Hamiltonian Sequential Monte Carlo

Svetoslav Kostov* Nick Whiteley†

Abstract

We present the basics of a new sampling algorithm - Hamiltonian Sequential Monte Carlo (HSMC), which combines ideas from Hamiltonian Monte Carlo and Sequential Monte Carlo, allowing us to move from an initial, easy-to-sample-from distribution, to the distribution of interest via a sequence of intermediate distributions. The algorithm produces a sample from the desired distribution, as well as an estimate of the ratio of the normalizing constants of the final and the initial distributions. We show that for a particular choice of the transition kernels, the HSMC algorithm performs better in terms of mean squared error of the estimate of the ratio of the normalizing constants, compared to other standard algorithms. This is achieved through bias-variance trade off. We discuss some of the properties of the new algorithm and present simulation results for couple of toy examples, as well as for a 20-dimensional linear regression, where we estimate the Bayes factor for two competing models.

Key Words: Simulation, Normalizing Constants, Bias-variance trade off, Bayes factor, Sequential Monte Carlo, Hamiltonian Monte Carlo

1. Introduction and Motivation

Sampling from high-dimensional, multimodal distributions proved to be a really difficult task using standard Monte Carlo methods. Some more advanced methods like Sequential Monte Carlo (SMC), introduced in Gordon et al. [8] show big potential to complete this task. In recent years a relatively new method - Hamiltonian Monte Carlo (HMC), introduced first by Duane et al. [5] become popular among the statistical community as a sampling algorithm, which partially alleviates the random walk behavior of the standard, random - walk type of MCMC algorithms. In our work we combine ideas from SMC and HMC to get a new sampling algorithm, which we call Hamiltonian Sequential Monte Carlo (HSMC). This new algorithm will produce approximate samples from a target distribution, while also producing an estimate of the ratio of normalizing constants for this target distribution and an arbitrary initial one.

Standard HMC has problems in exploring multiple modes of a distribution, because its inherent tendency to propose values, which lie on a single contour of constant energy, i.e. of constant value of the Hamiltonian H. In contrast, because it is based on SMC, the HSMC algorithm proves to deal well with sampling from multimodal distributions. On the other hand, because it is based on HMC, it also benefits from the ability to make distant, long-range proposals. Another novelty is the fact, that we introduce a natural parameter which controls how much noise we use during the simulation process. This allows us to better control the MSE of the estimate of the ratio of the normalizing constants that we obtain.

*School of Mathematics, University of Bristol, University Walk, Bristol, BS8 1TW, UK
†School of Mathematics, University of Bristol, University Walk, Bristol, BS8 1TW, UK

2. HSMC algorithm

With a continuous, non-decreasing "schedule" function $\tau : [0,1] \to [0,1]$, such that $\tau(0) = 0$ and $\tau(1) = 1$, and for some $d \geq 1$ a family of functions $\{U_\tau; \tau \in [0,1]\}$ each $U_\tau : \mathbb{R}^d \to \mathbb{R}$ and such that $\int_{\mathbb{R}^d} \exp\left[-U_\tau(q)\right] dq < \infty$, we consider a family of unnormalized probability densities $\{\bar{\pi}_\tau; \tau \in [0,1]\}$ with each $\bar{\pi}_\tau : \mathbb{R}^{2d} \to \mathbb{R}_+$ given by

$$\begin{aligned} \bar{\pi}_\tau(q,p) &:= \exp\left[-H_\tau(q,p)\right], & (1) \\ H_\tau(q,p) &:= \frac{p^T M^{-1} p}{2} + U_\tau(q) & (2) \end{aligned}$$

where $p = (p_1, ..., p_d)$ and $M > 0$ is a positive-semidefinite matrix, called the mass matrix (except when we explicitly state different, we will assume, that M is a constant matrix).

We shall regard the potential energy is equal to the minus logarithm of some probability density on $\mathbb{R}^d$, $f_\tau(q) = \exp[-U_\tau(q)] / \int_{\mathbb{R}^d} \exp[-U_\tau(q')] dq'$. With $x = (q,p)$ we shall write the normalized version of each $\bar{\pi}_\tau$ as π_τ, i.e., $\pi_\tau(x) = \bar{\pi}_\tau(x)/Z_\tau$, $Z_\tau = \int_{\mathbb{R}^{2d}} \bar{\pi}_\tau(x) dx$. Hence, because of the definition in Equation (2), we can write $\pi_\tau(x) = \mu(p) f_\tau(q)$, where $\mu(p)$ is the density of a normal distribution $\mathcal{N}(0, M)$. We shall sometimes write also the probability measure $\pi_\tau(dx) = \pi_\tau(x)dx$, where dx is Lebesgue measure on $\mathbb{R}^{2d}$.

Again with $x = (q,p)$, denote:

$$\begin{aligned} G_{n,k-1}(x) &= \frac{\bar{\pi}_{\tau(k/n)}(x)}{\bar{\pi}_{\tau((k-1)/n)}(x)} \\ &= \exp[-H_{\tau(k/n)}(q) + H_{\tau((k-1)/n)}(q)] \in (0, +\infty), \quad k = 1, ..., n. \end{aligned} \tag{3}$$

We assume throughout that for each $n \geq 1$ and $k = 0, ..., n-1$,

$$c_{n,k} := \sup_x G_{n,k}(x) < +\infty.$$

Let $\{L_\tau; \tau \in [0,1]\}$, $\{M_\tau; \tau \in [0,1]\}$ be two families of Markov kernels, where each of these kernels is defined on $\mathbb{R}^{2d}$. We will specify the exact functional form of these kernels later. Then for each $n \geq 1$, $k = 1, ..., n$ and η a probability measure on $\mathbb{R}^{2d}$, we define the Markov kernel $K^\eta_{n,k}(x, dx')$ as:

$$\begin{aligned} K^\eta_{n,k}(x, dx') : &= \frac{1}{c_{n,k-1}} G_{n,k-1}(x) L_{\tau(k/n)}(x, dx') \\ &+ \left[1 - \frac{1}{c_{n,k-1}} G_{n,k-1}(x)\right] \\ &\times \int \Psi_{n,k-1}(\eta)(d\zeta) M_{\tau(k/n)}(\zeta, dx'), \end{aligned} \tag{4}$$

where $\Psi_{n,k}$ maps probability measures to probability measures, according to

$$\Psi_{n,k}(\eta)(\varphi) := \frac{\eta(G_{n,k}\varphi)}{\eta(G_{n,k})}, \quad k = 0, ..., n-1. \tag{5}$$

where φ is a bounded test function, and where we use the notation $\eta(f) = \int_{\mathbb{R}^d} f(x)\eta(dx)$ for arbitrary function f. As we shall see in more detail later, the Markov kernels $K^{\pi_{\tau((k-1)/n)}}_{n,k}$

can be used to describe the evolution of the probability distributions $\pi_{\tau((k-1)/n)}$, and inspires the following particle approximation.

With $n \geq 1$ fixed, we will introduce a sequence of generations of particles $\{\xi_{n,k}\}_{k=0}^{n}$, where $\xi_{n,k} = \left\{\xi_{n,k}^i\right\}_{i=1}^{N}$ and each $\xi_{n,k}^i$ is valued in $\mathbb{R}^{2d}$. We will assume that we have already obtained an empirical distribution, $\eta_{n,k-1}^N(dx) := N^{-1}\sum_{i=1}^N \delta_{\xi_{n,k-1}^i}(dx)$ which is an approximation of $\pi_{\tau((k-1)/n)}(dx)$. Here, $\delta_a(dx)$ denotes the Dirac point measure.

We define the HSMC algorithm explicitly in Algorithm 1. Note, that to estimate the $c_{n,k-1}$ coefficient, we have to calculate the empirical maximum

$$c_{n,k-1}^N = \max_{\xi_{n,k-1}} G_{n,k-1}(\xi_{n,k-1}^i)$$

and we note that:

Algorithm 1 Hamiltonian SMC: a generic algorithm

Fix $n \geq 1$

For $k = 0$,

sample $\left\{\xi_{n,0}^i\right\}_{i=1}^N$ iid according to $\pi_{\tau(0)} := \pi_0$

For $k = 1, ..., n$

for $i = 1, ..., N$

with probability $\dfrac{1}{c_{n,k-1}^N} G_{n,k-1}(\xi_{n,k-1}^i)$, sample

$$\xi_{n,k}^i \sim L_{\tau(k/n)}(\xi_{n,k-1}^i, \cdot)$$

otherwise sample

$$\xi_{n,k}^i \sim \frac{\sum_{j=1}^N G_{n,k-1}(\xi_{n,k-1}^i) M_{\tau(k/n)}\left(\xi_{n,k-1}^j, \cdot\right)}{\sum_{j=1}^N G_{n,k-1}(\xi_{n,k-1}^i)}$$

$$\int \Psi_{n,k-1}(\eta_{n,k-1}^N)(dx) M_{\tau(k/n)}(x, \cdot) = \frac{\sum_{j=1}^N G_{n,k-1}(\xi_{n,k-1}^i) M_{\tau(k/n)}\left(\xi_{n,k-1}^j, \cdot\right)}{\sum_{j=1}^N G_{n,k-1}(\xi_{n,k-1}^i)}.$$

The "otherwise" step in Algorithm 1 amounts to multinomial resampling, followed by sampling according to the $M_{\tau(k/n)}$ kernel.

2.1 The $L_{\tau(k/n)}$ kernel

Our idea is to build $L_{\tau(k/n)}(x, dx')$ using the Hamiltonian dynamics, associated with $H_{\tau(k/n)}$. This Hamiltonian flow is a continuous map that gives the time evolution of (q, p) through the solution of the system of Hamiltonian equations. For more details about Hamiltonian Monte Carlo and Hamiltonian mechanics itself we will refer the reader to [5], [1], [11] and [13]. In practice, we cannot integrate the Hamiltonian equations exactly, so an approximation using a numerical integrator is needed. We will use the Störmer-Verlet (or leapfrog) integrator - for more details we refer the reader to [10].

We will denote by $\phi_{\tau(k/n)}(x)$ the discretized Hamiltonian flow that we obtain using the leapfrog integrator for a dynamics, associated with $H_{\tau(k/n)}(x)$. We note here, that the leapfrog integrator depends on the discretization parameters - the step size ϵ and the number of steps l, but we do not include that explicitly in our notation for $\phi_{\tau(k/n)}(x)$. We also introduce the so called momentum flip operator $F : \mathbb{R}^{2d} \to \mathbb{R}^{2d}$ as $F : (q, p) \mapsto (q, -p)$. We also use the notation $a \wedge b = \min(a, b)$. We have summarized the sampling according to $L_{\tau(k/n)}$ in Algorithm 2.

Algorithm 2 Sampling according to $L_{\tau(k/n)}(x, dx')$

1. With current state x, calculate $\tilde{x} = \phi_{\tau(k/n)}(x)$
2. With probability $1 \wedge \frac{\pi_{\tau(k/n)}(\tilde{x})}{\pi_{\tau(k/n)}(x)}$, set $x' = \tilde{x}$, otherwise set $x' = Fx$.

To put it in words, the sampling according to the $L_{\tau(k/n)}$ kernel consists of several steps. First, we apply the leapfrog integrator to the current particle $\xi^i_{n,k-1}$ and we obtain a new value for the particle $\xi'^i_{n,k-1}$. After that, with probability $1 \wedge \frac{\pi_{\tau(k/n)}(\xi'^i_{n,k-1})}{\pi_{\tau(k/n)}(\xi^i_{n,k-1})}$, we set the newly proposed particle to be equal to $\xi'^i_{n,k-1}$. Otherwise, we set the new particle to be equal to the old particle $\xi^i_{n,k-1}$, but with flipped second component (momentum).

We can write an explicit expression for the kernel $L_{\tau(k/n)}$ as the composition of F and $P_{\tau(k/n)}(x, dx')$, where

$$
\begin{aligned}
L_{\tau(k/n)}(x, dx') &= F P_{\tau(k/n)}(x, dx') && (6)\\
P_{\tau(k/n)}(x, dx') &= \left[1 \wedge \frac{\pi_{\tau(k/n)}(F\phi_{\tau(k/n)}(x))}{\pi_{\tau(k/n)}(x)}\right] \delta_{F\phi_{\tau(k/n)}(x)}(dx') && (7)\\
&+ \left[1 - \left(1 \wedge \frac{\pi_{\tau(k/n)}(F\phi_{\tau(k/n)}(x))}{\pi_{\tau(k/n)}(x)}\right)\right] \delta_x(dx')
\end{aligned}
$$

Notice, that there is no randomness in the proposals that we produce using $F\phi_{\tau(k/n)}(x)$. The $P_{\tau(k/n)}$ kernel is a special case of the more general Metropolis Hastings proposal kernel with deterministic proposals - see [14] and [15] for more details on deterministic proposals in MCMC. This is important to be stressed, as it will turn out that this lack of randomness will play significant role in the explanation of the performance of the HSMC algorithm later.

2.2 The $M_{\tau(k/n)}$ kernel

We will define the $M_{\tau(k/n)}(x, dx')$ kernel to be

$$
\begin{aligned}
M_{\tau(k/n)}(x, dx') &= M_{\tau(k/n)}((q, p), d(q', p')) = \mu(dp')\delta_q(dq') && (8)\\
&= \frac{1}{(2\pi)^{d/2}\det(M)} \exp\left(-\frac{1}{2} p'^T M^{-1} p'\right) dp'\ \delta_q(dq')
\end{aligned}
$$

Here we denote with $dp' = d(p'_1, ..., p'_d)$ the d - dimensional Lebesgue measure. Let us explain what is the definition of $M_{\tau(k/n)}(x, dx')$ essentially saying. To sample according to $M_{\tau(k/n)}(x, dx')$, we start with a particle $\xi^i_{n,k-1}$ - we keep the value of the q - component of

$\xi^i_{n,k-1}$ intact, whereas at the same time we sample new value p' for the momentum (second component) of the particle from a normal distribution $\mathcal{N}(0, M)$. Combining these two, we obtain the newly proposed particle according to $M_{\tau(k/n)}$.

2.3 Estimate of the ratio of the normalizing constants

We can obtain the estimate of the ratio of the normalizing constants Z_1/Z_0 of the initial $\pi_{\tau(0)}$ and the final distribution $\pi_{\tau(1)}$ by calculating

$$\widehat{\frac{Z_1}{Z_0}} = \prod_{k=0}^{n-1} \eta^N_{n,k}\left(G_{n,k}\right) \tag{9}$$

where $\left\{\eta^N_{n,k}\right\}_{k=0}^{n-1}$ is the sequence of empirical measures that we have obtained by running the HSMC algorithm. The expression in Equation (9) is motivated by the identity:

$$\frac{Z_1}{Z_0} = \prod_{k=0}^{n-1} \pi_{\tau(k/n)}\left(G_{n,k}\right), \tag{10}$$

the equality following from the definitions of G_n, k and $\pi_{\tau(k/n)}$. For further discussion of (9) the reader is referred to [4] and [3] for more details.

2.4 Comments

We can clearly see, that Algorithm 1 is a generalization of the ϵ - algorithm, defined by Del Moral in Section 7.2.1 of Chapter 7 in [3]. The ϵ - algorithm samples a sequence of generations of particles $\{\xi_{n,0}, \xi_{n,1}, \ldots, \xi_{n,n}\}$ as in the case of the HSMC algorithm, but it does that according to the kernel, described in Equation (4), where we take $L_{\tau(k/n)}(x, dy) = M_{\tau(k/n)}(x, dy) = R_{\tau(k/n)}(x, dy)$, where $R_{\tau(k/n)}$ is some proposal kernel. It is important to note, that the ϵ - algorithm produces unbiassed estimate of the ratio of the normalizing constants, as defined in Equation (9). We refer the reader to [3] for more details about the ϵ - algorithm and its properties.

In the case of HSMC, the introduction of second Markov kernel $L_{\tau(k/n)}(x, dy)$ helps us to "separate" the standard HMC proposal scheme into two. By doing this separation we would like to decrease the noise in the estimate of Z_1/Z_0, defined in Equation (9). In contrast to the ϵ - algorithm, the HSMC produces biased estimate of Z_1/Z_0, which is a direct consequence of the introduction of second kernel.

3. Why HSMC works?

In this section we would like to discuss the foundations of the HSMC algorithm. We will give some reasoning behind the algorithm. To start, we will observe the following important property, defined as a Lemma

Lemma 1. *For any $n \geq 1$ and $1 \leq k \leq n$, $\Psi_{n,k-1}(\pi_{\tau((k-1)/n)}) = \pi_{\tau(k/n)}$, where $\Psi_{n,k-1}$ is defined in Equation* (5)*, and the sequence of distributions $\pi_{\tau(k/n)}$ is the same, as the one, defined already in Equation* (1)*.*

Proof. From (3) and (5), we have for the action of $\Psi_{n,k-1}$ on $\pi_{\tau((k-1)/n)}$

$$\begin{aligned}
&\Psi_{n,k-1}(\pi_{\tau((k-1)/n)})(\varphi) \\
&= \frac{\pi_{\tau((k-1)/n)}(G_{n,k-1}\varphi)}{\pi_{\tau((k-1)/n)}(G_{n,k-1})} \\
&= \frac{\int \pi_{\tau((k-1)/n)}(x)\frac{\bar{\pi}_{\tau(k/n)}(x)}{\bar{\pi}_{\tau((k-1)/n)}(x)}\varphi(x)dx}{\int \pi_{\tau((k-1)/n)}(x)\frac{\bar{\pi}_{\tau(k/n)}(x)}{\bar{\pi}_{\tau((k-1)/n)}(x)}dx} \\
&= \frac{\int \pi_{\tau((k-1)/n)}(x)\frac{Z_{\tau(k/n)}}{Z_{\tau((k-1)/n)}}\frac{\pi_{\tau(k/n)}(x)}{\pi_{\tau((k-1)/n)}(x)}\varphi(x)dx}{\int \pi_{\tau((k-1)/n)}(x)\frac{Z_{\tau(k/n)}}{Z_{\tau((k-1)/n)}}\frac{\pi_{\tau(k/n)}(x)}{\pi_{\tau((k-1)/n)}(x)}dx} \\
&= \int \pi_{\tau((k-1)/n)}(x)\frac{\pi_{\tau(k/n)}(x)}{\pi_{\tau((k-1)/n)}(x)}\varphi(x)dx \\
&= \pi_{\tau(k/n)}(\varphi).
\end{aligned}$$

which is exactly what we wanted to show in the first place. □

What Lemma 1 is essentially saying, is that the $\Psi_{n,k-1}$ operator maps the distribution $\pi_{\tau((k-1)/n)}$ into the distribution $\pi_{\tau(k/n)}$. This Lemma is at the basis of the ϵ - algorithm as well, so we refer the reader to [3] for more details.

To be able to justify the HSMC algorithm we will need also an auxiliary result from [15], namely Special case 2 of Theorem 2, which we state in a Lemma

Lemma 2. *Let ψ be a one-to-one transformation from space E to the same space E, i.e. $\psi : E \to E$ and let ψ is such, that $\psi^{-1} = \psi$ (i.e. ψ is an involution). Let us have a Metropolis - Hastings transition kernel $T(x, dy) : E \times \mathcal{E} \to [0, 1]$, a proposal kernel $\delta_{\psi(x)}(dy)$ and a target distribution with density $\pi(x)$, where*

$$\begin{aligned}
T(x, dy) &= \alpha(x, y)\,\delta_{\psi(x)}(dy) + (1 - a(x))\,\delta_x(dy) && (11)\\
a(x) &= \alpha(x, \psi(x)) \\
\alpha(x, y) &= 1 \wedge \frac{\pi(\psi(x))}{\pi(x)} && (12)
\end{aligned}$$

Then if the current state is x, and the proposed state y is equal to $\psi(x)$, then the combination of transition kernel $T(x, dy)$ and the target distribution $\pi(x)$ satisfy the detailed balance equation.

Proof. See the proof of Theorem 2 in Tierney's paper [15]. □

The result in Lemma 2 is important part of the proof, that $L_{\tau(k/n)}$ and $M_{\tau(k/n)}$ leave $\pi_{\tau(k/n)}$ invariant. We are stating this in the following

Lemma 3. *For each $\tau \in [0, 1]$, $L_{\tau(k/n)}$ and $M_{\tau(k/n)}$ both admit $\pi_{\tau(k/n)}$ as an invariant distribution.*

Proof. First we will prove, that the $L_{\tau(k/n)}$ kernel leaves $\pi_{\tau(k/n)}$ invariant. To do this, we will first show, that the composition of the flip operator F and the leapfrog integrator $\phi_{\tau(k/n)}$ is indeed an involution. First, F is an involution, which is obvious from its definition - we

have, that $F^2(x) = F(F(q,p)) = F(q,-p) = x$, or $F = F^{-1}$. Moreover, F leaves $\pi_{\tau(k/n)}$ invariant, because $F\pi_{\tau(k/n)}(x) = \pi_{\tau(k/n)}(q,-p) = \pi_{\tau(k/n)}(x)$, because of Equation (2).

Then, we will also use the standard fact, that the Hamiltonian dynamics and the leapfrog integrator, associated with $H_{\tau(k/n)}$, as defined in Equation (2), are time reversible - in other words, that the inverse map $\phi^{-1}_{\tau(k/n)}$ is obtained by first negating the momentum p, then applying $\pi_{\tau(k/n)}$ and after that negating the momentum again, i.e. $\phi^{-1}_{\tau(k/n)} = F\phi_{\tau(k/n)}F$. We refer the reader to [13] for more details about the time reversibility property. From the reversibility and the fact, that F is an involution, we can easily see, why the composition $F\phi_{\tau(k/n)}$ is an involution.

Knowing this, and using Lemma 2, we see, that $P_{\tau(k/n)}(x, dx')$, as defined in Section 2.1, and $\pi_{\tau(k/n)}$ satisfy the detailed balance equation. This automatically means, that $P_{\tau(k/n)}(x, dx')$ also leaves $\pi_{\tau(k/n)}$ invariant. Because $L_{\tau(k/n)}(x, dx') = FP_{\tau(k/n)}(x, dx')$, we easily see, that $L_{\tau(k/n)}$ also leaves $\pi_{\tau(k/n)}$ invariant.

To prove, that $M_{\tau(k/n)}$ leaves $\pi_{\tau(k/n)}$ invariant, we make the following simple calculation

$$\begin{aligned}
(\pi_{\tau(k/n)}M_{\tau(k/n)})(d(q',p')) &= \int_{\mathbb{R}^d}\int_{\mathbb{R}^d} \pi_{\tau(k/n)}(d(q,p))M_{\tau(k/n)}((q,p),d(q',p'))) \\
&= \int_{\mathbb{R}^d}\int_{\mathbb{R}^d} f_{\tau(k/n)}(dq)\mu(dp)\delta_q(dq')\mu(dp') \\
&= \int_{\mathbb{R}^d} f_{\tau(k/n)}(dq)\delta_q(dq')\mu(dp') \\
&= f_{\tau(k/n)}(dq')\mu(dp') = \pi_{\tau(k/n)}(d(q',p'))
\end{aligned}$$

which is exactly what we wanted to show. □

Having Lemma 3 and Lemma 1 in hand, we can now prove the following Lemma, which will be used to justify the basis of the definition of the HSMC algorithm.

Lemma 4. *We have for any* $n \geq 1$ *and* $1 \leq k \leq n$,

$$\begin{aligned}
\int \pi_{\tau((k-1)/n)}(x)K_{n,k}^{\pi_{\tau((k-1)/n)}}(x,A)dx &= \int_A \pi_{\tau(k/n)}(x)dx, \\
&\qquad k = 1,...,n, \quad A \in \mathcal{B}(\mathbb{R}^{2d}).
\end{aligned}$$

Proof. We have

$$\begin{aligned}
& \int \pi_{\tau((k-1)/n)}(x)K_{n,k}^{\pi_{\tau((k-1)/n)}}(x,A)dx \\
= & \int \pi_{\tau((k-1)/n)}(x)\frac{1}{c_{n,k-1}}\frac{Z_{\tau(k/n)}}{Z_{\tau((k-1)/n)}}\frac{\pi_{\tau(k/n)}(x)}{\pi_{\tau((k-1)/n)}(x)}L_{\tau(k/n)}(x,A)dx \\
& + \left[1 - \int \pi_{\tau((k-1)/n)}(x)\frac{1}{c_{n,k-1}}\frac{Z_{\tau(k/n)}}{Z_{\tau((k-1)/n)}}\frac{\pi_{\tau(k/n)}(x)}{\pi_{\tau((k-1)/n)}(x)}dx\right] \\
& \int \Psi_{n,k-1}(\pi_{\tau((k-1)/n)})(d\zeta)M_{\tau(k/n)}(\zeta,A) \\
= & \frac{1}{c_{n,k-1}}\frac{Z_{\tau(k/n)}}{Z_{\tau((k-1)/n)}}\int \pi_{\tau(k/n)}(x)L_{\tau(k/n)}(x,A)dx
\end{aligned}$$

$$
\begin{aligned}
&+\left[1-\int \frac{1}{c_{n,k-1}}\frac{Z_{\tau(k/n)}}{Z_{\tau((k-1)/n)}}\pi_{\tau(k/n)}(x)dx\right]\\
&\int \Psi_{n,k-1}(\pi_{\tau((k-1)/n)})(d\zeta)M_{\tau(k/n)}(\zeta,A)\\
=\;& \frac{1}{c_{n,k-1}}\frac{Z_{\tau(k/n)}}{Z_{\tau((k-1)/n)}}\int_A \pi_{\tau(k/n)}(x)dx\\
&+\left[1-\frac{1}{c_{n,k-1}}\frac{Z_{\tau(k/n)}}{Z_{\tau((k-1)/n)}}\right]\int_A \pi_{\tau(k/n)}(x)dx\\
=\;& \int_A \pi_{\tau(k/n)}(x)dx,
\end{aligned}
$$

where for the penultimate equality, Lemma 3 and Lemma 1 have been used. □

Lemma 4 says, that if we apply the Markov kernel $K_{n,k}^{\pi_{\tau((k-1)/n)}}$ to $\pi_{\tau((k-1)/n)}$ we will get $\pi_{\tau(k/n)}$. This is the basis of the HSMC algorithm - we see, that starting from an initial distribution $\pi_{\tau(0)}$, we can move from the $k-1$ - th to the k - the distribution in the sequence by applying $K_{n,k}^{\pi_{\tau((k-1)/n)}}$ kernel. As for the approximating empirical measures $\eta_{n,k}^N$, Lemma 4 suggests to us that if $\eta_{n,k-1}^N$ is a "good" approximation of $\pi_{\tau((k-1)/n)}$, then $\eta_{n,k}^N := N^{-1}\sum_{i=1}^N \delta_{\xi_{n,k}^i}$ should be a "good" approximation of the next distribution in the sequence, $\pi_{\tau(k/n)}$.

4. Numerical results

In this section we are going to present three numerical examples. In the first example we will present simulations for the HSMC algorithm, when applied in a really simple, two - dimensional setting. We will compare the performance of the HSMC algorithm with the performance of the more standard, trivial algorithm, which we call $L = M$ - algorithm. This $L = M$ - algorithm is just the standard ϵ - algorithm (as defined in Del Moral in [3]), but in the case, where we choose both kernels L and M to be equal to the standard Hamiltonian Monte Carlo transition kernel, i.e. the transition kernel consists of drawing a momentum p from a Gaussian distribution $\mathcal{N}(0, M)$, where M is the mass matrix; then we propose new value for the position q and the momentum p with the help of the leapfrog integrator; and after that we make Metropolis - Hastings accept - reject step. We will compare both algorithms in terms of the bias and the MSE of the estimate of the ratios of the normalizing constants produced.

With the second numerical example we will investigate the dependence of the MSE of the estimate of the ratio of the normalizing constants that we obtain with the HSMC algorithm on the choice of the mass matrix. We will compare this dependence to the dependence for the more standard $L = M$ algorithm.

In the third, more realistic example, we will compare the performance of the HSMC and the Annealed Importance Sampling (AIS) algorithm, introduced by Neal in [12]. The example will be a 3 and a 20 dimensional heteroscedastic regressions. We will estimate the Bayes factor for the two sub - examples with the help of the HSMC and AIS algorithms and compare them in terms of accuracy and variability of the estimates.

We note, that in all numerical examples, we use linear schedule function $\tau(k/n) = k/n$, where $n \geq 1$ and $0 \leq k \leq n$. We also assume, that throughout all of the examples we

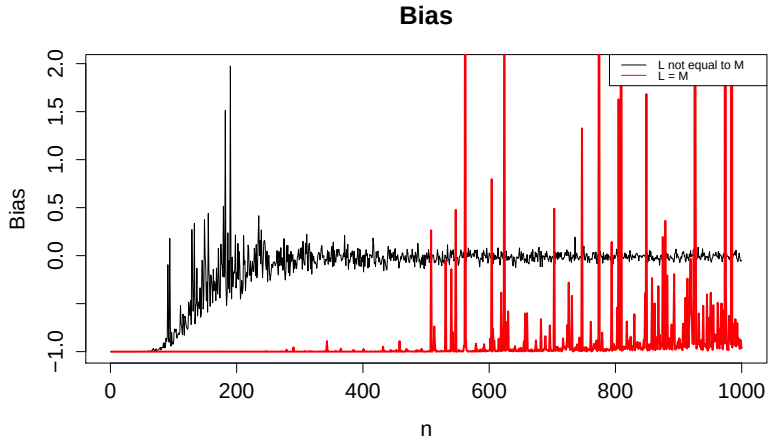

Figure 1: Bias of the estimate of the ratio of the normalizing constants for the cases where $L = M =$ standard HMC, and where $L \neq M$, and L as defined in Section 2.1, and M as defined in Section 2.2.

have the following expression for the sequence of densities, with given initial f_0 and final distributions f_1: $f_{\tau(k/n)} = f_0^{1-\tau(k/n)} f_1^{\tau(k/n)}$.

4.1 Toy example - bias and MSE

In our first toy example, we are going to investigate the performance of the HSMC algorithm, in the following setup - π_1 is a bimodal normal distribution, and π_0 is an unimodal normal distribution. The target bimodal distribution has two clearly separated modes. We will compare the performance of the HSMC (which we will also call $L \neq M$ algorithm) and the performance of the $L = M$ algorithm, which we have already defined.

The exact details of the example are as follows - let us suppose that we have

$$f_0 \sim \mathcal{N}(0, \Sigma_0); \quad f_1 \sim \omega_1 \mathcal{N}(\mu_1, \Sigma_1) + \omega_2 \mathcal{N}(\mu_2, \Sigma_2)$$

where in our particular we have $\mu_1 = (10, 5)$, $\mu_2 = (-10, 5)$, $\mu_0 = (0, 0)$, $\Sigma_0 = \Sigma_1 = \Sigma_2 = I$ and $\omega = (0.5, 0.5)$. The initial and final distributions f_0 and f_1 correspond to the initial and final distributions in the sequence of distributions, defined in Section 2. For this example we will have for the parameters of the leapfrog discretization (refer to [10]) - for the stepsize $\epsilon = 0.1$, and for the number of steps $l = 1$. We will run the both algorithms $m = 100$ times with $N = 10$ particles.

In Figure 1 we plot the bias of the estimate of the ratio of the normalizing constants as a function of the number of intermediate distributions n for both algorithms. We can clearly see, that the estimate of the ratio that we obtain using the HSMC algorithm (the black line) has much less variability and is much more accurate even for small values of n.

We see also, that the standard combination of HMC and SMC (the $L = M$ algorithm, or the red line at the plot) performs really poorly in terms of the variability of the estimate of the ratio. We see that even for large values of n, the $L = M$ algorithm could not produce a reliable and accurate estimate of the ratio. In Figure 2 we plot the mean squared error of the estimate of the ratio for both algorithms. We again see easily, that the HSMC outperforms

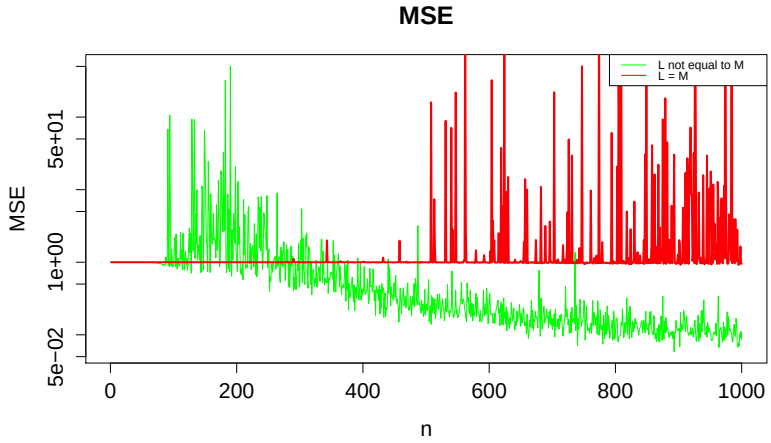

Figure 2: MSE of the estimate of the ratio of the normalizing constants for the cases where $L = M =$ standard HMC, and where $L \neq M$, and L as defined in Section 2.1, and M as defined in Section 2.2.

the $L = M$ algorithm in terms of MSE for large values of n. Because the example that we have is really extreme, we see, that both algorithms perform poorly for small values of n. This is due to the fact, that the initial and final distributions are so different, that the estimate of the ratio that we obtain for both algorithms is close to zero (the particles that we propagate could not keep up with the change in the sequence of distributions, in other words we are far from equilibrium).

4.2 Mass matrix dependence

In this section, we will investigate empirically an interesting property of the HSMC algorithm. We know, that the choice of the mass matrix M is crucial when it comes to the performance of the standard HMC algorithm - see for instance [11] or [13]. A major attempt to cope with this problem was done by the authors of [7], where they introduce the Riemannian Manifold HMC (RMHMC). In this paper, the authors introduce a mass matrix, which is a function of the geometry of the statistical inference problem we have in hand. This choice has several benefits - first, we save time, because we do not have to tune the mass matrix by hand. Secondly, including information about the geometry of the target distribution could help us improve the performance of the standard HMC algorithm. The major drawbacks of the RMHMC are that it involves costly derivatives / matrix computations, as well as it is still a classical MCMC based simulation algorithm, so it is not suitable for sampling of multimodal distributions.

It turns out, that the HSMC deals with the problem of choice of the mass matrix in a very natural way. Our numerical investigations show, that for large number of intermediate distributions n, the HSMC algorithm seems to be less sensitive to the choice of the mass matrix. This corresponds with our intuition - when we increase n, we are sampling less and less according to the $M_{\tau(k/n)}$ kernel, which depends on the mass matrix M.

Let us define the mass matrix as a multiple of the identity, i.e. $M = aI$, where $a \in \mathbb{R}^+$. In this example we also fix $n = 10000$, $N = 10$, $m = 100$, $\epsilon = 0.1$ and $l = 1$. In Figure 3 we plot the dependence of the HSMC and $L = M$ algorithms on the mass matrix scale

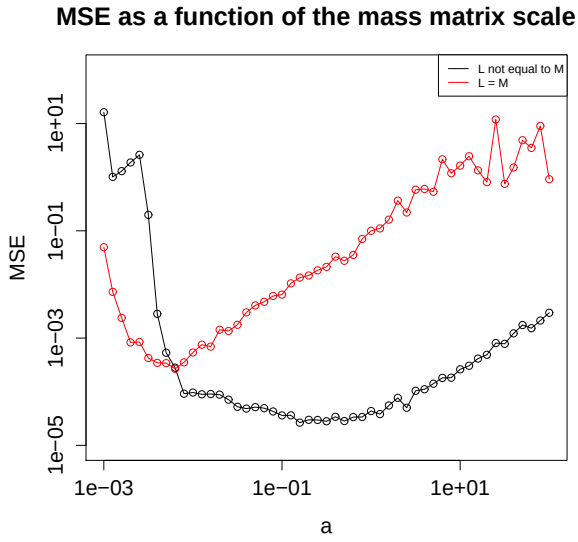

Figure 3: Dependence on the scale of the mass matrix

parameter a (both the x and y axis are on a log - scale). The example we are using now is simpler than the one, we used in the previous section - we have a unimodal initial and unimodal final distributions, described as

$$f_0 \sim \mathcal{N}(\mu_0, \Sigma_0); \quad f_1 \sim \mathcal{N}(\mu_1, \Sigma_1)$$

where we have $\mu_1 = (3, 3)$, $\mu_0 = (0, 0)$ and $\Sigma_0 = \Sigma_1 = I$.

We clearly see, that the scale of the mass matrix has almost no effect over the MSE of the estimate of the ratio of the normalizing constants that we obtain from the HSMC algorithm (the black line and points). On the other hand, we can see that MSE of the estimate of the ratio obtained from the $L = M$ (the red line and points on the plot) algorithm depends strongly on the choice of a. We also see that the HSMC algorithm outperforms the $L = M$ algorithm (around 10x) even for the best possible choice of scale parameter a.

4.3 HSMC for Bayes factor estimation

In this more real - world example, we are going to test the performance of the HSMC algorithm on a linear regression problem, in both low and high - dimensional cases. The example that we are going to use is the heteroscedastic linear regression example from [6] and [9]. The regression model could be written as

$$Y \mid \beta, \sigma, r \sim N\left(X\beta, \sigma^2 r^{-\theta}\right)$$

where r is a vector of weights, and $\theta \in [0, 1]$ is a model parameter, that controls the heteroscedasticity of the generated sample. If $\theta = 0$, we have a regular, homoscedastic regression, whereas for all other values of θ the model has some degree of heteroscedasticity, which depends on the weights r.

In the two papers of Gelman et al. - [6] and [9], there is a specific practical problem that the authors attack with this model - in the papers, they try to model with linear regression the

properties of the outcome of a general election in the USA, having a data set. In our case, we will simulate all of the observed data y, based on simulated values for the regression parameters β, as well as chosen by hand values for the other parameters. We will estimate the Bayes factor for the two competing models with $\theta = 0$ and $\theta = 1.$, i.e.

$$B = \frac{P\left(y \mid \theta = 1\right)}{P\left(y \mid \theta = 0\right)}$$

where y is the observed data. The basic rule of using the Bayes factor is to calculate the ratio, and then compare its value to a table of predetermined values, which tell us how much support do we have for the two models. For more information about the Bayes factor, we refer the reader to specialized literature on the subject like the classic book of Bernardo et al. [2].

We will now specify the exact model of the regression and the weights function r. Following the paper by Boscardin and Gelman [9], we will choose for the functional form of r

$$r = \exp\left(1.5 X_2\right)$$

In other words, the model we are going to simulate and estimate is

$$y_i \mid \beta, \sigma \sim N\left(\left(X\beta\right)_i, \sigma^2 \exp\left(-3\theta X_{i2}\right)\right)$$

where we choose by hand $\sigma = 10$. In this model, θ is an external parameter, that will not be estimated.

We will test the HSMC algorithm and compare it to the standard AIS with random walk proposals in two cases - the first one will be only 3 dimensional, whereas the second one will be more challenging - a 20 dimensional regression. For all examples, we have simulated 100 observations of the dependent variable y. The parameters sets that are going to be estimated in these two cases are $\left(\beta = \left(\beta_1, \beta_2\right); \sigma\right)$ and $\left(\beta = \left(\beta_1, \ldots, \beta_{19}\right); \sigma\right)$ respectively. There is a specific detail that we would like to mention - because of the functional form of the variance in the case where $\theta \neq 0$, we see, that the posterior distribution over the parameters of interest β, σ will be bi-modal - the σ parameter is allowed to have positive and negative values, and for given value for it, both σ and $-\sigma$ results in the same linear regression model. So the choice of this model has another interesting feature - we will be able to test the HSMC algorithm in a high-dimensional and multimodal setup, where the two modes will be well separated. To make a connection with the HSMC setup, where we have f_0 and f_1 - in this case, f_0 is a multivariate normal distribution over β and σ. For f_1 - it is the posterior distribution over (β, σ), which is a product of the likelihood and the prior distribution over (β, σ). The prior distribution we use is a product of a uniform distribution on $\mathbb{R}$ for β, and Jeffrey's prior on σ, which is $p(\sigma) \sim 1/\sigma$.

The number of independent instances of the algorithm is $m = 10$, number of particles is $N = 100$ and the number of intermediate distributions n is 1000. Here again we have for the leapfrog parameters $\epsilon = 0.1$ and $l = 1$. We present in Table 1 the results of the simulations using HSMC for the 3 dimensional regression model. In the second column, called Data, we specify which data set we have used to calculate the Bayes factor. We test with both data sets in order to see whether the factor that we calculate will give the right indication about the real model, that gave rise to the observed data y. We calculate the log - likelihood, because of the numerical instabilities that appear during the calculation process. The Bayes factor then is estimate as the exponent of the difference of this two marginal

Table 1: HSMC - 3 dimensional example. Estimates of the marginal likelihoods and the Bayes factor for the two models.

	Data	Model	Log of the marginal likelihood estimate	Standard deviation	Estimated Bayes factor
$n = 1000$	$\theta = 0$ data	$\theta = 0$	22.27	0.4	1.6×10^{-10}
		$\theta = 1$	-0.27	0.4	
	$\theta = 1$ data	$\theta = 0$	-14.87	0.3	4.1×10^{9}
		$\theta = 1$	7.25	0.4	

Table 2: AIS - 3 dimensional example. Estimates of the marginal likelihoods and the Bayes factor for the two models.

	Data	Model	Log of the marginal likelihood estimate	Standard deviation	Estimated Bayes factor
$n = 1000$	$\theta = 0$ data	$\theta = 0$	19.95	1.4	4.9×10^{-10}
		$\theta = 1$	-1.48	0.8	
	$\theta = 1$ data	$\theta = 0$	-15.38	0.6	2.5×10^{9}
		$\theta = 1$	6.27	1.1	

likelihoods. We see from Table 1, that the HSMC decisively determines, through the value of the calculated factor, the true value of the heteroscedasticity factor θ.

In Table 2 we see the same results as in the previous Table 1, but this time for the AIS algorithm. We see clearly, that the estiamted Bayes factor is close to the one, estimated with the HSMC algorithm. The difference here is, that the standard deviation of the estimates is lower for the HSMC algorithm, compared to the AIS.

Now we look at the results for the 20 - dimensional example. The results for the HSMC algorithm are in Table 4. We see that for $n = 1000$ the HSMC algorithm manages again to estimate the Bayes factor correctly, although the example that we have is high - dimensional and bimodal. This is not the case for the AIS algorithm, as seen from the results in Table 3. We see, that because of the large variability in the estimates of the log marginal likelihood, the AIS algorithm estimates the Bayes factors really poorly. Another interesting fact is, that in the case of the AIS algorithm, the estimates of the log marginal likelihoods are substantially different from the estimates that we obtain using the HSMC algorithm. This could mean, that the AIS algorithm was far from equilibrium during the runs (which is supported by the large variance of the estimates as well).

We can clearly see, that in this particular case, the AIS algorithm produces wrong value for one of the Bayes factors, whereas the HSMC algorithm performs really well in both cases - homo- or heteroscedastic data.

5. Conclusions

In this paper we present a novel simulation algorithm - Hamiltonian Sequential Monte Carlo, that produces both samples of a desired target distribution, as well as an estimate of the ratio of the normalizing constants. We have developed this new simulation algorithm to tackle several goals. First, we introduce an extension of the standard Hamiltonian Monte Carlo with the idea to be able to use HMC for sampling of multimodal distributions. Our second goal was to introduce a way to use HMC to estimate ratios of normalizing constants.

Table 3: Estimates of the Bayes factors for the aforementioned heteroscedastic regression example, obtained by using standard AIS, in the case, where the dimension of the parameters space is 20.

	Data	Model	Log of the marginal likelihood estimate	Standard deviation	Estimated Bayes factor
$n = 1000$	$\theta = 0$ data	$\theta = 0$	-176.80	5.6	1.1×10^{-10}
		$\theta = 1$	-199.75	4.6	
	$\theta = 1$ data	$\theta = 0$	-176.68	4.5	2.8×10^{-9}
		$\theta = 1$	-196.38	5.5	

Table 4: Estimates of the Bayes factors for the aforementioned heteroscedastic regression example, obtained by using standard HSMC, in the case, where the dimension of the parameters space is 20.

	Data	Model	Log of the marginal likelihood estimate	Standard deviation	Estimated Bayes factor
$n = 1000$	$\theta = 0$ data	$\theta = 0$	-88.86	1.4	8.6×10^{-9}
		$\theta = 1$	-107.43	1.2	
	$\theta = 1$ data	$\theta = 0$	-106.70	1.1	1.7×10^{7}
		$\theta = 1$	-90.04	2.6	

For this purpose, we extend the method proposed in [3] by introducing a second Markov kernel. We saw that the introduction of a second kernel produces additional benefits - like the fact, we can control in a natural way the amount of noise we induce into the sampling algorithm while running it. This also means, that in certain regime of large value of the number of intermediate distributions n we can avoid losing time and resources in choosing an optimal mass matrix M. We show that by an example where we clearly see that for large n the HSMC algorithm does not depend on the choice of the scale of the mass matrix.

We also compare the performance of the HSMC to a standard SMC in a toy model, and also with the AIS algorithm in a real - world example of a heteroscedastic data, generated from a linear model. These two examples show really well that we were able to achieve our goal to introduce a sampling algorithm, based on HMC, that could sample from multimodal and high - dimensional distributions.

As a summary, we can say, that the HSMC algorithm provides a novel way to sample from multimodal distributions. It also makes possible the estimation of a ratio of normalizing constants of two distributions using techniques from standard HMC. As a by product we achieve a remarkable result - we have a way to avoid tuning by hand the mass matrix parameter of the standard HMC. All these properties make the HSMC algorithm really attractive research area in terms of theory, as well as in terms of methodological extensions.

Acknowledgements

Svetoslav Kostov would like to thank University of Bristol for the financial support through the University of Bristol Postgraduate Scholarship. SK would also like to thank Dr. Nick Whiteley for his help and guidance.

References

[1] V. I. Arnold. *Mathematical Methods of Classical Mechanics (Graduate Texts in Mathematics, Vol. 60)*. Springer, 2nd edition, Sept. 1997.

[2] J. M. Bernardo and A. F. M. Smith. *Bayesian Theory*. Wiley Series in Probability and Statistics. John Wiley and Sons, Inc., 2008.

[3] P. Del Moral. *Feynman-Kac Formulae. Genealogical and Interacting Particle Systems with Applications*. Probability and its Applications. Springer Verlag, New York, 2004.

[4] P. Del Moral, A. Doucet, and A. Jasra. Sequential Monte Carlo samplers. *J. R. Stat. Soc. Ser. B. Stat. Methodol.*, 68(3):411–436, June 2006.

[5] S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth. Hybrid Monte Carlo. *Physics Letters B*, 195(2):216 – 222, 1987.

[6] A. Gelman and X.-L. Meng. Simulating normalizing constants: from importance sampling to bridge sampling to path sampling. *Statist. Sci.*, 13(2):163–185, 05 1998.

[7] M. Girolami and B. Calderhead. Riemann manifold langevin and hamiltonian monte carlo methods. *J. R. Stat. Soc. Ser. B. Stat. Methodol.*, 73(2):123–214, 2011.

[8] N. Gordon, D. Salmond, and A. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE PROC-F*, 140(2):107–113, 1993.

[9] A. G. J.W. Boscardin. Bayesian regression with parametric models for heteroscedasticity. *Adv. Econom.*, 11(A):87–109, 1996.

[10] B. J. Leimkuhler and S. Reich. *Simulating Hamiltonian dynamics*. Cambridge monographs on applied and computational mathematics. Cambridge Univ., Cambridge, 2004.

[11] J. S. Liu. *Monte Carlo Strategies in Scientific Computing (Springer Series in Statistics)*. Springer-Verlag New York, Inc., 2001.

[12] R. M. Neal. Annealed importance sampling. *Stat. Comput.*, 11(2):125–139, 2001.

[13] R. M. Neal. MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 54:113–162, 2010.

[14] L. Tierney. Markov chains for exploring posterior distributions. *Ann. Statist.*, 22(4):1701–1728, 12 1994.

[15] L. Tierney. A note on Metropolis-Hastings Kernels for General State Spaces. *Ann. Appl. Probab.*, 8(1):pp. 1–9, 1998.