Indian Statistical Institute, Delhi Centre

BSDS

Hamiltonian Monte Carlo: From Physics Intuition to Advanced Applications

Author:
Ansh Sarraf
Roll Number: BSD-DH-2406
bsddh2406@isid.ac.in

Co-Author:
Antareep
Roll Number: BSD-CC-2406
bsdcc2406@isid.ac.in

May 2026

Abstract

Markov Chain Monte Carlo (MCMC) methods are the backbone of Bayesian computation, yet classical algorithms such as the Metropolis–Hastings sampler suffer from slow exploration in high-dimensional and highly correlated target distributions. This report provides a self-contained, mathematically rigorous treatment of the Hamiltonian Monte Carlo (HMC) algorithm, which resolves these limitations by embedding the sampling problem within the framework of classical Hamiltonian mechanics. We begin by reviewing the deficiencies of the standard Metropolis algorithm and motivating the HMC construction. We then derive the leapfrog integrator in full detail, prove its two essential geometric properties—*time-reversibility* and *volume preservation* (symplecticity)—and use these properties to rigorously establish that HMC satisfies the detailed balance condition, guaranteeing convergence to the correct target distribution. In the second part, we implement and visualise the HMC trajectory in R for a bivariate Gaussian target, introduce the No-U-Turn Sampler (NUTS) as an adaptive extension, and demonstrate a concrete Bayesian application: posterior inference for the parameters of the Vasicek stochastic differential equation using Stan via `rstan`. The report aims to serve as a complete reference for a mathematically literate reader wishing to understand both the theory and practice of HMC.

Contents

1 Introduction and Motivation

The fundamental challenge of Bayesian statistics is the evaluation of integrals of the form

$$\mathbb{E}_\pi[f(x)] = \int_{\mathcal{X}} f(x)\,\pi(x)\,\mathrm{d}x, \tag{1}$$

where $\pi(x) \propto e^{-S(x)}$ is a target density (here written in the physicist's convention with *action* $S(x) = -\log \tilde{\pi}(x)$ for unnormalised density $\tilde{\pi}$) and the integration domain $\mathcal{X} \subseteq \mathbb{R}^d$ is typically of high dimension. Except in specially conjugate models, these integrals are analytically intractable, necessitating computational approximation.

Monte Carlo methods replace the integral by a sample average: given i.i.d. draws $x^{(1)}, \ldots, x^{(T)} \sim \pi$, the law of large numbers gives

$$\frac{1}{T}\sum_{t=1}^{T} f(x^{(t)}) \xrightarrow{\text{a.s.}} \mathbb{E}_\pi[f(x)]. \tag{2}$$

When direct sampling is impossible, **Markov Chain Monte Carlo (MCMC)** constructs a Markov chain $\{x^{(t)}\}$ whose stationary distribution is π, so that the ergodic average converges to the same limit.

1.1 The Metropolis–Hastings Algorithm and Its Limitations

The Metropolis–Hastings (MH) algorithm [3, 4] is the prototypical MCMC method. Given a symmetric proposal kernel $q(x' \mid x)$ (typically a Gaussian random walk), each iteration:

1. proposes $x' \sim q(\cdot \mid x^{(t)})$,
2. accepts with probability $\alpha = \min\left(1, \dfrac{\pi(x')}{\pi(x^{(t)})}\right) = \min\left(1, e^{S(x^{(t)}) - S(x')}\right)$.

While correct in principle, the random-walk proposal produces two pathological behaviours in high dimensions:

- **High rejection rates.** A proposal drawn uniformly at random in $\mathbb{R}^d$ is overwhelmingly likely to move into a low-density region, causing rejection.
- **Diffusive exploration.** Accepted proposals are locally correlated; the chain requires $O(d^2)$ steps to traverse the support of π, generating high autocorrelation and making the effective sample size far smaller than the nominal sample size.

The HMC algorithm, first introduced by Duane et al. [1] under the name *Hybrid Monte Carlo* and later popularised in statistics by Neal [2], resolves both issues by exploiting the gradient of $\log \pi$ to generate long-range, low-rejection proposals.

2 Hamiltonian Mechanics: The Physical Analogy

2.1 The Hamiltonian System

The key idea of HMC is to embed the sampling problem in a higher-dimensional *phase space* by augmenting the position variables $\boldsymbol{x} = (x_1, \ldots, x_d) \in \mathbb{R}^d$ (the original parameters of interest) with auxiliary *momentum* variables $\boldsymbol{p} = (p_1, \ldots, p_d) \in \mathbb{R}^d$. The dynamics on this phase space are governed by the **Hamiltonian**:

$$H(\boldsymbol{x}, \boldsymbol{p}) = S(\boldsymbol{x}) + K(\boldsymbol{p}), \qquad K(\boldsymbol{p}) = \frac{1}{2}\sum_{i=1}^{d} p_i^2 = \frac{1}{2}\|\boldsymbol{p}\|^2, \tag{3}$$

where $S(\boldsymbol{x}) = -\log \tilde{\pi}(\boldsymbol{x})$ plays the role of **potential energy** and $K(\boldsymbol{p}) = \frac{1}{2}\|\boldsymbol{p}\|^2$ is the **kinetic energy** (corresponding to a unit mass matrix; a general mass matrix M gives $K(\boldsymbol{p}) = \frac{1}{2}\boldsymbol{p}^\top M^{-1}\boldsymbol{p}$).

The joint distribution on phase space corresponding to H is the **canonical distribution**:

$$\pi(\boldsymbol{x}, \boldsymbol{p}) \propto e^{-H(\boldsymbol{x},\boldsymbol{p})} = e^{-S(\boldsymbol{x})} \cdot e^{-\frac{1}{2}\|\boldsymbol{p}\|^2}. \tag{4}$$

Because H is separable, $\boldsymbol{x}$ and $\boldsymbol{p}$ are *independent* under $\pi(\boldsymbol{x}, \boldsymbol{p})$: the marginal in $\boldsymbol{x}$ is the target $\pi(\boldsymbol{x}) \propto e^{-S(\boldsymbol{x})}$, and the marginal in $\boldsymbol{p}$ is a standard multivariate Gaussian $\mathcal{N}(\boldsymbol{0}, I_d)$.

2.2 Hamilton's Equations

The evolution of the system through fictitious time τ is determined by **Hamilton's equations**:

$$\frac{\mathrm{d}x_i}{\mathrm{d}\tau} = \frac{\partial H}{\partial p_i} = p_i, \qquad \frac{\mathrm{d}p_i}{\mathrm{d}\tau} = -\frac{\partial H}{\partial x_i} = -\frac{\partial S}{\partial x_i}. \tag{5}$$

These equations define a flow $\Phi_\tau : (\boldsymbol{x}(0), \boldsymbol{p}(0)) \mapsto (\boldsymbol{x}(\tau), \boldsymbol{p}(\tau))$ with three fundamental properties that are essential to the correctness of HMC:

Proposition 2.1 (Properties of Hamiltonian Flow)**.** *The flow Φ_τ generated by Hamilton's equations satisfies:*

(i) ***Energy conservation:*** *$H(\boldsymbol{x}(\tau), \boldsymbol{p}(\tau)) = H(\boldsymbol{x}(0), \boldsymbol{p}(0))$ for all τ.*

(ii) ***Time reversibility:*** *If $(\boldsymbol{x}^*, \boldsymbol{p}^*) = \Phi_\tau(\boldsymbol{x}, \boldsymbol{p})$, then $(\boldsymbol{x}, -\boldsymbol{p}) = \Phi_\tau(\boldsymbol{x}^*, -\boldsymbol{p}^*)$.*

(iii) ***Volume preservation (Liouville's theorem):*** *$|\det J_{\Phi_\tau}| = 1$, where J_{Φ_τ} is the Jacobian of the map Φ_τ.*

Proof. (i) Energy conservation. By the chain rule and Hamilton's equations:

$$\frac{\mathrm{d}H}{\mathrm{d}\tau} = \sum_{i=1}^{d}\left(\frac{\partial H}{\partial x_i}\frac{\mathrm{d}x_i}{\mathrm{d}\tau} + \frac{\partial H}{\partial p_i}\frac{\mathrm{d}p_i}{\mathrm{d}\tau}\right) = \sum_{i=1}^{d}\left(\frac{\partial H}{\partial x_i}p_i - \frac{\partial H}{\partial p_i}\frac{\partial S}{\partial x_i}\right).$$

Since $\partial H/\partial x_i = \partial S/\partial x_i$ and $\partial H/\partial p_i = p_i$, each summand is $p_i \cdot \partial S/\partial x_i - p_i \cdot \partial S/\partial x_i = 0$.

(ii) Time reversibility. The momentum negation map $\mathcal{N} : (\boldsymbol{x}, \boldsymbol{p}) \mapsto (\boldsymbol{x}, -\boldsymbol{p})$ is an involution ($\mathcal{N}^2 = \mathrm{id}$) and reverses the direction of Hamiltonian flow, since negating $\boldsymbol{p}$ negates $\mathrm{d}\boldsymbol{x}/\,\mathrm{d}\tau = \boldsymbol{p}$ while leaving $\mathrm{d}\boldsymbol{p}/\,\mathrm{d}\tau = -\nabla_{\boldsymbol{x}} S$ unchanged in magnitude. Hence $\mathcal{N} \circ \Phi_\tau \circ \mathcal{N} = \Phi_{-\tau} = \Phi_\tau^{-1}$, which is precisely the statement of reversibility.

(iii) Volume preservation. Liouville's theorem from classical mechanics states that Hamiltonian flows are volume-preserving. This follows from the divergence theorem: the divergence of the Hamiltonian vector field $\mathcal{V} = (\nabla_{\boldsymbol{p}} H, -\nabla_{\boldsymbol{x}} H) = (\boldsymbol{p}, -\nabla_{\boldsymbol{x}} S)$ is

$$\nabla_{(\boldsymbol{x},\boldsymbol{p})} \cdot \mathcal{V} = \sum_i \frac{\partial p_i}{\partial x_i} + \sum_i \frac{\partial(-\partial S/\partial x_i)}{\partial p_i} = 0 + 0 = 0.$$

A vector field with zero divergence generates a flow that preserves volume. □

2.3 Why This Helps Sampling

Because the Hamiltonian is conserved along the flow, proposing the state $(\boldsymbol{x}^*, \boldsymbol{p}^*) = \Phi_\tau(\boldsymbol{x}, \boldsymbol{p})$ yields $\Delta H = H(\boldsymbol{x}^*, \boldsymbol{p}^*) - H(\boldsymbol{x}, \boldsymbol{p}) = 0$, and the Metropolis acceptance probability is $\min(1, e^{-\Delta H}) = 1$. In other words, **the exact Hamiltonian flow always accepts**. In practice we use a numerical integrator (the leapfrog) which introduces small discretisation errors, but acceptance rates remain high. Moreover, the momentum carries the trajectory along level sets of H in phase space, which correspond to the high-probability "shells" of π—hence the intuition of the ball rolling along the bottom of the valley.

3 The HMC Algorithm

Definition 3.1 (The HMC Algorithm)**.** Given the current position $\boldsymbol{x}^{(t)}$, one HMC transition proceeds as:

1. **Momentum refreshment.** Sample fresh momentum independently:

$$\boldsymbol{p}^{(0)} \sim \mathcal{N}(\boldsymbol{0}, I_d), \quad \text{i.e.,} \quad p_i^{(0)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1). \tag{6}$$

2. **Leapfrog integration.** Starting from $(\boldsymbol{x}^{(0)}, \boldsymbol{p}^{(0)}) = (\boldsymbol{x}^{(t)}, \boldsymbol{p}^{(0)})$, simulate Hamilton's equations for N_τ steps of size $\Delta\tau$ using the leapfrog integrator (Section 4) to obtain a proposal $(\boldsymbol{x}^*, \boldsymbol{p}^*)$.

3. **Metropolis correction.** Accept the proposal with probability

$$\alpha = \min\Big(1, e^{H(\boldsymbol{x}^{(0)},\boldsymbol{p}^{(0)}) - H(\boldsymbol{x}^*,\boldsymbol{p}^*)}\Big) = \min\big(1, e^{-\Delta H}\big)\,. \tag{7}$$

   If accepted, set $\boldsymbol{x}^{(t+1)} = \boldsymbol{x}^*$; otherwise, set $\boldsymbol{x}^{(t+1)} = \boldsymbol{x}^{(t)}$.

Remark 3.2. The momentum refreshment in Step 1 ensures *ergodicity*: by independently resampling $\boldsymbol{p}$ at the start of each iteration, the chain can explore all of phase space and is not confined to a single energy surface. The Metropolis correction in Step 3 corrects for the numerical error introduced by the leapfrog integrator. When using the exact Hamiltonian flow, $\Delta H = 0$ and the acceptance rate is 1.

4 The Leapfrog Integrator

Hamilton's equations (5) are a system of $2d$ first-order ODEs and generally lack closed-form solutions. We must therefore numerically integrate them. The leapfrog (Störmer–Verlet) method is the integrator of choice because it exactly preserves the two geometric properties needed for detailed balance: time-reversibility and volume preservation.

4.1 Derivation of the Leapfrog Scheme

Definition 4.1 (Leapfrog Integrator)**.** Given a step size $\Delta\tau > 0$ and total number of steps $N_\tau \in \mathbb{N}$, the leapfrog method advances the state $(\boldsymbol{x}(0), \boldsymbol{p}(0))$ to $(\boldsymbol{x}(N_\tau\Delta\tau), \boldsymbol{p}(N_\tau\Delta\tau))$ via the following sequence of updates:

Step 1 (initial half-step for position):

$$x_i(\Delta\tau/2) = x_i(0) + p_i(0) \cdot \frac{\Delta\tau}{2}. \tag{8}$$

Steps $n = 1, \ldots, N_\tau - 1$ (full alternating updates):

$$p_i(n\Delta\tau) = p_i((n-1)\Delta\tau) - \frac{\partial S}{\partial x_i}\big((n-\tfrac{1}{2})\Delta\tau\big) \cdot \Delta\tau, \tag{9}$$

$$x_i\big((n+\tfrac{1}{2})\Delta\tau\big) = x_i\big((n-\tfrac{1}{2})\Delta\tau\big) + p_i(n\Delta\tau) \cdot \Delta\tau. \tag{10}$$

Final steps (last full momentum update and final half-step for position):

$$p_i(N_\tau\Delta\tau) = p_i((N_\tau-1)\Delta\tau) - \frac{\partial S}{\partial x_i}\big((N_\tau-\tfrac{1}{2})\Delta\tau\big) \cdot \Delta\tau, \tag{11}$$

$$x_i(N_\tau\Delta\tau) = x_i\big((N_\tau-\tfrac{1}{2})\Delta\tau\big) + p_i(N_\tau\Delta\tau) \cdot \frac{\Delta\tau}{2}. \tag{12}$$

The key structural feature is that $\boldsymbol{x}$ and $\boldsymbol{p}$ are evaluated at times offset by half a step (hence "leapfrog"): $\boldsymbol{x}$ lives on the half-integer grid $\{(n-1/2)\Delta\tau\}$ while $\boldsymbol{p}$ lives on the integer grid $\{n\Delta\tau\}$. This staggering is not cosmetic—it is the source of both reversibility and symplecticity, as we now prove.

4.2 Time Reversibility of the Leapfrog

Theorem 4.2 (Time Reversibility)**.** *Let $(\boldsymbol{x}^*, \boldsymbol{p}^*)$ denote the output of the leapfrog integrator applied to $(\boldsymbol{x}, \boldsymbol{p})$ for N_τ steps of size $\Delta\tau$. Then applying the leapfrog to $(\boldsymbol{x}^*, -\boldsymbol{p}^*)$ for N_τ steps of the same size returns exactly $(\boldsymbol{x}, -\boldsymbol{p})$. Equivalently, the leapfrog map $\mathcal{L}$ satisfies*

$$\mathcal{N} \circ \mathcal{L} \circ \mathcal{N} = \mathcal{L}^{-1}, \tag{13}$$

where $\mathcal{N} : (\boldsymbol{x}, \boldsymbol{p}) \mapsto (\boldsymbol{x}, -\boldsymbol{p})$ is the momentum-negation map.

Proof. We prove this for a single component i and one full leapfrog cycle of N_τ steps; the general case follows by applying the argument to each component and composing over steps.

Forward pass: Starting from $(x_i(0), p_i(0))$, the leapfrog performs the sequence

$$\begin{aligned}
x_i\left(\tfrac{\Delta\tau}{2}\right) &= x_i(0) + p_i(0)\tfrac{\Delta\tau}{2}, \\
p_i(\Delta\tau) &= p_i(0) - \frac{\partial S}{\partial} x_i\left(\tfrac{\Delta\tau}{2}\right)\Delta\tau, \\
x_i\left(\tfrac{3\Delta\tau}{2}\right) &= x_i\left(\tfrac{\Delta\tau}{2}\right) + p_i(\Delta\tau)\Delta\tau, \\
&\vdots \\
p_i(N_\tau\Delta\tau) &= p_i((N_\tau - 1)\Delta\tau) - \frac{\partial S}{\partial} x_i\left((N_\tau - \tfrac{1}{2})\Delta\tau\right)\Delta\tau, \\
x_i(N_\tau\Delta\tau) &= x_i\left((N_\tau - \tfrac{1}{2})\Delta\tau\right) + p_i(N_\tau\Delta\tau)\tfrac{\Delta\tau}{2}.
\end{aligned}$$

Let the final state be $(x_i^*, p_i^*) = (x_i(N_\tau\Delta\tau), p_i(N_\tau\Delta\tau))$.

Reversed pass: Now start from $(x_i^*, -p_i^*)$ and apply the leapfrog. The first half-step gives

$$\tilde{x}_i\left(\tfrac{\Delta\tau}{2}\right) = x_i^* + (-p_i^*)\tfrac{\Delta\tau}{2} = x_i(N_\tau\Delta\tau) - p_i(N_\tau\Delta\tau)\tfrac{\Delta\tau}{2} = x_i\left((N_\tau - \tfrac{1}{2})\Delta\tau\right),$$

which is exactly where the forward pass was at position index $N_\tau - 1/2$. The full momentum update gives

$$\tilde{p}_i(\Delta\tau) = -p_i^* - \frac{\partial S}{\partial} x_i\left(\tilde{x}_i\left(\tfrac{\Delta\tau}{2}\right)\right)\Delta\tau = -p_i(N_\tau\Delta\tau) + \frac{\partial S}{\partial} x_i\left((N_\tau - \tfrac{1}{2})\Delta\tau\right)(-\Delta\tau)\cdot(-1)$$

Carefully tracking the signs, the leapfrog update for momentum is $p \leftarrow p - (\partial S/\partial x)\Delta\tau$.

Starting from $-p_i^*$:

$$\begin{aligned}\tilde{p}_i(\Delta\tau) &= -p_i^* - \frac{\partial S}{\partial} x_i\big((N_\tau - \tfrac{1}{2})\Delta\tau\big)\,\Delta\tau \\ &= -p_i(N_\tau\Delta\tau) - \frac{\partial S}{\partial} x_i\big((N_\tau - \tfrac{1}{2})\Delta\tau\big)\,\Delta\tau \\ &= -\left[p_i(N_\tau\Delta\tau) + \frac{\partial S}{\partial} x_i\big((N_\tau - \tfrac{1}{2})\Delta\tau\big)\,\Delta\tau\right] \\ &= -p_i((N_\tau - 1)\Delta\tau),\end{aligned}$$

where the last equality uses (11) in the form $p_i(N_\tau\Delta\tau) = p_i((N_\tau - 1)\Delta\tau) - (\partial S/\partial x_i)\Delta\tau$. Continuing this backward unrolling step by step, each leapfrog update in the reversed pass exactly undoes one step of the forward pass (with momentum negated), until after N_τ steps we arrive at $(\tilde{x}_i(N_\tau\Delta\tau), \tilde{p}_i(N_\tau\Delta\tau)) = (x_i(0), -p_i(0))$, establishing (13). □

4.3 Volume Preservation (Symplecticity) of the Leapfrog

Theorem 4.3 (Volume Preservation)**.** *Each elementary update of the leapfrog integrator—whether a half- or full-step—is a shear map in phase space with unit Jacobian determinant. Consequently, the full leapfrog map $\mathcal{L}$ satisfies $|\det J_{\mathcal{L}}| = 1$, i.e., the leapfrog preserves $2d$-dimensional phase-space volume.*

Proof. We analyse the two elementary updates separately.

Position update (momentum fixed). The half-step position update is

$$\boldsymbol{x} \leftarrow \boldsymbol{x} + \frac{\Delta\tau}{2}\boldsymbol{p}, \qquad \boldsymbol{p} \leftarrow \boldsymbol{p}.$$

Writing $(\boldsymbol{x}', \boldsymbol{p}') = (\boldsymbol{x} + \frac{\Delta\tau}{2}\boldsymbol{p}, \boldsymbol{p})$, the Jacobian of this map is the $(2d) \times (2d)$ block matrix

$$J_{\text{pos}} = \begin{pmatrix} I_d & \frac{\Delta\tau}{2} I_d \\ 0 & I_d \end{pmatrix}, \qquad \det J_{\text{pos}} = \det(I_d) \cdot \det(I_d) = 1, \tag{14}$$

using the formula $\det\begin{pmatrix} A & B \\ 0 & D \end{pmatrix} = \det A \cdot \det D$ for block triangular matrices.

Momentum update (position fixed). The full-step momentum update is

$$\boldsymbol{x} \leftarrow \boldsymbol{x}, \qquad \boldsymbol{p} \leftarrow \boldsymbol{p} - \Delta\tau\, \nabla_{\boldsymbol{x}} S(\boldsymbol{x}).$$

Writing $(\boldsymbol{x}', \boldsymbol{p}') = (\boldsymbol{x}, \boldsymbol{p} - \Delta\tau\, \nabla_{\boldsymbol{x}} S(\boldsymbol{x}))$, the Jacobian is

$$J_{\text{mom}} = \begin{pmatrix} I_d & 0 \\ -\Delta\tau\, \nabla^2_{\boldsymbol{x}} S & I_d \end{pmatrix}, \qquad \det J_{\text{mom}} = 1. \tag{15}$$

Since the full leapfrog map is a composition of such elementary updates, its Jacobian determinant is the product of individual determinants:

$$|\det J_{\mathcal{L}}| = \prod_k |\det J_k| = 1.$$

□

Remark 4.4. The leapfrog is in fact a *symplectic* integrator, meaning it preserves the symplectic 2-form $\omega = \sum_i \mathrm{d}x_i \wedge \mathrm{d}p_i$ on phase space, which is a stronger statement than mere volume preservation (Liouville's theorem is a consequence of symplecticity). This symplectic structure is what gives the leapfrog its superior long-time energy behaviour compared to non-symplectic integrators such as Euler or Runge–Kutta methods.

4.4 Discretisation Error

The leapfrog is a second-order method: the local truncation error per step is $O((\Delta\tau)^3)$, and the global error over a trajectory of fixed length $\tau_{\text{fin}} = N_\tau \Delta\tau$ is $O((\Delta\tau)^2)$. Consequently, the Hamiltonian error satisfies $|\Delta H| = O((\Delta\tau)^2)$, and the acceptance rate approaches 1 as $\Delta\tau \to 0$.

5 Detailed Balance and Convergence

We now prove rigorously that the HMC Markov kernel satisfies the detailed balance condition with respect to the target distribution, ensuring correctness of the algorithm.

5.1 The Extended Target and the Transition Kernel

The HMC algorithm operates on the extended phase space $(\boldsymbol{x}, \boldsymbol{p}) \in \mathbb{R}^{2d}$ with invariant distribution

$$\Pi(\boldsymbol{x}, \boldsymbol{p}) \propto e^{-H(\boldsymbol{x},\boldsymbol{p})} = e^{-S(\boldsymbol{x})} \cdot e^{-\frac{1}{2}\|\boldsymbol{p}\|^2}. \tag{16}$$

One HMC transition from $(\boldsymbol{x}, \boldsymbol{p})$ consists of:

1. Momentum refreshment: $\boldsymbol{p} \to \boldsymbol{p}^{\text{new}} \sim \mathcal{N}(\boldsymbol{0}, I)$, yielding $(\boldsymbol{x}, \boldsymbol{p}^{\text{new}})$.
2. Leapfrog proposal: $(\boldsymbol{x}^*, \boldsymbol{p}^*) = \mathcal{L}(\boldsymbol{x}, \boldsymbol{p}^{\text{new}})$.
3. Metropolis accept/reject with probability $\alpha = \min(1, e^{H(\boldsymbol{x},\boldsymbol{p}^{\text{new}}) - H(\boldsymbol{x}^*,\boldsymbol{p}^*)})$.

Theorem 5.1 (Detailed Balance for HMC)**.** *The HMC Markov kernel* $T\big((\boldsymbol{x}, \boldsymbol{p}) \to (\boldsymbol{x}', \boldsymbol{p}')\big)$ *satisfies the detailed balance condition with respect to* Π*:*

$$\Pi(\boldsymbol{x}, \boldsymbol{p})\, T\big((\boldsymbol{x}, \boldsymbol{p}) \to (\boldsymbol{x}', \boldsymbol{p}')\big) = \Pi(\boldsymbol{x}', \boldsymbol{p}')\, T\big((\boldsymbol{x}', \boldsymbol{p}') \to (\boldsymbol{x}, \boldsymbol{p})\big). \tag{17}$$

Proof. We write the proof in stages, leveraging Theorems 4.2 and 4.3.

Step 1: The proposal step. Consider a proposed move $(\boldsymbol{x}, \boldsymbol{p}) \to (\boldsymbol{x}^*, \boldsymbol{p}^*)$ generated by the leapfrog map $\mathcal{L}$. The transition density (in the continuous case, working with density kernels) for proposing $(\boldsymbol{x}^*, \boldsymbol{p}^*)$ starting from $(\boldsymbol{x}, \boldsymbol{p})$ is, by change of variables using the Jacobian of $\mathcal{L}$:

$$q\big((\boldsymbol{x}, \boldsymbol{p}) \to (\boldsymbol{x}^*, \boldsymbol{p}^*)\big) = \delta_{(\boldsymbol{x}^*, \boldsymbol{p}^*) - \mathcal{L}(\boldsymbol{x}, \boldsymbol{p})}, \quad |\det J_{\mathcal{L}}| = 1.$$

By time reversibility (Theorem 4.2), the reverse proposal $(\boldsymbol{x}^*, \boldsymbol{p}^*) \to (\boldsymbol{x}, \boldsymbol{p})$ under negated momentum is generated by $\mathcal{L}(\boldsymbol{x}^*, -\boldsymbol{p}^*) = (\boldsymbol{x}, -\boldsymbol{p})$.

Step 2: Metropolis accept/reject. The acceptance probability for the forward move is

$$\alpha_{\text{fwd}} = \min\left(1, \frac{e^{-H(\boldsymbol{x}^*, \boldsymbol{p}^*)}}{e^{-H(\boldsymbol{x}, \boldsymbol{p})}}\right),$$

and for the reverse move (from $(\boldsymbol{x}^*, -\boldsymbol{p}^*)$ to $(\boldsymbol{x}, -\boldsymbol{p})$):

$$\alpha_{\text{rev}} = \min\left(1, \frac{e^{-H(\boldsymbol{x}, -\boldsymbol{p})}}{e^{-H(\boldsymbol{x}^*, -\boldsymbol{p}^*)}}\right) = \min\left(1, \frac{e^{-H(\boldsymbol{x}, \boldsymbol{p})}}{e^{-H(\boldsymbol{x}^*, \boldsymbol{p}^*)}}\right),$$

using $H(\boldsymbol{x}, -\boldsymbol{p}) = S(\boldsymbol{x}) + \frac{1}{2}\|-\boldsymbol{p}\|^2 = H(\boldsymbol{x}, \boldsymbol{p})$.

Step 3: Verifying detailed balance. The flow of probability from $(\boldsymbol{x}, \boldsymbol{p})$ to $(\boldsymbol{x}^*, \boldsymbol{p}^*)$ is

$$\Pi(\boldsymbol{x}, \boldsymbol{p}) \cdot \alpha_{\text{fwd}} = e^{-H(\boldsymbol{x}, \boldsymbol{p})} \cdot \min\big(1, e^{H(\boldsymbol{x}, \boldsymbol{p}) - H(\boldsymbol{x}^*, \boldsymbol{p}^*)}\big).$$

Without loss of generality, assume $H(\boldsymbol{x}^*, \boldsymbol{p}^*) \geq H(\boldsymbol{x}, \boldsymbol{p})$. Then:

$$\begin{aligned}
\Pi(\boldsymbol{x}, \boldsymbol{p}) \cdot \alpha_{\text{fwd}} &= e^{-H(\boldsymbol{x}, \boldsymbol{p})} \cdot e^{H(\boldsymbol{x}, \boldsymbol{p}) - H(\boldsymbol{x}^*, \boldsymbol{p}^*)} = e^{-H(\boldsymbol{x}^*, \boldsymbol{p}^*)}, \\
\Pi(\boldsymbol{x}^*, \boldsymbol{p}^*) \cdot \alpha_{\text{rev}} &= e^{-H(\boldsymbol{x}^*, \boldsymbol{p}^*)} \cdot 1 = e^{-H(\boldsymbol{x}^*, \boldsymbol{p}^*)}.
\end{aligned}$$

The two sides are equal, establishing (17). The case $H(\boldsymbol{x}, \boldsymbol{p}) \geq H(\boldsymbol{x}^*, \boldsymbol{p}^*)$ is symmetric. □

Corollary 5.2 (Invariance of Π)**.** Π *is stationary for the HMC kernel, i.e., if* $(\boldsymbol{x}^{(0)}, \boldsymbol{p}^{(0)}) \sim \Pi$*, then* $(\boldsymbol{x}^{(1)}, \boldsymbol{p}^{(1)}) \sim \Pi$.

Proof. Detailed balance implies stationarity by integrating both sides of (17) over $(\boldsymbol{x}, \boldsymbol{p})$. □

Remark 5.3. Since momentum is refreshed at the start of each iteration, the HMC chain is also *aperiodic* and *irreducible* under mild regularity conditions on S (e.g., S continuous and the support of π connected), which together with stationarity imply *ergodicity*: the time average $\frac{1}{T}\sum_t f(\boldsymbol{x}^{(t)})$ converges to $\mathbb{E}_\pi[f]$ for Π-almost all starting points.

6 Applications and Parameter Tuning

6.1 The Role of N_τ and $\Delta\tau$

The two free parameters of HMC are the step size $\Delta\tau$ and the number of leapfrog steps N_τ. Their product, the trajectory length $\tau_{\text{fin}} = N_\tau \Delta\tau$, controls the distance the system travels in phase space before a Metropolis test is applied.

- **Large $\Delta\tau$:** coarser discretisation, larger ΔH, lower acceptance rate.
- **Small $\Delta\tau$:** finer discretisation, smaller ΔH, near-unit acceptance rate, but higher computational cost per proposal (since N_τ must be increased to maintain trajectory length).
- **Large $N_\tau\Delta\tau$:** longer trajectories, less correlated samples, but $O(N_\tau)$ gradient evaluations per step.
- **Small N_τ:** cheap proposals but potentially high autocorrelation (in the extreme, $N_\tau = 1$ reduces to a Langevin-type step).

The *optimal* acceptance rate for HMC in the limit of a d-dimensional target with i.i.d. components is 0.651, achieved by scaling $\Delta\tau \propto d^{-1/4}$ [6]. In practice, one targets acceptance rates in the range $[0.6, 0.9]$.

6.2 Handling Complex Targets

A notable application from the original literature [1] is sampling from matrix-valued actions appearing in lattice field theory:

$$S(\phi) = N \operatorname{Tr}\left(\frac{1}{2}\phi^2 + \frac{1}{4}\phi^4\right), \qquad \phi \in \mathbb{R}^{N\times N}.$$

The gradient $\nabla_\phi S(\phi) = N(\phi + \phi^3)$ is tractable, making HMC far more efficient than random-walk MCMC in the $d = N^2$-dimensional parameter space.

7 Visualising the HMC Trajectory in R

To develop intuition, we implement the leapfrog integrator in R and visualise the trajectory on a two-dimensional standard Gaussian target $\pi(\boldsymbol{x}) \propto \exp\left(-\frac{1}{2}(x_1^2 + x_2^2)\right)$, for which the potential energy and its gradient are:

$$U(\boldsymbol{x}) = S(\boldsymbol{x}) = \frac{1}{2}(x_1^2 + x_2^2), \qquad \nabla_{\boldsymbol{x}} U = \boldsymbol{x}.$$

Listing 1: R implementation of the leapfrog algorithm and HMC trajectory visualisation on a 2D standard Gaussian target.

```r
# Load required library
library(ggplot2)

# Define the gradient of the potential energy U(q) = 0.5 * ||q||^2
grad_U <- function(q) {
  return(q)  # Gradient of 0.5 * q^2 is simply q
}

# Leapfrog Algorithm
# q      : position vector (length 2)
# p      : momentum vector (length 2)
# epsilon: step size Delta_tau
# L      : number of leapfrog steps N_tau
leapfrog <- function(q, p, epsilon, L) {
  q_path <- matrix(NA, nrow = L + 1, ncol = 2)
  q_path[1, ] <- q

  # Initial half-step for momentum
  p <- p - epsilon * grad_U(q) / 2

  for (i in 1:L) {
    # Full position step
    q <- q + epsilon * p
    q_path[i + 1, ] <- q
    # Full momentum step (except at the last iteration)
    if (i != L) p <- p - epsilon * grad_U(q)
  }

  # Final half-step for momentum
  p <- p - epsilon * grad_U(q) / 2
  return(q_path)
}

# Initial states
q_init <- c(-2, 2)    # Starting position
p_init <- c(1, 0.5)   # Initial random momentum
epsilon <- 0.15        # Step size
L <- 40                # Number of leapfrog steps

# Generate trajectory
path    <- leapfrog(q_init, p_init, epsilon, L)
path_df <- data.frame(x = path[,1], y = path[,2], step = 1:(L+1))

```

```r
# Generate grid for contour plot of the 2D Gaussian PDF
grid_vals <- seq(-3, 3, length.out = 100)
grid      <- expand.grid(x = grid_vals, y = grid_vals)
grid$z    <- exp(-0.5 * (grid$x^2 + grid$y^2))

# Plot using ggplot2
ggplot() +
  geom_contour_filled(data = grid, aes(x = x, y = y, z = z),
    alpha = 0.8) +
  geom_path(data = path_df, aes(x = x, y = y),
            color = "red", size = 1,
            arrow = arrow(length = unit(0.15, "cm"))) +
  geom_point(data = path_df[1, ], aes(x = x, y = y),
             color = "yellow", size = 4) +
  geom_point(data = path_df[nrow(path_df), ], aes(x = x, y = y)
    ,
             color = "cyan", size = 4) +
  labs(title = "HMC Leapfrog Trajectory on 2D Target
    Distribution",
       subtitle = "Yellow = Start, Cyan = End",
       x = "x1", y = "x2") +
  theme_minimal() +
  scale_fill_viridis_d(option = "magma") +
  theme(legend.position = "none")
```

The resulting figure (Figure 1) shows the leapfrog trajectory (red) overlaid on the contour plot of the target density. The trajectory follows the high-density elliptical shell of the Gaussian, illustrating how HMC exploits the gradient to propose distant states while remaining in regions of high probability.

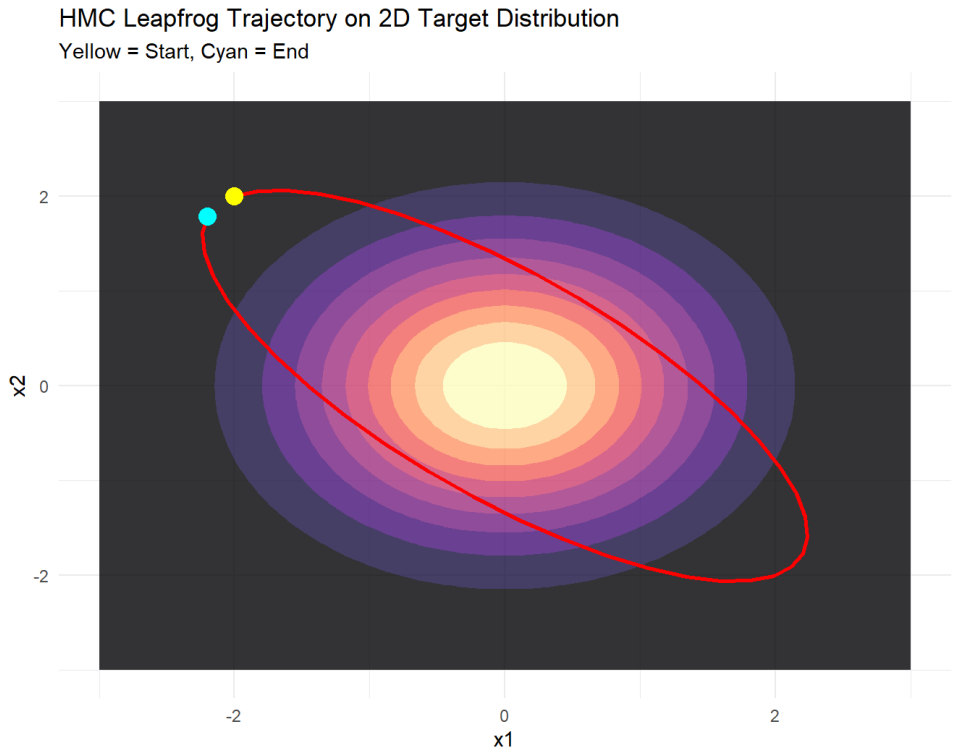

Figure 1: HMC Leapfrog Trajectory on 2D Target Distribution. The trajectory (red) efficiently explores the high-density region while conserving Hamiltonian energy.

8 The No-U-Turn Sampler (NUTS)

A practical obstacle to deploying HMC is the need to manually specify N_τ (the trajectory length). Too small a value gives short, correlated proposals; too large a value wastes computation once the trajectory begins to "turn back" on itself. The **No-U-Turn Sampler (NUTS)**, introduced by Hoffman and Gelman [5], solves this problem by adaptively determining the trajectory length.

8.1 The U-Turn Criterion

NUTS extends the trajectory by doubling in both directions simultaneously using a tree-building procedure, stopping when the trajectory begins to turn back toward its origin. Formally, a U-turn is detected when

$$(\boldsymbol{x}^+ - \boldsymbol{x}^-) \cdot \boldsymbol{p}^+ < 0 \quad \text{or} \quad (\boldsymbol{x}^+ - \boldsymbol{x}^-) \cdot \boldsymbol{p}^- < 0, \tag{18}$$

where $\boldsymbol{x}^+, \boldsymbol{p}^+$ (resp. $\boldsymbol{x}^-, \boldsymbol{p}^-$) are the rightmost (resp. leftmost) states of the current trajectory. When condition (18) is satisfied, the tree stops growing and a new position is sampled from the current subtree using a biased progressive sampling scheme that maintains detailed balance.

8.2 Dual Averaging for Step-Size Adaptation

NUTS also incorporates the **dual averaging** scheme of Nesterov [10] to adaptively tune $\Delta\tau$ during a warm-up phase, targeting a user-specified acceptance rate (default 0.8 in Stan). After warm-up, $\Delta\tau$ is fixed and NUTS runs with standard detailed balance.

8.3 Stan and rstan

Stan [8] is a probabilistic programming language that compiles user-specified models to efficient HMC/NUTS samplers with automatic differentiation for gradient computation. The `rstan` package provides an R interface to Stan.

9 Application: Bayesian Calibration of the Vasicek SDE via HMC

We demonstrate the power of HMC/NUTS in a concrete Bayesian inference problem: recovering the parameters of the **Vasicek short-rate model** [7] from discrete observations.

9.1 The Vasicek Model

The Vasicek model specifies the short rate r_t as the solution of the linear SDE:

$$\mathrm{d}r_t = a(b - r_t)\,\mathrm{d}t + \sigma\,\mathrm{d}W_t, \tag{19}$$

where $a > 0$ is the speed of mean reversion, $b \in \mathbb{R}$ is the long-run mean, $\sigma > 0$ is the volatility, and $\{W_t\}$ is a standard Brownian motion. The three parameters $\theta = (a, b, \sigma)$ are to be inferred from observed data $\boldsymbol{r} = (r_1, r_2, \ldots, r_N)$ at equally spaced times with step Δt.

9.2 Exact Transition Density

The Vasicek SDE is an Ornstein–Uhlenbeck process and admits a closed-form conditional distribution. Given r_t, the next observation $r_{t+\Delta t}$ satisfies:

$$r_{t+\Delta t} \mid r_t \sim \mathcal{N}\big(\mu_t,\ \nu^2\big)\,, \tag{20}$$

where

$$\mu_t = r_t\, e^{-a\Delta t} + b\left(1 - e^{-a\Delta t}\right), \tag{21}$$

$$\nu^2 = \frac{\sigma^2}{2a}\left(1 - e^{-2a\Delta t}\right). \tag{22}$$

Derivation of the exact transition density. From the integrating factor solution of the Vasicek SDE (as derived in detail in the companion report [**?**]), for $s < t$:

$$r_t = b + (r_s - b)e^{-a(t-s)} + \sigma \int_s^t e^{-a(t-u)}\, \mathrm{d}W_u. \tag{23}$$

Since r_s is $\mathcal{F}_s$-measurable and the stochastic integral $\int_s^t e^{-a(t-u)}\, \mathrm{d}W_u$ is independent of $\mathcal{F}_s$, the conditional distribution of r_t given r_s is Gaussian with mean

$$\mathbb{E}[r_t \mid r_s] = b + (r_s - b)e^{-a(t-s)} = r_s e^{-a(t-s)} + b(1 - e^{-a(t-s)}),$$

and variance, by the Itô isometry:

$$\mathrm{Var}(r_t \mid r_s) = \sigma^2 \int_s^t e^{-2a(t-u)}\, \mathrm{d}u = \frac{\sigma^2}{2a}\left(1 - e^{-2a(t-s)}\right).$$

Setting $t - s = \Delta t$ recovers (21)–(22). □

9.3 Bayesian Model and Likelihood

The exact log-likelihood for N observations (conditioning on r_1) is:

$$\ell(\theta \mid \boldsymbol{r}) = \sum_{t=1}^{N-1} \log \phi\big(r_{t+1}; \mu_t, \nu^2\big)\,, \tag{24}$$

where $\phi(\cdot; \mu, \nu^2)$ is the Gaussian density. We place the following weakly informative priors:

$$a \sim \mathcal{N}(0, 2^2) \cdot \mathbf{1}[a > 0], \quad b \sim \mathcal{N}(0.05, 0.05^2), \quad \sigma \sim \text{Half-Cauchy}(0, 0.1). \tag{25}$$

The posterior is $\pi(\theta \mid \boldsymbol{r}) \propto e^{\ell(\theta|\boldsymbol{r})} \cdot \pi_0(\theta)$, which does not have a closed form due to the nonlinear dependence of μ_t and ν^2 on a. This is exactly where HMC excels: the gradient of the log-posterior with respect to (a, b, σ) can be computed via automatic differentiation, and NUTS traverses the correlated posterior geometry efficiently.

9.4 Simulation Study: Data Generation

We first generate synthetic observations from the known true parameters $(a^*, b^*, \sigma^*) = (0.5, 0.05, 0.02)$ using the Euler–Maruyama discretisation with $\Delta t = 1/252$ (one trading day) over $N = 250$ steps, with $r_1 = 0.04$.

9.5 Stan Model and R Implementation

Listing 2: Full R implementation: data generation, Stan model specification, HMC sampling via rstan, and trace plot diagnostics.

```r
library(rstan)

# ---- 1. Simulate synthetic Vasicek data ----
set.seed(42)
N          <- 250
dt         <- 1 / 252    # daily step size
a_true     <- 0.5
b_true     <- 0.05
sigma_true <- 0.02

r    <- numeric(N)
r[1] <- 0.04
for (t in 2:N) {
  dr   <- a_true * (b_true - r[t-1]) * dt + sigma_true * sqrt(dt) * rnorm(1)
  r[t] <- r[t-1] + dr
}

stan_data <- list(N = N, dt = dt, r = r)

# ---- 2. Stan model using the exact transition density ----
stan_code <- "
data {
  int<lower=1> N;
  real dt;
  vector[N] r;
}
parameters {
  real<lower=0> a;
  real b;
  real<lower=0> sigma;
}
model {
  // Weakly informative priors
  a     ~ normal(0, 2);
  b     ~ normal(0.05, 0.05);
  sigma ~ cauchy(0, 0.1);

  // Exact Gaussian transition density (Ornstein-Uhlenbeck)
  for (t in 2:N) {
    real mean_r = r[t-1] * exp(-a * dt) + b * (1 - exp(-a * dt));
    real var_r  = (square(sigma) / (2 * a)) * (1 - exp(-2 * a * dt));
```

```
    r[t] ~ normal(mean_r, sqrt(var_r));
  }
}
"

# ---- 3. Run NUTS via rstan ----
fit <- stan(
  model_code = stan_code,
  data       = stan_data,
  iter       = 2000,     # 1000 warmup + 1000 sampling per chain
  chains     = 4,
  seed       = 123
)

# ---- 4. Diagnostics: trace plots ----
# A well-mixed chain produces overlapping "fuzzy caterpillar"
    traces.
traceplot(fit, pars = c("a", "b", "sigma"), inc_warmup = FALSE)

# ---- 5. Print posterior summary ----
print(fit, pars = c("a", "b", "sigma"), probs = c(0.025, 0.5,
    0.975))
```

9.6 Interpreting the Trace Plots

The trace plot (Figure 2) shows four independent Markov chains over 1000 post-warmup iterations for each of the three parameters (a, b, σ). Convergence is assessed via:

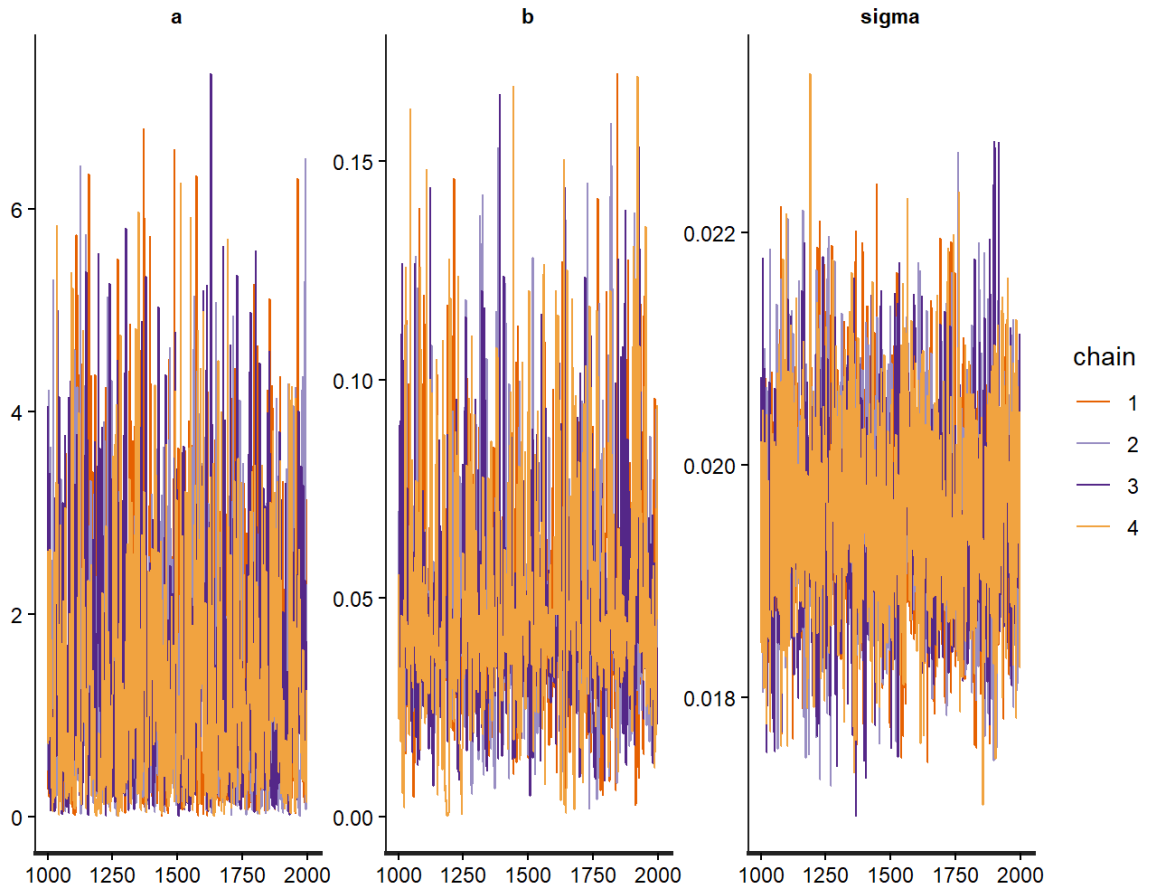

Figure 2: Trace plots for the Vasicek parameters (a, b, σ) across 4 chains. The "fuzzy caterpillar" appearance indicates excellent mixing and convergence.

1. **Visual mixing:** The overlapping "fuzzy caterpillar" pattern indicates that all four chains are exploring the same region of the posterior, consistent with convergence.
2. $\hat{R}$ **statistic (Gelman–Rubin):** The potential scale reduction factor $\hat{R} = \sqrt{\hat{V}/W}$, where $\hat{V}$ is the between-chain variance estimator and W the within-chain variance, should satisfy $\hat{R} < 1.01$ for good mixing. Stan computes this automatically.
3. **Effective sample size (ESS):** The number of effectively independent samples from the chain; NUTS typically achieves $\text{ESS}/T \gg 0.1$, far exceeding random-walk MCMC on correlated posteriors.

Remark 9.1. The Vasicek posterior is highly correlated in the (a, b) subspace because the long-run mean b and the speed a jointly determine the stationary mean $\mathbb{E}[r_\infty] = b$. Standard random-walk MH would require many more iterations to explore this curved ridge, while NUTS traverses it efficiently due to gradient-guided proposals.

10 Discussion: HMC vs. Standard MCMC

Table 1 summarises the key differences between the standard Metropolis–Hastings (MH) random walk and HMC/NUTS.

Table 1: Comparison of Metropolis–Hastings and HMC/NUTS.

Property	**Metropolis–Hastings**	**HMC / NUTS**
Proposal mechanism	Random walk	Gradient-guided leapfrog
Gradient required	No	Yes
Scaling with d	$O(d^2)$ steps	$O(d^{5/4})$ steps [6]
Autocorrelation	High in $d \gg 1$	Low
Parameter tuning	Step size h only	$\Delta\tau$, N_τ (NUTS: automatic)
Typical acceptance	~ 0.234 (optimal 1D)	~ 0.65 (optimal, any d)

The fundamental scaling advantage of HMC—$O(d^{5/4})$ gradient evaluations vs. $O(d^2)$ proposals for MH—makes it indispensable for modern Bayesian computation with hundreds to thousands of parameters, as exemplified by its adoption as the default sampler in Stan and PyMC.

11 Conclusion

This report has provided a complete, rigorous treatment of the Hamiltonian Monte Carlo algorithm, from first principles through to practical implementation. The main contributions are:

1. A self-contained derivation of the *physics intuition* underlying HMC, connecting Hamiltonian mechanics to MCMC sampling.
2. Detailed proofs of the two essential properties of the leapfrog integrator—*time reversibility* (Theorem 4.2) and *volume preservation* (Theorem 4.3)—and their use in proving the *detailed balance condition* (Theorem 5.1), which guarantees correct convergence to the target distribution.
3. An R implementation and visualisation of the HMC trajectory on a 2D Gaussian target, providing geometric intuition.
4. An introduction to NUTS as an adaptive extension that eliminates manual tuning of the trajectory length.
5. A concrete Bayesian application: posterior inference for the Vasicek SDE parameters using Stan, demonstrating HMC's superiority over random-walk MCMC in highly correlated parameter spaces.

The HMC algorithm exemplifies a beautiful connection between theoretical physics and computational statistics: the same equations that govern the motion of a ball rolling on

a landscape allow us to explore complex probability distributions with unprecedented efficiency. As models grow in dimension and complexity—from Bayesian neural networks to large-scale latent variable models—gradient-based samplers such as HMC and NUTS will remain central tools of the statistical practitioner.

References

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

[2] Neal, R.M. (2011). MCMC Using Hamiltonian Dynamics. In S. Brooks, A. Gelman, G. Jones, & X.L. Meng (Eds.), *Handbook of Markov Chain Monte Carlo* (Chapter 5). Chapman & Hall / CRC Press.

[3] Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., & Teller, E. (1953). Equation of State Calculations by Fast Computing Machines. *Journal of Chemical Physics*, 21(6), 1087–1092.

[4] Hastings, W.K. (1970). Monte Carlo Sampling Methods Using Markov Chains and Their Applications. *Biometrika*, 57(1), 97–109.

[5] Hoffman, M.D., & Gelman, A. (2014). The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15, 1593–1623.

[6] Beskos, A., Pillai, N., Roberts, G., Sanz-Serna, J.M., & Stuart, A. (2013). Optimal Tuning of the Hybrid Monte Carlo Algorithm. *Bernoulli*, 19(5A), 1501–1534.

[7] Vašíček, O. (1977). An Equilibrium Characterization of the Term Structure. *Journal of Financial Economics*, 5(2), 177–188.

[8] Stan Development Team (2023). Stan Modeling Language Users Guide and Reference Manual, Version 2.32. https://mc-stan.org.

[9] Hanada, M., & Matsuura, S. (2022). *MCMC from Scratch: A Practical Introduction to Markov Chain Monte Carlo*. Springer Nature Singapore.

[10] Nesterov, Y. (2009). Primal-Dual Subgradient Methods for Convex Problems. *Mathematical Programming*, 120(1), 221–259.

[11] Document provided by professor (Excerpt on Hamiltonian Monte Carlo algorithm).