15-780: Graduate AI
Lecture 18. Learning

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

Admin

- *HW4 questions?*
- *HW4 extension: was due Tue, now Thu*
- *Project interim reports*
  - *reminder: due 4/14*
- *Midterm 4/9*
  - *prep sessions TBA*

Review

Probability

- *Expectation*
- *Conditional expectation*
  - *law of iterated expectations*
- *Sample vs. population quantities*
- *Estimators; consistency*

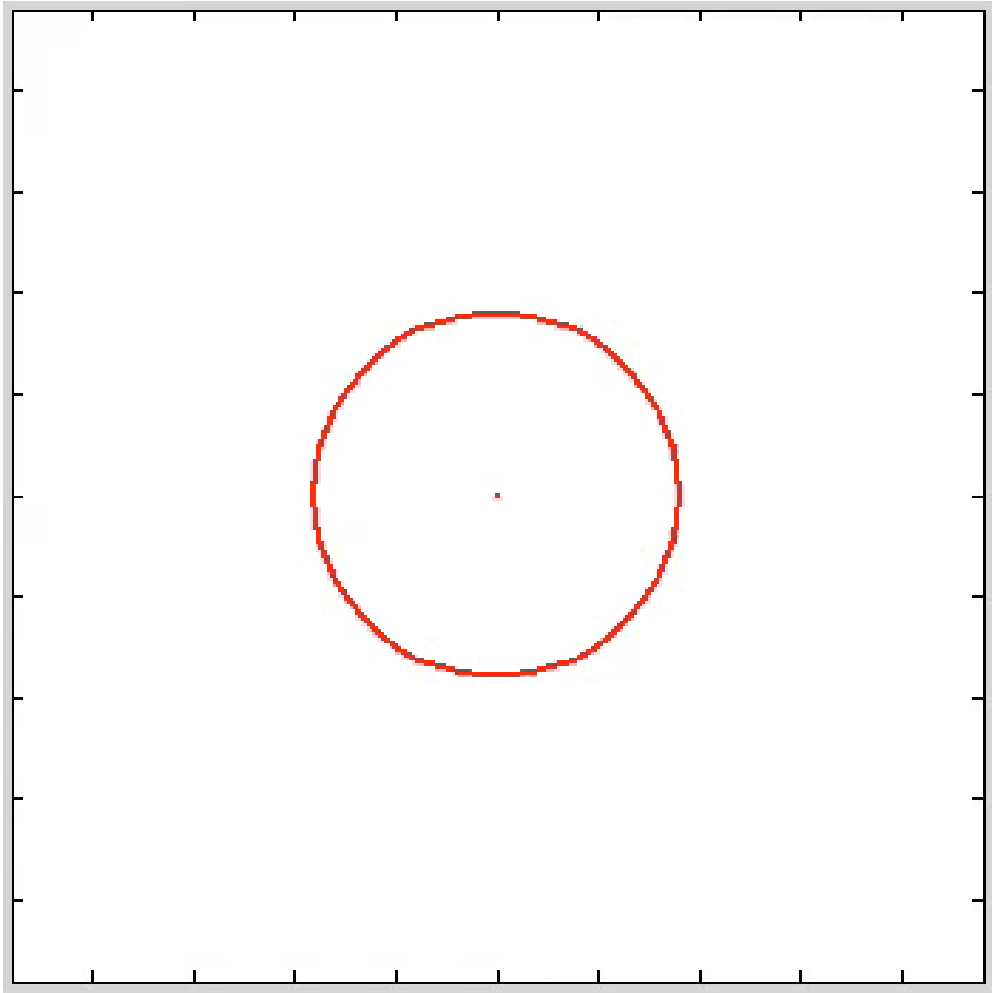
Markov-Chain Monte Carlo

- *For computing:*

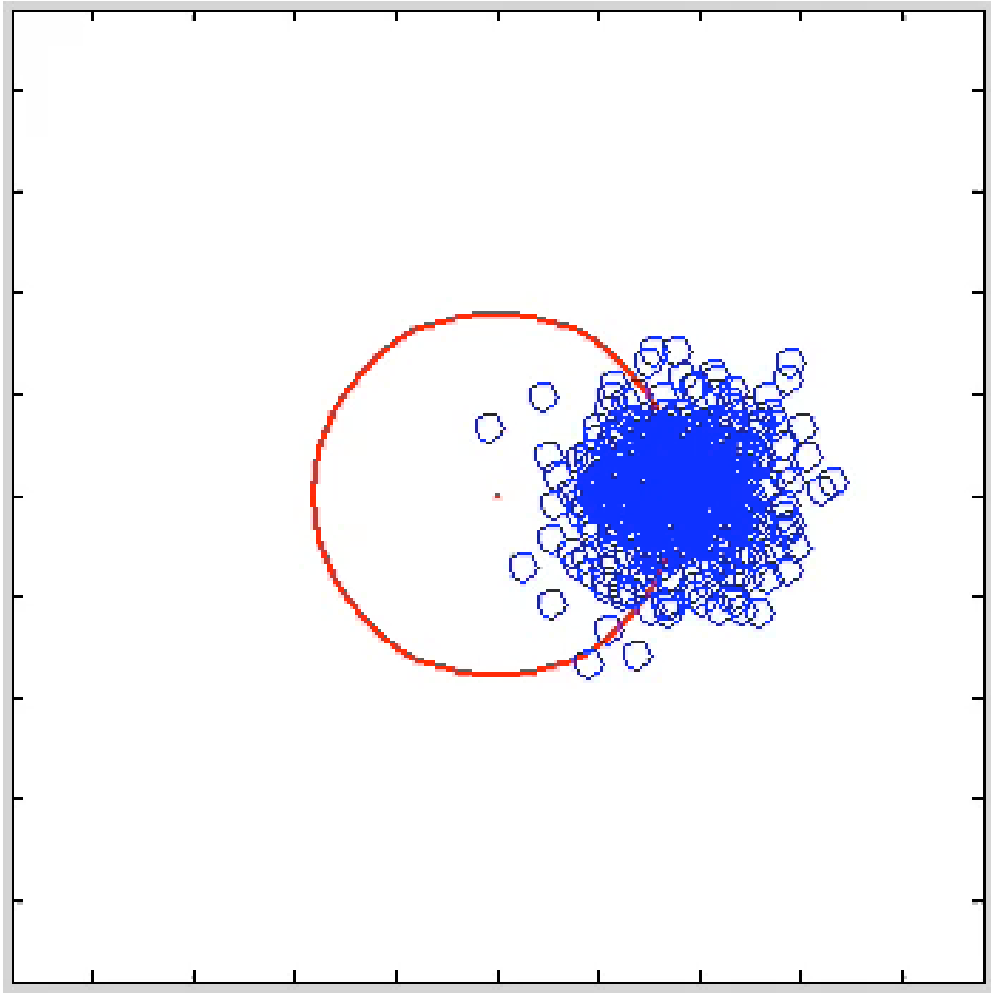
$$E_P(g(X)) = \int g(x)P(x)dx = \int f(x)dx$$

- *Chief difficulty: finding a good importance distribution Q(x)*
- *Metropolis-Hastings: optimal distribution (Q = P) using randomized search*

Markov chain



Stationary distribution



Stationary distribution

$$Q(\mathbf{x}_t) = \mathbb{P}(\mathbf{x}_t) \quad \Rightarrow \quad Q(\mathbf{x}_{t+1}) = \mathbb{P}(\mathbf{x}_{t+1})$$

$$Q(\mathbf{x}_t) = \mathbb{P}(\mathbf{x}_t) \quad \Rightarrow \quad Q(\mathbf{x}_{t+1}) = \int \mathbb{P}(\mathbf{x}_{t+1}, \mathbf{x}_t) d\mathbf{x}_t$$

$$Q(\mathbf{x}_t) = \mathbb{P}(\mathbf{x}_t) \quad \Rightarrow \quad Q(\mathbf{x}_{t+1}) = \int \mathbb{P}(\mathbf{x}_{t+1} \mid \mathbf{x}_t) \mathbb{P}(\mathbf{x}_t) d\mathbf{x}_t$$

$$Q(\mathbf{x}_t) = \mathbb{P}(\mathbf{x}_t) \quad \Rightarrow \quad Q(\mathbf{x}_{t+1}) = \int \mathbb{P}(\mathbf{x}_{t+1} \mid \mathbf{x}_t) Q(\mathbf{x}_t) d\mathbf{x}_t$$

Stationary distribution

$$Q(\mathbf{x}_t) = \mathbb{P}(\mathbf{x}_t) \quad \Rightarrow \quad Q(\mathbf{x}_{t+1}) = \mathbb{P}(\mathbf{x}_{t+1})$$

$$Q(\mathbf{x}_t) = \mathbb{P}(\mathbf{x}_t) \quad \Rightarrow \quad Q(\mathbf{x}_{t+1}) = \int \mathbb{P}(\mathbf{x}_{t+1}, \mathbf{x}_t) d\mathbf{x}_t$$

$$Q(\mathbf{x}_t) = \mathbb{P}(\mathbf{x}_t) \quad \Rightarrow \quad Q(\mathbf{x}_{t+1}) = \int \mathbb{P}(\mathbf{x}_{t+1} \mid \mathbf{x}_t) \mathbb{P}(\mathbf{x}_t) d\mathbf{x}_t$$

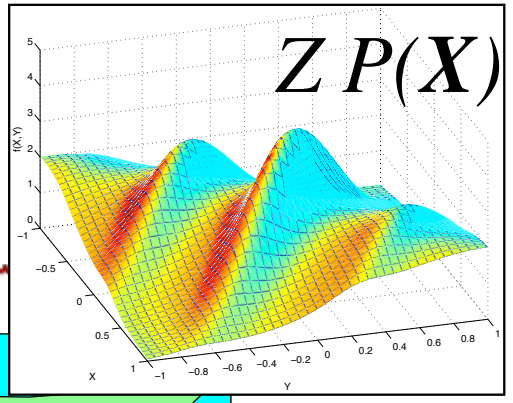
$$Q(\mathbf{x}_t) = \mathbb{P}(\mathbf{x}_t) \quad \Rightarrow \quad \boxed{Q(\mathbf{x}_{t+1}) = \int \mathbb{P}(\mathbf{x}_{t+1} \mid \mathbf{x}_t) Q(\mathbf{x}_t) d\mathbf{x}_t}$$

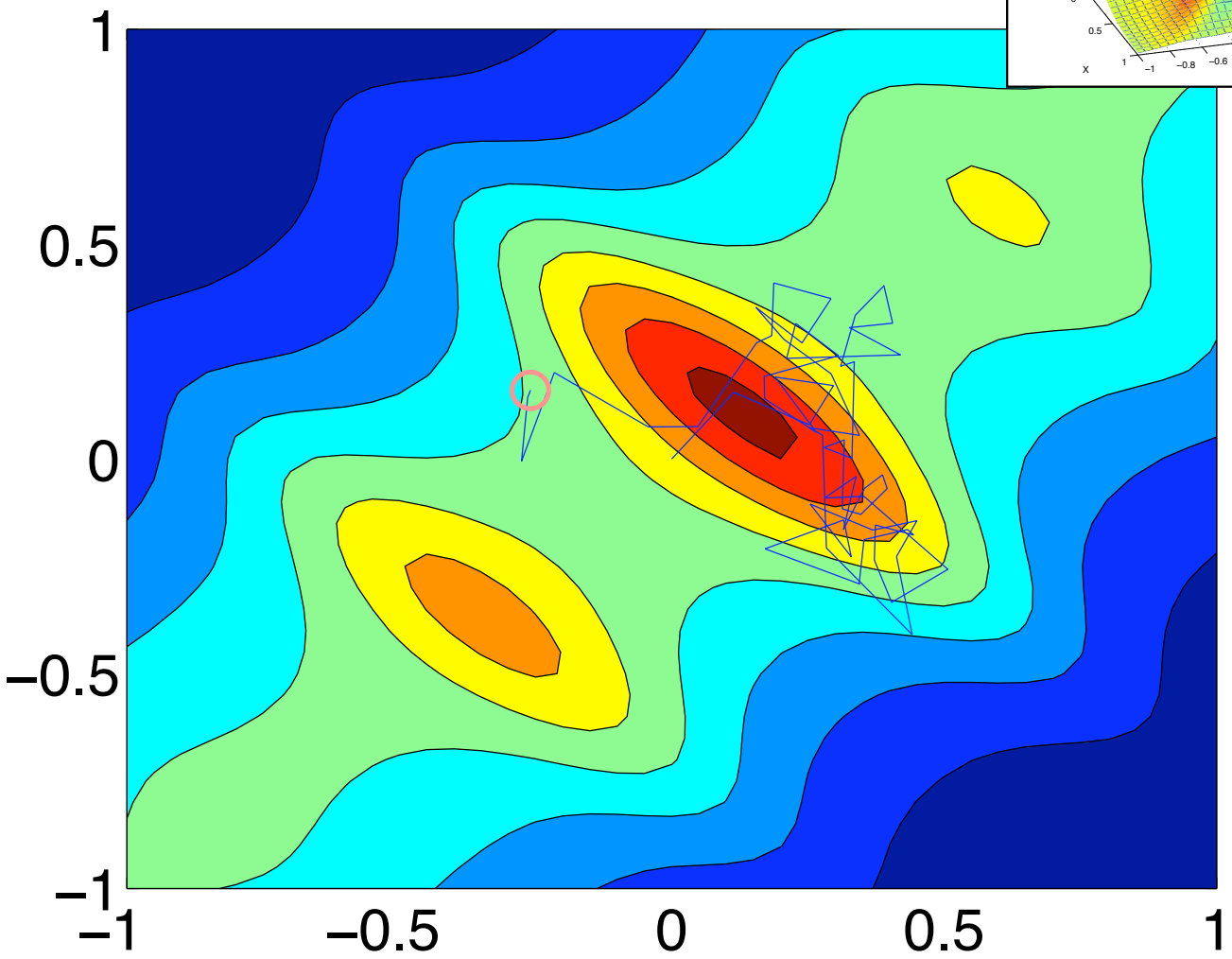
MH algorithm

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





MH considerations

- *Acceptance rate*
- *Mixing rate = 1 / mixing time*
- *Annealing (start at high temperature, reduce to T=1)*

MH proof

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

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

- *Proof based on **detailed balance***
- *If we can show detailed balance, P(x) is our stationary distribution:*
  - *take integral dx on both sides*
  - *use law of total probability on RHS*

$$\int P(x) T(x'|x) dx = \int P(x') T(x|x') dx$$

$$= P(x') \int T(x|x') dx$$

$$P(x) T(x'|x)$$

$$= P(x) Q(x'|x) p(x'|x)$$

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

$$= \begin{cases} P(x) Q(x'|x) & \text{case 1} \\ P(x') Q(x|x') & \text{case 2} \end{cases}$$

$$P(x')\,T(x \mid x')$$

$$= P(x')\,Q(x \mid x')\,\beta(x \mid x')$$

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

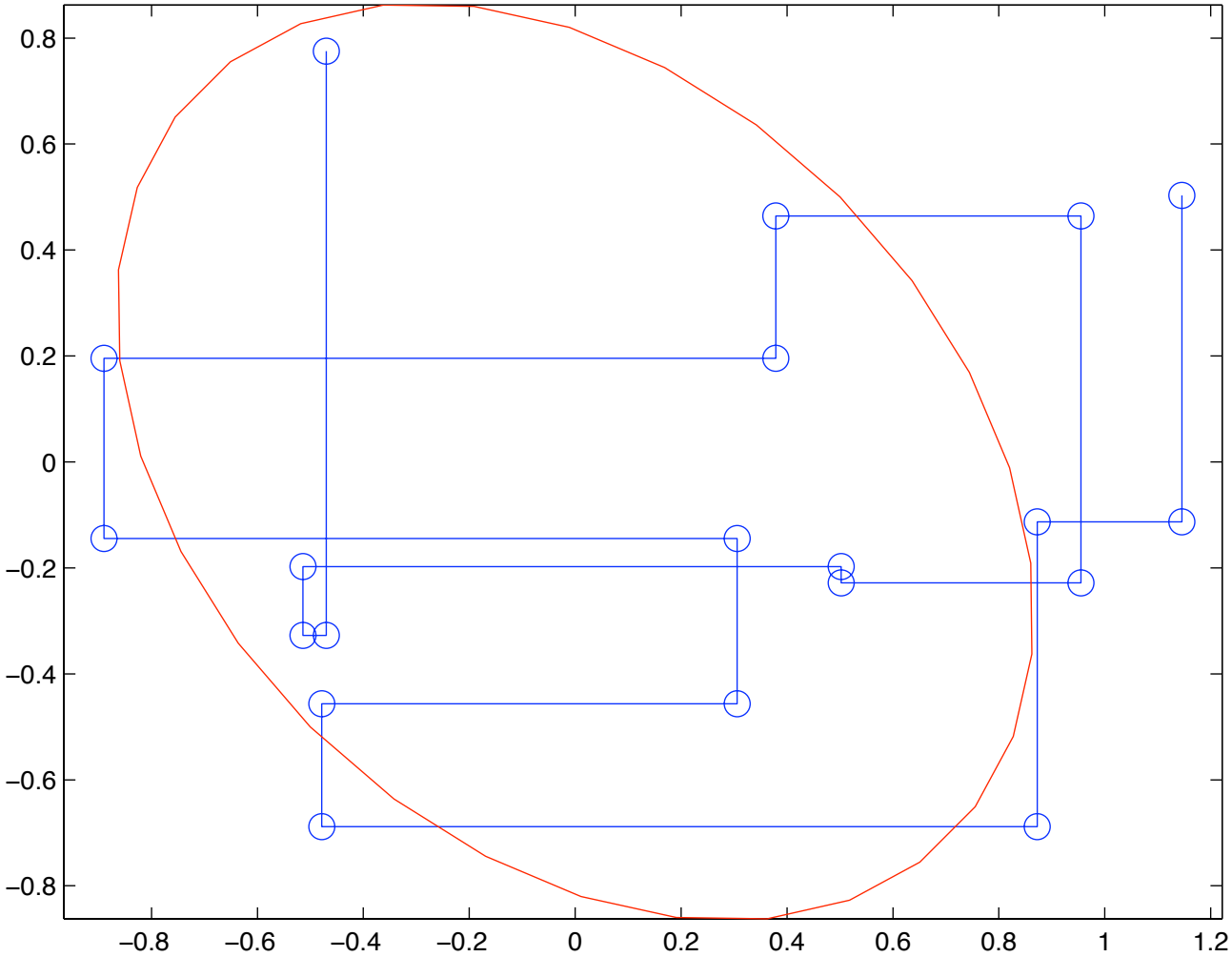
$$= \begin{cases} P(x')\,Q(x \mid x') & \text{case 2} \\ P(x)\,Q(x' \mid x) & \text{case 1} \end{cases}$$

Gibbs

Gibbs sampler

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

Gibbs example



Why is Gibbs useful?

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

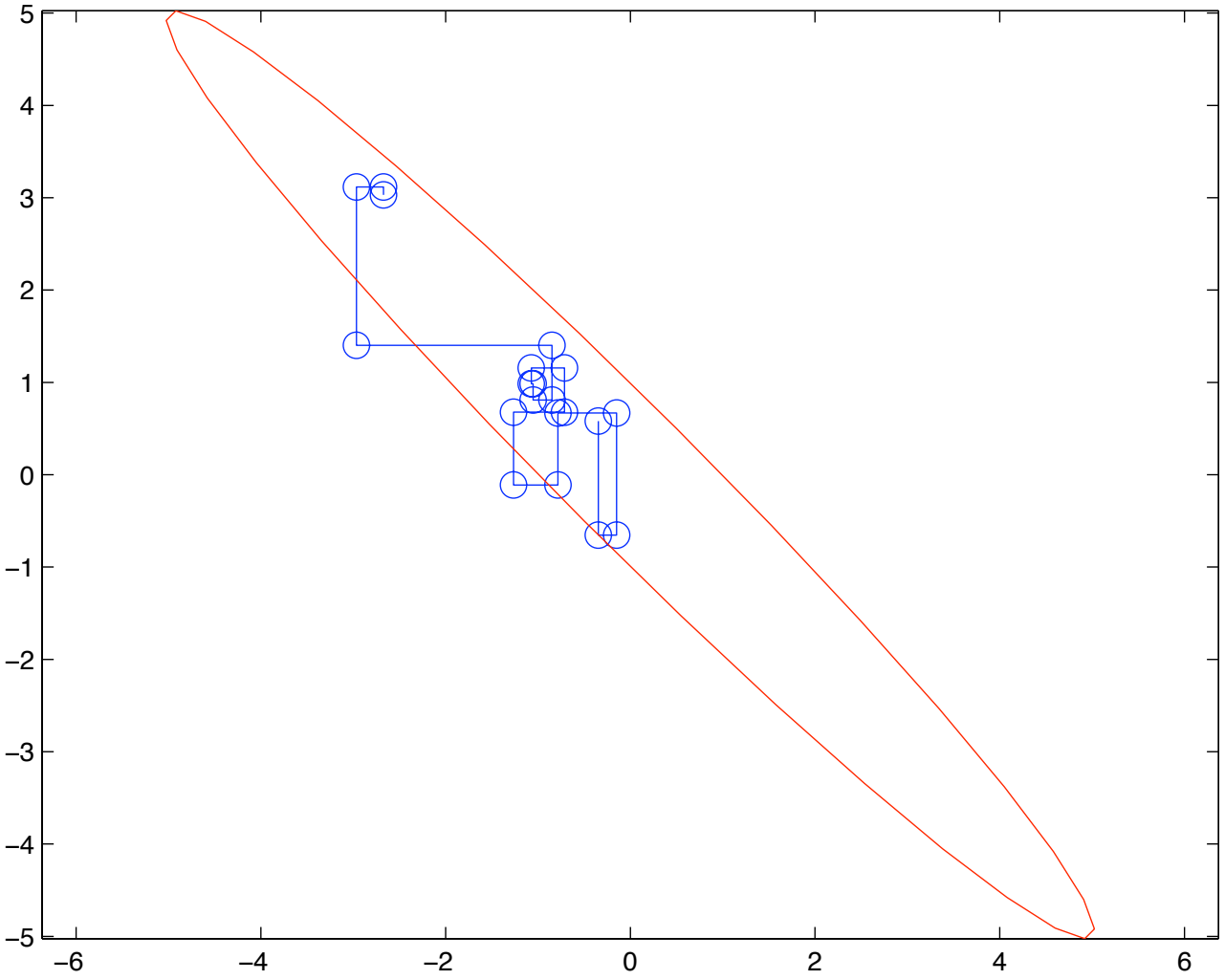
Gibbs derivation

$$
\begin{aligned}
& \frac{P(x'_i, x'_{\neg i})}{P(x_i, x_{\neg i})} \frac{P(x_i \mid x'_{\neg i})}{P(x'_i \mid x_{\neg i})} \\
= \quad & \frac{P(x'_i, x_{\neg i})}{P(x_i, x_{\neg i})} \frac{P(x_i \mid x_{\neg i})}{P(x'_i \mid x_{\neg i})} \\
= \quad & \frac{P(x'_i, x_{\neg i})}{P(x_i, x_{\neg i})} \frac{P(x_i, x_{\neg i})/P(x_{\neg i})}{P(x'_i, x_{\neg i})/P(x_{\neg i})} \\
= \quad & 1
\end{aligned}
$$

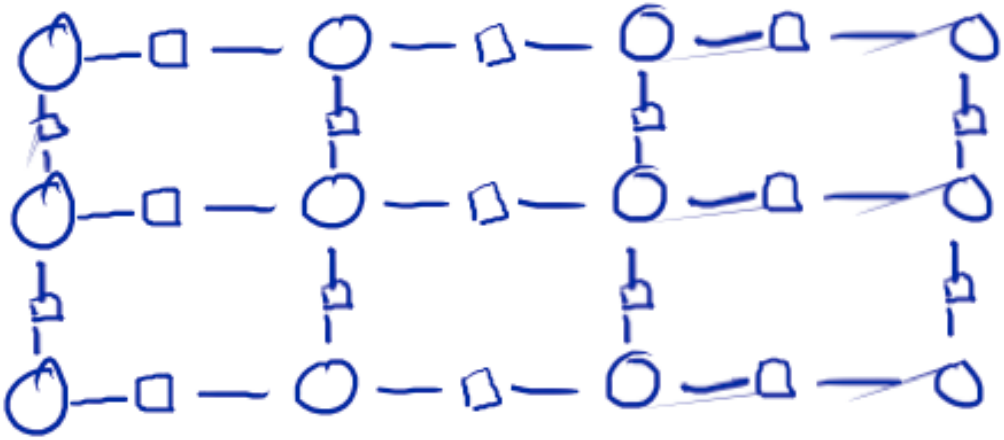
Gibbs in practice

- *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



Sequential sampling



- *In an HMM or DBN, to sample $P(\mathbf{X}_T)$, start from $\mathbf{X}_1$ and sample forward step by step*
  - $\mathbf{X}_{t+1} \sim P(\mathbf{X}_{t+1} \mid \mathbf{X}_t)$
- $P(\mathbf{X}_{1:T}) = P(\mathbf{X}_1)\, P(\mathbf{X}_2 \mid \mathbf{X}_1)\, P(\mathbf{X}_3 \mid \mathbf{X}_2) \ldots$

Particle filter

- *Can sample* $\mathbf{X}_{t+1} \sim P(\mathbf{X}_{t+1} \mid \mathbf{X}_t)$ *using any algorithm from above*
- *If we use* ***parallel importance sampling*** *to get N samples at once from each* $P(\mathbf{X}_t)$*, we get a* ***particle filter***
- *Write* $\mathbf{x}_{t,i}$ $(i = 1 \ldots N)$ *for sample at time t*

Particle filter

- *Want one sample from each of* $P(\mathbf{X}_{t+1} \mid \mathbf{x}_{t,i})$
- *Have only* Z $P(\mathbf{X}_{t+1} \mid \mathbf{x}_{t,i})$
- *For each* i*, pick* $\mathbf{x}_{t+1,i}$ *from proposal* $Q(x)$
- *Compute unnormalized importance weight*

$$\hat{w}_i = ZP(\mathbf{x}_{t+1,i} \mid \mathbf{x}_{t,i})/Q(\mathbf{x}_{t+1,i})$$

Particle filter

- *Normalize weights:*

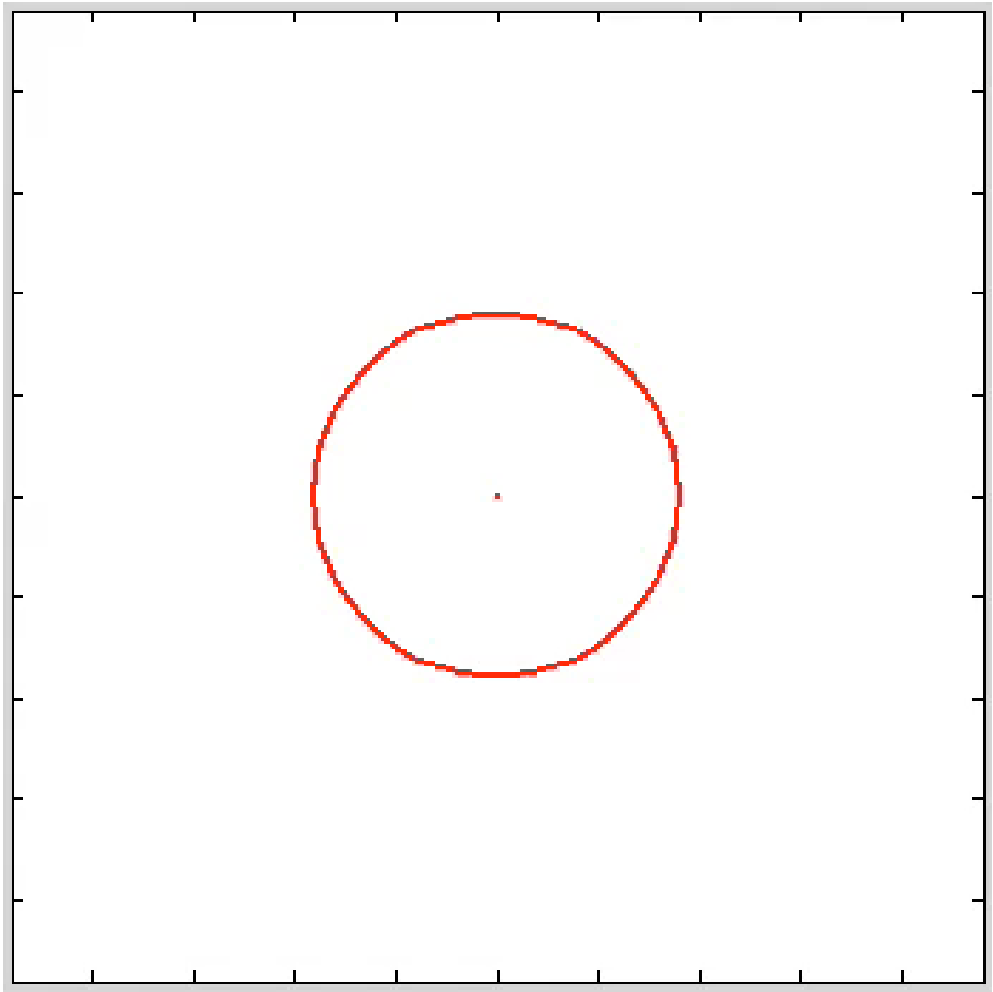
$$\bar{w} = \frac{1}{N}\sum_i \hat{w}_i \qquad w_i = \hat{w}_i / \bar{w}$$

- *Now, (w_i, $\mathbf{x}_{t+1,i}$) is an approximate* ***weighted*** *sample from P($\mathbf{X}_{t+1}$)*
- *To get an unweighted sample,* ***resample***

Resampling

- *Sample N times (with replacement) from* $\boldsymbol{x}_{t+1,i}$ *with probabilities* w_i/N
  - *alternate method: deterministically take floor*(w_i) *copies of* $\boldsymbol{x}_{t+1,i}$ *and sample only from* $[w_i - \text{floor}(w_i)]$
- *Each* $\boldsymbol{x}_{t+1,i}$ *appears* w_i *times on average, so we're still a sample from* $P(\boldsymbol{X}_{t+1})$

Particle filter example



Learning

Learning

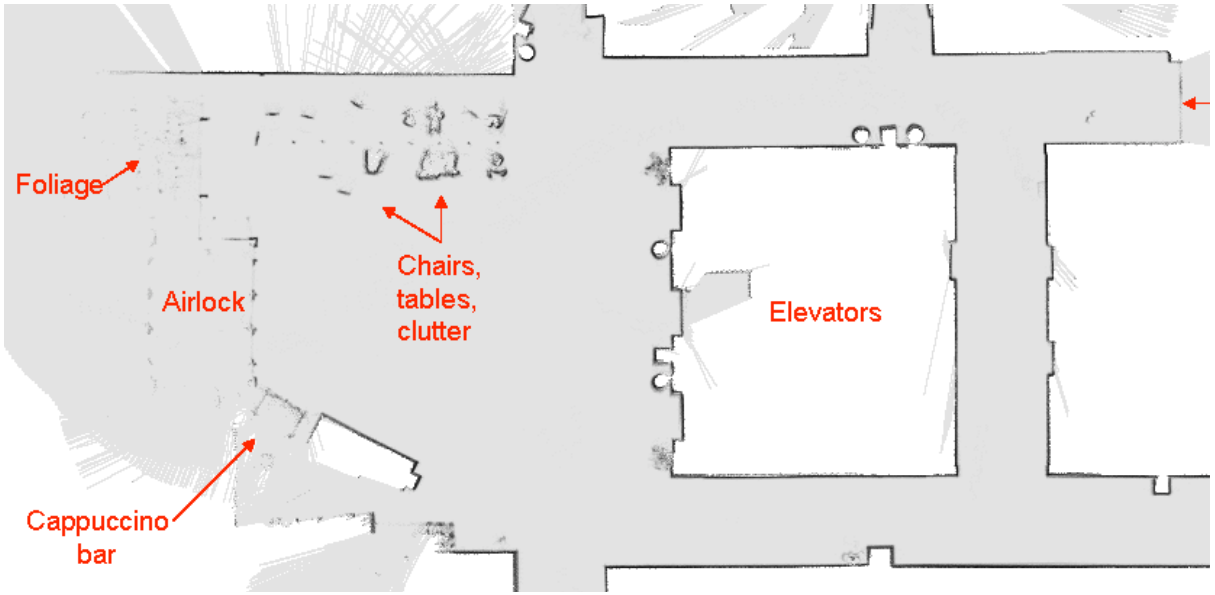
- *So far we've assumed our model of the world (factor graph, or SAT formula, or MILP, etc.) was given*
- *In fact, one of the most important attributes of an intelligent agent is that it* ***learns*** *from experience*

Learning

- *Basic learning problem: given some experience, find a new or improved model*
- *Experience: a sample* $x_1, \ldots, x_N$
- *Model: want to predict* $x_{N+1}, \ldots$

Example

- *Experience = range sensor readings & odometry from robot*
- *Model = map of the world*



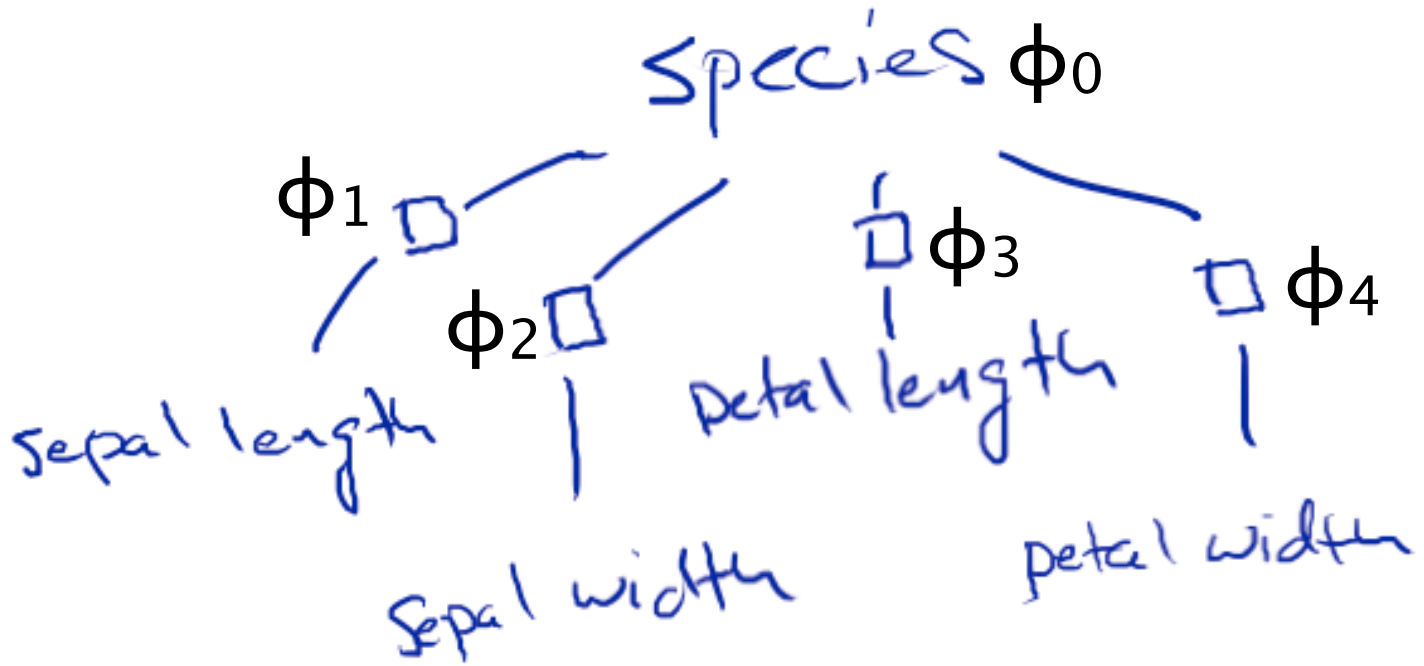
Example

- *Experience = physical measurements of surveyed specimens & expert judgements of their true species*
- *Model = factor graph relating species to measurements*

Sample data

sepal length	*sepal width*	*petal length*	*petal width*	*species*
5.1	*3.5*	*1.4*	*0.2*	*Iris setosa*
5.6	*3.0*	*4.5*	*1.5*	*Iris versicolor*
4.9	*3.0*	*1.4*	*0.2*	*Iris setosa*
6.4	*2.8*	*5.6*	*2.1*	*Iris virginica*
5.8	*2.7*	*4.1*	*1.0*	*Iris versicolor*

Factor graph



- *One of many possible factor graphs*
- *Values of Φs not shown, but part of model*

In general

- *For our purposes, a model is exactly a joint distribution P(**X**) over possible samples*

Comparing models

- *When is a model P(**X**) **better than** another model P'(**X**)?*
- *Need to make future decisions based on model; better model = better decisions*
- *E.g., suppose robot runs on I. versicolor, but I. setosa is poisonous to it*
- *Knowing P(I. setosa | petal length = 4.2) lets us weigh risks of eating specimen*

The problem

- *We don't know what future examples we'll see, or what decisions we'll have to make about them*
- *So, we'll use various proxies*

Conditional model

- *Split variables into* $(\mathbf{X}, \mathbf{Y})$
- *Suppose we always observe* $\mathbf{X}$
- *Two ways* $P(\mathbf{X}, \mathbf{Y})$ *and* $P'(\mathbf{X}, \mathbf{Y})$ *can differ:*
  - $P(\mathbf{X}) \neq P'(\mathbf{X})$, *and/or*
  - $P(\mathbf{Y} \mid \mathbf{X}) \neq P'(\mathbf{Y} \mid \mathbf{X})$
- *First way doesn't matter for decisions*
- ***Conditional model****: only specifies* $P(\mathbf{Y} \mid \mathbf{X})$

Conditional model example

- *Experience = samples of (**X**, **Y**)*
- ***X** = features of object*
- ***Y** = whether object is a "framling"*
- *Model = rule for deciding whether a new object is a framling*

Sample data & possible model

tall	*pointy*	*blue*	*framling*
T	*T*	*F*	*T*
T	*F*	*F*	*T*
F	*T*	*F*	*F*
T	*T*	*T*	*F*
T	*F*	*F*	*T*

$$H = tall \wedge \neg blue$$

Hypothesis space

- *Hypothesis space $\mathcal{H}$ = set of models we are willing to consider*
  - *for philosophical or computational reasons*
- *E.g., all factor graphs of a given structure*
- *Or, all conjunctions of up to two literals*

A simple learning algorithm

- *Conditional learning: samples* $(\boldsymbol{x}_i, y_i)$
- *Let* $\mathscr{H}$ *be a set of propositional formulae*
  - $\mathscr{H} = \{ H_1, H_2, \ldots \}$
- *H is* ***consistent*** *if* $H(\boldsymbol{x}_i) = y_i$ *for all i*
- ***Version space*** $V = \{$ *all consistent H* $\} \subseteq \mathscr{H}$
- ***Version space algorithm****: predict y = majority vote of* $H(\boldsymbol{x})$ *over all* $H \in V$

Framlings

tall	*pointy*	*blue*	*framling*
T	*T*	*F*	*T*
T	*F*	*F*	*T*
F	*T*	*F*	*F*
T	*T*	*T*	*F*
T	*F*	*F*	*T*

- $\mathcal{H}$ = { *conjunctions of up to 2 literals* } = { *T*, *F*, *tall*, *pointy*, *blue*, ¬*tall*, ¬*pointy*, ¬*blue*, *tall* ∧ *pointy*, *tall* ∧ *blue*, *pointy* ∧ *blue*, ¬*tall* ∧ *pointy*, … }

Analysis

- ***Mistake*** *= make wrong prediction*
- *If some* $H \in \mathscr{H}$ *is always right, eventually we'll eliminate all competitors, and make no more mistakes*
- *If no* $H \in \mathscr{H}$ *is always right, eventually V will become empty*
  - *e.g., if* ***label noise*** *or* ***feature noise***

Analysis

- *Suppose* $|\mathcal{H}| = N$
- *How many mistakes could we make?*

Analysis

- *Suppose* $|\mathcal{H}| = N$
- *How many mistakes could we make?*
- *Since we predict w/* ***majority*** *of V, after any mistake, we eliminate half (or more) of V*
- *Can't do that more than* $\log_2(N)$ *times*

Discussion

- *In example, $N = 20$, $\log_2(N) = 4.32$*
- *Made only 2 mistakes*
- *Mistake bound: limits wrong decisions, as desired*
- *But, required strong assumptions (no noise, true H contained in $\mathcal{H}$)*
- *Could be very slow!*