Lecture Notes for STAT546: Computational Statistics

—Lecture 3: Monte Carlo

Faming Liang

Purdue University

August 23, 2026

Hit-and-Run algorithm:

1. Draw $d \sim g(d)$ $(d \in \mathbb{O})$ and $\lambda \sim l(\lambda|d, x)$ over $\mathcal{X}_{x,d}$, and compute an MH acceptance probability $\alpha(x, y)$, where $x = X^{(t)}$.
2. Generate U from Unif[0,1] and set

$$X^{(t+1)} = \begin{cases} x + \lambda d, & \text{if } U \leq \alpha(x, y); \\ x, & \text{otherwise.} \end{cases}$$

Chen *et al.* (2000) note that the most common choice of $g(d)$ is the uniform distribution on $\mathbb{O}$. They also discuss common choices for $g(.|x, d)$ and $\alpha(x, y)$. It has been recognized by Berger (1993) that Hit-and-Run is particularly useful for the problems with a sharply constrained parameter space.

This algorithm is rooted in the Langevin diffusion process, which is defined by a stochastic differential equation

$$dX_t = dB_t + \frac{1}{2}\nabla \log f(X_t), \tag{1}$$

where B_t is the standard Brownian motion. It is known this process leaves f as the stationary distribution. The actual implementation of the diffusion algorithm involves a discretiation step which replaces (1) by a random-walk like transition

$$x^{(t+1)} = x^{(t)} + \frac{\sigma^2}{2}\nabla \log f(x^{(t)}) + \sigma\epsilon_t, \tag{2}$$

where $\epsilon_t \sim N_d(0, I_d)$ and σ is the step size of discretization. However, as shown by Roberts and Tweedie (1995), the discretized process may be transient and is no longer reversible with respect to f.

To correct this negative behavior, Besag (1994) suggested to moderate the discretization step by the MH acceptance-rejection rule; that is, treating (2) as a conventional MH proposal distribution. In summary, one iteration of the Langevin algorithm can be described as follows:

1. Propose a new state by setting

$$x^* = x^{(t)} + \frac{\sigma^2}{2}\nabla \log f(x^{(t)}) + \sigma\epsilon_t,$$

   where σ is a user-specified parameter.

2. Calculate the MH ratio

$$r = \frac{f(x^*)}{f(x^{(t)})} \frac{\exp(-\|x^{(t)} - x^* - \frac{\sigma^2}{2}\nabla \log f(x^*)\|^2/2\sigma^2)}{\exp(-\|x^* - x^{(t)} - \frac{\sigma^2}{2}\nabla \log f(x^{(t)})\|^2/2\sigma^2)}.$$

   Set $x^{(t+1)} = x^*$ with probability $\min(1, r)$, and set set $x^{(t+1)} = x^{(t)}$ with the remaining probability.

Langevin Dynamics:

$$\Delta\theta_t = \frac{\epsilon}{2}\left(\nabla \log p(\theta_t) + \sum_{x \in D} \nabla \log p(x|\theta_t)\right) + \eta_t,$$

$$\eta_t = N(0, \epsilon).$$

- The noise variance ϵ is proportional to the learning rate.
- Decreasing ϵ also decrease the discretization error.
- Converges to the posterior as $\epsilon \to 0$.

Stochastic gradient Langevin Dynamics:

$$\Delta\theta_t = \frac{\epsilon_t}{2}\left(\nabla \log p(\theta_t) + \frac{|\tilde{D}|}{|D|}\sum_{x\in\tilde{D}} \nabla \log p(x|\theta_t)\right) + \eta_t,$$

$$\eta_t = N(0, \epsilon_t).$$

- The learning rate ϵ_t follows a rule similar to the Robbins-Monro algorithm: $\epsilon_t \to 0$ and $\sum_{t=1}^{\infty} \epsilon_t \to \infty$.
- Metropolos correction will no longer be needed!

- During training, we collect a set of samples $\{\theta_1, \theta_2, \ldots, \theta_T\}$.
- We will use them to approximate an expectation $E[h(\theta)]$.
- Simple sample averaging

$$\frac{1}{T}\sum_{t=1}^{T} h(\theta_t)$$

  It works if $\epsilon_t \succ 1/t$, e.g., $\epsilon_t = O(1/t^{\alpha})$ for some $0 < \alpha < 1$ (Song, Sun, Ye, and Liang, 2020, Biometrika).
- Alternatively, do weighted averaging

$$\frac{\sum_{t=1}^{T} \epsilon_t h(\theta_t)}{\sum_{t=1}^{T} \epsilon_t}$$

For the Metropolis-Hastings transition rule based Markov chain, its efficiency mainly depends on the proposal distribution.
Let $\lambda(x, y)$ be a nonnegative symmetric function in x and y.
Suppose that $\lambda(x, y) > 0$ whenever $q(y|x) > 0$. Define

$$w(x, y) = f(x)q(y|x)\lambda(x, y). \tag{3}$$

Multiple-Try Metropolis:

Current is at $\boldsymbol{x}$

(a) Draw $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_k$ from the proposal $T(\boldsymbol{x} \to \boldsymbol{y})$.

(b) Select $\boldsymbol{y} = \boldsymbol{y}_j$ with probability $\propto w(\boldsymbol{y}_j, \boldsymbol{x})$.

(c) Draw $\boldsymbol{x}_1^*, \ldots, \boldsymbol{x}_{k-1}^*$ from $T(\boldsymbol{y} \to \boldsymbol{x})$. Let $\boldsymbol{x}_k^* = \boldsymbol{x}$.

(d) Accept the proposed $\boldsymbol{y}$ with probability

$$p = \min\{1, \frac{w(\boldsymbol{y}_1, \boldsymbol{x}) + \cdots + w(\boldsymbol{y}_k, \boldsymbol{x})}{w(\boldsymbol{x}_1^*, \boldsymbol{y}) + \cdots + w(\boldsymbol{x}_k^*, \boldsymbol{y})}\}$$

See Liu, Liang and Wong (2000, JASA) for details.
When $q(x|y)$ is symmetric, for example, one can choose $\lambda(x, y) = 1/q(y|x)$, and then $w(x, y) = f(x)$. In this case, the MTM algorithm is reduced to the orientational bias Monte Carlo algorithm used in the field of molecular simulation.

Reversible Jump MCMC:

Consider the problem of Bayesian model selection. Let $\{\mathcal{M}_k : k \in \mathcal{K}\}$ denote a countable collection of models to be fitted to the observed data Y. Each model $\mathcal{M}_k$ has its own parameter space $\Theta_k \subseteq \mathbb{R}^{d_k}$. Without loss of generality, we assume here that different models have different dimensions. A full Bayesian model can be written as

$$p(k)p(\theta_k|k)p(Y|k,\theta_k), \tag{4}$$

where $p(k)$ is the prior probability imposed on the model $\mathcal{M}_k$, $p(\theta_k|k)$ is the prior distribution imposed on the parameter θ_k, and $p(Y|k,\theta_k)$ represents the sampling model for the observed data Y.

RJMCMC takes the auxiliary variable approach to the problem of "dimension matching". Let $x_t = (k^{(t)}, \theta_k^{(t)})$ denote the current state. Suppose that $x^* = (k^*, \theta_{k^*}^*)$ is the proposed state for $X^{(t+1)}$. If $k^* = k$, the proposed move explores different locations within the same subspace $\mathcal{X}_k$ and therefore the dimension-matching problem does not exist. If $k^* \neq k$, generate s random variables $\boldsymbol{u} = (u_1, \ldots, u_s)$ from a distribution $\psi_{k^{(t)} \to k^*}(u)$, and consider a bijection

$$(\theta_{k^*}^*, \boldsymbol{u}^*) = T(\theta_k^{(t)}, \boldsymbol{u}), \tag{5}$$

where $\boldsymbol{u}^* = (u_1, \ldots, u_{s^*})$ is a random vector of s^*-dimension, and s and s^* satisfy the dimension-matching condition $s + d_{k^{(t)}} = s^* + d_{k^*}$. The general idea of "augmenting less" suggests that $s^* = 0$ be taken if $d_{k^{(t)}} \leq d_{k^*}$ and vice versa.

Reversible jump MCMC Algorithm

1. Select model $\mathcal{M}_{k^*}$ with probability $q(k^{(t)}, k^*)$.
2. Generate $u_1, \ldots, u_s \sim \psi_{k^{(t)} \to k^*}(u)$.
3. Set $(\theta_{k^*}, u^*) = T(\theta_k^{(t)}, u)$.
4. Compute the MH ratio

$$r = \frac{f(k^*, \theta_{k^*}^* | Y) q(k^*, k^{(t)}) \psi_{k^* \to k^{(t)}}(\boldsymbol{u}^*)}{f(k^{(t)}, \theta_k^{(t)} | Y) q(k^{(t)}, k^*) \psi_{k^{(t)} \to k^*}(\boldsymbol{u})} \left| \frac{\partial(\theta_{k^*}^*, \boldsymbol{u}^*)}{\partial(\theta_k^{(t)}, \boldsymbol{u})} \right| \quad (6)$$

   where $\partial(\theta_{k^*}^*, \boldsymbol{u}^*)/\partial(\theta_k^{(t)}, \boldsymbol{u})$ is the Jacobian of the transformation of (5).
5. Set $X^{(t+1)} = (k^*, \theta_{k*}^*)$ with probability $\min(1, r)$ and $X^{(t+1)} = X_t$ with the remaining probability.

Let $\boldsymbol{Z} = (z_1, \ldots, z_n)$ denote a sequence of independent observations drawn from a mixture distribution with the likelihood function

$$f(\boldsymbol{Z}|m, \boldsymbol{p}_m, \Phi_m, \eta) = \prod_{i=1}^{n} \left[p_1 f(z_i; \phi_1, \eta) + \cdots + p_m f(z_i; \phi_m, \eta) \right],$$

where m is the unknown number of components, $\boldsymbol{p}_m = (p_1, \ldots, p_m)$, $\Phi_m = (\phi_1, \ldots, \phi_m)$, and η is a common parameter vector for all components.

The posterior distribution of $(k, \boldsymbol{p}_k, \Phi_k, \eta)$ is then

$$\pi(k, \boldsymbol{p}_k, \Phi_k, \eta | \boldsymbol{Z}) \propto f(\boldsymbol{Z} | k, \boldsymbol{p}_k, \Phi_k, \eta) \pi(k, \boldsymbol{p}_k, \Phi_k, \eta). \qquad (7)$$

For simulating from (7), RJMCMC consists of three types of moves, "birth", "death", and "parameter updating", which can be prescribed as in Stephens (2000). In the "birth" move, a new component is generated and $(\boldsymbol{p}_k, \Phi_k)$ is updated to be

$$\{(p_1(1-p), \phi_1), \ldots, (p_k(1-p), \phi_k), (p, \phi)\}$$

In the "death" move, a component, say component i, is randomly chosen to remove and $(\boldsymbol{p}_k, \Phi_k)$ is updated to be

$$\left\{ (\frac{p_1}{1-p_i}, \phi_1), \ldots, (\frac{p_{i-1}}{1-p_i}, \phi_{i-1}), (\frac{p_{i+1}}{1-p_i}, \phi_{i+1}), \ldots, (\frac{p_k}{1-p_i}, \phi_k) \right\}.$$

In the "parameter updating" move, the parameters $(\boldsymbol{p}_k, \Phi_k, \eta)$ are updated using the MH algorithm.

- Propose a value of k^* according to a stochastic matrix Q, where, for example, we set $Q_{k,k+1} = Q_{k,k-1} = Q_{k,k} = 1/3$ for $K_{\min} < k < K_{\max}$, $Q_{K_{\min},K_{\min}+1} = Q_{K_{\max},K_{\max}-1} = 1/3$, and $Q_{K_{\min},K_{\min}} = Q_{K_{\max},K_{\max}} = 2/3$.

- According to the value of k^*, do step (a), (b) or (c):
  (a) If $k^* = k + 1$, make a "birth" move: Draw $p \sim Unif[0, 1]$ and draw ϕ from a proposal distribution $g(\phi|\boldsymbol{p}_k, \Phi_k, \eta)$, and accept the new component with probability

$$\min\left\{1, \frac{\pi(k+1, \boldsymbol{p}_{k+1}, \Phi_{k+1}, \eta|\boldsymbol{Z})}{\pi(k, \boldsymbol{p}_k, \Phi_k, \eta|\boldsymbol{Z})} \frac{Q_{k+1,k}}{Q_{k,k+1}} \frac{1}{(k+1)g(\phi|\boldsymbol{p}_k, \Phi_k, \eta)}\right\}.$$

  (b) If $k^* = k - 1$, make a "death" move: Randomly choose a component, say component i, to remove, and accept the new state with probability

$$\min\left\{1, \frac{\pi(k-1, \boldsymbol{p}_{k-1}, \Phi_{k-1}, \eta|\boldsymbol{Z})}{\pi(k, \boldsymbol{p}_k, \Phi_k, \eta|\boldsymbol{Z})} \frac{Q_{k-1,k}}{Q_{k,k-1}} \frac{kg(\phi_i|\boldsymbol{p}_{k-1}, \Phi_{k-1}, \eta)}{1}\right\}.$$

(c) If $k^* = k$, make a "parameter updating" move, updating the parameters $(\boldsymbol{p}_k, \Phi_k, \eta)$ using the MH algorithm: Generate $(\boldsymbol{p}_k^*, \Phi^*, \eta^*)$ from a proposal distribution $q(\boldsymbol{p}_k^*, \Phi^*, \eta^* | \boldsymbol{p}_k, \Phi, \eta)$, and accept the proposal with probability

$$\min \left\{ 1, \frac{\pi(k, \boldsymbol{p}_k^*, \Phi_k^*, \eta^* | \boldsymbol{Z})}{\pi(k, \boldsymbol{p}_k, \Phi_k, \eta | \boldsymbol{Z})} \frac{q(\boldsymbol{p}_k, \Phi_k, \eta | \boldsymbol{p}_k^*, \Phi^*, \eta^*)}{q(\boldsymbol{p}_k^*, \Phi^*, \eta^* | \boldsymbol{p}_k, \Phi, \eta)} \right\}.$$

Metropolis-within-Gibbs sampler

Consider the Gibbs algorithm described in Lecture 2. When some some components cannot be easily simulated, rather than resorting to a customized algorithm such as the acceptance-rejection algorithm in each of these cases, Müller (1991, 1993) suggested a compromised Gibbs algorithm—the Metropolis-within-Gibbs sampler. In any step of the Gibbs sampler with difficulty in sampling from $f_k(x_k|x_i, i \neq k)$, substitute a MH simulation. Müller's algorithm can be described as follows:

Metropolis-within-Gibbs Sampler

For $i = 1, \ldots, K$, given $(x_1^{(t+1)}, \ldots, x_{i-1}^{(t+1)}, x_i^{(t)}, \ldots, x_K^{(t)})$:

1. Generate $x_i^* \sim q_i(x_i | x_1^{(t+1)}, \ldots, x_{i-1}^{(t+1)}, x_i^{(t)}, \ldots, x_K^{(t)})$.
2. Calculate

$$r = \frac{f_i(x_i^* | x_1^{(t+1)}, \ldots, x_{i-1}^{(t+1)}, x_{i+1}^{(t)}, \ldots, x_K^{(t)})}{f_i(x_i^{(t)} | x_1^{(t+1)}, \ldots, x_{i-1}^{(t+1)}, x_{i+1}^{(t)}, \ldots, x_K^{(t)})} \times \frac{q_i(x_i^{(t)} | x_1^{(t+1)}, \ldots, x_{i-1}^{(t+1)}, x_i^*, x_{i+1}^{(t)}, \ldots, x_K^{(t)})}{q_i(x_i^* | x_1^{(t+1)}, \ldots, x_{i-1}^{(t+1)}, x_i^{(t)}, x_{i+1}^{(t)}, \ldots, x_K^{(t)})}.$$

3. Set $x_i^{(t+1)} = x_i^*$ with probability $\min(1, r)$ and $x_i^{(t+1)} = x_i^{(t)}$ with the remaining probability.

An important point about this substitution is that the MH step is only performed once at each iteration, while $f(x_1, \ldots, x_K)$ still being admitted as the stationary distribution. One may wonder whether one should run multiple MH steps to produce a precise approximation of $f_i(\cdot)$. As noted by Chen and Schmeiser (1998), this is not necessary. A precise approximation of $f_i(\cdot)$ does not necessarily lead to a better approximation of $f(\cdot)$, and a single step substitution may be beneficial for the speed of excursion of the chain on the sample space of $f(\cdot)$.