Markov Chain Monte Carlo (MCMC) Sampling

ELG 5218 - Uncertainty Evaluation in Engineering Measurements and Machine Learning

Miodrag Bolic

University of Ottawa

February 1, 2026

Outline

- ✓ **Motivation & Problem Setup**
  - The inference challenge
  - Why sampling beats integration
- ✓ **Core Concepts**
  - Markov chains, stationarity
  - Detailed balance
- ✓ **Metropolis-Hastings**
  - Intuition and mechanics
  - Evidence cancellation

Notebook: Lec3a_python_numpyro.ipynb

- ✓ **Hamiltonian Monte Carlo**
- ✓ **Convergence Diagnostics**
  - Burn-in, ACF, ESS
  - Gelman-Rubin $\hat{R}$
- ✓ **Modern Samplers**
  - HMC, NUTS,

The Fundamental Problem

We want to compute the **posterior distribution**:

$$P(\theta|x) = \frac{P(x|\theta)P(\theta)}{P(x)}$$

Key Components:

- $P(\theta|x)$: **posterior** (what we want)
- $P(x|\theta)$: **likelihood** (easy to compute)
- $P(\theta)$: **prior** (chosen by us)
- $P(x) = \int P(x|\theta)P(\theta)d\theta$: **evidence** (hard!)

The evidence integral is the computational bottleneck.

Why Bayesian Inference Is Hard

The evidence integral:

$$P(x) = \int_{\Theta} P(x|\theta)P(\theta)\, d\theta$$

Problems:

- High-dimensional integrals are intractable
- No closed form for most realistic models
- Standard numerical integration fails in high dimensions (curse of dimensionality)

Why classical methods fail:

Method	Dimensionality	Status
Direct sampling	Any	✓Requires inverting posterior
Grid approximation	$D = 5$	✓Intractable (10^5 points)
Rejection sampling	$D > 3$	✓Exponential rejection rate

The Surprising Insight: MCMC

Key Idea: Instead of computing $P(\theta|x)$ directly, we can:

1. **Construct** a Markov chain with stationary distribution $= P(\theta|x)$
2. **Sample** from this chain long enough to reach stationarity
3. **Approximate** the posterior using collected samples

Key observation: The chain's stationary distribution "remembers" the target distribution—we don't need to compute the evidence explicitly!

This is the magic of MCMC:

Avoiding direct computation of $P(x)$ while still sampling from $P(\theta|x)$

Why Sampling Works: Monte Carlo Integration

Approximating expectations via sampling:

$$\mathbb{E}[f(\theta)] = \int f(\theta) p(\theta|x)\, d\theta \approx \frac{1}{N} \sum_{i=1}^{N} f(\theta_i)$$

Advantages of sampling over quadrature:

- Curse of dimensionality disappears: N samples work for any dimension D
- Unbiased estimator: By the ergodic theorem, samples converge to truth
- Easy uncertainty quantification: Standard error $\sim 1/\sqrt{N}$
- Flexible: Works with unnormalized densities

Markov Chains: Definition

A sequence $\{\theta_t\}_{t=1}^{\infty}$ is a **first-order Markov chain** if:

$$P(\theta_{t+1}|\theta_t, \theta_{t-1}, \ldots) = P(\theta_{t+1}|\theta_t)$$

Key property: Future only depends on present, not history.

Transition distribution:

$$P(\theta_{t+1}|\theta_t) = P(\theta'|\theta)$$

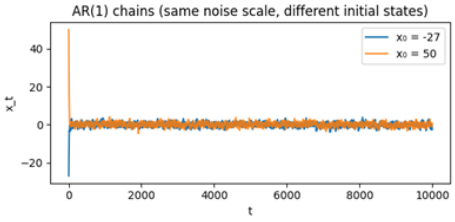
Simple Example:

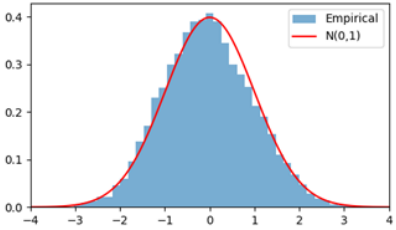
- AR(1) process: $x(t+1) = 0.9 \cdot x(t) + v(t)$ where $v(t) \sim \mathcal{N}(0, 0.19)$
- Starting at $x_0 = -27$ or $x_0 = 50$, both converge to $\mathcal{N}(0, 1)$
- This is a Markov chain with stationary distribution $\mathcal{N}(0, 1)$

AR(1) Example: Convergence to Stationarity

Key observations:

- Despite initialization at $x_0 = -27$, variance converges to 1
- Mean converges to 0
- After burn-in, samples are approximately from $\mathcal{N}(0, 1)$





Transition Function (Kernel) and Matrix Form

Transition function / kernel:

$$P(\theta' \mid \theta) \quad \text{with} \quad P(\theta' \mid \theta) \geq 0, \qquad \int P(\theta' \mid \theta)\, d\theta' = 1.$$

Finite state space ($\theta \in \{1, \ldots, K\}$):

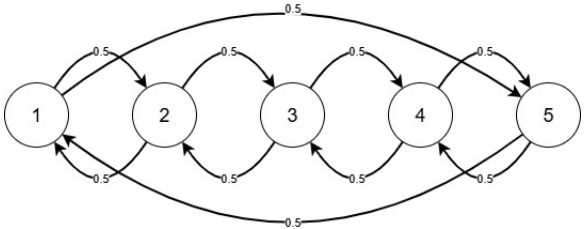
$$T_{ij} \;:=\; P(\theta' = j \mid \theta = i), \qquad \sum_{j=1}^{K} T_{ij} = 1.$$

If p_t is the row-vector distribution of θ_t, then:
$p_{t+1} = p_t\, T$, and more generally $p_{t+m} = p_t\, T^m$.

Finite-state matrix form:

$$P(\theta_{t+m} = j \mid \theta_t = i) = (T^m)_{ij}.$$

Example of a discrete chain



Discrete Example:
Transition matrix:

$$T = \begin{bmatrix} 0 & 0.5 & 0 & 0 & 0.5 \\ 0.5 & 0 & 0.5 & 0 & 0 \\ 0 & 0.5 & 0 & 0.5 & 0 \\ 0 & 0 & 0.5 & 0 & 0.5 \\ 0.5 & 0 & 0 & 0.5 & 0 \end{bmatrix}$$

Two-step Probability Example

For the 5-state random walk, the two-step probability is:

$$P(\theta_{t+2} = \theta_3 \mid \theta_t = \theta_1) = (T^2)_{1,3}.$$

There is exactly one 2-step path from θ_1 to θ_3:

$$\theta_1 \to \theta_2 \to \theta_3,$$

so

$$P(\theta_{t+2} = \theta_3 \mid \theta_t = \theta_1) = 0.5 \times 0.5 = 0.25.$$

Stationary distribution: $\pi = [0.2, 0.2, 0.2, 0.2, 0.2]$ (uniform)

Check: After 30 steps, $T^{30} \approx \mathbf{1}\boldsymbol{\pi}^T$ (each row approaches π)

Stationarity = Unchanged by One Step

A Markov chain evolves using the transition kernel $P(\theta' \mid \theta)$.

One-step update:

$$p_{t+1}(\theta') = \int p_t(\theta)\, P(\theta' \mid \theta)\, d\theta.$$

(Discrete version: $p_{t+1}(j) = \sum_i p_t(i) T_{ij}$ with $T_{ij} = P(j \mid i)$.)

Stationary distribution: π is stationary if applying one step does not change it:

$$\pi(\theta') = \int \pi(\theta)\, P(\theta' \mid \theta)\, d\theta.$$

Intuition: If $\theta_0 \sim \pi$, then $\theta_1, \theta_2, \ldots$ all have distribution π.

Ergodicity and Convergence

A Markov chain is **ergodic** if it satisfies three properties:

1. **Aperiodic:** Cannot return to a state at regular intervals
2. **Irreducible:** All states reachable from all other states with positive probability
3. **Positive recurrent:** Returns to any state in finite expected time

Fundamental Theorem of Markov Chains:

For an ergodic chain, regardless of initialization:

$$\lim_{t \to \infty} P(\theta_t) = \pi(\theta)$$

All chains converge to the same stationary distribution!

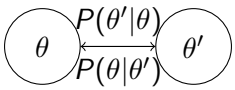
For MCMC: This guarantees that no matter where we start, we eventually sample from the target posterior.

Detailed Balance (Reversibility)

A chain satisfies **detailed balance** if:

$$\pi(\theta)P(\theta'|\theta) = \pi(\theta')P(\theta|\theta')$$

Intuition: Flow from state θ to θ' equals flow from θ' to θ.



Consequence: If detailed balance holds, then π is stationary!

Proof:

$$\int \pi(\theta)P(\theta'|\theta)\,d\theta = \int \pi(\theta')P(\theta|\theta')\,d\theta = \pi(\theta')\int P(\theta|\theta')\,d\theta = \pi(\theta').$$

Why Detailed Balance Matters for MCMC

Goal (MCMC design problem): We want a Markov chain with transition kernel $P(\theta' \mid \theta)$ such that

$$\text{(stationary)} \qquad \pi(\theta') = \int \pi(\theta)\, P(\theta' \mid \theta)\, d\theta.$$

Why this matters: If the chain is **ergodic** (e.g., irreducible + aperiodic in finite state), then starting from almost any p_0 we get

$$p_t \longrightarrow \pi,$$

so after burn-in the chain produces samples distributed like π.

Key idea: Detailed balance is an easy sufficient condition.

$$\boxed{\pi(\theta)\, P(\theta' \mid \theta) = \pi(\theta')\, P(\theta \mid \theta')} \quad \Longrightarrow \quad \pi \text{ is stationary.}$$

Metropolis–Hastings: Enforcing Detailed Balance

The Challenge:

- **Given:** Target distribution $\pi(\theta)$ (the posterior)
- **Find:** Transition probability $P(\theta'|\theta)$ such that π is stationary
- **Constraint:** Don't want to compute $P(x)$ (evidence)

Metropolis Solution (1953):

1. Propose new state $\theta^* \sim q(\theta^*|\theta)$ from arbitrary proposal
2. Accept with probability $\alpha(\theta \to \theta^*)$ (acceptance ratio)
3. Choose α to guarantee detailed balance

Important result:

By carefully choosing the acceptance probability, we can construct a chain that satisfies detailed balance with the target posterior—**without computing the evidence**!

Metropolis-Hastings Algorithm: Complete Description

Algorithm 1 Metropolis-Hastings MCMC

1: **Initialize** θ_0 arbitrarily
2: **for** $t = 1, 2, \ldots, T$ **do**
3: **Propose:** Draw $\theta^* \sim q(\theta^* | \theta_{t-1})$
4: **Compute acceptance ratio:**

$$\alpha = \min\left(1, \frac{\pi(\theta^*) q(\theta_{t-1} | \theta^*)}{\pi(\theta_{t-1}) q(\theta^* | \theta_{t-1})}\right)$$

5: **Accept/Reject:**
6: **if** rand() $< \alpha$ **then**
7: $\theta_t = \theta^*$ (accept proposal)
8: **else**
9: $\theta_t = \theta_{t-1}$ (reject, stay put)
10: **end if**
11: **end for**
12: **Return** $\{\theta_1, \ldots, \theta_T\}$ (post burn-in)

Why the Evidence Cancels

Acceptance ratio:

$$r = \frac{\pi(\theta^*)q(\theta_{t-1}|\theta^*)}{\pi(\theta_{t-1})q(\theta^*|\theta_{t-1})}$$

Expand using $\pi(\theta) = \frac{P(x|\theta)P(\theta)}{P(x)}$:

$$\begin{aligned} r &= \frac{\frac{P(x|\theta^*)P(\theta^*)}{P(x)} \cdot q(\theta_{t-1}|\theta^*)}{\frac{P(x|\theta_{t-1})P(\theta_{t-1})}{P(x)} \cdot q(\theta^*|\theta_{t-1})} \\ &= \frac{P(x|\theta^*)P(\theta^*) \cdot q(\theta_{t-1}|\theta^*)}{P(x|\theta_{t-1})P(\theta_{t-1}) \cdot q(\theta^*|\theta_{t-1})} \end{aligned}$$

The $P(x)$ cancels!

Why this is important:

- The evidence $P(x)$ is intractable, but it cancels in the ratio!
- We only need the **unnormalized** posterior (likelihood $\times$ prior)
- This is why MCMC is practical for high-dimensional problems

Acceptance Ratio: What Each Term Means

Define the MH ratio:

$$r(\theta \to \theta^*) = \frac{\tilde{\pi}(\theta^*)\, q(\theta \mid \theta^*)}{\tilde{\pi}(\theta)\, q(\theta^* \mid \theta)}, \qquad \alpha(\theta \to \theta^*) = \min(1, r).$$

Here $\tilde{\pi}(\theta) = p(x \mid \theta)p(\theta)$ is the unnormalized posterior (evidence cancels).

Interpretation: r has three terms

$$r = \underbrace{\frac{p(x \mid \theta^*)}{p(x \mid \theta)}}_{\text{likelihood}} \cdot \underbrace{\frac{p(\theta^*)}{p(\theta)}}_{\text{prior}} \cdot \underbrace{\frac{q(\theta \mid \theta^*)}{q(\theta^* \mid \theta)}}_{\text{proposal correction}}$$

Forward vs reverse proposals:

- **Forward:** $q(\theta^* \mid \theta)$ = propose θ^* when currently at θ.
- **Reverse:** $q(\theta \mid \theta^*)$ = propose returning to θ if currently at θ^*.

Decision rule:

- If $r \geq 1$: accept always (move to higher posterior density).
- If $r < 1$: accept with probability r (sometimes move to lower density = "downhill").

Metropolis Algorithm (Symmetric Proposal)

When proposal is symmetric: $q(\theta^*|\theta) = q(\theta|\theta^*)$

The proposal ratio term cancels:

$$\alpha = \min\left(1, \frac{\pi(\theta^*)}{\pi(\theta_{t-1})}\right)$$

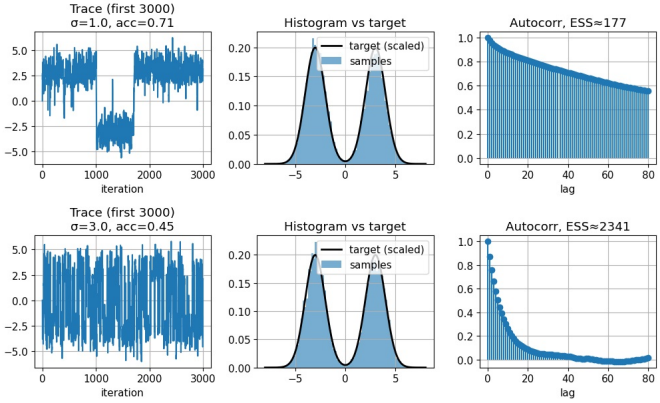
Random-Walk Metropolis (RWM):

- Propose: $\theta^* = \theta_{t-1} + \epsilon$ where $\epsilon \sim \mathcal{N}(0, \sigma^2)$
- This is symmetric: probability of going from θ to $\theta + \epsilon$ equals going from θ to $\theta - \epsilon$

Properties:

- If σ is too small $\rightarrow$ high acceptance but slow exploration (high autocorrelation).
- If σ too large $\rightarrow$ many rejections.
- Rule of thumb (high-dim random walk): acceptance 0.23 is often near-optimal

Random-Walk Metropolis Algorithm



Interpretation:

- Early samples (burn-in): chain moves toward high posterior regions
- Later samples: chain explores posterior, sometimes ventures into low-probability areas
- Oscillations show exploratory behavior

Proposal Width Tuning: The Critical Trade-off

The problem: Proposal standard deviation σ_{proposal} critically affects performance.

Scenario	Acceptance Rate	Mixing
$\sigma_{\text{prop}} \to 0$	$\approx 100\%$	Very slow (tiny steps)
σ_{prop} small	$\approx 60\%$	? Decent but still local
σ_{prop} optimal	$\approx 44\%$ (1D) / 24% (high-D)	Best ESS
σ_{prop} large	$\approx 10\%$	? Many rejections
$\sigma_{\text{prop}} \to \infty$	$\approx 0\%$	No movement

Adaptive tuning:

- Monitor acceptance rate during burn-in
- Increase σ_{prop} if acceptance too high
- Decrease σ_{prop} if acceptance too low

Detailed Balance Verification: Why MH Works

Claim: Metropolis-Hastings satisfies detailed balance.

Proof sketch:
The joint transition to $\theta' \neq \theta$ is:

$$P(\theta \to \theta') = q(\theta'|\theta) \cdot \min\left(1, \frac{\pi(\theta')}{\pi(\theta)}\right)$$

Detailed balance requires:

$$\pi(\theta)P(\theta \to \theta') = \pi(\theta')P(\theta' \to \theta)$$

Two cases:
Case 1: $\pi(\theta') > \pi(\theta)$ (uphill move)

$$\text{LHS} = \pi(\theta) \cdot q(\theta'|\theta) \cdot 1$$

$$\text{RHS} = \pi(\theta') \cdot q(\theta|\theta') \cdot \frac{\pi(\theta)}{\pi(\theta')} = \pi(\theta) \cdot q(\theta|\theta')$$

Equals by symmetry of proposal (or ratio term for general MH).

Case 2: $\pi(\theta') < \pi(\theta)$ (downhill move)

$$\text{LHS} = \pi(\theta) \cdot q(\theta'|\theta) \cdot \frac{\pi(\theta')}{\pi(\theta)} = \pi(\theta') \cdot q(\theta'|\theta)$$

$$\text{RHS} = \pi(\theta') \cdot q(\theta|\theta') \cdot 1$$

Equals by symmetry. ✓

What is "Wrong" with Random-Walk MH?

- Random-walk proposals move locally: the chain explores by many small steps.
- In high dimensions, step sizes must shrink to keep acceptance reasonable $\Rightarrow$ slow exploration.
- Strong correlations / narrow valleys: proposals often point in unhelpful directions $\Rightarrow$ many rejections.
- Bimodality: crossing low-probability gaps between modes can take a very long time.

Bottom line

MH is correct, but can be inefficient when the posterior geometry is challenging.

See interactive demos at `https://github.com/chi-feng/mcmc-demo`

Geometry Intuition: Why MH Can Be Slow

- Think of $\pi(\theta)$ as a landscape:
  - high probability = valleys/ridges where we want to spend time
  - low probability = steep cliffs or empty regions
- Random-walk MH keeps "bumping" around locally, so it diffuses slowly across the landscape.
- We want long moves that stay within high-probability regions.

Hamiltonian Monte Carlo (HMC): The Big Idea

- HMC augments the state with a temporary momentum (think: position + velocity).
- It uses gradient information to move in a directed, smooth trajectory through parameter space.
- Result: proposals can be far away but still land in high-probability regions.
- Then a Metropolis accept/reject step keeps the method exact.

Key intuition

HMC uses the gradient $\nabla_\theta \log \pi(\theta)$ to guide proposals (less random-walk diffusion).

HMC: What Happens Each Iteration (Conceptual)

1. Sample a fresh momentum (random direction + speed).
2. Simulate a short trajectory guided by the gradient (many small internal steps).
3. Propose the endpoint and accept/reject (corrects numerical error).

Why it is better than random-walk MH

- Momentum reduces back-and-forth dithering.
- Trajectories follow the posterior geometry.
- Lower autocorrelation $\Rightarrow$ higher effective sample size per compute.

HMC still needs tuning (step size and trajectory length).

NUTS: HMC Without Choosing a Trajectory Length

- In HMC, choosing how long to simulate (trajectory length) is tricky.
- Too short $\Rightarrow$ random-walk behavior returns; too long $\Rightarrow$ wasted compute (path loops back).

NUTS (No-U-Turn Sampler) removes this knob

- It grows a trajectory forward and backward.
- It stops automatically when the path starts doubling back (a "U-turn").
- Modern implementations also adapt step size during warm-up.

Practical Takeaways: When to Use What

- **Random-walk MH:**
  - simple, no gradients needed
  - can struggle in high dimensions or strong correlations
- **HMC / NUTS:**
  - best for continuous, differentiable parameters
  - often faster exploration and lower autocorrelation
  - NUTS reduces manual tuning
- For strong multimodality, even HMC/NUTS may need additional strategies (e.g., tempering).

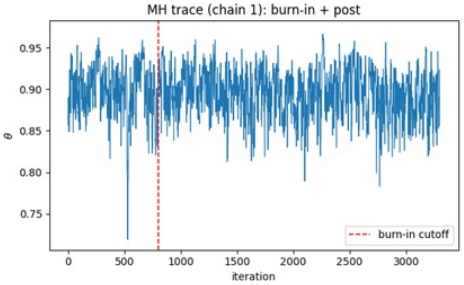
Burn-in and Warm-up Phases

The problem: Early samples depend on starting point, not stationary distribution.

Solution: Discard early **burn-in** (also called **warm-up**) samples.

Rules of thumb:

- Remove $\sim$ 10%–50% of total samples
- Visual inspection: when does trace look "stable"?



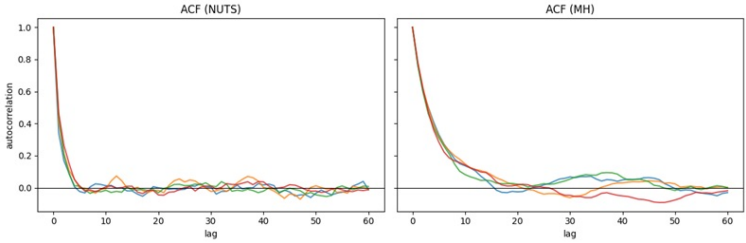
Autocorrelation Function (ACF)

The autocorrelation at lag ℓ:

$$\rho(\ell) = \text{Corr}(\theta_t, \theta_{t+\ell})$$

Interpretation:

- $\rho(1) \approx 1$: Samples very correlated (poor mixing)
- $\rho(1) \approx 0$: Samples independent (good mixing)
- $\rho(\ell) \to 0$: Correlation decays over lags



Effective Sample Size (ESS)

Key idea: Correlated samples are less informative than independent samples.

Define autocorrelation time:

$$\tau = 1 + 2\sum_{\ell=1}^{\infty} \rho(\ell)$$

Effective sample size:

$$N_{\text{eff}} = \frac{N}{\tau}$$

Intuition: N correlated samples $\approx N_{\text{eff}}$ independent samples.

Monte Carlo standard error: $\text{SE}(\hat{\theta}) = \frac{\sigma(\theta)}{\sqrt{N_{\text{eff}}}}$

Example from your notes:

- MH sampler: $N = 3000$, $N_{\text{eff}} \approx 1133$
- HMC sampler: $N = 3000$, $N_{\text{eff}} \approx 2972$
- **Conclusion:** HMC produces $\sim 3\times$ more useful samples!

Gelman-Rubin Statistic ($\hat{R}$): Convergence Assessment

Idea: Run J parallel chains with different random initializations.

After burn-in, keep L samples from each chain.

Between-chain variance:

$$B = \frac{L}{J-1} \sum_{j=1}^{J} (\bar{\theta}_j - \bar{\theta}_\cdot)^2$$

Within-chain variance:

$$W = \frac{1}{J} \sum_{j=1}^{J} s_j^2$$

where s_j^2 is sample variance of chain j.

Potential Scale Reduction Factor:

$$\hat{R} = \sqrt{\frac{(L-1)W + B}{L \cdot W}} = \sqrt{\frac{\text{between}}{\text{within}}}$$

Gelman–Rubin Statistic ($\hat{R}$): Setup and Derivation

Setup (after burn-in)

- Run J independent chains with different initializations.
- Keep L post–burn-in samples from each chain.
- Denote draws: $\{\theta_{j\ell}\}$, with $j = 1, \ldots, J$ and $\ell = 1, \ldots, L$.

Per-chain summaries

$$\bar{\theta}_j = \frac{1}{L}\sum_{\ell=1}^{L}\theta_{j\ell}, \qquad s_j^2 = \frac{1}{L-1}\sum_{\ell=1}^{L}(\theta_{j\ell} - \bar{\theta}_j)^2$$

$$\bar{\theta}_{\cdot} = \frac{1}{J}\sum_{j=1}^{J}\bar{\theta}_j \quad \text{(grand mean)}$$

Within-chain variance $W = \frac{1}{J}\sum_{j=1}^{J} s_j^2$

Between-chain variance $B = \frac{L}{J-1}\sum_{j=1}^{J}(\bar{\theta}_j - \bar{\theta}_{\cdot})^2$

Why the L in B?
The variance of a chain mean is roughly $\mathrm{Var}(\bar{\theta}_j) \approx \sigma^2/L$. Multiplying by L puts B on the same scale as W (a variance scale), so they are directly comparable.

Estimating the marginal variance

$$\widehat{\mathrm{Var}}^{+}(\theta) = \left(1 - \frac{1}{L}\right) W + \frac{1}{L} B = \frac{L-1}{L} W + \frac{1}{L} B$$

Potential Scale Reduction

$$\hat{R} = \sqrt{\frac{\widehat{\mathrm{Var}}^{+}(\theta)}{W}} = \sqrt{\frac{(L-1)W + B}{L\,W}}$$

Convergence cue

- If chains have converged and mix well: $B \approx W \Rightarrow \hat{R} \approx 1$.
- If chain means still differ: $B > W \Rightarrow \hat{R} > 1$ (keep sampling / reparameterize / retune).

Gelman-Rubin: Interpretation

$$\hat{R} = \sqrt{\frac{\text{between-chain variance}}{\text{within-chain variance}}}$$

$\hat{R}$ value	Status	Action
≈ 1.00	Converged	Use samples
$1.00 - 1.05$	Nearly converged	Probably fine
$1.05 - 1.10$	Unconverged	? Continue sampling
> 1.10	Not converged	Chains still diverging

Intuition:

- $\hat{R} = 1.0$: Chains indistinguishable (converged!)
- $\hat{R} = 1.10$: Could shrink estimate by $\sim 10\%$ with more sampling
- $\hat{R} > 1.20$: Chains exploring different regions (not converged)

Convergence Diagnostic Summary

Use all four diagnostics together:

Diagnostic	What to look for	Good value	Bad sign
Trace plot	Stable, mixed paths	No trends	Drifting, stuck
ACF	Decays quickly to 0	$\rho(10) < 0.1$	Slow decay
ESS/N	Efficiency	> 0.10	< 0.05
$\hat{R}$	Convergence	< 1.05	> 1.10

Red flags that indicate problems:

- Trace plot shows trends or doesn't explore space
- ACF doesn't decay
- ESS is very small (< 100)
- $\hat{R} > 1.05$ or shows high variance

When problems occur:

1. Increase number of iterations
2. Improve sampler (switch to HMC/NUTS)
3. Re-examine model (is it well-specified?)
4. Standardize data (helps numerics)

Core Ideas Summary

The Pillars of MCMC:

1. **Problem:** High-dimensional integrals intractable
2. **Solution:** Construct Markov chain with correct stationary distribution
3. **Theory:** Detailed balance + ergodicity $\Rightarrow$ convergence
4. **Algorithm:** Metropolis-Hastings makes evidence cancel in acceptance ratio
5. **Diagnostics:** Multiple checks ensure convergence (trace, ACF, ESS, $\hat{R}$)
6. **Modern:** Gradient-based (HMC/NUTS) samplers solve high-dimensional problems

Practical takeaway:

Use probabilistic programming (Turing.jl, Stan, PyMC) to automate sampler selection and diagnostics!

Common Pitfalls to Avoid

× **Ignoring burn-in:** Starting distribution effects can dominate early samples.

× **Single chain only:** Can't assess convergence without multiple chains. Always run $J \geq 2$ chains!

× **Over-interpretation of single diagnostic:** Use all four: trace plot, ACF, ESS, $\hat{R}$

× **Confusing thinning with efficiency:** Thinning reduces auto-correlation but wastes computation. With modern samplers, don't thin.

× **Ignoring effective sample size:** N samples with $\tau = 10$ autocorrelation time $\Rightarrow$ only $N/10$ effective samples!

× **Using wrong sampler:** MH on high-dimensional problems $\Rightarrow$ slow. Use NUTS instead.

× **Forgetting to standardize data:** Helps with numerics and sampling efficiency. Center and scale features.

Research Frontiers & Extensions

Approximate methods: Variational Inference

- Fast (no sampling loop)
- Less accurate (mean-field approximation)
- Useful for exploratory analysis

Adaptive MCMC:

- Automatically tune proposal distribution during sampling
- Reduces manual tuning burden

Parallel tempering:

- Multiple chains at different "temperatures"
- Helps escape local modes

Delayed acceptance:

- Two-stage proposals for expensive model evaluations
- Cheap pretest before expensive full evaluation

Further Reading & Resources

Textbooks / monographs:

- *Bayesian Data Analysis* (3rd ed.) — Gelman, Carlin, Stern, Dunson, Vehtari, & Rubin (2013).
- *Information Theory, Inference, and Learning Algorithms* — MacKay (2003).
- *An Introduction to Probabilistic Programming* — van de Meent, Paige, Yang, & Wood (2018; revised 2021).

Review articles / chapters:

- Geyer (2011): *Introduction to Markov Chain Monte Carlo*. In *Handbook of Markov Chain Monte Carlo*.
- Andrieu, de Freitas, Doucet, & Jordan (2003): *An Introduction to MCMC for Machine Learning*. *Machine Learning*, 50, 5–43.
- Betancourt (2017): *A Conceptual Introduction to Hamiltonian Monte Carlo*. arXiv:1701.02434.

Software & tutorials:

- `Turing.jl` documentation
- Stan user's guide
- PyMC tutorials and examples
- ArviZ: Exploratory analysis of Bayesian models

Interactive visualizations:

- MCMC visualizations `https://chi-feng.github.io/mcmc-demo/`
- Markov Chain Visualizer (online)