Markov Chains

What is a Markov Chain?

Definition: A discrete-time process *Markov Chain* is a sequence of random variables $\{X_t\}_{t\geq 0}$ with the property that, given the present state, the future and past states are independent. Formally,

$$P(X_{t+1}|X_t, X_{t-1}, \ldots, X_0) = P(X_{t+1}|X_t).$$

The Markov Chain is **time-homogeneous** if the transition probabilities do not depend on time t:

$$\forall n \in N, \quad P(X_t = y|X_{t-1} = x) = P(X_{t+m} = y|X_{t+m-1} = x)$$

i.e. transition probabilities do not depend on t. The **key idea** is MCMC is to construct a Markov Chains such that x_t converges to a desired distribution π as $t \to \infty$ and

$$\frac{1}{n}\sum_{t=1}^{n} \phi(x_t) \to \mathbb{E}_{x\sim\pi}[\phi(x)] \quad \text{as } n \to \infty.$$

What kinds of conditions are required for this to hold?

Invariant / stationary distribution

A distribution π is called **invariant** (or **stationary**) for a Markov Chain with transition kernel P if

$$\pi(y) = \int \pi(x) P(x, y) dx.$$

Intuitively, if the chain starts with distribution π, it remains in distribution π at all future times.

Time-homogeneous is not needed for invariant distribution. But it is often easier to verify in that case.

Irreducible

A Markov Chain is called **irreducible** if it is possible to get to any state from any state. Formally, for any states x and y, there exists an integer $0 \leq n < \infty$ such that

$$P^n(x, y) > 0,$$

where $P^n(x, y)$ is the n-step transition probability from state x to state y.

Aperiodicity

A Markov Chain is called **aperiodic** if it does not get trapped in cycles with fixed periods. Ensures actual convergence instead of oscillation. Formally, for any state x, the greatest common divisor of the set of integers

$$\{n \geq 1 : P^n(x, x) > 0\}$$

is 1. **Note:** If all states have a non-zero probability of remaining in the same state, the chain is aperiodic.

Positive recurrence

A Markov Chain is called **positive recurrent** if, starting from any state, the expected return time to that state is finite. Formally, for any state x,

$$\mathbb{E}[T_x | X_0 = x] < \infty,$$

where T_x is the return time to state x. **Note:** Positive recurrence ensures that the chain does not wander off to infinity and has a well-defined long-term behavior.

More on recurrence

- **Recurrent**: A Markov Chain is called recurrent if, starting from any state, the probability of returning to that state is 1.
- **Positive recurrence:** A Markov Chain is called positive recurrent if it is recurrent and the expected return time to any state is finite, i.e. the chain returns quickly on average.
- **Transient**:
- **Null recurrence:** A Markov Chain is called null recurrent if it is recurrent but the expected return time to any state is infinite.

Rejection Sampling

why

Basic idea: Sample from instrumental proposal $q \neq \pi$; correct through rejection step to obtain a sample from π.
Given two densities π, q with $\pi(x) \leq Mq(x)$ for all x, and some $M > 0$, we can generate a sample from π by

What is Rejection Sampling?

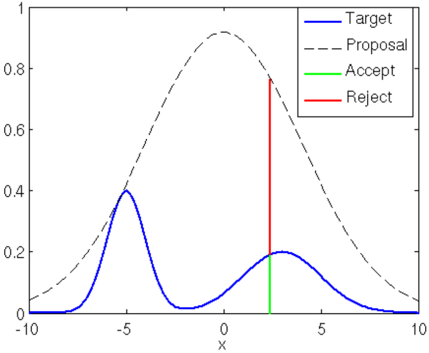
Intuition:

- Throw darts uniformly under $M \cdot q(x)$
- Keep only those under $\pi(x)$
- Kept points follow $\pi(x)$ exactly!

The Algorithm:

1. Sample $X \sim q(x)$
2. Sample $U \sim \text{Uniform}(0, 1)$
3. Accept if $U \leq \frac{\pi(X)}{M \cdot q(X)}$

Why it works: We're sampling uniformly from the area under $\pi(x)$!



Mathematical Foundation

Proposition

The distribution of accepted samples is exactly $\pi(x)$

Proof: We need to show $P(X \in A|X \text{ accepted}) = \pi(A)$. From the definition of conditional probability: $P(X \in A|X \text{ accepted}) = \frac{P(X\in A, X \text{ accepted})}{P(X \text{ accepted})}$.

$$\begin{aligned} P(X \in A, X \text{ accepted}) &= \int_X \int_0^1 \mathbb{I}_A(x) \cdot \mathbb{I}\left(u \leq \frac{\pi(x)}{Mq(x)}\right) q(x)\, du\, dx \\ &= \int_X \mathbb{I}_A(x) \cdot \frac{\pi(x)}{Mq(x)} \cdot q(x)\, dx \\ &= \int_X \mathbb{I}_A(x) \cdot \frac{\pi(x)}{M}\, dx = \frac{\pi(A)}{M} \end{aligned}$$

Similarly, $P(X \text{ accepted}) = \frac{1}{M}$. Therefore: $P(X \in A|X \text{ accepted}) = \frac{\pi(A)/M}{1/M} = \pi(A)$.

Does this work for un-normalised distributions?

Often we only know π and q up to some normalising constants; i.e.

$$\pi = \frac{\pi}{Z_\pi} \quad \text{and} \quad q = \frac{q}{Z_q}$$

where $\tilde{\pi}$, $\tilde{q}$ are known but Z_π, Z_q are unknown. You still need to be able to sample from $q(\cdot)$. If you can upper bound:

$$\frac{\tilde{\pi}(x)}{\tilde{q}(x)} \leq \tilde{M},$$

then using $\tilde{\pi}$, $\tilde{q}$ and $\tilde{M}$ in the algorithm is correct.
Indeed we have

$$\frac{\tilde{\pi}(x)}{\tilde{q}(x)} \leq \tilde{M} \iff \frac{\pi(x)}{q(x)} \leq \tilde{M} \cdot \frac{Z_q}{Z_\pi} \stackrel{\text{def}}{=} M$$

- forudsætninger og konstant M
- waiting time to accepted sample is geometric with mean M
- M storre end 1
- find M kan være svært
- kan være ineffektiv hvis M er stor
- dimensionalitet
- squeezing

Importance Sampling

What is Importance Sampling?

What?

- Monte Carlo technique for estimating $\mathbb{E}_\pi[\phi(X)]$
- Sample from proposal q instead of target π
- Reweight samples to correct bias

Why?

- Target π difficult to sample from
- Focus sampling in important regions
- Works with unnormalized distributions
- All samples are used (unlike rejection)

How? The key identity:

$$\begin{aligned}\mathbb{E}_\pi[\phi(X)] &= \int \phi(x)\pi(x)dx \\ &= \int \phi(x)\frac{\pi(x)}{q(x)}q(x)dx \\ &= \mathbb{E}_q[\phi(X)w(X)]\end{aligned}$$

Algorithm

1. Sample $X_1, \ldots, X_n \sim q$
2. Compute $w(X_i) = \pi(X_i)/q(X_i)$
3. Estimate: $\hat{I} = \frac{1}{n}\sum_{i=1}^n \phi(X_i)w(X_i)$

Key Properties and Unnormalized Distributions

Properties of IS Estimator:

- **Unbiased**: $\mathbb{E}_q[\hat{I}] = \mathbb{E}_\pi[\phi(X)]$
- **Consistent**: $\hat{I} \xrightarrow{n\to\infty} \mathbb{E}_\pi[\phi(X)]$ (LLN)
- **Variance**: $\text{Var}_q[\hat{I}] = \frac{1}{n}\text{Var}_q[\phi(X)w(X)]$

Requirements:

- $q(x) > 0$ whenever $\pi(x)\phi(x) \neq 0$
- $\mathbb{E}_q[|\phi(X)w(X)|] < \infty$

Unnormalized Distributions:

When $\pi(x) = \tilde{\pi}(x)/Z$ with unknown Z:

Self-Normalized IS

- Weights: $\tilde{w}(x) = \tilde{\pi}(x)/q(x)$
- Estimator:

$$\hat{I}_{SN} = \frac{\sum_{i=1}^n \phi(X_i)\tilde{w}(X_i)}{\sum_{i=1}^n \tilde{w}(X_i)}$$

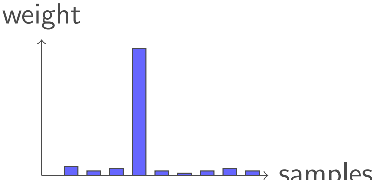
- Biased but consistent
- Bias: $O(1/n)$

Why Weight Distribution Matters

Weight Distribution Impact:

- High weight variance $\Rightarrow$ poor estimates
- Few samples dominate the sum
- Ideal case: all weights equal $(q = \pi)$
- $\text{Var}_q[w(X)]$ determines convergence

Example Weight Degeneracy:



One weight dominates

Effective Sample Size (ESS)

Definition

$$\text{ESS} = \frac{(\sum_{i=1}^{n} w_i)^2}{\sum_{i=1}^{n} w_i^2}$$

Interpretation:

- Number of "equivalent" samples from π
- Range: $1 \leq \text{ESS} \leq n$
- $\text{ESS} = n$ when all weights equal, $\text{ESS} = 1$ when one weight dominates

Why ESS Matters:

- $\text{ESS} \ll n$ indicates weight degeneracy
- Low ESS $\Rightarrow$ high variance
- Monitor ESS to diagnose problems
- Rule of thumb: $\text{ESS} > n/2$ is good

Choosing Good Proposals & Dimensional Scaling

Good Proposal Properties:

1. Heavier tails than π
2. Easy to sample from
3. Similar shape to $\pi|\phi|$
4. Covers support of π
5. Minimizes $\mathrm{Var}_q[\phi(X)w(X)]$

Common Choices:

- Student-t for Gaussian targets
- Mixture distributions
- Previous MCMC output
- Laplace approximation

Curse of Dimensionality:

Gaussian Example

For $\pi = \mathcal{N}(0, I_d)$, $q = \mathcal{N}(0, \sigma^2 I_d)$:

$$\mathrm{Var}_q[w(X)] = \left(\frac{\sigma^4}{2\sigma^2 - 1}\right)^{d/2} - 1$$

Numerical Example:

d	σ	$\mathrm{Var}_q[w(X)]$
10	1.2	≈ 5.6
50	1.2	≈ 850
100	1.2	$\approx \mathbf{1.8 \times 10^4}$

Importance Sampling vs. Rejection Sampling

Aspect	**Importance Sampling**	**Rejection Sampling**
Sample usage	All samples (weighted)	Some samples rejected
Efficiency	Depends on weight variance	Depends on acceptance rate
High dimensions	Poor (variance explodes)	Very poor (accept rate → 0)
Proposal req.	$q > 0$ where $\pi\phi \neq 0$	Need $Mq \geq \pi$ everywhere
Output	Weighted samples	Exact samples from π
Normalizing const.	Not required	Required (for bound M)
Bias	Unbiased (or consistent)	Unbiased (exact)
Failure mode	High variance	No samples produced

Key Insight

Both methods suffer from curse of dimensionality, but:

- IS degrades gracefully - still provides estimates (with high variance)

Rejection Sampling

Metropolis-Hastings Algorithm

- Target distribution on $\mathbb{X} = \mathbb{R}^d$ of density $\pi(x)$.
- Proposal distribution: for any $x, x' \in \mathbb{X}$, we have $q(x'|x) \geq 0$ and $\int_{\mathbb{X}} q(x'|x)\, dx' = 1$.

Algorithm

1. Starting with $X^{(1)}$, for $t = 2, 3, \ldots$
2. Sample $X^\star \sim q(\cdot|X^{(t-1)})$.
3. Compute $\alpha(X^\star|X^{(t-1)}) = \min\left(1, \frac{\pi(X^\star)q(X^{(t-1)}|X^\star)}{\pi(X^{(t-1)})q(X^\star|X^{(t-1)})}\right)$.
4. Sample $U \sim \mathcal{U}_{[0,1]}$. If $U \leq \alpha(X^\star|X^{(t-1)})$, set $X^{(t)} = X^\star$, otherwise set $X^{(t)} = X^{(t-1)}$.

Metropolis-Hastings Algorithm

- The proposal distribution q can be chosen quite freely, but it should be easy to sample from and to evaluate.
- The acceptance probability $\alpha(x'|x)$ ensures that the chain has π as its stationary distribution.
- If q is symmetric, i.e., $q(x'|x) = q(x|x')$, then the acceptance probability simplifies to $\alpha(x'|x) = \min\left(1, \frac{\pi(x')}{\pi(x)}\right)$. This is also known as Metropolist Random Walk.
- If q does not depend on the current state, i.e., $q(x'|x) = q(x')$, then the algorithm reduces to the Independent Metropolis-Hastings algorithm.
- The choice of q affects the efficiency of the algorithm. A poorly chosen q can lead to slow mixing and high autocorrelation in the samples.

Role of $\alpha(X^\star|X^{(t-1)})$

- The acceptance probability $\alpha(X^\star|X^{(t-1)})$ is crucial for ensuring that the Markov chain has the desired stationary distribution π.
- It corrects for the discrepancy between the proposal distribution q and the target distribution π.
- If the proposed state $X^\star$ has a higher density under π than the current state $X^{(t-1)}$, it is always accepted ($\alpha = 1$).
- If $X^\star$ has a lower density, it may still be accepted with a probability proportional to the ratio of densities, allowing exploration of the state space.
- This mechanism helps to avoid getting stuck in local modes and promotes better mixing of the chain.

Role of $\alpha(X^\star|X^{(t-1)})$

- If $\pi(x^*) > \pi(x)$, then the proposed state has higher probability than current state which favors acceptance
- If $\pi(x^*) < \pi(x)$, then the proposed state has lower probability than current state which favors rejection
- Intuition: want to spend time in high-probability regions, so moves towards them should be favored
- Reverse proposal ratio: how easy is it to propose reverse move compared to forward move?
- Intuition: if proposal mechanism makes it easy to reacg certain states, we need to be more selective accepting such moves. otherwise the chain will be biased towards such easy-to-propos-to regions rather than high probability regions

Transition Kernel

Lemma

The kernel of the Metropolis–Hastings algorithm is given by

$$K(x, y) = \alpha(y \mid x) q(y \mid x) + (1 - a(x))\delta_x(y).$$

Proof: We have

$$\begin{aligned}
K(x, y) &= \int q(x^* \mid x)\{\alpha(x^* \mid x)\delta_{x^*}(y) + (1 - \alpha(x^* \mid x))\delta_x(y)\}\, dx^* \\
&= q(y \mid x)\alpha(y \mid x) + \left\{\int q(x^* \mid x)(1 - \alpha(x^* \mid x))\, dx^*\right\}\delta_x(y) \\
&= q(y \mid x)\alpha(y \mid x) + \left\{1 - \int q(x^* \mid x)\alpha(x^* \mid x)\, dx^*\right\}\delta_x(y) \\
&= q(y \mid x)\alpha(y \mid x) + \{1 - a(x)\}\delta_x(y).
\end{aligned}$$

Reversibility

Proposition

The Metropolis–Hastings kernel K is π-reversible and thus admits π as invariant distribution.

Proof: For any $x, y \in \mathbb{X}$, with $x \neq y$

$$\begin{aligned}
\pi(x)K(x,y) &= \pi(x)q(y \mid x)\alpha(y \mid x) \\
&= \pi(x)q(y \mid x)\left(1 \wedge \frac{\pi(y)q(x \mid y)}{\pi(x)q(y \mid x)}\right) \\
&= (\pi(x)q(y \mid x) \wedge \pi(y)q(x \mid y)) \\
&= \pi(y)q(x \mid y)\left(\frac{\pi(x)q(y \mid x)}{\pi(y)q(x \mid y)} \wedge 1\right) = \pi(y)K(y,x).
\end{aligned}$$

If $x = y$, then obviously $\pi(x)K(x,y) = \pi(y)K(y,x)$.

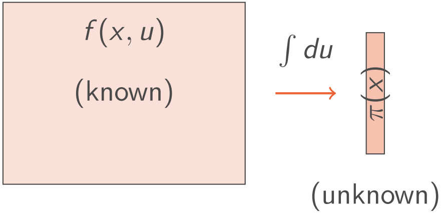
Pseudo-marginal MCMC

The Challenge: Intractable Marginals

The Problem:

- Target: $\pi(x) = \int f(x, u)du$
- $f(x, u)$ is known (complete data)
- Integral is intractable
- Standard MCMC requires exact $\pi(x)$

Key Insight: We can estimate $\pi(x)$ unbiasedly!



The Pseudo-marginal Solution

Key Prerequisites

For pseudo-marginal MCMC to be applicable, we need:

1. Ability to **evaluate** $f(x, u)$ pointwise for any (x, u)
2. Ability to **sample** from an importance distribution $q_x(\cdot)$ over the u-space
3. The importance distribution must have appropriate support: $q_x(u) > 0$ whenever $f(x, u) > 0$

Importance Sampling Estimator:

$$\hat{\pi}(x) = \frac{1}{N} \sum_{i=1}^{N} \frac{f(x, U_i)}{q_x(U_i)}, \quad U_i \sim q_x(\cdot)$$

Key Property: $\mathbb{E}[\hat{\pi}(x)] = \pi(x)$ (unbiased!)

The Magic: Replace π with $\hat{\pi}$ in MH ratio!

$$\alpha = \min\left\{1, \frac{\hat{\pi}(y)q(y, x)}{\hat{\pi}(x)q(x, y)}\right\}$$

Result: Still targets correct $\pi(x)$!

Why It Works: Extended Target

One can think of estimator (the “pseudo-marginal”) as the product of the true target and a random variable:

$$\hat{\pi}(x) = \pi(x)Z_x$$

where Z_x satifies:

1. is non-negative: $Z_x \geq 0$,
2. has density $g_x()$: $\int_0^\infty g_x(z)dz = 1$
3. has expectation 1: $\mathbb{E}[Z_x] = \int_0^\infty z g_x(z)dz = 1$.

Why It Works: Extended Target

Extended Target Construction:

$$\bar{\pi}(x, z) = \pi(x) \cdot z \cdot g_x(z)$$

where $g_x(z)$ is the density of Z_x

Key Property:

$$\int \bar{\pi}(x, z) dz = \pi(x)$$

Now apply Metropolis–Hastings with proposal

$$\bar{q}((x, z), (y, w)) := q(x, y) \cdot g_y(w).$$

Intuition:

- Run exact MCMC on (x, z) space
- Marginal in x gives correct target
- z represents the "noise" in estimates

Equivalence to MH on Extended Space

Theorem (Equivalence)

Metropolis-Hastings on the extended target $\bar{\pi}$ with proposal $\bar{q}$ is equivalent to the pseudo-marginal algorithm using estimates $\hat{\pi}$.

Proof Sketch: The MH acceptance ratio on the extended space is:

$$\begin{aligned}
\alpha_{ext} &= \min\left\{1, \frac{\bar{\pi}(y,w)\bar{q}((y,w),(x,z))}{\bar{\pi}(x,z)\bar{q}((x,z),(y,w))}\right\} \\
&= \min\left\{1, \frac{\pi(y)\cdot w \cdot g_y(w)\cdot q(y,x)\cdot g_x(z)}{\pi(x)\cdot z\cdot g_x(z)\cdot q(x,y)\cdot g_y(w)}\right\} \\
&= \min\left\{1, \frac{\pi(y)\cdot w\cdot q(y,x)}{\pi(x)\cdot z\cdot q(x,y)}\right\} = \min\left\{1, \frac{\hat{\pi}(y)q(y,x)}{\hat{\pi}(x)q(x,y)}\right\} = \alpha_{pm}
\end{aligned}$$

In the last step, we used $\hat{\pi}(x) = \pi(x)z$ and $\hat{\pi}(y) = \pi(y)w$, which is exactly the pseudo-marginal acceptance probability.

Pseudo-marginal MCMC Algorithm

Given $(X^{(t-1)}, \hat{\pi}^{(t-1)})$:

1. **Propose:** $Y \sim q(X^{(t-1)}, \cdot)$
2. **Estimate:**
   - Sample $U_i \sim q_Y(\cdot)$
   - $\hat{\pi}(Y) = \frac{1}{N} \sum_i \frac{f(Y, U_i)}{q_Y(U_i)}$
3. **Accept with probability:**

$$\alpha = \min \left\{ 1, \frac{\hat{\pi}(Y) q(Y, X^{(t-1)})}{\hat{\pi}^{(t-1)} q(X^{(t-1)}, Y)} \right\}$$

4. **Update:**
   - If accept: $(X^{(t)}, \hat{\pi}^{(t)}) = (Y, \hat{\pi}(Y))$
   - Else: $(X^{(t)}, \hat{\pi}^{(t)}) = (X^{(t-1)}, \hat{\pi}^{(t-1)})$

Critical Points:

- Store estimates with states! In the next iteration, use the stored $\hat{\pi}(X^{(t-1)})$.
- Fresh randomness for each proposal. Every time you propose a new state Y, you must generate a completely new, independent estimate $\hat{\pi}(Y)$ using fresh random samples.
- Works with *any* MH proposal q

Slice Sampling

Slice Sampling

What is Slice Sampling?

A "black-box" auxiliary variable Markov Chain Monte Carlo (MCMC) method that avoids the need to tune hyperparameters. Introduced by Neal (2003).

The idea of slice sampling. Suppose we wish to sample from a distribution for a variable, x, taking values in some subset of R^n , whose density is proportional to some function $f(x)$. We can do this by sampling uniformly from the (n $+$ 1)-dimensional region that lies under the plot of $f(x)$.

Introduction

This idea can be formalized by introducing an auxiliary real variable, y, and defining a joint distribution over x and y that is uniform over the region $U = \{(x, y) : 0 < y < f(x)\}$ below the curve or surface defined by $f(x)$. That is, the joint density for (x, y) is

$$p(x, y) = \frac{1}{Z} \begin{cases} 1, & \text{if } 0 < y < f(x) \\ 0, & \text{otherwise} \end{cases}$$

where $Z = \int f(x)dx$. The marginal density for x is then

$$p(x) = \int_0^{f(x)} \frac{1}{Z} dy = \frac{f(x)}{Z}$$

which is the desired distribution. Thus, if we can sample from the joint distribution $p(x, y)$, we can obtain samples from the marginal distribution $p(x)$.

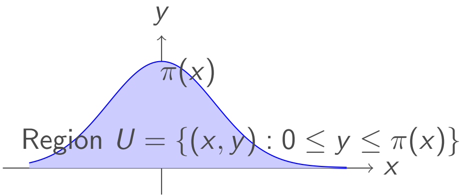
Intuition Behind Slice Sampling

Step 1: Vertical Slice

- Given current position x
- Sample height $y \sim \text{Uniform}(0, \pi(x))$
- Defines horizontal "slice" at height y

Step 2: Horizontal Slice

- Sample new x uniformly from slice
- $S = \{x : \pi(x) \geq y\}$
- Gives new sample from $\pi(x)$



Key Insight: By alternating between sampling $y|x$ and $x|y$, we create a Markov chain that explores the space under $\pi(x)$ uniformly, with marginal distribution for x being exactly $\pi(x)$

The Slice Sampling Algorithm

Basic Algorithm

Given: Current state x_t, target distribution $\pi(x)$

Algorithm

1. **Sample auxiliary variable:** Draw $y \sim \text{Uniform}(0, \pi(x_t))$
2. **Find the slice:** Identify $S = \{x : \pi(x) \geq y\}$
3. **Sample from the slice:** Draw $x_{t+1} \sim \text{Uniform}(S)$

The Challenge: Finding and Sampling from S

In practice, finding $S = \{x : \pi(x) \geq y\}$ can be difficult!

Key Idea: Leave Distribution Invariant

How to sample from S

The Stepping Out Procedure

1. **Create initial interval:**
2. $L = x_t - w \cdot U$, $R = L + w$, where $U \sim \text{Uniform}(0, 1)$
3. **Step out left:**
4. While $\pi(L) \geq y$: $L = L - w$
5. **Step out right:**
6. While $\pi(R) \geq y$: $R = R + w$

The Shrinking Procedure

1. **Sample and shrink:**
2. Loop: $x' \sim \text{Uniform}(L, R)$
3. If $\pi(x') \geq y$: accept $x_{t+1} = x'$
4. Else: shrink $[L, R]$ by setting $L = x'$ or $R = x'$

Adaptive Nature: The algorithm automatically adapts to the local scale of $\pi(x)$. Wide regions are explored with large steps, narrow regions with small steps. Alternative procedures exist for sampling from S e.g., doubling.

Why Slice Sampling Converges

Formal Convergence Properties

1. Detailed Balance

Let $T(x'|x)$ be the transition kernel. We need: $\pi(x) \cdot T(x'|x) = \pi(x') \cdot T(x|x')$

Proof sketch:

- Given x, sample $y \sim \text{Uniform}(0, \pi(x))$
- Probability density of moving from x to x':

$$T(x'|x) = \int_0^{\min(\pi(x),\pi(x'))} \frac{1}{\pi(x)} \cdot \frac{1}{|S_y|} dy$$

  where $|S_y|$ is the length of slice $\{z : \pi(z) \geq y\}$
- This is symmetric: $T(x'|x) = T(x|x') \Rightarrow$ detailed balance holds

Convergence - Continued

2. Irreducibility

For any x, x' where $\pi(x) > 0$ and $\pi(x') > 0$:

$$P(x \to x') \geq \int_0^{\min(\pi(x),\pi(x'))} \frac{1}{\pi(x)} \cdot P(x' \text{ sampled from } S_y) dy > 0$$

3. Aperiodicity

$P(x \to x) > 0$ (can stay at current state) $\Rightarrow$ period $= 1$

Ergodic Theorem

Since the chain is irreducible, aperiodic, with stationary distribution $\pi(x)$:

$$\lim_{n\to\infty} \|P(X_n \in \cdot | X_0 = x_0) - \pi(\cdot)\|_{TV} = 0$$

Various topics

- How to choose initial width w?
- Extensions to Multivariate Slice Sampling. Coordinate-wise: Apply one-dimensional slice sampling to each x_i in turn. (Gibbs sampling)
- Elliptical Slice Sampling for Gaussian priors (Murray et al., 2010)

MALA and Barker's Proposal: Gradient-Based MCMC Methods

- Background: From RWM to gradient-based methods
- Langevin dynamics and discretization
- Metropolis-Adjusted Langevin Algorithm (MALA)
- Optimal scaling theory
- Barker's Proposal: An alternative approach
- Comparison and practical considerations

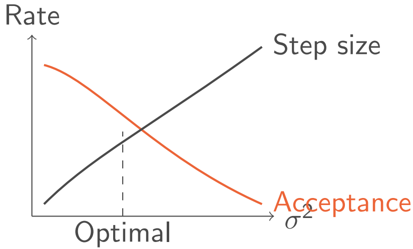
Random Walk Metropolis: The Challenge

Random Walk Metropolis (RWM):

$$q^* = q + \sigma W, \quad W \sim N(0, I_d)$$

Fundamental Trade-off:

- Large σ: Low acceptance
- Small σ: Slow exploration
- Optimal: $\sigma = \mathcal{O}(d^{-1})$



Problem: In high dimensions, RWM becomes inefficient

- Optimal acceptance rate: 0.234
- Curse of dimensionality: step size $\propto 1/d$

From Langevin Diffusion to MALA

Continuous Langevin Diffusion:

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

- Has π as stationary distribution
- Gradient provides drift toward high-probability regions

Euler-Maruyama Discretization (ULA):

$$X^{(t)} = X^{(t-1)} + \frac{\epsilon}{2}\nabla \log \pi(X^{(t-1)}) + \sqrt{\epsilon}W$$

Problem π is **not** the invariant distribution of ULA!
Solution: Add Metropolis-Hastings correction ⇒ MALA

Metropolis-Adjusted Langevin Algorithm

Algorithm 1 MALA

```
Input: Initial $X^{(0)}$, step size $\epsilon$, target $\pi$, proposal $q$
for $t = 1, 2, \dots$ do
    Propose: $X^* = X^{(t-1)} + \frac{\epsilon}{2}\nabla \log \pi(X^{(t-1)}) + \sqrt{\epsilon}W$
    Compute acceptance ratio:
        $\alpha = \min\left\{1, \frac{\pi(X^*)q(X^{(t-1)}|X^*)}{\pi(X^{(t-1)})q(X^*|X^{(t-1)})}\right\}$
    Accept $X^{(t)} = X^*$ with probability $\alpha$, else $X^{(t)} = X^{(t-1)}$
end for
```

Optimal Scaling Theory

Maximizing Expected Squared Jump Distance (ESJD)

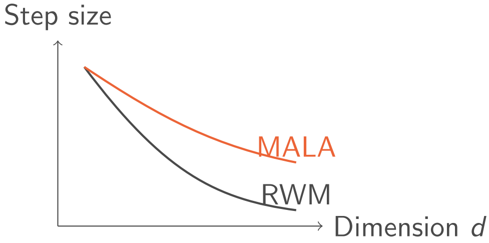
$$\mathbb{E}\left[\|X^{(t+1)} - X^{(t)}\|^2\right]$$

Dimension Scaling:

- RWM: $\sigma = \mathcal{O}(d^{-1})$
- MALA: $\sigma = \mathcal{O}(d^{-1/3})$

Optimal Acceptance:

- RWM: 0.234
- MALA: 0.574



Implication: MALA maintains larger step sizes in high dimensions

- Better exploration efficiency
- Faster convergence to target distribution
- **Catch** - requires gradient computation

Local-Balanced Proposals

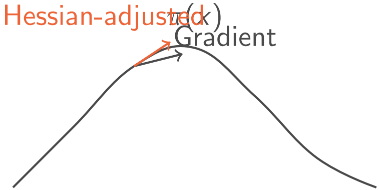
General Framework: Use local information about π

First-order (MALA):

$$X^* = X^{(t-1)} + \frac{\epsilon}{2}\nabla \log \pi(X^{(t-1)}) + \sqrt{\epsilon}W$$

Second-order:

$$X^* = X^{(t-1)} + \frac{\epsilon}{2}[\nabla^2 \log \pi(X^{(t-1)})]^{-1}\nabla \log \pi(X^{(t-1)}) + \sqrt{\epsilon}W$$



Higher-order methods better approximate local geometry

Barker's Proposal: An Alternative Approach

Key Idea: Use gradient to stochastically bias proposal direction

Proposal Density: $Q_B(x, dy) = \frac{2}{1+e^{-\nabla \log \pi(x)^T (y-x)}} K(x, dy)$
where $K(x, dy)$ is a base kernel (e.g., Gaussian)

Algorithm 2 1D case with Gaussian kernel

```
  Sample $Z \sim N(0, \sigma^2)$
  Calculate $p(x, z) = 1/(1 + \exp(-Z^T \nabla \log \pi(x)))$:
  Set $b(x, z) = 1$ with probability $p(x, z)$, else $b(x, z) = -1$
  Propose $Y = x + b(x, z)Z$
  Apply Metropolis-Hastings acceptance
```

MALA vs Barker's Proposal

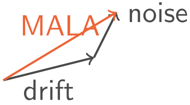
Both use gradient information, but differently

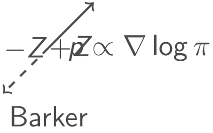
MALA:

- Deterministic drift
- $X^* = X + \frac{\epsilon}{2}\nabla \log \pi + \text{noise}$
- Gradient always adds to proposal
- Well-studied optimal scaling
- Proven efficiency in high dimensions

Barker:

- Stochastic direction choice
- Probability depends on gradient
- May flip proposal direction
- More recent theoretical development
- Potentially better for certain targets





Summary and Practical Considerations

Key Takeaways:

- Gradient information dramatically improves MCMC efficiency
- MALA: Proven workhorse with $O(d^{-1/3})$ scaling
- Barker: Promising alternative with different mixing properties
- Both methods correct discretization bias via Metropolis step

When to use which?

Choose MALA when:

- High-dimensional problems
- Gradients are cheap
- Well-conditioned targets
- Need proven reliability

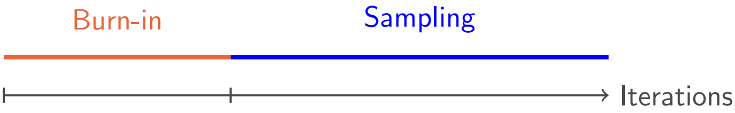
Consider Barker when:

- Exploring alternatives
- Specific target structure
- Research applications
- Robustness needed

Both methods: Major improvements over RWM!

The Convergence Challenge in MCMC

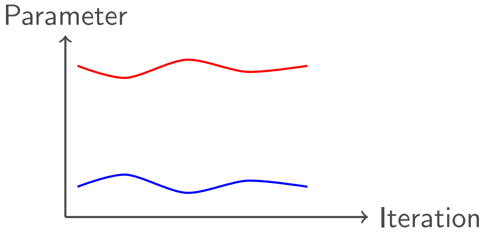
- **Ideal goal**: Assess whether MCMC chains have converged
- **Fundamental problem**:
  - In general, impossible to know for sure that there is no problem
  - But we can sometimes know for sure that there *is* a problem
- **Two phases of MCMC** [2]:
  - Transient phase (burn-in): mixing time
  - Stationary phase: Monte Carlo estimation



Why Convergence Matters

Non-converged chains:

- Biased estimates
- Incorrect uncertainty quantification
- Missing important modes
- Unreliable inference



Chains in different regions

Key Question

How can we diagnose whether our MCMC chains have converged to the target distribution?

The Intuition Behind Gelman-Rubin

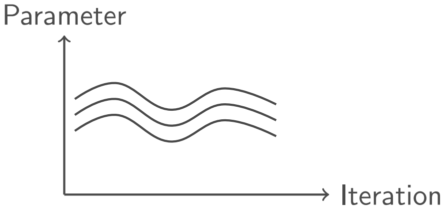
Core Idea

If MCMC chains have converged to the target distribution, then:

- Multiple chains started from different points should look similar
- Between-chain variance ≈ Within-chain variance

Compare two sources of variance:

1. Within-chain variance (W)
   How much each chain varies
2. Between-chain variance (B)
   How different chains are from each other



Converged chains overlap

Mathematical Foundation

Consider M chains, each of length T:

Variance Decomposition

Total sum of squares = Inter-group + Intra-group

$$\sum_{m=1}^{M}\sum_{t=1}^{T}(X_{m,t}-\bar{X}_{..})^2 = \sum_{m=1}^{M}\sum_{t=1}^{T}(\bar{X}_{m}-\bar{X}_{..})^2 + \sum_{m=1}^{M}\sum_{t=1}^{T}(X_{m,t}-\bar{X}_{m})^2$$

- Intra-group = Within-chain variance (W)
- Inter-group = Between-chain variance (B)

Key insight: After convergence, both estimate the same target variance!

The Gelman-Rubin Statistic

Definition

$$\hat{R} = \sqrt{\frac{\hat{V}}{W}}$$

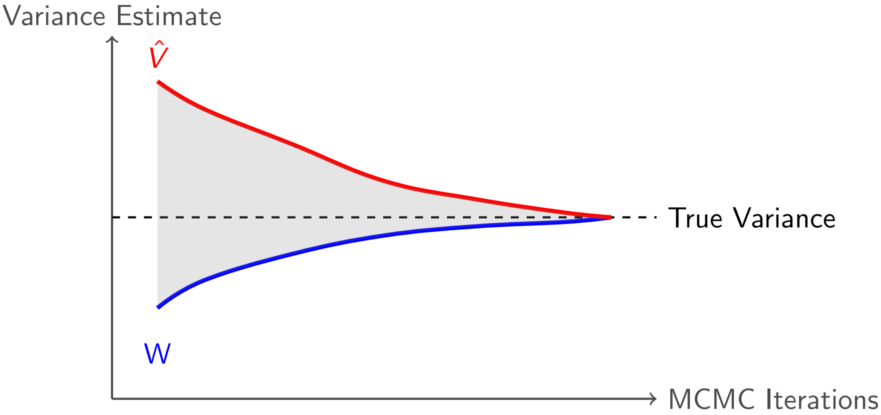
Where:

$$W = \frac{1}{M}\sum_{m=1}^{M} s_m^2 \quad \text{(average within-chain variance)}$$

$$B = \frac{T}{M-1}\sum_{m=1}^{M}(\bar{X}_m - \bar{X}_{\cdot\cdot})^2 \quad \text{(between-chain variance)}$$

$$\hat{V} = \frac{T-1}{T}W + \frac{1}{T}B \quad \text{(pooled variance estimate)}$$

Why It Works: The Variance Sandwich



- Initially: $W <$ True Variance $< \hat{V}$
- As chains converge: Both W and $\hat{V} \to$ True Variance
- Therefore: $\hat{R} = \sqrt{\hat{V}/W} \to 1$

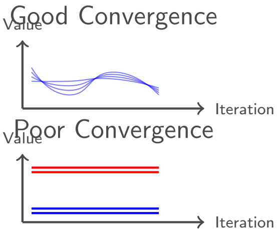
Example: Detecting Convergence Issues

Good convergence:

```
# Chains sampling from same distribution
chains_good = np.random.normal(0, 1, (4, 1000))
R_good = gelman_rubin(chains_good)
print(f"R-hat␣=␣{R_good:.3f}")
# Output: R-hat = 1.002
```

Poor convergence:

```
# Chains stuck in different modes
chains_bad = np.array([
    np.random.normal(-5, 0.5, 1000),
    np.random.normal(-5, 0.5, 1000),
    np.random.normal(5, 0.5, 1000),
    np.random.normal(5, 0.5, 1000)
])
R_bad = gelman_rubin(chains_bad)
print(f"R-hat␣=␣{R_bad:.3f}")
# Output: R-hat = 3.764
```



Evolution of Convergence Thresholds

Historical Development

- **1992**: Gelman & Rubin propose the diagnostic
- **2004**: Gelman recommends $\hat{R} < 1.1$
- **2021**: Vehtari et al. recommend $\hat{R} < 1.01$

Why the stricter threshold?

- More computing power available
- Better understanding of convergence
- Need for more reliable inference
- Connection to effective sample size

$\hat{R}$ threshold	ESS per chain
1.1	≈ 5
1.05	≈ 20
1.01	≈ 50

Connection to Effective Sample Size

Key Approximation (Vats & Knudson, 2021)

$$\hat{R} \approx \sqrt{1 + \frac{M}{\text{ESS}}}$$

Where:

- M = number of chains
- ESS = effective sample size (accounting for autocorrelation)

Implications:

- $\hat{R} = 1.1 \Rightarrow \text{ESS} \approx 5M$ (5 independent samples per chain)
- $\hat{R} = 1.01 \Rightarrow \text{ESS} \approx 50M$ (50 independent samples per chain)

5 effective samples per chain is too small for reliable inference!

Major Weaknesses of Gelman-Rubin

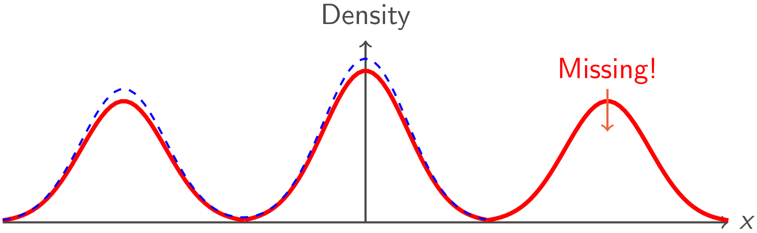
1. **Cannot detect if all modes are found**
   - Only checks if chains agree with each other
   - All chains might miss the same modes
2. **Sensitive to initialization**
   - Chains starting in the same wrong place
3. **Struggles with metastable states**
   - Chains get stuck but occasionally jump
   - Similar statistics but poor mixing
4. **Poor for heavy-tailed distributions**
   - Variance might not exist or be unstable

Remember

$\hat{R} < 1.01$ is necessary but not sufficient for convergence!

Example: Missing Modes

True distribution: Mixture of 3 Gaussians



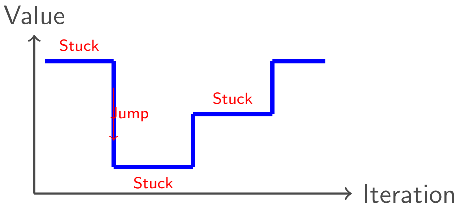
Chains sample only 2 modes

Result: $\hat{R} < 1.01$ but completely wrong posterior!
All chains agree because they all miss the same mode.

Example: Metastable States

Pathological behavior:

- Chains get "stuck" for long periods
- Occasionally jump to other regions
- All chains show same behavior
- $\hat{R} \approx 1$ despite poor mixing!



Detection:

- Very high autocorrelation
- Low effective sample size
- Visual inspection of trace plots

Despite poor mixing:

- Similar means across chains
- Similar variances
- $\hat{R} \approx 1$

Comprehensive Convergence Assessment

Use Multiple Diagnostics

1. **Gelman-Rubin statistic**: $\hat{R} < 1.01$
2. **Effective Sample Size**: ESS $>$ 400 (minimum)
3. **Trace plots**: Visual inspection
4. **Autocorrelation**: Check mixing quality
5. **Geweke test**: Compare chain beginning and end

Best Practices:

- Use at least 4 chains (preferably more)
- Initialize chains from overdispersed starting points
- Run chains longer than you think necessary
- Use rank-normalized $\hat{R}$ (more robust)
- Check both bulk and tail $\hat{R}$

Modern Extensions

Rank-Normalized $\hat{R}$ (Vehtari et al., 2021)

- Transform samples to ranks (more robust to outliers)
- Split chains in half (detect within-chain problems)
- Separate bulk and tail diagnostics

Bulk-$\hat{R}$:

- Convergence of center
- Mean, median
- Usually converges faster

Tail-$\hat{R}$:

- Convergence of extremes
- 5%, 95% quantiles
- Needs more samples

Modern tools (Stan, ArviZ) implement these improvements

Summary Checklist

MCMC Convergence Checklist

1. Run at least 4 chains with dispersed starts
2. Check $\hat{R} < 1.01$ for all parameters
3. Verify ESS > 400 (bulk and tail)
4. Examine trace plots visually
5. Check autocorrelation is low
6. Run sensitivity analysis with different seeds
7. Compare results from different samplers if possible

Remember:

No single diagnostic is perfect

Key Takeaways

1. **Gelman-Rubin compares within vs between chain variance**
   - Elegant idea: converged chains should agree
2. **Modern threshold is** $\hat{R} < 1.01$
   - Old threshold (1.1) gives only 5 effective samples
   - New threshold ensures 50 effective samples
3. $\hat{R}$ **has important limitations**
   - Can miss modes
   - Fooled by metastable states
   - Necessary but not sufficient
4. **Always use multiple diagnostics**
   - ESS, trace plots, autocorrelation
   - Visual inspection remains crucial

Good MCMC diagnostics = Reliable scientific inference

References

- Gelman, A. and Rubin, D.B. (1992). Inference from iterative simulation using multiple sequences. *Statistical Science*, 7(4), 457-472.
- Gelman, A., et al. (2004). *Bayesian Data Analysis* (2nd ed.). Chapman & Hall/CRC.
- Vehtari, A., Gelman, A., Simpson, D., Carpenter, B., and Bürkner, P.C. (2021). Rank-normalization, folding, and localization: An improved $\hat{R}$ for assessing convergence of MCMC. *Bayesian Analysis*, 16(2), 667-718.
- Vats, D. and Knudson, C. (2021). Revisiting the Gelman-Rubin diagnostic. *Statistical Science*, 36(4), 518-529.
- Brooks, S.P. and Gelman, A. (1998). General methods for monitoring convergence of iterative simulations. *Journal of Computational and Graphical Statistics*, 7(4), 434-455.

Thank You!

Questions?

Parallel Tempering MCMC

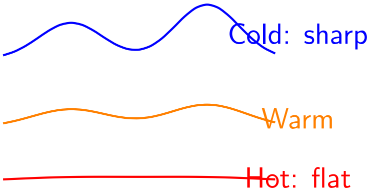
The Problem and Solution

The Problem:

- Standard MCMC gets stuck in local modes
- Can't explore separated peaks
- Dilemma: small steps (stuck) vs. large steps (rejected)

The Solution: Temperature Ladder

- Run N chains targeting π^{γ_n}
- $0 < \gamma_1 < \cdots < \gamma_N = 1$
- Hot ($\gamma \approx 0$): Explores freely
- Cold ($\gamma = 1$): Exploits peaks



Key Insight Different temperatures see the same distribution differently - hot chains explore, cold chains exploit

The Temperature Mechanism

Key Idea: Tempered Distributions

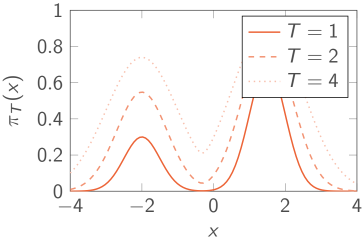
Define a family of distributions indexed by inverse temperature $0 < \gamma_1 < \gamma_2 < \ldots < \gamma_N = 1$:

$$\pi_{\gamma_n}(x) \propto \pi(x)^{\gamma_n}$$

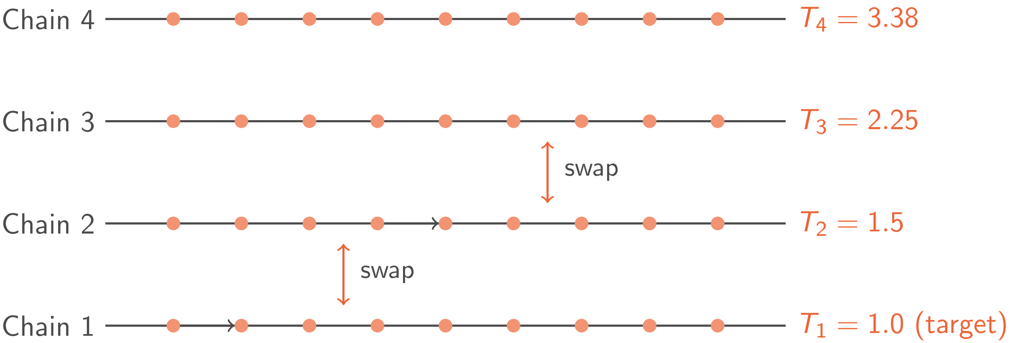
where $n = 1, \ldots, N$ and $\pi(x)$ is our target distribution.

Properties:

- $\gamma_N = 1$: Original target distribution
- $\gamma_N \approx 0$: Uniform, explores broadly



Parallel Chains Architecture



Two Types of Moves

1. **Within-chain updates**: Standard MCMC at each temperature
2. **Between-chain swaps**: Exchange states between adjacent temperatures

Parallel Tempering Algorithm Prerquisites

- **Target Distribution**: $\pi(x)$
- **Proposal Distribution**: For each tempered chain $q(x'|x)$ - could potentially depend on temperature
- **Initialization**: $x_n^{(0)}$ for $n = 1, \dots, N$
- **Standard MCMC Step**: Any MCMC kernel (e.g., RWM, MALA)
- **Number of Chains**: N
- **Number of Samples per Chain**: T
- **Temperature Schedule**: $\{\gamma_n\}_{n=1}^N$ with $\gamma_N = 1$

Parallel Tempering Algorithm

Algorithm 3 Parallel Tempering MCMC

```
for $t = 1$ to $T$ do
    for all $n \in \{1, \ldots, N\}$ in parallel do
        Sample $x_n^{(t)}$ using a standard MCMC step targeting $\pi^{\gamma_n}$
    end for
    $k \sim \text{Uniform}\{1, \ldots, N-1\}$
    $\alpha_{\text{swap}} = \min\left\{1, \left(\frac{\pi(x_{k+1}^{(t)})}{\pi(x_k^{(t)})}\right)^{\gamma_k - \gamma_{k+1}}\right\}$
    Swap $(x_k^{(t)}, x_{k+1}^{(t)})$ with probability $\alpha_{\text{swap}}$
end for
return $\{x_N^{(t)}\}_{t=1}^{T}$
```

Swap Acceptance Ratio Derivation

Propose swap: Exchange states $x_{k_1} \leftrightarrow x_{k_2}$ between chains k_1 and k_2

$$
\begin{aligned}
\alpha_{\text{swap}} = \frac{\text{Prob of proposed state}}{\text{Prob of current state}} &= \min\left\{1, \frac{\pi^{\gamma_{k_1}}(x_{k_2}) \cdot \pi^{\gamma_{k_2}}(x_{k_1})}{\pi^{\gamma_{k_1}}(x_{k_1}) \cdot \pi^{\gamma_{k_2}}(x_{k_2})}\right\} \\
&= \min\left\{1, \frac{[\pi(x_{k_2})]^{\gamma_{k_1}}}{[\pi(x_{k_2})]^{\gamma_{k_2}}} \cdot \frac{[\pi(x_{k_1})]^{\gamma_{k_2}}}{[\pi(x_{k_1})]^{\gamma_{k_1}}}\right\} \\
&= \min\left\{1, [\pi(x_{k_2})]^{\gamma_{k_1}-\gamma_{k_2}} \cdot [\pi(x_{k_1})]^{\gamma_{k_2}-\gamma_{k_1}}\right\} \\
&= \min\left\{1, \frac{[\pi(x_{k_2})]^{\gamma_{k_1}-\gamma_{k_2}}}{[\pi(x_{k_1})]^{\gamma_{k_1}-\gamma_{k_2}}}\right\} = \min\left\{1, \left(\frac{\pi(x_{k_2})}{\pi(x_{k_1})}\right)^{\gamma_{k_1}-\gamma_{k_2}}\right\}
\end{aligned}
$$

Parallel Tempering: Swap Move Acceptance

Metropolis-Hastings Derivation

Joint target: $\pi^{\gamma_1} \otimes \pi^{\gamma_2} \otimes \cdots \otimes \pi^{\gamma_N}$ where γ_i (inverse temperature)
MH acceptance ratio:

$$\alpha = \frac{\text{Probability of proposed state}}{\text{Probability of current state}} = \frac{\pi^{\gamma_{k_1}}(x_{k_2})\pi^{\gamma_{k_2}}(x_{k_1})}{\pi^{\gamma_{k_1}}(x_{k_1})\pi^{\gamma_{k_2}}(x_{k_2})}$$

Accept with probability: $\min(1, \alpha)$
Swapping state of two chains doesn't change the joint target distribution. This ensures detailed balance w.r.t. the joint distribution!

Parallel Tempering MCMC: Normalization Requirements

Target Distribution

NOT required to be normalized

Tempered Distributions

Also NOT normalized

Why It Works: Acceptance Ratios

Within-chain moves:

$$\alpha = \min\left(1, \frac{\pi_i(\mathbf{x}')}{\pi_i(\mathbf{x})}\right)$$

Normalizing constants cancel!

Between-chain swaps:

$$\alpha = \min\left(1, \frac{\pi_i(\mathbf{x}_j)\pi_j(\mathbf{x}_i)}{\pi_i(\mathbf{x}_i)\pi_j(\mathbf{x}_j)}\right)$$

Constants cancel again!

Why Swapping Works

The Relay Race Mechanism:

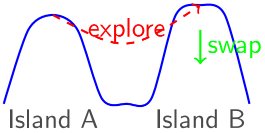
1. Hot chain randomly discovers new mode
2. Swap propagates discovery downward
3. Cold chain thoroughly explores it
4. Information flows both ways

Swap Acceptance (adjacent chains):

$$\alpha = \min\left\{1, \left(\frac{\pi(x_{k+1})}{\pi(x_k)}\right)^{\gamma_k - \gamma_{k+1}}\right\}$$

- Favors moving high-prob states to cold
- Favors moving low-prob states to hot
- Adjacent swaps → high acceptance

Example: Two Islands



Without PT: Stuck on Island A forever
With PT: Both islands sampled!

Detailed Balance and Ergodicity

Proposition (Detailed Balance)

The parallel tempering algorithm satisfies detailed balance with respect to the joint distribution:

$$\pi(x_1, \ldots, x_K) = \prod_{i=1}^{K} \frac{1}{Z_i} \pi(x_i)^{1/T_i}$$

Proof Sketch:

1. Within-chain moves: Standard MCMC detailed balance
2. Swap moves: Show $\pi(\mathbf{x})P(\mathbf{x} \to \mathbf{x}') = \pi(\mathbf{x}')P(\mathbf{x}' \to \mathbf{x})$
3. Symmetry of proposal + Metropolis ratio ensures balance

Unbiased MCMC

Unbiased MCMC

- Standard MCMC estimators are biased for any fixed number of iterations.
- We do not know in practice how many iterations are needed to reduce bias to an acceptable level.
- Burn-in period is often used to reduce bias, but it is difficult to choose appropriately.
- Unbiased MCMC estimators can be constructed using coupling techniques.
- Glynn–Rhee estimator provides a way to obtain unbiased estimates of expectations with respect to the target distribution.
- Key assumptions include moment conditions, geometric tail bounds on meeting times, and the property that chains stay together after meeting.
- Since the estimator is unbiased, we can generate shorter chains (in parallel) and average them to reduce variance without introducing bias.
- The cost is more complexity in implementation and the need to design effective coupling strategies.

Unbiased MCMC

- **Unbiased estimation using coupling** (Glynn and Rhee, 2014, Jacob et al., 2020 & 2024)

$$X_0 \sim \pi_0, \quad Y_0 \sim \pi_0, \quad X_1 \sim K(X_0, \cdot)$$
$$(X_{t+1}, Y_t) \sim \bar{K}((X_t, Y_{t-1}), (\cdot, \cdot)), \quad \forall t \geq 1$$

- **Assumption 1**: $E_\pi[h(X)] = \lim_{t\to\infty} E[h(X_t)]$ and there is $\eta > 0$ and $D < \infty$ that $E[|h(X_t)|^{2+\eta}] \leq D$ for all t.
- **Assumption 2**: meeting/coupling time

  $$\tau = \inf\{t \geq 1 : X_t = Y_{t-1}\} \text{ satisfies } \Pr(\tau > t) \leq C\delta^t \text{ for all}$$

  t, for some constant $C < \infty$ and $\delta < 1$.
- **Assumption 3**: the chains stay together after meeting: $X_t = Y_{t-1}$ for all $t \geq \tau$.

Glynn–Rhee estimator - assumptions

- **Assumption 1**: To ensure existence of moments
- **Assumption 2**: This is the hardest to verify in practice. It requires the coupling to be efficient enough so that the meeting time has geometric tails
- **Assumption 3**: This is often easy to ensure in practice by designing the coupling appropriately

Glynn–Rhee debiasing formula

Theorem

Under assumptions A1-A3, for any fixed $k \geq 0$

$$E_\pi[h(X)] = E[h(X_k) + \sum_{t=k+1}^{\tau-1} [h(X_t) - h(Y_{t-1})]]$$

$$E_\pi[h(X)] \underset{\text{by A1}}{=} \lim_{t\to\infty} E[h(X_t)] = E[h(X_k)] + \sum_{t=k+1}^{\infty} \{E[h(X_t)] - E[h(X_{t-1})]\}$$

$$\underset{\text{by A1 and A2}}{=} E\left[h(X_k) + \sum_{t=k+1}^{\infty} [h(X_t) - h(Y_{t-1})]\right] \underset{\text{by A3}}{=} E\left[h(X_k) + \sum_{t=k+1}^{\tau-1} [h(X_t) - h(Y_{t-1})]\right]$$

Example of Coupling: Maximal Coupling

Algorithm 4 Sampling a coupling of p and q. The coupling maximizes $\mathbb{P}(X = Y)$.

```
Sample $X \sim p$.
Sample $U \sim \text{Uniform}(0, 1)$.
**if** $U \leq q(X)/p(X)$ **then**
    set $Y = X$.
**else**
    **while** true **do**
        sample $Y^* \sim q$ and $W^* \sim \text{Uniform}(0, 1)$ until $W^* > p(Y^*)/q(Y^*)$
    **end while**
    set $Y = Y^*$.
**end if**
Return $(X, Y)$.
```

Maximal Coupling of two Gaussians

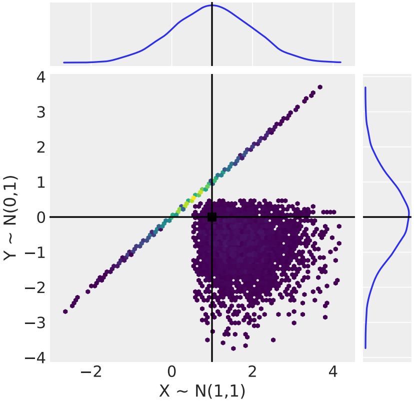


Figure: Maximal coupling of $\mathcal{N}(0,1)$ and $\mathcal{N}(1,1)$. Geometric interpretation: Maximizes mass on the diagonal.

Coupling

Couplings of MCMC algorithms can be devised using maximal couplings, reflection couplings, and common random numbers. We have focused on couplings that can be implemented without further analytical knowledge about the target distribution or about the MCMC kernels. However, these couplings might result in prohibitively large meeting times, either because the marginal chains mix slowly or because the coupling strategy is ineffective

Coupling

Algorithm 5 Successful coupling of chains with lag L and length ℓ. Coupled initial distribution: $\tilde{\pi}_0$, transition: P, coupled transition: $\tilde{P}$, meeting time $\tau = \inf\{t \geq L : X_t = Y_{t-L}\}$.

```
Sample $(X_0, Y_0)$ from $\tilde{\pi}_0$.
if $L \geq 1$ then
    for $t = 1, \dots, L$ do
        sample $X_t$ from $P(X_{t-1}, \cdot)$
    end for
end if
for $t \geq L$ do
    while true do
        sample $(X_{t+1}, Y_{t-L+1})$ from $\tilde{P}((X_t, Y_{t-L}), \cdot)$ until $X_{t+1} = Y_{t-L+1}$ and $t+1 \geq \ell$
    end while
end for
```

Vanilla HMC

- Hamiltonian Monte Carlo (HMC) is an MCMC algorithm that leverages concepts from physics to propose new states in the Markov chain.
- It introduces auxiliary momentum variables and simulates Hamiltonian dynamics to explore the target distribution more efficiently.
- In class we saw how MALA improved upon RW-Metropolis by using gradient information; HMC takes this further by simulating trajectories in the state space.
- In high-dimensional spaces it is not enough to explore regions around the modes.
- In high dimensions, probability mass concentrates on a thin shell away from modes
- This typical set has lower density but massive volume → contains most probability mass
- Example: In 100D Gaussian, samples lie 10 units from origin, not at origin!
- So we need a method that makes proposals based on more than the local moves or local gradient at the current position.

Physical Interpretation

Neal, 2011

In two dimensions, we can visualize the dynamics as that of a frictionless puck that slides over a surface of varying height. The state of this system consists of the position of the puck, given by a 2D vector q, and the momentum of the puck (its mass times its velocity), given by a 2D vector p.

On a level part of the surface, the puck moves at a constant velocity. If it encounters a rising slope, the puck's momentum allows it to continue, with its kinetic energy $K(p)$ decreasing and its potential energy $U(q)$ increasing, until the kinetic energy is zero, at which point it will slide back down (with kinetic energy increasing and potential energy decreasing)

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

Hamiltonian Dynamics and Equations

Our target distribution is defined in terms of a potential energy function $U(q)$, which encodes the negative log probability of the target distribution $\pi(q)$ that we wish to sample from.
The dynamics of the system can be described by Hamilton's equations, which govern the time evolution of the position and momentum variables. In our case, these equations take the form:

$$\frac{dq}{dt} = \frac{\partial H}{\partial p} \quad \text{and} \quad \frac{dp}{dt} = -\frac{\partial H}{\partial q}$$

where $H(q, p)$ is the Hamiltonian function, representing the total energy of the system, given by the sum of kinetic and potential energy:

$$H(q, p) = K(p) + U(q) = \frac{1}{2} p^T M^{-1} p + U(q)$$

Leapfrog Integrator

To numerically simulate Hamiltonian dynamics, we use the leapfrog integrator, which is a symplectic method that preserves the volume in phase space and is time-reversible.

$$\begin{aligned} p\left(t+\frac{\varepsilon}{2}\right) &= p(t) - \frac{\varepsilon}{2}\nabla U(q(t)) \\ q(t+\varepsilon) &= q(t) + \varepsilon M^{-1} p\left(t+\frac{\varepsilon}{2}\right) \\ p(t+\varepsilon) &= p\left(t+\frac{\varepsilon}{2}\right) - \frac{\varepsilon}{2}\nabla U(q(t+\varepsilon)) \end{aligned}$$

where ε is the step size.

The Leapfrog integrator is used to simulate the Hamiltonian dynamics over a series of steps, so it is the backbone of the proposal mechanism in HMC or in other words it transforms the current state to a proposed new state. Being symplectic means that this transformation preserves volume in phase space and that the Jacobian determinant of the transformation is equal to one and hence the Metropolis Acceptance ratio needs no volume correction factor.

Vanilla HMC Algorithm

Algorithm

Requires: Leapfrog integrator φ, step-size ε, number of steps L, current position q_n and positive definite matrix M.

1. **Energy Lift**: given q_t, draw $p_t \sim N(0, M)$ - (random)
   This "lifts" our position into phase space by adding kinetic energy
2. **Hamilton flow**: $q^*, p^* = \varphi_\varepsilon^L(q_t, p_t)$ - (deterministic)
   Simulate dynamics for L steps using leapfrog integrator. Follow energy-conserving trajectory through phase space. The chain is constructed on the joint (q, p); marginalizing p yields $\pi(q)$ stationarity.
3. **Metropolis acceptance step** - (random)
   accept $q_{t+1} = q^*$ with probability $\min\left\{1, \exp(H(q_t, p_t) - H(q^*, -p^*))\right\}$
   Corrects for numerical errors in integration. No Jacobian term; leapfrog is volume-preserving

Choosing parameters in HMC

Another story...

Step-size ε: optimal scaling

- Dimension dependence of stepsize:
  - RWM: $\mathcal{O}(d^{-1})$
  - MALA: $\mathcal{O}(d^{-1/3})$
  - HMC: $\mathcal{O}(d^{-1/4})$
- Optimal acceptance rates:
  - RWM: 0.234 (see Roberts, et. al., 1997)
  - MALA: 0.574 (see Roberts and Rosenthal, 1998)
  - HMC: 0.651 (see Beskos, et. al., 2013)

Choose L **adaptively**: NUTS sampler