Universität Stuttgart
IPVS – Institute for Parallel and Distributed Systems

Analytic Computing

Advanced Topics in Machine Learning
7 Inference - Part 2

Prof. Dr. Steffen Staab
Dr. Rafika Boutalbi
Zihao Wang
https://www.ipvs.uni-stuttgart.de/departments/ac/

Learning Objectives

Inference Methods:

- MAP: Max-Sum
- Sampling
  - Forward sampling
  - MCMC
  - Gibbs Sampling
  - Metropolis Hasting

Disclaimer

Figures and examples not marked otherwise are taken from the book by Koller & Friedman

1 Max-Sum Exact Inference for Cluster Trees

Turn Product into Sum

$$P_{\Phi}(\boldsymbol{x}) \propto \prod_{k} \phi_k\left(D_k = \boldsymbol{x}_k\right)$$

$$\text{argmax} \prod_{k} \phi_k\left(D_k\right)$$

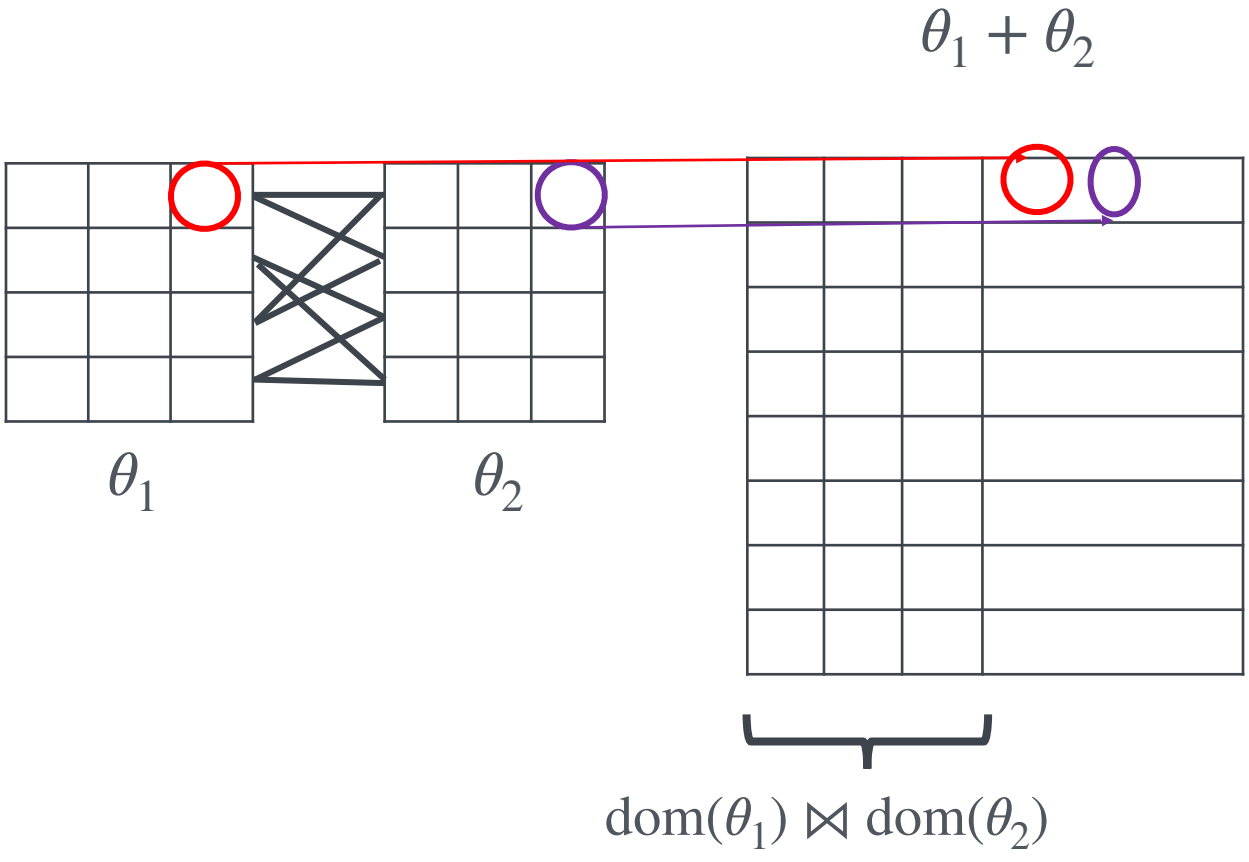
$$\text{argmax} \sum_{k} \theta_k\left(D_k\right) = \text{argmax}\, \theta\left(\mathcal{X}\right),$$

$$\text{where } \theta_k(D_k) = \log \phi_k\left(D_k\right)$$

Remember, e.g., Naive Bayes in ML lecture!

Factor Summation

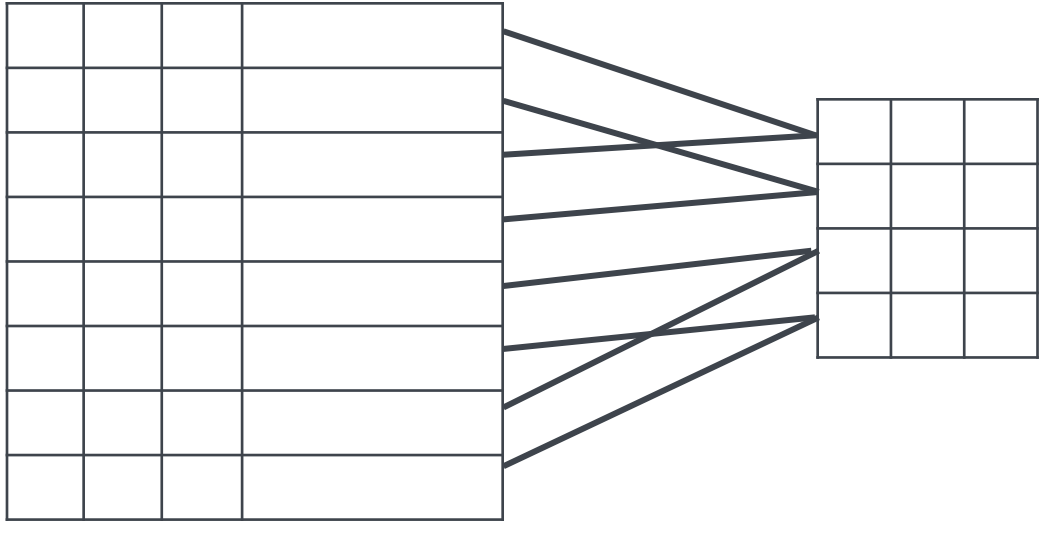
Example



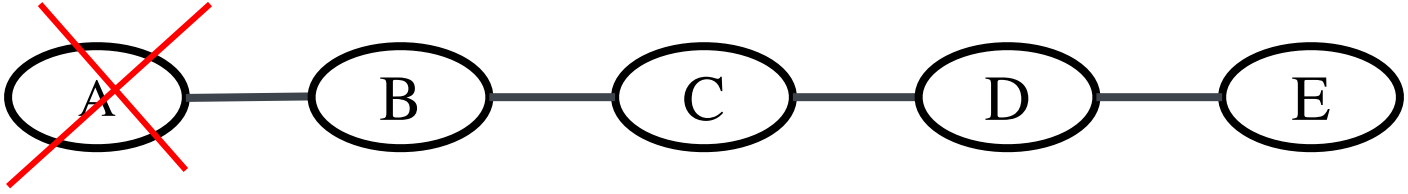
Max Marginalization

$$\theta_1 + \theta_2$$

$$\operatorname*{argmax}_{b \in B}(\theta_1 + \theta_2)$$

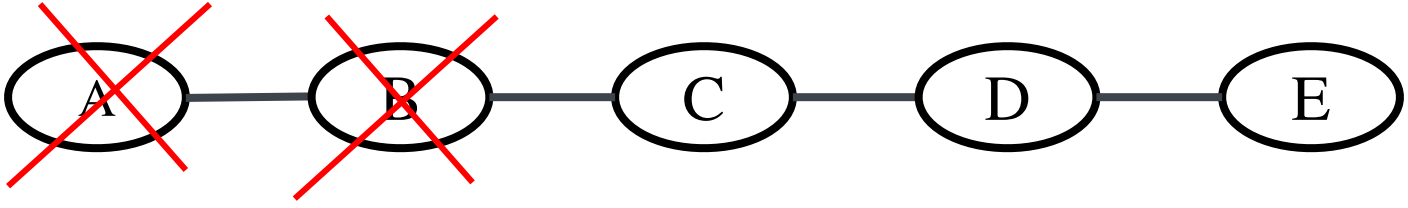


Max-Sum elimination in chains



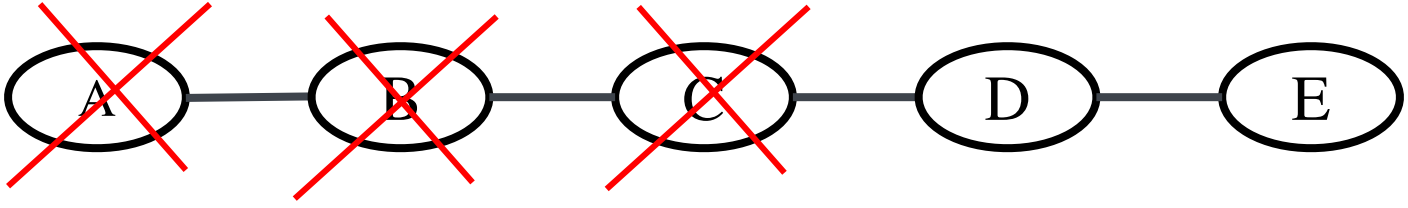
$$\max_{D}\max_{C}\max_{B}\max_{A}\left(\theta_1(A,B)+\theta_2(B,C)+\theta_3(C,D)+\theta_4(D,E)\right) = = \max_{D}\max_{C}\max_{B}\left(\theta_2(B,C)+\theta_3(C,D)+\theta_4(D,E)+\max_{A}\theta_1(A,B)\right) = = \max_{D}\max_{C}\max_{B}\left(\theta_2(B,C)+\theta_3(C,D)+\theta_4(D,E)+\lambda_1(B)\right)$$

Max-Sum elimination in chains



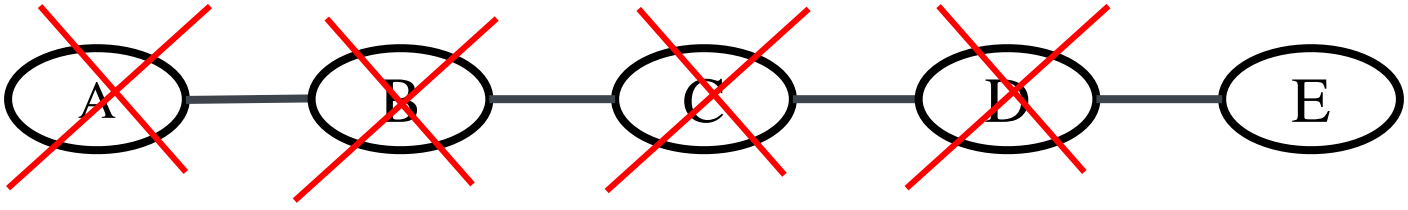
$$\max_{D}\max_{C}\max_{B}\max_{A}\left(\theta_1(A,B)+\theta_2(B,C)+\theta_3(C,D)+\theta_4(D,E)\right) = = \max_{D}\max_{C}\max_{B}\left(\theta_2(B,C)+\theta_3(C,D)+\theta_4(D,E)+\max_{A}\theta_1(A,B)\right) = = \max_{D}\max_{C}\max_{B}\left(\theta_2(B,C)+\theta_3(C,D)+\theta_4(D,E)+\lambda_1(B)\right) = = \max_{D}\max_{C}\left(\theta_3(C,D)+\theta_4(D,E)+\max_{B}(\theta_2(B,C)+\lambda_1(B))\right) = = \max_{D}\max_{C}\left(\theta_3(C,D)+\theta_4(D,E)+\lambda_2(C)\right)$$

Max-Sum elimination in chains



$$\max_{D}\max_{C}\max_{B}\max_{A}\left(\theta_1(A,B)+\theta_2(B,C)+\theta_3(C,D)+\theta_4(D,E)\right) = = \max_{D}\max_{C}\max_{B}\left(\theta_2(B,C)+\theta_3(C,D)+\theta_4(D,E)+\max_{A}\theta_1(A,B)\right) = = \max_{D}\max_{C}\max_{B}\left(\theta_2(B,C)+\theta_3(C,D)+\theta_4(D,E)+\lambda_1(B)\right) = = \max_{D}\max_{C}\left(\theta_3(C,D)+\theta_4(D,E)+\lambda_2(C)\right) = = \max_{D}\left(\theta_4(D,E)+\lambda_3(D)\right)$$

Max-Sum elimination in chains



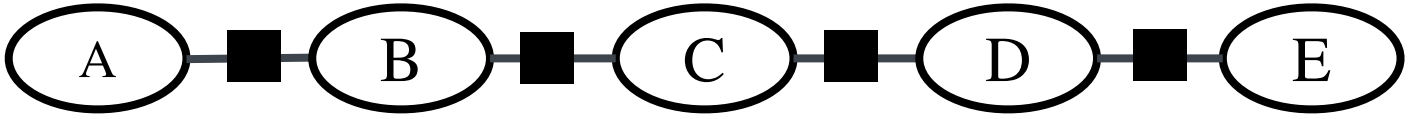
$$\max_{D}\max_{C}\max_{B}\max_{A}\left(\theta_1(A,B)+\theta_2(B,C)+\theta_3(C,D)+\theta_4(D,E)\right) = = \max_{D}\max_{C}\max_{B}\left(\theta_2(B,C)+\theta_3(C,D)+\theta_4(D,E)+\max_{A}\theta_1(A,B)\right) = = \max_{D}\max_{C}\max_{B}\left(\theta_2(B,C)+\theta_3(C,D)+\theta_4(D,E)+\lambda_1(B)\right) = = \max_{D}\max_{C}\left(\theta_3(C,D)+\theta_4(D,E)+\lambda_2(C)\right) = = \max_{D}\left(\theta_4(D,E)+\lambda_3(D)\right) =$$

$$= \lambda_4(e)$$

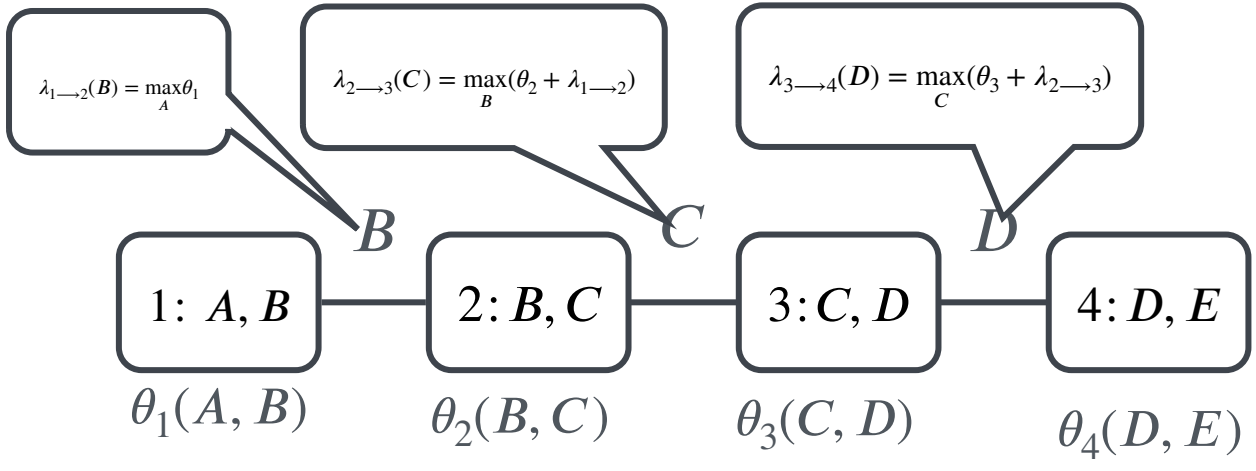
Max marginal

Max-Sum in Cluster Trees

Factor Graph

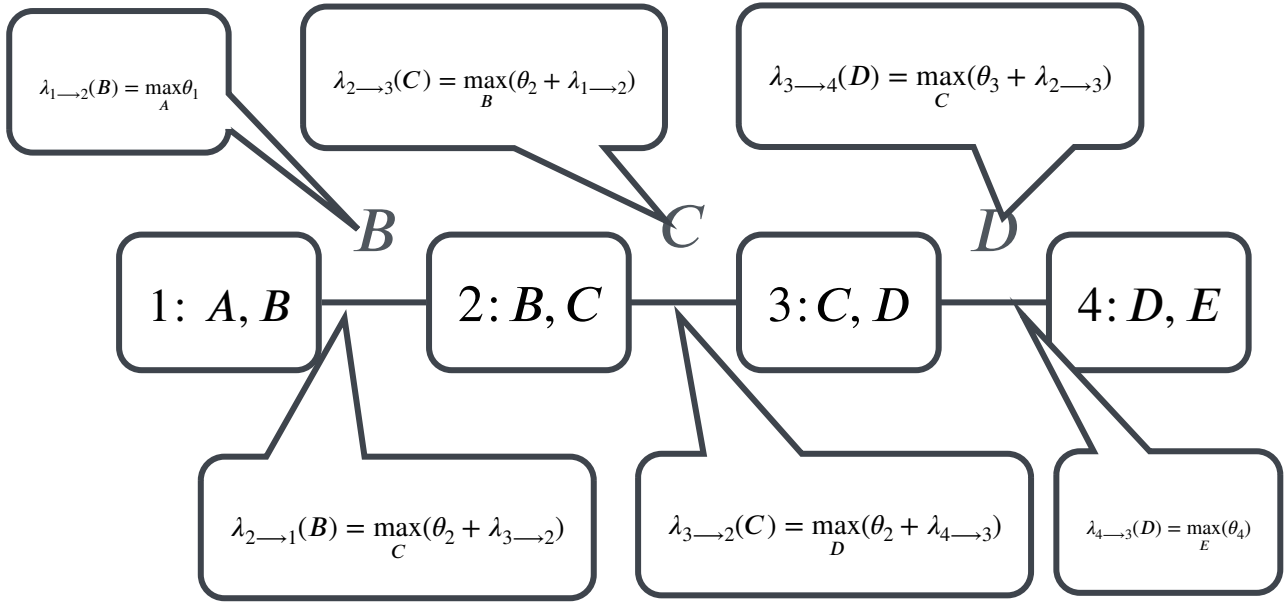


Cluster Graph



Max-Sum in Cluster Trees

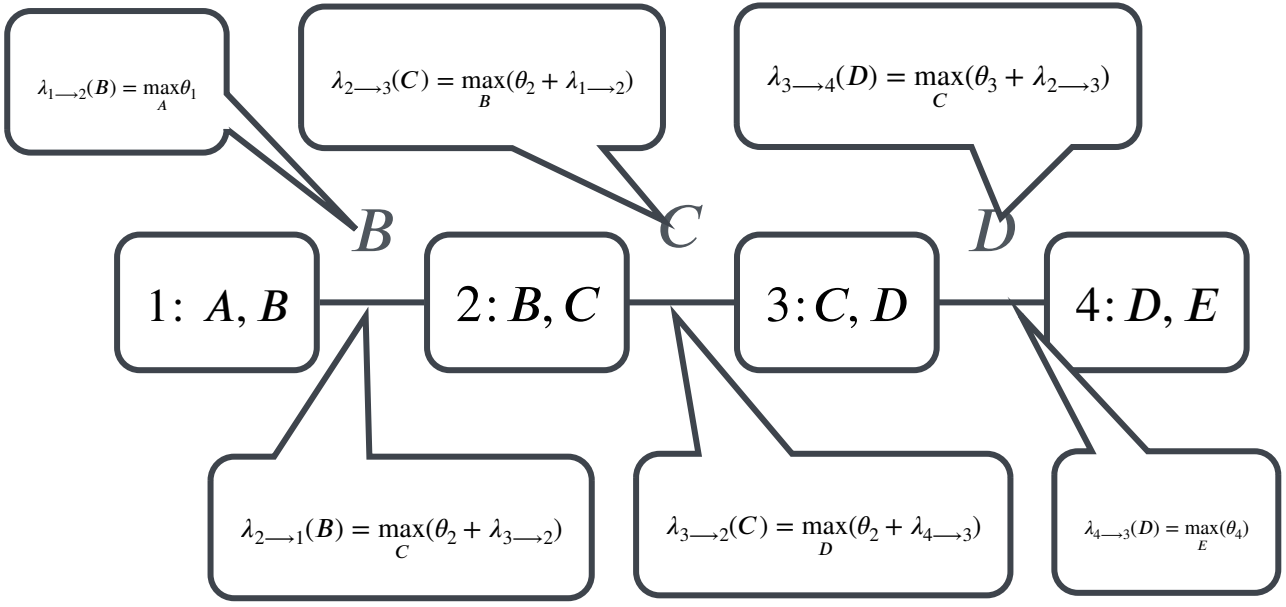
Cluster Graph



$\theta_1(A, B)$ $\theta_2(B, C)$ $\theta_3(C, D)$ $\theta_4(D, E)$

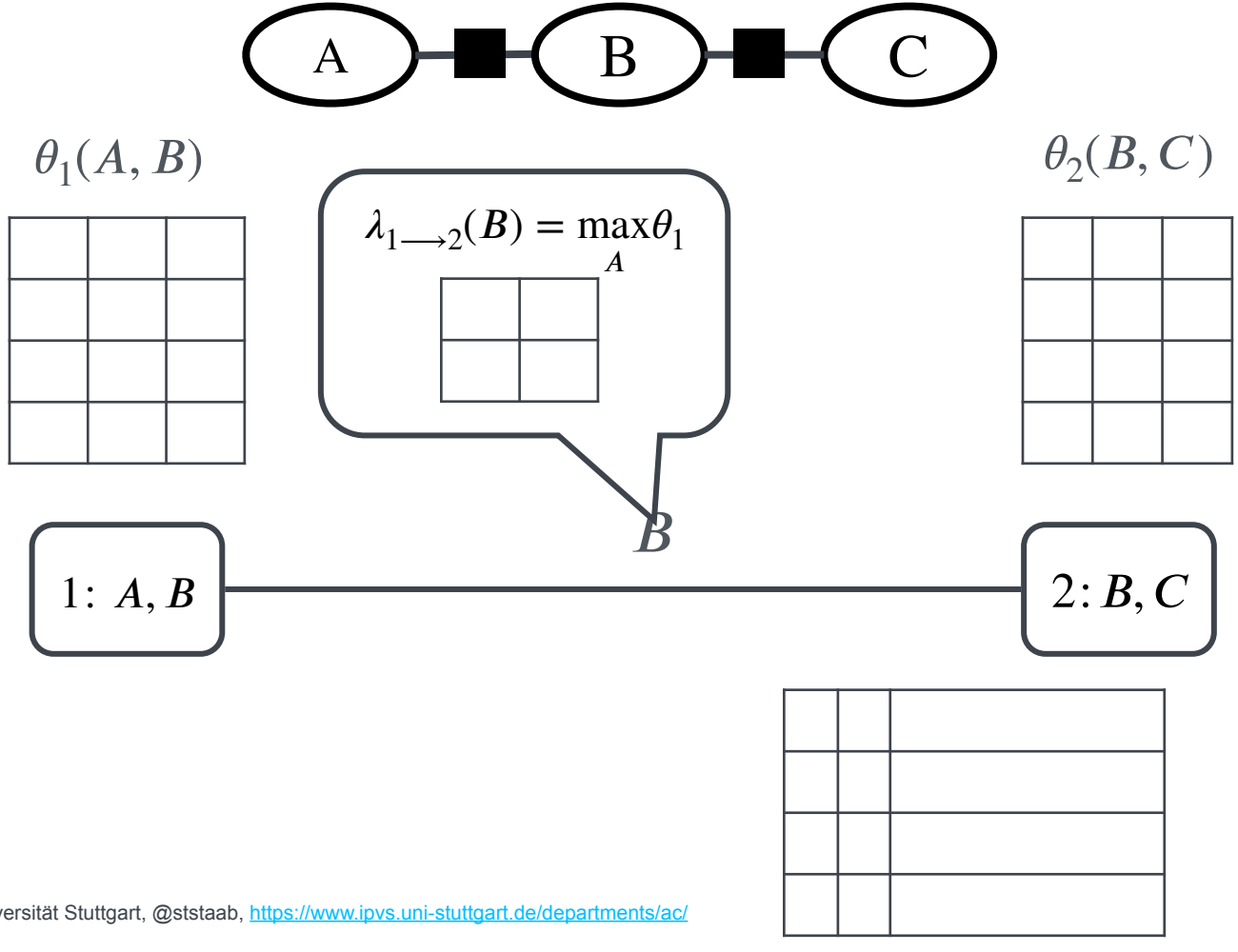
Convergence of Message Passing

Once cluster C_i receives a final message from all neighbors except C_j then $\lambda_{i \longrightarrow j}$ is also final



$\theta_1(A, B)$ $\theta_2(B, C)$ $\theta_3(C, D)$ $\theta_4(D, E)$

Example from before (upward pass)



Max-Sum Beliefs at Convergence

- Beliefs at each Cluster are max-marginals

$$\beta_i(C_i) = \theta_i(C_i) + \sum_k \lambda_{k \longrightarrow i}$$

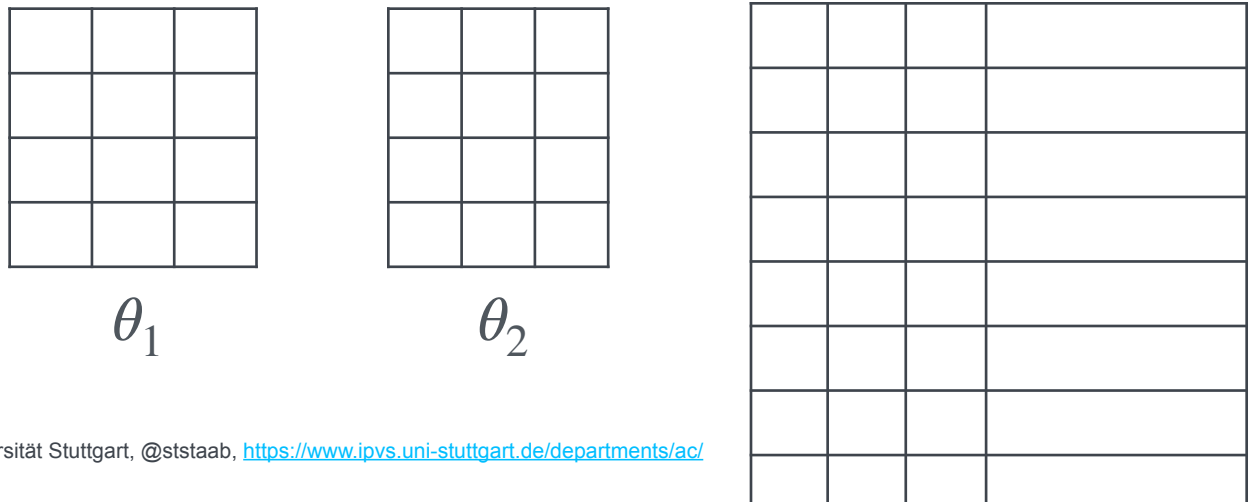
$$\beta_i(C_i) = \max_{W_i} \theta(C_i, W_i), \quad \text{where } W_i = \{X_1, \ldots, X_n\} - C_i$$

- Calibration: clusters agree on shared variables

$$\max_{C_i - S_{i,j}} \beta_i(C_i) = \max_{C_j - S_{i,j}} \beta_j(C_j)$$

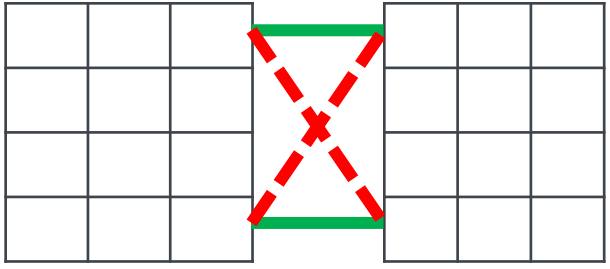
From Max to Argmax: Decoding a MAP Assignment

- Easy if MAP assignment is unique
  - Single maximizing assignment at each cluster
  - whose value is the θ value of the MAP assignment
  - Due to calibration, choices at all clusters must agree



Decoding Non-unique MAP assignment

- XOR Problem

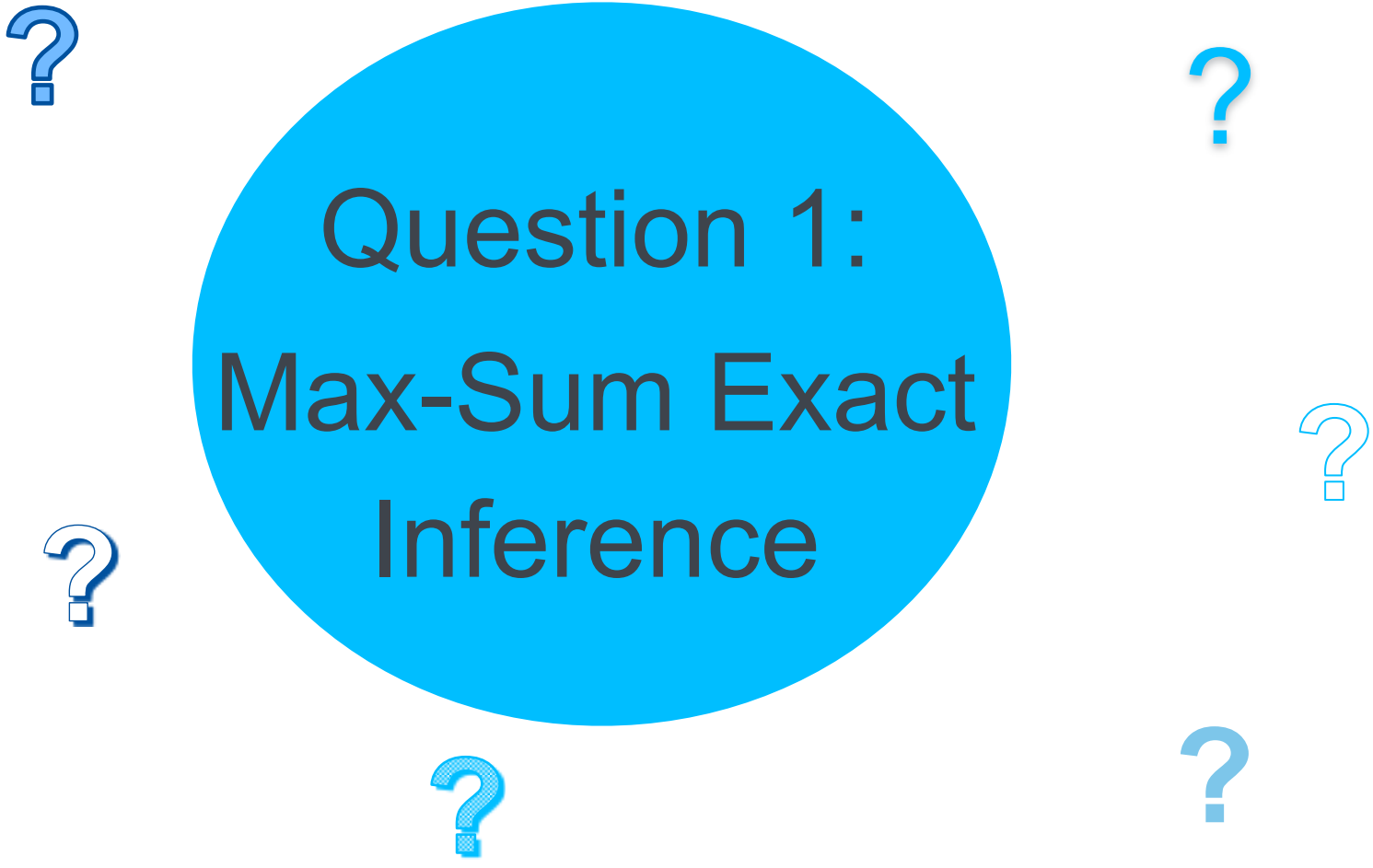


- If MAP assignment is not unique there are multiple choices at some clusters
- Arbitrary tie-breaking may not produce a MAP assignment
- Option 1: Slightly perturb parameters to make MAP unique
- Option 2: Use traceback that builds MAP assignment incrementally, one variable at a time

Summary

- The same cluster tree algorithm used for sum-product can be used for max-sum
- In cluster trees, convergence is achieved after a single up-down pass
- Result is max-marginal at each cluster

There are tractable algorithms for MAP queries for many models
if the models have special properties


Question 1:
Max-Sum Exact
Inference

2 Forward Sampling

Sampling-Based Estimation

- Data set $\mathcal{D} = \{x[1], \ldots x[M]\}$
  sampled iid (independent, identically distributed) from $\boldsymbol{P}(\boldsymbol{\mathcal{X}})$
- Let $p := P(X = 1),\ \mathrm{X} = \{0,1\}$
- Maximum likelihood estimation for p: $p_{\mathcal{D}} = \frac{1}{M} \sum_{m=1}^{M} x[m]$
- Expectation for a function f:

$$\mathbb{E}_{\boldsymbol{P}(\boldsymbol{\mathcal{X}})}[f] \approx \frac{1}{M} \sum_{m=1}^{M} f(x[m])$$

- If f is an indicator function, e.g. $f_0(x) = [x = 0]$ or $f_1(x) = [x = 1]$, we can use this to count occurrences

Sampling from a Discrete Distribution

$$\mathrm{Val}(\mathcal{X}) = \{x^1, \ldots, x^k\}, \qquad P(x^i) = \theta^i$$

Python: Return random float uniformly from range [0.0, 1.0).

`random.random()`

$$\theta^1 + \theta^2 + \theta^3 + \ldots + \theta^k = 1$$

Quality of Sampling-Based Estimation

- Hoeffding Bound:

$$P_{\mathscr{D}}(p_{\mathscr{D}} \notin [p-\epsilon, p+\epsilon]) \leq 2e^{-2M\epsilon^2}$$

- For additive bound ϵ on error with probability $> 1-\delta$:

$$M \geq \frac{\ln\left(\frac{2}{\delta}\right)}{2\epsilon^2}$$

- Chernoff Bound:

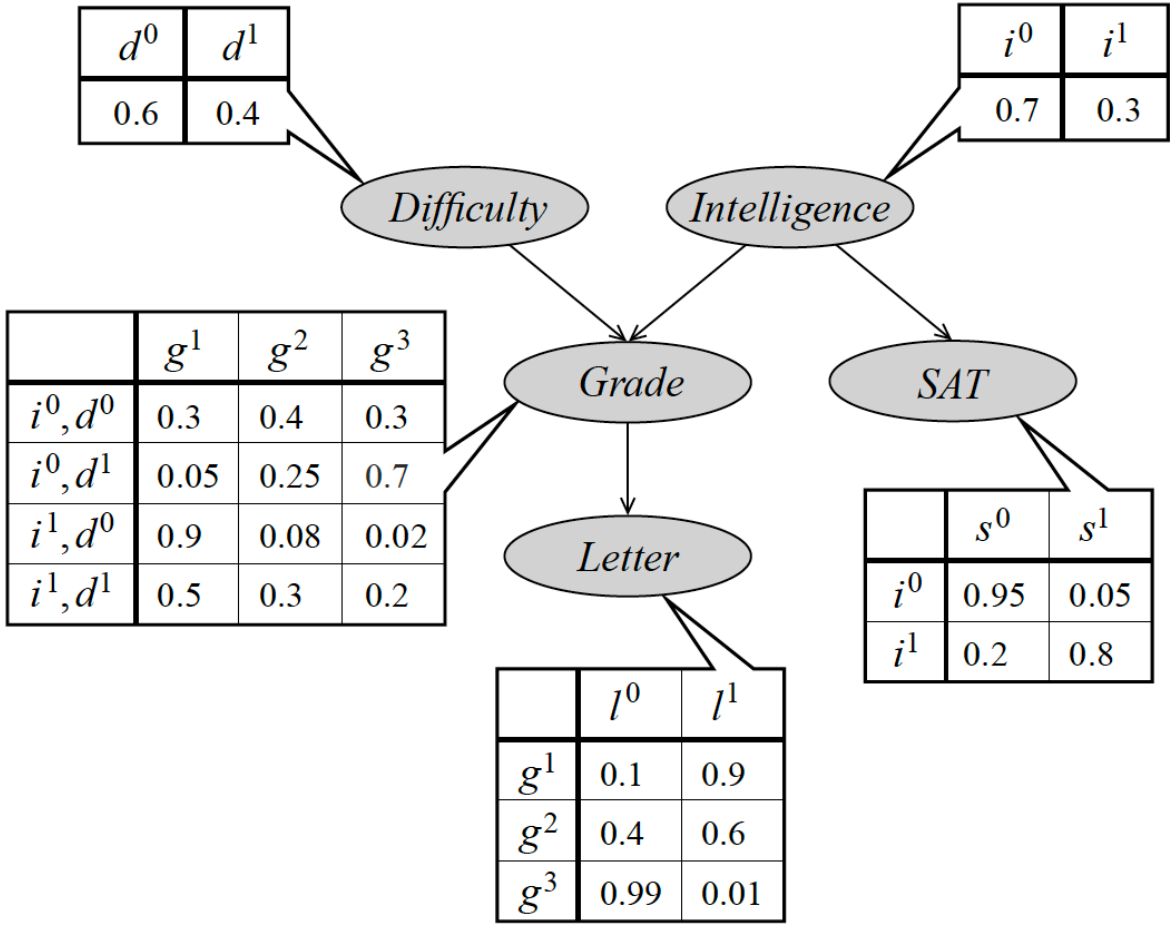
$$P_{\mathscr{D}}(p_{\mathscr{D}} \notin [p(1-\epsilon), p(1+\epsilon)]) \leq 2e^{-Mp\epsilon^2/3}$$

- For multiplicative bound ϵ on error with probability $> 1-\delta$:

$$M \geq 3\frac{\ln\left(\frac{2}{\delta}\right)}{p\epsilon^2}$$

In other words: sampling is problematic when p is small

Forward sampling from a Bayesian Network



d^0, i^1, g^1, s^0, l^1

d^1, i^1, g^2, s^1, l^0

…

Sample individual variables starting with parents before children

Querying by Forward Sampling

- Goal: Estimate $P(\boldsymbol{Y} = \boldsymbol{y})$
  1. Generate M samples from the Bayesian network
  2. Compute fraction where $\boldsymbol{Y} = \boldsymbol{y}$

- For additive bound ϵ on error with probability $> 1 - \delta$: $M \geq \frac{\ln\left(\frac{2}{\delta}\right)}{2\epsilon^2}$

- For multiplicative bound ϵ on error with probability $> 1 - \delta$: $M \geq 3\frac{\ln\left(\frac{2}{\delta}\right)}{p\epsilon^2}$

Querying with Evidence by Rejection Sampling

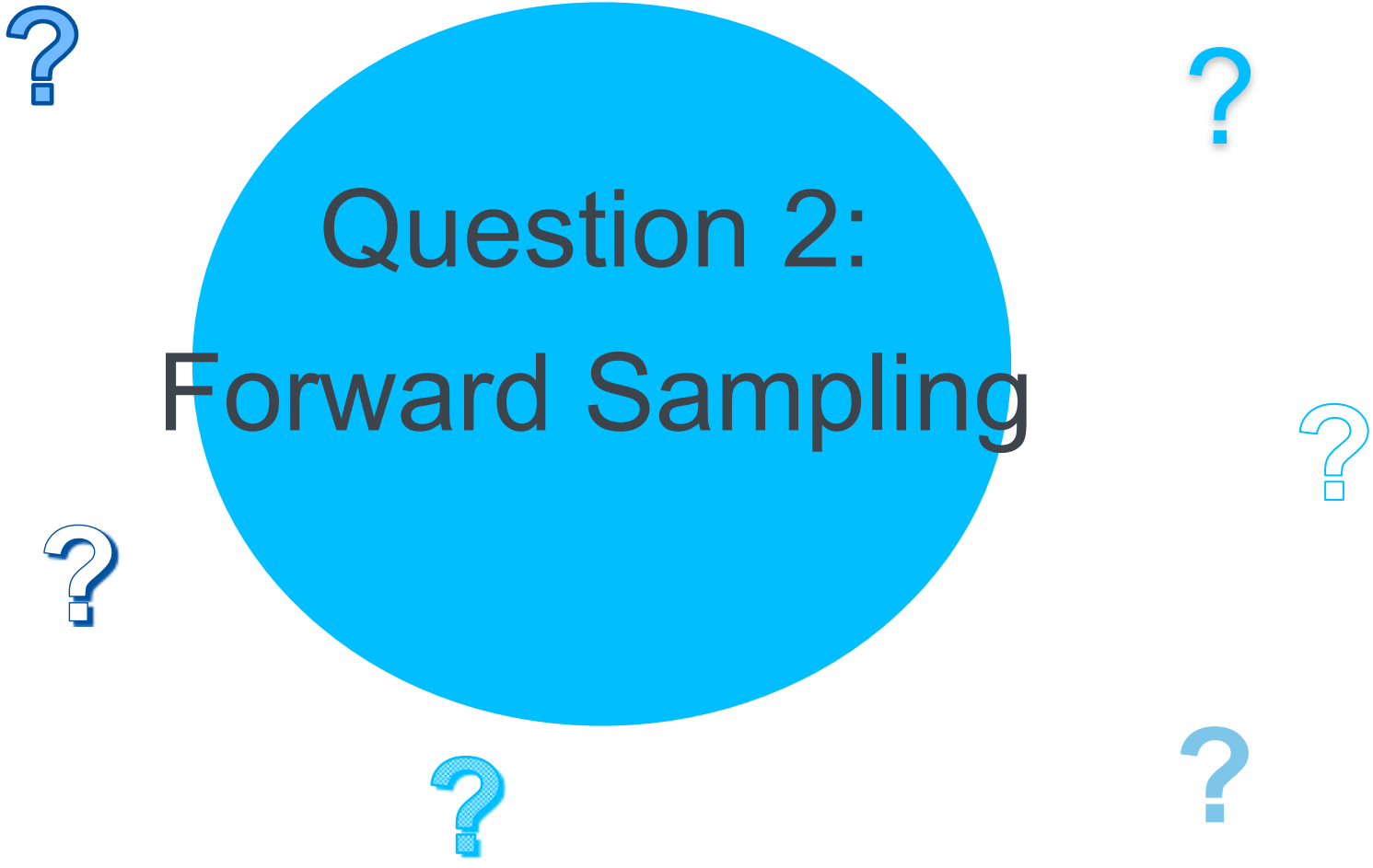
- Goal: Estimate $P(Y = y \mid E = e)$
  1. Generate samples from the Bayesian network
  2. Reject all samples where $E \neq e$
  3. Compute fraction of remaining samples where $Y = y$
- Expected fraction of samples kept: $P(e)$
- Curse of dimensionality:

Rejection sampling methods suffer from the ***curse of dimensionality***, where the probability of rejection increases exponentially as a function of the number of dimensions $|E|$.

Summary

- Generating samples from a Bayesian Network is easy
- (ϵ, δ)-bounds exist, but usefulness is limited:
  - additive bounds: useless for low probability events
  - multiplicative bounds: # of samples grows as $\frac{1}{P(y)}$
- With evidence, # of required samples grows exponentially with # of observed variables

Forward sampling generally infeasible for Markov Networks.


Question 2:
Forward Sampling

3 Markov Chain

Markov Chain Monte Carlo (MCMC)

Markov chain Monte Carlo (MCMC) methods are a class of algorithms for ***sampling from a probability distribution***.

They are ***based on a Markov chain*** that has the desired distribution of its **equilibrium** distribution.

The state of the chain after a number of steps is then used for sampling from the desired distribution.

MCMC algorithms are generally used for sampling from multi-dimensional distributions, especially when the ***number of dimensions is high***.

Variations:

- Gibbs sampling
- Metropolis Hastings

Markov Chain Monte Carlo (MCMC)

Markov chain Monte Carlo (MCMC) methods are a class of algorithms for ***sampling from a probability distribution***.

They are ***based on a Markov chain*** that has the desired distribution of its **equilibrium** distribution.

MCMC algorithms are generally used for sampling from multi-dimensional distributions, especially when the ***number of dimensions is high***.

Our roadmap today:

1. What is a Markov chain?
2. How do we sample from a Markov chain?
3. How to use MCMC for sampling from target distribution 1
4. How to use MCMC for sampling from target distribution 2

Unit 3 Markov Chain

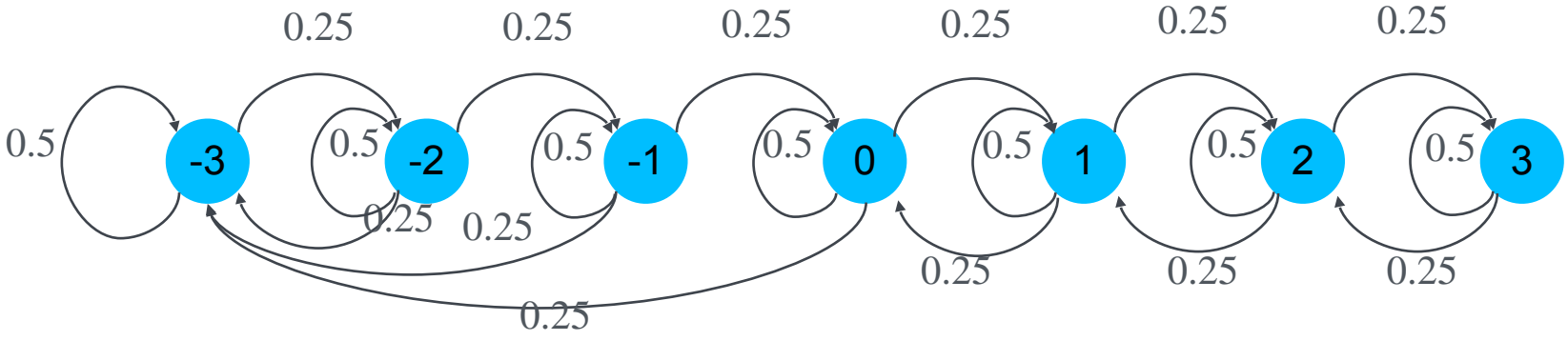
Unit 4 MCMC

Unit 5 Gibbs Samplin

Unit 6 Metropolis Hastings

Markov Chain

Example

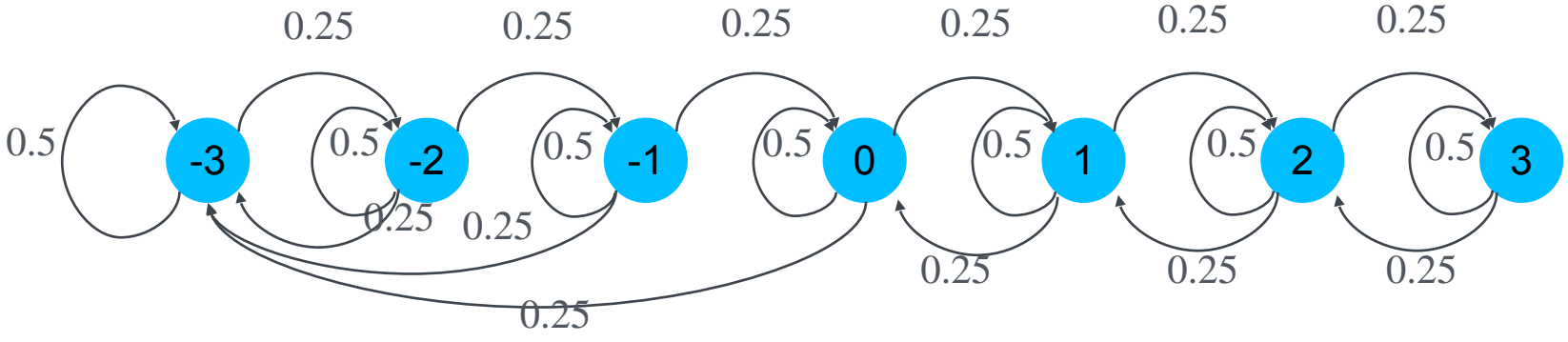


Definition: A Markov chain defines a probabilistic transition model $T(s \rightarrow s')$ over states S:

- For all $s \in S$: $\sum_{s'} T(s \rightarrow s') = 1$

Temporal Dynamics of Random Variables $X^{(t)}$

$$P^{(t+1)}\left(X^{(t+1)}=s'\right)=\sum_{s\in S}P^{(t)}\left(X^{(t)}=s\right)T\left(s\rightarrow s'\right)$$



Stationary Process

General definition: A stochastic process $\left\{ X^{(t)} : t \geq 0 \right\}$ is stationary if for any time points $i_1, \ldots, i_n$ and any $m \geq 0$, the joint distribution of $(X^{(i_1)}, \ldots, X^{(i_n)})$ is the same as the joint distribution of the time-shifted $(X^{(m + i_1)}, \ldots, X^{(m + i_n)})$.

Implies: If a process $\left\{ X^{(t)} : t \geq 0 \right\}$ is stationary, the distributions $X^{(t)}$ are the same for all t.

Stationary Distribution π

$$P^{(t)}(s') \approx P^{(t+1)}(s') = \sum_{s \in S} P^{(t)}\left(X^{(t)} = s\right) T(s \to s')$$

$$\pi(s') = \sum_{s \in S} \pi(s) T(s \to s')$$

- Matrix $T \in \mathbb{R}^{|S| \times |S|}$ is a stochastic matrix:

$$\forall s \in S: \quad \sum_{s' \in S} T[s \to s'] = 1$$

- $\pi(s)$ is constant for all $s \in S$,

this leads to the following **eigenvalue problem**:

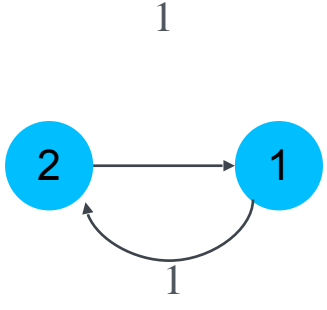
$$\pi^\intercal \lambda = \pi^\intercal T, \text{ with } \lambda = 1$$

Not all eigenvalue problems have real-valued solutions!
How can we ensure this problem does?

Regular Markov Chain

Definition: A Markov chain is regular if there exists k such that, for every s, s' the probability of getting from s to s' in exactly k steps is larger than 0.

Counterexample:



Regular Markov Chain

Definition: A Markov chain is regular if there exists k such that, for every s, s' the probability of getting from s to s' in exactly k steps is larger than 0.

Theorem: A regular Markov chain converges to a unique stationary distribution regardless of start state.

Making a Markov Chain regular

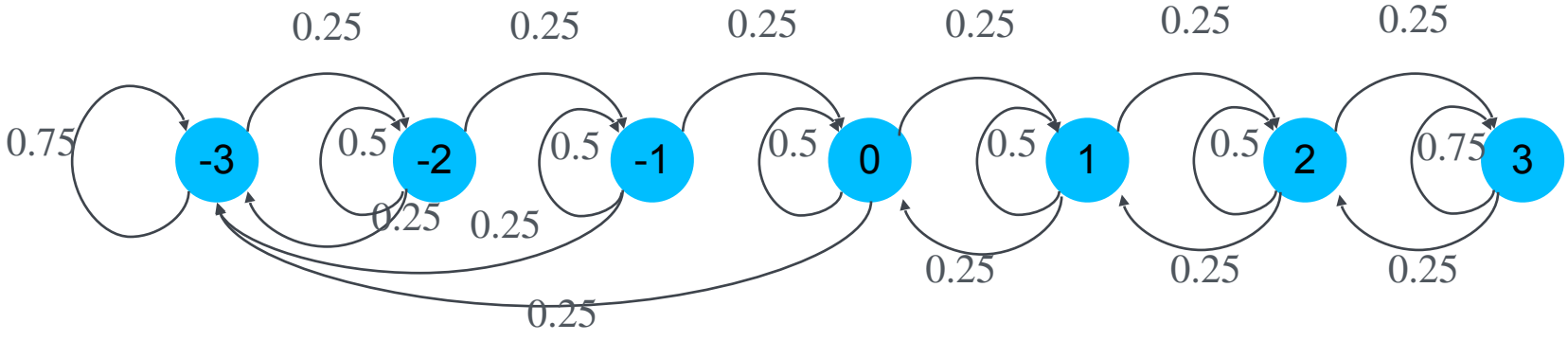
If one can transition from every state in a Markov Chain to every other state (and it is not regular yet),
you can make it regular by adding self transitions.

$$T^{\text{new}} = \frac{1}{2}(T + I)$$


Question 3:
Markov Chain

4 Markov Chain Monte Carlo

Sampling from a Regular Markov Chain



- Very simple idea:
  1. Start in a randomly chosen state
  2. Repeat until probability distribution is stationary
     - Randomly transition into next state s' with probability $T[s \longrightarrow s']$
  3. Repeat until sample is large enough
     - sample current state
     - Randomly transition into next state s' with probability $T[s \longrightarrow s']$

Using a Markov Chain for a Non-Markov Chain distribution

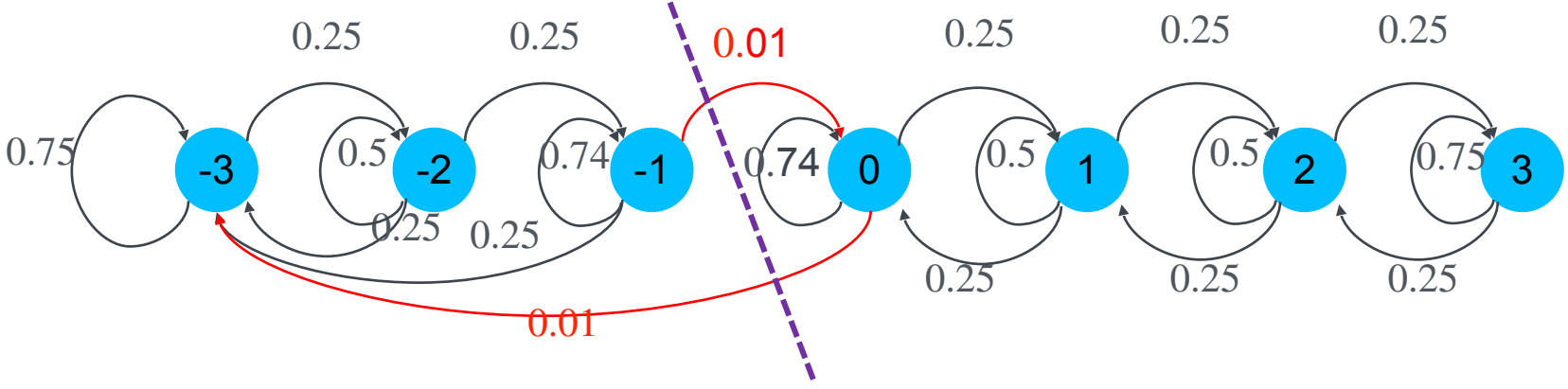
- Goal: compute $P(x \in X)$
  - but: $P(X)$ is too hard to sample from directly
- Approach:
  1. Construct a Markov chain T whose unique stationary distribution is $P(X)$
  2. Sample $x^{(0)}$ from some $P^{(0)}$
  3. For $t = 0, 1, 2, \ldots$
     - Generate $x^{(t+1)}$ from $T(x^{(t)} \longrightarrow x')$

- Considerations
  - Only sample if distribution is close to $P(X)$
  - At early iterations, $P^t(X)$ is usually far from $P(X) = \pi(X)$
  - Thus: collect samples only after the chain has run long enough to ***mix***

Mixing

- Common mixing problem

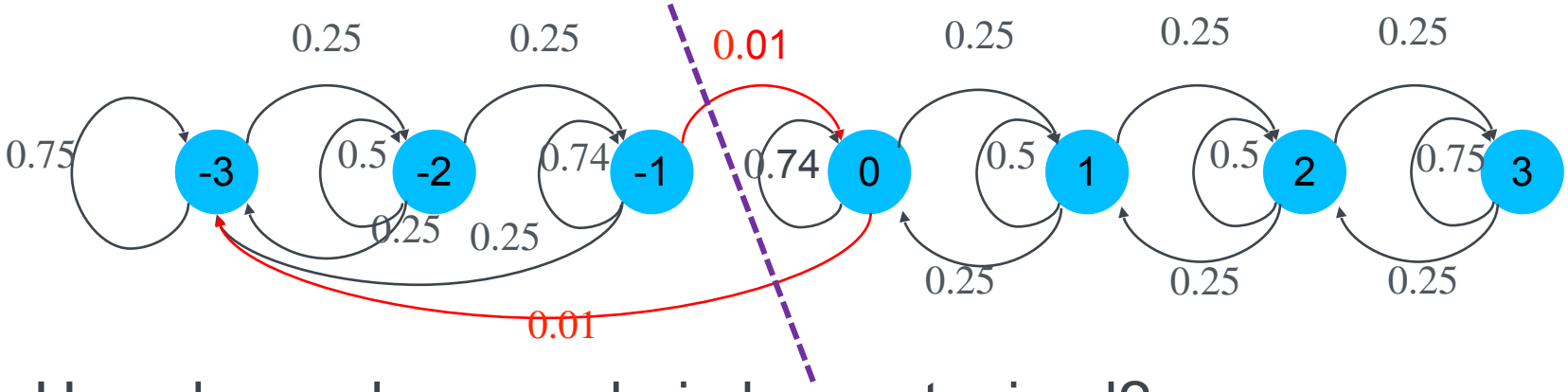
samples **are not** independent!



- How do you know if a chain has mixed or not?
  - In general, you cannot prove a chain has mixed
  - Often, you can show that is has NOT mixed

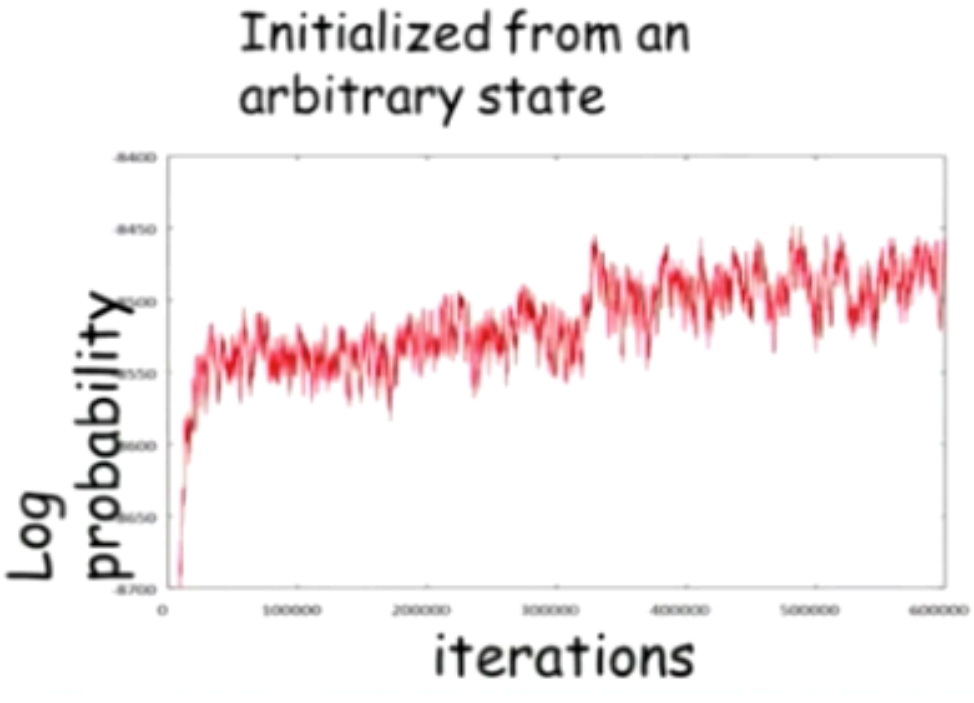
Mixing

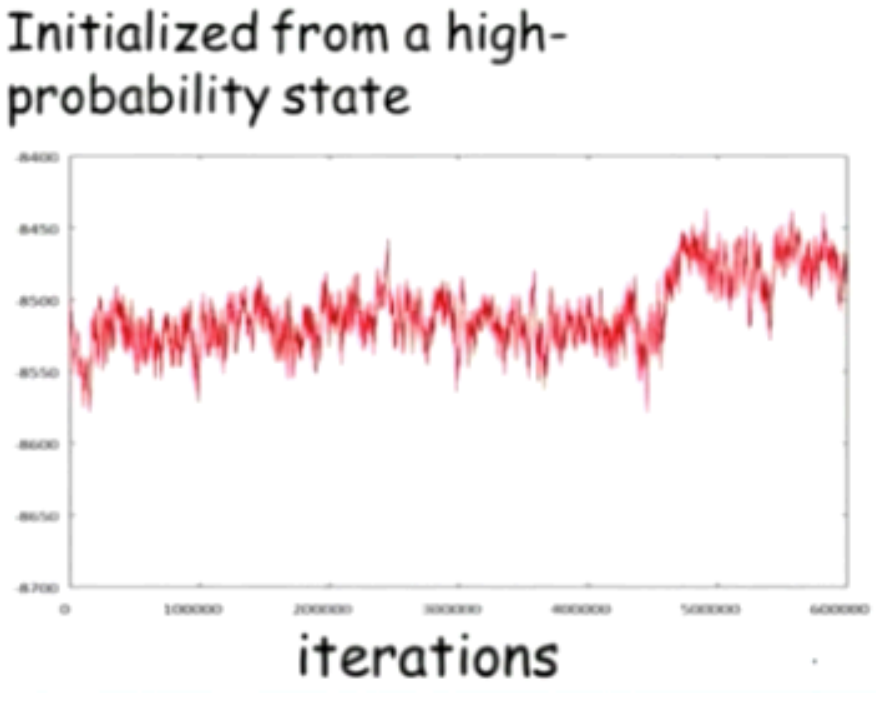
- Common mixing problem



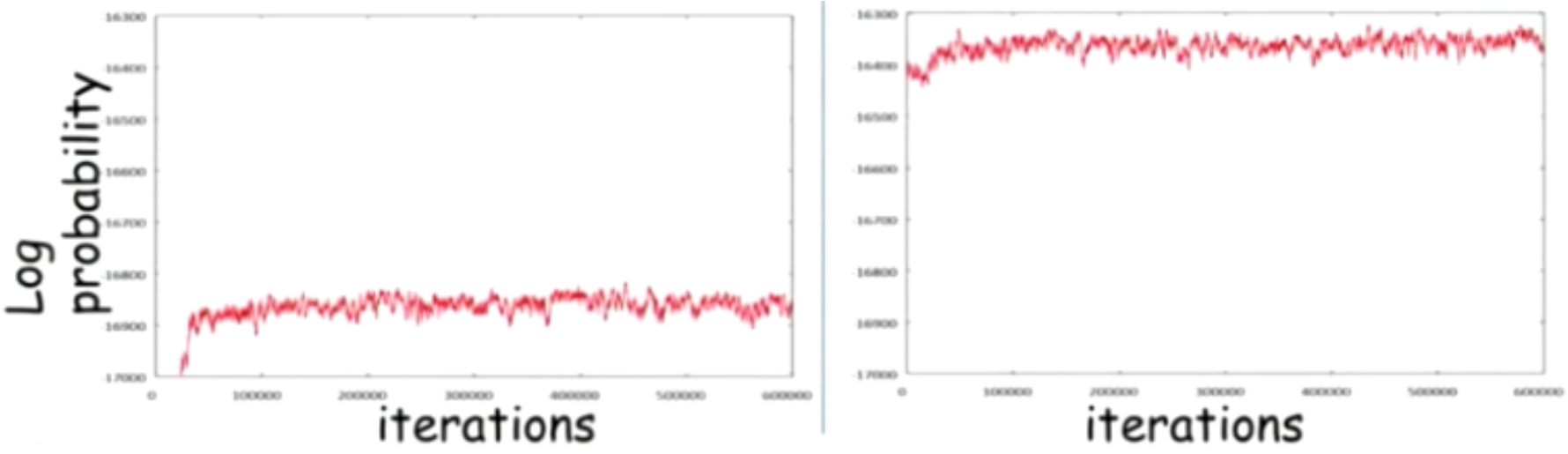
- How do you know a chain has not mixed?
  - Compare chain statistics in different windows *within a single run* of the chain
  - and *across different runs*, initialized differently

Comparing two runs





Comparing two runs



Tools

- All kind of statistics
  - mean
  - variance
- All kind of visualizations
  - scatter plots mapping two series
  - ...

About the Samples

- Once the chain mixes (at $t^{(\text{mix})}$),
  all samples $x^{(t)}$ are from the stationary distribution π
  - Use all $x^{(t)}$ for $t > t^{(\text{mix})}$
- But: nearby samples are correlated (***not iid*** !)
  - do not overestimate quality of sample, just based on size M
- When a chain mixes faster,
  the samples are less correlated and more useful

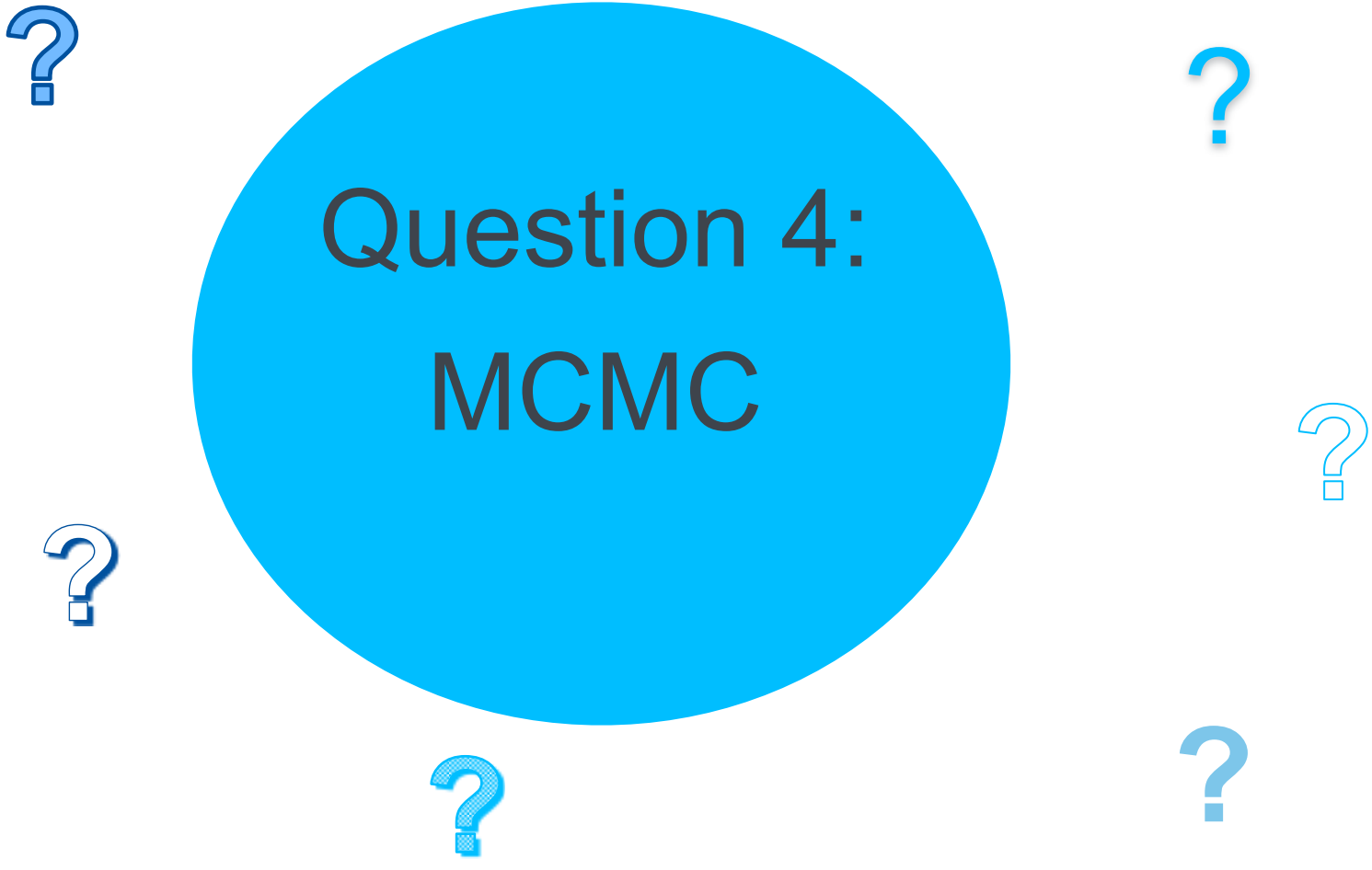
MCMC (refined)

1. Iterate over runs $r = 1 \ldots R$
   - Sample $x^{(r,0)}$ from $P^{(0)}$
2. Repeat until mixing
   - For $r = 1 \ldots R$
     - Generate $x^{(r,t+1)}$ from $T(x^{(r,t)} \rightarrow x')$
   - Compare window statistics in different chains to determine mixing
   - $t := t + 1$
3. Repeat until sufficient samples
   - $D := \varnothing$
   - For $r = 1 \ldots R$
     - Generate $x^{(r,t+1)}$ from $T(x^{(r,t)} \rightarrow$
     - $D := D \cup \left\{ x^{(r,t+1)} \right\}$
   - $t := t + 1$
4. $D = \{ x[1] \ldots x[M] \}$

$$\mathbb{E}_{\boldsymbol{P}(\mathcal{X})}[f] \approx \frac{1}{M} \sum_{m=1}^{M} f(x[m])$$

Issues with MCMC

- Many parameters
  - how many samples to collect?
    - how correlated are they?
  - how long until mixing?
  - how to compute statistics that determines mixing?
    - window size
    - which test statistics ?
    - how close should test statistics be?


Question 4:
MCMC

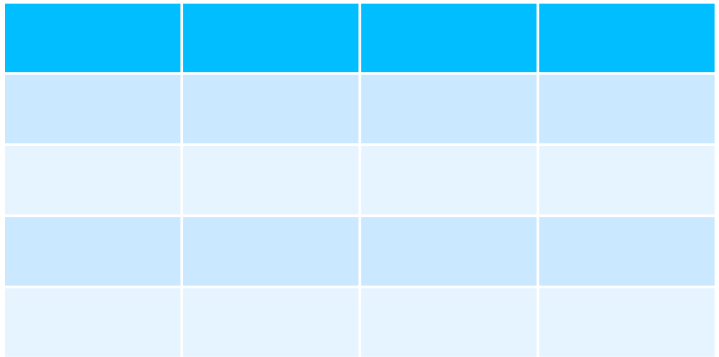
5 Gibbs Sampling

How to design a markov chain? Answer 1: Gibbs chain

- Target distribution $\boldsymbol{P_{\Phi}}(\mathcal{X}) = \boldsymbol{P_{\Phi}}(X_1, \ldots, X_n)$
- Markov chain state space:
  each complete assignment $\boldsymbol{x}$ to $\mathcal{X}$ is a state of the Markov chain
- Let $\boldsymbol{x_{-i}}$ denote $(x_1 \ldots x_{i-1}\ x_{i+1}\ \ldots x_n)$

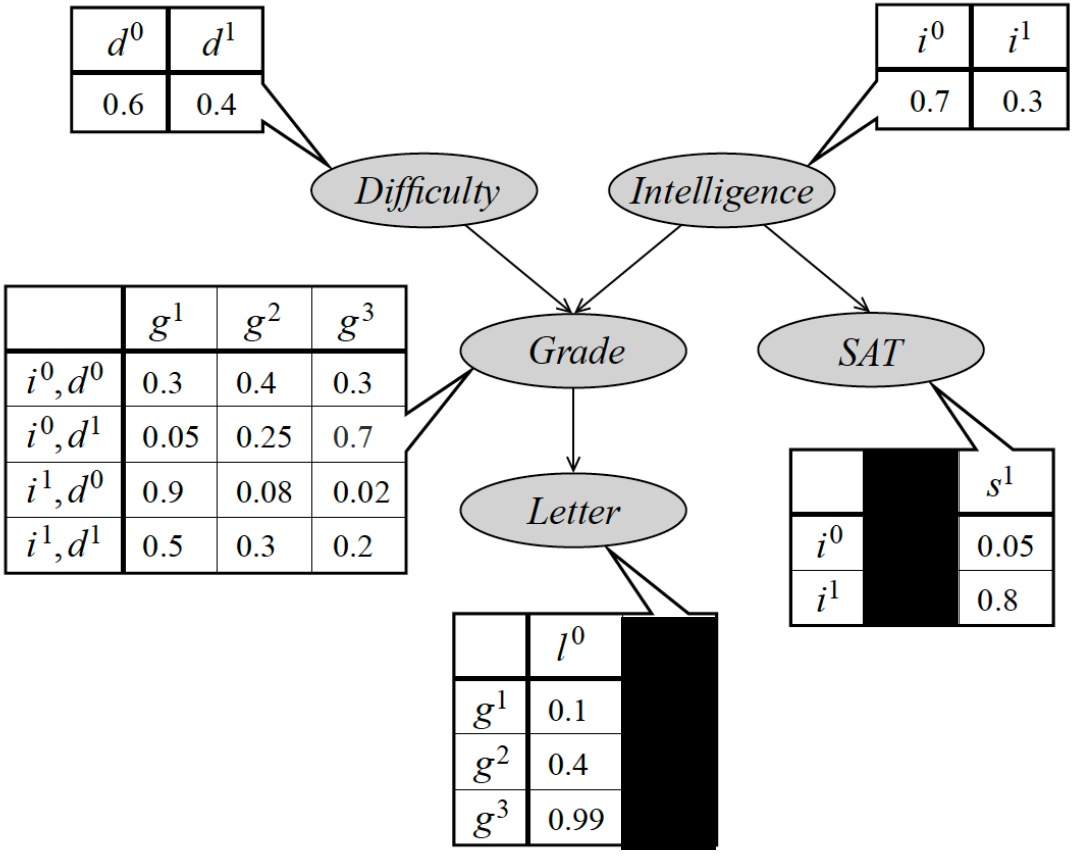
No explicit definition of T

- Random transition step defined given state $\boldsymbol{x}$
  - For $i = 1 \ldots n$
    - Sample $x_i' \sim \boldsymbol{P_{\Phi}}(X_i \mid \boldsymbol{x_{-i}})$
    - Set $\boldsymbol{x} = \boldsymbol{x_{-i}} \circ x_i'$
  - Return $\boldsymbol{x}' = \boldsymbol{x}$



Gibbs sampling with evidence

Sample $P_\Phi(D, I, G \mid s^1, l^0)$



Start assignment, state

$s^0 = (d^0, i^0, g^0, s^1, l^0)$ constant

$d \sim P(D \mid i^0, g^0, s^1, l^0)$

$i \sim \ldots$

$g \sim \ldots$

State $s^1 = (d, i, g, s^1, l^0)$

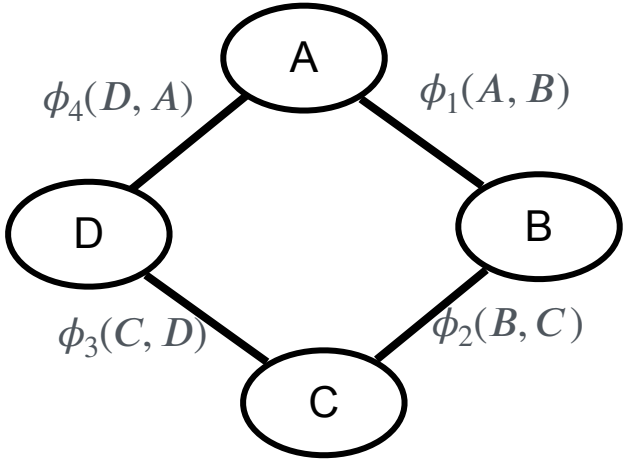
...

Computational cost

$$P_{\Phi}(A \mid b, c, d) = \frac{\tilde{P}_{\Phi}(A, b, c, d)}{\sum_{A'} \tilde{P}_{\Phi}(A', b, c, d)} = = \frac{\phi_1(A, b)\phi_2(b, c)\phi_3(c, d)\phi_4(d, A)}{\sum_{A'} \phi_1(A', b)\phi_2(b, c)\phi_3(c, d)\phi_4(d, A')} = = \frac{\phi_1(A, b)\phi_4(d, A)}{\sum_{A'} \phi_1(A', b)\ \phi_4(d, A')}$$

- For $i = 1 \ldots n$
  - Sample $x_i' \sim \boldsymbol{P_{\Phi}}(X_i \mid \boldsymbol{x_{-i}})$

$$\boldsymbol{P_{\Phi}}(X_i \mid \boldsymbol{x_{-i}}) = \frac{\boldsymbol{P_{\Phi}}(X_i, \boldsymbol{x_{-i}})}{P_{\Phi}(\boldsymbol{x_{-i}})} = = \frac{\frac{1}{Z}\tilde{\boldsymbol{P}}_{\boldsymbol{\Phi}}(X_i, \boldsymbol{x_{-i}})}{\frac{1}{Z}\tilde{P}_{\Phi}(\boldsymbol{x_{-i}})} = \frac{\tilde{\boldsymbol{P}}_{\boldsymbol{\Phi}}(X_i, \boldsymbol{x_{-i}})}{\tilde{P}_{\Phi}(\boldsymbol{x_{-i}})}$$

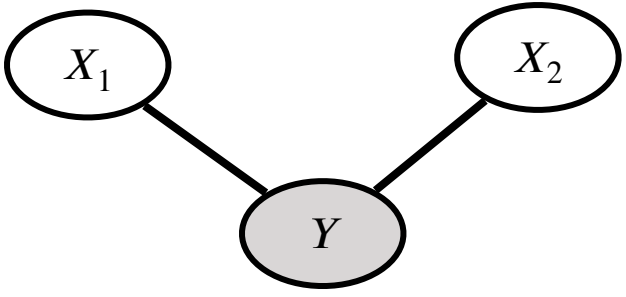


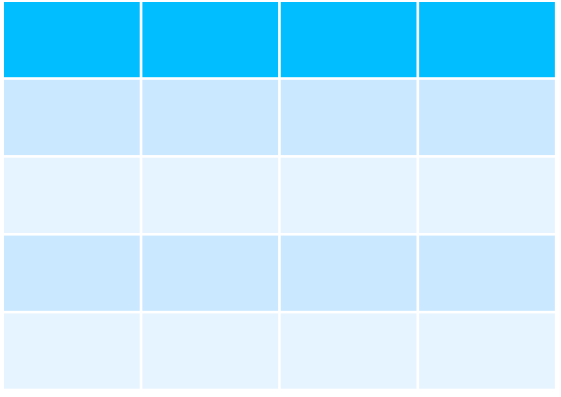
Only have to consider factors related to X_i

Not all Gibbs chains are regular

Mixing can remain slow

- Counterexample to regularity: XOR



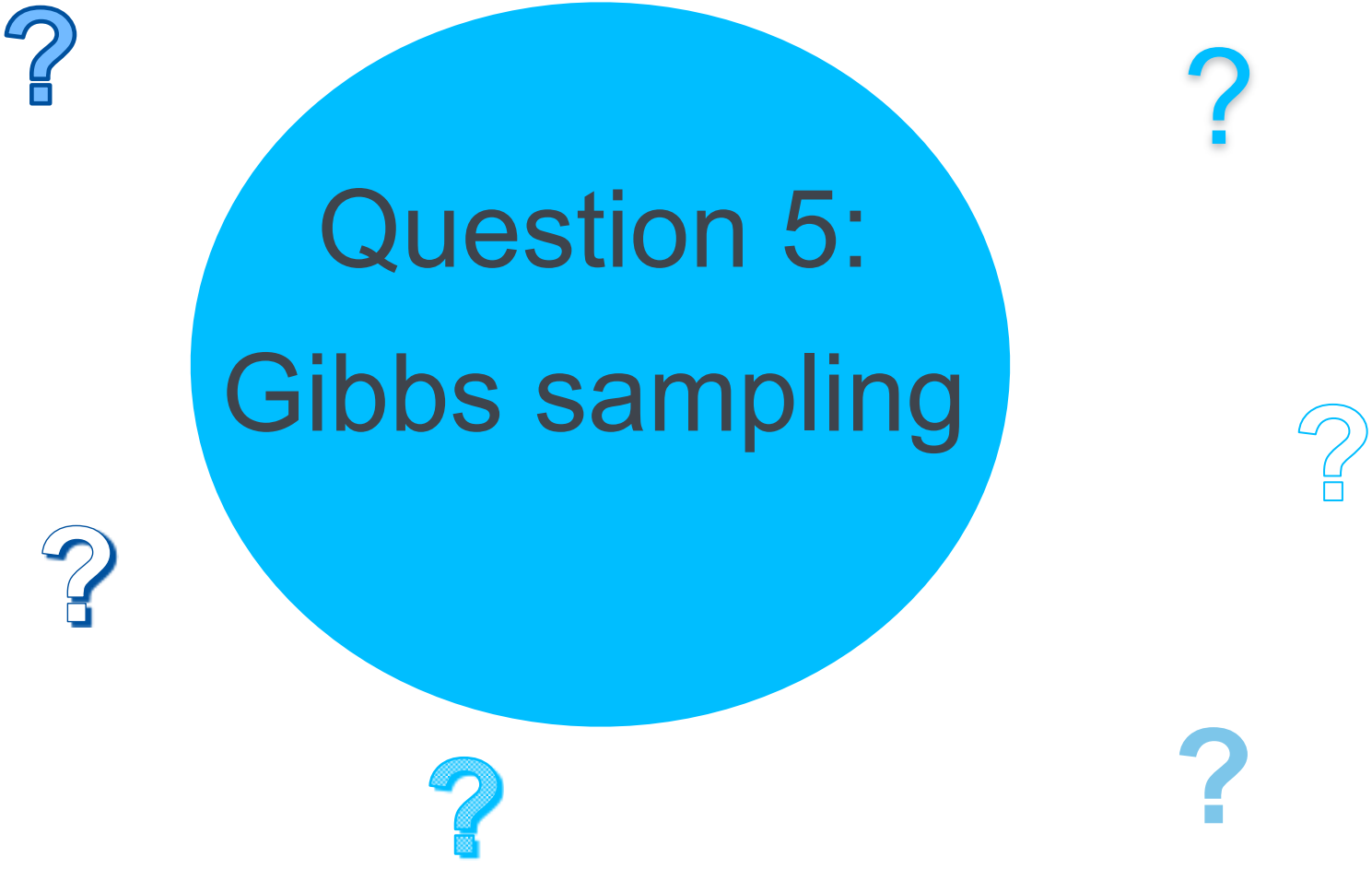


- Compute $P(X_1, X_2 \mid Y = 1)$

If all factors are positive, Gibbs chain is regular
(connected, aperiodic)

Conclusion of Gibbs sampling

- Handles high-dimensional problem
- Handles continuous and discrete random variables
- Sampling from conditionals can be hard!
  - Possible way out: „Metropolis within Gibbs“
- If random variables are strongly correlated, the Markov chain converges slowly and has highly auto-correlated samples


Question 5:
Gibbs sampling

6 Metropolis Hastings Algorithm

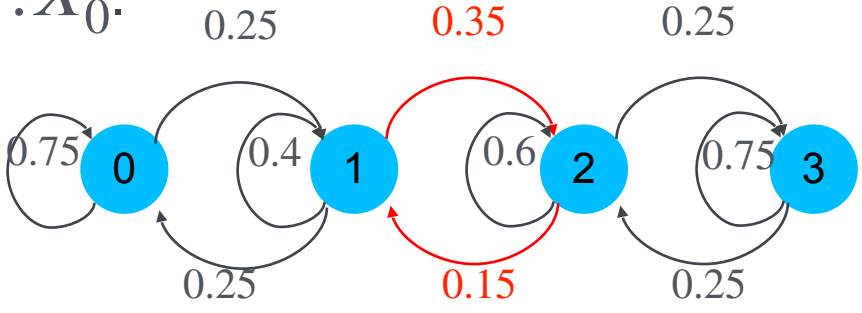
How to apply MCMC?

- Gibbs:
  - states correspond to complete assignments
  - sample from posteriors $P(X_i \mid \boldsymbol{x}_{-i})$
- Metropolis Hastings:
  - sample proportional to $\boldsymbol{P_\Phi}(\mathcal{X})$
    - unnormalized $\tilde{\boldsymbol{P}}_{\boldsymbol{\Phi}}(\mathcal{X})$!
    - can also be proportional to density for continuous probability distributions

Different strengths and weaknesses wrt mixing.
No single best choice!

Reversible Markov Chain

Definition: A Markov chain is called reversible, if for every n the sequence $X_0,\ X_1, \ldots X_n$ has the same joint probability as the time-reversed sequence $X_n,\ X_{n-1}, \ldots X_0$.



In particular this implies:

Theorem: A Markov chain with probability distribution over states π is reversible if and only if the detailed balance equation holds:

$$\pi(x)T(x \longrightarrow x') = \pi(x')T(x' \longrightarrow x)$$

Reversible Markov Chain

$$\pi(x)T(x \longrightarrow x') = \pi(x')T(x' \longrightarrow x)$$

Theorem: If T is regular, it has a unique stationary distribution.

Theorem: If T is regular and detailed balance holds,
the unique stationary distribution is π.

Proof:

$$\sum_x \pi(x)T(x \longrightarrow x') = \sum_x \pi(x')T(x' \longrightarrow x)$$

$$\sum_x \pi(x)T(x \longrightarrow x') = \pi(x') \sum_x T(x' \longrightarrow x) = \pi(x')$$

Metropolis Hastings Chain

- Proposal distribution $Q(x \to x') \in [0,1]$
  - describes how to explore the space $\mathcal{X}$
- Acceptance probability $A(x \to x')$ - Bernoulli random variable
  - reflects $P_{\Phi}(\mathcal{X})$
  - Turns ergodic Markov chain Q into a reversible Markov chain T
- Targeted Markov chain

$$T(x \to x') = Q(x \to x')A(x \to x'), \text{ if } x \neq x'$$

$$T(x \to x') = Q(x \to x') + \sum_{x' \in X} Q(x \to x')(1 - A(x \to x')), \text{ else}$$

- Meaning: Q proposes transition to next sample and A accepts it or not

Make Metropolis Hastings Chain Reversible

- Proposal distribution $Q(x \to x') \in [0,1]$
- Acceptance probability $A(x \to x') \in \{0,1\}$
- Targeted Markov chain: $T(x \to x') = Q(x \to x')A(x \to x'), \text{ if } x \neq x'$
- Enforce reversibility

$$\pi(x)T(x \longrightarrow x') = \pi(x')T(x' \longrightarrow x)$$

$$\pi(x)Q(x \to x')A(x \to x') = \pi(x')Q(x' \to x)A(x' \to x)$$

$$\frac{A(x \to x')}{A(x' \to x)} = \frac{\pi(x')Q(x' \to x)}{\pi(x)Q(x \to x')} = p < 1, \text{ set } A(x' \to x) = 1$$

$$A(x \to x') = \min\left(1, \frac{\pi(x')Q(x' \to x)}{\pi(x)Q(x \to x')}\right)$$

Deliver $P_{\Phi}(\mathcal{X})$

- Proposal distribution $Q(x \rightarrow x') \in [0,1]$
- Acceptance probability $A(x \rightarrow x') \in \{0,1\}$
- Targeted Markov chain:
  $T(x \rightarrow x') = Q(x \rightarrow x')A(x \rightarrow x'), \text{ if } x \neq x'$
- Enforce reversibility

$$A(x \rightarrow x') = \min\left(1, \frac{\pi(x')Q(x' \rightarrow x)}{\pi(x)Q(x \rightarrow x')}\right)$$

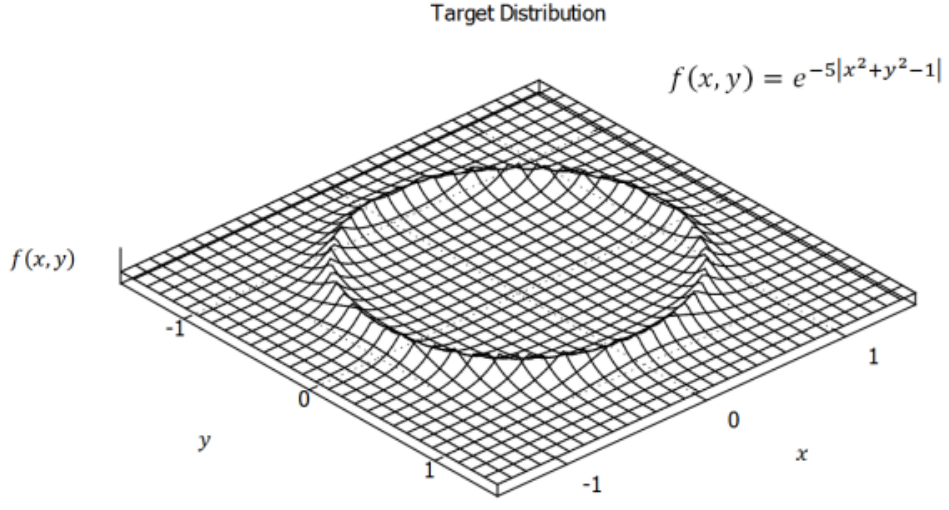
$$\frac{\pi(x')}{\pi(x)} = \frac{P_{\Phi}(x')}{P_{\Phi}(x)}$$

Choice of Q

$$A(x \to x') = \min\left(1, \frac{P_{\Phi}(x')Q(x' \to x)}{P_{\Phi}(x)Q(x \to x')}\right)$$

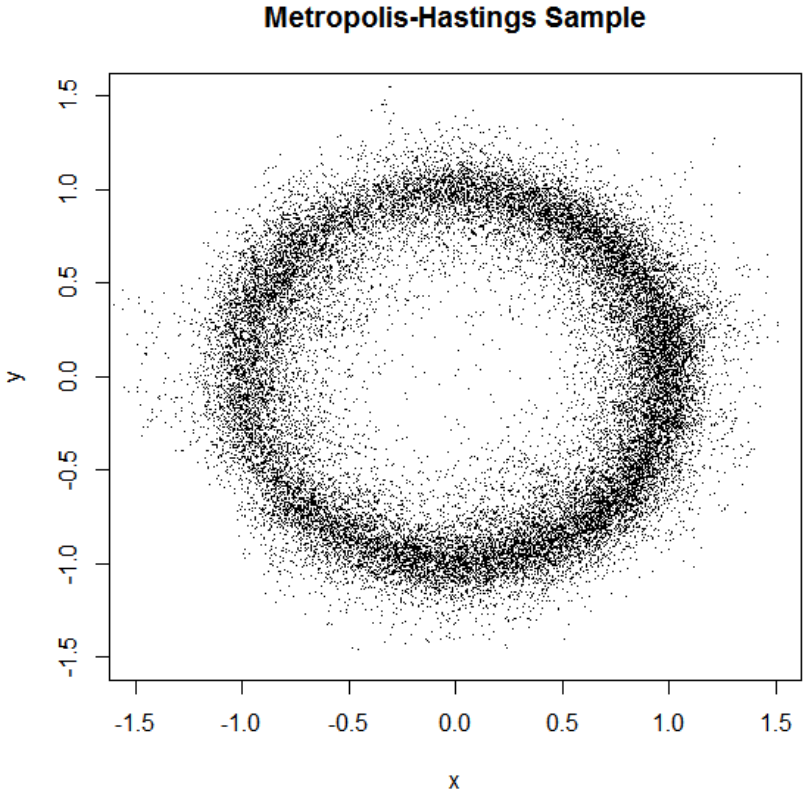
- Q must be reversible:
  - $Q(x' \to x) > 0 \Longrightarrow Q(x \to x') > 0$
- describes how to explore the space $\mathcal{X}$
  - movements should be neither to small nor too large
    - large „jumps“ often end in area of low probability $\Longrightarrow$ high rejection rate
  - often a normal distribution (for continuous distributions $\mathcal{X}$)

Illustration Metropolis-Hastings in Continuous Space



From: https://blog.abhranil.net/2014/02/08/r-code-for-multivariate-random-walk-metropolis-hastings-sampling/

Illustration Metropolis-Hastings



From: https://blog.abhranil.net/2014/02/08/r-code-for-multivariate-random-walk-metropolis-hastings-sampling/

Lessons learned - MH Sampling is susceptible to:

- Start points
  - AlphaGo uses neural network just to predict good start point for sampling
- Proposal distribution
  - Proposal steps should not be too large or too small
    - Too large – many rejections
    - Too small – slow diffusion
  - Tremendous flexibility in designing proposal distributions that explore the space quickly
    - good proposal distribution makes a huge difference
- Mixing time (also called *burn-in* time)
  - Do not sample too early
- Lag (also referred to as sampling intervals)
  - Succeeding states are closely coupled
- Shape of target function
  - Non-differentiable is problematic
  - Low or zero slopes are problematic
- Discrete distributions can be harder to sample from than continuous ones

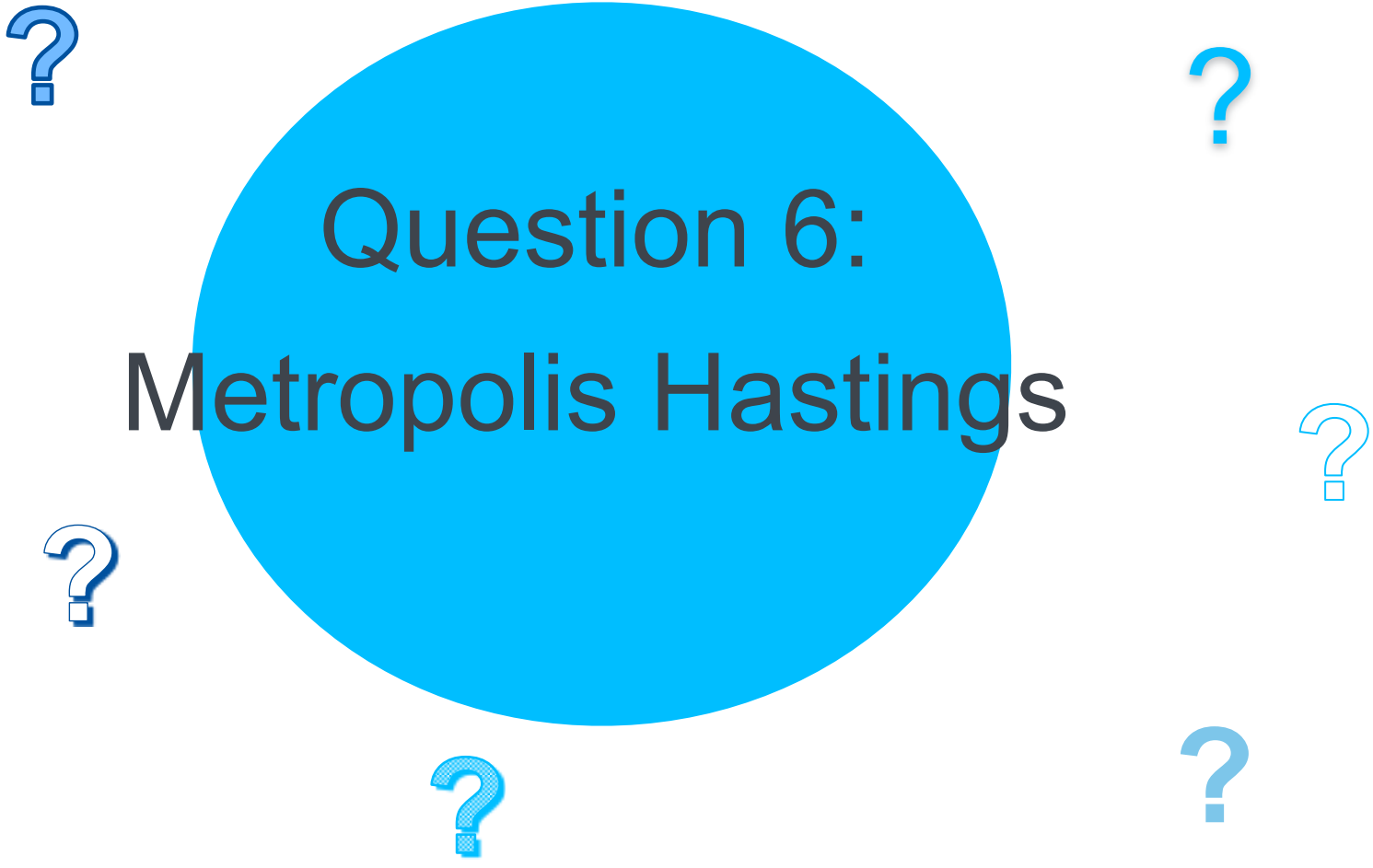
Metropolis Hastings Algorithm

Also used to compute

- expected values
- integrals

First version developed in 1950 in statistical physics

Metropolis Hastings is a general framework for building Markov chains with a particular stationary distribution


Question 6:
Metropolis Hastings

Universität Stuttgart
IPVS

Thank you!

Steffen Staab

E-Mail Steffen.staab@ipvs.uni-stuttgart.de
Telefon +49 (0) 711 685 to be defined
www. ipvs.uni-stuttgart.de/departments/ac/

Universität Stuttgart
Analytic Computing, IPVS
Universitätsstraße 32, 50569 Stuttgart