Statistical Models & Computing Methods

Lecture 5: Advanced MCMC

Cheng Zhang
School of Mathematical Sciences, Peking University

November 5, 2020
Simple MCMC methods, such as Metropolis algorithm and Gibbs sampler explore the posterior distribution using simple mechanism (e.g., a random walk).

While this strategy might work well for low-dimensional distributions, it could become very inefficient (e.g., high autocorrelation, missing isolated modes) for high-dimensional distributions.

In this lecture, we discuss several advanced techniques to improve the efficiency of MCMC methods.
Auxiliary variable strategies can be used to improving mixing of Markov chains

When standard MCMC methods mix poorly, one potential remedy is to augment the state space of the variable of interest

This approach can lead to chains that mix faster and require less tuning than the standard MCMC methods

Main idea: construct a Markov chain over \((X, U)\) (\(U\) is the auxiliary variable) with stationary distribution marginalizes to the target distribution of \(X\)

As we will see later, this includes a large family of modern MCMC methods
Suppose that we have a challenging target distribution $f(x) \propto \exp(-U(x))$

We can introduce temperatures to construct a sequence of distributions that are easier to sample from

$$f_k(x) \propto \exp \left( -U(x)/T_k \right), \quad k = 0, \ldots, K$$

where $1 = T_0 < T_1 < \ldots < T_K$

When simulating Markov chains with different temperature $T$, the chain with high temperature (hot chain) is likely to mix better than the chain with cold temperature (cold chain)

Therefore, we can run parallel chains and swap states between the chains to improve mixing
Double-well Potential Distribution

\[ f_T(x) \propto \exp\left(-\left(x^2 - 1\right)^2/T\right), \quad T = 1/\gamma \]
We run parallel Markov chains for distributions with different temperatures. In each iteration

- Follow regular Metropolis steps in each chain to get new states $x_0^{(t)}, \ldots, x_K^{(t)}$
- Select two temperatures, say $(i, j), i < j$, and swap the states

$$x_0^{(t)}, \ldots, x_i^{(t)}, \ldots, x_j^{(t)}, \ldots x_K^{(t)} \rightarrow x_0^{(t)}, \ldots, x_j^{(t)}, \ldots, x_i^{(t)}, \ldots x_K^{(t)}$$

- Accept the swapped new states with the following probability

$$\min \left( 1, \frac{f_i(x_j^{(t)}) f_j(x_i^{(t)})}{f_i(x_i^{(t)}) f_j(x_j^{(t)})} \right)$$
Both the within-chain Metropolis updates and the between-chain swap preserves

\[ p(x_0, \ldots, x_K) \propto f_0(x_0)f_1(x_1)\cdots f_K(x_K) \]

Therefore, the joint distribution of \((x_0^{(t)}, \ldots, x_K^{(t)})\) will converge to \(p(x)\), and the marginal distribution of \(x_0\) (cold chain) is the target distribution.

There are many ways to swap chains. For example, we can pick a pair of temperatures uniformly at random or only swap chains with successive temperatures.

The design of temperature levels could be crucial for the performance.
Example: Double-well Potential Distribution
Slice sampling was introduced by Neal (2003) to accelerate mixing of Metropolis (or MH)

It is essentially a Gibbs sampler in the augmented space $(X, U)$ with density

$$f(x, u) = f(x)f(u|x)$$

where $U$ is the auxiliary variable and $f(u|x)$ is designed to be a uniform distribution $\mathcal{U}(0, f(x))$
For this purpose, slice sampling alternates between two steps:

- Given the current state of the Markov chain, $x$, we uniformly sample a new point $u$ from the interval $(0, f(x))$

$$U|x \sim \mathcal{U}(0, f(x))$$

- Given the current value of $u$, we uniformly sample from the region $S = \{x : f(x) > u\}$, which is referred to as the slice defined by $u$

$$X|u \sim \mathcal{U}(S)$$

- As mentioned by Neal (2003), in practice it is safer to compute $g(x) = \log(f(x))$, and use the auxiliary variable $z = \log(u) = g(x) - e$, where $e$ has exponential distribution with mean one, and define the slice as $S = \{x : z < g(x)\}$
Move Between Modes with Slice Sampler

- One advantage of slice sampling is for sampling from multimodal distributions.
- Unlike standard Metropolis (or MH) that struggles between distant modes, sampling from the slice allows us to easily jump between different modes.

![Diagram showing slice sampling](image)
Sampling an independent point uniformly from $S$ might be difficult. In practice, we can substitute this step by any update that leaves the uniform distribution over $S$ invariant.

There are several methods to perform this task.

Here, we introduce a simple but effective procedure that consists of two phases:

- **Stepping-out.** A procedure for finding an interval around the current point.
- **Shrinkage.** A procedure for sampling from the interval obtained.

For a detail description of these methods, see Neal (2003).
Sampling $u \sim \mathcal{U}(0, f(x_0))$ and stepping out (of size $w$) until we reach points outside the slice.
Shrinkage of interval to a point, $x'$, which is sampled (uniformly) from the interval but it has $f(x') < y$
Continue shrinkage until we reach a point $x_1$ such that $y < f(x_1)$. We accept $x_1$ as our new sample.
Random walk Metropolis (RWM) is struggling with a banana-shaped distribution
Simple MCMC is Not Enough

Random walk Metropolis (RWM) is struggling with a banana-shaped distribution
Random proposals are likely to be inefficient, since they completely ignore the target distribution.

A better way would be to use information from the target distribution to guide our proposals.

Note that in optimization, the gradient points to an ascent direction, which would also be useful when designing the proposal distributions.

\[ x' = x + \epsilon \nabla \log p(x) \]

when \( \epsilon \) is small,

\[ \log p(x') > \log p(x) \]
We can incorporate the gradient information into our proposal distribution.

Let $x$ be the current state, instead of using a random perturbation centered at $x$ (e.g., $\mathcal{N}(x, \sigma^2)$), we can shift toward the gradient direction which leads to the following proposal distribution:

$$Q(x' | x) = \mathcal{N}(x + \frac{\sigma^2}{2} \nabla \log p(x), \sigma^2 I)$$

This looks like GD with noise!

No longer symmetric, use Metropolis-Hasting instead.

This is called **Metropolis Adjusted Langevin Algorithm (MALA)**.
Metropolis Adjusted Langevin Algorithm

RWM

MALA
It turns out that we can combine multiple MALA together, resulting in an algorithm that can generate distant proposals with high acceptance rate.

The new algorithm is based on Hamiltonian dynamics, a system introduced by Alder and Wainwright (1959) to simulate motion of molecules deterministically based on Newton’s law of motion.

In 1987, Duane et al. combine the standard MCMC and the Hamiltonian dynamics, and derived a method they called Hybrid Monte Carlo (HMC).

Nowadays, this abbreviation has also been used for Hamiltonian Monte Carlo.
Construct a landscape with potential energy $U(x)$

$$p(x) \propto e^{-U(x)}, \quad U(x) = - \log P(x)$$

Introduce momentum $r$ carrying kinetic energy $K(r) = \frac{1}{2} r^T M^{-1} r$, and define total energy or Hamiltonian $H(x, r) = U(x) + K(r)$

Hamiltonian equations

$$\frac{dx}{dt} = \frac{\partial H}{\partial r}, \quad \frac{dr}{dt} = - \frac{\partial H}{\partial x}$$

Some physics:

- The two equations are about velocity and force, respectively.
- Frictionless ball rolling $(x, r) \rightarrow (x', r')$ satisfies $H(x', r') = H(x, r)$
The joint probability of \((x, r)\) is
\[
p(x, r) \propto \exp(-H(x, r)) \propto p(x) \cdot \mathcal{N}(r|0, M)
\]

- \(x\) and \(r\) are independent and \(r\) follows a Gaussian distribution
- The marginal distribution is the target distribution \(p(x)\)
- We then use MH to sample from the joint parameter space and \(x\) samples are collected as samples from the target distribution
- HMC is an auxiliary variable method
We follow two steps to make proposals in the joint parameter space

- Gibbs sample momentum: \( r \sim \mathcal{N}(0, M) \)
- Simulate Hamiltonian dynamics and flip the sign of the momentum

\[
(x, r) = (x^{(0)}, r^{(0)}) \xrightarrow{\text{HD}} (x^{(t)}, r^{(t)}), \quad (x', r') = (x^{(t)}, -r^{(t)})
\]

Important Properties

- **Time reversibility**: The trajectory is time reversible
- **Volume preservation**: Hamiltonian flow does not change the volume - the jacobian determinant is 1
- **Conservation of Hamiltonian**: Total energy is conserved, meaning the proposal will always be accepted
In practice, Hamiltonian dynamics can not be simulated exactly. We need to use numerical integrators

**Leap-frog scheme**

\[
\begin{align*}
  r(t + \frac{\epsilon}{2}) &= r(t) - \frac{\epsilon}{2} \frac{\partial U}{\partial x}(x(t)) \\
  x(t + \epsilon) &= x(t) + \epsilon \frac{\partial K}{\partial r}(r(t + \frac{\epsilon}{2})) \\
  r(t + \epsilon) &= r(t + \epsilon/2) - \frac{\epsilon}{2} \frac{\partial U}{\partial x}(x(t + \epsilon))
\end{align*}
\]

**Important Properties**

- **Reversibility and volume preservation**: still hold
- **Conservation of Hamiltonian**: broken. Acceptance probability becomes

\[
a(x', r'|x, r) = \min \left(1, \exp(-H(x', r') + H(x, r))\right)
\]
Comparison of Numerical Integrators

\[ H(x, r) = \frac{x^2}{2} + \frac{r^2}{2} \]

Euler, \( \epsilon = 0.3 \)  

Leap-frog, \( \epsilon = 0.3 \)

Adapted from Neal (2011)
Hamiltonian Monte Carlo

HMC in one iteration

- Sample momentum $r \sim \mathcal{N}(0, M)$
- Run numerical integrators (e.g., leapfrog) for $L$ steps
- Accept new position with probability

$$\min \left(1, \exp(-H(x', r') + H(x, r))\right)$$
HMC in one iteration

- Sample momentum $r \sim \mathcal{N}(0, M)$
- Run numerical integrators (e.g., leapfrog) for $L$ steps
- Accept new position with probability

$$\min \left(1, \exp(-H(x', r') + H(x, r))\right)$$
Since Hamiltonian is conserved, every Hamiltonian trajectory is confined to an energy level set

\[ H^{-1}(E) = \{ x, r | H(x, r) = E \} \]

Adapted from Betancourt (2017)
Since Hamiltonian is conserved, every Hamiltonian trajectory is confined to an energy level set

$$H^{-1}(E) = \{x, r | H(x, r) = E\}$$

Adapted from Betancourt (2017)
The choice of the conditional probability distribution over the momentum, or equivalently, the kinetic energy, affects HMC’s behavior over different energy level sets.

Ideally, the kinetic energy will interact with the target distribution to ensure that the energy level sets are uniformly distributed.

In HMC, we often use Euclidean-Gaussian kinetic energy $K(r) = \frac{r^T r}{2}$. This sets $M = I$ and completely ignore local geometric information of the target distribution.

Preconditioning mass matrix may help, but it is also quite limited.

Instead of using a fixed $M$, how about using an adaptive one?
Consider the symmetric KL divergence between two densities $p$ and $q$

\[ D_{\text{KL}}^S(p\|q) = D_{\text{KL}}(p\|q) + D_{\text{KL}}(q\|p) \]

Let $p(y|x)$ be the likelihood. Then

\[ D_{\text{KL}}^S(p(y|x + \delta x)\|p(y|x)) \] is approximately

\[ \delta x^T \mathbb{E}_{y|x} (\nabla_x \log p(y|x) \nabla_x \log p(y|x)^T) \delta x = \delta x^T G(x) \delta x \]

where $G(x)$ is the **Fisher Information** matrix

This induces a **Riemannian manifold** (Amari 2000) over the parameter space of a statistical model, which defines the **natural geometric structure** of density $p(x)$
Based on the Riemannian manifold formulation, Girolami and Calderhead (2011) introduce a new method, called Riemannian manifold HMC (RMHMC).

Hamiltonian on a Riemannian manifold

\[ H(x, r) = U(x) + \frac{1}{2} \log((2\pi)^d |G(x)|) + \frac{1}{2} r^T G(x)^{-1} r \]

The joint probability is

\[ p(x, r) \propto \exp(-H(x, r)) \propto p(x) \cdot \mathcal{N}(r|0, G(x)) \]

\( x \) and \( r \) now are correlated, and the conditional distribution of \( r \) given \( x \) follows a Gaussian distribution.

The marginal distribution is the target distribution.
The resulting dynamics is non-separable, so instead of the standard leapfrog we need to use the *generalized* leapfrog method (Leimkuhler and Reich, 2004)

**The generalized leapfrog scheme**

\[
\begin{align*}
  r(t + \frac{\epsilon}{2}) &= r(t) - \frac{\epsilon}{2} \nabla_x H(x(t), r(t + \frac{\epsilon}{2})) \\
  x(t + \epsilon) &= x(t) + \frac{\epsilon}{2} \left( G(x(t))^{-1} + G(x(t + \epsilon))^{-1} \right) r(t + \frac{\epsilon}{2}) \\
  r(t + \epsilon) &= r(t + \frac{\epsilon}{2}) - \frac{\epsilon}{2} \nabla_x H(x(t + \epsilon), r(t + \frac{\epsilon}{2}))
\end{align*}
\]

The above scheme is time reversible and volume preserving. However, the first two equations are defined implicitly (can be solved via several fixed point iterations)
Consider a 2D banana-shaped posterior distribution as follows

\[ y_i \sim \mathcal{N}(\theta_1 + \theta_2^2, \sigma_y^2), \quad \theta = (\theta_1, \theta_2) \sim \mathcal{N}(0, \sigma_\theta^2 I) \]

the log-posterior is (up to an ignorable constant)

\[
\log p(\theta|Y, \sigma_y^2, \sigma_\theta^2) = - \sum_i (y_i - \theta_1 - \theta_2^2)^2 \frac{2}{2\sigma_y^2} - \theta_1^2 + \theta_2^2 \frac{2}{2\sigma_\theta^2}
\]

Fisher information for the joint likelihood

\[
G(\theta) = \mathbb{E}_{Y|\theta} \left( -\nabla_\theta^2 \log p(Y, \theta) \right) = \frac{n}{\sigma_y^2} \begin{bmatrix} 1 & 2\theta_2 \\ 2\theta_2 & 4\theta_2^2 \end{bmatrix} + \frac{1}{\sigma_\theta^2} I
\]
Examples: Banana Shape Distribution

[Graphs showing HMC and RMHMC with distributions and trajectories in 2D space.

HMC (left) and RMHMC (right) with theta axes.

\( \theta_1 \) and \( \theta_2 \) ranges from -2.0 to 2.0.]

HMC

RMHMC
Consider a Bayesian logistic regression model with design matrix $X$ and regression coefficients $\beta \in \mathbb{R}^d$, with a simple prior $\beta \sim \mathcal{N}(0, \alpha I_d)$.

Neglecting constants, the log-posterior is

$$
\log p(\beta|X, Y, \alpha) = L(\beta) - \frac{1}{2\alpha} \beta^T \beta \\
= \beta^T X^T Y - \sum_i \log(1 + \exp(x_i^T \beta)) - \frac{1}{2\alpha} \beta^T \beta
$$

Use the joint likelihood to compute the fisher information

$$
G(\beta) = \mathbb{E}_{Y|X, \beta, \alpha} \left( -\nabla^2_\beta L(\beta) + \frac{1}{\alpha} I_d \right) = X^T W X + \frac{1}{\alpha} I_d
$$
Examples: Bayesian Logistic Regression

Adapted form Girolami and Calderhead (2011)
Choice of Integration Time

Integration time determines the exploration efficiency of Hamiltonian trajectory in each energy level set

- Too short integration time lose the advantage of the coherent exploration of the Hamiltonian trajectory (e.g., one step HMC is equivalent to MALA)
- Too long integration time wastes computation since trajectories are likely to return to explored regions

The No-U-Turn Sampler (Hoffman and Gelman, 2011).

- Idea: use the distance to the initial position as a criteria for selecting integration time - avoid U-Turn
- Naive implementation is not time reversible. Use a strategy similar to the doubling procedure in slice sampling (Neal 2003).
Generally speaking, the efficiency of MCMC depends on its proposal distribution, which usually involves several hyper-parameters.

Most MCMC algorithms, therefore, need tuning to be efficient and reliable in large scale applications.

However, tuning could be painful and sometimes not practical (requires computing time, human time, and typically expert knowledge, too many variables, when to stop tuning, tuning criterion not clear, etc).

Adaptive MCMC is about tuning MCMC without human intervention.

It uses the trajectory so far to tune the sampling kernel on the fly (so it is not a Markov chain anymore).
Examples: Random Walk Metropolis

- Proposal distribution:

\[ x' \sim Q_\sigma(\cdot|x) = x + \sigma \mathcal{N}(0, I_d) \]

- Plots for different \( \sigma \) - Goldilock’s principle
Random Scan Gibbs Sampler for 50-d Truncated Multivariate Normals. Are uniform $1/d$ selection probabilities optimal?
First, we need a parameterized family of proposal distributions for a given MCMC class.

We also need an optimization rule that is mathematically sound and computationally cheap.

We need it to work in practice.

Ergodicity of Adaptive MCMC

How do we know that the chain will converge to the target distribution if it is not even Markovian?

Two conditions (see Roberts and Rosenthal 2007):

- **Diminishing adaption**: the dependency on earlier states of the chain goes to zero.
- **Bounded convergence**: convergence times for all adapted transition kernels are bounded in probability.
Consider random walk Metropolis for a $d$-dimensional target distribution with proposal $Q(x'|x_n) = \mathcal{N}(x_n, \sigma^2 \Sigma^{(n)})$.

If the target distribution is Gaussian with covariance $\Sigma$, the optimal proposal is $\mathcal{N}(x_n, \frac{2.38^2}{d} \Sigma)$, which leads to an acceptance rate $\alpha^* \approx 0.23$ (see Gelman et al 1996).

This gives a simple criterion for random walk Metropolis in practice.

We can use it to design an adaptive Metropolis algorithm.
Adaptive Scaling Algorithm

- Draw proposal
  \[ x' \sim Q(\cdot|x_n) = x_n + \sigma_n \mathcal{N}(0, I_d) \]

- Select the value \( x_{n+1} \) according to the Metropolis acceptance rate
  \[ \alpha_n = \alpha(x'|x_n) \]

- Update scale by
  \[ \log \sigma_{n+1} = \log \sigma_n + \gamma_n (\alpha_n - \alpha^*) \]
  where the adaptation parameter \( \gamma_n \to 0 \)
Optimal scaling is not the whole story. In fact, the optimal proposal suggests to learn the covariance matrix of the target distribution (e.g., use the empirical estimates)

The algorithm runs as follows:

- Sample a candidate value from $\mathcal{N}(x_n, \frac{2.38^2}{d}\Sigma_n)$
- Select the value $x_{n+1}$ as in the usual Metropolis (or MH)
- Update the proposal distribution in two steps:

$$
\begin{align*}
\mu_{n+1} &= \mu_n + \gamma_{n+1}(x_{n+1} - \mu_n) \\
\Sigma_{n+1} &= \Sigma_n + \gamma_{n+1}((x_{n+1} - \mu_n)(x_{n+1} - \mu_n)^T - \Sigma_n)
\end{align*}
$$

where $\gamma_n \to 0$

- Many variants exist (e.g., adapting the scale, block updates, and batch adaption, etc)
The performance of HMC would be sensitive to its hyperparameters, mainly the stepsize $\epsilon$ and trajectory length $L$. 
Optimal acceptance rate strategy might not work well. The example shown on the previous slides all have similar acceptance rate.

Effective sample size is impractical since high order auto-correlation are hard to estimate.

Wang et al (2013) uses normalized expected squared jumping distance (ESJD)

$$\text{ESJD}_\gamma = \mathbb{E}_\gamma \| x^{(t+1)} - x^{(t)} \|^2 / \sqrt{L}$$

where $\gamma = (\epsilon, L)$

Update $\gamma$ via Bayesian optimization, with an annealing adapting rate.
Instead of using a fixed trajectory length $L$, we can sample it from some distribution (e.g., $\mathcal{U}(1, L_{\text{max}})$)

Split the Hamiltonian

$$H(x, r) = H_1(x, r) + H_2(x, r) + \cdots + H_k(x, r)$$

simulate Hamiltonian dynamics on each $H_i$ (sequentially or randomly) give the Hamiltonian dynamics on $H$. Can save computation if some of the $H_i$ are analytically solvable

Partial momentum refreshment

Acceptance using windows of states

See Neal (2010) for more complete and detailed discussion


References


References
