



Computer  
Science

# CSC696H: Probabilistic Methods in ML

**Monte Carlo Methods**

Prof. Jason Pacheco

Some material from:  
Prof. Erik Sudderth & Prof. Kobus Barnard

# Administrative Items

- Sign up for paper presentation
- Create Github repository
  - Title “CSC969H Spring 2024 – <Name>”
  - Add Markdown document “critical\_summary.md”
  - Add me as collaborator “pacheco”
  - Set repository as Private

# Critical Reading Summaries

Starting next week **all readings** will require critical summaries

I will add a grade item to D2L for 1<sup>st</sup> half of summaries

- Nothing to hand in on D2L
- Append summaries to `critical_summary.md` in Github repo
- For full credit make sure to push summaries to Github regularly

**Short** paragraph that answers the following:

- What are the strengths?
- What are the weaknesses (what would you improve)?
- What are some points that would be helpful to discuss in class?

# Outline

- Monte Carlo Estimation
- Sequential Monte Carlo
- Markov Chain Monte Carlo



# Outline

- Monte Carlo Estimation
- Sequential Monte Carlo
- Markov Chain Monte Carlo

# Motivation for Monte Carlo Methods

- Now consider computing the expectation of a function  $f(z)$  over  $p(z)$  .
- Recall that this looks like  $E_{p(z)}[f] = \int_z f(z)p(z)dz$
- How can we approximate or estimate  $E[f]$ ?

## A bad plan...

Discretize the space where  $z$  lives into  $L$  blocks

Then compute  $E_{p(z)}[f] \cong \frac{1}{L} \sum_{l=1}^L p(z) f(z)$

**Scales poorly with dimension of  $Z$**

## A better plan...

Given independent samples  $z^{(l)}$  from  $p(z)$

Estimate  $E_{p(z)}[f] \cong \frac{1}{L} \sum_{l=1}^L f(z)$

# Motivation for Monte Carlo Methods

- Generally,  $Z$  lives in a very high dimensional space.
- Generally, regions of high  $p(z)$  is very little of that space.
- IE, the probability mass is very localized.
- Watching samples from  $p(z)$  should provide a good maximum (one of our inference problems)

# Motivation for Monte Carlo Methods

- Real problems are typically complex and high dimensional.
- Suppose that we *could* generate samples from a distribution that is proportional to one we are interested in.
- Typically we want posterior samples,

$$p(z | \mathcal{D}) = \frac{p(z)p(\mathcal{D} | z)}{p(\mathcal{D})} \propto \tilde{p}(z) \leftarrow \text{Unnormalized posterior}$$

↑  
Don't know marginal  
likelihood / normalizer

- Typically,  $\tilde{p}(z)$  is easier to evaluate (though not always)

# Inference (and related) Tasks

- Simulation:  $x \sim p(x) = \frac{1}{Z} f(x)$
- Compute expectations:  $\mathbb{E}[\phi(x)] = \int p(x) \phi(x) dx$
- Optimization:  $x^* = \arg \max_x f(x)$
- Compute normalizer / marginal likelihood:  $Z = \int f(x) dx$

# Inference (and related) Tasks

- **Simulation:**  $x \sim p(x) = \frac{1}{Z} f(x)$
- Compute expectations:  $\mathbb{E}[\phi(x)] = \int p(x) \phi(x) dx$
- Optimization:  $x^* = \arg \max_x f(x)$
- Compute normalizer / marginal likelihood:  $Z = \int f(x) dx$

# Sampling Continuous RVs

Recall that the CDF is the integral of the PDF and (left) tail probability,

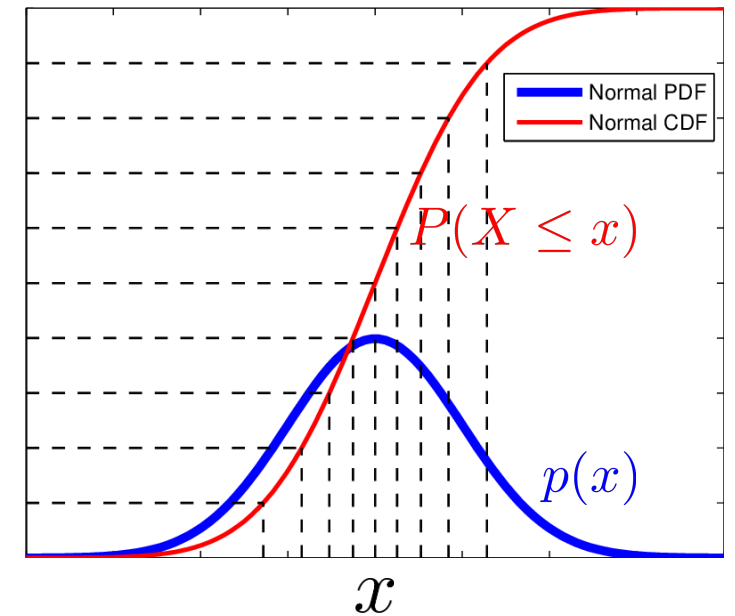
$$P(X \leq x) = \int_{-\infty}^x p(X = t) dt$$

**Observation 1** Equally spaced intervals of CDF correspond to regions of equal event probability

**Observation 2** The same events have unequal regions under PDF

**Question** Given samples  $\{x_i\}_{i=1}^N \sim p(x)$  what is the probability distribution of the CDF values,

$$\{P(X \leq x_i)\}_{i=1}^N \sim ???$$



# Sampling Continuous RVs

**Answer** The CDF of iid samples has a **standard uniform** distribution!

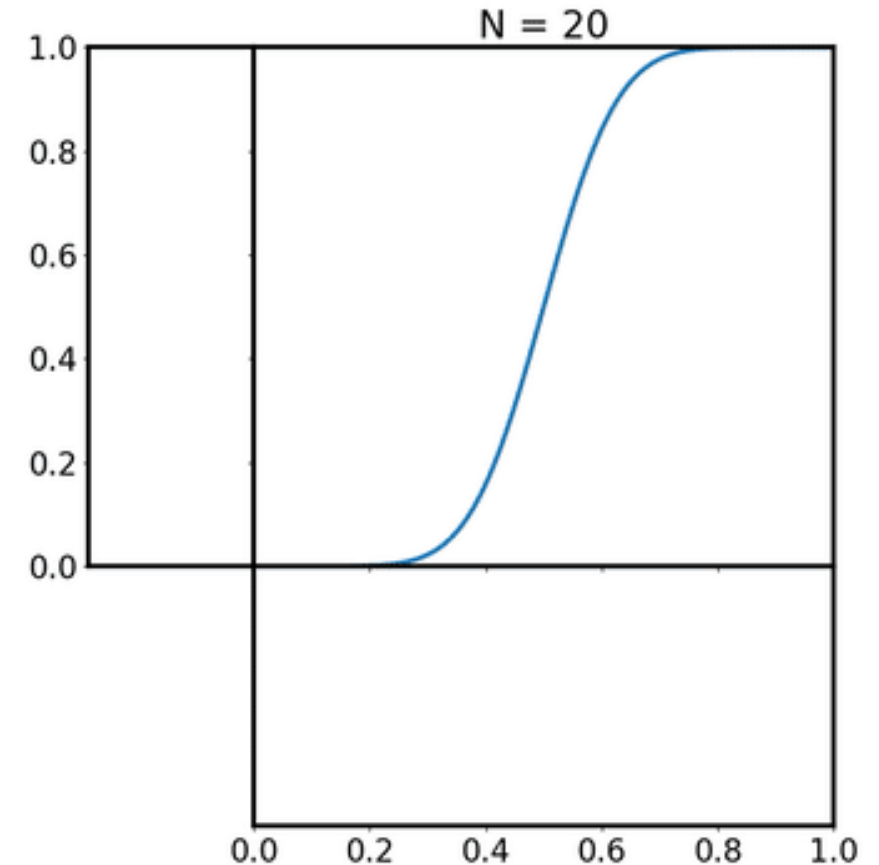
$$\{P(X \leq x_i)\}_{i=1}^N \sim \text{Uniform}(0, 1)$$

**Question** How can we use this fact to sample *any* RV?

**Answer** Apply this relationship in reverse:

1. Sample iid standard uniform RVs
2. Compute *inverse* CDF
3. Result are samples from the target

This property is called the **probability integral transform**





# Inverse Transform Sampling

- Input: Independent standard uniform variables  $U_1, U_2, U_3, \dots$
- We can use these to exactly sample from any continuous distribution using the cumulative distribution function:

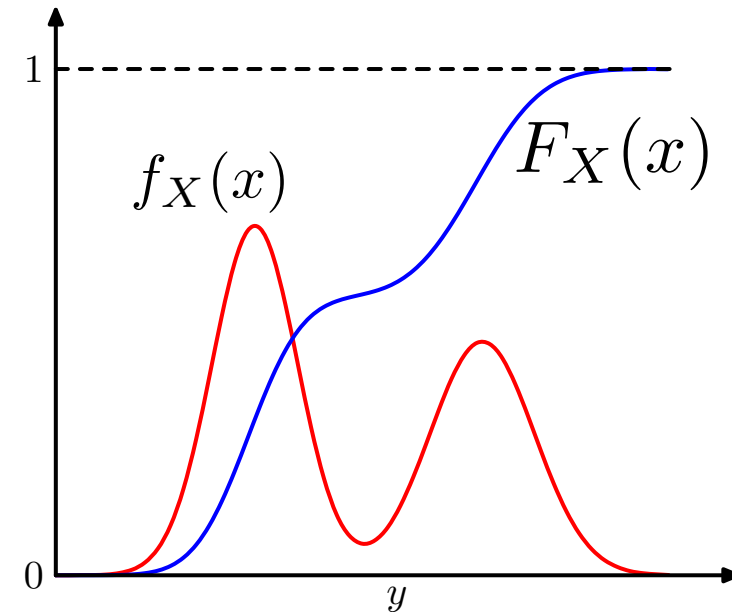
$$F_X(x) = P(X \leq x) = \int_{-\infty}^x f_X(z) dz$$

- Assuming continuous CDF is invertible:

$$h(u) = F_X^{-1}(u)$$

$$X_i = h(U_i)$$

Requires us to have access to inverse CDF



$$P(X_i \leq x) = P(h(U_i) \leq x) = P(U_i \leq F_X(x)) = F_X(x)$$

*This function transforms uniform variables to our target distribution!*

# Inverse Transform Sampling

- Very nice trick that applies to *all* continuous RVs (in theory)
- Yay, we know how to sample any RV right? Wrong...
- Don't always have the *inverse* CDF (or cannot calculate it)
- Doesn't extend easily to multivariate RVs (that's why I only showed 1-dimensional)

# Rejection Sampling

## Assume

- Access to easy-to-sample distribution  $q(z)$
- Constant  $k$  such that  $\tilde{p}(z) \leq k \cdot q(z)$

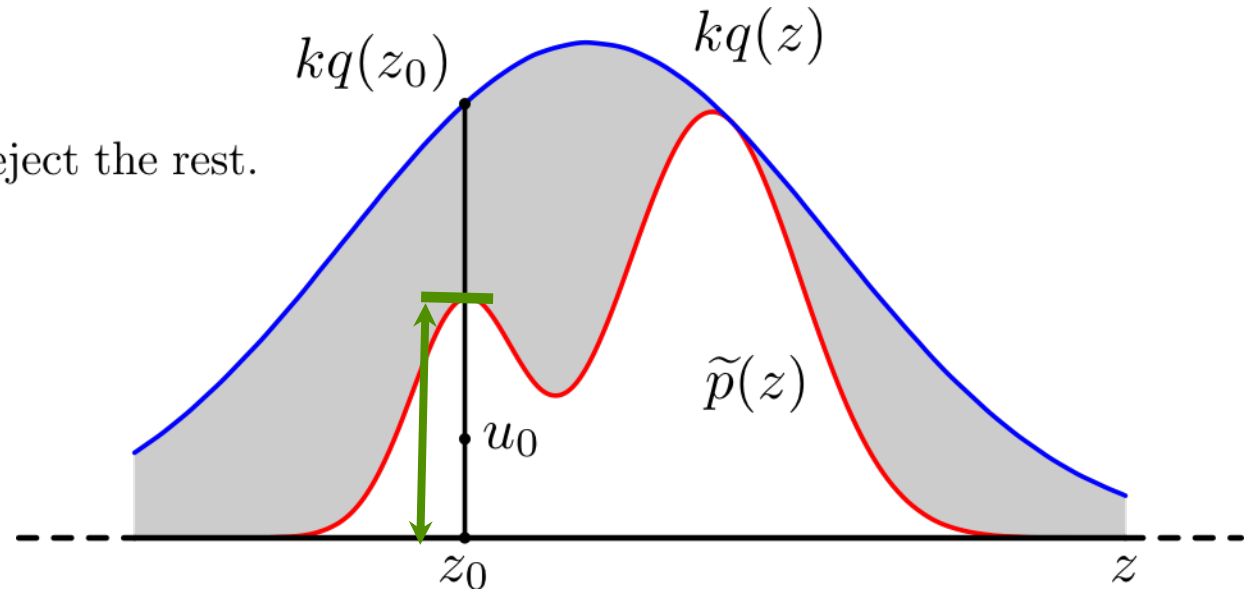
*Proposal Distribution*  
Where we can use one of methods on previous slides to sample efficiently

## Algorithm

1) Sample  $q(z)$

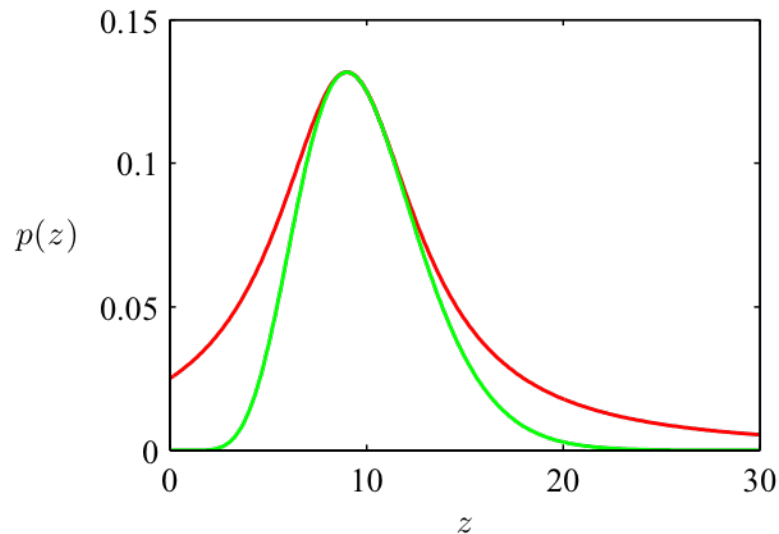
2) Keep samples in proportion to  $\frac{\tilde{p}(z)}{k \cdot q(z)}$  and reject the rest.

**Example** Uses Gaussian proposal  $q$  to draw samples from multimodal distribution  $p$



# Rejection Sampling

- Rejection sampling is hopeless in high dimensions, but is useful for sampling low dimensional “building block” functions.
- For example, the Box-Muller method for generating samples from a Gaussian uses rejection sampling.



A second example where a gamma distribution is approximated by a Cauchy proposal distribution.

# Inference (and related) Tasks

- Simulation:  $x \sim p(x) = \frac{1}{Z} f(x)$
- **Compute expectations:**  $\mathbb{E}[\phi(x)] = \int p(x) \phi(x) dx$
- Optimization:  $x^* = \arg \max_x f(x)$
- Compute normalizer / marginal likelihood:  $Z = \int f(x) dx$

# Monte Carlo Integration

*One reason to sample a distribution is to approximate expected values under that distribution...*

Expected value of function  $f(x)$  w.r.t. distribution  $p(x)$  given by,

$$\mathbb{E}_p[f(x)] = \int p(x)f(x) dx \equiv \mu$$

- Doesn't always have a closed-form for arbitrary functions
- Suppose we have iid samples:  $\{x_i\}_{i=1}^N \sim p(x)$
- *Monte Carlo* estimate of expected value,

$$\hat{\mu}_N = \frac{1}{N} \sum_{i=1}^N f(x_i) \approx \mathbb{E}_p[f(x)]$$

# Monte Carlo Integration

$$\hat{\mu}_N = \frac{1}{N} \sum_{i=1}^N f(x_i) \approx \mathbb{E}_p[f(x)]$$

- Expectation estimated from *empirical distribution* of  $N$  samples:

$$\hat{p}_N(x) = \frac{1}{N} \sum_{i=1}^N \delta_{x_i}(x) \quad \{x_i\}_{i=1}^N \sim p(x)$$

- The *Dirac delta* is *loosely* defined as a piecewise function:

$$\delta_{x_i}(x) = \begin{cases} +\infty & x = x_i \\ 0 & x \neq x_i \end{cases} \quad \text{Caveat This is technically incorrect. Dirac is only well-defined within integrals, } \int \delta_{\bar{x}}(x) f(x) dx = f(\bar{x}) \text{ but it gets the intuition across.}$$

- For any  $N$  this estimator, a random variable, is *unbiased*:

$$\mathbb{E}[\hat{\mu}_N] = \frac{1}{N} \sum_{i=1}^N f(x_i) = \mathbb{E}_p[f(x)]$$

# Monte Carlo Asymptotics

$$\hat{\mu}_N = \frac{1}{N} \sum_{i=1}^N f(x_i) \approx \mathbb{E}_p[f(x)]$$

- Estimator variance reduces at rate 1/N:

$$\text{Var}[\hat{\mu}_N] = \frac{1}{N} \text{Var}[f] = \frac{1}{N} \mathbf{E} [(f(x) - \mu)^2]$$

Independent of dimensionality  
of random variable X

- If the true variance is **finite** have *central limit theorem*:

$$\sqrt{N}(\hat{\mu}_N - \mu) \xrightarrow{N \rightarrow \infty} \mathcal{N}(0, \text{Var}[f])$$

- Even if true variance is **infinite** have *laws of large numbers*:

*Weak Law*

$$\lim_{N \rightarrow \infty} \Pr (|\hat{\mu}_N - \mu| < \epsilon) = 1, \quad \text{for any } \epsilon > 0$$

*Strong Law*

$$\Pr \left( \lim_{N \rightarrow \infty} \hat{\mu}_N = \mu \right) = 1$$



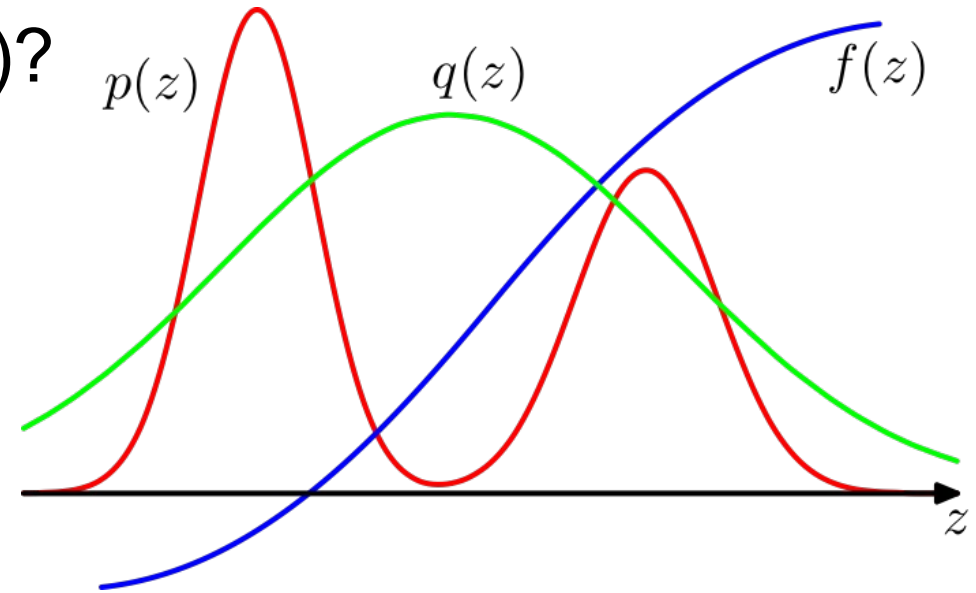
# Importance Sampling

Can we estimate  $\mathbb{E}_p[f]$  without sampling  $p(z)$ ?

$$\mathbb{E}_p[f] = \int f(z)p(z) dz$$

$q(z)$  is an easy-to-sample  
proposal distribution

$$= \int f(z) \frac{p(z)}{q(z)} q(z) dz$$



Monte Carlo estimate over samples  $\{z_i\}_{i=1}^N \sim q$  from proposal  $q(z)$ :

$$\mathbb{E}_p[f] \approx \frac{1}{N} \sum_{i=1}^N \frac{p(z_i)}{q(z_i)} f(z_i)$$

**Key: We can sample from an “easy” distribution  $q(z)$  instead!**

# Importance Sampling

IS weights are the ratio of target / proposal distributions:

$$\mathbb{E}_p[f] \approx \frac{1}{N} \sum_{i=1}^N w_i f(z_i) \quad \text{where} \quad w_i = \frac{p(z_i)}{q(z_i)}$$

But we often do not know the normalizer of the target distribution,

$$p(z) = \frac{1}{Z_p} \tilde{p}(z) \quad \text{where} \quad Z_p = \int \tilde{p}(z) dz$$

 **Can only evaluate unnormalized target**

Can we evaluate IS estimate in terms of unnormalized weights?

$$\tilde{w}_i = \frac{\tilde{p}(z_i)}{q(z_i)} \quad \text{Yes! Let's see how...}$$

# Importance Sampling (Normalized)

Recall, the importance sampling estimate is given by,

$$\mathbb{E}_p[f] \approx \frac{1}{N} \sum_{i=1}^N \frac{p(z_i)}{q(z_i)} f(z_i)$$

With normalized target and proposal distributions, respectively:

$$p(z) = \frac{1}{Z_p} \tilde{p}(z) \qquad q(z) = \frac{1}{Z_q} \tilde{q}(z)$$

Substitute and pull out ratio of normalizers,

$$\mathbb{E}_p[f] \approx \left( \frac{Z_q}{Z_p} \right) \frac{1}{N} \sum_{i=1}^N \frac{\tilde{p}(z_i)}{\tilde{q}(z_i)} f(z_i)$$

**Need to compute this...**



**Easy to compute**



# Importance Sampling (Normalized)

**Idea** Compute importance sampling estimate of target normalizer:

$$Z_p = \int \tilde{p}(z) dz = \int \frac{\tilde{p}(z)}{q(z)} q(z) dz \approx \frac{1}{N} \sum_{i=1}^N \frac{\tilde{p}(z_i)}{q(z_i)}$$

Typically we have normalized proposal  $q(z)$  so  $Z_q=1$  and,

$$\frac{Z_p}{Z_q} \approx \frac{1}{N} \sum_{i=1}^N \frac{\tilde{p}(z_i)}{q(z_i)} = \frac{1}{N} \sum_{i=1}^N \tilde{w}_i$$

Where  $\tilde{w}_i$  are our *unnormalized importance weights*,

$$\tilde{w}_i = \frac{\tilde{p}(z_i)}{q(z_i)} \quad \text{We can compute this!}$$

# Importance Sampling (normalized)

Given samples  $\{z_i\}_{i=1}^N \sim q$  we can write the IS estimate as,

$$\mathbb{E}_p[f] \approx \left( \frac{Z_q}{Z_p} \right) \frac{1}{N} \sum_{i=1}^N \tilde{w}_i f(z_i)$$

The ratio of normalizers is approximated by normalized weights,

$$\frac{Z_p}{Z_q} \approx \frac{1}{N} \sum_{i=1}^N \tilde{w}_i$$

Substituting the normalized weights yields,

$$\mathbb{E}_p[f] \approx \frac{\sum_{i=1}^N \tilde{w}_i f(z_i)}{\sum_{j=1}^N \tilde{w}_j} \quad \text{where} \quad \tilde{w}_j = \frac{\tilde{p}(z_j)}{\tilde{q}(z_j)}$$

# Importance Sampling On-A-Slide

1. Simulate from tractable distribution

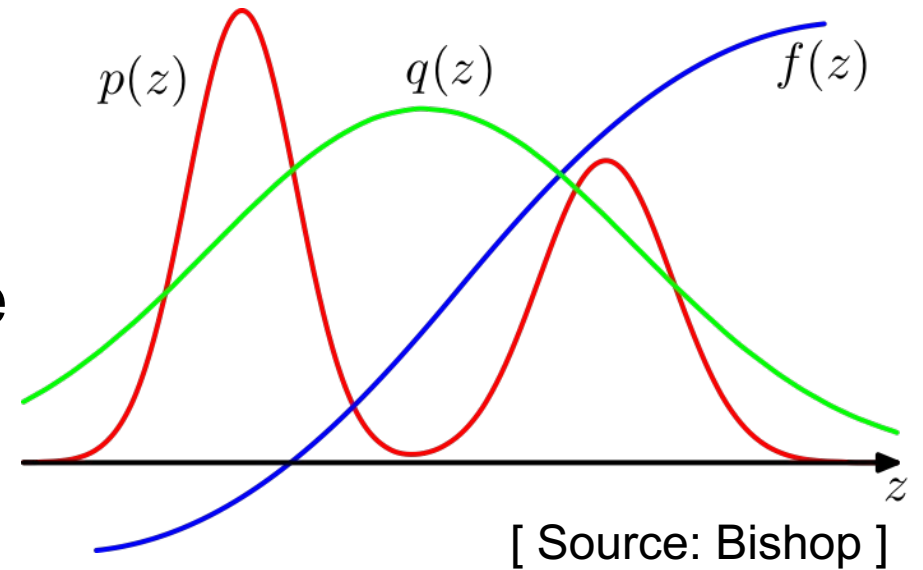
$$\{z_i\}_{i=1}^N \sim q(z)$$

2. Compute importance weights & normalize

$$\tilde{w}_i = \frac{\tilde{p}(z_i)}{q(z_i)} \quad w_i = \frac{\tilde{w}_i}{\sum_{i=1}^N \tilde{w}_i}$$

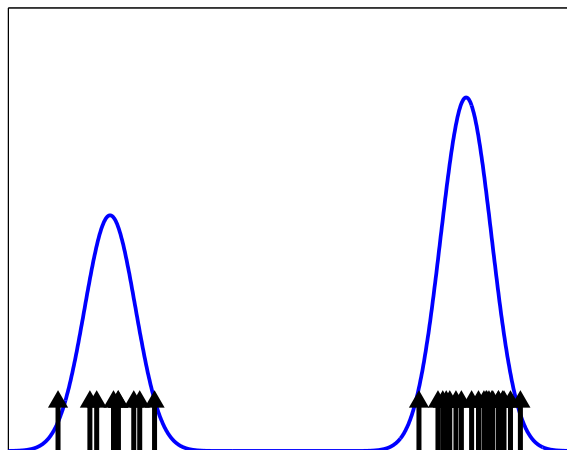
3. Compute importance-weighted expectation

$$\mathbf{E}_p[f(z)] \approx \sum_{i=1}^N w_i f(z_i) \equiv \hat{f}$$

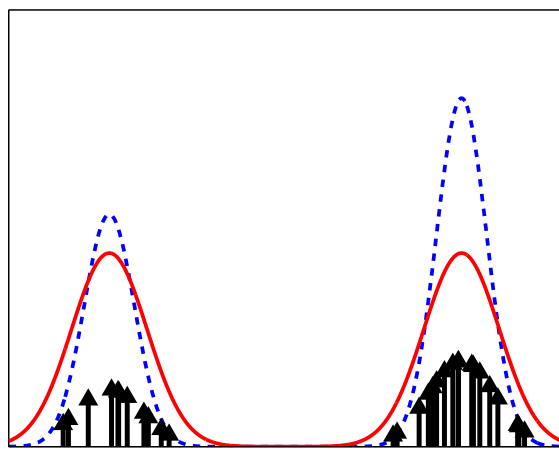


**Note** There is no  $1/N$  term since it is part of the normalized IS weights

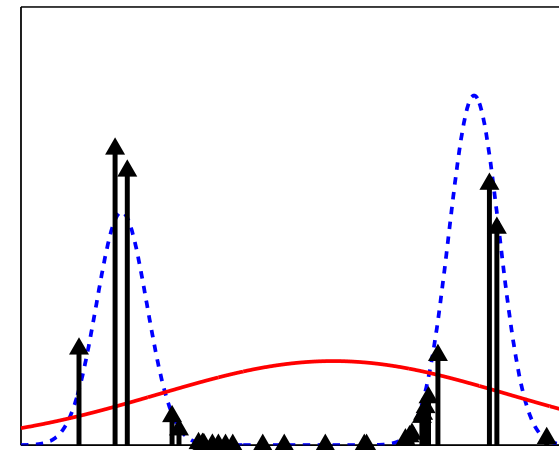
# Selecting Proposal Distributions



*Target Distribution*



*Good Proposal*

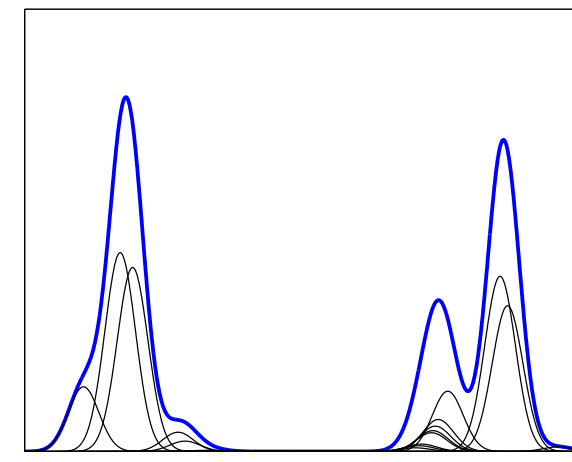
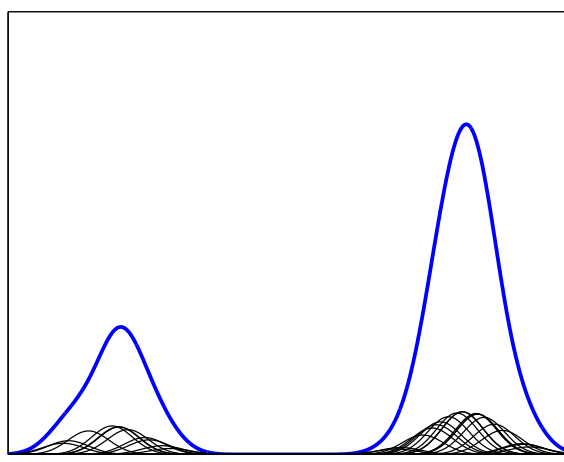
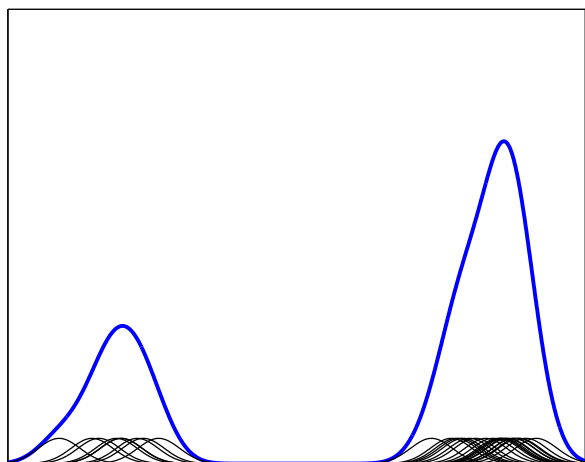


*Poor Proposal*

*Kernel or Parzen window estimators  
interpolate to predict density:*

$$\hat{p}(x) = \sum_{\ell=1}^L w^{(\ell)} \mathcal{N}(x; x^{(\ell)}, \Lambda)$$

$$w^{(\ell)} \propto \frac{p(x^{(\ell)})}{q(x^{(\ell)})}$$



# Importance Sampling

**Q:** What is a good proposal distribution?

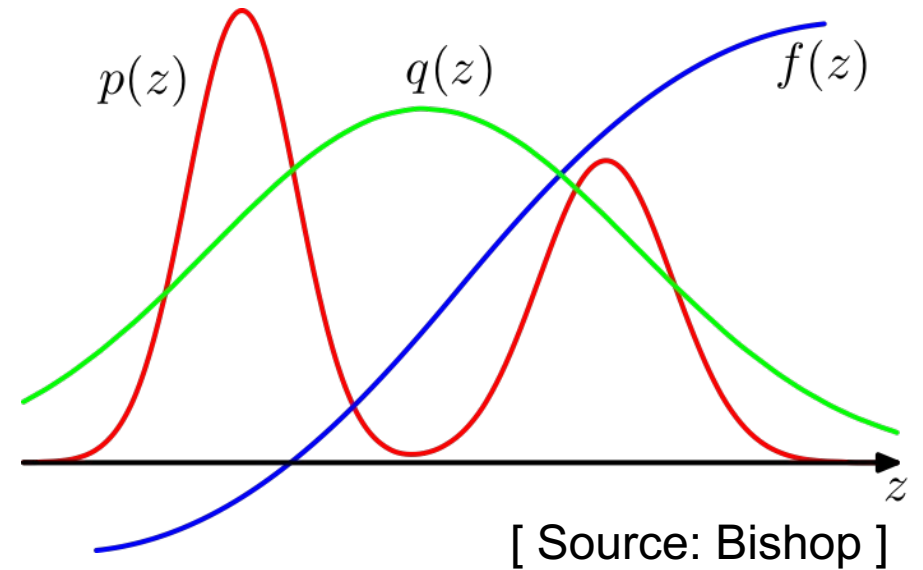
**A:** Minimize estimator variance

$$q^* = \arg \min_q \text{Var}_q(\hat{f})$$

Minimum variance obtained when,

$$q^* \propto |f(z)|p(z)$$

**E.g. can do better than  $q=p$**



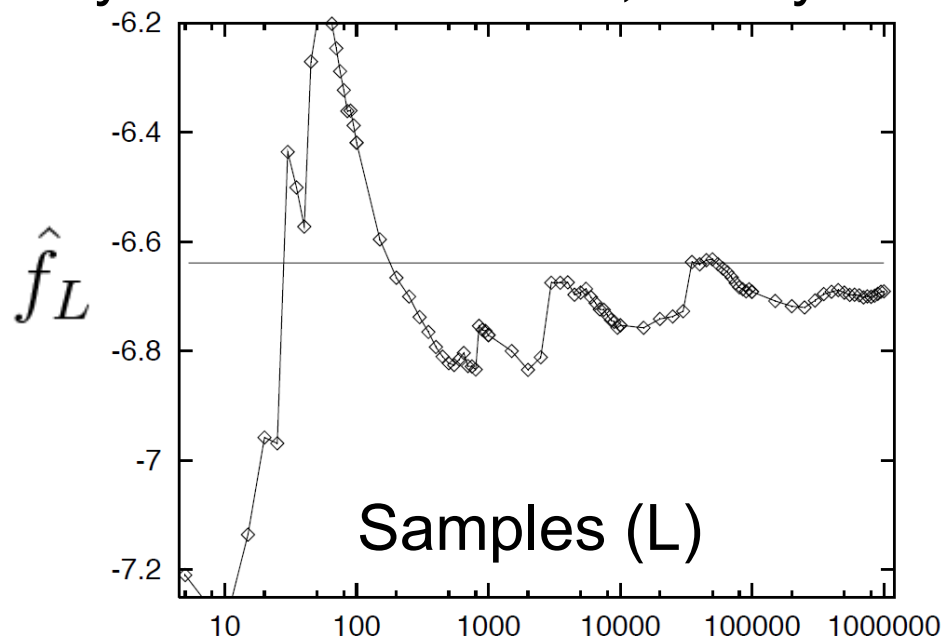
Estimator variance scales catastrophically with dimension:

e.g. for N-dim.  $X$  and Gaussian  $q(x)$ :  $\text{Var}_{q^*}(\hat{f}) = \exp(\sqrt{2N})$

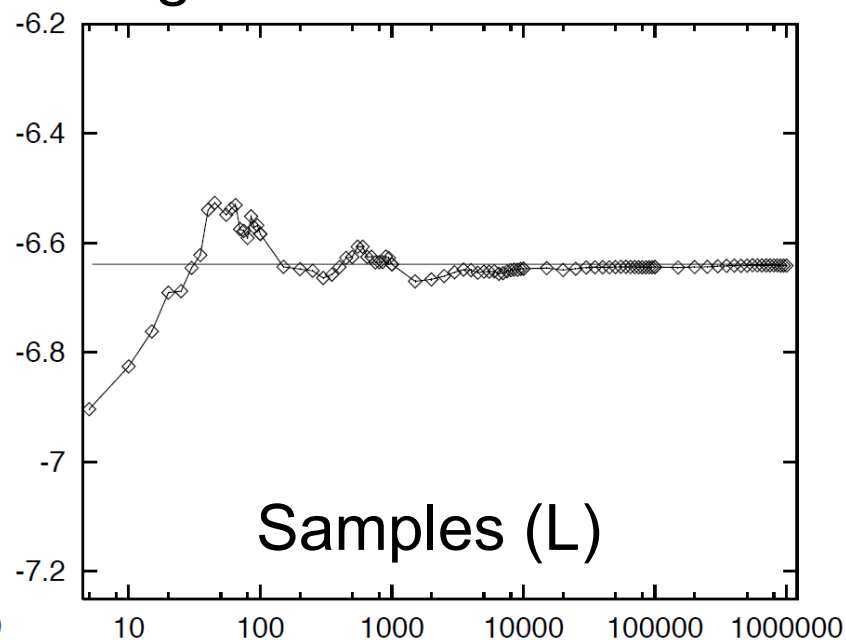


# Selecting Proposal Distributions

- For a toy one-dimensional, heavy-tailed target distribution:



*Gaussian Proposal*



*Cauchy (Student's-t) Proposal*

***Empirical variance of weights may not predict estimator variance!***

- Always (asymptotically) unbiased, but variance of estimator can be enormous unless weight function bounded above:

$$\mathbb{E}_q[\hat{f}_L] = \mathbb{E}_p[f] \quad \text{Var}_q[\hat{f}_L] = \frac{1}{L} \text{Var}_q[f(x)w(x)] \quad w(x) = \frac{p(x)}{q(x)}$$

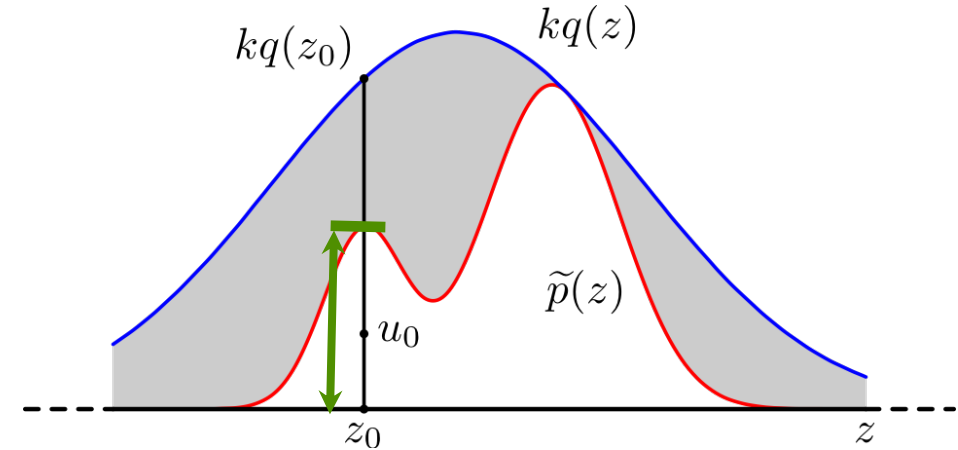
# Monte Carlo Methods Summary

## Rejection sampling

- Choose  $q$  such that:  $\tilde{p}(z) \leq k \cdot q(z)$
- Sample  $q(z)$  and keep with probability:  $\frac{\tilde{p}(z)}{k \cdot q(z)}$

Pro: Efficient, easy to implement

Con: Acceptance rate evaporates as dimension increases

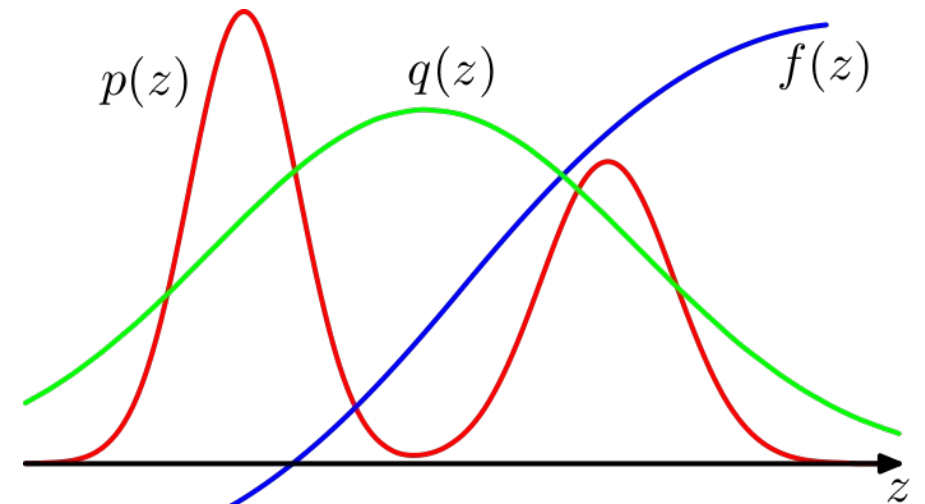


## Importance Sampling

$$\mathbf{E}_p[f(z)] \approx \sum_{l=1}^L \frac{\tilde{r}^{(l)}}{\sum_{i=1}^L \tilde{r}^{(i)}} f(z^{(l)}) \quad \tilde{r}^{(l)} = \frac{\tilde{p}(z^{(l)})}{q(z^{(l)})}$$

Pro: Efficient, easy to implement

Con: Variance grows exponentially in dimension



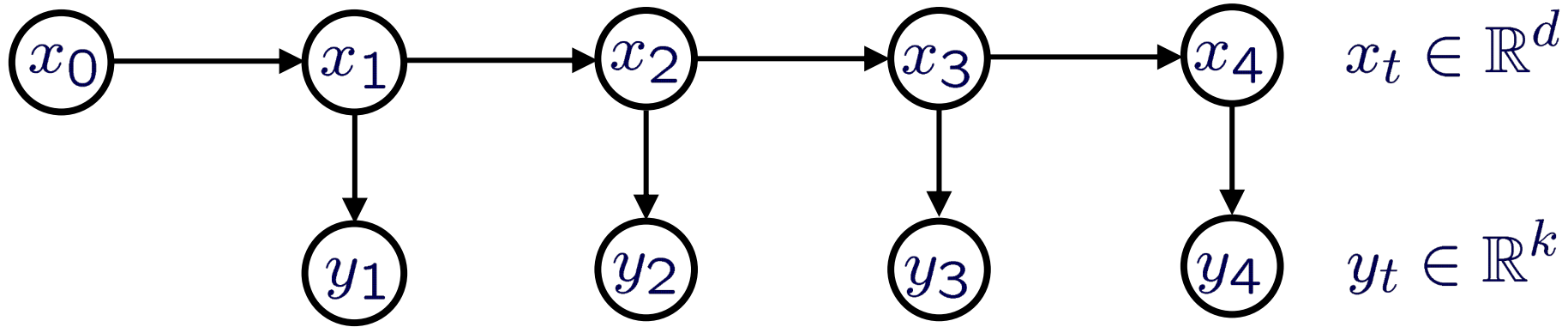
# Outline

- Monte Carlo Estimation
- Sequential Monte Carlo
- Markov Chain Monte Carlo

# Outline

- Monte Carlo Estimation
- **Sequential Monte Carlo**
- Markov Chain Monte Carlo

# Non-linear State Space Models



$$x_{t+1} = f(x_t, w_t)$$

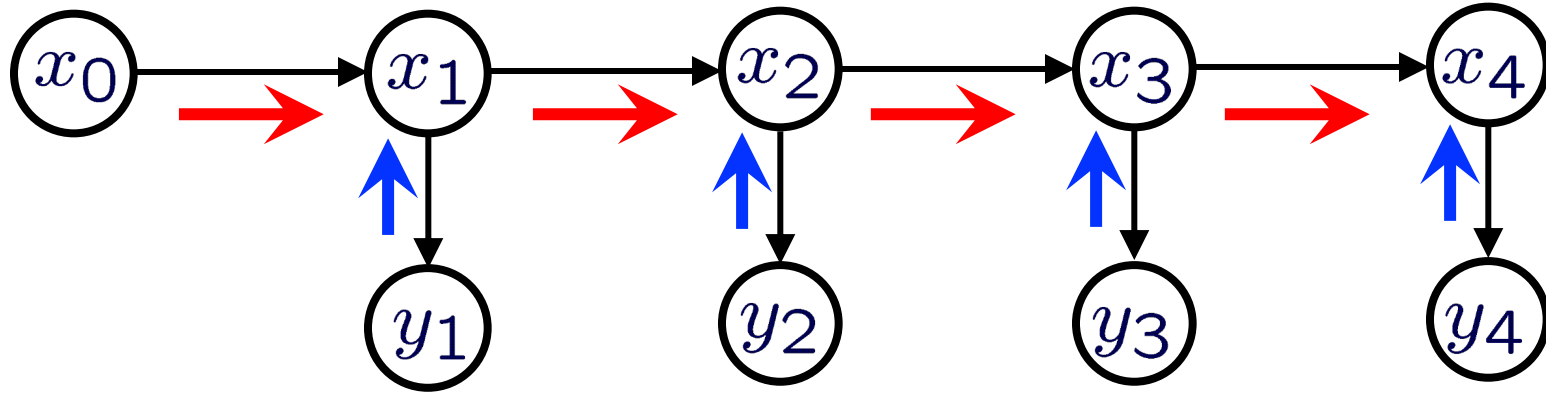
$$w_t \sim \mathcal{F}$$

$$y_t = g(x_t, v_t)$$

$$v_t \sim \mathcal{G}$$

- State dynamics and measurements given by potentially complex *nonlinear functions*
- Noise sampled from *non-Gaussian* distributions
- Usually no closed form for messages or marginals

# Sequential Importance Sampling (SIS)



- Suppose interested in some complex, global function of state:

$$\mathbb{E}[f] = \int f(x)p(x | y) dx \approx \sum_{\ell=1}^L w_{\ell} f(x^{(\ell)}) \quad w_{\ell} \propto \frac{p(x^{(\ell)} | y)}{q(x^{(\ell)} | y)} \quad x^{(\ell)} \sim q(x | y)$$

- Construct efficient proposal using Markov structure

$$q(x | y) = q(x_0) \prod_{t=1}^T q(x_t | x_{t-1}, y_t) \quad q(x_t | x_{t-1}, y_t) \approx p(x_t | x_{t-1}, y)$$

*Computing the weights is easy with this type of proposal!*

# Recursive Weight Updating

Recall the importance weights are given by,

$$w^{(\ell)} \propto \frac{p(x^{(\ell)} | y)}{q(x^{(\ell)} | y)} \propto \frac{p(x^{(\ell)}, y)}{q(x^{(\ell)} | y)}$$

Plugging in the factorization of  $p$  and  $q$  weights at time  $t$  are:

$$w_t^{(\ell)} \propto \frac{p(x_0^{(\ell)})}{q(x_0^{(\ell)})} \frac{p(x_1^{(\ell)} | x_0^{(\ell)})p(y_1 | x_1^{(\ell)})}{q(x_1^{(\ell)} | x_0^{(\ell)}, y_1)} \cdots \frac{p(x_t^{(\ell)} | x_{t-1}^{(\ell)})p(y_t | x_t^{(\ell)})}{q(x_t^{(\ell)} | x_{t-1}^{(\ell)}, y_t)}$$

Therefore, by recursion we have that weights at time  $t+1$  are:

$$w_{t+1}^{(\ell)} \propto w_t^{(\ell)} \frac{p(x_{t+1}^{(\ell)} | x_t^{(\ell)})p(y_{t+1} | x_{t+1}^{(\ell)})}{q(x_{t+1}^{(\ell)} | x_t^{(\ell)}, y_t)}$$

# Sequential Importance Sampling (SIS)

**For  $\ell = 1, \dots, N$**

Sample initial  $N$  particles from proposal prior:  $x_0^{(\ell)} \sim q_0$

Compute initial importance weights:  $w_0^{(\ell)} \propto p(x_0^{(\ell)}) \div q(x_0^{(\ell)})$

**For  $t=1, \dots, T$**

**For  $\ell = 1, \dots, N$**

Propagate particles:  $x_t^{(\ell)} \sim q(x_t | x_{t-1}^{(\ell)}, y_t)$

Compute *unnormalized* weights,

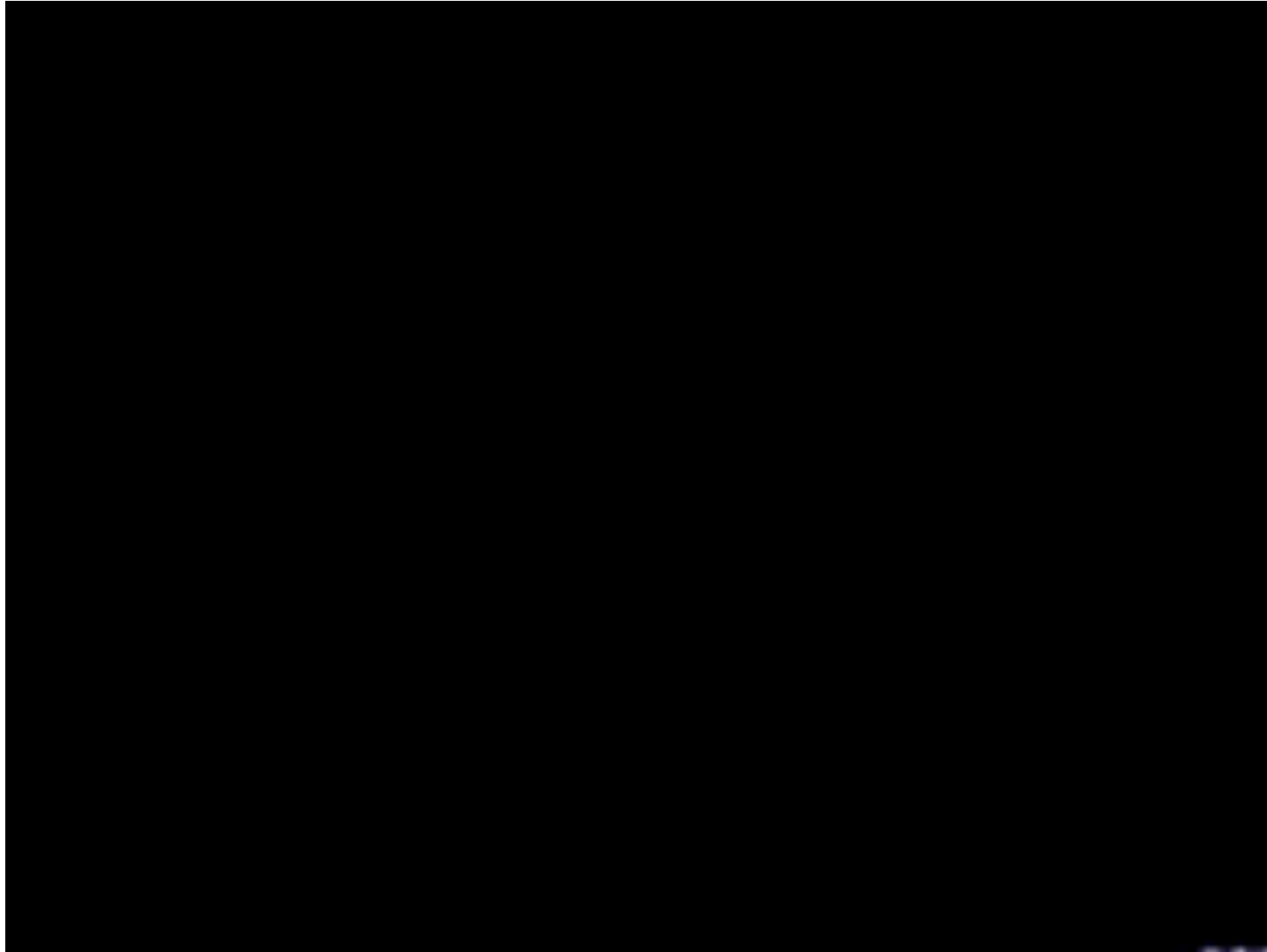
$$\tilde{w}_t^{(\ell)} = w_{t-1}^{(\ell)} \frac{p(x_t^{(\ell)} | x_{t-1}^{(\ell)}) p(y_t | x_t^{(\ell)})}{q(x_t^{(\ell)} | x_{t-1}^{(\ell)}, y_t)}$$

Normalize weights:  $w_t^{(\ell)} = \tilde{w}_t^{(\ell)} \div \sum_i \tilde{w}_t^{(i)}$

Filter mean estimate:  $\hat{x}_t = \sum_{\ell} w_t^{(\ell)} x_t^{(\ell)}$



# Particle Filters: The Movie



(M. Isard, 1996)

# Weight Degeneration

*Sequential importance sampling does not work!*

- Sample trajectories  $x^{(\ell)}$  are high-dimensional and become unlikely
- In time, unnormalized weights approach zero with high probability,

$$\lim_{t \rightarrow \infty} \tilde{w}_t^{(\ell)} = 0$$

- Normalized weights approach *one-hot vector*,

$$w_t^{(\ell)} = \tilde{w}_t^{(\ell)} \div \sum_i \tilde{w}_t^{(i)} = \begin{cases} 1 & \text{if } \tilde{w}_t^{(\ell)} = \max \tilde{w}_t \\ 0 & \text{otherwise} \end{cases}$$

# Particle Resampling

$$p(x_t | y_{\bar{t}}) \approx \sum_{\ell=1}^L \omega_t^{(\ell)} \delta_{x_t^{(\ell)}}(x_t)$$

where  $y_{\bar{t}} = \{y_1, \dots, y_t\}$

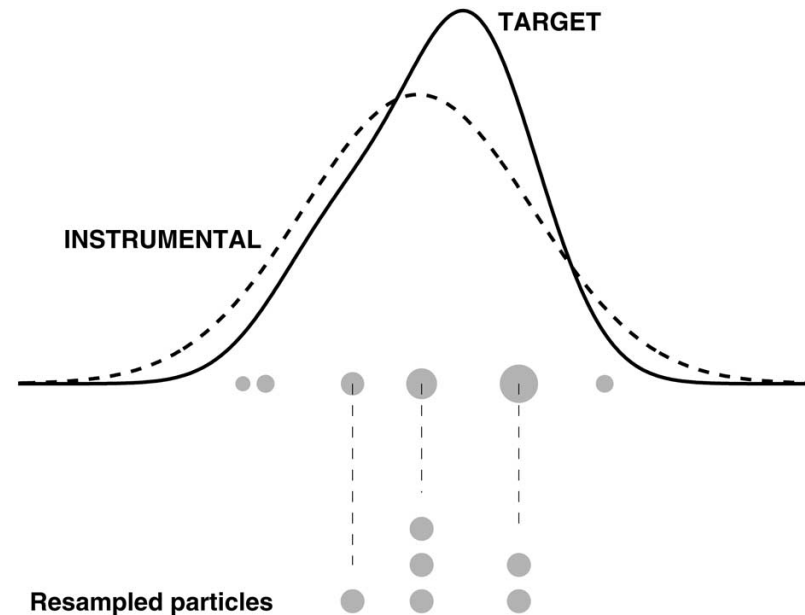
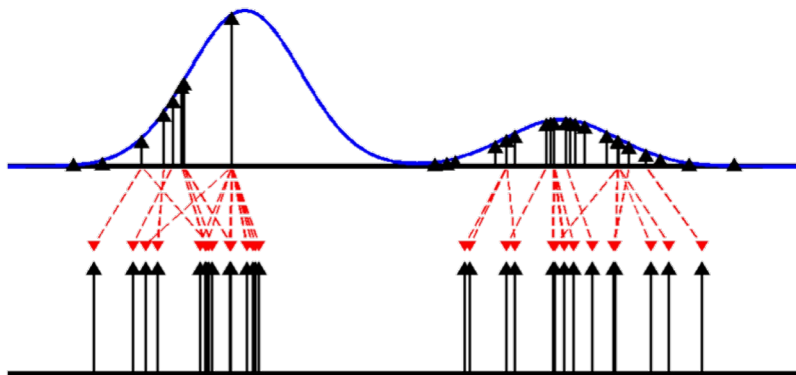
$$p(x_t | y_{\bar{t}}) \approx \sum_{\ell=1}^L \frac{1}{L} \delta_{\bar{x}_t^{(\ell)}}(x_t)$$

$$\bar{x}_t^{(\ell)} = x_t^{(j_\ell)}$$

$$j_\ell \sim \text{Cat}(\omega_t)$$

Resample with replacement produces *random discrete distribution* with same mean as original distribution

While remaining unbiased, resampling avoids degeneracies in which most weights go to zero



# Sequential IS with Resampling : Particle Filter

**Initialize:** N samples  $\tilde{x}_0^{(\ell)} \sim q_0$  and weights  $w_0^{(\ell)} \propto p(x_0^{(\ell)}) \div q(x_0^{(\ell)})$

**For t=1,...T**

**If Resampling:**

Resample  $x_{t-1}^{(\ell)}$  from  $\tilde{x}_{t-1}$  according to normalized weights  $w_{t-1}$  (with replacement)

Set uniform weights  $w_{t-1} = 1/N$

**Else:** Set  $x_{t-1} = \tilde{x}_{t-1}$

**For  $\ell=1,\dots,N$**

Propagate particles:  $x_t^{(\ell)} \sim q(x_t | x_{t-1}^{(\ell)}, y_t)$

Compute *unnormalized* weights,  $\tilde{w}_t^{(\ell)} = w_{t-1}^{(\ell)} \frac{p(x_t^{(\ell)} | x_{t-1}^{(\ell)}) p(y_t | x_t^{(\ell)})}{q(x_t^{(\ell)} | x_{t-1}^{(\ell)}, y_t)}$

Normalize weights:  $w_t^{(\ell)} = \tilde{w}_t^{(\ell)} \div \sum_i \tilde{w}_t^{(i)}$

Filter mean estimate:  $\hat{x}_t = \sum_{\ell} w_t^{(\ell)} x_t^{(\ell)}$

# “Bootstrap” Proposal

Recall that the full proposal distribution factorizes as,

$$q(x | y) = q(x_0) \prod_{t=1}^T q(x_t | x_{t-1}, y_t)$$

A convenient choice is to sample from the prior distribution,

$$q(x) = p(x_0) \prod_{t=1}^T p(x_t | x_{t-1})$$

This is easy to sample, and weight updates simplify,

$$w_{t+1}^{(\ell)} \propto w_t^{(\ell)} \frac{\cancel{p(x_{t+1}^{(\ell)} | x_t^{(\ell)})} p(y_{t+1} | x_{t+1}^{(\ell)})}{\cancel{p(x_{t+1}^{(\ell)} | x_t^{(\ell)})}} = w_t^{(\ell)} p(y_{t+1} | x_{t+1}^{(\ell)})$$

**“Correct” weights with data likelihood**

# Bootstrap Particle Filter

**Initialize:** N samples  $\tilde{x}_0^{(\ell)} \sim q_0$  and weights  $w_0^{(\ell)} \propto p(x_0^{(\ell)}) \div q(x_0^{(\ell)})$

**For t=1,...T**

**If Resampling:**

Resample  $x_{t-1}^{(\ell)}$  from  $\tilde{x}_{t-1}$  according to normalized weights  $w_{t-1}$  (with replacement)

Set uniform weights  $w_{t-1} = 1/N$

**Else:** Set  $x_{t-1} = \tilde{x}_{t-1}$

**For  $\ell=1,\dots,N$**

Propagate particles:  $\tilde{x}_t^{(\ell)} \sim p(x_t | x_{t-1}^{(\ell)})$

Compute *unnormalized* weights,  $\tilde{w}_t^{(\ell)} = w_{t-1}^{(\ell)} p(y_t | \tilde{x}_t^{(\ell)})$

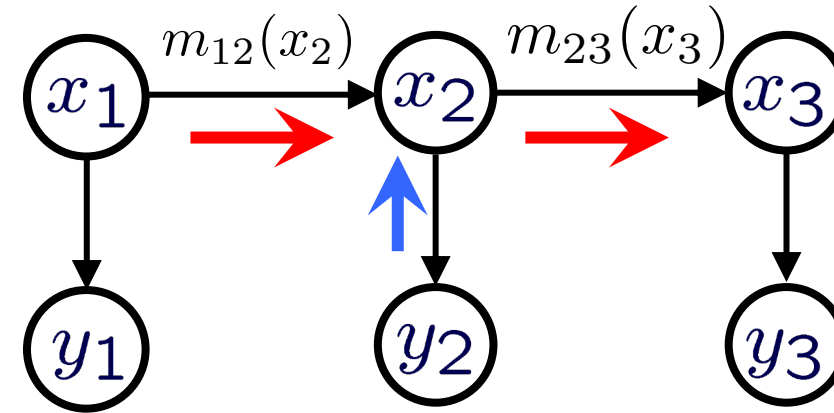
**Changes for  
Bootstrap**

Normalize weights:  $w_t^{(\ell)} = \tilde{w}_t^{(\ell)} \div \sum_i \tilde{w}_t^{(i)}$

Filter mean estimate:  $\hat{x}_t = \sum_{\ell} w_t^{(\ell)} x_t^{(\ell)}$

# Particle Filtering Algorithms

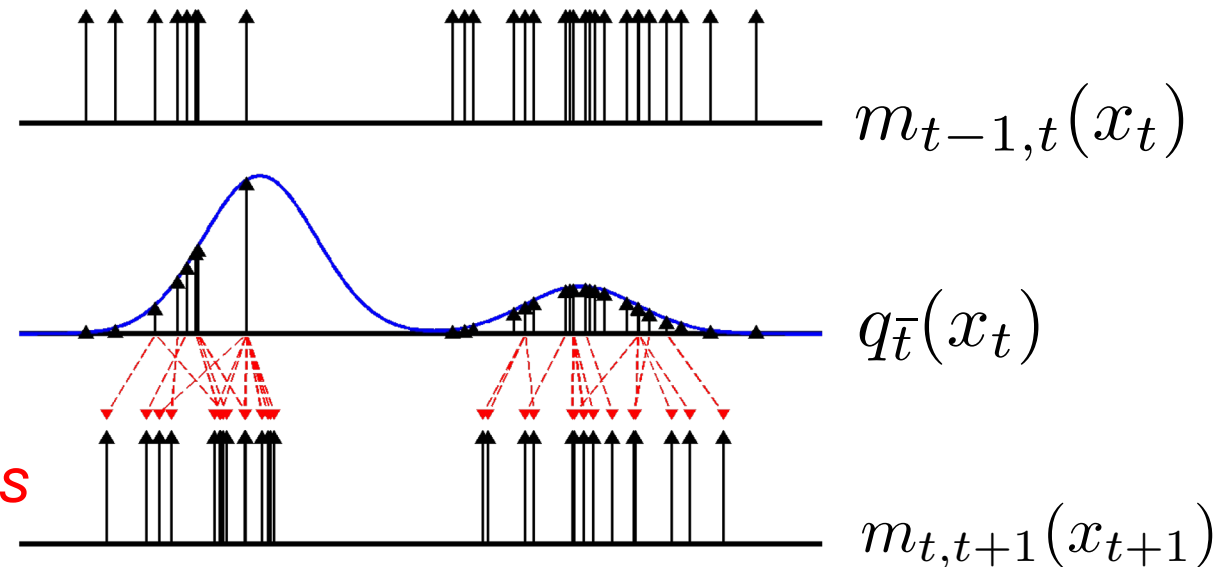
- Represent state estimates using a set of samples
- Propagate over time using *sequential importance sampling* with *resampling*



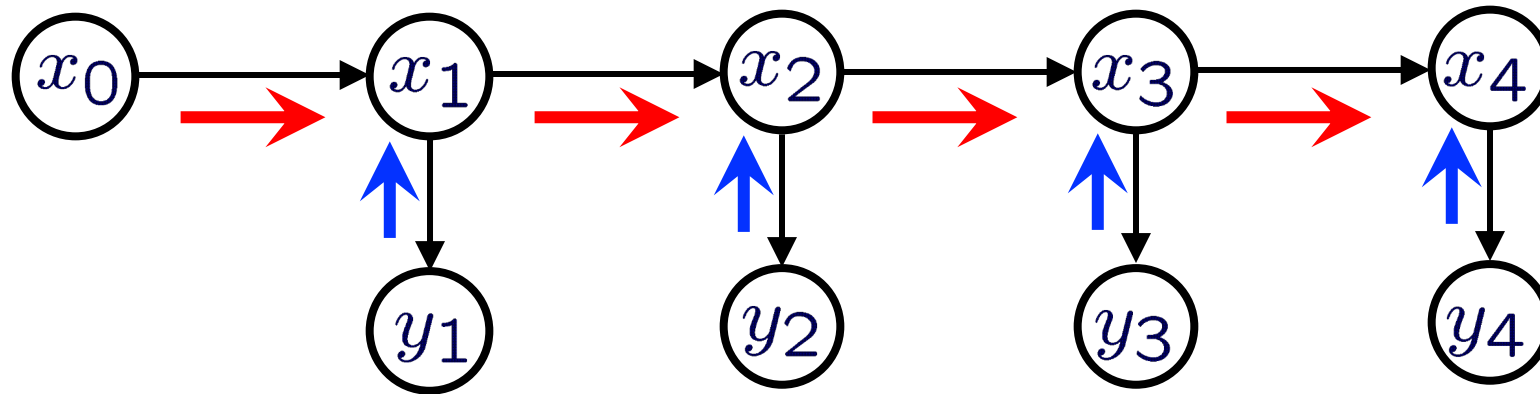
*Sample-based density estimate*

*Weight by observation likelihood*

*Resample & propagate by dynamics*



# BP for State-Space Models



$$m_{t-1,t}(x_t) \propto p(x_t | y_{\bar{t}-1}) \quad \text{where} \quad y_{\bar{t}} = \{y_1, \dots, y_t\}$$

$$m_{t-1,t}(x_t) p(y_t | x_t) \propto p(x_t | y_{\bar{t}}) = q_{\bar{t}}(x_t)$$

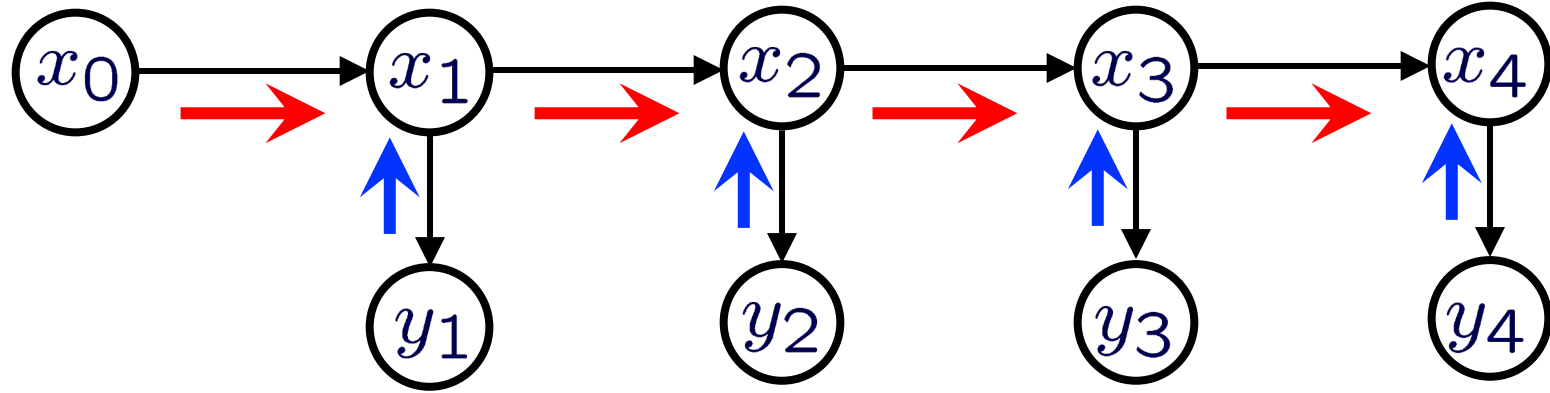
**Prediction (Integral/Sum step of BP):**

$$m_{t-1,t}(x_t) \propto \int p(x_t | x_{t-1}) q_{\bar{t}-1}(x_{t-1}) dx_{t-1}$$

**Inference (Product step of BP):**  $q_{\bar{t}}(x_t) = \frac{1}{Z_t} m_{t-1,t}(x_t) p(y_t | x_t)$



# Particle Filter: Measurement Update



**Incoming message:** A set of  $L$  weighted particles

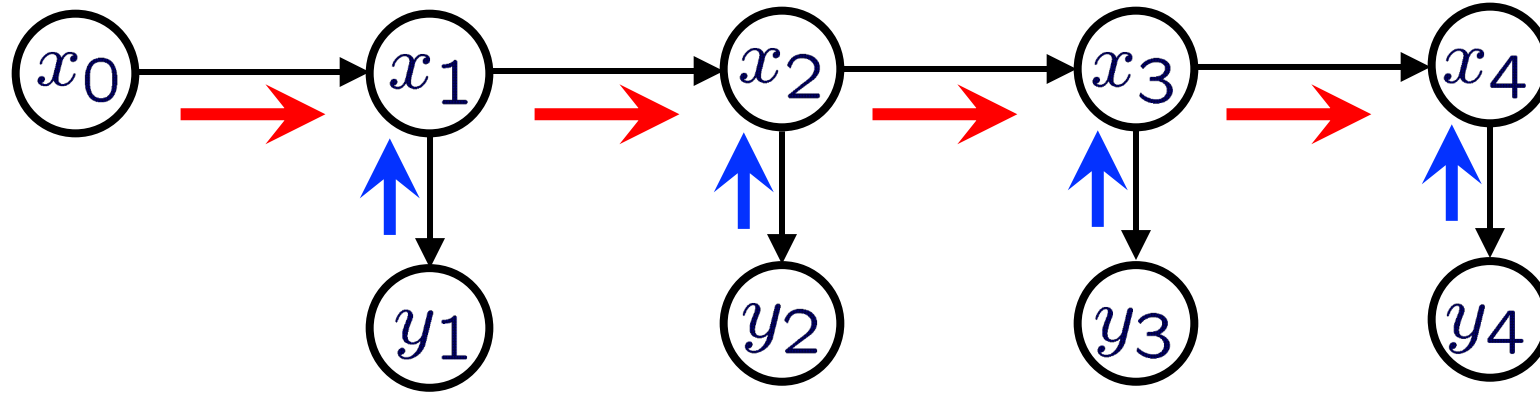
$$m_{t-1,t}(x_t) \approx \sum_{\ell=1}^L w_{t-1,t}^{(\ell)} \delta(x_t, x_t^{(\ell)}) \quad \sum_{\ell=1}^L w_{t-1,t}^{(\ell)} = 1$$

**Bayes' Rule:** Posterior at particles proportional to prior times likelihood

$$q_{\bar{t}}(x_t) \propto m_{t-1,t}(x_t) p(y_t | x_t) \propto \sum_{\ell=1}^L w_{t-1,t}^{(\ell)} p(y_t | x_t^{(\ell)}) \delta(x_t, x_t^{(\ell)})$$
$$q_{\bar{t}}(x_t) = \sum_{\ell=1}^L w_t^{(\ell)} \delta(x_t, x_t^{(\ell)}) \quad w_t^{(\ell)} \triangleq \frac{w_{t-1,t}^{(\ell)} p(y_t | x_t^{(\ell)})}{\sum_{m=1}^L w_{t-1,t}^{(m)} p(y_t | x_t^{(m)})}$$

*Variance of importance weights increases with each update*

# Particle Filter: Sample Propagation



**State Posterior Estimate:** A set of  $L$  weighted particles

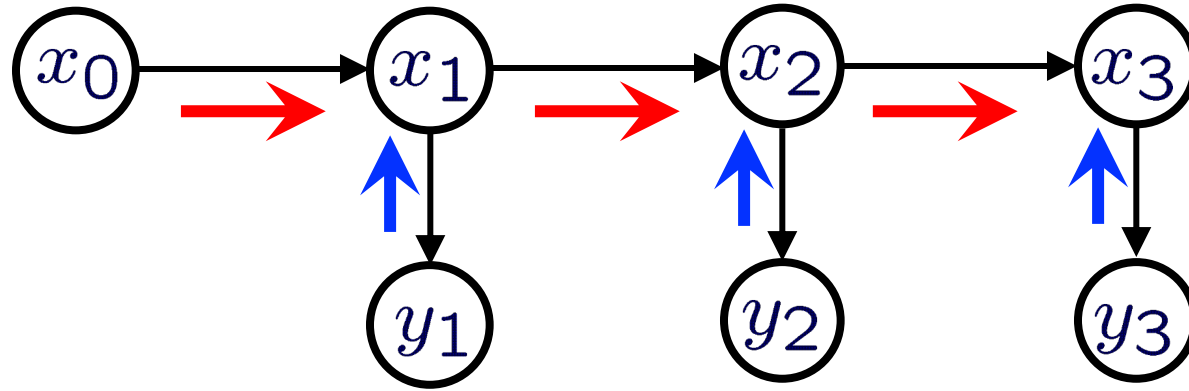
$$q_{\bar{t}}(x_t) = \sum_{\ell=1}^L w_t^{(\ell)} \delta(x_t, x_t^{(\ell)}) \quad \sum_{\ell=1}^L w_t^{(\ell)} = 1$$

**Prediction:** Sample next state conditioned on current particles

$$m_{t,t+1}(x_{t+1}) = \sum_{\ell=1}^L w_{t,t+1}^{(\ell)} \delta(x_{t+1}, x_{t+1}^{(\ell)}) \quad \begin{aligned} x_{t+1}^{(\ell)} &\sim p(x_{t+1} | x_t^{(\ell)}) \\ w_{t,t+1}^{(\ell)} &= w_t^{(\ell)} \end{aligned}$$

*Assumption for now: Can exactly simulate temporal dynamics*

# Particle Filter: Resampling



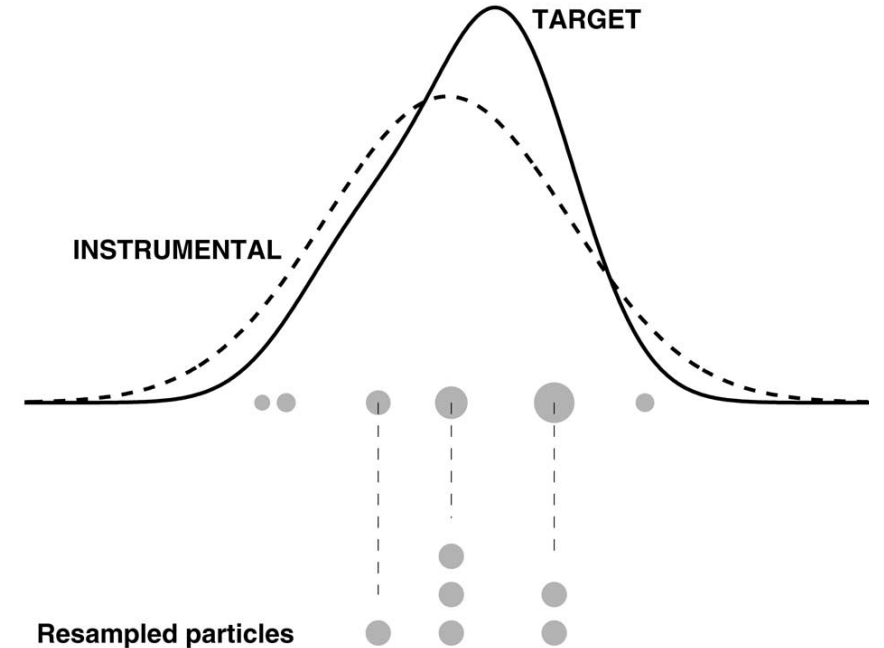
## State Posterior Estimate:

$$q_{\bar{t}}(x_t) = \sum_{\ell=1}^L w_t^{(\ell)} \delta(x_t, x_t^{(\ell)})$$

**Prediction:** Sample next state conditioned on randomly chosen particles

$$m_{t,t+1}(x_{t+1}) = \sum_{\ell=1}^L w_{t,t+1}^{(\ell)} \delta(x_{t+1}, x_{t+1}^{(\ell)})$$

*Resampling with replacement preserves expectations, but increases the variance of subsequent estimators*



$$\tilde{x}_t^{(\ell)} \sim q_{\bar{t}}(x_t)$$

$$x_{t+1}^{(\ell)} \sim p(x_{t+1} | \tilde{x}_t^{(\ell)})$$

$$w_{t,t+1}^{(\ell)} = 1/L$$

# Particle Filter: Resampling

## Effective Sample Size:

$$L_{\text{eff}} = \left( \sum_{\ell=1}^L \left( w^{(\ell)} \right)^2 \right)^{-1}$$

$$1 \leq L_{\text{eff}} \leq L$$

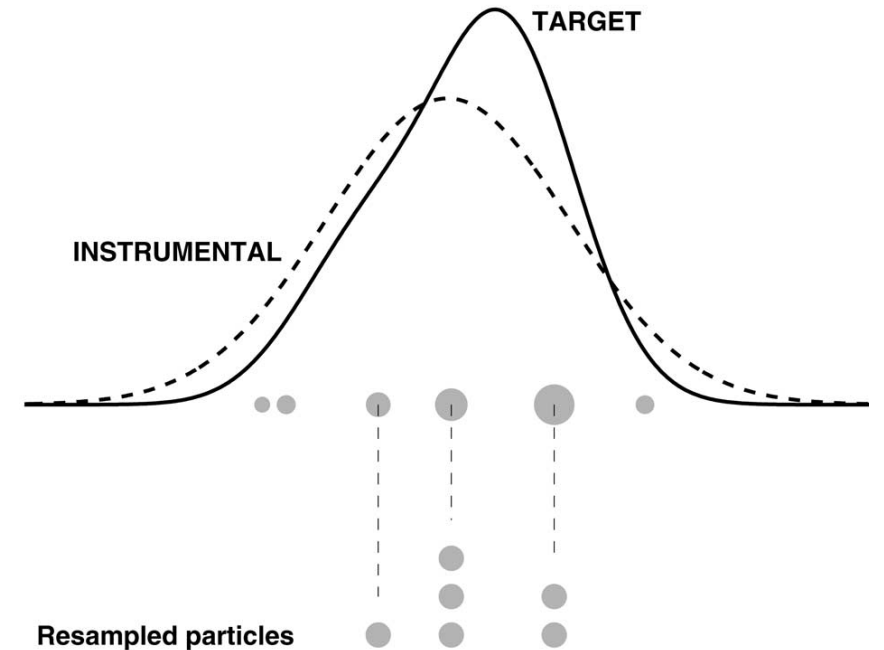
## State Posterior Estimate:

$$q_{\bar{t}}(x_t) = \sum_{\ell=1}^L w_t^{(\ell)} \delta(x_t, x_t^{(\ell)})$$

**Prediction:** Sample next state conditioned on randomly chosen particles

$$m_{t,t+1}(x_{t+1}) = \sum_{\ell=1}^L w_{t,t+1}^{(\ell)} \delta(x_{t+1}, x_{t+1}^{(\ell)})$$

*Resampling with replacement preserves expectations, but increases the variance of subsequent estimators*



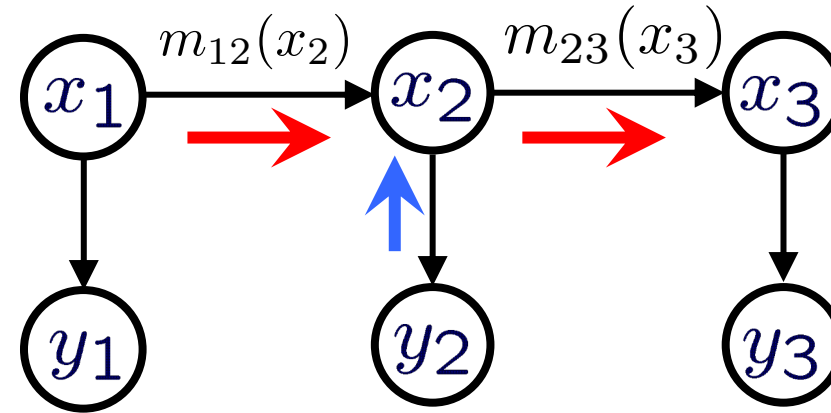
$$\tilde{x}_t^{(\ell)} \sim q_{\bar{t}}(x_t)$$

$$x_{t+1}^{(\ell)} \sim p(x_{t+1} | \tilde{x}_t^{(\ell)})$$

$$w_{t,t+1}^{(\ell)} = 1/L$$

# Particle Filtering Algorithms

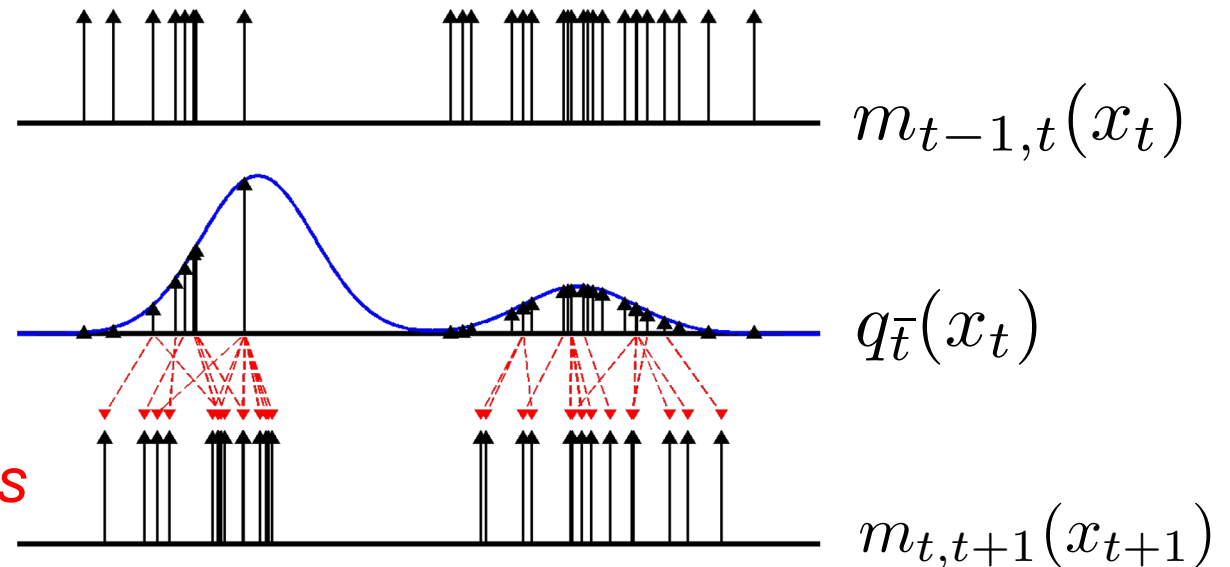
- Represent state estimates using a set of samples
- Propagate over time using *sequential importance sampling* with *resampling*



*Sample-based density estimate*

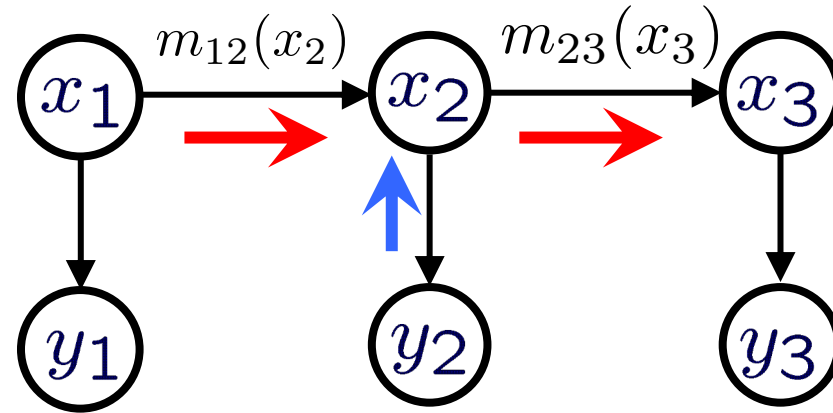
*Weight by observation likelihood*

*Resample & propagate by dynamics*



# Bootstrap Particle Filter Summary

- Represent state estimates using a set of samples
- Propagate over time using *sequential importance sampling with resampling*



Assume sample-based approximation of incoming message:

$$m_{t-1,t}(x_t) = p(x_t | y_{t-1}, \dots, y_1) \approx \sum_{\ell=1}^L \frac{1}{L} \delta_{x_t^{(\ell)}}(x_t)$$

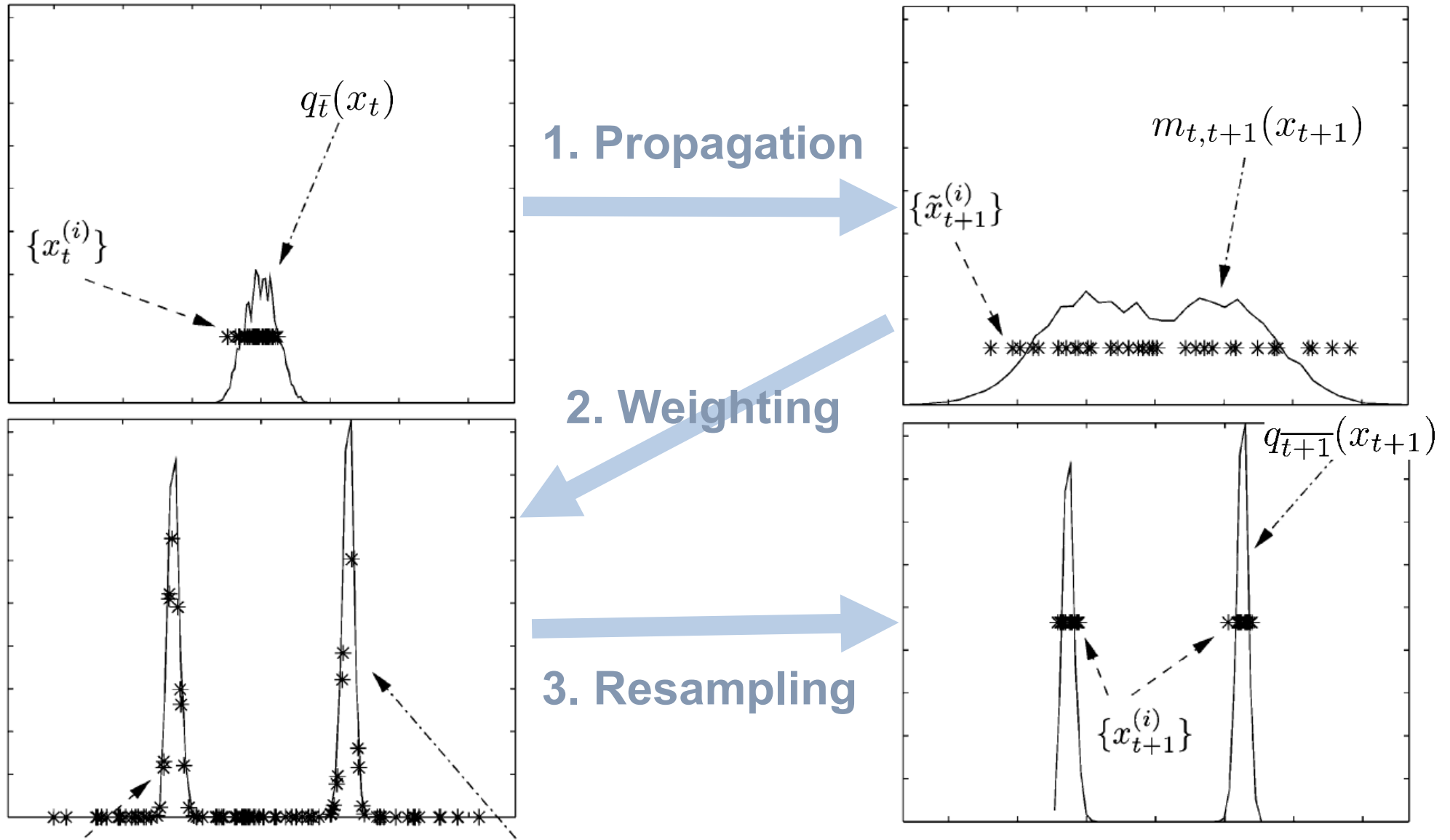
Account for observation via importance weights:

$$p(x_t | y_t, y_{t-1}, \dots, y_1) \approx \sum_{\ell=1}^L w_t^{(\ell)} \delta_{x_t^{(\ell)}}(x_t) \quad w_t^{(\ell)} \propto p(y_t | x_t^{(\ell)})$$

Sample from forward dynamics distribution of next state:

$$m_{t,t+1}(x_{t+1}) \approx \sum_{m=1}^L \frac{1}{L} \delta_{x_{t+1}^{(m)}}(x_{t+1}) \quad x_{t+1}^{(m)} \sim \sum_{\ell=1}^L w_t^{(\ell)} p(x_{t+1} | x_t^{(\ell)})$$

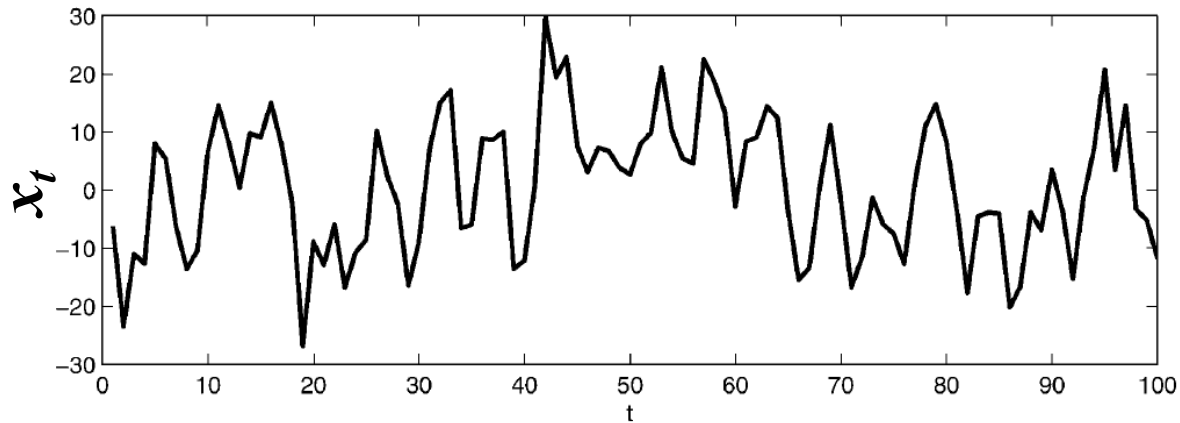
# Bootstrap Particle Filter Summary



# Toy Nonlinear Model

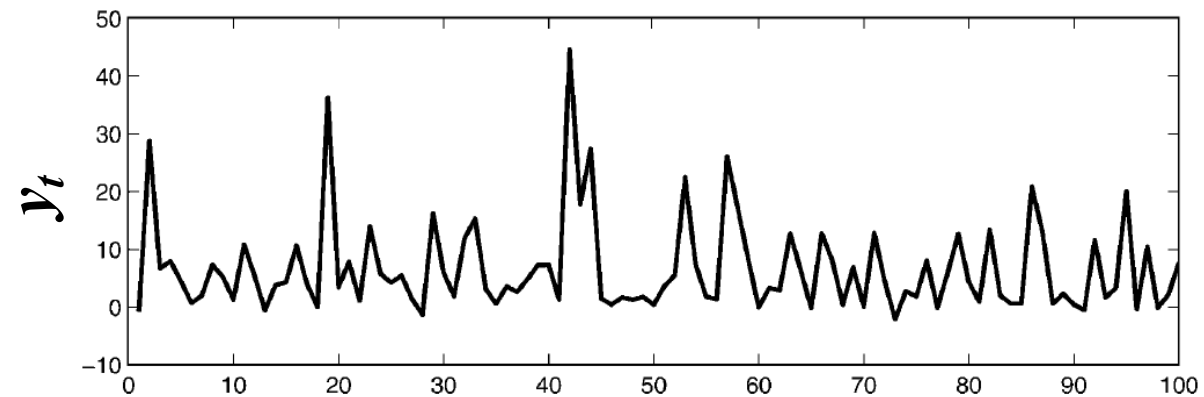
## Nonlinear dynamics and observation model...

### Dynamics



$$x_t = \frac{x_{t-1}}{2} + 25 \frac{x_{t-1}}{1 + x_{t-1}^2} + 8 \cos(1.2t) + u_t$$

### Measurement



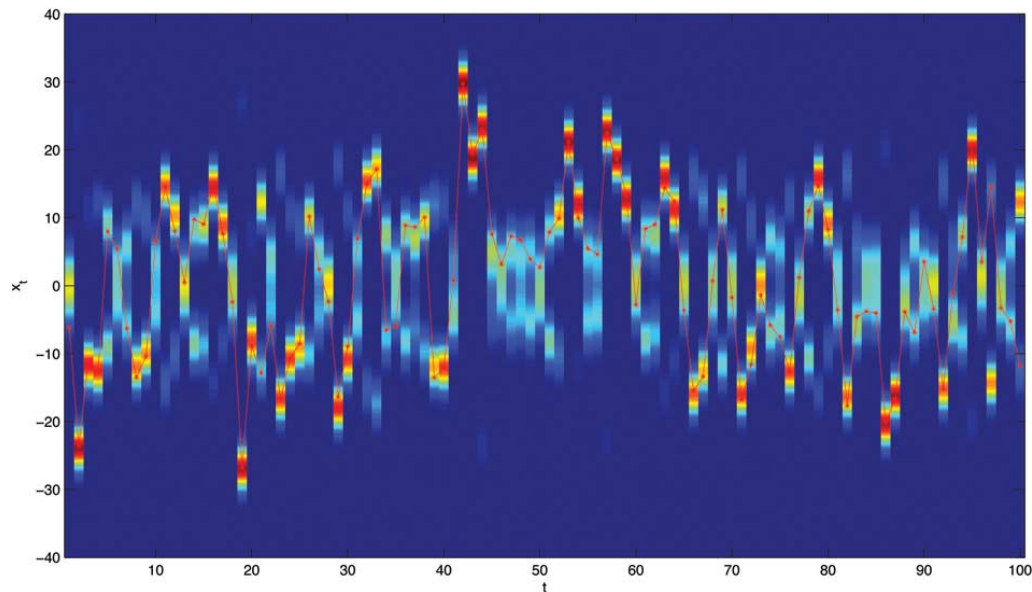
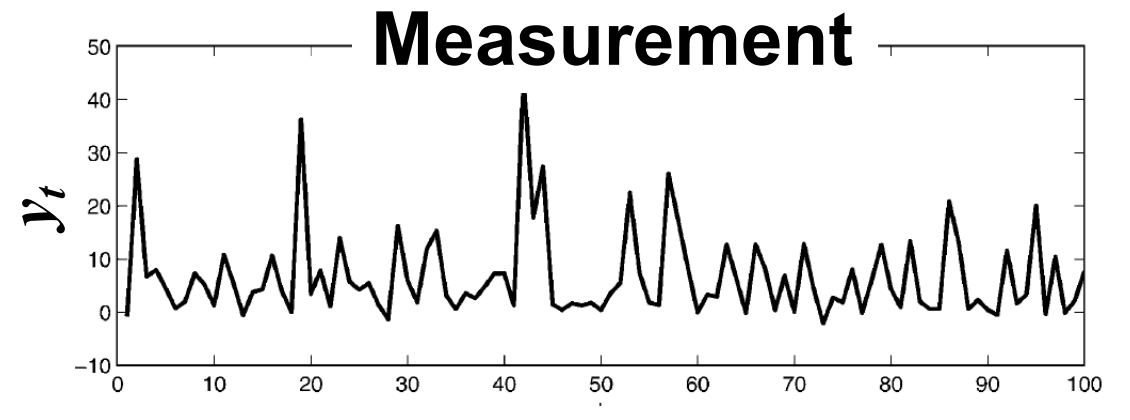
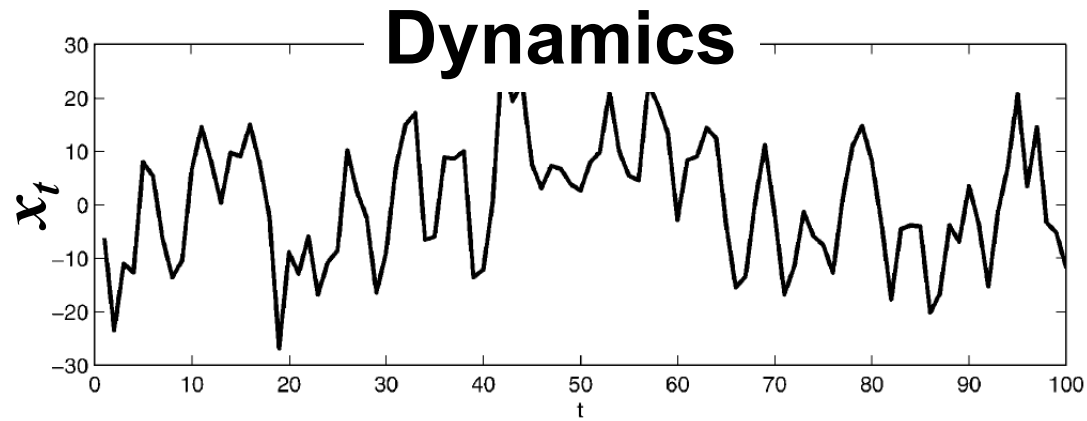
$$y_t = \frac{x_t^2}{20} + v_t$$

Gaussian noise model,  $u_t \sim \mathcal{N}(0, \sigma_x^2)$  and  $v_t \sim \mathcal{N}(0, \sigma_y^2)$

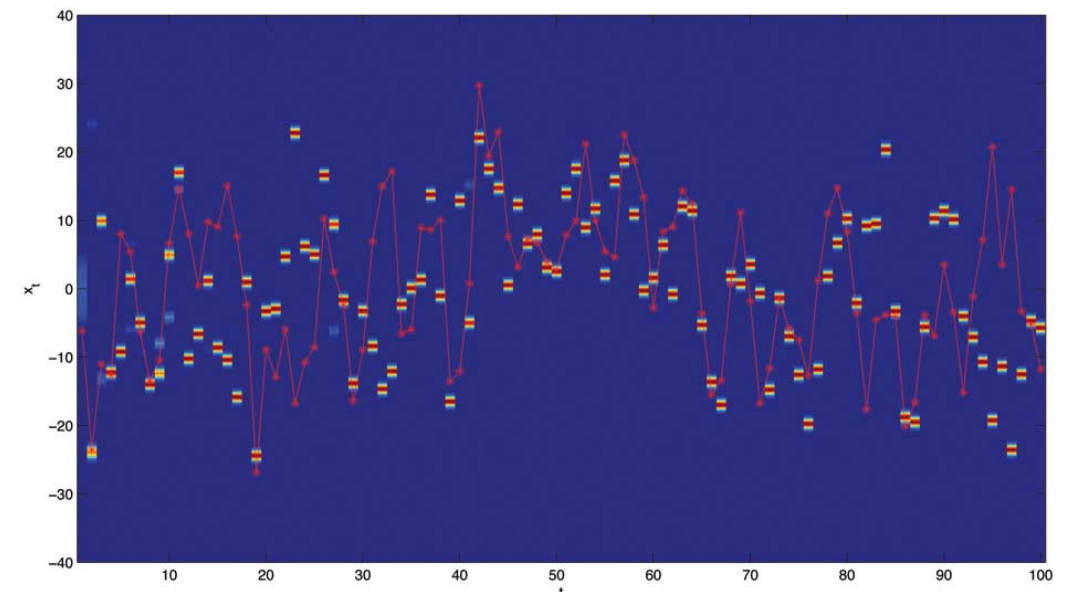
**...filter equations lack closed form.**



# Toy Nonlinear Model



*Particle Filter Marginal KDEs*



*Full Sequence Importance Sampling*

*What is the probability that a state sequence, sampled from the prior model, is consistent with all observations?*

# A More General Particle Filter

- Assume sample-based approximation of previous state's marginal:

$$p(x_{t-1} | y_{t-1}, \dots, y_1) \approx \sum_{\ell=1}^L \frac{1}{L} \delta_{x_{t-1}^{(\ell)}}(x_{t-1})$$

- Sample from a *proposal distribution*  $q$ :

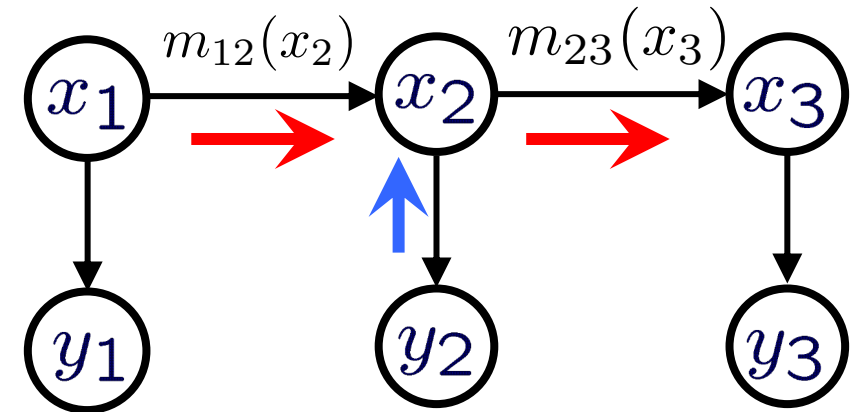
$$x_t^{(\ell)} \sim q(x_t | x_{t-1}^{(\ell)}, y_t) \approx p(x_t | x_{t-1}^{(\ell)}, y_t)$$

- Account for *observation and proposal* via importance weights:

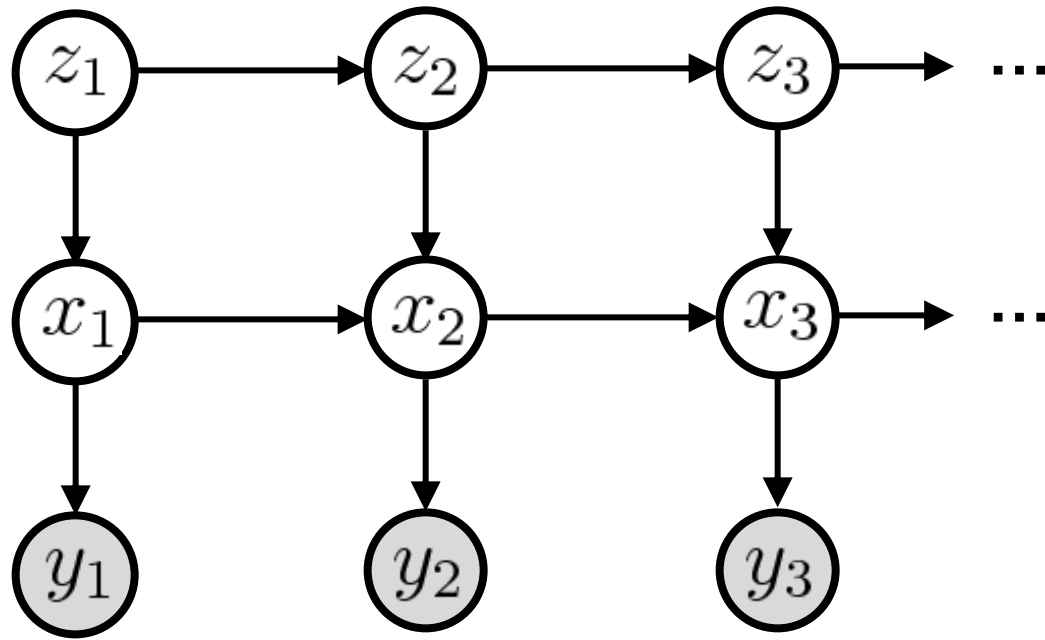
$$w_t^{(\ell)} \propto \frac{p(x_t^{(\ell)} | x_{t-1}^{(\ell)})p(y_t | x_t^{(\ell)})}{q(x_t^{(\ell)} | x_{t-1}^{(\ell)}, y_t)}$$

- Resample to avoid particle degeneracy:

$$p(x_t | y_t, \dots, y_1) \approx \sum_{\ell=1}^L \frac{1}{L} \delta_{x_t^{(\ell)}}(x_t) \quad x_t^{(\ell)} \sim \sum_{m=1}^L w_t^{(m)} \delta_{x_t^{(m)}}(x_t)$$



# Switching State-Space Model

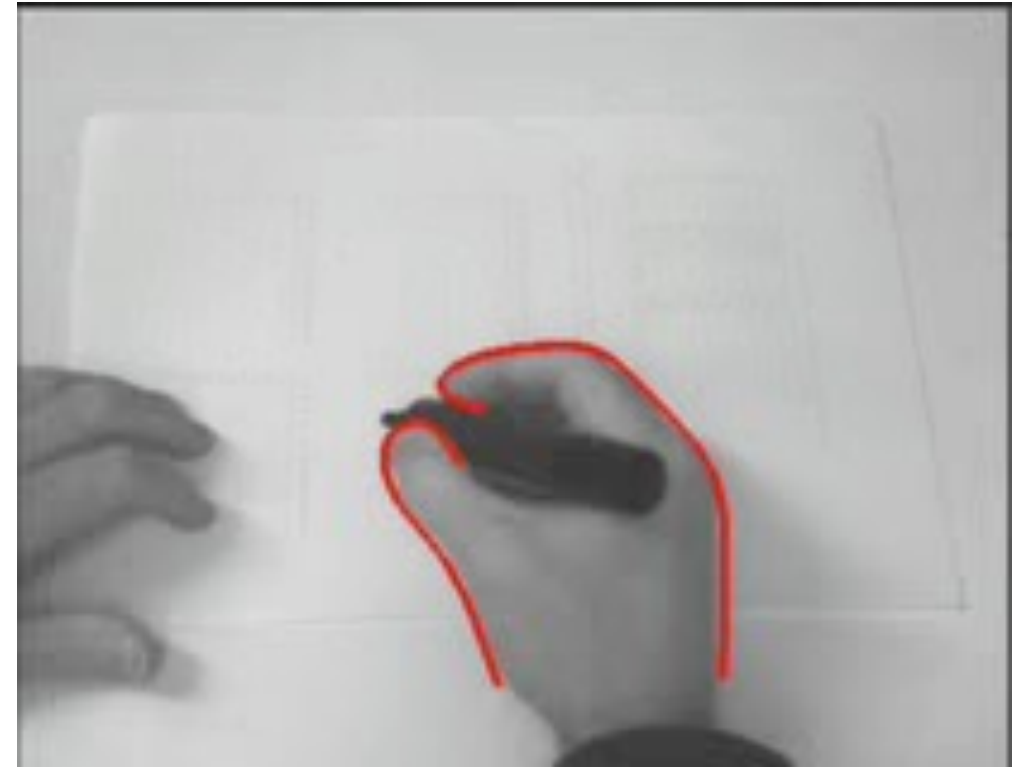


**Discrete switching state:**

$$z_t \mid z_{t-1} \sim \text{Cat}(\pi(z_{t-1})) \quad \text{With stochastic transition matrix } \pi$$

**Switching state selects dynamics:**

$$x_t \mid x_{t-1} \sim \mathcal{N}(A_{z_t} x_{t-1}, \Sigma_{z_t}) \quad (\text{e.g. Nonlinear Gaussian})$$

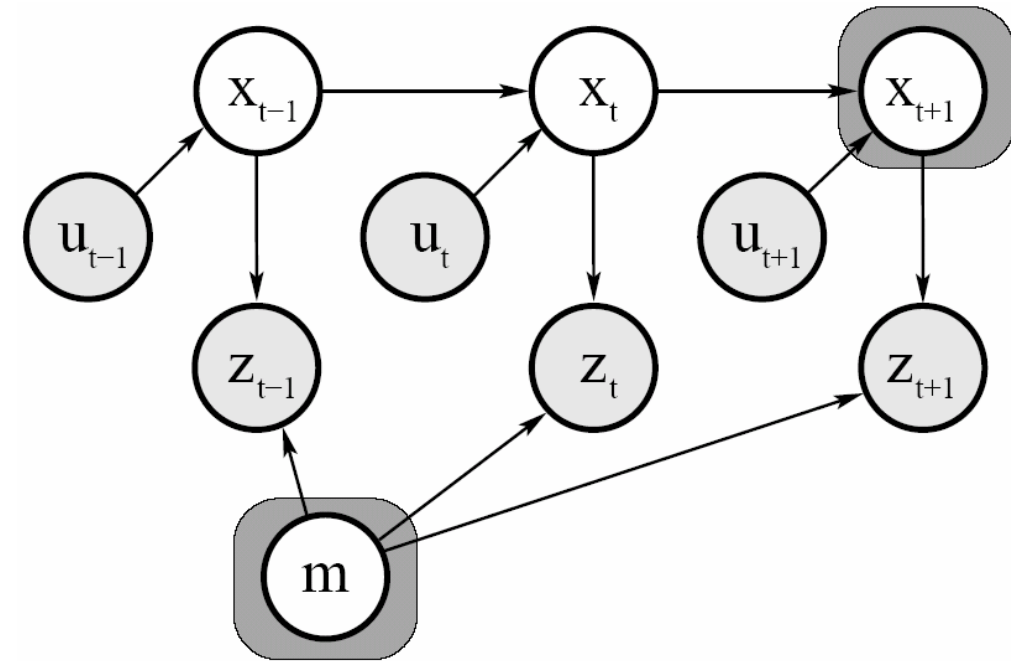


**Colors indicate 3 writing modes**

[ Video: Isard & Blake, ICCV 1998. ]

# Example: Particle Filters for SLAM

*Simultaneous Localization & Mapping (FastSLAM, Montemerlo 2003)*



$$p(x_t, m \mid z_{1:t}, u_{1:t})$$

$x_t$  = State of the robot at time  $t$

$m$  = Map of the environment

$z_{1:t}$  = Sensor inputs from time 1 to  $t$

$u_{1:t}$  = Control inputs from time 1 to  $t$

*Raw odometry (controls)*

*True trajectory (GPS)*

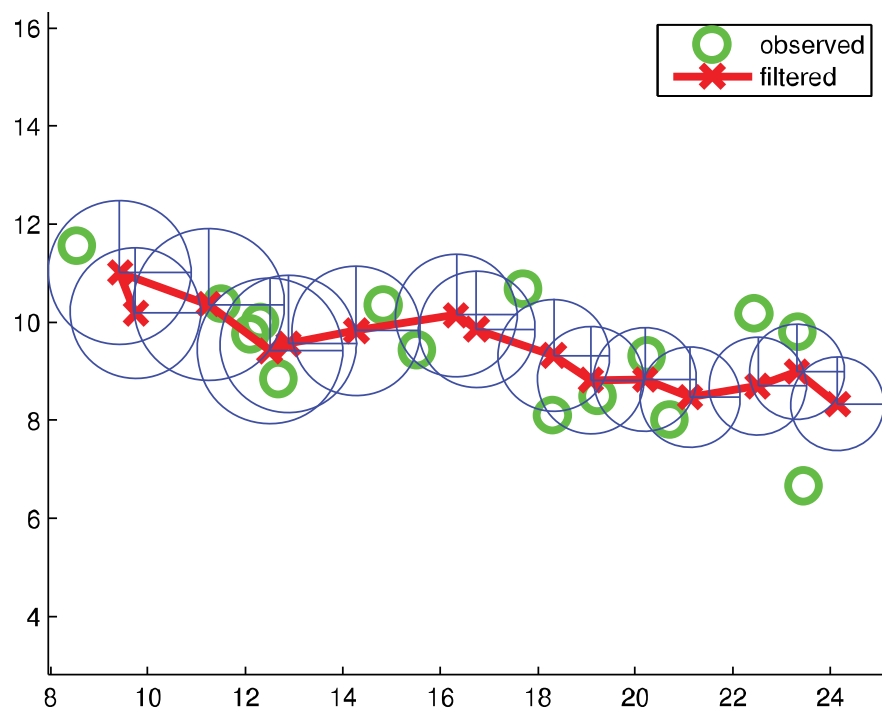
*Inferred trajectory & landmarks*



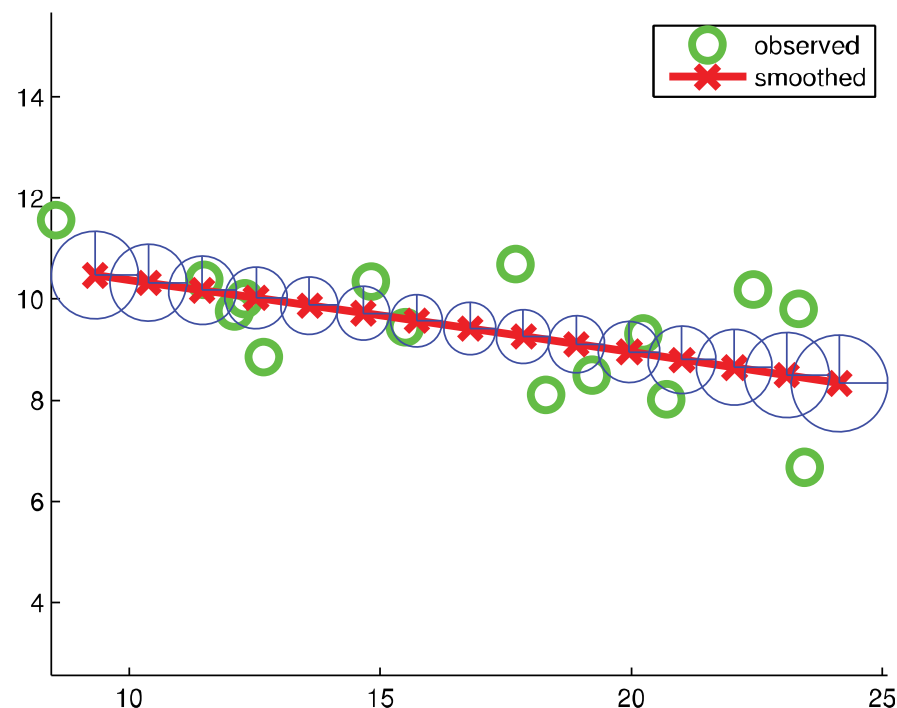
# Dynamical System Inference

Define shorthand notation:  $y_1^{t-1} \triangleq \{y_1, \dots, y_{t-1}\}$

## Filtering



## Smoothing



Compute  $p(x_t | y_1^t)$  at each time  $t$

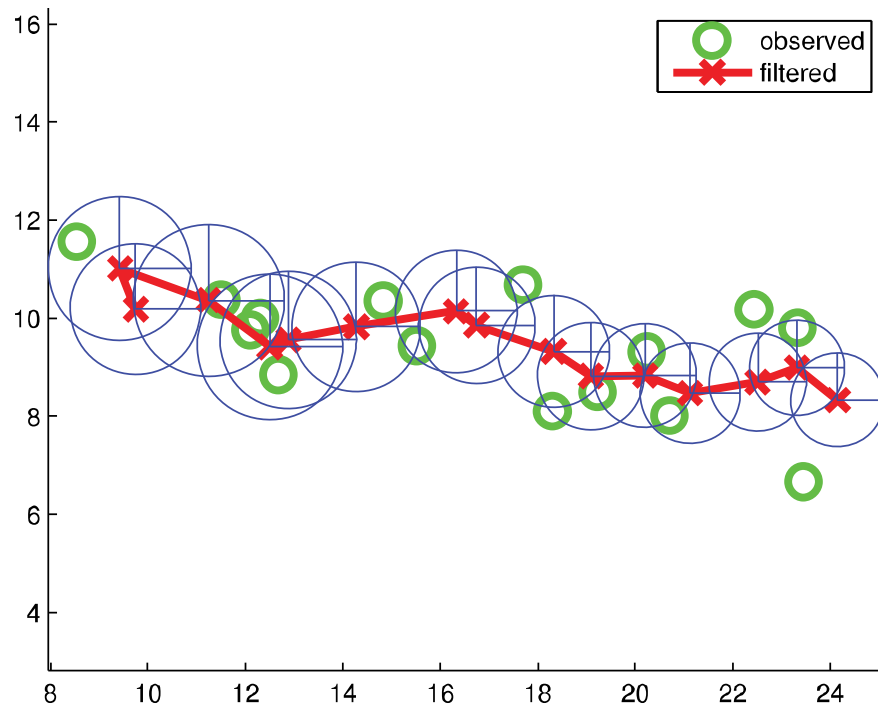
Compute full posterior marginal  $p(x_t | y_1^T)$  at each time  $t$



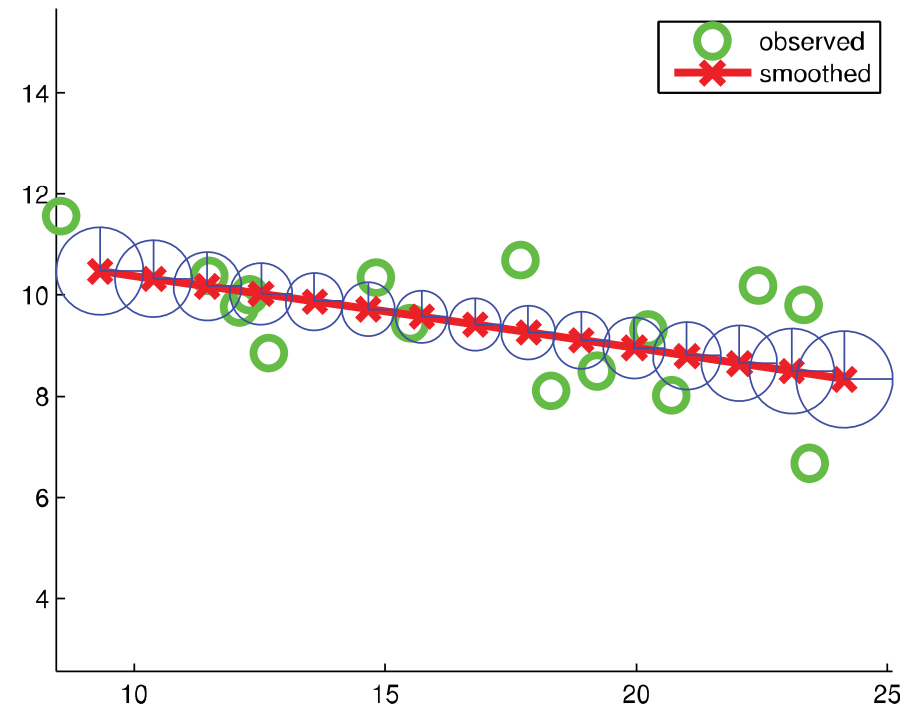
# Dynamical System Inference

Define shorthand notation:  $y_1^{t-1} \triangleq \{y_1, \dots, y_{t-1}\}$

## Filtering

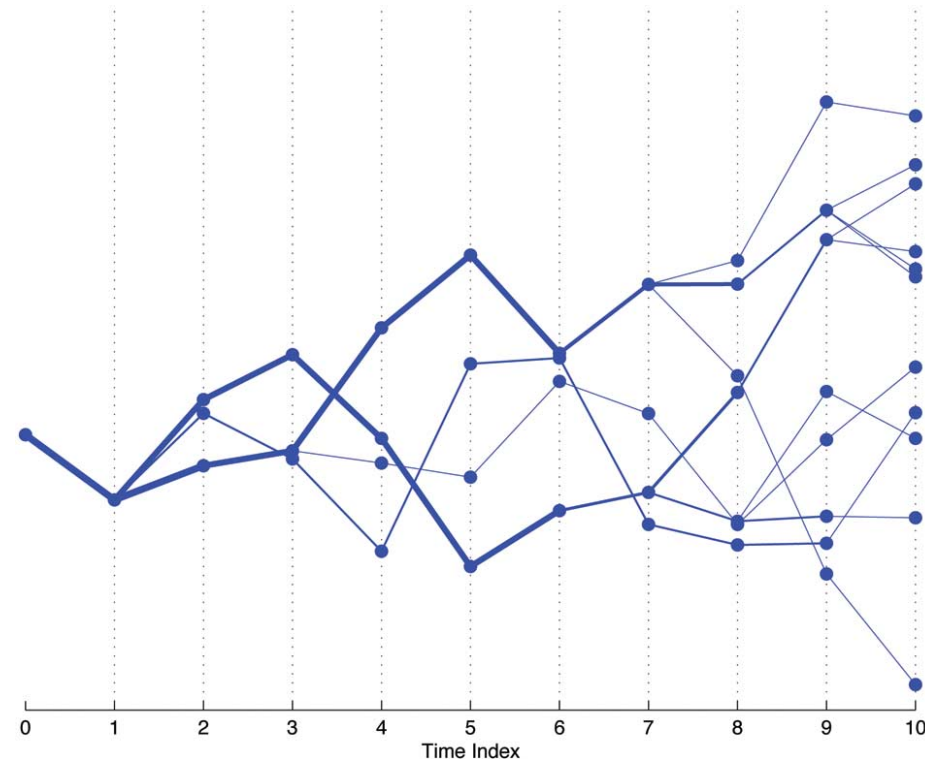


## Smoothing



If estimates at time  $t$  are not needed *immediately*, then better *smoothed* estimates are possible by incorporating future observations

# A Note On Smoothing



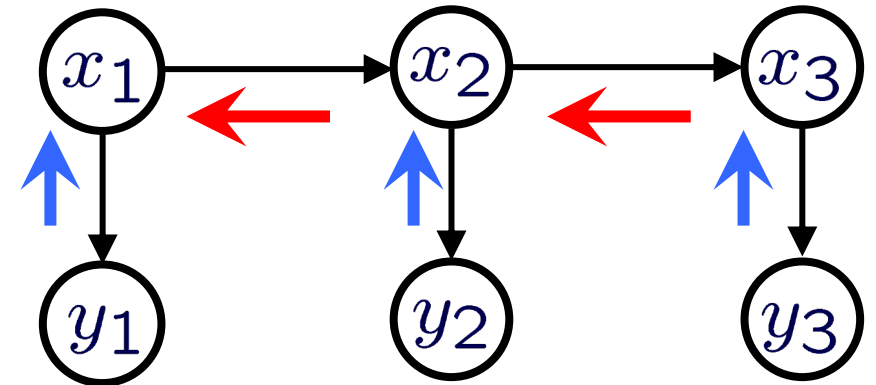
- Each resampling step discards states and they cannot subsequently be restored
- Resampling introduces dependence across trajectories (common ancestors)
- Smoothed marginal estimates are generally poor
- Backwards simulation improves estimates of smoothed trajectories

# Particle Filter Smoothing

Smoothing distribution factorizes as,

$$p(x_1^T | y_1^T) = p(x_T | y_1^T) \prod_{t=1}^{T-1} p(x_t | x_{t+1}, y_1^T)$$
$$= p(x_T | y_1^T) \prod_{t=1}^{T-1} p(x_t | x_{t+1}, y_1^t)$$

Filter distribution at time T



Markov property removes dependence on  $y_{t+1} \dots y_T$

Suggests an algorithm to sample from  $p(x_1^T | y_1^T)$ :

1. Compute and store filter marginals,  $p(x_t | y_1^t)$  for  $t=1, \dots, T$
2. Sample final state from full posterior marginal,  $x_T \sim p(x_T | y_1^T)$
3. Sample in reverse for  $t=(T-1), (T-2), \dots, 2, 1$  from,  $x_t \sim p(x_t | x_{t+1}, y_1^t)$

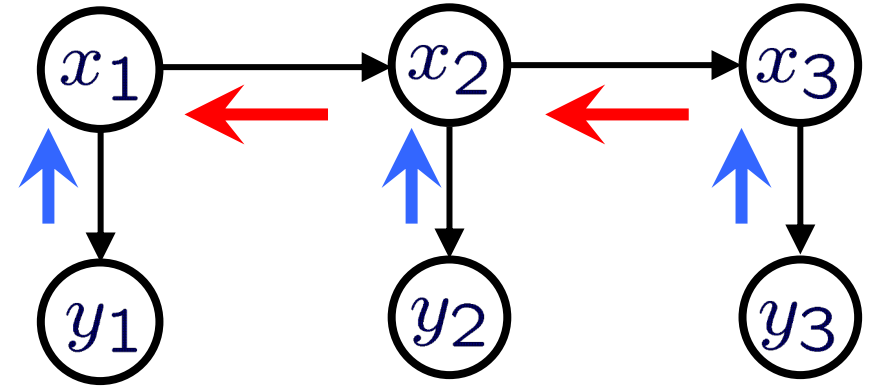
**Use resampling idea to sample from current particle trajectories in reverse**



# Particle Filter Smoothing

Reverse conditional given by def'n of conditional prob.:

$$p(x_t | x_{t+1}, y_1^t) = \frac{p(x_{t+1} | x_t) p(x_t | y_1^t)}{p(x_{t+1} | y_1^t)} \\ \propto p(x_{t+1} | x_t) p(x_t | y_1^t)$$



Forward pass sample-based filter marginal estimates:

$$p(x_t | y_1^t) \approx \sum_{\ell=1}^L w_t^{(\ell)} \delta(x_t - x_t^{(\ell)})$$

Thus particle estimate of reverse prediction is:

$$p(x_t | x_{t+1}, y_1^T) \approx \sum_{\ell=1}^L \rho_t^{(\ell)}(x_{t+1}) \delta(x_t - x_t^{(i)}) \quad \text{where} \quad \rho_t^{(i)}(x_{t+1}) = \frac{w_t^{(i)} p(x_{t+1} | x_t^{(i)})}{\sum_{l=1}^L w_t^{(l)} p(x_{t+1} | x_t^{(l)})}$$

# Particle Filter Smoothing

---

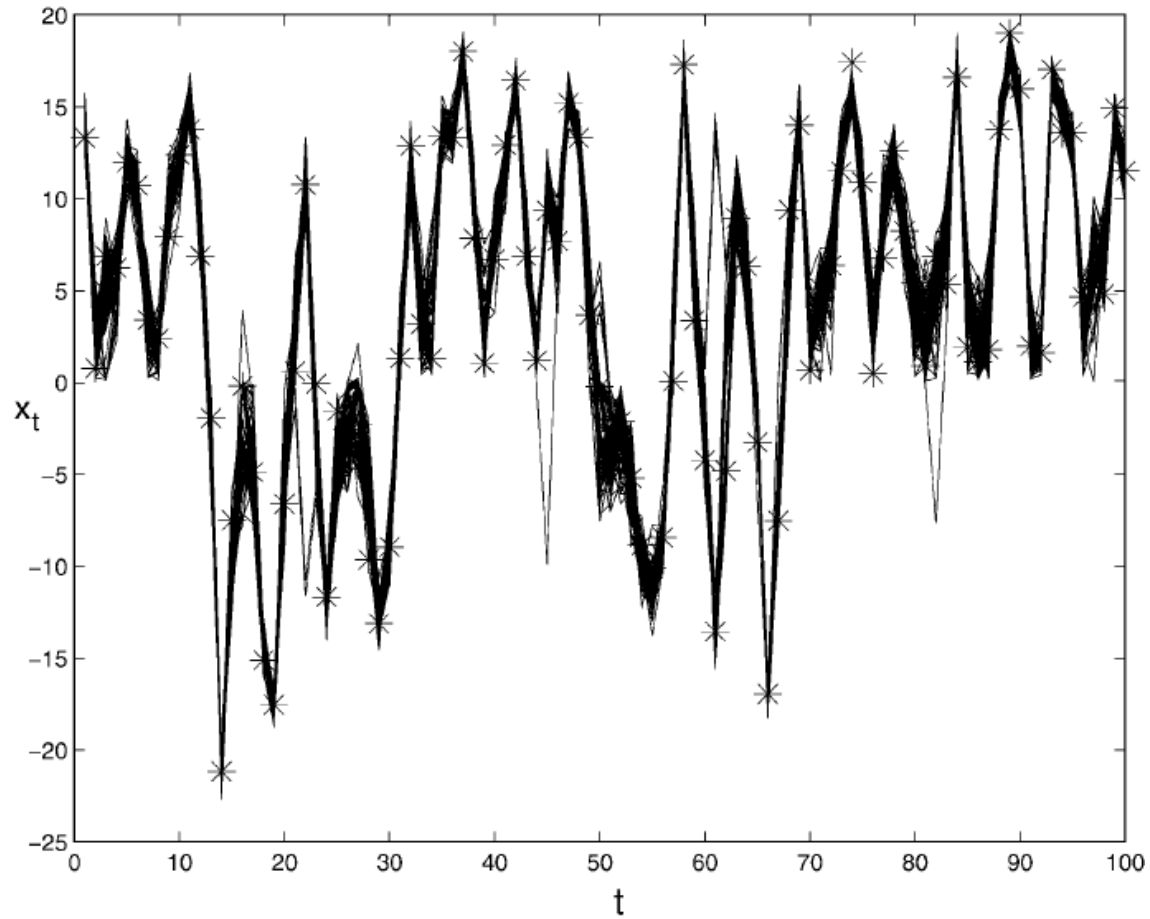
**Algorithm 5** Particle Smoother

---

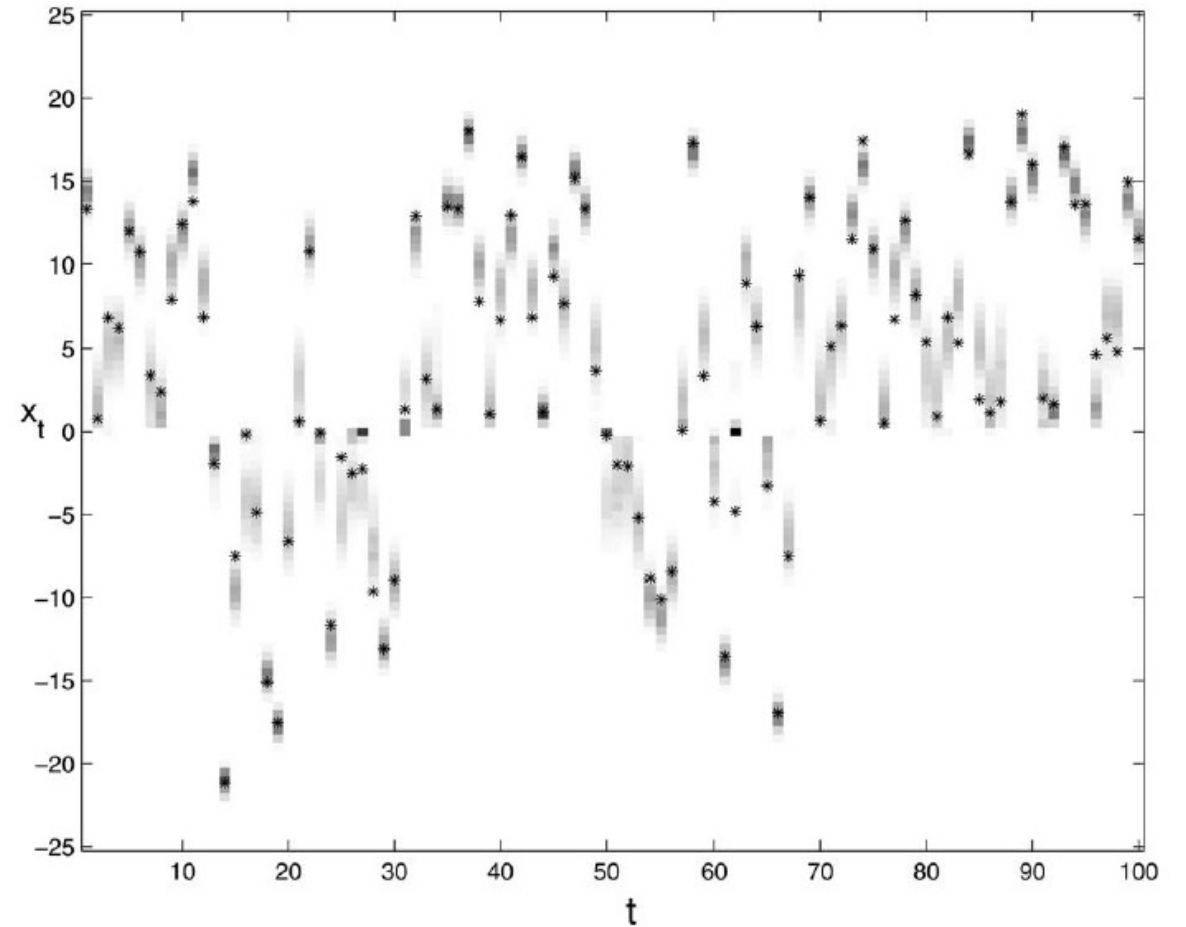
**for**  $t = 0$  to  $T$  **do** ▷ Forward Pass Filter  
Run Particle filter, storing at each time step the particles  
and weights  $\{x_t^{(i)}, \omega_t^{(i)}\}_{1 \leq i \leq L}$   
**end for**  
Choose  $\tilde{x}_T = x_T^{(i)}$  with probability  $\omega_T^{(i)}$ .  
**for**  $t = T - 1$  to  $1$  **do** ▷ Backward Pass Smoother  
Calculate  $\rho_t^{(i)} \propto \omega_t^{(i)} p(\tilde{x}_{t+1} | x_t^{(i)})$  for  $i = 1, \dots, L$  and  
normalize the modified weights.  
Choose  $\tilde{x}_t = x_t^{(i)}$  with probability  $\rho_t^{(i)}$ .  
**end for**

---

# Particle Smoothing Example



Smoothing trajectories for  $T=100$ .  
True states (\*).



Kernel density estimates based on  
smoothed trajectories. True states (\*).

# Additional Particle Filter Topics

- Auxiliary particle filter – bias samples towards those more likely to “survive”
- Rao-Blackwell PF – analytically marginalize tractable sub-components of the state (e.g. linear Gaussian terms)
- MCMC PF – apply MC kernel with correct target  $p(x_1^t | y_1^t)$  to sample trajectory prior to the resampling step
- Other smoothing topics:
  - Generalized two-filter smoothing
  - MC approximation of posterior marginals  $p(x_t | y_1^T)$
- *Maximum a posteriori* (MAP) particle filter
- Maximum likelihood parameter estimation using PF

# Sequential Monte Carlo Summary

- Importance sampling for inference in nonlinear dynamical systems
- Using model dynamics as proposal allows recursive weight updates

$$q(x | y) = q(x_0) \prod_{t=1}^T p(x_t | x_{t-1}) \quad w_t^{(\ell)} \propto w_{t-1}^{(\ell)} p(y_t | x_t^{(\ell)})$$

- All but one weight go to zero as prior/posterior diverge (degeneracy)
- Periodic resampling (with replacement) avoids weight degeneracy
- Each resampling step increases estimator variance (use sparingly)
- In practice, resample when effective sample size (ESS) below thresh

# Outline

- Monte Carlo Estimation
- Sequential Monte Carlo
- **Markov Chain Monte Carlo**

# Monte Carlo Estimation

*One reason to sample a distribution is to approximate expected values under that distribution...*

Expected value of function  $f(x)$  w.r.t. distribution  $p(x)$  given by,

$$\mathbb{E}_p[f(x)] = \int p(x)f(x) dx \equiv \mu$$

- Doesn't always have a closed-form for arbitrary functions
- Suppose we have iid samples:  $\{x_i\}_{i=1}^N \sim p(x)$
- *Monte Carlo* estimate of expected value,

$$\hat{\mu}_N = \frac{1}{N} \sum_{i=1}^N f(x_i) \approx \mathbb{E}_p[f(x)]$$

**Samples must be independent!**

# Markov chain Monte Carlo methods

- The approximations of expectation that we have looked at so far have assumed that the samples are independent draws.
- This sounds good, but in high dimensions, we do not know how to get good independent samples from the distribution.
- MCMC methods drop this requirement.
- Basic intuition
  - If you have finally found a region of high probability, stick around for a bit, enjoy yourself, grab some more samples.



# Markov chain Monte Carlo methods

- Samples are conditioned on the previous one (this is the Markov chain).
- MCMC is often a good hammer for complex, high dimensional, problems.
- Main downside is that it is not “plug-and-play”
  - Doing well requires taking advantage to the structure of your problem
  - MCMC tends to be expensive (but take heart---there may not be any other solution, and at least your problem is being solved).
  - If there are faster solutions, you can incorporate that (and MCMC becomes a way to improve/select these good guesses).

# Metropolis Algorithm

We want samples  $z^{(1)}, z^{(2)}, \dots$

Again, write  $p(z) = \tilde{p}(z)/Z$

Assume that  $q(z|z^{(prev)})$  can be sampled easily

Also assume that  $q(\cdot)$  is symmetric, i.e.,  $q(z_A|z_B) = q(z_B|z_A)$

For example,  $q(z|z^{(prev)}) \sim \mathbb{N}(z; z^{(prev)}, \sigma^2)$

# Metropolis Algorithm

While not\_bored

{

Sample  $q(z|z^{(prev)})$

Accept with probability  $A(z, z^{(prev)}) = \min\left(1, \frac{\tilde{p}(z)}{\tilde{p}(z^{(prev)})}\right)$

If accept, emit  $z$ , otherwise, emit  $z^{(prev)}$ .

}

Always emit one or the other

If things get better, always accept. If they get worse, sometimes accept.

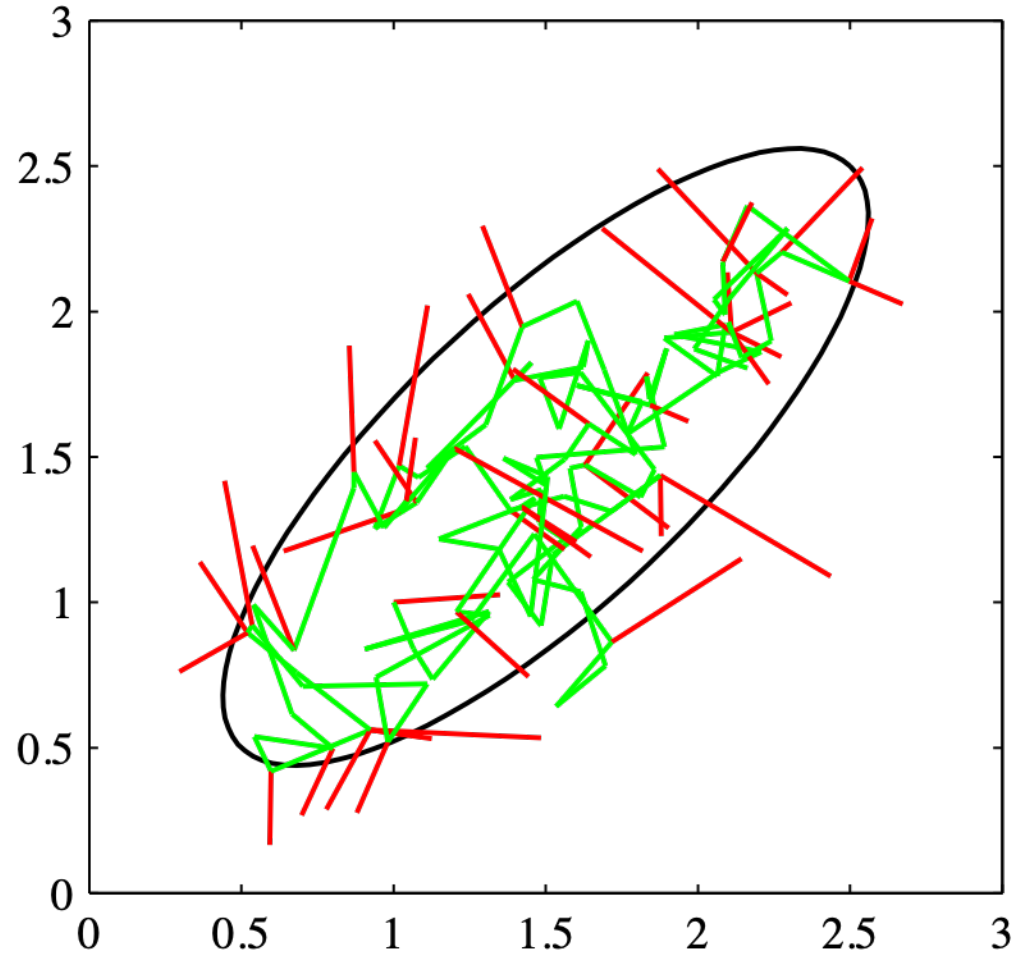
# Metropolis Algorithm

Note that

$$A(z, z^{(prev)}) = \min\left(1, \frac{\tilde{p}(z)}{\tilde{p}(z^{(prev)})}\right) = \min\left(1, \frac{p(z)}{p(z^{(prev)})}\right)$$

So we do not need to normalize  $p(z)$

# Metropolis Example



Green follows accepted proposals  
Red are rejected moves.

# Markov chain view

Denote an initial probability distribution by  $p(z^{(1)})$

Define transition probabilities by:

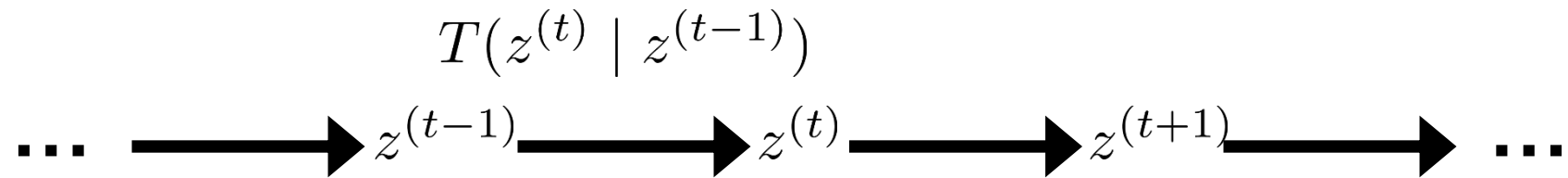
$$T(z^{(prev)}, z) = p(z | z^{(prev)}) \quad (\text{a probability distribution})$$

$T$  can change over time, but for now, assume that it is always the same (homogeneous chain)

A given chain evolves from a sample of  $p(z^{(1)})$ , and is an instance from an ensemble of chains.

# Markov Chain Monte Carlo (MCMC)

- Stochastic 1<sup>st</sup> order Markov process with transition kernel:



- Each  $x^{(t)}$  full N-dimensional state vector
- MCMC samples  $\dots, z^{(t-1)}, z^{(t)}, z^{(t+1)}, \dots$  **not independent**
- New superscript notation indicates dependence:

$$\{z^{(\ell)}\}_{\ell=1}^L \quad \{z^{(t)}\}_{t=1}^T$$

Independent                  Dependent

**Key Question:** How many MCMC samples T are needed to draw L independent samples from  $p(x)$ ?

# Stationary Markov chains

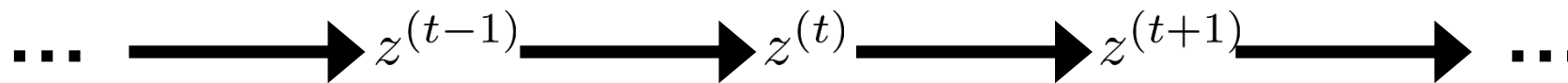
- Recall that our goal is to have our Markov chain emit samples from our **target distribution**  $p(z)$ .
- This implies that the distribution being sampled at time  $t+1$  would be the same as that of time  $t$  (**stationary**).
- If our stationary (target) distribution is  $p()$ , then if we imagine an ensemble of chains, they are in each state with (long-run) probability  $p()$ .
  - On average, a switch from  $s_1$  to  $s_2$  happens as often as going from  $s_2$  to  $s_1$ , otherwise, the percentage of states would not be stable.



# Markov Chain Monte Carlo (MCMC)

- Stochastic 1<sup>st</sup> order Markov process with transition kernel:

$$T(z^{(t)} | z^{(t-1)})$$

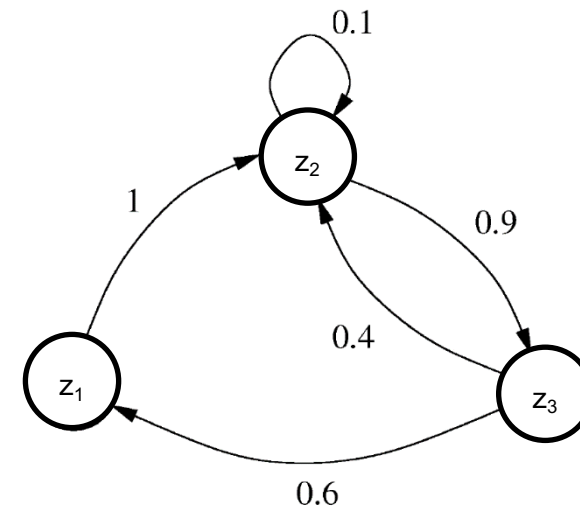


E.g. Let,  $T = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0.1 & 0.9 \\ 0.6 & 0.4 & 0 \end{bmatrix}$

- Initial state dist'n:  $\mu(z^{(1)}) = (0.5, 0.2, 0.3)$
- Repeated transitions converge to target  $\mu(z^{(1)}) \cdot T \cdot T \cdot \dots \cdot T = (0.2, 0.4, 0.4) = p(z)$

**True for any initial state distribution**

**How can we formalize this?**



[ Source: Andrieu et al. ]

# Detailed balance

- Detailed balance is defined by:

$$p(z)T(z,z') = p(z')T(z',z)$$

(We assume that  $T(\cdot) > 0$ )

- Detailed balance is a sufficient condition for  $p()$  to be a stationary distribution with respect to the positive T.

**Sufficient but *not* necessary**

# Detailed balance implies stationary

$$p(z) = \sum_{z'} p^{(prev)}(z') T(z', z)$$

(because?)

# Detailed balance implies stationary

$$p(z) = \sum_{z'} p^{(prev)}(z') T(z', z) \quad (\text{marginalization})$$
$$= \sum_{z'} p^{(prev)}(z) T(z, z') \quad (\text{because?})$$

# Detailed balance implies stationary

$$\begin{aligned} p(z) &= \sum_{z'} p^{(prev)}(z') T(z', z) && \text{(marginalization)} \\ &= \sum_{z'} p^{(prev)}(z) T(z, z') && \text{(detailed balance)} \\ &= p^{(prev)}(z) \sum_{z'} T(z, z') && \text{(because?)} \end{aligned}$$

# Detailed balance implies stationary

$$\begin{aligned} p(z) &= \sum_{z'} p^{(prev)}(z') T(z', z) && \text{(marginalization)} \\ &= \sum_{z'} p^{(prev)}(z) T(z, z') && \text{(detailed balance)} \\ &= p^{(prev)}(z) \sum_{z'} T(z, z') && \text{(moving constant out of sum)} \\ &= p^{(prev)}(z) \sum_{z'} p(z'|z) && \text{(because?)} \end{aligned}$$

# Detailed balance implies stationary

$$\begin{aligned} p(z) &= \sum_{z'} p^{(prev)}(z') T(z', z) && \text{(marginalization)} \\ &= \sum_{z'} p^{(prev)}(z) T(z, z') && \text{(detailed balance)} \\ &= p^{(prev)}(z) \sum_{z'} T(z, z') && \text{(moving constant out of sum)} \\ &= p^{(prev)}(z) \sum_{z'} p(z'|z) && \text{(definition of T)} \\ &= p^{(prev)}(z) \sum_{z'} \frac{p(z', z)}{p(z)} && \text{(because?)} \end{aligned}$$

# Detailed balance implies stationary

$$\begin{aligned} p(z) &= \sum_{z'} p^{(prev)}(z') T(z', z) && \text{(marginalization)} \\ &= \sum_{z'} p^{(prev)}(z) T(z, z') && \text{(detailed balance)} \\ &= p^{(prev)}(z) \sum_{z'} T(z, z') && \text{(moving constant out of sum)} \\ &= p^{(prev)}(z) \sum_{z'} p(z'|z) && \text{(definition of T)} \\ &= p^{(prev)}(z) \sum_{z'} \frac{p(z', z)}{p(z)} && \text{(definition of "|")} \\ &= p^{(prev)}(z) \frac{p(z)}{p(z)} && \text{(because?)} \end{aligned}$$



# Detailed balance implies stationary

$$\begin{aligned} p(z) &= \sum_{z'} p^{(prev)}(z') T(z', z) && \text{(marginalization)} \\ &= \sum_{z'} p^{(prev)}(z) T(z, z') && \text{(detailed balance)} \\ &= p^{(prev)}(z) \sum_{z'} T(z, z') && \text{(moving constant out of sum)} \\ &= p^{(prev)}(z) \sum_{z'} p(z'|z) && \text{(definition of T)} \\ &= p^{(prev)}(z) \sum_{z'} \frac{p(z', z)}{p(z)} && \text{(definition of "|")} \\ &= p^{(prev)}(z) \frac{p(z)}{p(z)} && \text{(marginalization)} \\ &= p^{(prev)}(z) && \text{(because?)} \end{aligned}$$

# Detailed balance implies stationary

$$\begin{aligned} p(z) &= \sum_{z'} p^{(prev)}(z') T(z', z) && \text{(marginalization)} \\ &= \sum_{z'} p^{(prev)}(z) T(z, z') && \text{(detailed balance)} \\ &= p^{(prev)}(z) \sum_{z'} T(z, z') && \text{(moving constant out of sum)} \\ &= p^{(prev)}(z) \sum_{z'} p(z'|z) && \text{(definition of T)} \\ &= p^{(prev)}(z) \sum_{z'} \frac{p(z', z)}{p(z)} && \text{(definition of "|")} \\ &= p^{(prev)}(z) \frac{p(z)}{p(z)} && \text{(marginalization)} \\ &= p^{(prev)}(z) && \text{(canceling)} \end{aligned}$$

# Detailed balance (continued)

- Detailed balance (**for  $p()$** ) means that *if* our chain was generating samples from  $p()$ , it would continue to do so.
  - We will address how it gets there soon.
  - For MCMC algorithms like Metropolis, it is important that the stationary state is the distribution **we want** (most Markov chains converge to *something*),
- Does the Metropolis algorithm have detailed balance?

# Metropolis has detailed balance

Recall that in Metropolis,  $A(z, z') = \min\left(1, \frac{p(z)}{p(z')}\right)$

For detailed balance, we need to show (in general)

$$p(z')T(z', z) = p(z)T(z, z')$$

# Metropolis has detailed balance

Recall that in Metropolis,  $A(z, z') = \min\left(1, \frac{p(z)}{p(z')}\right)$

For detailed balance, we need to show (in general)  
 $p(z')T(z', z) = p(z)T(z, z')$

In Metropolis this is

$$p(z')q(z|z')A(z, z') = p(z)q(z'|z)A(z', z)$$

Probability of transition from  $z$  to  $z'$  is the probability that  $z'$  is proposed, **and** it is accepted.

# Metropolis has detailed balance

Recall that in Metropolis,  $A(z, z') = \min\left(1, \frac{p(z)}{p(z')}\right)$

$$p(z')q(z|z')A(z, z') = q(z|z')\min(p(z'), p(z)) \quad \text{(because?)}$$

# Metropolis has detailed balance

Recall that in Metropolis,  $A(z, z') = \min\left(1, \frac{p(z)}{p(z')}\right)$

$$\begin{aligned} p(z')q(z|z')A(z, z') &= q(z|z')\min(p(z'), p(z)) && \text{(bring } p(z') \text{ into } A) \\ &= q(z'|z)\min(p(z'), p(z)) && \text{(because?)} \end{aligned}$$

# Metropolis has detailed balance

Recall that in Metropolis,  $A(z, z') = \min\left(1, \frac{p(z)}{p(z')}\right)$

$$\begin{aligned} p(z')q(z|z')A(z, z') &= q(z|z')\min(p(z'), p(z)) && \text{(bring } p(z') \text{ into } A) \\ &= q(z'|z)\min(p(z'), p(z)) && q() \text{ is symmetric} \\ &= p(z)q(z'|z)\min\left(\frac{p(z')}{p(z)}, 1\right) && \text{(because?)} \end{aligned}$$



# Metropolis has detailed balance

Recall that in Metropolis,  $A(z, z') = \min\left(1, \frac{p(z)}{p(z')}\right)$

$$\begin{aligned} p(z')q(z|z')A(z, z') &= q(z|z')\min(p(z'), p(z)) && \text{(bring } p(z') \text{ into } A) \\ &= q(z'|z)\min(p(z'), p(z)) && q() \text{ is symmetric} \\ &= p(z)q(z'|z)\min\left(\frac{p(z')}{p(z)}, 1\right) && \text{(divide the min() by } p(z)) \\ &= p(z)q(z'|z)\min\left(1, \frac{p(z')}{p(z)}\right) && \text{(switch order in min())} \\ &= p(z)q(z'|z)A(z', z) && \text{(because?)} \end{aligned}$$

# Metropolis has detailed balance

Recall that in Metropolis,  $A(z, z') = \min\left(1, \frac{p(z)}{p(z')}\right)$

$$\begin{aligned} p(z')q(z|z')A(z, z') &= q(z|z')\min(p(z'), p(z)) && \text{(bring } p(z') \text{ into } A) \\ &= q(z'|z)\min(p(z'), p(z)) && q() \text{ is symmetric} \\ &= p(z)q(z'|z)\min\left(\frac{p(z')}{p(z)}, 1\right) && \text{(divide the min() by } p(z)) \\ &= p(z)q(z'|z)\min\left(1, \frac{p(z')}{p(z)}\right) && \text{(switch order in min())} \\ &= p(z)q(z'|z)A(z', z) && \text{(definition of } A(z', z)) \end{aligned}$$

# Ergodic chains

- Different starting probabilities will give different chains
- We want our chains to converge (in the limit) to the same stationary state, regardless of starting distribution.
- Such chains are called ergodic, and the common stationary state is called the equilibrium state.
- Ergodic chains have a unique equilibrium.

# When do our chains converge?

- Important theorem tells us that for finite state spaces\* our chains converge to equilibrium under two relatively weak conditions.
  - (1) Irreducible
    - We can get from any state to any other state
  - (2) Aperiodic
    - The chain does not get trapped in cycles
- These are true for detailed balance (there exists a stationary state) with  $T > 0$  (you can get there).
  - Detailed balance is sufficient, but not necessary for convergence—it is a stronger property than (1) & (2)

\*Infinite or uncountable state spaces introduces additional complexities, but the main thrust is similar.

# Evolution of ergodic chains

Let  $p^{(t)}(z)$  be the distribution at some time (e.g., initial distribution)

Let  $\pi(z)$  be the stationary distribution

Let  $p^{(t)}(z) = \pi(z) - \Delta^{(t)}(z)$

What is  $p^{(t+1)}(z)$  in terms of  $\pi(z)$  ?

# Evolution of ergodic chains

Let  $p^{(t)}(z)$  be the distribution at some time (e.g., initial distribution)

Let  $\pi(z)$  be the stationary distribution

Let  $p^{(t)}(z) = \pi(z) - \Delta^{(t)}(z)$

$$\begin{aligned} p^{(t+1)}(z) &= \sum_{z'} p^{(t)}(z') T(z, z') \\ &= \sum_{z'} \pi(z') T(z, z') - \sum_{z'} \Delta^{(t)}(z') T(z, z') \\ &= \pi(z) - \Delta^{(t+1)}(z) \end{aligned}$$

# Evolution of ergodic chains

Let  $p^{(t)}(z) = \pi(z) - \Delta^{(t)}(z)$

$$\begin{aligned} p^{(t+1)}(z) &= \sum_{z'} p^{(t)}(z') T(z, z') \\ &= \sum_{z'} \pi(z') T(z, z') - \sum_{z'} \Delta^{(t)}(z') T(z, z') \\ &= \pi(z) - \Delta^{(t+1)}(z) \end{aligned}$$

Cannot die!

Dies out

# Evolution of ergodic chains

$$\begin{aligned} p^{(t+1)}(z) &= \sum_{z'} p^{(t)}(z') T(z, z') \\ &= \sum_{z'} \pi(z') T(z, z') - \sum_{z'} \Delta^{(t)}(z') T(z, z') \\ &= \pi(z) - \Delta^{(t+1)}(z) \end{aligned}$$

Claim that  $|\Delta^{(t)}(z)| < (1 - \nu)^t$

where  $\nu = \min_z \min_{z': \pi(z') > 0} \frac{T(z, z')}{\pi(z)}$

and we have  $0 < \nu \leq 1$



# Matrix-vector representation

Chains (think ensemble) evolve according to:

$$p(z) = \sum_{z'} p(z') T(z', z)$$

Matrix vector representation:

$$\mathbf{p} = \mathbf{T}\mathbf{p}'$$

And, after  $n$  iterations after a starting point:

$$\mathbf{p}^{(n)} = \mathbf{T}^n \mathbf{p}^{(0)}$$

# Matrix representation

A single transition is given by

$$\mathbf{p} = \mathbf{T}\mathbf{p}'$$

Note what happens for stationary state:

$$\mathbf{p}^* = \mathbf{T}\mathbf{p}^*$$

**What does this equation look like?**

So,  $\mathbf{p}^*$  is an eigenvector with eigenvalue one.

# Matrix representation

A single transition is given by

$$\mathbf{p} = \mathbf{T}\mathbf{p}'$$

Note what happens for stationary state:

$$\mathbf{p}^* = \mathbf{T}\mathbf{p}^*$$

So,  $\mathbf{p}^*$  is an eigenvector with eigenvalue one.

And, intuitively, if things converge,  $\mathbf{p}^* = \mathbf{T}^\infty \mathbf{p}^{(0)}$

For any  $\mathbf{p}^{(0)}$ !

# Aside on stochastic matrices

- A right (row) stochastic matrix has non-negative entries, and its rows sum to one.
- A left (column) stochastic matrix has non-negative entries, and its columns sum to one.
- A doubly stochastic matrix has both properties.

# Aside on stochastic matrices

- In our problem,  $T$  is a left (column) stochastic matrix.
  - If you want to be right handed, take the transpose
- The column vector,  $\mathbf{p}$ , also has non-negative elements, that sum to one (stochastic vector).

# Aside on stochastic matrices

- In our problem,  $T$  is a left (column) stochastic matrix.
  - If you want to be right handed, take the transpose
- The column vector,  $\mathbf{p}$ , also has non-negative elements, that sum to one (stochastic vector).
- Fun facts
  - The product of a stochastic matrix and vector is a stochastic vector.
  - The product of two stochastic matrices is a stochastic matrix.

# Aside on (stochastic) matrix powers

Consider the eigenvalue decomposition of  $T$ ,  $T = E\Lambda E^{-1}$

$$T^N = ?$$

# Aside on (stochastic) matrix powers

Consider the eigenvalue decomposition of  $T$ ,  $T = E\Lambda E^{-1}$

$$T^N = E\Lambda^N E^{-1}$$



# Aside on (stochastic) matrix powers

Consider the eigenvalue decomposition of  $T$ ,  $T = E\Lambda E^{-1}$

$$T^N = E\Lambda^N E^{-1}$$

$T^N$  cannot grow without bound,

Why not?

# Aside on (stochastic) matrix powers

Consider the eigenvalue decomposition of  $T$ ,  $T = E\Lambda E^{-1}$

$$T^N = E\Lambda^N E^{-1}$$

$T^N$  cannot grow without bound,  
because it is a stochastic matrix.

# Aside on (stochastic) matrix powers

Consider the eigenvalue decomposition of  $T$ ,  $T = E\Lambda E^{-1}$

$$T^N = E\Lambda^N E^{-1}$$

$T^N$  cannot grow without bound,

because it is a stochastic matrix.

## Logic:

- Product of stochastic matrix is a stochastic matrix
- Columns of (left) stochastic matrix sum to 1
- Power is a bunch of products

# Aside on (stochastic) matrix powers

Consider the eigenvalue decomposition of  $T$ ,  $T = E\Lambda E^{-1}$

$$T^N = E\Lambda^N E^{-1}$$

Since  $T^N$  cannot grow without bound, the eigenvalue magnitudes (remember they can be complex) are inside  $[0,1]$ .

# Aside on (stochastic) matrix powers

Consider the eigenvalue decomposition of  $T$ ,  $T = E\Lambda E^{-1}$

$$T^N = E\Lambda^N E^{-1}$$

Since  $T^N$  cannot grow without bound, the eigenvalue magnitudes (remember they can be complex) are inside  $[0,1]$ .

In fact, for our situation, the second biggest absolute value of the eigenvalues is less than one (not so easy to prove), which also means the biggest one is 1 (otherwise  $T$  will go to zero).

# Aside on (stochastic) matrix powers

We have  $T^N = E\Lambda^N E^{-1}$

$$\Lambda = \begin{pmatrix} 1 & & & \\ & \lambda_2 & & \\ & & \dots & \\ & & & \lambda_K \end{pmatrix} \text{ and } \Lambda^\infty = \begin{pmatrix} & & & \\ & ? & & \\ & & & \\ & & & \end{pmatrix}$$

# Aside on (stochastic) matrix powers

We have  $T^N = E\Lambda^N E^{-1}$

$$\Lambda = \begin{pmatrix} 1 & & & \\ & \lambda_2 & & \\ & & \dots & \\ & & & \lambda_K \end{pmatrix} \text{ and } \Lambda^\infty = \begin{pmatrix} 1 & & & \\ & 0 & & \\ & & \dots & \\ & & & 0 \end{pmatrix}$$

# Aside on (stochastic) matrix powers

We have  $T^N = E\Lambda^N E^{-1}$

$$\Lambda = \begin{pmatrix} 1 & & & \\ & \lambda_2 & & \\ & & \dots & \\ & & & \lambda_K \end{pmatrix} \text{ and } \Lambda^\infty = \begin{pmatrix} 1 & & & \\ & 0 & & \\ & & \dots & \\ & & & 0 \end{pmatrix}$$

$$\Lambda^\infty E^{-1} = \begin{pmatrix} & & & \\ & & & \\ & & & \\ & & & \end{pmatrix}$$



# Aside on (stochastic) matrix powers

We have  $T^N = E\Lambda^N E^{-1}$

$$\Lambda = \begin{pmatrix} 1 & & & \\ & \lambda_2 & & \\ & & \dots & \\ & & & \lambda_K \end{pmatrix} \text{ and } \Lambda^\infty = \begin{pmatrix} 1 & & & \\ & 0 & & \\ & & \dots & \\ & & & 0 \end{pmatrix}$$

$$\Lambda^\infty E^{-1} = \begin{pmatrix} E^{-1}(1,:) \\ \mathbf{0}^T \\ \dots \\ \mathbf{0}^T \end{pmatrix}$$

# Aside on (stochastic) matrix powers

Write  $\mathbf{p}$  in terms of the eigen basis

$$\mathbf{p} = \sum_i a_i \mathbf{e}_i$$

$$E^{-1}(1, :) \cdot \mathbf{p} = \sum_i a_i E^{-1}(1, :) \cdot \mathbf{e}_i = a_1$$

# Aside on (stochastic) matrix powers

Write  $\mathbf{p}$  in terms of the eigen basis

$$\mathbf{p} = \sum_i a_i \mathbf{e}_i$$

$$E^{-1}(1,:) \cdot \mathbf{p} = \sum_i a_i E^{-1}(1,:) \cdot \mathbf{e}_i = a_1$$

$$\left( \begin{array}{l} E^{-1} \cdot E = I \\ \text{And the columns of } E \text{ are } \mathbf{e}_i \\ \text{So, } E^{-1}(1,:) \cdot E = (1,0,0,\dots,0) \\ \text{(first row of the inverse), and} \\ \text{so } E^{-1}(1,:) \cdot \mathbf{e}_1 = 1 \\ \text{and } E^{-1}(1,:) \cdot \mathbf{e}_{i \neq 1} = 0 \end{array} \right)$$

# Aside on (stochastic) matrix powers

Write  $\mathbf{p}$  in terms of the eigen basis

$$\mathbf{p} = \sum_i a_i \mathbf{e}_i$$

$$E^{-1}(1,:) \cdot \mathbf{p} = \sum_i a_i E^{-1}(1,:) \cdot \mathbf{e}_i = a_1$$

$$\left( \begin{array}{l} E^{-1} \cdot E = I \\ \text{And the columns of } E \text{ are } \mathbf{e}_i \\ \text{So, } E^{-1}(1,:) \cdot E = (1,0,0,\dots,0) \\ \text{(first row of the inverse), and} \\ \text{so } E^{-1}(1,:) \cdot \mathbf{e}_1 = 1 \\ \text{and } E^{-1}(1,:) \cdot \mathbf{e}_{i \neq 1} = 0 \end{array} \right)$$

$$\text{and, } \Lambda^\infty E^{-1} \mathbf{p} = \begin{pmatrix} E^{-1}(1,:) \cdot \mathbf{p} \\ 0 \\ \dots \\ 0 \end{pmatrix} = \begin{pmatrix} a_1 \\ 0 \\ \dots \\ 0 \end{pmatrix}$$

# Aside on (stochastic) matrix powers

Recall that we are studying  $E\Lambda^\infty E^{-1}\mathbf{p}$

$$\Lambda^\infty E^{-1}\mathbf{p} = \begin{pmatrix} a_1 \\ 0 \\ \dots \\ 0 \end{pmatrix}$$

So,  $E\Lambda^\infty E^{-1}\mathbf{p} = ?$

# Aside on (stochastic) matrix powers

Recall that we are studying  $\mathbb{E} \Lambda^\infty E^{-1} \mathbf{p}$

$$\Lambda^\infty E^{-1} \mathbf{p} = \begin{pmatrix} a_1 \\ 0 \\ \dots \\ 0 \end{pmatrix}$$

So,  $\mathbb{E} \Lambda^\infty E^{-1} \mathbf{p} = a_1 \mathbf{e}_1$

# Aside on (stochastic) matrix powers

So,  $\mathbf{p}^* = E\Lambda^\infty E^{-1}\mathbf{p}$

$E\Lambda^\infty E^{-1}\mathbf{p} = \mathbf{p}^*$  no matter what the initial point  $\mathbf{p}$  is.

So, glossing over details, we have convergence to equilibrium.

# Justification relies on Perron Frobenius theorem

Let  $A = (a_{ij})$  be an  $n \times n$  positive matrix:  $a_{ij} > 0$  for  $1 \leq i, j \leq n$ . Then the following statements hold.

1. There is a positive real number  $r$ , called the **Perron root** or the **Perron–Frobenius eigenvalue**, such that  $r$  is an eigenvalue of  $A$  and any other eigenvalue  $\lambda$  (possibly, **complex**) is strictly smaller than  $r$  in **absolute value**,  $|\lambda| < r$ . Thus, the **spectral radius**  $\rho(A)$  is equal to  $r$ .
2. The Perron–Frobenius eigenvalue is simple:  $r$  is a simple root of the **characteristic polynomial** of  $A$ . Consequently, the **eigenspace** associated to  $r$  is one-dimensional. (The same is true for the left eigenspace, i.e., the eigenspace for  $A^T$ .)
3. There exists an eigenvector  $v = (v_1, \dots, v_n)$  of  $A$  with eigenvalue  $r$  such that all components of  $v$  are positive:  $A v = r v$ ,  $v_i > 0$  for  $1 \leq i \leq n$ . (Respectively, there exists a positive left eigenvector  $w$ :  $w^T A = r w^T$ ,  $w_i > 0$ .)
4. There are no other positive (moreover non-negative) eigenvectors except  $v$  (respectively, left eigenvectors except  $w$ ), i.e. all other eigenvectors must have at least one negative or non-real component.
5.  $\lim_{k \rightarrow \infty} A^k / r^k = v w^T$ , where the left and right eigenvectors for  $A$  are normalized so that  $w^T v = 1$ . Moreover, the matrix  $v w^T$  is the **projection onto the eigenspace** corresponding to  $r$ . This projection is called the **Perron projection**.
6. **Collatz–Wielandt formula**: for all non-negative non-zero vectors  $x$ , let  $f(x)$  be the minimum value of  $[Ax]_i / x_i$  taken over all those  $i$  such that  $x_i \neq 0$ . Then  $f$  is a real valued function whose **maximum** is the Perron–Frobenius eigenvalue.
7. A "Min-max" Collatz–Wielandt formula takes a form similar to the one above: for all strictly positive vectors  $x$ , let  $g(x)$  be the maximum value of  $[Ax]_i / x_i$  taken over  $i$ . Then  $g$  is a real valued function whose **minimum** is the Perron–Frobenius eigenvalue.
8. The Perron–Frobenius eigenvalue satisfies the inequalities

$$\min_i \sum_j a_{ij} \leq r \leq \max_i \sum_j a_{ij}.$$

From Wikipedia



# Main points about P-F for positive square matrices

- The maximal eigenvalue is strictly maximal and real valued (item 1).
- Its eigenvector (as computed by software\*) has all positive (or negative) real components (item 3).
- The maximal eigenvalue of a stochastic matrix has absolute value 1 (item 8 applied to stochastic matrix).

\*P-F says that the positive version exists, but software might hand you the negative of that, but you can negate it to be consistent with P-F.

# Summary on matrix version of stationarity

$\mathbf{p}^* = \mathbf{T}\mathbf{p}^*$  is an eigenvector with eigenvalue one.

We have written it as  $\mathbf{p}^* \parallel \mathbf{e}^1$  because  $\mathbf{e}^1$  is the eigenvector normalized to norm 1 (not stochastic).

Intuitively (perhaps),  $\mathbf{T}$  will reduce any component of  $\mathbf{p}$  orthogonal to  $\mathbf{p}^*$ , and  $\mathbf{T}^N$  will kill off such components as  $N \rightarrow \infty$ .

Neal '93 provides an algebraic proof which does not rely on spectral theory.

# MCMC so far

- Under reasonable conditions (ergodicity) ensembles of chains over discretized states converge to an equilibrium state (stationary distribution)
- Easiest way to prove (or check) that this is the case is to show **detailed balance** and use  $T>0$  (sufficient but not necessary)
- There is a nice analogy with powers of stochastic matrices, which converge to an operator based on the largest magnitude eigenvector (with  $|\text{eigenvalue}|=1$ )
- In theory, to use MCMC for sampling a distribution, we simply need to ensure that our target distribution is the equilibrium state.
- In practice we do not know even know if we have visited the best place yet. (The ensemble metaphor runs into trouble if you have a small number of chains compared to the number of states).

# MCMC Theory vs. Practice

- The time it takes to get reasonably close to equilibrium (where samples come from the target distribution) is called “burn in” time.
  - I.E., how long does it take to forget the starting state.
  - There is no general way to know when this has occurred.
- The average time it takes to visit a state is called “hit time”.
- What if we really want independent samples?
  - In theory we can take every  $N^{\text{th}}$  sample (some theories about how long to wait exist, but it depends on the algorithm and distribution).

# MCMC for ML in practice

- We use MCMC for machine learning problems with very complex distributions over high dimensional spaces.
- Variables can be either discrete or continuous (often both)
- Despite the gloomy worst case scenario, MCMC is often a good way to find good solutions (either by MAP or integration).
  - Key reason is that there is generally structure in our distributions.
  - We need to exploit this knowledge in our proposal distributions.
  - Instead of getting hung up about whether you actually have convergence
    - Enjoy that fact that what you are doing is principled and can improve any answer (with respect to your model) that you can get by other means
      - Your model should be able to tell you which proposed solution are good.

# A View of Metropolis

[ Source: D. MacKay ]

**Transition kernel** with target distribution:

$$p(x) = \frac{1}{Z} \tilde{p}(x)$$

1. Sample proposal:  $x' \mid x^{(t-1)} \sim q(\cdot)$
2. Accept with probability:

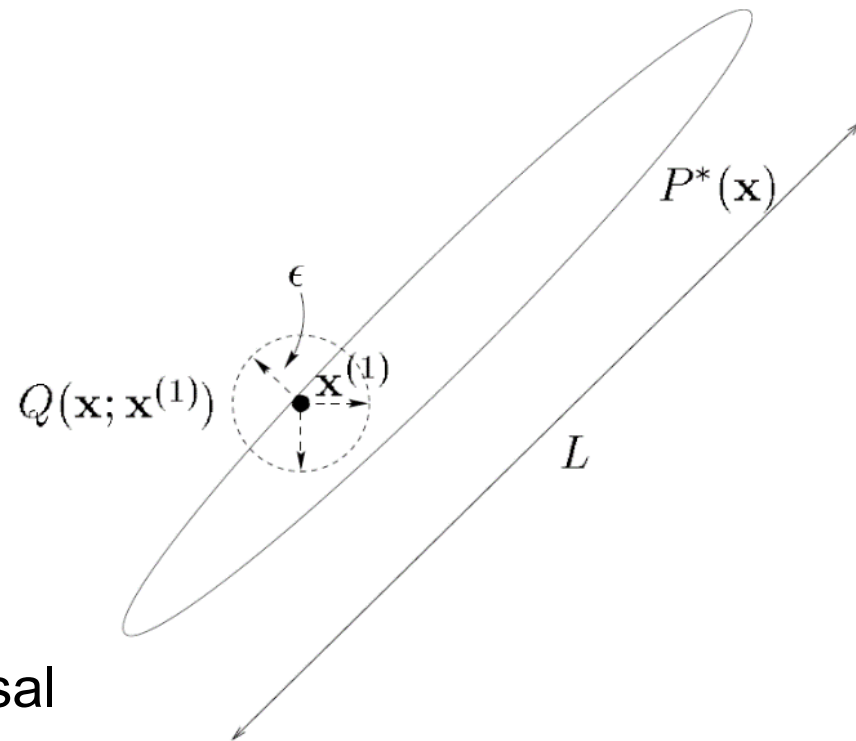
$$\min\{1, a\} \quad \text{where} \quad a = \frac{\tilde{p}(x')}{p(x^{(t-1)})}$$

**Proposal must be symmetric**

$$q(x^{(t)} \mid x^{(t-1)}) = q(x^{(t-1)} \mid x^{(t)})$$

**Example: Symmetric Gaussian proposal**

$$q(x^{(t)} \mid x^{(t-1)}) = \mathcal{N}(x^{(t-1)}, \epsilon^2)$$

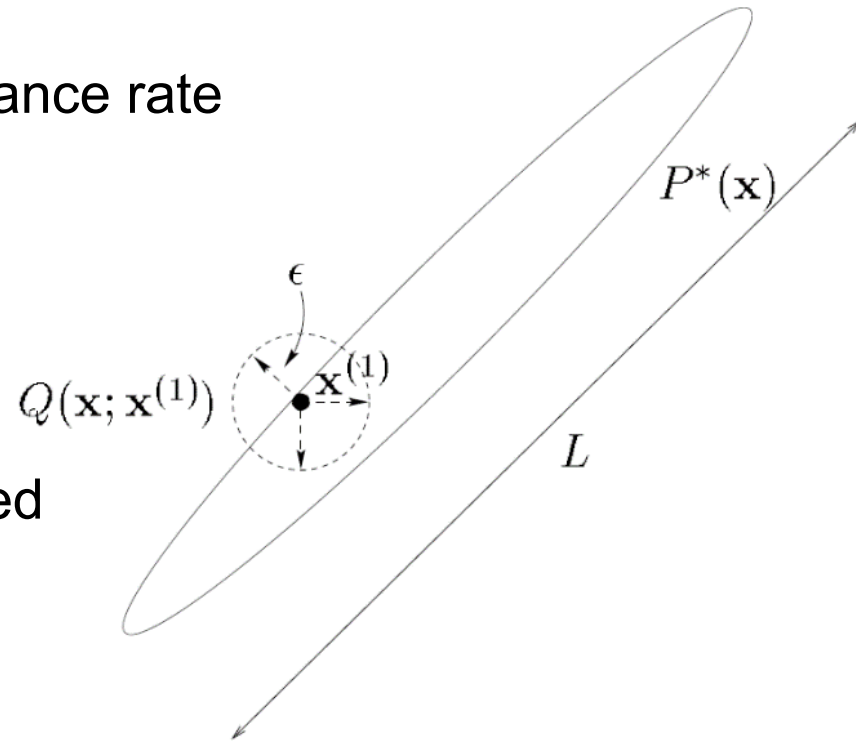


# Metropolis Efficiency

*How many samples needed for an independent sample?*

Consider Gaussian proposal:  $q(x^{(t)} | x^{(t-1)}) = \mathcal{N}(x^{(t-1)}, \epsilon^2)$  [ Source: D. MacKay ]

- Typically  $\epsilon \ll L$  for adequate acceptance rate
- Leads to random walk dynamics that are slow to converge
- Rule of Thumb:  
If average acceptance is  $f \in (0, 1)$  need to run for roughly  $T \approx (L/\epsilon)^2 / f$  iterations for an independent sample



**This is only a lower bound (and potentially very loose)**

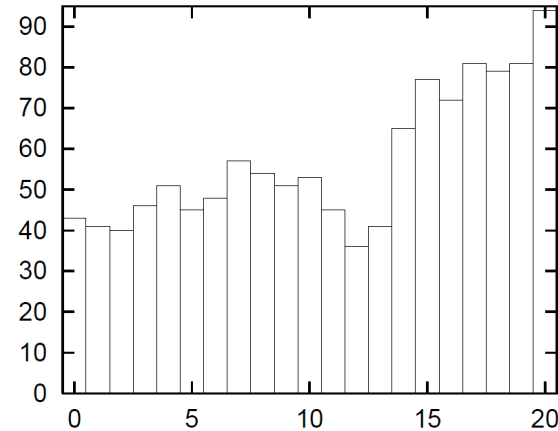
# Example: Random Walk Dynamics

← State evolution for  $t=1\dots 600$ , horizontal bars denote intervals of 50



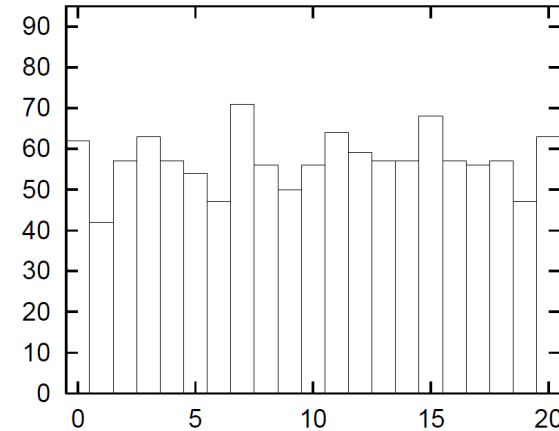
## Metropolis

1200 iterations



## Independent

1200 iterations



[ Source: D. MacKay ]

**Target:** 
$$p(x) = \begin{cases} \frac{1}{21} & x \in \{0, \dots, 20\} \\ 0 & \text{otherwise} \end{cases}$$

**Proposal:** 
$$q(x' | x) = \begin{cases} \frac{1}{2} & x' = x \pm 1 \\ 0 & \text{otherwise} \end{cases}$$

From  $x_0 = 10$  need  $\sim 400$  steps to reach both end states (0 and 20).  
So,  $\sim 400$  steps to generate 1 independent sample!

**Very important to avoid random walk dynamics**



# Beyond the Metropolis Method

Metropolis requires the proposal to be symmetric,

$$q(z' | z) = q(z | z')$$

This often results in a chain that takes a long time to converge to a stationary distribution (long burn in time)

**Example** The most common proposal (Gaussian),

$$q(z' | z) = \mathcal{N}(z' | z, \sigma^2 I)$$

exhibits random walk dynamics that are inefficient

*Metropolis-Hastings relaxes this symmetry requirement...*

# Metropolis-Hastings MCMC method

While not\_bored

{

Sample  $q(z|z^{(prev)})$

Accept with probability  $A(z, z^{(prev)}) = \min\left(1, \frac{\tilde{p}(z)q(z^{(prev)}|z)}{\tilde{p}(z^{(prev)})q(z|z^{(prev)})}\right)$

If accept, emit  $z$ , otherwise, emit  $z^{(prev)}$ .

}

## Does Metropolis-Hastings converge to the target distribution?

- Like Metropolis, but now  $q()$  is not necessarily symmetric.
- If Metropolis-Hastings has detailed balance, then it converges to the target distribution under weak conditions.
  - The converse is not true, but generally samplers of interest will have detailed balance

# Does Metropolis-Hastings have detailed balance?

To show detailed balance we need to show

$$p(z')q(z|z')A(z,z') = p(z)q(z'|z)A(z',z)$$

$$p(z')q(z|z')A(z,z') = \min(p(z')q(z|z'), p(z)q(z'|z))$$

# Does Metropolis-Hastings have detailed balance?

To show detailed balance we need to show

$$p(z')q(z|z')A(z,z') = p(z)q(z'|z)A(z',z)$$

$$p(z')q(z|z')A(z,z') = \min(p(z')q(z|z'), p(z)q(z'|z))$$

$$\left\{ \text{Recall that } A(z,z') = \min\left(1, \frac{p(z)q(z'|z)}{p(z')q(z|z')}\right) \right\}$$

# Does Metropolis-Hastings have detailed balance?

To show detailed balance we need to show

$$p(z')q(z|z')A(z,z') = p(z)q(z'|z)A(z',z)$$

$$\begin{aligned} p(z')q(z|z')A(z,z') &= \min(p(z')q(z|z'), p(z)q(z'|z)) \\ &= p(z)q(z'|z) \min\left(\frac{p(z')q(z|z')}{p(z)q(z'|z)}, 1\right) \end{aligned}$$

# Does Metropolis-Hastings have detailed balance?

To show detailed balance we need to show

$$p(z')q(z|z')A(z,z') = p(z)q(z'|z)A(z',z)$$

$$\begin{aligned} p(z')q(z|z')A(z,z') &= \min(p(z')q(z|z'), p(z)q(z'|z)) \\ &= p(z)q(z'|z) \min\left(\frac{p(z')q(z|z')}{p(z)q(z'|z)}, 1\right) \\ &= p(z)q(z'|z) \min\left(1, \frac{p(z')q(z|z')}{p(z)q(z'|z)}\right) \end{aligned}$$

# Does Metropolis-Hastings have detailed balance?

To show detailed balance we need to show

$$p(z')q(z|z')A(z,z') = p(z)q(z'|z)A(z',z)$$

$$\begin{aligned} p(z')q(z|z')A(z,z') &= \min(p(z')q(z|z'), p(z)q(z'|z)) \\ &= p(z)q(z'|z) \min\left(\frac{p(z')q(z|z')}{p(z)q(z'|z)}, 1\right) \\ &= p(z)q(z'|z) \min\left(1, \frac{p(z')q(z|z')}{p(z)q(z'|z)}\right) \\ &= p(z)q(z'|z)A(z',z) \end{aligned}$$



# Metropolis-Hastings comments

- Again it does not matter if we use unnormalized probabilities in the M-H acceptance ratio  $A(z,z')$
- It should be clear that the Metropolis method (where  $q()$  is symmetric) is a special case of M-H
- $q(z'|z)$  can be anything, but you need to specify the reverse move  $q(z|z')$ , which can be tricky

# MCMC So Far...

## Metropolis Algorithm

- Sample RV from proposal  $z \sim q(z | z^{(\text{prev})})$
- Proposal must be *symmetric*  $q(z | z^{(\text{prev})}) = q(z^{(\text{prev})} | z)$
- Accept with probability  $\min \{ 1, \tilde{p}(z) \div \tilde{p}(z^{(\text{prev})}) \}$

## Metropolis-Hastings Algorithm

- Proposal does not have to be symmetric
- Accept with probability

$$\min \left\{ 1, \frac{\tilde{p}(z)q(z^{(\text{prev})} | z)}{\tilde{p}(z^{(\text{prev})})q(z | z^{(\text{prev})})} \right\}$$

*Both methods require choosing proposal, which can be hard*

# Gibbs Sampling

Let  $p(x)$  be the target distribution on random variables,

$$x_1, x_2, \dots, x_N$$

Consider the *complete conditional distribution*  $x_i$

$$p(x_i \mid x_{\neg i})$$

where  $x_{\neg i} = \{x_1, x_2, \dots, x_N\} \setminus x_i$  all RVs *except*  $x_i$

**Idea** Don't sample all RVs from one proposal. Sample each from its corresponding complete conditional,

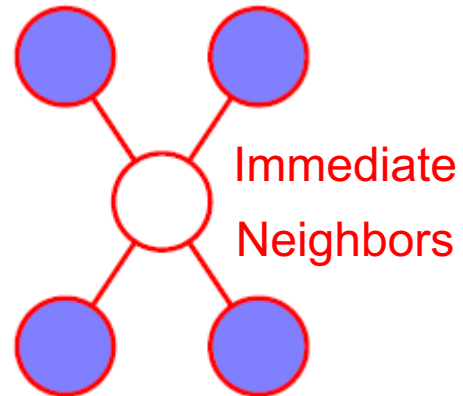
$$q_i(x_i \mid x_{\neg i}) = p(x_i \mid x_{\neg i})$$

*We call this method Gibbs Sampling*

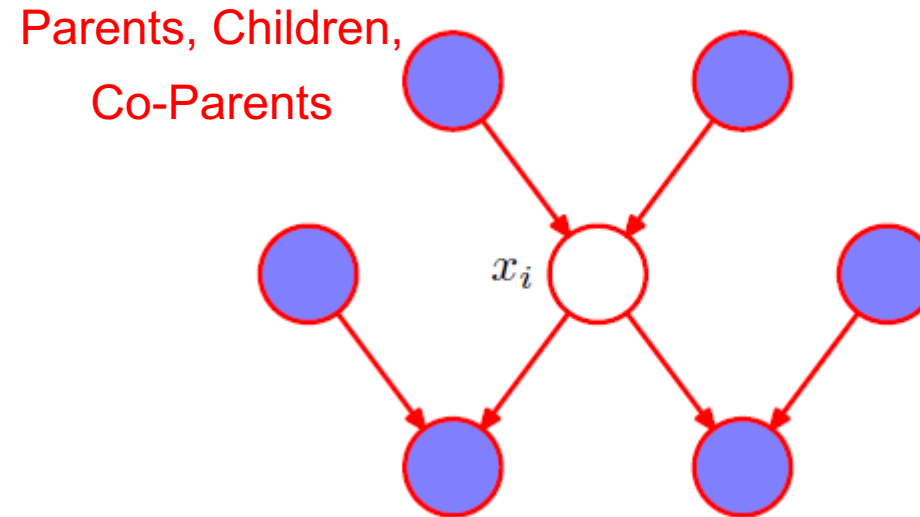
# Gibbs Sampling

Recall that an RV is conditionally independent of all RVs given its *Markov Blanket*

**MRF**



**Bayes Net**



So complete conditionals only depend on Markov Blanket,

$$p(x_i | x_{\neg i}) = p(x_i | x_{\text{Mb}(i)})$$

Consider a set of  $N$  variables,  $z_1, z_1, \dots, z_N$ . Then Gibbs says

Initialize  $\{z_i^{(0)} : i = 1, \dots, N\}$

While not\_bored

Can choose any order (or randomize)

{

For  $i=1$  to  $N$

Condition on most recent samples

{

Sample  $z_i^{(\tau+1)} \sim p\left(z_i \mid z_1^{(\tau+1)}, \dots, z_{i-1}^{(\tau+1)}, z_{i+1}^{(\tau)}, \dots, z_M^{(\tau)}\right)$

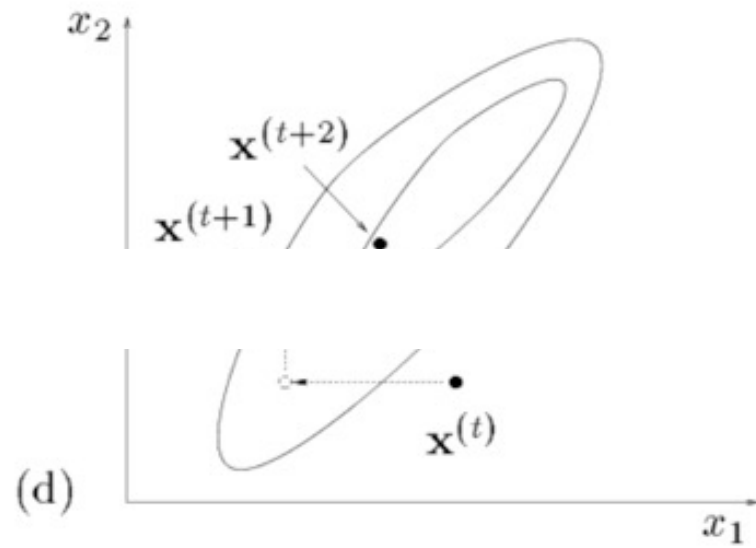
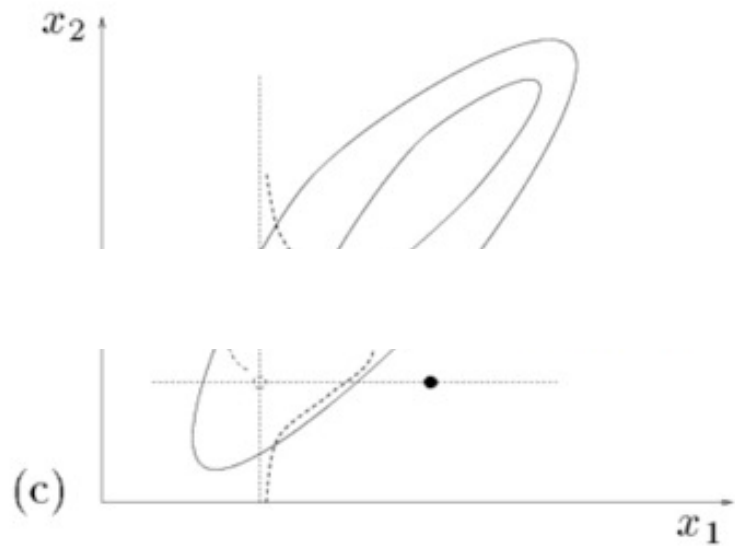
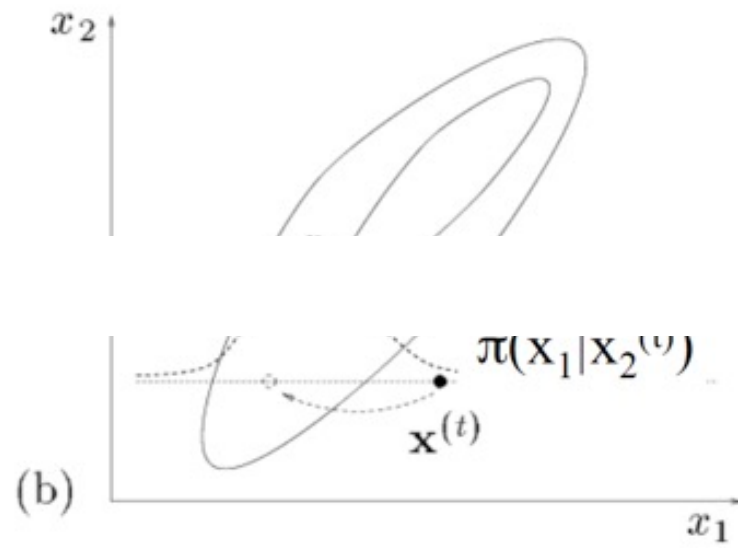
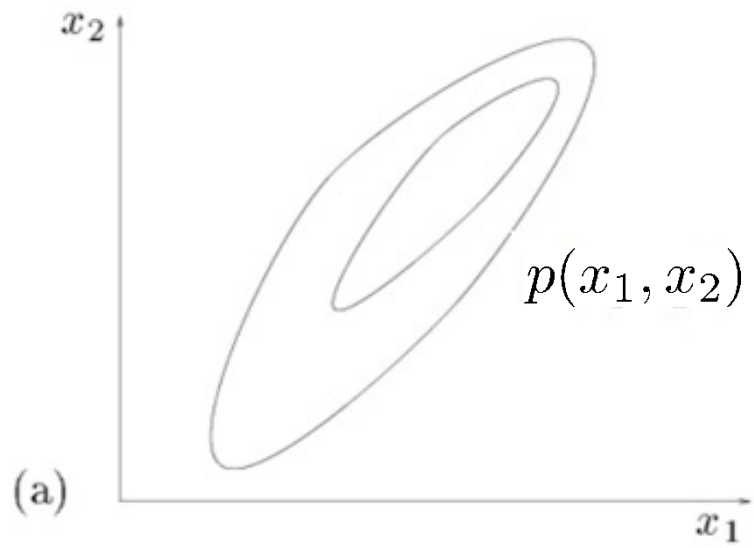
Always accept (i.e., emit  $z = z_1^{(\tau+1)}, \dots, z_{i-1}^{(\tau+1)}, z_i^{(\tau+1)}, z_{i+1}^{(\tau)}, \dots, z_M^{(\tau)}$ )

}

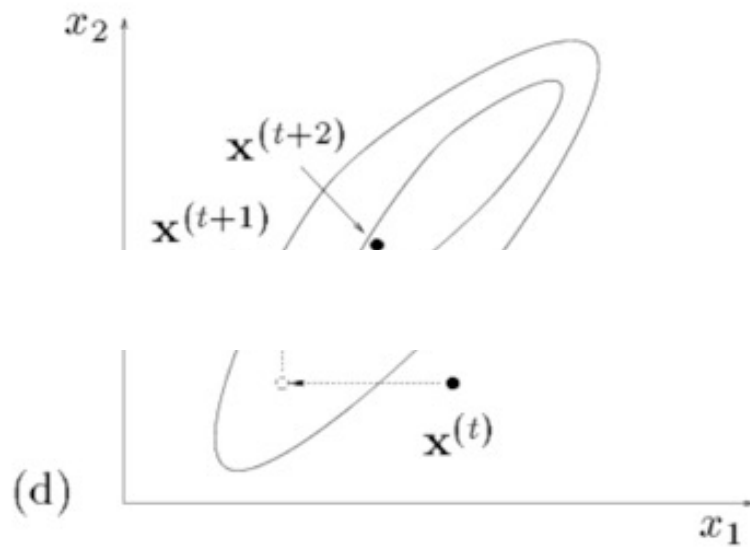
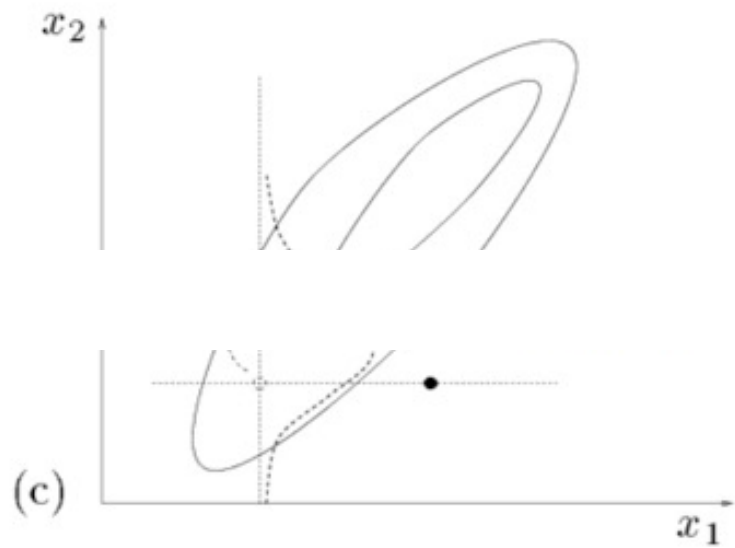
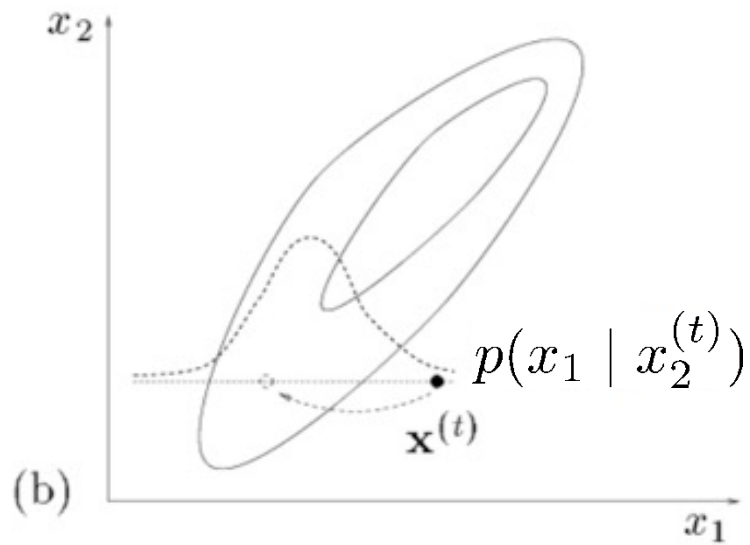
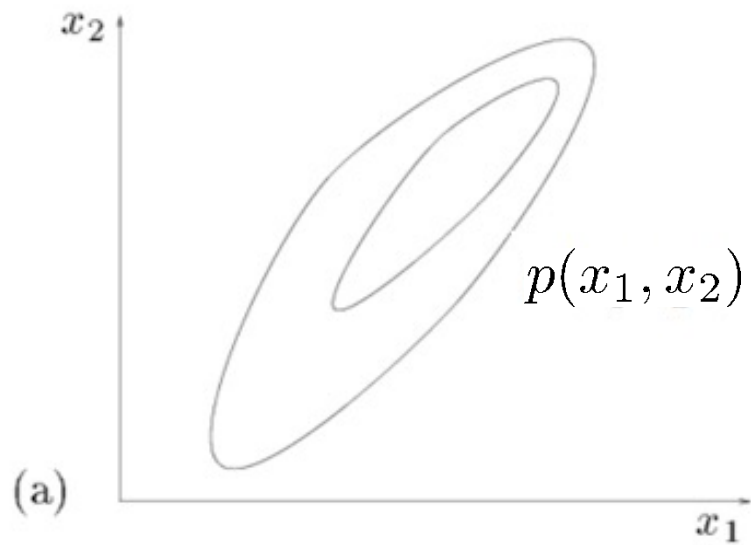
}

# Gibbs sampling

- Gibbs sampling is special case of M-H (but we always accept)
- Unlike M-H we *do not have to choose proposal*
- The proposal distribution will be cycle over  $p(z_i | z_{-i})$
- Transition function  $T()$  varies (cycles) over time
  - Relaxation of our assumption used to provide intuition about convergence
  - It still OK because the concatenation of the  $T()$  for a cycle converge
- We must be able to compute and sample from  $p(z_i | z_{-i})$ 
  - This is not always possible in general!
- This is **not** the same as sampling from the generative model, e.g. Ancestral Sampling in a Bayes Net samples from  $p(z_i | z_{\text{Pa}(i)})$

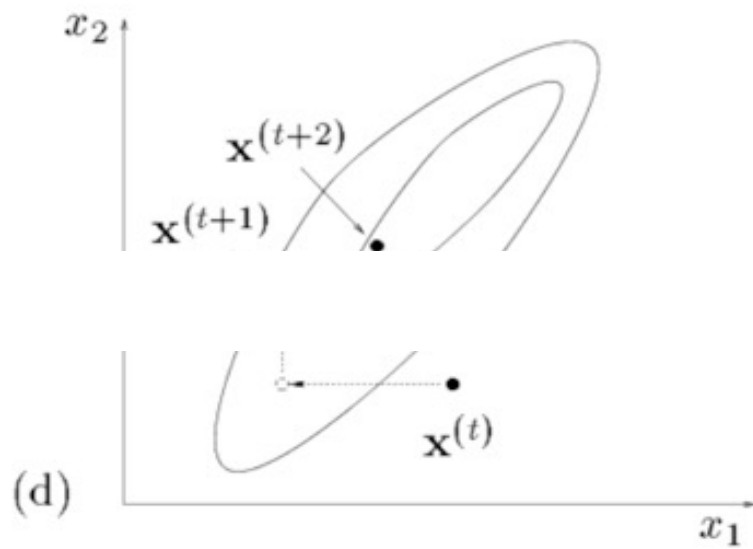
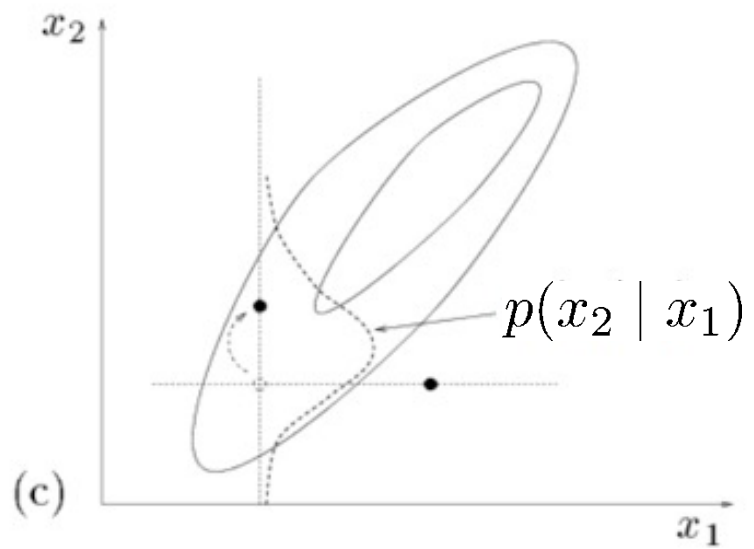
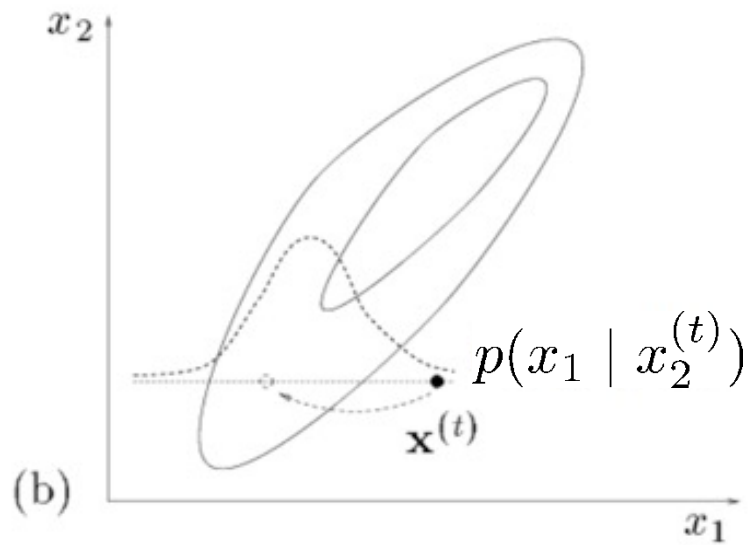
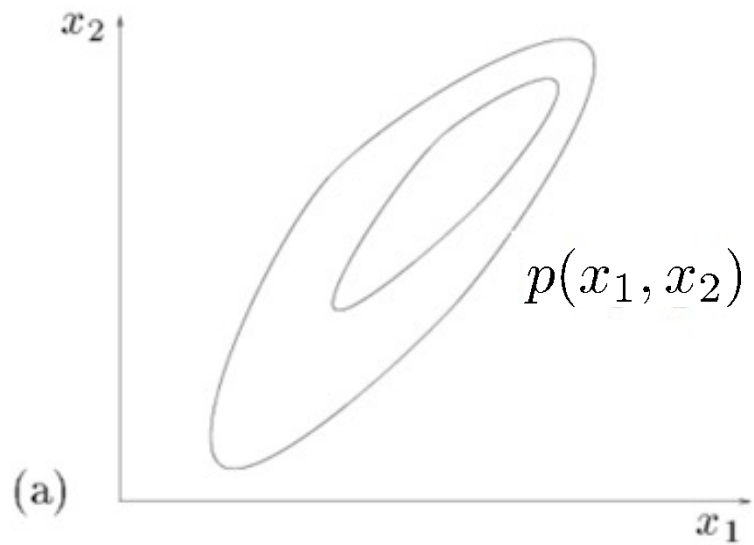


(Source: D. MacKay)

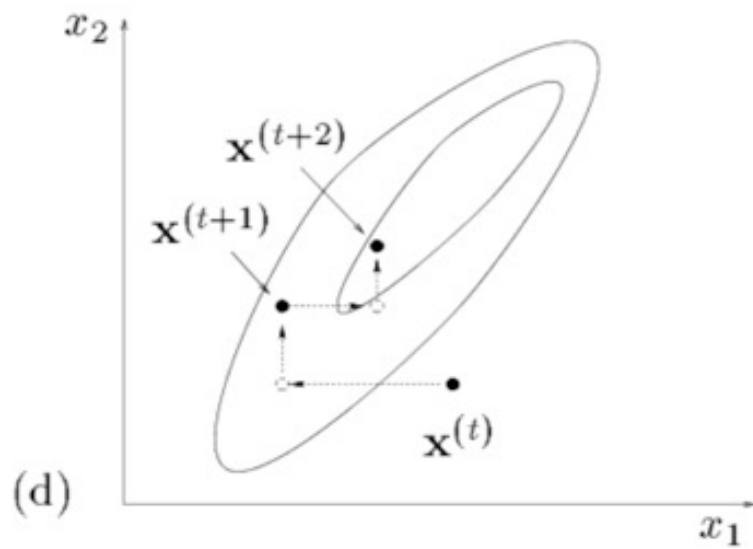
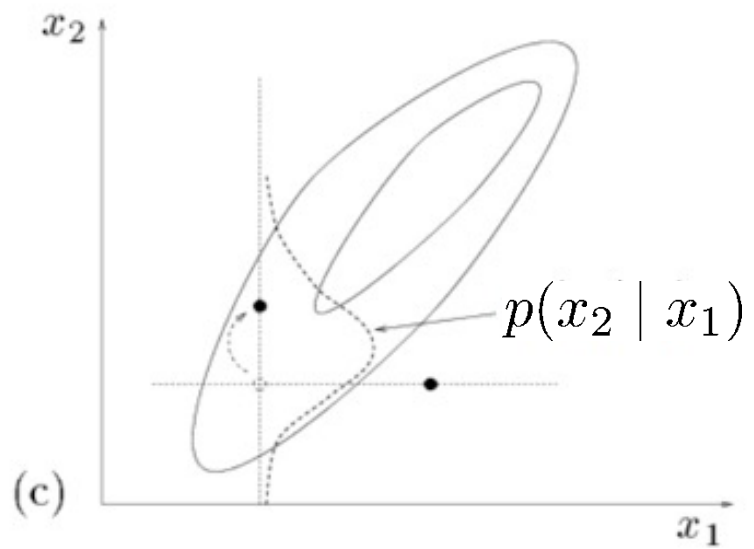
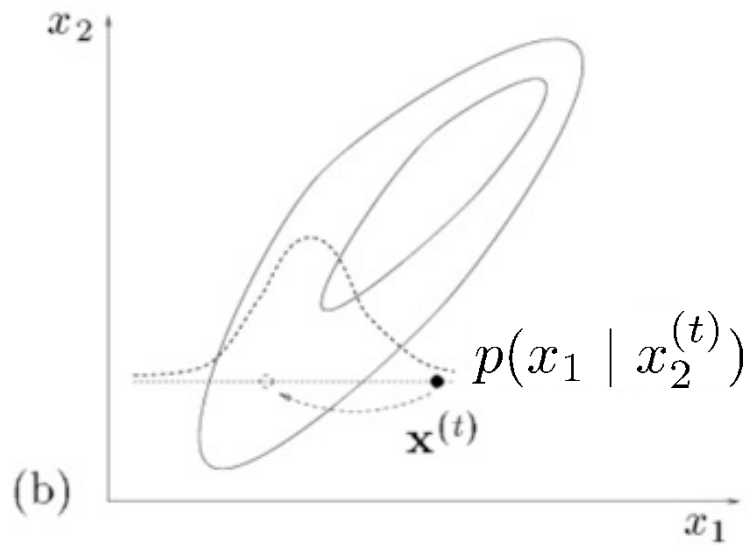
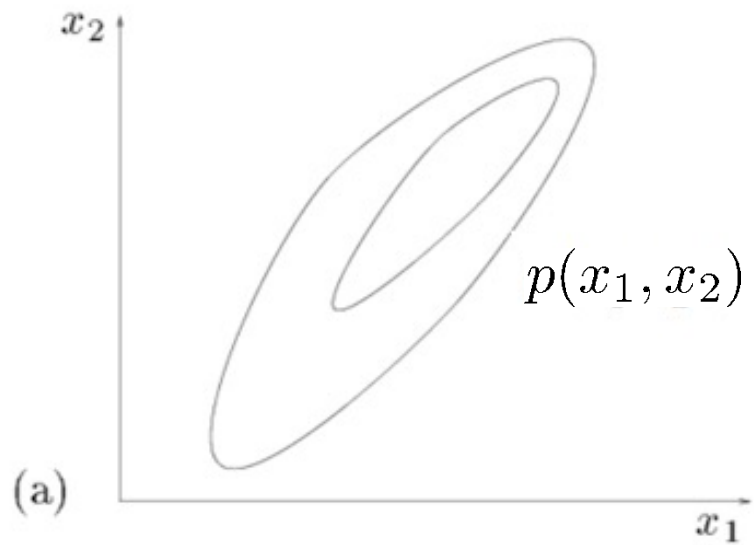


(Source: D. MacKay)





(Source: D. MacKay)



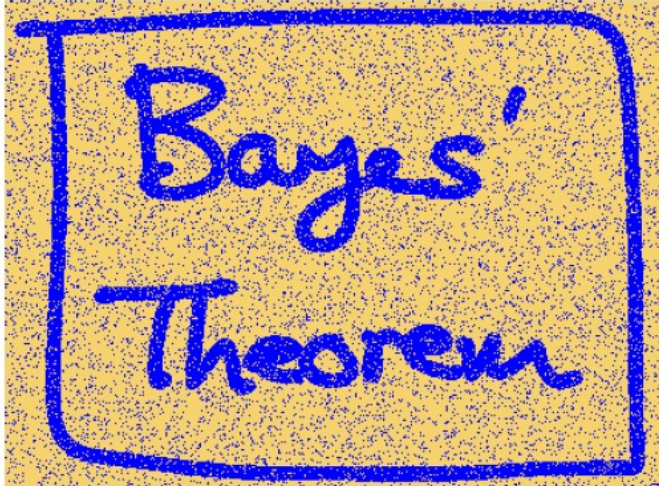
(Source: D. MacKay)

# Examples of Gibbs

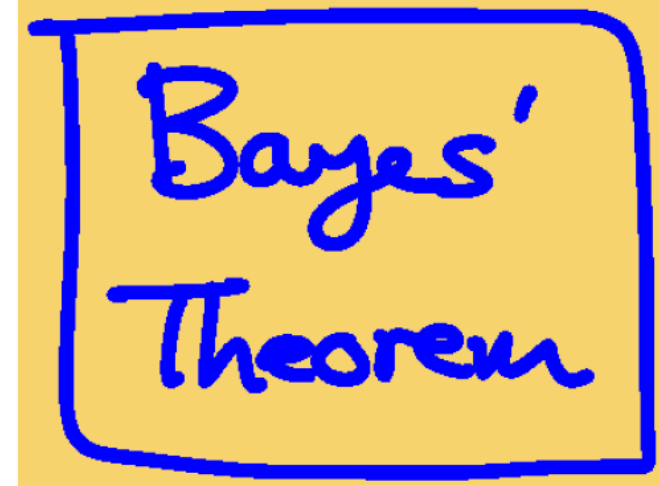
- Gibbs can be very good if one can compute and sample from the complete conditional distributions
- This is often feasible for MRFs of discrete RVs
  - Typical examples include symmetric systems like the Markov random field grids we had for images
  - Complete conditionals only depend on immediate neighboring pixels
- Continuous models are more complicated, and typically restricted to *exponential family distributions* (we will discuss in the next lecture)

# Example: Image Denoising

Noisy Image



Latent Image

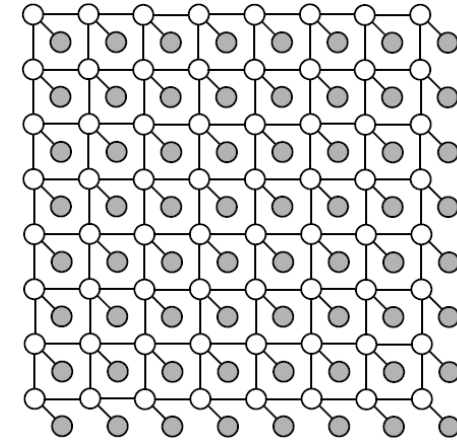


**Problem** Given observed image corrupted by i.i.d. noise, infer “clean” denoised image.

# Example: Image Denoising

Use a “grid graph” where each pixel is connected to its up/down/left/right neighbors,

$$p(x | y) \propto \prod_i \phi_i(x_i, y_i) \prod_{j \in \Gamma(i)} \phi_{ij}(x_i, x_j)$$



Where  $x_i, y_i \in \{-1, +1\}$  for convenience

**Observation Likelihood:**  $\log \phi_i(x_i) = \eta x_i y_i$

**Pairwise Similarity:**  $\log \phi_{ij}(x_i, x_j) = \beta x_i x_j$

Observation noise

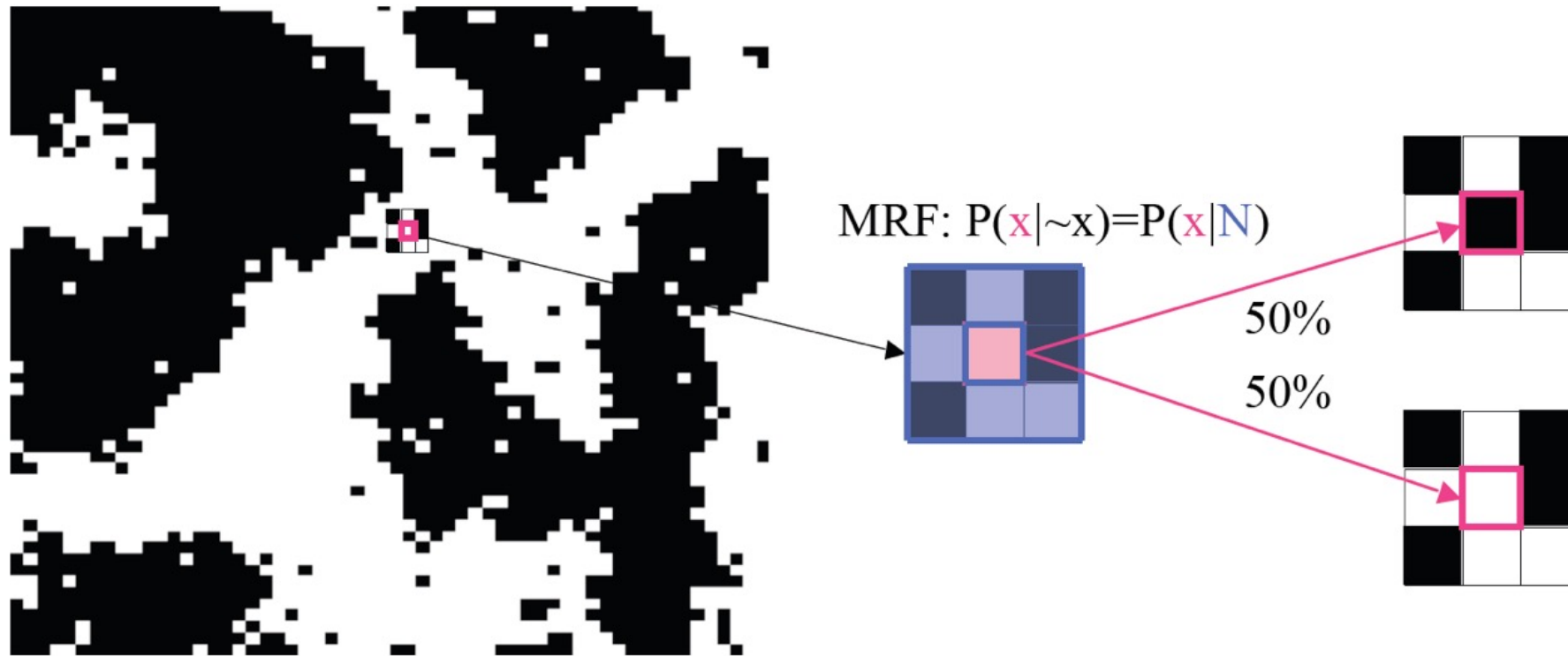
Smoothness prior

Complete conditional only depends on immediate neighbors,

$$p(x_i | x_{-i}) \propto \phi_i(x_i, y_i) \prod_{j \in \Gamma(i)} \phi_{ij}(x_i, x_j)$$

Normalizer only requires summing over 4 neighbors  $\mathcal{O}(2^4)$

# Examples of Gibbs



(From Dellaert and Zhu tutorial)

# Examples of Gibbs



Weak Affinity to Neighbors



Strong Affinity to Neighbors

(From Dellaert and Zhu tutorial)

# Gibbs as Metropolis Hastings (M-H)

To see Gibbs as MH, and to understand why we always accept, consider that if it were MH, then our proposal distribution,  $q_i()$ , for a given variable,  $i$ , would be

$$q_i(\mathbf{z}^* | \mathbf{z}) = p(z_i^* | \mathbf{z}_{\setminus i}) \quad \text{and} \quad q_i(\mathbf{z} | \mathbf{z}^*) = p(z_i | \mathbf{z}_{\setminus i}^*)$$

And we have  $\mathbf{z}_{\setminus i}^* = \mathbf{z}_{\setminus i}$  because only  $i$  changes.

The “\*” here means next state, NOT stationary state.



# Gibbs as M-H

$$A(\mathbf{z}^*, \mathbf{z}) = \min \left( 1, \frac{p(\mathbf{z}^*) q_i(\mathbf{z} | \mathbf{z}^*)}{p(\mathbf{z}) q_i(\mathbf{z}^* | \mathbf{z})} \right) \quad (\text{def'n of } A())$$

# Gibbs as M-H

$$A(\mathbf{z}^*, \mathbf{z}) = \min \left( 1, \frac{p(\mathbf{z}^*) q_i(\mathbf{z} | \mathbf{z}^*)}{p(\mathbf{z}) q_i(\mathbf{z}^* | \mathbf{z})} \right) \quad (\text{def'n of } A())$$
$$= \min \left( 1, \frac{p(\mathbf{z}_{\setminus i}^*) p(z_i^* | \mathbf{z}_{\setminus i}^*) q_i(\mathbf{z} | \mathbf{z}^*)}{p(\mathbf{z}_{\setminus i}) p(z_i | \mathbf{z}_{\setminus i}) q_i(\mathbf{z}^* | \mathbf{z})} \right) \quad (\text{because?})$$

# Gibbs as M-H

$$A(\mathbf{z}^*, \mathbf{z}) = \min \left( 1, \frac{p(\mathbf{z}^*) q_i(\mathbf{z} | \mathbf{z}^*)}{p(\mathbf{z}) q_i(\mathbf{z}^* | \mathbf{z})} \right)$$

(def'n of A())

$$= \min \left( 1, \frac{p(\mathbf{z}_{\setminus i}^*) p(z_i^* | \mathbf{z}_{\setminus i}^*) q_i(\mathbf{z} | \mathbf{z}^*)}{p(\mathbf{z}_{\setminus i}) p(z_i | \mathbf{z}_{\setminus i}) q_i(\mathbf{z}^* | \mathbf{z})} \right)$$

(def'n of "bar")

$$= \min \left( 1, \frac{p(\mathbf{z}_{\setminus i}^*) p(z_i^* | \mathbf{z}_{\setminus i}^*) p(z_i | \mathbf{z}_{\setminus i}^*)}{p(\mathbf{z}_{\setminus i}) p(z_i | \mathbf{z}_{\setminus i}) p(z_i^* | \mathbf{z}_{\setminus i})} \right)$$

**(because?)**

# Gibbs as M-H

$$\begin{aligned} A(\mathbf{z}^*, \mathbf{z}) &= \min \left( 1, \frac{p(\mathbf{z}^*) q_i(\mathbf{z} | \mathbf{z}^*)}{p(\mathbf{z}) q_i(\mathbf{z}^* | \mathbf{z})} \right) && \text{(def'n of } A()) \\ &= \min \left( 1, \frac{p(\mathbf{z}_{\setminus i}^*) p(z_i^* | \mathbf{z}_{\setminus i}^*) q_i(\mathbf{z} | \mathbf{z}^*)}{p(\mathbf{z}_{\setminus i}) p(z_i | \mathbf{z}_{\setminus i}) q_i(\mathbf{z}^* | \mathbf{z})} \right) && \text{(def'n of "bar")} \\ &= \min \left( 1, \frac{p(\mathbf{z}_{\setminus i}^*) p(z_i^* | \mathbf{z}_{\setminus i}^*) p(z_i | \mathbf{z}_{\setminus i}^*)}{p(\mathbf{z}_{\setminus i}) p(z_i | \mathbf{z}_{\setminus i}) p(z_i^* | \mathbf{z}_{\setminus i})} \right) && \text{(Gibbs, coloring)} \\ &= \min(1, 1) && \text{(cancel colors using } \mathbf{z}_{\setminus i}^* = \mathbf{z}_{\setminus i}, \text{ as only } z_i \text{ changes)} \\ &= 1 \end{aligned}$$

# Gibbs Sampling Extensions

Standard Gibbs suffers same random walk behavior as M-H  
(but no adjustable parameters, so that's a plus...)

**Block Gibbs** Jointly sample subset  $S \subset \mathcal{V}$  from  $p(x_S | x_{\neg S})$

- Reduces random walk caused by highly correlated variables
- Requires that conditional  $p(x_S | x_{\neg S})$  can be sampled efficiently

**Collapsed Gibbs** Marginalize some variables out of joint:

$$p(x_{\mathcal{V} \setminus S}) = \int p(x) dx_S$$

- Reduces dimensionality of space to be sampled
- Requires that marginals are computable in closed-form

# Combined samplers

Different samplers fail in different ways, so combine them...


1. Initialise  $x^{(0)}$ .
2. For  $i = 0$  to  $N - 1$ 
  - Sample  $u \sim \mathcal{U}_{[0,1]}$ .
  - If  $u < \nu$ 
    - Apply the MH algorithm with a global proposal.
  - else
    - Apply the MH algorithm with a random walk proposal.

...can also combine with Gibbs proposals

# Mixing MCMC Kernels

*Can do this more generally....*

Consider a set of MCMC kernels  $T_1, T_2, \dots, T_K$  all having target distribution  $p(x)$  then the mixture:

$$T = \sum_{k=1}^K \pi_k T_k$$


Mixing weights

Is a valid MCMC kernel with target distribution  $p(x)$

**Mixture MCMC** Transition kernel given by:

1. Sample  $k \sim \pi$
2. Sample  $x^{(t+1)} \sim T_k(x | x^{(t)})$

# Inference (and related) Tasks

- Simulation:  $x \sim p(x) = \frac{1}{Z} f(x)$
- Compute expectations:  $\mathbb{E}[\phi(x)] = \int p(x) \phi(x) dx$
- Optimization:  $x^* = \arg \max_x f(x)$
- Compute normalizer:  $Z = \int f(x) dx$



# Inference (and related) Tasks

- Simulation:  $x \sim p(x) = \frac{1}{Z} f(x)$
- Compute expectations:  $\mathbb{E}[\phi(x)] = \int p(x) \phi(x) dx$
- Optimization:  $x^* = \arg \max_x f(x)$
- Compute normalizer:  $Z = \int f(x) dx$

# Simulated Annealing

- Analogy with physical systems
- Relevant for optimization (not integration)
- Powers of probability distributions emphasize the peaks
- If we are looking for a maximum within a lot of distracting peaks, this can help.

# Simulated Annealing

- Define a temperature  $T$ , and a cooling schedule (black magic part)
- Lower temperatures correspond to emphasized maximal peaks.
  - Hence we exponentiate by  $(1/T)$ .
- The terminology makes sense because the number of states accessible to a physical system decreases with temperature.

# Simulated Annealing

1. Initialise  $x^{(0)}$  and set  $T_0 = 1$ .

2. For  $i = 0$  to  $N - 1$

– Sample  $u \sim \mathcal{U}_{[0,1]}$ .

– Sample  $x^* \sim q(x^* | x^{(i)})$ .

– If  $u < \mathcal{A}(x^{(i)}, x^*) = \min \left\{ 1, \frac{p^{\frac{1}{T_i}}(x^*) q(x^{(i)} | x^*)}{p^{\frac{1}{T_i}}(x^{(i)}) q(x^* | x^{(i)})} \right\}$

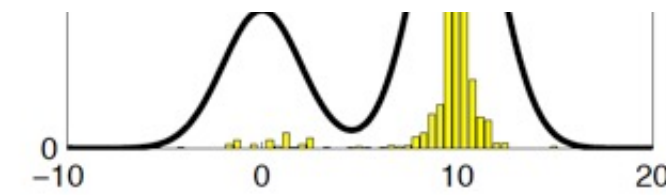
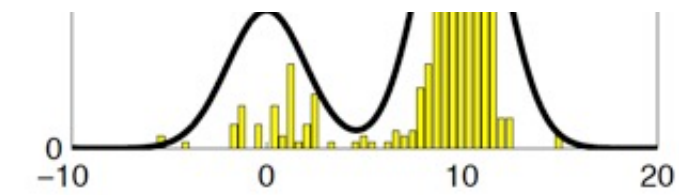
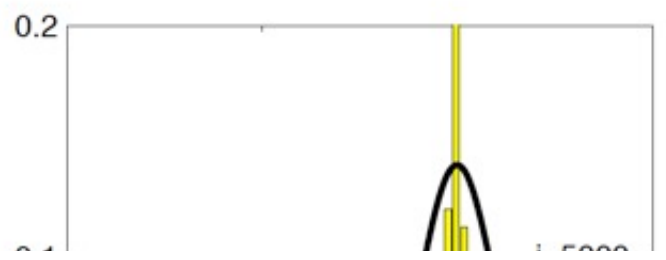
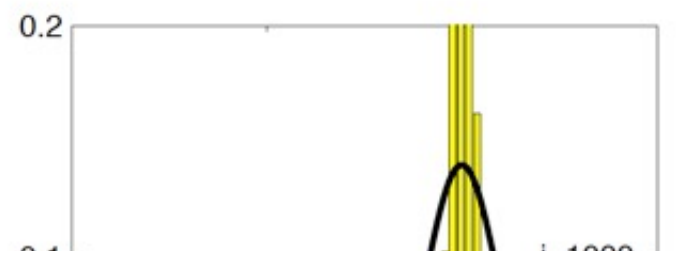
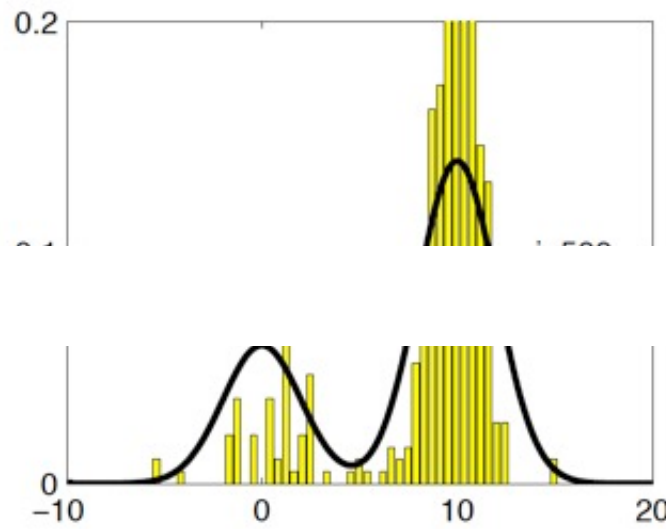
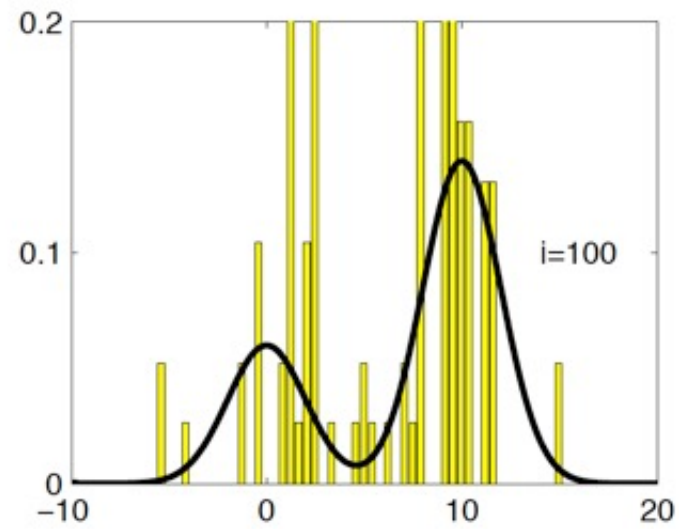
$$x^{(i+1)} = x^*$$

else

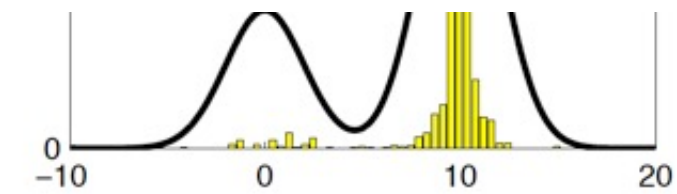
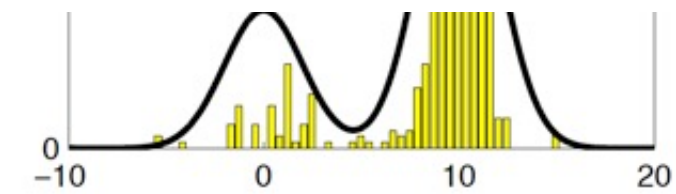
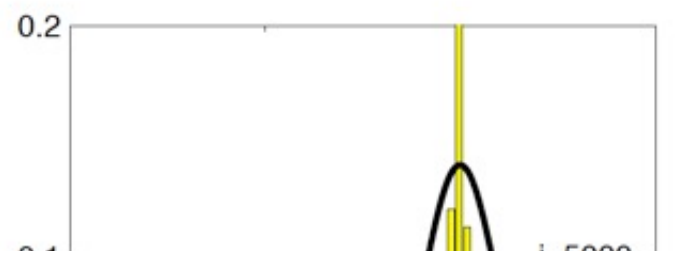
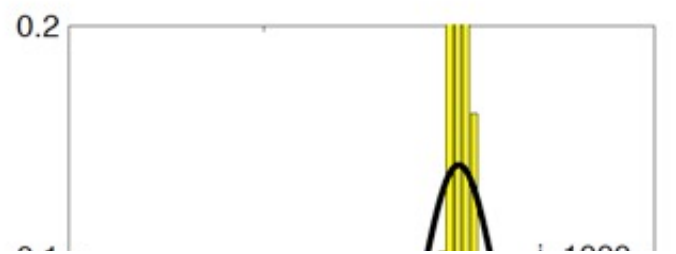
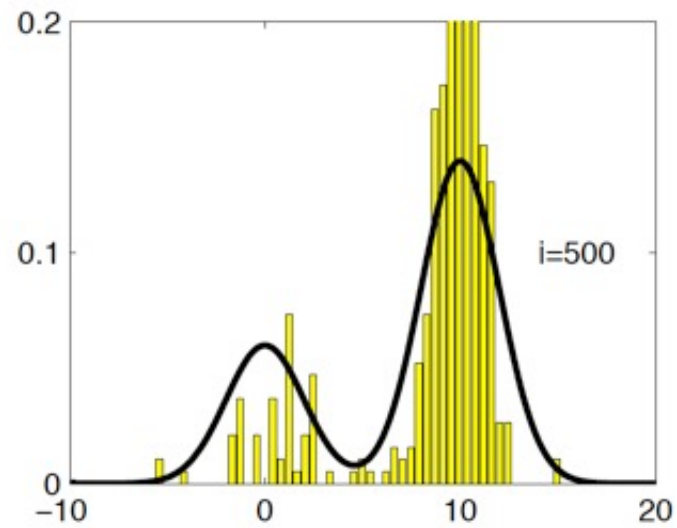
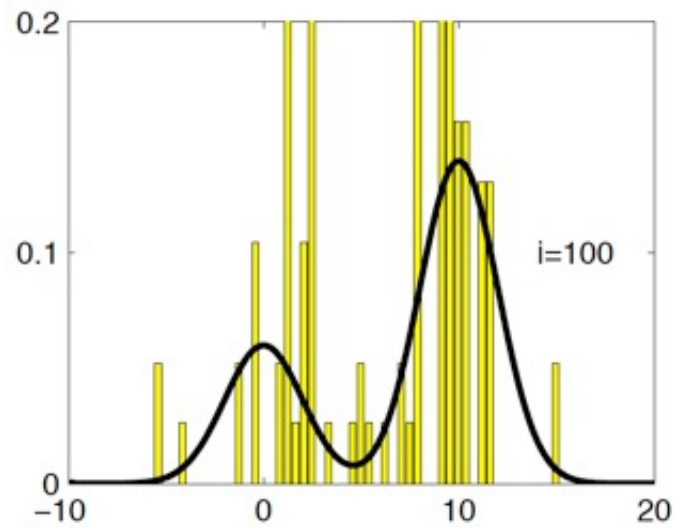
$$x^{(i+1)} = x^{(i)}$$

– Set  $T_{i+1}$  according to a chosen cooling schedule.

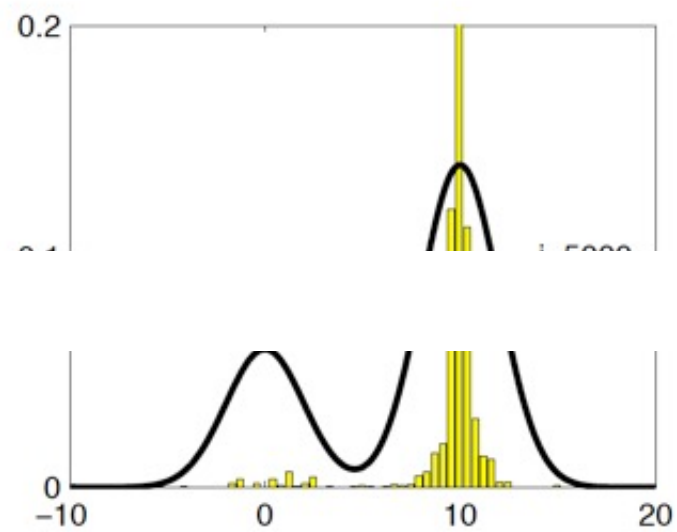
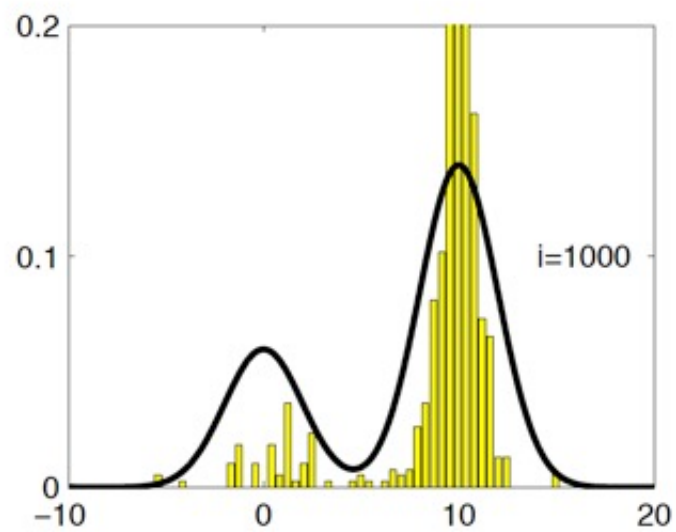
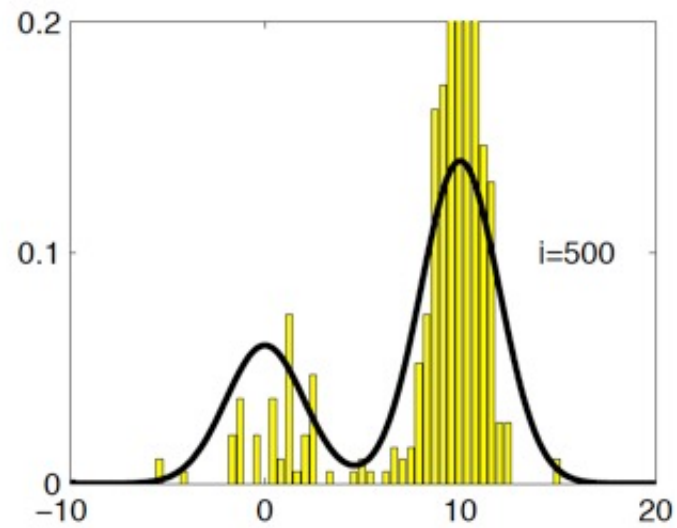
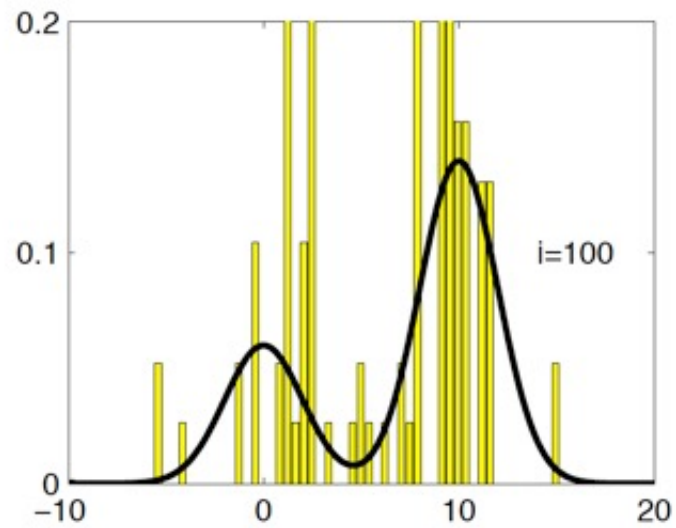
Basically M-H but we are *annealing*  
target distribution with temperature T



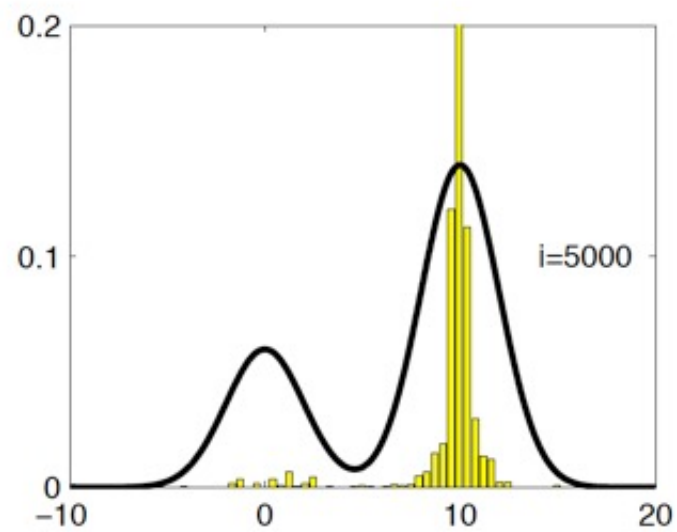
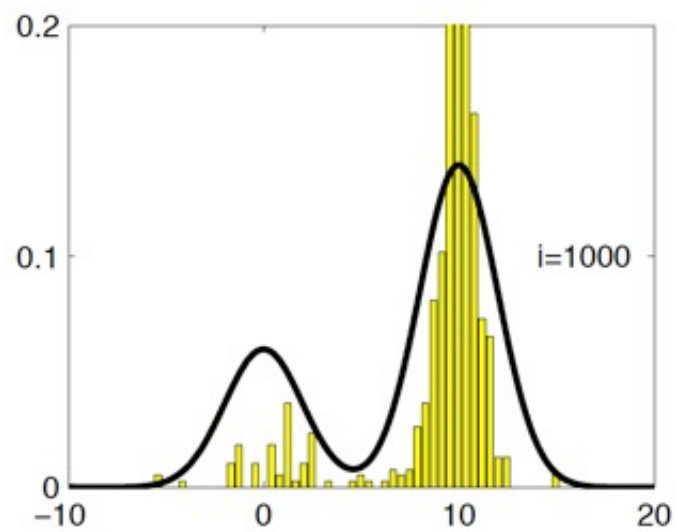
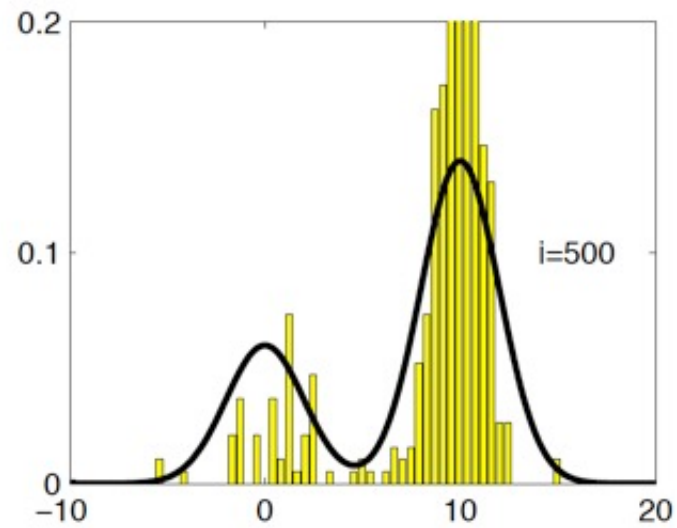
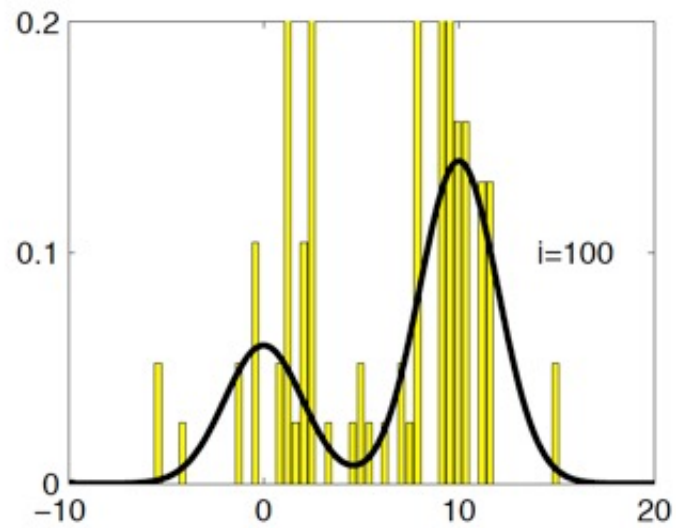
(From Andrieu et al)



(From Andrieu et al)



(From Andrieu et al)



(From Andrieu et al)



# Simulated Annealing

Let *annealing distribution* at temp  $\tau$  be given by:

$$p_\tau(x) \propto (f(x))^{1/\tau}$$

As  $\tau \rightarrow 0$  we have:

$$\lim_{\tau \rightarrow 0} p_\tau(x) = \delta(x^*) \quad \text{where} \quad x^* = \arg \max_x f(x)$$

## **Simulated Annealing (SA) for Global Optimization:**

Annealing schedule  $\tau_0 \geq \dots \geq \tau_t \geq \dots \geq 0$

1. Sample  $x^{(t)}$  from MCMC kernel  $T_t$  with target  $p_{\tau_t}(x)$
2. Set  $\tau_{t+1}$  according to annealing schedule

**SA for Convergence:**  $\tau_0 \geq \dots \geq 1$  Final temperature = 1

# MCMC Summary

- Markov chain induced by MCMC transition kernel  $T(z, z')$
- Converges to stationary distribution iff chain is **ergodic**
  - Chain is ergodic if it is **irreducible** (can get from any  $z$  to any  $z'$ ) and **aperiodic** (doesn't get trapped in cycles)
- Easier to prove **detailed balance**, which implies ergodicity

$$p(z)T(z, z') = p(z')T(z', z)$$

- Metropolis algorithm samples from symmetric proposal  $q(z'|z)$  and accepts sample  $z'$  with probability,

$$A = \min \left( 1, \frac{\tilde{p}(z')}{\tilde{p}(z)} \right)$$

# MCMC Summary

- Metropolis-Hastings allows non-symmetric proposal  $q(z'|z)$  and accepts sample  $z'$  with probability,

$$A = \min \left( 1, \frac{\tilde{p}(z')}{\tilde{p}(z)} \frac{q(z|z')}{q(z'|z)} \right)$$

- Gibbs sampler on random vector  $z = (z_1, \dots, z_d)^T$  successively samples from *complete conditionals*,

$$z_1^{\text{new}} \sim p(z_1 | z_2^{\text{old}}, \dots, z_d^{\text{old}})$$

$$z_2^{\text{new}} \sim p(z_2 | z_1^{\text{new}}, z_3^{\text{old}}, \dots, z_d^{\text{old}})$$

...

$$z_d^{\text{new}} \sim p(z_d | z_1^{\text{new}}, \dots, z_{d-1}^{\text{new}})$$

- Gibbs is instance of M-H which *always accepts*

# MCMC Summary

- Simulated annealing adjusts target distribution at each stage with temperature  $T$

$$(p(z))^{\frac{1}{T_i}}$$

- For decreasing temperatures  $\lim T_i \rightarrow 0$  support of target approaches set of global maximizers
  - Convenient to use for global maximization
  - Can prove that this will find the global maximum in the limit (need to wait for the heat death of the universe, however...)
- For increasing temp ending at  $\lim T_i \rightarrow 1$  approaches  $p(x)$ 
  - Helps avoid getting stuck in local optima

# Monte Carlo Methods Summary

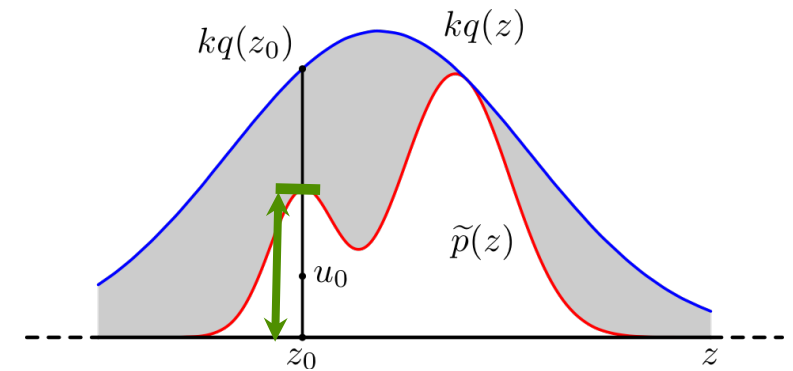
- Simulation:  $x \sim p(x) = \frac{1}{Z} f(x)$  **Rejection sampling, MCMC**
- Compute expectations:  $\mathbb{E}[\phi(x)] = \int p(x) \phi(x) dx$  **Importance sampling or any simulation method**
- Optimization:  $x^* = \arg \max_x f(x)$  **Simulated annealing**
- Compute normalizer / marginal likelihood:  $Z = \int f(x) dx$  **Reverse importance sampling (Did not cover)**

# Monte Carlo Methods Summary

- In complex models we often have no other choice than to simulate realizations
- Rejection sampler choose proposal/constant s.t.  $\tilde{p}(z) \leq kq(z)$

1) Sample  $q(z)$

2) Keep samples in proportion to  $\frac{\tilde{p}(z)}{k \cdot q(z)}$  and reject the rest.



- Monte carlo estimate via independent samples  $\{z^{(i)}\}_{i=1}^L \sim p$ ,

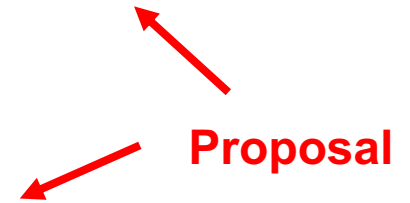
$$\mathbf{E}_p[f] \approx \frac{1}{L} \sum_{i=1}^L f(z^{(i)})$$

- Unbiased
- Consistent
- Law of large numbers
- Central limit theorem (if  $f$  is finite variance)

# Monte Carlo Methods Summary

- Importance sampling estimate over samples  $\{z^{(i)}\}_{i=1}^L \sim q$ ,

$$\mathbf{E}_p[f] \approx \sum_{i=1}^L w^{(i)} f(z^{(i)})$$

$$w^{(i)} \propto \frac{\tilde{p}(z^{(i)})}{q(z^{(i)})}$$


**Importance Weights**

- Avoids simulation of  $p(z)$  but variance scales exponentially with dim.
- Sequential importance sampling extends IS for sequence models, with proposal given by dynamics,

$$q(z) = q(z_0) \prod_{t=1}^T p(z_t | z_{t-1})$$

**“Bootstrap” Particle Filter**

$$w_t(z^{(i)}) \propto w_{t-1}(z^{(i-1)}) p(y_t | z_t^{(i)})$$

**Recursively update weights**

- **Resampling** step necessary to avoid weight degeneracy

# Monte Carlo Methods Summary

- Lots of other methods to explore...
  - Hamiltonian Monte Carlo
  - Slice Sampling
  - Reversible Jump MCMC (and other *transdimensional samplers*)
  - Parallel Tempering
- Some good resources if you are interested...
  - Neal, R. “Probabilistic Inference Using Markov Chain Monte Carlo Methods”, U. Toronto, 1993
  - MacKay, D. J. “Introduction to Monte Carlo Methods”, Cambridge U., 1998
  - Andrieu, C., et al., “Introduction to MCMC for Machine Learning”, 2001