Conceptual motivation of MCMC using Hamiltonian Dynamics

Anubhav Mittal
anubhavm@iitk.ac.in
Supervised by: Prof. Satyadev Nandakumar and Prof. K. Vasudevan

***Abstract*—Hamiltonian Monte-Carlo was originally developed in the late 1980s as Hybrid Monte Carlo to tackle calculations in Lattice Quantum Chromodynamics. From there, it has found its way into the mainstream of statistical computing through the work of Prof. Radford M. Neal [1] [2] [3] and the like [4] [5].**

The aim of this review is to first show the problem of computing expectations encountered in high dimensions and show how the usual MCMC performs. Then we describe Hamiltonian dynamics, and show how to use it to construct a Markov chain Monte Carlo method. We motivate the choice of different energy functions, the different approximations involved, as well as discuss the problems that arise and how to address them.

The first step is to define a Hamiltonian function in terms of the probability distribution we wish to sample from. In addition to the variables we are interested in (the position variables), we must introduce auxiliary momentum variables, which typically have independent Gaussian distributions. The Hamiltonian Monte Carlo method alternates simple updates for these momentum variables with Metropolis updates in which a new state is proposed by computing a trajectory according to Hamiltonian dynamics, implemented with the "leapfrog method". A state proposed in this way can be distant from the current state but nevertheless have a high probability of acceptance. This bypasses the slow exploration of the state space that occurs when Metropolis updates are done using a simple random-walk proposal distribution.

We also include some of the popular extensions to Hamiltonian Monte Carlo and how they address specific problems.

I. Computing expectations by exploring probability distributions

A. Expectation of a given function

The ultimate undertaking in statistical computing is evaluating expectations with respect to some distinguished target probability distribution. For example, we might be interested in extracting information from a posterior distribution over model configuration space in Bayesian inference. Here we will be agnostic, considering only a target distribution, π, on a D-dimensional sample space, Q, and the corresponding expectations of functions, $E_\pi[f]$. Assuming the distribution is smooth over the space Q, we have:

$$E_\pi[f] = \int_Q \pi(q) f(q) dq$$

Unfortunately, for any nontrivial target distribution we will not be able to evaluate these integrals analytically, and we must instead resort to numerical methods which only approximate them. For a method to scale to the complex problems at the frontiers of applied statistics, it has to make effective use of each and every evaluation of the target density, $\pi(q)$, and relevant functions, $f(q)$. Optimizing these evaluations is a subtle problem frustrated by the natural geometry of probability distributions, especially over high-dimensional parameter spaces.

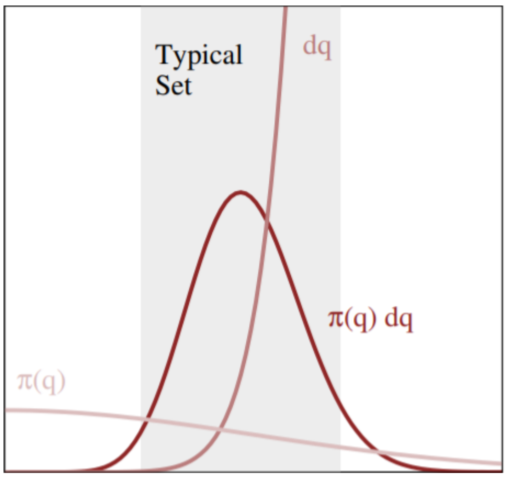

Fig. 1. Typical Set

B. Typical set

One way to ensure computational inefficiency is to waste computational resources evaluating the target density and relevant functions in regions of parameter space that have negligible contribution to the desired expectation. Intuitively, we should consider the neighborhood around the mode where the density is maximized. However, expectation values are given by accumulating the integrand over a volume of parameter space and, while the density is largest around the mode, there is not much volume there.

The neighborhood immediately around the mode features large densities, but in more than a few dimensions the small volume of that neighborhood prevents it from having much contribution to any expectation. On the other hand, the complimentary neighborhood far away from the mode features a much larger volume, but the vanishing densities lead to similarly negligible contributions expectations. The only significant contributions come from the neighborhood between these two extremes known as the **typical set**, as shown in figure 1.

Consequently, in order to compute expectations efficiently, we have to be able to identify, and then focus our computational resources into, the typical set.

II. Markov Chain Monte Carlo

A. Estimation using MCMC

Markov chain Monte Carlo uses a Markov chain to stochastically explore the typical set, generating a random grid across the region of high probability from which we can construct accurate expectation estimates. Given sufficient computational resources a properly designed Markov chain will eventually explore the typical set of any distribution. The more practical, and much more challenging question, however, is whether a given Markov chain will explore a typical set in the finite time available in a real analysis.

For constructing a Markov chain, we can think of a Markov transition as a conditional probability density, $T(q'|q)$, defining to which point, q', we are most likely to jump from the initial point, q.

$$\pi(q) = \int_Q dq' \pi(q') T(q|q'))$$

So long as this condition holds, at every initial point the Markov transition will concentrate towards the typical set. Consequently, no matter where we begin in parameter space the corresponding Markov chain will eventually drift into, and then across, the typical set. Using the samples generated by the Markov chain $q_0, ..., q_N$, we use a monte-carlo estimate of the function:

$$\hat{f}_N = \frac{1}{N} \sum_{n=0}^{N} f(q_n)$$

If the chain is run long enough, and under certain assumption on the target density, we can claim that the estimate will converge to the true expectation:

$$\lim_{N \to \infty} \hat{f}_N = E_\pi[f]$$

Ideally, the chain will have three phases:

- . In the first phase the Markov chain converges towards the typical set from its initial position in parameter space while the Markov chain Monte Carlo estimators suffer from strong biases.
- The second phase begins once the Markov chain finds the typical set and persists through the first sojourn across the typical set. This initial exploration is extremely effective and the accuracy of Markov chain Monte Carlo estimators rapidly improves as the bias from the initial samples is eliminated.
- The third phase consists of all subsequent exploration where the Markov chain refines its exploration of the typical set and the precision of the Markov chain Monte Carlo estimators improves, albeit at a slower rate

The Metropolis-Hastings Algorithm This algorithm defines the transition function for constructing our Markov chain. The Metropolis-Hastings algorithm is comprised of two steps: a proposal and a correction. The proposal is any stochastic perturbation of the initial state while the correction rejects any proposals that stray too far away from the typical set of the target distribution. More formally, let $Q(q'|q)$ be the probability density defining each proposal. The probability of accepting a given proposal is then given by

$$a(q'|q) = \min(1, \frac{Q(q|q')\pi(q')}{Q(q'|q)\pi(q)})$$

The most common proposal distribution used is Gaussian, where the algorithm is now called Random-Walk Metropolis and due to symmetry of the gaussian distribution, the acceptance probability takes the following form:

$$a(q'|q) = \min(1, \frac{\pi(q')}{\pi(q)})$$

B. Problems

The idealized behavior of MCMC requires that the Markov transition is compatible with the structure of the target distribution. When the target distribution exhibits pathological behavior, like in a target probability distribution where the typical set pinches into a region of high curvature, Markov transitions will have trouble exploring and Markov chain Monte Carlo will fail.

Even if we have a well-behaved target distribution, we may still encounter problems. Because of its conceptual simplicity and the ease in which it can be implemented by practitioners, Random Walk Metropolis is still popular in many applications. Unfortunately, that seductive simplicity hides a performance that scales poorly with increasing dimension and complexity of the target distribution. As the dimension of the target distribution increases, the volume exterior to the typical set overwhelms the volume interior to the typical set, and almost every Random Walk Metropolis proposal will produce a point on the outside of the typical set, towards the tails. The density of these points, however, is so small, that the acceptance probability becomes negligible. In this case almost all of the proposals will be rejected and the resulting Markov chain will only rarely move.

C. Need for better transition functions

As seen in the problems encountered above, we need to define a different transition function. We need to exploit information about the geometry of the typical set. Specifically, we need transitions that can follow those contours of high probability mass, coherently gliding through the typical set.

When the sample space is continuous, a natural way of encoding this direction information is with a vector field aligned with the typical set. A vector field is the assignment of a direction at every point in parameter space, and if those directions are aligned with the typical set then they act as a guide through this neighborhood of largest target probability.

To construct such a vector field, we exploit the differential structure of the target distribution which we can query through the gradient of the target probability density function. In particular, the gradient defines a vector field in parameter space sensitive to the structure of the target distribution.

Unfortunately, that sensitivity is not sufficient as the gradient will never be aligned with the typical set. Following the guidance of the gradient pulls us away from the typical set and towards the mode of the target density. To utilize the information in the gradient we need to complement it with additional geometric constraints, carefully removing the dependence on any particular parameterization while twisting the directions to align with the typical set.

III. Hamiltonian Dynamics

Hamiltonian dynamics has a physical interpretation that can provide useful intuitions. In two dimensions, we can visualize the dynamics as that of a frictionless puck that slides over a surface of varying height. The state of this system consists of the position of the puck, given by a 2D vector q, and the momentum of the puck (its mass times its velocity), given by a 2D vector p. The potential energy, $U(q)$, of the puck is proportional to the height of the surface at its current position, and its kinetic energy, $K(p)$, is equal to $|p|^2/(2m)$, where m is the mass of the puck. On a level part of the surface, the puck moves at a constant velocity, equal to p/m. If it encounters a rising slope, the pucks momentum allows it to continue, with its kinetic energy decreasing and its potential energy increasing, until the kinetic energy (and hence p) is zero, at which point it will slide back down (with kinetic energy increasing and potential energy decreasing).

In non-physical MCMC applications of Hamiltonian dynamics, the position will correspond to the variables of interest. The potential energy will be minus the log of the probability density for these variables. Momentum variables, one for each position variable, will be introduced artificially.

A. Hamilton's equations

Hamiltonian dynamics operates on a d-dimensional position vector, q, and a d-dimensional momentum vector, p, so that the full state space has $2d$ dimensions. The system is described by a function of q and p known as the Hamiltonian, $H(q, p)$.

Equations of motion The partial derivatives of the Hamiltonian determine how q and p change over time, t, according to Hamiltons equations:

$$\frac{dq_i}{dt} = \frac{\partial H}{\partial p_i}$$

$$\frac{dp_i}{dt} = -\frac{\partial H}{\partial q_i}$$

for $i = 1, ..., d$.

Potential and kinetic energy For Hamiltonian Monte Carlo, we usually use Hamiltonian functions that can be written as follows:

$$H(q, p) = U(q) + K(p)$$

Here, $U(q)$ is called the potential energy, and will be defined to be minus the log probability density of the distribution for q that we wish to sample, plus any constant that is convenient. $K(p)$ is called the kinetic energy, and is usually defined as

$$K(p) = \frac{p^T M^1 p}{2}$$

Here, M is a symmetric, positive-definite mass matrix, which is typically diagonal, and is often a scalar multiple of the identity matrix.

With these forms, the Hamilton's equations reduce to the following:

$$\frac{dq_i}{dt} = [M^{-1}p]_i$$

$$\frac{dp_i}{dt} = -\frac{\partial U}{\partial q_i}$$

B. Properties of Hamilton's equations

C. Discretizing the Hamilton's equations

For simplicity, we assume that M is diagonal, with diagonal elements

$$m_1, ..., m_d$$

, so that:

$$K(p) = \sum_{i=1}^{d} \frac{p_i^2}{m_i}$$

Eulers method For Hamiltons equations, this method performs the following steps, for each component of position and momentum, indexed by $i = 1, ..., d$:

$$p_i(t+\epsilon) = p_i(t) + \epsilon \frac{dp_i}{dt}(t) = p_i(t) - \epsilon \frac{\partial U}{\partial q_i}(t)$$

$$q_i(t+\epsilon) = q_i(t) + \epsilon \frac{dq_i}{dt}(t) = q_i(t) + \epsilon \frac{p_i(t)}{m_i}$$

In practise, however, the results are not that good, so we slightly modify the method to our liking.

A modification of Eulers method Much better results can be obtained by slightly modifying Eulers method, as follows:

$$p_i(t+\epsilon) = p_i(t) - \epsilon \frac{\partial U}{\partial q_i}(t)$$

$$q_i(t+\epsilon) = q_i(t) + \epsilon \frac{p_i(t+\epsilon)}{m_i}$$

We simply use the new value for the momentum variables, p_i , when computing the new value for the position variables, q_i . A method with similar performance can be obtained by instead updating the q_i first and using their new values to update the p_i . We get better results using this method, but we can still improve further with almost no extra overhead, as described below.

The leapfrog method Even better results can be obtained with the leapfrog method, which works as follows:

$$p_i(t+\frac{\epsilon}{2}) = p_i(t) - \frac{\epsilon}{2} \frac{\partial U}{\partial q_i}(t)$$

$$q_i(t+\epsilon) = q_i(t) + \epsilon \frac{p_i(t+\frac{\epsilon}{2})}{m_i}$$

$$p_i(t+\epsilon) = p_i(t+\frac{\epsilon}{2}) - \frac{\epsilon}{2} \frac{\partial U}{\partial q_i}(t+\epsilon)$$

IV. MCMC FROM HAMILTONIAN DYNAMICS

Using Hamiltonian dynamics to sample from a distribution requires translating the density function for this distribution to a potential energy function and introducing momentum variables to go with the original variables of interest (now seen as position variables). We can then simulate a Markov chain in which each iteration resamples the momentum and then does a Metropolis update with a proposal found using Hamiltonian dynamics.

A. *Canonical distributions*

The distribution we wish to sample can be related to a potential energy function via the concept of a canonical distribution from statistical mechanics. Given some energy function, $E(x)$, for the state, x, of some physical system, the canonical distribution over states has probability or probability density function:

$$P(x) = \frac{1}{Z} exp(-E(x)/T)$$

Viewing this the opposite way, if we are interested in some distribution with density function $P(x)$, we can obtain it as a canonical distribution with $T = 1$ by setting $E(x) = logP(x)logZ$, where Z is any convenient positive constant.

$$P(q,p) = \frac{1}{Z} exp(-E(q,p))$$

Since $H(q,p) = U(q) + K(p)$,

$$P(q,p) = \frac{1}{Z} exp(-U(q))exp(-K(p))$$

we see that q and p are independent, and each have canonical distributions, with energy functions $U(q)$ and $K(p)$. We will use q to represent the variables of interest, and introduce p just to allow Hamiltonian dynamics to operate.

In Bayesian statistics, the posterior distribution for the model parameters is the usual focus of interest, and hence these parameters will take the role of the position, q. We can express the posterior distribution as a canonical distribution using a potential energy function defined as follows:

$$U(q) = -\log[\pi(q)L(q|D)]$$

where $\pi(q)$ is the prior density, and $L(q|D)$ is the likelihood function given data D.

B. *The algorithm for Hamiltonian MCMC*

We make some general assumptions for ease of notation. We assume that the density is non-zero everywhere. We must also be able to compute the partial derivatives of the log of the density function. These derivatives must therefore exist, except perhaps on a set of points with probability zero, for which some arbitrary value could be returned.

We use the values of $K(p)$ and $U(q)$ as defined before. There are two main steps in the algorithm:

1) In the first step, new values for the momentum variables are randomly drawn from their Gaussian distribution, independently of the current values of the position variables. For the kinetic energy, the d momentum variables are independent, with p_i having mean zero and variance m_i . Since q isnt changed, and p is drawn from its correct conditional distribution given q (the same as its marginal distribution, due to independence), this step obviously leaves the canonical joint distribution invariant.
2) For the second step, we need some background for the details. There is one important exception to performance of the leapfrog method. Long time accuracy can be compromised when the exact energy level sets feature neighborhoods of high curvature that the finite time discretization is not able to resolve. These neighborhoods induce a divergence that almost immediately propels the numerical trajectory towards infinite energies. This distinctive behavior proves beneficial in practice, however, because it makes the failures of the method straightforward to identify and hence diagnose. An intuitive way to correct this is to treat the Hamiltonian transition as the proposal for a Metropolis Hastings scheme on phase space.
   In the second step, a Metropolis update is performed, using Hamiltonian dynamics to propose a new state. Starting with the current state, (q, p), Hamiltonian dynamics is simulated for L steps using the Leapfrog method (or some other reversible method that preserves volume), with a stepsize of ϵ. Here, L and ϵ are parameters of the algorithm, which need to be tuned to obtain good performance.
   We encounter a problem here.Because Hamiltonian trajectories, and their numerical approximations, are deterministic and non-reversible, Metropolis-Hastings proposals are always rejected. In particular, we have positive proposal probabilities going forwards in time but vanishing proposal probabilities going backwards in time which renders the Metropolis-Hastings acceptance probability identically zero.
   If we modify the Hamiltonian transition to be reversible, however, then the ratio of proposal densities becomes non-zero and we achieve a useful correction scheme. The simplest way of achieving a reversible proposal is to augment the the numerical integration with a negation step that flips the sign of momentum,

$$(p_L, q_L) \rightarrow (p_L, -q_L)$$

   Hence, the momentum variables at the end of the L-step trajectory are then negated, giving a proposed state $(q*, p*)$. This proposed state is accepted as the next state of the Markov chain with probability:

$$min[1, exp(-H(q*, p*) + H(q, p)]$$
$$= min[1, exp(-U(q*) + U(q) - K(p*) + K(p)]$$

If the proposed state is not accepted (ie, it is rejected), the next state is the same as the current state (and is counted again when estimating the expectation of some function of state by its average over states of the Markov chain).

Importance of first step The resampling of the momentum variables is still crucial to obtaining the proper distribution for q. Without resampling, $H(q, p) = U(q) + K(p)$ will be (nearly) constant, and since $K(p)$ and $U(q)$ are non-negative, $U(q)$ could never exceed the initial value of $H(q,p)$ if no resampling for p were done.

C. Avoiding random walks

Avoidance of random-walk behaviour is one major benefit of Hamiltonian Monte Carlo. To see this, note that the variance of the position after n iterations of random walk Metropolis from some start state will grow in proportion to n (until this variance becomes comparable to the overall variance of the state), since the position is the sum of mostly independent movements for each iteration. The standard deviation of the amount moved (which gives the typical amount of movement) is therefore proportional to $\sqrt{n}$.

The stepsize used for the leapfrog steps is similarly limited by the most constrained direction, but the movement will be in the same direction for many steps. The distance moved after n steps will therefore tend to be proportional to n, until the distance moved becomes comparable to the overall width of the distribution. The advantage compared to movement by a random walk will be a factor roughly equal to the ratio of the standard deviations in the least confined direction and most confined direction.

References

[1] R. M. Neal *et al.*, “Mcmc using hamiltonian dynamics,” *Handbook of markov chain monte carlo*, vol. 2, no. 11, p. 2, 2011.

[2] R. M. Neal, *Bayesian learning for neural networks*, vol. 118. Springer Science & Business Media, 2012.

[3] R. M. Neal, “An improved acceptance procedure for the hybrid monte carlo algorithm,” *Journal of Computational Physics*, vol. 111, no. 1, pp. 194–203, 1994.

[4] M. Betancourt, “A conceptual introduction to hamiltonian monte carlo,” 2017.

[5] M. Betancourt, “A general metric for riemannian hamiltonian monte carlo,” in *First International Conference on the Geometric Science of Information (F. Nielsen and F. Barbaresco, eds.). Lecture Notes in Computer Science*, vol. 8085, 2013.