Journal of the American Statistical Association

ISSN: 0162-1459 (Print) 1537-274X (Online) Journal homepage: https://www.tandfonline.com/loi/uasa20

Mini-batch Metropolis-Hastings with Reversible SGLD Proposal

Tung-Yu Wu, Y. X. Rachel Wang & Wing H. Wong

To cite this article: Tung-Yu Wu, Y. X. Rachel Wang & Wing H. Wong (2020): Mini-batch Metropolis-Hastings with Reversible SGLD Proposal, Journal of the American Statistical Association, DOI: 10.1080/01621459.2020.1782222

To link to this article: https://doi.org/10.1080/01621459.2020.1782222

View supplementary material

Accepted author version posted online: 24 Jun 2020.

Submit your article to this journal

Article views: 103

View related articles

View Crossmark data

Full Terms & Conditions of access and use can be found at
https://www.tandfonline.com/action/journalInformation?journalCode=uasa20

Check for updates

Mini-batch Metropolis-Hastings with Reversible SGLD Proposal

Tung-Yu Wu*

Institute for Computational & Mathematical Engineering, Stanford University

Y. X. Rachel Wang *‡

School of Mathematics and Statistics, University of Sydney

and

Wing H. Wong †§

Department of Statistics, Stanford University

*Equal contribution

†Co-corresponding author Wing H. Wong whwong@stanford.edu

‡Partially supported by the ARC DECRA Fellowship

§Partially supported by NSF grants DMS 1721550 and DMS 1811920

Abstract

Traditional MCMC algorithms are computationally intensive and do not scale well to large data. In particular, the Metropolis-Hastings (MH) algorithm requires passing over the entire dataset to evaluate the likelihood ratio in each iteration. We propose a general framework for performing MH-MCMC using mini-batches of the whole dataset and show that this gives rise to approximately a tempered stationary distribution. We prove that the algorithm preserves the modes of the original target distribution and derive an error bound on the approximation with mild assumptions on the likelihood. To further extend the utility of the algorithm to high dimensional settings, we construct a proposal with forward and reverse moves using stochastic gradient and show that the construction leads to reasonable acceptance probabilities. We demonstrate the performance of our algorithm in both low dimensional models and high dimensional neural network applications. Particularly in the latter case, compared to popular optimization methods, our method is more robust to the choice of learning rate and improves testing accuracy.

Keywords: scalable MCMC, tempering, neural networks

Accepted Manuscript

1 Introduction

Since its inception, Markov chain Monte Carlo (MCMC) sampling has been an indispensable tool in Bayesian modeling for obtaining parameter estimates and their uncertainty. However, traditional MCMC algorithms do not scale well to large data as they typically involve expensive computation using the full dataset. Additionally, scaling classical MCMCs toward modern high-dimensional applications can be problematic. The computational bottleneck led researchers to pursue lower accuracy, higher efficiency trade-offs such as variational inference. Despite its computational efficiency, theoretical guarantees for asymptotic convergence of variational approximations are given typically for specific models, and the objective function can contain multiple local optima trapping commonly used optimization algorithms [7, 13, 21]. In comparison, MCMC techniques have the potential to navigate non-convex surfaces and find better local optima in the process. As the amount of data continues to grow rapidly, the need for scalable MCMC methods for large-scale learning tasks remains critical. In this paper, we propose an MCMC algorithm that is scalable in both the size of the dataset and the dimension of the parameter space. Our algorithm leverages the traveling property of an MCMC sampler to find better solutions to optimization problems in machine learning.

The search for scalable MCMC methods has largely proceeded in two directions. The first approach divides the data into manageable batches and performs MCMC on each batch in parallel. To collectively process the results, most methods either require different machines to communicate with each other in different rounds of MCMC iteration [1], or combine the posterior distribution from each batch to approximate the target posterior [23, 31, 28]. Our work follows the second line of approach, which uses subsamples, or mini-batches, of the full data in each iteration of the MCMC algorithm. The key in analyzing such an algorithm is to understand the noise and bias introduced by the mini-batches.

The broad class of pseudo-marginal algorithms [3] use mini-batches of data to accelerate computation in the Metropolis-Hastings (MH) algorithm [14, 5, 20, 24].

Accepted Manuscript

The exact posterior (or some close approximation) is maintained by constructing an unbiased and nonnegative estimator, which can have a nontrivial form or require carefully chosen lower bound on the likelihood. Another class of methods performs approximate tests in the MH acceptance step using mini-batches. To control the approximation error, an adaptive approach is usually adopted to sequentially increase the size of a batch until an error bound is met [4, 16, 8]. Approaches based on non-reversible MCMC have also been proposed [6]. In practice, some of these methods were tested on large datasets with hundreds of parameters, but further scaling up in parameter dimension toward deep machine learning models would be challenging.

In another direction, past few years have witnessed the rise of stochastic gradient based MCMC algorithms which have shown strong potential in large-scale machine learning applications. These algorithms are developed from diffusion-based MCMC and approximate the gradient with noisy estimates based on mini-batches of data ([26]), a notable example being the Stochastic Gradient Langevin Dynamics (SGLD) and other variants [32, 2, 9, 18]. Many studies have since analyzed the convergence of SGLD by viewing the algorithm as a discrete-time simulation of a continuous stochastic differential equation (SDE) [29, 25]. Unlike algorithms such as MALA which uses the MH acceptance test to correct the errors in discretizing a continuous system (e.g. [27]), SGLD completely avoids the costly computation of the MH ratio by using a shrinking step size. In practice, this implies the algorithm eventually converges to a local optimum.

We propose a general mini-batch MH algorithm whose invariant distribution approximates a tempered version of the target posterior. By augmenting the system with a variable related to the subsampling procedure, we show our algorithm is a reversible Markov chain thus has an invariant distribution. The idea of augmenting the system to sample a tempered posterior was also explored by [19] to heuristically design a mini-batch Metropolis sampler, but their algorithm differs in the use of mini-batches and they did not offer theoretical support for the

Accepted Manuscript

method. [10] introduced a mini-batch Gibbs sampler capable of exact sampling from certain graphical models. Finally, a connection between tempering and subsample variance was also mentioned in [5]. Here, we provide a rigorous theoretical foundation for mini-batching in MH. We emphasize that our aim here is not Bayesian inference from the exact posterior. Rather, we exploit the tempered posterior with an efficient MCMC sampler to obtain better solutions from a global optimization.

With mild assumptions on the likelihood and allowing the parameter dimension to grow at a suitable rate, we provide full theoretical analysis to i) show the invariant distribution of our algorithm approximately preserves the modes of the true posterior, which is an important property for optimization tasks, and ii) bound the distance between the invariant distribution and the tempered posterior. To further enhance the utility of our algorithm in high dimensional applications, we design a proposal function based on Reversible Stochastic Gradient Langevin Dynamic (RSGLD) to make the calculation of MH ratio computationally efficient while ensuring reasonable acceptance probability. We show that the proposal significantly enhances acceptance probability in regions with strong gradient information and explores flat regions in a way similar to random walk. Empirically, we demonstrate the tempering effect inherent to our algorithm helps the Markov chain jump out of local optima and travel between differently modes more easily. Most importantly, we show our mini-batch MH algorithm combined with the RSGLD proposal can be applied to efficiently train neural networks.

The rest of the paper is organized as follows. In Section 2, we introduce our algorithm and provide theoretical analysis of its stationary distribution. In the high dimensional setting, we also design a proposal function called RSGLD and show that adding the reverse move significantly increases the acceptance probability when the gradient is strong. In Section 3, we demonstrate with an array of examples from simple Gaussian models to neural networks with $>10^5$ parameters that our algorithm combines the traveling property of an MCMC

Accepted Manuscript

sampler and the computational efficiency of stochastic optimization methods, thus showing good promise for optimization tasks in deep machine learning applications. In the neural network examples, our algorithm shows higher accuracy overall and better stability for larger learning rates compared to other popular optimization methods.

2 Methods

We first introduce our algorithm and outline its connection to tempering using an augmented variable. We then show under appropriate assumptions, the stationary distribution of the mini-batch MH approximately preserves the modes of the target posterior and is close to a tempered posterior in distribution. In the high dimensional setting, we design a proposal function that can navigate a complex surface guided by gradient information and ensure the acceptance probability does not diminish too quickly as the dimension grows.

2.1 MH MCMC with batch tempering (MHBT)

Under the usual Bayesian setting, let $\mathbf{x} = (x_1, \ldots, x_n) \in \mathsf{X}^n, \mathsf{X} \subset \mathbb{R}^p$, be iid samples drawn from distribution $p(\cdot \mid \boldsymbol{\theta}^*)$, where $\boldsymbol{\theta}^* \in \Theta \subset \mathbb{R}^d$ denotes the parameters. Let $\pi_0(\boldsymbol{\theta})$ be the prior on $\boldsymbol{\theta}$. We are interested in sampling from the target posterior $\pi(\boldsymbol{\theta}) \propto \pi_0(\boldsymbol{\theta}) \prod_{i=1}^{n} p(x_i \mid \boldsymbol{\theta})$ using the MH algorithm. In each iteration of classical MH, given some proposal function $q(\cdot)$, a move from $\boldsymbol{\theta}$ to $\boldsymbol{\theta}'$ is accepted with probability given by the MH ratio,

$$r(\boldsymbol{\theta} \to \boldsymbol{\theta}') = \min\left\{1, \frac{\pi(\boldsymbol{\theta}')q(\boldsymbol{\theta}' \to \boldsymbol{\theta})}{\pi(\boldsymbol{\theta})q(\boldsymbol{\theta} \to \boldsymbol{\theta}')}\right\}.$$

For large *n*, the evaluation of $\pi(\cdot)$ is costly. Now denote

$$\ell_i(\boldsymbol{\theta}) = \log p(x_i \mid \boldsymbol{\theta}),\ \mu(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n} \ell_i(\boldsymbol{\theta}),\ \hat{\mu}_I(\boldsymbol{\theta}) = \frac{1}{|I|}\sum_{j \in I} \ell_j(\boldsymbol{\theta}) \text{ with } I \subset \{1, \ldots, n\} = [n]$$

being an index subset. Let I_m be the collection of I such that $|I| = m$. We will use $\hat{\mu}_I(\boldsymbol{\theta})$ to approximate $\mu(\boldsymbol{\theta})$.

Accepted Manuscript

We next derive our algorithm, *MH MCMC with batch tempering (MHBT)*, using an augmented system [1]. Consider an auxiliary variable $\tau \in \{0,1\}^n$ with $I(\tau) = \{i : \tau_i = 1\}$ and $|I(\tau)| = m$, then we can write $\hat{\mu}_{I(\tau)} = \frac{1}{m}\sum_{i=1}^{n} \ell_i(\boldsymbol{\theta})\tau_i$. Jointly for $(\boldsymbol{\theta}, \tau)$, consider the proposal $q((\boldsymbol{\theta}, \tau) \to (\boldsymbol{\theta}', \tau')) = q(\boldsymbol{\theta} \to \boldsymbol{\theta}')\nu_{m,n}(\tau')$ and the target distribution

$$\tilde{\pi}(\boldsymbol{\theta}, \tau) \propto e^{c_n \hat{\mu}_{I(\tau)}(\boldsymbol{\theta})} \nu_{m,n}(\tau) \qquad (1)$$

where $\nu_{m,n}$ is the uniform distribution over I_m and c_n is a scaling constant that will be explained soon. Performing the classical MH algorithm on the augmented pair $(\boldsymbol{\theta}, \tau)$ with the above proposal and $\tilde{\pi}$, simple algebra shows the acceptance probability is given by

$$r((\boldsymbol{\theta}, \tau) \to (\boldsymbol{\theta}', \tau')) = \min\left\{1, \frac{\tilde{\pi}(\boldsymbol{\theta}', \tau')q((\boldsymbol{\theta}', \tau') \to (\boldsymbol{\theta}, \tau))}{\tilde{\pi}(\boldsymbol{\theta}, \tau)q((\boldsymbol{\theta}, \tau) \to (\boldsymbol{\theta}', \tau'))}\right\} = \min\left\{1, \frac{q(\boldsymbol{\theta}' \to \boldsymbol{\theta})e^{c_n \hat{\mu}_{I(\tau')}(\boldsymbol{\theta}')}}{q(\boldsymbol{\theta} \to \boldsymbol{\theta}')e^{c_n \hat{\mu}_{I(\tau)}(\boldsymbol{\theta})}}\right\}, \qquad (2)$$

which can be calculated efficiently using a new mini-batch $I(\tau')$ of the data. Since the stationary distribution of this Markov chain is $\tilde{\pi}$, marginalizing (1) over τ (with τ in the batches suppressed for clarity),

$$\tilde{\pi}(\boldsymbol{\theta}) \propto \left(\pi(\boldsymbol{\theta})\right)^{1/T} \binom{n}{m}^{-1} \sum_{I \in \mathsf{I}_m} e^{c_n(\hat{\mu}_I(\boldsymbol{\theta}) - \mu(\boldsymbol{\theta}))} \qquad (3)$$

where $T = n / c_n$ is the temperature. In this sense, the mini-batch stationary distribution is approximately a tempered version of the posterior, up to a bias term. Unlike pseudo-marginal MCMCs, we do not require constructing an unbiased estimate of the likelihood, which leads to improved computational efficiency. The bias becomes small (i.e. the bias term becomes close to 1) as n increases for appropriate m and c_n since $\hat{\mu}_I(\boldsymbol{\theta}) - \mu(\boldsymbol{\theta})$ becomes small. c_n controls the trade-off between approximation error and the tempering amount – a smaller

Accepted Manuscript

c_n leads to a smaller error but a higher temperature. The choice of c_n and the exact error rate will also be discussed in Section 2.2.

We summarize the mini-batch MH algorithm in Algorithm 1 (with τ suppressed for simplicity).

Algorithm 1 MH MCMC with batch tempering (MHBT)

Input: data **x**, batch size m, constant c_n, proposal $q(\boldsymbol{\theta} \to \boldsymbol{\theta}')$, log likelihood $\ell(\boldsymbol{\theta})$, initial $\boldsymbol{\theta}_0$, I_0.

for $t = 0,1,\ldots$ **do**

Draw $\boldsymbol{\theta}'$ from $q(\boldsymbol{\theta}_t \to \boldsymbol{\theta}')$, an index set $I' \in \mathrm{I}_m$ randomly, and $u \sim \mathrm{Unif}[0,1]$.

Compute acceptance probability $r = \min\left\{1, \dfrac{q(\boldsymbol{\theta}' \to \boldsymbol{\theta}_t) e^{c_n \hat{\mu}_{I'}(\boldsymbol{\theta}')}}{q(\boldsymbol{\theta}_t \to \boldsymbol{\theta}') e^{c_n \hat{\mu}_{I_t}(\boldsymbol{\theta}_t)}}\right\}$

if $u < r$ **then**

$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}', I_{t+1} = I'$,

else

$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t, I_{t+1} = I_t$.

2.2 Preservation of local optima and convergence to tempered posterior

In this section, we analyze the properties of the stationary distribution $\tilde{\pi}(\boldsymbol{\theta})$. In particular, we show the convergence rate of the bias term in (3) in terms of the two tuning parameters m and c_n. Throughout the rest of the paper, for two positive sequences a_n and b_n, we use the notation $a_n \asymp b_n$ if for large enough n, $a_n \le c_1 b_n, b_n \le c_2 a_n$ for some constants c_1, c_2 not depending on n. $\|\cdot\|_1, \|\cdot\|_2$ denote the ℓ_1, ℓ_2 norm for vectors, and $\|\cdot\|_{op}$ denotes the operator norm of a matrix. $\lfloor a \rfloor$

Accepted Manuscript

is the greatest integer smaller than or equal to *a*. $a \vee b = \max\{a, b\}$. E_{θ^*} is the expectation taken over the data which is generated by the true parameter $\boldsymbol{\theta}^*$.

Consider the regime where both *n* and *m* are large with $m \leq n$. We will also allow the dimension *d* to grow at some suitable rate with respect to *n*. We assume the likelihood function $p(x \mid \boldsymbol{\theta}) := p_{\boldsymbol{\theta}}(x), x \in \mathsf{X}$, belongs to a parametric family satisfying the following conditions.

Assumption 1. *There exist a function L and a vector of measurable function* T *such that* $|\log p_{\boldsymbol{\theta}}(x) - \log p_{\boldsymbol{\theta}}(y)| \leq L(\boldsymbol{\theta}) \| \mathsf{T}(x) - \mathsf{T}(y) \|_1, x, y \in \mathsf{X}$, *with* $L_0 := \sup_{\boldsymbol{\theta} \in \Theta} L(\boldsymbol{\theta}) < \infty$ *and* $\mathsf{E}_{\boldsymbol{\theta}^*} e^{\delta_1 \| \mathsf{T}(X) \|_1} < \infty$ *for some* $\delta_1 > 0$.

Assumption 2. *There exists a measurable function M such that* $|\log p_{\boldsymbol{\theta}}(x) - \log p_{\boldsymbol{\theta}'}(x)| \leq M(x) \| \boldsymbol{\theta} - \boldsymbol{\theta}' \|_1$ *for all* $\boldsymbol{\theta}, \boldsymbol{\theta}' \in \Theta$ *and* $x \in \mathsf{X}$. *In addition, there exists* $\delta_2 > 0$ *such that* $\mathsf{E}_{\boldsymbol{\theta}^*} e^{\delta_2 M(X)} < \infty$.

The above assumptions are mild and require the log likelihood $\log p_{\boldsymbol{\theta}}(x)$ to be suitably smooth in both $\boldsymbol{\theta}$ and *x*. Unlike some pseudo-marginal MCMC algorithms [3, 20], we do not require the likelihood to be bounded. We show in Supplement S2 that these assumptions can be carried over to a number of commonly used models in statistics and machine learning, such as mixtures of exponential family distributions, linear regression with random feature vectors, and classification tasks with fully connected neural networks (which include logistic regression as a special case). In the exponential family example, $\mathsf{T}(\cdot)$ and $M(\cdot)$ are in fact functions of the sufficient statistic. In the neural network example, the constant $L(\boldsymbol{\theta})$ is related to the network complexity measure.

For large *n*, it suffices to consider the population log likelihood $\mu_{\boldsymbol{\theta}} := \mathsf{E}_{\boldsymbol{\theta}^*} \log p_{\boldsymbol{\theta}}(X)$. Let $\boldsymbol{\theta}_0$ be a stationary point of $\mu_{\boldsymbol{\theta}}$ such that it represents a well-separated local optimum in the following sense.

Accepted Manuscript

Assumption 3. *μ_θ is twice continuously differentiable in θ. $\theta_0 \in Int(\Theta)$ and the Hessian of μ_θ at θ_0 has eigenvalues $\lambda_i(H_{\theta_0}) < 0$ for all $i = 1, \ldots, d$.*

Note that the assumption implies there exist $\kappa_0, \delta_0 > 0$ such that $\mu_{\theta_0} - \mu_\theta \geq \kappa_0 \| \theta - \theta_0 \|_2$ for all $\| \theta - \theta_0 \|_2 \leq \delta_0$. Then we have the next theorem showing $\tilde{\pi}(\theta)$ approximately preserves any well-separated local optimum.

Theorem 1. *Suppose θ_0 is a stationary point of μ_θ satisfying Assumption 3. For some $\alpha > 0$, let $c_n \to \infty$ be a sequence such that $\frac{dc_n^{2+\alpha} \log c_n}{m} \to 0$, t be a fixed constant with $t \in (0, 1/2)$, and $\delta_n = \sqrt{\frac{3\log(1/(1-2t))}{\kappa_0 c_n}}$. Then under Assumptions 1, 2, for large n,*

$$\sup_{\theta \in R_n} \log \tilde{\pi}(\theta) \leq \sup_{\theta \in \mathrm{B}(\theta_0; \delta_n)} \log \tilde{\pi}(\theta) - \log(1/(1-2t)), \qquad (4)$$

with probability at least $1 - \eta_n$. Here $R_n = \{\theta \in \Theta : \delta_n < \| \theta - \theta_0 \|_2 < \delta_0\}$, $\mathrm{B}(\theta_0; \delta_n) = \{\theta \in \Theta : \| \theta - \theta_0 \|_2 < \delta_n\}$, and $\eta_n \coloneqq \frac{1}{t^2 \left\lfloor \frac{n}{m} \right\rfloor c_n^\alpha}$.

The theorem states that with high probability, the supremum of $\log \tilde{\pi}(\theta)$ in the shrinking ball $\mathrm{B}(\theta_0; \delta_n)$ is larger than any point in the surrounding region R_n by a constant margin. This guarantees with high probability $\tilde{\pi}(\theta)$ has a local optimum lying in a shrinking neighborhood centered at θ_0. The preservation of local optima is important for optimization tasks.

We can further bound the distance between $\tilde{\pi}(\theta)$ and the tempered posterior $\pi^{1/T}(\theta)$ with one more assumption.

Assumption 4. *Θ is compact.*

Accepted Manuscript

Theorem 2. *Denote $\pi_T(\boldsymbol{\theta}) \propto \pi^{1/T}(\boldsymbol{\theta})$ the tempered posterior. Under Assumptions 1, 2 and 4, for some $\alpha > 0, \delta_n \to 0$ slower than $c_n^{-\alpha/2}, c_n \to \infty$ such that $\frac{dc_n^{2+\alpha} \log c_n}{m} \to 0, D_{KL}(\pi_T \,||\, \tilde{\pi}) \leq \delta_n$ with probability at least $1 - \eta'_n, \eta'_n \geq \frac{1}{\delta_n^2 \left\lfloor \frac{n}{m} \right\rfloor c_n^{\alpha}}$, for large n.*

The proofs of the above theorems can be found in Supplement S1.

Remark 1.

1. *Both Theorems 1 and 2 require $\frac{dc_n^{2+\alpha} \log c_n}{m} \to 0$, meaning c_n and d need to go to infinity at a controlled rate. The convergence regime in both theorems covers a wide spectrum of batch size m, from $\omega(1)$ to O(n).*
2. *For a given m, if d is fixed, we can choose c_n to be a value close to but smaller than $\sqrt{m}$ to make sure the temperature is not too high while the convergence holds. In Section 3.1, we show using numerical experiments that the choice of c_n is very robust in low dimensional models.*
3. *The convergence requirement has a linear dependence on d. If $m = n^{\gamma}$ for some fraction γ, d can also go to infinity at the rate of n raised to some fractional power.*

2.3 MHBT with stochastic gradient based proposal for neural networks

In large-scale machine learning tasks such as training deep neural networks (DNN), the high dimensionality and complex nature of the loss function surface have posed significant challenges for designing an MCMC sampler that can i) efficiently navigate the high dimensional surface, ii) result in a reasonable acceptance probability in the MH test, and iii) be computationally feasible. Recent studies on stochastic gradient MCMC have demonstrated their potential in training DNNs [9, 18, 34]. However, these methods are derived from continuous-time SDEs, and each discretization step introduces some error which ideally

Accepted Manuscript

could be corrected with an MH acceptance test. Many of these methods require a shrinking learning rate in order to circumvent the MH test. In this section, we propose and analyze a stochastic gradient-based proposal with appropriate MH correction, which is computationally efficient for DNN applications.

Proposal with Reversible Stochastic Gradient Langevin Dynamics (RSGLD)

Our goal is to design a proposal function that can explore a complex high dimensional surface efficiently guided by gradient information. We will start by considering the proposal used in SGLD, which has been widely adopted in the literature for large-scale training tasks. Let $\hat{g}_I(\boldsymbol{\theta}) = \frac{1}{|I|}\sum_{i\in I}\nabla_{\boldsymbol{\theta}}\ell_i(\boldsymbol{\theta})$ be the average gradient of mini-batch I, the proposal move for SGLD is given by

$$\boldsymbol{\theta}' = \boldsymbol{\theta} + \frac{\epsilon}{2}\hat{g}_I(\boldsymbol{\theta}) + \frac{\sqrt{2\epsilon}}{n}N(0, I_d), \qquad (5)$$

where ϵ is the learning rate, $N(0, I_d)$ is the iid Gaussian noise. Note that we have written the learning rate in a form that is consistent with the convention for SGD, so ϵ differs from the learning rate in the convention for SGLD by a factor of n. The original SGLD avoids the MH correction step since it is costly to compute using the full data.

In practice, in addition to the computational efficiency issue, another difficulty arises from the acceptance probability as d increases. Using (5) as the proposal in Algorithm 1, it can be treated as a mini-batch version of the MALA algorithm [27] (and the more general Hamiltonian MCMC). It is known that in these full-batch algorithms, ϵ needs to scale like $d^{-\frac{1}{4}}n^{-1}$ to maintain a reasonable acceptance probability [22]. As an illustration, we consider using (5) as the proposal in Algorithm 1 to sample from the d-dimensional Gaussian $N(0, I_d)$, where $d = 1, 10, 10^2, 10^3$, and $n = 10^4, m = 1000, c_n = 20$. In Figure 1(a), we computed the average acceptance probability for the first 2000 iterations initializing at the origin and then selected the largest learning rate ϵ with average acceptance

Accepted Manuscript

probability at least 0.5 and 0.1. ϵ was chosen from a grid that scales like $d^{-\frac{1}{4}}n^{-1}$. As can be seen, ϵ quickly diminishes to below 10^{-7} when the dimension reaches 10^3, if we still want to maintain a reasonable acceptance probability. Such a small learning rate results in very slow convergence and is therefore usually infeasible for practical use.

Our proposal, Reversible Stochastic Gradient Langevin Dynamics (RSGLD), is based on SGLD but enhances the acceptance probability by allowing the sampler to move in the direction of either ascending or descending gradient with an adjusted Gaussian noise. Using RSGLD as the proposal in Algorithm 1 gives us a mini-batch MH algorithm that both utilizes gradient information and is computationally efficient. Our proposal modifies (5) in two ways: i) a coin flip decides whether the move will be in the positive or negative direction of the gradient. For convenience, we will henceforth refer to a move in the positive (or negative) gradient direction as a *forward* (or *backward*) step; ii) the backward step is coupled with a larger Gaussian noise. The new state $\boldsymbol{\theta}'$ is sampled by

$$\boldsymbol{\theta}' = \begin{cases} \boldsymbol{\theta} + \epsilon \hat{g}_I(\boldsymbol{\theta}) + \frac{\sqrt{2\epsilon}}{n} N(0, I_d), & \text{with probability } 1/2, \\ \boldsymbol{\theta} - \epsilon \hat{g}_I(\boldsymbol{\theta}) + \frac{\sqrt{2\epsilon}}{n} \beta N(0, I_d), & \text{with probability } 1/2 \end{cases} \quad (6)$$

for some constant $\beta \geq 1$. Denote this proposal $q_I(\boldsymbol{\theta} \to \boldsymbol{\theta}')$, then

$$q_I(\boldsymbol{\theta} \to \boldsymbol{\theta}') = \frac{1}{2}\phi\left(\boldsymbol{\theta}' - \boldsymbol{\theta} - \epsilon \hat{g}_I(\boldsymbol{\theta}); \frac{2\epsilon}{n^2} I_d\right) + \frac{1}{2}\phi\left(\boldsymbol{\theta}' - \boldsymbol{\theta} + \epsilon \hat{g}_I(\boldsymbol{\theta}); \frac{2\epsilon\beta^2}{n^2} I_d\right), \quad (7)$$

where $\phi(\cdot; \Sigma)$ is the density of a multivariate Gaussian with zero mean and covariance matrix Σ.

In Algorithm 1, the acceptance probability for moving from $(\boldsymbol{\theta}_t, I_t) \to (\boldsymbol{\theta}', I')$ becomes

Accepted Manuscript

$$\min\left\{1, \frac{q_{I'}(\boldsymbol{\theta}' \to \boldsymbol{\theta}_t) e^{c_n \hat{\mu}_{I'}(\boldsymbol{\theta}')}}{q_{I_t}(\boldsymbol{\theta}_t \to \boldsymbol{\theta}') e^{c_n \hat{\mu}_{I_t}(\boldsymbol{\theta}_t)}}\right\}. \quad (8)$$

Similar to the argument in Section 2.1, we can show using an auxiliary variable the above mini-batch MH algorithm is closely related to a tempered MCMC. We refer to Supplement S3 for details.

As an illustration to show both the backward step and its associated, enlarged Gaussian noise increase the acceptance probability, we used the same Gaussian setting as before (sampling from $N(0, I_d)$, where $d = 10, 10^2, 10^3$, and $n = 10^4, m = 1000, c_n = 20$) and tested β = 1, which corresponds to only adding the backward move; and β = 2, which increases the size of the Gaussian noise in the backward move. In Figure 1(b)-(d), we can see both adding the backward move and increasing the Gaussian noise significantly improve the acceptance probability, and the trend is consistent for different dimensions.

Analysis of acceptance probability

In this section, we show that the RSGLD proposal leads to larger proposal ratio, thus increasing the MH ratio and acceptance probability overall. To focus on the behavior of the algorithm, we take the data **x** as given and fixed, and the only randomness lies in the selection of data batch and the Gaussian perturbation. Let $\mathbf{Z} \sim N(0, I_d)$ and $H_I(\boldsymbol{\theta})$ be the Hessian matrix of $\hat{g}_I(\boldsymbol{\theta})$ on mini-batch *I*. We assume the following conditions hold.

Assumption 5. $\sup_{I \in \mathrm{I}_m, \boldsymbol{\theta}} \| H_I(\boldsymbol{\theta}) \|_{op} \leq \lambda$, *where* $\|\cdot\|_{op}$ *is the operator norm.*

Assumption 6. *For every* $\boldsymbol{\theta}$, *all batches give similar gradients. More specifically, for any two batches I and J,*

$$\| \hat{g}_J(\boldsymbol{\theta}) - \hat{g}_I(\boldsymbol{\theta}) \|_2 = O(\text{ò} \| \hat{g}_I(\boldsymbol{\theta}) \|_2). \quad (9)$$

Accepted Manuscript

Proposition 1. *For large n, suppose Assumptions 5 and 6 hold. Then depending on where the sampler is in the landscape of the target likelihood, we have the following approximations for the proposal ratio* $\frac{q_J(\boldsymbol{\theta}' \to \boldsymbol{\theta})}{q_I(\boldsymbol{\theta} \to \boldsymbol{\theta}')}$*, where* $\boldsymbol{\theta}$ *is the current parameter value to be updated and I is the current batch.*

Case 1)*. Assume there exists a small constant* η_0 *such that* $\| \mathbf{Z} \|_2 \leq \eta_0 \cdot n\sqrt{\delta/2} \| \hat{g}_I(\boldsymbol{\theta}) \|_2$ *with high probability (i.e. with probability approaching 1), and the learning rate* ϵ *is small enough such that* $\frac{n^2\delta^2(\delta \vee \eta_0)}{\beta^2 - 1} \| \hat{g}_I(\boldsymbol{\theta}) \|_2^2 = o(d)$ *for large d,* $\beta > 1$*. Then*

- *if the update in (6) results in a forward move, we have* $\frac{q_J(\boldsymbol{\theta}' \to \boldsymbol{\theta})}{q_I(\boldsymbol{\theta} \to \boldsymbol{\theta}')} > 1$ *with high probability.*
- *if the update in (6) results in a backward move, we have* $\frac{q_J(\boldsymbol{\theta}' \to \boldsymbol{\theta})}{q_I(\boldsymbol{\theta} \to \boldsymbol{\theta}')} = o_P(1)$ *.*

Case 2)*. Assume* $\| \hat{g}_I(\boldsymbol{\theta}) \|_2 = 0$*, and the learning rate* ϵ *is small enough such that* $\delta = o(d^{-1})$ *for large d. Then we have* $\frac{q_J(\boldsymbol{\theta}' \to \boldsymbol{\theta})}{q_I(\boldsymbol{\theta} \to \boldsymbol{\theta}')} = 1 + o_P(1)$ *for both directions in (6).*

We defer the proof to Supplement S4.

Remark 2.

1. *In this proposition, we consider the behavior of the proposal ratio in different regions of the landscape. The condition in Case 1) means the sampler is at a location where gradient information is strong. Simple rearranging in (6) shows in this case, the gradient part dominates the Gaussian noise. In Case 2), the sampler has reached a flat region of the landscape.*

Accepted Manuscript

2. *If* $\| \hat{g}_I(\boldsymbol{\theta}) \|_2 = O(\sqrt{d})$ *, in Case 1)* ϵ *needs to satisfy* ò $n^{-1}\sqrt{\beta^2 - 1} / \sqrt{\eta_0}$ *, the rate of which no longer depends on d and scales better than before (* $d^{-\frac{1}{4}} n^{-1}$*). Sparse* $\hat{g}_I(\boldsymbol{\theta})$ *(such as in typical neural networks) and large β can allow for even larger learning rates.*
3. *The result in Case 1) implies it is more likely for the MH step to accept a forward move than a backward move when the gradient is strong. This is a desirable property in optimization tasks for maintaining efficiency. In particular, the proposal ratio is lower bounded by 1 in the forward direction and hence will no longer shrink the overall MH ratio to zero. In Case 2), the proposal in the sampler behaves like a random walk if the learning rate is sufficiently small.*

3 Experiments

3.1 Distributions in low dimensions

Convergence to known posterior

We first examined the convergence behavior of MHBT compared to the conventional MCMC sampler using the full dataset (termed full batch MCMC). As the analysis in Section 2.2 suggests, MHBT converges to a tempered version of the original posterior distribution. In order to explicitly measure the distance from this posterior, we considered *d*-dimensional (*d* = 2 and 5) Gaussian distributions with unknown mean $\boldsymbol{\theta}$, known covariance I_d, where the prior of $\boldsymbol{\theta}$ was set to be $N(0, I_d)$. We generated $n = 10^5$ samples from this distribution with each true $\theta_i^* = 2$. It follows then the posterior of $\boldsymbol{\theta}$ given the data **x** is $N\left(\frac{n}{n+1}\bar{\mathbf{x}}, (n+1)^{-1} I_d\right)$, where $\bar{\mathbf{x}}$ is the sample average. Raising the posterior to temperature *T* changes the variance to $\frac{T}{n+1} I_d$. Mini-batch sampling was performed with Algorithm 1, setting the proposal $q(\cdot)$ as a Gaussian random walk with step size δ and mini-batch size *m* = 1000. We found no significant difference in the results varying *m* from 500 to 5000. Full batch MCMC was performed on the tempered posterior

Accepted Manuscript

also with the same type of random walk proposal. The same step size δ was chosen for both algorithms and the average acceptance probability was around 0.3.

Figure 2(a) shows the total variation distance between the sampled distributions and true tempered posterior for the two MCMC algorithms on d-dimensional Gaussian, as the number of iterations increases. The distance was calculated by running 10^5 independent MCMC chains and taking the same number of independent samples from the tempered distribution, followed by discretization to group the values into d-dimensional histograms. The results shown correspond to $c_n = 20$, which is smaller than $\sqrt{m}$ as discussed in Remark 1, although we note that a range of c_n values (5-30) led to very similar results. For both $d = 2$ and 5, MHBT converges at a rate almost identical to full batch MCMC to the tempered posterior.

Gaussian mixture

To illustrate the tempering effect of MHBT and examine the accuracy of the approximation in Section 2.2, we consider an example in [32]. We generated $n = 10^5$ samples from a 2-component mixture Gaussian model with parameters $\boldsymbol{\theta} = (\theta_1, \theta_2)$ following:

$$\boldsymbol{\theta} \sim N(0, \mathrm{diag}(\sigma_1^2, \sigma_2^2)), \quad X_i \sim 0.5N(\theta_1, \sigma_x^2) + 0.5N(\theta_1 + \theta_2, \sigma_x^2),$$

where $\sigma_x^2 = 2, \sigma_1^2 = 10, \sigma_2^2 = 1$. The posterior distribution of $\boldsymbol{\theta}$ given $\mathbf{x} = (x_1, \ldots, x_n)$ can be calculated explicitly as

$$\pi(\boldsymbol{\theta}) \propto e^{-\frac{1}{2}\left(\theta_1^2/\sigma_1^2 + \theta_2^2/\sigma_2^2\right)} \prod_{i=1}^{n} \left(e^{-\frac{1}{4}(\theta_1^2 - 2\theta_1 x_i)} + e^{-\frac{1}{4}((\theta_1+\theta_2)^2 - 2(\theta_1+\theta_2)x_i)} \right).$$

We sampled $\boldsymbol{\theta}$ using Algorithm 1, where the proposal $q(\cdot)$ is the Gaussian random walk with step size δ. We set the mini-batch size m to 1000 (varying m from 500 to 5000 did not change the results noticeably). There remain two tuning

Accepted Manuscript

parameters in the algorithm: c_n and δ. We chose c_n to be 20 and δ such that the average acceptance probability was around 0.3. Very similar results can be obtained by a range of c_n values (e.g. 5-30).

Figure 2(b)-(c) show the sampled θ from 10^5 iterations and the contour plot of the tempered log posterior, $\log \pi_T(\boldsymbol{\theta}) \propto 1/T \log \pi(\boldsymbol{\theta})$. We can see that the two modes in these plots coincide well.

Figure 2(d)-(f) compare the trajectory of MHBT with that of the full batch MCMC in one of the two dimensions. The latter sampling was performed on the original posterior distribution, and the step size of the random walk was chosen so that the average probability was around 0.3. We fixed $\theta_1 = 0$ and increased θ_2 from 0.5 to 4 so that the two modes in the posterior distribution became increasingly separated. In each case, MHBT is capable of visiting the two modes of the distribution whereas the full batch MCMC is trapped in one of the modes. This highlights the effect of tempering brought about by the mini-batch algorithm, which makes the landscape smoother and easier for the sampler to travel.

3.2 Neural networks

Fully connected neural networks

We first tested MHBT with RSGLD on the standard MNIST handwritten digit classification task. The dataset was loaded directly from TensorFlow tutorial and consists of 55,000 instances for training and 10,000 instances for testing. We considered a neural network containing one hidden layer with 600 nodes and ReLU activation function ($\sim 4 \times 10^5$ parameters). The outputs from the layer are connected to a 10-class softmax layer for classification. In this case, the log likelihood function is the negative of the cross entropy loss. The batch size was set to 100, which is a typical size used for neural networks. In Supplement S5, we provide additional results showing how our algorithm behaves under different m, as well as the interaction between m and the learning rate ϵ. We compared the performance of our method with a number of popular optimization methods in

Accepted Manuscript

the neural network literature for a range of learning rates. In each training, we started RSGLD with a large β to initiate the moves and gradually decreased it as the training progressed.

Choosing β. Throughout training, we monitored the overall acceptance probability for each epoch, where by convention one epoch equals the total number of iterations it takes to step through the whole training dataset (in this case $55000/100 = 550$ iterations). We decreased β according to the following adjustment phase once the acceptance probability became larger than 0.4 at the end of each epoch. During the adjustment phase, we ran 100 forward steps using the current parameter values and computed the MH acceptance probability. If the average probability of these forward steps exceeded 0.7, we decreased β by 5%. The new β value was then tested again with 100 forward steps. The maximum reduction allowed in each adjustment phase was 50%. The next epoch of training was then run with the new β value. On the other hand, when the average probability for one epoch dropped below 0.2, we increased β by 5%. We observed that in all experiments, β eventually stabilized to some constant slightly larger than 1.

Comparison with other methods. We performed extensive comparison with SGD and SGLD using various learning rates and multiple rounds of training to assess the stability of each method. Each round of training lasted 2.75×10^5 iterations (500 epochs), and all the parameters were initialized with independent $N(0, 0.03)$ distribution. The same batch size (100) was used for all the methods. In this high dimensional setting, we explored a range of c_n values around the batch size and show results using c_n = 100. We additionally tested $c_n = 50, 200$ under the same settings; the results are very similar thus omitted.

Table 1 shows the prediction errors of the three methods on the testing set, using the top class from the softmax layer as the predicted label. Each number is the median error obtained from 30 training rounds with the corresponding standard

Accepted Manuscript

deviation shown in parentheses. Overall, the performance of RSGLD improves with large learning rate and eventually achieves better accuracy (smaller error) than that attainable by SGD or SGLD at any learning rate. RSGLD shows substantially better stability for large learning rate than the other two methods. In particular, when the learning rate is 0.2 or larger, SGD and SGLD can fail to converge completely for a significant fraction of the training rounds, which explains the large standard deviations. In general, the standard deviation of errors increases with the learning rate for all the methods, showing stability is hard to achieve with a large learning rate although it can lead to faster convergence and potentially better prediction. As explained in [33], using a large learning rate can help algorithms maintain a trajectory high from the valley floor and more easily overcome energy barriers as they explore the loss surface with stochastic gradients. In this sense, the stability of RSGLD under large learning rates is beneficial for training DNNs. We also observe that in all the experiments, the backward step in RSGLD was much less likely to be accepted compared to the forward step, which is discussed in Case 1) of Proposition 1 and is desirable for optimization efficiency. Since the forward step is identical to SGLD, this suggests the added MH acceptance-rejection step plays a role in improving the optimization result. In Supplement S5, we compare prediction error trajectories from running RSGLD and SGLD to illustrate how the MH acceptance-rejection step implicitly selects a more efficient trajectory through the parameter space.

In addition to checking the average performance of the methods from multiple training rounds, we also examine the lowest prediction error achieved under each learning rate from 30 rounds of training. Since SGD and SGLD did not converge most of the time under large learning rates, showing the average or median error would make the plot scale badly. Fig 3(a) shows a trend similar to Table 1 with RSGLD outperforming the other two methods for large learning rates. Overall the lowest error is achieved by RSGLD with learning rate around 0.4-0.5. Examples of detailed testing error trajectories for various methods are shown in Fig 3(b), where for each method we selected the learning rate with the best performance.

Accepted Manuscript

We have further included RMSprop [30] with learning rate 0.005 and Adam [15] with learning rate 0.001 for comparison. The learning rate was chosen by optimization via grid search for these two methods.

Convolutional neural networks (CNN)

We next tested a standard three-layer CNN on the CIFAR-10 RGB image dataset [17], the detailed architecture of which is listed in Supplement Table S1. The network has around 4×10^6 parameters. The dataset consists of 60000 32 × 32 RGB images in 10 classes, with 50000 for training and 10000 for testing. All parameters were initialized independently with $N(0,0.02)$ distribution. The same batch size and c_n were used, and the same schedule was used for decreasing β as in the last example. Similar to the comparison performed on MNIST, we used 20 rounds of independent training for each learning rate to check the accuracy and stability of RSGLD, SGD and SGLD, with each round lasting for 10^5 iterations. As shown in Table 2, RSGLD consistently outperformed the other two methods and the margin of difference becomes larger as the learning rate increases.

4 Conclusion

In this paper, we study an efficient MH-MCMC algorithm which uses mini-batches of data. We draw connections between the stationary distribution of this Markov chain and the tempered posterior, and provide the approximation errors for a general class of likelihood functions. We also propose RSGLD, a stochastic gradient based proposal to help the sampler navigate complex high dimensional surface with reasonable acceptance probability in the MH acceptance test. Empirically, we demonstrate the algorithm has good convergence behavior and the tempering effect helps move between well separated modes in classical low dimensional models. We demonstrate the efficacy of RSGLD in training neural networks with the MNIST and CIFAR-10 datasets and show that compared to

Accepted Manuscript

popular optimization methods, we achieve improved accuracy and stability when the learning rate is large.

There are several interesting directions for future work. We can think of the MH acceptance test implicitly performing batch selection by sometimes rejecting a move based on the current batch, thus it would be interesting to draw connections with the machine learning literature on optimal batch selections in SGD [35, 36]. In addition, although we have shown the bias term in the stationary distribution is negligible asymptotically, it requires $c_n \to \infty$ at a controlled rate implying the temperature is always much larger than 1. To extend the application of our algorithm from optimization to sampling, it would be helpful to reduce the bias term further while paying careful attention to the additional computational cost it incurs. One can also extend the algorithm in a simulated tempering framework, although the implementation of which in neural networks would be even more challenging. Finally, there are a number of popular stochastic optimization algorithms in the machine learning literature which use adaptive learning rates (e.g. Adagrad [12], RMSprop [30], second order methods [11]). We intend to explore how to incorporate such a feature in an adaptive MCMC framework in future work.

SUPPLEMENTARY MATERIAL

Title: Supplementary Material for “Mini-batch Metropolis-Hastings with Reversible SGLD Proposal”.

Note

[1]For simplicity of description, we assume the prior $\pi_0(\boldsymbol{\theta}) \propto 1$; the algorithm and theoretical results generalize with minor modifications to other priors for large *n*.

References

Accepted Manuscript

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

[2] Sungjin Ahn, Anoop Korattikara, and Max Welling. Bayesian posterior sampling via stochastic gradient Fisher scoring. *arXiv preprint arXiv:1206.6380*, 2012.

[3] Christophe Andrieu, Gareth O Roberts, et al. The pseudo-marginal approach for efficient monte carlo computations. *The Annals of Statistics*, 37(2):697–725, 2009.

[4] Rémi Bardenet, Arnaud Doucet, and Chris Holmes. Towards scaling up Markov chain Monte Carlo: an adaptive subsampling approach. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 405–413, 2014.

[5] Rémi Bardenet, Arnaud Doucet, and Chris Holmes. On Markov chain Monte Carlo methods for tall data. *arXiv preprint arXiv:1505.02827*, 2015.

[6] Joris Bierkens, Paul Fearnhead, Gareth Roberts, et al. The zig-zag process and super-efficient sampling for bayesian analysis of big data. *The Annals of Statistics*, 47(3):1288–1320, 2019.

[7] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, (just-accepted), 2017.

[8] Haoyu Chen, Daniel Seita, Xinlei Pan, and John Canny. An efficient minibatch acceptance test for Metropolis-Hastings. *arXiv preprint arXiv:1610.06848*, 2016.

[9] Tianqi Chen, Emily Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo. In *International Conference on Machine Learning*, pages 1683–1691, 2014.

Accepted Manuscript

[10] Christopher De Sa, Vincent Chen, and Wing Wong. Minibatch gibbs sampling on large graphical models. *arXiv preprint arXiv:1806.06086*, 2018.

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

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

[13] Behrooz Ghorbani, Hamid Javadi, and Andrea Montanari. An instability in variational inference for topic models. *arXiv preprint arXiv:1802.00568*, 2018.

[14] Pierre E Jacob and Alexandre H Thiery. On nonnegative unbiased estimators. *The Annals of Statistics*, 43(2):769–784, 2015.

[15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[16] Anoop Korattikara, Yutian Chen, and Max Welling. Austerity in MCMC land: Cutting the Metropolis-Hastings budget. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 181–189, 2014.

[17] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[18] Chunyuan Li, Changyou Chen, David E Carlson, and Lawrence Carin. Preconditioned stochastic gradient langevin dynamics for deep neural networks. In *AAAI*, volume 2, page 4, 2016.

Accepted Manuscript

[19] Dangna Li and Wing H Wong. Mini-batch tempered mcmc. *arXiv preprint arXiv:1707.09705*, 2017.

[20] Dougal Maclaurin and Ryan P Adams. Firefly Monte Carlo: Exact MCMC with subsets of data. In *UAI*, pages 543–552, 2014.

[21] Soumendu Sundar Mukherjee, Purnamrita Sarkar, YX Rachel Wang, and Bowei Yan. Mean field for the stochastic blockmodel: Optimization landscape and convergence issues. In *Advances in Neural Information Processing Systems*, pages 10717–10727, 2018.

[22] Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011.

[23] Willie Neiswanger, Chong Wang, and Eric Xing. Asymptotically exact, embarrassingly parallel mcmc. In *UAI*, 2013.

[24] Matias Quiroz, Robert Kohn, Mattias Villani, and Minh-Ngoc Tran. Speeding up mcmc by efficient data subsampling. *Journal of the American Statistical Association*, pages 1–13, 2018.

[25] Maxim Raginsky, Alexander Rakhlin, and Matus Telgarsky. Non-convex learning via stochastic gradient langevin dynamics: a nonasymptotic analysis. *arXiv preprint arXiv:1702.03849*, 2017.

[26] Herbert Robbins and Sutton Monro. A stochastic approximation method. In *Herbert Robbins Selected Papers*, pages 102–109. Springer, 1985.

[27] Gareth O Roberts, Richard L Tweedie, et al. Exponential convergence of langevin distributions and their discrete approximations. *Bernoulli*, 2(4):341–363, 1996.

Accepted Manuscript

[28] Steven L Scott, Alexander W Blocker, Fernando V Bonassi, Hugh A Chipman, Edward I George, and Robert E McCulloch. Bayes and big data: The consensus monte carlo algorithm. *International Journal of Management Science and Engineering Management*, 11(2):78–88, 2016.

[29] Yee Whye Teh, Alexandre H Thiery, and Sebastian J Vollmer. Consistency and fluctuations for stochastic gradient langevin dynamics. *Journal of Machine Learning Research*, 17:1–33, 2016.

[30] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

[31] Xiangyu Wang and David B Dunson. Parallelizing mcmc via weierstrass sampler. *arXiv preprint arXiv:1312.4605*, 2013.

[32] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 681–688, 2011.

[33] Chen Xing, Devansh Arpit, Christos Tsirigotis, and Yoshua Bengio. A walk with sgd. *arXiv preprint arXiv:1802.08770*, 2018.

[34] Nanyang Ye, Zhanxing Zhu, and Rafal K Mantiuk. Langevin dynamics with continuous tempering for training deep neural networks. *arXiv preprint arXiv:1703.04379*, 2017.

[35] Dong Yin, Ashwin Pananjady, Max Lam, Dimitris Papailiopoulos, Kannan Ramchandran, and Peter Bartlett. Gradient diversity: a key ingredient for scalable distributed learning. In *AISTATS*, volume 84, pages 1998–2007, 2018.

Accepted Manuscript

[36] Cheng Zhang, Cengiz Öztireli, Stephan Mandt, and Giampiero Salvi. Active mini-batch sampling using repulsive point processes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5741–5748, 2019.

Accepted Manuscript

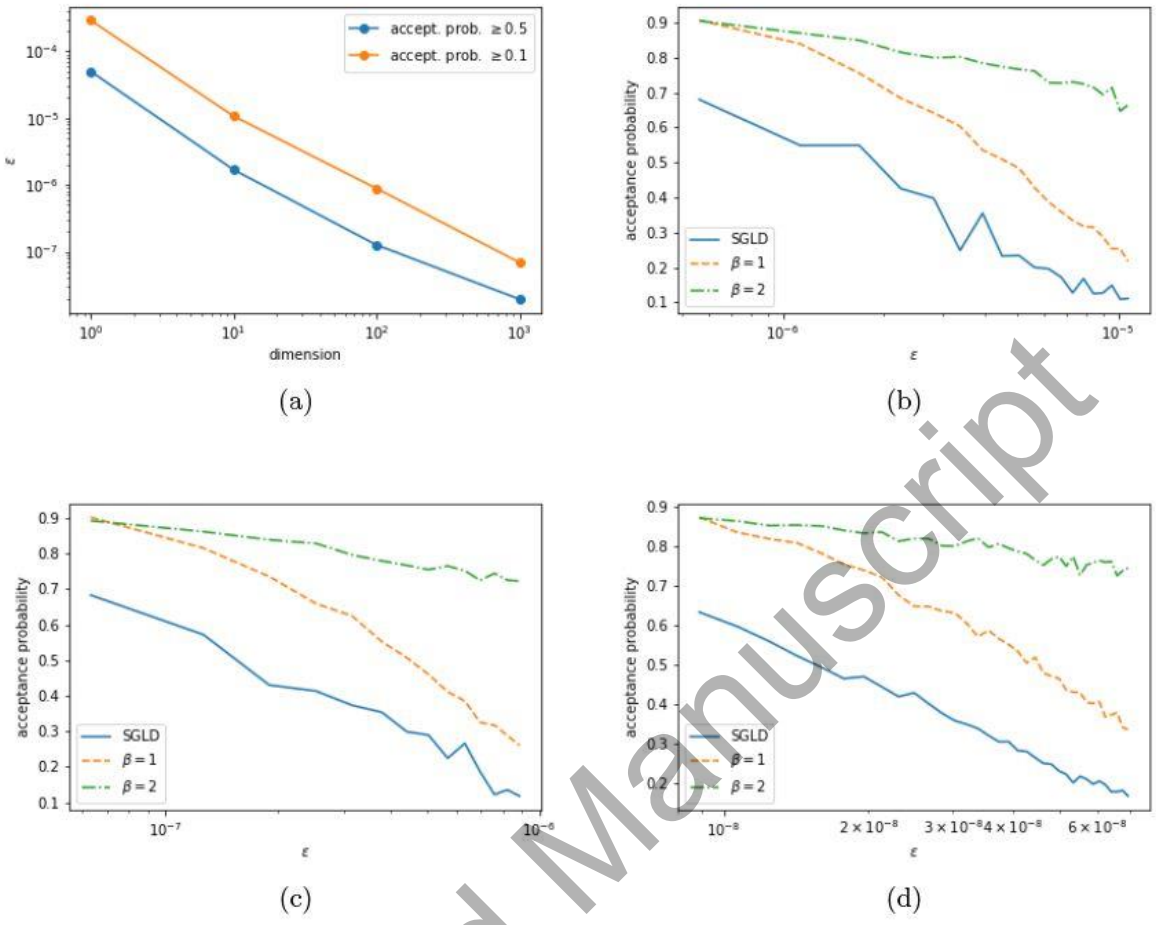

Fig. 1 (a) The largest ϵ allowed to achieve reasonable average acceptance probability on $N(0, I_d), d = 1, 10, 10^2, 10^3$. (b), (c), (d), the average acceptance probability for SGLD, RSGLD ($\beta = 1, 2$) for (b) d = 10, (c) $d = 10^2$, (d) $d = 10^3$.

Accepted Manuscript

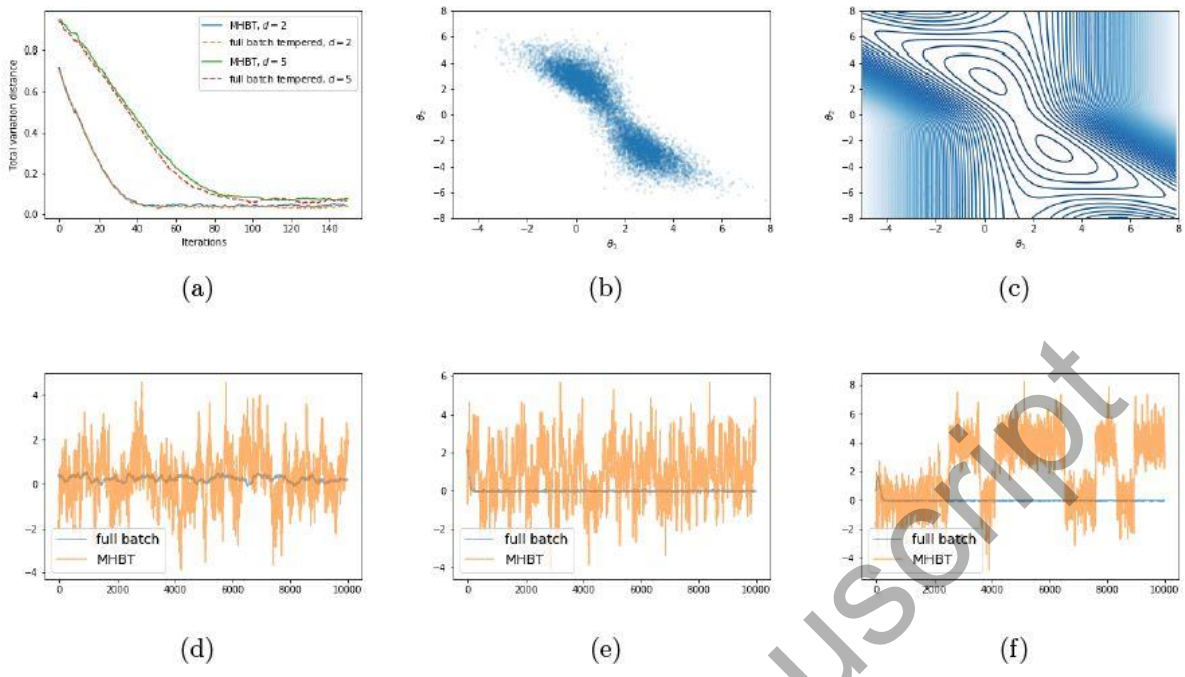

Fig. 2 (a) Total variation distance between the sampled distribution and true tempered posterior for *d*-dimensional Gaussian. *d* = 2, 5. (b), (c) Scatter plot of sampled θ values vs. contour plot of the tempered log posterior; (d), (e), (f) trajectories of MHBT and full batch MCMC for the 2-component Gaussian mixture model with fixed $\theta_1 = 0$, and $\theta_2 = 0.5, 2, 4$ respectively.

Accepted Manuscript

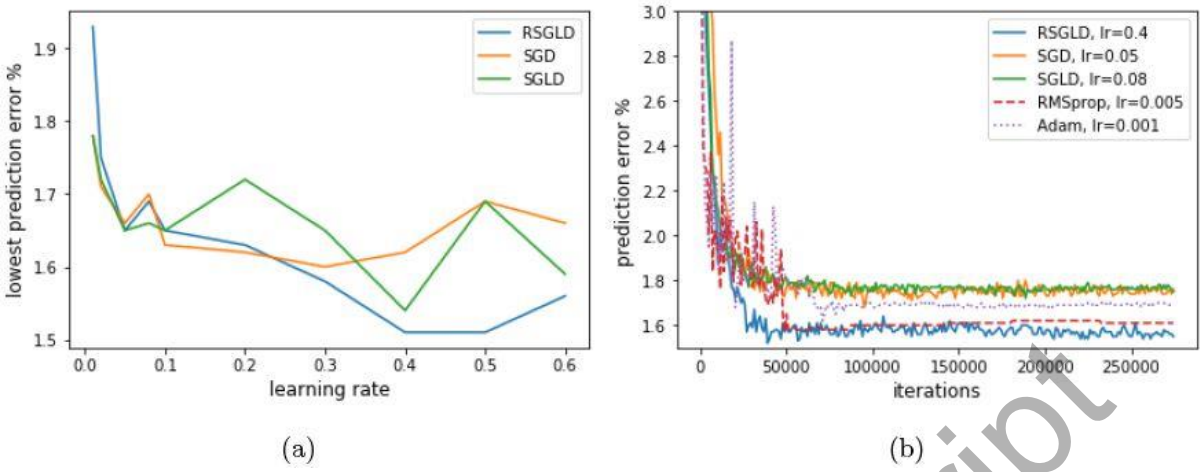

Fig. 3 (a) Lowest error rate in % achieved by the three methods out of 30 training rounds using various learning rates.(b) Examples of testing error trajectories using different training methods.

Accepted Manuscript

Table 1 MNIST top class prediction error (%) on the testing set using 30 training rounds for each learning rate. Each number is the median error with standard deviation in parentheses.

Top class prediction error (%) on the testing set					
ϵ	0.01	0.02	0.05	0.08	0.1
RSGLD	2.01 (0.03)	1.82 (0.03)	1.72 (0.03)	1.75 (0.03)	1.73 (0.04)
SGD	1.81 (0.02)	1.78 (0.02)	1.73 (0.03)	1.75 (0.03)	1.75 (0.06)
SGLD	1.81 (0.02)	1.78 (0.02)	1.73 (0.03)	1.72 (0.03)	1.75 (0.07)
	0.2	0.3	0.4	0.5	0.6
RSGLD	1.75 (0.13)	1.7 (0.18)	1.68 (0.33)	1.66 (11.4)	1.71 (26.9)
SGD	1.76 (16.1)	1.85 (42.1)	89.7 (44.4)	89.7 (33.4)	89.8 (24.0)
SGLD	1.8 (33.3)	1.84 (42.1)	89.7 (43.3)	89.9 (26.9)	89.8 (37.9)

Accepted Manuscript

Table 2 CIFAR-10 top class prediction error (%) on the testing set using 20 training rounds for each learning rate. The numbers shown are the median/lowest errors out of 20 rounds.

Top class prediction error (%) on the testing set				
ϵ	0.005	0.008	0.02	0.04
RSGLD	26.93/26.29	26.81/26.01	26.95/26.36	27.34/26.75
SGD	27.03/26.55	27.03/26.43	27.27/26.85	27.85/27.14
SGLD	27.00/26.60	26.88/26.19	27.31/26.70	27.85/27.08

Accepted Manuscript