Inference: Sampling

CS 6355: Structured Prediction

So far in the class

- Thinking about structures
  - A graph, a collection of parts that are labeled jointly, a collection of decisions
- Algorithms for learning
  - Local learning
    - Learn parameters for individual components independently
    - Learning algorithm not aware of the full structure
  - Global learning
    - Learn parameters for the full structure
    - Learning algorithm "knows" about the full structure
- This section: ***Prediction***
  - Sets structured prediction apart from binary/multiclass

Inference

- What is inference?
  - An overview of what we have seen before
  - Combinatorial optimization
  - Different views of inference
- Graph algorithms
  - Dynamic programming, greedy algorithms, search
- Integer programming
- Heuristics for inference
  - Sampling
- Learning to search

Lecture outline

- Inexact inference
- Markov chains and stationary distributions
- MCMC algorithms
  - Metropolis Hastings
  - Gibbs Sampling

Lecture outline

- *Inexact inference*
- Markov chains and stationary distributions
- MCMC algorithms
  - Metropolis Hastings
  - Gibbs Sampling

So far we have seen…

- Inference as graph search
- Inference with dynamic programming
- Inference via integer linear programming
- But…
  - Inference can be intractable
  - The number of possible outcomes can be very large

Inference questions

- This class:
  - Mostly we use inference to mean “What is the highest scoring assignment to the output random variables for a given input?”
  - Maximum A Posteriori (MAP) inference (if the score is probabilistic)

Inference questions

- This class:
  - Mostly we use inference to mean "What is the highest scoring assignment to the output random variables for a given input?"
  - Maximum A Posteriori (MAP) inference (if the score is probabilistic)

- Other inference questions
  - What is the highest scoring assignment to *some* of the output variables given the input?
  - Sample from the posterior distribution over the **Y**
  - Loss-augmented inference: Which structure most violates the margin for a given scoring function?
  - Computing marginal probabilities over **Y**

A problem: Exact inference is hard!

- We have a probabilistic graphical model
  - A conditional distribution $P(\mathbf{y} \mid \mathbf{x}) \propto \exp(\mathbf{w}^T \phi(\boldsymbol{x}, \boldsymbol{y}))$

- Inference questions can broadly take two forms

A problem: Exact inference is hard!

- We have a probabilistic graphical model
  - A conditional distribution $P(\mathbf{y} \mid \mathbf{x}) \propto \exp(\mathbf{w}^T \phi(\boldsymbol{x}, \boldsymbol{y}))$

- Inference questions can broadly take two forms
  - MAP inference: Find the most likely structure

$$\arg\max_{y} P(\mathbf{y} \mid \mathbf{x}) = \arg\max_{\mathbf{y}} \mathbf{w}^T \phi(\mathbf{x}, \mathbf{y})$$

A problem: Exact inference is hard!

- We have a probabilistic graphical model
  - A conditional distribution $P(\mathbf{y} \mid \mathbf{x}) \propto \exp(\mathbf{w}^T \phi(\boldsymbol{x}, \boldsymbol{y}))$

- Inference questions can broadly take two forms
  - MAP inference: Find the most likely structure

$$\arg\max_{y} P(\mathbf{y} \mid \mathbf{x}) = \arg\max_{\mathbf{y}} \mathbf{w}^T \phi(\mathbf{x}, \mathbf{y})$$

  - Marginal inference: Compute probability over a set of output variables

$$P_i(y_i) = \sum_{\substack{\hat{\boldsymbol{y}} \\ s.t. \hat{y}_i = y_i}} P(\hat{\mathbf{y}} \mid \mathbf{x})$$

A problem: Exact inference is hard!

- We have a probabilistic graphical model
  - A conditional distribution $P(\mathbf{y} \mid \mathbf{x}) \propto \exp(\mathbf{w}^T \phi(\boldsymbol{x}, \boldsymbol{y}))$

- Inference questions can broadly take two forms
  - MAP inference: Find the most likely structure

$$\arg\max_{y} P(\mathbf{y} \mid \mathbf{x}) = \arg\max_{\mathbf{y}} \mathbf{w}^T \phi(\mathbf{x}, \mathbf{y})$$

  - Marginal inference: Compute probability over a set of output variables

$$P_i(y_i) = \sum_{\substack{\hat{\mathbf{y}} \\ s.t. \hat{y}_i = y_i}} P(\hat{\mathbf{y}} \mid \mathbf{x})$$

- Computationally, exact inference (in both cases) is
  - Easy for tree-like models
  - In the general case, however, intractable

How can we address this problem?

Exact inference can be computationally expensive except in the case of certain restricted families of models

Two broad strategies present themselves:

1. Restrict ourselves to these “well-behaved” models
   (And potentially lose the correctness of our model)

2. Give up on exactness of inference
   (And potentially lose the correctness of our predictor)

How can we address this problem?

Exact inference can be computationally expensive except in the case of certain restricted families of models

Two broad strategies present themselves:

1. Restrict ourselves to these "well-behaved" models
   (And potentially lose the correctness of our model)

Involves modeling choices

2. Give up on exactness of inference
   (And potentially lose the correctness of our predictor)

Involves algorithmic choices

A problem: Exact inference is hard!

- We need to solve inference anyway
  - Inference constructs the final output!
  - Inference can play an important part in learning
    - We have seen examples of global learning

A problem: Exact inference is hard!

- We need to solve inference anyway
  - Inference constructs the final output!
  - Inference can play an important part in learning
    - We have seen examples of global learning

- A solution: Give up exactness
  - Already seen *beam search*: A heuristic graph search
  - Most common "approximate" inference methods:
    1. Monte Carlo sampling
    2. Message passing, belief propagation with non-tree-like graphical models

Inference by sampling

- Monte Carlo methods: A large class of algorithms
  - Origins in physics

- Basic idea:
  - Repeatedly sample from a distribution
  - Compute aggregate statistics from samples
    - Eg: The marginal distribution

- Useful when we have many, many interacting variables

Why sampling works

- Suppose we have some probability distribution $P(z)$
  - Might be a cumbersome function

- We want to answer questions about this distribution
  - What is the mean, mode, etc?

Why sampling works

- Suppose we have some probability distribution $P(z)$
  - Might be a cumbersome function
- We want to answer questions about this distribution
  - What is the mean, mode, etc?
- Approximate with samples from the distribution $\{z_1, z_2, \cdots, z_n\}$
  - Example: Expectation $E_P[f] \approx \frac{1}{n}\sum_i f(z_i)$
- Bounds on large deviations tell us that this is a good estimator
  - Chernoff-Hoeffding style bounds

Why sampling works

- Suppose we have some probability distribution $P(z)$
  - Might be a cumbersome function
- We want to answer questions about this distribution
  - What is the mean, mode, etc?
- Approximate with samples from the distribution $\{z_1, z_2, \cdots, z_n\}$
  - Example: Expectation $E_P[f] \approx \frac{1}{n}\sum_i f(z_i)$

How do we generate samples from this cumbersome distribution?

- Bounds on large devi… …ator
  - Chernoff-Hoeffding style bounds

The Markov Chain Monte Carlo revolution

- Goal: To generate samples from a distribution $P(\mathbf{y}|\mathbf{x})$
  - The target distribution could be intractable to sample from

The Markov Chain Monte Carlo revolution

- Goal: To generate samples from a distribution P($\mathbf{y}|\mathbf{x}$)
  - The target distribution could be intractable to sample from
- Idea: Construct a Markov chain of structures whose *stationary* distribution converges to P
  - An iterative process that constructs examples
  - Initially samples might not be from the target distribution
  - But after a long enough time, the samples are from a distribution that is increasingly close to P

Outline

- Inexact inference
- *Markov chains and stationary distributions*
- MCMC algorithms
  - Metropolis Hastings
  - Gibbs Sampling

A detour

Recall: Markov Chains

A collection of random variables $y_0, y_1, y_2, \cdots$ form a Markov chain if the i^{th} state depends only on the previous one

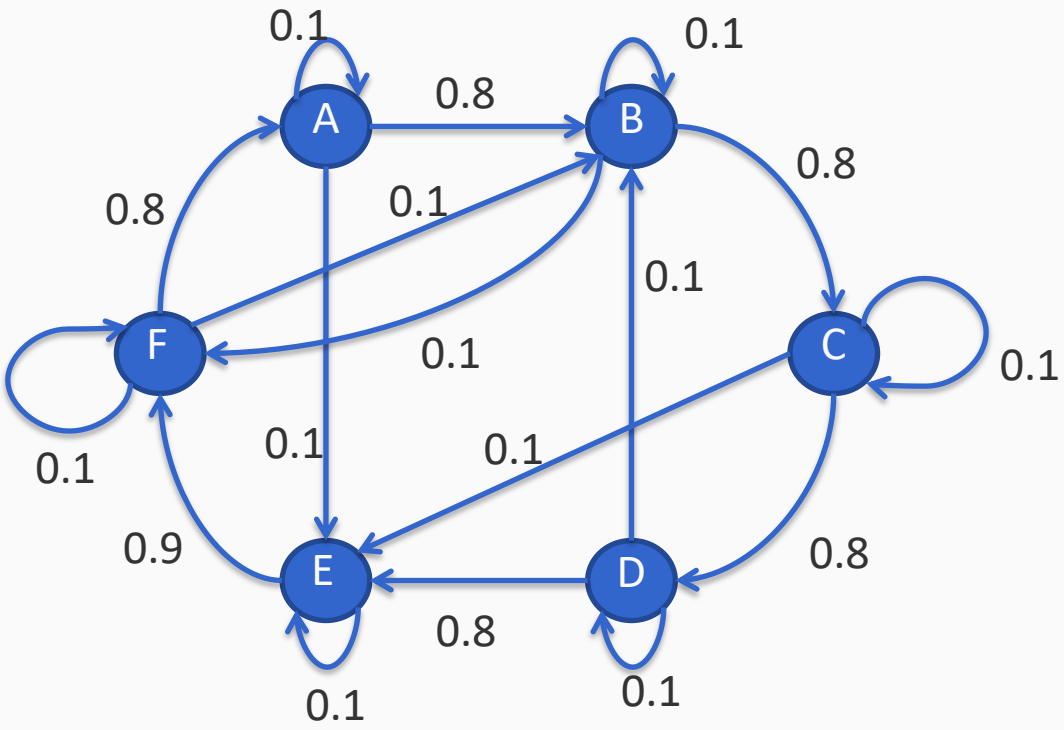
$$P(y_i \mid y_0, y_1, \cdots, y_{i-1}) = P(y_i \mid y_{i-1})$$

A detour

Recall: Markov Chains

A collection of random variables $y_0, y_1, y_2, \cdots$ form a Markov chain if the i^{th} state depends only on the previous one

$$P(y_i \mid y_0, y_1, \cdots, y_{i-1}) = P(y_i \mid y_{i-1})$$



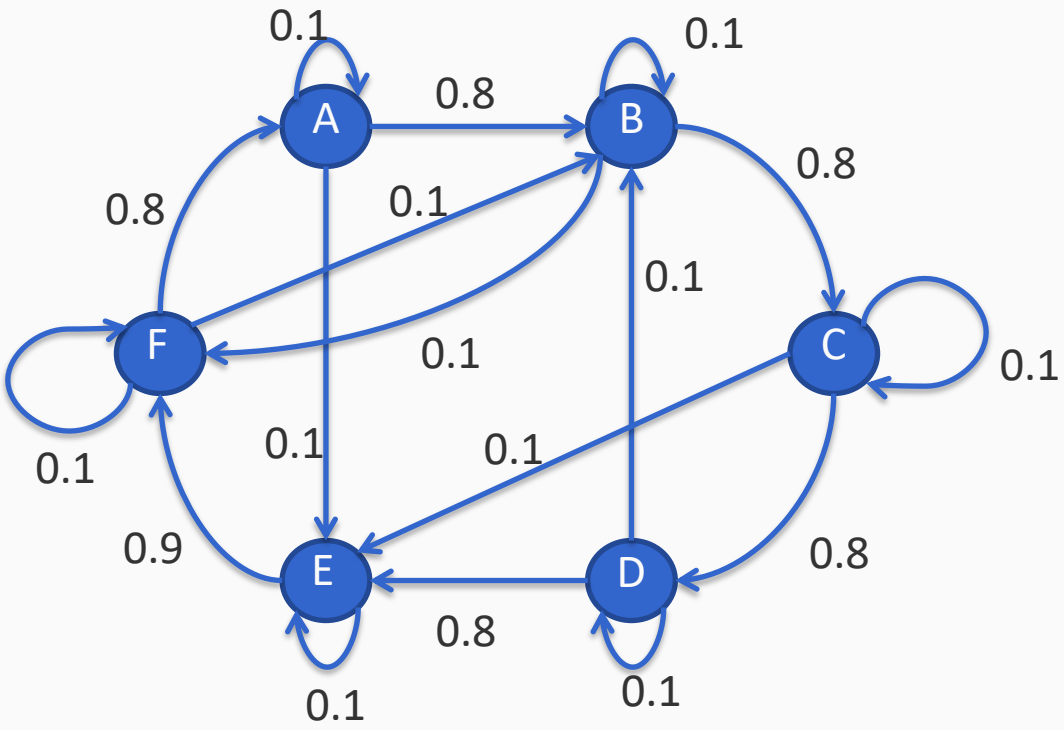
Imagine a state transition diagram of this type

A detour

Recall: Markov Chains

A collection of random variables $y_0, y_1, y_2, \cdots$ form a Markov chain if the i^{th} state depends only on the previous one

$$P(y_i \mid y_0, y_1, \cdots, y_{i-1}) = P(y_i \mid y_{i-1})$$



Imagine a state transition diagram of this type

Sampling from this chain would generate state sequences such as:

A → B → C → D → E → F

F → A → A → E → F → B→ C

Temporal dynamics of a Markov chain

What is the probability that a chain is in a state z at time t+1?

$$P_{t+1}(state_{t+1} = z)$$

Temporal dynamics of a Markov chain

What is the probability that a chain is in a state z at time t+1?

$$P_{t+1}(state_{t+1} = z) = \sum_{z'} P_t(state_t = z')P(z' \rightarrow z)$$

Temporal dynamics of a Markov chain

What is the probability that a chain is in a state z at time t+1?

$$P_{t+1}(state_{t+1} = z) = \sum_{z'} P_t(state_t = z')P(z' \to z)$$

The probability that it was in some state z'at the previous step

The probability that the system moved from state z' to the state z.

Temporal dynamics of a Markov chain

Suppose we have a Markov chain with n states has transition probabilities given by an $n \times n$ matrix $\mathbf{A}$.

Denote the initial state probability of the chain as a vector

$$\mathbf{P}_0 = [P_0(1), P_0(2), \cdots, P_0(n)\ ]$$

Temporal dynamics of a Markov chain

Suppose we have a Markov chain with n states has transition probabilities given by an $n \times n$ matrix $\mathbf{A}$.

Denote the initial state probability of the chain as a vector

$$\mathbf{P}_0 = [P_0(1), P_0(2), \cdots, P_0(n)\ ]$$

After one step, the probability of the system existing in a state i is:

$$P_1(i) = \sum_j P_0(j) P(j \to i) = \sum_j P_0(j) \mathbf{A}_{ji}$$

Temporal dynamics of a Markov chain

Suppose we have a Markov chain with n states has transition probabilities given by an $n \times n$ matrix $\mathbf{A}$.

Denote the initial state probability of the chain as a vector

$$\mathbf{P}_0 = [P_0(1), P_0(2), \cdots, P_0(n)\ ]$$

After one step, the probability of the system existing in a state i is:

$$P_1(i) = \sum_j P_0(j) P(j \to i) = \sum_j P_0(j) \mathbf{A}_{ji}$$

The distribution $P_1(\cdot)$ over all the states is a vector $\mathbf{P}_1 = \mathbf{P_0 A}$.

Temporal dynamics of a Markov chain

Suppose we have a Markov chain with n states has transition probabilities given by an $n \times n$ matrix $\mathbf{A}$.

Denote the initial state probability of the chain as a vector

$$\mathbf{P}_0 = [P_0(1), P_0(2), \cdots, P_0(n)\ ]$$

After one step, the probability of the system existing in a state i is:

$$P_1(i) = \sum_j P_0(j) P(j \to i) = \sum_j P_0(j) \mathbf{A}_{ji}$$

The distribution $P_1(\cdot)$ over all the states is a vector $\mathbf{P}_1 = \mathbf{P_0 A}$.

After two steps, the probability of the system existing in a state i is:

$$P_2(i) = \sum_j P_1(j) P(j \to i) = \sum_j P_1(j) \mathbf{A}_{ji}$$

Temporal dynamics of a Markov chain

Suppose we have a Markov chain with n states has transition probabilities given by an $n \times n$ matrix $\mathbf{A}$.

Denote the initial state probability of the chain as a vector

$$\mathbf{P}_0 = [P_0(1), P_0(2), \cdots, P_0(n)\ ]$$

After one step, the probability of the system existing in a state i is:

$$P_1(i) = \sum_j P_0(j) P(j \to i) = \sum_j P_0(j) \mathbf{A}_{ji}$$

The distribution $P_1(\cdot)$ over all the states is a vector $\mathbf{P}_1 = \mathbf{P_0 A}$.

After two steps, the probability of the system existing in a state i is:

$$P_2(i) = \sum_j P_1(j) P(j \to i) = \sum_j P_1(j) \mathbf{A}_{ji}$$

As before, we have $\mathbf{P}_2 = \mathbf{P}_1 \mathbf{A} = \mathbf{P_0 A A}$

Temporal dynamics of a Markov chain

Suppose we have a Markov chain with n states has transition probabilities given by an $n \times n$ matrix $\mathbf{A}$.

Denote the initial state probability of the chain as a vector

$$\mathbf{P}_0 = [P_0(1), P_0(2), \cdots, P_0(n)\ ]$$

After one step, the probability of the system existing in a state i is:

$$P_1(i) = \sum_j P_0(j) P(j \to i) = \sum_j P_0(j) \mathbf{A}_{ji}$$

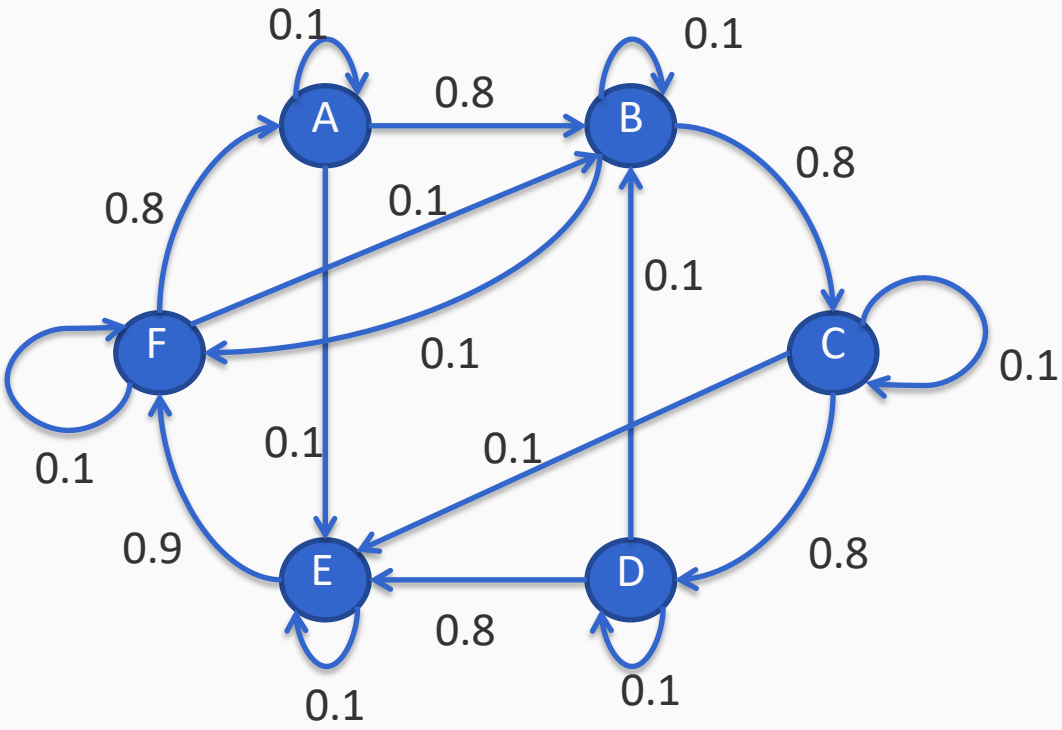
The distribution $P_1(\cdot)$ over all the states is a vector $\mathbf{P}_1 = \mathbf{P_0 A}$.

In general, after n steps, the probability of the system is given by

$$\mathbf{P}_n = \mathbf{P}_{n-1} \mathbf{A}$$

Exercise

Suppose a Markov chain for these transition probabilities starts at state C. What is the distribution over states after three steps?



Stationary distributions

Suppose we have a Markov chain with transition probability $\mathbf{A}$.

A distribution over states π is a stationary distribution if, *after a transition*, the probability of the system being at any state is unchanged.

Stationary distributions

Suppose we have a Markov chain with transition probability $\mathbf{A}$.

A distribution over states π is a stationary distribution if, *after a transition*, the probability of the system being at any state is unchanged.

Mathematically: if $\mathbf{A}$ is the transition matrix, we have $\pi = \pi\mathbf{A}$.

Stationary distributions

Suppose we have a Markov chain with transition probability $\mathbf{A}$.

A distribution over states π is a stationary distribution if, *after a transition*, the probability of the system being at any state is unchanged.

Mathematically: if $\mathbf{A}$ is the transition matrix, we have $\pi = \pi\mathbf{A}$.

How do we get to a stationary distribution?

- A regular Markov chain: There is an non-zero probability of getting from any state to any other in a finite number of steps
- If transition matrix is regular, just run it for a long time
- Steady-state behavior is the stationary distribution

Outline

- Inexact inference
- Markov chains and stationary distributions
- *MCMC algorithms*
  - *Metropolis Hastings*
  - *Gibbs Sampling*

Markov Chain Monte Carlo for inference

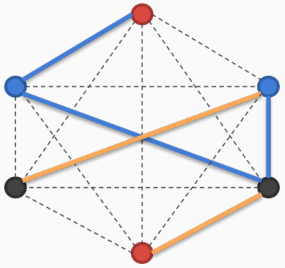
- Design a Markov chain such that
  - Every state is a structure
  - The stationary distribution of the chain is the probability distribution we care about P(**y** |**x**)

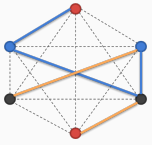
Markov Chain Monte Carlo for inference

- Design a Markov chain such that
  - Every state is a structure
  - The stationary distribution of the chain is the probability distribution we care about P(**y** |**x**)

- How to do inference?
  - Run the Markov chain for a long time till we think it gets to steady state
  - Let the chain wander around the space and collect samples
  - We have samples from P(**y**|**x**)

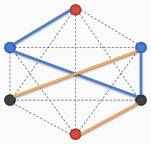
MCMC for inference

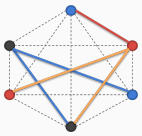




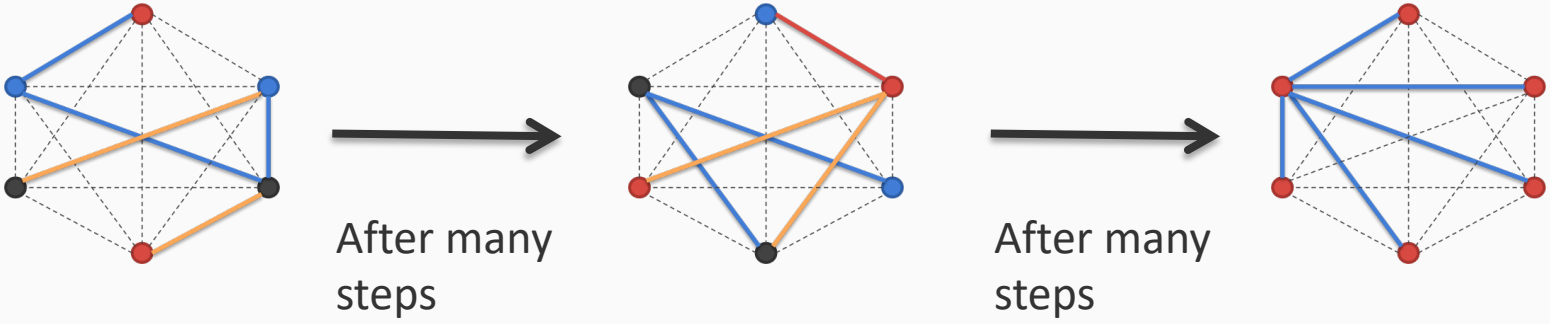
1

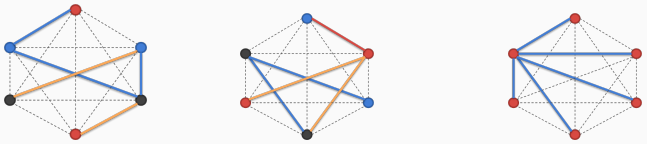
MCMC for inference


1

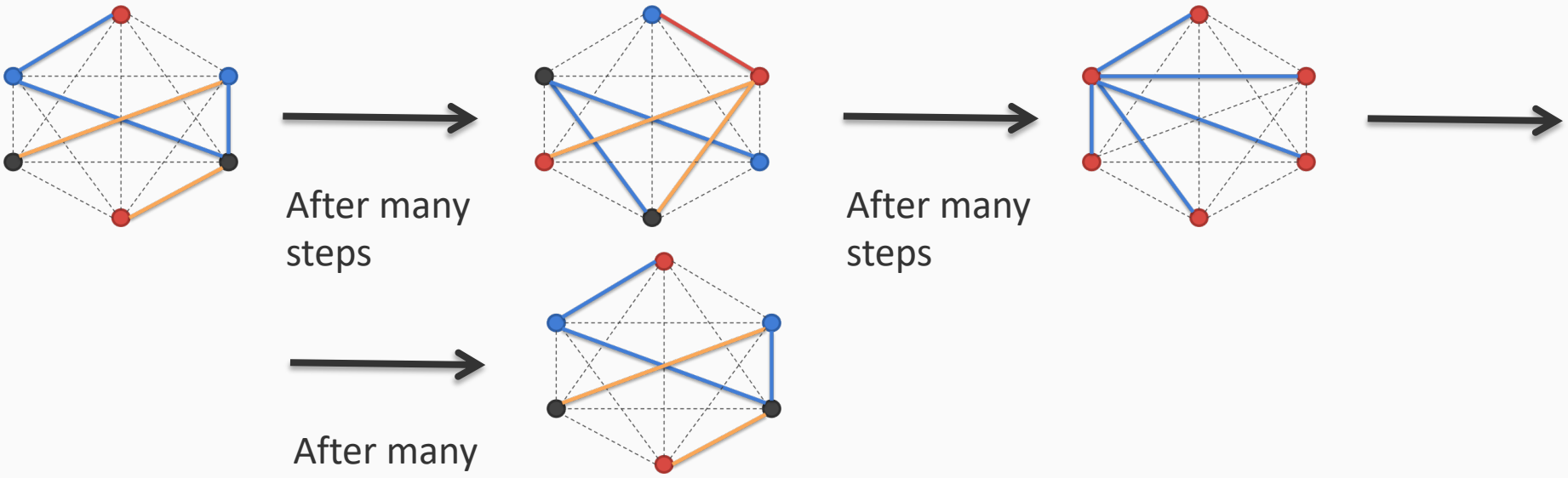

1

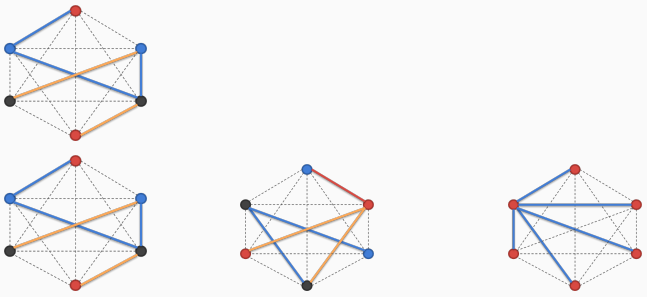
MCMC for inference



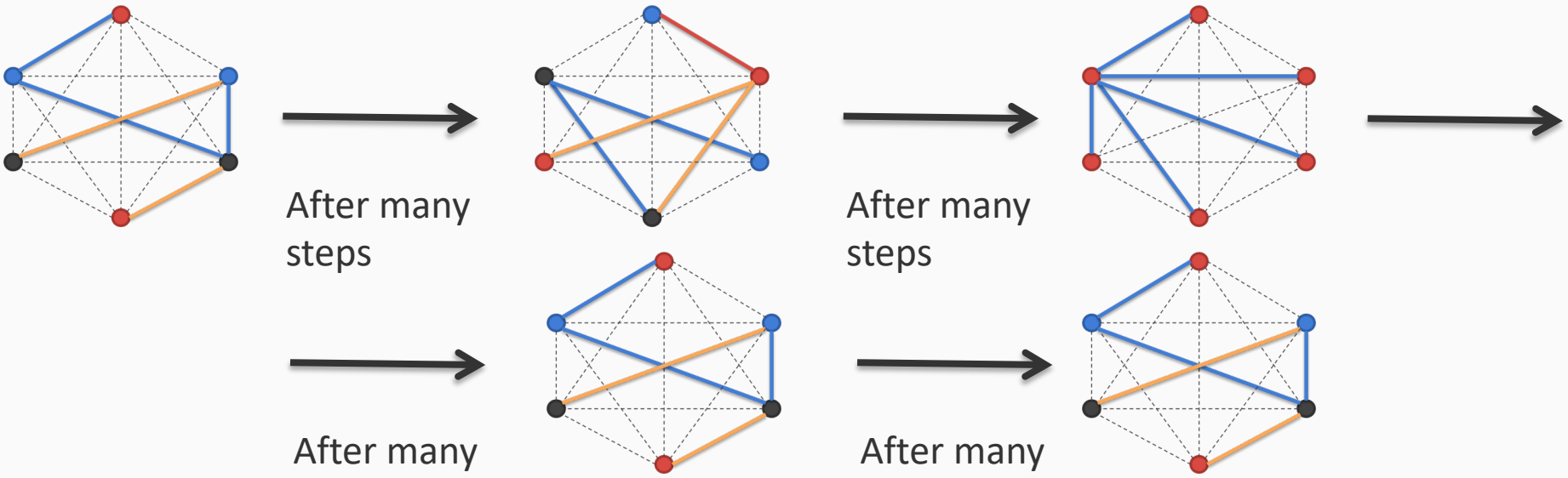


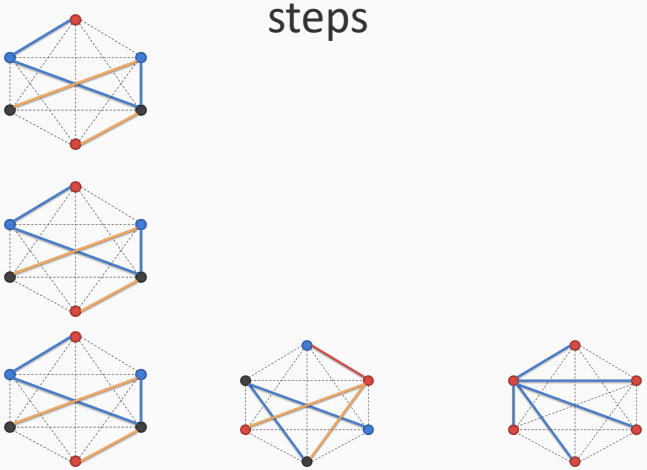
MCMC for inference



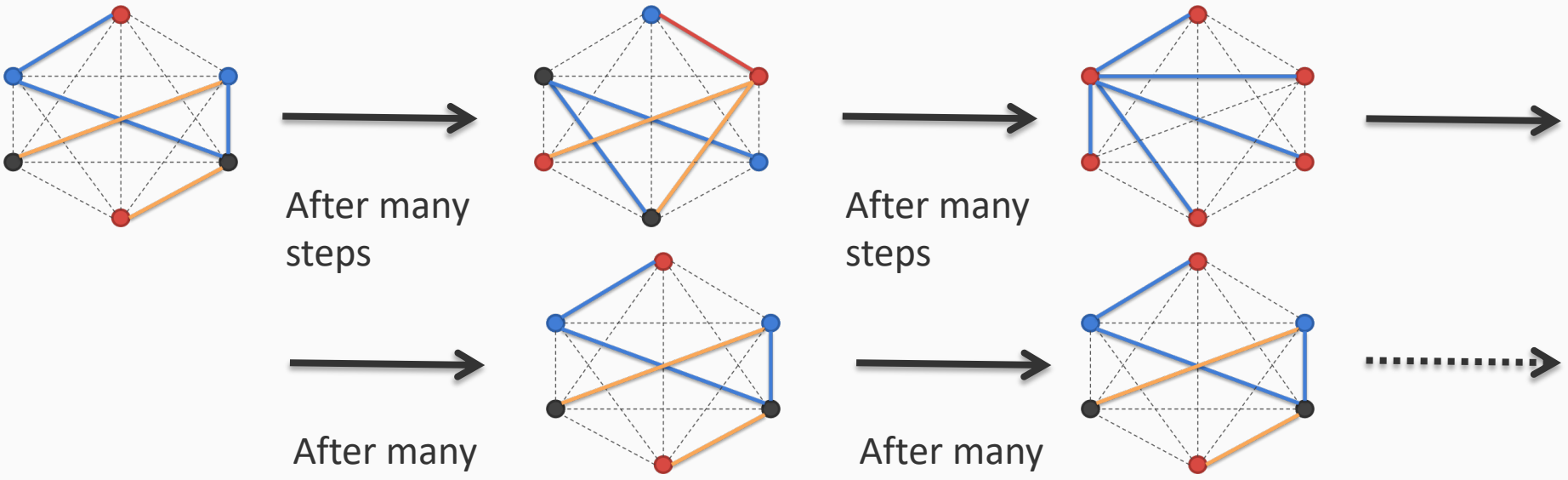


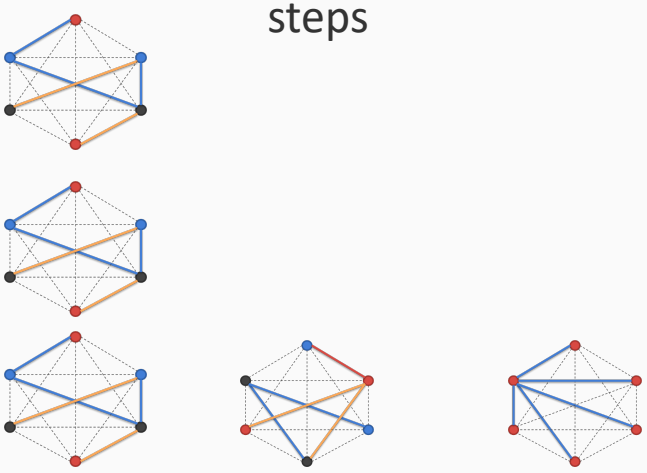
MCMC for inference





MCMC for inference





With sufficient samples, we can answer inference questions like calculating the partition function (just sum over the samples)

MCMC algorithms

- Metropolis-Hastings algorithm

- Gibbs sampling
  - An instance of the Metropolis Hastings algorithm
  - Many variants exist

- Remember: We are sampling from an exponential state space
  - All possible assignments to the random variables

Metropolis-Hastings

[Metropolis, Rosenbluth, Rosenbluth, Teller & Teller 1953]
[Hastings 1970]

- Proposal distribution $q(\mathbf{y} \rightarrow \mathbf{y}')$
  - Shorthand for $q(state_{t+1} = \mathbf{y}' \mid state_t = \mathbf{y})$
  - Proposes changes to the state
  - Could propose large changes to the state

- Acceptance probability α
  - Should the proposal be accepted or not
  - If yes, move to the proposed state, else remain in the previous state
  - Intended to strike a balance between:
    1. Preferring higher probability states and
    2. Not getting stuck in one region of the state space

Metropolis-Hastings

[Metropolis, Rosenbluth, Rosenbluth, Teller & Teller 1953]
[Hastings 1970]

- Metropolis et al 1953:
  - Symmetric proposal distributions
  - Invented to study the states of packed rigid spheres in two and three dimensions
  - Implemented on `MANIAC I` (an early computer)
    - 1024 vacuum tubes (i.e. bits) of memory!

- Hastings 1970:
  - Updated to general distributions

For a interesting retrospective on how the algorithm came about, see
Marshall N. Rosenbluth. 2003. Genesis of the Monte Carlo Algorithm for Statistical Mechanics. In *AIP Conference Proceedings*, volume 690, pages 22–30, Los Alamos, New Mexico (USA). AIP.

Metropolis-Hastings Algorithm

The distribution we care about is $P(\mathbf{y}|\mathbf{x})$

- Start with an initial guess $\mathbf{y}^0$
- Loop for t = 1, 2, … N
  - Propose next state $\mathbf{y}'$
  - Calculate acceptance probability $\alpha(\mathbf{y}^t, \mathbf{y}')$
  - With probability α, accept proposal
    - If accepted: $\mathbf{y}^{t+1} \leftarrow \mathbf{y}'$, else $\mathbf{y}^t \leftarrow \mathbf{y}'$
- Return the last M samples

Metropolis-Hastings Algorithm

The distribution we care about is $P(\mathbf{y}|\mathbf{x})$

- Start with an initial guess $\mathbf{y}^0$
- Loop for t = 1, 2, … N
  - Propose next state $\mathbf{y}'$
  - Calculate acceptance probability $\alpha(\mathbf{y}^t, \mathbf{y}')$
  - With probability α, accept proposal
    - If accepted: $\mathbf{y}^{t+1} \leftarrow \mathbf{y}'$, else $\mathbf{y}^t \leftarrow \mathbf{y}'$
- Return the last M samples

Sample from $q(\mathbf{y} \rightarrow \mathbf{y}')$

The proposal distribution

Metropolis-Hastings Algorithm

The distribution we care about is $P(\mathbf{y}|\mathbf{x})$

- Start with an initial guess $\mathbf{y}^0$
- Loop for t = 1, 2, … N
  - Propose next state $\mathbf{y}'$
  - Calculate acceptance probability $\alpha(\mathbf{y}^t, \mathbf{y}')$
  - With probability α, accept proposal
    - If accepted: $\mathbf{y}^{t+1} \leftarrow \mathbf{y}'$, else $\mathbf{y}^t \leftarrow \mathbf{y}'$
- Return the last M samples

Sample from $q(\mathbf{y} \rightarrow \mathbf{y}')$

The proposal distribution

$$\min\left(1, \frac{P(\mathbf{y}'|\mathbf{x})}{P(\mathbf{y}|\mathbf{x})} \frac{q(\mathbf{y}' \rightarrow \mathbf{y})}{q(\mathbf{y} \rightarrow \mathbf{y}')}\right)$$

Metropolis-Hastings Algorithm

The distribution we care about is $P(\mathbf{y}|\mathbf{x})$

- Start with an initial guess $\mathbf{y}^0$
- Loop for t = 1, 2, … N
  - Propose next state $\mathbf{y}'$
  - Calculate acceptance probability $\alpha(\mathbf{y}^t, \mathbf{y}')$
  - With probability α, accept proposal
    - If accepted: $\mathbf{y}^{t+1} \leftarrow \mathbf{y}'$, else $\mathbf{y}^t \leftarrow \mathbf{y}'$
- Return the last M samples

Sample from $q(\mathbf{y} \rightarrow \mathbf{y}')$

The proposal distribution

$$\min\left(1, \frac{P(\mathbf{y}'|\mathbf{x})}{P(\mathbf{y}|\mathbf{x})} \frac{q(\mathbf{y}' \rightarrow \mathbf{y})}{q(\mathbf{y} \rightarrow \mathbf{y}')}\right)$$

Important: This does not require us to calculate the partition function. **Why?**

And why is this good?

Metropolis-Hastings guarantees

As the number of samples grows large, the empirical distribution of the samples approaches the distribution we want to sample from.

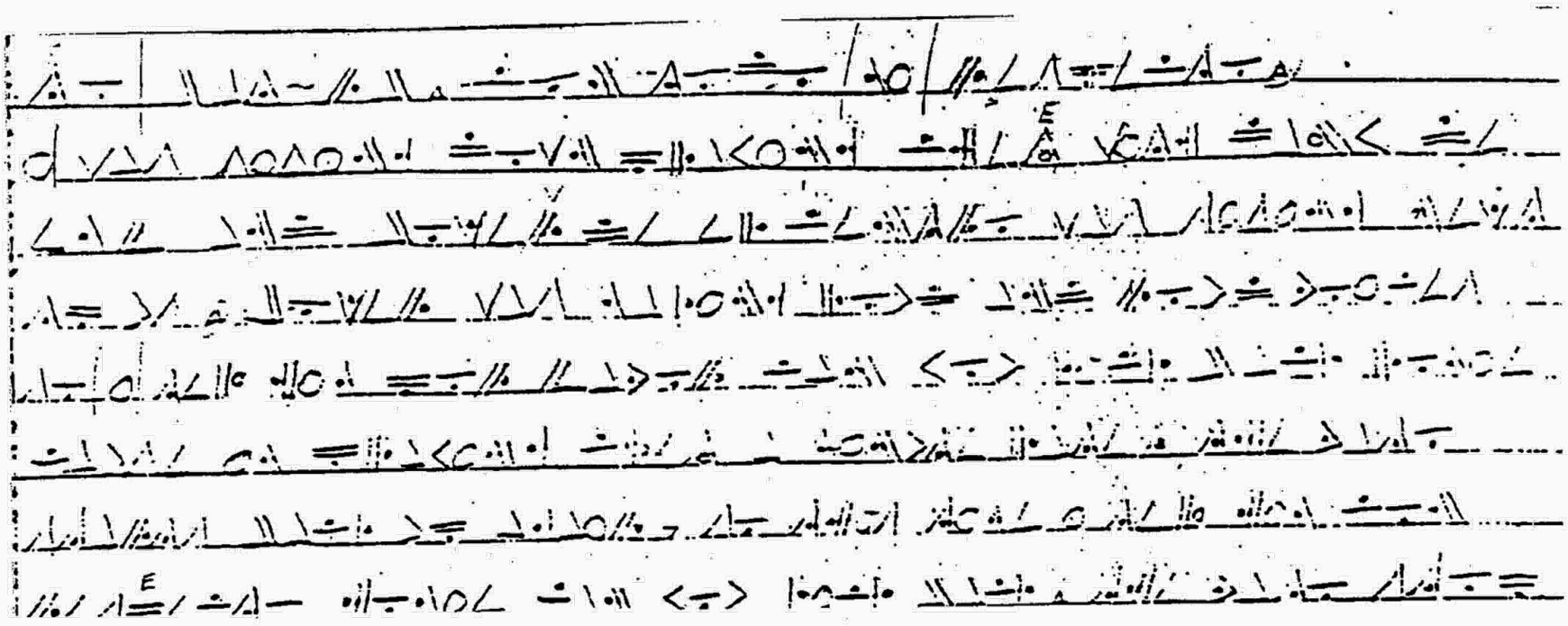
- Subject to some mild restrictions on the probability P we want to sample from and the proposal distribution Q

Proposal functions for Metropolis

- A design choice
- Different possibilities
  - Only make local changes to the factor graph
    - But the chain might not explore widely
  - Make big jumps in the state space
    - But the chain might move very slowly
- Doesn’t have to depend on the size of the graph

A non-machine learning example

Decoding text: What is the coded message here?



[From Persi Diaconis]

Decoding the text

- Suppose we have a way of getting a good estimate for character bigram probability
  - How?
- We can now compute P(English string) using this
- Assume that a substitution cipher is used for encoding
  - Every character is mapped to a different one
- Our goal is to find the mapping
- Let's try Metropolis-Hastings

Metropolis example

- The state space: The set of all possible complete character mappings
  - Eg: {A → B, B → Z, C → 1, …} could be a state
- The distribution we want to sample from

  P(decoded string)
- The proposal:
  - Given a state, pick two characters randomly and swap their mappings


The Markov Chain Monte Carlo Revolution

Persi Diaconis*

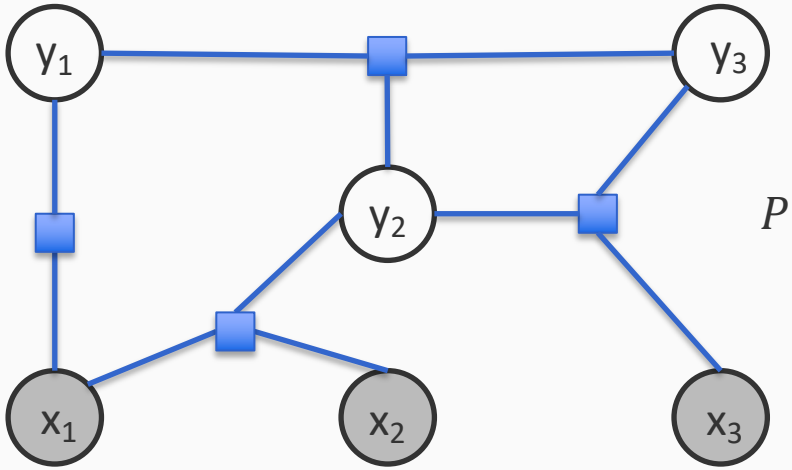
Gibbs sampling

- Start with an initial guess $\mathbf{y} = (y_1, y_2, \cdots, y_n)$
- Loop several times
  - For i = 1 to n:
    - Sample y_i from $P(y_i | y_1, y_2, \cdots y_{i-1}, y_{i+1}, \cdots, y_n, \mathbf{x})$
  - We now have a complete sample

The ordering is arbitrary

A specific instance of Metropolis-Hastings algorithm, no proposal needs to be designed

Gibbs sampling: An example



$$P(y_1, y_2, y_3 | \mathbf{x}) \propto \exp\left(w^T \phi(\mathbf{x}, \mathbf{y})\right)$$

Suppose we start with (A, B, A)

What will one step of Gibbs sampling look like?

Once again, no need to calculate Z

Questions?

Gibbs sampling has many variants

The task: Sample from $P(y_1, y_2, y_3 | x)$

- Gibbs sampling:
  - Fix y_2^t, y_3^t and draw y_1^{t+1}
  - Fix y_1^{t+1}, y_3^t and draw y_2^{t+1}
  - Fix $y_1^{t+1}, y_2^{t+1}$and draw y_3^{t+1}

Gibbs sampling has many variants

The task: Sample from $P(y_1, y_2, y_3 | x)$

- Gibbs sampling:
  - Fix y_2^t, y_3^t and draw y_1^{t+1}
  - Fix y_1^{t+1}, y_3^t and draw y_2^{t+1}
  - Fix y_1^{t+1}, y_2^{t+1} and draw y_3^{t+1}

- **Block** Gibbs sampling: Group the random variables into blocks. Say y_1, y_2 form a block and y_3 forms another
  - Fix y_3^t and draw y_1^{t+1}, y_2^{t+1} from $P(y_1, y_2 | y_3^t, x)$
  - Fix y_1^{t+1}, y_2^{t+1} and draw y_3^{t+1} from $P(y_3 | y_1^{t+1}, y_2^{t+1}, x)$

Gibbs sampling has many variants

The task: Sample from $P(y_1, y_2, y_3 | x)$

- Gibbs sampling:
  - Fix y_2^t, y_3^t and draw y_1^{t+1}
  - Fix y_1^{t+1}, y_3^t and draw y_2^{t+1}
  - Fix y_1^{t+1}, y_2^{t+1} and draw y_3^{t+1}

- **Block** Gibbs sampling: Group the random variables into blocks. Say y_1, y_2 form a block and y_3 forms another
  - Fix y_3^t and draw y_1^{t+1}, y_2^{t+1} from $P(y_1, y_2 | y_3^t, x)$
  - Fix y_1^{t+1}, y_2^{t+1} and draw y_3^{t+1} from $P(y_3 | y_1^{t+1}, y_2^{t+1}, x)$

Other variants include Collapsed Gibbs sampling, parallel sampling, etc

Gibbs sampling: summary

- A specific instance of the Metropolis Hastings algorithm
  - Fix all random variables except one, sample the remaining one
  - Repeat till all random variables are sampled
- Easy to implement
- No need to implement a proposal function
  - Acceptance ratio is 1

Practical concerns

- Burn-in
  - The chain needs to get to the stationary state
  - Ignore all samples during a burn-in period, a parameter to the algorithm
- Thinning
  - Samples that follow each other are correlated
  - For statistical guarantees, we need independent samples
  - After burn-in, don't take every sample, take every K^{th} sample
  - K is the thinning period, another parameter
- Must start at a state that has non-zero probability

MAP inference with MCMC

- So far we have only seen how to collect samples
- Marginal inference with samples is easy
  - Compute the marginal probabilities from the samples
- MAP inference:
  - Find the sample with highest probability
  - To help convergence to the maximum , acceptance condition becomes

$$\min\left(1, \left(\frac{P(\mathbf{y}'|\mathbf{x})}{P(\mathbf{y}|\mathbf{x})}\right)^{1/t} \frac{q(\mathbf{y}' \to \mathbf{y})}{q(\mathbf{y} \to \mathbf{y}')}\right)$$

T is a temperature parameter that increases with every step
Similar to simulated annealing

Summary of MCMC methods

- A different approach for inference
  - No guarantee of exactness
- General idea
  - Set up a Markov chain whose stationary distribution is the probability distribution that we care about
  - Run the chain, collect samples, aggregate
- Metropolis-Hastings, Gibbs sampling
  - Many, many, many variants abound, we have barely scratched the surface here!
- Useful when exact inference is intractable
  - Typically low memory costs, local changes only for Gibbs sampling

Questions?