Machine Learning Department
School of Computer Science
Carnegie Mellon University

# Markov Chain Monte Carlo

Matty O'Gormley
Lecture 13
Mar., 17 2021

# Reminders

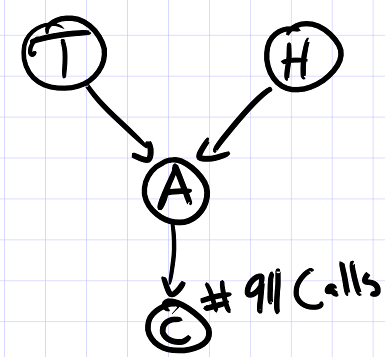
- **Project Team Formation**
  - **Due: Mon, Mar. 22 at 11:59pm**
- **Homework 3: Structured SVM**
  - **Out: Wed, Mar. 10**
  - **Due: Wed, Mar. 24 at 11:59pm**

Metropolis, Metropolis-Hastings, Gibbs Sampling

# MCMC (BASIC METHODS)

# Sampling from a Joint Distribution

Ex: Tornado




$T \sim \text{Bernoulli}(\tau)$ $\quad \tau = 1/2$

$H \sim \text{Bernoulli}(\eta)$ $\quad \eta = 1/3$

$A \sim \text{Bernoulli}(\alpha_{H,T})$ $\quad \alpha =$

| | T=0 | T=1 |
|---|---|---|
| H=0 | 0 | 1/2 |
| H=1 | 1/2 | 1 |

$C \sim \text{Unif}(\{1, \ldots, 63\}) + A * \text{Unif}(\{1, \ldots, 63\})$

integer

We can use these samples to estimate many different probabilities!

| T | H | A | C |
|---|---|---|---|
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

# A Few Problems for a Factor Graph

Suppose we already have the parameters of a Factor Graph…

1. How do we compute the probability of a specific assignment to the variables?
   $P(T=t, H=h, A=a, C=c)$

2. How do we draw a sample from the joint distribution?
   $t,h,a,c \sim P(T, H, A, C)$

3. How do we compute marginal probabilities?
   $P(A) = \ldots$

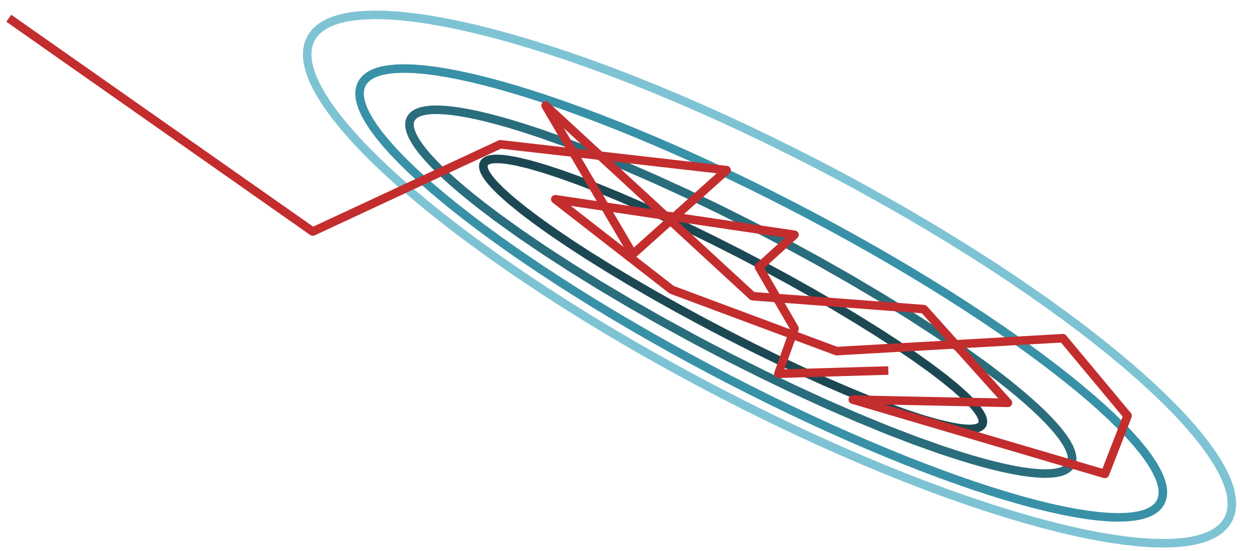
4. How do we draw samples from a conditional distribution?
   $t,h,a \sim P(T, H, A \mid C = c)$

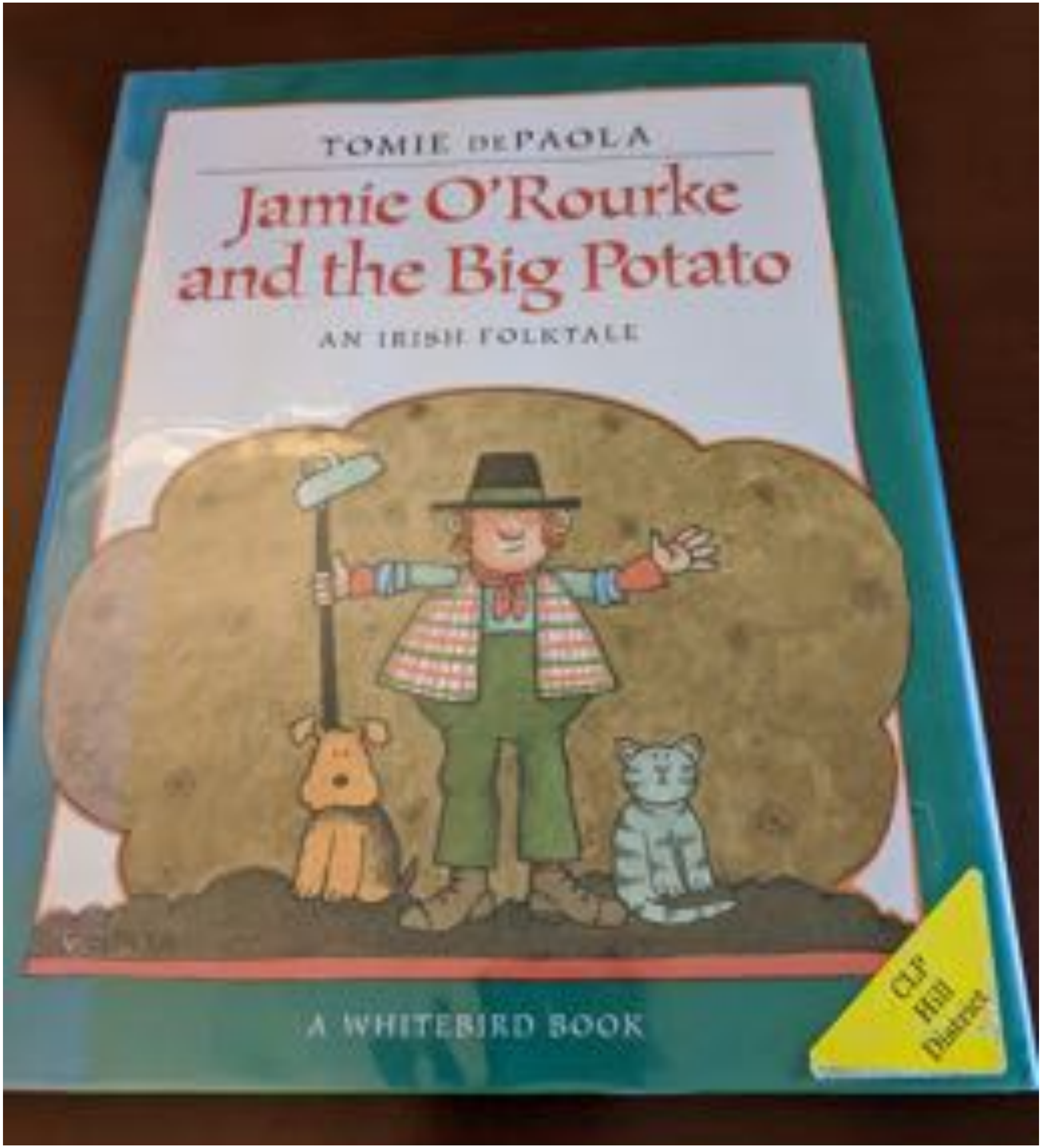
5. How do we compute conditional marginal probabilities?
   $P(H \mid C = c) = \ldots$

Can we use samples?

# MCMC

- **Goal:** Draw approximate, correlated samples from a target distribution p(x)
- **MCMC:** Performs a biased random walk to explore the distribution




TOMIE dePAOLA
Jamie O'Rourke and the Big Potato
AN IRISH FOLKTALE
A WHITEBIRD BOOK
CLF Hill District


https://commons.wikimedia.org/wiki/File:Map_of_ireland.jpg

# Simulations of MCMC

Visualization of Metroplis-Hastings, Gibbs Sampling, and Hamiltonian MCMC:

https://chi-feng.github.io/mcmc-demo/

http://twiecki.github.io/blog/2014/01/02/visualizing-mcmc/

# GIBBS SAMPLING

# Gibbs Sampling

***Whiteboard***

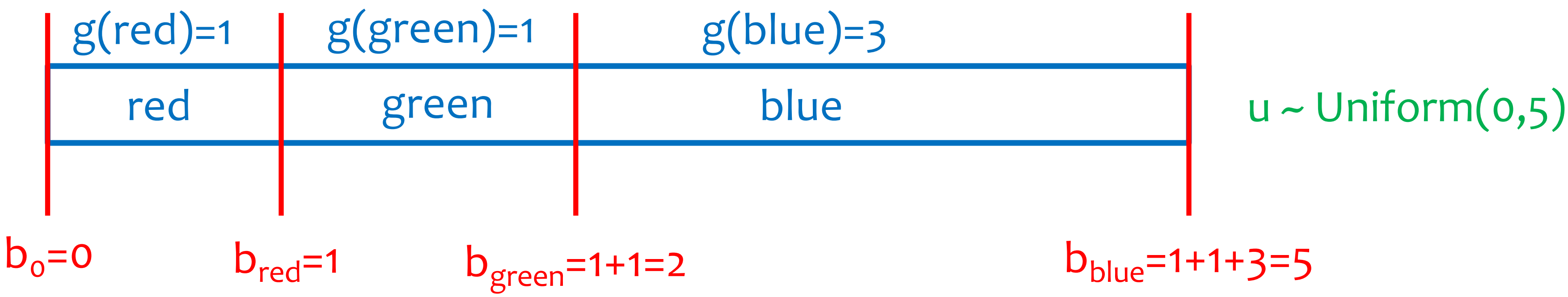
- Gibbs Sampling

# Sampling from a Discrete Distribution

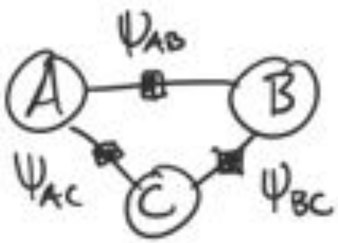
- To sample from a discrete distribution p(y) we only need a function proportional to it
e.g., g(·) s.t. p(y) ∝ g(y)
- **Recipe:**
  - Define a bin cutoff $b_y$ for each value $y \in \{1, \ldots, V\}$

$$b_y = \sum_{t=1}^{y} g(t),\ \forall y \in \{1, \ldots, V\} \qquad b_0 = 0$$

  - Sample u ~ Uniform(0, $b_V$)
  - Return value y if u lands in bin $[b_{y-1}, b_y,]$



# Example: Gibbs Sampling




$A, B, C \in \{+, -\}$

| a | b | $\psi_{AB}(a,b)$ |
|---|---|---|
| + | + | 1 |
| + | − | 2 |
| − | + | 1 |
| − | − | 1 |

| a | c | $\psi_{AC}$ |
|---|---|---|
| + | + | 2 |
| + | − | 2 |
| − | + | 2 |
| − | − | 1 |

| b | c | $\psi_{BC}$ |
|---|---|---|
| + | + | 1 |
| + | − | 1 |
| − | + | 2 |
| − | − | 1 |

full conditionals:

① $p(a \mid b, c) \propto \psi(a,b)\psi(a,c) \rightarrow g(a) = [\ + \mid - \ ]$

② $p(b \mid a, c) \propto \psi(a,b)\psi(b,c) \rightarrow g(b) = [\ + \mid - \ ]$

③ $p(c \mid a, b) \propto \psi(a,c)\psi(b,c) \rightarrow g(c) = [\ + \mid - \ ]$

fixed while sampling

might change at each iteration.

Algo: Initialize $a, b, c$ randomly $\in \{+, -\}$

For $i = 1, 2, 3, \dots$

$a \sim p(a \mid b, c)$

$b \sim p(b \mid a, c)$

$c \sim p(c \mid a, b)$

# table entries: 2 or 8

$$p(a \mid b, c) = \frac{p(a,b,c)}{p(b,c)} \propto p(a,b,c)$$

# table entries: 8

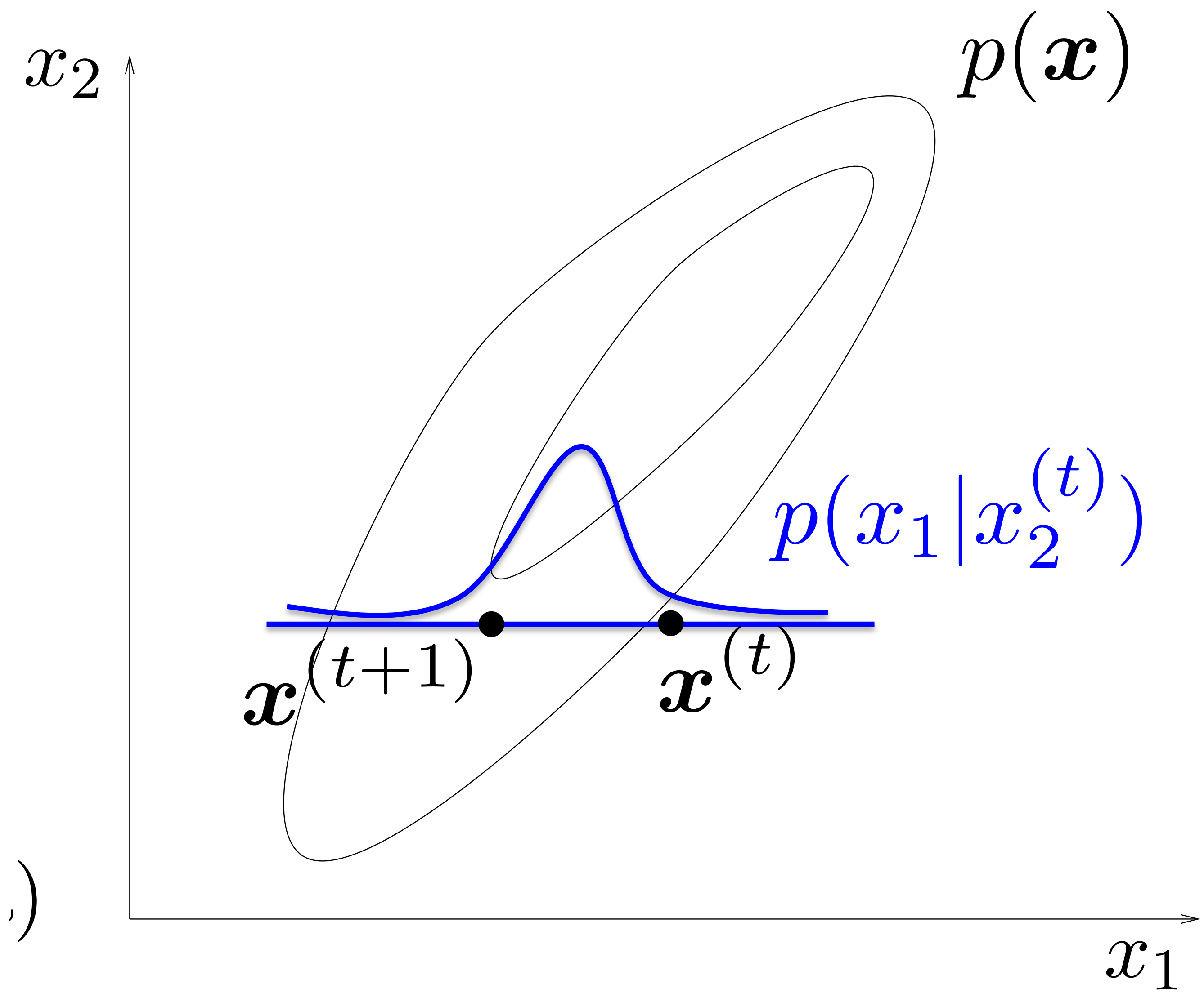
$$p(a,b,c) \triangleq \frac{1}{Z}\psi(a,b)\psi(a,c)\psi(b,c)$$

# Example: Gibbs Sampling

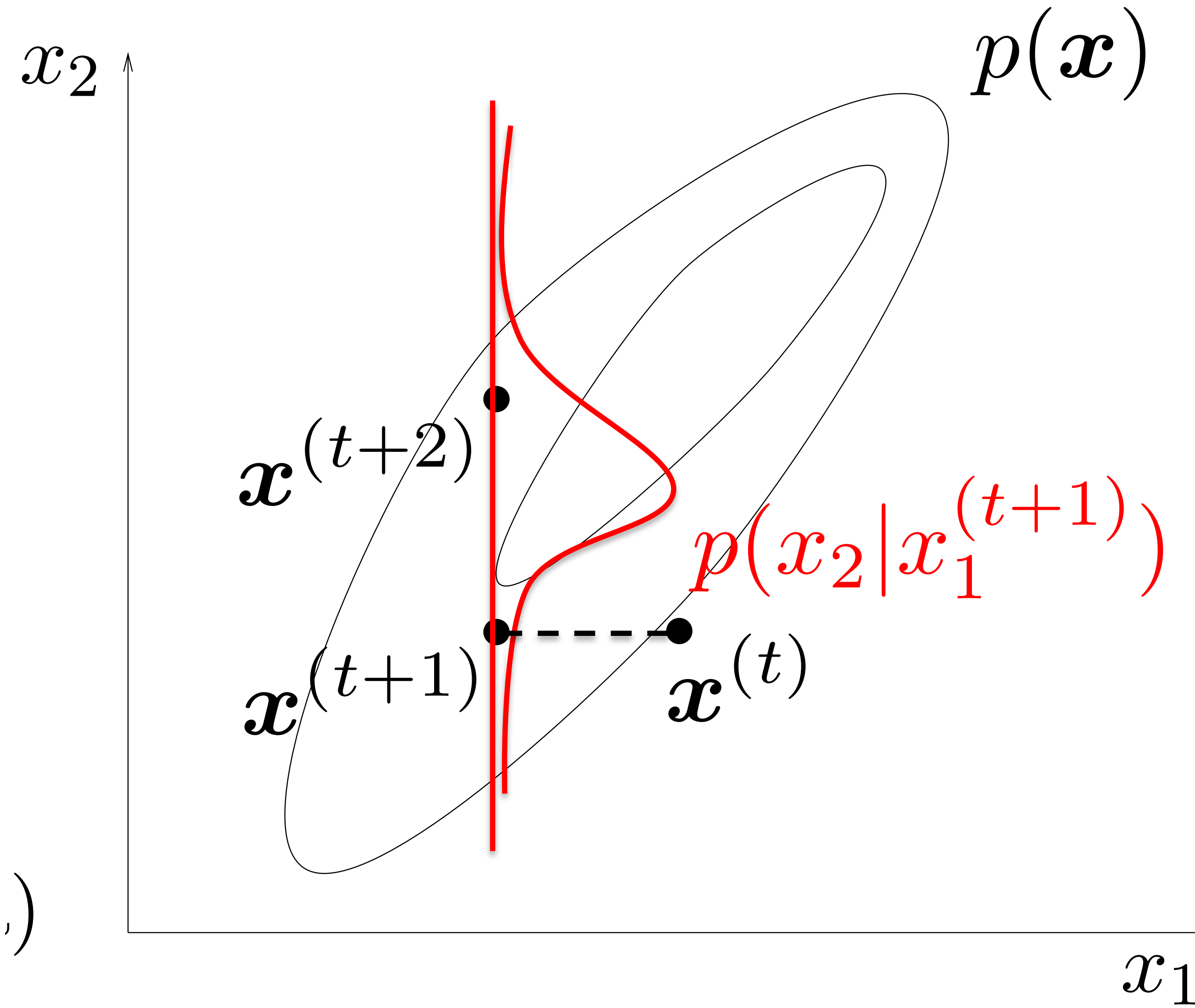
Example: 3-node Factor Graph

```python
import numpy as np
import random

def sample01(g0, g1):
    u = random.uniform(0, g0 + g1)
    if u < g0:
        return 0
    else:
        return 1

def gibbs_sampling():
    # Define factor graph
    psi_ab = np.array([[1, 2], [1, 1]])
    psi_ac = np.array([[2, 2], [2, 1]])
    psi_bc = np.array([[1, 1], [2, 1]])

    # Initialize variable values
    a = random.choice([0,1])
    b = random.choice([0,1])
    c = random.choice([0,1])

    counts = np.array([[0, 0], [0, 0], [0, 0]])
    # Gibbs sampling
    for i in range(10):
        a = sample01(psi_ab[0,b] * psi_ac[0,c],
                     psi_ab[1,b] * psi_ac[1,c])
        b = sample01(psi_ab[a,0] * psi_bc[0,c],
                     psi_ab[a,1] * psi_bc[1,c])
        c = sample01(psi_ac[a,0] * psi_bc[b,0],
                     psi_ac[a,1] * psi_bc[b,1])
        print(a, b, c)
        counts[0, a] += 1
        counts[1, b] += 1
        counts[2, c] += 1

    print('p(a = 0) ~= %.2f' % (counts[0,0] / (counts[0,0] + counts[0,1])))
    print('p(b = 0) ~= %.2f' % (counts[1,0] / (counts[1,0] + counts[1,1])))
    print('p(c = 0) ~= %.2f' % (counts[2,0] / (counts[2,0] + counts[2,1])))

if __name__ == '__main__':
    gibbs_sampling()
```

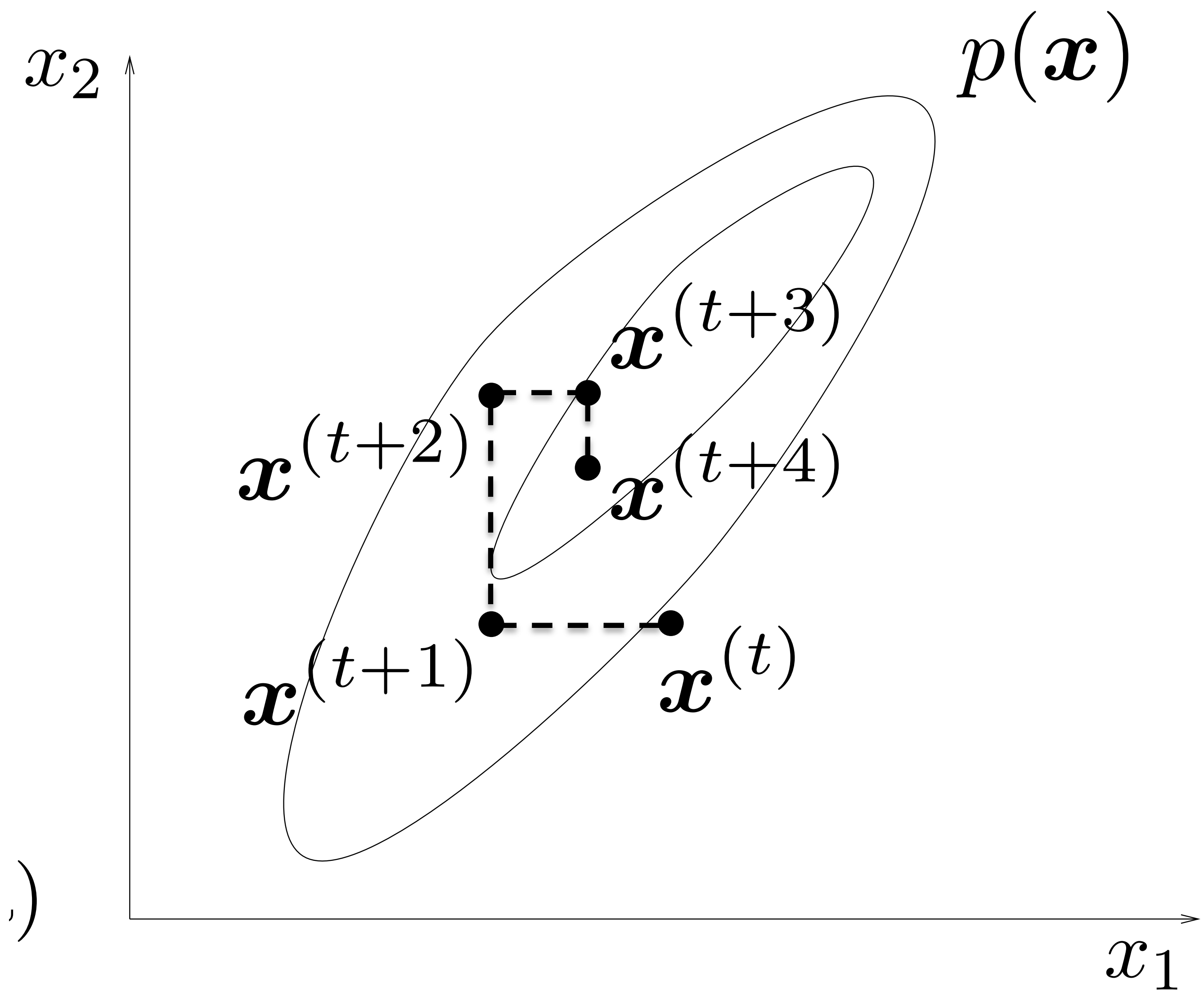
# Gibbs Sampling



# Gibbs Sampling



# Gibbs Sampling



# Gibbs Sampling

**Question:**
How do we draw samples from a conditional distribution?
$y_1, y_2, \ldots, y_J \sim p(y_1, y_2, \ldots, y_J \mid x_1, x_2, \ldots, x_J)$

**(Approximate) Solution:**

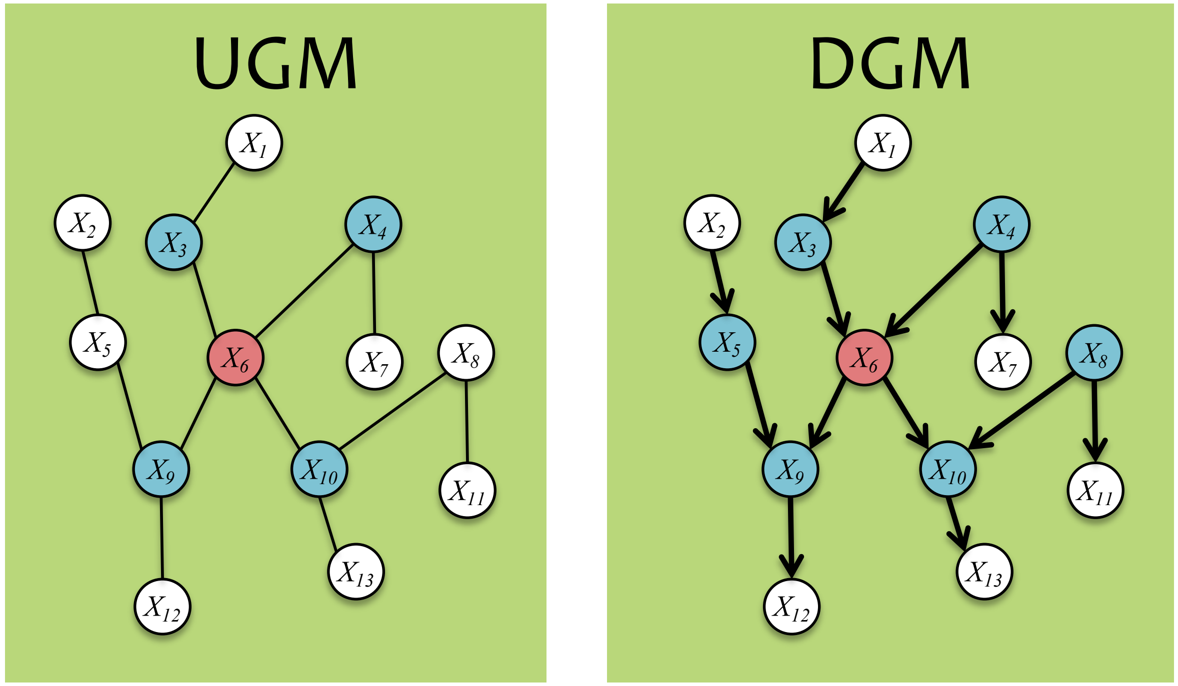
- Initialize $y_1^{(0)}, y_2^{(0)}, \ldots, y_J^{(0)}$ to arbitrary values
- For $t = 1, 2, \ldots$:
  - $y_1^{(t+1)} \sim p(y_1 \mid y_2^{(t)}, \ldots, y_J^{(t)}, x_1, x_2, \ldots, x_J)$
  - $y_2^{(t+1)} \sim p(y_2 \mid y_1^{(t+1)}, y_3^{(t)}, \ldots, y_J^{(t)}, x_1, x_2, \ldots, x_J)$
  - $y_3^{(t+1)} \sim p(y_3 \mid y_1^{(t+1)}, y_2^{(t+1)}, y_4^{(t)}, \ldots, y_J^{(t)}, x_1, x_2, \ldots, x_J)$
  - …
  - $y_J^{(t+1)} \sim p(y_J \mid y_1^{(t+1)}, y_2^{(t+1)}, \ldots, y_{J-1}^{(t+1)}, x_1, x_2, \ldots, x_J)$

**Properties:**

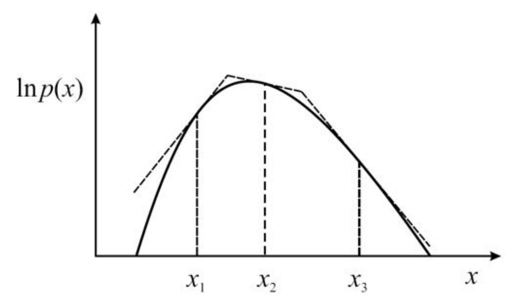
- This will eventually yield samples from $p(y_1, y_2, \ldots, y_J \mid x_1, x_2, \ldots, x_J)$
- But it might take a long time -- just like other Markov Chain Monte Carlo methods

# Gibbs Sampling

**Full conditionals** only need to condition on the **Markov Blanket**




- Must be "easy" to sample from conditionals
- Many conditionals are log-concave and are amenable to adaptive rejection sampling



# METROPOLIS-HASTINGS

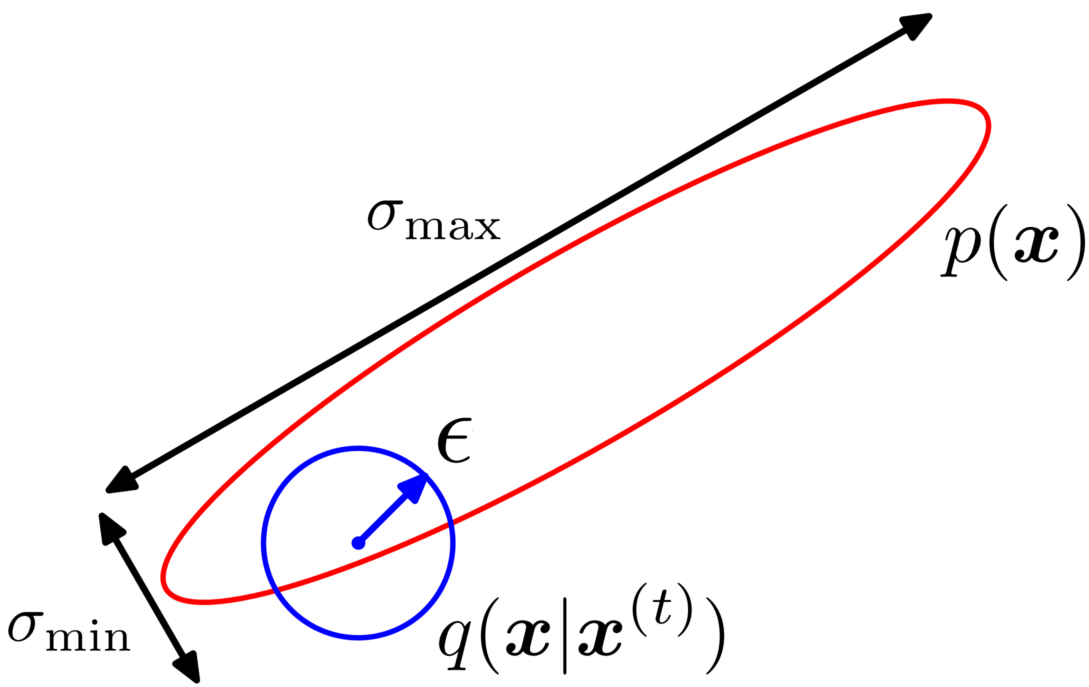
# Metropolis-Hastings

***Whiteboard***

- Metropolis Algorithm
- Metropolis-Hastings Algorithm

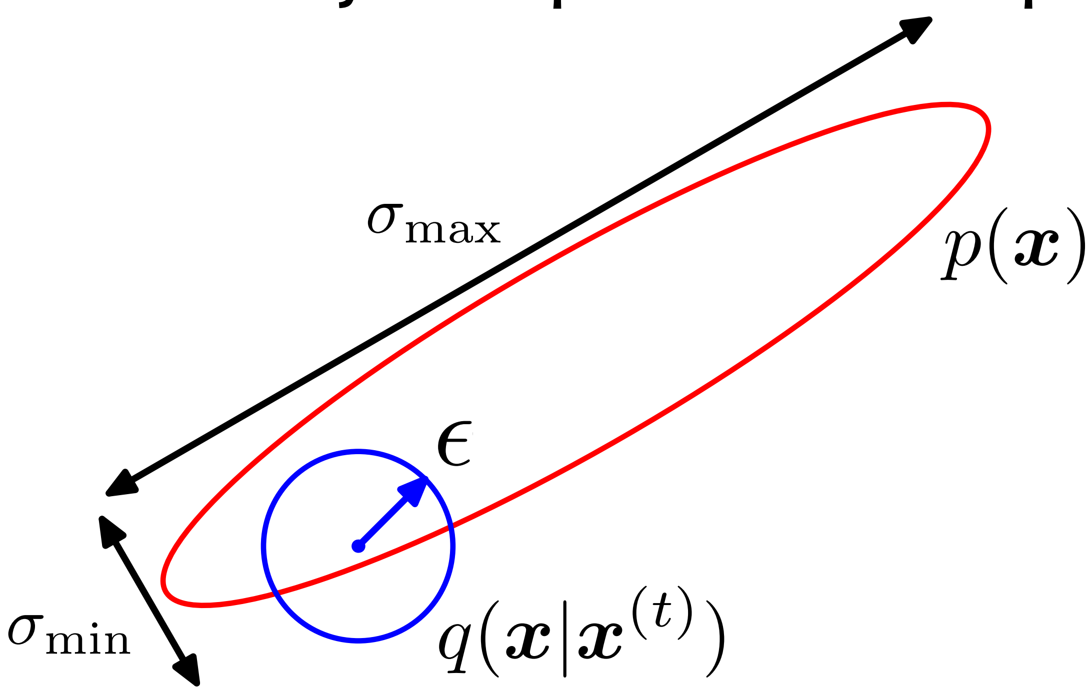
# Random Walk Behavior of M-H

- For Metropolis-Hastings, a generic proposal distribution is: $q(x|x^{(t)}) = \mathcal{N}(0, \epsilon^2)$
- If ϵ is large, many rejections
- If ϵ is small, slow mixing



# Random Walk Behavior of M-H

- For Rejection Sampling, the accepted samples are are **independent**
- But for Metropolis-Hastings, the samples are **correlated**
- **Question:** How long must we wait to get effectively independent samples?




**A:** independent states in the M-H random walk are separated by roughly $(\sigma_{\max}/\sigma_{\min})^2$ steps

# Whiteboard

- Gibbs Sampling as M-H

# MCMC IN PRACTICE

# Practical Issues

- **Question:** Is it better to move along one dimension or many?

- **Answer:** For Metropolis-Hasings, it is sometimes better to sample one dimension at a time
  - Q: Given a sequence of 1D proposals, compare rate of movement for **one-at-a-time** vs. **concatenation.**

- **Answer:** For Gibbs Sampling, sometimes better to sample a block of variables at a time
  - Q: When is it tractable to sample a block of variables?

# Blocked Gibbs Sampling

**Goal:**
Draw samples from a distribution $y_1, y_2, \ldots, y_J \sim p(y_1, y_2, \ldots, y_J)$

**Algorithm:**

- Initialize $y_1, y_2, \ldots, y_J$ to arbitrary values
- For t = 1, 2, … :

  for b in B: where $b \subseteq \{1, \ldots, J\}$

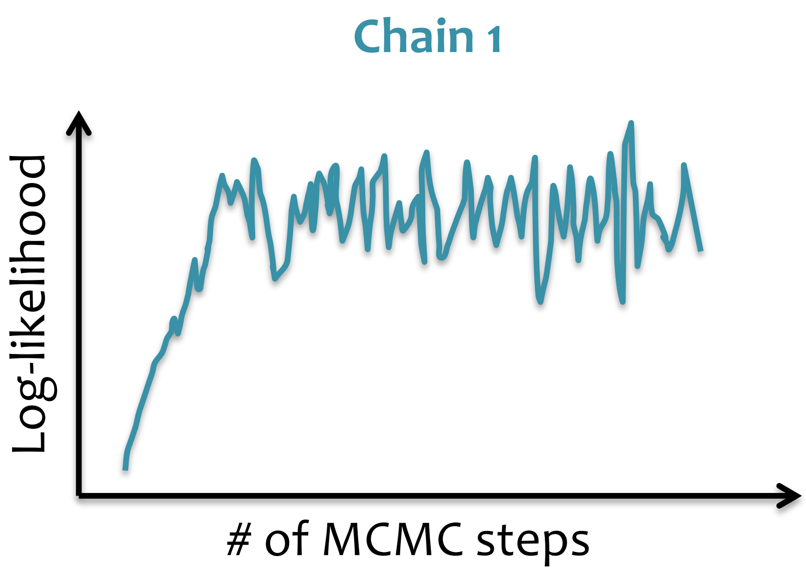
  $$y_b \sim p(y_b \mid y_{\neg b})$$
- Example: B = set of factors in a factor graph

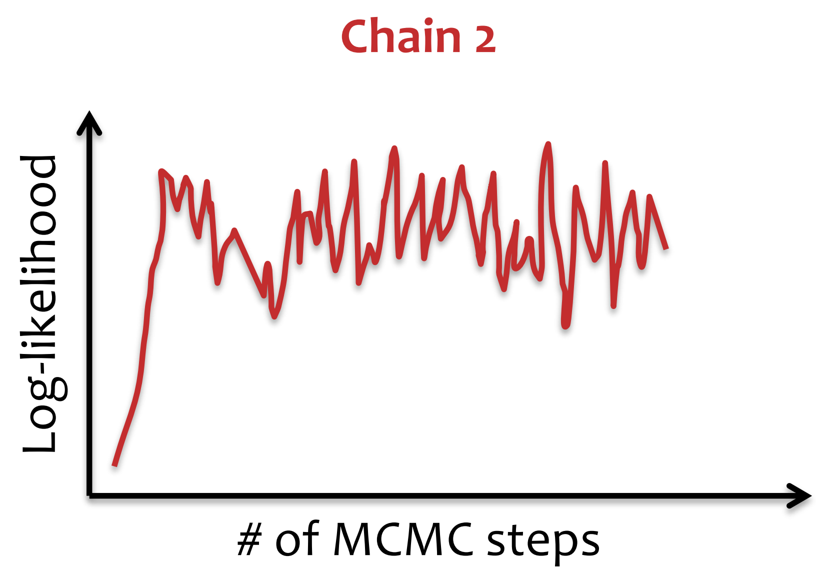
**Why use blocks?**

- As in Gibbs Sampler, this will eventually yield samples from $p(y_1, y_2, \ldots, y_J)$
- **Might improve mixing time** (i.e. "eventually" will be a bit sooner)

# Practical Issues

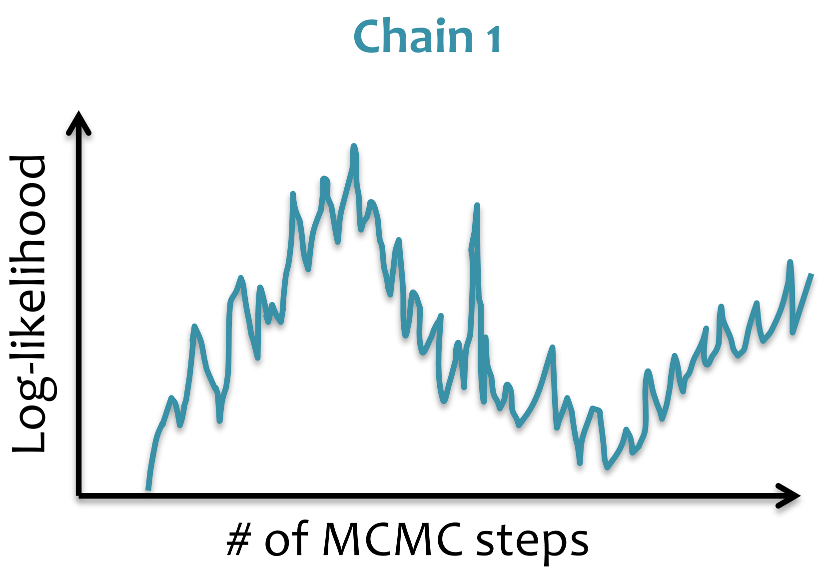
- **Question:** How do we assess convergence of the Markov chain?
- **Answer:** It's not easy!
  - Compare statistics of multiple independent chains
  - Ex: Compare log-likelihoods

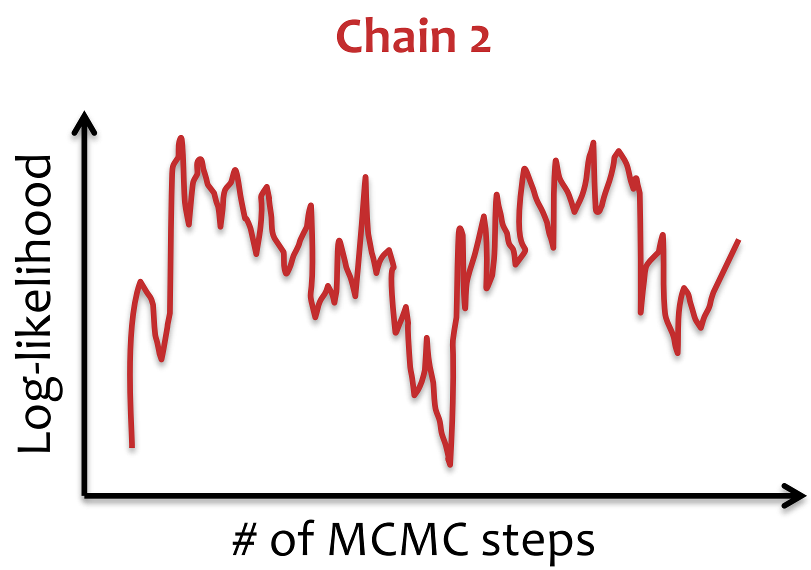


# Practical Issues

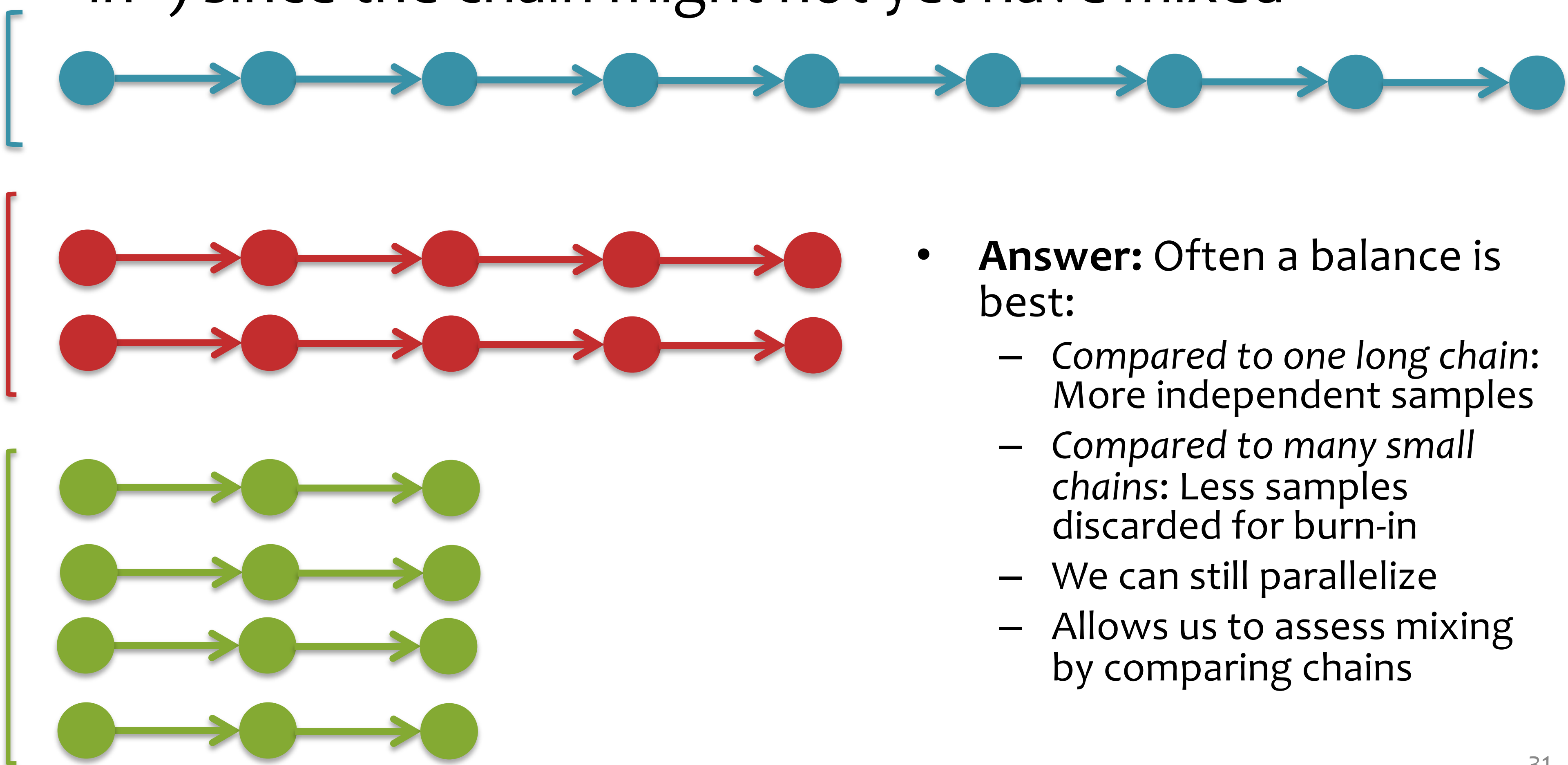
- **Question:** How do we assess convergence of the Markov chain?
- **Answer:** It's not easy!
  - Compare statistics of multiple independent chains
  - Ex: Compare log-likelihoods



# Practical Issues

- **Question:** Is one long Markov chain better than many short ones?
- **Note:** typical to discard initial samples (aka. "burn-in") since the chain might not yet have mixed



- **Answer:** Often a balance is best:
  – *Compared to one long chain*: More independent samples
  – *Compared to many small chains*: Less samples discarded for burn-in
  – We can still parallelize
  – Allows us to assess mixing by comparing chains