Hamiltonian Dynamics Sampling

Abstract

Hamiltonian Dynamics Monte Carlo is a popular method used in simulating complicated distribution. The performance of HMC highly depends on geometry of the momentum-state dual space; we investigate several pathological dual space that leads to the failure of HMC from high level geometric perspective. This includes isolated modes, high curvature and multiple spatial scales occurs in dual space. We summaries several approaches that improve the results. We reinterpret the Riemannian Manifold HMC from the perspective of the relationship between Hessian and Co-variance as this can better explain the frame to propose a General Riemannian Metrics for a better generalization of Manifold HMC. We also adapt the stepsize and leapfrog steps of continuously tempered HMC to improve the performance and generate a larger number of effective samples.

Contents

1 Introduction

1.1 Motivation

We like to draw samples from a probability distribution for many reasons. Consider the Bayes' Rule

$$P(Z = z|X = x) = \frac{P(x,z)}{P(X = x)} = \frac{P(x,z)}{\sum_z P(X = x, Z = z)}$$

Notice the normalizer term (we call target distribution) can be hard to compute. Let's say that Z takes only $\{0,1\}$ with dimension D; there are 2^D possible combination for the values of Z. Since computing exact form of normalizer is generally intractable ; in practice, we approximate the sum values by drawing some random samples from distribution and compute sum of the samples. In general, the intractable sum can be encountered when one interested in marginal/conditional probability distribution in the directed graph, posterior distribution of parameters in Bayesian Inference.

1.2 Basics

Using only random samples to approximate the sum or integral is justified by thinking the sum or integral as an expectation under the target distribution; then average out to approximate the expectation

$$s = \sum_{\mathbf{x}} p(\mathbf{x}) f(\mathbf{x}) = \mathbb{E}_{\mathbf{x}}[f(\mathbf{x})]$$

Let $\{\mathbf{x}^{(i)}\}_{i=1}^N$ be iid from distribution p; the estimator

$$\hat{s} = \frac{1}{N} \sum_i f(\mathbf{x}^{(i)})$$

is an unbiased estimator. And by Law of Large Number, $\hat{s} \to s$ as $n \to \infty$. And the variances is

$$Var(\hat{s}) = Var(\frac{1}{N} \sum_i f(\mathbf{x}^{(i)})) = \frac{1}{N^2} \sum_i Var(f(\mathbf{x}^{(i)})) = \frac{Var(f(\mathbf{x})}{N} \to 0$$

as $n \to \infty$

1.3 Challenges

Notice that the major challenges here is how to sample from the target distribution. To directly sample from target distribution we actually have to compute the probability for each possible combination first (which would already solve our problem). Therefore, sample techniques are used based on the assumption that we can't evaluate the probability distribution p but we are able to evaluate its density function. That is

$$p(\mathbf{x}) = \frac{\tilde{p}(\mathbf{x})}{\sum_z \tilde{p}(\mathbf{x})} = \frac{\tilde{p}(\mathbf{x})}{Z}$$

We can't evaluate normalized $p(\mathbf{x})$ but able to evaluate $\tilde{p}(\mathbf{x})$ in polynomial time for all $\mathbf{x}$.
Another problem is the curse of dimensionality. If the dimension of $\mathbf{x}$ is large, each $\mathbf{x}$ will behave weirdly. In a simple example, consider a $d-$dimensional hyper-sphere with radius r and each point is uniformly distributed, The fraction of its hyper-volume lying between values $r - c$ and r, where $0 < c < r$, is

$$V = 1 - (1 - \frac{c}{r})^d$$

For any fixed radius r, c, as $d \to \infty$, we see that $V \to 1$, it means that most of points will be found closed to the surface. That is even points are uniformly distributed, in high dimension, many points concentrated in

a small region. When one computing the sum or integral based on uniform sampling, we must make sure that we sample large enough number of samples such that we can sample points from that small region. In general, the number of samples are so large make the uniform sampling useless in practice.

1.4 Monte Carlo Method

In this summary, we focus on one particular methods that allowing us to solve the problem. A special version of MCMC algorithm–Hamiltonian Monte Carlo (HMC) that explores the space of interest without knowledge of normalized target distribution and meanwhile collecting samples during its exploration. These collected samples can be used to compute mean, variances, or used for target density estimation. After the presentation of basic HMC, several variations of the method will be discussed for further improvement.

2 Hamiltonian Monte Carlo

2.1 Motivation

In traditional Metropolis Hasting Monte Carlo, one performs the random walk in the space of interest. Every time when a proposed move is made, there is an acceptance probability to determine whether such a move is accepted. The Metropolis Hasting is simple to implement and satisfies all sufficient conditions for Markov Chain to converge. However, when scales to high dimension, with limited computational resources, this algorithm behaves badly when explore the typical set of target density. Without efficiently explore the typical set of target density, we facing the very low acceptance rate and bad samples (tends to bias). HMC approaches this problem from the study of the geometry of typical set such that allows the algorithm to have high acceptance rate and collecting efficient samples.

2.2 Typical Set

In high dimension space, given any neighborhood, the volume inside the neighborhood is much smaller than the volume outside the neighborhood. As we have shown above, in high dimensional sphere, the volume concentrated on the shell of the sphere; the center of the sphere becomes negligible as dimension increase.

Consider the simplest case, uni-mode Gaussian distribution; its contour will be a high dimensional sphere where the mode is the center. Most of the volume is away from the mode (lies in the "tail" of the density). Consequently, when one integrates over the density space to compute the normalizing constant or expectation, the "tail region" (shell of the sphere, with low density) contributes massive volume.

$$\mathbb{E}(f(x)) = \int f(x)p(x)dx \tag{1}$$

In the equation (1), dx denotes the small neighborhood around point x. This values is very large for point x in tail region ($p(x)$ is very small). On the other hand, if x lies in the near mode region, dx is small and $p(x)$ is large. The same argument can be made for one wants to compute CDF or normalizing the density function

Hence, for any neighborhood around the mode, with large density but very small volume; while its complementary space, far away from the mode, with small density but very large volume. Both space makes small contribution to the integration. Most of the contribution to the integral comes from the region where

probability density and volume is well balanced. This region is called typical set. However, as dimension goes to infinite, shell (low probability density) extracts all of the volume, typical set will tends to singular. We see that, evaluating the integrand over the region outside the typical set yields insignificant results and waste of computation resources and only the typical region has important effect. This motivates Hamiltonian Monte Carlo. Our goal is to design an algorithm that avoid random walk, and stay in the typical set as long as possible to collect efficient samples. The ideal behaviour would be: start with any initial position, and travels to the typical set, and explore the typical set as long as possible.

2.3 Hamiltonian Dynamics

2.3.1 Analogous

Consider in the continuous state space; starts with some initial point, instead of random walk in the state space, we follow some carefully designed direction that guides us to trace along the typical region and lands at a new point. The natural way of design such trace direction would be constructing a vector field where its center is the mode of the target distribution. An analogous example, one can think of the Earth is the mode of interest distribution, and the Earth's gravitational field is our vector field. Imagine a satellite (its position in the gravitational field represents the our algorithm's current state) travelling in the gravity field of the Earth. There is a gravitational force that pulls the satellite towards the the Earth, but with a proper momentum, the satellite doesn't fall into the Earth but orbits around.

Hence, we aim to construct a vector field such that our algorithm (satellite) orbits around the Earth (mode of target distribution). However, we allow the satellite to orbits at different level of level set; that is allowing satellite can orbits along at different distance to the Earth.

2.3.2 Condition for HMC

The question is that what do we require to construct such vector field for different target distribution ?

Since we are given the un-normalized target distribution; we can compute its gradient. The gradient at point x points to which direction does the increasing fastest at that point. More specifically, the gradient of the un-normalized target distribution points to the mode of the distribution. Hence, the gradient can serve the purpose of gravitational force in our analogous example.

Second, we need momentum force. The choice of momentum is crucial. If the momentum is too small compare to gravitational force, the gravity pulls the satellites down to the earth. If the momentum is too large, it drifts away the level curve (orbits track). In addition to the state parameter, we need to create another random parameter called "momentum" to our parameter space; where the momentum parameter endows with a probabilistic distribution assumed to be easy to compute and sampled.

2.3.3 Geometry of Dual Space

We extend the original state space (space of interest) with additional momentum space. That is for every state variable q takes the value q_i at iteration i, we have an auxiliary momentum variable p with value q_i with same dimension where p_i is sampled from momentum distribution of our choice. To illustrate the idea of extended space, consider an uni-variate state variable q where it takes values from $\mathbb{R}$

Figure 1: This is our original state variable takes the value in the yellow dot before introduce auxiliary variable

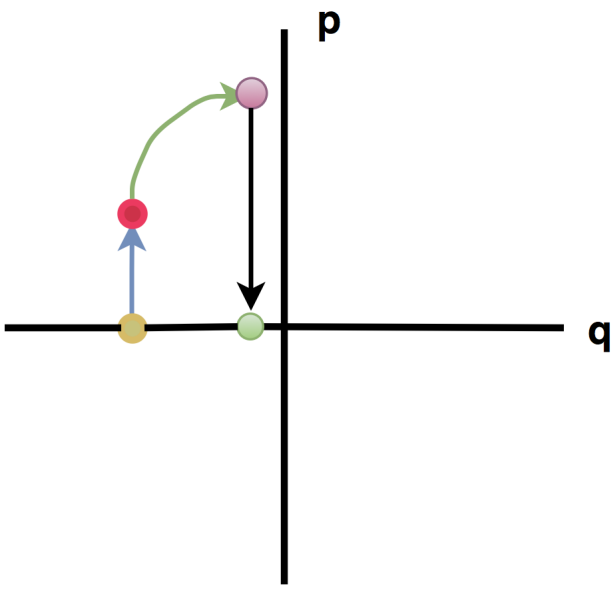

Figure 2

Figure 2 shows what happen for one iterations of HMC, where yellow spot q_i is the value for state space at current iteration i; then we sample a value p_i for momentum; then new position (q_i, p_i) which is the red dot will be our dual state. Then, we let the red dot travelling along the green line to purple dot then map back to state space. The new state green dot will be either accept or reject based on the acceptance criteria.

In our analogous example above, one can think the origin of Figure 2 to be our earth, and the green travelling curve to be our satellite orbiting path. Immediately, one notices that whether the new state green dot is accepted or not depends on two things: (1) the position of red dot (2) the position of purple dot. More specifically, the value of the sampled momentum (which will represents the vertical axes position for red spot) and how much do we trajectory along the green line to purple dot.

2.3.4 Hamiltonian Dynamics

Hamiltonian System is described by a scalar function $H(\boldsymbol{q}, \boldsymbol{p}, t)$ where $\boldsymbol{q}, \boldsymbol{p}$ denotes the state and momentum, and t denotes the time (in above section, t describes how much time we spend on travelling along the

green line). Note that Hamiltonian is not time dependent; $H(\boldsymbol{q}, \boldsymbol{p}, t) = H(\boldsymbol{q}, \boldsymbol{p})$. It means that regardless how long we travelling on that green line, the value of H doesn't vary. This implies that we can represent the green line to be part of a level curve. For any level curve, we can describe all dual points.

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

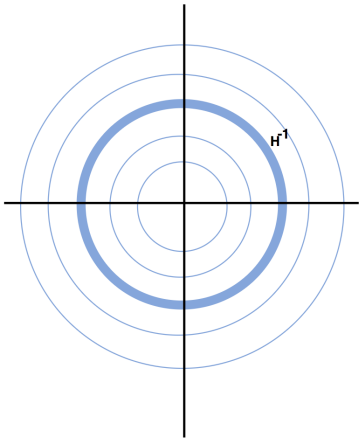

Figure 3

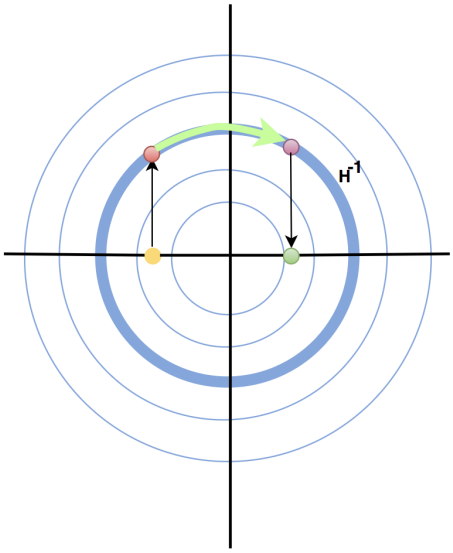

Figure 4

The Hamiltonian equation:

$$H(q, p) = U(q) + K(p) \tag{2}$$

Where potential function $U(q)$ is minus log of interest un-normalized density; while kinetic function $K(p)$ is choice of our own. In general, the most standard choice of K would be

$$K(p) = \frac{p^T M^{-1} p}{2} \tag{3}$$

Where M is a symmetric semi-definite mass matrix. We usually use matrix M as co-variance matrix for momentum distribution. So we can think of momentum has distribution $p \sim Gaussian(\mathbf{0}, M)$.
Define the joint distribution of momentum and state to be as

$$\pi(q, p) \sim exp(-H(q, p)/T) \tag{4}$$

This joint probability is in form of Canonical ensemble, it can help us to sample a point on any given level curve. Then one can perform marginalization $\pi(q) = \sum_p \pi(q|p)\pi(p)$. T is the hyper-parameter called temperature.

Hamiltonian dynamics can also be understood as the result from a certain set of differential equations satisfies the following two conditions

$$\frac{dq}{dt} = \frac{\partial H}{\partial p} \tag{5}$$

$$\frac{dp}{dt} = -\frac{\partial H}{\partial q} \tag{6}$$

In HMC, we would have the following results:

$$\frac{dq}{dt} = M^{-1} p \tag{7}$$

$$\frac{dp}{dt} = -\frac{\partial U}{\partial q} \tag{8}$$

Solving the above differential equation, one can derive the function $q(t), p(t)$. This two functions will tell us the value of (q, p) for each time on a given level curve.

2.3.5 Leapfrog

Once we define our probability distribution for momentum, we need the method of trajectory; namely, how do we traveling along a given level curve. As shown above, travelling along the level curve is equivalent to solve the differential equations of (7), (8). Leapfrog is an iterative method of numerical integration that updating q, p at each time step. During the iteration, we can collect all approximated values of (q, p) along

the curve. The leapfrog method is as follows:

Algorithm 1: Leapfrog

input : step size ϵ, step length L, initial position (q_0, p_0)
$p = p_0 - \epsilon * \frac{\partial U}{\partial q}(q_0)/2$
for $i : 1 \rightarrow L$ **do**
  $q \leftarrow q + \epsilon * p$
  $p \leftarrow p - \epsilon * \frac{\partial U}{\partial q}(q)$
end
$p = p - \epsilon * \frac{\partial U}{\partial q}(q)/2$
return $(q, -p)$
; // negation of momemtum isn't part of leapfrog but in HMC, it's used for proposal symmetric

2.3.6 Simple HMC Algorithm

Just like all other MCMC method, there is criteria for the newly sampled state to be accepted or not. Starts with (q_0, p_0), by performing leapfrog method, we will have a updated proposed dual point (q, p) that is at end of the travelling along the level curve. This proposed point is accepted as next state with probability

Algorithm 2: AcceptanceCriteria

input : proposed position (q, p), initial state (q_0, p_0)
$u \sim unif(0, 1)$
if $u < exp(-H(q, p) + H(q_0, p_0))$ **then**
  return q
end
else
  return q_0
end

If the proposed state is not accepted, the next state is the same as the current state. The full algorithm is performed as follow:

Algorithm 3: Simple HMC

input : initial state q_0, step size ϵ, step length L, num of iterations N,mass matrix $\boldsymbol{M}$, potential U
$i = 0$
$p_0 \sim N(\mathbf{0}, \boldsymbol{M})$
$S = \{\}$; // set to save all samples
while $i < N$ **do**
  $(q, p) \leftarrow Leapfrog(\epsilon, L, (q_0, p_0))$
  $q \leftarrow AcceptanceCriteria((q, p), (q_0, p_0))$
  $S \leftarrow S \cup \{q\}$
  $p_0 \sim N(\mathbf{0}, \boldsymbol{M})$; // sample a new momentum
  $q_0 \leftarrow q$; // update the current state
  $i++$
end
return (q, p)

3 Variation on Simple HMC

3.1 Motivation for Variation

3.1.1 The Importance of Hyper-parameters

We see that the result of Hamiltonian MC is very sensitive to many hyper-parameters; it highly depends on trajectory length L, step size ϵ, and our choice of kinetic function K, namely, the mass matrix M.

First, let's see how a mass matrix effects our sampling process. Consider uni-variate state variable $q \sim N(0,1)$ to be the distribution of our interests. Supposed we choose our mass matrix $\tilde{\boldsymbol{M}}$ to be diagonal matrix where the diagonal entry has value 10. The kinetic function will be $K(p) = p^T \tilde{\boldsymbol{M}}^{-1} p/2$. The momentum variable $q \sim N(0, M)$. Here, we see that q has a distribution very different than the true distribution of interest. Our dual space will be the contour of bi-variate normal distribution as follows.

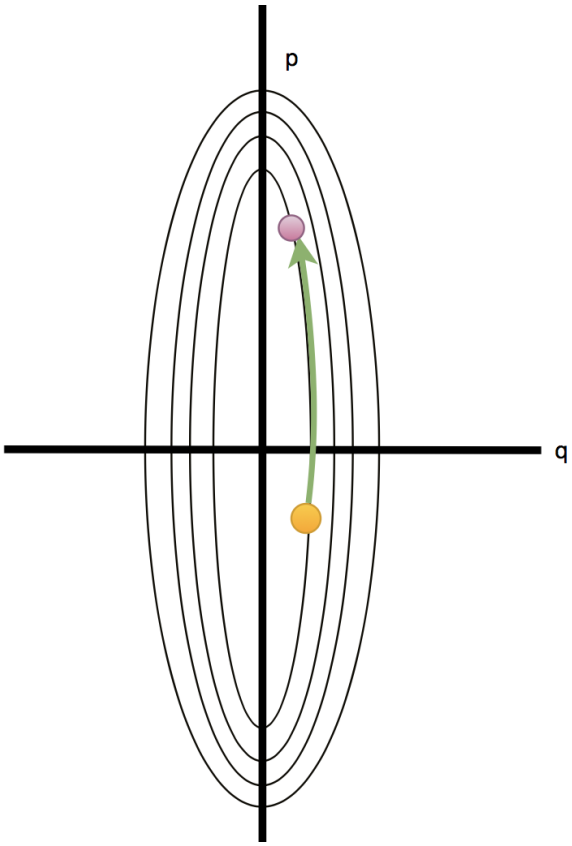

Figure 5

We see that starts with yellow dot as our original state, we trajectory along the green line for quit while and end up with purple. However, the travelling of $(q_0, p_0) \to (q, p)$ doesn't give us any thing interest since the magnitude between q_0 and q is very small. It means that large amount of computation when doing leapfrog moves us very little.

Consider another extreme situation, let our target distribution to have large variance (heavy tail). Heavy tail target implies that typical set might likely lies towards the tail. If the momemtum distribution has small variance. Our dual space will be like more horizontally squashed:

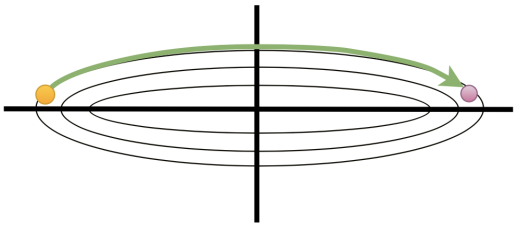

Figure 6

If one starts at yellow dot, which is at the tail of the distribution of interest, it requires less time to reach another end (requires smaller L) to collect efficient samples from another end. However, if our kinetic is badly chosen, the above contour will be more uniform sphere, then travelling from yellow to purple requires much longer time. However, if contour is so horizontally squashed, we are unlikely to explore the level curves lies outer leads to incomplete informative samples

This two extreme cases illustrates how does the choice of mass matrix effect the level set of our dual space. One may notice that the trajectory length L (how long we travel along a given level curve) should be tuned accordingly to the dual space. In other word, the value of L length might work for some level curve but not work well at others.

Third, the step size ϵ is also crucial to determine how well one travelling along the given level curve. The following two figures shows how step size can effect the approximation of leapfrog along the curve.

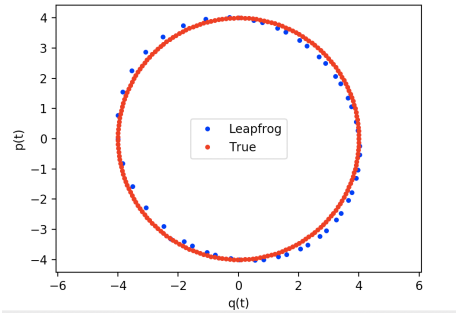

Figure 7: $\epsilon = 0.5$

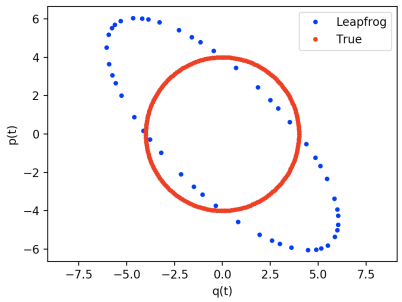

Figure 8: $\epsilon = 1.5$, large step size leads to low density region of dual space and leads to rejection

Note that small ϵ isn't always the best choice because small step size requires large number of jumps for each run of leapfrog to travel sufficiently long distance. Small step size leads to waste of computational resources and lead to inefficient exploration of state space.

3.1.2 Transition between Modes in Simple HMC

HMC can easily be trapped in isolated mode. Each point along a trajectory is distributed approximately according to the target distribution; so the trajectory are unlikely to pass through points that separate the modes. Consider the following experiment: Let our target distribution to be equally weighted mixture of Gaussian: $\frac{1}{2}N_1(1, 0.1) + \frac{1}{2}N_2(-3, 0.1)$. Two modes are so isolated where transition between two modes has very low probability.

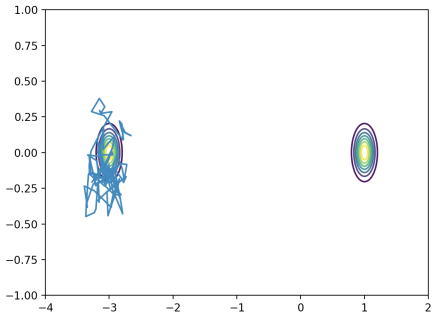

Figure 9: $\epsilon = 0.001, L = 100$; the Chain stuck in leftside mode

3.1.3 High Curvature in Dual Space

So far, we have only restricted ourselves to smooth dual space; however, in practice, we might encounter the situation where dual space has high curvature region (contains a sharp corner). The problem with high curvature is that, it's very difficult for leapfrog to step into the region accurately. If such region actually contains massive information of target distribution, we are missing them entirely. This leads the result to be bias. However, since the guarantee of convergence of Markov Chain, we expect it eventually step into the corner region to correct the bias. However, if the simulation is stop somewhere before that happens, we have incorrect results. (Following two Figures show that trajectory misses the corner)

(a) (b)

Figure 10: Momentum and State are two Mutually Independent Laplace Distributions random variables

4 Continuously tempered HMC

4.1 Motivation

Let us consider the Hamiltonian equation of HMC as $H(x,p) = U(x) + K(p)$. The minimum amount of kinetic energy needed to reach proposal state from initial state is called energy barrier.

$$B(x_1, x_2; U) := inf_{\gamma \in C^0}\{max_{0 \le t \le 1} U(\gamma(t)) - U(x_1) | \gamma(0) = x_1, \gamma(1) = x_2\} \tag{9}$$

Notice that the trajectory generated by Hamiltonian dynamics satisfies $U(x(t)) - U(x(0)) = K(p(0)) - K(p(t)) \le K(p(0))$ due to the conservation of energy. Thus, the trajectory will be unable to reach proposal state if the kinetic energy of initial state is lower than the energy barrier. Unfortunately, the energy barrier will be high if the proposal state and initial state are located in different modes of the target distribution. Therefore, Simple HMC algorithm works badly on multimodal target distribution.

4.2 Introduction

A common approach to deal with multimodal target distributions is to introduce a new temperature variable(β). Density function was flattened and energy barrier will become lower at high temperature. Compared with tempered MCMC algorithm, continuously tempered HMC use a continuously varying inverse temperature variable $\beta = \beta(u)$ to generate a geometric bridge between target density $exp(-\phi(x))/Z$ at $\beta = 1$ and prior density $exp(-\psi(x))$ at $\beta = 0$.

Extended Hamiltonian equation:

$$H(x,u,p,v) = U(x,u) + K(p,v) \tag{10}$$

$$= \beta(u)[\phi(x) + log(\xi)] + [1 - \beta(u)]\psi(x) - log|\frac{\partial \beta}{\partial u}| + \frac{1}{2}p^\intercal M^{-1} p + \frac{v^2}{2m} \tag{11}$$

Here, $log(\xi)$ is chosen to be an estimation of log(normalizing constant) and temperature control variable u is mapped to [s,1], $0 < s < 1$ via a smooth piecewise defined function $\beta : \mathbb{R} \to [s,1]$ for a pair of thresholds θ_1, θ_2 with conditions:

$$\beta(u) = s \ \ \forall \ |u| \ge \theta_2 \tag{12}$$

$$s < \beta(u) < 1 \ \ \forall \ \theta_1 < |u| < \theta_2 \tag{13}$$

$$\beta(u) = 1 \ \ \forall \ |u| \le \theta_1 \tag{14}$$

$$0 < \theta_1 < \theta_2 \tag{15}$$

An addition momentum variable v with mass m is introduced to keep Hamiltonian system retain a symplectic structure and Hamiltonian equation remain separable. Thus, the dynamic can be simulated efficiently with a leapfrog integrator.

4.3 Algorithm

Algorithm 4: Leapfrog

input : step size ϵ, step length L, initial position (q_0, u_0, p_0, v_0)
$p' = (p, v)$
$q' = (q, u)$
$p' = p'_0 - \epsilon \times \frac{\partial U}{\partial q'}(q'_0)/2$
for $i : 1 \rightarrow L$ **do**
  $q' \leftarrow q' + \epsilon \times p'$
  if $i! = L$ **then**
    $p' \leftarrow p' - \epsilon \times \frac{\partial U}{\partial q'}(q')$
  end
end
$p' = p' - \epsilon * \frac{\partial U}{\partial q'}(q')/2$
return $(q', -p')$

Algorithm 5: Acceptance Criteria

input : proposed position (q', p'), initial state $(q'_0, p'_0) = (q_0, u_0, p_0, v_0)$
$u \sim unif(0, 1)$
if $log(u) < -H(q', p') + H(q'_0, p'_0)$ **then**
  return q'
end
else
  return q'_0
end

If the proposed state is not accepted, the next state is the same as the current state. The full algorithm is performed as follow:

Algorithm 6: Continuously tempered HMC

input : initial state q'_0, step size ϵ, step length L, num of iterations N,mass matrix $\boldsymbol{M}$, potential U, kinetic K, coefficient c
$i = 0$
$p_0 \sim N(\mathbf{0}, \boldsymbol{M})$; `// M is a squared matrix with nrow = dim+1`
$S = \{\}$; `// set to save all samples`
while $i < N$ **do**
  $(q', p') \leftarrow Leapfrog(c, \epsilon, L, (q'_0, p'_0))$
  $q' \leftarrow AcceptanceCriteria((q', p'), (q'_0, p'_0))$
  $S \leftarrow S \cup \{q'\}$
  $p^* \sim N(\mathbf{0}, \boldsymbol{M})$
  $p'_0 \leftarrow c \times p^* + \sqrt{1 - c^2} \times p'_0$; `// Partially/fully resample a new momentum`
  $q'_0 \leftarrow q'$; `// update the current state`
  $i++$
end
return (q', p')

4.4 Experiments

4.4.1 Two-dimensional bimodal Gaussian distribution

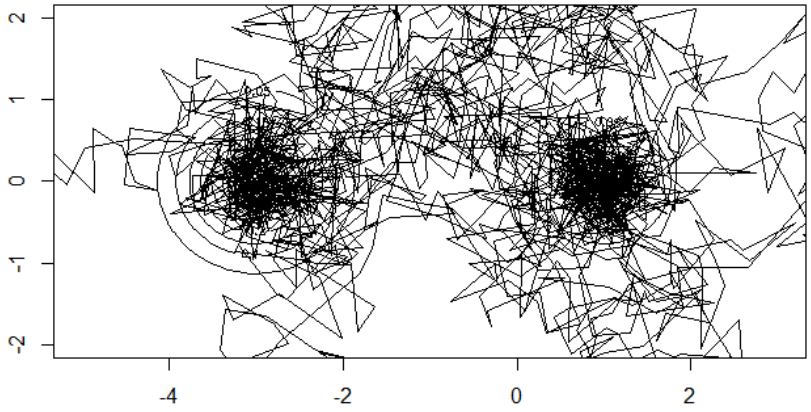

Figure 11: Contour plot and trajectory (continuously tempered HMC)

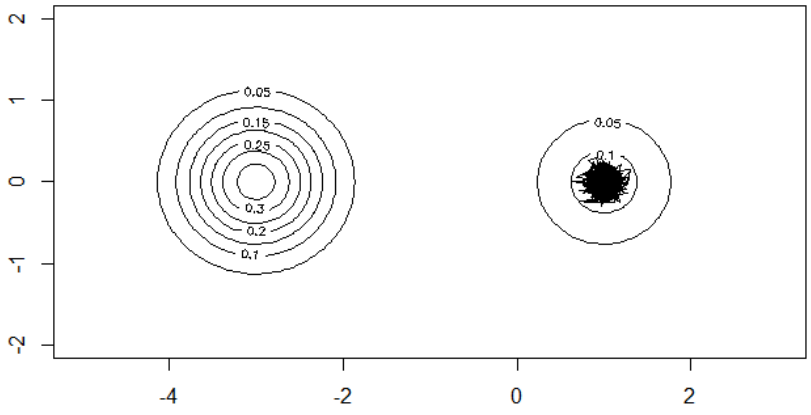

Figure 12: Contour plot and trajectory (simple HMC)

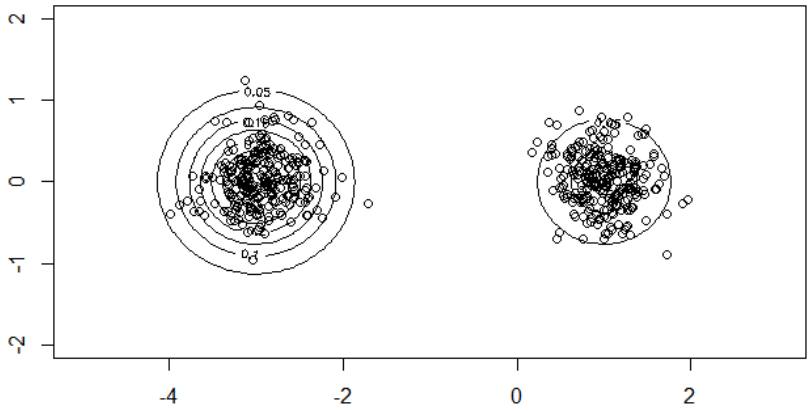

Figure 13: Contour plot and sampled $\mathbf{x}$ with $\beta(u) \approx 1$

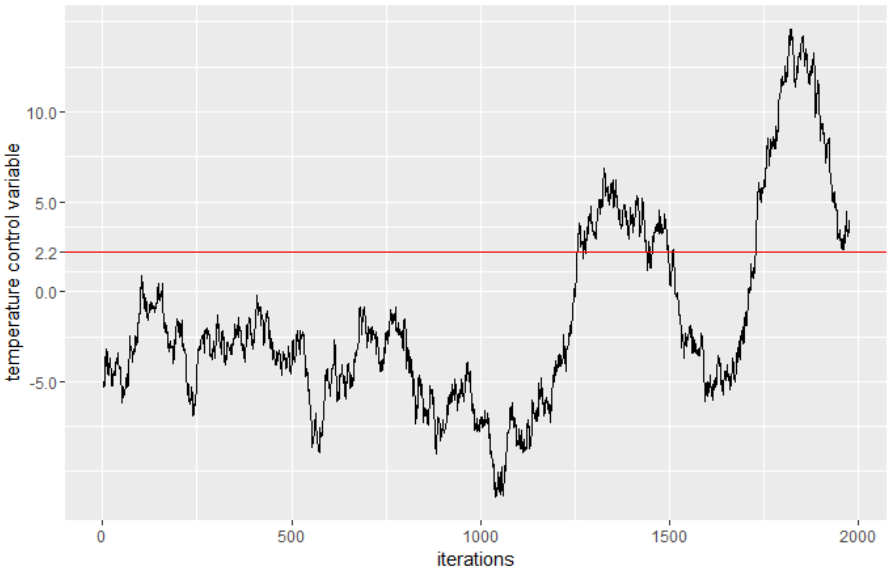

Figure 14: Temperature control variable

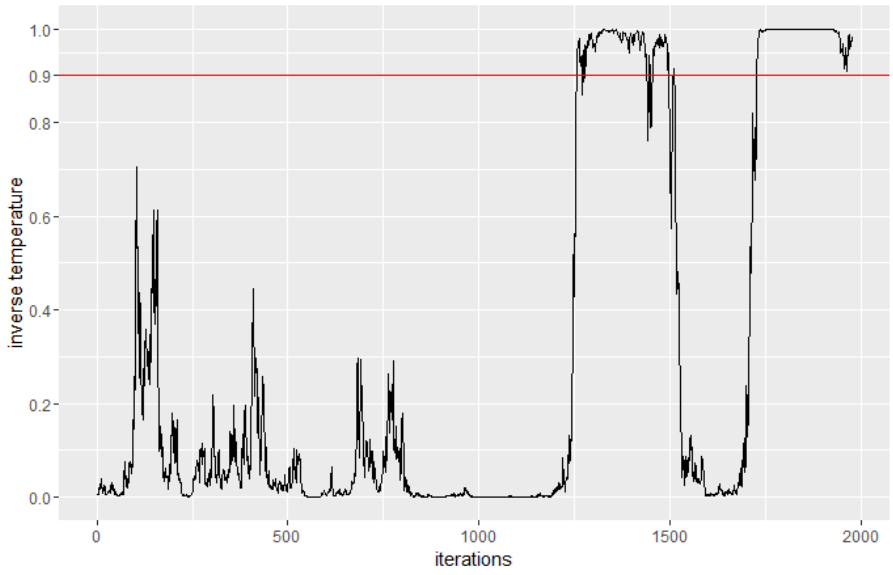

Figure 15: Inverse temperature $\beta(u)$

We start with a multivariate bi-modal Gaussian distribution to see the performance of Hamiltonian dynamics on extended joint space (x, u). The prior density was chosen to be a uni-modal multivariate Gaussian distribution with mode located between two modes of target distribution. The initial state x_0 was selected to lie on one mode of target distribution to check if the trajectory is able to reach another mode. ξ was set to be equal to normalizing constant of target density to create the "best-case" for continuously tempered HMC experiment. Besides, logarithm Sigmoid function was chosen to be the inverse temperature function and initial value of temperature control variable. u was chosen to be -5 to make $\beta(u) \approx 0$.

Running length was set to be 2000 since several tests reflect that 1000 iterations are not large enough for extended Hamiltonian dynamics to build a bridge between isolated modes.

As shown in Figure 11, extended Hamiltonian dynamics generate a path between two modes of target distribution with enough exploration. As long as the trajectory reaches another mode, there is an high probability that the proposed state is distributed approximately as the target distribution. Figure 13 shows all the sampled configuration states $\mathbf{x}$ with associated $u > 2.2$ (so that inverse temperature > 0.9). We can see most points lie inside the outermost contour line and most of them lie around the center (mode). Some points lie outside since the choice of inverse function results in that inverse temperature could only converge to 1. Figure 14 and Figure 15 shows how temperature control variable and inverse temperature varies along the trajectory. Temperature control variable tends to decrease before around 1100 iterations and then increase with wide fluctuation. It reflects that the dynamics start working as expected after around 1100 iterations. Thus, as shown in Figure 14, sampled states (with $\beta(u) \approx 1$) that asymptotically converge in distribution to the target has a relatively low acceptance rate. (24.27% for this experiment). Figure 12 shows the trajectory simulated by standard HMC algorithm. Standard HMC is easily trapped in one mode while continuously tempered HMC is able to identify isolated modes and generates a path between them.

5 High Curvature

In this section, we introduce a particular class of distributions that illustrates the weakness of simple HMC. This type of distribution is introduced in Radford Neal's earlier paper[2], and given a name "Funnel Distribution". This is a rather simple distribution but appears in practical situation frequently. Unfortunately, funnel distribution has exhibited many pathologies that leads to the failure of simple HMC and also many other Monte Carlo methods, including MCMC, Gibbs Sampling, etc. Let our target distribution to be π, D to be the dimension of our random variable, and funnel distribution is defined as follows:

$$\pi(\boldsymbol{q}) = \prod_{i=1}^{D} N(q_i|0, e^{q_0}) N(q_0|0, 9) \tag{16}$$

Note that this distribution, in face, to be very simple to be sampled. One first needs to sample $q_0 \sim N(0, 9)$, then independently sample each component of $\boldsymbol{q}$ such that $q_i \sim N(0, e^{q_0})$. However, if one wants to simulate this distribution, it turns out to be very difficult. The reason is that such distribution has multiple spatial scales. It has smooth, flatten and high volume in the region where density to be small; while it has sharp, spike and low volume in the region where density to be large. It's level curve to be like this

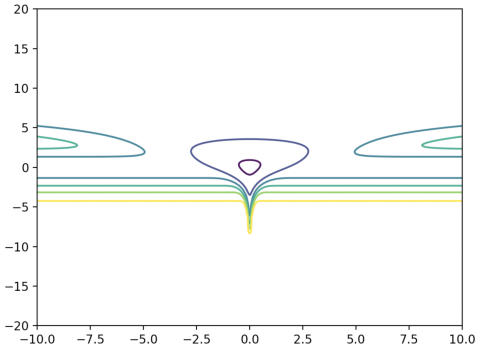

Figure 16: x-axis is one of our q_i and y-axis is our variabel q_0

In term of graphical probabilistic model, we can graph as follows:

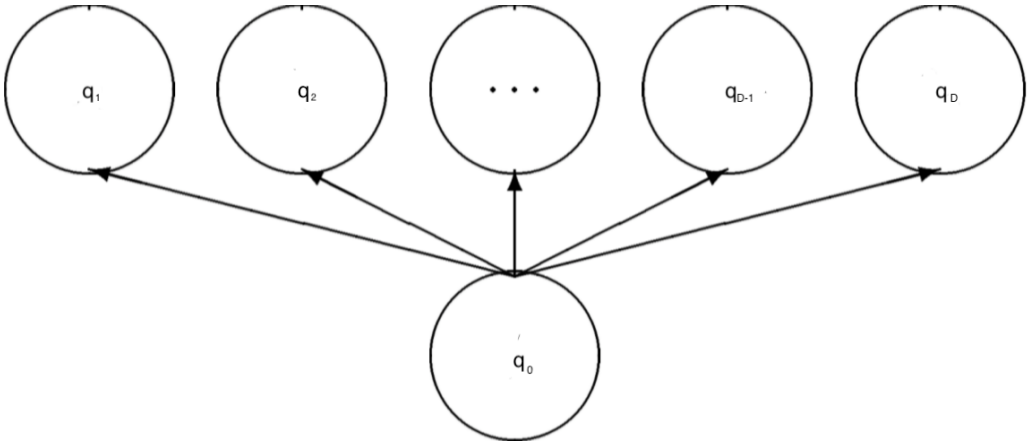

Figure 17

The distribution has illustrated the simplest version of hierarchical model (with only one level of hierarchical). We have a global parameter q_0 and independent local parameters q_i that conditional depends on q_0. The model can be further complicated by adding more hierarchical levels (introducing more conditional random variables that depends on each q_i) or having more complicated groups (connect some q_i to build cliques); such model creates a stronger correlations between each random variables. As the value of each random variable varies, their correlation also varies strongly. As a result, near each point, it's neighborhood has its own scales and rotations.

We perform simple HMC on the one level hierarchical model.

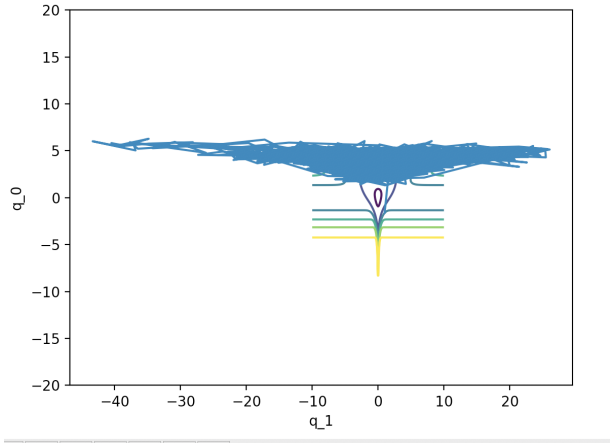

Figure 18: $\epsilon = 0.3$

Notice that the region we explore is focus on the low density region. That region is smooth with high volume. This gives us biased estimation on the value of q_0.

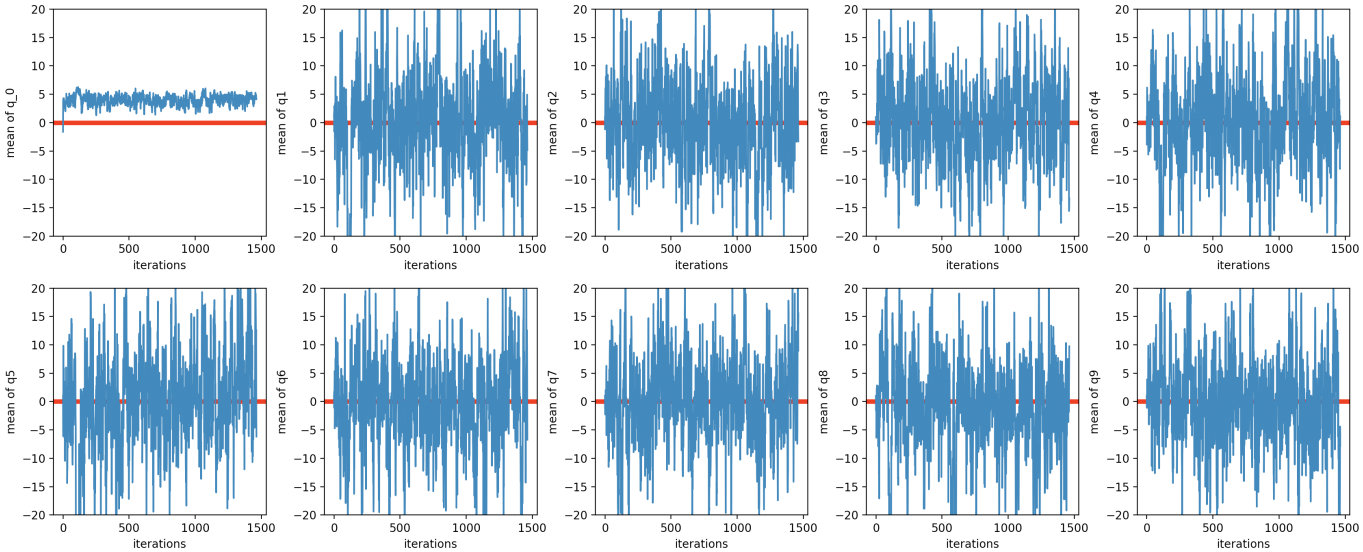

Figure 19: The red line is the true value

The left top corner plot shows that our estimation of q_0 is only near value 5 (The low density region).

Clearly, the failure of the above experiment caused by the badly tuned hyper-parameter ϵ. It's too large and if the algorithm travels to the high density but low volume region, big step size leads to the region of high rejection. Hence, the algorithm is forced to travel upwards to the high volumn region when path is smooth. (In face, we let our initial value to be right in the typical region; but it quickly jumps to upper region as one may notices the travelling line)

To ensure the algorithm travel nicely in the typical region, we requires the ϵ to be very small. However, the small step size traps our algorithm in the sharp region and for a very long run of chain, we still stuck in that region and fails to fully explore and once again we have biased estimation.

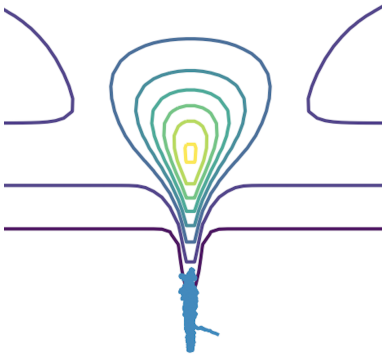

Figure 20: $\epsilon = 0.01$

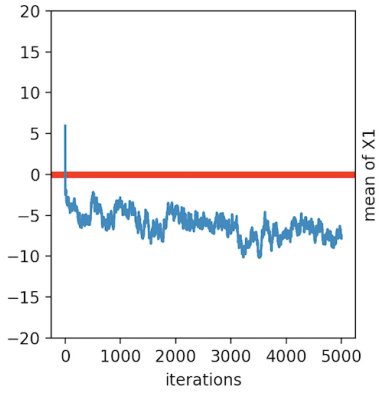

Figure 21

Funnel Distribution is a typical example that a single fixed ϵ fails when the target distribution has different curvatures around some points' neighborhood. One popular approache to solve such problem is to adaptive correct the mass matrix. We introduce this approach in section 6.

6 Riemannian HMC

As mention earlier, when we introduce momentum variable into our model, our algorithm will be travelling in the duel space. When the target has high curvature region, by choose the correct mass matrix (co-variance matrix for momentum variable), the resulting dual space will be less likely to have high curvature. In face, with perfect choice of mass matrix, we will end up with a perfect uniformly hyper-sphere dual space, where the exploration space is smooth with same curvature everywhere. The perfect uniformly hyper-sphere dual space occurs when the momentum distribution happens to be exactly of our target distribution.

6.1 Globally Correct Mass Matrix

Based on above idea, we starts with a naive approach of mass matrix adaptive method. Let the chain runs for several iterations, then use the most recent samples collected to estimate the empirical co-variance matrix of the target distribution, then change our mass matrix to be this empirical estimation and sample our momentum from this new co-variance matrix.

Algorithm 7: Global Adaptive Mass Matrix HMC

$i \leftarrow 0$
$S \leftarrow \{\}$
while *not converge* **do**
  Run a single iteration of Simple HMC
  $S \leftarrow S \cup \{q_i\}$; `// Collect the sample from this iteration run of HMC`
  $i \leftarrow i + 1$
  if $i\%\ 1000 == 0$; `// Estimate the target co-variance every 1000 iterations` **then**
    $\boldsymbol{M} \leftarrow Covaraince(S)$
    ; `// Sample covariance matrix; then use it as new mass matrix`
  end
end

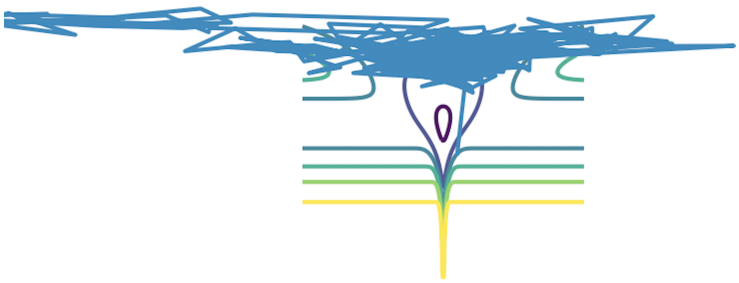

Figure 22: $\epsilon = 0.3$ with global adaptive mass matrix HMC

Unfortunately, this doesn't gives us a good result; in face, it doesn't improve compared to our simple HMC at all. We still have the biased estimation of q_0. The very reason is that the funnel distribution is not global correlations. Every time when we estimate the target distribution, we use all of the past samples and approximate a global co-variance matrix of the target distribution; which it can estimate badly in the

local region as it over generalizes. In funnel distribution, we have a global variable q_0 but there is local conditional variable q_i features strongly local correlation.

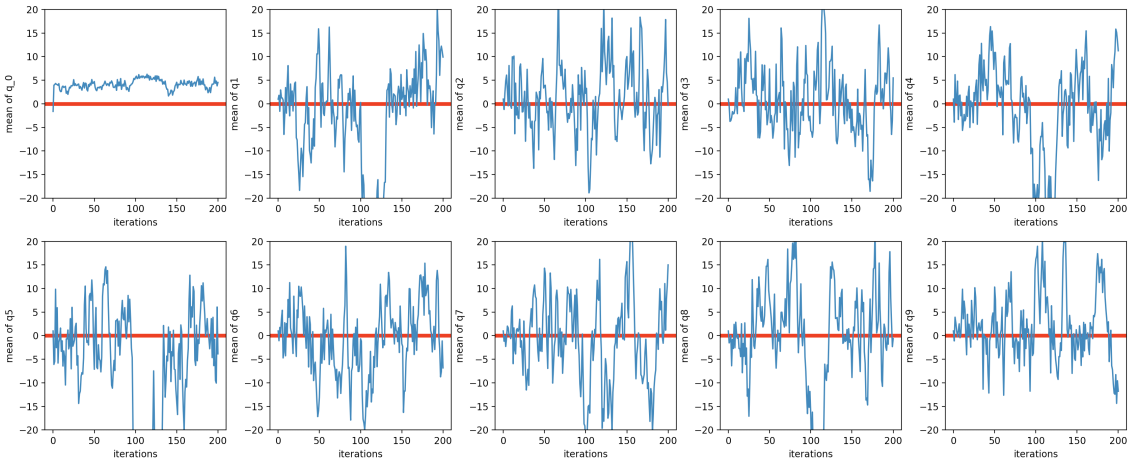

Figure 23: Globally Adaptive HMC still gives us biased estimation on q_0

6.2 Locally Correct Mass Matrix

Motivated by the failure of globally adaptive mass matrix HMC, as a global estimation of co-variance matrix of target distribution can be locally miscalibrated; we turn to the locally adaptive mass matrix. More specific, instead of using the historical samples to compute the global co-variance, for each point q, we look around some small neighborhood around that point, and estimate the variance within that neighborhood, this estimation will be used to sample the corresponding momentum variable p. In this case, we expect that within some small neighborhood around q, its corresponding momentum p will have the same variance. Though, the global dual-space not perfect uniformly hyper-sphere, but locally, it's smooth and uniformly.

6.2.1 Curvature and Hessian

We explain in the high level intuition that why using the local estimation of variance can make the dual-space locally uniform and smooth.

Consider one dimensional space of our target distribution. Let q be the current position of our algorithm. The second derivation of our negative log likelihood function (our potential function) evaluated at q is $U''(q)$. The curvature defined on that point is $\kappa = \frac{|U''(q)|}{(1+(U'(q))^2)^{\frac{3}{2}}}$. The second derivative can give us an idea of how a graph is shaped (how does the tangent line changes near that point). The larger second derivative implies the larger curvature, hence the graph is sharper near the point (vice verse, small second derivative, smaller curvature and the graph is smoother near the point). If the second derivative is large, the potential function has very different value near q, the value of the function is more sensitive to the permutation of the q. This indicates a small variance near that point. If we use the inverse of the second derivative, it will be a good estimator of the variance within the neighborhood. If we sample q from a distribution with similar variance, we expect the shape of momentum-position dual space within that neighborhood to be more uniformly and smooth.

6.2.2 Relationship with Manifolds

Unfortunately, directly using the second derivative as a variance for the momentum distribution is way too naive because we can't always guaranteed that second derivative is always positive unless the potential function is strictly convex within the neighborhood. Hence, relax the assumption of using only the second derivative evaluated at the current position, we make it more generalized such that we require a local map that maps the current position to a valid variance. In high dimension space, we are dealing with co-variance matrix, hence we need a map such that $G : q \to \tilde{\boldsymbol{M}}$ where $\tilde{\boldsymbol{M}}$ is space of all valid co-variance matrix for momentum (namely, the mass matrix); and $G(q)$ as co-variance matrix to sample p, must to have the following properties:

- $G(q)$ is a symmetric matrix
- $G(q)$ is a positive definite matrix
- $G(q)$ is bi-linear map such that $G(q)(aX + bY, Z) = a\text{G(q)}(X, Z) + b\text{G(q)}(Y, Z) \forall a, b \in \mathbb{R}$

Notice that if our momentum variable has distribution with such co-variance matrix that depends on the current position q, the kinetic function is in form of

$$K(q, p) = \frac{1}{2} p^T (G(q))^{-1} p + \frac{1}{2} log|(G(q)| \tag{17}$$

Since $G(q)$ is matrix, then it satisfies that $p^T G(q) p$ for $p \in \mathbb{R}^D$. This is can be seen as $G(q) : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$. Along with three properties above, this is exactly what we need to define a Riemannian Metric. In other word, this is equivalently to say that, we treat the dual-space as a manifolds, and at current position q, let $\phi : [a, b] \to M$ to be some curve pass q at $c \in [a, b]$, we define a tangent plane $T_q M := \{\frac{d}{dt}|_{t=c} \phi(t); \forall \phi \text{ that pass through point } q\}$ (where $\frac{d}{dt}|_{t=c} \phi(t)$ is the velocity vector at point q) such tangent plane equipped with an inner product $G(q)$ (Riemannian Metric). In some literature, the locally correct mass matrix approach is also called Riemannian Manifolds HMC (in these literature, the motivation arises from Manifolds perspective but it will results in the same conclusion as in this summary).

A local metric map $G : q \to \tilde{\boldsymbol{M}}$ with three properties is required to do locally correction. If $G(q)$ is a constant (let's say identity matrix),then there is no local correction nor global correction. It reduces the HMC to simple HMC. In the following sections, we introduce several local metric map.

6.2.3 Fisher Information Metric

Since we motivate our locally correction from local curvature perspective, we would like to take advantage of Hessian matrix. Moreover, Hessian Matrix already satisfies the symmetric and bi-linear properties. We show that Fisher Information Matrix is a proper choice of local metric map.

Theorem 1

If the target distribution is conditioned on some random variable θ (for the simplicity of notation, let $\pi(\boldsymbol{q}|\theta) = \pi(\boldsymbol{q})$). Then the Fisher Information Matrix is the expectation (under θ) of Hessian Matrix (w.r.t. $\boldsymbol{q}$) of the potential function.

Proof.

$$\nabla_{\boldsymbol{q}}^2 U(\boldsymbol{q}) = \nabla_{\boldsymbol{q}}^2 - log(\boldsymbol{q}) \tag{18}$$

$$= \nabla(\frac{\nabla\pi(\boldsymbol{q})}{\pi(\boldsymbol{q})}) \tag{19}$$

$$= \frac{\pi(\boldsymbol{q})\nabla^2\pi(\boldsymbol{q}) - \nabla\pi(\boldsymbol{q})\nabla\pi(\boldsymbol{q})^T}{\pi(\boldsymbol{q})^2} \tag{20}$$

$$= \frac{\pi(\boldsymbol{q})\nabla^2\pi(\boldsymbol{q})}{\pi(\boldsymbol{q})^2} - \frac{\nabla\pi(\boldsymbol{q})\nabla\pi(\boldsymbol{q})^T}{\pi(\boldsymbol{q})^2} \tag{21}$$

Take the Expectation under. θ on both side yields

$$\mathbb{E}(\nabla_{\boldsymbol{q}}^2 U(\boldsymbol{q})) = -\mathbb{E}(\frac{\pi(\boldsymbol{q})\nabla^2\pi(\boldsymbol{q})}{\pi(\boldsymbol{q})^2} - \frac{\nabla\pi(\boldsymbol{q})\nabla\pi(\boldsymbol{q})^T}{\pi(\boldsymbol{q})^2}) \tag{22}$$

$$= -(\mathbb{E}(\frac{\pi(\boldsymbol{q})\nabla^2\pi(\boldsymbol{q})}{\pi(\boldsymbol{q})^2}) - \mathbb{E}(\frac{\nabla\pi(\boldsymbol{q})\nabla\pi(\boldsymbol{q})^T}{\pi(\boldsymbol{q})^2})) \tag{23}$$

$$= -(\int \nabla^2\pi(\boldsymbol{q})) - \mathbb{E}(\nabla log(\pi(\boldsymbol{q}))\nabla log(\pi(\boldsymbol{q}))^T) \tag{24}$$

$$= -(\nabla^2 \int \pi(\boldsymbol{q})) + \mathbb{E}(\nabla log(\pi(\boldsymbol{q}))\nabla log(\pi(\boldsymbol{q}))^T) \tag{25}$$

$$= \mathbb{E}(\nabla log(\pi(\boldsymbol{q}))\nabla log(\pi(\boldsymbol{q}))^T) \tag{26}$$

The equation(26) is the definition of Fisher Information. □

Theorem 2

The Fisher Information Matrix is positive semi-definite.

Proof. Let $I_{ij} = \mathrm{E}_\theta\left[(\nabla \log \pi(\boldsymbol{q}))\,(\nabla \log \pi(\boldsymbol{q}))\right]$; and let $u = (u_1, \ldots, u_D)^\top \in \mathbb{R}^D$ be some nonzero vector

$$\sum_{i,j=1}^{D} u_i I_{ij} u_j = \sum_{i,j=1}^{D} \left(u_i \mathrm{E}_\theta\left[(\partial_i \log \pi(\boldsymbol{q}))\,(\partial_j \log \pi(\boldsymbol{q}))\right] u_j\right) \tag{27}$$

$$= \mathrm{E}_\theta\left[\left(\sum_{i=1}^{D} u_i \partial_i \log \pi(\boldsymbol{q})\right)\left(\sum_{j=1}^{D} u_j \partial_j \log \pi(\boldsymbol{q})\right)\right] \tag{28}$$

$$= \mathrm{E}_\theta\left[\left(\sum_{i=1}^{D} u_i \partial_i \log \pi(\boldsymbol{q})\right)^2\right] \geq 0\,. \tag{29}$$

□

Equation (26) also gives us an advantage in term of computational complexity. When compute the Hessian Matrix of potential function to be tedious, one can compute its gradient and take the dot product(The right hand side). Unfortunately, there are few short-cuts of such choice of local metric. First, the Fisher Information is only positive semi-definite, and still can be singular and leads to invalid values. Secondly, evaluating expectation over random variable θ will not be practical. Even we compute the empirical Fisher $\frac{1}{5}\sum_i p(\theta_i)\pi(\boldsymbol{q}|\theta_i)$. We need to know the distribution $p(\theta)$. Unless the target distribution is naturally equipped with such conditional random variable θ with a simple distribution to sample it, we couldn't take such advantages.

6.2.4 Soft Absolute Metric

This is a more generalized metric introduced by Betancourt[4] that relax the assumption of the conditional random variable requirement in Fisher Information metric and also forces the metric to be positive definite.

The Hessian matrix H is symmetric; by Spectral Theorem, it can be diagonalized as $H = Q\Lambda Q^T$ where Q is a matrix consists of orthonormalized eigenvectors and Λ is a diagonal matrix where the diagonal is the corresponding eigenvalues. Also note that, when all the eigenvalues are positive, then H must be a positive definite matrix. The idea lies in Soft Absolute Metric is that designing a differentiable function ϕ to approximate the absolute value function. We apply function ϕ to each of eigenvalues to force the eigenvalues to be positive. Moreover, we want to regularize the small value of eigenvalues to avoid the numerical instability. The approximation map is

$$\phi(\lambda) = \lambda \frac{exp(a\lambda) + exp(-a\lambda)}{exp(a\lambda) - exp(-a\lambda)} \tag{30}$$

Since exponential function positive all time, it assures that even when the true absolute value of λ is so close to zero, the transformed λ is bounded below by some positive number. The value of $a > 0$ is the approximation control that controls how close we want our map ϕ to be with true absolute function. As $a \to \infty$, $\phi(\lambda) \to |\lambda|$.

One needs to be more careful when implement the soft absolute metric as the exponential function might be easily explode and leads to numerical overflow. A more numerical stable implementation is as follows:

If $\lambda > 0$, then multiply both of the denominator and numerator by $exp(a\lambda)$; then equation (30) becomes

$$\phi(\lambda) = \lambda \frac{1 + exp(-2a\lambda)}{1 - exp(-2a\lambda)} \tag{31}$$

If $\lambda < 0$, then multiply both of the denominator and numerator by $exp(-a\lambda)$; then equation (30) becomes

$$\phi(\lambda) = \lambda \frac{exp(2a\lambda) + 1}{exp(2a\lambda) - 1} \tag{32}$$

The Soft Absolute Metric will be $\phi(H) = Q\phi(\Lambda)Q^T$. The approximation control a can be set as hyper-parameter or it can be automatically adapted through the training based on the acceptance rate.

6.2.5 Smooth Metric

Based the same idea, we introduce another general metric called Smooth Metric. It is almost same as Soft Absolute Metric with one thing different. We use a different absolute value function approximation.

$$\psi(\lambda) = (\lambda^2 + a^2)^{1/2} \tag{33}$$

As $a \to 0$, $\psi(\lambda) \to |\lambda|$ and the function is bounded below by a. The approximator will be narrower than the Soft Metric so that for each λ, its approximated value will be slighter larger than the Soft Metric and further reduces the possibility of the numerical instability.

6.3 Locally Correction Algorithm

6.3.1 General Form

Let $G(q)$ be a local metric we choose. Now the kinetic function will be position-dependent and has the form of

$$K(q,p) = \frac{1}{2}p^T G(q)^{-1}p + \frac{1}{2}log|G(q)| \tag{34}$$

The Hamiltonian joint distribution becomes

$$H(q,p) = U(q) + K(q,p) \tag{35}$$

The sampling process is can be written as Gibbs sampling as

$$q^*|p \sim N(\mathbf{0}, G(q)) \tag{36}$$
$$p^*|q^* \propto exp(-H(q^* + \delta q^*, p^*)) \tag{37}$$

The momentum-position dual variable $(q^* + \delta q^*, p^*)$ is obtained through performing leapfrog algorithm. Note that Simple HMC leapfrog no longer will be working here because now, kinetic function involves variable q, the Hamiltonian is non-separable, implies that $\partial_q H$ requires differentiate through $K(q,p)$. And this needed to be taken into consideration when doing leapfrog. Here, we adopt the generalized leapfrog algorithm[5]

$$p^{n+1/2} \leftarrow p^n - \frac{\epsilon}{2}\partial_q H(q^n, p^{n+1/2}) \tag{38}$$
$$q^{n+1} \leftarrow q^n + \frac{\epsilon}{2}[\partial_p H(q^n, p^{n+1/2}) + \partial_p H(q^{n+1}, p^{n+1/2})] \tag{39}$$
$$p^{n+1} \leftarrow p^{n+1/2} - \frac{\epsilon}{2}\partial_q H(q^{n+1}, p^{n+1/2}) \tag{40}$$

6.3.2 Fixed Point Method

Note, equation (38) and equation (39) are defined implicitly.Use fixed point iteration. In more general case, we are solving situation like $g(x) = x$ Randomly choose a point x_0, consider a recursive process

$$x_{n+1} = g(x_n)$$

Theoretically, if $g(x) - x$ is a continuous function and $\{x_n\}$ converges, then it converges to the solution of $g(x) = x$. Hence, equation (38) and (39) will be solved through such recursive method. (the number of iteration is a hyper-parameters we choose beforehand)

Algorithm 8: Local Correct Mass Matrix HMC

input : G : a specified local metric;
ϵ : step-size;
I : iteration for fixed point method;
q_0 : the inital position;
U : the potentila function;
N : the number of interested samples

output: S : a set of samples obtained from HMC

$j \leftarrow 0$
$S \leftarrow \{\}$
for $j < N$ **do**
  $q_0 \sim N(0, G(q_0))$; // Sample a new momentum
  $L \sim randint(2, 200)$; // Sample a positive integer for number of leapfrog
  for $i < L$ **do**
    $p \leftarrow p_0$
    $q \leftarrow q_0$
    Perform equation (38) for I number of times to update q
    Perform equation (39) for I number of times to update p
    Perform equation (40) for a single time to update q
  end
  $u \sim uniform(0, 1)$
  $r \leftarrow -log(H(q, p)) + log(H(q_0, p_0))$
  if $r > log(u)$ **then**
    ; // Accept the proposed new position
    $S \leftarrow S \cup \{q\}$
    $q_0 \leftarrow q$
  end
end

6.4 Experiments with Locally Correction

We use both Soft metric, Smooth metric and Squared Hessian matrix ($H6TH$) to obtain the following results.

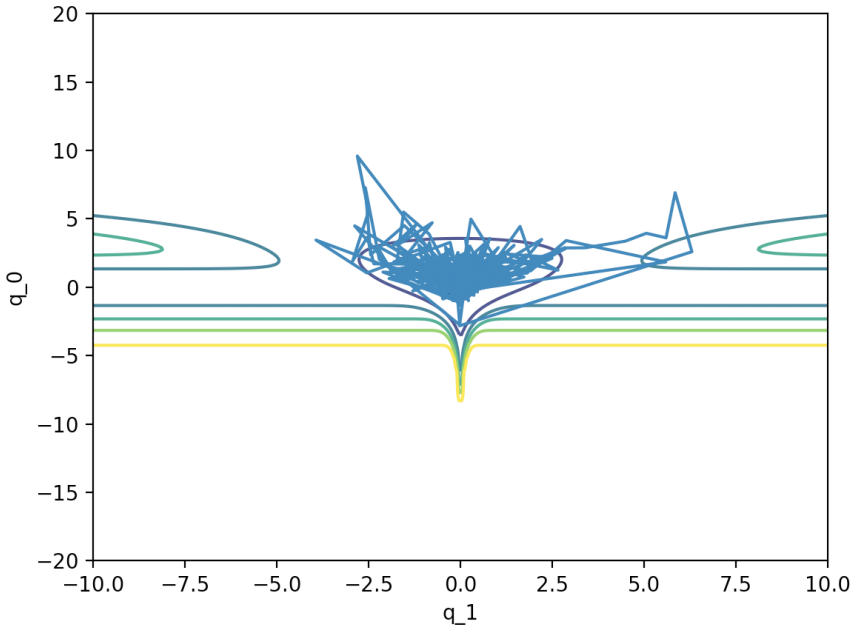

Figure 24: Soft Metric Exploration

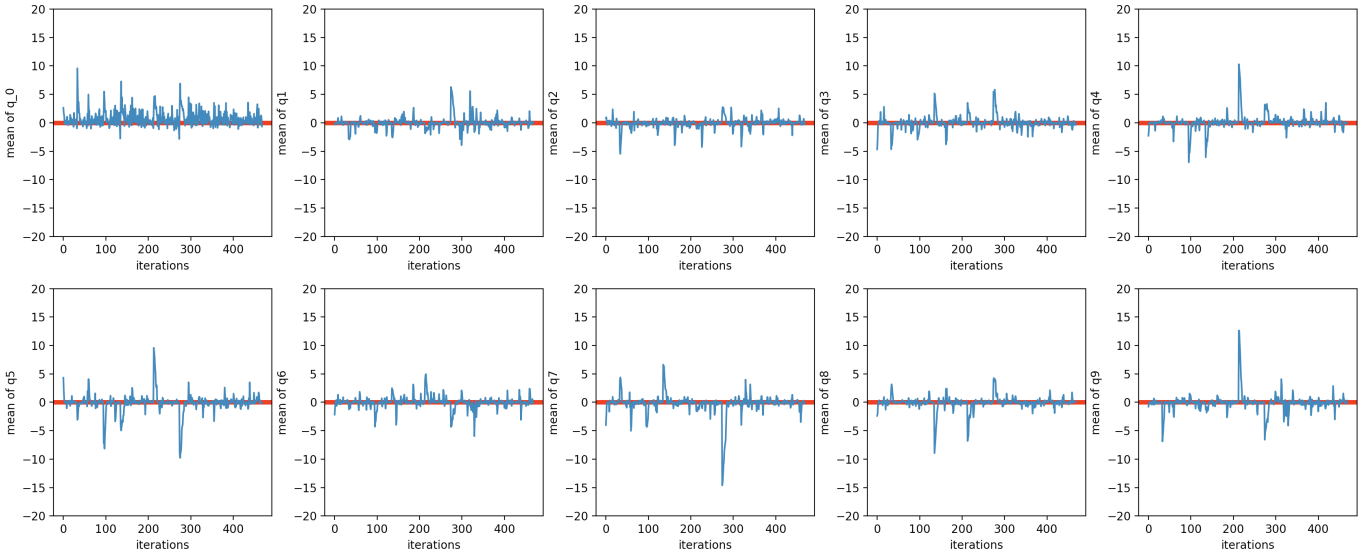

Figure 25: Soft Metric Estimation

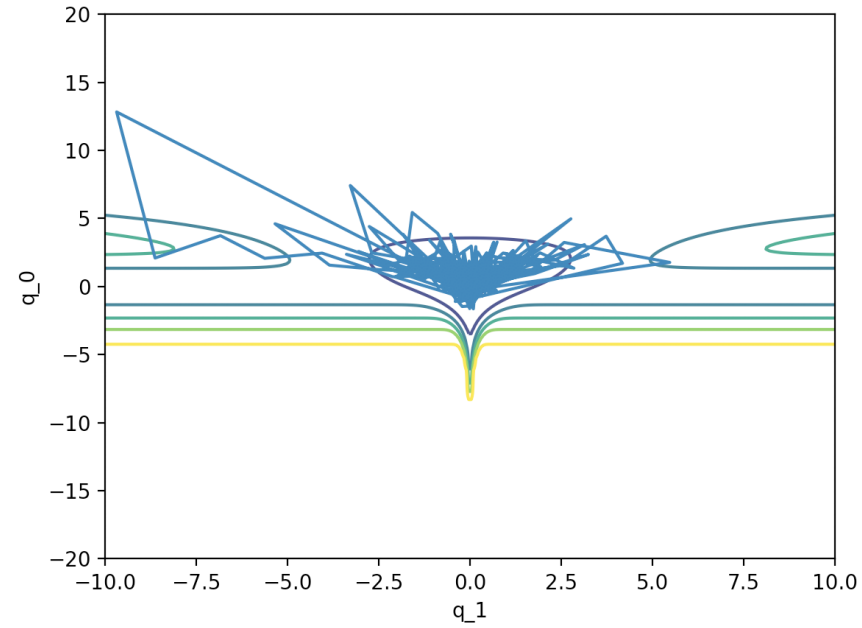

Figure 26: Smooth Metric Exploration

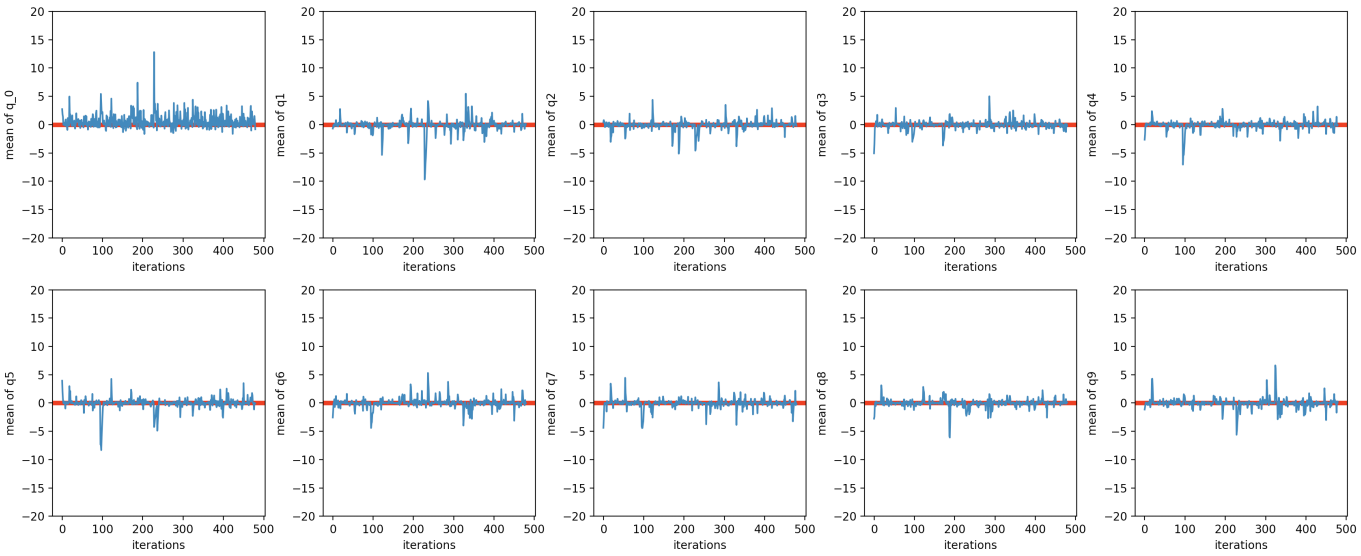

Figure 27: Smooth Metric Estimation

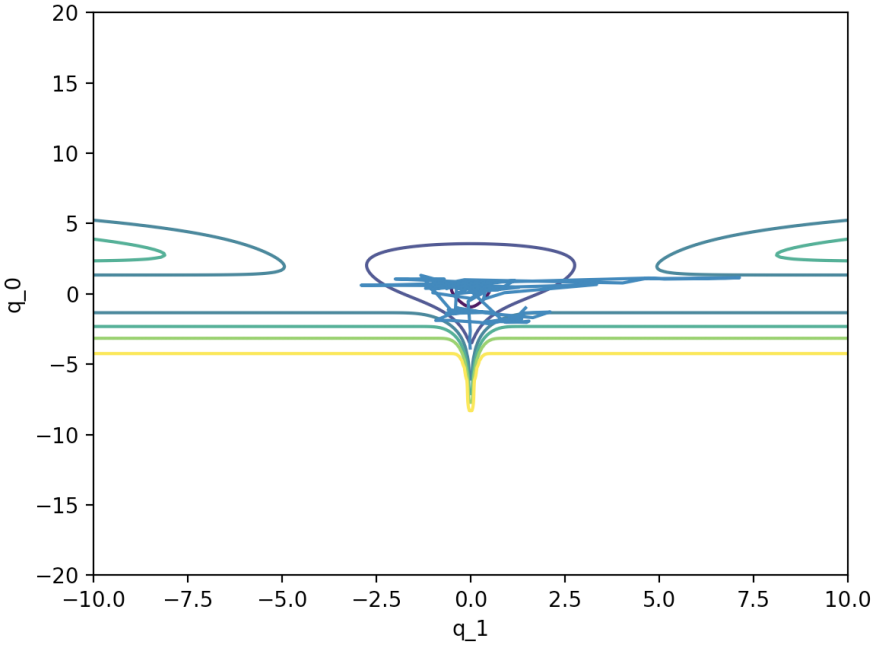

Figure 28: Squared Hessian Exploration

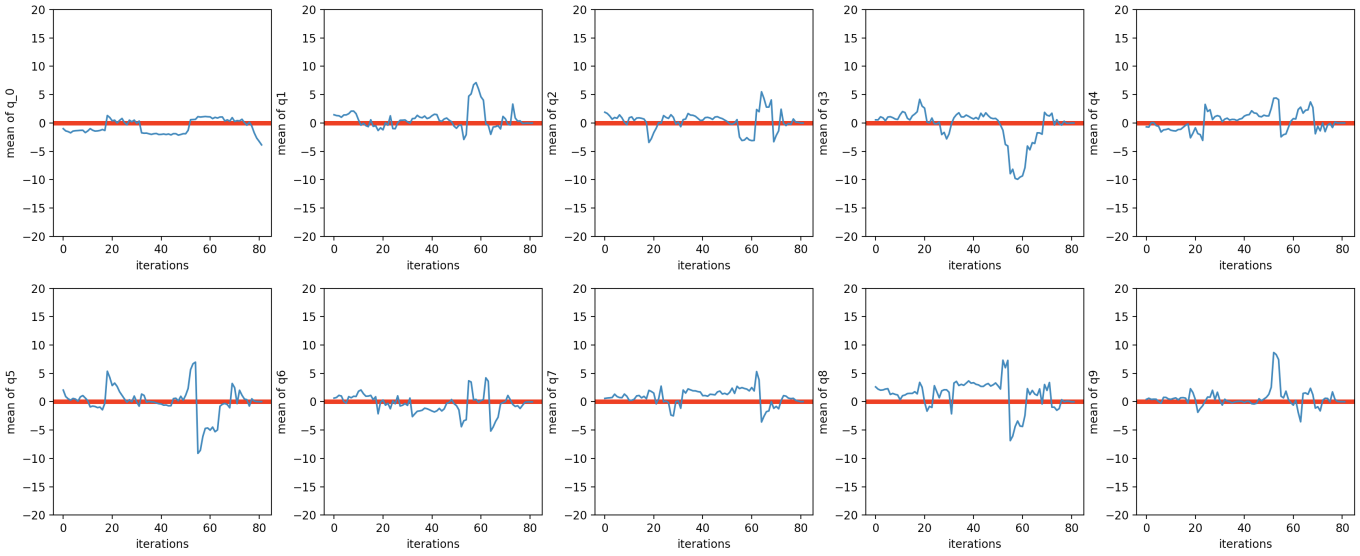

Figure 29: Squared Hessian Estimation

Three tings we notice from above is that allowing the momentum variable distribution depends on the current position gives much more flexibility of exploration even our step-size ϵ is fixed through the whole iterations. The algorithm is capable of maneuver through the shallow region and also travel in the high volume region. Local metric not only prevents the algorithm gives biased estimation (q_0); also, for unbiased estimation obtained from simple HMC, the locally correction HMC has much smaller varinces.

7 Adaptive HMC - tunning step size and leapfrog steps

7.1 Motivation

Hamiltonian is sensitive to the choice of initial step size(ϵ) and leapfrog steps(L). A poor choice of these parameters will result in a high rejection rate as well as poor mixing and high auto-correlation. To address this problem, finding a way to automatically tune two parameters in burning period is a necessity. Based on this idea, an approach of adaptive HMC was introduced.

7.2 Objective function

First, we need to define an objective function. Our goal is to find a group of step size and leapfrog steps that minimize the auto-correlation of the sampler. The choice of our objective function come from an objective measure called expected squared jumping distance(EJSD):

$$ESJD(\gamma) = \mathbb{E}_\gamma ||\boldsymbol{x}^{t+1} - \boldsymbol{x}^t||^2 \ \ where \ \ \gamma = (L, \epsilon) \tag{41}$$

The γ that maximizing this objective measure will also minimize the first order auto-correlation. If the higher order auto-correlation increase monotonically with respect to first order auto-correlation, this measure is also efficient. In this situation, maximizing the objective measure is equivalent to minimize auto-correlation with any lag. However, this measure is not suitable for HMC samplers since it is almost guaranteed to have better samples by increasing leapfrog steps at each iteration. That means the EJSD measure will have no global maximum. Thus, computing time is necessary to be taken into consideration and following objective function will be used:

$$f(\gamma) = \frac{\mathbb{E}_\gamma ||\boldsymbol{x}^{t+1} - \boldsymbol{x}^t||^2}{\sqrt{L}} \tag{42}$$

Since we have an expectation in the normalized measure, it can be estimated by the empirical estimator:

$$\hat{f(\gamma)} = \frac{\frac{1}{m}\sum_{t=1}^{m-1} ||\boldsymbol{x}^{t+1} - \boldsymbol{x}^t||^2}{\sqrt{L}} \tag{43}$$

For calculation, We can run simple HMC for several iterations with some γ at each iteration of adaptive HMC. As a result, there will also be some random error and a noisy evaluation of the measure will be used here:

$$r(\gamma) = \hat{f(\gamma)} + \epsilon \tag{44}$$

7.3 Optimization

There are two common optimization methodology. One is stochastic approximation which is commonly used since the exact form of objective function is intractable. Another one is Bayesian optimization which we use here to optimize the objective function. Bayesian optimization is an efficient way since our objective function can be treated as a black box. Following Bayesian optimization methodology, we first set a box constraint Γ:

$$\Gamma = \{(\epsilon, L) : \epsilon \in [b_l^\epsilon, b_u^\epsilon], L \in [b_l^L, b_u^L]\} \ \ where \ \ b_l^\epsilon < b_u^\epsilon, b_l^L < b_u^L \tag{45}$$

Note that ϵ is a continuous parameter where L is a discrete parameter that is greater than zero. It is necessary to discretize it in a very fine grid. For the choice of interval boundaries, they can be set to be large enough to cover all reasonable γ. A better way is to run adaptive HMC several times and narrow the interval.

Since true objective is unknown, we need to find a surrogate model that can be maximized by γ. Here, We construct a Gaussian process to provide a posterior normal distribution over the objective function. First, we specify a prior distribution over the objective function:

$$f \sim N(0, k(\gamma_i, \gamma_j)) \tag{46}$$

$k(\gamma_i, \gamma_j)$ is some covariance function with argument γ_i and γ_j. Here, we use the Gaussian ARD covariance function so that $k(\gamma_i, \gamma_j) = exp(-\frac{1}{2}\gamma_i^T \Sigma^{-1} \gamma_j)$. At each iteration, we will compute the value of $r(\gamma)$ and store the value and γ into a dataset $D = \{\gamma, r(\gamma)\}$. (at n^{th} iteration, $D_n = \{\{\gamma\}_{i=1}^n, \{r\}_{i=1}^n\}$). Given D_k, we can arrive the posterior distribution over the objective function :

$$f|D_i, \gamma \sim N(\mu_i(\gamma), \sigma_i^2(\gamma)) \tag{47}$$

$$\mu_i(\gamma) = \boldsymbol{k}^T (\boldsymbol{K} + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{r}_i \tag{48}$$

$$\sigma_i^2(\gamma) = k(\gamma, \gamma) - \boldsymbol{k}^T (\boldsymbol{K} + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{k} \tag{49}$$

$$where \;\; \boldsymbol{k} = [k(\gamma, \gamma_1), ..., k(\gamma, \gamma_i)]^T, \;\; \boldsymbol{r}_i = [r_1, ..., r_i]^T \tag{50}$$

$$\boldsymbol{K} = \begin{pmatrix} k(\gamma_1, \gamma_1) & \dots & k(\gamma_1, \gamma_i) \\ \vdots & \ddots & \vdots \\ k(\gamma_i, \gamma_1) & \dots & k(\gamma_i, \gamma_i) \end{pmatrix} \tag{51}$$

Note that the dimension of K will increase as running length increases. This increase the difficulty of calculation and decrease efficiency. Thus, we need to carefully choose the number of iterations before the experiment.
Gaussian process enables us to build an acquisition function to tell us which γ maximize the objective function. The function uses the Gaussian process posterior mean to predict regions of potentially higher objective values and posterior variance to detect regions of high uncertainty (Wang 2013). We use a variant of Upper Confidence bound (UCB) as the acquisition function:

$$u(\gamma, s|D_i) = \mu_i(\gamma, s) + p_i \beta_{i+1}^{\frac{1}{2}} \sigma_i(\gamma) \;\; where \tag{52}$$

$$\mu_i(\gamma, s) = \mu_i(\gamma) \times s, \;\; p_i = (max\{i - k + 1, 1\})^{-0.5} \tag{53}$$

$$\beta_{i+1} = 2log(\frac{(i+1)^{\frac{d}{2}+2} \pi^2}{3\delta}) \;\; for \; k \in \mathbb{N}^+ \tag{54}$$

p_i will continuously decrease to zero after the number of current iterations are greater than k where k was set to be Burning period B divided by running length of standard HMC at each iteration (m). Thus, It becomes increasingly difficult for Bayesian optimization to propose new γ as p_i decreases. It is better to predict known good γ with a good initial value rather than exploring a better one.

7.4 Algorithm

Algorithm 9: Adaptive HMC

input : Γ : box constraint;
m : running length of standard HMC at each iteration (Burning period;
k, α
n : total number of iterations
γ_1 : initial value of γ
output: S : a set of samples obtained from HMC
$S \leftarrow \{\}$
for $i = 1, ..., m$ **do**
  Run standard HMC for m iterations with $\gamma_i = (\epsilon_i, L_i)$.
  Store the drawn samples into S
  Calculate the value of objective function r_i using the drawn samples.
  Store γ_i, r_i into dataset D_i
  if $r_i > sup_{j \in \{1,...,i-1\}}$ **then**
    $s = \frac{\alpha}{r_i}$
  end
  Draw $u \sim U(0, 1)$
  Calculate the value of $p_i = (max\{i - k + 1, 1\})^{-0.5}$
  if $u < p_i$ **then**
    $\gamma_{i+1} = argmax_{\gamma \in \Gamma} u(\gamma, s|D_i)$
  end
end
Run standard HMC for $n - m$ times and store drawn samples into S.

7.5 Experiment

Effective sample size (ESS) is a number that represents a sample draws from posterior distribution such that observations in this sample are correlated or weighted. In this experiment, we use $ESS = R \times (1 + 2\sum_k \rho_k)$ to evaluate the performance of different samplers. In above equation, R is the number of accepted states and ρ_k is the auto-correlations with lag k. If $\rho_k = 0 \ \ \forall k$, effective sample size is equal to the number of posterior samples. Since small running length with high effective sample size is the characteristics of an efficient algorithm, we use ESS per leapfrog steps (ESS/L) as the evaluation. We compute the maximum, minimum and median of ESS/L obtained and compare these summary statistics obtained from adaptive HMC and standard HMC.

7.5.1 Multivariate Multimodal data

Since the idea of our adaptive HMC algorithm is to find "good" hyperparameters for standard HMC, we use the same target distribution as in section 4.4.1. We generate 2000 samples after a burnin phase of 400 iterations and repeat this process 10 times with different initial state but same initial hyperparameters. The interval boundaries was set to be $[0.1, 3]$ and $[5, 25]$ for ϵ and L separately. Initial value of α was set to be 0.02 and m,k was both set to be 20. We then repeat the same progress 10 times using CTHMC with same setups and compare ESS/L obtained from two algorithm.

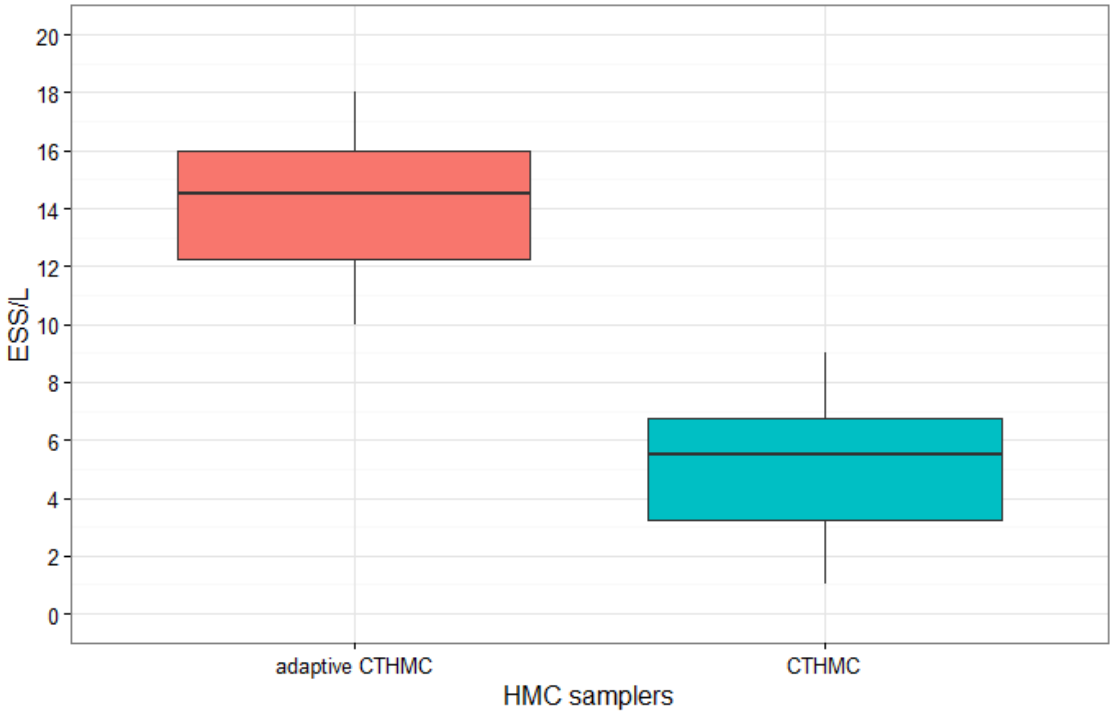

Figure 30: Comparing ESS/L using box plots

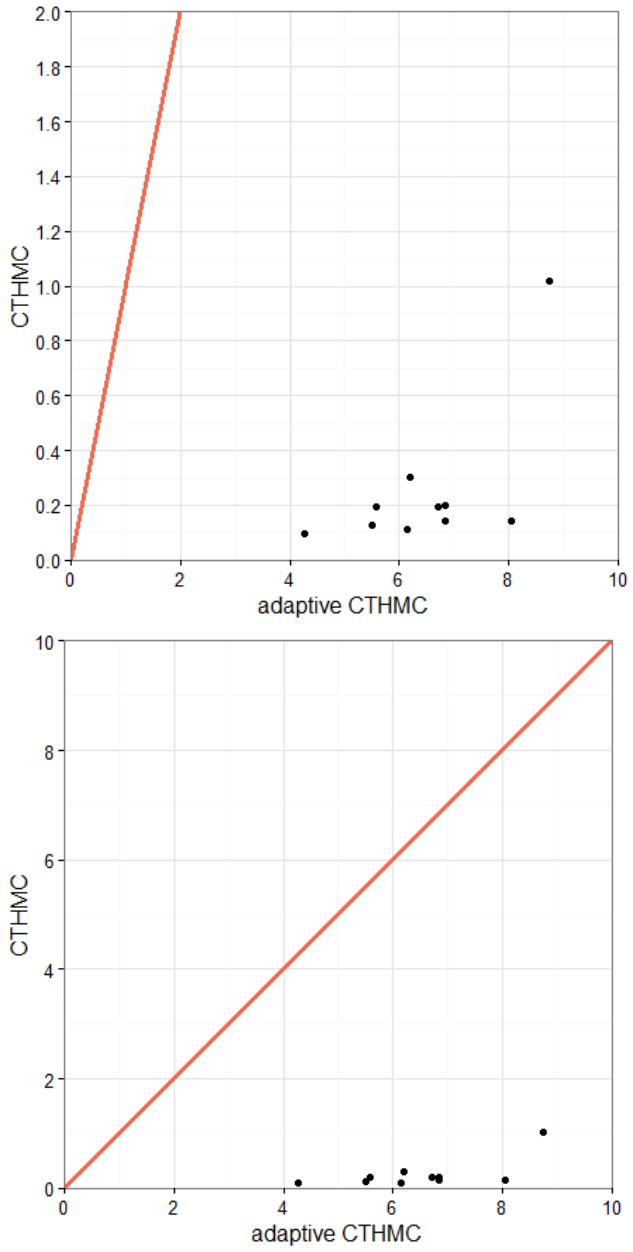

Figure 31: Comparing ESS/L obtained from CTHMC and adaptive CTHMC

Figure 30 reflects maximum, minimum and median of ESS/L obtained from CTHMC and adaptive CTHMC. We can see that adaptive CTHMC have a better performance than CTHMC. The box plots reflect that the minimum of ESS/L obtained from adaptive CTHMC is much higher than that of CTHMC and it is even higher than the maximum of ESS/L obtained from CTHMC. Figure 31 compares 10 ESS/L obtained for adaptive CTHMC and CTHMC and the bottom plot is equivalent to the top one with a different scale. It can be seen that all the points lie under the diagonal line and all very close to x-axis (adaptive CTHMC). Our adaptive CTHMC provides a much more efficient methodology to generate samples from multimodal problem. We can conclude that adaptive CTHMC has a higher effective sample size per unit of computation and does improves efficiency of CTHMC.

7.6 Challenges

Although adaptive CTHMC also has its own problems. The performance of this algorithm is highly depend on the computation ability of the programming language. For example, this experiment is writing in R programming language and R will recognize a parameter as INF if its value is too large (i.e. $a = INF$ if $a > 10^{31}$). There is a high probability that β_{i+1} has a large value after several iterations depends on the initial choice of δ. The algorithm will stop to find the "better" hyperparameters if a parameter with INF appears. In addition, the dimension of K matrix will increasingly large with more iterations. Thus, time-cost will be high if larger samples are generated. Also, the goal of this algorithm is to tunning standard HMC; so, ti is better to make sure that standard HMC is able to generate expected samples from the target distribution with some initial choice of hyperparameters.

References

[1] Radford M.Neal**.
MCMC Using Hamiltonian Dynamics--Chapter 5 of the Handbook of Markov Chain Monte Carlo, 2011

[2] Radford M.Neal**.
The Short-Cut Metropolis Method, 2005

[3] Matthew M. Graham and Amos J, Storkey**.
Continuously tempered Hamiltonian Monte Carlo, 2017

[4] Michael Betancourt**.
A General Metric for Riemannian Manifolds Hamiltonian Monte Carlo, 2013

[5] Leimkuhler. B. and Reich. S**.
Simulating Hamiltonian Dynamics--Cambridge University Press, 2004

[6] Michael Betancourt**.
Hamiltonian Monte Carlo for Hierarchical Model, 2013

[7] Michael Betancourt**.
A Conceptual Introduction to Hamiltonian Monte Carlo, 2017

[8] Ziyu Wang, Shakir Mohamed and Nando de Freitas**.
Adaptive Hamiltonian and Riemann Manifold Monte Carlo Samplers, 2013

[9] Akihiko Nishimura and David B. Dunson**
Geometrically Tempered Hamiltonian Monte Carlo, 2017

.