Efficiency Ordering of Stochastic Gradient Descent

Anonymous Author(s)
Affiliation
Address
email

Abstract

We consider the stochastic gradient descent (SGD) algorithm driven by a general stochastic sequence, including *i.i.d* noise and random walk on an arbitrary graph, among others; and analyze it in the asymptotic sense. Specifically, we employ the notion of 'efficiency ordering', a well-analyzed tool for comparing the performance of Markov Chain Monte Carlo (MCMC) samplers, for SGD algorithms in the form of Loewner ordering of covariance matrices associated with the scaled iterate errors in the long term. Using this ordering, we show that input sequences that are more efficient for MCMC sampling also lead to smaller covariance of the errors for SGD algorithms in the limit. This also suggests that an arbitrarily weighted MSE of SGD iterates in the limit becomes smaller when driven by more efficient chains. Our finding is of particular interest in applications such as decentralized optimization and swarm learning, where SGD is implemented in a random walk fashion on the underlying communication graph for cost issues and/or data privacy. We demonstrate how certain non-Markovian processes, for which typical mixing-time based non-asymptotic bounds are intractable, can outperform their Markovian counterparts in the sense of efficiency ordering for SGD. We also show the utility of our method by applying it to gradient descent with shuffling and mini-batch gradient descent, reaffirming key results from existing literature under a unified framework.

1 Introduction

Stochastic gradient descent (SGD) is widely used in machine learning, signal processing and other engineering fields to solve the optimization problem

$$\theta^* = \arg\min_{\theta\in\Theta} \left\{ f(\theta) \triangleq \frac{1}{n}\sum_{i=1}^{n} F(\theta, i) \right\}, \tag{1}$$

where $\Theta \subset \mathbb{R}^d$ is some closed and convex set, and $F(\cdot, i) : \mathbb{R}^d \to \mathbb{R}$ for $i \in [n] \triangleq \{1, \cdots, n\}$ are smooth functions on Θ, not necessarily convex, such that their summation $f : \mathbb{R}^d \to \mathbb{R}$ exhibits a minimizer $\theta^* \in \Theta$ satisfying $\nabla f(\theta^*) = 0$. The update rule of the iterative SGD scheme is of the form

$$\theta_{t+1} = \text{Proj}_\Theta \left(\theta_t - \gamma_{t+1}\nabla_\theta F\left(\theta_t, X_{t+1}\right)\right), \tag{2}$$

where γ_t is the step size that can be constant or diminishing as $t \to \infty$, Proj_Θ is a projection operator onto the constraint set Θ, and $\{X_t\}_{t\geq 0}$ is some sequence taking values in $[n]$. This sequence is often generated in a stochastic manner, and samples can be drawn from temporally independent and identically distributed *(i.i.d)* random variables that are either uniformly distributed over $[n]$ [53, 48, 11], or leverage importance sampling techniques for variance reduction [46, 9, 24]. $\{X_t\}_{t\geq 0}$ can also be constructed by repeatedly shuffling over all possible states without repetition,[1] leading to faster convergence than stochastic counterparts drawing *i.i.d* samples from $[n]$ [57, 3, 29, 66].

[1] One complete pass over the entire set $[n]$ is typically called an epoch. Shuffling can refer to passing over $[n]$ in the same order for every epoch (single shuffling), or in a random order (random shuffling).

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

Random Walk Stochastic Gradient Descent (RWSGD): Some applications observe restricted access to the state space, such as decentralized optimization [58, 64, 40], where communication occurs between nodes in a network to collaboratively solve the optimization problem (1). For instance, disease classification in confidential clinical swarm learning [62] considers peer-to-peer networks due to the highly private nature of medical data. In such a setting, the random sequence $\{X_t\}_{t\geq 0}$ is usually realized as a Markov chain on a general graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ that only samples local gradients of the nodes in $\mathcal{V} \triangleq [n]$ and traverses the network via edges connecting them without divulging the update history or its own gradient. The randomness of the communication path ensures that the compromised node can not easily leak the data of its neighbors [40].

Apart from the privacy concern, such dynamics are also employed in swarm learning/optimization in robotics [14] and wireless sensor networks [38] due to their low communication cost and asynchronous nature. The need for data privacy and demand for communication-efficient algorithms for decentralized optimization has spurred the study of RWSGD algorithms in recent years [51, 31, 60], with the underlying Markov chain in the form of Metropolis-Hasting random walk (MHRW) [42].

Common analytical approach - Finite time bounds based on mixing time: Most of the existing works analyzing iteration (2) provide so-called finite-time upper bounds on expected error in either the objective function $\mathbb{E}[f(\tilde{\theta}_t) - f(\theta^*)]$, where $\tilde{\theta}_t$ is some weighted average of the iterates, or its gradient $\mathbb{E}[\|\nabla f(\theta_t)\|_2^2]$; and are used to infer the convergence rate of the iterate sequence [51, 22, 60]. For diminishing step sizes $\gamma_t = t^{-\alpha}$ with $\alpha \in (0.5, 1)$,[2] the upper bound on $\mathbb{E}[\|\nabla f(\theta_t)\|_2^2]$ reads as

$$\mathbb{E}[\|\nabla f(\theta_t)\|_2^2] \leq O\left(\frac{\max\{M, 1/\log(1/\beta)\}}{t^{1-\alpha}}\right), \tag{3}$$

and a similar form for $\mathbb{E}[f(\tilde{\theta}_t) - f(\theta^*)]$ as well [60, 21, 6]. Here, $\beta \in (0, 1)$ is the second largest eigenvalue modulus (SLEM) of the underlying Markov chain's transition matrix and is related to its mixing time property, since smaller SLEM leads to faster mixing of the Markov chain [15, 39]. On the other hand, $M > 0$ is usually a quantity proportional to the local gradients evaluated at the minimizer, or their upper bound. Both the gradient information and the mixing time play a key role in quantifying the convergence rate derived from this upper bound, and the mixing time is especially important since it hints that convergence rate of the SGD algorithm can potentially be accelerated using faster mixing Markov chains for the input driving sequence. It has also been noted that the inherent correlation of the underlying random walk has to be addressed in any analysis concerning Markov-chain-driven gradient descent [60]. The mixing time technique, by capturing the rate at which the chain converges to its stationary distribution [15, 39], is one way of doing so.

Alternative approach - Asymptotic analysis and efficiency ordering: In addition to the aforementioned mixing time, another widely used metric for characterizing the second order properties of Markov chains is the asymptotic variance (AV). For any scalar valued function $g : [n] \to \mathbb{R}$, the estimator $\hat{\mu}_t(g) \triangleq \frac{1}{t}\sum_{i=1}^{t} g(X_i)$, associated with an irreducible Markov chain $\{X_t\}_{t\geq 0}$ with stationary distribution $\boldsymbol{\pi}$, is the average of the samples of $g(\cdot)$ obtained along the chain's sample path up to time $t > 0$. The AV of the Markov chain, denoted by $\sigma_X^2(g)$, is then defined as the the limiting variance of the estimator; that is,

$$\sigma_X^2(g) \triangleq \lim_{t\to\infty} t \cdot \text{Var}(\hat{\mu}_t(g)). \tag{4}$$

For all functions $g(\cdot)$ satisfying $\mathbb{E}_{\boldsymbol{\pi}}(g^2) < \infty$, the AV is associated with the Central Limit Theorem (CLT) for any Markovian kernel on a finite state space, as the variance of the normally distributed estimates in the limit [54, 32, 15]. More formally, we have

$$\sqrt{t} \cdot [\hat{\mu}_t(g) - \mathbb{E}_{\boldsymbol{\pi}}(g)] \xrightarrow[t\to\infty]{dist} \mathcal{N}(0, \sigma_X^2(g)). \tag{5}$$

A smaller AV means that fewer samples are required *post* mixing of the chain[3] in order to obtain a desired accuracy - in some sense quantifying the chain's *efficiency*.

Both the AV and the mixing time of a Markov chain are very strongly related concepts[4]. In fact, the AV has an upper bound in terms of the SLEM, which decreases as the SLEM gets smaller (chain

[2] We only need the step size to be $O(t^{-\alpha})$, but we omit the $O(\cdot)$ notation for simplicity. We also consider a slightly more general case, allowing for $\alpha = 1$ as well.

[3] Achieved by employing a burn-in period to get rid of the correlation with the initial state [26].

[4] For reversible Markov chains, the AV can be written explicitly as an increasing function of *every* eigenvalue of the transition matrix [15], while the mixing time is related to the SLEM as mentioned earlier.

mixes faster) [44]. However, an ordering of the SLEM between two Markov chains does not imply an ordering of their AV, as we shall demonstrate later in Section 4 for a special case. Both of these second-order properties therefore lead to different notions of optimality; and the comparison of two chains based on their AV leads to the concept of *efficiency ordering* [44], where we say that a chain is more efficient than the other if it has a smaller AV, uniformly over all functions $g : [n] \to \mathbb{R}$.

As mentioned earlier, the common intuition asserted by finite time bounds such as (3) is that Markov chains with smaller SLEM lead to faster convergence of the SGD iteration (2) to the minimizer [60, 6]. We put this logic to test by simulating the RWSGD algorithm with three different reversible Markov chains (w.r.t. uniform stationary distribution) as the stochastic inputs - the MHRW, a modification of MHRW, which is also shown in Appendix I [1] to be more *efficient* than MHRW, and the so-called 'fastest mixing Markov chain' (FMMC) as defined in [13] as the Markov chain obtained by minimizing the SLEM over the entire class of reversible chains for a given graph topology. We employ RWSGD to minimize a quadratic objective function for two underlying graphs. The exact details of the setup are deferred to Appendix I [1], and our numerical results in Figure 1 show that even though the FMMC is theoretically guaranteed to have the smallest SLEM (β_i for $i \in \{1, 2, 3\}$) of the three reversible chains simulated, it is the worst performing one with largest mean square error (MSE). Although MHRW and Modified-MHRW share the same SLEM in the lower plot of Figure 1, they still have performance differences. This is contradictory to the intuition derived from (3), and could be attributed to the finite time results providing upper bounds for *all* times $t > 0$, which may therefore not necessarily be tight. On the other hand, the performance of the chains seem to be ordered according to their AV (σ_i^2 for $i \in \{1, 2, 3\}$) evaluated for a test function. This lends credence to developing techniques based on AV, for judging the performance of different stochastic inputs for SGD, as possible alternatives to using SLEM as the sole performance metric.

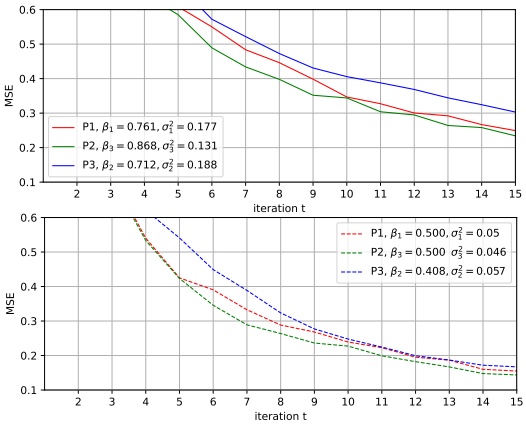


Figure 1: Comparison of MHRW (P1), Modified-MHRW (P2) and FMMC (P3) as stochastic inputs for RWSGD on two different graphs.

The asymptotic variance also appears in the CLT for stochastic approximation (SA) algorithms [8, 18], though this time not directly as the variance in the limit, but as a component of the limiting covariance matrix of the scaled iterate errors. Recent works [18, 45] point out that the covariance matrix itself is of special interest, and typically contains more information than the non-asymptotic MSE bounds [45]. In the sense of SGD algorithms, we will show in Section 3 that it embeds explicit information of the exact vector-valued gradient evaluated at the optimizer as well as the entire spectrum of the transition matrix; as opposed to the upper bound M of the gradients and only the second largest eigenvalue modulus commonly found in mixing time based non-asymptotic bounds. It has been suggested [17, 20], and also proved for the special case of linear SA [18], that the covariance matrix emerging out of the CLT dominates as a leading term of the finite-time MSE bounds. This also holds true for finite-time bounds on weighted MSE for any preferred weight; the weighted MSE being utilized in fields such as wireless MIMO [61] and process optimization [27]. Overall, while finite-time bounds have enjoyed great success in the literature, the potential for performance gains out of the asymptotic analysis of SGD algorithms have remained largely unexplored.

Contributions: We employ asymptotic analysis to propose a general framework that offers seamless connection between AV in the MCMC literature with efficiency ordering and covariance matrix in the SGD algorithms. Our framework can be used to design different random walk variants and also to systematically compare the existing sampling methods in the SGD iteration (2) with diminishing step size, not just limited to random walks. In particular, we show that any two random walks following an efficiency ordering have their covariance matrices Loewner ordered, including *non-Markovian* stochastic processes versus its Markovian counterpart, which defies any mixing-time (SLEM) based analysis. Such ordering can be harnessed into improving the accuracy of SGD iterates, which implies a reduction in the weighted MSE with arbitrary weights. Moreover, via a specific augmentation of the state space, we are able to analyze SGD for both single and random shuffling and show

the efficiency of shuffling over *i.i.d* sampling for a set of objective functions that may not satisfy 'Polyak-Łojasiewicz inequality'. We further extend such comparison to mini-batch SGD algorithms. Lastly, we present numerical results where the efficiency ordering via asymptotic analysis tends to hold over all time periods and input sequences with higher efficiency have smaller errors in SGD.

2 Modeling setup

Basic notations: We use lower case, bold faced letters to denote vectors ($\mathbf{v} \in \mathbb{R}^d$), and use upper case, bold faced letters to denote matrices ($\mathbf{M}$" $\in \mathbb{R}^{d\times d}$). $\|\cdot\|_2$ denotes the l^2 norm for vectors or 2-norm for matrices. We use $\nabla \mathbf{f}(\cdot)$ as Jacobian matrix of vector-valued function $\mathbf{f}(\cdot)$, and $\nabla^2 g(\cdot)$ as Hessian matrix of scalar-valued function $g(\cdot)$. We let $\nabla g(\theta, X)$ be the partial derivative of scalar-valued function $g(\theta, X)$ with respect to θ. Loewner ordering of matrices is denote by '$\leq_L$' such that $\mathbf{A} \leq_L \mathbf{B} \iff \mathbf{x}^T(\mathbf{A}-\mathbf{B})\mathbf{x} \leq 0$ for any $\mathbf{x} \in \mathbb{R}^d$. The term $Tr(\mathbf{A})$ denotes the trace of matrix $\mathbf{A}$, and let $\mathbb{1}_{\{\cdot\}}$ be the indicator function. We write $\mathcal{N}(0, \mathbf{V})$ to represent a multivariate Gaussian distribution with zero mean and covariance matrix $\mathbf{V}$. For a connected and undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ with node set $\mathcal{V}$ and edge set $\mathcal{E}$, we use $N(i)$ for the set of neighbors of node $i \in \mathcal{V}$ and $\mathbf{d} \triangleq [d_1, d_2, \cdots, d_n]^T$ for the degree vector where $d_i = |N(i)|$.

SGD algorithm with arbitrary input sequence: We consider random walks $\{X_t\}_{t\geq 0}$ for which the limit $\pi_i \triangleq \lim_{t\to\infty} \frac{1}{t}\sum_{k=1}^{t} \mathbb{1}_{\{X_k=i\}}$ exists almost surely and is positive for all $i \in [n]$, with $\boldsymbol{\pi} = [\pi_i]_{i\in[n]}$ denoting the limiting or *stationary* distribution. This is trivially satisfied via strong law of large numbers [23] when X_t for each $t > 0$ are *i.i.d* random variables with distribution $\boldsymbol{\pi}$ over $[n]$, and via the ergodic theorem [15] when $\{X_t\}_{t\geq 0}$ is an irreducible, aperiodic and positive recurrent (ergodic) Markov chain. Note however that this way of defining the stationary distribution $\boldsymbol{\pi}$ allows for the input sequence $\{X_t\}_{t\geq 0}$ to be more general, possibly being non-Markov on $[n]$. Then, we can use $\boldsymbol{\pi}$ to rewrite the objective in (1) as

$$f(\theta) = \frac{1}{n}\sum_{i=1}^{n} F(\theta, i) = \mathbb{E}_{X\sim\boldsymbol{\pi}}\left[G(\theta, X)\right], \tag{6}$$

where function $G(\theta, i) \triangleq \frac{1}{n\pi_i} F(\theta, i)$ for any $\theta \in \Theta, i \in [n]$. The generalized update rule then becomes

$$\theta_{t+1} = \text{Proj}_{\Theta}\left(\theta_t - \gamma_{t+1}\nabla G\left(\theta_t, X_{t+1}\right)\right). \tag{7}$$

This change of notation allows us to consider input sequences having possibly non-uniform stationary distributions, and is a version of importance sampling for RWSGD schemes, as in [6]. For example, the iteration (7) with the input sequence generated from a MHRW with uniform target distribution $\boldsymbol{\pi} = \mathbf{1}/n$ will reduce down to (2) with $G(\theta, i) = F(\theta, i)$ for all $\theta \in \Theta$, $i \in [n]$. If the input sequence is instead a simple random walk on a connected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ with $\mathcal{V} = [n]$, we have $\boldsymbol{\pi} \propto \mathbf{d}$, and $G(\theta, i) = \frac{\mathbf{1}^T\mathbf{d}}{nd_i} F(\theta, i)$ for all $\theta \in \Theta$, $i \in \mathcal{V}$.[5]

Asymptotic covariance matrix. We now quickly review the multivariate CLT for Markov chains, since it is a natural way to introduce the *asymptotic covariance* matrix, used heavily throughout the paper. For any finite, irreducible Markov chain $\{X_t\}_{t\geq 0}$ with stationary distribution $\boldsymbol{\pi}$, its *estimator* is defined as $\hat{\mu}_t(\mathbf{g}) \triangleq \frac{1}{t}\sum_{k=1}^{t} \mathbf{g}(X_t)$ for any vector-valued function $\mathbf{g} : [n] \to \mathbb{R}^d$. Then, the ergodic theorem [15, 16] states that for any initial distribution and any $\mathbf{g}(\cdot)$ such that $\mathbb{E}_{\boldsymbol{\pi}}(\mathbf{g}) = \sum_{i\in[n]} \mathbf{g}(i)\pi_i < \infty$, we have $\hat{\mu}_t(\mathbf{g}) \xrightarrow[t\to\infty]{a.s.} \mathbb{E}_{\boldsymbol{\pi}}(\mathbf{g})$. Similarly to the asymptotic variance $\sigma_X^2(g)$ for a scalar-valued function $g(\cdot)$, we can also define the *asymptotic covariance* matrix $\boldsymbol{\Sigma}_X(\mathbf{g})$ for vector-valued function $\mathbf{g}(\cdot)$,

$$\boldsymbol{\Sigma}_X(\mathbf{g}) \triangleq \lim_{t\to\infty} t\cdot \text{Var}(\hat{\mu}_t(\mathbf{g})) = \lim_{t\to\infty} \frac{1}{t}\cdot \mathbb{E}\left\{\Delta_t\Delta_t^T\right\}, \tag{8}$$

where $\Delta_t \triangleq \sum_{s=1}^{t}(\mathbf{g}(X_s) - \mathbb{E}_{\boldsymbol{\pi}}(\mathbf{g}))$. The associated multivariate CLT is then given as follows.

Theorem 2.1 (Chapter 1 [16]). *For any function $\mathbf{g} : [n] \to \mathbb{R}^d$ that satisfies $\mathbb{E}_{\boldsymbol{\pi}}(\mathbf{g}^2) < \infty$, we have*

$$\sqrt{t}\cdot\left[\hat{\mu}_t(\mathbf{g}) - \mathbb{E}_{\boldsymbol{\pi}}(\mathbf{g})\right] \xrightarrow[t\to\infty]{dist} \mathcal{N}(0, \boldsymbol{\Sigma}_X(\mathbf{g})). \qquad \square$$

In the next section, we will show how the the asymptotic covariance matrix $\boldsymbol{\Sigma}_X(\cdot)$ also appears as part of the CLT result for SGD algorithms.

[5] In practice, knowing π_i up to a multiplicative constant is enough to converge to the optimal point.

3 Efficiency Ordering of SGD Algorithms

In this section, we present our main result concerning the performance comparison of different SGD algorithms to solve (1). We first begin by stating our assumptions on the objective function and the stochastic input sequence, providing a CLT result for SGD algorithms, and analyzing the covariance matrix arising therein. We then introduce the notion of *efficiency ordering* of Markov chains in the context of MCMC sampling, and form the connection with covariance matrices as our main result in Theorem 3.6.

For the rest of this section we assume that the functions $F(\cdot, i)$ (possibly non-convex), the summands of the objective function in (1), and the input process $\{X_t\}_{t\geq 0}$ for the SGD iteration (7) satisfy:

- (A1) The step size is given by $\gamma_t = t^{-\alpha}$ for $\alpha \in (1/2, 1]$;
- (A2) There exists a unique minimizer θ^* in the interior of the compact set Θ with $\nabla f(\theta^*) = 0$, and matrix $\nabla^2 f(\theta^*)$ (resp. $\nabla^2 f(\theta^*) - \mathbf{I}/2$) is positive definite for $a \in (1/2, 1)$ (resp. $a = 1$);
- (A3) Gradients are bounded in the compact set Θ, that is, $\sup_{\theta\in\Theta} \sup_{i\in[n]} \|\nabla F(\theta, i)\|_2 < \infty$;
- (A4) For every $z \in [n], \theta \in \mathbb{R}^d$, the solution $\tilde{F}(\theta, z) \in \mathbb{R}^d$ of the Poisson equation $\tilde{F}(\theta, z) - \mathbb{E}[\tilde{F}(\theta, X_{t+1}) \mid X_t = z] = \nabla F(\theta, z) - \nabla f(\theta)$ exists, and $\sup_{\theta\in\Theta, z\in[n]} \|\tilde{F}(\theta, z)\|_2 < \infty$;
- (A5) The functions $F(\theta, i)$ are L-smooth for all $i \in [n]$, that is, $\forall \theta_1, \theta_2 \in \Theta, \forall i \in [n]$, we have $\|\nabla F(\theta_1, i) - \nabla F(\theta_2, i)\|_2 \leq L\|\theta_1 - \theta_2\|_2$.

We then have the following CLT result for SGD algorithms.

Lemma 3.1. *For iterates $\{\theta_t\}_{t\geq 0}$ of the SGD algorithm* (7) *satisfying (A1)–(A5), we have*

$$\theta_t \xrightarrow[t\to\infty]{a.s.} \theta^*, \quad \textit{and} \quad (\theta_t - \theta^*)/\sqrt{\gamma_t} \xrightarrow[t\to\infty]{Dist} \mathcal{N}(0, \mathbf{V}_X), \tag{9}$$

where covariance matrix $\mathbf{V}_X$ is the unique solution to the Lyapunov equation $\mathbf{\Sigma}_X + \mathbf{K}\mathbf{V}_X + \mathbf{V}_X\mathbf{K}^T = \mathbf{0}$ when $\alpha \in (0.5, 1)$ (resp. $\mathbf{\Sigma}_X + \big(\mathbf{K} + \frac{\mathbf{I}}{2}\big)\mathbf{V}_X + \mathbf{V}_X\big(\mathbf{K} + \frac{\mathbf{I}}{2}\big)^T = \mathbf{0}$) when $\alpha = 1$). Here, $\mathbf{\Sigma}_X \triangleq \mathbf{\Sigma}_X(\nabla F(\theta^*, \cdot))$ is the asymptotic covariance matrix[6] *as in* (8)*, and $\mathbf{K} \triangleq \nabla^2 f(\theta^*)$.*

Additionally, for the averaged iterates $\{\bar{\theta}_t\}_{t\geq 0}$ where $\bar{\theta}_t \triangleq \frac{1}{t}\sum_{i=0}^{t-1} \theta_t$, we have

$$\bar{\theta}_t \xrightarrow[t\to\infty]{a.s.} \theta^*, \quad \textit{and} \quad \sqrt{t}(\bar{\theta}_t - \theta^*) \xrightarrow[t\to\infty]{Dist} \mathcal{N}(0, \mathbf{V}'_X), \tag{10}$$

where $\mathbf{V}'_X = \mathbf{K}^{-1}\mathbf{\Sigma}_X(\mathbf{K}^{-1})^T$ with the same matrices $\mathbf{K}$ and $\mathbf{\Sigma}_X$ as in the non-averaged case. □

Remark 3.2. Lemma 3.1 is itself a special case of the more general CLT result for SA algorithms provided in Appendix A [1], and as proved in Appendix B [1]. □

Remark 3.3. While (A2) may appear to be too strict at first, it can be relaxed to the setting of the objective function $f(\cdot)$ having multiple minimizers, by leveraging more general CLT results from SA literature, such as Theorem 2.1 in [25]. However, this comes at a cost of cumbersome notation, requiring conditioning of iterates converging to one of the minimizers, potentially making the mathematical parts harder to follow. We also show in Appendix C [1] that (A2) is no stricter than the Polyak-Łojasiewicz inequality – a popularly adopted weak assumption in recent SGD literature studying non-convex objective functions [33, 41, 63, 66]. □

Remark 3.4. Assumptions (A3) and (A5) are widely seen in the RWSGD literature [51, 31, 60], while (A4) is automatically satisfied for any ergodic Markov chain (see [43, 18] for details), a common assumption for the stochastic noise sequence [31, 22, 6]. The compactness in (A3) can also be relaxed, given assumptions on the objective function in [34], such that the estimator θ_t generated by Markov-driven sequences can still be 'locked in' a compact set after a sufficiently long time. □

Lemma 3.1 implicitly indicates that the asymptotic convergence rate (in distribution) for $\theta_t - \theta^*$ (resp. $\bar{\theta}_t - \theta^*$) is $O(\sqrt{\gamma_t})$ (resp. $O(1/\sqrt{t})$). While this does not necessarily translate to $O(\sqrt{\gamma_t})$ convergence rate for $\mathbb{E}[\|\theta_t - \theta^*\|_2]$ ($O(1/\sqrt{t})$ for $\mathbb{E}[\|\bar{\theta}_t - \theta^*\|_2]$), it has been suggested [17, 20], and

[6] We slightly abuse the notation and shorten $\mathbf{\Sigma}_X(\nabla F(\theta^*, \cdot))$, that is, the asymptotic covariance matrix evaluated at $\nabla F(\theta^*, \cdot))$, to $\mathbf{\Sigma}_X$ for better readability.

is in fact true for cases such as quadratic objective functions since they satisfy the linear stochastic approximation in [18], which is of the form

$$\theta_{t+1} = \theta_t - \gamma_{t+1}(\mathbf{A}\theta_t - \mathbf{b}(X_{t+1})), \tag{11}$$

for which the connection between finite-time MSE and covariance matrix $\mathbf{V}_X$ has been established [18]. This is also true for arbitrarily weighted MSE, which can be obtained as a weighted sum of diagonal entries of the covariance matrices $\mathbf{V}_X$ and $\mathbf{V}'_X$.

In addition to the apparent connection to MSE, the covariance matrix plays a wider role in SGD performance. Given any vector of weights $\mathbf{w} \in \mathbb{R}^d$, from Lemma 3.1 we also have that the weighted sum of errors $\mathbf{w}^T(\theta_t - \theta^*)$ converges to zero almost surely, and that $\mathbf{w}^T(\theta_t - \theta^*)/\sqrt{\gamma_t} \xrightarrow[t\to\infty]{Dist} \mathcal{N}(0, \mathbf{w}^T\mathbf{V}_X\mathbf{w})$. This means that, for sufficiently large t, we can estimate

$$P\left(\frac{\mathbf{w}^T(\theta_t - \theta^*)}{\sqrt{\gamma_t \mathbf{w}^T\mathbf{V}_X\mathbf{w}}} > \alpha\right) \approx \frac{1}{2\pi}\int_\alpha^\infty e^{-x^2/2}dx,$$

such that, for instance, the 95% confidence interval for $\mathbf{w}^T\theta_t$ is approximately $\mathbf{w}^T\theta^* \pm 2\sqrt{\gamma_t \mathbf{w}^T\mathbf{V}_X\mathbf{w}}$. In other words, smaller $\mathbf{w}^T\mathbf{V}_X\mathbf{w}$ leads to narrower confidence interval and higher accuracy. The form $\mathbf{w}^T\mathbf{V}_X\mathbf{w}$ for any vector $\mathbf{w} \in \mathbb{R}^d$ naturally implies that Loewner ordering should come into play when concerning the performance of SGD algorithms.

To proceed, we first employ the widely used notion of *efficiency ordering* of Markov chains. The efficiency of different chains is compared by ordering them using their respective AV as follows.

Definition 3.5 (**Efficiency Ordering** [44])**.** For two random walks $\{X_t\}_{t\geq 0}$ and $\{Y_t\}_{t\geq 0}$ with the same stationary distribution $\boldsymbol{\pi}$, we say $\{X_t\}_{t\geq 0}$ is more *efficient* than $\{Y_t\}_{t\geq 0}$, which we write as $X \geq_E Y$, if and only if $\sigma^2_X(g) \leq \sigma^2_Y(g)$ for any $g : [n] \to \mathbb{R}$. □

We are now ready to state our main result. We first extend the efficiency ordering of Markov chains by proving the equivalence of comparing their scalar-valued AVs, to comparing their asymptotic covariance matrices via Loewner ordering. We then use this extension to show that more efficient inputs $\{X_t\}_{t\geq 0}$ (as in Definition 3.5) to the SGD algorithm lead to performance improvements in the form of smaller covariance matrices in the Loewner ordering sense.

Theorem 3.6. *Consider two SGD iterations with random walks $\{X_t\}_{t\geq 0}$ and $\{Y_t\}_{t\geq 0}$ as input sequences, with the same stationary distribution $\boldsymbol{\pi}$, satisfying (A1)–(A5). Then,*

(i) $X \geq_E Y$ if and only if $\boldsymbol{\Sigma}_X \leq_L \boldsymbol{\Sigma}_Y$;

(ii) If $X \geq_E Y$, then $\mathbf{V}_X \leq_L \mathbf{V}_Y$ ($\mathbf{V}'_X \leq_L \mathbf{V}'_Y$ for the case of averaged iterates);

where $\boldsymbol{\Sigma}_X, \mathbf{V}_X$, and $\mathbf{V}'_X$ (resp. $\boldsymbol{\Sigma}_Y, \mathbf{V}_Y$, and $\mathbf{V}'_Y$) are the covariance matrices from Lemma 3.1, corresponding to $\{X_t\}_{t\geq 0}$ (resp. $\{Y_t\}_{t\geq 0}$) as the stochastic input sequence. □

Theorem 3.6 enables us to provide a sense of *efficiency ordering of SGD algorithms* which are driven by different stochastic inputs. Since this is achieved via Loewner ordering, it also leads to smaller confidence intervals in the long run as mentioned earlier, as well as potentially smaller MSE[7] depending on the objective function.

Remark 3.7. In addition to the CLT result for SGD algorithms with diminishing step size described in Lemma 3.1, we include in Appendix E [1] similar results for constant step sizes and quadratic objective functions, where the statement of Theorem 3.6 still holds. □

4 Applications: Towards More Efficient SGD

In this section, we present some SGD variants and compare them in terms of efficiency ordering of SGD. Specifically, we first show that a certain class of non-Markov random walks can provide a better input sequence than its Markovian counterpart. We then analyze shuffling-based gradient

[7]The mean square error can be retrieved as the trace of the covariance matrix (weighted sum of its diagonal entries in case of weighted MSE). Loosely speaking, an iterate having a smaller covariance matrix in the Loewner ordering will then also have a smaller MSE (weighted MSE).

descent and compare it to the SGD with *i.i.d* input in terms of efficiency ordering for SGD algorithm. We also extend our approach to a more general mini-batch version.

High-Order Efficient Random Walk for SGD: The simple random walk (SRW) is a popular Markov chain that has been extensively studied in the literature [55, 52, 26]. Several recent works have focused on the non-backtracking random walk (NBRW) on a connected undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ in the MCMC literature, which is an extension of SRW with the same limiting distribution $\boldsymbol{\pi} = \mathbf{d}/\mathbf{1}^T\mathbf{d}$ [47, 5, 37, 35, 7]. Intuitively speaking, NBRW is a random walk that selects one of its neighbors uniformly at random *except* the one it just came/transitioned from. Specifically, the NBRW $\{Y_t\}_{t\geq 0}$ is a second-order non-reversible Markov chain (i.e., it is non-Markov on $\mathcal{V} = [n]$) with its transition probability given by

$$P(Y_{t+1} = j | Y_t = i, Y_{t-1} = k) = \begin{cases} \frac{1}{d_i - 1} & \text{if } j \neq k, j \in N(i), d_i > 1, \\ 1 & \text{if } d_i = 1, j \in N(i), \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

From the nature of same limiting distribution, NBRW can be used as the input for SGD iterations (7) with the same re-weighted local functions $G(\theta^*, i)$ as that of SRW for all $i \in [n]$ whenever the applications call for random-walk type of inputs. Let $\boldsymbol{\Sigma}_Y(\nabla G(\theta^*, \cdot))$ be the asymptotic covariance matrix of this NBRW $\{Y_t\}_{t\geq 0}$, as defined in (8). One of the main results in [37] concerns the efficiency ordering of NBRW and SRW. They show that NBRW has a smaller AV, or equivalently, from our Theorem 3.6 (i), a smaller asymptotic covariance in terms of Loewner ordering. Our next result forms the necessary connection between the asymptotic covariance matrix arising in the CLT result and $\boldsymbol{\Sigma}_Y(\nabla G(\theta^*, \cdot))$.

Proposition 4.1. *Consider the SGD iteration* (7) *with two input sequences SRW* $\{X_t\}_{t\geq 0}$ *and NBRW* $\{Y_t\}_{t\geq 0}$ *respectively. Then, both the respective estimators* $\theta_t^X, \theta_t^Y \xrightarrow[t\to\infty]{a.s.} \theta^*$*, and* $\mathbf{V}_Y \leq_L \mathbf{V}_X$*, that is, NBRW is more efficient than SRW in the SGD algorithm.* □

By augmenting the state space, we can represent NBRW as a Markov chain $Z_t = (Y_{t-1}, Y_t) \in \mathcal{V} \times \mathcal{V}$, as was done in [47, 37]. This transformation then allows us to build CLT for an SGD iteration with $\{Z_t\}_{t\geq 0}$ as the input. The subtlety here is to prove that the asymptotic covariance matrix arising out of the CLT with respected to the augmented process $\{Z_t\}_{t\geq 0}$ is indeed equal to $\boldsymbol{\Sigma}_Y(\nabla G(\theta^*, \cdot))$. This is shown by cultivating the relationship between the stationary distribution of $\{Z_t\}_{t\geq 0}$ on the augmented state space $\mathcal{V} \times \mathcal{V}$ and $\{Y_t\}_{t\geq 0}$ on the node space $\mathcal{V}$, as provided in [47].

Thus, our Theorem 3.6 together with the existing works on efficiency ordering of NBRW versus SRW in the MCMC literature [47, 37] enable us to show that NBRW is a more efficient input sequence than SRW for the SGD iteration (7). Interestingly, it has been shown that non-backtracking walks mix faster when the underlying graph is $d-$regular [5]. In this case, a faster convergence rate is also suggested by mixing time based non-asymptotic bounds prevalent in RWSGD literature. However, no such results concerning mixing time and SLEM exists for NBRW on a general graph. Thus, in the form of Proposition 4.1, we demonstrate the utility of our approach in settings where mixing time based comparisons are unavailable.

Shuffling versus *i.i.d* Input Sequence: Shuffling-based methods have been widely used in machine learning applications [10]. They work by repeatedly passing over the entire state space $[n]$ without repetition, each complete pass forming an *epoch*. *Random shuffling* and *single shuffling* are two versions therein and differ in the order in which they pass over $[n]$. Random shuffling, as the name suggests, makes the pass in a randomly chosen order in each epoch, while single shuffling maintains the same predetermined order (often randomly chosen once at the beginning) for all epochs. Shuffling-based methods are known to show better empirical performance than *i.i.d* input [12], although intense theoretical analysis for shuffling-based gradient descent has only emerged in recent years [59, 30, 57, 3, 29]. In what follows, we use our results from Section 3 to compare shuffling-based gradient descent to SGD with *i.i.d* input. To do so, we first analyze the asymptotic covariance matrix for shuffling-based methods.

Lemma 4.2. *Let the input process* $\{X_t\}_{t\geq 0}$ *be single or random shuffling. Then, for any vector-valued function* $\mathbf{g} : [n] \to \mathbb{R}^d$*,* $\boldsymbol{\Sigma}_X(\mathbf{g}) = \mathbf{0}$*, where* $\boldsymbol{\Sigma}_X(\mathbf{g})$ *is defined in* (8). □

For *i.i.d* input sequence with distribution $\hat{\boldsymbol{\pi}}$, the asymptotic covariance from Lemma 3.1 reduces to

$$\boldsymbol{\Sigma}_X(\nabla G(\theta^*, \cdot)) \triangleq \text{Var}_{X_0 \sim \hat{\boldsymbol{\pi}}}(\nabla G(\theta^*, X_0)) \tag{13}$$

following its definition in (8), and thus, trivially, $\mathbf{\Sigma}_X(\nabla G(\theta^*, \cdot)) \geq_L \mathbf{0}$. Lemma 4.2 shows that shuffling-based methods are more efficient than *i.i.d* input sequence due to a smaller asymptotic covariance matrix in Loewner ordering. Next, we show that they also outperform *i.i.d* input when used for driving the input sequence of SGD algorithms.

Proposition 4.3. *Consider the SGD iteration* (7) *with stochastic inputs single/random shuffling* $\{X_t\}_{t\geq 0}$ *and i.i.d sampling* $\{Y_t\}_{t\geq 0}$*, we have* $\theta_t^X, \theta_t^Y \xrightarrow[t\to\infty]{a.s.} \theta^*$ *and* $\mathbf{V}_X = \mathbf{0} \leq_L \mathbf{V}_Y$. □

Though it may seem so at first, Proposition 4.3 is not a simple application of Theorem 3.6, especially for random shuffling because it is hard to check if random shuffling, formulated as a time-inhomogeneous Markov chain, indeed satisfies (A4). To overcome this difficulty, in Appendix H [1] we come up with a non-trivial augmentation to a much higher dimensional state space ($[n]^{n+1}$) to make random shuffling a time-homogeneous periodic Markov chain in order to show that both single shuffling and random shuffling satisfy (A4) and thus apply Theorem 3.6.

The case of shuffling versus *i.i.d* inputs is an example of a setting where the sequence with larger SLEM is more efficient than one with smaller SLEM[8] as an input sequence to the SGD iteration (7). For quadratic objective functions that satisfy the linear SA iteration in [18], it also attains a faster convergence speed in terms of MSE than *i.i.d* inputs to SGD algorithms. Although some recent works provide more informative finite-time error bounds on the MSE of the objective function for shuffling-based methods, by studying a special case of the matrix norm AM-GM inequality and proving faster convergence rate than *i.i.d* inputs [50, 3, 29], our result is not a subset of theirs. To be precise, we show in Appendix C [1] that our assumption (A2) on the objective function is no less general than their most general setting based on the Polyak-Łojasiewicz inequality.

Mini-Batch Gradient Descent with Shuffling: Mini-batch gradient descent is another popular gradient descent variant and is widely used in the machine learning tools [19, 2, 49] to accelerate the learning process when compared to SGD. Instead of sampling a single element, mini-batch gradient descent samples multiple elements from $[n]$ in each iteration that form a batch.

To incorporate the notion of mini-batches in our SGD framework, we provide a reformulation of the general SGD iteration based on a similar formulation in [28] for the general analysis of SGD with *i.i.d* inputs. Consider a stochastic process $\{B_t\}_{t\geq 0}$ as the driving sequence, which randomly samples batches of size S (without replacement) from the state space $[n]$, that is $B_t \subset [n]$ and $|B_t| = S$ for all $t \geq 0$. Here we assume $[n] \bmod S = 0$ for simplicity. B_t will therefore refer to the batch chosen at any time $t > 0$. We assume that B_t for all $t > 0$ are *i.i.d* random variables drawn from a distribution $\mathcal{P}$, such that $\mathcal{P}(B) > 0$ is the probability with which a batch $B \subset [n]$ is picked. We associate with any batch B, $\mathbf{v}(B) \triangleq \left[\sum_{i\in B} \mathbf{e}_i\right] / \binom{N}{S} \mathcal{P}(B)$, where $\mathbf{e}_i$ is the i'th vector of the canonical basis of $\mathbb{R}^d$. We then denote $\mathbf{F}(\theta) \triangleq [F(\theta, 1), \cdots, F(\theta, n)]^T$, and $\nabla\mathbf{F}(\theta) \triangleq [\nabla F(\theta, 1), \cdots, \nabla F(\theta, n)]^T$ for all $\theta \in \Theta$. With this notation, we can rewrite the general update rule for mini-batch SGD as

$$\theta_{t+1} = \text{Proj}_\Theta \left(\theta_t - \gamma_{t+1} \nabla\mathbf{F}(\theta_t)^T \mathbf{v}(B_{t+1})\right). \tag{14}$$

Note that this way of defining the mini-batch based random input ensures that $\mathbb{E}_{\mathcal{P}}[\mathbf{F}(\theta)^T \mathbf{v}(\cdot)] = f(\theta)$ for all $\theta \in \Theta$, maintaining the same objective function irrespective of the distribution from which batches are sampled.

With $X_t = B_t$ for all $t \geq 0$, and $\nabla G(\theta_t, X_{t+1}) = \nabla\mathbf{F}(\theta_t)^T \mathbf{v}(B_{t+1})$, the iteration (14) can still be written in the form of (7) with *i.i.d* input sequence $\{X_t\}_{t\geq 0}$. We can thus apply the CLT for SGD algorithms to the mini-batch SGD with *i.i.d* input, and in a similar fashion as (13) derive the explicit form of the asymptotic covariance matrix of (14), that is,

$$\mathbf{\Sigma}_B(\nabla\mathbf{F}(\theta^*)^T \mathbf{v}(\cdot)) \triangleq \text{Var}_{B_0\sim\mathcal{P}}(\nabla\mathbf{F}(\theta^*)^T \mathbf{v}(B_0)). \tag{15}$$

In practice, mini-batch gradient descent with shuffling is more widely used than $i.i.d$ sampling [2], in which B_t is generated by shuffling-based method instead of independent drawn from a distribution.[9] At the beginning of each epoch, *Mini-batch gradient descent with random shuffling* shuffles the whole

[8] The single shuffling when realized as a periodic Markov chain has SLEM = 1 (transition matrix is unitary), while the *i.i.d* input sequence has SLEM = 0 (transition matrix is rank one).

[9] The reformulation (14) enables us to analyze mini-batch gradient descent with various stochastic processes that samples B_t, not just i.i.d input and shuffling. However, discussing general processes $\{B_t\}_{t\geq 0}$ is beyond the scope of this paper.

dataset $[n]$ and split it into small batches. On the other hand, *mini-batch gradient descent with single shuffling* only shuffles the dataset $[n]$ once before dividing it into batches, sticking to a predetermined sequence of batches for all epochs of the training process. As pointed out by [65], there is still a gap between practical implementation and theoretical analysis for mini-batch gradient descent with shuffling. Nevertheless, by extrapolating the analysis from Proposition 4.3, we are able to analyze the efficiency ordering of shuffling and *i.i.d* sampling in the mini-batch version, as stated next.

Proposition 4.4. *Consider the mini-batch gradient descent* (14) *with stochastic inputs single/random shuffling* $\{X_t\}_{t\geq 0}$ *and i.i.d sampling* $\{Y_t\}_{t\geq 0}$*, we have* $\theta_t^X, \theta_t^Y \xrightarrow[t\to\infty]{a.s.} \theta^*$ *and* $\mathbf{V}_X = \mathbf{0} \leq_L \mathbf{V}_Y$. □

Proposition 4.4 generalizes Proposition 4.3 (special case with mini-batch of size $S = 1$) in that the same efficiency ordering between shuffling and *i.i.d* input holds true even with mini-batches.

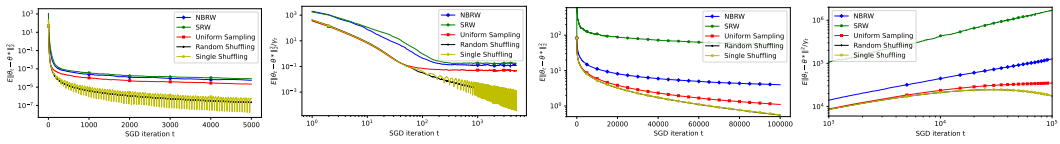
5 Numerical Experiments

In this section, we empirically validate our theoretical analysis. We compare the efficiency ordering of the SGD algorithm for various stochastic inputs on two objective functions in (16), $f(\theta)$ being strongly convex and $g(\theta)$ being nonconvex.

$$\tilde{f}(\theta) = \frac{1}{n}\sum_{i=1}^{n} \log(1+\exp(-y_i \mathbf{x}_i^T \theta)) + \frac{1}{2}\|\theta\|_2^2\,, \quad \hat{f}(\theta) = \frac{1}{n}\sum_{i=1}^{n} \theta^T(\mathbf{a}_i\mathbf{a}_i^T + \mathbf{D}_i)\theta + \mathbf{b}^T\theta. \tag{16}$$

For l_2-regularized logistic regression $\tilde{f}(\theta)$, we choose the dataset CIFAR-10 [36] where n is the total number of data points. Here, $\mathbf{x}_i \in \mathbb{R}^{108}$ is the vector flattened from the cropped image i with shape $(6, 6, 3)$, and $y_i \in \mathbb{R}$ is the label. For sum-of-non-convex functions $\hat{f}(\theta)$, which is based on the experiment setup in [28, 4], we random generate a diagonal matrix $\mathbf{D}_i \in \mathbb{R}^{10\times 10}$ and ensures $\sum_{i=1}^{n} \mathbf{D}_i = \mathbf{0}$. Vectors $\mathbf{a}_1, \cdots, \mathbf{a}_n \in \mathbb{R}^{10}$ are randomly generated and $\sum_{i=1}^{n} \mathbf{a}_i\mathbf{a}_i^T$ should be invertible. For both experiments, we assign a data point to each node i on the general graph 'Dolphins' (62 nodes) [56]. We set the step size in the SGD algorithm to be $1/t^{0.9}$, and use MSE $\mathbb{E}[\|\theta_t - \theta^*\|_2^2]$ as our performance metric to measure the relative performance of different inputs. We also employ the scaled MSE $\mathbb{E}[\|\theta_t - \theta^*\|_2^2]/\gamma_t$ to empirically show its relationship to the CLT result (9). More simulation results are deferred to Appendix I [1].

In Figure 2 we compare NBRW and SRW as input sequences on the graph 'Dolphins' for two objective functions in (16). We also compare uniform sampling, random shuffling and single shuffling, assuming that they can access any node on the graph in each iteration. We can see in Figure 2a and 2c that NBRW always falls below SRW throughout all time periods, which indicates that NBRW tends to have smaller MSE than SRW. Single and random shuffling are both better than uniform sampling in terms of smaller MSE. The oscillation of single shuffling comes from a predetermined fixed data sampling sequence, while random shuffling changes the permutation whenever traversing all nodes. Figure 2b shows that the scaled MSEs of NBRW, SRW and uniform sampling approach some constants after some time, which is consistent with the CLT result (9). The curves of NBRW are still below that of SRW, showing that the input with smaller scaled MSE tends to have higher efficiency, which supports Proposition 4.1. We can see from Figure 2d that NBRW, SRW have not yet entered the regime when the covariance matrix becomes the main factor (the curve is increasing) while uniform sampling and both shuffling methods are just entering this regime. The curves of single and random shuffling in Figure 2b and 2d fall below that of uniform sampling and still decrease because eventually their covariance matrices will be zero matrix, as indicated in Proposition 4.3.



(a) Logistic regression (metric: MSE) (b) Logistic regression (metric: scaled MSE) (c) sum of non-convex fn. (metric: MSE) (d) sum of non-convex fn. (metric: scaled MSE)

Figure 2: Performance comparison of different stochastic inputs on the graph 'Dolphins'.

References

[1] Efficiency ordering of stochastic gradient descent – supplementary material, 2022.

[2] 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 *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283, 2016.

[3] Kwangjun Ahn, Chulhee Yun, and Suvrit Sra. Sgd with shuffling: optimal rates without component convexity and large epoch requirements. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, volume 33, pages 17526–17535, 2020.

[4] Zeyuan Allen-Zhu and Yang Yuan. Improved svrg for non-strongly-convex or sum-of-non-convex objectives. In *International conference on machine learning*, pages 1080–1089. PMLR, 2016.

[5] Noga Alon, Itai Benjamini, Eyal Lubetzky, and Sasha Sodin. Non-backtracking random walks mix faster. *Communications in Contemporary Mathematics*, 9(04):585–603, 2007.

[6] Ghadir Ayache and Salim El Rouayheb. Private weighted random walk stochastic gradient descent. *IEEE Journal on Selected Areas in Information Theory*, 2(1):452–463, 2021.

[7] Anna Ben-Hamou, Eyal Lubetzky, and Yuval Peres. Comparing mixing times on sparse random graphs. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1734–1740. SIAM, 2018.

[8] Albert Benveniste, Michel Métivier, and Pierre Priouret. *Adaptive algorithms and stochastic approximations*, volume 22. Springer Science & Business Media, 2012.

[9] Zalán Borsos, Sebastian Curi, Kfir Yehuda Levy, and Andreas Krause. Online variance reduction with mixtures. In *International Conference on Machine Learning*, pages 705–714. PMLR, 2019.

[10] Léon Bottou. Curiously fast convergence of some stochastic gradient descent algorithms. Unpublished open problem offered to the attendance of the SLDS 2009 conference, 2009.

[11] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[12] Léon Bottou. Stochastic gradient descent tricks. In *Neural networks: Tricks of the trade*, pages 421–436. Springer, 2012.

[13] Stephen Boyd, Persi Diaconis, and Lin Xiao. Fastest mixing markov chain on a graph. *SIAM review*, 46(4):667–689, 2004.

[14] Manuele Brambilla, Eliseo Ferrante, Mauro Birattari, and Marco Dorigo. Swarm robotics: a review from the swarm engineering perspective. *Swarm Intelligence*, 7(1):1–41, 2013.

[15] Pierre Brémaud. *Markov chains: Gibbs fields, Monte Carlo simulation, and queues*, volume 31. Springer Science & Business Media, 2013.

[16] Steve Brooks, Andrew Gelman, Galin Jones, and Xiao-Li Meng, editors. *Handbook of Markov Chain Monte Carlo (1st ed.)*. Chapman and Hall/CRC, New York, 2011.

[17] Shuhang Chen, Adithya Devraj, Andrey Bernstein, and Sean Meyn. Accelerating optimization and reinforcement learning with quasi stochastic approximation. In *2021 American Control Conference (ACC)*, pages 1965–1972, 2021.

[18] Shuhang Chen, Adithya Devraj, Ana Busic, and Sean Meyn. Explicit mean-square error bounds for monte-carlo and linear stochastic approximation. In *International Conference on Artificial Intelligence and Statistics*, pages 4173–4183. PMLR, 2020.

[19] Francois Chollet et al. Keras, 2015.

[20] Adithya M. Devraj and Sean P. Meyn. Q-learning with uniformly bounded variance. *IEEE Transactions on Automatic Control*, 2021.

[21] Thinh T Doan, Lam M Nguyen, Nhan H Pham, and Justin Romberg. Finite-time analysis of stochastic gradient descent under markov randomness. *arXiv preprint arXiv:2003.10973*, 2020.

[22] John C Duchi, Alekh Agarwal, Mikael Johansson, and Michael I Jordan. Ergodic mirror descent. *SIAM Journal on Optimization*, 22(4):1549–1578, 2012.

[23] Rick Durrett. *Probability: theory and examples*, volume 49. Cambridge university press, 2019.

[24] Ayoub El Hanchi and David Stephens. Adaptive importance sampling for finite-sum optimization and sampling with decreasing step-sizes. In *Advances in Neural Information Processing Systems*, volume 33, pages 15702–15713, 2020.

[25] Gersende Fort. Central limit theorems for stochastic approximation with controlled markov chain dynamics. *ESAIM: Probability and Statistics*, 19:60–80, 2015.

[26] Minas Gjoka, Maciej Kurant, Carter T Butts, and Athina Markopoulou. Practical recommendations on crawling online social networks. *IEEE Journal on Selected Areas in Communications*, 29(9):1872–1892, 2011.

[27] JHF Gomes, AP Paiva, SC Costa, Pedro Paulo Balestrassi, and EJ Paiva. Weighted multivariate mean square error for processes optimization: A case study on flux-cored arc welding for stainless steel claddings. *European Journal of Operational Research*, 226(3):522–535, 2013.

[28] Robert Mansel Gower, Nicolas Loizou, Xun Qian, Alibek Sailanbayev, Egor Shulgin, and Peter Richtárik. Sgd: General analysis and improved rates. In *International Conference on Machine Learning*, pages 5200–5209. PMLR, 2019.

[29] Mert Gürbüzbalaban, Asu Ozdaglar, and Pablo A Parrilo. Why random reshuffling beats stochastic gradient descent. *Mathematical Programming*, 186(1):49–84, 2021.

[30] Jeff Haochen and Suvrit Sra. Random shuffling beats sgd after finite epochs. In *International Conference on Machine Learning*, pages 2624–2633. PMLR, 2019.

[31] Björn Johansson, Maben Rabi, and Mikael Johansson. A randomized incremental subgradient method for distributed optimization in networked systems. *SIAM Journal on Optimization*, 20(3):1157–1170, 2010.

[32] Galin L Jones. On the markov chain central limit theorem. *Probability surveys*, 1:299–320, 2004.

[33] Hamed Karimi, Julie Nutini, and Mark Schmidt. Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 795–811. Springer, 2016.

[34] Prasenjit Karmakar and Shalabh Bhatnagar. Stochastic approximation with iterate-dependent markov noise under verifiable conditions in compact state space with the stability of iterates not ensured. *IEEE Transactions on Automatic Control*, 66(12):5941–5954, 2021.

[35] Mark Kempton. Non-backtracking random walks and a weighted ihara's theorem. *Open Journal of Discrete Mathematics*, 6(4):207–226, 2016.

[36] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009.

[37] Chul-Ho Lee, Xin Xu, and Do Young Eun. Beyond random walk and metropolis-hastings samplers: why you should not backtrack for unbiased graph sampling. *ACM SIGMETRICS Performance evaluation review*, 40(1):319–330, 2012.

[38] Victor Lesser, Charles L Ortiz Jr, and Milind Tambe. *Distributed sensor networks: A multiagent perspective*, volume 9. Springer Science & Business Media, 2003.

[39] David A Levin and Yuval Peres. *Markov chains and mixing times*, volume 107. American Mathematical Soc., 2017.

[40] Xianghui Mao, Kun Yuan, Yubin Hu, Yuantao Gu, Ali H Sayed, and Wotao Yin. Walkman: A communication-efficient random-walk algorithm for decentralized optimization. *IEEE Transactions on Signal Processing*, 68:2513–2528, 2020.

[41] Panayotis Mertikopoulos, Nadav Hallak, Ali Kavis, and Volkan Cevher. On the almost sure convergence of stochastic gradient descent in non-convex problems. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 1–32, 2020.

[42] Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

[43] Sean P Meyn and Richard L Tweedie. *Markov chains and stochastic stability*. Springer Science & Business Media, 2012.

[44] Antonietta Mira. Ordering and improving the performance of monte carlo markov chains. *Statistical Science*, pages 340–350, 2001.

[45] Wenlong Mou, Chris Junchi Li, Martin J Wainwright, Peter L Bartlett, and Michael I Jordan. On linear stochastic approximation: Fine-grained polyak-ruppert and non-asymptotic concentration. In *Conference on Learning Theory*, pages 2947–2997. PMLR, 2020.

[46] Hongseok Namkoong, Aman Sinha, Steve Yadlowsky, and John C Duchi. Adaptive sampling probabilities for non-smooth optimization. In *International Conference on Machine Learning*, pages 2574–2583. PMLR, 2017.

[47] Radford M Neal. Improving asymptotic variance of mcmc estimators: Non-reversible chains are better. Technical report, Department of Statistics, University of Toronto, July 2004.

[48] Arkadij Semenovič Nemirovskij and David Borisovich Yudin. *Problem complexity and method efficiency in optimization*. Wiley-Interscience, 1983.

[49] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32:8026–8037, 2019.

[50] Shashank Rajput, Anant Gupta, and Dimitris Papailiopoulos. Closing the convergence gap of sgd without replacement. In *International Conference on Machine Learning*, pages 7964–7973. PMLR, 2020.

[51] S Sundhar Ram, A Nedić, and Venugopal V Veeravalli. Incremental stochastic subgradient algorithms for convex optimization. *SIAM Journal on Optimization*, 20(2):691–717, 2009.

[52] Amir Hassan Rasti, Mojtaba Torkjazi, Reza Rejaie, Nick Duffield, Walter Willinger, and Daniel Stutzbach. Respondent-driven sampling for characterizing unstructured overlays. In *IEEE INFOCOM 2009*, pages 2701–2705. IEEE, 2009.

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

[54] Gareth O Roberts and Jeffrey S Rosenthal. General state space markov chains and mcmc algorithms. *Probability surveys*, 1:20–71, 2004.

[55] Sheldon M Ross, John J Kelly, Roger J Sullivan, William James Perry, Donald Mercer, Ruth M Davis, Thomas Dell Washburn, Earl V Sager, Joseph B Boyce, and Vincent L Bristow. *Stochastic processes*, volume 2. Wiley New York, 1996.

[56] Ryan A. Rossi and Nesreen K. Ahmed. The network data repository with interactive graph analytics and visualization. In *AAAI*, 2015.

[57] Itay Safran and Ohad Shamir. How good is sgd with random shuffling? In *Conference on Learning Theory*, pages 3250–3284. PMLR, 2020.

[58] Kevin Scaman, Francis Bach, Sébastien Bubeck, Yin Tat Lee, and Laurent Massoulié. Optimal algorithms for smooth and strongly convex distributed optimization in networks. In *International Conference on Machine Learning*, pages 3027–3036. PMLR, 2017.

[59] Ohad Shamir. Without-replacement sampling for stochastic gradient methods. *Advances in neural information processing systems*, 29:46–54, 2016.

[60] Tao Sun, Yuejiao Sun, and Wotao Yin. On markov chain gradient descent. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 9918–9927, 2018.

[61] Fanggang Wang, Xiaojun Yuan, Soung Chang Liew, and Dongning Guo. Wireless mimo switching: Weighted sum mean square error and sum rate optimization. *IEEE transactions on information theory*, 59(9):5297–5312, 2013.

[62] Stefanie Warnat-Herresthal, Hartmut Schultze, Krishnaprasad Lingadahalli Shastry, Sathyanarayanan Manamohan, Saikat Mukherjee, Vishesh Garg, Ravi Sarveswara, Kristian Händler, Peter Pickkers, N Ahmad Aziz, et al. Swarm learning for decentralized and confidential clinical machine learning. *Nature*, 594(7862):265–270, 2021.

[63] Stephan Wojtowytsch. Stochastic gradient descent with noise of machine learning type. part i: Discrete time analysis. *arXiv preprint arXiv:2105.01650*, 2021.

[64] Ran Xin, Shi Pu, Angelia Nedić, and Usman A Khan. A general framework for decentralized optimization with first-order methods. *Proceedings of the IEEE*, 108(11):1869–1889, 2020.

[65] Chulhee Yun, Shashank Rajput, and Suvrit Sra. Minibatch vs local sgd with shuffling: Tight convergence bounds and beyond. *arXiv preprint arXiv:2110.10342, To appear in ICLR 2022*, 2021.

[66] Chulhee Yun, Suvrit Sra, and Ali Jadbabaie. Open problem: Can single-shuffle sgd be better than reshuffling sgd and gd? In *Proceedings of Thirty Fourth Conference on Learning Theory*, volume 134, pages 4653–4658. PMLR, Aug 2021.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section __.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [No]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] All the proofs are included in the supplementary material.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] The details of the simulation setup are provided in the supplementary material.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] The citations can be found in Section 5.
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]