15-780: Graduate AI

Lecture 16. Inference

Geoff Gordon (this lecture)
Tuomas Sandholm
TAs Sam Ganzfried, Byron Boots

Review

Probability

- *Conditioning*
- *Independence*
- *Bayes Rule*
- *Continuous distributions*

Factor graphs

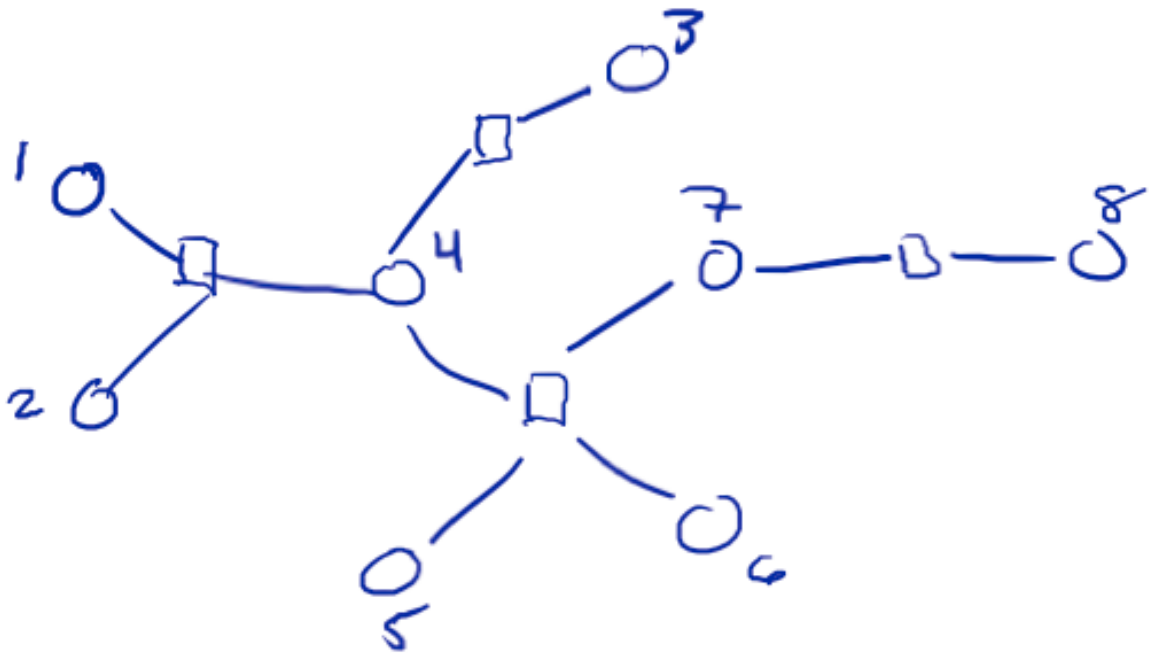
- *Generalization of SAT, ILP to include probability*
- *R.v.s connected by factors (= soft or hard constraints)*
- *P = product of factors / Z*
- *Partition function Z is hard part—makes most tasks NP- or #P-complete*

Inference

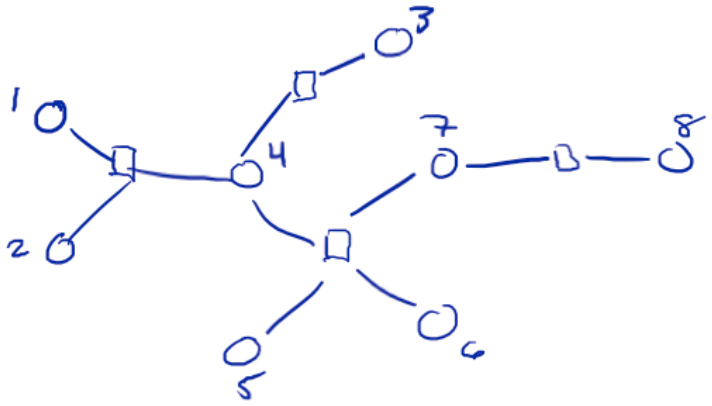
- *Dynamic programming for counting support or for calculating Z*
- *Build probability tables for subsets of r.v.s*
- *Marginalize onto r.v.s shared with neighboring factors, extend domain and multiply into neighboring factor*
- *Allows us to forget exact settings of non-shared r.v.s*

Example

Calculate Z for this graph

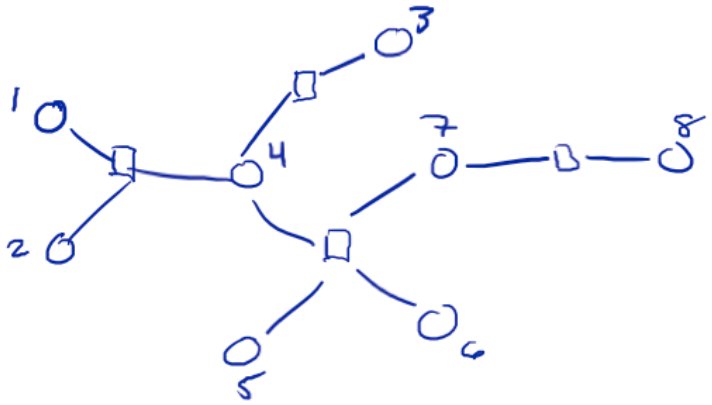


Eliminating partway



- *For Z, want to sum over all* $\mathbf{X}$
- *For marginal* $P(\mathbf{X}_{45})$*, eliminate* $\mathbf{X}_{123678}$
- *Sum out 123, then 876, to get Z* $P(\mathbf{X}_{45})$
- *Sum out 45 to get Z*

Eliminating partway

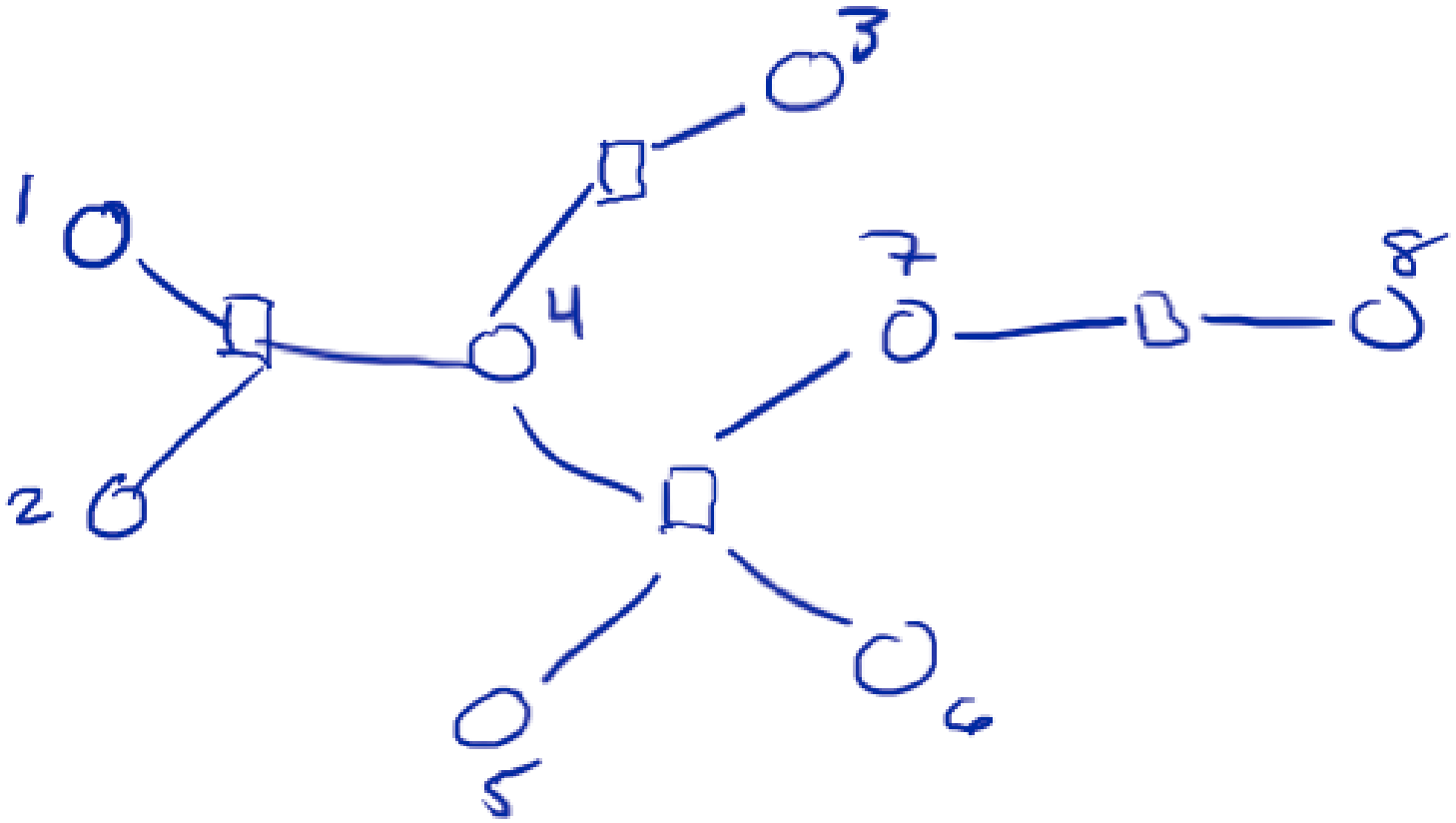


- *Conditional* $P(\mathbf{X}_{56} \mid \mathbf{X}_4) = P(\mathbf{X}_{456}) / P(\mathbf{X}_4)$
- *Sum out 123, then 87, to get Z* $P(\mathbf{X}_{456})$
- *Sum out 56 to get Z* $P(\mathbf{X}_4)$
- *Divide*

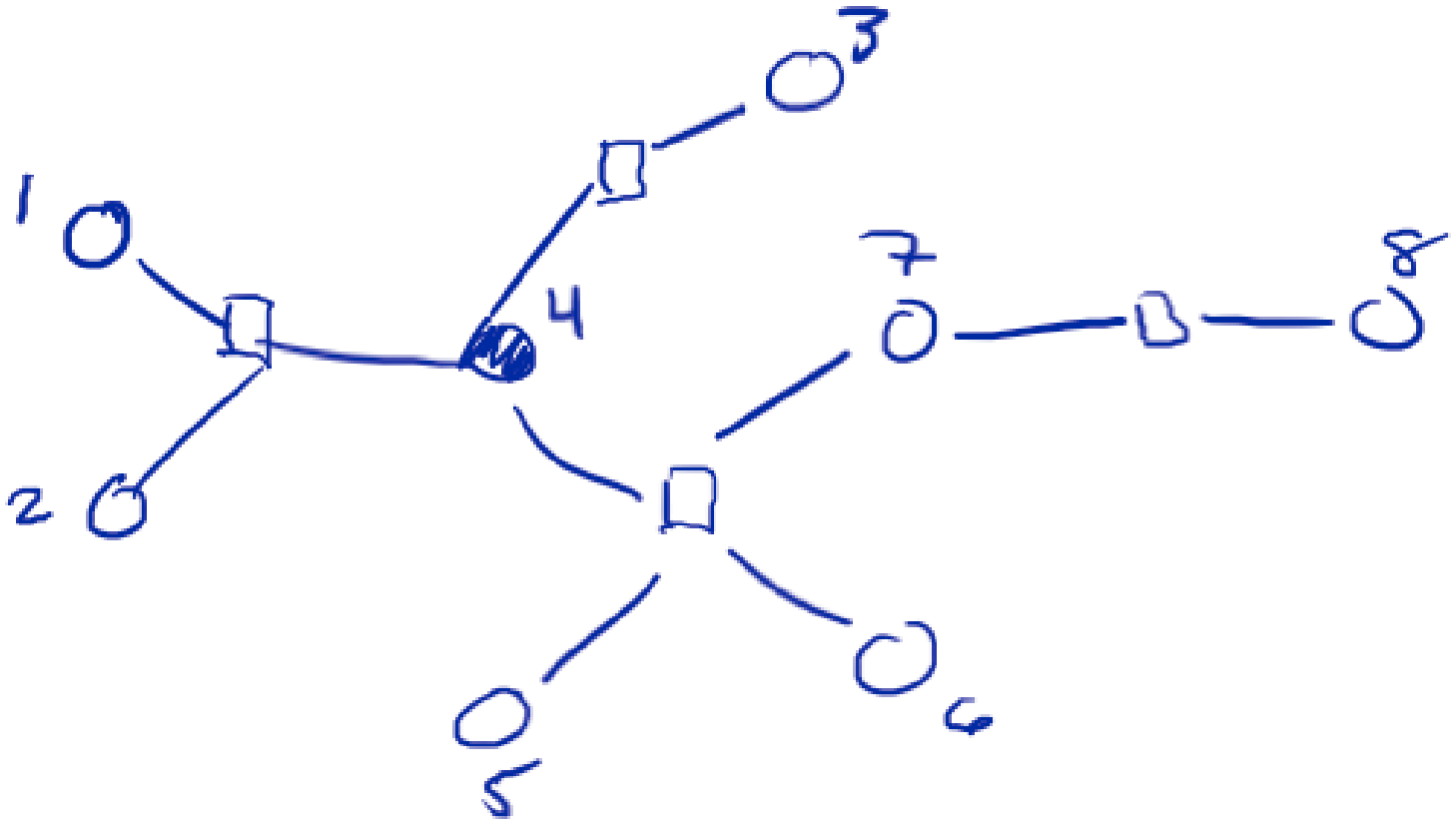
Conditioning on observations

- *We just computed* $P(\boldsymbol{X}_{56} \mid \boldsymbol{X}_4)$
- *What if we only want* $P(\boldsymbol{X}_{56} \mid \boldsymbol{x}_4)$
  - *e.g., if we observed* $\boldsymbol{X}_4 = \boldsymbol{x}_4$

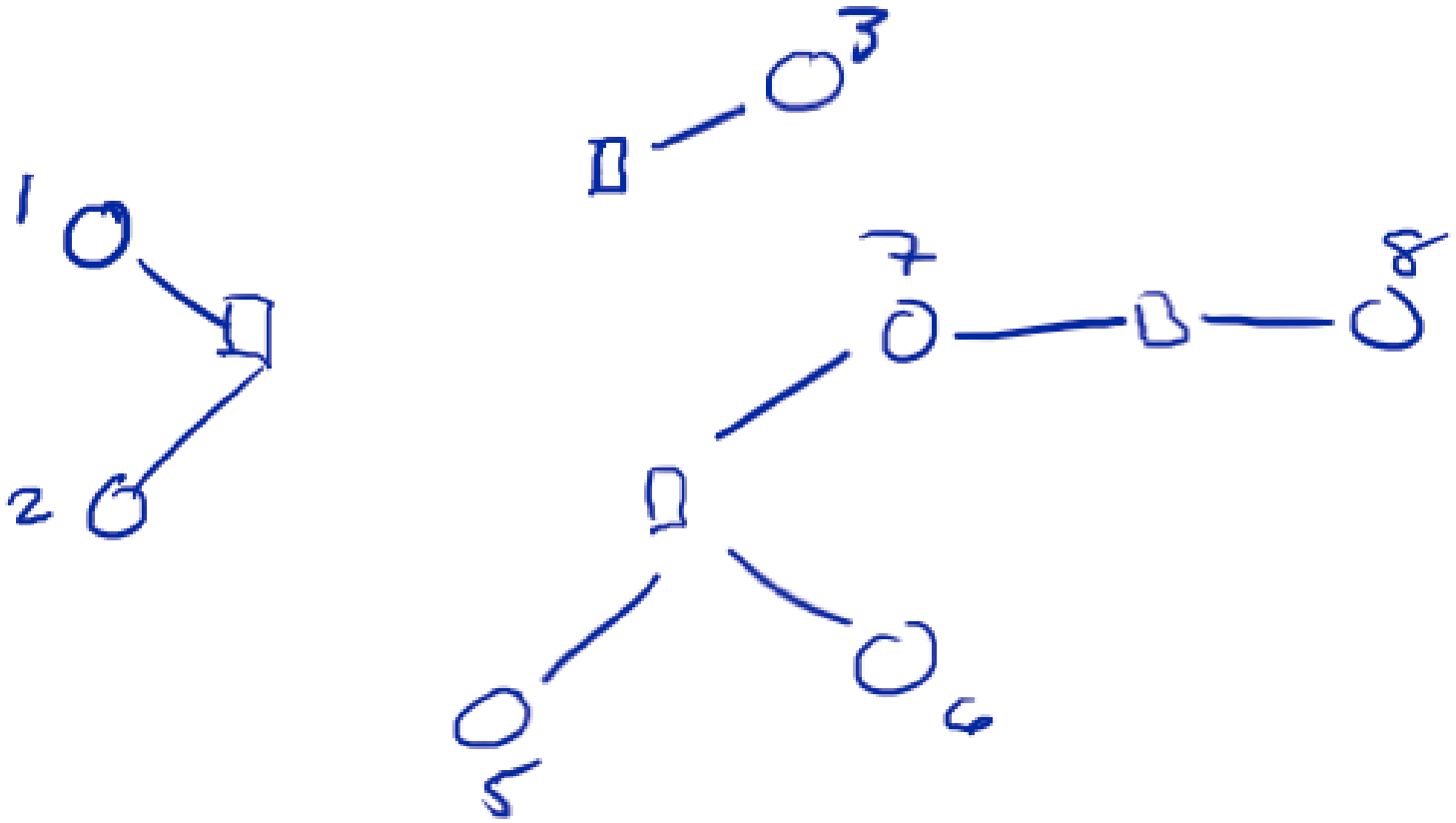
Conditioning on observations



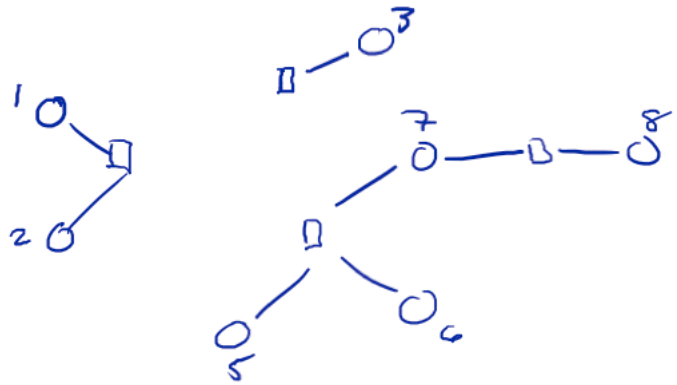
Conditioning on observations



Conditioning on observations



Graph separation = independence



- *Recall independence: P(X, Y) = P(X)P(Y)*

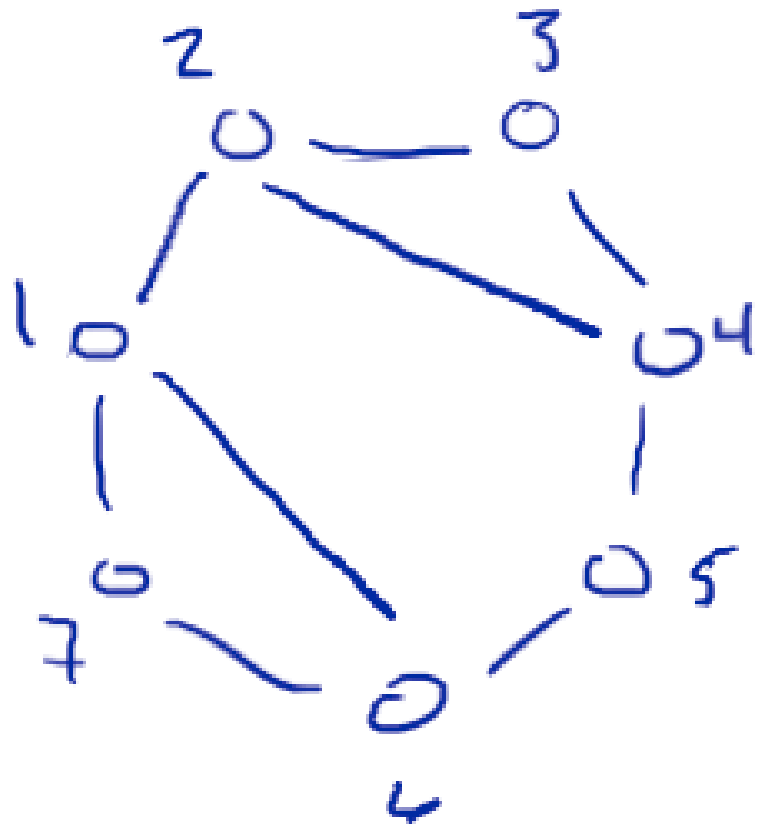
$$\sum_{1}\sum_{2}\sum_{3}\sum_{5}\sum_{6}\sum_{7}\sum_{8}\phi_{12}\phi_{3}\phi_{567}\phi_{78}$$

$$= \left(\sum_{1}\sum_{2}\phi_{12}\right)\left(\sum_{3}\phi_{3}\right)\left(\sum_{5}\sum_{6}\sum_{7}\sum_{8}\phi_{567}\phi_{78}\right)$$

Using independence

- *So, for $P(\mathbf{X}_{56} \mid \mathbf{x}_4)$, we can ignore the $\mathbf{X}_{12}$ piece and the $\mathbf{X}_3$ piece, and just work with the $\mathbf{X}_{5678}$ piece*
- *Eliminate $\mathbf{X}_{78}$*

A more difficult example



"Marrying" neighbors

- *When we sum out a variable X, we create a new factor whose domain is all neighboring r.v.s of X*
- *If lots of neighbors, this can be very costly*
- *Then, when we sum out another r.v. Y, we might create an even bigger factor, etc.*

Treewidth

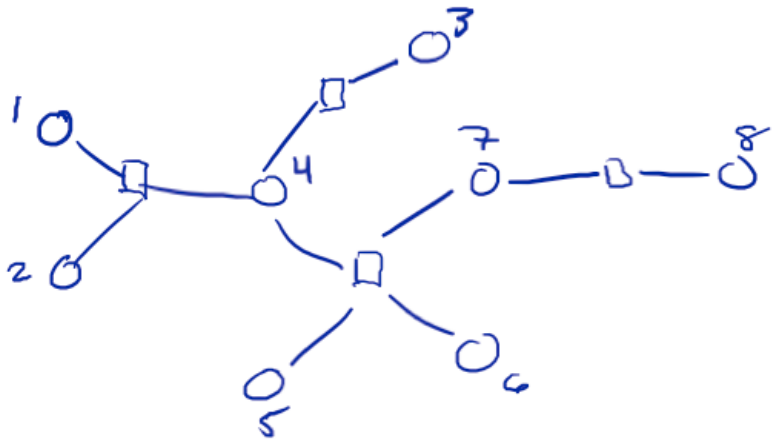
- *Elimination order E: sum x_1, then x_5, …*
- *treewidth(E) =*

  (size of largest factor formed) – 1
- *treewidth = $\min_E$ treewidth(E)*
- *Variable elimination uses space, time* ***exponential*** *in treewidth*
- *Worse: even computing treewidth is NP-complete*

Treewidth examples

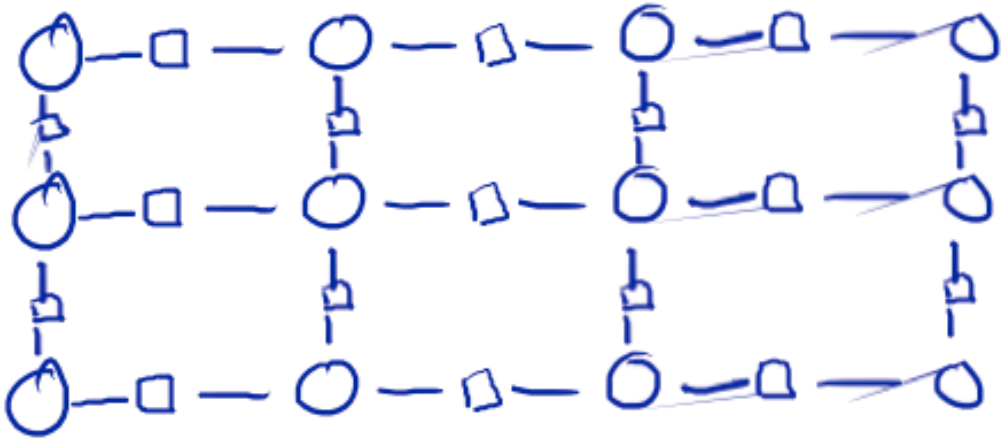
Chain

Tree

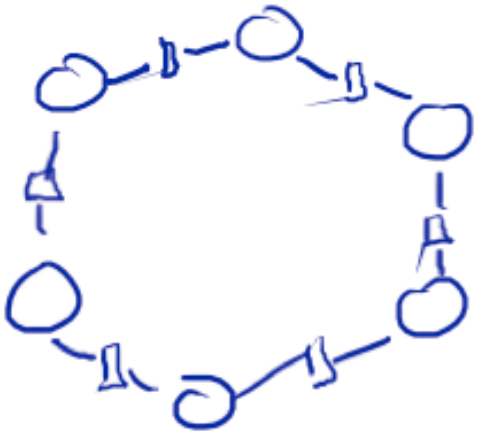


Treewidth examples

Parallel chains



Cycle



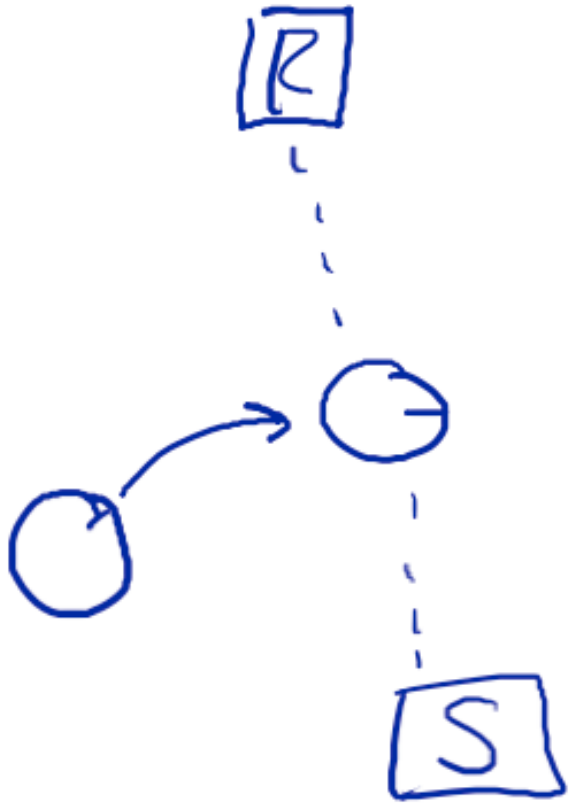
Aside: belief propagation

- *Suppose we want all 1-variable marginals*
- *Could do N runs of variable elimination*
- *Or: the BP algorithm simulates N runs for the price of 2*
- *For details: Kschischang et al. reading*

HMMs and DBNs

Inference over time

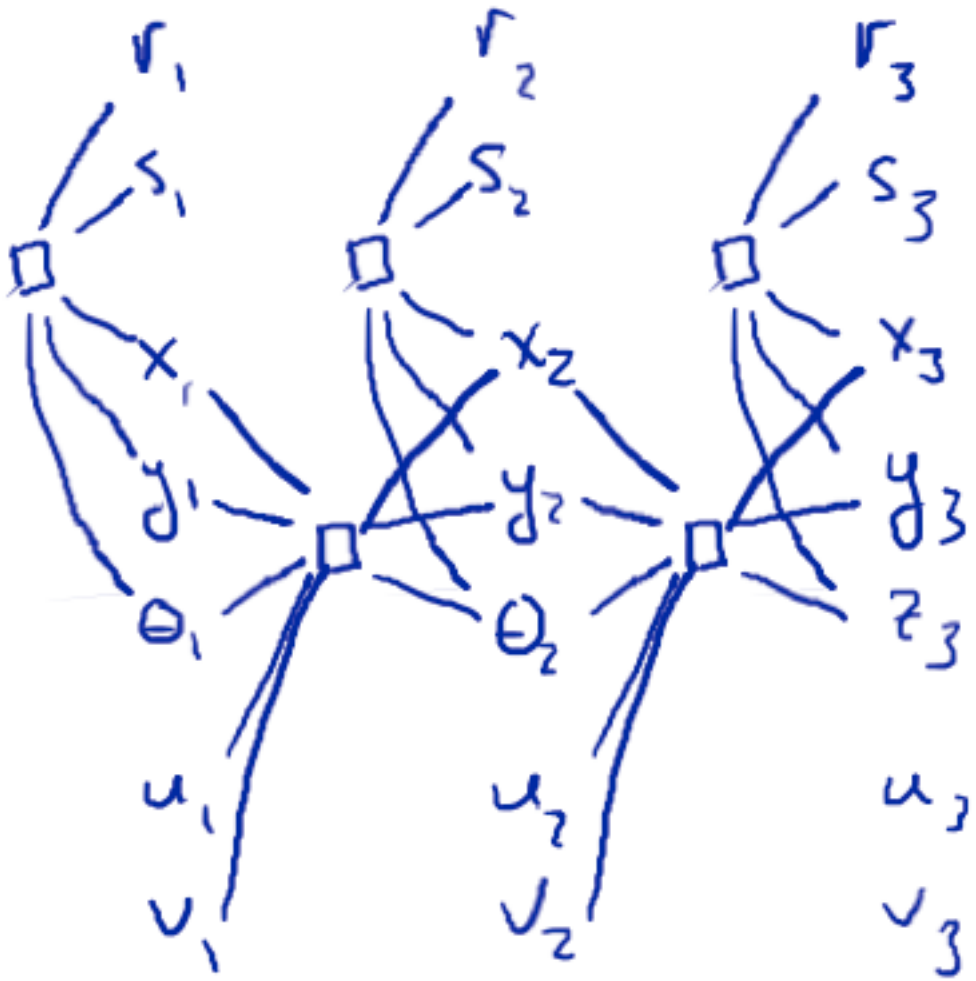
- *Consider a robot:*
  - *true state* (x, y, θ)
  - *controls* (v, w)
  - *two range sensors* (r, s)



Model

$$
\begin{aligned}
x_{t+1} &= x_t + v_t \cos\theta_t + \text{noise} \\
y_{t+1} &= y_t + v_t \sin\theta_t + \text{noise} \\
\theta_{t+1} &= \theta_t + w_t + \text{noise} \\
r_t &= \sqrt{(x_t - x^R)^2 + (y_t - y^R)^2} + \text{noise} \\
s_t &= \sqrt{(x_t - x^S)^2 + (y_t - y^S)^2} + \text{noise}
\end{aligned}
$$

Factor graph



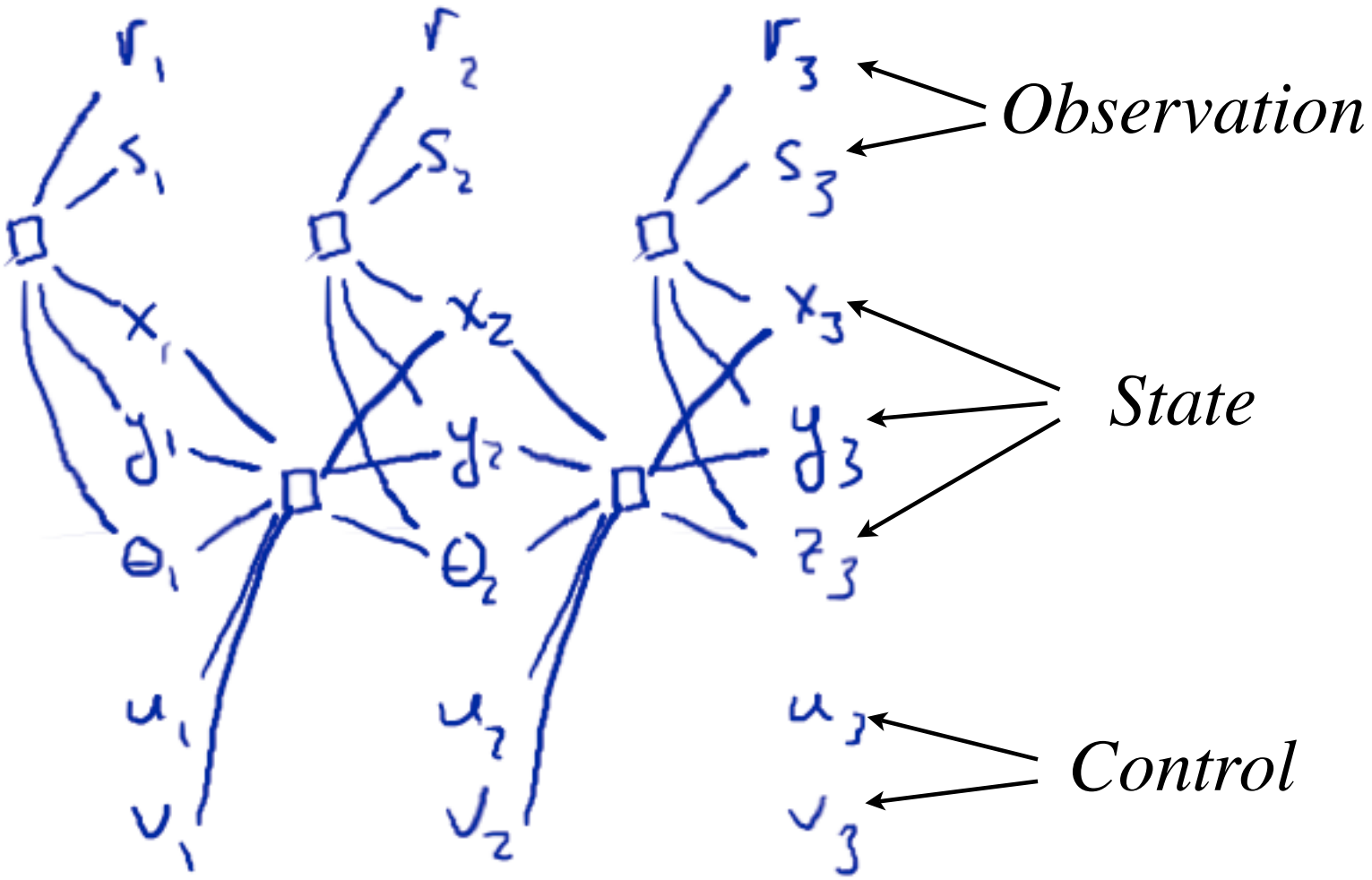
Dynamic Bayes Network

- *DBN: factor graph composed of a single structural unit repeated over time*
  - *conceptually infinite to right, but in practice cut off at some maximum T*
- *Factors **must** be conditional distributions*

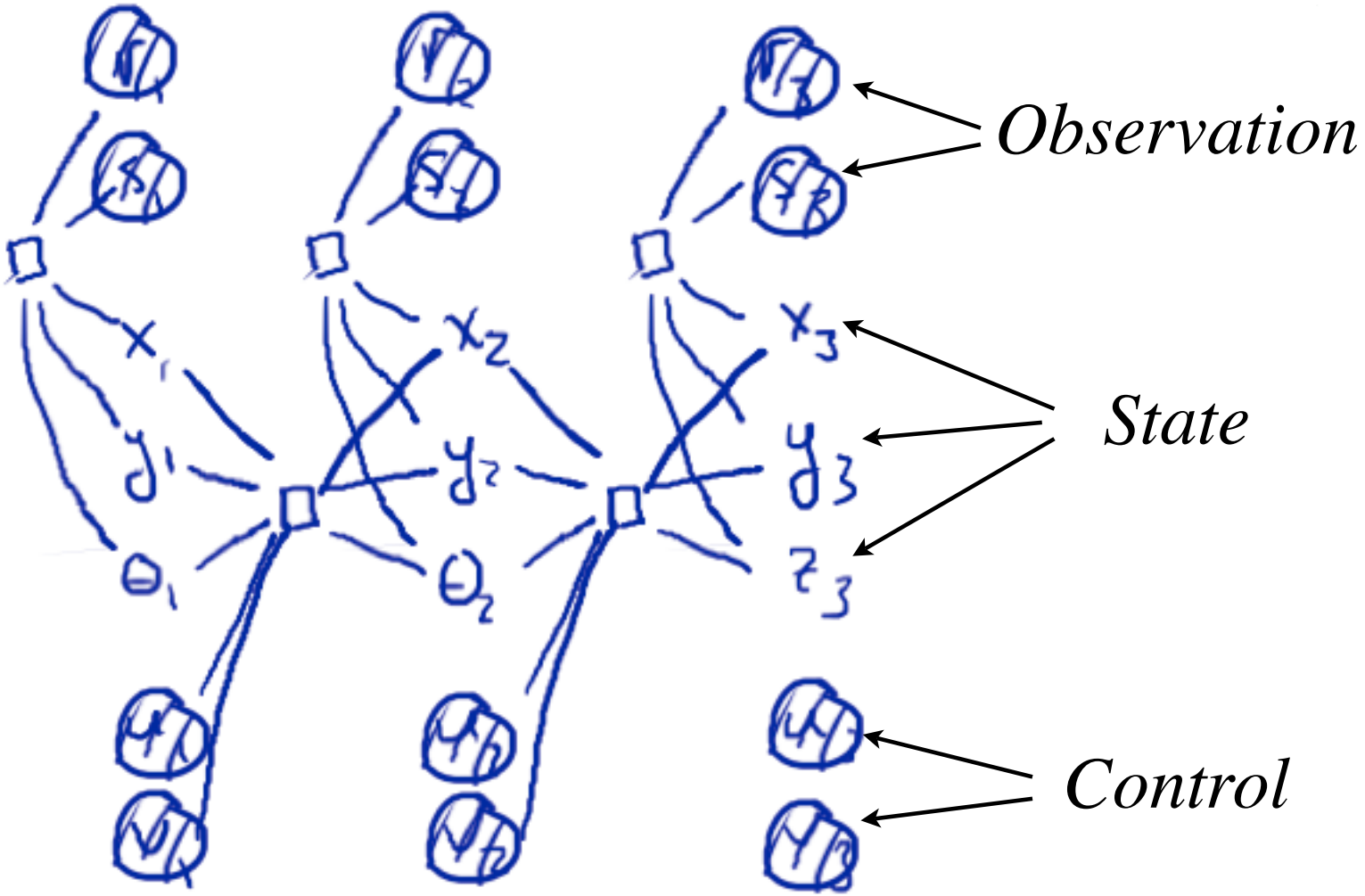
$$\forall x_t. \sum_{x_{t+1}} \phi(x_t, x_{t+1}) = 1$$

$$\forall x_t. \sum_{y_t} \phi(x_t, y_t) = 1$$

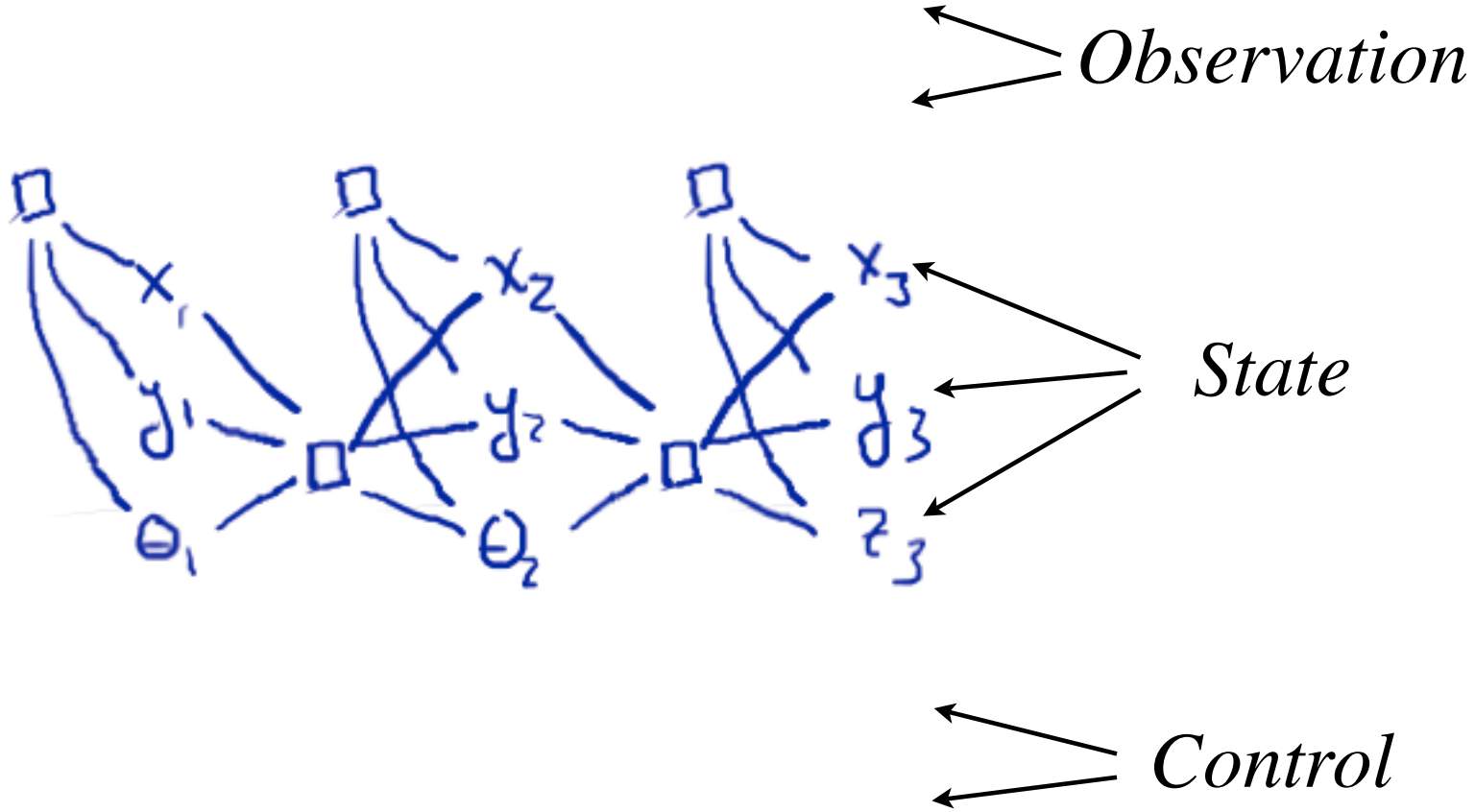
Three kinds of variable



Condition on obs, control



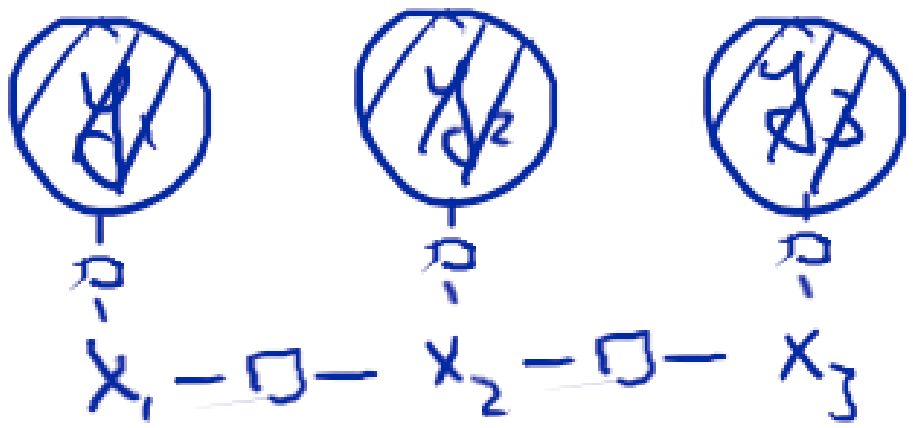
Condition on obs, control



Simplified version

- *State:* $x_t \in \{1, 2, 3\}$
- *Observation:* $y_t \in \{L, R\}$
- *Control: just one, "move randomly"*

Factor graph



Potentials

		X_{t+1}		
		1	2	3
X_t	1	.67	.33	0
	2	.33	.33	.33
	3	0	.33	.67

		Y_t	
		L	H
X_t	1	.67	.33
	2	.5	.5
	3	.33	.67

Hidden Markov Models

- *This is an HMM—a DBN with:*
  - *one state variable*
  - *one observation variable*

HMM inference

- *Condition on* $y_1 = H, y_2 = H, y_3 = L$
- *What is* $P(X_2 \mid HHL)$*?*

Forward-backward

- *You may recognize the above as the forward-backward algorithm*
- *Special case of belief propagation*

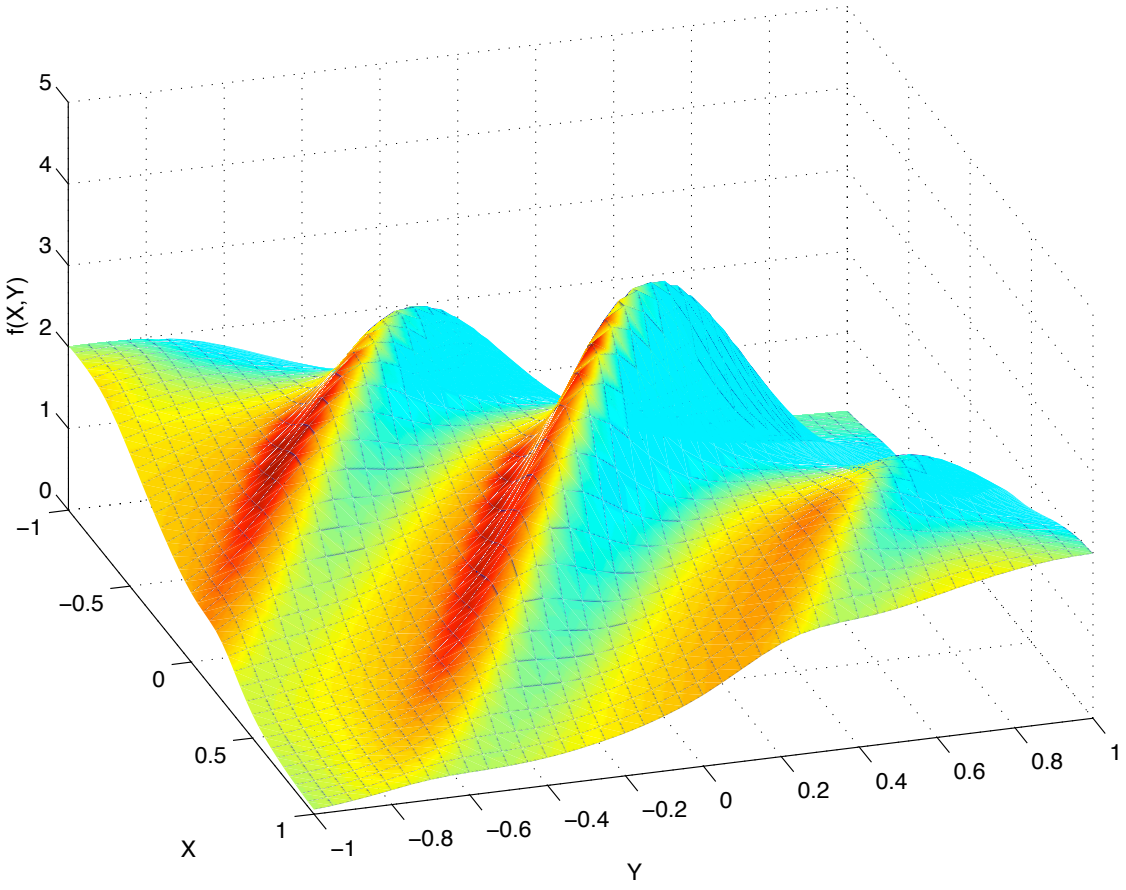
Approximate Inference

Most of the time…

- *Treewidth is big*
- *Variables are high-arity or continuous*
- *Can't afford exact inference*

- *Partition function = numerical integration (and/or summation)*
- *We'll look at randomized algorithms*

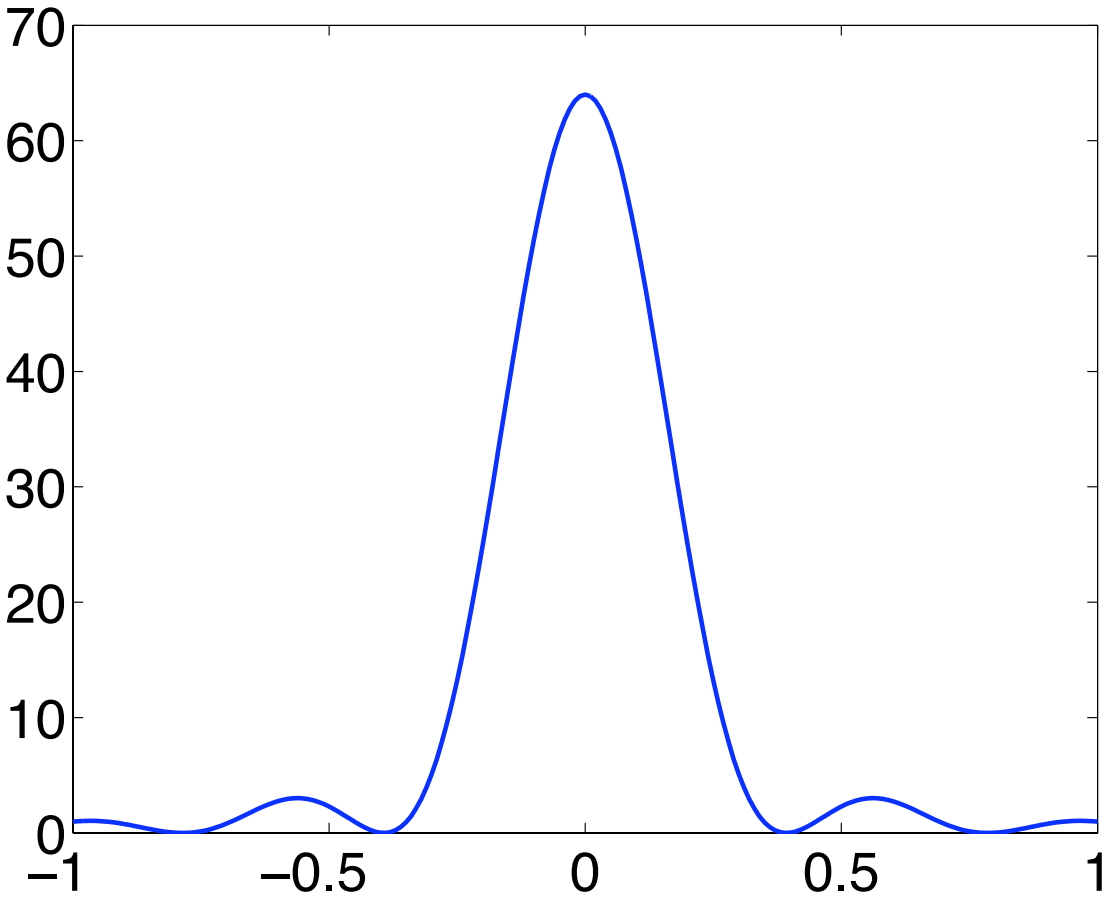
Numerical integration



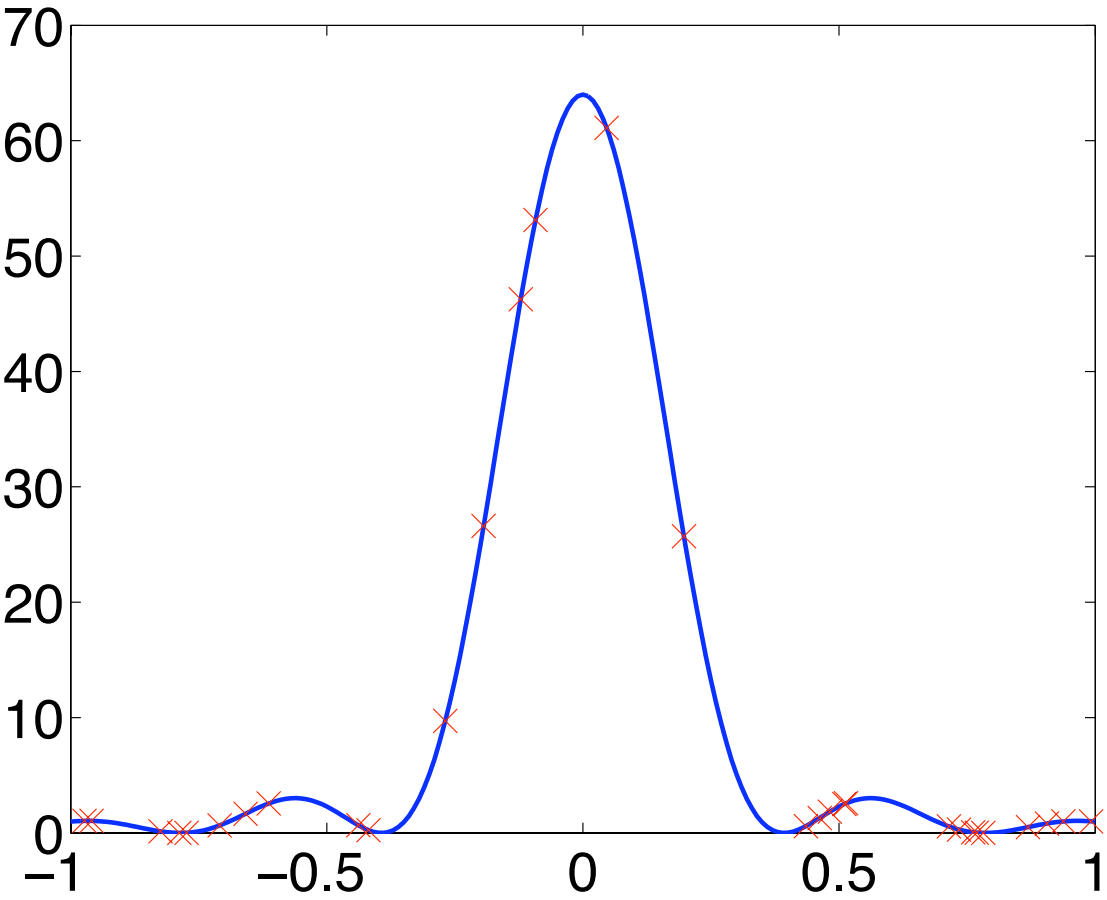
Integration in 1000s of dims

Eliazar and Parr, IJCAI-03

Simple 1D problem



Simplest randomized algorithm



- *Uniform sampling: sum($f(x_i)$)/N*

Uniform sampling

$$E(f(X)) = \int P(x) f(x) dx$$

$$= \frac{1}{V} \int f(x) dx$$

- *So, V E(f(X)) is desired integral*
- *But standard deviation can be big*
- *Can reduce it by averaging many samples*
- *But only at rate 1/sqrt(N)*

Importance sampling

- *Instead of x ~ uniform, use x ~ Q(x)*
- *Q = importance distribution*
- *Should have Q(x) large where f(x) is large*
- *Problem:*

$$E_Q(f(X)) = \int Q(x) f(x) dx$$

Importance sampling

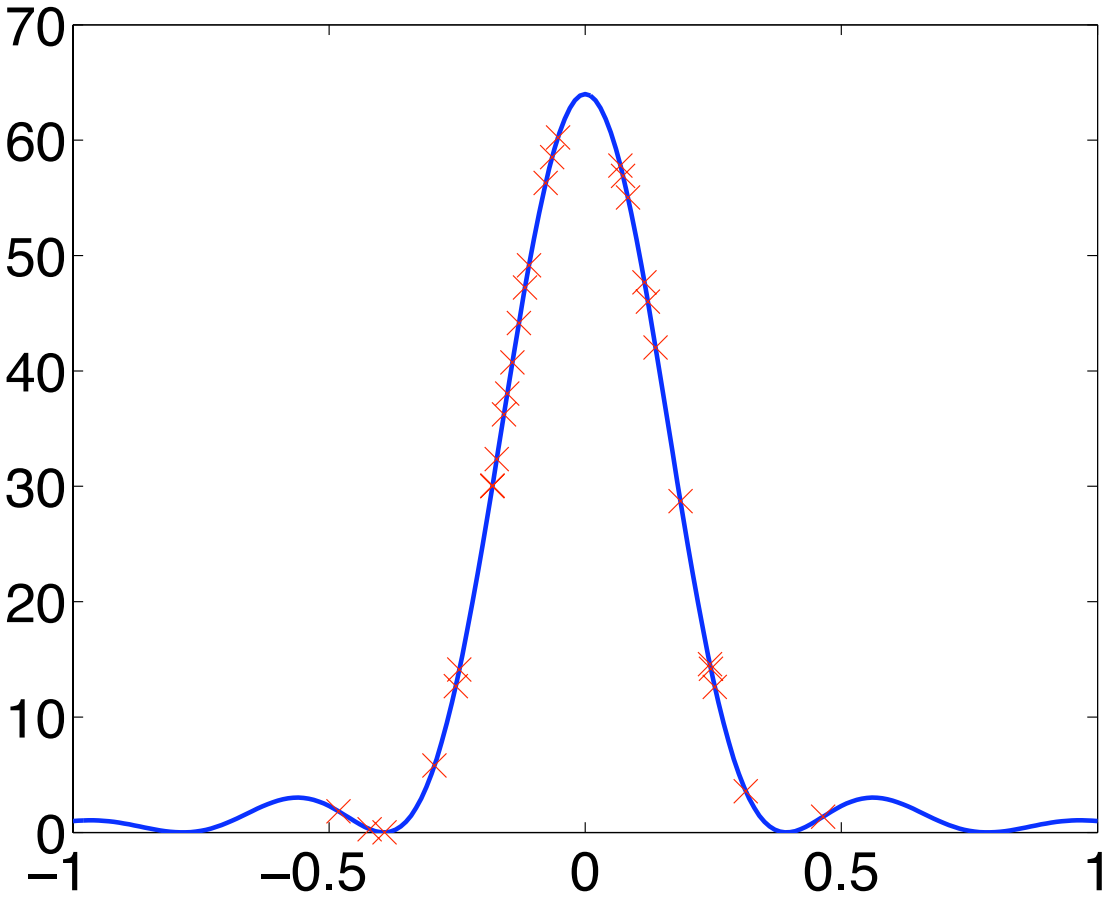
$$h(x) \equiv f(x)/Q(x)$$

$$\begin{aligned} E_Q(h(X)) &= \int Q(x)h(x)dx \\ &= \int Q(x)f(x)/Q(x)dx \\ &= \int f(x)dx \end{aligned}$$

Importance sampling

- *So, take samples of h(X) instead of f(X)*
- $w_i = 1/Q(x_i)$ *is* ***importance weight***
- *Q = uniform yields uniform sampling*

Importance sampling



Variance

- *How does this help us control variance?*
- *Suppose f big ==> Q big*
- *And Q small ==> f small*
- *Then h = f/Q never gets too big*
- *Variance of each sample is lower ==> need fewer samples*
- *A good Q makes a good IS*

Importance sampling, part II

- *Suppose we want*

$$\int f(x)dx = \int P(x)g(x)dx = E_P(g(X))$$

- *Pick N samples x_i from proposal Q(X)*
- *Average w_i $g(x_i)$, where $w_i = P(x_i)/Q(x_i)$ is importance weight*

$$E_Q(Wg(X)) = \int Q(x)[P(x)/Q(x)]g(x)dx = \int P(x)g(x)dx$$

Parallel importance sampling

- *Suppose we want*

$$\int f(x)dx = \int P(x)g(x)dx = E_P(g(X))$$

- *But P(x) is unnormalized (e.g., represented by a factor graph)—know only Z P(x)*

Parallel IS

- *Pick N samples x_i from proposal Q(X)*
- *If we knew $w_i = P(x_i)/Q(x_i)$, could do IS*
- *Instead, set* $\hat{w}_i = ZP(x_i)/Q(x_i)$

Parallel IS

$$
\begin{aligned}
E(\hat{W}) &= \int Q(x)(ZP(x)/Q(x))dx \\
&= Z\int P(x)dx \\
&= Z
\end{aligned}
$$

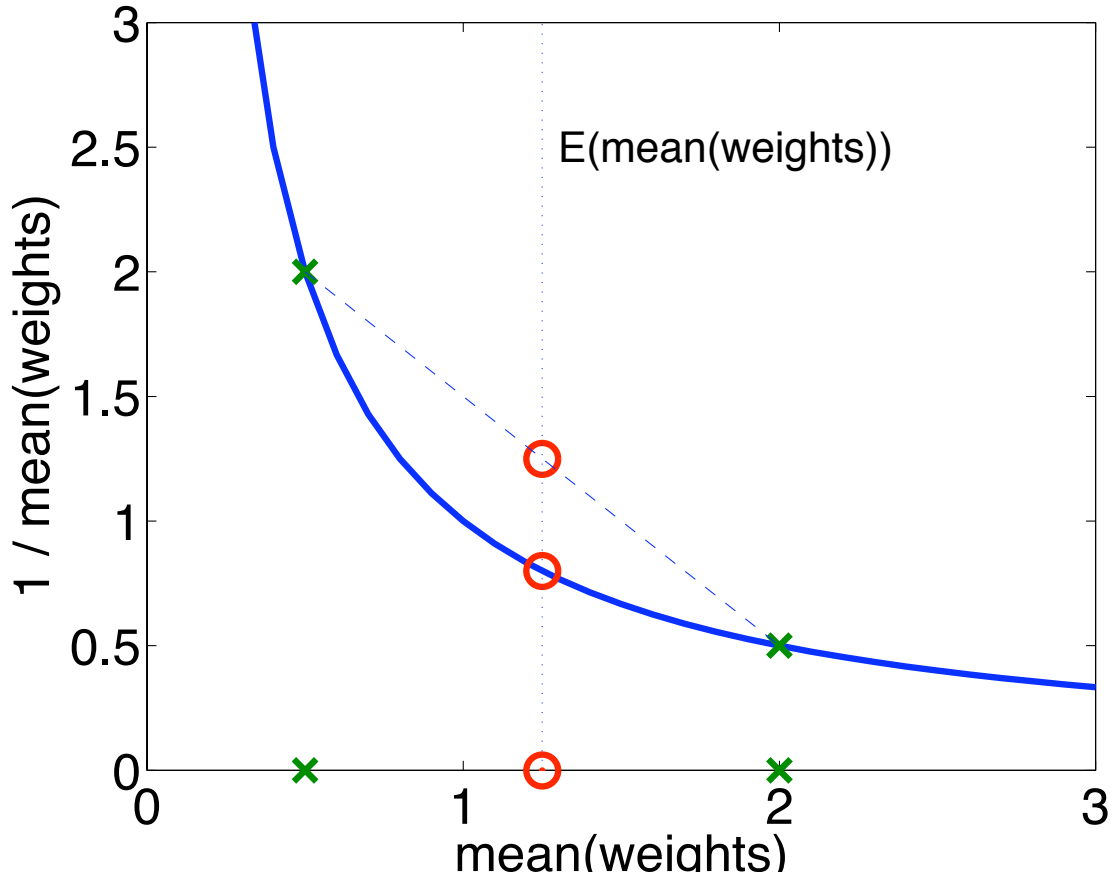
- *So,* $\bar{w} = \frac{1}{N}\sum_i \hat{w}_i$ *is an unbiased estimate of* Z

Parallel IS

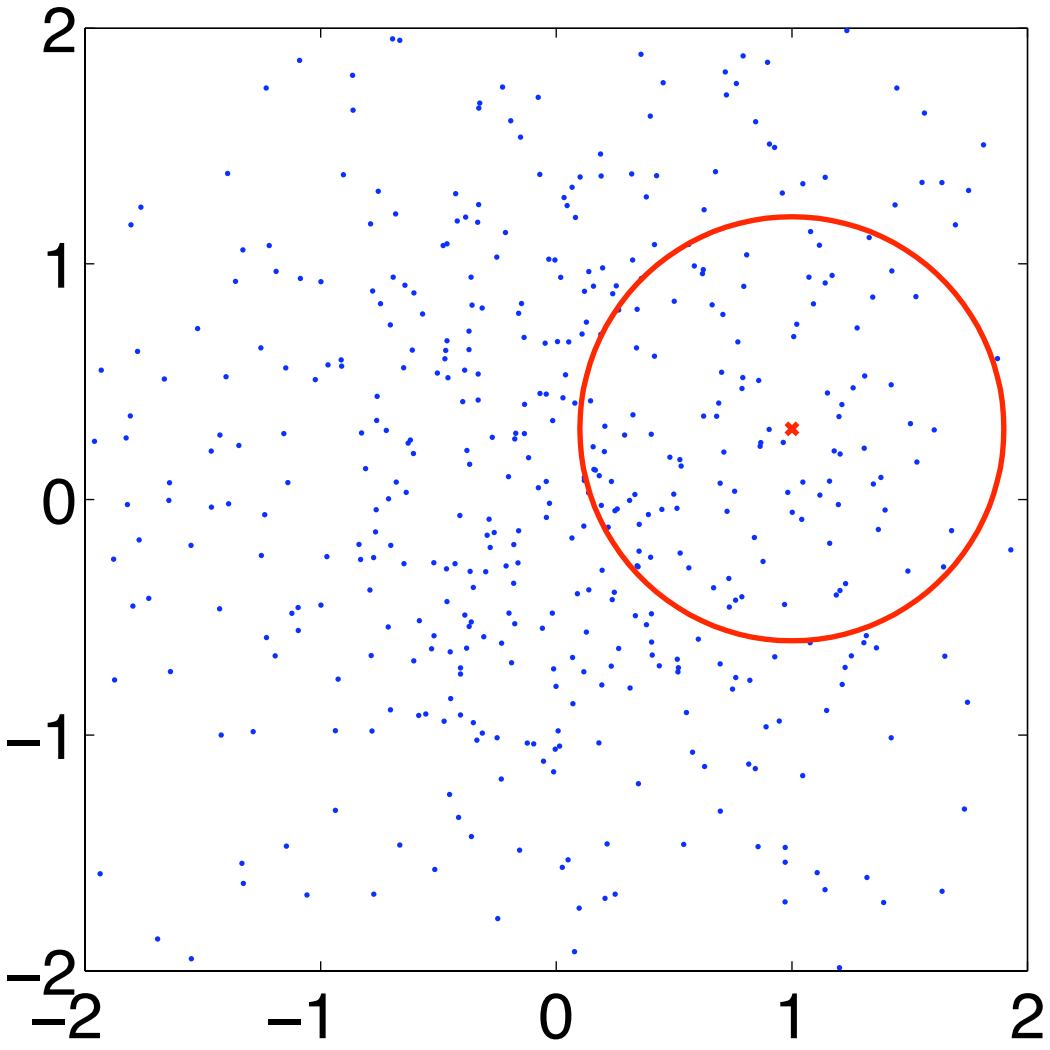
- *So, $\hat{w}_i/\bar{w}$ is an estimate of w_i, computed without knowing Z*
- *Final estimate:*

$$\int f(x)dx \approx \frac{1}{n}\sum_i \frac{\hat{w}_i}{\bar{w}} g(x_i)$$

Parallel IS is biased

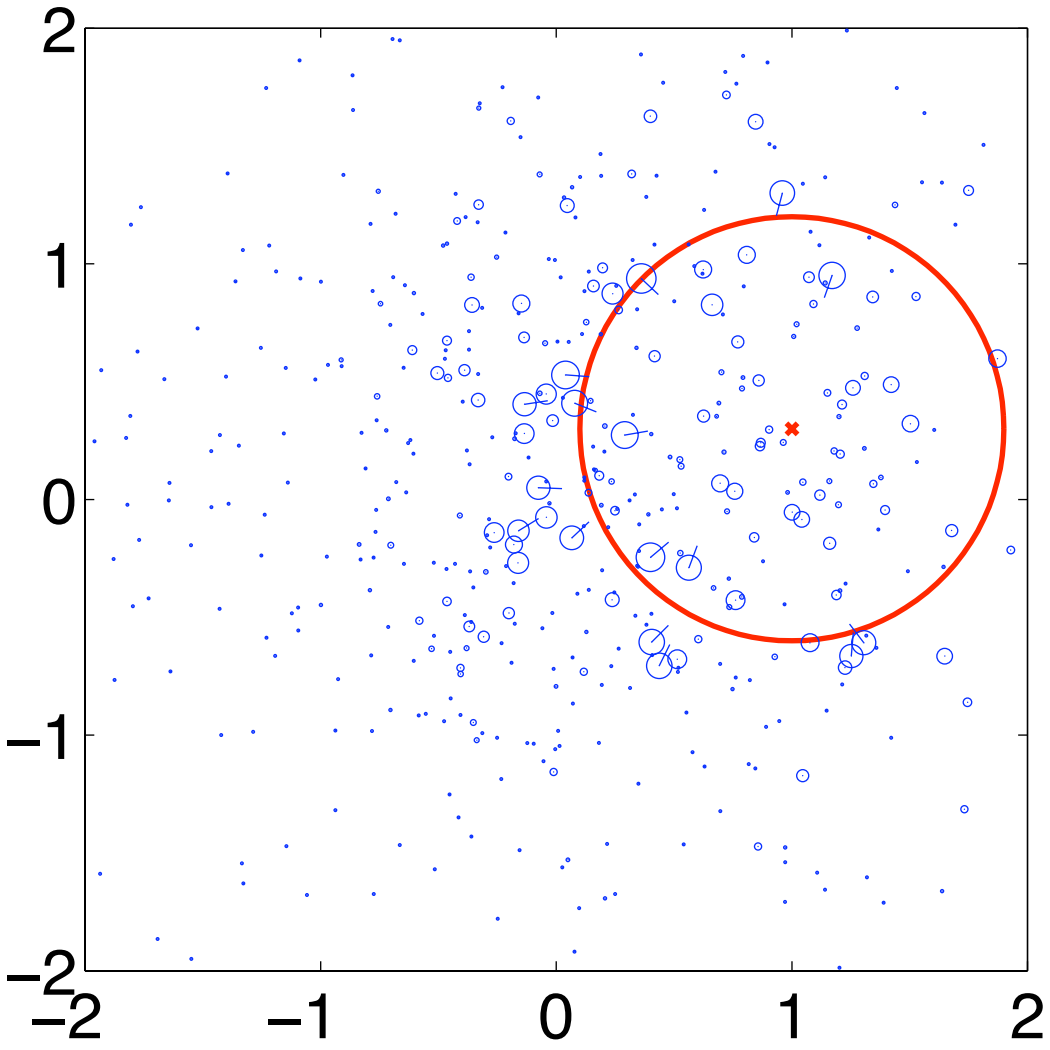


$$E(\bar{W}) = Z, \text{ but } E(1/\bar{W}) \neq 1/Z \text{ in general}$$



$$Q : (X, Y) \sim N(1, 1) \qquad \theta \sim U(-\pi, \pi)$$

$$f(x, y, \theta) = Q(x, y, \theta) P(o = 0.8 \mid x, y, \theta)/Z$$



$$\textit{Posterior}\ \ E(X, Y, \theta) = (0.496, 0.350, 0.084)$$

Back to high dimensions

- *Picking a good importance distribution is hard in high-D*
- *Major contributions to integral can be hidden in small areas*
  - *recall, want (P big ==> Q big)*
- *Would like to search for areas of high P(x)*
- *But searching could bias our estimates*

MCMC

Markov-Chain Monte Carlo

- *Design a randomized search procedure M which tends to increase P(x) if it is small*
- *Run M for a while, take resulting x as a sample*
- *Importance weight P(x)/Q(x)?*

Markov-Chain Monte Carlo

- *Design a randomized search procedure M which tends to increase P(x) if it is small*
- *Run M for a while, take resulting x as a sample*
- *Importance weight P(x)/Q(x)?*
  Q = stationary distribution of M

Stationary distribution

- *If we run M a long time, eventually we won't* * *be able to tell where we started*
- *Now look at current value of x*
- *This is a sample from* ***stationary distribution*** *of M*
- *…which is why we use Q = stationary distribution in importance weight*

Designing a search chain

$$\int f(x)dx = \int P(x)g(x)dx = E_P(g(x))$$

- *Would like Q(x) = P(x)*
  - *makes importance weight = 1*
- *Turns out we can get this exactly, using Metropolis-Hastings*

Metropolis-Hastings

- *Way of designing chain w/ $Q(x) = P(x)$*
- *Basic strategy: start from arbitrary x*
- *Repeatedly tweak x to get x'*
- *If $P(x') \geq P(x)$, move to x'*
- *If $P(x') << P(x)$, stay at x*
- *In intermediate cases, randomize*

Proposal distribution

- *Left open: what does "tweak" mean?*
- *Parameter of MH: Q(x' | x)*
  - *one-step proposal distribution*
- *Good proposals explore quickly, but remain in regions of high P(x)*
- *Optimal proposal?*

MH algorithm

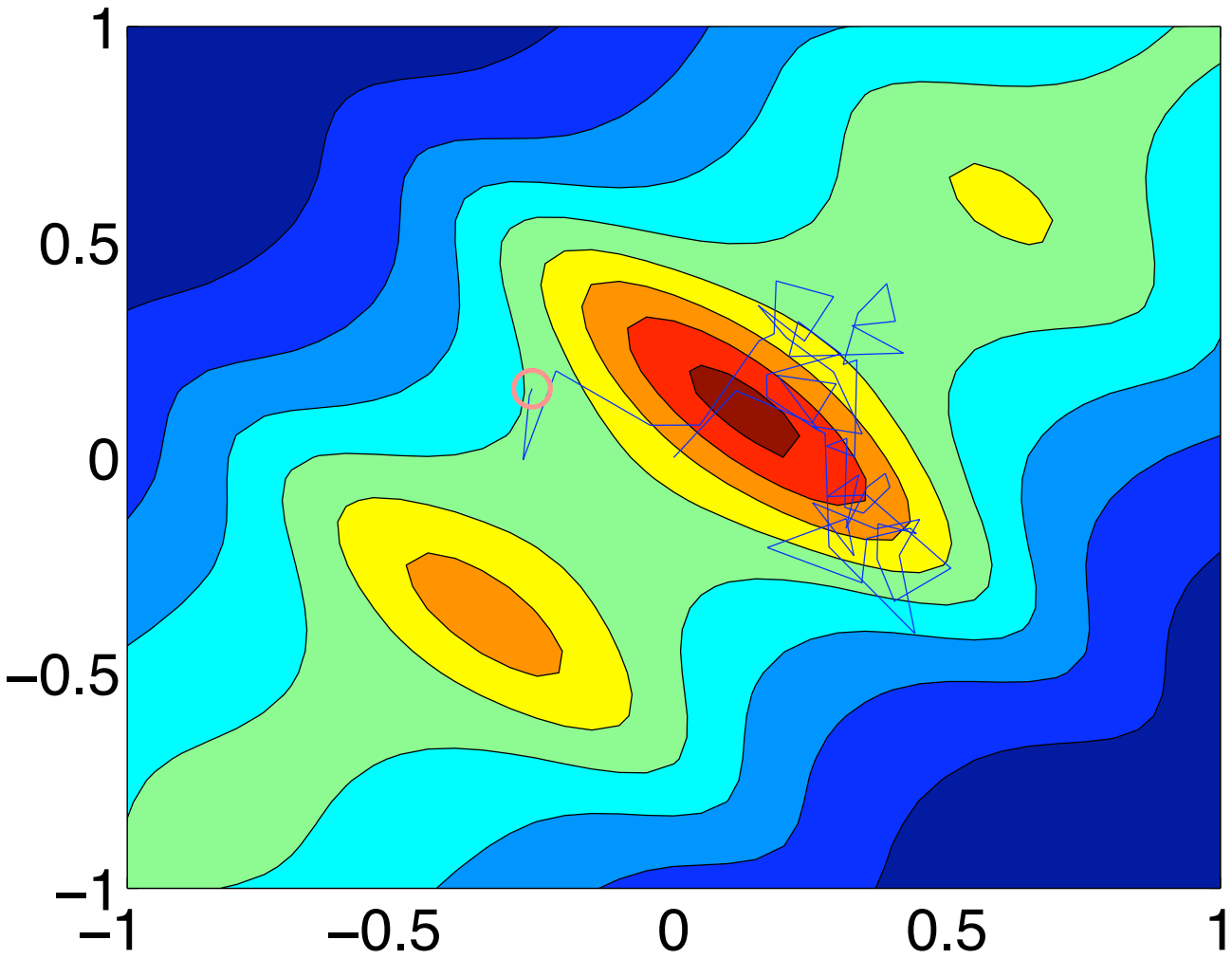
- *Sample x' ~ Q(x' | x)*
- *Compute* $p = \frac{P(x')}{P(x)} \frac{Q(x \mid x')}{Q(x' \mid x)}$
- *With probability min(1, p), set x := x'*
- *Repeat for T steps; sample is* $x_1, \ldots, x_T$ *(will usually contain duplicates)*

MH algorithm

note: we don't need to know Z

- *Sample x' ~ Q(x' | x)*
- *Compute* $p = \frac{P(x')}{P(x)} \frac{Q(x \mid x')}{Q(x' \mid x)}$
- *With probability min(1, p), set x := x'*
- *Repeat for T steps; sample is* $x_1, \ldots, x_T$ *(will usually contain duplicates)*

MH example



Acceptance rate

- *Moving to new x' is **accepting***
- *Want **acceptance rate** (avg p) to be large (so we don't get big runs of the same x)*
- *Want Q(x' | x) to move long distances (to explore quickly)*
- *Tension between Q and P(accept):*

$$p = \frac{P(x')}{P(x)} \frac{Q(x \mid x')}{Q(x' \mid x)}$$

Mixing rate, mixing time

- *If we pick a good proposal, we will move rapidly around domain of P(x)*
- *After a short time, won't be able to tell where we started*
- *This is fast* ***mixing rate =*** *1 / (mixing time)*
- ***Mixing time*** *= # steps until we can't tell accurately which starting point we used*

MH estimate

- *Once we have our samples x_1, x_2, …*
- *Optional: discard initial "burn-in" range*
  - *allows time to reach stationary dist'n*
- *Estimated integral:* $\frac{1}{N}\sum_{i=1}^{N} f(x_i)$

In example

- $f(x) = x^2$
- *True E(f(x)) = 0.28…*
- *Proposal:* $Q(x' \mid x) = N(x' \mid x, 0.25^2 I)$
- *Acceptance rate 55–60%*
- *After 1000 samples, minus burn-in of 100:*

```
final estimate 0.282361
final estimate 0.271167
final estimate 0.322270
final estimate 0.306541
final estimate 0.308716
```

MH proof

- *Write T(x' | x) for transition probability*
- *Write p(x' | x) for acceptance probability*

$$\min\left(1, \frac{P(x')}{P(x)} \frac{Q(x \mid x')}{Q(x' \mid x)}\right)$$

- *If x' ≠ x, then*
  - *T(x' | x) = Q(x' | x) p(x' | x)*

Detailed balance

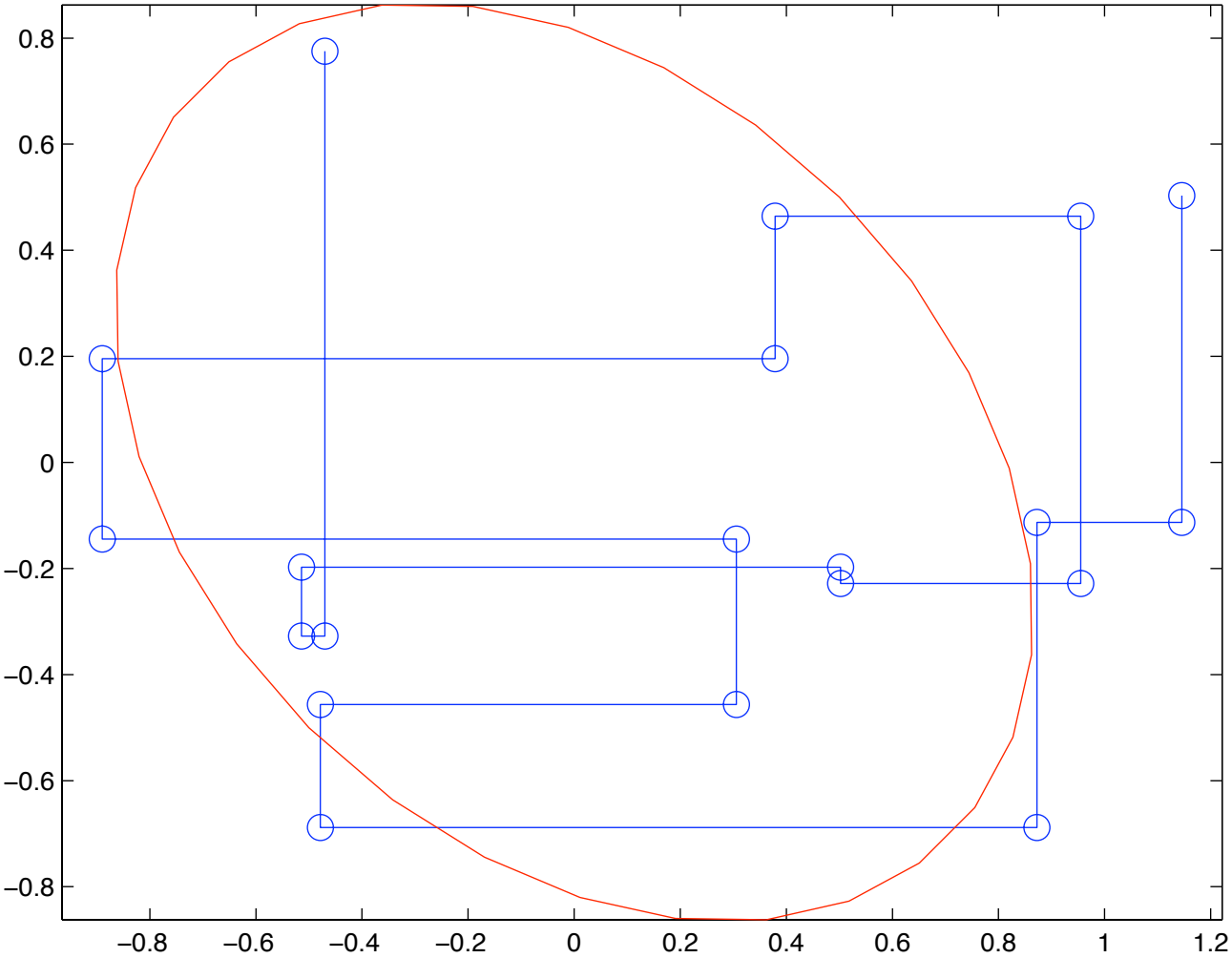
$$P(x)T(x' \mid x) = P(x')T(x \mid x') \qquad \forall x, x'$$

- *Detailed balance implies that P(x) is our stationary distribution:*
  - *take integral dx on both sides*
  - *use Bayes rule, law of total probability on RHS*

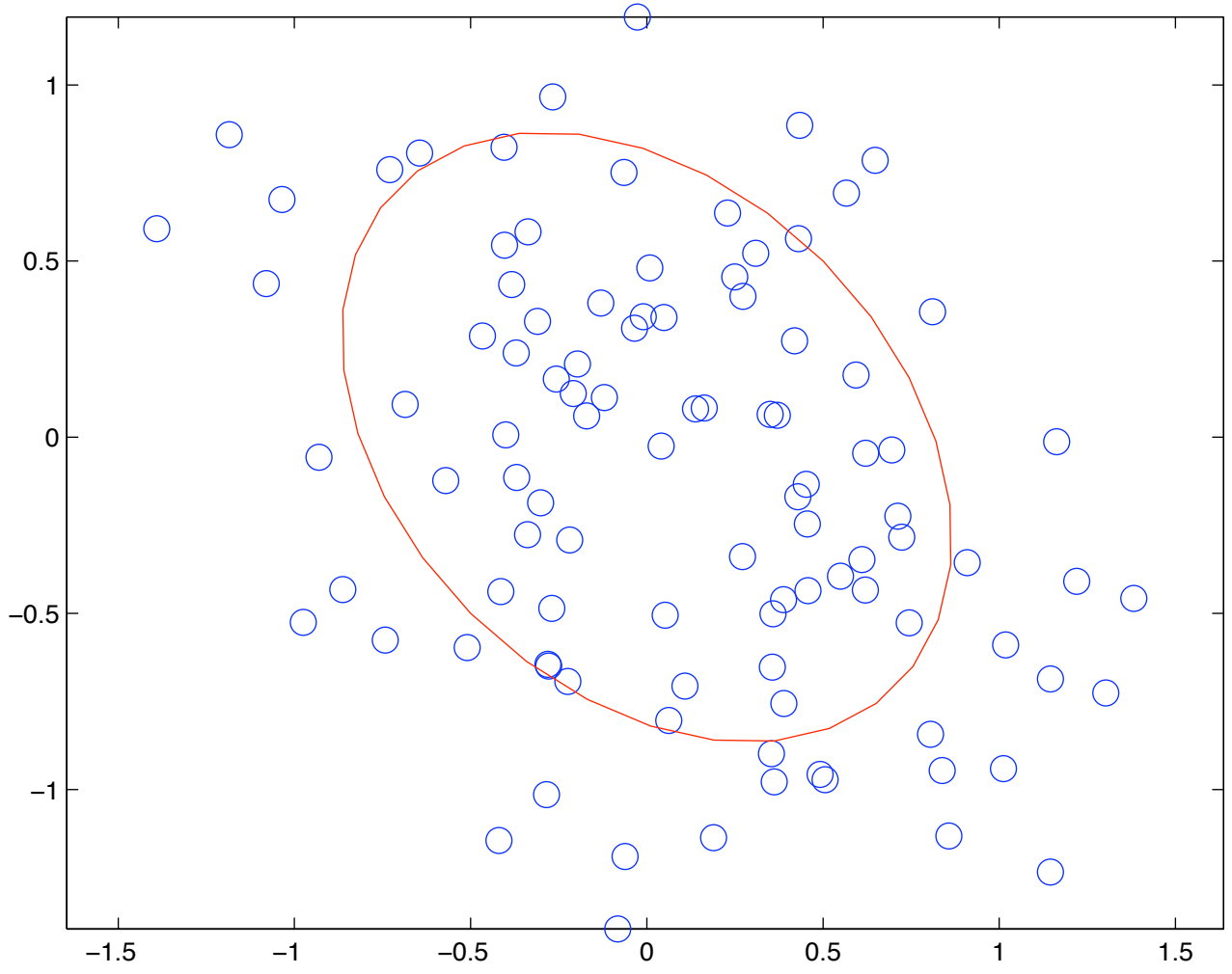
Gibbs sampler

- *Special case of MH*
- *Divide* $\mathbf{X}$ *into blocks of r.v.s B(1), B(2), …*
- *Proposal Q:*
  - *pick a block i uniformly*
  - *sample* $\mathbf{X}_{B(i)} \sim P(\mathbf{X}_{B(i)} \mid \mathbf{X}_{\neg B(i)})$

Gibbs example



Gibbs example



Why is Gibbs useful?

- *For Gibbs,* $p = \frac{P(x'_i, x'_{\neg i})}{P(x_i, x_{\neg i})} \frac{P(x_i \mid x'_{\neg i})}{P(x'_i \mid x_{\neg i})}$

Gibbs in practice

- *Above fact about p means Gibbs is often easy to implement*
- *Often works well*
  - ***if** we choose good blocks (but there may be no good blocking!)*
- *Fancier version: adaptive blocks, based on current **x***

Gibbs failure example