Markov-Chain Monte-Carlo Sampling

Spring 2024

Instructor: Shandian Zhe
zhe@cs.utah.edu
School of Computing
Outline

• General ideas and Markov chain basics
• Metropolis-Hastings algorithm
• Gibbs sampling
• Hybrid Monte-Carlo
Outline

• General ideas and Markov chain basics
• Metropolis-Hastings algorithm
• Gibbs sampling
• Hybrid Monte-Carlo
MCMC: Goal

- Given a probabilistic model

\[ p(D, z) = p(z)p(D|z) \]

- How to generate samples from the posterior distribution (the samples are NOT necessarily independent!)

\[ z_1, z_2, \ldots, z_N \sim p(z|D) \]
MCMC: Goal

• Given the posterior samples, what can we do?

• A lot of things
  – Approximate the (marginal) posterior posterior over any subset of variable (unlike message-passing)
    \[ p(z|\mathcal{D}) \approx \frac{1}{N} \sum_{n=1}^{N} \delta(z - z_n) \]
  – Estimation of any interested statistics/moments
    \[ \mathbb{E}[f(z)] = \int f(z)p(z|\mathcal{D})dz \approx \frac{1}{N} \sum_{n=1}^{N} f(z_n) \]
  – Predictive distribution
    \[ p(y^*|\mathcal{D}) = \int p(y^*|z)p(z|\mathcal{D})dz \approx \frac{1}{N} \sum_{n=1}^{N} p(y^*|z_n) \]
MCMC: Pros and Cons

• Pros
  – Asymptotic convergence to the true posterior (note: deterministic approximation, such as VI, always has discrepancy with the true posterior)
  – Robust to initialization
  – Empirically best and often used as a gold-standard to test other approximate inference algorithms
  – Samples are more convenient to use than approximate distributions
MCMC: Pros and Cons

• Cons
  – Orders of magnitude slower than VB
  – Hard to diagnosis the convergence
  – Hard for parallelization (sequential sampling nature)
  – Hard for large-scale applications
  – Easily trap into single modes (this is the same as VB)

How to scale up MCMC to big data is a hot research topic!
MCMC: Basic ideas

Sample a sequence of variables using a Markov chain that converges to the desired posterior

\[
\mathbf{z}_1 \rightarrow \mathbf{z}_2 \rightarrow \ldots \rightarrow \mathbf{z}_n \rightarrow \mathbf{z}_{n+1} \rightarrow \ldots
\]

\[
\mathbf{z}_{n+1} \sim p(\mathbf{z}_{n+1}|\mathbf{z}_n) \quad \lim_{n \to \infty} p(\mathbf{z}_n) = p(\mathbf{z}|\mathcal{D})
\]

Therefore, the MCMC samples are strongly correlated!
Basics of Markov chains

• A Markov chain is determined by
  – $p(Z_1)$: we do not care it much in MCMC sampling
  – Transition kernel: determines the speed of convergence

\[
T(z_n \rightarrow z_{n+1}) = p(z_{n+1} | z_n)
\]

if the kernel is the same for all $n$, the Markov chain is called \textit{homogeneous}

The development of MCMC sampling is the art to design the transition kernel
Basics of Markov chains

• What distribution does a MC converge to?
  – Invariant distribution

\[
\int p^*(z')T(z' \rightarrow z)dz' = p^*(z)
\]

We claim that \( p^*(\cdot) \) is invariant to the transition kernel \( T \)

Also called stationary distribution

Obviously, we want to design a kernel to which the target posterior is invariant
Basics of Markov chains

• How to examine invariance?

Sufficient condition (not necessary): *detailed balance*

\[ p^*(z)T(z \rightarrow z') = p^*(z')T(z' \rightarrow z) \]
Basics of Markov chains

• How does *detailed balance* lead to *invariance*?

An MC whose stationary distribution and transition kernel respect detailed balance is called *reversible*.
Basics of Markov chains

• An MC can have multiple stationary distributions; converging to which one depends on $p(z_1)$

• We want our MC only converges to the desired posterior no matter what initial distribution is chosen!

• This property is called **ergodicity**: an ergodic MC only converges to one invariant (stationary) distribution
Basics of Markov chains

• Informally, in an ergodic chain, it is possible to go from *every* state to *every* state (not necessarily in one move)

• An ergodic chain is also called *irreducible*

• The invariant (or stationary) distribution of an ergodic chain is called the *equilibrium* distribution
Basics of Markov chains

• In MCMC sampling procedure
  – Invariance guarantees the samples will converge to the true posterior (unbiased)
  – Ergodicity guarantees the sample space can be fully explored (rather than partially)

• It can be shown that a homogeneous MC will be ergodic, subject only to weak restrictions on the invariant distribution and transitional kernels
Basics of Markov chains

• Conceptually, the sampling contains two stages
  – Before burn-in: the MC has yet converged to the invariant distribution. In practice, we usually set up the maximum number of steps before burn-in, and usually various tricks to verify convergence empirically (e.g., look at trace plots).
  – After burn-in: the MC has converged. Then we generate the posterior samples. To reduce the strong correlation, we often take every M-th sample (e.g., M = 5, 10, 20). We also need to compute the effective sample size (ESS) to ensure the collected samples are enough.
Outline

• General ideas and Markov chain basics
• Metropolis-Hastings algorithm
• Gibbs sampling
• Hybrid Monte-Carlo
Metropolis-Hastings algorithm

• A general framework for MCMC
Metropolis-Hastings algorithm

- A general framework for MCMC

- In each step, we first use a proposal distribution to generate a candidate sample, and then decide whether to accept this new sample
**Metropolis-Hastings algorithm**

- Denote the proposal distribution (not the transition kernel) by \( q(z'|z_n) \), e.g., \( \mathcal{N}(z'|z_n, \sigma^2 I) \). Sample the the proposal \( z' \) first.

- Accept \( z' \) with probability

\[
\min(1, \frac{p(z', D)q(z_n|z')}{p(z_n, D)q(z'|z_n)})
\]

Unnormalized posterior

Jump back

Jump out
Metropolis-Hastings algorithm

- Accept $z'$ with probability

$$
\min(1, \frac{p(z', D)q(z_n|z')}{p(z_n, D)q(z'|z)})
$$

How do we implement it in practice?

Sample a uniform R.V. $u$ in $[0,1]$, and test if

$$
u \leq \exp\left\{ \min(0, \log p(z', D) + \log q(z_n|z') - \log p(z_n, D) - \log q(z'|z_n)) \right\}
$$
Metropolis-Hastings algorithm

• If we accept \( z' \)

\[
\text{Set } \quad z_{n+1} = z'
\]

otherwise

\[
\text{Set } \quad z_{n+1} = z_n
\]

Note: the chain may contain many duplicated samples due to rejections
Metropolis-Hastings algorithm

• Proof: MH guarantees the detailed balance

Given arbitrary \( z_n \) and \( z_{n+1} \), if \( z_{n+1} \neq z_n \), \( z_{n+1} \) must be obtained from accepting a proposal

\[
T(z_n \rightarrow z_{n+1}) = q(z_{n+1}|z_n) \min(1, \frac{p(z_{n+1}, \mathcal{D})q(z_n|z_{n+1})}{p(z_n, \mathcal{D})q(z_{n+1}|z_n)})
\]

\[
= q(z_{n+1}|z_n) \min(1, \frac{p(z_{n+1}, \mathcal{D})/p(\mathcal{D})q(z_n|z_{n+1})}{p(z_n, \mathcal{D})/p(\mathcal{D})q(z_{n+1}|z_n)})
\]

\[
= q(z_{n+1}|z_n) \min(1, \frac{p(z_{n+1}|\mathcal{D})q(z_n|z_{n+1})}{p(z_n|\mathcal{D})q(z_{n+1}|z_n)})
\]


Metropolis-Hastings algorithm

• Proof: MH guarantees the detailed balance

Given arbitrary \( z_n \) and \( z_{n+1} \), if \( z_{n+1} \neq z_n \), \( z_{n+1} \) must be obtained from accepting a proposal

\[
T(z_n \rightarrow z_{n+1}) = q(z_{n+1}|z_n) \min(1, \frac{p(z_{n+1}|\mathcal{D})q(z_n|z_{n+1})}{p(z_n|\mathcal{D})q(z_{n+1}|z_n)})
\]

\[
p(z_n|\mathcal{D})T(z_n \rightarrow z_{n+1}) = p(z_n|\mathcal{D})q(z_{n+1}|z_n) \min(1, \frac{p(z_{n+1}|\mathcal{D})q(z_n|z_{n+1})}{p(z_n|\mathcal{D})q(z_{n+1}|z_n)})
\]

\[
= \min \left( p(z_n|\mathcal{D})q(z_{n+1}|z_n), p(z_{n+1}|\mathcal{D})q(z_n|z_{n+1}) \right)
\]

\[
p(z_{n+1}|\mathcal{D})T(z_{n+1} \rightarrow z_n)
\]

\[
= \min \left( p(z_{n+1}|\mathcal{D})q(z_n|z_{n+1}), p(z_n|\mathcal{D})q(z_{n+1}|z_n) \right)
\]
Metropolis-Hastings algorithm

- Proof: MH guarantees the detailed balance

  if \( z_{n+1} = z_n \)

  \[
  T(z_n \rightarrow z_{n+1}) = \begin{align*}
  &p(\text{reject the proposal}) + \\
  &p(\text{proposal is } z_{n+1} \text{ and accept})
  \end{align*}
  \]

  \[
  p(z_n | D)T(z_n \rightarrow z_{n+1}) = p(z_n | D) \cdot [p(\text{reject the proposal}) + \\
  p(\text{proposal is } z_{n+1} \text{ and accept})]
  \]

  \[
  p(z_{n+1} | D)T(z_{n+1} \rightarrow z_n) = p(z_n | D) \cdot [p(\text{reject the proposal}) + \\
  p(\text{proposal is } z_n \text{ and accept})]
  \]
Metropolis algorithm

• If we choose a symmetric proposal distribution

\[ q(z' | z_n) = q(z_n | z') \quad \text{e.g.,} \quad \mathcal{N}(z' | z_n, \sigma^2 I) \]

Accept probability:

\[
\min(1, \frac{p(z', \mathcal{D}) q(z_n | z')}{p(z_n, \mathcal{D}) q(z' | z_n)})
\]

\[
= \min(1, \frac{p(z', \mathcal{D})}{p(z_n, \mathcal{D})})
\]

If the proposal increases the model probability, the accept rate is one!
Nightmare: random walk behavior

• We need to collect samples that fit the target posterior (e.g., their histogram should be more and more like the posterior). That means, we require many samples on the high-density regions and much less samples on the low-density regions

• However, if the proposals are generated like a random walk through the sample space, a great many proposals will be discarded (due to being in the low-density regions); and much computational cost is wasted
Nightmare: random walk behavior

• Take the commonly used Gaussian proposal as an example

• So a key goal to design MCMC algorithms is to reduce random walk behavior!
Outline

• General ideas and Markov chain basics
• Metropolis-Hastings algorithm
• Gibbs sampling
• Hybrid Monte-Carlo
Gibbs sampling

• A special type of MH algorithm
• Use conditional distribution to sample each single (or subset of) random variable in the model
• Accept rate is always one
• A good choice when the conditional distribution is tractable and easy to draw samples
Gibbs sampling

\[ \mathbf{z} = [z_1, \ldots, z_m]^\top \quad p(\mathbf{z}, D) = p(z_1, \ldots, z_m, D) \]

Assume each \( p(z_i | \mathbf{z}_{-i}, D) \) is tractable and easy to generate samples

\[ \mathbf{z}_{-i} = [z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_m]^\top \]
Gibbs sampling

- Initialize $\mathbf{z}^{(1)} = [z_1^{(1)}, \ldots, z_m^{(1)}]^{\top}$
- For $t = 1, \ldots, T$
  - Sample $z_1^{(n+1)} \sim p(z_1|z_2^{(n)}, z_3^{(n)}, \ldots, z_m^{(n)}, D)$
  - Sample $z_2^{(n+1)} \sim p(z_2|z_1^{(n+1)}, z_3^{(n)}, \ldots, z_m^{(n)}, D)$
  - Sample $z_3^{(n+1)} \sim p(z_3|z_1^{(n+1)}, z_2^{(n+1)}, \ldots, z_m^{(n)}, D)$
  - Sample $z_j^{(n+1)} \sim p(z_j|z_1^{(n+1)}, \ldots, z_{j-1}^{(n+1)}, z_{j+1} \ldots, z_m^{(n)}, D)$
  
  ...  

  - Sample $z_m^{(n+1)} \sim p(z_j|z_1^{(n+1)}, z_2^{(n+1)}, \ldots, z_{m-1}^{(n+1)}, D)$
Gibbs sampling

- We can also partition the random variables into sub-vectors, and perform similar alternative sampling

\[ z = [z_1, \ldots, z_t]^\top \]

\[ p(z_i | z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_t, D) \]

- This is called block Gibbs sampling
Gibbs sampling: examples

• Matrix factorization

<table>
<thead>
<tr>
<th></th>
<th>Movie 1</th>
<th>Movie 2</th>
<th>Movie 3</th>
<th>Movie 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>User 1</td>
<td>3.2</td>
<td>1.2</td>
<td>5</td>
<td>4.0</td>
</tr>
<tr>
<td>User 2</td>
<td>2.2</td>
<td>1.0</td>
<td>?</td>
<td>3.0</td>
</tr>
<tr>
<td>User 3</td>
<td>2.5</td>
<td>?</td>
<td>4.3</td>
<td>?</td>
</tr>
</tbody>
</table>
Gibbs sampling: examples

<table>
<thead>
<tr>
<th></th>
<th>Movie 1</th>
<th>Movie 2</th>
<th>Movie 3</th>
<th>Movie 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>User 1</td>
<td>3.2</td>
<td>1.2</td>
<td>5</td>
<td>4.0</td>
</tr>
<tr>
<td>User 2</td>
<td>2.2</td>
<td>1.0</td>
<td>?</td>
<td>3.0</td>
</tr>
<tr>
<td>User 3</td>
<td>2.5</td>
<td>?</td>
<td>4.3</td>
<td>?</td>
</tr>
</tbody>
</table>

For each user $i$, introduce a $k$-dimensional latent feature vector $\mathbf{u}_i$.

For each movie $j$, introduce a $k$-dimensional latent feature vector $\mathbf{v}_j$.

$$ p(\mathbf{u}_i) = \mathcal{N}(\mathbf{u}_i | \mathbf{0}, \mathbf{I}) \quad p(\mathbf{v}_j) = \mathcal{N}(\mathbf{v}_j | \mathbf{0}, \mathbf{I}) $$

The rating is sampled from a Gaussian

$$ p(R_{ij} | \mathbf{U}, \mathbf{V}) = \mathcal{N}(R_{ij} | \mathbf{u}_i^\top \mathbf{v}_j, \tau) $$
Gibbs sampling: examples

<table>
<thead>
<tr>
<th></th>
<th>Movie 1</th>
<th>Movie 2</th>
<th>Movie 3</th>
<th>Movie 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>User 1</td>
<td>3.2</td>
<td>1.2</td>
<td>5</td>
<td>4.0</td>
</tr>
<tr>
<td>User 2</td>
<td>2.2</td>
<td>1.0</td>
<td>?</td>
<td>3.0</td>
</tr>
<tr>
<td>User 3</td>
<td>2.5</td>
<td>?</td>
<td>4.3</td>
<td>?</td>
</tr>
</tbody>
</table>

The joint probability

\[
p(U, V, R) = \prod_{i} p(u_i) \prod_{j} p(v_j) \prod_{(i,j) \in \mathcal{O}} p(r_{ij} | u_i^\top v_j, \tau)
\]
Gibbs sampling: examples

\[ p(U, V, R) = \prod_{i} p(u_i) \prod_{j} p(v_j) \prod_{(i,j) \in O} p(r_{ij} | u_i^T v_j, \tau) \]

We can use Gibbs sampling to sequentially sample each \( u_i \) and \( v_j \)

The conditional distribution will be Gaussian!
Gibbs sampling: correctness

• Proof: the target posterior is invariant to the chain

What is the transition kernel?

\[ T(z^{(n)} \rightarrow z^{(n+1)}) = p(z_1^{(n+1)} | z_2^{(n)}, \ldots, z_m^{(n)}, D) \]
\[ \cdot p(z_2^{(n+1)} | z_1^{(n+1)}, z_3^{(n)}, \ldots, z_m^{(n)}, D) \]
\[ \ldots \]
\[ \cdot p(z_m^{(n+1)} | z_1^{(n+1)}, z_2^{(n+1)}, \ldots, z_{m-1}^{(n+1)}, D) \]
Gibbs sampling: correctness

- Proof: the target posterior is invariant to the chain

\[ z^{(n)} \sim p(z|D) \] respect the target posterior

\[
T(z^{(n)} \rightarrow z^{(n+1)}) = p(z_1^{(n+1)}|z_2^{(n)}, \ldots, z_m^{(n)}, D) \\
\times p(z_2^{(n+1)}|z_1^{(n+1)}, z_3^{(n)}, \ldots, z_m^{(n)}, D) \\
\times \ldots \\
\times p(z_m^{(n+1)}|z_1^{(n+1)}, z_2^{(n+1)}, \ldots, z_{m-1}^{(n+1)}, D)
\]

\[
[z_1^{(n+1)}, z_2^{(n+1)}, \ldots, z_m^{(n+1)}]^{\top} \]

\[
[z_1^{(n)}, z_2^{(n)}, \ldots, z_m^{(n)}]^{\top} \\
\ldots \\
[z_1^{(n+1)}, \ldots, z_m^{(n+1)}] \\
\ldots \\
z^{(n+1)}
\]
Gibbs sampling: correctness

• Note that you need also to ensure ergodicity
• A sufficient condition is that none of the conditional distributions be zero anywhere in the sample space (not hard for continuous distributions)
• If the sufficient condition is NOT satisfied, you must explicitly prove the ergodicity!
Gibbs sampling: An instance of MH

• One iteration of Gibbs sampling is equivalent to $m$ steps of MH updates, each step with accept prob. 1.

• Let us look at one step, w.l.o.g., sample the first element (the other elements are the same).
Gibbs sampling: An instance of MH

- Let us look at one step, w.l.o.g., sampling the first element (sampling the other elements are the same)

\[
\mathbf{z}_n = [z_1^{(n)}, z_2^{(n)}, \ldots, z_m^{(n)}]^\top \quad \rightarrow \quad \mathbf{z}' = [z_1^{(n+1)}, z_2^{(n)}, \ldots, z_m^{(n)}]^\top
\]

Acceptance probability

\[
\min \left( 1, \frac{p(z_1^{(n+1)}, z_2^{(n)}, \ldots, z_m^{(n)}, \mathcal{D})p(z_1^{(n)} | z_2^{(n)}, \ldots, z_m^{(n)}, \mathcal{D})}{p(z_1^{(n)}, z_2^{(n)}, \ldots, z_m^{(n)}, \mathcal{D})p(z_1^{(n+1)} | z_2^{(n)}, \ldots, z_m^{(n)}, \mathcal{D})} \right)
\]

\[
\min \left( 1, \frac{p(z_1^{(n+1)} | z_2^{(n)}, \ldots, z_m^{(n)}, \mathcal{D})p(z_1^{(n)} | z_2^{(n)}, \ldots, z_m^{(n)}, \mathcal{D})}{p(z_1^{(n)} | z_2^{(n)}, \ldots, z_m^{(n)}, \mathcal{D})p(z_1^{(n+1)} | z_2^{(n)}, \ldots, z_m^{(n)}, \mathcal{D})} \right) = 1
\]
Gibbs sampling: inefficient exploration

- Although Gibbs sampling won’t reject samples, it may still suffer from inefficient exploration due to strong correlations.
Outline

• General ideas and Markov chain basics
• Metropolis-Hastings algorithm
• Gibbs sampling
• Hybrid Monte-Carlo
The MCMC algorithms we learned so far

• Random walk behavior --- waste a lot of samples
• High correlation between different RVs --- slow exploration
• Can we address both problems?
Hybrid Monte-Carlo Sampling (HMC)

• Also called Hamiltonian MC
• An augmented approach
• Turn the probability to the energy of a physical system
• Augment with other physical properties
• Use the evolution of the physical system (usually described by a set of partial/ordinary differential equations)
• Theoretically can explore the sample space more efficiently, acceptance prob = 1
• Practically limited by the numerical integration error.
Hamiltonian system

• Consider a small ball in a $m$-dimensional space, without any friction
• Given an initial position and momentum, how does the ball move?
Hamiltonian system

• Characterize how the system evolves
• \( z(t) \): position vector at time \( t \)
• Potential energy: \( U(z(t)) \)
• \( r(t) \): momentum vector at time \( t \)
• Kinetic energy: \( K(r(t)) \)
• Total energy: \( H(z, r) = U(z) + K(r) \)
Hamiltonian system

- $z(t)$: position vector at time $t$
- Potential energy: $U(z(t))$
- $r(t)$: momentum vector at time $t$
- Kinetic energy: $K(r(t))$
- Total energy: $H(z, r) = U(z) + K(r)$

Evolving:

\[
\frac{d z_i}{dt} = \frac{\partial H}{\partial r_i}, \quad \frac{d r_i}{dt} = -\frac{\partial H}{\partial z_i}
\]

\[z = [z_1, \ldots, z_m]^\top\]
\[r = [r_1, \ldots, r_m]^\top\]
Hamiltonian system

• How to map our probabilistic model into the system?

\[ p(z, D) = p(z_1, \ldots, z_m, D) \]

• We take

\[ U(z) = - \log (p(z, D)) \]

\[ K(r) = \frac{1}{2} r^\top \mathbf{M}^{-1} r \quad \text{often takes identity/diagonal matrix} \]

\[ H(z, r) = U(z) + K(r) \quad \text{energy dist.} \]

\[ p(z, r) \propto \exp \left( - H(z, r) \right) \]

What does it include?
Hamiltonian system

\[ U(z) = -\log \left( p(z, D) \right) \]
\[ K(r) = \frac{1}{2} r^\top M^{-1} r \]
\[ H(z, r) = U(z) + K(r) \]

\[ \frac{dz_i}{dt} = \frac{\partial H}{\partial r_i} \]
\[ \frac{dr_i}{dt} = -\frac{\partial H}{\partial z_i} \]

\[ \frac{dz_i}{dt} = [M^{-1} r]_i \]
\[ \frac{dr_i}{dt} = -\frac{\partial U}{\partial z_i} \]
The key idea: use the current sample $z_n$ and random sample of $r$, as the initial state of the Hamiltonian system; and then evolve the system to a time $t$, pick the $z(t)$ as the proposal and test whether to accept it as $z_{n+1}$.

Note: the proposal is not randomly generated; it is generated deterministically.
Hamiltonian system

• Nice properties to guarantee the detailed balance

1. Reversibility:

\[
p^*(\mathbf{z}) T(\mathbf{z} \rightarrow \mathbf{z}') = p^*(\mathbf{z}') T(\mathbf{z}' \rightarrow \mathbf{z})
\]

Now \( T \) is a delta function, we need to be able to jump back!
Numerical Integration

• Nice properties to guarantee the detailed balance

2. Conservation: \( \frac{dH}{dt} = 0 \)  Totally energy does not change

3. Volume preservation: Determinant of Jacobian is always 1

For any \( t,s \geq 0 \):

\[
|\det \frac{\partial [u(t + s), r(t + s)]^T}{\partial [u(t), r(t)]^T}| = 1
\]

Volume does not change after transformation
General theorem (proof omitted)

Consider an arbitrary dynamic system \( \Psi_t \)

Let \( \mathbf{v} = (\mathbf{z}, \mathbf{r}) \) be the extended variable. Define \( \mathbf{v}' = \Psi_t(\mathbf{v}) \)

If the following conditions are satisfied:

- \( \Psi_t \) is reversible under \( R \), i.e., \( \mathbf{v} = \Psi_t^{-1}(\mathbf{v}') = R(\Psi_t(R(\mathbf{v}'))) \)
- \( R \) is an involution, i.e., \( R \circ R(\mathbf{x}) = \mathbf{x} \)
- The proposed sample \( R(\mathbf{v}') \) is accepted with prob. 
  \[
  \min\{1, \frac{p(R(\mathbf{v}'))}{p(\mathbf{v})} | \det \frac{\partial R \circ \Psi_t(\mathbf{v})}{\partial \mathbf{v}} | \} \quad \text{otherwise keep } \mathbf{v}
  \]

Then \( p(\mathbf{v}) \) is stationary distribution of the Markov chain generated by this \( \Psi_t \) and accept test
**General theorem (proof omitted)**

Consider an arbitrary dynamic system $\Psi_t$

Let $\mathbf{v} = (\mathbf{z}, \mathbf{r})$ be the extended variable. Define $\mathbf{v}' = \Psi_t(\mathbf{v})$

If the following conditions are satisfied:

- $\Psi_t$ is reversible under $R$, i.e., $\mathbf{v} = \Psi_t^{-1}(\mathbf{v}') = R(\Psi_t(R(\mathbf{v}')))$
- $R$ is an involution, i.e., $R \circ R(\mathbf{x}) = \mathbf{x}$  
  - $R$ is negating the momentum
- The proposed sample $R(\mathbf{v}')$ is accepted with prob.

$$\min\{1, \frac{p(R(\mathbf{v}'))}{p(\mathbf{v})} |\det \frac{\partial R \circ \Psi_t(\mathbf{v})}{\partial \mathbf{v}}|\} \quad \text{otherwise keep } \mathbf{v}$$

Then $p(\mathbf{v})$ is stationary distribution of the Markov chain generated by this $\Psi_t$ and accept test

*Apply the theorem to Hamiltonian system, the accept rate is always 1*
However, we cannot exactly evolve Hamiltonian system (do not know solution)

\[
U(z) = - \log(p(z, D))
\]
\[
K(r) = \frac{1}{2} r^\top M^{-1} r
\]

\[
H(z, r) = U(z) + K(r)
\]

\[
\begin{align*}
\frac{dz_i}{dt} &= \frac{\partial H}{\partial r_i} \\
\frac{dr_i}{dt} &= -\frac{\partial H}{\partial z_i}
\end{align*}
\]

\[
\begin{align*}
\frac{dz_i}{dt} &= [M^{-1} r]_i \\
\frac{dr_i}{dt} &= -\frac{\partial U}{\partial z_i}
\end{align*}
\]
Numerical integration

\[
\frac{d\tilde{z}_i}{dt} = [M^{-1}r]_i \quad \text{In practice we often choose} \\
\frac{dr_i}{dt} = -\frac{\partial U}{\partial z_i}
\]

Euler’s method: choose step size $\mathcal{E}$, and # of step size $L$

\[
\begin{align*}
    r_i(t + \mathcal{E}) &= r_i(t) + \mathcal{E} \frac{dr_i(t)}{dt} = r_i(t) - \mathcal{E} \frac{\partial U(z(t))}{\partial z_i} \\
    z_i(t + \mathcal{E}) &= z_i(t) + \mathcal{E} \frac{dz_i(t)}{dt} = z_i(t) + \mathcal{E} \frac{r_i(t)}{s_i}
\end{align*}
\]
Leapfrog method

- Euler’s method is a first-order method $O(\varepsilon)$
- In practice, people choose Leapfrog method, a second-order method $O(\varepsilon^2)$

\[
\begin{align*}
    r_i(t + \varepsilon/2) &= r_i(t) - (\varepsilon/2) \frac{\partial U(z)}{\partial z_i} \\
    z_i(t + \varepsilon) &= z_i(t) + \varepsilon \frac{r_i(t + \varepsilon/2)}{s_i} \\
    r_i(t + \varepsilon) &= r_i(t + \varepsilon/2) - (\varepsilon/2) \frac{\partial U(z(t + \varepsilon))}{\partial z_i}
\end{align*}
\]
Leapfrog method \((\varepsilon, L)\)

- Key properties
  - Reversibility under momentum negation
  - Volume preservation: each leap-frog step is a shear transformation and preserves volumes

Question: does conservation still hold?
Leapfrog method ($\varepsilon, L$)

• Key properties
  – Reversibility under momentum negation
  – Volume preservation: each leap-frog step is a shear transformation and preserves volumes

Question: does conservation still hold? No, because it is a numerical approximation!
General theorem (proof omitted)

Consider an arbitrary dynamic system $\Psi_t$

Let $\mathbf{v} = (z, r)$ be the extended variable. Define $\mathbf{v}' = \Psi_t(\mathbf{v})$

If the following conditions are satisfied:

- $\Psi_t$ is reversible under $R$, i.e., $\mathbf{v} = \Psi_t^{-1}(\mathbf{v}') = R(\Psi_t(R(\mathbf{v}')))$
- $R$ is an involution, i.e., $R \circ R(\mathbf{x}) = \mathbf{x}$, \textbf{R: momentum negation}
- The proposed sample $R(\mathbf{v}')$ is accepted with prob.
  \[
  \min\{1, \frac{p(R(\mathbf{v}'))}{p(\mathbf{v})} | \det \frac{\partial R \circ \Psi_t(\mathbf{v})}{\partial \mathbf{v}} | \} \quad \text{otherwise keep } \mathbf{v}
  \]

Then $p(\mathbf{v})$ is stationary distribution of the Markov chain generated by this $\Psi_t$ and accept test

Note that: due to the numerical error, the accept rate is not guaranteed to be 1
HMC based on leap-frog

• We augment the latent variable $z$, with momentum variables $r$

• Construct energy distribution

\[
U(z) = - \log (p(z, D)) \quad K(r) = \frac{1}{2} r^\top M^{-1} r
\]

\[
H(z, r) = U(z) + K(r)
\]

\[
p(z, r) \propto \exp (- H(z, r))
\]

• We construct a MC to generate samples from $p(z, r)$
HMC based on leap-frog

- Step 1: generate new sample for \( r \)
  \[ r_i \sim \mathcal{N}(r_i|0, s_i) \]
  (This is a Gibbs sampling step, why? Because the \( r \) and \( z \) are independent!)

- Step 2: start with current \((z, r)\) and run Leap-frog for \( L \) steps with step size \( \epsilon \), obtain \((z', r')\), set \( r' = -r' \), accept \( z' \) with probability
  \[
  \min\{1, \exp \left( -H(z', r') + H(z, r) \right) \} = \min\{1, \exp \left( -U(z') - K(r') + U(z) + K(r) \right) \}
  \]
  otherwise keep \( z \)
  (This is a Metropolis-hasting step)

- Repeat Step 1 & 2 until get all the samples after burn-in
HMC-correctness

• Combining multiple Metropolis-hasting steps still yields one valid MH step, so the target posterior is invariant to the transitional kernel of the chain

• Ergodicity: typically satisfied because any value can be sampled from the momentum; only failed when the Leapfrog will produce periodicity; we can overcome this issue by randomly choosing $\epsilon$ and $L$ routinely.
HMC applications

- Apply to continuous distributions only, because Leapfrog needs the gradient information
- Very powerful MCMC algorithms.
- Usually much better than original Metropolis Hasting
HMC discussion

• There is a trade-off for the choice \((\epsilon, L)\) in the Leapfrog

\[
\min\{1, \exp \left( -H(z', r') + H(z, r) \right) \}
\]

• Large \(\epsilon\) and \(L\) will allow you to explore the space further away, but increase the numerical error and lower the acceptance rate.

• Small \(\epsilon\) and \(L\) will be more accurate and so the acceptance rate increases, but the generated samples are not distant.

• In practice, it is very important to tune the two parameters!
What you need know

- Basic idea of MCMC
- Key concepts: transitional kernel, stationary/invariant/equilibrium distribution, detailed balance...
- Metropolis Hasting and random walk behavior
- Gibbs sampling
- Hybrid Monte-Carlo sampling
- You should be able to implement these algorithms!