Hamiltonian Monte Carlo: Algorithm, Theory, and Experiments

Genghis Luo KL4747@NYU.EDU
NYU Shanghai
Alex Ni JN2294@NYU.EDU
NYU

Editor: Joan Bruna, Nikolaos Tsilivis

Abstract

Efficient sampling from complex, high-dimensional probability distributions remains a central challenge in computational statistics and machine learning. In this report, we present a comprehensive treatment of Hamiltonian Monte Carlo (HMC), an advanced Markov Chain Monte Carlo (MCMC) technique that leverages Hamiltonian dynamics to propose distant, low-autocorrelation moves in the target distribution. We begin by introducing the HMC algorithm—momentum augmentation, leapfrog integration, and the Metropolis acceptance step—and discuss its key theoretical properties, including reversibility, symplecticity, and near-conservation of the Hamiltonian. We then relate HMC to overdamped and underdamped Langevin dynamics and survey important extensions such as the No-U-Turn Sampler (NUTS) and Riemannian Manifold HMC. Through a series of experiments on Gaussian and "Donut" distributions across varying dimensions, as well as comparisons of integrators, kinetic energy functions, and proposal schemes, we empirically demonstrate HMC's superior effective sample size and acceptance rates relative to Random-Walk Metropolis, albeit at higher computational cost. Finally, we discuss practical considerations, limitations—such as tuning requirements and gradient dependence—and outline future research directions, including adaptive mass matrices, higher-order integrators, and integration with normalizing flows. Github codes for our experiments can be found here.

Keywords: Hamiltonian Monte Carlo; Markov Chain Monte Carlo; leapfrog integrator; symplecticity; high-dimensional sampling; effective sample size; No-U-Turn Sampler; Riemannian Manifold HMC

1. Introduction and Background

Efficiently sampling from complex, high-dimensional probability distributions is a fundamental challenge in computational statistics and machine learning. Traditional Markov Chain Monte Carlo (MCMC) methods like the Random-Walk Metropolis (RWM) algorithm often suffer from slow convergence and poor scalability in high dimensions. This is largely due to the diffusive "random walk" behavior of proposal moves, which leads to highly autocorrelated samples and long wait times for the chain to thoroughly explore the target distribution. In highly correlated or ill-conditioned settings, RWM must take very small steps to maintain a reasonable acceptance rate, further exacerbating the inefficiency. These limitations are sometimes referred to as a "curse of dimensionality" for MCMC, where the effective sample size per iteration drops precipitously as dimensionality grows.

Hamiltonian Monte Carlo (HMC) offers a powerful alternative by leveraging concepts from physics – specifically Hamiltonian dynamics – to guide the sampling process more effectively. Originally introduced as Hybrid Monte Carlo by Duane et al. (1987) in the physics literature, HMC augments the state space with auxiliary momentum variables and then simulates Hamiltonian dynamics to propose moves. By doing so, HMC can generate proposals that travel long distances through

the target distribution with high acceptance rates, mitigating the random-walk behavior and high autocorrelation that plague naive methods. In essence, HMC uses the gradient information of the target density (through the Hamiltonian equations of motion) to inform the direction of proposals, allowing the chain to "coast" through high-probability regions rather than diffusing slowly. This approach has proven remarkably successful in practice, enabling efficient exploration of complex posterior landscapes in Bayesian inference where simpler methods struggle.

HMC's performance advantages have made it a cornerstone of modern Bayesian computing. It was popularized in statistics by Neal (2012), who provided a detailed exposition and empirical evidence of its benefits. Since then, HMC has been adopted in probabilistic programming frameworks (such as Stan) and extended in various ways to improve its robustness and ease of use. For example, the No-U-Turn Sampler (NUTS) of Hoffman and Gelman (2011) eliminates the need to set a trajectory length by adaptively stopping when the trajectory starts to turn back on itself. Riemannian Manifold HMC (RMHMC) of Girolami and Calderhead (2011) adapts the HMC method to the local geometry of the target distribution by using position-specific mass matrices. In large-scale settings, Stochastic Gradient HMC (SGHMC) and related algorithms Ma et al. (2015) use noisy subsampled gradients with friction terms to scale HMC to massive datasets. These developments position HMC within a broader literature of gradient-based MCMC methods, alongside Langevin algorithms, as an indispensable tool for high-dimensional inference.

In this report, we provide a detailed account of the HMC algorithm and its theoretical foundations, and we summarize experimental results comparing HMC to RWM on prototypical problems. We begin with a description of the HMC algorithm, including the introduction of momentum, simulation of Hamilton's equations via a leapfrog integrator, and the Metropolis acceptance step. We then discuss the key theoretical guarantees that underpin HMC's validity and efficiency including reversibility, symplecticity, and approximate Hamiltonian conservation, and also light on its relations with (overdamped/underdamped) Langevin dynamics. Next, we present experimental results from our simulations (reproducing and extending experiments by Neal (2012) and others) that highlight HMC's performance on low-dimensional and high-dimensional Gaussian and Donut distributions, compared to RWM. Finally, we conclude with a discussion of HMC's practical advantages and limitations, its impact on modern statistical computing, and possible future directions for improvement and research.

2. Algorithm

2.1. Motivation

The Hamiltonian Monte Carlo algorithm augments the original variable of interest (often called the "position" variable) with an auxiliary momentum variable. If $x \in \mathbb{R}^d$ denotes the **position** (the variables we ultimately care to sample from the target distribution $\pi(x)$), HMC introduces a **momentum** $p \in \mathbb{R}^{\widehat{d}}$. Let $\widetilde{d} = d + \widehat{d}$. Typically, a **Hamiltonian** $H(x,p) : \mathbb{R}^{\widetilde{d}} \to \mathbb{R}$ is then defined as the sum of a **potential energy** $U(x) : \mathbb{R}^d \to \mathbb{R}$ and **kinetic energy** $K(p) : \mathbb{R}^{\widehat{d}} \to \mathbb{R}$:

$$H(x,p) = U(x) + K(p) \tag{1}$$

Canonically, $U(x)$ is deterministically chosen as minus the log target density, so that $U(x) = -\log \pi(x)$. The kinetic energy $K(p)$ is flexible in theory while usually taken to be quadratic practically as $K(p) = \frac{1}{2} p^T M^{-1} p$, where M is a mass matrix (often set to the identity or diagonal). This choice corresponds to p being assigned a Gaussian distribution $\mathcal{N}(0, M)$ as its marginal

(since p and x are independent). The introduction of p thus defines an lifted target distribution $\widetilde{\pi}(x,p) \propto \exp[-H(x,p)] = \exp[-U(x) - K(p)]$ on $\mathbb{R}^{\widetilde{d}}$, whose x-marginal is the original $\pi(x)$ by construction.

HMC proceeds by alternating between refreshing the momentum and updating (x,p) using Hamiltonian dynamics proposals. A single iteration of HMC can be summarized as follows:

- **Momentum resampling**: First, sample a new momentum p from its Gaussian distribution independent of the current state. For example, one can draw $p \sim \mathcal{N}(0, M)$, so each component p_i is drawn from $\mathcal{N}(0, m_i)$ if $M = \text{diag}(m_1, \ldots, m_d)$. This step ensures the momentum is randomized at each iteration, it is crucial in the sense that if momentum was not refreshed, the dynamics alone would conserve H almost exactly, trapping the state on a single constant-energy surface, so that the irreducibility of the Markov Chain is not satisfied. More details will be discussed in the next step and in Section 3.

- **Hamiltonian dynamics**: Starting from the current position x and the freshly sampled p, simulate Hamilton's equations of motion for a fixed time length s using a numerical integrator. Hamilton's equations are given by the coupled ordinary differential equations:

$$\frac{dx}{dt} = \nabla_p H(x,p) = \nabla_p K(p), \quad \frac{dp}{dt} = -\nabla_x H(x,p) = -\nabla_x U(x), \tag{2}$$

  In particular for our choice, we have $\nabla_p K(p) = M^{-1}p$, and $\nabla_x U(x) = -\nabla_x \log \pi(x)$. These equations describe how (x,p) would evolve in an artificial "physics" system with Hamiltonian H. To see this, we can generalize and rewrite the coupled ODE system (2) as:

$$\frac{d}{dt}\widetilde{y}^{(t)} = -\widetilde{J}\left(\widetilde{y}^{(t)}\right)\nabla^{\mathrm{T}} H\left(\widetilde{y}^{(t)}\right) + \text{div}\,\widetilde{J}\left(\widetilde{y}^{(t)}\right) \tag{3}$$

  for $\widetilde{y} = \widetilde{y}(x,p) : \mathbb{R}^{\widetilde{d}} \to \mathbb{R}^{\widetilde{d}}$ as the trajectory in the lifted $\mathbb{R}^{\widetilde{d}}$ space, and by the choice of $\widetilde{J} = \begin{bmatrix} 0 & -I_{d\times\widehat{d}} \\ I_{\widehat{d}\times d} & 0 \end{bmatrix}$, we see $M\frac{d^2}{dt^2}\widehat{y}^{(t)} = -\nabla^T U\left(\widehat{y}^{(t)}\right)$, which is in turn the **Newton's Second Law**. Indeed, the generator of the Hamiltonian ODE system (3) has good properties, it is **invariant** w.r.t. $\widetilde{\pi}_H \propto e^{-H}$ (further, **skew-reversible**), **time-symmetrical**, and **Hamiltonian-conserving** (thus, non-irreducibility), while skew-reversibility of the Hamiltonian generator can imply the reversibility of the chain under mild conditions, and non-irreducibility is bypassed by previous momentum resampling step. See more details in Section 3.

  As most realistic $U(x)$ are not analytically integrable, HMC uses a **symplectic integrator** (one that preserves volume and approximates energy well) to simulate the dynamics. The most common choice is the **leapfrog integrator** (we have also tested other symplectic integrators like modified Euler scheme and non-symplectic integrators like classical Euler schemes, it is natural to expect symplectic over non-symplectic integrators, see details in Section 4), which discretizes time length into small steps of size ϵ and alternates updates of x and p:

  - $p \leftarrow p - \frac{\epsilon}{2}\nabla_x U(x)$ (half-step update of momentum),
  - $x \leftarrow q + \epsilon M^{-1}p$ (full-step update of position using the new momentum),

- $p \leftarrow p - \frac{\epsilon}{2}\nabla_x U(x)$ (another half-step for momentum using updated position).

These three sub-steps constitute one leapfrog step of size ϵ, after which (x, p) has been advanced by a discrete time ϵ. To simulate for a total of s units of time, one performs $L = [s/\epsilon]$ leapfrog steps in sequence. The result is a proposal state (x^*, p^*) obtained by following the approximate Hamiltonian trajectory from the starting point. Crucially, the leapfrog integrator (or generally symplectic integrator) is **time-reversible** and **volume-preserving**. It also accumulates only small error in the Hamiltonian, which tends to remain bounded even for many steps if ϵ is well-chosen. This means the simulated trajectory closely follows a constant-energy contour of the true Hamiltonian, reaching a point that has nearly the same H value as the start. See more details in Section 3 and 4.

- **Metropolis acceptance step**: After the L leapfrog steps, we have a candidate new state (x^*, p^*). To correct for the discretization error (since the leapfrog integration is not exact), HMC uses a Metropolis–Hastings acceptance criterion. We accept the proposed state with probability

$$\alpha = \min\{1, \exp[-H(x^*, p^*) + H(x, p)]\} = \min\{1, \exp(-\Delta H)\},$$

where $\Delta H = H(x^*, p^*) - H(x, p)$ is the change in the Hamiltonian along the numerically simulated trajectory. If energy were exactly conserved ($\Delta H = 0$) as in the continuous limit, the acceptance probability would be always 1. If the move is rejected, the state remains at (x, p). If the proposal is accepted, we take the new position x^* as the next sample and set $p^* \leftarrow -p^*$ as a negation step. This step is crucial in the theoretical sense to keep the proposal distribution symmetric in HMC (thus does not appear in the acceptance ratio), while it can be simply neglected because the kinetic energy we use is often even and we will resample the momentum next round.

In summary, HMC generates a Markov chain in the lifted $(x, p) \in \mathbb{R}^{\widetilde{d}}$ space where proposals are informed by Hamiltonian dynamics. Each iteration uses gradient information to propose a distant move that is then accepted or rejected with a probability that corrects for any simulation error. The use of momentum and dynamics helps the chain "inertially" navigate the energy landscape, avoiding the diffusive behavior of random-walk proposals.

2.2. Pseudocode

Pseudocode for a basic HMC update is given in Algorithm 1.

2.3. Relevant Extensions and Variants

Many improvements to the basic HMC algorithm have been developed. As mentioned, the No-U-Turn Sampler (NUTS) automatically tunes the number of leapfrog steps L by doubling the trajectory length s until a U-turn is detected, thereby eliminating the need for the user to set a path length a priori. Riemannian Manifold HMC (RMHMC) generalizes HMC to use a position-dependent mass matrix $M(x)$, effectively integrating geodesics on a manifold informed by the local curvature of the target density Girolami and Calderhead (2011). This can dramatically improve sampling for distributions with heterogeneous curvature (e.g. Neal's funnel distribution) but at a higher computational

Algorithm 1 Hamiltonian Monte Carlo with Metropolis

Require: initial position $X^{(1)}$, step size ϵ, mass matrix M, # of leapfrog steps L, sample size N

for $t = 1, 2, \ldots, N$ **do**
  Sample momentum: $P^{(t)} \sim \mathcal{N}(0,\, M)$
  Set $(X_0,\, P_0) \leftarrow (X^{(t)},\, P^{(t)})$
  $P_0 \leftarrow P_0 \;-\; \frac{\epsilon}{2}\nabla U(X_0)$
  for $i = 1$ **to** s **do**
    $X_i \;\leftarrow\; X_{i-1} \;+\; \epsilon\, M^{-1} P_{i-1}$
    $P_i \;\leftarrow\; P_{i-1} \;-\; \epsilon\, \nabla U(X_i)$
  end for
  $P_L \leftarrow P_L \;-\; \frac{\epsilon}{2}\nabla U(X_L)$
  Set proposal $(X^*,\, P^*) \leftarrow (X_L,\, -P_L)$ {A negation step that can be often neglected}
  Draw $u \sim \mathrm{Uniform}(0, 1)$
  Compute $\rho \;=\; \exp\big(H(X^*, P^*) \;-\; H(X^{(t)}, P^{(t)})\big)$
  if $u < \min(1, \rho)$ **then**
    $X^{(t+1)} \leftarrow X^*$
  else
    $X^{(t+1)} \leftarrow X^{(t)}$
  end if
end for
return $\{X^{(t)}\}_{t=1}^{N}$

cost (one must compute matrix derivatives or factorings). Partial momentum refreshment schemes Horowitz (1991) have been proposed where only a fraction of the momentum is renewed each iteration, to improve mixing while retaining some persistent direction between iterations. For big-data problems, where evaluating $\nabla_x U(x)$ on the full dataset is expensive, stochastic gradient HMC methods using minibatch estimates of the gradient was introduced by Ma et al. (2015). Finally, recent research has explored combining HMC with normalizing flows (e.g. Neural Hamiltonian Flow in Toth et al. (2019)) to transform the sample space into one easier for HMC to explore. By using deep learning to find a better representation or by iteratively morphing the distribution, these approaches aim to further reduce autocorrelation and improve stability, pushing the limits of HMC in challenging problem domains.

3. Theoretical Guarantees

The effectiveness of Hamiltonian Monte Carlo rests on solid theoretical foundations that ensure the sampler maintains the correct target distribution and explores it efficiently.

3.1. Guarantees for Hamiltonian dynamics

The key properties inherited from Hamiltonian dynamics are: (1) **invariant** w.r.t. $\widetilde{\pi}_H \;\propto\; e^{-H}$ (further, **skew-reversible**), (2) **time-symmetricity** of the trajectory, (3) **Hamiltonian-conserving** (thus, non-irreducibility), and (4) **symplecticity** when $d = \widehat{d}$. We discuss each in turn:

We first lay a solid foundation for preparation. Let $\widetilde{x} = (x, p) \in \mathbb{R}^{\widetilde{d}}$, we assume $\liminf_{\widetilde{x}\to\infty} \frac{H(\widetilde{x})}{\|\widetilde{x}\|} > 0$ to make sure that $\exp(-H(\widetilde{x}))$ is integrable, and thus the Boltzmann distribution $\widetilde{\pi}_H(\widetilde{x}) \propto \exp(-H(\widetilde{x}))$ is well-defined. Specifically, in terms of the Boltzmann density, we can rewrite the ODE (3) as:

$$\frac{d}{dt}y^{(t)} = \frac{1}{\widetilde{\pi}_H\left(y^{(t)}\right)} \operatorname{div}\left(\widetilde{\pi}_H\left(y^{(t)}\right)\widetilde{J}\left(y^{(t)}\right)\right)$$

The generator corresponding to the ODE (3) is:

$$\mathcal{L}_H f = \nabla f \frac{1}{\widetilde{\pi}_H} \operatorname{div}\left(\widetilde{\pi}_H \widetilde{J}\right) \tag{4}$$

The action of $\mathcal{L}_H$ on a density μ is found (by an integration by parts) to be:

$$\begin{aligned}\mu\mathcal{L}_H &= -\operatorname{div}\left(\frac{\mu}{\widetilde{\pi}_H}\operatorname{div}^{\mathrm{T}}\left(\widetilde{\pi}_H\widetilde{J}\right)\right)\\ &= -\nabla\left(\frac{\mu}{\widetilde{\pi}_H}\right)\operatorname{div}\left(\widetilde{\pi}_H\widetilde{J}\right) - \frac{\mu}{\widetilde{\pi}_H}\operatorname{div}\left(\operatorname{div}^{\mathrm{T}}\left(\widetilde{\pi}_H\widetilde{J}\right)\right) = -\nabla\left(\frac{\mu}{\widetilde{\pi}_H}\right)\operatorname{div}\left(\widetilde{\pi}_H\widetilde{J}\right)\end{aligned} \tag{5}$$

where the second term of the second equation vanishes due to $\widetilde{J}$'s anti-symmetricity.

- Invariance w.r.t. $\widetilde{\pi}_H \propto e^{-H}$ and further, skew-reversibility:

  Plugging in $\mu = \widetilde{\pi}_H$ in (5) we find that:

  $$\widetilde{\pi}_H\mathcal{L}_H = 0 \tag{6}$$

  In fact, with respect to $\widetilde{\pi}_H$, the operator $\mathcal{L}_H$ satisfies the even stronger property

  $$\int g(\widetilde{x})\mathcal{L}_H f(\widetilde{x})\widetilde{\pi}_H(d\widetilde{x}) = -\int f(\widetilde{x})\mathcal{L}_H g(\widetilde{x})\widetilde{\pi}_H(d\widetilde{x}) \tag{7}$$

  for any test function f and g, which can be referred to as skew-reversibility w.r.t. $\widetilde{\pi}_H$. Thus, in turn, it implies that:

  $$\int f(\widetilde{x})g\left(y^{(t)}(\widetilde{x})\right)\widetilde{\pi}_H(d\widetilde{x}) = \int g(\widetilde{x})f\left(y^{(-t)}(\widetilde{x})\right)\widetilde{\pi}_H(d\widetilde{x}) \tag{8}$$

  which can be verified by fixing $s \in [0, t]$ and let $w(s) = \int f\left(y^{(s-t)}(\widetilde{x})\right) g\left(y^{(s)}(\widetilde{x})\right)\widetilde{\pi}_H(d\widetilde{x})$ and then showing that the derivative of w is zero.

- Time-symmetricity of the trajectory $\widetilde{y}^{(t)}$ on the lifted space $\mathbb{R}^{\widetilde{d}}$:

  If we make two more canonical assumptions, say: (1) $H(\widetilde{x})$ is an even function of the momentum variable p, i.e.,

  $$H(x, p) = H(x, -p),$$

  and (2) $\widetilde{J}$ has a particular form:

  $$\widetilde{J}(\widetilde{x}) = \begin{bmatrix} 0 & -J(x) \\ J^{\mathrm{T}}(x) & 0 \end{bmatrix} \tag{9}$$

where J is a $d \times \widehat{d}$ matrix valued function of only the x variables. Then, under these assumptions, we can write ODE (3) as:

$$\frac{d}{dt}\begin{pmatrix} y^{(t)} \\ \widehat{y}^{(t)} \end{pmatrix} = \begin{pmatrix} J\left(y^{(t)}\right)\nabla_x^{\mathrm{T}} H\left(y^{(t)}\right) \\ -J^{\mathrm{T}}\left(y^{(t)}\right)\nabla_x^{\mathrm{T}} H\left(y^{(t)}\right) + \operatorname{div} J^{\mathrm{T}}\left(y^{(t)}\right) \end{pmatrix}.$$

The action of the operator $\mathcal{L}_H$ on functions f becomes

$$\mathcal{L}_H f = \nabla_x f J \nabla_p^{\mathrm{T}} H - \nabla_p f J^{\mathrm{T}} \nabla_x^{\mathrm{T}} H + \nabla_p f \operatorname{div} J^{\mathrm{T}}, \tag{10}$$

and its action on probability densities u becomes

$$\mu\mathcal{L}_H = -\nabla_p \mu J \nabla_p^{\mathrm{T}} H + \nabla_p \mu J^{\mathrm{T}} \nabla_x^{\mathrm{T}} H - (\mu \nabla_p H + \nabla_p \mu) \operatorname{div} J^{\mathrm{T}}$$

Notice that if for some test function f we set $f_-(\widetilde{x}) = f(x, -p)$, then we have:

$$\mathcal{L}_H f_-(x, -p) = -\mathcal{L}_H f(\widetilde{x}). \tag{11}$$

A similar formula holds if we apply $\mathcal{L}_H$ to $\mu_-(x) = \mu(x, -p)$.

Indeed, (11) has several remarkable and useful ramifications. For one, it implies that if f is an even (odd) function of p then $\mathcal{L}_H f$ is an odd (even) function of p. In particular, if f and g are both even functions of p, then:

$$\int g(x)\mathcal{L}_H f(x)dx = 0$$

For another, (11) implies that both the functions

$$\widetilde{y}^{(-t)}(\widetilde{x}) \quad \text{and} \quad \left(y^{(t)}(x, -p), -\widehat{y}^{(t)}(x, -p)\right)$$

solves ODE (3) in general with the sign of the right hand side reversed and with initial condition $\widetilde{x}$. By the uniqueness of solutions to the ODE, we find therefore that the two functions are equal, i.e.,

$$\widetilde{y}^{(-t)}(\widetilde{x}) = \left(y^{(t)}(x, -p), -\widehat{y}^{(t)}(x, -p)\right) \tag{12}$$

Equation (12) is often referred to as the time-symmetricity of the trajectory $\widetilde{y}^{(t)}$ on the lifted space $\mathbb{R}^{\widetilde{d}}$. It tells us that the inverse of the flow map at time t can also be written as a forward-in-time integration using an initial condition with reversed sign in p. Time reversal symmetry of the trajectory is not to be confused with the notion of reversibility of the chain, but indeed time reversal symmetry can be used to show that a Markov process incorporating Hamilton's ODE are indeed **reversible**. See in Section 3.3.

Last but not least, time reversal symmetry combined with expression (8) implies that if f and g are even functions of $\tilde{x}$ then

$$\int g(\tilde{x}) f\left(\widetilde{y}^{(t)}(\tilde{x})\right) \widetilde{\pi}_H(d\tilde{x}) = \int f(\tilde{x}) g\left(\widetilde{y}^{(t)}(\tilde{x})\right) \widetilde{\pi}_H(d\tilde{x}) \tag{13}$$

- Hamiltonian-conserving (thus, non-irreducible)

  When $\widetilde{J}$ is a constant matrix (e.g., a canonical choice of $\widetilde{J} = \begin{bmatrix} 0 & -I_{d\times\widehat{d}} \\ I_{\widehat{d}\times d} & 0 \end{bmatrix}$), by (10), we have:

$$\mathcal{L}_H H = \nabla H \cdot \frac{\widetilde{J}\nabla\widetilde{\pi}_H + \left(\nabla\cdot\widetilde{J}\right)\widetilde{\pi}_H}{\widetilde{\pi}_H} = \nabla H \cdot \frac{\widetilde{J}\nabla\widetilde{\pi}_H}{\widetilde{\pi}_H} = (\nabla H)^T \widetilde{J}(\nabla H) = 0 \tag{14}$$

  where the last equation is because for $\forall v \in \mathbb{R}^{\widetilde{d}}$, as $v^T\widetilde{J}v \in \mathbb{R}$, we have $v^T\widetilde{J}v = \left(v^T\widetilde{J}v\right)^T = -v^T\widetilde{J}v$ due to skew-symmetricity of $\widetilde{J}$. Therefore, the Hamiltonian system (3) is Hamiltonian conservative, i.e., with fixed initial point, every trajectory $\widetilde{y}^{(t)}$ preserves the Hamiltonian.

  Moreover, when J is constant, $\mu\mathcal{L}_H = 0$ for an density μ of the form $\mu(x) = \rho(H(x))$ for some function ρ. In words, the flow map $y^{(t)}(x)$ preserves any density of this form (including the constant density). However, it is also clear that if the value of H is preserved, the solutions to (3) cannot be ergodic (or, cannot be irreducible). Indeed, this irreducibility issue is bypassed by the momentum resampling step in the MCMC algorithm.

- Symplecticity when $d = \widehat{d}$:

  Last but not least, under very specific situation when $d = \widehat{d}$, the Hamiltonian dynamics (3) is **volume-preserving** in the sense that for the vector field of (3): $V : \mathbb{R}^{2d} \to \mathbb{R}^{2d}, (x,p) \mapsto \binom{\nabla_p H(x,p)}{-\nabla_x H(x,p)}$, we have:

$$\mathrm{div}V = \sum_{i=1}^{d}\left[\frac{\partial}{\partial x_i}\frac{dx_i}{dt} + \frac{\partial}{\partial p_i}\frac{dp_i}{dt}\right] = \sum_{i=1}^{d}\left[\frac{\partial}{\partial x_i}\frac{\partial H}{\partial p_i} - \frac{\partial}{\partial p_i}\frac{\partial H}{\partial x_i}\right] = 0. \tag{15}$$

  Volume preservation is also a consequence of Hamiltonian dynamics being **symplectic**, in the sense that the Jacobian matrix, B_s, of the transition operator $\mathcal{T}_s$ from $(x^{(t)}, p^{(t)})$ to $(x^{(t+s)}, p^{(t+s)})$ satisfies

$$B_s^T J^{-1} B_s = J^{-1}.$$

  This implies volume conservation, since $\det\left(B_s^T\right)\det\left(J^{-1}\right)\det\left(B_s\right) = \det\left(J^{-1}\right)$ implies that $\det\left(B_s\right)^2$ is one. When $d > 1$, the symplecticity condition is stronger than volume preservation. Although this symplecticity property is only constrained in $d = \widehat{d}$, it motivates us to determine the integrators we use in practice for running the discretized Hamiltonian ODE 3. We illustrate three integrators in Section 3.2, and compare their performances in Section 4.

3.2. Guarantees for leapfrog integrator and its BAD alternatives

Recall that there are three integrators we are interested at (for simplicity, let $M = \mathrm{diag}(m_1, \ldots, m_d)$):

- Euler's method

$$p_i(t+\varepsilon) = p_i(t) + \varepsilon\frac{dp_i}{dt}(t) = p_i(t) - \varepsilon\frac{\partial U}{\partial x_i}(x(t)),$$
$$x_i(t+\varepsilon) = x_i(t) + \varepsilon\frac{dx_i}{dt}(t) = x_i(t) + \varepsilon\frac{p_i(t)}{m_i}.$$

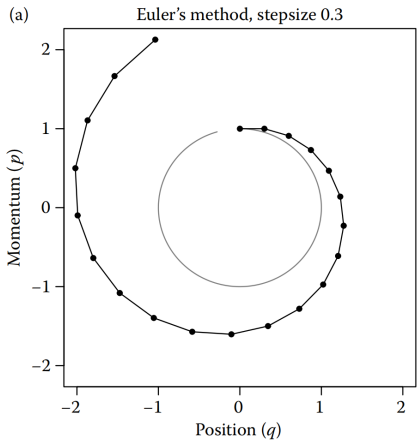


Figure 1: (q is x) Euler's method initialized at $(x, p) = (0, 1)$, with $H(x, p) = x^2/2 + p^2/2$ and step size $\varepsilon = 0.3$ as in Neal (2012)

- Modified Euler's method

$$p_i(t + \varepsilon) = p_i(t) - \varepsilon \frac{\partial U}{\partial x_i}(x(t)),$$
$$x_i(t + \varepsilon) = x_i(t) + \varepsilon \frac{p_i(t + \varepsilon)}{m_i}.$$

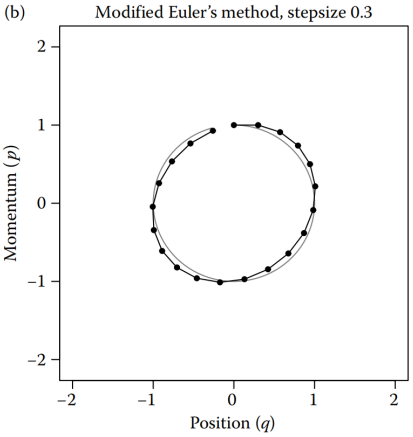


Figure 2: (q is x) Modified Euler's method initialized at $(x, p) = (0, 1)$, with $H(x, p) = x^2/2+p^2/2$ and step size $\varepsilon = 0.3$ as in Neal (2012)

- Velocity-Verlet / Leapfrog method

$$p_i(t+\varepsilon/2) = p_i(t) - (\varepsilon/2)\frac{\partial U}{\partial x_i}(x(t)),$$
$$x_i(t+\varepsilon) = x_i(t) + \varepsilon\frac{p_i(t+\varepsilon/2)}{m_i},$$
$$p_i(t+\varepsilon) = p_i(t+\varepsilon/2) - (\varepsilon/2)\frac{\partial U}{\partial x_i}(x(t+\varepsilon)).$$

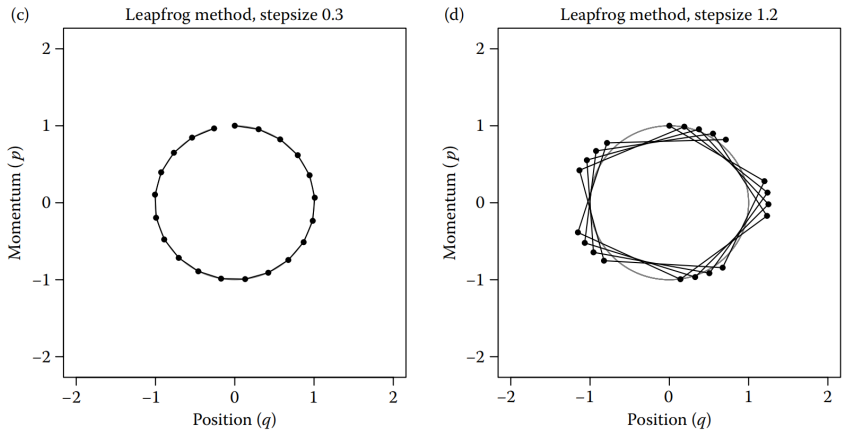


Figure 3: (q is x) Leapfrog method initialized at $(x, p) = (0, 1)$, with $H(x, p) = x^2/2 + p^2/2$ and step size $\varepsilon = 0.3$ for (c), step size $\varepsilon = 1.2$ for (d) as in Neal (2012)

We briefly discuss how the error from discretizing the dynamics behaves in the limit as the step size, ε, goes to zero; Leimkuhler and Reich (2005) provide a much more detailed discussion. For useful methods, the error goes to zero as ε goes to zero, so that any upper limit on the error will apply (apart from a usually unknown constant factor) to any differentiable function of state-for example, if the error for (x, p) is no more than order ε^2, the error for $H(x, p)$ will also be no more than order ε^2.

The local error is the error after one step, that moves from time t to time $t + \varepsilon$. The global error is the error after simulating for some fixed time interval, s, which will require s/ε steps. If the local error is order ε^p, the global error will be order ε^{p-1} —the local errors of order ε^p accumulate over the s/ε steps to give an error of order ε^{p-1}. If we instead fix ε and consider increasing the time, s, for which the trajectory is simulated, the error can in general increase exponentially with s. Interestingly, however, this is often not what happens when simulating Hamiltonian dynamics with a symplectic(volume-preserving) method, as can be seen in Figure 2 and Figure 3. See in Neal (2012) for a detailed discussion.

The Euler method and its modification above have order ε^2 local error and order ε global error. The leapfrog method has order ε^3 local error and order ε^2 global error. As shown by Leimkuhler and Reich (2005) Section 4.3.3, this difference is a consequence of leapfrog being reversible, since any reversible method must have global error that is of even order in ε.

3.3. Further guarantees for Hamiltonian Monte Carlo

In Algorithm 1, we in fact have a reversible chain if we choose the kinetic energy $K(p)$ to be an even function in p, and our $\widetilde{J}$ as (9). To see this note that the transition operator for the Markov chain $X^{(t)}$ generated by Algorithm 1 is given by:

$$\mathcal{T}_s f(x) = \frac{\int f\left(y^{(s)}(x,p)\right) e^{-K(p)} dp}{\mathcal{Z}_p}$$

Appealing to time reversal symmetry (12) and the fact that K is an even function we can apply (13) to see that:

$$\begin{aligned}\int g(x)\mathcal{T}_s f(x)\pi(dx) &= \int g(x) f\left(y^{(s)}(x)\right)\widetilde{\pi}_H(dx) \\ &= \int f(x) g\left(y^{(s)}(x)\right)\widetilde{\pi}_H(dx) = \int f(y)\mathcal{T}_s g(y)\pi(dy),\end{aligned}$$

i.e. that the $X^{(k)}$ process is reversible with respect to π.

3.4. Relations with Langevin dynamics

Indeed, Hamiltonian Monte Carlo methods have close relations with Langevin dynamics. If we choose the integration step to be 1 for each generation round, assume that we also make a typical choice of kinetic energy as: $K(p) = \|p\|_2^2/2$, then we fully recovered the overdamped Langevin dynamics:

$$\begin{aligned}X_h^{(k+1)} = X_h^{(k)} + hS\left(X_h^{(k)}\right)\nabla^{\mathrm{T}}\log\pi\left(X_h^{(k)}\right) \\ + h\,\mathrm{div}\,S\left(X_h^{(k)}\right) + \sqrt{2hS\left(X_h^{(k)}\right)}\xi^{(k+1)}\end{aligned} \tag{16}$$

where for any $h > 0$, X_h is a Markov chain step in $\mathbb{R}$ generated according to the Metropolis-Hastings rule with proposal density $q(y \mid x) = \mathcal{N}(x, 2h)$.

Therefore, we expect that if we choose n very large in Algorithm 1, we will expend substantial effort to generate a single update of the chain $X^{(k)}$ and the scheme will become inefficient. On the other hand, if we choose n to be small, the performance of this scheme is similar to the corresponding overdamped Langevin scheme. It is then often the case that for intermediate choices of n, the HMC scheme outperforms its overdamped Langevin analogue (even accounting for the additional cost of the multiple evaluations of $\nabla \log \pi$). Ma et al. (2015) discusses using one-pass SGD to overcome this issue.

An alternative approach to deriving possibly ergodic schemes based on the Hamiltonian ODE is to add appropriate random terms at each integration step. This corresponds to the idea of underdamped Langevin dynamics:

$$\begin{aligned}X_h^{(k+1)} = X_h^{(k)} - h(J + S\left(X_h^{(k)}\right)\nabla^{\mathrm{T}}H\left(X_h^{(k)}\right) \\ + h\,\mathrm{div}(J+S)\left(X_h^{(k)}\right) + \sqrt{2hS\left(X_h^{(k)}\right)}\xi^{(k)}\end{aligned} \tag{17}$$

for independent $\xi^{(k)}$ with $\mathbf{E}\left[\xi^{(k)}\right] = 0$ and $\mathrm{cov}\left[\xi^{(k)}\right] = I$ (and finite higher moments). See more details discussed in Septier and Peters (2015).

4. Experimental Results

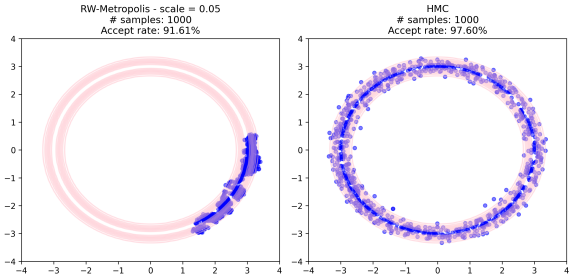
In this section, we present some experiment results to show some of the most important attributes of HMC and modify the components of HMC for structural comparisons.

4.1. Experiment 1: Motivations

In this experiment, we want to show that the Hamiltonian Monte Carlo (HMC) sampler generally performs better than the Markov Chain Monte Carlo (MCMC) sampler on the 2D Donut distribution, which is specially tailored to the HMC sampler.

Parameter	Value / Description
Dimension (d)	Configurable, default: 2
Number of Samples	1000
Donut Radius (r)	3.0
Shell Thickness (σ^2)	0.05
Initial State	$[r, 0, \ldots, 0]$
Proposal Type (Metropolis)	Gaussian (NormalProposal)
Metropolis Scales	0.05 (small), 1.0 (large)
HMC Step Size (ϵ)	0.1
HMC Leapfrog Steps (L)	50
Output File	`donut_comparison_{dim}d.png`
Visualization	2D scatter plots / pairwise projections
Target Distribution	Donut: $\exp\left(-\frac{(\|x\|-r)^2}{\sigma^2}\right)$

HMC vs MCMC: Settings



HMC vs RWM: Results

Figure 4: Comparison between HMC and MCMC

As shown in the Figure 4, the HMC sampler covers the probability space more efficiently by exploring the majority of the high-density region, while the MCMC sampler covers only a very small proportion. Even if the acceptance-rate is comparable, the caveat is that when computing the target distribution's moments (mean, variance, etc.), using samples from the MCMC sampler could have a larger bias. For high-dimensional probability space, the problem will be more obvious.

Notice that in this experiment we fix the proposal distribution for MCMC to be standard Gaussian, the standard Gaussian kinetic energy function $K(p) = \frac{1}{2}p^\top M^{-1}p$ with Leapfrog integrator for the HMC sampler configuration. This is because these settings are oftenly used in practice and prove to be stable and excel in performance. However, we are interested in whether changing these settings would affect the overall performance for HMC and MCMC, which leads to the following experiments (Experiment 2 to Experiment 5).

4.2. Experiment 2: MCMC Proposal Comparison

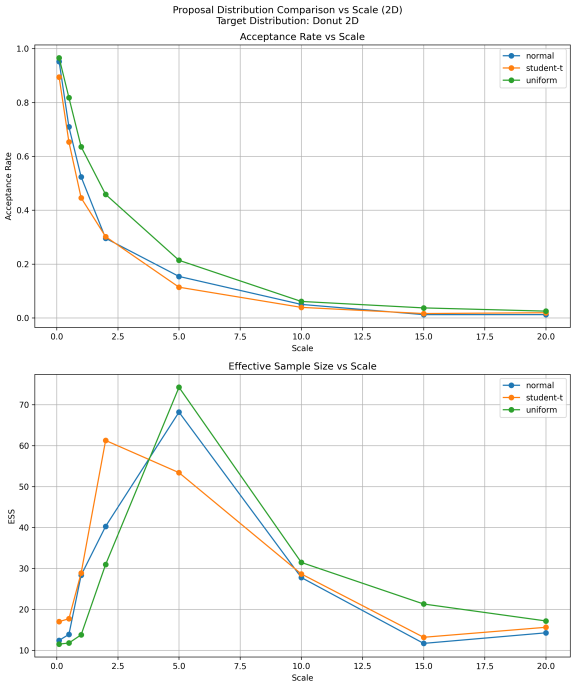
In experiment 1, we already show that MCMC performs worse than HMC on 2D Donut distribution with standard gaussian proposal. So we aim to show whether other proposal choices (student-t with df = 3, uniform) will give a performance bonus on MCMC.

The experiment setting and results of the experiment are shown in Figrue 5, and there are three things worth noting:

1. The proposal is defined as $x' = x + \epsilon$ where ϵ belongs to our proposal distribution. For different candidates in our experiment, we have:

Parameter	Value
Target Distribution	Donut Distribution
Target Dimension	2D
Donut Radius	3.0
Donut Thickness Variance (σ^2)	0.5
Number of Samples	1,000
Initial State	$(3.0, 0.0)$
Proposal Types	Gaussian, Student-t (df=3), Uniform
Proposal Scales	0.1, 0.5, 1.0, 2.0, 5.0, 10, 15, 20
Sampler	MCMC with different proposals
Acceptance Rate Estimation	Accepted samples / Total samples(1000)
Evaluation Metrics	Acceptance Rate, Effective Sample Size (ESS)

HMC vs MCMC: Settings



HMC vs RWM: Results

Figure 5: Comparison between HMC and MCMC

(a) For gaussian proposal, it is defined as $\epsilon \sim \mathcal{N}\left(0, s^2 I\right)$ and each ϵ_i is sampled independently according to the marginal gaussian distribution.

(b) For student-t proposal, it is defined as $x^* = x + \epsilon$, where $\epsilon_i \sim s \cdot t_\nu$, $f(\epsilon_i) = \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\nu\pi}\Gamma\left(\frac{\nu}{2}\right)s}\left(1 + \frac{1}{\nu}\left(\frac{\epsilon_i}{s}\right)^2\right)^{-\frac{\nu+1}{2}}$ and each ϵ_i is sampled independently.

(c) For uniform proposal, it is defined as $x^* = x + \epsilon, \quad \epsilon_i \sim \mathcal{U}(-s, s)$ where $f(\epsilon_i) = \begin{cases} \frac{1}{2s}, & \text{if } \epsilon_i \in [-s, s] \\ 0, & \text{otherwise} \end{cases}$ and each ϵ_i is sampled independently.

2. We choose to vary the **scale**(denoted as s), which can be interpreted as the exploration radius of proposal distribution; the larger the scale, the farther the distance will be between the current sample and the next hop proposal. Mathematically, we have:

3. As the scale increases, the acceptance rate of the MCMC sampler drops. Larger scale increase the likelihood that the proposal sample falls outside of the high-density region and is rejected. Thus the acceptance rate function w.r.t. scale is monotonically decreasing.

4. ESS is computed by $\text{ESS} = \frac{n}{1+2\sum_{k=1}^{K}\rho_k}$ where $K = \min\left(\left\lfloor \frac{n}{3} \right\rfloor, \max\{k : \rho_k \geq 0.05\}\right)$. For smaller scales, the adjacent samples are highly correlated, leading to a bigger denominator and thus a smaller ESS. For bigger scales, lots of samples are rejected, leading to a smaller numerator and thus a smaller ESS. Only with proper s value will we reach a decent level of ESS.

In conclusion, we have to nitpick both the scale and proposal distribution in order to have a comparable performance with HMC on 2D Donut, which shows the power of HMC.

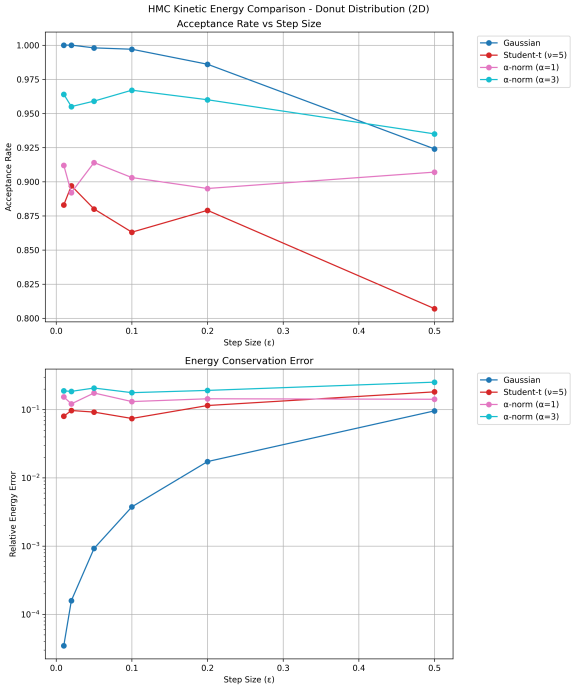
4.3. Experiment 3: HMC Kinetic Energy Function vs Step Size Comparison

Having shown that HMC generally performs better than MCMC, we want to now modify the component of HMC. In this experiment, we modify the kinetic energy function definition. From the Hamiltonian's definition in (1), the $U(q)$ term is generally fixed for Hamiltonian dynamics, but we can modify $K(p)$ the definition, where our candidates include:

1. Standard Gaussian kinetic energy function: $K(\mathbf{p}) = \frac{1}{2}\mathbf{p}^\top \mathbf{M}^{-1}\mathbf{p}$
2. Student-t Kinetic energy function: $K(\mathbf{p}) = \frac{\nu}{2}\log\left(1 + \frac{1}{\nu}\mathbf{p}^\top \mathbf{M}^{-1}\mathbf{p}\right)$ suggested by Livingstone and Girolami (2014)
3. Alpha-Norm Kinetic energy function: $K_\alpha(\mathbf{p}) = \frac{1}{\alpha}\sum_{i=1}^{d}|z_i|^\alpha = \frac{1}{\alpha}\left\|\mathbf{M}^{-1/2}\mathbf{p}\right\|_\alpha^\alpha$ suggested by Betancourt (2018)

Parameter	Value
Target Distribution	Donut Distribution
Target Dimension	2D
Number of Samples	1000
Donut Thickness Variance (σ^2)	0.5
Initial State	$(3.0, 0.0)$
Trajectory Length ($s = \epsilon \times L$)	0.5
Step Sizes (ϵ)	0.01, 0.02, 0.05, 0.1, 0.2, 0.5
Kinetic Energy Types	Gaussian, Student-t ($\nu = 5$), ℓ_1-norm ($\alpha = 1$), ℓ_3-norm ($\alpha = 3$)
Integrator	Leapfrog (fixed)
Sampler	HMC with varying kinetic energy
Acceptance Rate Estimation	Accepted samples / Total samples
Evaluation Metrics	Acceptance Rate, Relative Energy Error

HMC vs MCMC: Settings



HMC vs RWM: Results

Figure 6: Comparison between HMC kinetic energy function vs step size

As Figure 6 shows,

1. Gaussian kinetic energy leads to simple linear momentum updates and pairs naturally with the assumption of independent normal momentum variables. This ensures the numerical stability of leapfrog integrator is maximized, especially when using an identity mass matrix.

In contrast, student-t kinetic energy introduces heavy tails, which can destabilize momentum updates. Moreover, α-norms (like L1 or L3) are non-quadratic, leading to less smooth Hamiltonian surfaces, making gradients less stable and energy error more sensitive to step size.

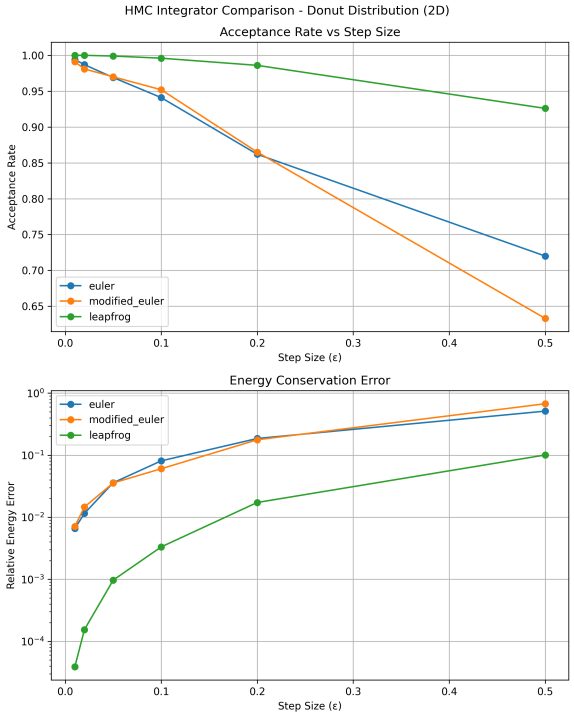
2. With small ε, all integrators approximate the true continuous trajectory well. As ε increases, discretization introduces more error, especially for non-Gaussian forms which are less amenable to numerical integration. The leapfrog integrator remains stable up to a critical ϵ, beyond which energy conservation breaks down.

4.4. Experiment 4: HMC Integrator vs Step Size Comparison

Next we modify the integrator for HMC during the discretization of hamiltonian surfaces. Our candidates include the Euler's method, modified Euler's method and leapfrog integrator, as introduced here()

Parameter	Value
Target Distribution	Donut Distribution
Target Dimension	2D
Donut Radius	3.0
Donut Thickness Variance (σ^2)	0.5
Number of Samples	1000
Initial State	(3.0, 0.0)
Trajectory Length ($s = \epsilon \times L$)	0.5
Step Sizes (ϵ)	0.01, 0.02, 0.05, 0.1, 0.2, 0.5
Integrators	Euler, Modified Euler, Leapfrog
Sampler	HMC with different integrators
Acceptance Rate Estimation	Accepted samples / Total samples
Evaluation Metrics	Acceptance Rate, Relative Energy Error

HMC vs MCMC: Settings



HMC vs RWM: Results

Figure 7: Comparison between HMC integrators vs step size

We observer the following from Figure 7:

1. Leapfrog (green) maintains high acceptance rates even as it ε increases, while Euler's (blue) and Modified Euler's (orange) show rapidly declining acceptance as they ε increase. One of the reasons is that Leapfrog is symplectic and time-reversible - it conserves energy better and keeps the proposal within the "correct" region of phase space, while Euler's methods are

not symplectic, and they accumulate integration error much faster, causing large deviations in energy and $\rightarrow$ low acceptance. Even if modified Euler's method is also symplectic, it is not strictly following the Hamiltonian surface, causing bigger error.

2. On a log scale, Leapfrog has orders of magnitude lower error, especially at large step sizes. Euler's/modified Euler's have much higher and steeper energy error growth. The reason is that The Hamiltonian is supposed to be conserved along the simulated trajectory. The leapfrog integrator introduces second-order errors that largely cancel out, while Euler's introduces first-order drift. This means:

$$\left|H\left(x^*, p'\right) - H(x, p)\right| \ll \text{ for Leapfrog vs Euler's ,}$$

where H is the Hamiltonian.

The final takeawawy is that the leapfrog integrator is the backbone of HMC because it is specifically designed to simulate Hamiltonian systems accurately. It preserves the geometry of the system, which is crucial for MCMC methods that rely on detailed balance and reversibility.

4.5. Experiment 5: MCMC vs HMC Across Dimensions Comparison

After the experiments focused on 2D Donut distribution, we want to extend our experiment settings to high dimensional distribution, which is shown in Table 1.

For the evaluation metric, we choose **acceptance rate, ESS**, and **Number of floating-point operations (FLOPs)**. Table 2 shows the definition of our FLOPS:

The experiment results is shown in Figure 8, and we observe:

1. For Donut distribution:
   (a) **Acceptance Rate**: HMC achieves higher acceptance rates than MCMC at low dimensions due to its ability to make informed proposals using gradients. However, as dimensionality increases, the probability mass becomes increasingly concentrated on a thin shell, making it difficult for any sampler—including HMC—to propose moves that stay on this shell. Consequently, the acceptance rate for all samplers drops steeply, highlighting the challenge of exploring curved manifolds in high dimensions.
   (b) **ESS**: HMC consistently achieves higher ESS than MCMC, especially at low to moderate dimensions. This is due to its ability to produce less correlated samples by simulating Hamiltonian dynamics. However, as dimension increases, the high curvature of the shell degrades performance, causing ESS for all samplers—including HMC—to drop sharply. MCMC methods struggle even more because their random proposals often jump outside the thin shell, leading to low acceptance and highly autocorrelated chains.
2. For standard gaussian distribution:
   (a) **Acceptance Rate**: HMC maintains a high acceptance rate across all dimensions, significantly outperforming MCMC methods. This is because the Gaussian's smooth, convex landscape is well-suited for HMC's gradient-based dynamics, allowing it to propose long-range moves that remain within high-density regions. In contrast, MCMC proposals deteriorate quickly with increasing dimensions, as random walks are less likely to land in regions of high probability, leading to a marked decline in acceptance rate.

Table 1: Experimental Configuration Overview

Category	Configuration
General Settings	
Number of Points to Sample	1000
Warmup Samples	100
Dimensions Tested	[2, 5, 10, 20, 50, 100, 200, 300]
Target Distributions	
Standard Gaussian	
Type	Multivariate Normal
Mean	Zero vector
Covariance	Identity matrix
Donut Distribution	
Type	N-dimensional shell
Radius	3.0
Donut Thickness Variance (σ^2)	0.5
MCMC Settings — Proposal Distributions	
Gaussian	Covariance = Identity matrix
Student-t	Degrees of freedom = 3
Uniform	Range = [–1, 1]
HMC Settings	
Leapfrog Integrator	
Trajectory Length ($s = \epsilon \times L$)	0.5
Step Size (ϵ)	0.1
Kinetic Energy Distributions	
Gaussian	Standard normal momentum
Student-t	Degrees of freedom (ν) = 3.0

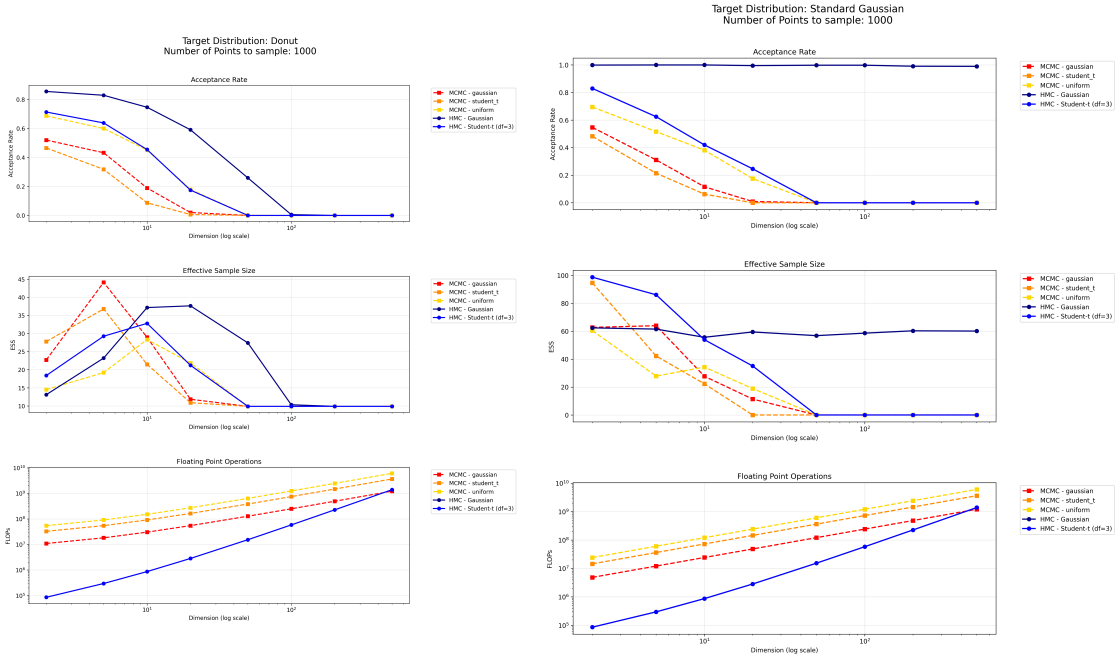
(b) **ESS**: HMC delivers dramatically higher ESS across all dimensions, maintaining strong performance even in high dimensions. **This is due to the fact that we choose a gaussian kinetic energy function, which captures the shape of the target distribution very well.** In contrast, MCMC's ESS rapidly declines due to poor scaling of random walk proposals, which increasingly fail to explore the space efficiently as dimensionality grows.

3. **FLOPS**: Across both distributions, HMC incurs significantly higher computational cost (FLOPs) than MCMC due to its reliance on gradient evaluations and multiple leapfrog steps per sample. MCMC remains relatively cheap per sample, but its low ESS means more samples are needed for the same statistical accuracy. Notably, HMC's computational cost increases linearly (or worse) with dimension, but its efficiency per effective sample remains competitive—especially in the Gaussian case, where its FLOPs/ESS ratio is superior to that of MCMC.

Table 2: FLOP Accounting Definitions for HMC and MCMC Samplers

Operation	**FLOPs Count**	**Remarks**
Vector Addition ($\mathbf{x} + \mathbf{y}$)	d	One addition per element
Scalar-Vector Multiplication ($a \cdot \mathbf{x}$)	d	One multiplication per element
Matrix-Vector Multiplication ($\mathbf{A} \cdot \mathbf{x}$)	d^2	Assumes dense matrix
Gradient Evaluation ($\nabla \log p(\theta)$)	$2d$	Approximated per call
Dot Product ($\mathbf{x}^\top \mathbf{y}$)	d	Inner product
Logarithm or Exponential ($\log x$, $\exp x$)	1	Approximated as unit cost
Division (e.g. $\frac{f(\theta')}{f(\theta)}$)	1	Used in acceptance ratio
Acceptance Decision (Comparison)	0	Logic-only, no arithmetic FLOPs

4. **Curse of dimensionality**: Due to its special property, there is no cure-for-all sampler (proposal distribution choices, integrator choices, kinetic energy function choices) in high dimension for Donut distribution.



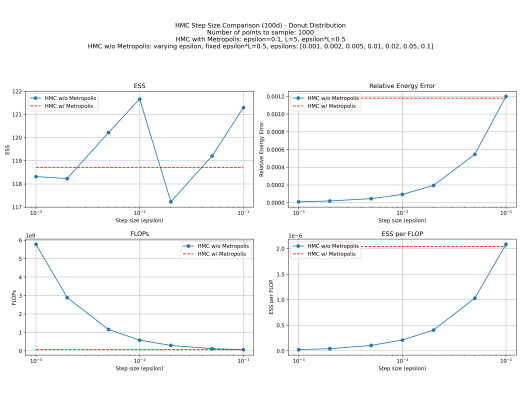
Donut Distribution

Standard Gaussian Distribution

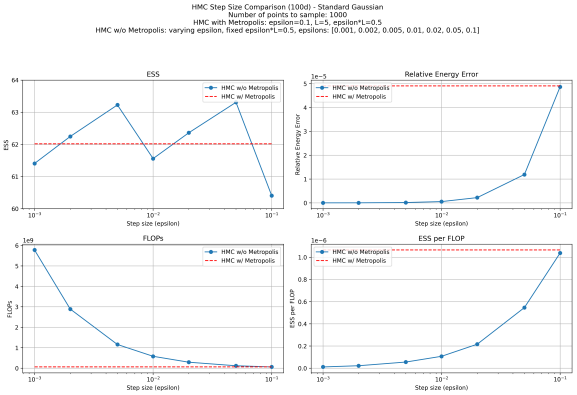
Figure 8: MCMC vs HMC Across Dimensions Comparison

4.6. Experiment 6: High Dimension HMC Variants Comparison

In this experiment, we aim to verify the hypothesis that the Metropolis-Hastings correction step may not be necessary in high-dimension settings and decreasing step size may be more helpful. Results are shown in Figure 9, Figure 10, Figure 11.

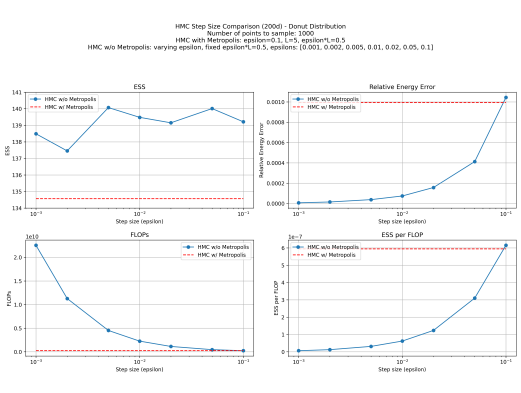


Donut Distribution

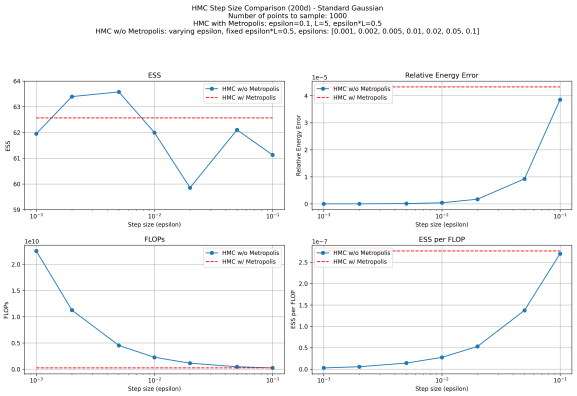


Standard Gaussian Distribution

Figure 9: HMC Variants Comparison - 100d



Donut Distribution



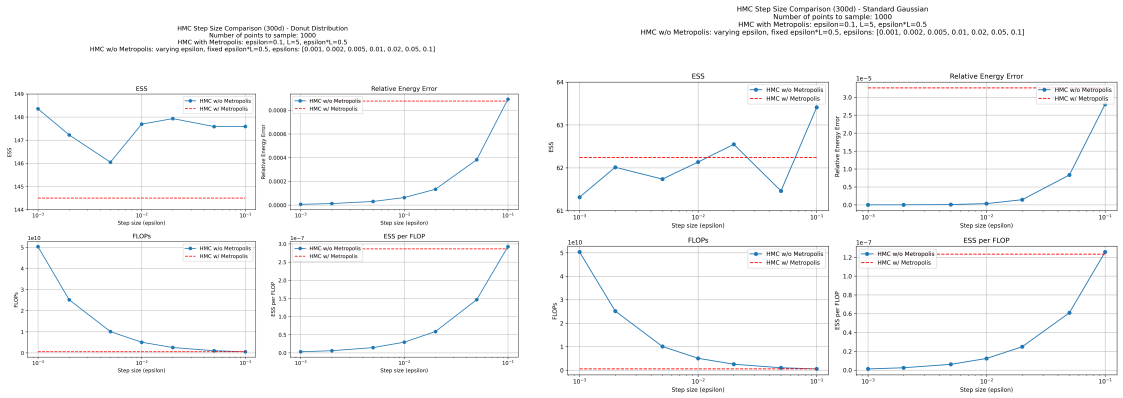
Standard Gaussian Distribution

Figure 10: HMC Variants Comparison - 200d

In conclusion, the metropolis correction step is not necessary in high dimensions when the step size is small enough. We can reach the same level of or even better ESS performance without metropolis correction just by using a small step size(ϵ). But we cannot ignore the fact that with Metropolis correction we can significantly decrease the FLOPs and reach a similar level of ESS as without Metropolis correction.

5. Discussion and Conclusion

This is a great project with a very interesting topic, we enjoy a lot working on HMC for this semester. We learned the motivation of HMC algorithm, we delved deep into the theoretical foundations of it, and spent huge chunks of time into testing this algorithm and experimenting on new ideas. Below are a few words we would like to discuss about the HMC algorithm based on our experimental observations and theoretical understandings.



Donut Distribution

Standard Gaussian Distribution

Figure 11: HMC Variants Comparison - 300d

5.1. Practical advantages

One of HMC's major practical advantages is its ability to handle high-dimensional targets with complex geometry (e.g., Donut distribution). Where Gibbs or Metropolis samplers might get stuck exploring one dimension at a time or rejecting proposals that don't fit a narrow ridge of probability, HMC can simultaneously adjust all dimensions in a coordinated way, guided by the gradient towards high-density regions. This makes it particularly powerful for highly correlated posteriors (common in hierarchical Bayesian models) and targets with multimodal or curved densities (where local Euclidean proposals struggle). HMC generates proposals that are often near-independently distributed (especially with partial momentum refreshment, successive HMC states can be almost uncorrelated), which means fewer iterations are needed to achieve a desired estimation accuracy. Moreover, HMC's use of continuous trajectories means it naturally avoids diffusive behavior and can tunnel through energy barriers more effectively (though it is not a panacea for multimodality, it often explores modes connected by reasonably low-energy paths well). In practice, we also find that HMC provides useful diagnostics: the acceptance rate and energy error can indicate when the sampler is behaving well or when step sizes are mis-tuned, and divergences (instances of integrator instability) can highlight problematic regions of the posterior Betancourt (2018). Moreover, we have tested on the idea of "*It is often better to increase the amount of discretization steps rather than using Metropolis acceptance in high dimensional problems*", which is **not very correct** for testing on high-dimensional Donut and Gaussian distributions. In fact, Metropolis acceptance step compensates for the discretization error in a very efficient manner, which is somehow crucial in the increasingly higher dimensional problems.

5.2. Limitations

Despite its strengths, HMC is not without limitations. Firstly, HMC requires gradient computations of the log-density $U(q)$ at each step of the trajectory. In problems where $\nabla U(q)$ is expensive or not available in closed form, HMC can be computationally heavy. This is particularly an issue in "big data" Bayesian problems, where $U(q)$ sums contributions from millions of data points; straightforward HMC would require a full pass through the data for each leapfrog step, which is

infeasible. Stochastic gradient variants (SGHMC) Ma et al. (2015) have been developed to address this by using minibatches and adding noise, at the cost of introducing some bias or requiring careful calibration of friction. Secondly, HMC has several tunable parameters – notably the step size ϵ and number of leapfrog steps L (or equivalently the trajectory length $T = L\epsilon$), and the mass matrix M – which must be chosen appropriately. Poor choices can lead to either inefficient sampling (if ϵ is too small or L too short, producing small moves) or rejection of many proposals or even numerical divergence (if ϵ is too large or L too long for the integrator to remain stable). In high dimensions, setting these parameters by hand can be challenging; indeed, this was a barrier to wider adoption of HMC until the advent of automatic tuning strategies. Algorithms like NUTS (which adapts L on the fly) Hoffman and Gelman (2011) and adaptive step-size schemes (which adjust ϵ during a warm-up phase to target a desired acceptance rate) have largely mitigated this issue, making HMC more user-friendly. Another limitation is that HMC operates in continuous state spaces – it cannot be directly applied to discrete parameter models, since gradients are required. Techniques exist to extend HMC-like ideas to discrete spaces (e.g., Hamiltonian jumps on lattices in physics, or continuous relaxations for discrete variables), but these are specialized. Furthermore, while HMC mixes faster than RWM in most scenarios, it is still a local sampler that can struggle with truly multimodal distributions that have very isolated modes separated by high energy barriers. In such cases, any local method (HMC included) may fail to jump between modes unless combined with other techniques like tempering.

5.3. Future directions

Looking ahead, several avenues exist to further enhance HMC or address its shortcomings. One direction is adaptive and self-tuning HMC: while NUTS and step-size adaptation are great strides, further adaptation of the mass matrix (especially in non-diagonal forms) during sampling can improve efficiency in anisotropic distributions. Riemannian HMC already moves in this direction, but practical and robust implementations are still an active area of research (due to the difficulty of computing Hessians or metric tensors in complex models). Another direction is symmetric splitting integrators or higher-order integrators for HMC: the leapfrog integrator is simple and effective, but in some cases higher-order integrators or tailored integrators that exactly conserve some invariants could allow larger step sizes or reduce bias. Recent work in computational physics on symplectic integrators could translate to better HMC performance. Additionally, temperature-assisted sampling techniques could be combined with HMC – e.g., tempering or annealed trajectories that allow crossing between modes, marrying HMC's local efficiency with global exploration strategies. There is also interest in distributed and parallel HMC: while HMC is inherently sequential (each step depends on the previous), one can run multiple chains in parallel or use speculative moves in parallel and accept one, etc., to utilize modern multi-core hardware. Combining HMC with Sequential Monte Carlo (SMC) or particle filtering (as explored by Septier and Peters (2015), for high-dimensional filtering problems) is another promising route, where HMC proposals can move particles efficiently in an SMC framework. Finally, continuing the theme of combining learning and sampling, we expect to see more of normalizing flows or learned transport guiding HMC. By preconditioning the space with an invertible neural network that approximately "gaussianizes" the posterior, one could alleviate the burden on HMC to handle complex geometries, thereby further reducing autocorrelation and making each HMC trajectory more effective. Early research in this vein is underway.

In conclusion, Hamiltonian Monte Carlo stands as a landmark development in MCMC methodology, offering a principled and highly effective way to sample from difficult distributions. Its grounding in physics provides both an intuitive picture (samples as particles gliding through the landscape) and strong mathematical guarantees (symplectic integrators, invariants) that together yield a formidable algorithm. Through both theoretical analysis and extensive experimentation, we see that HMC can greatly improve sampling efficiency, especially in the high-dimensional regimes that are common in modern Bayesian inference. HMC has transformed what practitioners expect from sampling algorithms: problems once deemed intractable are now routinely handled. As we refine the method further and integrate it with other innovations, HMC and its descendants will continue to play a central role in the future of computational statistics, enabling us to tackle ever more complex models and datasets with confidence in the correctness and efficiency of our Monte Carlo estimates. As Betancourt (2018) emphasizes, understanding why HMC works so well – and also when it can fail – is key to using it effectively. This report has sought to illuminate those points, demonstrating both the elegant theory of Hamiltonian Monte Carlo and its powerful impact in practice.

References

Michael Betancourt. A conceptual introduction to hamiltonian monte carlo, 2018. URL https://arxiv.org/abs/1701.02434.

Simon Duane, A.D. Kennedy, Brian J. Pendleton, and Duncan Roweth. Hybrid monte carlo. *Physics Letters B*, 195(2):216–222, 1987. ISSN 0370-2693. doi: https://doi.org/10.1016/0370-2693(87)91197-X. URL https://www.sciencedirect.com/science/article/pii/037026938791197X.

Mark Girolami and Ben Calderhead. Riemann manifold langevin and hamiltonian monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(2):123–214, 2011. doi: https://doi.org/10.1111/j.1467-9868.2010.00765.x. URL https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-9868.2010.00765.x.

Matthew Hoffman and Andrew Gelman. The no-u-turn sampler: Adaptively setting path lengths in hamiltonian monte carlo. *Journal of Machine Learning Research*, 15, 11 2011.

Alan M. Horowitz. A generalized guided monte carlo algorithm. *Physics Letters B*, 268(2):247–252, 1991. ISSN 0370-2693. doi: https://doi.org/10.1016/0370-2693(91)90812-5. URL https://www.sciencedirect.com/science/article/pii/0370269391908125.

Benedict Leimkuhler and Sebastian Reich. *Simulating Hamiltonian Dynamics*. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 2005.

Samuel Livingstone and Mark Girolami. Information-geometric markov chain monte carlo methods using diffusions. *Entropy*, 16(6):3074–3102, 2014. ISSN 1099-4300. doi: 10.3390/e16063074. URL https://www.mdpi.com/1099-4300/16/6/3074.

Yi-An Ma, Tianqi Chen, and Emily B. Fox. A complete recipe for stochastic gradient mcmc. In *Proceedings of the 29th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, page 2917–2925, Cambridge, MA, USA, 2015. MIT Press.

Radford Neal. Mcmc using hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 06 2012. doi: 10.1201/b10905-6.

François Septier and Gareth Peters. Langevin and hamiltonian based sequential mcmc for efficient bayesian filtering in high-dimensional spaces. *IEEE Journal of Selected Topics in Signal Processing*, 10, 04 2015. doi: 10.1109/JSTSP.2015.2497211.

Peter Toth, Danilo Jimenez Rezende, Andrew Jaegle, Sébastien Racanière, Aleksandar Botev, and Irina Higgins. Hamiltonian generative networks. *ArXiv*, abs/1909.13789, 2019. URL https://api.semanticscholar.org/CorpusID:203593936.