A Tutorial on the Hamiltonian Monte Carlo

Bryan Chang

1 Introduction

The Metropolis-Hastings algorithm is a powerful MCMC algorithm that allows us to sample from complex target distributions. However, its proposal step, which usually involves sampling from a Gaussian distribution, is highly inefficient, especially when the target distribution is high-dimensional. To explore the parameter space more efficiently, the Hamiltonian Monte Carlo (HMC) algorithm borrows the idea of Hamiltonian system from Physics and introduces an auxiliary momentum variable to its sampling scheme. Instead of sampling from the target distribution $\pi(q)$, we would now sample from $\pi(p, q)$, which is the joint distribution of the parameter of interest and the auxiliary momentum variable. As a result, at each iteration, new samples would be proposed according to the laws of a Hamiltonian system instead of a random walk.

2 The Hamiltonian System

A Hamiltonian system describes the dynamics between the position q and the momentum p of a particle. The Hamiltonian $H(q, p) = U(q) + K(p)$, where U is the potential energy and K is the kinetic energy, is the total energy of a particle and is conserved over time. Suppose the position is d-dimensional, then for each q_d, we introduce a momentum variable p_d.

In a Hamiltonian system, the movement of the particles over time is entirely governed by the Hamiltonian equations:

$$\frac{dq_i}{dt} = +\frac{\partial H}{\partial p_i} = +\frac{\partial K}{\partial p_i}$$
$$\frac{dp_i}{dt} = -\frac{\partial H}{\partial q_i} = -\frac{\partial U}{\partial q_i}$$

To solve the Hamiltonian equations (i.e., to propose samples of q and p that follow the rules of the Hamiltonian equations), we would use a symplectic integrator, in particular the Leapfrog Integrator.

The **Leapfrog Integrator**: Given staring values for q and p, we update them through the following steps:

$$\tilde{p} \leftarrow p - (\epsilon/2)\nabla_q U(q)$$
$$\tilde{q} \leftarrow q + (\epsilon)\nabla_p K(\tilde{p})$$
$$\tilde{p} \leftarrow \tilde{p} - (\epsilon/2)\nabla_q U(\tilde{q})$$

3 Probability Distributions and the Hamiltonian

To link the Hamiltonian to the distribution we wish to sample from, which is now $\pi(q, p)$, we make the following definitions:

The (unnormalized) **joint density of the parameter of interest and the momentum**:

$$\pi(q, p) \propto \exp(-H(q, p))$$

The **potential energy function**:

$$U(q) \equiv -\mathcal{L}(q)$$

, where $\mathcal{L}(q)$ is the log likelihood of the target distribution $\pi(q)$, up to a normalizing constant.

The **kinetic energy function**:

$$K(p) = \frac{1}{2}\, p^t p$$

(I made this choice of kinetic energy function for tutorial purpose, but it could be more complicated.)

Those definitions yield us the following equalities:

$$\begin{aligned} \pi(q, p) &\propto \exp(-H(q, p)) \\ &= \exp(-U(q) - K(p)) \\ &= \underbrace{\exp(\mathcal{L}(q)}_{\pi(q)} - \frac{1}{2}\, p^t p) \end{aligned}$$

Also, the updating equations for the leapfrog integrator becomes:

$$\begin{aligned} \tilde{p} &\leftarrow p + (\epsilon/2)\nabla_q \mathcal{L}(q) \\ \tilde{q} &\leftarrow q + (\epsilon)\tilde{p} \\ \tilde{p} &\leftarrow \tilde{p} + (\epsilon/2)\nabla_q \mathcal{L}(\tilde{q}) \end{aligned}$$

4 Hamiltonian Monte Carlo

The Hamiltonian Monte Carlo algorithm is split into two steps at each iteration: a Gibbs step for the momentum $\pi(p)$, and a Metropolis step for the joint distribution $\pi(q, p)$. In the first step, we resample a random momentum $p \sim N_d(0, I)$ (suppose the parameter of interest is d-dimensional). In the second step, we propose new values for q and p, $\tilde{q}$ and $\tilde{p}$, using the leapfrog integrator, then accept the proposal with probability:

$$\min\{1, \frac{\pi(\tilde{q}, \tilde{p})}{\pi(q, p)}\} = \min\{1, \frac{\exp(-H(\tilde{q}, \tilde{p}))}{\exp(-H(q, p))}\}$$

After getting the samples from $\pi(q, p)$, we can then marginalize out the auxiliary momentum variables to obtain samples from the target distribution $\pi(q)$.

Note that the acceptance probability becomes 1 if we could simulate the Hamiltonian system exactly, since the energy is conserved. Because the leapfrog integration is just an approximation, we need the Metropolis step to correct for the approximation error.

On the other hand, the Gibbs step for the momentum is necessary for the samples to jump between different **energy level sets**:

$$H^{-1}(E) = \{q, p \mid H(q, p) = E\}$$

Without resampling the momentum, we would always be drawing samples on the same energy level set (since the leapfrog integrator only gives us samples under the same Hamiltonian).

Below is the outline of the algorithm. There are two hyper-parameters for the leapfrog integrator: ϵ, the step size, and L, the number of steps. The effects of the hyper-parameters will be discussed in the last section. In addition, we also have to solve for the gradient of the log likelihood function $\nabla_q \mathcal{L}(q)$, which could be easily done through automatic differentiation (available in probabilistic programming languages such as Pyro/Edward2/Stan/PyMC3).

Algorithm 1: Hamiltonian Monte Carlo

Given q^0, ϵ, L, $\mathcal{L}$, $N_{samples}$;
for $n = 1$ *to* $N_{samples}$ **do**
  Sample $p^0 \sim \mathcal{N}(0, I)$;
  Set $\tilde{q} \leftarrow q^{n-1}$, $\tilde{p} \leftarrow p^0$;
  for $i = 1$ *to* L **do**
    Set $\tilde{q}, \tilde{p} \leftarrow$ Leapfrog $(\tilde{q}, \tilde{p}, \epsilon)$
  end
  With probability $\alpha = \min\{1, \frac{\pi(\tilde{q}, \tilde{p})}{\pi(q^{n-1}, p^0)}\}$, set $q^n \leftarrow \tilde{q}$, else set $q^n \leftarrow q^{n-1}$
end
function Leapfrog (q, p, ϵ);
  Set $\tilde{p} \leftarrow p + (\epsilon/2)\nabla_q \mathcal{L}(q)$;
  Set $\tilde{q} \leftarrow q + (\epsilon)\tilde{p}$;
  Set $\tilde{p} \leftarrow \tilde{p} + (\epsilon/2)\nabla_q \mathcal{L}(\tilde{q})$;
  Return $(\tilde{q}, \tilde{p})$

5 Example: Standard Normal

Here I demonstrate sampling from a standard normal using HMC. We only need two information: one is $\mathcal{L}(q)$ (up to an normalizing constant), for the Metropolis step, and one is $\nabla_q \mathcal{L}(q)$, for the leapfrog integrator. The gradient $\nabla_q \mathcal{L}(q)$ could be calculated simply by hand in this example. We have:

$$\begin{aligned} \mathcal{L}(q) &\propto -\frac{1}{2}q^2 \\ \nabla_q \mathcal{L}(q) &= -q \end{aligned}$$

Note that the Hamiltonian equations for this example is:

$$\begin{aligned} \frac{dq}{dt} &= +\frac{\partial K}{\partial p} = p \\ \frac{dp}{dt} &= -\frac{\partial U}{\partial q} = \nabla_q \mathcal{L}(q) = -q \end{aligned}$$

, which has an analytical solution:

$$\begin{aligned} q(t) &= r\cos(a+t) \\ p(t) &= -r\sin(a+t) \end{aligned}$$

, where r and a are some constants.

The solution corresponds to a clockwise rotation around the origin in the (q, p) plane.

Below is a demonstration of 4 iterations of the HMC sampler, with step size = 0.2 and number of steps = 15 (Also, all proposed values are accepted in the figure).

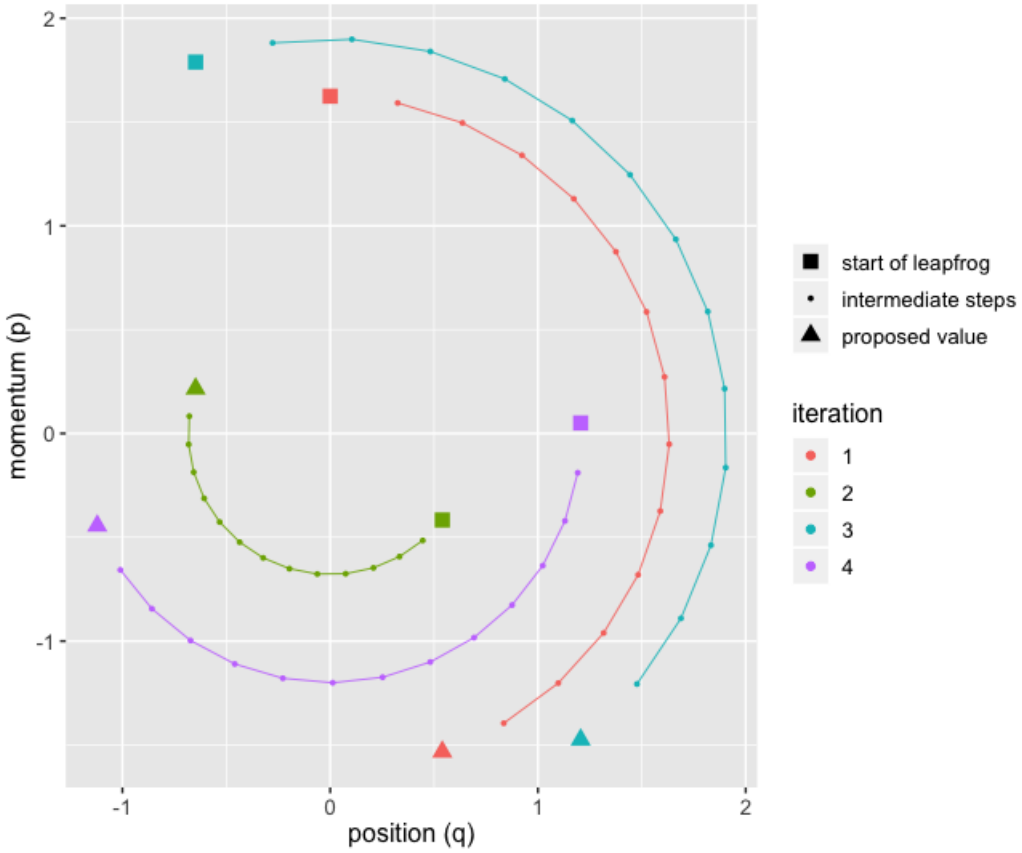

Figure 1: Evolution of position and momentum over 4 iterations.

We could see that at the start of each iteration, the resampled momentum brings the sampler to a different energy level set.

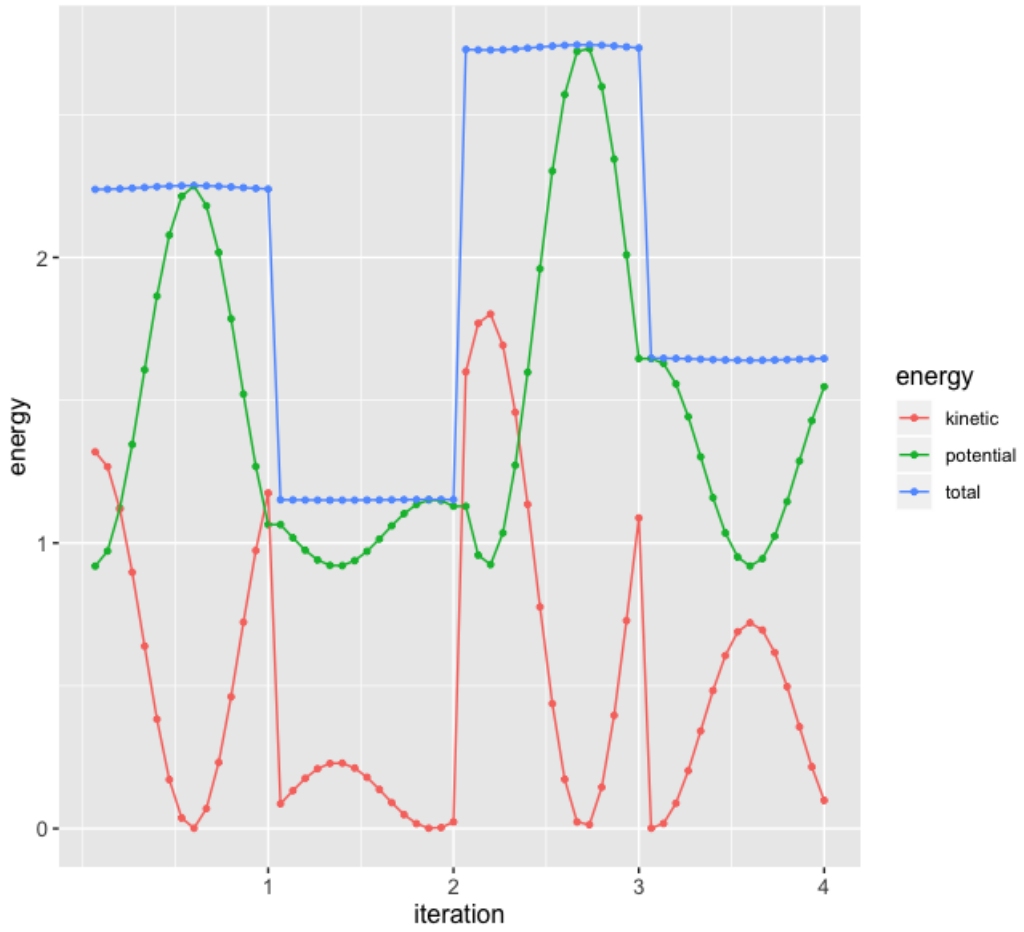

Figure 2: Evolution of energy over 4 iterations.

Figure 2 shows the evolution of energy over 4 iterations. We could make two observations. First, the total energy (the Hamiltonian) is nearly conserved within each iteration since we chose a small step size. Second, the change in energy between each iteration is sheerly caused by the change in kinetic energy (due to the resampling of the momentum).

6 Choosing Hyper-parameters

The choice of hyper-parameters is critical for the efficiency of HMC. Both hyper-parameters appear in the leapfrog integrator: the step size ϵ, and the number of steps L. A large ϵ may lead to poor approximation to the solution of the Hamiltonian equations, hence resulting in a high rejection rate. On the other hand, an ϵ that is too small may lead to a waste of computation time. If L is too small, the proposed value would be too close to the previous value. If L is too large, we might end up looping along the Hamiltonian trajectories, which also results in a waste of computation time.

Below we show two leapfrog integrators for the standard normal, one with $\epsilon = 0.2$, and one with $\epsilon = 0.5$. Both of them have $L = 30$.

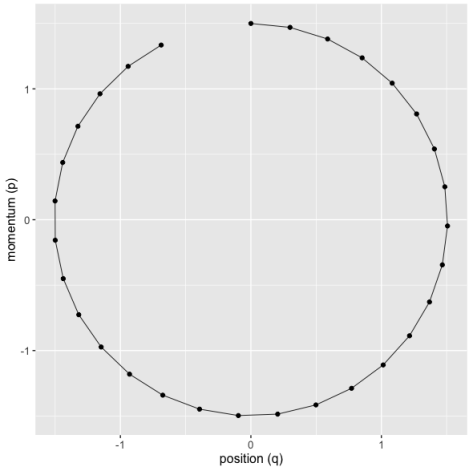

Figure 3: Evolution of position and momentum, $\epsilon = 0.2$, $L = 30$

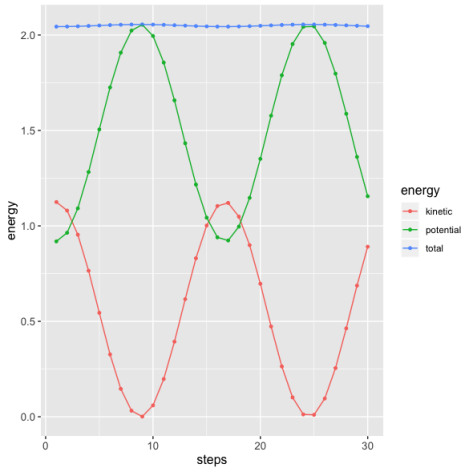

Figure 4: Evolution of energy, $\epsilon = 0.2$, $L = 30$

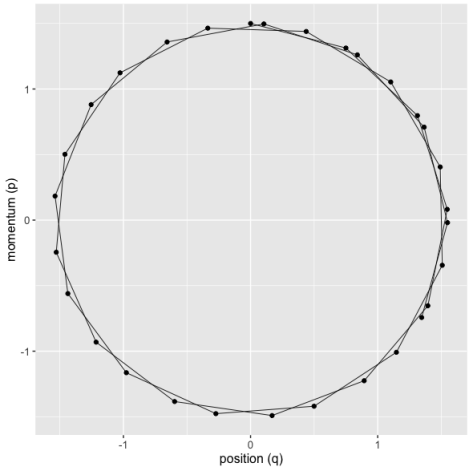

Figure 5: Evolution of position and momentum, $\epsilon = 0.5$, $L = 30$

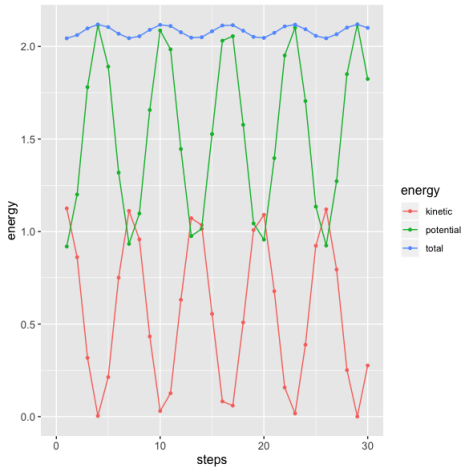

Figure 6: Evolution of energy, $\epsilon = 0.5$, $L = 30$

When setting $\epsilon = 0.2$, we are almost perfectly simulating from the exact solutions. However, the number of steps seems to be a bit too large since we almost make a loop back to the start. The total energy is nearly conserved in this example, which should make the acceptance probability of this proposal close to 1.

When setting $\epsilon = 0.5$, the approximation error becomes slightly visible, and we start looping. (we could tell from figure 6 that we make a loop in around 13 steps). The total energy is clearly not conserved in this example. We could calculate the acceptance probability of the proposal, which is: $\exp(H(q,p) - H(\tilde{q},\tilde{p})) = \exp(2.04 - 2.1) = 0.94$.

In practice, it is often hard to find a proper choice of ϵ and L. The No-U-Turn Sampler (NUTS), which is now commonly used in most probabilistic programming languages, is an extension of the HMC that eliminates the need to set L and tunes the ϵ automatically.

7 References

Betancourt, M. (2017). A conceptual introduction to Hamiltonian Monte Carlo. arXiv preprint arXiv:1701.02434.

Matthew D. Hoffman and Andrew Gelman. (2014) The no-U-turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. Journal of Machine Learning Research, 15: 1351-1381.

Neal, R. M. (2010) MCMC using Hamiltonian dynamics. In Handbook of Markov Chain Monte Carlo (eds S. Brooks, A. Gelman, G. Jones and X.-L. Meng). Boca Raton: Chapman and Hall–CRC Press.