Stochastic Quasi-Newton Langevin Monte Carlo

Umut Şimşekli, Roland Badeau, Taylan Cemgil, Gael Richard

Stochastic Quasi-Newton Langevin Monte Carlo

Umut Şimşekli[1] UMUT.SIMSEKLI@TELECOM-PARISTECH.FR
Roland Badeau[1] ROLAND.BADEAU@TELECOM-PARISTECH.FR
A. Taylan Cemgil[2] TAYLAN.CEMGIL@BOUN.EDU.TR
Gaël Richard[1] GAEL.RICHARD@TELECOM-PARISTECH.FR
1: LTCI, CNRS, Télécom ParisTech, Université Paris-Saclay, 75013, Paris, France
2: Department of Computer Engineering, Boğaziçi University, 34342, Bebek, İstanbul, Turkey

Abstract

Recently, Stochastic Gradient Markov Chain Monte Carlo (SG-MCMC) methods have been proposed for scaling up Monte Carlo computations to large data problems. Whilst these approaches have proven useful in many applications, vanilla SG-MCMC might suffer from poor mixing rates when random variables exhibit strong couplings under the target densities or big scale differences. In this study, we propose a novel SG-MCMC method that takes the local geometry into account by using ideas from Quasi-Newton optimization methods. These second order methods directly approximate the inverse Hessian by using a limited history of samples and their gradients. Our method uses dense approximations of the inverse Hessian while keeping the time and memory complexities linear with the dimension of the problem. We provide a formal theoretical analysis where we show that the proposed method is asymptotically unbiased and consistent with the posterior expectations. We illustrate the effectiveness of the approach on both synthetic and real datasets. Our experiments on two challenging applications show that our method achieves fast convergence rates similar to Riemannian approaches while at the same time having low computational requirements similar to diagonal preconditioning approaches.

1. Introduction

Markov Chain Monte Carlo (MCMC) methods are one of the most important family of algorithms in Bayesian machine learning. These methods lie in the core of various applications such as the estimation of Bayesian predictive densities and Bayesian model selection. They also provide important advantages over optimization-based point estimation methods, such as having better predictive accuracy and being more robust to over-fitting (Chen et al., 2014; Ahn et al., 2015). Despite their well-known benefits, during the last decade, conventional MCMC methods have lost their charm since they are often criticized as being computationally very demanding. Indeed, classical approaches based on batch Metropolis Hastings would require passing over the whole data set at each iteration and the acceptance-rejection test makes the methods even more impractical for modern data sets.

Recently, alternative approaches have been proposed to scale-up MCMC inference to large-scale regime. An important attempt was made by Welling & Teh (2011), where the authors combined the ideas from Langevin Monte Carlo (LMC) (Rossky et al., 1978; Roberts & Stramer, 2002; Neal, 2010) and stochastic gradient descent (SGD) (Robbins & Monro, 1951; Kushner & Yin, 2003), and developed a scalable MCMC framework referred to as stochastic gradient Langevin dynamics (SGLD). Unlike conventional batch MCMC methods, SGLD uses subsamples of the data per iteration similar to SGD. With this manner, SGLD is able to scale up to large datasets while at the same time being a valid MCMC method that forms a Markov chain asymptotically sampling from the target density. Several extensions of SGLD have been proposed (Ahn et al., 2012; Patterson & Teh, 2013; Ahn et al., 2014; Chen et al., 2014; Ding et al., 2014; Ma et al., 2015; Chen et al., 2015; Shang et al., 2015; Li et al., 2016), which are coined under the term Stochastic Gradient MCMC (SG-MCMC).

One criticism that is often directed at SGLD is that it suffers from poor mixing rates when the target densities exhibit strong couplings and scale differences across dimensions. This problem is caused by the fact that SGLD is based on an isotropic Langevin diffusion, which implicitly assumes that different components of the latent variable are uncor-

Proceedings of the 33^{rd} International Conference on Machine Learning, New York, NY, USA, 2016. JMLR: W&CP volume 48.

related and have the same scale, a situation which is hardly encountered in practical applications.

In order to be able to generate samples from non-isotropic target densities in an efficient manner, SGLD has been extended in several ways. The common theme in these methods is to incorporate the geometry of the target density to the sampling algorithm. Towards this direction, a first attempt was made by (Ahn et al., 2012), where the authors proposed Stochastic Gradient Fisher Scoring (SGFS), a preconditioning schema that incorporates the curvature via Fisher scoring. Later, Patterson & Teh (2013) presented Stochastic Gradient Riemannian Langevin Dynamics (SGRLD), where they defined the sampler on the Riemannian manifold by borrowing ideas from (Girolami & Calderhead, 2011). SGRLD incorporates the local curvature information through the expected Fisher information matrix (FIM), which defines a natural Riemannian metric tensor for probability distributions. SGRLD has shown significant performance improvements over SGLD; however, its computational complexity is very intensive since it requires storing and inverting huge matrices, computing Cholesky factors, and expensive matrix products for generating each sample. It further requires computing the third order derivatives and the expected FIM to be analytically tractable, which limit the applicability of the method to a narrow variety of statistical models (Calderhead & Sustik, 2012).

In a very recent study, Li et al. (2016) proposed Preconditioned SGLD (PSGLD), that aims to handle the scale differences in the target density by making use of an adaptive preconditioning matrix that is chosen to be diagonal for computational purposes. Even though this method is computationally less intensive than SGRLD, it still cannot handle highly correlated target densities since the preconditioning matrix is forced to be diagonal. Besides, in order to reduce the computational burden, the authors discard a correction term in practical applications, which introduces permanent bias that does not vanish asymptotically.

In this study, we propose Hessian Approximated MCMC (HAMCMC), a novel SG-MCMC method that computes the local curvature of the target density in an accurate yet computationally efficient manner. Our method is built up on similar ideas from the Riemannian approaches; however, instead of using the expected FIM, we consider the local Hessian of the negative log posterior, whose expectation coincides with the expected FIM. Instead of computing the local Hessian and inverting it for each sample, we borrow ideas from Quasi-Newton optimization methods and directly approximate the inverse Hessian by using a limited history of samples and their stochastic gradients. By this means, HAMCMC is (i) computationally more efficient than SGRLD since its time and memory complexities are linear with the dimension of the latent variable, (ii) it can be applied to a wide variety of statistical models without requiring the expected FIM to be analytically tractable, and (iii) it is more powerful than diagonal preconditioning approaches such as PSGLD, since it uses dense approximations of the inverse Hessian, hence is able to deal with correlations along with scale differences.

We provide rigorous theoretical analysis for HAMCMC, where we show that HAMCMC is asymptotically unbiased and consistent with posterior expectations. We evaluate HAMCMC on a synthetic and two challenging real datasets. In particular, we apply HAMCMC on two different matrix factorization problems and we evaluate its performance on a speech enhancement and a distributed link prediction application. Our experiments demonstrate that HAMCMC achieves fast convergence rates similar to SGRLD while at the same time having low computational requirements similar to SGLD and PSGLD.

We note that SGLD has also been extended by making use of higher order dynamics (Chen et al., 2014; Ding et al., 2014; Ma et al., 2015; Shang et al., 2015). These extensions are rather orthogonal to our contributions and HAMCMC can be easily extended in the same directions.

2. Preliminaries

Let θ be a random variable in $\mathbb{R}^D$. We aim to sample from the posterior distribution of θ, given as $p(\theta|x) \propto \exp(-U(\theta))$, where $U(\theta)$ is often called the *potential energy* and defined as $U(\theta) = -[\log p(x|\theta) + \log p(\theta)]$. Here, $x \triangleq \{x_n\}_{n=1}^N$ is a set of observed data points, and each $x_n \in \mathbb{R}^P$. We assume that the data instances are independent and identically distributed, so that we have:

$$U(\theta) = -[\log p(\theta) + \sum_{n=1}^{N} \log p(x_n|\theta)]. \tag{1}$$

We define an unbiased estimator of $U(\theta)$ as follows:

$$\tilde{U}(\theta) = -[\log p(\theta) + \frac{N}{N_\Omega} \sum_{n\in\Omega} \log p(x_n|\theta)] \tag{2}$$

where $\Omega \subset \{1, \dots, N\}$ is a random data subsample that is drawn with replacement, $N_\Omega = |\Omega|$ is the number of elements in Ω. We will occasionally use $\tilde{U}_\Omega(\theta)$ for referring to $\tilde{U}(\theta)$ when computed on a specific subsample Ω.

2.1. Stochastic Gradient Langevin Dynamics

By combining ideas from LMC and SGD, Welling & Teh (2011) presented SGLD that asymptotically generates a sample θ_t from the posterior distribution by iteratively applying the following update equation:

$$\theta_t = \theta_{t-1} - \epsilon_t \nabla \tilde{U}(\theta_{t-1}) + \eta_t \tag{3}$$

where ϵ_t is the step-size, and η_t is Gaussian noise: $\eta_t \sim \mathcal{N}(0, 2\epsilon_t I)$, and I stands for the identity matrix.

This method can be seen as the Euler discretization of the Langevin dynamics that is described by the following stochastic differential equation (SDE):

$$d\theta_t = -\nabla U(\theta_t)dt + \sqrt{2}dW_t \tag{4}$$

where W_t is the standard Brownian motion. In the simulations $\nabla U(\theta)$ is replaced by the stochastic gradient $\nabla \tilde{U}(\theta)$, which is an unbiased estimator of $\nabla U(\theta)$. Convergence analysis of SGLD has been studied in (Sato & Nakagawa, 2014; Teh et al., 2016). Since W_t is isotropic, SGLD often suffers from poor mixing rates when the target density is highly correlated or contains big scale differences.

2.2. The L-BFGS algorithm

In this study, we consider a *limited-memory* Quasi-Newton (QN) method, namely the L-BFGS algorithm (Nocedal & Wright, 2006) that iteratively applies the following equation in order to find the maximum a-posteriori estimate:

$$\theta_t = \theta_{t-1} - \epsilon_t H_t \nabla U(\theta_{t-1}) \tag{5}$$

where H_t is an approximation to the inverse Hessian at θ_{t-1}. The L-BFGS algorithm directly approximates the inverse of the Hessian by using the M most recent values of the past iterates. At each iteration, the inverse Hessian is approximated by applying the following recursion: (using $H_t = H_t^{M-1}$)

$$H_t^m = (I - \frac{s_\tau y_\tau^\top}{y_\tau^\top s_\tau}) H_t^{m-1} (I - \frac{y_\tau s_\tau^\top}{y_\tau^\top s_\tau}) + \frac{s_\tau s_\tau^\top}{y_\tau^\top s_\tau} \tag{6}$$

where $s_t \triangleq \theta_t - \theta_{t-1}$, $y_t \triangleq \nabla U(\theta_t) - \nabla U(\theta_{t-1})$, and $\tau = t - M + m$ and the initial approximation is chosen as $H_t^1 = \gamma I$ for some $\gamma > 0$. The matrix-vector product $H_t \nabla U(\theta_{t-1})$ is often implemented either by using the *two-loop recursion* (Nocedal & Wright, 2006) or by using the *compact form* (Byrd et al., 1994), which both have linear time complexity $\mathcal{O}(MD)$. Besides, since L-BFGS needs to store only the latest $M-1$ values of s_t and y_t, the space complexity is also $\mathcal{O}(MD)$, as opposed to classical QN methods that have quadratic memory complexity.

Surprisingly, QN methods have not attracted much attention from the MCMC literature. Zhang & Sutton (2011) combined classical Hamiltonian Monte Carlo with L-BFGS for achieving better mixing rates for small- and medium-sized problems. This method considers a batch scenario with full gradient computations which are then appended with costly Metropolis acceptance steps. Recently, Dahlin et al. (2015) developed a particle Metropolis-Hastings schema based on (Zhang & Sutton, 2011).

In this study, we consider a stochastic QN (SQN) framework for MCMC. The main difference in these approaches is that they use stochastic gradients $\nabla \tilde{U}(\theta)$ in the L-BFGS computations. These methods have been shown to provide better scalability properties than QN methods, and more favorable convergence properties than SGD-based approaches. However, it has been shown that straightforward extensions of L-BFGS to the stochastic case would fail in practice and therefore special attention is required (Schraudolph et al., 2007; Byrd et al., 2014).

3. Stochastic Quasi-Newton LMC

Recent studies have shown that Langevin and Hamiltonian Monte Carlo methods have strong connections with optimization techniques. Therefore, one might hope for developing stronger MCMC algorithms by borrowing ideas from the optimization literature (Qi & Minka, 2002; Zhang & Sutton, 2011; Bui-Thanh & Ghattas, 2012; Pereyra, 2013; Chaari et al., 2015; Bubeck et al., 2015; Li et al., 2016). However, incorporating ideas from optimization methods to an MCMC framework requires careful design and straightforward extensions often do not yield proper algorithms (Chen et al., 2014; Ma et al., 2015). In this section, we will first show that a naïve way of developing an SG-MCMC algorithm based on L-BFGS would not target the correct distribution. Afterwards, we will present our proposed algorithm HAMCMC and show that it targets the correct distribution and it is asymptotically consistent with posterior expectations.

3.1. A Naïve Algorithm

A direct way of using SQN ideas in an LMC framework would be achieved by considering H_t as a preconditioning matrix in an SGLD context (Welling & Teh, 2011; Ahn et al., 2012) by combining Eq. 3 and Eq. 5, which would yield the following update equation:

$$\theta_t = \theta_{t-1} - \epsilon_t H_t(\theta_{t-M:t-1}) \nabla \tilde{U}(\theta_{t-1}) + \eta_t \tag{7}$$

where $H_t(\cdot)$ is the approximate inverse Hessian computed via L-BFGS, M denotes the memory size in L-BFGS and $\eta_t \sim \mathcal{N}\big(0, 2\epsilon_t H_t(\theta_{t-M:t-1})\big)$. Here, we use a slightly different notation and denote the L-BFGS approximation via $H_t(\theta_{t-M:t-1})$ instead of H_t in order to explicitly illustrate the samples that are used in the L-BFGS calculations.

Despite the fact that such an approach would introduce several challenges, which will be described in the next section, for now let us assume computing Eq. 7 is feasible. Even so, this approach does not result in a proper MCMC schema.

Proposition 1. The Markov chain described in Eq. 7 does not target the posterior distribution $p(\theta|x)$ unless $\sum_{j=1}^{D} \frac{\partial}{\partial \theta_j} H_{ij}(\theta) = 0$ for all $i \in \{1, \dots, D\}$.

Proof. Eq. 7 can be formulated as a discretization of a continuous-time diffusion that is given as follows:

$$d\theta_t = -H(\theta_t)\nabla U(\theta_t)dt + \sqrt{2H(\theta_t)}dW_t \tag{8}$$

where $\nabla U(\theta_t)$ is approximated by $\nabla \tilde{U}(\theta_t)$ in simulations. Theorems 1 and 2 of (Ma et al., 2015) together show that SDEs with state dependent volatilities leave the posterior distribution invariant only if the drift term contains an additional 'correction' term, $\Gamma(\theta_t)$, defined as follows:[1]

$$\Gamma_i(\theta) = \sum_{j=1}^{D} \frac{\partial}{\partial \theta_j} H_{ij}(\theta). \tag{9}$$

Due to the hypothesis, Eq. 8 clearly does not target the posterior distribution, therefore Eq. 7 does not target the posterior distribution either. □

In fact, this result is neither surprising nor new. It has been shown that in order to ensure targeting the correct distribution, we should instead consider the following SDE (Girolami & Calderhead, 2011; Xifara et al., 2014; Patterson & Teh, 2013; Ma et al., 2015):

$$d\theta_t = -[H(\theta_t)\nabla U(\theta_t) + \Gamma(\theta_t)]dt + \sqrt{2H(\theta_t)}dW_t$$

where $\Gamma(\theta_t)$ must be taken into account when generating the samples. If we choose $H(\theta_t) = \mathbb{E}[\nabla^2 U(\theta_t)]^{-1}$ (i.e., the inverse of the expected FIM), discretize this SDE with the Euler scheme, and approximate $\nabla U(\theta_t)$ with $\nabla \tilde{U}(\theta_t)$, we obtain SGRLD. Similarly, we obtain PSGLD if $H(\theta_t)$ has the following form:

$$H(\theta_t) = \text{diag}(1/(\lambda + \sqrt{v(\theta_t)}))$$
$$v(\theta_t) = \alpha v(\theta_{t-1}) + (1-\alpha)\bar{g}_{\Omega_{t-1}}(\theta_{t-1}) \circ \bar{g}_{\Omega_{t-1}}(\theta_{t-1})$$

where $\bar{g}_\Omega(\theta) \triangleq (1/N_\Omega)\sum_{n\in\Omega} \nabla \log p(x_n|\theta)$, the symbols $\cdot/\cdot$ and $\cdot \circ \cdot$ respectively denote element-wise division and multiplication, and $\alpha \in [0,1]$ and $\lambda \in \mathbb{R}_+$ are the parameters to be tuned.

The correction term $\Gamma(\theta_t)$ does not vanish except for a very limited variety of models and computing this term requires the computation of the second derivatives of $\tilde{U}(\theta_t)$ in our case (whereas it requires computing the third derivatives in the Riemannian methods). This conflicts with our motivation of using QN methods for avoiding the Hessian computations in the first place.

In classical Metropolis-Hastings settings, where Langevin dynamics is only used in the proposal distribution, the correction term $\Gamma(\theta_t)$ can be discarded without violating the convergence properties thanks to the acceptance-rejection step (Qi & Minka, 2002; Girolami & Calderhead, 2011; Zhang & Sutton, 2011; Calderhead & Sustik, 2012). However, such an approach would result in slower convergence (Girolami & Calderhead, 2011). On the other hand, in SG-MCMC settings which do not include an acceptance-rejection step, this term can be problematic and should be handled with special attention.

In (Ahn et al., 2012), the authors discarded the correction term in a heuristic manner. Recently, Li et al. (2016) showed that discarding the correction term induces permanent bias which deprives the methods of their asymptotic consistency and unbiasedness. In the sequel, we will show that the computation of $\Gamma(\theta_t)$ can be avoided without inducing permanent bias by using a special construction that exploits the limited memory structure of L-BFGS.

3.2. Hessian Approximated MCMC

In this section, we present our proposed method HAMCMC. HAMCMC applies the following update rules for generating samples from the posterior distribution:

$$\theta_t = \theta_{t-M} - \epsilon_t H_t\Big(\theta^{\neg(t-M)}_{t-2M+1:t-1}\Big)\nabla \tilde{U}(\theta_{t-M}) + \eta_t \tag{10}$$

where $\theta^{\neg c}_{a:b} \equiv (\theta_{a:b} \setminus \theta_c)$, $\eta_t \sim \mathcal{N}\big(0, 2\epsilon_t H_t(\cdot)\big)$, and we assume that the initial $2M+1$ samples, i.e. $\theta_0, \ldots, \theta_{2M}$ are already provided.

As opposed to the naïve approach, HAMCMC generates the sample θ_t based on θ_{t-M} and it uses a history samples $\theta_{t-2M+1}, \ldots, \theta_{t-M-1}, \theta_{t-M+1}, \ldots, \theta_{t-1}$ in the L-BFGS computations. Here, we define the displacements and the gradient differences in a slightly different manner from usual L-BFGS, given as follows: $s_t = \theta_t - \theta_{t-M}$ and $y_t = \nabla \tilde{U}(\theta_t) - \nabla \tilde{U}(\theta_{t-M})$. Therefore, M must be chosen at least 2 or higher. HAMCMC requires to store only the latest M samples and the latest $M-1$ memory components, i.e. $\theta_{t-M:t-1}$, $s_{t-M+1:t-1}$, and $y_{t-M+1:t-1}$, resulting in linear memory complexity.

An important property of HAMCMC is that the correction term $\Gamma(\theta)$ vanishes due to the construction of our algorithm, which will be formally proven in the next section. Informally, since H_t is independent of θ_{t-M} conditioned on $\theta^{\neg(t-M)}_{t-2M+1:t-1}$, the change in θ_{t-M} does not affect H_t and therefore all the partial derivatives in Eq. 9 become zero. This property saves us from expensive higher-order derivative computations or inducing permanent bias due to the negligence of the correction term. On the other hand, geometrically, our approach corresponds to choosing a flat Riemannian manifold specified by the metric tensor H_t, whose curvature is constant (i.e. $\Gamma(\theta) = 0$).

We encounter several challenges due to the usage of stochastic gradients that are computed on random sub-

[1] The correction term presented in (Girolami & Calderhead, 2011) and (Patterson & Teh, 2013) has a different form than Eq. 9. Xifara et al. (2014) later showed that this term corresponded to a non-Lebesgue measure and Eq. 9 should be used instead for the Lebesgue measure.

Algorithm 1: Hessian Approximated MCMC

input: M, γ, λ, N_Ω, $\theta_0, \dots, \theta_{2M}$
for $t = 2M+1, \cdots, T$ **do**
  Draw $\Omega_t \subset \{1, \dots, N\}$
  Generate $z_t \sim \mathcal{N}(0, I)$
  // L-BFGS computations
  $\xi_t = H_t\big(\theta^{\neg(t-M)}_{t-2M+1:t-1}\big)\nabla\tilde{U}_{\Omega_t}(\theta_{t-M})$ (Eq. 6)
  $\eta_t = S_t\big(\theta^{\neg(t-M)}_{t-2M+1:t-1}\big)z_t$ (Eq. 11)
  // Generate the new sample
  $\theta_t = \theta_{t-M} - \epsilon_t\xi_t + \sqrt{2\epsilon_t}\eta_t$ (Eq. 10)
  // Update L-BFGS memory
  $s_t = \theta_t - \theta_{t-M}$, $y_t = \nabla\tilde{U}_{\Omega_t}(\theta_t) - \nabla\tilde{U}_{\Omega_t}(\theta_{t-M}) + \lambda s_t$

sets of the data. The first challenge is the *data consistency* (Schraudolph et al., 2007; Byrd et al., 2014). Since we draw different data subsamples Ω_t at different epochs, the gradient differences y_t would be simply 'inconsistent'. Handling this problem in incremental QN settings (i.e., the data subsamples are selected via a deterministic mechanism, rather than being chosen at random) is not as challenging, however, the randomness in the stochastic setting prevents simple remedies such as computing the differences in a cyclic fashion. Therefore, in order to ensure consistent gradient differences, we slightly increase the computational needs of HAMCMC and perform an additional stochastic gradient computation at the end of each iteration, i.e. $y_t = \nabla\tilde{U}_{\Omega_t}(\theta_t) - \nabla\tilde{U}_{\Omega_t}(\theta_{t-M})$ where both gradients are evaluated on the same subsample Ω_t.

In order to have a valid sampler, we are required to have positive definite $H_t(\cdot)$. However, even if $U(\theta)$ was strongly convex, due to data subsampling L-BFGS is no longer guaranteed to produce positive definite approximations. One way to address this problem is to use variance reduction techniques as presented in (Byrd et al., 2014); however, such an approach would introduce further computational burden. In this study, we follow a simpler approach and address this problem by approximating the inverse Hessian in a trust region that is parametrized by λ, i.e. we use L-BFGS to approximate $(\nabla^2 U(\theta) + \lambda I)^{-1}$. For large enough λ this approach will ensure positive definiteness. The L-BFGS updates for this problem can be obtained by simply modifying the gradient differences as: $y_t \leftarrow y_t + \lambda s_t$ (Schraudolph et al., 2007).

The final challenge is to generate η_t whose covariance is a scaled version of the L-BFGS output. This requires to compute the square root of H_t, i.e. $S_t S_t^\top = H_t$, so that we can generate η_t as $\sqrt{2\epsilon_t}S_t z_t$, where $z_t \sim \mathcal{N}(0, I)$. Fortunately, we can directly compute the product $S_t z_t$ within the L-BFGS framework by using the following recursion (Brodlie et al., 1973; Zhang & Sutton, 2011):

$$
\begin{aligned}
&H_t^m = S_t^m (S_t^m)^\top,\ S_t^m = (I - p_\tau q_\tau^\top) S_t^{m-1} \\
&B_t^m = C_t^m (C_t^m)^\top,\ C_t^m = (I - u_\tau v_\tau^\top) C_t^{m-1} \\
&v_\tau = \frac{s_\tau}{s_\tau^\top B_t^{m-1} s_\tau},\quad u_\tau = \sqrt{\frac{s_\tau^\top B_t^{m-1} s_\tau}{s_\tau^\top y_\tau}}\, y_\tau + B_t^{m-1} s_\tau, \\
&p_\tau = \frac{s_\tau}{s_\tau^\top y_\tau},\qquad q_\tau = \sqrt{\frac{s_\tau^\top y_\tau}{s_\tau^\top B_t^{m-1} y_\tau}}\, B_t^{m-1} s_\tau - y_\tau
\end{aligned} \tag{11}
$$

where $S_t \equiv S_t^{M-1}$ and $B_t^m = (H_t^m)^{-1}$. By this approach, η_t can be computed in $\mathcal{O}(M^2 D)$ time, which is still linear in D. We illustrate HAMCMC in Algorithm 1.

Note that large M would result in a large gap between two consecutive samples and therefore might require larger λ for ensuring positive definite L-BFGS approximations. Such a circumstance would be undesired since H_t would get closer to the identity matrix and therefore result in HAMCMC being similar to SGLD. In our computational studies, we have observed that choosing M between 2 and 5 is often adequate in several applications.

3.3. Convergence Analysis

We are interested in approximating the posterior expectations by using sample averages:

$$
\bar{h} \triangleq \int_{\mathbb{R}^D} h(\theta)\pi(d\theta) \approx \hat{h} \triangleq \frac{1}{W_T}\sum_{t=1}^{T} \epsilon_t h(\theta_t), \tag{12}
$$

where $\pi(\theta) \triangleq p(\theta|x)$, $W_T \triangleq \sum_{t=1}^T \epsilon_t$, and θ_t denotes the samples generated by HAMCMC. In this section we will analyze the asymptotic properties of this estimator. We consider a theoretical framework similar to the one presented in (Chen et al., 2015). In particular, we are interested in analyzing the bias $\big|\mathbb{E}[\hat{h}] - \bar{h}\big|$ and the mean squared-error $\mathbb{E}\big[(\hat{h} - \bar{h})^2\big]$ of our estimator.

A useful way of analyzing dynamics-based MCMC approaches is to first analyze the underlying continuous dynamics, then consider the MCMC schema as a discrete-time approximation (Roberts & Stramer, 2002; Sato & Nakagawa, 2014; Chen et al., 2015). However, this approach is not directly applicable in our case. Inspired by (Roberts & Gilks, 1994) and (Zhang & Sutton, 2011), we convert Eq. 10 to a first order Markov chain that is defined on the product space $\mathbb{R}^{D(2M-1)}$ such that $\Theta_t \equiv \{\theta_{t-2M+2}, \dots, \theta_t\}$. In this way, we can show that HAMCMC uses a transition kernel which modifies only one element of Θ_t per iteration and rearranges the samples that will be used in the L-BFGS computations. Therefore, the overall HAMCMC kernel can be interpreted as a composition of M different preconditioned SGLD kernels whose

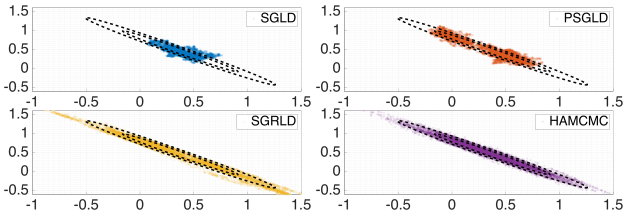

Figure 1. The illustration of the samples generated by different SG-MCMC methods. The contours of the true posterior distribution is shown with dashed lines. We set $M = 2$ for HAMCMC.

volatilities are independent of their current states. By considering this property and assuming certain conditions that are provided in the Supplement hold, we can bound the bias and MSE as shown in Theorem 1.

Theorem 1. Assume the number of iterations is chosen as $T = KM$ where $K \in \mathbb{N}_+$ and $\{\theta_t\}_{t=1}^T$ are obtained by HAMCMC (Eq. 10). Define $L_K \triangleq \sum_{k=1}^K \epsilon_{kM}$, $Y_K \triangleq \sum_{k=1}^K \epsilon_{kM}^2$, and the operator $\Delta V_t \triangleq (\nabla \tilde{U}(\theta_t) - \nabla U(\theta_t))^\top H_t \nabla$. Then, we can bound the bias and MSE of HAMCMC as follows:

(a) $\left|\mathbb{E}[\hat{h}] - \bar{h}\right| = \mathcal{O}\left(\frac{1}{L_K} + \frac{Y_K}{L_K}\right)$

(b) $\mathbb{E}\left[(\hat{h} - \bar{h})^2\right] = \mathcal{O}\left(\sum_{k=1}^K \frac{\epsilon_{kM}^2}{L_K^2} \mathbb{E}\|\Delta V_{k^\star}\|^2 + \frac{1}{L_K} + \frac{Y_K^2}{L_K^2}\right)$

where h is smooth and $\Delta V_{k^\star} = \Delta V_{m_k^* + kM}$, where $m_k^* = \arg\max_{1 \le m \le M} \mathbb{E}\|\Delta V_{m+kM}\|^2$, and $\|\Delta V_t\|$ denotes the operator norm.

The proof is given in the Supplement. The theorem implies that as T goes to infinity, the bias and the MSE of the estimator vanish. Therefore, HAMCMC is asymptotically unbiased and consistent.

4. Experiments

4.1. Linear Gaussian Model

We conduct our first set of experiments on synthetic data where we consider a rather simple Gaussian model whose posterior distribution is analytically available. The model is given as follows:

$$\theta \sim \mathcal{N}(0, I), \quad x_n|\theta \sim \mathcal{N}(a_n^\top \theta, \sigma_x^2), \tag{13}$$

for all n. Here, we assume $\{a_n\}_{n=1}^N$ and σ_x^2 are known and we aim to draw samples from the posterior distribution $p(\theta|x)$. We will compare the performance of HAMCMC against SGLD, PSGLD, and SGRLD. In these experiments, we determine the step size as $\epsilon_t = (a_\epsilon/t)^{0.51}$. In all our experiments, for each method, we have tried several values for the hyper-parameters with the same rigor and we report the best results. We also report all of the hyper-parameters used in the experiments (including Sections 4.2-4.3) in the Supplement. These experiments are conducted on a standard laptop computer with 2.5GHz Quad-core Intel Core i7 CPU and all the methods are implemented in Matlab.

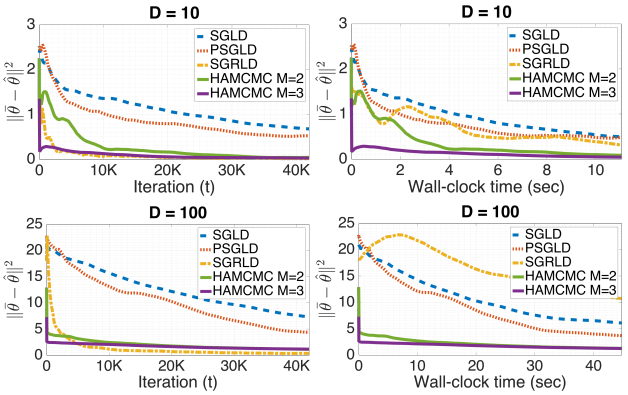

Figure 2. The error between the true posterior mean $\bar{\theta}$ and the Monte Carlo estimates $\hat{\theta}$ for $D = 10$ and $D = 100$. On the left column, we show the error versus iterations and on the right column we show the error versus wall-clock time.

In the first experiment, we first generate the true θ and the observations x by using the generative model given in Eq. 13. We set $D = 2$ for visualization purposes and $\sigma_x^2 = 10$, then we randomly generate the vectors $\{a_n\}_n$ in such a way that the posterior distribution will be correlated. Then we generate $T = 20000$ samples by using the SG-MCMC methods, where the size of the data subsamples is selected as $N_\Omega = N/100$. We discard the first half of the samples as burn-in. Figure 1 shows the typical results of this experiment. We can clearly observe that SGLD cannot explore the posterior efficiently and gets stuck around the mode of the distribution. PSGLD captures the shape of the distribution in a more accurate way than SGLD since it is able to take the scale differences into account; however, it still cannot explore lower probability areas efficiently. Besides, we can see that SGRLD performs much better than SGLD and PSGLD. This is due to the fact that it uses the inverse expected FIM of the *full* problem for generating each sample, where this matrix is simply $(\sum_n a_n a_n^\top/\sigma_x^2 + I)^{-1}$ for this model. Therefore, SGRLD requires much heavier computations as it needs to store a $D \times D$ matrix, computes the Cholesky factors of this matrix, and performs matrix-vector products at each iteration. Furthermore, in more realistic probabilistic models, the expected FIM almost always depends on θ; hence, the computation burden of SGRLD is further increased by the computation of the inverse of the $D \times D$ matrix at each iteration. On the other hand, Fig. 1 shows that HAMCMC takes the best of both worlds: it is able to achieve the quality of SGRLD since it makes use of dense approximations to the inverse Hessian and it has low computational requirements similar to SGLD and PSGLD. This observation becomes more clear in our next experiment.

In our second experiment, we again generate the true θ and the observations by using Eq. 13. Then, we approximate the posterior mean $\bar{\theta} = \int \theta p(\theta|x)d\theta$ by using the aforementioned MCMC methods. For comparison, we monitor $\|\bar{\theta} - \hat{\theta}\|^2$ where $\hat{\theta}$ is the estimate that is obtained from MCMC. Figure 2 shows the results of this experiment for $D = 10$ and $D = 100$. In the left column, we show the error as a function of iterations. These results show that the convergence rate of SGRLD (with respect to iterations) is much faster than SGLD and PSGLD. As expected, HAMCMC yields better results than SGLD and PSGLD even in the simplest configuration ($M = 2$), and it performs very similarly to SGRLD as we set $M = 3$. The performance difference becomes more prominent as we increase D.

An important advantage of HAMCMC is revealed when we take into account the total running time of these methods. These results are depicted in the right column of Fig. 2, where we show the error as a function of wall-clock time [2]. We can observe that, since SGLD, PSGLD, and HAMCMC have similar computational complexities, the shapes of their corresponding plots do not differ significantly. However, SGRLD appears to be much slower than the other methods as we increase D.

4.2. Alpha-Stable Matrix Factorization

In our second set of experiments, we consider a recently proposed probabilistic matrix factorization model, namely alpha-stable matrix factorization (αMF), that is given as follows (Şimşekli et al., 2015):

$$W_{ik} \sim \mathcal{GG}(a_w, b_w, -2/\alpha), \quad H_{kj} \sim \mathcal{GG}(a_h, b_h, -2/\alpha)$$
$$X_{ij}|W, H \sim \mathcal{S}\alpha\mathcal{S}_c\big([\textstyle\sum_k W_{ik}H_{kj}]^{1/\alpha}\big) \qquad (14)$$

where $X \in \mathbb{C}^{I\times J}$ is the observed matrix with *complex* entries, and $W \in \mathbb{R}_+^{I\times K}$ and $H \in \mathbb{R}_+^{K\times J}$ are the latent factors. Here $\mathcal{GG}$ denotes the generalized gamma distribution (Stacy, 1962) and $\mathcal{S}\alpha\mathcal{S}_c$ denotes the complex symmetric α-stable distribution (Samoradnitsky & Taqqu, 1994). Stable distributions are heavy-tailed distributions and they are the limiting distributions in the generalized central limit theorem. They are often characterized by four parameters: $\mathcal{S}(\alpha, \beta, \sigma, \mu)$, where $\alpha \in (0, 2]$ is the characteristic exponent, $\beta \in [-1, 1]$ is the skewness parameter, $\sigma \in (0, \infty)$ is the dispersion parameter, and $\mu \in (-\infty, \infty)$ is the location parameter. The distribution is called symmetric ($\mathcal{S}\alpha\mathcal{S}$) if $\beta = 0$. The location parameter is also chosen as 0 in αMF.

The probability density function of the stable distributions cannot be written in closed-form except for certain special cases; however it is easy to draw samples from them. Therefore, MCMC has become the de facto inference tool for α-stable models like αMF (Godsill & Kuruoglu, 1999). By using data augmentation and the product property of stable distributions, Şimşekli et al. (2015) proposed a partially collapsed Gibbs sampler for making inference in this model. The performance of this approach was then illustrated on a speech enhancement problem.

[2]The inverse of the FIM for this model can be precomputed. However, this will not be the case in more general scenarios. Therefore, for fair comparison we perform the matrix inversion and Cholesky decomposition operations at each iteration.

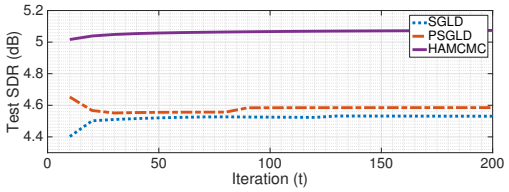

Figure 3. The performance of HAMCMC on a speech denoising application in terms of SDR (higher is better).

In this section, we derive a new inference algorithm for αMF that is based on SG-MCMC. By using the product property of stable models (Kuruoglu, 1999), we express the model as conditionally Gaussian, given as follows:

$$W_{ik} \sim \mathcal{GG}(a_w, b_w, -2/\alpha), \quad H_{kj} \sim \mathcal{GG}(a_h, b_h, -2/\alpha)$$
$$\Phi_{ij} \sim \mathcal{S}\big(\alpha/2, 1, 2\big(\cos(\pi\alpha)/4\big)^{2/\alpha}, 0\big)$$
$$X_{ij}|W, H, \Phi \sim \mathcal{N}_c\big(0, \Phi_{ij}[\textstyle\sum_k W_{ik}H_{kj}]^{2/\alpha}\big), \qquad (15)$$

where $\mathcal{N}_c$ denotes the complex Gaussian distribution. Here, by assuming α is already provided, we propose a Metropolis and SG-MCMC within Gibbs approach, where we generate samples from the full conditional distributions of the latent variables in such a way that we use a Metropolis step for sampling Φ where the proposal is the prior and we use SG-MCMC for sampling W and H from their full conditionals. Since all the elements of W and H must be non-negative, we apply mirroring at the end of each iteration where we replace negative elements with their absolute values (Patterson & Teh, 2013).

We evaluate HAMCMC on the same speech enhancement application described in (Şimşekli et al., 2015) by using the same settings. The aim here is to recover clean speech signals that are corrupted by different types of real-life noise signals (Hu & Loizou, 2007). We use the Matlab code that can be downloaded from the authors' websites in order to generate the same experimental setting. We first train W on a clean speech corpus, then we fix W at denoising and sample the rest of the variables. For each test signal (out of 80) we have $I = 256$ and $J = 140$. The rank is set $K = 105$ where the first 100 columns of W is reserved for clean speech and for speech $\alpha = 1.2$ and for noise $\alpha = 1.89$. Note that the model used for denoising is slightly different from Eq. 14 and we refer the readers to the original paper for further details of the experimental setup.

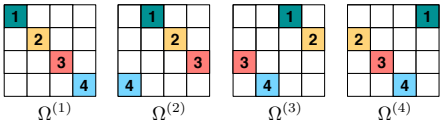

Figure 4. The illustration of the data partitioning that is used in the distributed matrix factorization experiments. The numbers in the blocks denote the computer that stores the corresponding block.

In this experiment, we compare HAMCMC against SGLD and PSGLD. SGRLD cannot be applied to this model since the expected FIM is not tractable. For each method, we set $N_\Omega = IJ/10$ and we generate 2500 samples where we compute the Monte Carlo averages by using the last 200 samples. For HAMCMC, we set $M = 5$. Fig. 3 shows the denoising results on the test set in terms of signal-to-distortion ratio (SDR) when the mixture signal-to-noise ratio is 5dB. The results show that SGLD and PSGLD perform similarly, where the quality of the outputs of PSGLD is slightly higher than the ones obtained from SGLD. We can observe that HAMCMC provides a significant performance improvement over SGLD and PSGLD, thanks to the usage of local curvature information.

4.3. Distributed Matrix Factorization

In our final set of experiments, we evaluate HAMCMC on a distributed matrix factorization problem, where we consider the following probabilistic model: $W_{ik} \sim \mathcal{N}(0, \sigma_w^2)$, $H_{kj} \sim \mathcal{N}(0, \sigma_h^2)$, $X_{ij}|\cdot \sim \mathcal{N}\big(\sum_k W_{ik}H_{kj}, \sigma_x^2\big)$. Here, $W \in \mathbb{R}^{I\times K}$, $H \in \mathbb{R}^{K\times J}$, and $X \in \mathbb{R}^{I\times J}$. This model is similar to (Salakhutdinov & Mnih, 2008) and is often used in distributed matrix factorization problems (Gemulla et al., 2011; Şimşekli et al., 2015b).

Inspired by Distributed SGLD (DSGLD) for matrix factorization (Ahn et al., 2015) and Parallel SGLD (Şimşekli et al., 2015a), we partition X into disjoint subsets where each subset is further partitioned into disjoint blocks. This schema is illustrated in Fig. 4, where X is partitioned into 4 disjoint subsets $\Omega^{(1)}, \dots, \Omega^{(4)}$ and each subset contains 4 different blocks that will be distributed among different computation nodes. By using this approach, we extend PSGLD and HAMCMC to the distributed setting similarly to DSGLD, where at each iteration the data subsample Ω_t is selected as $\Omega^{(i)}$ with probability $|\Omega^{(i)}|/\sum_j |\Omega^{(j)}|$. At the end of each iteration the algorithms transfer a small block of H to a neighboring node depending on Ω_{t+1}. PSGLD and HAMCMC further need to communicate other variables that are required for computing H_t, where the communication complexity is still linear with D. Finally, HAMCMC requires to compute six vector dot products at each iteration that involves all the computation nodes, such as $s_t^\top y_t$. These computations can be easily implemented by using a *reduce* operation which has a communication complexity of $\mathcal{O}(M^2)$. By using this approach, the methods have the same theoretical properties except that the stochastic gradients would be expected to have larger variances.

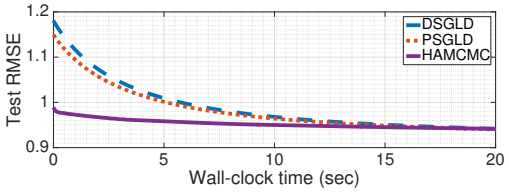

Figure 5. The performance of HAMCMC on a distributed matrix factorization problem.

For this experiment, we have implemented all the algorithms in C by using a low-level message passing protocol, namely the OpenMPI library. In order to keep the implementation simple, for HAMCMC we set $M = 2$. We conduct our experiments on a small-sized cluster with 4 interconnected computers each of them with 4 Intel Xeon 2.93GHz CPUs and 192 GB of memory.

We compare distributed HAMCMC against DSGLD and distributed PSGLD on a large movie ratings dataset, namely the MovieLens 1M (`grouplens.org`). This dataset contains about 1 million ratings applied to $I = 3883$ movies by $J = 6040$ users, resulting in a sparse observed matrix X with 4.3% non-zero entries. We randomly select 10% of the data as the test set and partition the rest of the data as illustrated in Fig. 4. The rank of the factorization is chosen as $K = 10$. We set $\sigma_w^2 = \sigma_h^2 = \sigma_x^2 = 1$. In these experiments, we use a constant step size for each method and discard the first 50 samples as burn-in. Note that for constant step size, we no longer have asymptotic unbiasedness; however, the bias and MSE are still bounded.

Fig. 5 shows the comparison of the algorithms in terms of the root mean squared-error (RMSE) that are obtained on the test set. We can observe that DSGLD and PSGLD have similar converges rates. On the other hand, HAMCMC converges faster than these methods while having almost the same computational requirements.

5. Conclusion

We presented HAMCMC, a novel SG-MCMC algorithm that achieves high convergence rates by taking the local geometry into account via the local Hessian that is efficiently computed by the L-BFGS algorithm. HAMCMC is both efficient and powerful since it uses dense approximations of the inverse Hessian while having linear time and memory complexity. We provided formal theoretical analysis and showed that HAMCMC is asymptotically consistent. Our experiments showed that HAMCMC achieves better convergence rates than the state-of-the-art while keeping the computational cost almost the same. As a next step, we will explore the use of HAMCMC in model selection applications (Şimşekli et al., 2016).

Acknowledgments

The authors would like to thank to Ş. İlker Birbil and Figen Öztoprak for fruitful discussions on Quasi-Newton methods. The authors would also like to thank to Charles Sutton for his explanations on the Ensemble Chain Adaptation framework. This work is partly supported by the French National Research Agency (ANR) as a part of the EDISON 3D project (ANR-13-CORD-0008-02). A. Taylan Cemgil is supported by TÜBİTAK 113M492 (Pavera) and Boğaziçi University BAP 10360-15A01P1.

References

Ahn, S., Korattikara, A., and Welling, M. Bayesian posterior sampling via stochastic gradient Fisher scoring. In *International Conference on Machine Learning*, 2012.

Ahn, S., Shahbaba, B., and Welling, M. Distributed stochastic gradient MCMC. In *International Conference on Machine Learning*, 2014.

Ahn, S., Korattikara, A., Liu, N., Rajan, S., and Welling, M. Large-scale distributed Bayesian matrix factorization using stochastic gradient MCMC. In *International Conference on Knowledge Discovery and Data Mining*, August 2015.

Brodlie, K. W., Gourlay, A. R., and Greenstadt, J. Rank-one and rank-two corrections to positive definite matrices expressed in product form. *IMA J APPL MATH*, 11 (1):73–82, 1973.

Bubeck, S., Eldan, R., and Lehec, J. Finite-time analysis of projected Langevin Monte Carlo. In *Advances in Neural Information Processing Systems*, pp. 1243–1251, 2015.

Bui-Thanh, T. and Ghattas, O. A scaled stochastic Newton algorithm for Markov Chain Monte Carlo simulations. *SIAM Journal on Uncertainty Quantification*, 2012.

Byrd, R. H., Nocedal, J., and Schnabel, R. B. Representations of quasi-Newton matrices and their use in limited memory methods. *Mathematical Programming*, 63(1-3): 129–156, 1994.

Byrd, R. H., Hansen, S. L., Nocedal, J., and Singer, Y. A stochastic quasi-Newton method for large-scale optimization. *arXiv preprint arXiv:1401.7020*, 2014.

Calderhead, B. and Sustik, M. Sparse approximate manifolds for differential geometric MCMC. In *Advances in Neural Information Processing Systems*, pp. 2879–2887, 2012.

Chaari, L., Tourneret, J.Y., Chaux, C., and Batatia, H. A Hamiltonian Monte Carlo method for non-smooth energy sampling. *arXiv preprint arXiv:1401.3988*, 2015.

Chen, C., Ding, N., and Carin, L. On the convergence of stochastic gradient MCMC algorithms with high-order integrators. In *Advances in Neural Information Processing Systems*, pp. 2269–2277, 2015.

Chen, T., Fox, E. B., and Guestrin, C. Stochastic gradient Hamiltonian Monte Carlo. In *International Conference on Machine Learning*, 2014.

Şimşekli, U., Liutkus, A., and Cemgil, A. T. Alpha-stable matrix factorization. *IEEE Signal Processing Letters*, 22 (12):2289–2293, 2015.

Şimşekli, U., Badeau, R., Richard, G., and Cemgil, A. T. Stochastic thermodynamic integration: efficient Bayesian model selection via stochastic gradient MCMC. In *41st International Conference on Acoustics, Speech and Signal Processing*, 2016.

Dahlin, J., Lindsten, F., and Schön, T. B. Quasi-Newton particle Metropolis-Hastings. In *IFAC Symposium on System Identification*, 2015.

Ding, N., Fang, Y., Babbush, R., Chen, C., Skeel, R. D., and Neven, H. Bayesian sampling using stochastic gradient thermostats. In *Advances in Neural Information Processing Systems*, pp. 3203–3211, 2014.

Gemulla, R., Nijkamp, E., J., Haas. P., and Sismanis, Y. Large-scale matrix factorization with distributed stochastic gradient descent. In *ACM SIGKDD*, 2011.

Girolami, M. and Calderhead, B. Riemann manifold Langevin and Hamiltonian Monte Carlo methods. *J R Stat Soc Series B Stat Methodol*, 73(2):123–214, 2011.

Godsill, S. and Kuruoglu, E. E. Bayesian inference for time series with heavy-tailed symmetric α-stable noise processes. *Heavy Tails' 99, Applications of Heavy Tailed Distributions in Economics, Engineering and Statistics*, pp. 3–5, 1999.

Hu, Y. and Loizou, P. C. Subjective comparison and evaluation of speech enhancement algorithms. *Speech communication*, 49(7):588–601, 2007.

Kuruoglu, E. E. *Signal processing in α-stable noise environments: a least lp-norm approach*. PhD thesis, University of Cambridge, 1999.

Kushner, H. and Yin, G. *Stochastic Approximation and Recursive Algorithms and Applications*. Springer, New York, 2003.

Li, C., Chen, C., Carlson, D., and Carin, L. Preconditioned stochastic gradient Langevin dynamics for deep neural networks. In *AAAI Conference on Artificial Intelligence*, 2016.

Ma, Y. A., Chen, T., and Fox, E. A complete recipe for stochastic gradient MCMC. In *Advances in Neural Information Processing Systems*, pp. 2899–2907, 2015.

Neal, R. M. MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 54, 2010.

Nocedal, J. and Wright, S. J. *Numerical optimization*. Springer, Berlin, 2006.

Patterson, S. and Teh, Y. W. Stochastic gradient Riemannian Langevin dynamics on the probability simplex. In *Advances in Neural Information Processing Systems*, 2013.

Pereyra, M. Proximal Markov Chain Monte Carlo algorithms. *arXiv preprint arXiv:1306.0187*, 2013.

Qi, Y. and Minka, T. P. Hessian-based Markov Chain Monte Carlo algorithms. In *First Cape Cod Workshop on Monte Carlo Methods*, 2002.

Robbins, H. and Monro, S. A stochastic approximation method. *Ann. Math. Statist.*, 22(3):400–407, 1951.

Roberts, G. O. and Stramer, O. Langevin Diffusions and Metropolis-Hastings Algorithms. *Methodology and Computing in Applied Probability*, 4(4):337–357, December 2002. ISSN 13875841.

Roberts, G.O. and Gilks, W.R. Convergence of adaptive direction sampling. *Journal of Multivariate Analysis*, 49 (2):287 – 298, 1994.

Rossky, P. J., Doll, J. D., and Friedman, H. L. Brownian dynamics as smart Monte Carlo simulation. *The Journal of Chemical Physics*, 69(10):4628–4633, 1978.

Salakhutdinov, R. and Mnih, A. Bayesian probabilistic matrix factorization using Markov Chain Monte Carlo. In *International Conference on Machine learning*, pp. 880–887, 2008.

Samoradnitsky, G. and Taqqu, M. *Stable non-Gaussian random processes: stochastic models with infinite variance*, volume 1. CRC Press, 1994.

Sato, I. and Nakagawa, H. Approximation analysis of stochastic gradient Langevin dynamics by using Fokker-Planck equation and Ito process. In *International Conference on Machine Learning*, pp. 982–990, 2014.

Schraudolph, N. N., Yu, J., and Günter, S. A stochastic quasi-Newton method for online convex optimization. In *International Conference on Artificial Intelligence and Statistics*, pp. 436–443, 2007.

Shang, X., Zhu, Z., Leimkuhler, B., and Storkey, A. J. Covariance-controlled adaptive Langevin thermostat for large-scale Bayesian sampling. In *Advances in Neural Information Processing Systems*, pp. 37–45, 2015.

Şimşekli, U., Koptagel, H., Güldaş, H., Cemgil, A. T., Öztoprak, F., and Birbil, Ş. İ. Parallel stochastic gradient Markov Chain Monte Carlo for matrix factorisation models. *arXiv preprint arXiv:1506.01418*, 2015a.

Şimşekli, U., Koptagel, H., Öztoprak, F., Birbil, Ş. İ., and Cemgil, A. T. HAMSI: Distributed incremental optimization algorithm using quadratic approximations for partially separable problems. *arXiv preprint arXiv:1509.01698*, 2015b.

Stacy, E. W. A generalization of the gamma distribution. *Ann. Math. Statist.*, 33(3):1187–1192, 09 1962.

Teh, Y. W., Thiéry, A., and Vollmer, S. Consistency and fluctuations for stochastic gradient Langevin dynamics. *Journal of Machine Learning Research*, 17(7):1–33, 2016.

Welling, M. and Teh, Y. W. Bayesian learning via stochastic gradient Langevin dynamics. In *International Conference on Machine Learning*, pp. 681–688, 2011.

Xifara, T., Sherlock, C., Livingstone, S., Byrne, S., and Girolami, M. Langevin diffusions and the Metropolis-adjusted Langevin algorithm. *Statistics & Probability Letters*, 91:14–19, 2014.

Zhang, Y. and Sutton, C. A. Quasi-Newton methods for Markov Chain Monte Carlo. In *Advances in Neural Information Processing Systems*, pp. 2393–2401, 2011.