequality is due to (16) and (56). Using (34) we have from the preceding relation

$$f(x_k) \leq f(x_{k-1}) - \gamma_k \left(1 - \frac{L \gamma_k}{2}\right) \|\nabla f(y_k)\|^2 + L \gamma_k \|x_{k-1} - \bar{x}_{k-1}\| \|\nabla f(y_k)\|$$

$$- \gamma_k \langle \nabla f(x_{k-1}) - \nabla f(y_k), G(y_k, \xi_k) - \nabla f(y_k) \rangle + 2M^2 L \gamma_k^2$$

$$\leq f(x_{k-1}) - \gamma_k \left(1 - \frac{L \gamma_k}{2}\right) \|\nabla f(y_k)\|^2 + \frac{L (1 - \alpha_k)^2}{2} \|x_{k-1} - \bar{x}_{k-1}\|^2$$

$$- \gamma_k \langle \nabla f(x_{k-1}) - \nabla f(y_k), G(y_k, \xi_k) - \nabla f(y_k) \rangle + 2M^2 L \gamma_k^2,$$

where the last inequality we apply the relation $2ab \leq a^2 + b^2$ to the third term. Next, using (32)--(34) we have

$$\bar{x}_k - x_k = (1 - \alpha_k)(\bar{x}_{k-1} - x_{k-1}) + (\gamma_k - \beta_k)G(y_k, \xi_k),$$

which dividing both sides by $\Gamma_k$, using (11) and $\alpha_1 = 1$, and summing up both sides yields

$$\bar{x}_k - x_k = \Gamma_k \sum_{t=1}^{k} \frac{\gamma_t - \beta_t}{\Gamma_t} G(y_t, \xi_t).$$

Thus, by using the Jensen’s inequality for $\| \cdot \|^2$ and (53) we have from the preceding equation

$$\|\bar{x}_k - x_k\|^2 = \left\| \Gamma_k \sum_{t=1}^{k} \frac{\gamma_t - \beta_t}{\Gamma_t} G(y_t, \xi_t) \right\|^2$$

$$\leq \Gamma_k \sum_{t=1}^{k} \frac{\alpha_t}{\Gamma_t} \left\| \frac{\gamma_t - \beta_t}{\alpha_t} G(y_t, \xi_t) \right\|^2$$

$$\leq M^2 \Gamma_k \sum_{t=1}^{k} \frac{(\gamma_t - \beta_t)^2}{\Gamma_t \alpha_t},$$
where the last inequality is due to (36). Substituting the preceding relation into (60) and since \((1-\alpha_k)^2\Gamma_{k-1} \leq \Gamma_k\) we have

\[
f(x_k) \leq f(x_{k-1}) - \gamma_k (1 - L\gamma_k) \|\nabla f(y_k)\|^2 + \frac{M^2 L \Gamma_k}{2} \sum_{t=1}^{k} \frac{(\gamma_t - \beta_t)^2}{\Gamma_t \alpha_t} \\
- \gamma_k \langle \nabla f(x_{k-1}) - L\gamma_k \nabla f(y_k), G(y_k, \xi_k) - \nabla f(y_k) \rangle + 2M^2 L \gamma_k^2 \\
\leq f(x_{k-1}) - \gamma_k (1 - L\gamma_k) \|\nabla f(y_k)\|^2 + \frac{M^2 L \Gamma_k}{2} \sum_{t=1}^{k} \frac{(\gamma_t - \beta_t)^2}{\Gamma_t \alpha_t} \\
- \gamma_k \langle \nabla f(x_{k-1}), G(y_k, \xi_k) - \nabla f(y_k) \rangle + 4M^2 L \gamma_k^2,
\]

where the last inequality is due to (36). We next analyze the inner product on the right-hand side of (61)

\[
- \gamma_k \langle \nabla f(x_{k-1}), G(y_k, \xi_k) - \nabla f(y_k) \rangle \\
= -\gamma_k \langle \nabla f(x_{k-\tau(\gamma_k)}), G(y_k, \xi_k) - \nabla f(y_k) \rangle - \gamma_k \langle \nabla f(x_{k-1}) - \nabla f(x_{k-\tau(\gamma_k)}), G(y_k, \xi_k) - \nabla f(y_k) \rangle.
\]

First, we denote by \(\mathcal{F}_k\) the filtration containing all the history generated by the algorithm up to time \(k\). Using Assumption 2 (Eq. (8)) we consider

\[
\mathbb{E}[\gamma_k \langle \nabla f(x_{k-\tau(\gamma_k)}), G(y_k, \xi_k) - \nabla f(y_k) \rangle | \mathcal{F}_{k-\tau(\gamma_k)}] \\
= -\gamma_k \langle \nabla f(x_{k-\tau(\gamma_k)}), \mathbb{E}[G(y_k, \xi_k) - \nabla f(y_k) | \mathcal{F}_{k-\tau(\gamma_k)}] \rangle \\
\leq M\gamma_k \mathbb{E}[\|G(y_k, \xi_k) - \nabla f(y_k) | \mathcal{F}_{k-\tau(\gamma_k)}\|] \\
\leq M\gamma_k^2.
\]

Second, using (36), Assumption 4 and since \(\gamma_k\) is nonnegative and nonincreasing we obtain

\[
\gamma_k \langle \nabla f(x_{k-1}) - \nabla f(x_{k-\tau(\gamma_k)}), G(y_k, \xi_k) - \nabla f(y_k) \rangle \leq 2LM\gamma_k \|x_{k-1} - x_{k-\tau(\gamma_k)}\| \\
\leq 2LM\gamma_k \sum_{t=k-\tau(\gamma_k)+1}^{k-1} \|x_t - x_{t-1}\| = 2LM\gamma_k \sum_{t=k-\tau(\gamma_k)+1}^{k-1} \|\gamma_t G(y_t, \xi_t)\| \leq 2LM^2 \tau(\gamma_k) \gamma_k - \tau(\gamma_k) \gamma_k.
\]

Taking the expectation on both sides of (62) and using the preceding two relations we obtain

\[
\mathbb{E}[-\gamma_k \langle \nabla f(x_{k-1}), G(y_k, \xi_k) - \nabla f(y_k) \rangle] \leq M\gamma_k^2 + 2LM^2 \tau(\gamma_k) \gamma_k - \tau(\gamma_k) \gamma_k,
\]

which by taking the expectation on both sides of (61) and using the equation above immediately yields (37).
B  More details of environments in Section 5

We present all parameters setup for all environments and algorithms in Table 2. The network parameters are specified as \((\text{input} \times \text{hidden layers} \times \text{output})\), i.e. a \(4 \times 8 \times 2\) network contains 1 hidden layer of 8 neurons.

Table 2: Parameters for Acrobot-v1, CartPole-v0, Swimmer-v2, HalfCheetah-v2, and Ant-v2 environments.

| Environment   | Algorithm     | Policy Network | Discount Factor | Episode Length | Baseline | Batch Size | Learning Rate |
|---------------|---------------|----------------|----------------|----------------|----------|------------|--------------|
| CartPole-v0   | REINFORCE     | 4x8x2          | 0.99           | 200            | None     | 25         | 0.1          |
|               | REINFORCE-Acc |                |                |                |          |            |              |
| Acrobot-v1    | REINFORCE     | 6x16x3         | 0.99           | 500            | None     | 25         | 0.1          |
|               | REINFORCE-Acc |                |                |                |          |            |              |
| Swimmer-v2    | REINFORCE     | 8x32x32x2      | 0.99           | 1000           | Linear   | 100        | 0.01         |
|               | REINFORCE-Acc |                |                |                |          |            |              |
| HalfCheetah-v2| REINFORCE     | 17x32x32x6     | 0.99           | 1000           | Linear   | 100        | 0.05         |
|               | REINFORCE-Acc |                |                |                |          |            |              |
| Ant-v2        | REINFORCE     | 111x128x64x32x8| 0.99           | 1000           | Linear   | 100        | 0.01         |
|               | REINFORCE-Acc |                |                |                |          |            |              |

Table 3 specifies the descriptions of 5 environments used in this paper including the observation space and action space.

Table 3: Descriptions of environments used for numerical experiments.

| Environment   | Observation Space | Action Space | Action Type | Descriptions                                                                                     |
|---------------|-------------------|--------------|-------------|--------------------------------------------------------------------------------------------------|
| Acrobot-v1    | 2                 | 3            | Discrete    | The Acrobot-v1 environment contains two joints and two links where we can actuate the joints between two links. The links are hanging downwards at the beginning and the goal is to swing the end of the lower link up to a given height. |
| CartPole-v0   | 4                 | 2            | Discrete    | A pole is attached by an un-actuated joint to a cart moving along a frictionless track. The pole starts upright, and the goal is to prevent it from falling over. |
| Swimmer-v2    | 8                 | 2            | Continuous  | The goal is to make a four-legged creature walk forward as fast as possible.                      |
| HalfCheetah-v2| 17                | 6            | Continuous  | Make a two-legged creature move forward as fast as possible.                                      |
| Ant-v2        | 111               | 8            | Continuous  | Make a four-legged creature walk forward as fast as possible.                                     |