Lecture 6.2: Inference Pat 2

谢丹
清华大学数学系

November 12, 2025

Section 2: EM algorithm

Problem Setup

We often need to introduce the latent variable to deal with complex probability. The evaluation of the marginal probability of the observed variable is crucial.

- Observed data: $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$
- Latent variables: $\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_N\}$
- Model parameters: θ

We want to maximize the marginal likelihood (evidence):

$$\log p_\theta(\mathbf{X}) = \log \int p_\theta(\mathbf{X}, \mathbf{Z}) d\mathbf{Z}$$

Challenge

The integral is typically intractable for complex models!

Variational Inference Approach

- Introduce variational distribution $q_\phi(\mathbf{Z})$
- Approximate true posterior $p_\theta(\mathbf{Z}|\mathbf{X})$
- Find ϕ that makes $q_\phi(\mathbf{Z})$ close to true posterior

Deriving the ELBO - Step 1

Start with the evidence:

$$\log p_\theta(\mathbf{X}) = \log \int p_\theta(\mathbf{X}, \mathbf{Z}) d\mathbf{Z}$$

Introduce variational distribution:

$$\log p_\theta(\mathbf{X}) = \log \int q_\phi(\mathbf{Z}) \frac{p_\theta(\mathbf{X}, \mathbf{Z})}{q_\phi(\mathbf{Z})} d\mathbf{Z}$$

Deriving the ELBO - Step 2

Apply Jensen's inequality (since log is concave):

$$\log \int q_\phi(\mathbf{Z}) \frac{p_\theta(\mathbf{X}, \mathbf{Z})}{q_\phi(\mathbf{Z})} d\mathbf{Z} \geq \int q_\phi(\mathbf{Z}) \log \frac{p_\theta(\mathbf{X}, \mathbf{Z})}{q_\phi(\mathbf{Z})} d\mathbf{Z}$$

This gives us the Evidence Lower Bound (ELBO):

$$\boxed{\mathcal{L}(\theta, \phi) = \mathbb{E}_{q_\phi(\mathbf{Z})} \left[\log \frac{p_\theta(\mathbf{X}, \mathbf{Z})}{q_\phi(\mathbf{Z})} \right]}$$

which is valid for any distribution $q_\phi(\mathbf{Z})$, which is often defined by a neural network.

Alternative Forms of ELBO

Form 1

$$\mathcal{L}(\theta, \phi) = \mathbb{E}_{q_\phi(\mathbf{Z})} \left[\log p_\theta(\mathbf{X}, \mathbf{Z}) \right] - \mathbb{E}_{q_\phi(\mathbf{Z})} \left[\log q_\phi(\mathbf{Z}) \right]$$

Form 2

$$\mathcal{L}(\theta, \phi) = \mathbb{E}_{q_\phi(\mathbf{Z})} \left[\log p_\theta(\mathbf{X}|\mathbf{Z}) \right] - D_{KL}(q_\phi(\mathbf{Z}) \| p_\theta(\mathbf{Z}))$$

Where D_{KL} is the Kullback-Leibler divergence.

Interpretation

- $\mathbb{E}_{q_\phi(\mathbf{Z})}[\log p_\theta(\mathbf{X}|\mathbf{Z})]$: **Reconstruction term**
  - Measures how well we can reconstruct data from latents
- $D_{KL}(q_\phi(\mathbf{Z})\|p_\theta(\mathbf{Z}))$: **Regularization term**
  - Keeps approximate posterior close to prior
  - Prevents overfitting

Optimization

We maximize the ELBO:

$$\theta^*, \phi^* = \arg\max_{\theta,\phi} \mathcal{L}(\theta, \phi)$$

This gives us:

- Good model parameters θ
- Good variational approximation $q_\phi(\mathbf{Z})$

Since

$$\boxed{\log p_\theta(\mathbf{X}) = \mathcal{L}(\theta, \phi) + D_{KL}(q_\phi(\mathbf{Z}) \| p_\theta(\mathbf{Z}|\mathbf{X}))}$$

Maximizing ELBO minimizes $D_{KL}(q_\phi(\mathbf{Z}) \| p_\theta(\mathbf{Z}|\mathbf{X}))$

The EM Strategy: A Lower Bound

Let's prove

$$\boxed{\log p_\theta(\mathbf{X}) = \mathcal{L}(\theta, \phi) + D_{KL}(q_\phi(\mathbf{Z}) \| p_\theta(\mathbf{Z}|\mathbf{X}))}$$

Proof:

$$\mathcal{L}(\theta, \phi) + D_{KL}(q_\phi(\mathbf{Z}) \| p_\theta(\mathbf{Z}|\mathbf{X})) =$$

$$\int q_\phi(Z) \left[\log \frac{p_\theta(\mathbf{X}, \mathbf{Z})}{q_\phi(\mathbf{Z})} \right] dZ + \int q_\phi(Z) \log \frac{q_\phi(Z)}{p_\theta(\mathbf{Z}|\mathbf{X})} dZ$$

Using the equation $p_\theta(X, Z) = p_\theta(Z|X)p_\theta(X)$, we get the important identity.

Remarks

1. Notice that the sum is independent of the distribution $q_\phi(Z)$.
2. $D_{KL}(q_\phi(\mathbf{Z}) \| p_\theta(\mathbf{Z}|\mathbf{X})) \geq 0$, and ELBO gives the lower bound.

The Two Steps of EM I

EM is an iterative algorithm that alternates between:

E-Step: Fix $\boldsymbol{\theta}$, maximize $\mathcal{L}$ w.r.t. q

(using the property of KL divergence) Hold $\boldsymbol{\theta}^{old}$ fixed. The optimal q is the posterior:

$$q^{opt}(\mathbf{Z}) = p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{old})$$

We compute the **Q-function**:

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{old}) = \mathbb{E}_{p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{old})}[\log p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})] + H(q^{opt})$$

This "fills in" the missing data $\mathbf{Z}$ in the log-likelihood.

M-Step: Fix q, maximize $\mathcal{L}$ w.r.t. $\boldsymbol{\theta}$

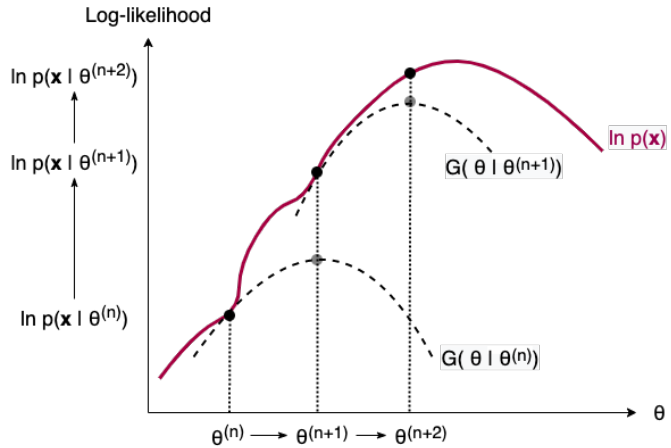
The Two Steps of EM II

Hold q fixed. Find new parameters that maximize the Q-function:

$$\boldsymbol{\theta}^{new} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{old})$$

This is often a much easier optimization problem.

Visualization of the EM Algorithm



Why Does It Work? The Guarantee

Theorem (Monotonic Increase of Log-Likelihood)

The EM algorithm never decreases the log-likelihood:

$$\log p(\mathbf{X}|\boldsymbol{\theta}^{new}) \geq \log p(\mathbf{X}|\boldsymbol{\theta}^{old})$$

Proof.

$$\begin{aligned}\log p(\mathbf{X}|\boldsymbol{\theta}) &= \mathcal{L}(q, \boldsymbol{\theta}) + D_{\mathsf{KL}}(q(\mathbf{Z}) \parallel p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta})) \\ &\geq \mathcal{L}(q, \boldsymbol{\theta}) \quad \text{(since KL divergence } \geq 0\text{)}\end{aligned}$$

In the E-Step, we set $q = p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{old})$, making KL$= 0$, so $\log p(\mathbf{X}|\boldsymbol{\theta}^{old}) = \mathcal{L}(q, \boldsymbol{\theta}^{old})$. In the M-Step, $\mathcal{L}(q, \boldsymbol{\theta}^{new}) \geq \mathcal{L}(q, \boldsymbol{\theta}^{old})$. Since KL$\geq 0$, $\log p(\mathbf{X}|\boldsymbol{\theta}^{new}) \geq \mathcal{L}(q, \boldsymbol{\theta}^{new})$. Thus, $\log p(\mathbf{X}|\boldsymbol{\theta}^{new}) \geq \log p(\mathbf{X}|\boldsymbol{\theta}^{old})$.

Summary and Applications

- **Purpose**: Find MLE/MAP estimates for models with latent variables.
- **Core Idea**: Iteratively maximize a lower bound (ELBO) on the log-likelihood.
- **Steps**:
  1. **E-Step**: Impute the latent variables (compute expectations).
  2. **M-Step**: Update the parameters using the "complete" data.
- **Guarantee**: Monotonically increases the log-likelihood.

Common Applications

- Gaussian Mixture Models (GMMs)
- Hidden Markov Models (HMMs)
- Topic Models (e.g., Latent Dirichlet Allocation)
- Clustering with soft assignments
- Missing data imputation

General Pseudo-Code

Input/Output

- **Input**: Observed data X, latent variables Z, parameters θ
- **Output**: Converged parameters θ_{final}

Initialize θ_{old}, threshold ϵ, max_iters
for iteration $= 1$ to max_iters **do**
    E-Step: Compute $Q(\theta|\theta_{old}) = E_{P(Z|X,\theta_{old})}[\log P(X, Z|\theta)]$
    M-Step: $\theta_{new} = \arg\max_\theta Q(\theta|\theta_{old})$
    if $|\log P(X|\theta_{new}) - \log P(X|\theta_{old})| < \epsilon$ **then**
        break
    else
        $\theta_{old} \leftarrow \theta_{new}$
    end if
end for
return $\theta_{final} = \theta_{new}$

Gaussian Mixture Model (GMM) Example

Problem Setup

- **Observed**: Data points X
- **Latent**: Component assignments Z
- **Parameters**: $\theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$

E-Step: Responsibilities

For each point i and component k:

$$\gamma_{ik} = \frac{\pi_k \cdot \mathcal{N}(X_i|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \cdot \mathcal{N}(X_i|\mu_j, \Sigma_j)}$$

M-Step: Parameter Updates

- $N_k = \sum_{i=1}^{N} \gamma_{ik}$
- $\pi_k^{new} = \frac{N_k}{N}$
- $\mu_k^{new} = \frac{1}{N_k} \sum_{i=1}^{N} \gamma_{ik} X_i$

The E step is $Q(\theta|\theta_{old}) = E_{P(Z|X,\theta_{old})}[\log P(X, Z|\theta)]$, with

Section 3: Variational methods

What are Variational Methods?

- Mathematical framework for approximating complex probability distributions
- Transform intractable problems into tractable optimization problems
- Widely used in Bayesian inference and deep learning
- Core idea: Find a simpler distribution that approximates the true posterior

Key Concepts

The crucial equation

$$\boxed{\log p_\theta(\mathbf{X}) = \mathcal{L}(\theta, q_\phi) + D_{KL}(q_\phi(\mathbf{Z}) \| p_\theta(\mathbf{Z}|\mathbf{X}))}$$

Evidence Lower Bound (ELBO)

$$\mathcal{L}(q) = \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}, \mathbf{z})] - \mathbb{E}_{q(\mathbf{z})}[\log q(\mathbf{z})]$$

Kullback-Leibler Divergence

$$KL(q\|p) = \int q(\mathbf{z}) \log \frac{q(\mathbf{z})}{p(\mathbf{z}|\mathbf{x})} d\mathbf{z}$$

Variational Inference Framework

- **Goal**: Approximate posterior $p(z|x)$ with simpler distribution $q(z)$
- **Objective**: Minimize $KL(q(z)||p(z|x))$
- **Approach**: Maximize ELBO (equivalent to minimizing KL divergence)
- **Family**: Choose variational family $q(z;\lambda)$ with parameters λ, and maximize the ELBO over space λ.

In the EM algorithm, we find the parameter maximizing the evidence $p(X|\theta)$. Variational method also gives an approximation for the evidence and the normalization constant Z (X is absent).

Example: Mean field of Ising model

Core Idea

Approximate complex probability distribution $p(\mathbf{s})$ with simpler distribution $q(\mathbf{s})$ from tractable family

Exact Distribution

Boltzmann distribution:

$$p(\mathbf{s}) = \frac{1}{Z} e^{\beta J \sum_{\langle ij \rangle} s_i s_j + \beta h \sum_i s_i}$$

Variational Distribution

Mean field assumption:

$$q(\mathbf{s}) = \prod_{i=1}^{N} q_i(s_i)$$

Key Insight

Mean field theory = Specific case of variational inference with factorized q

Mathematical Foundation

Variational Distribution Parametrization

For binary spins $s_i = \pm 1$:

$$q_i(s_i) = \frac{1 + m_i s_i}{2}, \quad \mathbb{E}_q[s_i] = m_i$$

Evidence Lower Bound (ELBO)

Maximize:

$$\mathcal{L}[q] = \mathbb{E}_q[\log p(\mathbf{s})] - \mathbb{E}_q[\log q(\mathbf{s})]$$

- **Energy term**: $\mathbb{E}_q[\log p(\mathbf{s})]$
- **Entropy term**: $-\mathbb{E}_q[\log q(\mathbf{s})]$

ELBO Derivation I

Energy Term

$$\begin{aligned}\mathbb{E}_q[\log p(\mathbf{s})] &= \beta J \sum_{\langle ij \rangle} \mathbb{E}_q[s_i s_j] + \beta h \sum_i \mathbb{E}_q[s_i] - \log Z \\ &= \beta J \sum_{\langle ij \rangle} m_i m_j + \beta h \sum_i m_i - \log Z\end{aligned}$$

Entropy Term

$$\begin{aligned}\mathbb{E}_q[\log q(\mathbf{s})] &= \sum_i \mathbb{E}_{q_i}[\log q_i(s_i)] \\ &= \sum_i \left[\frac{1+m_i}{2} \log\left(\frac{1+m_i}{2}\right) + \frac{1-m_i}{2} \log\left(\frac{1-m_i}{2}\right) \right]\end{aligned}$$

ELBO Derivation II

Complete ELBO

$$\mathcal{L}[\{m_i\}] = \beta J \sum_{\langle ij \rangle} m_i m_j + \beta h \sum_i m_i - \sum_i S(m_i) - \log Z$$

where $S(m_i)$ is binary entropy.

Optimization and Self-Consistency

Coordinate Ascent

Take derivatives of ELBO:

$$\frac{\partial \mathcal{L}}{\partial m_i} = 2\beta J \sum_{j \in \text{n.n.}(i)} m_j + \beta h - \frac{1}{2} \log \left(\frac{1+m_i}{1-m_i} \right)$$

Self-Consistency Equations

Setting $\partial \mathcal{L} / \partial m_i = 0$:

$$\frac{1}{2} \log \left(\frac{1+m_i}{1-m_i} \right) = 2\beta J \sum_{j \in \text{n.n.}(i)} m_j + \beta h$$

Using $\text{arctanh}(x) = \frac{1}{2} \log \left(\frac{1+x}{1-x} \right)$:

$$m_i = \tanh \left(2\beta J \sum_{j \in \text{n.n.}(i)} m_j + \beta h \right)$$

Algorithmic Implementation

Coordinate Ascent VI (CAVI)

Initialize $m_i \sim \text{Uniform}(-0.1, 0.1)$
for iteration $= 1$ to max_iters **do**
    for each site i **do**
        neighbor_sum $\leftarrow \sum_{j \in \text{neighbors}(i)} m_j$
        $m_i^{\text{new}} \leftarrow \tanh(2\beta J \cdot \text{neighbor\_sum} + \beta h_i)$
    end for
    if $\max_i |m_i^{\text{new}} - m_i| <$ tol **then break**
    end if
    $m \leftarrow m^{\text{new}}$
end for

Direct ELBO Optimization

Alternatively, maximize ELBO directly using gradient methods:

$$\nabla_{m_i} \mathcal{L} = 2\beta J \sum_{j \in \text{n.n.}(i)} m_j + \beta h - \text{arctanh}(m_i)$$

Section 4: Sampling methods

Why Sampling?

Key Motivations

- Approximate complex integrals:

$$\mathbb{E}[f(x)] = \int f(x)p(x)dx \approx \frac{1}{N}\sum_{i=1}^{N} f(x_i)$$

- Perform inference in complex probabilistic models
- Generate synthetic data for simulations

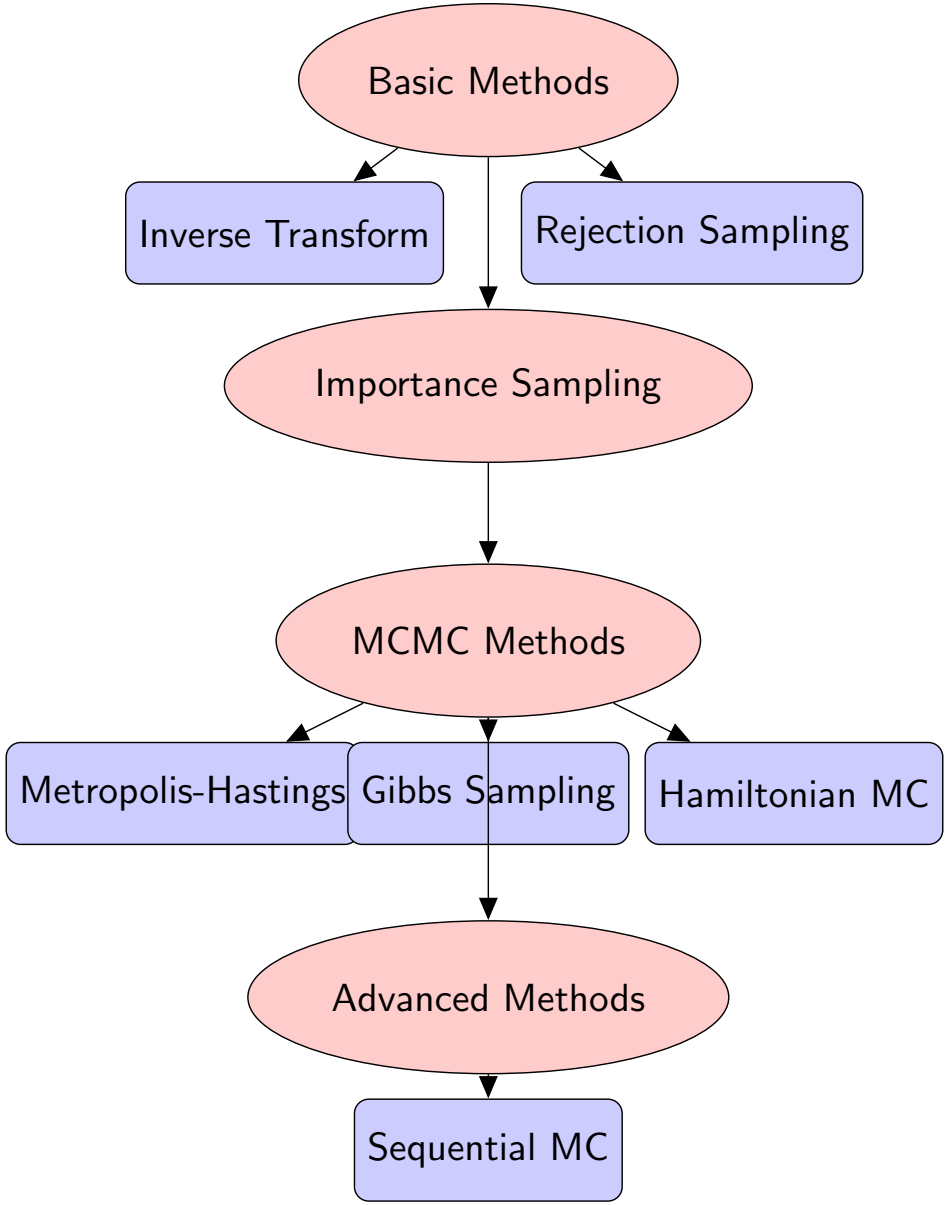
Example

Monte Carlo Integration Instead of solving hard integrals analytically, we approximate them empirically using samples from the distribution.

The Fundamental Challenge

How to efficiently generate samples from complex, high-dimensional probability distributions?

Taxonomy of Sampling Methods



Inverse Transform Sampling

Core Idea

Generate samples using uniform random variables and the inverse CDF.

Algorithm

1. Generate $u \sim \text{Uniform}(0, 1)$
2. Compute $x = F^{-1}(u)$
3. Return x as sample

Theorem (Probability Integral Transform)

If $U \sim \textit{Uniform}(0, 1)$, *then* $X = F^{-1}(U)$ *has distribution* F.

Example

Exponential Distribution

$p(x) = \lambda e^{-\lambda x}$

CDF: $F(x) = 1 - e^{-\lambda x}$

Inverse CDF: $F^{-1}(u) = -\frac{\ln(1-u)}{\lambda}$

Pros and Cons

- **Pros**: Exact sampling, simple implementation
- **Cons**: Requires analytical inverse CDF, inefficient in high dimensions

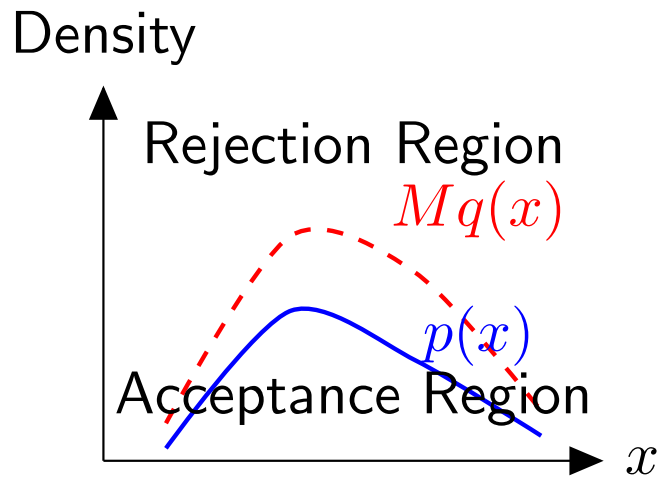
Rejection Sampling I

Concept

Sample from proposal $q(x)$ and accept/reject based on target $p(x)$.

Algorithm

1. Find M such that $p(x) \leq Mq(x)$
2. Sample $x \sim q(x)$
3. Sample $u \sim \text{Uniform}(0, 1)$
4. Accept if $u < \frac{p(x)}{Mq(x)}$



Acceptance Rate and Limitations

Rejection Sampling II

$$\text{Acceptance Rate} = \frac{1}{M}$$

- Efficiency depends on how well $q(x)$ matches $p(x)$
- Curse of dimensionality: acceptance rate decreases exponentially
- Difficult to find good M in high dimensions

Importance Sampling I

Core Idea

Weight samples from proposal distribution rather than generating exact samples.

Algorithm

1. Sample $x_i \sim q(x)$ for $i = 1, \ldots, N$
2. Compute weights: $w_i = \frac{p(x_i)}{q(x_i)}$
3. Normalize: $\tilde{w}_i = \frac{w_i}{\sum_j w_j}$

Expectation Estimation

$$\mathbb{E}_{p(x)}[f(x)] \approx \sum_{i=1}^{N} \tilde{w}_i f(x_i)$$

Effective Sample Size

$$\text{ESS} = \frac{1}{\sum_{i=1}^{N} \tilde{w}_i^2}$$

Measures how many "useful" samples we have.

Weight Degeneracy

In high dimensions, few samples dominate the weights, making estimation unreliable.

Importance Sampling II

Advantages and Limitations

- **Pros**: Always works, provides unbiased estimates
- **Cons**: Weight degeneracy, sensitive to proposal choice

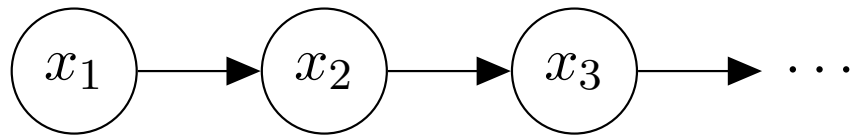
Markov Chain Monte Carlo (MCMC) Fundamentals

Core Idea

Construct a Markov chain whose stationary distribution is the target distribution $p(x)$.

Key Properties

- **Detailed Balance**: $p(x)T(xx') = p(x')T(x'x)$
- **Ergodicity**: Chain converges to stationary distribution
- **Burn-in**: Discard initial samples



Example

Markov Property

$p(x_{t+1}|x_t, x_{t-1}, \ldots, x_1) = p(x_{t+1}|x_t)$

Convergence Guarantees

Under mild conditions, the chain will converge to the target distribution regardless of initial state.

Metropolis-Hastings Algorithm

The Workhorse of MCMC

Most general and widely used MCMC method.

Algorithm

For $t = 0, 1, 2, \ldots$:

1. Sample $x^* \sim q(x^*|x_t)$
2. Compute acceptance probability:

$$\alpha = \min\left(1, \frac{p(x^*)q(x_t|x^*)}{p(x_t)q(x^*|x_t)}\right)$$

3. Sample $u \sim \text{Uniform}(0, 1)$
4. If $u < \alpha$, accept: $x_{t+1} = x^*$
   Else reject: $x_{t+1} = x_t$

Proposal Variants

- **Random Walk MH**: $q(x^*|x) = \mathcal{N}(x, \sigma^2)$
- **Independent MH**: $q(x^*|x) = q(x^*)$

Example

Symmetric Proposals When $q(x^*|x) = q(x|x^*)$ (symmetric), acceptance simplifies to:

$$\alpha = \min\left(1, \frac{p(x^*)}{p(x_t)}\right)$$

Gibbs Sampling I

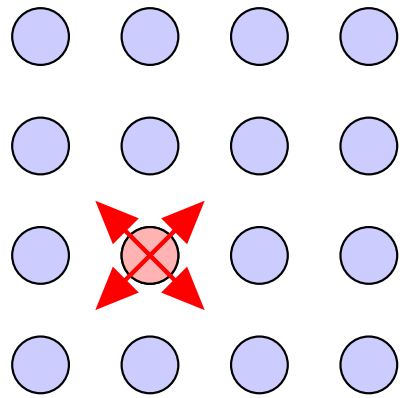
Special Case of Metropolis-Hastings

Sample one variable at a time from its full conditional distribution.

Algorithm

For target $p(x_1, x_2, \ldots, x_D)$:

1. Initialize $x_1^{(0)}, \ldots, x_D^{(0)}$
2. For $t = 1, 2, \ldots$:
   - Sample $x_1^{(t)} \sim p(x_1 | x_2^{(t-1)}, \ldots, x_D^{(t-1)})$
   - Sample $x_2^{(t)} \sim p(x_2 | x_1^{(t)}, x_3^{(t-1)}, \ldots)$
   - $\ldots$
   - Sample $x_D^{(t)} \sim p(x_D | x_1^{(t)}, \ldots, x_{D-1}^{(t)})$



Example

Conditional Dependencies Each variable is sampled given its Markov blanket.

Advantages and Limitations

Gibbs Sampling II

- **Pros**: No tuning parameters, acceptance rate $= 1$
- **Cons**: Requires sampling from conditionals, can mix slowly

Theoretical basis of MCMC

Goal

Sample from a complex target distribution $\pi(\boldsymbol{x})$.

- $\pi(\boldsymbol{x})$ is often high-dimensional and known only up to a constant: $\pi(\boldsymbol{x}) \propto P(\boldsymbol{x})$.
- Direct sampling (e.g., inverse transform) is impossible.
- Solution: Construct a Markov Chain whose **stationary distribution** is $\pi(\boldsymbol{x})$.

Markov Chains & Transition Kernels

Markov Property

The future state depends only on the present state.

$$P(X_{t+1} = \boldsymbol{x}' | X_t = \boldsymbol{x}, X_{t-1}, \dots) = P(X_{t+1} = \boldsymbol{x}' | X_t = \boldsymbol{x})$$

Transition Kernel

Describes the probability of moving from $\boldsymbol{x}$ to $\boldsymbol{x}'$.

- Discrete: $T(\boldsymbol{x} \to \boldsymbol{x}')$ or $P(\boldsymbol{x}'|\boldsymbol{x})$
- Continuous: $T(\boldsymbol{x}, \boldsymbol{x}')$

The Stationary Distribution

Definition (Stationary Distribution)

A distribution $\pi(\boldsymbol{x})$ is **stationary** for a Markov chain with transition kernel T if:

$$\pi(\boldsymbol{x}') = \sum_{\boldsymbol{x}} \pi(\boldsymbol{x}) T(\boldsymbol{x} \to \boldsymbol{x}')$$

(Replace sum with integral for continuous case).

Interpretation

Once the chain reaches distribution π, it *stays* there. The probability mass flowing **into** each state equals the mass flowing **out**.

This is the **Global Balance** condition.

From Global to Detailed Balance

Global Balance is often hard to check and enforce directly.

Definition (Detailed Balance Condition)

A Markov chain satisfies **detailed balance** with respect to π if for all $\boldsymbol{x}, \boldsymbol{x}'$:

$$\pi(\boldsymbol{x}) \cdot T(\boldsymbol{x} \to \boldsymbol{x}') = \pi(\boldsymbol{x}') \cdot T(\boldsymbol{x}' \to \boldsymbol{x})$$

Why Detailed Balance is Crucial

Theorem

If a Markov chain satisfies the Detailed Balance condition for a distribution π, then π is a stationary distribution of the chain.

Proof.

Start with detailed balance: $\pi(\boldsymbol{x})T(\boldsymbol{x} \to \boldsymbol{x}') = \pi(\boldsymbol{x}')T(\boldsymbol{x}' \to \boldsymbol{x})$.
Now, sum both sides over all $\boldsymbol{x}$:

$$\sum_{\boldsymbol{x}} \pi(\boldsymbol{x})T(\boldsymbol{x} \to \boldsymbol{x}') = \sum_{\boldsymbol{x}} \pi(\boldsymbol{x}')T(\boldsymbol{x}' \to \boldsymbol{x})$$

The right-hand side simplifies:

$$\pi(\boldsymbol{x}') \sum_{\boldsymbol{x}} T(\boldsymbol{x}' \to \boldsymbol{x}) = \pi(\boldsymbol{x}') \cdot 1 = \pi(\boldsymbol{x}')$$

Thus, $\sum_{\boldsymbol{x}} \pi(\boldsymbol{x})T(\boldsymbol{x} \to \boldsymbol{x}') = \pi(\boldsymbol{x}')$, which is the **global balance** condition.

Metropolis-Hastings: A Detailed-Balance Machine

How do we build a chain that satisfies detailed balance for *any* π?

The Algorithm

1. From current state $\boldsymbol{x}$, propose a new state $\boldsymbol{x}'$ using a **proposal distribution** $q(\boldsymbol{x}'|\boldsymbol{x})$.
2. Calculate the **acceptance probability**:

$$A(\boldsymbol{x}, \boldsymbol{x}') = \min\left(1, \frac{\pi(\boldsymbol{x}') \cdot q(\boldsymbol{x}|\boldsymbol{x}')}{\pi(\boldsymbol{x}) \cdot q(\boldsymbol{x}'|\boldsymbol{x})}\right)$$

3. With probability $A(\boldsymbol{x}, \boldsymbol{x}')$, accept the move and set the next state to $\boldsymbol{x}'$. Otherwise, reject and stay at $\boldsymbol{x}$.

The transition kernel is: $T(\boldsymbol{x} \to \boldsymbol{x}') = q(\boldsymbol{x}'|\boldsymbol{x}) \cdot A(\boldsymbol{x}, \boldsymbol{x}')$

Verifying Detailed Balance for M-H

We need to check:

$$\pi(\boldsymbol{x}) \cdot T(\boldsymbol{x} \to \boldsymbol{x}') = \pi(\boldsymbol{x}') \cdot T(\boldsymbol{x}' \to \boldsymbol{x})$$

Proof.

$$\pi(\boldsymbol{x}) \cdot T(\boldsymbol{x} \to \boldsymbol{x}') = \pi(\boldsymbol{x}) \cdot q(\boldsymbol{x}'|\boldsymbol{x}) \cdot A(\boldsymbol{x}, \boldsymbol{x}')$$
$$\pi(\boldsymbol{x}') \cdot T(\boldsymbol{x}' \to \boldsymbol{x}) = \pi(\boldsymbol{x}') \cdot q(\boldsymbol{x}|\boldsymbol{x}') \cdot A(\boldsymbol{x}', \boldsymbol{x})$$

Assume WLOG that $\pi(\boldsymbol{x}')q(\boldsymbol{x}|\boldsymbol{x}') > \pi(\boldsymbol{x})q(\boldsymbol{x}'|\boldsymbol{x})$.

- Then $A(\boldsymbol{x}, \boldsymbol{x}') = 1$
- And $A(\boldsymbol{x}', \boldsymbol{x}) = \frac{\pi(\boldsymbol{x})q(\boldsymbol{x}'|\boldsymbol{x})}{\pi(\boldsymbol{x}')q(\boldsymbol{x}|\boldsymbol{x}')}$

Substituting in:

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

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

The Random Walk Problem

Metropolis-Hastings Limitations

- Proposes new states **randomly**
- High rejection rates in high dimensions
- **Slow exploration** of parameter space
- Inefficient for correlated distributions

Hamiltonian MC: A Physics Analogy

Physical System

- **Position** q: Parameters
- **Potential Energy** $U(q)$: $-\log \pi(q)$
- **Momentum** p: Auxiliary variables
- **Kinetic Energy** $K(p)$: Quadratic in p

Hamiltonian Mechanics

Definition (Hamiltonian)

Total energy of the system:

$$H(q,p) = U(q) + K(p)$$

Hamilton's Equations

$$\frac{dq}{dt} = +\frac{\partial H}{\partial p} = \frac{\partial K}{\partial p}$$
$$\frac{dp}{dt} = -\frac{\partial H}{\partial q} = -\frac{\partial U}{\partial q}$$

Theorem (Conservation)

Hamiltonian is preserved: $\frac{dH}{dt} = 0$

Connection to Probability

Boltzmann Distribution

$$\pi(q,p) \propto \exp(-H(q,p)) = \exp(-U(q)) \cdot \exp(-K(p))$$

Clever Choices

- Potential energy: $U(q) = -\log \pi(q)$
- Kinetic energy: $K(p) = \frac{1}{2} p^\top M^{-1} p$
- Momentum: $p \sim \mathcal{N}(0, M)$

Key Insight

Marginal distribution of q is exactly our target distribution $\pi(q)$!

$$\pi(q) = \int \pi(q,p) dp \propto \exp(-U(q))$$

Leapfrog Integrator

Why Leapfrog?

- **Time-reversible**
- **Volume-preserving**
- **Symplectic** (approximately conserves Hamiltonian)

One Leapfrog Step

$$p \leftarrow p - \frac{\epsilon}{2}\frac{\partial U}{\partial q}$$
$$q \leftarrow q + \epsilon\frac{\partial K}{\partial p}$$
$$p \leftarrow p - \frac{\epsilon}{2}\frac{\partial U}{\partial q}$$

Complete HMC Algorithm

1. **Sample momentum**: $p \sim \mathcal{N}(0, M)$
2. **Simulate dynamics** (L leapfrog steps):

   for $i = 1$ to L **do**

   $p \leftarrow p - \frac{\epsilon}{2}\nabla U(q)$

   $q \leftarrow q + \epsilon M^{-1} p$

   $p \leftarrow p - \frac{\epsilon}{2}\nabla U(q)$

   end for
3. **Metropolis acceptance**:

$$\alpha = \min\left(1, \exp(H(q,p) - H(q^*,p^*))\right)$$

High Acceptance

Due to Hamiltonian conservation, $\alpha \approx 1$!

Properties and Advantages

Theoretical Properties

- **Time-reversible**
- **Volume-preserving**
- **Hamiltonian-preserving**
- **Ergodic** (under mild conditions)

Practical Advantages

- **Distant proposals**
- **High acceptance**
- **Avoids random walks**
- **Efficient in high dimensions**

Key Parameters

Step Size ϵ

- Too large: Poor integration, low acceptance
- Too small: Slow exploration, wasted computation
- Optimal: As large as possible while maintaining high acceptance

Trajectory Length L

- Too small: Random walk behavior
- Too large: Wasted computation (loops)
- Challenge: Fixed L is often suboptimal

No-U-Turn Sampler (NUTS)

Automating Trajectory Length

- Builds trajectory until it starts to double back ("U-turn")
- Automatically determines optimal L
- No hand-tuning required!

Practical Considerations

Gradient Requirements

- Need gradients $\nabla U(q) = -\nabla \log \pi(q)$
- Automatic differentiation makes this feasible
- No gradients $\rightarrow$ use Random Walk Metropolis

Mass Matrix M

- Can be adapted to target distribution geometry
- Diagonal M for axis-aligned scaling
- Full M for correlated parameters

Langevin Dynamics: Physical Origins

Brownian Motion in Potential Field

Describes particle motion with friction and random collisions:

$$m\frac{d^2q}{dt^2} = -\nabla U(q) - \gamma\frac{dq}{dt} + \text{random noise}$$

Overdamped Limit (High Friction)

When inertial effects are negligible:

$$\gamma\frac{dq}{dt} = -\nabla U(q) + \sqrt{2\gamma k_B T}\,\eta(t)$$

where $\langle \eta(t)\eta(s)\rangle = \delta(t-s)$

Connection to Sampling

Setting $\gamma = 1$, $k_B T = 1$ gives our sampling equation!

Mathematical Formulation

Stochastic Differential Equation

$$dq(t) = -\nabla U(q)dt + \sqrt{2}dW(t)$$
$$U(q) = -\log \pi(q)$$

Fokker-Planck Equation

Evolution of probability density $\rho(q,t)$:

$$\frac{\partial \rho}{\partial t} = \nabla \cdot (\rho \nabla U) + \Delta \rho$$

Stationary Distribution

Verify that $\rho(q) = \pi(q) \propto e^{-U(q)}$ is $\boxed{stationary}$:

$$\nabla \cdot (\pi \nabla U) + \Delta \pi = \nabla \cdot (\pi \nabla U - \nabla \pi) = 0$$

Unadjusted Langevin Algorithm (ULA)

Euler-Maruyama Discretization

$$q_{k+1} = q_k - \epsilon \nabla U(q_k) + \sqrt{2\epsilon}\xi_k, \quad \xi_k \sim \mathcal{N}(0, I)$$

Properties

- **Bias**: Stationary distribution $\pi_\epsilon \neq \pi$ due to discretization
- **Error**: $\|\pi_\epsilon - \pi\|_{TV} = O(\epsilon)$
- **Simple**: Easy to implement, no accept/reject step
- Useful for optimization and approximate sampling

Limitations

Requires decreasing step sizes $\epsilon_k \to 0$ for exact convergence:

$$\sum \epsilon_k = \infty, \quad \sum \epsilon_k^2 < \infty$$

Metropolis-Adjusted Langevin Algorithm (MALA)

Algorithm

1. Propose: $q^* = q_k - \epsilon \nabla U(q_k) + \sqrt{2\epsilon}\xi_k$
2. Accept with probability:

$$\alpha = \min\left(1, \frac{\pi(q^*)T(q_k|q^*)}{\pi(q_k)T(q^*|q_k)}\right)$$

where $T(x'|x) = \mathcal{N}(x'; x - \epsilon\nabla U(x), 2\epsilon I)$

Optimal Scaling

For d-dimensional distributions:

- Acceptance rate $\approx 57.4\%$ (optimal)
- Step size $\epsilon = O(d^{-1/3})$
- Much better than Random Walk: $\epsilon = O(d^{-1})$

Preconditioned Langevin

Ill-Conditioned Problems

When target has different length scales:

$$q_{k+1} = q_k - \epsilon P \nabla U(q_k) + \sqrt{2\epsilon P}\xi_k$$

Choice of Preconditioner

- **Diagonal**: $P = \text{diag}(\sigma_1^{-2}, \ldots, \sigma_d^{-2})$
- **Fisher information**: $P = I(\theta)^{-1}$
- **Empirical covariance**: $P = \text{Cov}(q)$

MALA with Preconditioning

Proposal becomes:

$$q^* = q_k - \epsilon P \nabla U(q_k) + \sqrt{2\epsilon P}\xi_k$$

Requires careful handling of proposal asymmetry.

Stochastic Gradient Langevin Dynamics (SGLD)

Big Data Setting

When $U(q) = \frac{1}{N}\sum_{i=1}^{N} U_i(q)$ is expensive:

$$q_{k+1} = q_k - \epsilon_k \nabla \hat{U}_B(q_k) + \sqrt{2\epsilon_k}\xi_k$$

where $\nabla \hat{U}_B(q) = \frac{1}{|B|}\sum_{i \in B} \nabla U_i(q)$

Theoretical Guarantees

With decreasing step sizes $\epsilon_k \to 0$:

- Converges to true stationary distribution
- Error analysis available
- Practical trade-off: fixed vs decreasing steps

Summary of MCMC methods

The Sampling Problem

We want to sample from target distribution $\pi(\theta)$ where we can evaluate $\pi(\theta)$ (possibly up to constant) but cannot sample directly.

1. **Random Walk MCMC**: Basic Metropolis-Hastings
2. **Langevin Dynamics**: Gradient-guided random walk
3. **Hamiltonian MCMC**: Physics-inspired momentum dynamics

Common Goal

All methods construct Markov chain with stationary distribution $\pi(\theta)$

Random Walk MCMC (Metropolis-Hastings)

Algorithm

1. Propose: $\theta^* \sim q(\theta^*|\theta_t)$
2. Accept with probability:

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

Typical Proposal

Random walk: $\theta^* = \theta_t + \epsilon\xi$, $\xi \sim \mathcal{N}(0, I)$

Properties

- **Simple**: Easy to implement
- **Flexible**: Works with any proposal
- **Slow**: Random walk behavior
- **Scaling**: $\epsilon = O(d^{-1})$ for optimal acceptance

Langevin Dynamics (MALA)

Gradient-Guided Proposals

Proposal: $\theta^* = \theta_t - \frac{\epsilon^2}{2}\nabla U(\theta_t) + \epsilon\xi$ where $U(\theta) = -\log \pi(\theta)$, $\xi \sim \mathcal{N}(0, I)$

Intuition

Combines gradient descent with random noise:

- Drift toward high probability regions
- Diffusion for exploration

Properties

- **Faster**: Gradient information improves mixing
- **Scaling**: $\epsilon = O(d^{-1/3})$ for optimal acceptance
- **Requires**: Gradients of target distribution

Hamiltonian Monte Carlo (HMC)

Physics-Inspired Dynamics

Introduces momentum variables p and uses Hamiltonian dynamics:

$$\frac{d\theta}{dt} = M^{-1}p$$
$$\frac{dp}{dt} = -\nabla U(\theta)$$

Leapfrog Integration

Discrete simulation with volume preservation:

$$p \leftarrow p - \frac{\epsilon}{2}\nabla U(\theta)$$
$$\theta \leftarrow \theta + \epsilon M^{-1}p$$
$$p \leftarrow p - \frac{\epsilon}{2}\nabla U(\theta)$$

Theoretical Comparison

Property	Random Walk	Langevin	Hamiltonian
Proposal Mechanism	Random	Gradient-guided	Hamiltonian dynamics
Required Gradients	No	Yes	Yes
Optimal Scaling	$O(d^{-1})$	$O(d^{-1/3})$	$O(d^{-1/4})$
Acceptance Rate	23.4%	57.4%	65% (typical)
Mixing Time	Slow	Medium	Fast
Complexity/Step	Low	Medium	High

Table: Theoretical properties for d-dimensional problems

Key Insight

More sophisticated methods use more information (gradients) to achieve better scaling with dimension

Computational Requirements

Per-Iteration Cost

- **RWM**: 1 target evaluation
- **MALA**: 1 gradient + 1 target
- **HMC**: L gradients + L targets

(L = number of leapfrog steps)

Memory

- **RWM**: Stores current state
- **MALA**: Stores current state
- **HMC**: Stores state + momentum

Effective Sample Size (ESS)

- **RWM**: Low ESS per evaluation
- **MALA**: Medium ESS per evaluation
- **HMC**: High ESS per evaluation

Tuning Complexity

- **RWM**: Step size only
- **MALA**: Step size only
- **HMC**: Step size + trajectory length

Performance in Different Scenarios

High-Dimensional Problems

- **RWM**: Becomes impractical for $d > 20$
- **MALA**: Works well for moderate dimensions
- **HMC**: Best for high-dimensional complex distributions

Correlated Distributions

- **RWM**: Struggles with strong correlations
- **MALA**: Handles mild correlations
- **HMC**: Naturally follows correlation structure

Multi-modal Distributions

- **RWM**: May get stuck in local modes
- **MALA**: Better at mode switching
- **HMC**: Can jump between distant modes

When to Use Each Method

Use Random Walk MCMC When:

- Target is low-dimensional ($d < 10$)
- Gradients are unavailable or expensive
- Implementation simplicity is priority
- Distribution is simple and well-conditioned

Use Langevin Dynamics When:

- Moderate dimensions ($d \approx 10 - 100$)
- Gradients are available
- Good balance of simplicity and efficiency needed
- Step size tuning is acceptable

Use Hamiltonian Monte Carlo When:

- High-dimensional complex distributions
- Gradients are available
- Computational efficiency is critical
- Willing to invest in tuning (or use NUTS)

Summary and Recommendations

Evolution of Sampling Methods

- **RWM**: Foundation, simple but inefficient
- **MALA**: Gradient information improves efficiency
- **HMC**: Physical intuition enables optimal exploration

Modern Best Practices

- Start with HMC/NUTS if gradients available
- Use MALA for moderate problems
- Use RWM only for simple low-dimensional cases
- Consider computational cost vs mixing time trade-offs

Key Trade-off

Simplicity vs Efficiency: More sophisticated methods require more implementation effort and tuning but provide dramatically better performance for complex problems.