Markov Chain Monte Carlo II

Probabilistic Decision Making VU

(Reinforcement Learning VO)

Thomas Wedenig

Dec 17, 2025

Institute of Machine Learning and Neural Computation
Graz University of Technology, Austria

Table of contents

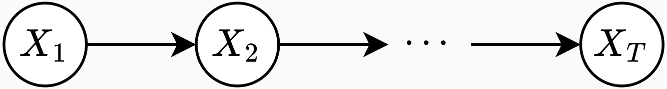
1. MCMC - Recap
2. Estimating Expectations with MCMC Samples
3. Gibbs Sampling
4. Langevin MCMC
5. Hamiltonian Monte Carlo
6. Diagnosing MCMC

MCMC - Recap

Setup

- **Goal**: Draw **samples** from **distribution** $\pi(\mathsf{x})$
- Typically, only $\tilde{\pi}(\mathsf{x}) \propto \pi(\mathsf{x})$ is known
  - e.g., **Bayesian inference**: $\tilde{\pi}_{\mathsf{y}}(\mathsf{x}) := p(\mathsf{x}, \mathsf{y}) \propto p(\mathsf{x} \mid \mathsf{y})$
  - e.g., **Boltzmann distributions**: $\tilde{\pi}(\mathsf{x}) := \exp(-E(\mathsf{x}))$, $E : \mathcal{X} \to \mathbb{R}$
- We care about **samples** because...
  - they might be **the object of interest** (e.g., sample image from generative model), or
  - we may need samples to **approximate expectations** (Monte Carlo)

MCMC – Idea



- **Idea**: Construct **Markov Chain**

$$p(x_1, x_2, \ldots, x_T) = p(x_1)\, p(x_2 \mid x_1)\, p(x_3 \mid x_2) \ldots p(x_T \mid x_{T-1})$$

  - with **unique stationary distribution** π,
  - its **marginal distribution converges to** π when $T \to \infty$,
  - $p(x_1)$ and $p(x_t \mid x_{t-1})$ are **easy to sample from**

Sampling from the chain for “long enough” yields
samples with **marginal distribution close to** π !

MCMC

```
Initialize $x$
repeat:
    $x \leftarrow$ some (random) update to $x$
```

Running a

Markov Chain Monte Carlo

Algorithm

Metropolis-Hastings

Given: **Target** $\tilde{\pi}$ and **proposal distribution** $q(\mathsf{y} \mid \mathsf{x})$.

Initialize $X_1 \sim p(X_1)$ and repeat for $t = 1, \dots, T-1$:

1. Given current state $X_t = \mathsf{x}$, **propose:**

$$\mathsf{y} \sim q(\mathsf{y} \mid \mathsf{x})$$

2. **Compute acceptance probability:**

$$\alpha(\mathsf{x} \to \mathsf{y}) = \min\left(1, \frac{\tilde{\pi}(\mathsf{y})\, q(\mathsf{x} \mid \mathsf{y})}{\tilde{\pi}(\mathsf{x})\, q(\mathsf{y} \mid \mathsf{x})}\right)$$

3. **Accept or reject:**

$$\text{Sample } U \sim \mathsf{Uniform}(0,1), \qquad X_{t+1} = \begin{cases} \mathsf{y} & \text{if } U \leq \alpha(\mathsf{x} \to \mathsf{y}) \\ \mathsf{x} & \text{else} \end{cases}$$

Metropolis-Hastings (cont.)

Metropolis-Hastings Update

1️⃣ Propose $\mathbf{y} \sim q(\mathbf{y} \mid \mathbf{x})$ 2️⃣ Compute $\alpha = \min\left(1, \frac{\tilde{\pi}(\mathbf{y})\, q(\mathbf{x}|\mathbf{y})}{\tilde{\pi}(\mathbf{x})\, q(\mathbf{y}|\mathbf{x})}\right)$ 3️⃣ Accept $\mathbf{y}$ with probability α

- For continuous state spaces, we could choose a **Gaussian proposal**

$$q(\mathbf{y} \mid \mathbf{x}) := \mathcal{N}(\mathbf{y}; \mathbf{x}, \sigma^2 I)$$

- Demo last week: Discrete state space with **proposal**

$$q(\mathbf{y} \mid \mathbf{x}) = \mathbf{Uniform}\left(\{\text{bitstrings that differ from } \mathbf{x} \text{ in exactly one bit}\}\right)$$

Analogy: Optimization

- Assume we want to find the **maximum** of $\tilde{\pi}$
- **Idea:** With current state $\mathbf{x}$, repeat
  1️⃣ Propose $\mathbf{y} \sim q(\mathbf{y} \mid \mathbf{x})$
  2️⃣ Set $\alpha = \tilde{\pi}(\mathbf{y})/\tilde{\pi}(\mathbf{x})$
  3️⃣ If $\alpha \geq 1$, accept $\mathbf{y}$, else stay at $\mathbf{x}$
- **"Greedy random search"**

Random-Walk Metropolis-Hastings Demo

`https://chi-feng.github.io/mcmc-demo/app.html`

Estimating Expectations with MCMC Samples

Dependency in MCMC Samples

Recall Goal: Draw i.i.d. samples from π.

Are samples from our Markov Chain **i.i.d.** samples from π? 🤔

No !

- If samples were i.i.d., X_t and X_{t+1} would be **independent**
- In the beginning, X_t is not even **marginally** distributed like π (not converged yet)
- Even when the chain has perfectly converged to π, and thus $X_t \sim \pi$, $X_{t+1} \sim \pi$ **marginally**, we still have **possibly large correlation** between X_t and X_{t+1}
- When the finite-state chain has converged, we have

$$p(X_t = j, X_{t+1} = i) = p(X_t = j)p(X_{t+1} = i \mid X_t = j) = \pi_j P_{ij} \neq \pi_j \pi_i = p(X_t = j)p(X_{t+1} = i)$$

Ergodic Theorem

Are we doomed? Monte Carlo estimators need **i.i.d.** samples 😰

Thankfully, no! 🥳

- There is an **MCMC counterpart** to classic i.i.d. Monte Carlo
- It assumes our Markov Chain has π as its **unique stationary distribution**
- And it assumes **certain properties**[1]

[1] Irreducible (when state space finite), irreducible and positive recurrent (state space countable), π-irreducible and Harris recurrent (state space uncountable.)

Ergodic Theorem

Ergodic Theorem

Assume $X_1, X_2, \ldots$ is a Markov Chain with above properties and stationary distribution π. Then,

$$\hat{\mathbb{E}}_T := \frac{1}{T}\sum_{t=1}^{T} f(X_t) \xrightarrow{\text{a.s.}} \mathbb{E}_{X\sim\pi}[f(X)]$$

as $T \to \infty$ for every initial state X_1.

- **Almost sure (a.s.) convergence** means

$$\mathbb{P}\left(\lim_{T\to\infty} \hat{\mathbb{E}}_T = \mathbb{E}_{X\sim\pi}[f(X)]\right) = 1$$

MCMC Estimate – Bias

Recap: Properties of MC Estimate

Given **i.i.d.** samples $Y_1, \ldots, Y_T \sim \pi$, the **Monte Carlo estimate** of $\mathbb{E}_{X\sim\pi}[f(X)]$ is

$$\hat{\mathbb{E}}_T^{(\mathrm{MC})} := \frac{1}{T}\sum_{t=1}^{T} f(Y_t).$$

- **Unbiased** for every T: $\mathbb{E}\left[\hat{\mathbb{E}}_T^{(\mathrm{MC})}\right] = \mathbb{E}_{X\sim\pi}[f(X)]$

Properties of MCMC Estimate

- MCMC estimate is generally[2] **biased** for any finite T
  - We can reduce bias by **discarding the first** B samples (**burn-in phase**)
- Bias goes to 0 as $T \to \infty$, i.e., **asymptotically unbiased**

[2] It would be unbiased if the initial distribution was already π, but we can't do that in practice.

MCMC Estimate – Variance

Recap: Properties of MC Estimate

- **Variance** decreases **linearly** with T: $\mathrm{var}\left[\hat{\mathbb{E}}_T^{(\mathrm{MC})}\right] = \frac{\mathrm{var}[f(X)]}{T}$

Properties of MCMC Estimate

- As $T \to \infty$, the **variance** of the MCMC estimator is approximately

$$\mathrm{var}\left[\hat{\mathbb{E}}_T\right] \approx \frac{\sigma^2_{\mathrm{as}}}{T}$$

with

$$\sigma^2_{\mathrm{as}} = \mathrm{var}_\pi[f(X)] + 2\sum_{n=1}^{\infty} \mathrm{cov}_\pi(f(X_0), f(X_n)) \, .$$

where $X_0 \sim \pi, X_1 \sim \pi, \ldots$ (i.e., chain in **stationary** regime)

- **Note:** σ^2_{as} typically **larger** than $\mathrm{var}_\pi[f(X)]$, but still decreases **linearly** with T

Gibbs Sampling

Gibbs Sampling

- Target: $\pi(\mathbf{x})$ with $\mathbf{x} = (x_1, \ldots, x_d)^\top \in \mathcal{X}$
    - e.g., $\mathcal{X} = \mathbb{R}^d$ or $\mathcal{X} = \{-1, +1\}^d$
- Let $\mathbf{x}_{-i} := (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_d)$
- Assume **conditionals** $\pi(x_i \mid \mathbf{x}_{-i})$ can be **sampled easily** for every i

Gibbs Sampling

Initialize $\mathbf{x}^{(1)} \sim p(\mathbf{x}^{(1)})$ and repeat for $t = 1, \ldots, T-1$:

- Given current state $\mathbf{x}^{(t)} := (x_1^{(t)}, \ldots, x_d^{(t)})$, sample

$$\begin{aligned}
x_1^{(t+1)} &\sim \pi\left(x_1 \mid x_2^{(t)}, x_3^{(t)}, \ldots, x_d^{(t)}\right) \\
x_2^{(t+1)} &\sim \pi\left(x_2 \mid x_1^{(t+1)}, x_3^{(t)}, \ldots, x_d^{(t)}\right) \\
x_3^{(t+1)} &\sim \pi\left(x_3 \mid x_1^{(t+1)}, x_2^{(t+1)}, x_4^{(t)}, \ldots, x_d^{(t)}\right) \\
&\vdots \\
x_d^{(t+1)} &\sim \pi\left(x_d \mid x_1^{(t+1)}, x_2^{(t+1)}, \ldots, x_{d-1}^{(t+1)}\right)
\end{aligned}$$

Gibbs Sampling Demo

`https://chi-feng.github.io/mcmc-demo/app.html`

Gibbs Sampling is a Metropolis-Hastings Algorithm

Metropolis-Hastings Update

1️⃣ Propose $\mathbf{y} \sim q(\mathbf{y} \mid \mathbf{x})$ 2️⃣ Compute $\alpha = \min\left(1, \frac{\tilde{\pi}(\mathbf{y})\, q(\mathbf{x}|\mathbf{y})}{\tilde{\pi}(\mathbf{x})\, q(\mathbf{y}|\mathbf{x})}\right)$ 3️⃣ Accept $\mathbf{y}$ with probability α

- Gibbs Sampling can be **viewed as an instance of Metropolis-Hastings (MH)**:

$$\mathbf{y} \sim q_i(\mathbf{y} \mid \mathbf{x}) \;:=\; \pi(y_i \mid \mathbf{x}_{-i})\, 1\{\mathbf{y}_{-i} = \mathbf{x}_{-i}\}.$$

- Hence, $\mathbf{y}_{-i} = \mathbf{x}_{-i}$, and thus

$$\pi(\mathbf{x}) = \pi(\mathbf{x}_{-i})\, \pi(x_i \mid \mathbf{x}_{-i}), \qquad \pi(\mathbf{y}) = \pi(\mathbf{x}_{-i})\, \pi(y_i \mid \mathbf{x}_{-i}),$$

- The MH ratio becomes

$$\frac{\pi(\mathbf{y})\, q_i(\mathbf{x} \mid \mathbf{y})}{\pi(\mathbf{x})\, q_i(\mathbf{y} \mid \mathbf{x})} = \frac{\left[\pi(\mathbf{x}_{-i})\, \pi(y_i \mid \mathbf{x}_{-i})\right]\, \pi(x_i \mid \mathbf{x}_{-i})}{\left[\pi(\mathbf{x}_{-i})\, \pi(x_i \mid \mathbf{x}_{-i})\right]\, \pi(y_i \mid \mathbf{x}_{-i})} = 1$$

and we accept with **probability $\alpha = 1$**.

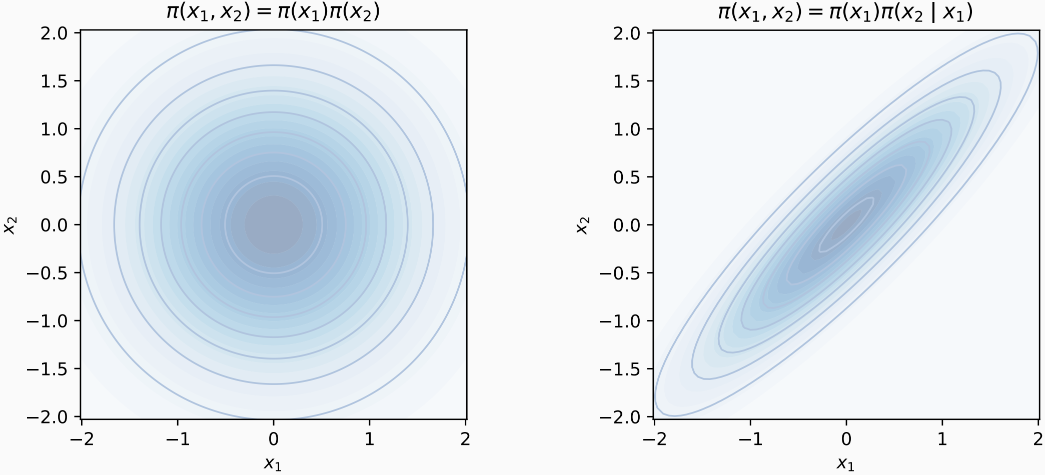
Gibbs Sampling is a Metropolis-Hastings Algorithm

- **Every coordinate update can be seen as a MH step that always accepts** 😍

- Gibbs Sampling typically only stores **every d^{th} MH sample**
  - Technically, this is not necessary if we e.g. apply the **ergodic theorem**
  - But it **reduces autocorrelation**
  - ...at the expense of “throwing away” samples

- Theoretical caveat: In this view, q_i is **time-dependent**
  - Issue: the ergodic theorem assumes a **time-homogeneous** chain
  - Luckily, we can make it time-homogeneous by augmenting the state space $\mathbf{z} := (\mathbf{x}, i)$
  - We then consider $q(\mathbf{y} \mid \mathbf{z})$ in this augmented space, which also updates i accordingly

Gibbs Sampling from a Gaussian



$\pi(x_1, x_2) = \pi(x_1)\pi(x_2)$

$\pi(x_1, x_2) = \pi(x_1)\pi(x_2 \mid x_1)$

Gibbs Sampling Highly Correlated Variables

- Per-coordinate updates may yield **large autocorrelation**
  - Especially for **highly correlated variables**
  - We cannot **“move diagonally”** in the state space
- In the above example, we of course could sample both coordinates at once ❗
  - In general, we can’t sample **all coordinates at once**
  - ...otherwise we would not need MCMC 😂
- Is there something in the middle? 🤔
- **Blocked Gibbs Sampling**: Sample **multiple coordinates** at once

Blocked Gibbs Sampling

Let

$$\mathbf{x} = (\mathbf{x}_{B_1}, \mathbf{x}_{B_2}, \ldots, \mathbf{x}_{B_K}), \qquad B_1 \cup \cdots \cup B_K = \{1, \ldots, d\}.$$

Initialize $\mathbf{x}^{(1)} \sim p(\mathbf{x}^{(1)})$ and repeat for $t = 1, \ldots, T-1$:

- Given current state $\mathbf{x}^{(t)} := (\mathbf{x}_{B_1}^{(t)}, \ldots, \mathbf{x}_{B_K}^{(t)})$, sample

$$\mathbf{x}_{B_1}^{(t+1)} \sim \pi\left(\mathbf{x}_{B_1} \,\middle|\, \mathbf{x}_{B_2}^{(t)}, \mathbf{x}_{B_3}^{(t)}, \ldots, \mathbf{x}_{B_K}^{(t)}\right)$$

$$\mathbf{x}_{B_2}^{(t+1)} \sim \pi\left(\mathbf{x}_{B_2} \,\middle|\, \mathbf{x}_{B_1}^{(t+1)}, \mathbf{x}_{B_3}^{(t)}, \ldots, \mathbf{x}_{B_K}^{(t)}\right)$$

$$\mathbf{x}_{B_3}^{(t+1)} \sim \pi\left(\mathbf{x}_{B_3} \,\middle|\, \mathbf{x}_{B_1}^{(t+1)}, \mathbf{x}_{B_2}^{(t+1)}, \mathbf{x}_{B_4}^{(t)}, \ldots, \mathbf{x}_{B_K}^{(t)}\right)$$

$$\vdots$$

$$\mathbf{x}_{B_K}^{(t+1)} \sim \pi\left(\mathbf{x}_{B_K} \,\middle|\, \mathbf{x}_{B_1}^{(t+1)}, \mathbf{x}_{B_2}^{(t+1)}, \ldots, \mathbf{x}_{B_{K-1}}^{(t+1)}\right)$$

Metropolis-within-Gibbs

What if we cannot directly sample from some **conditional distribution** ?

$$\mathbf{x}_{B_i}^{(t+1)} \sim \pi\left(\mathbf{x}_{B_i} \,\middle|\, \mathbf{x}_{B_1}^{(t+1)}, \ldots, \mathbf{x}_{B_{i-1}}^{(t+1)}, \mathbf{x}_{B_{i+1}}^{(t)}, \ldots, \mathbf{x}_{B_K}^{(t)}\right)$$

- We could use MCMC again **just for this block** !
    - e.g., random walk MH (or something smarter)
- This is called **Metropolis-within-Gibbs**

Langevin MCMC

Langevin MCMC – Motivation

Maximizing $\tilde{\pi}$ via greedy random search

With current state $\mathbf{x}$, repeat:

1️⃣ Propose $\mathbf{y} \sim q(\mathbf{y} \mid \mathbf{x})$ 2️⃣ Set $\alpha = \tilde{\pi}(\mathbf{y})/\tilde{\pi}(\mathbf{x})$ 3️⃣ If $\alpha \geq 1$, accept $\mathbf{y}$, else stay at $\mathbf{x}$

This is clearly **ridiculous** in most cases 😂

- If $\tilde{\pi}$ is a **differentiable** PDF, we can use **gradient information** to find a maximum
  - e.g., **gradient ascent**
- In MCMC **sampling**, we want to **explore regions of high probability**
  - Gradients can guide us towards regions of **high density**

Metropolis-Adjusted Langevin Algorithm (MALA)

- Metropolis-Adjusted Langevin Algorithm (MALA)

- MALA is an instance of Metropolis-Hastings with
$$q(\mathrm{y} \mid \mathrm{x}) = \mathcal{N}\left(\mathrm{y};\ \mathrm{x} + \frac{\eta}{2}\nabla_{\mathrm{x}} \log \pi(\mathrm{x}),\ \eta I\right)$$
  with step size $\eta > 0$.

- As usual, we accept $\mathbf{y}$ with probability
$$\alpha(\mathrm{x} \to \mathrm{y}) = \min\left(1, \frac{\tilde{\pi}(\mathrm{y})\, q(\mathrm{x} \mid \mathrm{y})}{\tilde{\pi}(\mathrm{x})\, q(\mathrm{y} \mid \mathrm{x})}\right)$$

- **Note:** We have to be able to compute $\nabla_{\mathrm{x}} \log \tilde{\pi}(\mathrm{x}) = \nabla_{\mathrm{x}} \log \pi(\mathrm{x})$
  - And evaluate $\tilde{\pi}$ in the **acceptance step**

MALA Demo

`https://chi-feng.github.io/mcmc-demo/app.html`

MALA – Background and Motivation

- This **choice of** q is motivated by the **Langevin Stochastic Differential Equation (SDE)**:

$$dX_t = \frac{1}{2}\nabla \log \pi(X_t)\, dt + dW_t$$

where W_t is a **Wiener process** (Brownian motion)

- Discretizing this SDE using **Euler-Maruyama** gives:

$$X_{t+1} = X_t + \frac{\eta}{2}\nabla \log \pi(X_t) + \sqrt{\eta}\mathbf{z}, \quad \mathbf{z} \sim \mathcal{N}(\mathbf{0}, I)$$

which is **exactly a sample from this** $q(\mathbf{y} \mid \mathbf{x})$

Unadjusted Langevin Algorithm

- For MALA, we need to know $\nabla_x \log \tilde{\pi}(x)$ and $\tilde{\pi}(x)$
- Sometimes, we **do not have access to** $\tilde{\pi}$, but **only to** $\nabla_x \log \tilde{\pi}(x)$
  - e.g., **Score-based Generative Models** (**Diffusion Models**)
- Idea: Let's just **always accept** 😂
- This is called the **Unadjusted Langevin Algorithm (ULA)**
- **Not a MH algorithm anymore**, and the stationary distribution is $\pi_\eta \neq \pi$ ❗
  - It has **stationary bias** $\mathbb{E}_{\pi_\eta}[f] - \mathbb{E}_\pi[f]$ of order $O(\eta)$ in general (even as $T \to \infty$)

Hamiltonian Monte Carlo

Hamiltonian Monte Carlo – Motivation

- To explore π, we want to make “large steps”
    - while still staying in **high probability regions**
- Also, we wish to incorporate **momentum**
    - Don’t forget direction immediately
    - Physical analog: ball with **mass**

Hamiltonian Monte Carlo

- If $\mathbf{x} \in \mathbb{R}^d$, augment the state space by a **new momentum variable $\mathbf{p} \in \mathbb{R}^d$**
- Define the **Hamiltonian**

$$H(\mathbf{x}, \mathbf{p}) = \overbrace{-\log \tilde{\pi}(\mathbf{x})}^{E(\mathbf{x})} + \overbrace{\tfrac{1}{2}\mathbf{p}^\top M^{-1}\mathbf{p}}^{K(\mathbf{p})}.$$

  where M is the **mass matrix**.

- This defines a **new joint distribution**

$$\pi_{\text{aug}}(\mathbf{x}, \mathbf{p}) = Z_H^{-1} \exp(-H(\mathbf{x}, \mathbf{p})) = Z_H^{-1} \tilde{\pi}(\mathbf{x}) \exp(-\tfrac{1}{2}\mathbf{p}^\top M^{-1}\mathbf{p}).$$

  with **marginal** $\pi_{\text{aug}}(\mathbf{x}) = \int \pi_{\text{aug}}(\mathbf{x}, \mathbf{p})\, d\mathbf{p} = \pi(\mathbf{x})$

- HMC is again an instance of **Metropolis-Hastings**, but **sampling from** $\pi_{\text{aug}}(\mathbf{x}, \mathbf{p})$
  - Since we want to **sample from** π (the marginal), we **throw away the sampled $\mathbf{p}$**

Hamiltonian Monte Carlo (cont.)

- In a single MH proposal step, we will consider $\mathbf{x}, \mathbf{p}$ to change over "time" t
    - This is **not** time t in the Markov chain
- We want that the **Hamiltonian is conserved over time**:

$$\frac{dH(\mathbf{x}, \mathbf{p})}{dt} = \frac{\partial H}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial t} + \frac{\partial H}{\partial \mathbf{p}} \frac{\partial \mathbf{p}}{\partial t} \overset{!}{=} 0$$

- Which we enforce by letting $\mathbf{x}, \mathbf{p}$ evolve via **Hamiltonian dynamics:**

$$\frac{\partial \mathbf{x}}{\partial t} := \frac{\partial H}{\partial \mathbf{p}} \qquad \frac{\partial \mathbf{p}}{\partial t} := -\frac{\partial H}{\partial \mathbf{x}}$$

- Let $(\mathbf{y}, \mathbf{p}') = \Phi_\tau(\mathbf{x}, \mathbf{p})$ be a **ODE solver** that **integrates this ODE** over some time τ
    - If we made no numerical error, then $H(\mathbf{x}, \mathbf{p}) = H(\Phi_\tau(\mathbf{x}, \mathbf{p}))$ for any τ and $(\mathbf{x}, \mathbf{p})$
    - In practice, we set step size ε and number of steps L, where then $\tau = L\varepsilon$

HMC – Algorithm

Hamiltonian Monte Carlo

Initialize $X_1 \sim p(X_1)$ and repeat for $t = 1, \ldots, T-1$:

1. Given current state $X_t = \mathbf{x}$, sample $\mathbf{p} \sim \mathcal{N}(\mathbf{p}; \mathbf{0}, M)$ and **propose** (deterministically)

$$(\mathbf{y}, \mathbf{p}') = \Phi_\tau(\mathbf{x}, \mathbf{p}) \qquad \text{i.e., } q\big((\mathbf{y}, \mathbf{p}') \mid (\mathbf{x}, \mathbf{p})\big) = \delta_{\Phi_\tau(\mathbf{x},\mathbf{p})}(\mathbf{y}, \mathbf{p}')$$

2. **Compute acceptance probability:**

$$\begin{aligned}\alpha((\mathbf{x}, \mathbf{p}) \to (\mathbf{y}, \mathbf{p}')) &= \min\big(1,\ \exp\big(-H(\mathbf{y}, \mathbf{p}') + H(\mathbf{x}, \mathbf{p})\big)\big) \\ &= 1 \qquad \text{if ODE solver has no numerical error}\end{aligned}$$

3. **Accept or reject:**

$$\text{Sample } U \sim \mathrm{Uniform}(0,1), \qquad X_{t+1} = \begin{cases} \mathbf{y} & \text{if } U \leq \alpha \\ \mathbf{x} & \text{else} \end{cases}$$

Extensions to HMC

- **Issue:** L and ε are sometimes **hard to tune**
  - Problem specific, brittle hyperparameters

- **NUTS (No U-Turn Sampler)**: Variant of HMC which
  - **Auto-tunes** ε in a **warm-up phase**
  - Picks L **adaptively** per MH step (integration stops when trajectory want to “turn around”)

HMC Demo

`https://chi-feng.github.io/mcmc-demo/app.html`

Diagnosing MCMC

MCMC – What can go wrong?

- Recall that for a **stationary** Markov Chain, the **MCMC estimator**

$$\hat{\mathbb{E}}_T = \frac{1}{T}\sum_{t=1}^{T} f(X_t), \qquad X_t \sim \pi$$

has variance

$$\mathrm{var}\left[\hat{\mathbb{E}}_T\right] \approx \frac{\sigma^2_{\mathrm{as}}}{T}, \qquad \sigma^2_{\mathrm{as}} = \mathrm{var}_\pi[f(X)] + 2\sum_{k=1}^{\infty} \mathrm{cov}_\pi(f(X_0), f(X_k)).$$

- **Note:** This formula is only valid **after burn-in**

Diagnostics

When running MCMC at time t, we want to answer **two questions**:

1. Are we **in stationarity**, i.e., is $p_t \approx \pi$ **?**
2. How **large** is σ^2_{as}, i.e., how **inefficient is the chain** **?**

Burn-in Phase

- We suffer from **initialization bias** since $p_1 \neq \pi$
- Thus, discard the first B samples (**burn-in phase**)
- **Unfortunately, there is no good data-driven way to pick** B 😩
- **Note:** B only affects **bias** and is **not related** to σ_{as} ⚠️

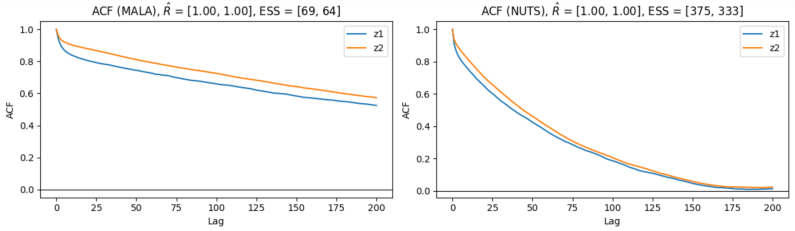
Autocorrelation Function (ACF)

- For a **stationary** chain, we can write

$$\rho_k = \frac{\text{cov}_\pi(f(X_0), f(X_k))}{\text{var}_\pi(f(X))}, \qquad \sigma^2_{\text{as}} = \text{var}_\pi[f(X)] \left(1 + 2\sum_{k=1}^{\infty} \rho_k\right)$$

  where ρ_k is the **autocorrelation** at lag k.

- We can plot an **estimate** $\hat{\rho}_k$ over k **after burn-in**:



- ACF diagnoses **within-chain dependence, not convergence to** π ⚠

Effective Sample Size (ESS)

- We would like to define **Effective Sample Size (ESS)** such that

$$\text{var}\left[\hat{\mathbb{E}}_T\right] = \frac{\text{var}_\pi[f(X)]}{\text{ESS}}$$

- ESS tells us **how many i.i.d. samples would give the same variance** !

- For a **stationary** chain, we thus have

$$\text{ESS} = \frac{\text{var}_\pi[f(X)]}{\sigma^2_{\text{as}}/T} = \frac{T}{1 + 2\sum_{k=1}^{\infty} \rho_k}$$

  which we can estimate from **samples post burn-in**

Gelman-Rubin $\hat{R}$

- We want to check if MCMC has **converged to “stationary” distribution**
  - By construction, we will **converge** to the **unique stationary distribution** π as $T \to \infty$
  - However, for finite T, we can get **stuck** in quasi-stationary distributions $\neq \pi$ **!**

- Run $M \geq 2$ **independent chains** and discard **burn-in**

- **Idea:** Compare **within-chain variance** W with **between-chain variance** B:

$$\hat{R} := \sqrt{\frac{(1 - \frac{1}{T})W + \frac{1}{T}B}{W}}$$

- If all chains converged to the same distribution, we should have $\hat{R} \approx 1$
  - Often, one checks if $\hat{R} \leq 1.01$

MCMC – Summary

- **Goal**: Sample from **complicated, high-dimensional distribution** π
  - Where typically only $\tilde{\pi}$ is known
- Almost all MCMC algorithms presented are **instances of Metropolis-Hastings**
  - Random-walk MH, Gibbs Sampling, MALA, HMC, NUTS
- In **high dimensions**, HMC/NUTS is the **go-to choice**
  - If you have **tractable conditionals**, Gibbs sampling may be a **good option**
- However, even well-tuned MCMC methods **can take a long time to converge**
  - This is called **mixing time**
- **Diagnosis** of MCMC is **hard in general**
  - Common tools: ACF, ESS, $\hat{R}$