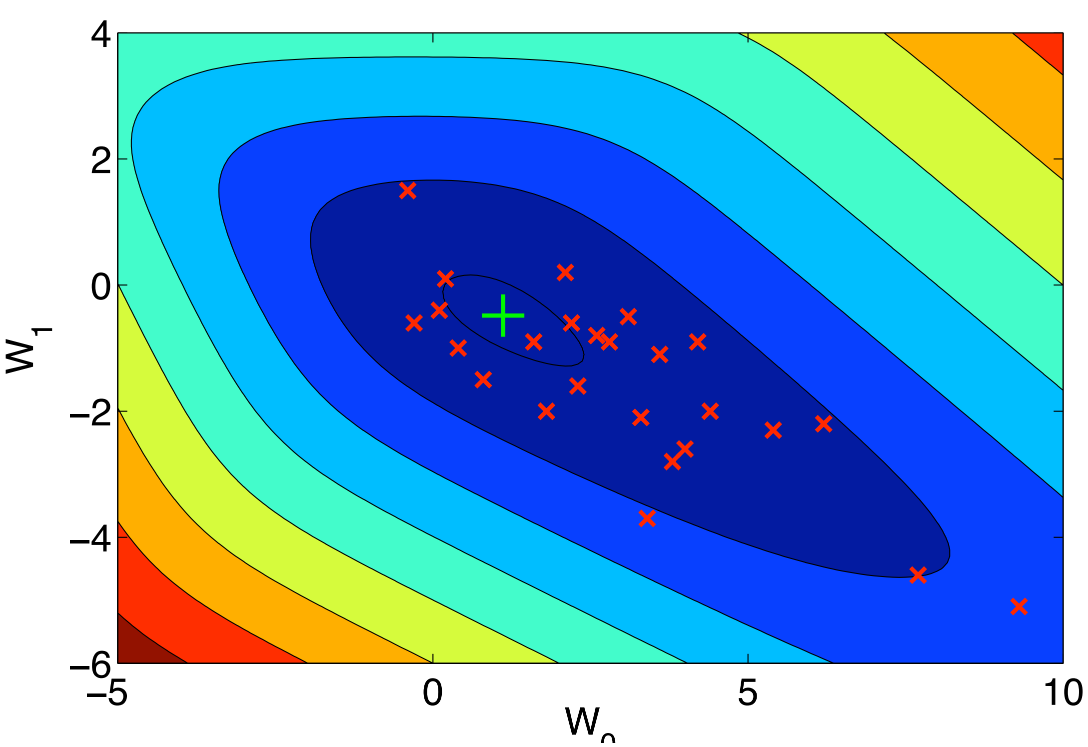
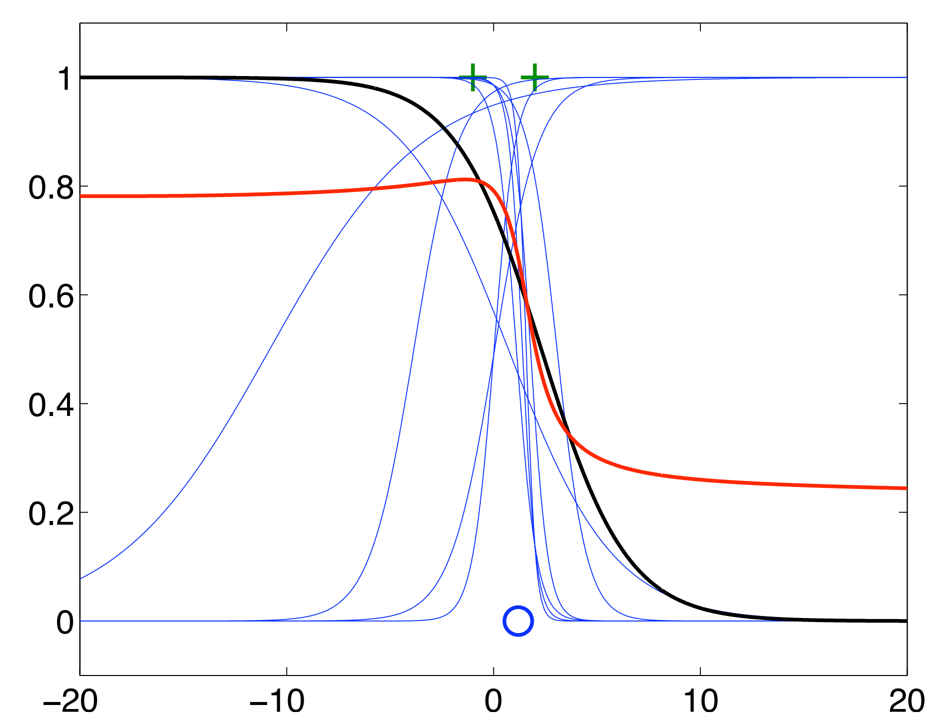
Review

- Multiclass logistic regression
- Priors, conditional MAP logistic regression
- Bayesian logistic regression
  - MAP is not always typical of posterior
  - posterior predictive can avoid overfitting





Review

- Finding posterior predictive distribution often requires numerical integration
  - uniform sampling
  - importance sampling
  - parallel importance sampling
- These are all ***Monte-Carlo algorithms***
  - another well-known MC algorithm coming up

Application: SLAM

Eliazar and Parr, IJCAI-03

Parallel IS

set $\hat{w}_i = Z P(x_i) / Q(x_i)$

$$\bar{w} = \frac{1}{N} \sum_{i=1}^{N} \hat{w}_i$$

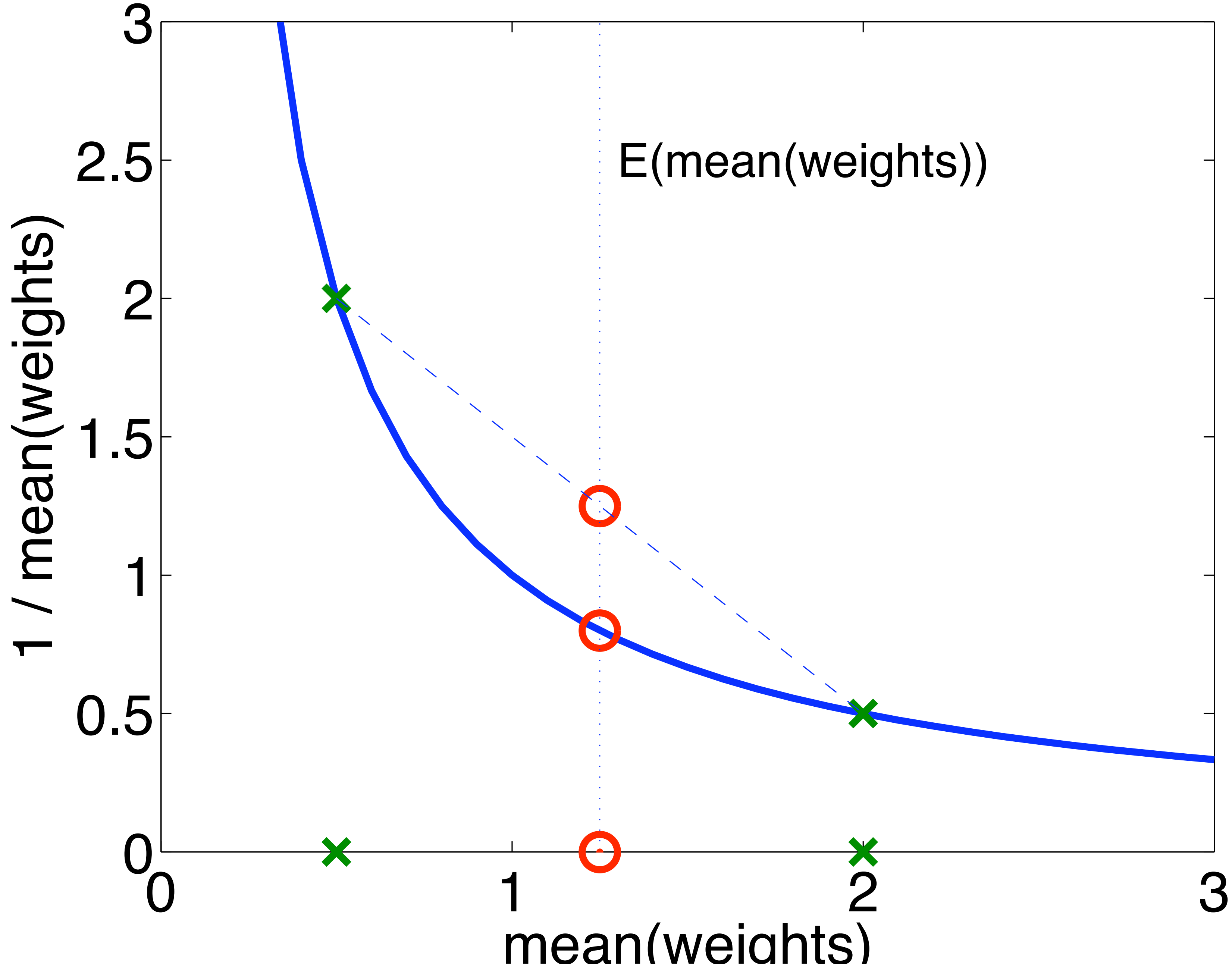
$$E(\hat{w}_i) = \int \cancel{Q(x)}\, Z P(x) / \cancel{Q(x)}\, dx$$

$$= Z \int P(x)\, dx = Z$$

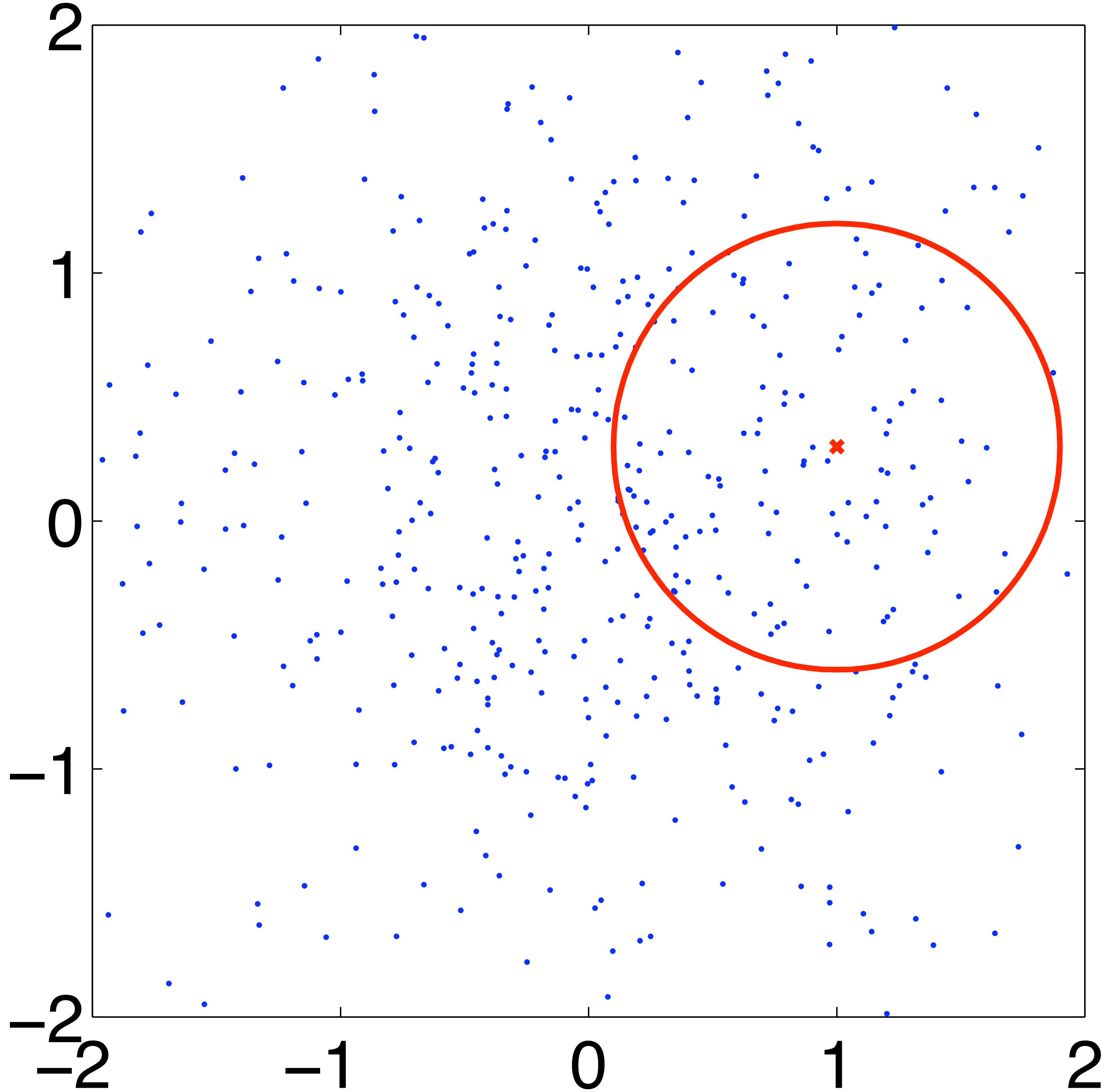
$$E(\bar{w}) = Z \quad \text{(lower variance)}$$

$$\mathbb{E}_P(g(X)) \approx \hat{G} = \frac{1}{N} \sum_{i=1}^{N} \underbrace{\frac{\hat{w}_i}{\bar{w}}}_{\text{normalized importance wts}} g(X_i)$$

Parallel IS is biased

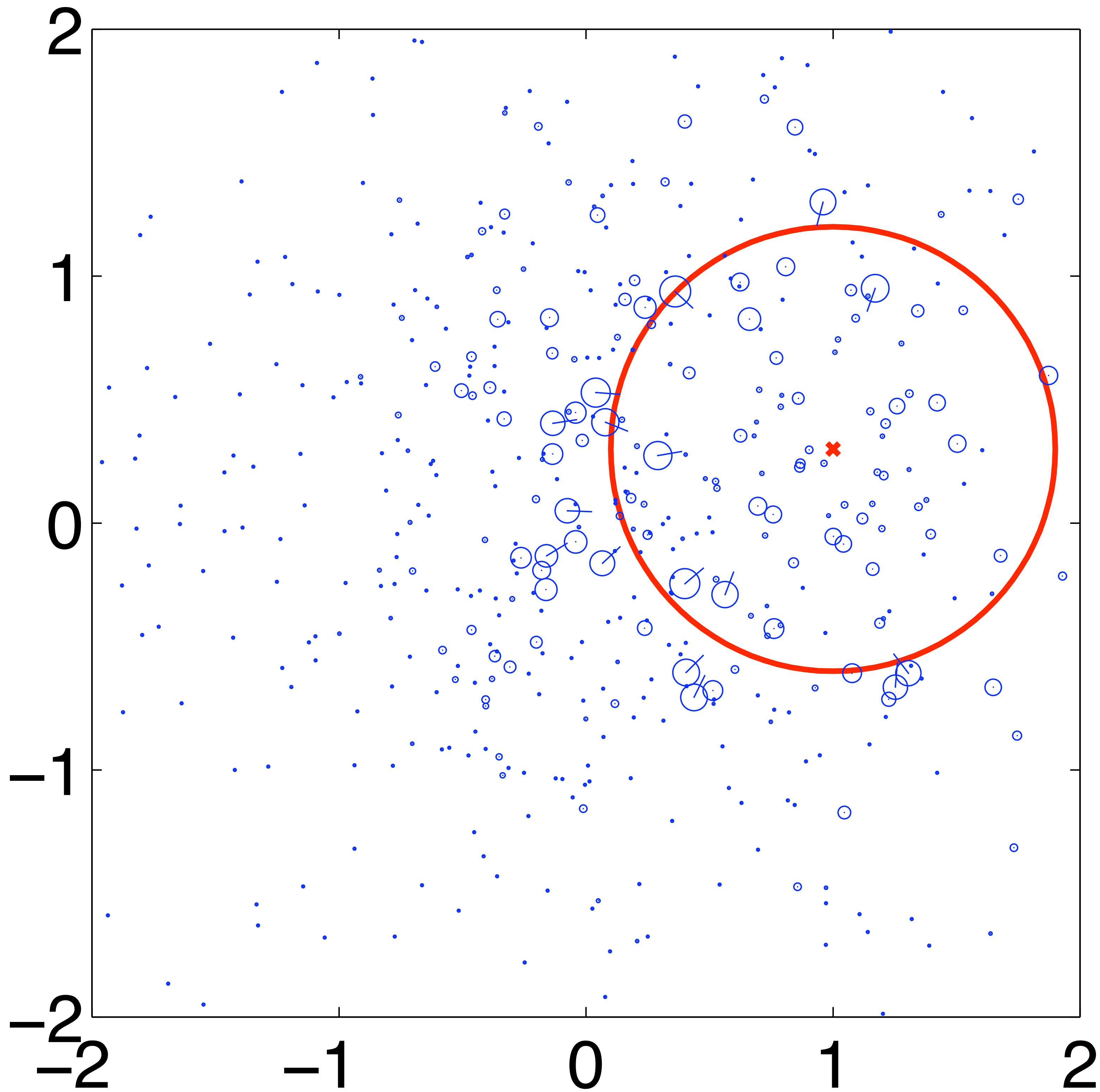


$$E(\bar{W}) = Z, \text{ but } E(1/\bar{W}) \neq 1/Z \text{ in general}$$



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

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

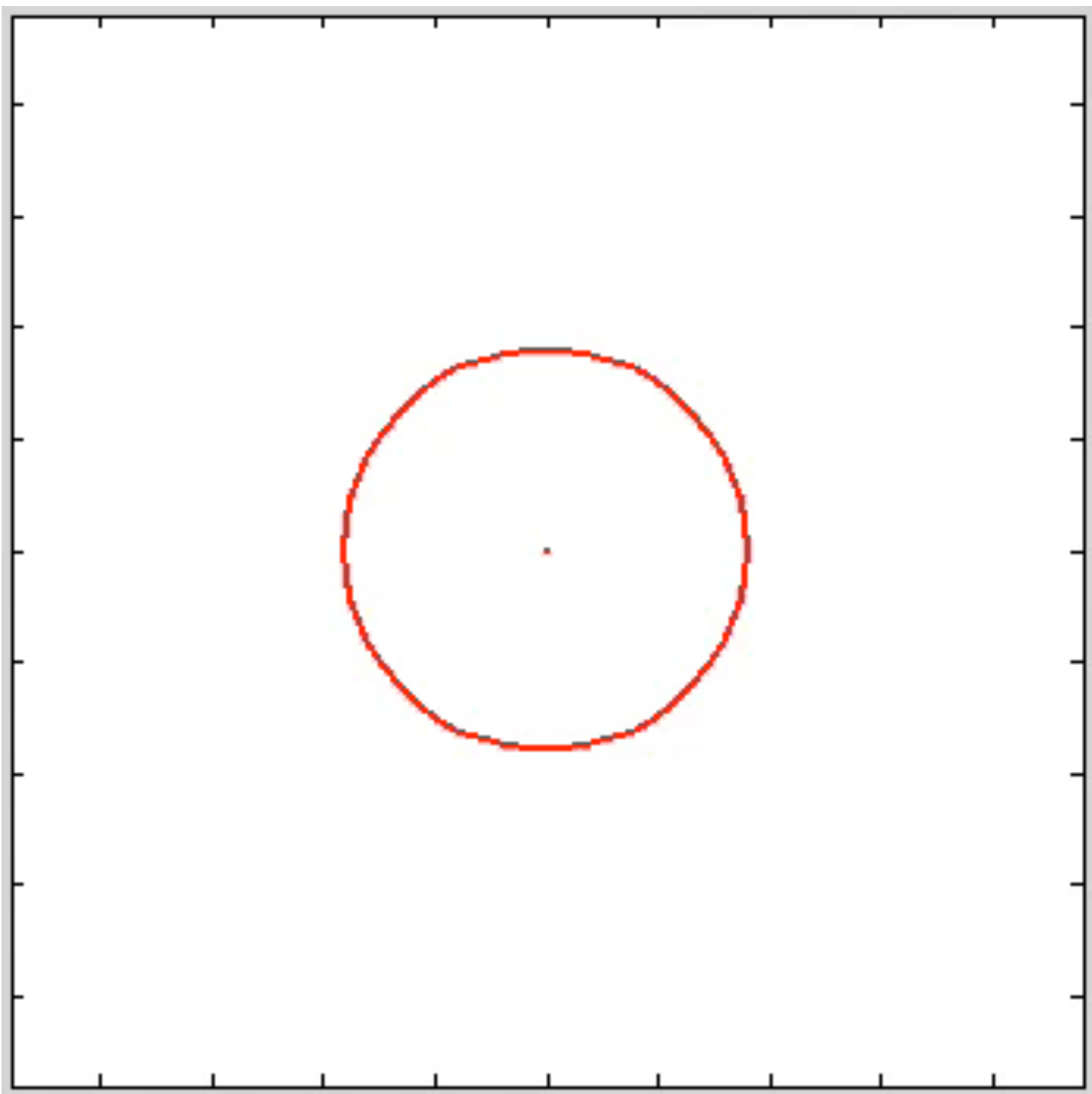


Posterior $E(X, Y, \theta) = (0.496, 0.350, 0.084)$

SLAM revisited

- Uses a recursive version of parallel importance sampling: ***particle filter***
  - each sample (particle) = trajectory over time
  - sampling extends trajectory by one step
  - recursively update importance weights and renormalize
  - resampling trick to avoid keeping lots of particles with low weights

Particle filter example



Monte-Carlo revisited

- Recall: wanted

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

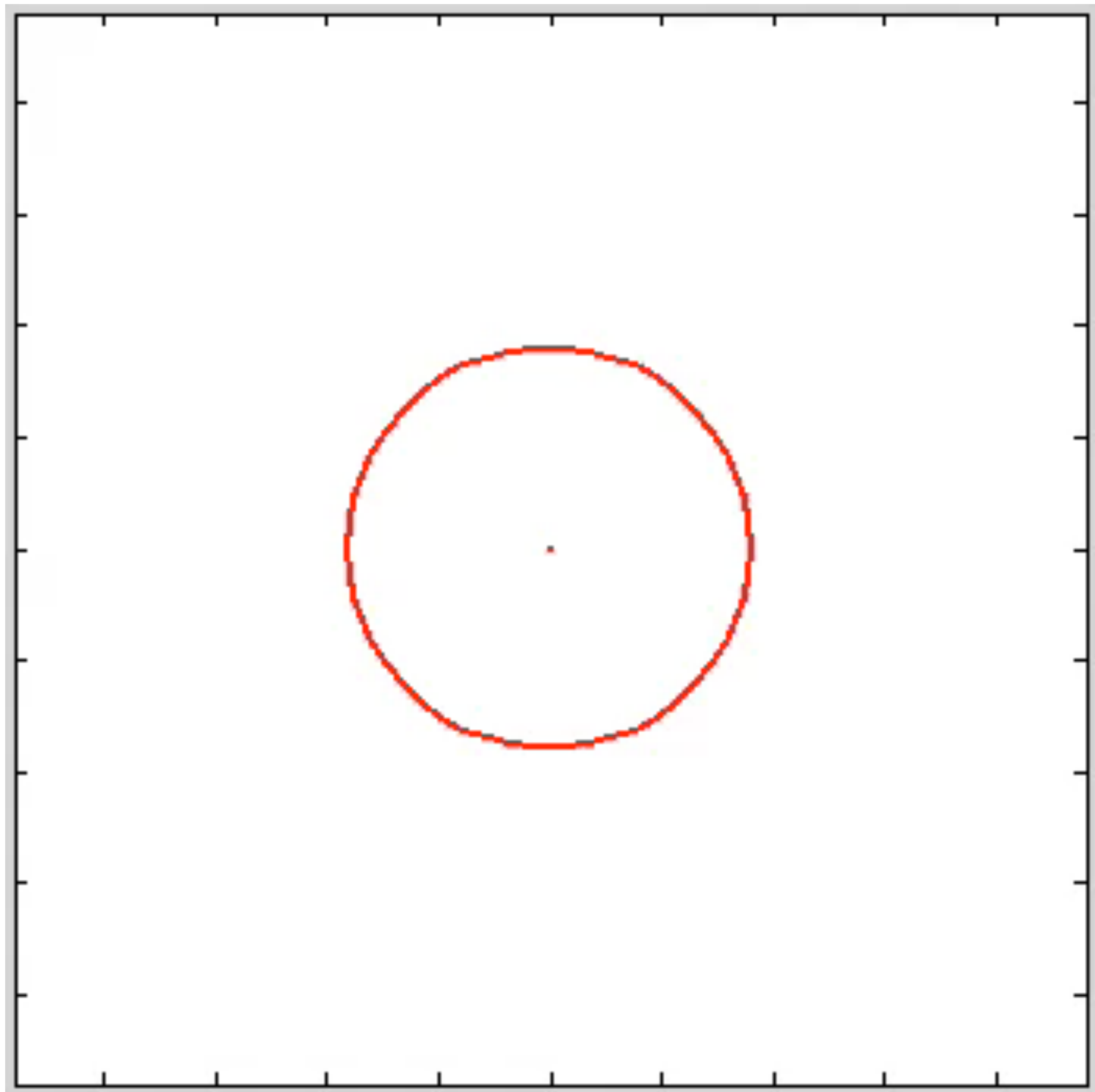
- Would like to search for areas of high P(x)
- But searching could bias our estimates

Markov-Chain Monte Carlo

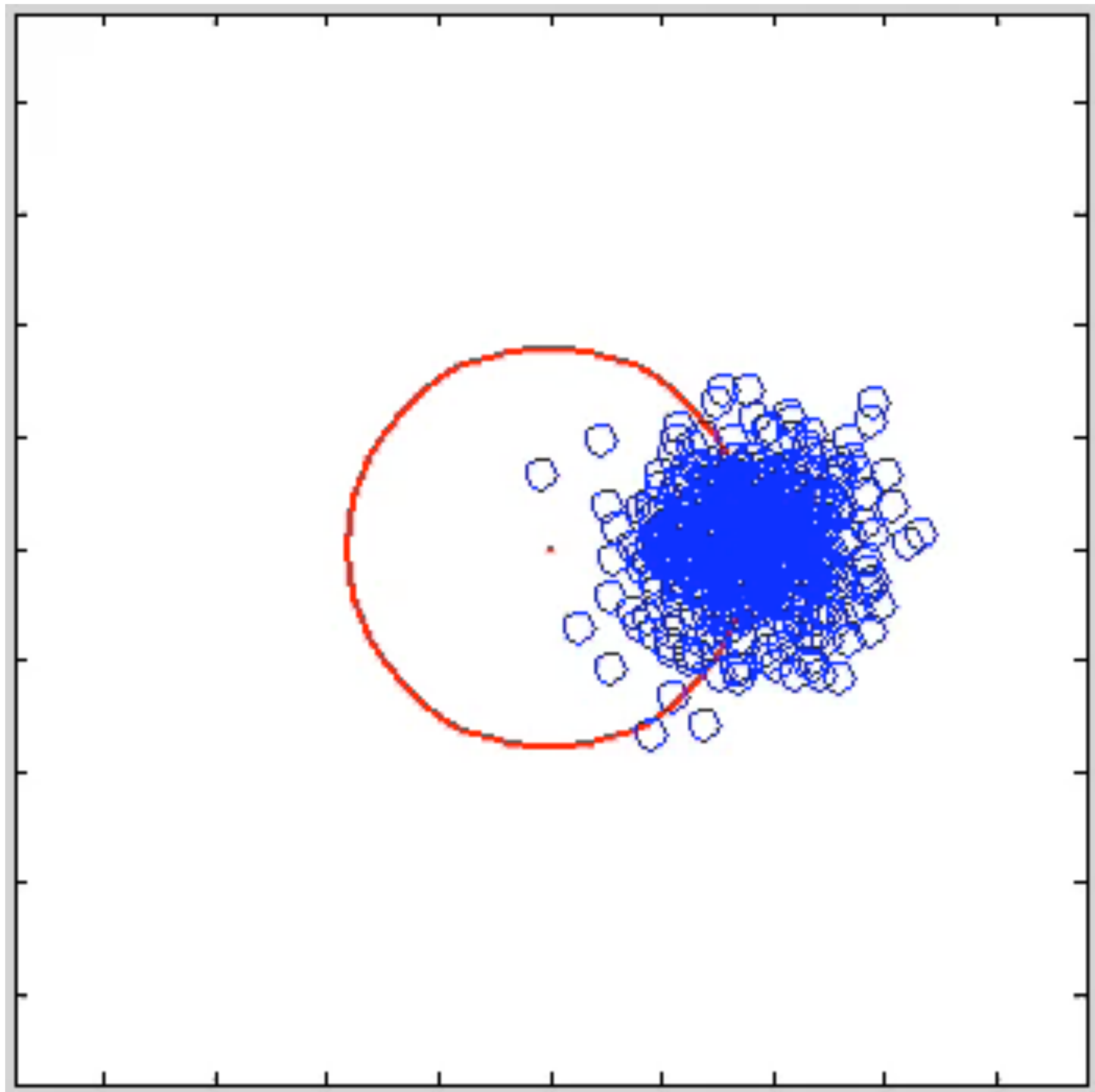
- Randomized search procedure
- Produces sequence of RVs $X_1, X_2, \ldots$
  - Markov chain: satisfies Markov property

- If $P(X_t)$ small, $P(X_{t+1})$ tends to be larger
- As $t \to \infty$, $X_i \sim P(X)$
- As $\Delta \to \infty$, $X_{t+\Delta} \perp X_t$

Markov chain



Stationary distribution



Markov-Chain Monte Carlo

- As $t \to \infty$, $X_i \sim P(X)$; as $\Delta \to \infty$, $X_{t+\Delta} \perp X_t$
- For big enough t and Δ, an approximately i.i.d. sample from P(X) is
  - $\{ X_t, X_{t+\Delta}, X_{t+2\Delta}, X_{t+3\Delta}, \ldots \}$
- Can use i.i.d. sample to estimate $E_P(g(X))$
- Actually, don't need independence:

Metropolis-Hastings

- Way to design chain w/ stationary dist'n P(X)
- Basic strategy: start from arbitrary X
- Repeatedly tweak X to get X'
  - If P(X') ≥ 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: $Q(X' \mid X)$

- Good proposals explore quickly, but remain in regions of high $P(X)$
- Optimal proposal?

MH algorithm

- Initialize X_1 arbitrarily
- For t = 1, 2, …:
  - Sample $X' \sim Q(X' \mid X_t)$
  - Compute p =
  - With probability min(1, p), set $X_{t+1} := X'$
    - else $X_{t+1} := X_t$
- Note: sequence $X_1, X_2, \ldots$ will usually contain duplicates

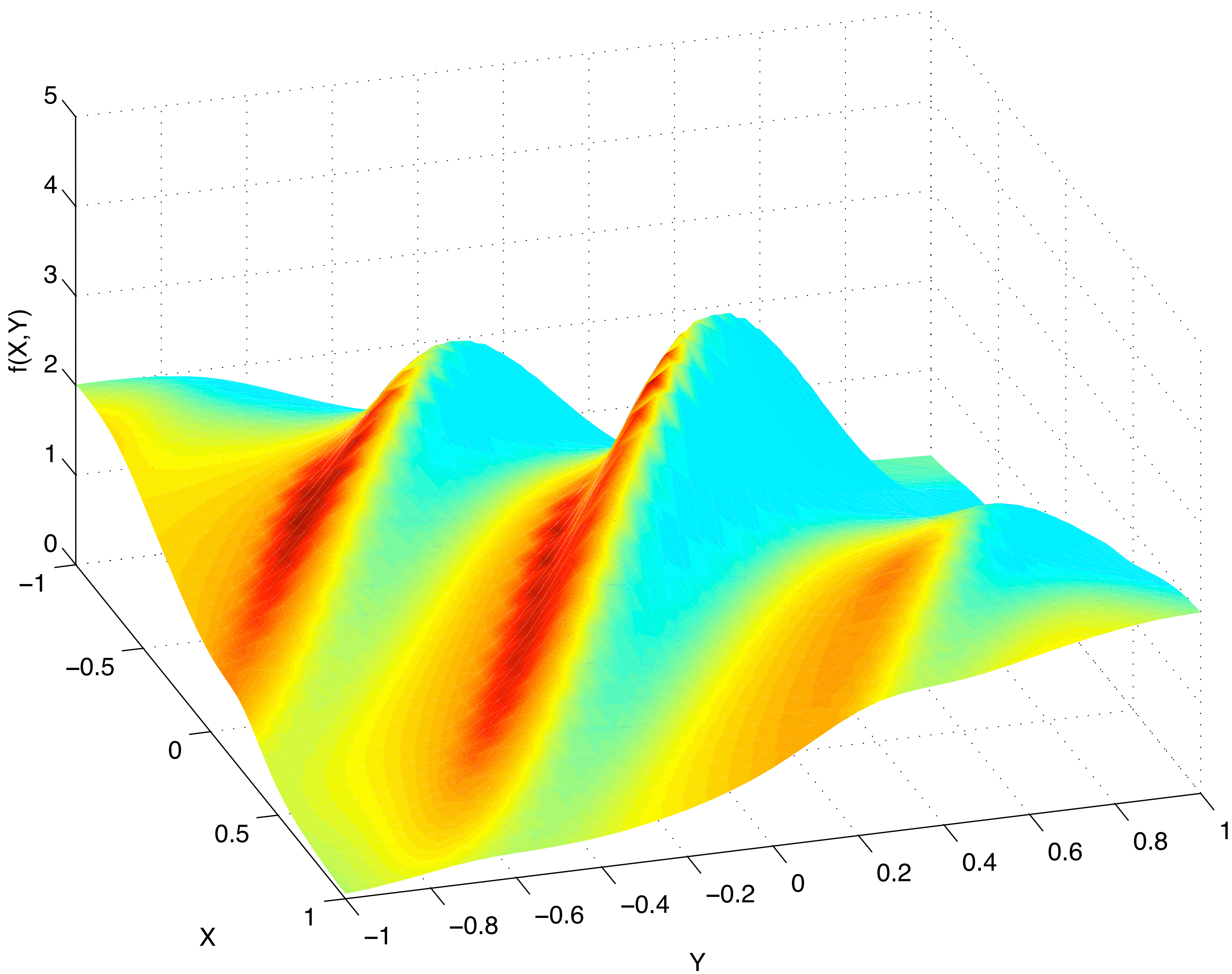
Acceptance rate

- 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 long moves and P(accept):

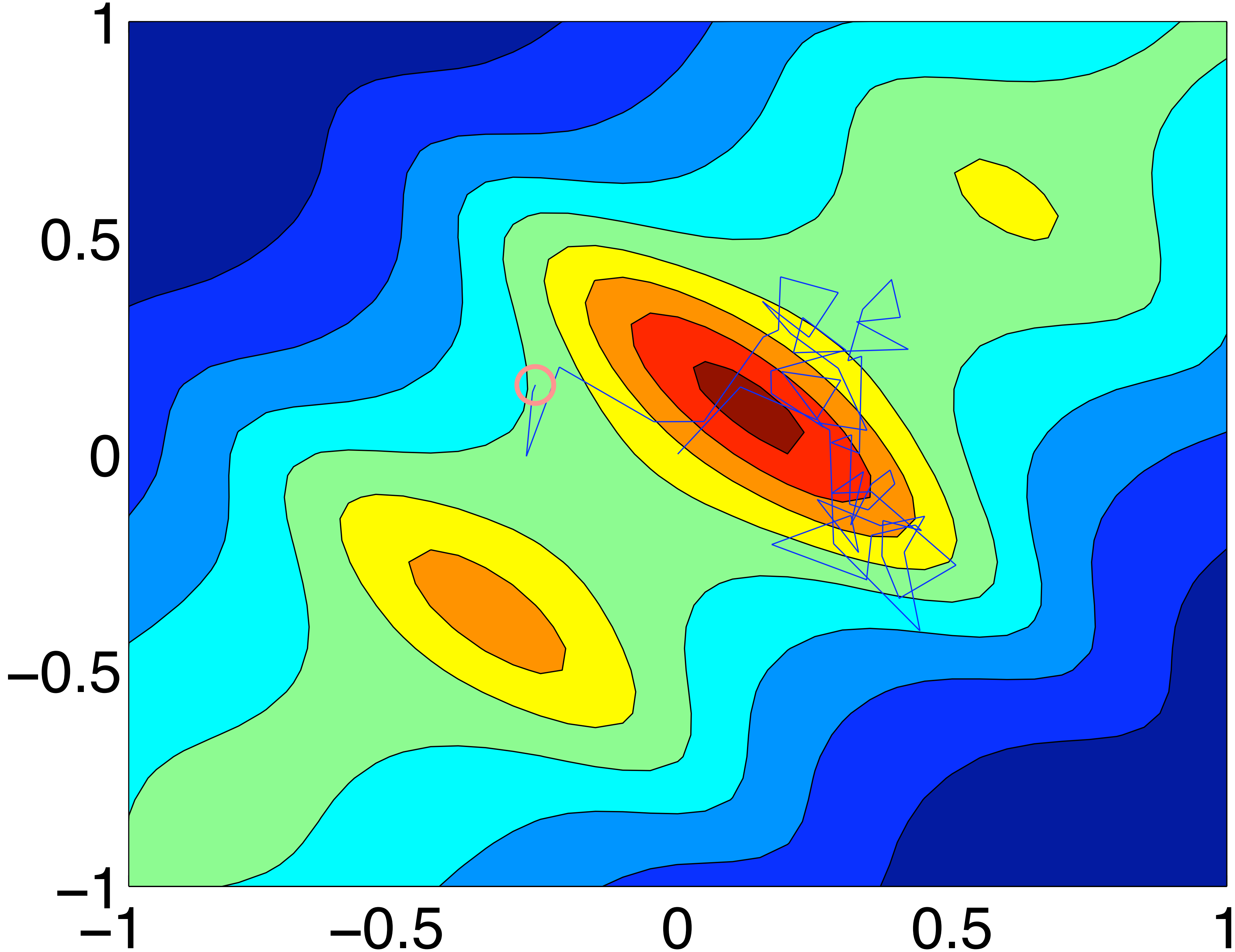
Mixing rate, mixing time

- 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 short **mixing time** = # steps until we can't tell which starting point we used
- **Mixing rate** = 1 / (mixing time)

MH example



MH example



In example

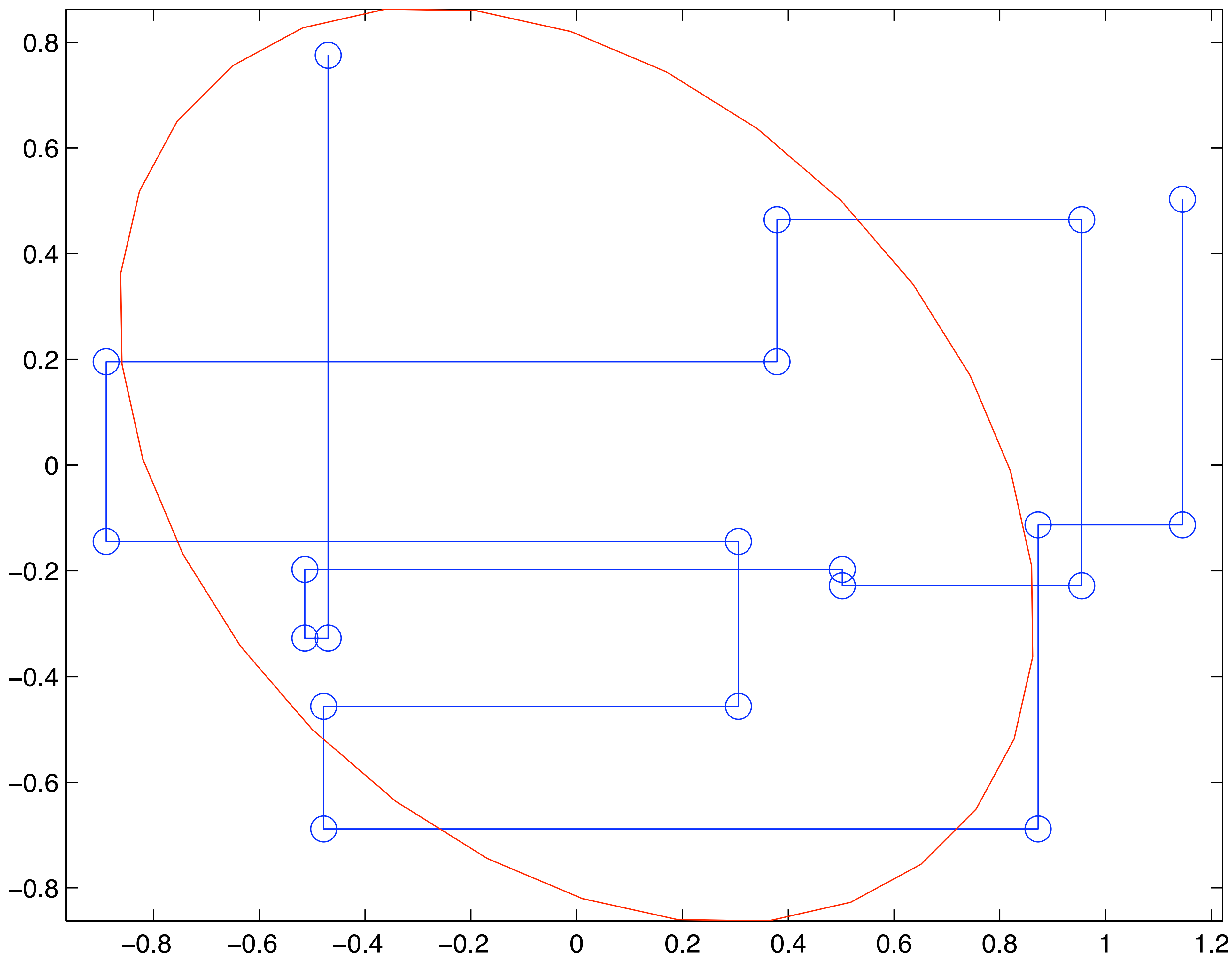
- $g(x) = x^2$
- True E(g(X)) = 0.28…
- Proposal: $Q(x' \mid x) = N(x' \mid x, 0.25^2 I)$
- Acceptance rate 55–60%
- After 1000 samples, minus burn-in of 100:

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

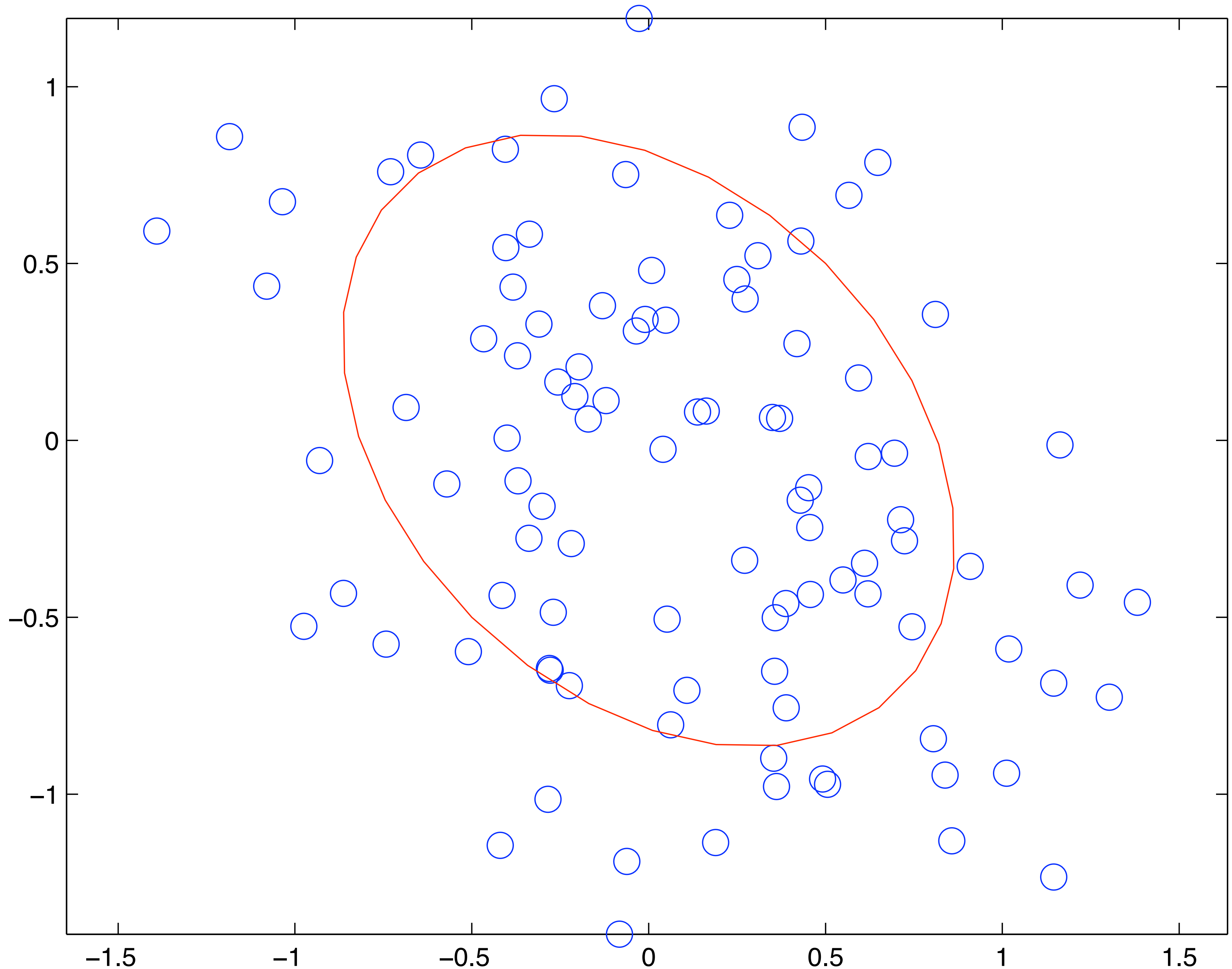
Gibbs sampler

- 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 $\mathbf{x}_{B(i)} \sim P(\mathbf{x}_{B(i)} \mid \mathbf{x}_{\neg B(i)})$
- Useful property: acceptance rate p = 1

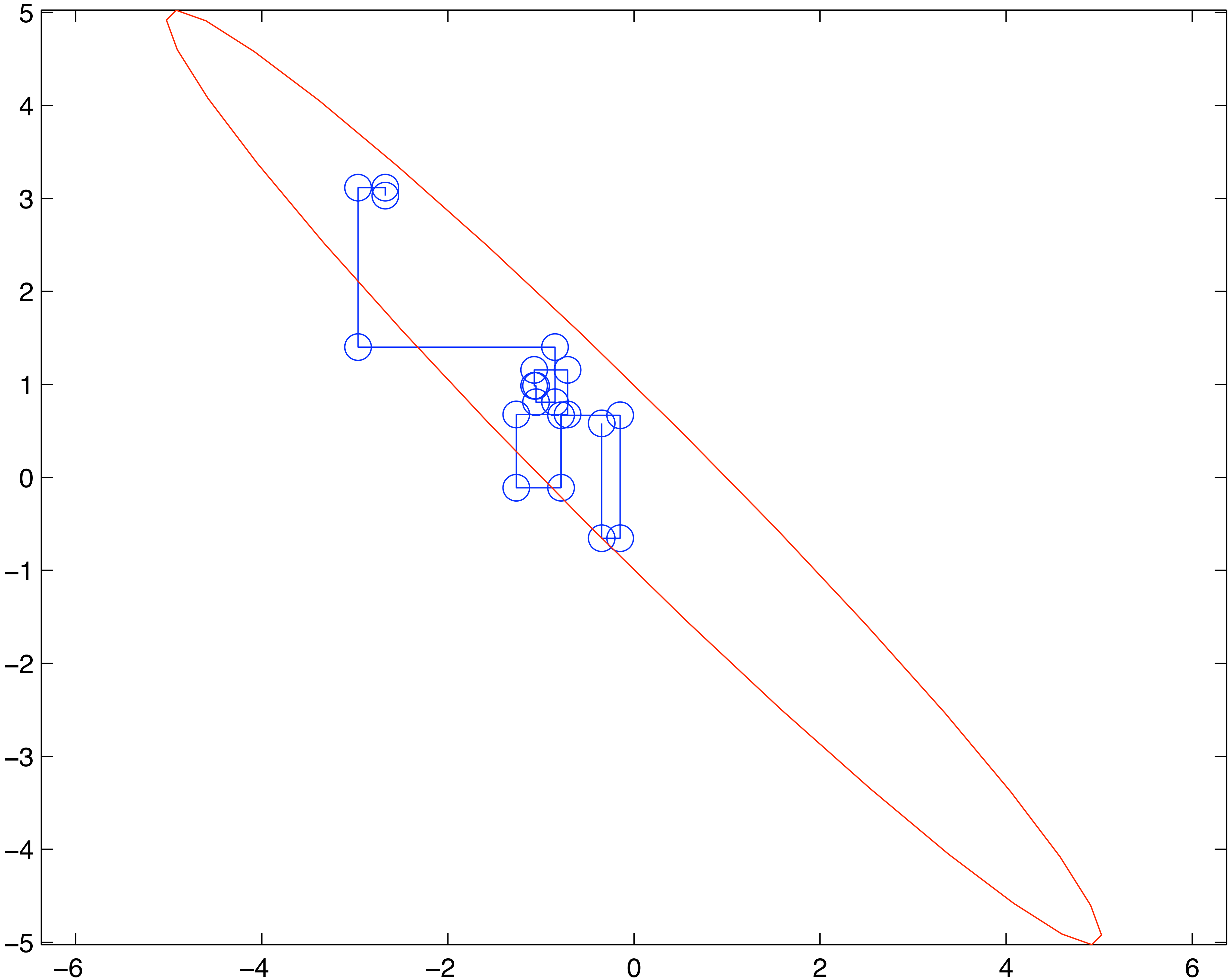
Gibbs example



Gibbs example



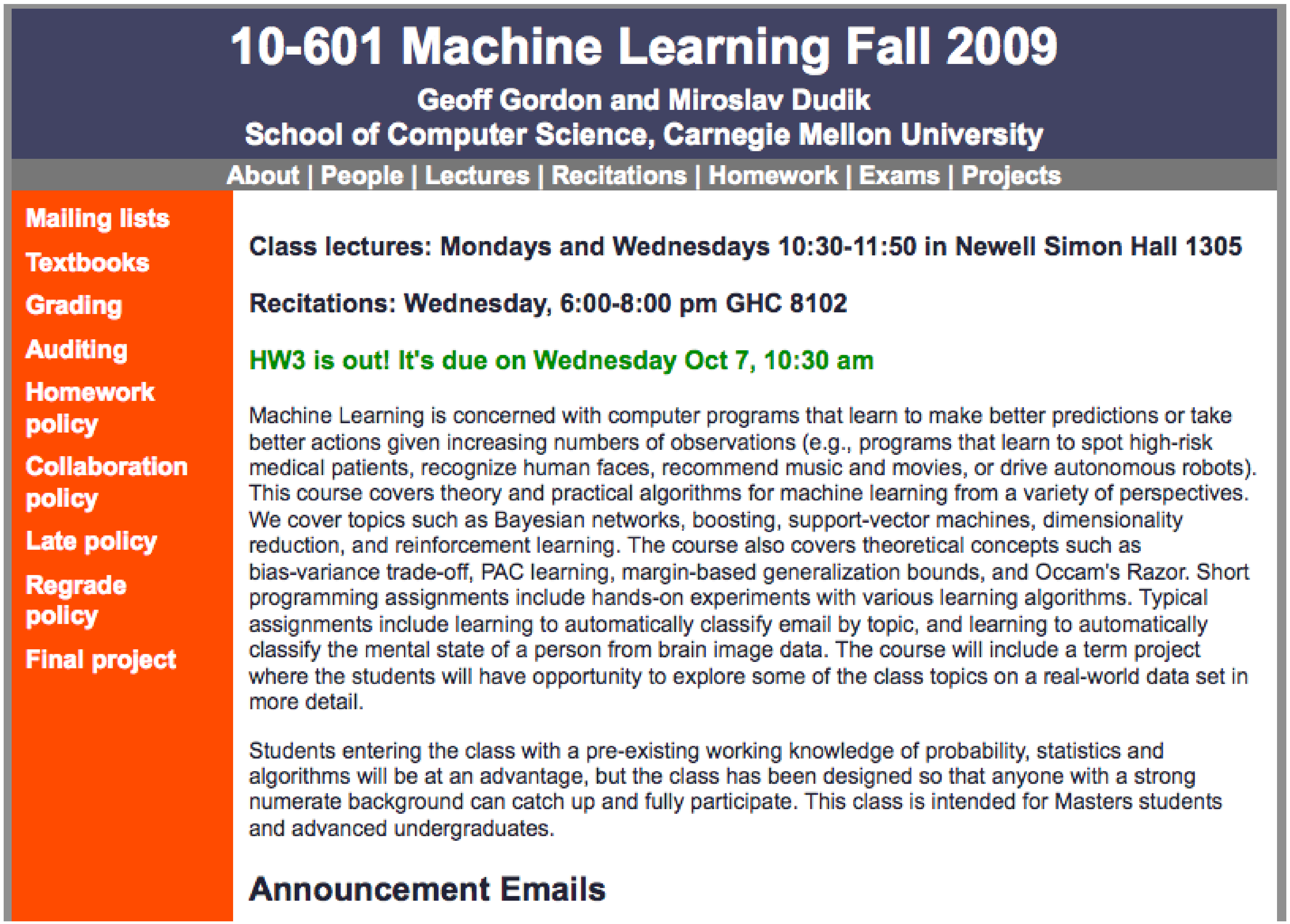
Gibbs failure example



Relational learning

- Linear regression, logistic regression: ***attribute-value*** learning
  - set of i.i.d. samples from P(X,Y)
- Not all data is like this
  - an attribute is a property of a single entity
  - what about properties of sets of entities?

Application: document clustering



10-601 Machine Learning Fall 2009

Geoff Gordon and Miroslav Dudik
School of Computer Science, Carnegie Mellon University

About | People | Lectures | Recitations | Homework | Exams | Projects

- Mailing lists
- Textbooks
- Grading
- Auditing
- Homework policy
- Collaboration policy
- Late policy
- Regrade policy
- Final project

Class lectures: Mondays and Wednesdays 10:30-11:50 in Newell Simon Hall 1305

Recitations: Wednesday, 6:00-8:00 pm GHC 8102

HW3 is out! It's due on Wednesday Oct 7, 10:30 am

Machine Learning is concerned with computer programs that learn to make better predictions or take better actions given increasing numbers of observations (e.g., programs that learn to spot high-risk medical patients, recognize human faces, recommend music and movies, or drive autonomous robots). This course covers theory and practical algorithms for machine learning from a variety of perspectives. We cover topics such as Bayesian networks, boosting, support-vector machines, dimensionality reduction, and reinforcement learning. The course also covers theoretical concepts such as bias-variance trade-off, PAC learning, margin-based generalization bounds, and Occam's Razor. Short programming assignments include hands-on experiments with various learning algorithms. Typical assignments include learning to automatically classify email by topic, and learning to automatically classify the mental state of a person from brain image data. The course will include a term project where the students will have opportunity to explore some of the class topics on a real-world data set in more detail.

Students entering the class with a pre-existing working knowledge of probability, statistics and algorithms will be at an advantage, but the class has been designed so that anyone with a strong numerate background can catch up and fully participate. This class is intended for Masters students and advanced undergraduates.

Announcement Emails

Application: recommendations

Latent-variable models

Best-known LVM: PCA

- Suppose X_{ij}, U_{ik}, V_{jk} all ~ Gaussian
  - yields ***principal components analysis***
  - or ***probabilistic PCA***
  - or ***Bayesian PCA***