SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

January 2019, Vol. 62 012101:1–012101:13
https://doi.org/10.1007/s11432-018-9656-y

A convergence analysis for a class of practical variance-reduction stochastic gradient MCMC

Changyou CHEN[1*], Wenlin WANG[2], Yizhe ZHANG[3], Qinliang SU[4] & Lawrence CARIN[2]

[1]*Department of Computer Science and Engineering, SUNY at Buffalo, Buffalo* 14260*, USA;*
[2]*Department of Electrical and Computer Engineering, Duke University, Durham* 27708*, USA;*
[3]*Microsoft Research, Redmond* 98052*, USA;*
[4]*School of Data and Computer Science, Sun Yat-sen University, Guangzhou* 510006*, China*

Received 26 June 2018/Revised 23 August 2018/Accepted 26 October 2018/Published online 19 December 2018

Abstract Stochastic gradient Markov chain Monte Carlo (SG-MCMC) has been developed as a flexible family of scalable Bayesian sampling algorithms. However, there has been little theoretical analysis of the impact of minibatch size to the algorithm's convergence rate. In this paper, we prove that at the beginning of an SG-MCMC algorithm, i.e., under limited computational budget/time, a larger minibatch size leads to a faster decrease of the mean squared error bound. The reason for this is due to the prominent noise in small minibatches when calculating stochastic gradients, motivating the necessity of variance reduction in SG-MCMC for practical use. By borrowing ideas from stochastic optimization, we propose a simple and practical variance-reduction technique for SG-MCMC, that is efficient in both computation and storage. More importantly, we develop the theory to prove that our algorithm induces a faster convergence rate than standard SG-MCMC. A number of large-scale experiments, ranging from Bayesian learning of logistic regression to deep neural networks, validate the theory and demonstrate the superiority of the proposed variance-reduction SG-MCMC framework.

Keywords Markov chain Monte Carlo, SG-MCMC, variance reduction, deep neural networks

Citation Chen C Y, Wang W L, Zhang Y Z, et al. A convergence analysis for a class of practical variance-reduction stochastic gradient MCMC. Sci China Inf Sci, 2019, 62(1): 012101, https://doi.org/10.1007/s11432-018-9656-y

1 Introduction

With the increasing size of datasets of interest to machine learning, stochastic gradient Markov Chain Monte Carlo (SG-MCMC) has been established as an effective tool for large-scale Bayesian learning, with applications in topic modeling [1, 2], matrix factorization [3–5], differential privacy [6], Bayesian optimization [7], and deep neural networks [8]. Typically, in each iteration of an SG-MCMC algorithm, a minibatch of data is used to generate the next sample, yielding computational efficiency comparable to stochastic optimization. While a large number of SG-MCMC algorithms have been proposed, their optimal convergence rates generally appear to share the same form, and are typically slower than stochastic gradient descent (SGD) [9]. The impact of stochastic gradient noise comes from a higher-order term (see Lemma 1 below), which was omitted in the analysis of [9]. In other words, current theoretical analysis does not consider the impact of minibatch size (corresponding to stochastic gradient noise), making the underlying convergence theory w.r.t. minibatch size unclear. Recent work by [10] on applying variance

* Corresponding author (email: cchangyou@gmail.com)

reduction in stochastic gradient Langevin dynamics (SGLD) claims to improve the convergence rate of standard SGLD [11–13].

The analysis in [10] omits certain aspects of variance reduction in SGLD, that we seek to address here: (i) how does the minibatch size (or equivalently the stochastic gradient noise) affect the convergence rate of an SG-MCMC algorithm? and (ii) how can one effectively reduce the stochastic gradient noise in SG-MCMC to improve its convergence rate, from both an algorithmic and a theoretical perspective? For (i), we provide theoretical results on the convergence rates of SG-MCMC w.r.t. minibatch size. For (ii), we propose a practical variance-reduction technique for SG-MCMC, as well as the theory to analyze improvements of the corresponding convergence rates. The resulting SG-MCMC algorithm is referred to as variance-reduction SG-MCMC (vrSG-MCMC).

For a clearer description, we first define the notation. In a Bayesian model, our goal is typically to evaluate the posterior average of a test function $\phi(\boldsymbol{x})$, defined as $\bar{\phi} \triangleq \int_{\mathcal{X}} \phi(\boldsymbol{x})\rho(\boldsymbol{x})\mathrm{d}\boldsymbol{x}$, where $\rho(\boldsymbol{x})$ is the target posterior distribution with $\boldsymbol{x}$ the possibly augmented model parameters (next section). Let $\{\boldsymbol{x}_l\}_{l=1}^L$ be the samples generated from an SG-MCMC algorithm. We use the sample average, $\hat{\phi}_L \triangleq \frac{1}{L}\sum_{l=1}^L \phi(\boldsymbol{x}_l)$, to approximate $\bar{\phi}$. The corresponding bias and mean square error (MSE) are defined as $|\mathbb{E}\hat{\phi}_L - \bar{\phi}|$ and $\mathbb{E}(\hat{\phi}_L - \bar{\phi})^2$, respectively. In vrSG-MCMC, unbiased estimates of full gradients are used, leading to the same bias bound as standard SG-MCMC. As a result, we focus here on analyzing the MSE bound for vrSG-MCMC.

We first analyze how minibatch size affects the MSE convergence rate of standard SG-MCMC. Our theoretical finding is motivated by the intuition that samples from a bad burn-in stage (reached by using too noisy gradients at the early stage of MCMC) are not helpful in reducing the MSE bound. Specifically, let N be the training set size. Our theoretical result is summarized in two cases: (i) for a limited computation budget proportional to N[6], which corresponds to the beginning (burn in) stage of SG-MCMC, the optimal MSE bound is achieved when using full gradients[1)]; (ii) for a large enough computational budget, i.e., after burn in, stochastic gradients with minibatches of size one are preferable. This indicates that stochastic gradient noise hurts SG-MCMC at the beginning of the algorithm. While it is computationally infeasible to use full gradients in practice, a remedy to overcome this issue is to use relatively small minibatches with variance reduction techniques to reduce stochastic gradient noise. Consequently, we propose a practical variance-reduction scheme, making SG-MCMC computationally efficient in a big-data setting. Finally, we develop the theory to analyze the benefit of the proposed variance-reduction technique and empirically show improvements of vrSG-MCMC over standard SG-MCMC algorithms.

2 Preliminaries

SG-MCMC is a family of scalable Bayesian sampling algorithms, developed recently to generate approximate samples from a posterior distribution $p(\boldsymbol{\theta}|\boldsymbol{D})$. Here $\boldsymbol{\theta} \in \mathbb{R}^r$ represents a model parameter vector and $\boldsymbol{D} = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_N\}$ represents the data available to learn the model. In general, SG-MCMC algorithms are discretized numerical approximations of continuous-time Itô diffusions [9, 14], which are equipped with stationary distributions coincident with the target posterior distributions. An Itô diffusion is written as $\mathrm{d}\boldsymbol{x}_t = F(\boldsymbol{x}_t)\mathrm{d}t + g(\boldsymbol{x}_t)\mathrm{d}\boldsymbol{w}_t$, where $\boldsymbol{x} \in \mathbb{R}^d$ is the state variable, t is the time index, and $\boldsymbol{w}_t \in \mathbb{R}^d$ is d-dimensional Brownian motion. Typically, $\boldsymbol{x} \supseteq \boldsymbol{\theta}$ is an augmentation of the model parameters, so $r \leqslant d$. Functions $F : \mathbb{R}^d \to \mathbb{R}^d$ and $g : \mathbb{R}^d \to \mathbb{R}^{d\times d}$ are assumed to satisfy the Lipschitz continuity condition [15].

All SG-MCMC algorithms can be formulated by defining appropriate functions F and g [14]. For example, the SGLD model corresponds to $\boldsymbol{x} = \boldsymbol{\theta}$, and $F(\boldsymbol{x}_t) = -\nabla_{\boldsymbol{\theta}} U(\boldsymbol{\theta})$, $g(\boldsymbol{x}_t) = \sqrt{2}\boldsymbol{I}_r$, where $U(\boldsymbol{\theta}) \triangleq -\log p(\boldsymbol{\theta}) - \sum_{i=1}^N \log p(\boldsymbol{d}_i|\boldsymbol{\theta})$ denotes the unnormalized negative log-posterior. Similar formula can be de-

1) Even though it is possible that a full gradient cannot be evaluated within this stage, our theory indicates that samples obtained by using small minibatches at this stage are not good at reducing MSE.

fined for other SG-MCMC algorithms, such as stochastic gradient Hamiltonian Monte Carlo (SGHMC) [3] and stochastic gradient thermostats (SGNHT) [4].

An SG-MCMC algorithm is usually developed by numerically solving the corresponding Itô diffusion and replacing the full gradient $\nabla_{\boldsymbol{\theta}} U(\boldsymbol{\theta})$ with an unbiased estimate from a minibatch of data $\nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta})$ in each iteration. For example, in SGLD, this yields an update equation of $\boldsymbol{\theta}_l = \boldsymbol{\theta}_{l-1} - \nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta}_{l-1}) h_l + \sqrt{2h_l}\zeta_l$ for the l-th iteration, where h_l is the stepsize, $\zeta_l \sim N(\mathbf{0}, \boldsymbol{I}_r)$. This brings two sources of error into the chain: numerical error (from discretization of the differential equation) and stochastic noise error from use of minibatches. In particular, Ref. [9] proved the following bias and MSE bounds for general SG-MCMC algorithms.

Lemma 1 ([9])**.** Under Assumption 1 in the supporting information (SI), the bias and MSE of SG-MCMC with a Kth-order integrator[2)] at time $t = hL$ are bounded as $|\mathbb{E}\hat{\phi}_L - \bar{\phi}| = O(\frac{\sum_l \|\mathbb{E}\Delta V_l\|}{L} + \frac{1}{Lh} + h^K)$, $\mathbb{E}(\hat{\phi}_L - \bar{\phi})^2 = O(\frac{\frac{1}{L}\sum_l \mathbb{E}\|\Delta V_l\|^2}{L} + \frac{1}{Lh} + h^{2K})$.

Here $\Delta V_l \triangleq (\mathcal{L} - \tilde{\mathcal{L}}_l)\phi$, where $\mathcal{L}$ is the infinitesimal generator of the corresponding Itô diffusion defined as $\mathcal{L}f(\boldsymbol{x}_t) = (F(\boldsymbol{x}_t)\cdot\nabla_{\boldsymbol{x}} + \frac{1}{2}(g(\boldsymbol{x}_t)g(\boldsymbol{x}_t)^{\mathrm{T}}) : \nabla_{\boldsymbol{x}}\nabla_{\boldsymbol{x}}^{\mathrm{T}}) f(\boldsymbol{x}_t)$, for any compactly supported twice differentiable function $f : \mathbb{R}^d \to \mathbb{R}$. $\boldsymbol{a}\cdot\boldsymbol{b} \triangleq \boldsymbol{a}^{\mathrm{T}}\boldsymbol{b}$ for two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$, $\boldsymbol{A} : \boldsymbol{B} \triangleq \mathrm{tr}\{\boldsymbol{A}^{\mathrm{T}}\boldsymbol{B}\}$ for two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$. $\|\cdot\|$ is defined as the standard operator norm acting on the space of bounded functions, e.g., $\|f\| \triangleq \sup_{\boldsymbol{x}} f(\boldsymbol{x})$ for a function f. $\tilde{\mathcal{L}}_l$ is the same as $\mathcal{L}$ except for the substitution of the stochastic gradient $\nabla\tilde{U}_l(\boldsymbol{\theta})$ for the full gradient due to the usage of a stochastic gradient in the l-th iteration. By substituting the definition of ΔV_l and $\mathcal{L}$, typically we have $\Delta V_l = (\nabla_{\boldsymbol{\theta}} U_l(\boldsymbol{\theta}) - \nabla_{\boldsymbol{\theta}}\tilde{U}_l(\boldsymbol{\theta}))\cdot\nabla\phi$.

Remark 1. It is probable that the constants factor corresponding to the big-O notation varies under different minibatch sizes. However, given the current theoretical framework, it does not seem possible to derive the exact formula. Consequently, we assume the constants to be the same in order to carry our theoretical results in this paper.

By using an unbiased estimate of the true gradient, the term $\mathbb{E}\Delta V_l$ in the bias bound in Lemma 1 vanishes, indicating that stochastic gradients (or equivalently minibatch size) only affect the MSE bound. Consequently, we focus on improving the MSE bound with the proposed variance-reduction SG-MCMC framework.

3 Practical variance-reduction SG-MCMC

We first motivate the necessity of variance reduction in SG-MCMC, by analyzing how minibatch size affects the MSE bound. A practical variance reduction scheme is then proposed, which is efficient from both computational and storage perspectives. Comparison with existing variance-reduction SG-MCMC approaches is also highlighted.

3.1 The necessity of variance reduction

It is clear from Lemma 1 that the variance of noisy stochastic gradients plays an important role in the MSE bound of an SG-MCMC algorithm. What is unclear is how exactly minibatch size affects the convergence rate. Intuitively, minibatch size appears to play the following roles in SG-MCMC: (i) smaller minibatch sizes introduce larger variance into stochastic gradients; (ii) smaller minibatch sizes allow an algorithm to run faster (thus more samples can be obtained in a given amount of computation time). To balance the two effects, in addition to using the standard assumptions for SG-MCMC (which basically requires the coefficients of Itô diffusions to be smooth and bounded, and is deferred to Assumption 1 in the SI), we assume that the algorithms with different minibatch sizes all run for a fixed computational time/budget T in the analysis, as stated in Assumption 1.

2) The order characterizes the accuracy of a numerical integrator, e.g., the Euler method is a 1st-order integrator. Readers could consider the order as a constant in this paper.

Assumption 1. For a fair comparison, all SG-MCMC algorithms with different minibatch sizes are assumed to run for a fixed amount of computation time/budget T. Further, we assume that T linearly depends on the minibatch size n and the number os MCMC samples L, i.e., $T \propto nL$.

For simplicity, we rewrite the gradient of the log-likelihood for data $\boldsymbol{d}_i$ in the l-th iteration as $\boldsymbol{\alpha}_{li} = \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{d}_i|\boldsymbol{\theta}_l)$. We first derive Lemma 2 about the property of $\{\boldsymbol{\alpha}_{li}\}$, which is useful in the subsequent developments, e.g., to guarantee a positive bound in Theorem 1 and an improved bound for the proposed vrSG-MCMC (Theorem 2).

Lemma 2. Under Assumption 1 in the SI, given $\boldsymbol{\theta}_l$ in the l-th iteration, $\Gamma_l \triangleq \frac{1}{N^2}\sum_{i=1}^N \sum_{j=1}^N \mathbb{E}[\boldsymbol{\alpha}_{li}^{\mathrm{T}}\boldsymbol{\alpha}_{lj}] - \frac{\sum_{i\neq j}\mathbb{E}\boldsymbol{\alpha}_{li}^{\mathrm{T}}\boldsymbol{\alpha}_{lj}}{N(N-1)} \geqslant 0$, where the expectation is taken over the randomness of an SG-MCMC algorithm[3)].

We next generalize Lemma 1 by incorporating the minibatch size n into the MSE bound. The basic idea in our derivation is to associate with each data $\boldsymbol{d}_i$ a binary random variable, z_i, to indicate whether data $\boldsymbol{d}_i$ is included in the current minibatch or not. These $\{z_i\}$ depend on each other such that $\sum_{i=1}^N z_i = n$ in order to guarantee minibatches of size n. Consequently, the stochastic gradient in the l-th iteration can be rewritten as $\nabla_{\boldsymbol{\theta}}\tilde{U}_l(\boldsymbol{\theta}) = -\nabla_{\boldsymbol{\theta}}\log p(\boldsymbol{\theta}_l) - \frac{N}{n}\sum_{i=1}^N \nabla_{\boldsymbol{\theta}}\log p(\boldsymbol{d}_i|\boldsymbol{\theta}_l)z_i$. Substituting the above gradient formula into the proof of standard SG-MCMC [9] and further summing out $\{z_i\}$ results in an alternative MSE bound for SG-MCMC, stated in Theorem 1. In the analysis, we assume to use a 1st-order numerical integrator for simplicity, e.g., the Euler method, though the results generalize to Kth-order integrators easily.

Theorem 1. Under Assumption 1 in the SI, let the minibatch size of an SG-MCMC be n, $\Gamma_M \triangleq \max_l \Gamma_l$. The finite-time MSE is bounded as

$$\mathbb{E}\left(\hat{\phi}_L - \bar{\phi}\right)^2 \leqslant O\left(\frac{2(N-n)N^2\Gamma_M}{nL} + \frac{1}{Lh} + h^2\right).$$

Theorem 1 represents the bound in terms of minibatch size n and sample size L. Note in our finite-time setting, L and N are considered to be constants. Consequently, Γ_M is also a bounded constant in our analysis. To bring in the computational budget T, based on Assumption 1, e.g., $T \propto nL$, the optimal MSE bound w.r.t. stepsize h in Theorem 1 can be written as $\mathbb{E}(\hat{\phi}_L - \bar{\phi})^2 \leqslant O(\frac{(N-n)N^2\Gamma_M}{T} + \frac{n^{2/3}}{T^{2/3}})$. The critic point for the bound w.r.t. the minibatch size n is $n = O(\frac{8T}{27N^6\Gamma_M^3})$. Therefore, depending on different value of T, and based on the fact that n is finite and an integer, we obtained different behaviors of the optimal minibatch sizes that minimizes the optimal bound, concluded in Corollary 1[4)].

Corollary 1. Under Assumptions 1 in both main text and the SI, we have three cases of optimal minibatch sizes, each corresponding to different levels of computational budget.

(1) When the computational budget is small, e.g., $T < O(\frac{27}{8}\Gamma_M^3 N^6)$, the optimal MSE bound is decreasing w.r.t. n in range $[1, N]$. The minimum MSE bound is achieved at $n = N$.

(2) When the computational budget is large, e.g., $T > O(\frac{27}{8}\Gamma_M^3 N^7)$, the optimal MSE bound is increasing w.r.t. n in range $[1, N]$. The minimum MSE bound is achieved at $n = 1$.

(3) When the computational budget is in between the above two cases, the optimal MSE bound first increases then decreases w.r.t. n in range $[1, N]$. The optimal MSE bound is obtained either at $n = 1$ or at $n = N$, depending on (N, T, Γ_M).

In large-scale machine learning applications where N is big and the computational budget is limited, one typically faces with the first case of Corollary 1, i.e., $T < O(\frac{27}{8}\Gamma_M^3 N^6)$. Our results suggest that it might be better to use full gradients instead of minibatches to get an optimal MSE. This is somewhat counter-intuitive but also reasonable because our measurement is in terms of the MSE. The MSE considers a balance between estimation bias and variance. If a minibatch is adopted when N is large enough, one might end up too high a variance, leading to a decreased MSE. In other words, even though SG-MCMC moves fast with a small minibatch, the variance is too high that the samples are not representative of the target distribution at all. Our synthetic experiment in Subsection 5.1 has demonstrated this phenomenon with a simple example. Even though, processing full data (i.e., no minibatch) is typically too

3) The same meaning holds for other expectations in the paper if not explicitly specified.

4) Note we only have that $T = C_1 nL$ for some unknown constant C_1, i.e., the specific value of T is unknown.

computationally expensive when N is large. A practical way to overcome this is to use small minibatches but still able to get low-variance gradient estimations. This motivates variance-reduction techniques to reduce the stochastic gradient noise.

Remark 2. In practice, one always chooses a minibatch size larger than one, which seems not consistent with our result. We conjecture that it might because Assumption 1 does not satisfy in practice. Nevertheless, the goal of Corollary 1 is only to demonstrate the necessary of variance reduction, thus this phenomenon is not our main focus, which is an interesting future work.

3.2 A practical variance reduction algorithm

For practical use, we require that a variance-reduction method should achieve both computational and storage efficiency. While variance reduction has been studied extensively in stochastic optimization, it is applied much less often in SG-MCMC. In this subsection we propose a vrSG-MCMC algorithm, a simple extension of the algorithm in [10], but it is more computationally practical in large-scale applications. A convergence theory is also developed.

The proposed vrSG-MCMC is illustrated in Algorithm 1. Similar to stochastic optimization [16], the idea of variance reduction is to balance the gradient noise with a less-noisy old gradient, i.e., a stochastic gradient is calculated based on a previous sample, as well as using a larger minibatch than that of the current stochastic gradient, resulting in a less noisy estimation. In each iteration of our algorithm, an unbiased stochastic gradient is obtained by combining the above two versions of gradients in an appropriate way (see g_{l+1} in Algorithm 1; see the SI for the unbiasedness proof). Such a construction of stochastic gradients essentially inherits a low variance with theoretical guarantees (detailed in convergence rate section). In Algorithm 1, the whole parameter $\boldsymbol{x}$ is decomposed into the model parameter $\boldsymbol{\theta}$ and the remaining algorithm-specific parameter $\boldsymbol{\tau}$, e.g., the momentum parameter. The expression "$\boldsymbol{\theta} \leftarrow \boldsymbol{x}$" means assigning the corresponding model parameter from $\boldsymbol{x}$ to $\boldsymbol{\theta}$. The old gradient is denoted as $\tilde{g}$, calculated with a minibatch of size n_1. The current stochastic gradient is calculated on a minibatch of size $n_2 < n_1$. We use $\boldsymbol{x}_{l+1} = \text{NextS}(\boldsymbol{x}_l, g_{l+1}, h_l)$ to denote a function which generates next sample $\boldsymbol{x}_{l+1}$ with an SG-MCMC algorithm, based on current sample $\boldsymbol{x}_l$, input stochastic gradient g_{l+1}, and step size h_l.

Algorithm 1 Practical variance-reduction SG-MCMC.

Input: $\bar{\boldsymbol{x}} = \boldsymbol{x}_0 = (\boldsymbol{\theta}_0, \boldsymbol{\tau}_0) \in \mathbb{R}^d$, minibatch sizes (n_1, n_2) such that $n_1 > n_2$, update interval m, total iterations L, stepsize $\{h_l\}_{l=1}^L$.
Output: Approximate samples $\{\boldsymbol{x}_l\}_{l=1}^L$.
for $l = 0$ to $L-1$ **do**
  if $(l \bmod m) = 0$ **then**
    Sample w/t replacement $\{\pi_i\}_{i=1}^{n_1} \subseteq \{1, \dots, N\}$;
    $\bar{\boldsymbol{x}} = \boldsymbol{x}_l$, $\tilde{\boldsymbol{\theta}}_l \leftarrow \bar{\boldsymbol{x}}$;
    $\tilde{g} = \frac{N}{n_1} \sum_{i \in \pi} \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{d}_i | \tilde{\boldsymbol{\theta}}_l)$;
  end if
  $\boldsymbol{\theta}_l \leftarrow \boldsymbol{x}_l$, $\tilde{\boldsymbol{\theta}}_l \leftarrow \bar{\boldsymbol{x}}$, {Fix $\tilde{\boldsymbol{\theta}}_l$ in the outer loop as a control variate};
  Sample w/t replacement $\{\tilde{\pi}_i\}_{i=1}^{n_2} \subseteq \{1, \dots, N\}$;
  $g_{l+1} = \tilde{g} + \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta}_l) + \frac{N}{n_2} \sum_{i \in \tilde{\pi}} (\nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{d}_i | \boldsymbol{\theta}_l) - \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{d}_i | \tilde{\boldsymbol{\theta}}_l))$;
  $\boldsymbol{x}_{l+1} = \text{NextS}(\boldsymbol{x}_l, g_{l+1}, h_{l+1})$;
end for

One should note that existing variance-reduction algorithms, e.g., [17], use a similar concept to construct low-variance gradients. However, most algorithms use the whole training data to compute $\tilde{g}$ in Algorithm 1, which is computationally infeasible in large-scale settings. Moreover, we note that like in stochastic optimization [18,19], instead of using a single parameter sample to compute $\tilde{g}$, similar methods can be adopted to compute $\tilde{g}$ based on an average of old parameter samples. The theoretical analysis can be readily adopted for such cases, which is omitted here for simplicity. More references are discussed in related work.

3.3 Comparison with existing variance-reduction SG-MCMC algorithms

The most related variance-reduction SG-MCMC algorithm we are aware of is a recent work on variance-reduction SGLD (SVRG-LD) [10]. SVRG-LD shares a similar flavor to our scheme from the algorithmic perspective, except that when calculating the old gradient $\tilde{g}$, the whole training data set is used in SVRG-LD. As mentioned above, this brings a computational challenge for large-scale learning. Although the problem is mitigated by using a moving average estimation of the stochastic gradient, this scheme does not match their theory. A more distinctive advantage of vrSG-MCMC over SVRG-LD [10] is in terms of theoretical analysis. Concerning SVRG-LD, (i) the authors did not show theoretically in which case variance reduction is useful in SGLD, and (ii) it is not clear in their theory whether SVRG-LD is able to speed up the convergence rate compared to standard SGLD. Specifically, the MSE of SVRG-LD was shown to be bounded by $O(\frac{N^2\min\{2\sigma^2,m^2(D^2h^2\sigma^2+hd)\}}{nL}+\frac{1}{Lh}+h^2)$, compared to $O(\frac{N^2\sigma^2}{nL}+\frac{1}{Lh}+h^2)$ for SGLD, where (d,D,σ) are constants, m is the same as defined in Algorithm 1. By inspecting the above bounds, it is not clear whether SVRG-LD improves SGLD because the two bounds are not directly comparable[5)]. More detailed explanations are provided in the SI.

3.4 Convergence rate

We derive convergence bounds for Algorithm 1 and analyze the improvement of vrSG-MCMC over the corresponding standard SG-MCMC. Using a similar approach as in Theorem 1, we first introduce additional binary random variables, $\{b_i\}_{i=1}^N$, to indicate which data points are included in calculating the old gradient $\tilde{g}$ in Algorithm 1. This results in the expression for the stochastic gradient used in the l-th iteration: $\nabla_{\boldsymbol{\theta}}\tilde{U}(\boldsymbol{\theta}_l)=\frac{N}{n_2}\sum_{i=1}^N(\nabla_{\boldsymbol{\theta}}\log p(\boldsymbol{d}_i|\boldsymbol{\theta}_l)-\nabla_{\boldsymbol{\theta}}\log p(\boldsymbol{d}_i|\tilde{\boldsymbol{\theta}}_l))z_i+\frac{N}{n_1}\sum_{i=1}^N\sum_{i=1}^N\nabla_{\boldsymbol{\theta}}\log p(\boldsymbol{d}_i|\tilde{\boldsymbol{\theta}}_l)b_i$. It is easy to verify that the above stochastic gradient is an unbiased estimation of the true gradient in the l-th iteration (see the SI for a formal proof).

In order to see how Algorithm 1 reduces the variance of stochastic gradients, from Lemma 1, it suffices to study ΔV_l, as the minibatch size only impacts this term. For notational simplicity, similar to the $\boldsymbol{\alpha}_{li}$ defined above, we denote $\boldsymbol{\beta}_{li}\triangleq\nabla_{\boldsymbol{\theta}}\log p(\boldsymbol{d}_i|\tilde{\boldsymbol{\theta}}_l)$, which is similar to $\boldsymbol{\alpha}_{li}$ but evaluated on the old parameter $\tilde{\boldsymbol{\theta}}_l$. Intuitively, since the old gradient $\tilde{g}$ is calculated from $\boldsymbol{\beta}$ to balance the stochastic gradient noise (calculated from $\boldsymbol{\alpha}$), $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are expected to be close to each other. Lemma 3 formulates the intuition, a key result in proving our main theorem, where we only consider the update interval m and stepsize h as factors. In Lemma 3, following [20] (Assumption 1), we further assume the gradient function $\nabla_{\boldsymbol{\theta}}U(\boldsymbol{\theta})$ to be Lipschitz.

Lemma 3. Under Assumption 1 in the SI and assume $\nabla_{\boldsymbol{\theta}}U(\boldsymbol{\theta})$ to be Lipschitz (Assumption 1 in [20]), $\boldsymbol{\alpha}_{li}$ and $\boldsymbol{\beta}_{li}$ are close to each other in expectation, i.e., $\mathbb{E}\boldsymbol{\alpha}_{li}=\mathbb{E}\boldsymbol{\beta}_{li}+O(mh)$.

In the SI, we further simplify $\mathbb{E}\|\Delta V_l\|^2$ in the MSE bound by decomposing it into several terms. Finally, we arrive at our main theorem for the proposed vrSG-MCMC framework.

Theorem 2. Under the setting of Lemma 3, let $A_M\triangleq\max_l A_l$, and $A_l=(\frac{N}{n_2}-1)\sum_{ij}\mathbb{E}\boldsymbol{\alpha}_{li}^{\mathrm{T}}\boldsymbol{\alpha}_{lj}-2\frac{N(N-n_2)}{n_2(N-1)}\sum_{i<j}\mathbb{E}\boldsymbol{\alpha}_{li}^{\mathrm{T}}\boldsymbol{\alpha}_{lj}$. The MSE of vrSG-MCMC with a Kth-order integrator is bounded as $\mathbb{E}(\hat{\phi}_L-\bar{\phi})^2=O(\frac{A_M}{L}+\frac{1}{Lh}+h^{2K}+\frac{mh}{L}-\frac{\lambda_M}{L})$, where we have defined $\lambda_M=\min_l\lambda_l$, and $\lambda_l\triangleq(\frac{N}{n_1}-\frac{N}{n_2})\sum_{ij}\mathbb{E}\boldsymbol{\beta}_{li}^{\mathrm{T}}\boldsymbol{\beta}_{li}-2\big(\frac{N(N-n_2)}{n_2(N-1)}-\frac{N(N-n_1)}{n_1(N-1)}\big)\times\sum_{i<j}\mathbb{E}\boldsymbol{\beta}_{li}^{\mathrm{T}}\boldsymbol{\beta}_{lj}$. Furthermore, we have $\lambda_l>0$ for $\forall l$, so that $\lambda_M>0$.

Note that for a fixed m, $\frac{mh}{L}$ in the above bound is a high-order term relative to $\frac{1}{Lh}$, thus is removed in the big-O notation. As a result, the MSE is bounded by $O(\frac{A_M}{L}+\frac{1}{Lh}+h^{2K}-\frac{\lambda_M}{L})$. Because the MSE of standard SG-MCMC is bounded by $O(\frac{A_M}{L}+\frac{1}{Lh}+h^{2K})$ (detailed in the SI) and $\lambda_M>0$ from Theorem 2, we conclude that vrSG-MCMC induces a lower MSE bound compared to the corresponding SG-MCMC algorithm[6)], with an improvement of $O(\frac{\lambda_M}{L})$.

It is worth noting that in Algorithm 1, the minibatch for calculating the old gradient $\tilde{g}$ is required to be larger than that for calculating the current stochastic gradient, i.e., $n_1>n_2$. Otherwise, λ_l in Theorem 2

5) The first term in the min of the SVRG-LD bound is strictly larger than the first term of the SGLD bound (if the term $2\sigma^2$ is used in the "min"), making the bounds not easily compared.

6) This is based on Remark 1 that both vrSG-MCMC and SG-MCMC share the same constant.

would become negative, leading to an increased MSE bound compared to standard SG-MCMC. This matches the intuition that old gradients need to be more accurate (thus with larger minibatches) than current stochastic gradients in order to reduce the stochastic gradient noise.

Remark 3. In the special case of [10] where $n_1 = N$ for SGLD, Theorem 2 gives an MSE bound of $O(\frac{A_M}{L} + \frac{1}{Lh} + h^2 + \frac{mh}{L} - \frac{\min_l \lambda_l}{L})$, with $\lambda_l = (\frac{N}{n_2} - 1)\sum_{ij} \mathbb{E}\boldsymbol{\beta}_{li}^{\mathrm{T}}\boldsymbol{\beta}_{li} - \frac{2N(N-n_2)}{n_2(N-1)}\sum_{i<j} \mathbb{E}\boldsymbol{\beta}_{li}^{\mathrm{T}}\boldsymbol{\beta}_{lj}$. According to Lemma 2, λ_l is also positive, thus leading to a reduced MSE bound. However, the bound is not necessarily better than that of vrSG-MCMC, where a minibatch is used instead of the whole data set to calculate $\tilde{g}$, leading to a significant decrease of computational time.

Remark 4. Following Corollary 1, Theorem 2 can also be formulated in terms of the computational budget T. Specifically, according to Algorithm 1, the computational budget T would be proportional to $\frac{n_1}{m} + n_2$. Substituting this into the MSE bound of Theorem 2 gives a reformulated bound of $O(\frac{(A_M + mh - \lambda_M)(\frac{n_1}{m} + n_2)}{T} + \frac{1}{Lh} + h^2)$. The optimal MSE w.r.t. n_1 and n_2 would be complicated since both A_M and λ_M depend on n_1 and n_2, an interesting problem for future research. Nevertheless, our experiments indicate that our algorithm always improves standard SG-MCMC algorithms for the same computational time.

4 Related work

Variance reduction was first introduced in stochastic optimization, which quickly became a popular research topic and has been actively developed in recent years. Refs. [16, 21] introduced perhaps the first variance reduction algorithm, called stochastic average gradient (SAG), where historical gradients are stored and continuously updated in each iteration. Later, stochastic variance reduction gradient (SVRG) was developed to reduce the storage bottleneck of SAG, at the cost of an increased computational time [17, 22]. Ref. [23] combined ideas of SAG and SVRG and proposed the SAGA algorithm, which improves SAG by using a better and unbiased stochastic-gradient estimation.

Variance reduction algorithms were first designed for convex optimization problems, followed by a number of recent studies extending the techniques for non-convex optimization [18, 19, 24, 25], as well as for distributed learning [26]. All these algorithms are mostly based on SVRG and are similar in algorithmic form, but differ in the techniques for proving the rigorous theoretical results.

Variance reduction has also been used in some minibatch-based MCMC algorithms, for example, Refs. [27, 28] use minibatches to run approximate Metropolis-Hastings/Gibbs sampling algorithms. For scalable Bayesian sampling with SG-MCMC, however, this topic has been studied little until a recent work on variance reduction for SGLD [10]. In this work, the authors adapted the SAG and SVRG ideas to SGLD. Although they provided corresponding convergence results, some fundamental problems, such as how minibatch size affects the convergence rate, were not fully studied. Furthermore, their algorithms suffer from an either high computational or storage cost in a big-data setting, because the whole data set needs to be accessed frequently. It is also worthwhile to mention that there are very recent studies such as [29–31] trying to investigate variance reduction for SG-MCMC from either algorithmic or theoretical perspectives. Their goals and techniques are different from ours in that they investigate the convergence of Wasserstein distance between the distribution induced by and algorithm and the true distribution. Furthermore, Ref. [29] only studied the case of log-concave distributions.

To reduce the computational cost of SVRG-based algorithms, the idea of using a minibatch of data to calculate the old gradient (corresponding to the $\tilde{g}$ in Algorithm 1) has also been studied in stochastic optimization. Representative studies include, but are not limited to [32–36]. The proposed approach adopts similar ideas, with the following main differences: (i) our algorithm represents the first work for large-scalable Bayesian sampling with a practical (computationally cheap) variance reduction technique; (ii) the techniques used here for analysis are different and appear to be simpler than those used for stochastic optimization; (iii) our theory addresses fundamental questions for variance reduction in SG-MCMC, such as those raised in the Introduction.

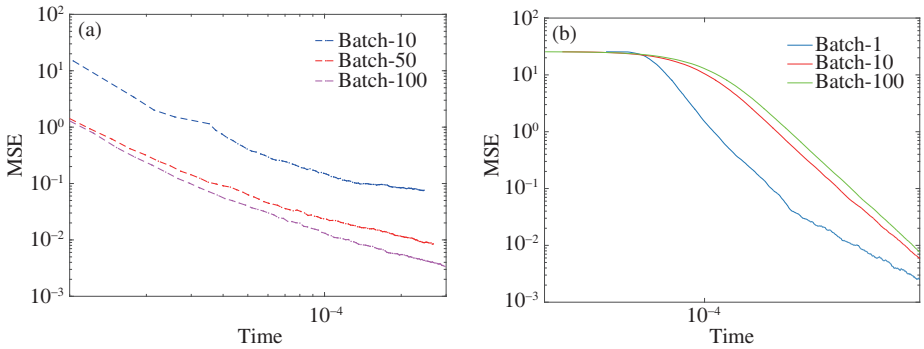

Figure 1 (Color online) MSE vs. wall-clock time for different computational budgets (running time) with varying minibatch sizes for standard SG-MCMC. For very limited computational resources, larger minibatch sizes tend to achieve better MSE (a); while for a larger computational budget, a minibatch size of 1 obtains the smallest MSE (b).

5 Experiments

5.1 A synthetic experiment

We first verify the conclusion of Corollary 1 when given both very limited and enough running time. Under a very limited-time setting, Corollary 1 indicates larger minibatch sizes should achieve lower MSE bound (which is expected to happen at the beginning); while when given enough computational budget (running the algorithm long enough, and can be regarded as after the burn-in stage), a minibatches of size 1 should achieve the optimal MSE bound. To achieve this, we test SG-MCMC on a simple Gaussian model, which runs very fast so that a little actual walk-clock time is considered long enough to get burn-in. The model is defined as $x_i \sim \mathcal{N}(\theta, 1)$, $\theta \sim \mathcal{N}(0, 1)$. We generate $N = 1000$ data samples $\{x_i\}$, and calculate the MSEs for different minibatch sizes. The test function is $\phi(\theta) = \theta^2$. The results are plotted in Figure 1, which in some sense is consistent with the theory (Corollary 1).

5.2 Real applications

We apply the proposed vrSG-MCMC framework to Bayesian learning of logistic regression and deep neural networks, including the multilayer perceptron (MLP), convolutional neural network (CNN), and recurrent neural network (RNN). The latter two have not been empirically evaluated in the variance-reduction setting in previous work. In the experiments, we are interested in modeling weight uncertainty of neural networks, which is an important topic and has been well studied [8,37–39]. We achieve this goal by applying priors to the weights (in our case, we use simple isotropic Gaussian priors) and performing posterior sampling with vrSG-MCMC or SG-MCMC. We implement vrSG-MCMC based on SGLD, and compare it to the standard SGLD and SVRG-LD [10] in our experiments[7]. For this reason, comparisons to other optimization-based methods such as the maximum likelihood are not considered. For simplicity, we set the update interval for the old gradient $\tilde{g}$ in Algorithm 1 to $m = 10$. For all the experiments, the minibatch sizes for vrSG-MCMC are set to $n_1 = 100$ and $n_2 = 10$. To be fair, this corresponds to a minibatch size of $n = 10$ in SGLD and SVRG-LD. Sensitivity of model performance w.r.t. minibatch size n_1 is also tested. For a fair comparison, following convention [10, 19], we plot the number of data passes versus error in the figures[8]. Note the number of data passes includes all the data accessed by an algorithm, e.g., the data used to compute both current gradients and history gradients $\tilde{g}$. Based on our assumption, this is also roughly proportional to the actual wall-clock time. Results on the number of data passes versus loss are given in the SI. In addition, we use fixed stepsizes in our algorithm for all except for the ResNet model specified below. Following relevant literatures [10,17], we tune the stepsizes and plot the best results for all the algorithms to ensure fairness. Note in our Bayesian setup, it is enough

7) The SAGA-LD algorithm in [10] is not compared here because it is too storage-expensive thus is not fair.
8) Since true posterior averages are infeasible, we plot sample averages in terms of accuracy/loss.

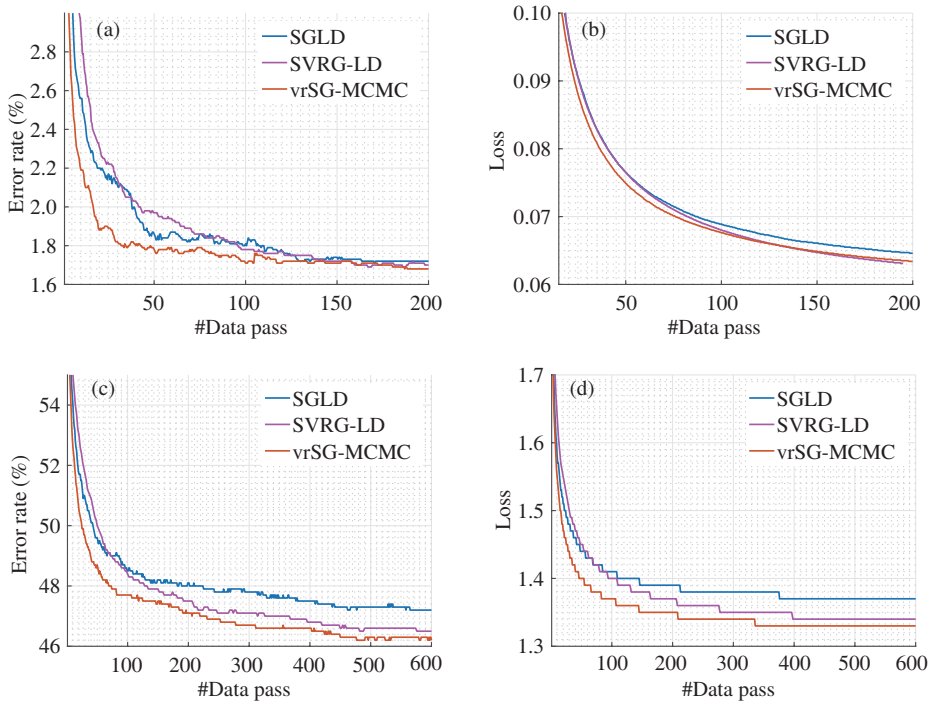

Figure 2 (Color online) Number of passes through data vs. testing error ((a) and (c)) / loss ((b) and (d)) on MNIST ((a) and (b)) and CIFAR-10 ((c) and (d)).

to run an algorithm for once since the uncertainty is encoded in the samples. Furthermore, we do not compared our method with optimization-based methods, which is beyond our goal.

Multilayer perceptron. We follow conventional settings [18,19] and use a single-layer MLP with 100 hidden units, using the sigmoid activation function as the nonlinear transformation. We test the MLP on the MNIST and CIFAR-10 datasets. The stepsizes for both vrSG-MCMC and SGLD are set to 0.25 and 0.01 in the two datasets, respectively. Figure 2 plots the number of passes through the data versus test error/loss. Results on the training datasets, including training results for the CNN and RNN-based models described below, are provided in the SI. It is clear that vrSG-MCMC leads to a much faster convergence speed than SGLD, resulting in much lower test errors and loss at the end, especially on the CIFAR-10 dataset. SVRG-LD, though it leads to potential lower errors/loss, converges slower than vrSG-MCMC, due to the high computational cost in calculating the old gradient $\tilde{g}$. As a result, we do not compare vrSG-MCMC with SVRG-LD in the remaining experiments[9)].

Convolutional neural networks. We use the CIFAR-10 dataset, and test two CNN architectures for image classification. The first architecture is a deep convolutional neural networks with 4 convolutional layers, denoted as C32-C32-C64-C32, where max-pooling is applied on the output of the first three convolutional layers, and a Dropout layer is applied on the output of the last convolutional layer. The second architecture is a 20-layers deep residual network (ResNet) with the same setup as in [40]. Specifically, we use a step-size-decrease scheme as $h_l = \frac{1}{10+1.8\text{e-}3\times l}$ for both vrSG-MCMC and SGLD, where l is the number of iterations so far.

Figure 3 plots the number of passes through the data versus test error/loss on both models. Similar to the results on MLP, vrSG-MCMC converges much faster than SGLD, leading to lower test errors and loss. Interestingly, the gap seems larger in the more complicated ResNet architecture; furthermore, the learning curves look much less noisy (smoother) for vrSG-MCMC because of the reduced variance in stochastic gradients.

Recurrent neural networks. The recurrent neural network with LSTM units [41] is a powerful

9) Actually, we found that in some experiments with complex deep-neural-network architectures and big data, SVRG-LD even converges slower than standard SGLD due to the need to evaluate full gradients frequently. Thus we did not incorporate the results for SVRG-LD in the following experiments.

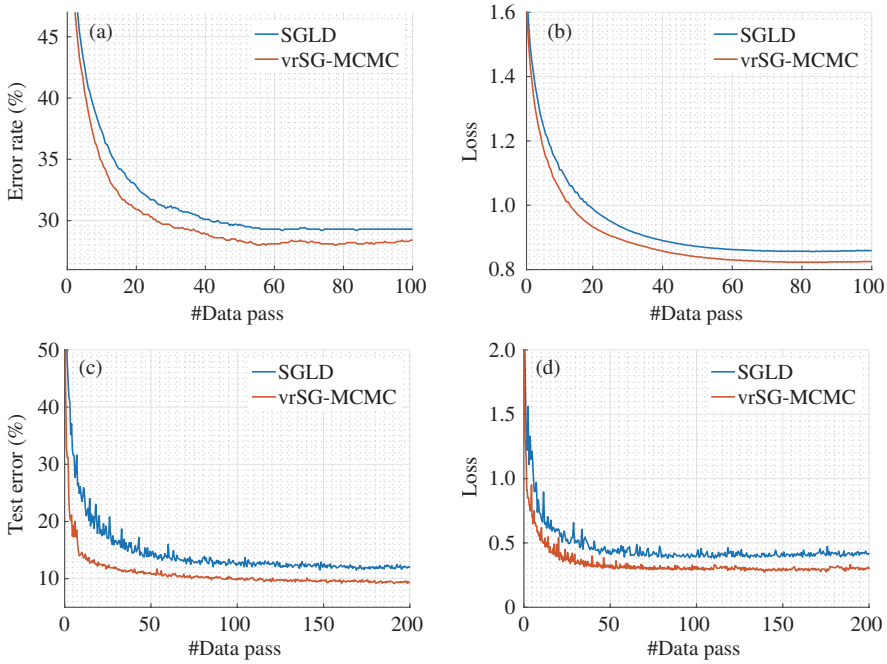

Figure 3 (Color online) Number of passes through data vs. testing error ((a) and (c)) / loss ((b) and (d)) with CNN-4 ((a) and (b)) and ResNet ((c) annd (d)) on CIFAR-10.

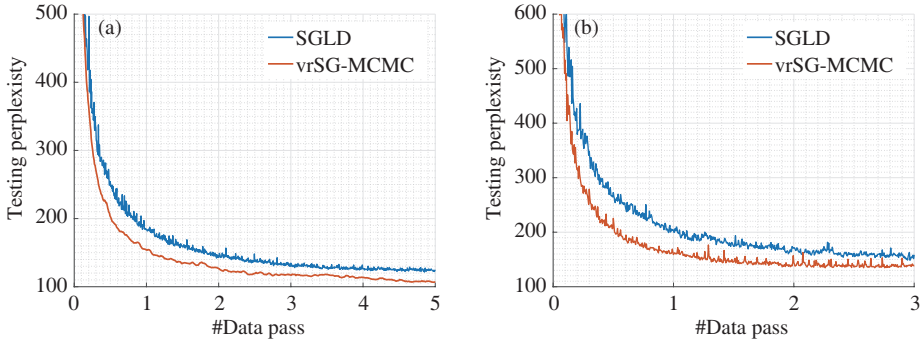

Figure 4 (Color online) Number of passes through data vs. testing perplexity on the PTB dataset (a) and WikiTest-2 dataset (b).

architecture used for modeling sequence-to-sequence data. We consider the task of language modeling on two datasets, i.e., the Penn treebank (PTB) dataset and WikiText-2 dataset [42]. PTB is the smaller dataset among the two, containing a vocabulary of size 10000. We use the default setup of 887521 tokens for training, 70390 for validation and 78669 for testing. WikiTest-2 is a large dataset with 2088628 tokens from 600 Wiki articles for training, 217649 tokens from 60 Wiki articles for validation, and 245569 tokens from an additional 60 Wiki articles for testing. The total vocabulary size is 33278.

We adopt the deep LSTM achitecture [43]. The hierarchy depth is set to 2, with each LSTM containing 200 hidden unites. The step size is set to 0.5 for both datasets. For more stable training, standard gradient clipping is adopted, where gradients are clipped if the norm of the parameter vector exceeds 5. Figure 4 plots the number of passes through the data versus test perplexity on both datasets. The results are consistent with the previous experiments on MLPs and CNNs, where vrSG-MCMC achieves faster convergence than SGLD; its learning curves in terms of testing error/loss are also much smoother.

Parameter sensitivity. Note that one of the main differences between vrSG-MCMC and the recently proposed SVRG-LD [10] is that the former uses minibatches of size n_1 to calculate the old gradient $\tilde{g}$ in Algorithm 1, leading to a much more computationally efficient algorithm, with theoretical guarantees. This section tests the sensitivity of model performance to the parameter n_1.

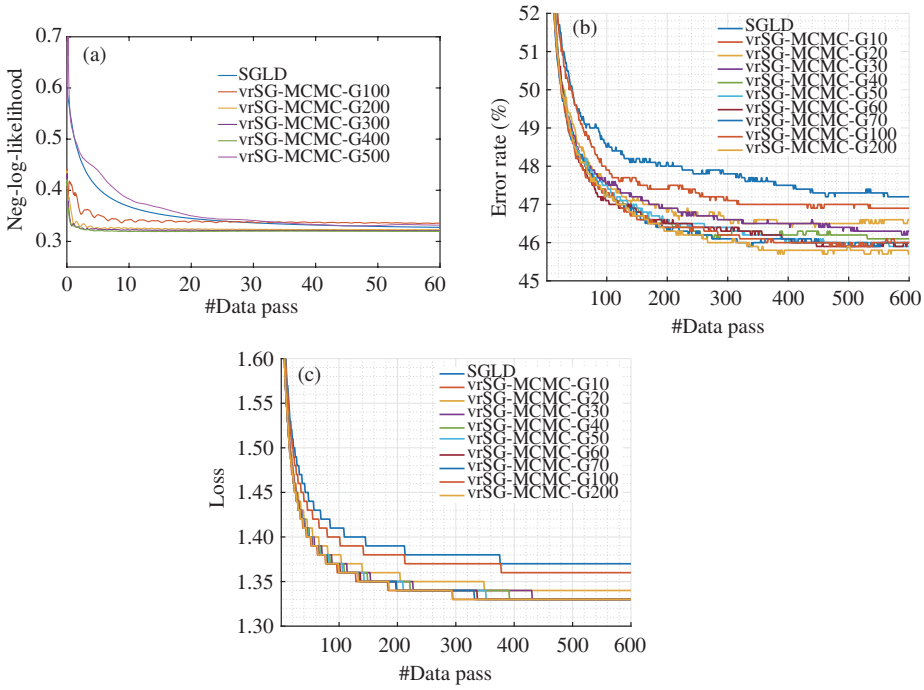

Figure 5 (Color online) Number of passes through data vs. testing negative log-likelihood on the Pima dataset for Bayesian logistic regression (a); number of passes through data vs. testing errors (b) / loss (c) on the CIFAR-10 dataset, with varying n_1 values.

For simplicity, we run on a Bayesian logistic regression model on the Pima dataset with the same setting as in [44], as well as the same MLP model described above on the CIFAR-10 dataset, where the same parameter settings are used, but varying n_1 in $\{100, 200, 300, 400, 500, 600, 700, 1000, 2000\}$. Figure 5 plots the number of passes through data versus test errors/loss, where we use "vrSG-MCMC-Gn" to denote vrSG-MCMC with $n_1 = n$. Interestingly, vrSG-MCMC outperforms the baseline SGLD on all n_1 values. Notably, when n_1 is large enough ($n_1 = 200$ in our case), their corresponding test errors and loss are very close. This agrees with the intuition that computing the old gradient using the whole training data is not necessarily a good choice in order to balance the stochastic gradient noise and computational time.

6 Conclusion

We investigate the impact of minibatches in SG-MCMC and propose a simple and practical variance-reduction SG-MCMC algorithm to reduce the stochastic gradient noise in SG-MCMC, based on existing work [10]. Compared to existing variance-reduction techniques for SG-MCMC, the proposed method is efficient from both the computational and storage perspectives. Theory is developed to guarantee faster convergence rates of vrSG-MCMC compared to standard SG-MCMC algorithms. Extensive experiments on Bayesian learning of logistic regression and DNNs verify the theory, obtaining significant speedup compared to the corresponding vanilla SG-MCMC algorithms.

Supporting information Appendixes A–F. The supporting information is available online at info.scichina.com and link.springer.com. The supporting materials are published as submitted, without typesetting or editing. The responsibility for scientific accuracy and content remains entirely with the authors.

References

1 Gan Z, Chen C Y, Henao R, et al. Scalable deep Poisson factor analysis for topic modeling. In: Proceedings of International Conference on Machine Learning, 2015

2 Liu C, Zhu J, Song Y. Stochastic gradient geodesic MCMC methods. In: Proceedings of Conference on Neural Information Processing System, 2016
3 Chen T, Fox E B, Guestrin C. Stochastic gradient Hamiltonian Monte Carlo. In: Proceedings of International Conference on Machine Learning, 2014
4 Ding N, Fang Y H, Babbush R, et al. Bayesian sampling using stochastic gradient thermostats. In: Proceedings of Conference on Neural Information Processing System, 2014
5 Şimşekli U, Badeau R, Cemgil A T, et al. Stochastic quasi-Newton Langevin Monte Carlo. In: Proceedings of International Conference on Machine Learning, 2016
6 Wang Y X, Fienberg S E, Smola A. Privacy for free: posterior sampling and stochastic gradient Monte Carlo. In: Proceedings of International Conference on Machine Learning, 2015
7 Springenberg J T, Klein A, Falkner S, et al. Bayesian optimization with robust Bayesian neural networks. In: Proceedings of Conference on Neural Information Processing System, 2016
8 Li C Y, Chen C Y, Carlson D, et al. Preconditioned stochastic gradient Langevin dynamics for deep neural networks. In: Proceedings of AAAI Conference on Artificial Intelligence, 2016
9 Chen C Y, Ding N, Carin L. On the convergence of stochastic gradient MCMC algorithms with high-order integrators. In: Proceedings of Conference on Neural Information Processing System, 2015
10 Dubey A, Reddi S J, Póczos B, et al. Variance reduction in stochastic gradient Langevin dynamics. In: Proceedings of Conference on Neural Information Processing System, 2016
11 Welling M, Teh Y W. Bayesian learning via stochastic gradient Langevin dynamics. In: Proceedings of International Conference on Machine Learning, 2011
12 Teh Y W, Thiery A H, Vollmer S J. Consistency and fluctuations for stochastic gradient Langevin dynamics. J Mach Learn Res, 2016, 17: 193–225
13 Vollmer S J, Zygalakis K C, Teh Y W. Exploration of the (Non-)asymptotic bias and variance of stochastic gradient Langevin dynamics. J Mach Learn Res, 2016, 17: 5504–5548
14 Ma Y A, Chen T Q, Fox E B. A complete recipe for stochastic gradient MCMC. In: Proceedings of International Conference on Machine Learning, 2015
15 Ghosh A P. Backward and forward equations for diffusion processes. Wiley Encyclopedia of Operations Research and Management Science, 2011. doi: 10.1002/9780470400531.eorms0080
16 Schmidt M, Le Roux N, Bach F. Minimizing finite sums with the stochastic average gradient. Math Program, 2017, 162: 83–112
17 Johnson R, Zhang T. Accelerating stochastic gradient descent using predictive variance reduction. In: Proceedings of Conference on Neural Information Processing System, 2013
18 Reddi S J, Hefny A, Sra S, et al. Stochastic variance reduction for nonconvex optimization. In: Proceedings of International Conference on Machine Learning, 2016
19 Allen-Zhu Z, Hazan E. Variance reduction for faster non-convex optimization. In: Proceedings of International Conference on Machine Learning, 2016
20 Chen C Y, Ding N, Li C Y, et al. Stochastic gradient MCMC with stale gradients. In: Proceedings of Conference on Neural Information Processing System, 2016
21 Schmidt M, Roux N L, Bach F. Minimizing finite sums with the stochastic average gradient. 2013. ArXiv:1309.2388
22 Zhang L J, Mahdavi M, Jin R. Linear convergence with condition number independent access of full gradients. In: Proceedings of Conference on Neural Information Processing System, 2013
23 Defazio A, Bach F, Lacoste-Julien S. SAGA: a fast incremental gradient method with support for non-strongly convex composite objectives. In: Proceedings of Conference on Neural Information Processing System, 2014
24 Reddi S J, Sra S, Póczos B. Fast stochastic methods for nonsmooth nonconvex optimization. In: Proceedings of Conference on Neural Information Processing System, 2016
25 Allen-Zhu Z, Richtárik P, Qu Z, et al. Even faster accelerated coordinate descent using non-uniform sampling. In: Proceedings of International Conference on Machine Learning, 2016
26 Reddi S J, Hefny A, Sra S, et al. On variance reduction in stochastic gradient descent and its asynchronous variants. In: Proceedings of Conference on Neural Information Processing System, 2015
27 Chen Y T, Ghahramani Z. Scalable discrete sampling as a multi-armed bandit problem. In: Proceedings of International Conference on Machine Learning, 2016
28 Bardenet R, Doucet A, Holmes C. On Markov chain Monte Carlo methods for tall data. J Mach Learn Res, 2017, 18: 1–43
29 Baker J, Fearnhead P, Fox E B, et al. Control variates for stochastic gradient MCMC. 2017. ArXiv:1706.05439
30 Chatterji N S, Flammarion N, Ma Y A, et al. On the theory of variance reduction for stochastic gradient Monte Carlo. In: Proceedings of International Conference on Machine Learning, 2018
31 Zou D F, Xu P, Gu Q Q. Subsampled stochastic variance-reduced gradient Langevin dynamics. In: Proceedings of Conference on Uncertainty in Artificial Intelligence, 2018
32 Harikandeh R, Ahmed M O, Virani A, et al. Stop wasting my gradients: practical SVRG. In: Proceedings of Conference on Neural Information Processing System, 2015
33 Frostig R, Ge R, Kakade S M, et al. Competing with the empirical risk minimizer in a single pass. In: Proceedings of Conference on Learning Theory, 2015
34 Shah V, Asteris M, Kyrillidis A, et al. Trading-off variance and complexity in stochastic gradient descent. 2016. ArXiv:1603.06861

35 Lei L H, Jordan M I. Less than a single pass: stochastically controlled stochastic gradient method. In: Proceedings of Conference on Neural Information Processing System, 2016
36 Lian X R, Wang M D, J Liu. Finite-sum composition optimization via variance reduced gradient descent. In: Proceedings of International Conference on Artificial Intelligence and Statistics, 2017
37 Hernández-Lobato J M, Adams R P. Probabilistic backpropagation for scalable learning of Bayesian neural networks. In: Proceedings of International Conference on Machine Learning, 2015
38 Blundell C, Cornebise J, Kavukcuoglu K, et al. Weight uncertainty in neural networks. In: Proceedings of International Conference on Machine Learning, 2015
39 Louizos C, Welling M. Structured and efficient variational deep learning with matrix Gaussian posteriors. In: Proceedings of International Conference on Machine Learning, 2016
40 He K M, Zhang X Y, Ren S Q, et al. Deep residual learning for image recognition. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2016
41 Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput, 1997, 9: 1735–1780
42 Merity S, Xiong C M, Bradbury J, et al. Pointer sentinel mixture models. 2016. ArXiv:1609.07843
43 Zaremba W, Sutskever I, Vinyals O. Recurrent neural network regularization. 2014. ArXiv:1409.2329
44 Zhang Y Z, Chen C Y, Gan Z, et al. Stochastic gradient monomial Gamma sampler. In: Proceedings of International Conference on Machine Learning, 2017