Markov Chain Monte Carlo

Advanced Statistical Inference

Simone Rossi

Sampling from an unknown distribution

- **Problem**: How to sample from a distribution $p(\boldsymbol{\theta} \mid \boldsymbol{y})$ when we don't know its form and the normalization constant Z?

Markov Chain Monte Carlo

Markov Chain Monte Carlo (MCMC): use randomness to generate samples from a distribution

> **i** History
>
> - **1946**: Stanislaw Ulam, John von Neumann, and Nicholas Metropolis working at Los Alamos National Laboratory develop the Metropolis algorithm.
> - **1947**: John von Neumann implements the algorithm on the ENIAC computer to simulate neutron diffusion.
> - **1953**: MCMC algorithms published in the Journal of Chemical Physics.

Metropolis-Hastings Algorithm

MH algorithm produces a sequence of samples $\boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}, \dots, \boldsymbol{\theta}^{(T)}$ that converges to the target distribution $p(\boldsymbol{\theta} \mid \boldsymbol{y})$.

Properties:

- **Markov Chain**: A sequence of random variables $\boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}, \dots, \boldsymbol{\theta}^{(T)}$ where the distribution of $\boldsymbol{\theta}^{(t)}$ depends only on $\boldsymbol{\theta}^{(t-1)}$.
- **Stationary Distribution**: The distribution of $\boldsymbol{\theta}^{(t)}$ converges to the target distribution $p(\boldsymbol{\theta} \mid \boldsymbol{y})$ as $t \to \infty$.

Metropolis-Hastings Algorithm

How to generate the sequence of samples $\boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}, \ldots, \boldsymbol{\theta}^{(T)}$?

1. **Initialization**: Start with an initial value $\boldsymbol{\theta}^{(1)}$.
2. **Proposal Distribution**: Generate a candidate sample $\boldsymbol{\theta}'$ from a proposal distribution $q(\boldsymbol{\theta}' \mid \boldsymbol{\theta}^{(t)})$.
3. **Accepts/Rejects**: Accept the candidate sample based on some criteria.
   - If *accepted*, set $\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}'$.
   - If *rejected*, set $\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)}$.
4. **Repeat**: Repeat steps 2-3 until we have T samples.

Metropolis-Hastings Algorithm: Proposal Distribution

$q(\boldsymbol{\theta}' \mid \boldsymbol{\theta}^{(t)})$ is a distribution that generates candidate samples $\boldsymbol{\theta}'$ given the current sample $\boldsymbol{\theta}^{(t)}$.

- We are free to choose any distribution as the proposal distribution (with the same or larger support as the target distribution).
- **Symmetric Proposal Distribution**: $q(\boldsymbol{\theta}' \mid \boldsymbol{\theta}^{(t)}) = q(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{\theta}')$.
- **Random Walk Proposal**: $q(\boldsymbol{\theta}' \mid \boldsymbol{\theta}^{(t)}) = \mathcal{N}(\boldsymbol{\theta}^{(t)}, \boldsymbol{\Sigma})$.

Metropolis-Hastings Algorithm: Acceptance Criteria

How to decide whether to accept or reject the candidate sample $\boldsymbol{\theta}'$?

In the general case, acceptance based on:

$$\alpha = \frac{p(\boldsymbol{\theta}' \mid \boldsymbol{y}) p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{\theta}')}{p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{y}) p(\boldsymbol{\theta}' \mid \boldsymbol{\theta}^{(t)})}$$

If proposal distribution is symmetric,

$$\alpha = \frac{p(\boldsymbol{\theta}' \mid \boldsymbol{y})}{p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{y})} = \frac{p(\boldsymbol{y} \mid \boldsymbol{\theta}') p(\boldsymbol{\theta}')/Z}{p(\boldsymbol{y} \mid \boldsymbol{\theta}^{(t)}) p(\boldsymbol{\theta}^{(t)})/Z} = \frac{p(\boldsymbol{y} \mid \boldsymbol{\theta}') p(\boldsymbol{\theta}')}{p(\boldsymbol{y} \mid \boldsymbol{\theta}^{(t)}) p(\boldsymbol{\theta}^{(t)})}$$

> **Tip**
>
> Compute the acceptance ratio in the log-space to be numerically stable.

Metropolis-Hastings Algorithm: Acceptance Criteria

1. **Acceptance**: If $\alpha \geq 1$, accept the candidate sample $\boldsymbol{\theta}'$.
2. **Rejection**: If $\alpha < 1$, accept the candidate sample $\boldsymbol{\theta}'$ with probability α.

Intuition:

1. If $\boldsymbol{\theta}'$ is more likely than $\boldsymbol{\theta}^{(t)}$, accept $\boldsymbol{\theta}'$.
2. If $\boldsymbol{\theta}'$ is less likely than $\boldsymbol{\theta}^{(t)}$, accept $\boldsymbol{\theta}'$ with some probability.

Metropolis-Hastings Algorithm

Steps:

1. Initialize $\boldsymbol{\theta}^{(1)}$.
2. For $t = 1, 2, \ldots, T$:
   - Generate a candidate sample $\boldsymbol{\theta}' \sim q(\boldsymbol{\theta}' \mid \boldsymbol{\theta}^{(t)})$.
   - Compute the acceptance ratio α.
   - Sample $u \sim \mathcal{U}(0, 1)$.
   - Compute $A = \min(1, \alpha)$.
   - Set new sample:
   $$\boldsymbol{\theta}^{(t+1)} = \begin{cases} \boldsymbol{\theta}' & \text{if } u \leq A \text{ (accept)} \\ \boldsymbol{\theta}^{(t)} & \text{if } u > A \text{ (reject)} \end{cases}$$
3. Repeat step 2 until we have T samples.

Assessing Convergence

When to stop?

- MCMC algorithms converge to the target distribution as $T \to \infty$.
- Practically, we need to decide when to stop the algorithm or to be able to assess its convergence.

1. **Burn-in Period**: Discard the initial samples to allow the chain to converge to the target distribution.
2. **Visual Inspection**: Plot the trace of the samples and check for convergence.
3. **Multiple Chains**: Run multiple chains from different initial values and compare their results.
4. **Compute heuristics**: Compute some metrics to assess convergence.

Assessing Convergence with $\hat{R}$

Intuition: If multiple chains have converged, in-chain variance should be similar to between-chain variance.

Potential Scale Reduction Factor ($\hat{R}$):

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

In practice:

- $\hat{R} > 1.01$: Some chains have not converged, you may need to run the chains longer.
- $\hat{R} < 1.01$: Convergence has been reached.

Propose better and accept more

Hamiltonian Monte Carlo

Hamiltonian Monte Carlo (HMC): A more sophisticated MCMC algorithm that uses the concept of Hamiltonian dynamics to propose better samples.

Idea:

- Use gradient information to guide local exploration of the target distribution.
- Propose samples by simulating the dynamics of a particle moving in a potential energy landscape.

Hamiltonian Dynamics

A particle moving in a potential energy landscape $\mathcal{U}(\theta)$ with momentum $\boldsymbol{\rho}$.

Hamiltonian:

$$\mathcal{H}(\theta, \boldsymbol{\rho}) = \mathcal{U}(\theta) + \mathcal{K}(\boldsymbol{\rho})$$

- $\mathcal{U}(\theta)$: Potential energy of the particle at position θ.
- $\mathcal{K}(\boldsymbol{\rho})$: Kinetic energy of the particle with momentum $\boldsymbol{\rho}$.

We consider:

- $\mathcal{U}(\theta) = -\log p(\theta \mid \boldsymbol{y})$, where $p(\theta \mid \boldsymbol{y})$ is the target distribution.
- $\mathcal{K}(\boldsymbol{\rho}) = \frac{1}{2}\boldsymbol{\rho}\boldsymbol{M}^{-1}\boldsymbol{\rho}$, where $\boldsymbol{M}$ is the mass matrix.

Energy Conservation

- In case of no-friction, the Hamiltonian is conserved.
- The trajectory of the particle in the potential energy landscape is governed by the **Hamiltonian equations**:

$$\frac{\mathrm{d}\theta}{\mathrm{d}t} = \nabla_{\boldsymbol{\rho}}\mathcal{H}(\theta, \boldsymbol{\rho}) = \nabla_{\boldsymbol{\rho}}\mathcal{K}(\boldsymbol{\rho})$$
$$\frac{\mathrm{d}\boldsymbol{\rho}}{\mathrm{d}t} = -\nabla_{\theta}\mathcal{H}(\theta, \boldsymbol{\rho}) = -\nabla_{\theta}\mathcal{U}(\theta)$$

The energy is conserved: $\frac{\mathrm{d}\mathcal{H}}{\mathrm{d}t} = 0$.

> **!** Important
>
> To simulate the dynamics of the particle, we don't need to know the normalization constant Z for $\mathcal{U}(\theta) = -\log p(\theta \mid \boldsymbol{y})$.
> Why? Because the dynamics only depend on the gradient of the log-density $\nabla_{\theta} \log p(\theta \mid \boldsymbol{y}) = \nabla_{\theta}[\log p(\boldsymbol{y} \mid \theta) + \log p(\theta) - \log Z]$.

Numerical Solution to the Hamiltonian Equations

$$\frac{\mathrm{d}\boldsymbol{\theta}}{\mathrm{d}t} = \boldsymbol{\nabla}_{\boldsymbol{\rho}}\mathcal{K}(\boldsymbol{\rho})$$
$$\frac{\mathrm{d}\boldsymbol{\rho}}{\mathrm{d}t} = -\boldsymbol{\nabla}_{\boldsymbol{\theta}}\mathcal{U}(\boldsymbol{\theta})$$

Solving exactly is infeasible, but we can use numerical methods to approximate the solution.

Leapfrog Integrator:

Start with $\boldsymbol{\theta}, \boldsymbol{\rho}$.

Repeat for L steps:

1. Update momentum: $\boldsymbol{\rho} \leftarrow \boldsymbol{\rho} - \frac{\epsilon}{2}\boldsymbol{\nabla}_{\boldsymbol{\theta}}\mathcal{U}(\boldsymbol{\theta})$.
2. Update position: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \epsilon\boldsymbol{\nabla}_{\boldsymbol{\rho}}\mathcal{K}(\boldsymbol{\rho})$.
3. Update momentum: $\boldsymbol{\rho} \leftarrow \boldsymbol{\rho} - \frac{\epsilon}{2}\boldsymbol{\nabla}_{\boldsymbol{\theta}}\mathcal{U}(\boldsymbol{\theta})$.

Simulating Hamiltonian Dynamics $\Leftrightarrow$ Sampling

$$p(\boldsymbol{\theta}, \boldsymbol{\rho}) = \frac{1}{Z_H}\exp(-\mathcal{H}(\boldsymbol{\theta}, \boldsymbol{\rho})) = \frac{1}{Z_H}\exp\left(-\mathcal{U}(\boldsymbol{\theta}) - \frac{1}{2}\boldsymbol{\rho}^\top \boldsymbol{M}^{-1}\boldsymbol{\rho}\right)$$

The marginal distribution of $\boldsymbol{\theta}$ is the target distribution $p(\boldsymbol{\theta} \mid \boldsymbol{y})$.

$$p(\boldsymbol{\theta} \mid \boldsymbol{y}) = \int p(\boldsymbol{\theta}, \boldsymbol{\rho})\,\mathrm{d}\boldsymbol{\rho} = \frac{1}{Z}\exp(-\mathcal{U}(\boldsymbol{\theta}))\frac{1}{Z_K}\int \exp\left(-\frac{1}{2}\boldsymbol{\rho}^\top \boldsymbol{M}^{-1}\boldsymbol{\rho}\right)\mathrm{d}\boldsymbol{\rho} = \frac{1}{Z}\exp(-\mathcal{U}(\boldsymbol{\theta}))$$

Hamiltonian Monte Carlo Algorithm

Suppose the last sample in the sequence is $\boldsymbol{\theta}^{(t-1)}$

1. **Initialization**:
   - Set $\boldsymbol{\theta}_0' = \boldsymbol{\theta}^{(t-1)}$.
   - Sample momentum $\boldsymbol{\rho}_0' \sim \mathcal{N}(0, \boldsymbol{M})$.
2. **Hamiltonian Dynamics**:
   - Simulate Hamiltonian dynamics for L steps to obtain $\boldsymbol{\theta}_L', \boldsymbol{\rho}_L'$.
3. **Acceptance**:
   - Accept with probability α

$$\alpha = \min\left(1, \frac{p(\boldsymbol{\theta}'_L, \boldsymbol{\rho}'_L)}{p(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{\rho}^{(t-1)})}\right) = \min\left(1, \exp\left(-\mathcal{H}(\boldsymbol{\theta}'_L, \boldsymbol{\rho}'_L) + \mathcal{H}(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{\rho}^{(t-1)})\right)\right)$$

Tuning HMC

- **Number of Steps**: The number of steps L to simulate the Hamiltonian dynamics.
- **Step Size**: The size of the leapfrog steps ϵ.
- **Mass Matrix**: The mass matrix $\boldsymbol{M}$.

Considerations:

1. L needs to be large enough to explore the target distribution, but small enough to avoid wasting computation.
2. ϵ needs to be small enough to avoid numerical instability, but large enough to have α close to 1.
3. $\boldsymbol{M}$ can be set to the identity matrix or estimated from the target distribution (after burn-in).

Extensions of HMC

- **No-U-Turn Sampler (NUTS)**: Automatically adapt the number of steps L.
- **Stochastic Gradient Hamiltonian Monte Carlo**: Use stochastic gradients to scale to large datasets

$$\mathcal{U}(\boldsymbol{\theta}) = -\log p(\boldsymbol{\theta} \mid \boldsymbol{y}) \propto -\sum_{i=1}^{n} \log p(\boldsymbol{y}_i \mid \boldsymbol{\theta}) - \log p(\boldsymbol{\theta})$$