Markov Chain Monte Carlo Methods
Problem Set

Diogo Ribeiro
ESMAD - Escola Superior de Média Arte e Design
Lead Data Scientist, Mysense.ai
dfr@esmad.ipp.pt
ORCID: 0009-0001-2022-7072

November 16, 2025

Abstract

This problem set covers fundamental and advanced concepts in Markov Chain Monte Carlo (MCMC) methods. Students should complete these exercises to gain practical experience with MCMC algorithms, convergence diagnostics, and applications to Bayesian inference. Solutions should be submitted as Jupyter notebooks or R Markdown documents with complete code and explanations.

1 Theoretical Foundations

Exercise 1 (Detailed Balance)**.** *Consider a Markov chain with transition kernel $K(x \to y)$ and stationary distribution $\pi(x)$.*

(a) Prove that if the detailed balance condition holds:

$$\pi(x)K(x \to y) = \pi(y)K(y \to x)$$

then π is a stationary distribution of the chain.

(b) Show that the Metropolis-Hastings algorithm satisfies detailed balance with respect to the target distribution $\pi(x)$.

(c) Give an example of a Markov chain that has π as a stationary distribution but does not satisfy detailed balance.

Exercise 2 (Acceptance Probability)**.** *The Metropolis-Hastings algorithm uses acceptance probability:*

$$\alpha(x \to x') = \min\left(1, \frac{\pi(x')q(x'|x)}{\pi(x)q(x|x')}\right)$$

(a) Derive this formula from the detailed balance condition.

(b) For a symmetric proposal $q(x'|x) = q(x|x')$, show that this reduces to the Metropolis algorithm acceptance probability.

(c) Consider an asymmetric proposal. How does the Hastings correction $\frac{q(x|x')}{q(x'|x)}$ affect the acceptance rate?

Exercise 3 (Convergence Diagnostics)**.** *Explain the following convergence diagnostics and when each is most useful:*

(a) Gelman-Rubin $\hat{R}$ statistic

(b) Effective Sample Size (ESS)

(c) Geweke diagnostic

(d) Autocorrelation plots

For each, provide:

- *Mathematical definition*
- *Interpretation guidelines*
- *When it might give misleading results*

2 Implementation Exercises

Problem 1 (Metropolis-Hastings from Scratch)**.** *Implement the Metropolis-Hastings algorithm to sample from a mixture of two Gaussians:*

$$\pi(x) = 0.3\mathcal{N}(-3, 1) + 0.7\mathcal{N}(2, 1.5)$$

Requirements:

1. *Use a Gaussian random walk proposal: $q(x'|x) = \mathcal{N}(x, \sigma^2)$*
2. *Experiment with different proposal variances: $\sigma \in \{0.5, 1.0, 2.5, 5.0\}$*
3. *Generate 10,000 samples with 2,000 burn-in*
4. *Plot:*
    - *Trace plots*
    - *Histograms comparing samples to true density*
    - *Autocorrelation functions*
    - *Acceptance rates vs. proposal variance*
5. *Calculate effective sample size for each σ*
6. *Which proposal variance gives the best efficiency?*

Problem 2 (Adaptive MCMC)**.** *Implement an adaptive Metropolis algorithm that automatically tunes the proposal variance during burn-in.*

Algorithm:

1. *Start with initial proposal variance σ_0^2*
2. *Every 50 iterations during burn-in:*
    - *If acceptance rate > 0.44: increase σ^2 by 10%*
    - *If acceptance rate < 0.23: decrease σ^2 by 10%*
3. *Fix σ^2 after burn-in*

Tasks:

1. *Implement this algorithm*

2. *Test on the mixture distribution from Problem 1*
3. *Plot the evolution of σ^2 during burn-in*
4. *Compare final performance to fixed proposals*

Note: *The target acceptance rates 0.44 and 0.23 are theoretically optimal for 1D and high-dimensional Gaussian targets respectively.*

Problem 3 (Hamiltonian Monte Carlo)**.** *Implement basic Hamiltonian Monte Carlo for a 2D correlated Gaussian:*

$$\pi(x) = \mathcal{N}(\mu, \Sigma)$$

where $\mu = [0, 0]^T$ *and*

$$\Sigma = \begin{pmatrix} 1 & 0.8 \\ 0.8 & 1 \end{pmatrix}$$

Requirements:

1. *Implement leapfrog integrator*
2. *Use $L = 20$ leapfrog steps with step size $\epsilon = 0.1$*
3. *Generate 5,000 samples*
4. *Compare to Metropolis-Hastings:*
   - *Acceptance rates*
   - *Effective sample size*
   - *Autocorrelation*
   - *Mixing (visual inspection of 2D scatter plots)*

Bonus: *Implement a simple step size adaptation during burn-in to achieve acceptance rate* ≈ 0.65.

3 Bayesian Inference Applications

Problem 4 (Logistic Regression)**.** *Use MCMC to perform Bayesian inference for logistic regression.*

Setup:

- *Generate synthetic data: $n = 200$ observations*
- *True parameters: $\beta_0 = -1.5, \beta_1 = 2.0, \beta_2 = -0.5$*
- *Model: $P(y_i = 1|x_i) = logit^{-1}(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2})$*
- *Prior: $\beta_j \sim \mathcal{N}(0, 10)$ for all j*

Tasks:

1. *Implement Metropolis-Hastings for the posterior $p(\beta|y, X)$*
2. *Run multiple chains from different starting points*
3. *Check convergence using $\hat{R}$ statistic*
4. *Compute posterior means and 95% credible intervals*

5. *Compare to maximum likelihood estimates*
6. *Make predictions and plot posterior predictive distribution*

Problem 5 (Hierarchical Model)**.** *Implement MCMC for a hierarchical (mixed effects) model.*
Model:

$$\begin{aligned} y_{ij} &\sim \mathcal{N}(\mu_i + \beta x_{ij}, \sigma^2) \quad \textit{(likelihood)} \\ \mu_i &\sim \mathcal{N}(\mu_0, \tau^2) \quad \textit{(group effects)} \\ \mu_0 &\sim \mathcal{N}(0, 100) \quad \textit{(hyperprior)} \\ \beta &\sim \mathcal{N}(0, 10) \\ \sigma^2, \tau^2 &\sim \textit{Inv-Gamma}(1, 1) \end{aligned}$$

where $i = 1, \ldots, J$ indexes groups and $j = 1, \ldots, n_i$ indexes observations within groups.
Tasks:

1. *Simulate data with $J = 10$ groups, $n_i = 20$ observations each*
2. *Implement Gibbs sampler (derive full conditionals)*
3. *Estimate all parameters*
4. *Compute shrinkage: compare μ_i to group-specific means*
5. *Visualize posterior distributions*

4 Advanced Topics

Problem 6 (Challenging Distributions)**.** *Test your MCMC implementations on challenging target distributions:*

(a) ***Neal's Funnel:***

$$\begin{aligned} v &\sim \mathcal{N}(0, 9) \\ x_i | v &\sim \mathcal{N}(0, e^v) \quad i = 1, \ldots, 9 \end{aligned}$$

(b) ***Rosenbrock's Banana:***

$$\pi(x_1, x_2) \propto \exp\left(-\frac{x_1^2}{200} - \frac{1}{2}(x_2 - x_1^2)^2\right)$$

(c) ***Multimodal Distribution:***

$$\pi(x) = 0.3\mathcal{N}(-5, 1) + 0.3\mathcal{N}(0, 0.5) + 0.4\mathcal{N}(5, 1)$$

For each distribution:

- *Try standard Metropolis-Hastings*
- *Try Hamiltonian Monte Carlo (where applicable)*
- *Document challenges encountered*
- *Propose solutions or modifications*
- *Compare mixing and convergence*

Problem 7 (Tempering)**.** *Implement parallel tempering to improve mixing for multimodal distributions.*

Algorithm:

1. *Run K chains in parallel with temperatures $T_1 = 1 < T_2 < \cdots < T_K$*
2. *Each chain targets $\pi^{1/T_k}(x)$*
3. *Periodically propose swaps between adjacent chains*
4. *Accept swaps with Metropolis-Hastings criterion*

Tasks:

1. *Implement parallel tempering*
2. *Test on the multimodal distribution from Problem 6*
3. *Use $K = 5$ chains with temperatures $[1.0, 1.5, 2.5, 5.0, 10.0]$*
4. *Track swap acceptance rates*
5. *Compare to standard MCMC (Problem 6c)*

5 Computational Considerations

Problem 8 (Vectorization and Efficiency)**.** *Optimize your MCMC implementations for performance.*

Tasks:

1. *Profile your Metropolis-Hastings code to identify bottlenecks*
2. *Vectorize operations where possible (avoid loops)*
3. *Compare runtime for:*
   - *Naive Python implementation*
   - *Vectorized NumPy implementation*
   - *Numba JIT-compiled version (bonus)*
4. *Generate timing benchmarks for different sample sizes*
5. *Discuss memory vs. speed tradeoffs*

6 Submission Guidelines

6.1 Format

- Submit as Jupyter notebook (.ipynb) or R Markdown (.Rmd)
- Include all code, outputs, and plots
- Add markdown cells with explanations
- Code should be well-commented and readable

6.2 Evaluation Criteria

- **Correctness (40%):** Algorithms implemented correctly
- **Analysis (30%):** Thoughtful interpretation of results
- **Presentation (20%):** Clear code and explanations
- **Creativity (10%):** Additional insights or extensions

6.3 Resources

- Course implementations: `code/mcmc/`
- Bibliography: `shared/bibliographies/mcmc_references.bib`
- Documentation: GitHub Repository

7 Recommended Reading

1. Metropolis et al. (1953). "Equation of state calculations by fast computing machines"
2. Hastings (1970). "Monte Carlo sampling methods using Markov chains"
3. Gelman & Rubin (1992). "Inference from iterative simulation"
4. Neal (2011). "MCMC using Hamiltonian dynamics"
5. Brooks et al. (2011). "Handbook of Markov Chain Monte Carlo"

Contact: Diogo Ribeiro, dfr@esmad.ipp.pt
Office Hours: By appointment
Course Materials: Available on repository