CS 8501 Advanced Topics in Machine Learning

Lecture 10: Monte Carlo Methods (II)

Yangfeng Ji

Information and Language Processing Lab

Department of Computer Science

University of Virginia

https://yangfengji.net/

Markov Chains

Reference

- Andrieu et al. An Introduction to MCMC for Machine Learning. 2003
- Holmes-Cerfon. Lecture 3 Markov Chains (II). Spring 2019

Markov Chain Monte Carlo

- Monte Carlo methods
  - Use samples to approximate a probability distribution
- Markov Chain Monte Carlo is a strategy for generating samples $\{x_t\}$ while exploring the sampling space using a Markov chain mechanism.
- The mechanism is constructed so that
  - the chain spends more time in the most important regions.
  - the samples mimic samples drawn from the target distribution $p(x)$

Markov Chain

A Markov chain defined on a finite state space $\mathcal{S} = \{s_1, \ldots, s_K\}$ is described as

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

where $T(x_t|x_{t-1})$ is a $K \times K$ matrix

- $p(x_t = s_k|x_{t-1} = s_{k'})$ describe the transition probability from $x_{t-1} = s_{k'}$ to $x_t = s_k$
- $\sum_{a_k} p(x_t = a_k|x_{t-1}) = 1$

Homogeneous Markov Chain

A Markov chain is homogeneous if

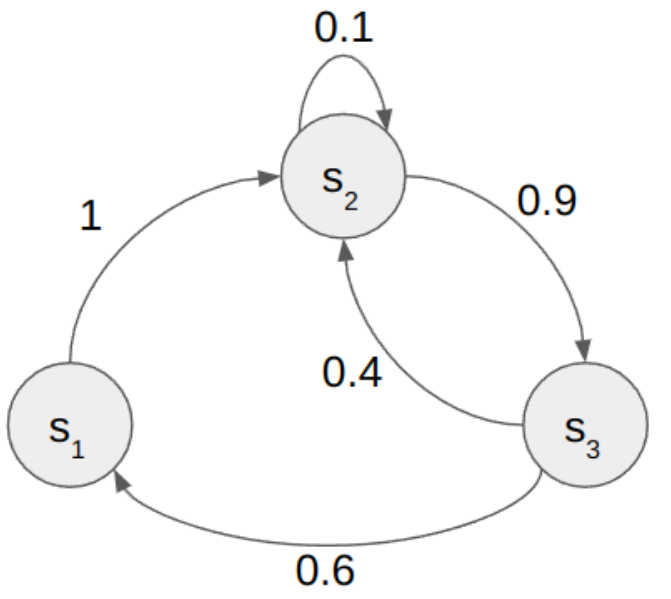
$$T = T(x_t | x_{t-1})$$

remains invariant for all t.

In this case, the evolution of the chain depends solely on

- current state of the chain, and
- a fixed transition matrix

Example



Transition matrix: row x_{t-1}; column x_t

$$T = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0.1 & 0.9 \\ 0.6 & 0.4 & 0 \end{bmatrix}$$

Example (II)

Consider the following initial state probability $p(x_1)$, $\pi = [1.0, 0, 0]$, then

- $p(x_2)$

$$p(x_2) = \sum_{x_1} p(x_1)p(x_2|x_1) = \pi \cdot T$$

- $p(x_{100})$

$$p(x_{100}) = \pi \cdot T^{100-1} = [0.2213, 0.4098, 0.3688]$$

Example (III)

With another initial state probability $\pi' = [0, 1.0, 0]$, we still have

$$p(x_{100}) = \pi' \cdot T^{100-1} = [0.2213, 0.4098, 0.3688]$$

- In fact, this is true for any initial probability
- And

  $$p(x) = [0.2213, 0.4098, 0.3688]$$

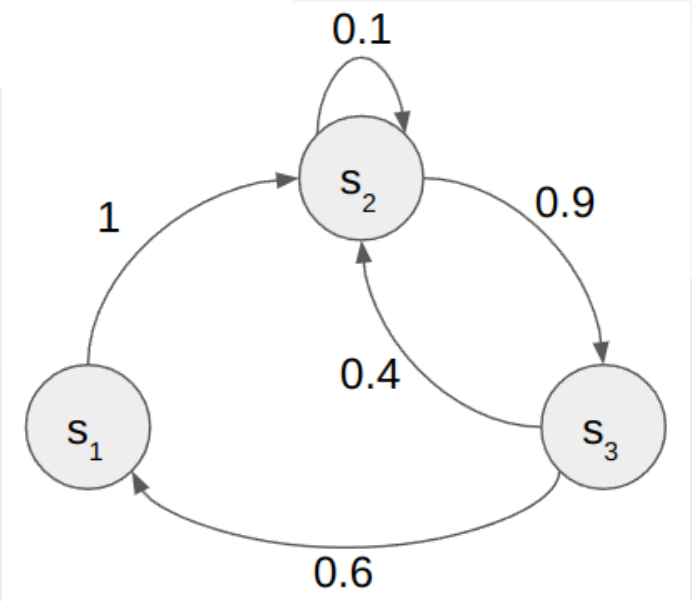
  is the stationary distribution of this Markov chain
- Mathematically, the stationary distribution π satisfies

  $$\pi P = \pi$$

Markov Chain (Cont.)

A Markov chain has a stationary distribution, as long as T obeys

- *Irreducibility*: from any state, there is a positive probability of visiting all other states
- *Aperiodicity*: the state transition should not be trapped in cycles



The Detailed Balance Condition

A sufficient (but not necessary) condition to ensure $p(x)$ to be an stationary distribution is the following detailed balance condition

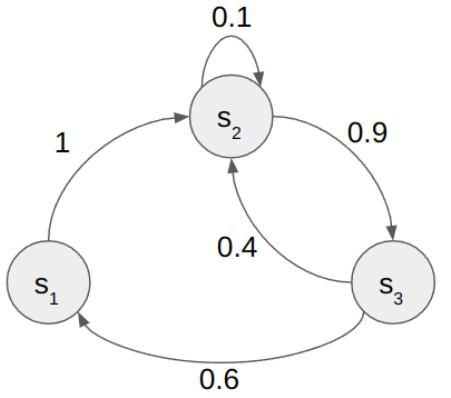
$$p(x_{t-1})T(x_t|x_{t-1}) = p(x_t)T(x_{t-1}|x_t)$$

- This is the key of MCMC
- Comparing to the definition of stationary distribution, this is defined on each edge of a transition graph
- An example of detailed balance graph

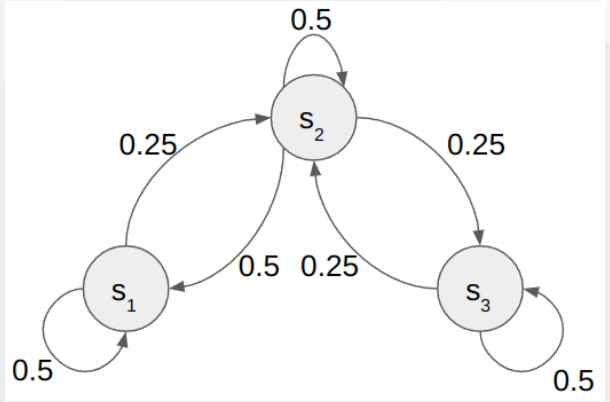
$$s_1 \rightleftharpoons s_2 \rightleftharpoons s_3$$

Examples

The previous example does not satisfy the detailed balance condition



A simple example that satisfies the detailed balance condition



Continuous State Space

In continuous state spaces, the transition matrix T becomes an integral kernel K,

$$p(x_t) = \int p(x_{t-1})K(x_t, x_{t-1})dx_{t-1}$$

Consider $K(x_t, x_{t-1})$ as a conditional probability $p(x_t|x_{t-1})$ would be easier

Metropolis-Hastings Algorithm

Metropolis-Hastings Algorithm

1. Initialize x_0
2. For $t = 0$ to T
    - Sample $u \sim U[0, 1]$
    - Sample $\tilde{x} \sim q(\tilde{x}|x_0)$ *// proposal distribution*
    - if $u < \min\{1, \frac{p(\tilde{x})q(x_t|\tilde{x})}{p(x_t)q(\tilde{x}|x_t)}\}$

$$x_{t+1} \leftarrow \tilde{x}$$

else

$$x_{t+1} \leftarrow x_t$$

Proposal Distribution

In the MH algorithm, the proposal distribution is defined as

$$q(\tilde{x}|x_t)$$

where x_t is the sample from the current time step

- $q(\tilde{x}|x_t)$ is the transition matrix/kernel function of the Markov chain
- There is a dependence between x_t and $\tilde{x}$

Acceptance Probability

The acceptance probability is defined as

$$A(x_t, \tilde{x}) = \min\{1, \frac{p(\tilde{x})q(x_t|\tilde{x})}{p(x_t)q(\tilde{x}|x_t)}\}$$

To understand the acceptance probability, let's consider a simple proposal function

$$q(\tilde{x}|x_t) = \mathcal{N}(\tilde{x}; x_t, I) \propto \exp(-\|\tilde{x} - x_t\|_2^2)$$

Symmetric Proposal Distribution

When the proposal distribution is symmetric, the acceptance probability is reduced as

$$A(x_t, \tilde{x}) = \min\{1, \frac{p(\tilde{x})}{p(x_t)}\}$$

Therefore, the algorithm will

- always accept a sample $\tilde{x}$, when $p(\tilde{x}) \geq p(x_t)$
- accept a sample $\tilde{x}$ by chance, when $p(\tilde{x}) < p(x_t)$

Asymmetric Proposal Distribution

For asymmetric proposal distribution, we have

$$A(x_t, \tilde{x}) = \min\{1, \frac{p(\tilde{x})/q(\tilde{x}|x_t)}{p(x_t)/q(x_t|\tilde{x})}\}$$

This will compensate the bias/preference in the transition matrix/kernel function

Why MH Works

The transition matrix of the MH algorithm is

$$p(\tilde{x}|x_t) = \begin{cases} q(\tilde{x}|x_t)A(\tilde{x}, x_t) & \text{if } \tilde{x} \neq x_t \\ q(x_t|x_t) + \sum_{\tilde{x} \neq x_t} q(\tilde{x}|x_t)(1 - A(\tilde{x}, x_t)) & \text{otherwise} \end{cases}$$

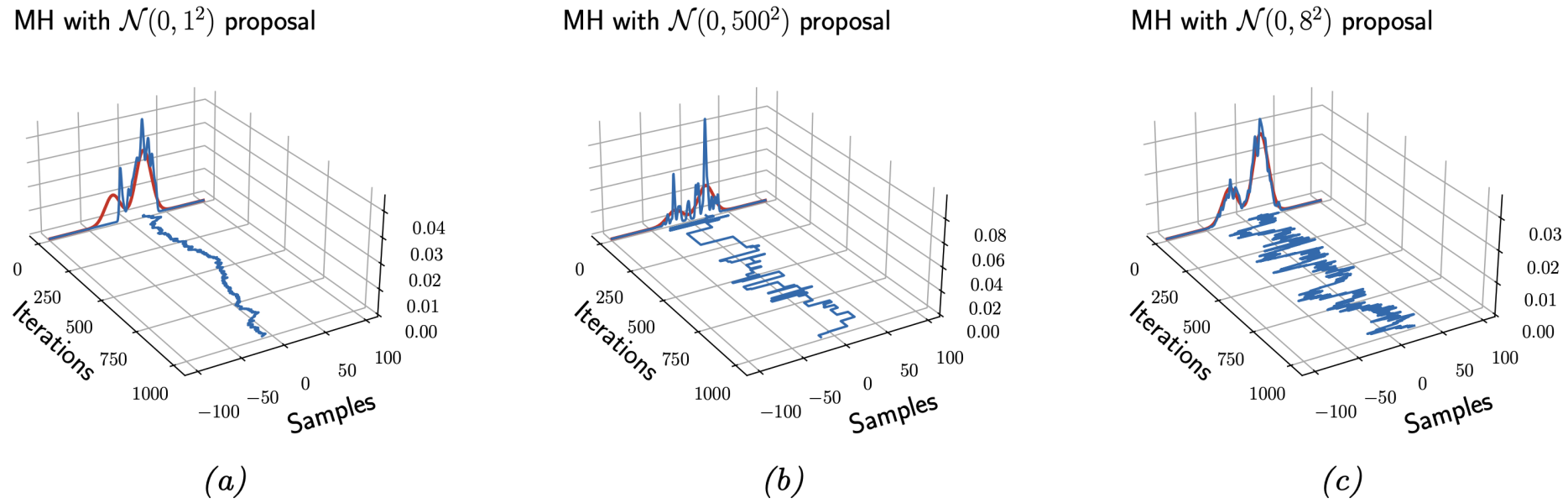
[Marphy 2023, sec 12.2.2] gives an excellent explanation of this formula

- $p(\tilde{x}|x_t)$ defines a Markov chain that satisfies the detailed balance condition
- $p(x)$ is its stationary distribution

Proposal Distribution: RWM algorithm

The random-walk Metropolis algorithm is the MH algorithm with the proposal distribution

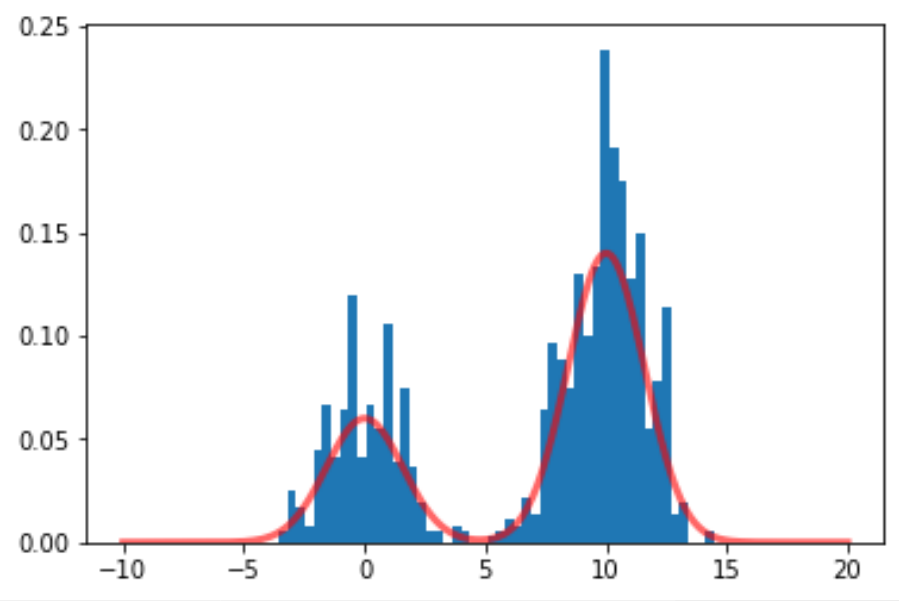
$$q(\tilde{x}|x_t) = \mathcal{N}(\tilde{x}; x_t, \tau^2 I)$$

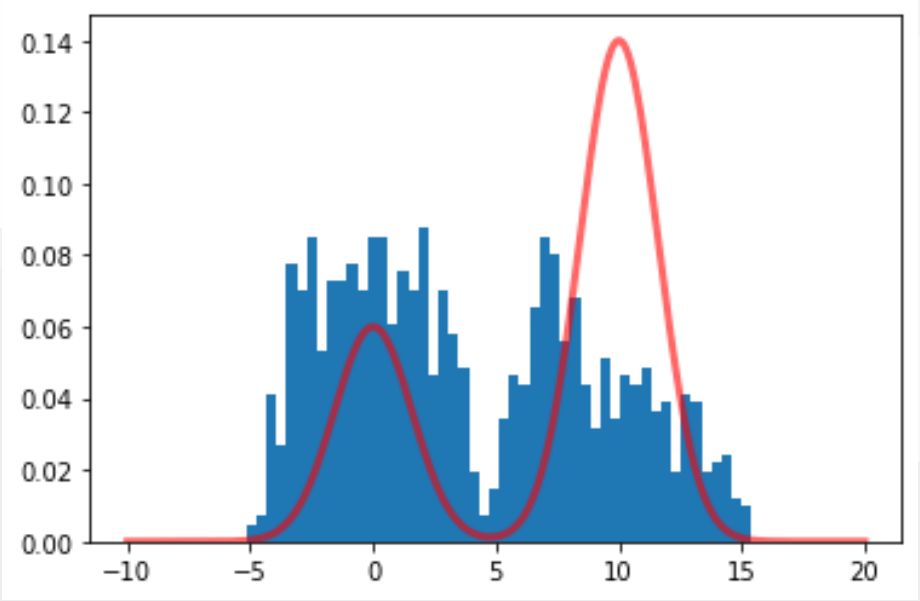


Comparison: MH vs. Accept-Reject

Proposal distributions

- For MH: $q(\tilde{x}|x) = \mathcal{N}(\tilde{x}; x, 10^2)$
- For accept-reject: $q(\tilde{x}) = \mathcal{N}(\tilde{x}, 0, 10^2)$

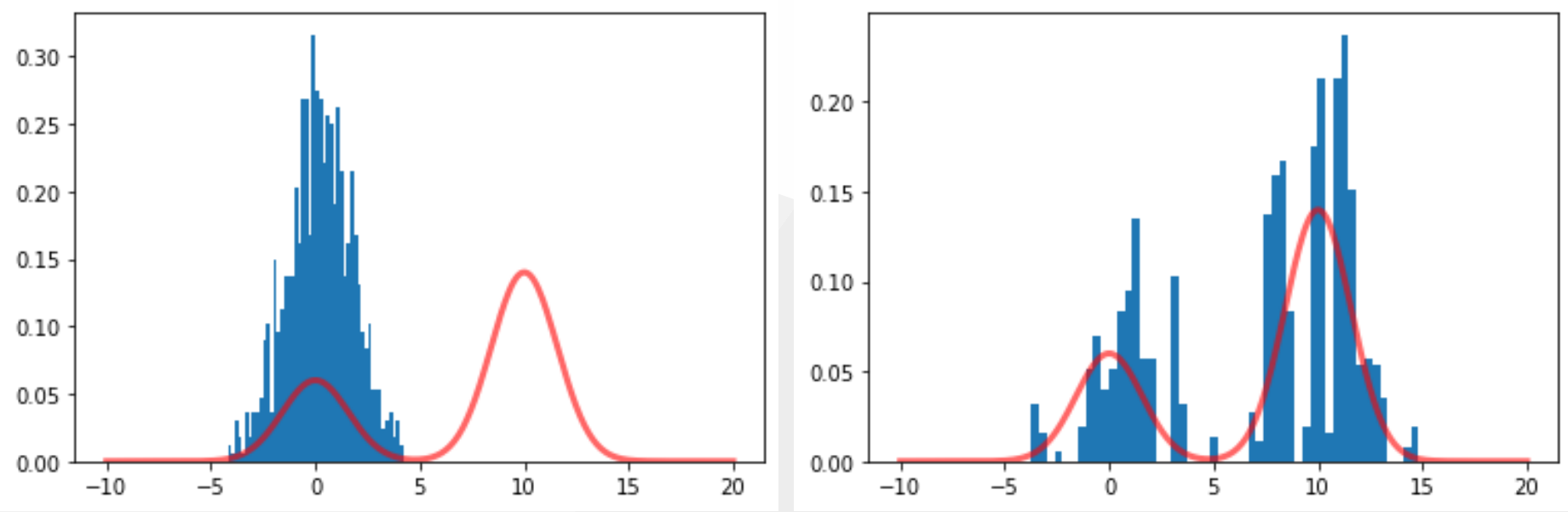




Comparison: MH with Different Proposals

Proposal distributions:

- Left: $q(\tilde{x}|x) = \mathcal{N}(\tilde{x}; x, 1^2)$
- Right: $q(\tilde{x}|x) = \mathcal{N}(\tilde{x}; x, 50^2)$



Gibbs Sampling

- Using $x^{(t)}$ to represent the sample at time step t

Gibbs Sampling

Consider a three-dimensional distribution $p(x_1, x_2, x_3)$, the sampling procedure for time step $t+1$

- $x_1^{(t+1)} \sim p(x_1 | x_2^{(t)}, x_3^{(t)})$
- $x_2^{(t+1)} \sim p(x_2 | x_1^{(t+1)}, x_3^{(t)})$
- $x_3^{(t+1)} \sim p(x_3 | x_1^{(t+1)}, x_2^{(t+1)})$

Return $(x_1^{(t+1)}, x_2^{(t+1)}, x_3^{(t+1)})$ as the sample at time step $t+1$

Sampling from Conditional Distributions

For example

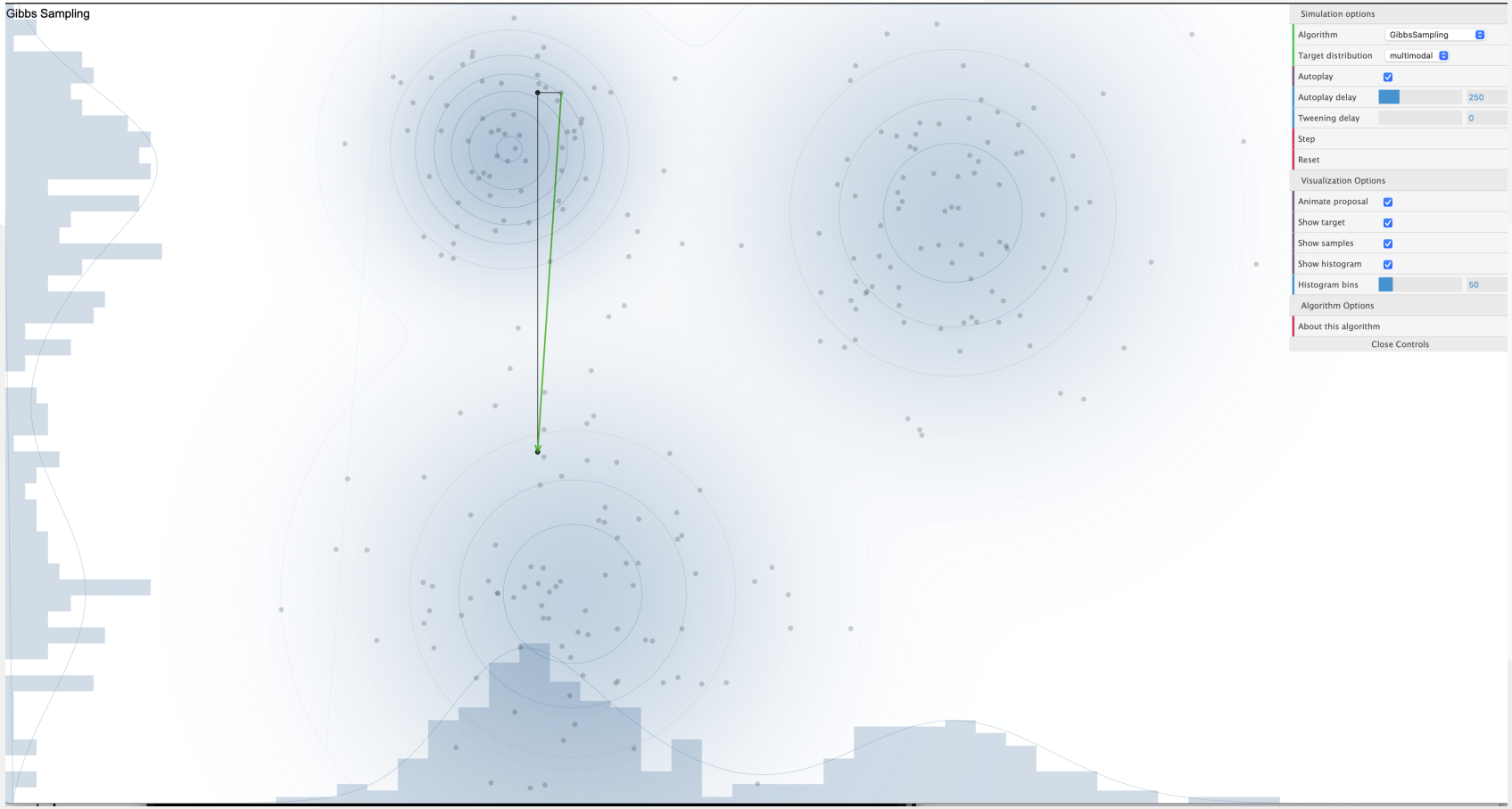
$$x_1^{(t+1)} \sim p(x_1 | x_2^{(t)}, x_3^{(t)})$$

With the sampling algorithm discussed in the last lecture, we can actually sample from the unnormalized distribution, by fixing x_2 and x_3 in this case

$$x_1^{(t+1)} \sim p(x_1, x_2^{(t)}, x_3^{(t)}) \propto p(x_1 | x_2^{(t)}, x_3^{(t)})$$

Demo

A demo with a multimodal distribution



Gibbs Sampling as a Special Case of MH

- The proposal distribution as shown in the previous page

$$q_i(\tilde{x}|x) = p(\tilde{x}_i|x_{-i})\mathbb{I}(\tilde{x}_{-i} = x_{-i})$$

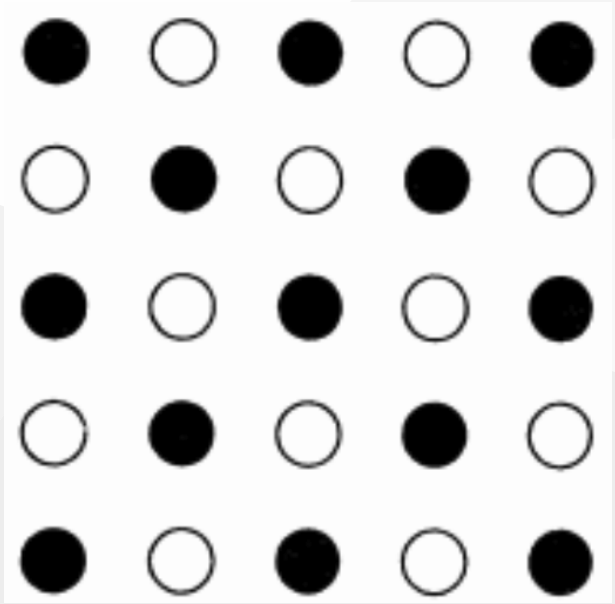
- The acceptance rate is 100%

$$\alpha = \frac{p(\boldsymbol{x}')q_i(\boldsymbol{x}|\boldsymbol{x}')}{p(\boldsymbol{x})q_i(\boldsymbol{x}'|\boldsymbol{x})} = \frac{p(x_i'|\boldsymbol{x}'_{-i})p(\boldsymbol{x}'_{-i})p(x_i|\boldsymbol{x}'_{-i})}{p(x_i|\boldsymbol{x}_{-i})p(\boldsymbol{x}_{-i})p(x_i'|\boldsymbol{x}_{-i})}$$
$$= \frac{p(x_i'|\boldsymbol{x}_{-i})p(\boldsymbol{x}_{-i})p(x_i|\boldsymbol{x}_{-i})}{p(x_i|\boldsymbol{x}_{-i})p(\boldsymbol{x}_{-i})p(x_i'|\boldsymbol{x}_{-i})} = 1$$

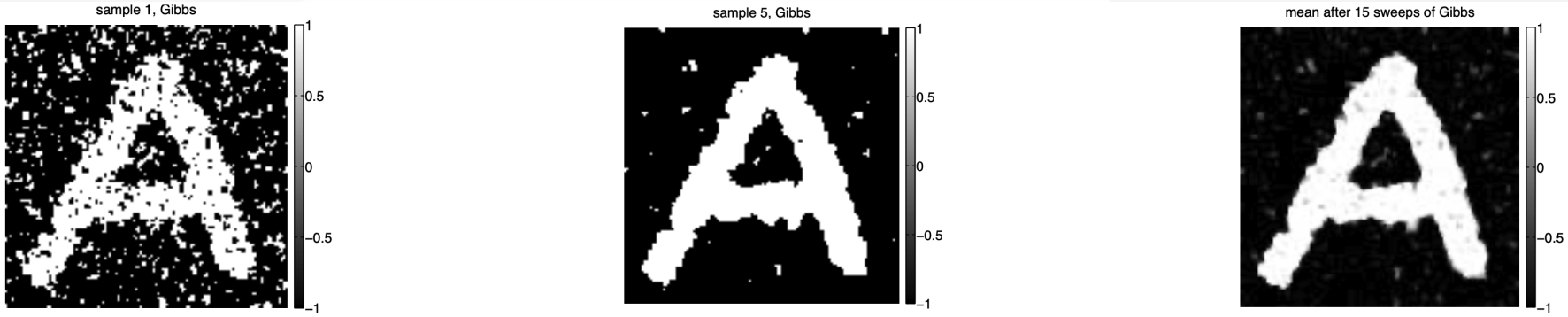
Gibbs Sampling on Ising Models

$$p(x) \propto \prod_{(i,j)\in E} \psi_{ij}(x_i, x_j; \theta)$$

Explore the conditional independence for parallel sampling



Example



(a) *(b)* *(c)*

Figure 12.3: Example of image denoising using Gibbs sampling. We use an Ising prior with $J = 1$ and a Gaussian noise model with $\sigma = 2$. (a) Sample from the posterior after one sweep over the image. (b) Sample after 5 sweeps. (c) Posterior mean, computed by averaging over 15 sweeps. Compare to Figure 10.3 which shows the results of mean field inference. Generated by ising_image_denoise_demo.ipynb.

Thank You!