Quasi-Newton Hamiltonian Monte Carlo

Tianfan Fu
Computer Science Dept.
Shanghai Jiao Tong University
Shanghai 200240, China

Luo Luo
Computer Science Dept.
Shanghai Jiao Tong University
Shanghai 200240, China

Zhihua Zhang
Computer Science Dept.
Shanghai Jiao Tong University
Shanghai 200240, China

Abstract

The Hamiltonian Monte Carlo (HMC) method has become significantly popular in recent years. It is the state-of-the-art MCMC sampler due to its more efficient exploration to the parameter space than the standard random-walk based proposal. The key idea behind HMC is that it makes use of first-order gradient information about the target distribution. In this paper, we propose a novel dynamics using second-order geometric information about the desired distribution. The second-order information is estimated by using a quasi-Newton method (say, the BFGS method), so it does not bring heavy computational burden. Moreover, our theoretical analysis guarantees that this dynamics remains the target distribution invariant. As a result, the proposed quasi-Newton Hamiltonian Monte Carlo (QNHMC) algorithm traverses the parameter space more efficiently than the standard HMC and produces a less correlated series of samples. Finally, empirical evaluation on simulated data verifies the effectiveness and efficiency of our approach. We also conduct applications of QNHMC in Bayesian logistic regression and online Bayesian matrix factorization problems.

1 Introduction

Hamiltonian Monte Carlo (HMC) (Neal, 2011) is the state-of-the-art MCMC sampling algorithm. It defines a Hamiltonian function in terms of a potential energy—the negative logarithm of the target distribution and a kinetic energy parameterized by an auxiliary variable called momentum. By simulating from such a dynamical system, the proposal of distant states can be achieved. The attractive property of HMC is its rapid exploration to the state space. The main reason is that HMC makes use of the first-order gradient about the target distribution so that random-walk behaviors are suppressed to a great extent.

Along the idea of HMC, stochastic gradient MCMC algorithms have received great attention (Welling and Teh, 2011; Ahn, Korattikara, and Welling, 2012; Patterson and Teh, 2013; Chen, Fox, and Guestrin, 2014; Ding et al., 2014). Recently, Ma, Chen, and Fox (2015) proposed a general framework for this kind of stochastic gradient MCMC algorithms, which is built on a skew-symmetry structure. This structure represents determining traversing effect in HMC procedure and becomes one motivation of our new dynamics.

On the other hand, it is well established that the Newton or quasi-Newton methods using second-order information are more advanced than first-order gradient methods in the numerical optimization community (Nocedal and Wright, 2006). Since the Newton method is deemed to be computationally intensive, the quasi-Newton method is widely used in practice. In this paper, we explore the possibility of marrying the second-order gradient with HMC. Intuitively, a naive approach is to replace the first-order gradient in HMC with the second-order gradient about the negative logarithm of the target distribution using the quasi-Newton method. However, we will see that the resulting dynamics leads to an incorrect stationary distribution. Thus, it is challenging to incorporate second-order gradient into HMC.

Motivated by the work of Ma, Chen, and Fox (2015), we construct a skew-symmetry structure in our dynamics via adding the approximation of the inverse Hessian matrix to the standard Hamiltonian dynamics. We theoretically prove that this new dynamics keeps the target distribution invariant. Accordingly, we develop a quasi-Newton Hamiltonian Monte Carlo (QNHMC) algorithm. In QNHMC, both momentum and position variables are rescaled into a better condition using the geometric information estimated via a quasi-Newton method. Such an algorithm would produce a better proposal, take a large movement in the extended Hamiltonian dynamical system and enable a faster convergence rate to the desired distribution, which is verified by empirical results.

The remainder of the paper is organized as follows. Basic quasi-Newton and HMC methods are introduced in Section 2. We then describe our approach QNHMC in Section 3. Related studies are briefly reviewed in Section 4 while empirical results are shown in Section 5. Finally, we conclude our work in Section 6. All the proofs of theoretical results are given in Appendix.

2 Background

In this section we describe backgrounds on both quasi-Newton Approximation and Hamiltonian Monte Carlo.

2.1 Quasi-Newton Approximation

It is well-known that the Hessian matrix describes second-order curvature of the objective function $f\colon \mathbb{R}^d \to \mathbb{R}$. The time complexity and space complexity for exactly computing the Hessian matrix are both $O(d^2)$, which is computationally prohibitive for high-dimensional problems. Alternatively, quasi-Newton methods, including the BFGS method and its variant limited-memory BFGS(L-BFGS), are widely used (Nocedal and Wright, 2006).

In particular, let $\boldsymbol{\theta} \in \mathbb{R}^d$ be the parameter that needs to be estimated. Given the previous m estimates $\{\boldsymbol{\theta}_{k-m+1}, \boldsymbol{\theta}_{k-m+2}, \ldots, \boldsymbol{\theta}_k\}$ of $\boldsymbol{\theta}$ and the k-th estimate $\mathbf{B}_k$ of the inverse Hessian matrix, the BFGS updates $\mathbf{B}_{k+1}$ in the following way:

$$\mathbf{B}_{k+1} = (\mathbf{I} - \frac{\mathbf{s}_k \mathbf{y}_k^T}{\mathbf{y}_k^T \mathbf{s}_k})\mathbf{B}_k(\mathbf{I} - \frac{\mathbf{y}_k \mathbf{s}_k^T}{\mathbf{y}_k^T \mathbf{s}_k}) + \frac{\mathbf{s}_k \mathbf{s}_k^T}{\mathbf{s}_k^T \mathbf{y}_k}, \quad (1)$$

where $\mathbf{I}$ is the identity matrix, $\mathbf{s}_k = \boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k$, and $\mathbf{y}_k = \nabla f(\boldsymbol{\theta}_{k+1}) - \nabla f(\boldsymbol{\theta}_k)$. It is implemented by storing the full $d \times d$ matrix $\mathbf{B}_k$. However, in the high-dimensional scenario (i.e., d is very large), the limited-memory BFGS is much more efficient. Specifically, in L-BFGS, $\mathbf{B}_{k-m+1}$ is set as $\gamma\mathbf{I}$ for some $\gamma > 0$, and $\{\mathbf{s}_{k-m+1}, \ldots, \mathbf{s}_{k-1}\}$ and $\{\mathbf{y}_{k-m+1}, \ldots, \mathbf{y}_{k-1}\}$ are stored. The involved matrix-vector product can be computed in linear time $O(md)$ by using a specially-designed recursive algorithm (Nocedal and Wright, 2006).

2.2 Hamiltonian Monte Carlo

We now briefly review the Hamiltonian Monte Carlo (Neal, 2011). HMC lies in Metropolis-Hastings (MH) framework. It can traverse long distances in the parameter space during a single transition. The proposals are generated from a Hamiltonian system by extending the state space via adding an auxiliary variable called momentum variable, and then simulating Hamiltonian dynamics to move long distances along the iso-probability contours in the extended parameter space.

Formally, suppose that $\boldsymbol{\theta} \in \mathbb{R}^d$ is the parameter of interest and $\pi(\boldsymbol{\theta})$ is the desired posterior distribution. Let $\mathbf{p}$ be the auxiliary variable which is independent of $\boldsymbol{\theta}$. For simplicity and generality, $\mathbf{p} \in \mathbb{R}^d$ is always assumed to be a zero-mean Gaussian with covariance $\mathbf{M}$.

Then a Hamiltonian is defined as the negative log-probability of the joint distribution $p(\boldsymbol{\theta}, \mathbf{p})$ as follows:

$$\begin{aligned} H(\boldsymbol{\theta}, \mathbf{p}) &= -\log p(\boldsymbol{\theta}, \mathbf{p}) \\ &= -\log \pi(\boldsymbol{\theta}) - \log p(\mathbf{p}) \\ &= U(\boldsymbol{\theta}) + \frac{1}{2}\mathbf{p}^T \mathbf{M}^{-1} \mathbf{p} + \text{const}, \end{aligned} \quad (2)$$

where $U(\boldsymbol{\theta}) = -\log \pi(\boldsymbol{\theta})$ is called the potential function. In the Hamiltonian system $\mathbf{M}$ is called a preconditioning mass matrix, $\boldsymbol{\theta}$ is regarded as a position variable, and $\mathbf{p}$ is called a momentum variable.

Given an initial state $(\boldsymbol{\theta}_0, \mathbf{p}_0)$, the state $(\boldsymbol{\theta}, \mathbf{p})$ is generated by deterministic simulation of Hamiltonian dynamics based on the following ordinary differential equation (ODE):

$$\begin{aligned} \dot{\boldsymbol{\theta}} &= \mathbf{M}^{-1}\mathbf{p}, \\ \dot{\mathbf{p}} &= -\nabla U(\boldsymbol{\theta}), \end{aligned} \quad (3)$$

where the dots denote the derivatives in time. In this paper we will also use $\mathbf{z} = (\boldsymbol{\theta}, \mathbf{p}) \in \mathbb{R}^{2d}$ to denote the joint variable of the position and momentum variables.

The trajectories of $\boldsymbol{\theta}$ and $\mathbf{p}$ produce proposals to the Metropolis-Hastings procedure. In particular, given the k-th estimate of the position variable $\boldsymbol{\theta}_k$, the standard HMC runs in the following steps: (i) draw $\mathbf{p}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{M})$; (ii) compute the proposal$(\boldsymbol{\theta}^\star, \mathbf{p}^\star)$ by simulation from Eq. (3) using ϵ-discretization; (iii) compute the error caused by discretization $\nabla H = H(\boldsymbol{\theta}_k, \mathbf{p}_k) - H(\boldsymbol{\theta}^\star, \mathbf{p}^\star)$; (iv) accept new proposal $\boldsymbol{\theta}^\star$ with probability at $\min(\exp(\nabla H), 1)$.

It is worth mentioning that the error is only caused from the discretization and highly related to the step size ϵ. If the discretization error vanishes, that is to say, Eq. (3) is solved exactly, then Hamiltonian $H(\boldsymbol{\theta}, \mathbf{p})$ is conserved exactly and new proposal is always accepted. The details of HMC can be found in Neal (2011).

In practice, the mainstream integrator is the well-known leapfrog method in Algorithm 1, which is symplectic and time-reversible. The error is second-order in the step size ϵ, keeping acceptance rate a reasonable value. However, when the ODE described in Eq. (2) is stiff, the step size ϵ is required to be small to maintain a reasonable acceptance rate. This will cause the high-correlated series and reduce the effective sample size. A main reason is that only first-order gradient is not enough. That is, HMC fails to make sufficient use of the local geometric information, as shown by Girolami and Calderhead (2011); Zhang and Sut-

Algorithm 1 Standard Hamiltonian Monte Carlo(HMC)

Input: target posterior distribution $\pi(\boldsymbol{\theta})$ and potential function $U(\boldsymbol{\theta}) = -\log \pi(\boldsymbol{\theta})$, step size ϵ, number of leapfrog L, total number of sample N, burn-in samples K, start point $\boldsymbol{\theta}_0$, mass matrix $\mathbf{M}$

Output: $\{\boldsymbol{\theta}_{K+1}, \boldsymbol{\theta}_{K+2}, \ldots, \boldsymbol{\theta}_N\}$

Iteration counter $t = 1$.
while $t < N$ **do**
  Let $\mathbf{q} = \boldsymbol{\theta}_t$.
  Draw $\mathbf{p} \sim \mathcal{N}(\mathbf{0}, \mathbf{M})$.
  Compute the current energy E_0 using Eq. (2).
  First half step: $\mathbf{p} = \mathbf{p} - \epsilon \nabla U(\mathbf{q})/2$
  for $i = 1 : L$ **do**
    $\mathbf{q} = \mathbf{q} + \epsilon \mathbf{M}^{-1}\mathbf{p}$
    if $i \neq L$ **then**
      $\mathbf{p} = \mathbf{p} - \epsilon \nabla U(\mathbf{q})$
    end if
  end for
  Last half step: $\mathbf{p} = \mathbf{p} - \epsilon \nabla U(\mathbf{q})/2$
  Compute the proposed energy E_1 using Eq. (2).
  Draw $u \sim \text{Uniform}(0, 1)$
  if $u < \min\{1, \exp(E_1 - E_0)\}$ **then**
    Accept the proposal $\mathbf{q}$, i.e., $\boldsymbol{\theta}_{t+1} = \mathbf{q}$.
  else
    Reject, $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t$.
  end if
  $t = t + 1$.
end while

ton (2011). We also demonstrate this problem empirically in Section 5.

3 Methodology

In this section, we first discuss the condition for variants of Hamiltonian dynamics to reach the correct stationary distribution and then propose a novel dynamics satisfying the condition. Accordingly, we develop a quasi-Newton Hamiltonian Monte Carlo (QNHMC) algorithm.

3.1 A Naive Replacement

Intuitively, the most straightforward approach to apply the quasi-Newton method on Hamiltonian Monte Carlo is simply to replace $\nabla U(\boldsymbol{\theta})$ in Algorithm 1 by $\mathbf{B}\nabla U(\boldsymbol{\theta})$, where $\mathbf{B}$ is the approximation to the inverse Hessian matrix. The resulting discrete time system can be viewed as an ϵ-discretization of the following continuous ODE:

$$\begin{aligned} \dot{\boldsymbol{\theta}} &= \mathbf{M}^{-1}\mathbf{p}, \\ \dot{\mathbf{p}} &= -\mathbf{B}(t)\nabla U(\boldsymbol{\theta}), \end{aligned} \tag{4}$$

where $\mathbf{B}(t) \in \mathbb{R}^{d\times d}$ is positive definite and varies with time t. For simplicity, we always ignore the time t and assume that $\mathbf{B}$ is not the identity matrix, i.e., $\mathbf{B} = \mathbf{B}(t) \neq \mathbf{I}$. Now we show that, as given by Corollary 1 below, $p(\boldsymbol{\theta}, \mathbf{p}) \propto \exp(-H(\boldsymbol{\theta}, \mathbf{p}))$ is no longer the stationary distribution of the dynamics described in Eq. (4). The following theorem shows a stronger result, a necessary condition for the invariance property, i.e., the dynamics governed by Eq. (4) can not conserve the entropy of p_t with time.

Theorem 1. *Let $p_t(\boldsymbol{\theta}, \mathbf{p})$ be the distribution of $(\boldsymbol{\theta}, \mathbf{p})$ at time t with dynamics described in Eq.* (4)*. Define the entropy of $p_t(\boldsymbol{\theta}, \mathbf{p})$ as $h(p_t) = -\int_{\boldsymbol{\theta},\mathbf{p}} f(p_t(\boldsymbol{\theta}, \mathbf{p}))d\boldsymbol{\theta}d\mathbf{p}$, where $f(x) = x \ln x$ is defined for measuring entropy. Assume p_t is a distribution with density and gradient vanishing at infinity and the gradient vanishes faster than $\frac{1}{\ln p_t}$. Then, the entropy of p_t varies over time.*

Intuitively, Theorem 1 is true because the standard Hamiltonian dynamics strictly preserve the entropy (Qian, 2012). The additional $\mathbf{B}$ can be seen as a noise, destroying the entropy preservation. This hints the fact that the distribution $p_t(z)$ tends toward far from the target distribution.

Based on Theorem 1, we conclude that the naive modification to Hamiltonian dynamics can not keep the target distribution invariant. That is,

Corollary 1. *The distribution of extended system $p(\boldsymbol{\theta}, \mathbf{p}) \propto exp(-H(\boldsymbol{\theta}, \mathbf{p}))$ is no longer invariant under the dynamics described by Eq.* (4)*.*

This corollary claims the failure of dynamics governed by Eq. (4). In what follows, we will consider a general framework which builds the necessary and sufficient condition for variants of Hamiltonian dynamics to satisfy the invariance of the target distribution. Accordingly, this leads us to a new dynamics.

3.2 Motivation: A general recipe

In this section, we introduce a general framework for stochastic gradient MCMC algorithms proposed recently by Ma, Chen, and Fox (2015). It provides the sufficient and necessary condition for Hamiltonian systems to reach the invariant distribution.

Consider the following Stochastic Differential Equation (SDE) for continuous Markov processes for sampling:

$$d\mathbf{z} = f(\mathbf{z})dt + \sqrt{2D(\mathbf{z})}dW(t), \tag{5}$$

where $f(\cdot) : \mathbb{R}^{2d} \to \mathbb{R}^{2d}$ denotes the deterministic drift and is often related to the gradients of Hamiltonian $\nabla H(\mathbf{z})$, $W(t)$ is a $2d$-dimensional Wiener process, and $D(\mathbf{z}) \in \mathbb{R}^{2d\times 2d}$ is a positive semi-definite diffusion matrix. Obviously, not all choices of $f(\mathbf{z})$ and $D(\mathbf{z})$ can yield the stationary distribution $p(\boldsymbol{\theta}, \mathbf{q}) = p(\mathbf{z}) \propto \exp(-H(\mathbf{z}))$.

Ma, Chen, and Fox (2015) devised a recipe for constructing SDEs with the correct stationary distributions. They

defined $f(\mathbf{z})$ directly in terms of the target distribution:

$$f(\mathbf{z}) = -[D(\mathbf{z}) + Q(\mathbf{z})]\nabla H(\mathbf{z}) + \Xi(\mathbf{z}),$$
$$\Xi_i(\mathbf{z}) = \sum_{j=1}^{d} \frac{\partial}{\partial \mathbf{z}_j}(D_{ij}(\mathbf{z}) + Q_{ij}(\mathbf{z})). \tag{6}$$

Here $\Xi_i(\cdot)$ is the i-th entry of the vector-valued function $\Xi(\cdot) : \mathbb{R}^{2d} \to \mathbb{R}^{2d}$, and $Q(\mathbf{z}) \in \mathbb{R}^{2d\times 2d}$ is a skew-symmetric curl matrix representing the determining traversing effects seen in the HMC procedure. It was proved that under certain conditions $p(\mathbf{z})$ is the unique stationary distribution of the dynamics governed by Eq. (5) (Ma, Chen, and Fox, 2015).

Theorem 2. *$p(\mathbf{z}) \propto \exp(-H(\mathbf{z}))$ is a stationary distribution of the dynamics described in Eq.* (5) *if f is restricted to the form of Eq.* (6)*, with $D(\mathbf{z})$ positive semidefinite and $Q(\mathbf{z})$ skew-symmetric.*

Since in this paper we restrict our attention to the deterministic dynamics, we can omit the term related to diffusion part $D(\mathbf{z})$. In this case, the SDE is reduced to an ODE. The skew-symmetry of $Q(\mathbf{z})$ inspires us to add $\mathbf{B}$ into update of the position variable $\boldsymbol{\theta}$ in the dynamics in Eq. (4).

In the next section we will consider skew-symmetric modification to the Hamiltonian dynamics that achieves the desired $p(\boldsymbol{\theta}, \mathbf{p})$ as the invariant distribution of the continuous Hamiltonian dynamical system.

3.3 Novel Dynamics in skew-symmetric structure

In this section, inspired by the skew-symmetric structure, we consider a variant on Hamiltonian dynamics as follows:

$$\dot{\boldsymbol{\theta}} = \mathbf{C}\mathbf{M}^{-1}\mathbf{p},$$
$$\dot{\mathbf{p}} = -\mathbf{C}\nabla U(\boldsymbol{\theta}), \tag{7}$$

where $\mathbf{C} \in \mathbb{R}^{d\times d}$ is a symmetric positive definite matrix, independent of $\boldsymbol{\theta}$ and $\mathbf{p}$.

Now we show that the new dynamics maintains the desired distribution as the invariant distribution.

Theorem 3. *$p(\boldsymbol{\theta}, \mathbf{p}) \propto exp(-H(\boldsymbol{\theta}, \mathbf{p}))$ is the unique stationary distribution of the dynamics governed by Eq.* (7)*.*

Then it is easy to prove the entropy preservation of proposed dynamics directly using the intermediate results in Theorem 3.

Corollary 2. *Let $p_t(\boldsymbol{\theta}, \mathbf{p})$ be the distribution of $(\boldsymbol{\theta}, \mathbf{p})$ at time t with dynamics described in Eq.* (7)*. Under almost the same condition with Theorem 1, i.e., the entropy of $p_t(\boldsymbol{\theta}, \mathbf{p})$ is defined as $h(p_t) = -\int_{\boldsymbol{\theta},\mathbf{p}} f(p_t(\boldsymbol{\theta}, \mathbf{p}))d\theta d\mathbf{p}$, where $f(x) = x \ln x$. Assume p_t is an arbitrary distribution with density and gradient vanishing at infinity. Then, the entropy of p_t is strictly conserved with time.*

Algorithm 2 Quasi-Newton HMC (QNHMC)

Input: target distribution $\pi(\boldsymbol{\theta})$ and potential function $U(\boldsymbol{\theta}) = -\log \pi(\boldsymbol{\theta})$, step size ϵ, number of Leapfrog L, total number of sample N, burn-in samples K, start point $\boldsymbol{\theta}_0$, mass matrix $\mathbf{M}$, approximation of Hessian Matrix $\mathbf{B} = \mathbf{I}$.
Output: $\boldsymbol{\theta}_{K+1}, \boldsymbol{\theta}_{K+2}, \cdots, \boldsymbol{\theta}_N$.

1: Iteration counter $t = 1$.
2: **while** $t < N$ **do**
3: Let $\mathbf{q} = \boldsymbol{\theta}_t$ and $\mathbf{C} = \mathbf{B}$.
4: Draw $\mathbf{p} \sim \mathcal{N}(\mathbf{0}, \mathbf{M})$.
5: Compute the current energy E_0 using Eq. (2).
6: First half step: $\mathbf{p} = \mathbf{p} - \epsilon\mathbf{C}\nabla U(\mathbf{q})/2$.
7: Update $\mathbf{B}$ using Eq. (1).
8: **for** $i = 1 : L$ **do**
9: $\mathbf{q} = \mathbf{q} + \epsilon\mathbf{C}\mathbf{M}^{-1}p$.
10: **if** $i \neq L$ **then**
11: $\mathbf{p} = \mathbf{p} - \epsilon\mathbf{C}\nabla U(\mathbf{q})$.
12: Update $\mathbf{B}$ using Eq. (1).
13: **end if**
14: **end for**
15: Last half step: $\mathbf{p} = \mathbf{p} - \epsilon\mathbf{C}\nabla U(\mathbf{q})/2$
16: Update $\mathbf{B}$ using Eq. (1).
17: Compute the proposed energy E_1 using Eq. (2).
18: Draw $u \sim$ Uniform$(0, 1)$.
19: **if** $u < \min\{1, \exp(E_1 - E_0)\}$ **then**
20: Accept the proposal $\mathbf{q}$, i.e., $\boldsymbol{\theta}_{t+1} = \mathbf{q}$.
21: **else**
22: Reject, $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t$. $\mathbf{B} = \mathbf{C}$.
23: **end if**
24: $t = t + 1$.
25: **end while**

In summary, we have shown that the dynamics given by Eq. (7) owns the invariance property. Furthermore, it is concise. When $\mathbf{C} = \mathbf{I}$, it reduces to Hamiltonian dynamics.

3.4 Quasi-Newton Hamiltonian Monte Carlo

In the previous section we mainly focus on the invariance property of the proposed dynamics given in Eq. (7), which in essence is a continuous ODE. However, in practice, we need a numerical solution to the continuous ODE in Eq. (7). Here, the discretization step employ leapfrog methods, inheriting from the standard HMC (Neal, 2011). In particular, the resulting Quasi-Newton Hamiltonian Monte Carlo (**QNHMC**) algorithm is shown in Algorithm 2.

It is worth noting that in Algorithm 2 the approximation $\mathbf{B}$ varies with iteration counter. In contrast, to keep the invariance property of the dynamics in Eq. (7), in each proposal (Step 3-15), $\mathbf{B}$ is kept as a constant $\mathbf{C}$. The update to $\mathbf{B}$ is only done when the proposal procedure ends and the proposal is accepted. The following theorem shows the correctness of Algorithm 2.

Theorem 4. *$\pi(\boldsymbol{\theta})$ is maintained as an invariant distribution for the whole chain produced by Algorithm 2.*

To gain some intuitions about the proposed QNHMC algorithm in physical interpretation, consider the well-known hockey puck instance widely used in the Hamiltonian dynamical system (Leimkuhler and Reich, 2004), where we can imagine the puck on an uneven surface. Here, $\mathbf{B}$ represents the approximation to the local geometric information. In HMC, when the local geometry is stiff, movements controlled by momentum $\mathbf{p}$ are always useless. By multiplying the matrix $\mathbf{B}$ ($\mathbf{C}$), the momentum variable and position variable are rescaled into the case where each dimension has a similar scale. This makes the movement in the extended Hamiltonian system more efficient. Empirically, it can traverse a large step in the state space. In the Bayesian scenario, this new dynamics produces a better proposal and less-autocorrelated series.

3.5 Computational Complexity

Two strategies (BFGS/L-BFGS) are used in estimating the matrix $\mathbf{B}$ according to the dimension of the parameter. When the dimension of the parameter d is high, the L-BFGS is adopted. In this case, $O(md)$ space complexity is needed. Moreover, since only matrix-vector product is required, the product $\mathbf{Bv}$ ($\mathbf{Cv}$) can be computed in linear time $O(md)$. It is worth noting that unlike in Algorithm 2, no need to update $\mathbf{B}$ in each leapfrog step when adopting L-BFGS. Instead, we choose to store $2m$ d-dimensional vectors as mentioned in Section 2. Only when the proposal is accepted, it is required to update $\mathbf{B}$.

On the other hand, when d is not high, BFGS is chosen. A $d \times d$ matrix needs to be stored so the complexity is $O(d^2)$. Each updating step requires $O(d)$ time without matrix-vector or matrix-matrix product.

In this paper, we mainly compare our QNHMC approach with the standard HMC (Neal, 2011). The gradient computation of the negative log-posterior $\nabla U(\boldsymbol{\theta})$ has already been done in the standard HMC in each leapfrog step. And, in practice, the gradient computation always contains expensive computations such as matrix-matrix/matrix-vector product. The size of matrix depends on both the dimension of the parameter and the number of the training instances. Our QNHMC algorithm can make use of the byproduct of leapfrog step, e.g., $\mathbf{s}_k$ and $\mathbf{y}_k$ in Eq. (1). Thus, the gradient computation is always dominant in computational time, which means that HMC and QNHMC cost the same order of magnitude running time in each proposal. Considering the larger step that QNHMC takes owing to more sufficient use of geometric information, QNHMC converges to the desired distribution faster than the standard HMC, which will be further validated by our empirical results.

4 Related Studies

In recent years, many approaches have been proposed to scale up Bayesian methods in machine learning community. As is well-known, gradient information of log-posterior distribution is widely used in Langevin/Hamiltonian dynamics. Among these methods, one large category is stochastic gradient Markov Chain Monte Carlo methods. The framework is to estimate the gradient from small mini-batches of observations instead of the whole dataset to cut the computational budget. The landmark of this kind of approaches is proposed by Welling and Teh (2011), which applied stochastic gradient on Langevin dynamics. Subsequently, a series of algorithms are developed to complete this framework (Ahn, Korattikara, and Welling, 2012; Patterson and Teh, 2013; Ahn, Shahbaba, and Welling, 2014; Chen, Fox, and Guestrin, 2014; Ding et al., 2014; Ma, Chen, and Fox, 2015). For instance, Chen, Fox, and Guestrin (2014) adapted stochastic gradient on Hamiltonian Monte Carlo (SGHMC) while Ding et al. (2014) devised a SGNHT algorithm by introducing a thermostat variable to make SGHMC more robust. Ma, Chen, and Fox (2015) developed a complete recipe for all stochastic gradient based MCMC approaches. There are some recent works (Bardenet, Doucet, and Holmes, 2014; Korattikara, Chen, and Welling, 2013; Maclaurin and Adams, 2015), attempting to perform Metropolis-Hastings rejection procedure using partial observations instead of whole dataset. These approaches mainly accelerate sampling procedure via a stochastic method applied on likelihood of data observations.

Another class of methods aim at sampler itself (Girolami and Calderhead, 2011; Zhang and Sutton, 2011; Calderhead and Sustik, 2012; Patterson and Teh, 2013; Wang, Mohamed, and Nando, 2013; Chao et al., 2015). Most of them are closely related to geometry. Concretely, they aim at finding the local geometric structure of posterior so that the random-walk behaviour in proposal can be significantly suppressed. Such algorithms allow a larger step size without loss of acceptance rate and make the sampling efficient. The representative work is Riemann Manifold Hamiltonian Monte Carlo proposed by Girolami and Calderhead (2011), which employed the high-order of geometric information about the local point. Chao et al. (2015) proposed an exponential integration to solve the high-oscillatory component of posterior more accurately than the integration used in the standard HMC method.

It is worth noting that Zhang and Sutton (2011) also proposed a quasi-Newton based Hamiltonian Monte Carlo method called HMC-BFGS. Different from our QNHMC, HMC-BFGS uses the geometric information in covariance of zero-mean proposal. In contrast, our QNHMC makes use of second-order information in scaling both momentum and position variables. Furthermore, HMC-BFGS devises

an extended chain. The specially-designed structure may sacrifice the convergence rate and leads to a poor estimate of Hessian, validated by our empirical evaluations.

5 Empirical Evaluation

In this section, we empirically analyze the quasi-Newton HMC. We mainly compare our quasi-Newton HMC (QNHMC) with the standard HMC (Neal, 2011) and HMC-BFGS (Zhang and Sutton, 2011) in both efficiency and effectiveness. In all cases, QNHMC outperforms HMC and HMC-BFGS under the same settings of hyperparameters in both burn-in time and effective sample size.

5.1 Simulated data

First, we evaluate the scalability of our model on a high-dimensional zero-mean Gaussian distribution. The target distribution is high-dimensional Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$. The covariance matrix is $\mathbf{\Sigma} = \mathbf{1}\mathbf{1}^T + 4\mathbf{I} \in \mathbb{R}^{d\times d}$, where $\mathbf{1}$ and $\mathbf{0}$ are the all-1 and all-0 vectors of d-dimension, respectively. This d-dimensional distribution is highly correlated in one direction. Hence, it is a challenging task for random-walk based sampler, such as MCMC.

The main evaluating metrics are as follows:

- Autocorrelation. Since the desired distribution is highly correlated, when computing the autocorrelation of samples, the samples are projected onto the direction of its largest eigenvalue ($\mathbf{x} = \mathbf{1}$). And the max number of lag m is set to 500. ρ_k is the autocorrelation at lag k.

- Effective sample size (ESS) is the common measurement, which summaries the amount of autocorrelation across different lags over all dimensions. ESS is formally defined as follows

$$\text{ESS} = \frac{n}{1 + 2\sum_{k=1}^{m} \rho_k},$$

  where n is the original sample size, m is the maximal number of lag and determined empirically. In this scenario, $m = 500$. Notice that ESS per second is also a metric to measure the efficiency of sampler.

- Convergence diagnostics. To compare the convergence rate fairly, we choose the starting position far away from the mode of the target distribution. Notice that the starting positions are same for all the samplers. Thus the chain should be run long enough to "forget" the starting position. This is the so-called burn-in period. Determining the length of burn-in period is critical. In this paper, we choose to monitor the probability of the samples, a mainstream method in convergence diagnostics (Gilks, Richardson, and Spiegelhalter, 1996). In particular, by observing the negative log-probability of samples $\mathbf{x}$ (i.e., $(\mathbf{x}-\mu)^T\mathbf{\Sigma}(\mathbf{x}-\mu)$, ignoring the normalizing constant), we can easily find that how long steps the chain takes to reach the highest posterior density (HPD) region and ends the burn-in period.

In this experiment, for fair comparison among these three methods, we adopt the same setting of the hyperparameters involved. In particular, we use the step size $\epsilon = 0.01$ and the number of leaps $L = 10$, the dimension of the problem $d = 100$. We draw 100K samples for each sampler. For HMC-BFGS, we need to choose an ensemble of K chains. If K is too large, there will be numerical unstability about the BFGS methods and the performance will degrade drastically. Here, K is chosen to be 5, following the setting of Zhang and Sutton (2011). The starting positions for these methods are the same, a randomly-generated point distant from the HPD region.

As illustrated in Figure 1, the chain of QNHMC only requires hundreds of samplings to end the burn-in period and reach the HPD region while the rest two algorithms need far more samplings. Hence, one advantage of QNHMC is that it can converge to the HDP region quicker than the other two algorithms.

Though QNHMC only requires hundreds of burn-in samples here, for fair comparison, when computing ESS and autocorrelation, we only use the last 50K samples for all the methods. The results are shown in Table 1. We observe that QNHMC can not only produce more "independent" series than the other methods, but also obtain the most effective sample size among the three methods under the same setting and draw samples from the desired distribution most efficiently.

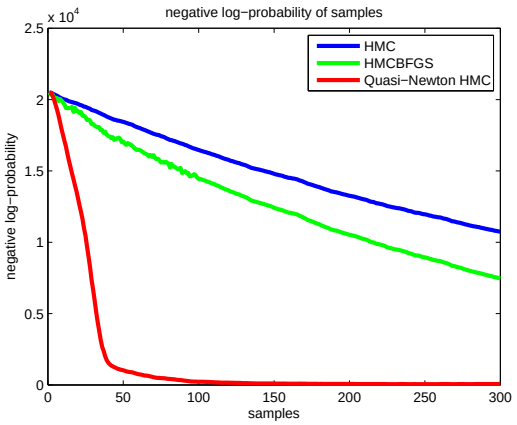

Figure 1: Performance of different samplers on simulated data: monitoring the convergence to HPD: negative log-probability of samples ($(\mathbf{x} - \mu)^T\mathbf{\Sigma}(\mathbf{x} - \mu)$) v.s. iteration

Table 1: Performance of different samplers on artificial data, as measured by sum of autocorrelation coefficient, burn-in time, Effective Sampler Size(ESS) and ESS per second. For the first two metric, lower is better while for the other two metric, higher is better.

Method	$\sum_{k=1}^{m} \lvert\rho_k\rvert$	burn-in time(second)	ESS	ESS per second
HMC	98.27	81	253	7.22
HMC-BFGS	49.74	47	497	9.65
QN-HMC	2.65	0.93	7936	93.36

Table 2: Performance of different methods on Bayesian Logistic Regression, as measured by sum of autocorrelation coefficient, burn-in time(in seconds) for sampling based methods, Effective Sampler Size(ESS) and ESS per second. GD is the abbreviation for gradient descent.

Method	burn-in time	ESS per second	Error
HMC	64	0.286	0.0493
HMC-BFGS	36	0.418	0.0501
QN-HMC	17	1.30	0.0496
GD	–	–	0.0578

5.2 Bayesian Logistic Regression

Next, we evaluate our method on a well-known handwritten digits classification task using the MNIST dataset[1]. In this task, we aim at discriminate digits "7" and "9". The number of training instances are 6265 and 5949 for "7" and "9", respectively while the number of test instances are 1028 and 1009 for "7" and "9", respectively.

We test four methods: gradient descent (GD), HMC, HMCBFGS and our QNHMC. For GD, we employ an ℓ_2 regularizer trial several time to choose the near-optimal hyperparameter. And in each iteration, we use all the training instances. For the sampling-based approaches, we take a fully Bayesian approach and place a weakly informative Gaussian prior on the weight, following Zhang and Sutton (2011); Girolami and Calderhead (2011).

The results for the sampling methods are shown in Table 2. The result with GD is provided as a baseline. We set the number of leapfrog L to 5 for all sampling based approaches. The step size ϵ is set to 0.1. Dimension $d = 784$ here, and L-BFGS method are used, where m is set to be 7. We can see that the Bayesian approaches are superior to the optimization based methods. Among the Bayesian approaches, QNHMC requires less burn-in time than the others and converges to a low test error faster. This shows its advantage over the other two HMC-related methods.

[1]http://yann.lecun.com/exdb/mnist/

5.3 Online Bayesian Matrix Factorization

Collaborative filtering is a popular theme. The target is to predict users' preference over a set of items, e.g., movies, music and produce recommendation. Owing to the sparsity in the ratings matrix (users versus items) in recommendation systems, over-fitting is a severe issue. Hence Bayesian approaches provide a natural solution. The most famous Bayesian algorithm for collaborative filtering is the online probabilistic matrix factorization proposed in Salakhutdinov and Mnih (2008).

In the experiment, the Root-Mean-Square Error (RMSE) is used to evaluate the performance of the three algorithms. It is defined as

$$\text{RMSE} = \|P_{\Omega_{\text{test}}}(\mathbf{X}) - P_{\Omega_{\text{test}}}(\mathbf{T})\|_F, \tag{8}$$

where Ω_{test} represents the index of all testing entries and $|\Omega_{test}|$ is the cardinality of Ω_{test}, and $\mathbf{X}$ is the solution from the algorithms and $P_{\Omega_{\text{test}}}(\mathbf{T})$ is corresponding partially observed labels, $\|\mathbf{X}\|_F$ represents the Frobenius norm of the matrix $\mathbf{X}$. $P_\Omega(\mathbf{X})$ represent the entry-wise projection of $\mathbf{X}$ onto Ω, defined by:

$$\{P_\Omega(\mathbf{X})\}_{ij} = \begin{cases} \mathbf{X}_{ij}, & (i,j) \in \Omega \\ 0, & \text{otherwise.} \end{cases}$$

We conduct the experiment in online Bayesian PMF on the Movielens-1M datasets[2]. The dataset contains about 1 million ratings of 3,952 movies by 6,040 users. The number of latent dimensions is set to 10. Other settings follow from the demo code given by Salakhutdinov and Mnih (2008). For HMC-related methods, step size ϵ is set to 0.01, length of leapfrog $L = 10$.

Performances of the different methods are shown in Table 3. The sampling-based methods (standard HMC, QNHMC and HMC-BFGS) provide better prediction results than the optimization-based methods (MAP), showing an advantage of Bayesian inference in this scenario, thus validating the need for scalable and efficient Bayesian algorithm such as QNHMC. In this experiment, prediction results for QNHMC, HMC and HMCBFGS are comparable. This experiment shows that QNHMC converges faster than the

[2]http://grouplens.org/datasets/movielens/

Table 3: Performance of different methods on Online Bayesian Matrix Factorization in terms of RMSE and running time. For RMSE, lower is better.

Method	RMSE	Running time(second)
MAP	0.8701	27.67
HMC	0.8607	202.5
QNHMC	0.8628	108.34
HMC-BFGS	0.8618	179.23

other two HMC-related approaches and can be seen as an effective choice for online Bayesian PMF.

6 Conclusion

In this paper we have proposed a novel quasi-Newton Hamiltonian Monte Carlo (QNHMC) algorithm to accelerate the Hamiltonian Monte Carlo sampler. Our theoretical analysis has guaranteed that the desired distribution is the unique stationary distribution under the proposed dynamics. The empirical results have verified the efficiency and effectiveness of our QNHMC on both simulated and practical datasets. A natural next step is to explore marrying stochastic gradient with QNHMC. More broadly, we believe that the unification of efficient optimization and sampling techniques, such as those described herein, will enable a significant scaling of Bayesian approaches.

Acknowledgements

This work has been supported by the National Natural Science Foundation of China (No. 61572017), Natural Science Foundation of Shanghai City (No. 15ZR1424200), and Microsoft Research Asia Collaborative Research Award.

Appendix

A: Proof of Theorem 1

Proof. Define

$$d\mathbf{z} = d\begin{pmatrix} \boldsymbol{\theta} \\ \mathbf{p} \end{pmatrix} = \begin{pmatrix} \mathbf{M}^{-1}\mathbf{p} \\ -\mathbf{B}\nabla U(\boldsymbol{\theta}) \end{pmatrix} dt = G(\mathbf{z})dt \quad (9)$$

where $G(\cdot) : \mathbb{R}^{2d} \to \mathbb{R}^{2d}$ is a function and the i-th component of $G(\mathbf{z})$ is denoted by $G_i(\mathbf{z})$. According to the definition of FPE (Fokker-Planck Equation) (Kadanoff, 2000), the corresponding FPE is given by

$$\partial_t p_t(\boldsymbol{\theta}, \mathbf{p}) = \nabla^T[G(\mathbf{z})p_t(\boldsymbol{\theta}, \mathbf{p})] \quad (10)$$

The entropy at time t is defined by integrating out the joint variable $\mathbf{z}$, as follows:

$$h(p_t) = -\int_{\mathbf{z}} f(p_t(\mathbf{z}))d\mathbf{z} \quad (11)$$

The evolution of the entropy is governed by

$$\begin{aligned} \partial_t h(p_t(\mathbf{z})) &= -\partial_t \int_{\mathbf{z}} f(p_t(z))d\mathbf{z} \\ &= -\int_{\mathbf{z}} f'(p_t(\mathbf{z}))\partial_t p_t(\mathbf{z})d\mathbf{z} \\ &= -\int_{\mathbf{z}} f'(p_t(\mathbf{z}))\nabla^T[G(\mathbf{z})p_t(\mathbf{z})]d\mathbf{z} \\ &= -\int_{\mathbf{z}} f'(p_t(\mathbf{z}))\nabla^T[G(z)]p_t(\mathbf{z})d\mathbf{z} \\ &\quad -\int_{\mathbf{z}} f'(p_t(\mathbf{z}))(\nabla p_t(\mathbf{z}))^T[G(\mathbf{z})]d\mathbf{z} \end{aligned} \quad (12)$$

where the last equality uses the fact that

$$\begin{aligned} &\nabla^T[G(\mathbf{z})p_t(\boldsymbol{\theta}, \mathbf{p})] \\ &= p_t(\theta, \mathbf{p})\nabla^T[G(\mathbf{z})] + (\nabla p_t(\boldsymbol{\theta}, \mathbf{p}))^T[G(\mathbf{z})] \end{aligned} \quad (13)$$

The second term on the RHS of Equation 12 can be simplified into following form:

$$\begin{aligned} &-\int_{\mathbf{z}} f'(p_t(\mathbf{z}))(\nabla p_t(\mathbf{z}))^T[G(\mathbf{z})]d\mathbf{z} \\ &= -\int_{\mathbf{z}} (\nabla f(p_t(\mathbf{z})))^T[G(\mathbf{z})]d\mathbf{z} \\ &= \int_{\mathbf{z}} f(p_t(\mathbf{z}))\nabla^T[G(\mathbf{z})]d\mathbf{z} = 0 \end{aligned} \quad (14)$$

where the second equality is given by integrations by parts, using the fact that

$$\int_{\mathbf{z}} \nabla^T[f(p_t(\mathbf{z}))G(\mathbf{z})]d\mathbf{z} = 0 \quad (15)$$

which is based on the assumption that the probability density vanishes at infinity and $f(x) \to 0$ as $x \to 0$ such that $f(p_t(\mathbf{z}))[G(\mathbf{z})] \to 0$ as $\mathbf{z} \to \infty$.

Hence the entropy of p_t varies with rate of

$$\begin{aligned} \partial_t h(p_t(\mathbf{z})) &= -\int_{\mathbf{z}} f^{'}(p_t(\mathbf{z}))\nabla^T[G(\mathbf{z})]p_t(\mathbf{z})d\mathbf{z} \\ &= -\int_{\mathbf{z}} f^{'}(p_t(\mathbf{z}))(\mathbf{p}^T\mathbf{M}^{-T}(\mathbf{B}-\mathbf{I})\nabla U(\theta))p_t(\mathbf{z})d\mathbf{z} \end{aligned} \quad (16)$$

This is not equal to zero for $\mathbf{B}$ obviously. □

B: Proof of Theorem 3

Proof. Using FPE (Kadanoff, 2000), Eq. (7) can be written in the following decomposed form:

$$\begin{aligned} d\mathbf{z} &= d\begin{pmatrix} \boldsymbol{\theta} \\ \mathbf{p} \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{0} & -\mathbf{C} \\ \mathbf{C} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \nabla U(\boldsymbol{\theta}) \\ \mathbf{M}^{-1}\mathbf{p} \end{pmatrix} dt \\ &= \begin{pmatrix} -\mathbf{C}\mathbf{M}^{-1}\mathbf{p} \\ \mathbf{C}\nabla U(\boldsymbol{\theta}) \end{pmatrix} dt \\ &= \mathbf{F}(\mathbf{z})dt. \end{aligned} \quad (17)$$

The distribution evolution under this dynamical system is governed by a Fokker-Planck Equation as following:

$$\begin{aligned}
&\partial_t p_t(\mathbf{z}) \\
&= -\nabla^T[F(\mathbf{z})p_t(\mathbf{z})] \\
&= \sum_{i=1}^{2d} \partial_{\mathbf{z}_i}[\mathbf{F}_i(\mathbf{z})p_t(\mathbf{z})] \\
&= \sum_{i=1}^{d} \partial_{\theta_i}[\mathbf{f}_i(\mathbf{z})p(\mathbf{z})] + \sum_{j=1}^{d} \partial_{\mathbf{p}_j}[\mathbf{g}_j(\mathbf{z})p(\mathbf{z})] \\
&= \sum_{i=1}^{d} [\partial_{\theta_i} p(\mathbf{z})]\mathbf{f}_i(\mathbf{z}) + \sum_{j=1}^{d} [\partial_{\mathbf{p}_j} p(\mathbf{z})]\mathbf{g}_j(\mathbf{z}) \\
&= \sum_{i=1}^{d} p(\mathbf{z})[-\nabla U(\theta)]_i \mathbf{f}_i(\mathbf{z}) + \sum_{j=1}^{d} p(\mathbf{z})[\mathbf{M}^{-1}\mathbf{p}]_j \mathbf{g}_j(\mathbf{z}) \\
&= p(\mathbf{z})(\nabla U(\theta))^T(-\mathbf{C}\mathbf{M}^{-1}\mathbf{p}) + p(\mathbf{z})(\mathbf{M}^{-1}\mathbf{p})^T(\mathbf{C}\nabla U(\theta)) \\
&= p(\mathbf{z})[(\nabla U(\theta))^T(-\mathbf{C}\mathbf{M}^{-1}\mathbf{p}) + (\nabla U(\theta))^T\mathbf{C}^T(\mathbf{M}^{-1}\mathbf{p})] \\
&= 0
\end{aligned} \tag{18}$$

where $f(\cdot)$ and $g(\cdot)$ are functions defined as: $f(\cdot) : \mathbb{R}^{2d} \to \mathbb{R}^d$, $g(\cdot) : \mathbb{R}^{2d} \to \mathbb{R}^d$. $\mathbf{f}(\mathbf{z}) = \mathbf{f}(\mathbf{p}, \boldsymbol{\theta}) = -\mathbf{C}\mathbf{M}^{-1}\mathbf{p}$ and $\mathbf{g}(\mathbf{z}) = \mathbf{g}(\mathbf{p}, \boldsymbol{\theta}) = C\nabla U(\boldsymbol{\theta})$. It satisfy that

$$F(\mathbf{z}) = \begin{pmatrix} -\mathbf{C}\mathbf{M}^{-1}\mathbf{p} \\ \mathbf{C}\nabla U(\boldsymbol{\theta}) \end{pmatrix} = \begin{pmatrix} f(\mathbf{z}) \\ g(\mathbf{z}) \end{pmatrix} \tag{19}$$

$\mathbf{f}_i(\mathbf{z})$ and $\mathbf{g}_j(\mathbf{z})$ are the i-th entry and j-th entry of $\mathbf{f}(\mathbf{z})$ and $\mathbf{g}(\mathbf{z})$ respectively.

The fourth equality follows from the fact that function $f(\mathbf{z})$ only depend on $\mathbf{p}$ while function $g(\mathbf{z})$ only depend on $\boldsymbol{\theta}$. Hence, following happens.

$$\begin{aligned}
\partial_{\boldsymbol{\theta}_i} f_i &= 0 \\
\partial_{\mathbf{p}_i} g_i &= 0
\end{aligned} \tag{20}$$

Since $p(\mathbf{z}) = \pi(\boldsymbol{\theta}, \mathbf{p}) = \exp(-U(\boldsymbol{\theta}) - 1/2\mathbf{p}^T\mathbf{M}^{-1}\mathbf{p})$, we expand the partial derivation as following:

$$\begin{aligned}
\partial_{\boldsymbol{\theta}_i} p(\mathbf{z}) &= p(\mathbf{z})[-\nabla U(\boldsymbol{\theta})]_i \\
\partial_{\mathbf{p}_i} p(\mathbf{z}) &= p(\mathbf{z})[\mathbf{M}^{-1}\mathbf{p}]_i
\end{aligned} \tag{21}$$

Hence the fifth equality satisfies.

The last equality is given by the fact that $\mathbf{C}$ is symmetric, i.e., $\mathbf{C}^T = \mathbf{C}$. By calculating $\partial_t p_t(\mathbf{z}) = 0$, we show that the distribution p_t does not vary with time, i.e., $p(\mathbf{z}) = \pi(\boldsymbol{\theta}, \mathbf{p})$ is invariant under the dynamics described in Equation 7. □

C: Proof of Corollary 2

Proof. With the entropy defined in Eq. (11) and compact form of dynamics defined in Eq. (17), The evolution of the entropy is governed by

$$\begin{aligned}
\partial_t h(p_t(\mathbf{z})) &= \partial_t \int_{\mathbf{z}} f(p_t(\mathbf{z}))d\mathbf{z} \\
&= -\int_{\mathbf{z}} f^{'}(p_t(\mathbf{z}))\partial_t p_t(\mathbf{z})d\mathbf{z} \\
&= -\int_{\mathbf{z}} f^{'}(p_t(\mathbf{z}))\nabla^T[F(\mathbf{z})p_t(\mathbf{z})]d\mathbf{z} \\
&= 0,
\end{aligned} \tag{22}$$

where the first two equalities can be referred from part of Eq. (12) and the last equality follows from the conclusion in Eq. (18). □

D: Proof of Theorem 4

The idea behind the correctness of Algorithm 2 is that Theorem 3 guarantees the detailed balance condition for any neighboring samples (e.g., $\boldsymbol{\theta}_{i-1}$ and $\boldsymbol{\theta}_i$) while Theorem 4 strengthen this results on the whole chain.

Proof. Firstly, we define $\{\boldsymbol{\theta}_i\}_{i=1}^n$, the chain generated by Algorithm 2. At i-th iteration, $\mathbf{B}$ is denoted by $\mathbf{B}_i$. For every i, the approximation to inverse Hessian $\mathbf{B}_i$ is symmetric positive definite (Nocedal and Wright, 2006). As seen from Algorithm 2, in i-th proposal (Step 3-15), $\mathbf{B}_i$ is fixed. Moreover, according to Theorem 3 and ϵ-discretization, we obtain that the detailed balance condition hold for any neighboring samples $\boldsymbol{\theta}_{i-1}$ and $\boldsymbol{\theta}_i$, i.e.,

$$\pi(\boldsymbol{\theta}_{i-1})\mathcal{T}_i(\boldsymbol{\theta}_{i-1} \to \boldsymbol{\theta}_i) = \pi(\boldsymbol{\theta}_i)\mathcal{T}_i(\boldsymbol{\theta}_i \to \boldsymbol{\theta}_{i-1}) \tag{23}$$

where $\mathcal{T}_i(\cdot \to \cdot)$ is the transition kernel at i-th proposal.

Integrating out $\boldsymbol{\theta}_{i-1}$ on both sides, we obtain that

$$\begin{aligned}
\pi(\boldsymbol{\theta}_i) &= \int \pi(\boldsymbol{\theta}_i)\mathcal{T}_i(\boldsymbol{\theta}_i \to \boldsymbol{\theta}_{i-1})d\boldsymbol{\theta}_{i-1} \\
&= \int \pi(\boldsymbol{\theta}_{i-1})\mathcal{T}_i(\boldsymbol{\theta}_{i-1} \to \boldsymbol{\theta}_i)d\boldsymbol{\theta}_{i-1}
\end{aligned} \tag{24}$$

That is, distribution π is a stationary distribution with a single transition kernel $\mathcal{T}_i$, i.e.,

$$\mathcal{T}_i(\pi) = \pi \tag{25}$$

holds for every i.

Thus,

$$\mathcal{T}_n\mathcal{T}_{n-1}...\mathcal{T}_1(\pi) = \pi \tag{26}$$

Proved. □

References

Ahn, S.; Korattikara, A.; and Welling, M. 2012. Bayesian posterior sampling via stochastic gradient fisher scoring. *arXiv preprint arXiv:1206.6380*.

Ahn, S.; Shahbaba, B.; and Welling, M. 2014. Distributed stochastic gradient mcmc. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 1044–1052.

Bardenet, R.; Doucet, A.; and Holmes, C. 2014. Towards scaling up markov chain monte carlo: an adaptive subsampling approach. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 405–413.

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

Chao, W.-L.; Solomon, J.; Michels, D. L.; and Sha, F. 2015. Exponential integration for hamiltonian monte carlo. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, 1142–01151.

Chen, T.; Fox, E. B.; and Guestrin, C. 2014. Stochastic gradient hamiltonian monte carlo. *arXiv preprint arXiv:1402.4102*.

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

Gilks, W. R.; Richardson, S.; and Spiegelhalter, D. J. 1996. Introducing markov chain monte carlo. *Markov chain Monte Carlo in practice* 1:19.

Girolami, M., and Calderhead, B. 2011. Riemann manifold langevin and hamiltonian monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 73(2):123–214.

Kadanoff, L. P. 2000. *Statistical physics: statics, dynamics and renormalization*. World Scientific.

Korattikara, A.; Chen, Y.; and Welling, M. 2013. Austerity in mcmc land: Cutting the metropolis-hastings budget. *arXiv preprint arXiv:1304.5299*.

Leimkuhler, B., and Reich, S. 2004. *Simulating hamiltonian dynamics*, volume 14. Cambridge University Press.

Ma, Y.-A.; Chen, T.; and Fox, E. 2015. A complete recipe for stochastic gradient mcmc. In *Advances in Neural Information Processing Systems*.

Maclaurin, D., and Adams, R. P. 2015. Firefly monte carlo: Exact mcmc with subsets of data. In *Proceedings of the 24th International Conference on Artificial Intelligence*, 4289–4295. AAAI Press.

Neal, R. M. 2011. Mcmc using hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo* 2.

Nocedal, J., and Wright, S. 2006. *Numerical optimization*. Springer Science & Business Media.

Patterson, S., and Teh, Y. W. 2013. Stochastic gradient riemannian langevin dynamics on the probability simplex. In *Advances in Neural Information Processing Systems*, 3102–3110.

Qian, H. 2012. A hamiltonian-entropy production connection in the skew-symmetric part of a stochastic dynamics. *arXiv preprint arXiv:1205.6552*.

Salakhutdinov, R., and Mnih, A. 2008. Bayesian probabilistic matrix factorization using mcmc. *ICML08*.

Wang, Z.; Mohamed, S.; and Nando, D. 2013. Adaptive hamiltonian and riemann manifold monte carlo. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, 1462–1470.

Welling, M., and Teh, Y. W. 2011. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 681–688.

Zhang, Y., and Sutton, C. A. 2011. Quasi-newton methods for markov chain monte carlo. In *Advances in Neural Information Processing Systems*, 2393–2401.