Markov-Chain Monte-Carlo Sampling

Spring 2024

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

THE UNIVERSITY OF UTAH

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

$$p(\mathcal{D}, \mathbf{z}) = p(\mathbf{z})p(\mathcal{D}|\mathbf{z})$$

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

$$\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N \sim p(\mathbf{z}|\mathcal{D})$$

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)

$$p(\mathbf{z}|\mathcal{D}) \approx \frac{1}{N}\sum_{n=1}^{N}\delta(\mathbf{z}-\mathbf{z}_n)$$

  - Estimation of any interested statistics/moments

$$\mathbb{E}[f(\mathbf{z})] = \int f(\mathbf{z})p(\mathbf{z}|\mathcal{D})\mathrm{d}\mathbf{z} \approx \frac{1}{N}\sum_{n=1}^{N} f(\mathbf{z}_n)$$

  - Predictive distribution

$$p(\mathbf{y}^*|\mathcal{D}) = \int p(\mathbf{y}^*|\mathbf{z})p(\mathbf{z}|\mathcal{D})\mathrm{d}\mathbf{z} \approx \frac{1}{N}\sum_{n=1}^{N} p(\mathbf{y}^*|\mathbf{z}_n)$$

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

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

$$\mathbf{z}_{n+1} \sim p\big(\mathbf{z}_{n+1} \big| \mathbf{z}_n\big) \qquad \lim_{n \rightarrow \infty} p(\mathbf{z}_n) = p(\mathbf{z}|\mathcal{D})$$

Therefore, the MCMC samples are strongly correlated!

Basics of Markov chains

- A Markov chain is determined by
  - $p(\mathbf{Z}_1)$: we do not care it much in MCMC sampling
  - Transition kernel: determines the speed of convergence

$$T(\mathbf{z}_n \rightarrow \mathbf{z}_{n+1}) = p(\mathbf{z}_{n+1} | \mathbf{z}_n)$$

if the kernel is the same for all n, the Markov chain is called ***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

$$\int p^*(\mathbf{z}')T(\mathbf{z}' \to \mathbf{z})\mathrm{d}\mathbf{z}' = p^*(\mathbf{z})$$

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***

$$p^*(\mathbf{z})T(\mathbf{z} \to \mathbf{z}') = p^*(\mathbf{z}')T(\mathbf{z}' \to \mathbf{z})$$

Basics of Markov chains

- How does *detailed balance* lead to invariance?

Let us prove it!

An MC whose stationary distribution and transition kernel respect detailed balance is called ***reversable***

Basics of Markov chains

- An MC can have multiple stationary distributions; converging to which one depends on $p(\mathbf{z}_1)$
- 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 # 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 $q(\mathbf{z}'|\mathbf{z}_n)$, e.g., $\mathcal{N}(\mathbf{z}'|\mathbf{z}_n, \sigma^2\mathbf{I})$. Sample the the proposal $\mathbf{z}'$ first.

- Accept $\mathbf{z}'$ with probability

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

Jump back

Jump out

Unnormalized posterior

Metropolis-Hastings algorithm

- Accept $\mathbf{z}'$ with probability

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

Jump back

Jump out

Unnormalized posterior

How do we implement it in practice?

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

$$u \leq \exp\left\{\min\left(0, \log p(\mathbf{z}', \mathcal{D}) + \log q(\mathbf{z}_n|\mathbf{z}') - \log p(\mathbf{z}_n, \mathcal{D}) - \log q(\mathbf{z}'|\mathbf{z}_n)\right)\right\}$$

Metropolis-Hastings algorithm

- If we accept $\mathbf{z}'$

Set $\mathbf{z}_{n+1} = \mathbf{z}'$

otherwise

Set $\mathbf{z}_{n+1} = \mathbf{z}_n$

Note: the chain may contain many duplicated samples due to rejections

Metropolis-Hastings algorithm

- Proof: MH guarantees the detailed balance

  Given arbitrary $\boldsymbol{z}_n$ and $\boldsymbol{z}_{n+1}$, if $\boldsymbol{z}_{n+1} \neq \boldsymbol{z}_n$, $\boldsymbol{z}_{n+1}$ must be obtained from accepting a proposal

$$T(\mathbf{z}_n \rightarrow \mathbf{z}_{n+1}) = q(\mathbf{z}_{n+1}|\mathbf{z}_n)\min(1, \frac{p(\mathbf{z}_{n+1}, \mathcal{D})q(\mathbf{z}_n|\mathbf{z}_{n+1})}{p(\mathbf{z}_n, \mathcal{D})q(\mathbf{z}_{n+1}|\mathbf{z}_n)})$$

$$= q(\mathbf{z}_{n+1}|\mathbf{z}_n)\min(1, \frac{\boxed{p(\mathbf{z}_{n+1}, \mathcal{D})/p(\mathcal{D})}q(\mathbf{z}_n|\mathbf{z}_{n+1})}{\boxed{p(\mathbf{z}_n, \mathcal{D})/p(\mathcal{D})}q(\mathbf{z}_{n+1}|\mathbf{z}_n)})$$

$$= q(\mathbf{z}_{n+1}|\mathbf{z}_n)\min(1, \frac{p(\mathbf{z}_{n+1}|\mathcal{D})q(\mathbf{z}_n|\mathbf{z}_{n+1})}{p(\mathbf{z}_n|\mathcal{D})q(\mathbf{z}_{n+1}|\mathbf{z}_n)})$$

Metropolis-Hastings algorithm

- Proof: MH guarantees the detailed balance

  Given arbitrary $\boldsymbol{z}_n$ and $\boldsymbol{z}_{n+1}$, if $\boldsymbol{z}_{n+1} \neq \boldsymbol{z}_n$, $\boldsymbol{z}_{n+1}$ must be obtained from accepting a proposal

$$T(\mathbf{z}_n \rightarrow \mathbf{z}_{n+1}) = q(\mathbf{z}_{n+1}|\mathbf{z}_n)\min(1, \frac{p(\mathbf{z}_{n+1}|\mathcal{D})q(\mathbf{z}_n|\mathbf{z}_{n+1})}{p(\mathbf{z}_n|\mathcal{D})q(\mathbf{z}_{n+1}|\mathbf{z}_n)})$$

$$p(\mathbf{z}_n|\mathcal{D})T(\mathbf{z}_n \rightarrow \mathbf{z}_{n+1}) = p(\mathbf{z}_n|\mathcal{D})q(\mathbf{z}_{n+1}|\mathbf{z}_n)\min(1, \frac{p(\mathbf{z}_{n+1}|\mathcal{D})q(\mathbf{z}_n|\mathbf{z}_{n+1})}{p(\mathbf{z}_n|\mathcal{D})q(\mathbf{z}_{n+1}|\mathbf{z}_n)})$$

$$= \min\left(p(\mathbf{z}_n|\mathcal{D})q(\mathbf{z}_{n+1}|\mathbf{z}_n), p(\mathbf{z}_{n+1}|\mathcal{D})q(\mathbf{z}_n|\mathbf{z}_{n+1})\right)$$

$$\Vert$$

$$p(\mathbf{z}_{n+1}|\mathcal{D})T(\mathbf{z}_{n+1} \rightarrow \mathbf{z}_n)$$

$$= \min\left(p(\mathbf{z}_{n+1}|\mathcal{D})q(\mathbf{z}_n|\mathbf{z}_{n+1}), p(\mathbf{z}_n|\mathcal{D})q(\mathbf{z}_{n+1}|\mathbf{z}_n)\right)$$

Metropolis-Hastings algorithm

- Proof: MH guarantees the detailed balance

if $\boldsymbol{z}_{n+1} = \boldsymbol{z}_n$

$$T(\mathbf{z}_n \rightarrow \mathbf{z}_{n+1}) = \text{ } p(\text{reject the proposal}) + p(\text{proposal is } \boldsymbol{z}_{n+1} \text{ and accept})$$

$$p(\mathbf{z}_n|\mathcal{D})T(\mathbf{z}_n \rightarrow \mathbf{z}_{n+1}) = p(\mathbf{z}_n|\mathcal{D})\cdot [p(\text{reject the proposal}) + p(\text{proposal is } \boldsymbol{z}_{n+1} \text{ and accept})]$$

$$p(\mathbf{z}_{n+1}|\mathcal{D})T(\mathbf{z}_{n+1} \rightarrow \mathbf{z}_n) = p(\mathbf{z}_n|\mathcal{D})\cdot [p(\text{reject the proposal}) + p(\text{proposal is } \boldsymbol{z}_n \text{ and accept})]$$

Metropolis algorithm

- If we choose a symmetric proposal distribution

$$q(\mathbf{z}'|\mathbf{z}_n) = q(\mathbf{z}_n|\mathbf{z}') \qquad \text{e.g.,} \qquad \mathcal{N}(\mathbf{z}'|\mathbf{z}_n, \sigma^2\mathbf{I})$$

Accept probability: $\min(1, \frac{p(\mathbf{z}', \mathcal{D})q(\mathbf{z}_n|\mathbf{z}')}{p(\mathbf{z}_n, \mathcal{D})q(\mathbf{z}'|\mathbf{z}_n)})$

$$= \min\left(1, \frac{p(\mathbf{z}', \mathcal{D})}{p(\mathbf{z}_n, \mathcal{D})}\right)$$

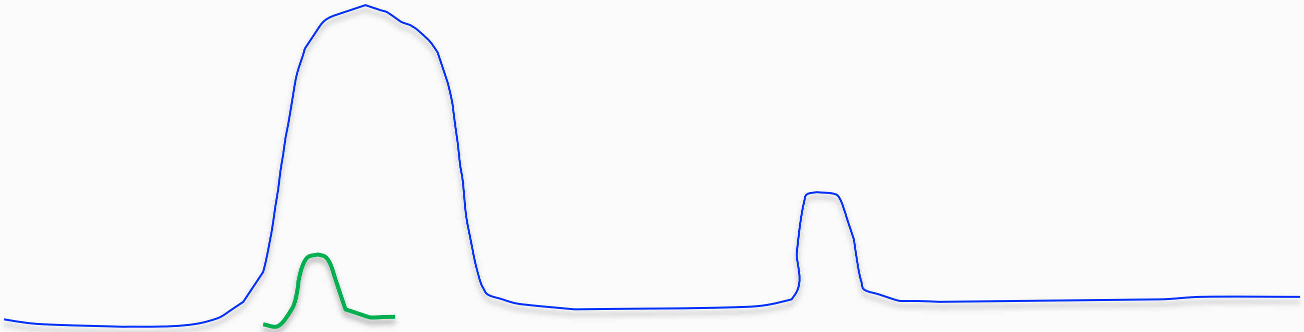
If the proposal increases the model probability, the accept rate is one!

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

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

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

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

Gibbs sampling

- Initialize $\boldsymbol{z}^{(1)} = [z_1^{(1)}, \ldots, z_m^{(1)}]^\top$
- For t = 1,…,T
  - Sample $z_1^{(n+1)} \sim p(z_1 | z_2^{(n)}, z_3^{(n)}, \ldots, z_m^{(n)}, \mathcal{D})$
  - Sample $z_2^{(n+1)} \sim p(z_2 | z_1^{(n+1)}, z_3^{(n)}, \ldots, z_m^{(n)}, \mathcal{D})$
  - Sample $z_3^{(n+1)} \sim p(z_3 | z_1^{(n+1)}, z_2^{(n+1)}, \ldots, z_m^{(n)}, \mathcal{D})$

  - Sample $z_j^{(n+1)} \sim p(z_j | z_1^{(n+1)}, \ldots z_{j-1}^{(n+1)}, z_{j+1}^{(n)} \ldots, z_m^{(n)}, \mathcal{D})$

  …

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

Gibbs sampling

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

$$\mathbf{z} = [\mathbf{z}_1, \ldots, \mathbf{z}_t]^\top$$

$$p(\mathbf{z}_i | \mathbf{z}_1, \ldots, \mathbf{z}_{i-1}, \mathbf{z}_{i+1}, \ldots, \mathbf{z}_t, \mathcal{D})$$

- This is called block Gibbs sampling

Gibbs sampling: examples

- Matrix factorization

	Movie 1	Movie 2	Movie 3	Movie 4
User 1	3.2	1.2	5	4.0
User 2	2.2	1.0	?	3.0
User 3	2.5	?	4.3	?

Gibbs sampling: examples

	Movie 1	Movie 2	Movie 3	Movie 4
User 1	3.2	1.2	5	4.0
User 2	2.2	1.0	?	3.0
User 3	2.5	?	4.3	?

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 $\mathbf{v}_j$

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

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

	Movie 1	Movie 2	Movie 3	Movie 4
User 1	3.2	1.2	5	4.0
User 2	2.2	1.0	?	3.0
User 3	2.5	?	4.3	?

The joint probability

$$p(\mathbf{U}, \mathbf{V}, \mathbf{R}) = \prod_i p(\mathbf{u}_i) \prod_j p(\mathbf{v}_j) \prod_{(i,j)\in\mathcal{O}} p(r_{ij} | \mathbf{u}_i^\top \mathbf{v}_j, \tau)$$

Gibbs sampling: examples

$$p(\mathbf{U}, \mathbf{V}, \mathbf{R}) \\ = \prod_i p(\mathbf{u}_i) \prod_j p(\mathbf{v}_j) \prod_{(i,j)\in\mathcal{O}} p(r_{ij}|\mathbf{u}_i^\top \mathbf{v}_j, \tau)$$

We can use Gibbs sampling to sequentially sample each $\mathbf{u}_i$ and $\mathbf{v}_j$

The conditional distribution will be Gaussian!

Gibbs sampling: correctness

- Proof: the target posterior is invariant to the chain

What is the transition kernel?

$$
\begin{aligned}
&T(\mathbf{z}^{(n)} \to \mathbf{z}^{(n+1)}) \\
&= p(z_1^{(n+1)} | z_2^{(n)}, \ldots, z_m^{(n)}, \mathcal{D}) \\
&\cdot p(z_2^{(n+1)} | z_1^{(n+1)}, z_3^{(n)}, \ldots, z_m^{(n)}, \mathcal{D}) \\
&\cdots \\
&\cdot p(z_m^{(n+1)} | z_1^{(n+1)}, z_2^{(n+1)}, \ldots, z_{m-1}^{(n+1)}, \mathcal{D})
\end{aligned}
$$

m steps

Gibbs sampling: correctness

- Proof: the target posterior is invariant to the chain

if $\mathbf{z}^{(n)} \sim p(\mathbf{z}|\mathcal{D})$ respect the target posterior

$$T(\mathbf{z}^{(n)} \rightarrow \mathbf{z}^{(n+1)})$$

$$= p(z_1^{(n+1)}|z_2^{(n)}, \ldots, z_m^{(n)}, \mathcal{D}) \qquad [z_1^{(n+1)}, z_2^{(n)}, \ldots, z_m^{(n)}]^\top$$

$$\cdot\, p(z_2^{(n+1)}|z_1^{(n+1)}, z_3^{(n)}, \ldots, z_m^{(n)}, \mathcal{D}) \qquad [z_1^{(n+1)}, z_2^{(n+1)}, z_3^{(n)} \ldots, z_m^{(n)}]^\top$$

$$\cdots \qquad \cdots$$

$$\cdot\, p(z_m^{(n+1)}|z_1^{(n+1)}, z_2^{(n+1)}, \ldots, z_{m-1}^{(n+1)}, \mathcal{D}) \qquad \underbrace{[z_1^{(n+1)}, \ldots, z_m^{(n+1)}]}_{\mathbf{z}^{(n+1)}}$$

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)

$$\mathbf{z}_n = [z_1^{(n)}, z_2^{(n)}, \dots, z_m^{(n)}]^\top \longrightarrow \mathbf{z}' = [z_1^{(n+1)}, z_2^{(n)}, \dots, z_m^{(n)}]^\top$$

divide $p(z_2^{(n)}, \dots, z_m^{(n)}, \mathcal{D})$

Acceptance probability

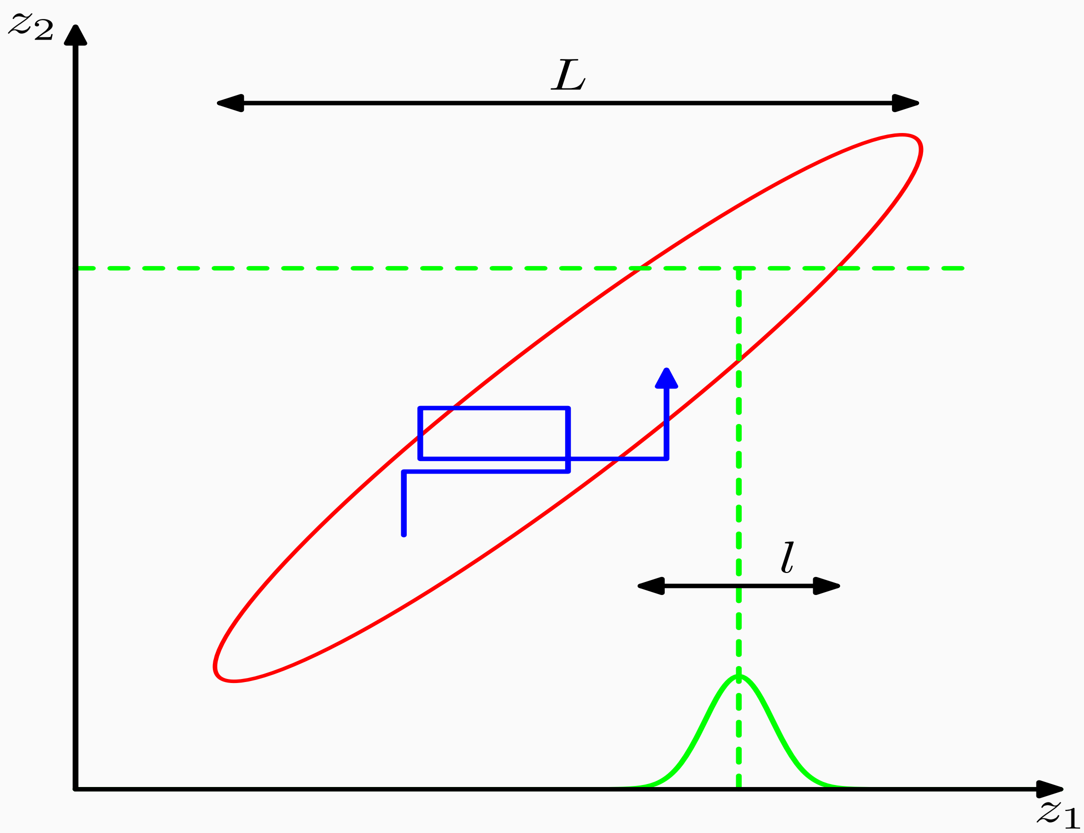
$$\min\left(1, \frac{p(z_1^{(n+1)}, z_2^{(n)}, \dots, z_m^{(n)}, \mathcal{D})p(z_1^{(n)}|z_2^{(n)}, \dots, z_m^{(n)}, \mathcal{D})}{p(z_1^{(n)}, z_2^{(n)}, \dots, z_m^{(n)}, \mathcal{D})p(z_1^{(n+1)}|z_2^{(n)}, \dots, z_m^{(n)}, \mathcal{D})}\right)$$

$$\|$$

$$\min\left(1, \frac{p(z_1^{(n+1)}|z_2^{(n)}, \dots, z_m^{(n)}, \mathcal{D})p(z_1^{(n)}|z_2^{(n)}, \dots, z_m^{(n)}, \mathcal{D})}{p(z_1^{(n)}|z_2^{(n)}, \dots, z_m^{(n)}, \mathcal{D})p(z_1^{(n+1)}|z_2^{(n)}, \dots, z_m^{(n)}, \mathcal{D})}\right) = 1$$

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

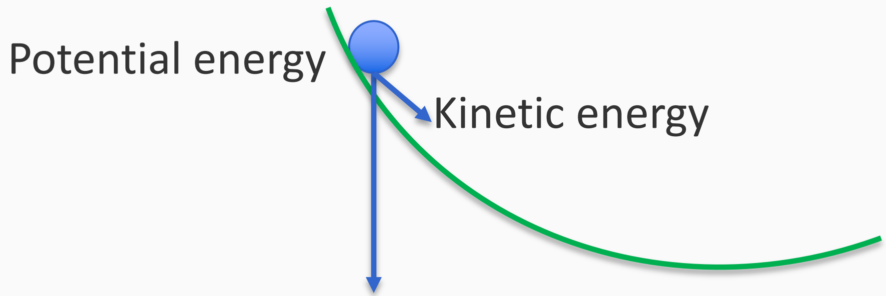
- Random walk behavior --- waste a lot of samples
- High correlation between different RVs --- slow exploration
- Can we address both problems?

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

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:

$$\frac{\mathrm{d}z_i}{\mathrm{d}t} = \frac{\partial H}{\partial r_i}$$

$$\frac{\mathrm{d}r_i}{\mathrm{d}t} = -\frac{\partial H}{\partial z_i}$$

$$\mathbf{z} = [z_1, \ldots, z_m]^\top$$

$$\mathbf{r} = [r_1, \ldots, r_m]^\top$$

Hamiltonian system

- How to map our probabilistic model into the system?

$$p(\mathbf{z}, \mathcal{D}) = p(z_1, \ldots, z_m, \mathcal{D})$$

- We take

$$U(\mathbf{z}) = -\log\big(p(\mathbf{z}, \mathcal{D})\big)$$

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

$$H(\mathbf{z}, \mathbf{r}) = U(\mathbf{z}) + K(\mathbf{r})$$ energy dist. → $$p(\mathbf{z}, \mathbf{r}) \propto \exp\big(-H(\mathbf{z}, \mathbf{r})\big)$$

What does it include?

Hamiltonian system

$$U(\mathbf{z}) = -\log\big(p(\mathbf{z}, \mathcal{D})\big)$$
$$K(\mathbf{r}) = \frac{1}{2}\mathbf{r}^\top \mathbf{M}^{-1}\mathbf{r}$$

$$H(\mathbf{z}, \mathbf{r}) = U(\mathbf{z}) + K(\mathbf{r})$$

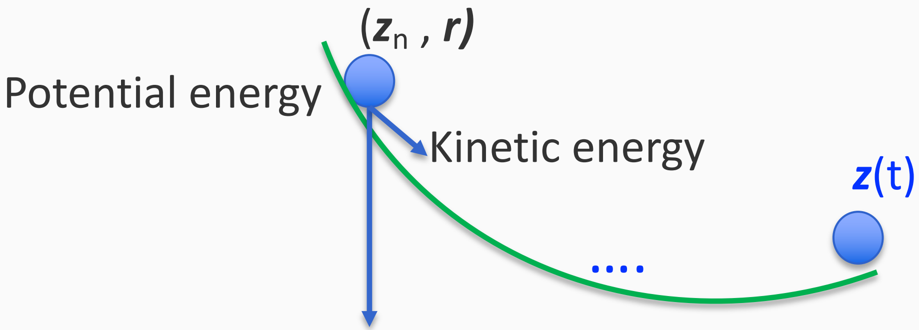
$$\frac{\mathrm{d}z_i}{\mathrm{d}t} = \frac{\partial H}{\partial r_i}$$
$$\frac{\mathrm{d}r_i}{\mathrm{d}t} = -\frac{\partial H}{\partial z_i}$$

$\Rightarrow$

$$\frac{\mathrm{d}z_i}{\mathrm{d}t} = [\mathbf{M}^{-1}\mathbf{r}]_i$$
$$\frac{\mathrm{d}r_i}{\mathrm{d}t} = -\frac{\partial U}{\partial z_i}$$

Hamiltonian System

- The key idea: use the current sample $\boldsymbol{z}_n$ and random sample of $\boldsymbol{r}$, as the initial state of the Hamiltonian system; and then evolve the system to a time t, pick the $\boldsymbol{z}$(t) as the proposal and test whether to accept it as $\boldsymbol{z}_{n+1}$



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

Hamiltonian system

- Nice properties to guarantee the detailed balance

1. Reversibility:

one-to-one

($\boldsymbol{z}$(t), $\boldsymbol{r}$(t)) → ($\boldsymbol{z}$(t+s), $\boldsymbol{r}$(t+s))

($\boldsymbol{z}$(t), -$\boldsymbol{r}$(t)) ← ($\boldsymbol{z}$(t+s), -$\boldsymbol{r}$(t+s))

Negate momentum

Rigorously speaking, we need to first evolve the system, and then negate the momentum to obtain the new proposal

Why is it important?

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

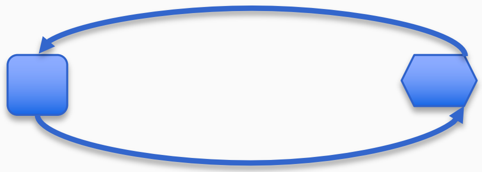
Now T is a delta function, we need to be able to jump back!

Numerical Integration

- Nice properties to guarantee the detailed balance

2. Conservation: $\frac{\mathrm{d}H}{\mathrm{d}t} = 0$ Totally energy does not change

3. Volume preservation: Determinant of Jacobian is always 1



For any t,s≥0: $|\det \frac{\partial[\mathbf{u}(t+s), \mathbf{r}(t+s)]^\top}{\partial[\mathbf{u}(t), \mathbf{r}(t)]^\top}| = 1$

Volume does not change after transformation

General theorem (proof omitted)

Consider an arbitrary dynamic system Ψ_t

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

If the following conditions are satisfied:

- Ψ_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.

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

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

General theorem (proof omitted)

Consider an arbitrary dynamic system Ψ_t

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

If the following conditions are satisfied:

- Ψ_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.

conservation

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

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

Energy dist.

Apply the theorem to Hamiltonian system, the accept rate is always 1

However, we cannot exactly evolve Hamiltonian system (do not know solution)

$$U(\mathbf{z}) = -\log\big(p(\mathbf{z}, \mathcal{D})\big)$$
$$K(\mathbf{r}) = \frac{1}{2}\mathbf{r}^\top \mathbf{M}^{-1}\mathbf{r}$$

$$H(\mathbf{z}, \mathbf{r}) = U(\mathbf{z}) + K(\mathbf{r})$$

$$\frac{\mathrm{d}z_i}{\mathrm{d}t} = \frac{\partial H}{\partial r_i}$$
$$\frac{\mathrm{d}r_i}{\mathrm{d}t} = -\frac{\partial H}{\partial z_i}$$

$\Rightarrow$

$$\frac{\mathrm{d}z_i}{\mathrm{d}t} = [\mathbf{M}^{-1}\mathbf{r}]_i$$
$$\frac{\mathrm{d}r_i}{\mathrm{d}t} = -\frac{\partial U}{\partial z_i}$$

Numerical integration

$$\frac{\mathrm{d}z_i}{\mathrm{d}t} = [\mathbf{M}^{-1}\mathbf{r}]_i$$
$$\frac{\mathrm{d}r_i}{\mathrm{d}t} = -\frac{\partial U}{\partial z_i}$$

In practice we often choose

$$\mathbf{M} = \mathrm{diag}[s_1, \ldots, s_m]$$

Euler's method: choose step size ϵ , and # of step size *L*

$$r_i(t+\epsilon) = r_i(t) + \epsilon \frac{\mathrm{d}r_i(t)}{\mathrm{d}t} = r_i(t) - \epsilon \frac{\partial U(\mathbf{z}(t))}{\partial z_i}$$
$$z_i(t+\epsilon) = z_i(t) + \epsilon \frac{\mathrm{d}z_i(t)}{\mathrm{d}t} = z_i(t) + \epsilon \frac{r_i(t)}{s_i}$$

Log joint probability

Leapfrog method

- Euler's method is a first-order method O(ϵ)
- In practice, people choose Leapfrog method, a second-order method O(ϵ^2)

$$r_i(t+\epsilon/2) = r_i(t) - (\epsilon/2)\frac{\partial U(\mathbf{z})}{\partial z_i}$$

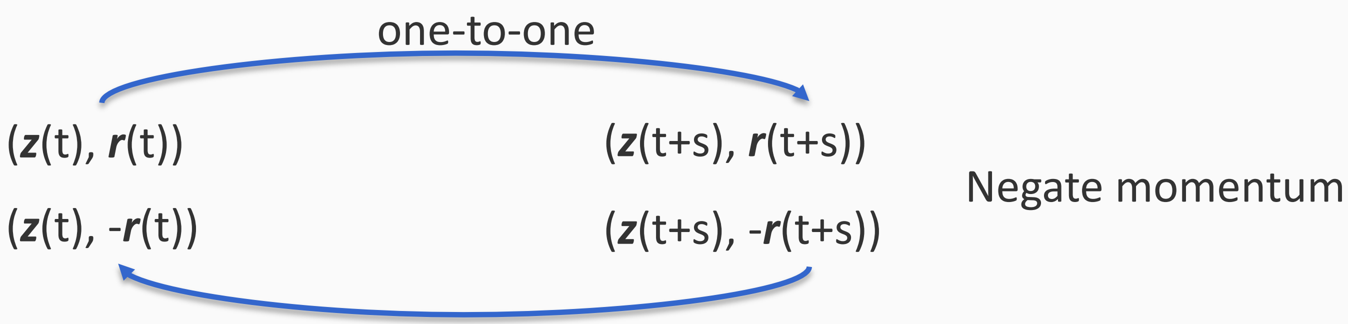
$$z_i(t+\epsilon) = z_i(t) + \epsilon\frac{r_i(t+\epsilon/2)}{s_i}$$

introduce half-step

$$r_i(t+\epsilon) = r_i(t+\epsilon/2) - (\epsilon/2)\frac{\partial U\big(\mathbf{z}(t+\epsilon)\big)}{\partial z_i}$$

Leapfrog method (ϵ, L)

- Key properties
  - Reversibility under momentum negation

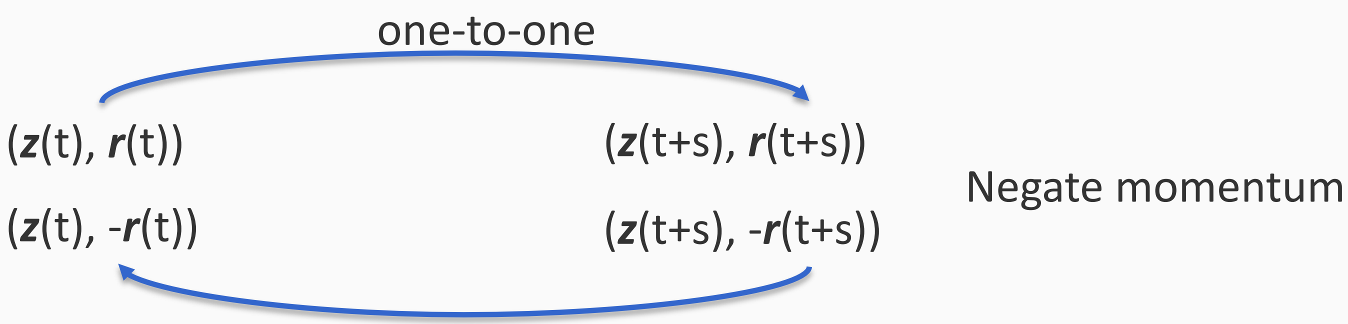


  - Volume preservation: each leap-frog step is a shear transformation and preserves volumes

Question: does conservation still hold?

Leapfrog method (ϵ, 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 Ψ_t ← Leapfrog

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

If the following conditions are satisfied:

- Ψ_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: momentum negation
- The proposed sample $R(\mathbf{v}')$ is accepted with prob.

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

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

$$U(\mathbf{z}) = -\log\big(p(\mathbf{z}, \mathcal{D})\big) \qquad K(\mathbf{r}) = \frac{1}{2}\mathbf{r}^\top \mathbf{M}^{-1}\mathbf{r}$$

$$H(\mathbf{z}, \mathbf{r}) = U(\mathbf{z}) + K(\mathbf{r})$$

$$p(\mathbf{z}, \mathbf{r}) \propto \exp\big(-H(\mathbf{z}, \mathbf{r})\big)$$

- 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 ϵ , obtain (***z***', ***r***'), set ***r***' = -***r***', accept z' with probability

$$\min\{1, \exp\big(-H(\mathbf{z}', \mathbf{r}') + H(\mathbf{z}, \mathbf{r})\big)\} = \min\{1, \exp\big(-U(\mathbf{z}') - K(\mathbf{r}') + U(\mathbf{z}) + K(\mathbf{r})\big)\}$$

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 ϵ 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
- Gold-standard for inference in Bayesian neural networks.

HMC discussion

- There is a trade-off for the choice (ϵ, L) in the Leapfrog

$$\min\{1, \exp\big(-H(\mathbf{z}', \mathbf{r}') + H(\mathbf{z}, \mathbf{r})\big)\}$$

- Large ϵ and L will allow you to explore the space further away, but increase the numerical error and lower the acceptance rate
- Small ϵ 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!