Computer Science

CSC535: Probabilistic Graphical Models

Monte Carlo Methods

Prof. Jason Pacheco

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

Administrivia

- HW3 has been graded / returned
- Midterm being graded
- HW4 out (Due: Monday, 11/16 ~ 1.5wks)

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

- 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,

$$p(z \mid \mathcal{D}) = \underbrace{\frac{p(z)p(\mathcal{D} \mid z)}{p(\mathcal{D})}}_{} \propto \widetilde{p}(z)$$

Unnormalized posterior

Don't know marginal likelihood / normalizer

- Typically, $\widetilde{p}(z)$ is easy to evaluate

Motivation for Monte Carlo Methods

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

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 independant samples $z^{(l)}$ from $\tilde{p}(z)$

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

Challenges for Monte Carlo Methods

- In real problems sampling $p(z)$ is very difficult
- Typically don't know normalization, so need to use $\widetilde{p}(z)$ instead
- Even if we **can** sample $p(z)$, it can be hard to know if/when they are "good" and if we have enough (e.g. to approximate E[f] well)
- Sometimes evaluating $\widetilde{p}(z)$ can also be hard

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: $Z = \int f(x)\,dx$

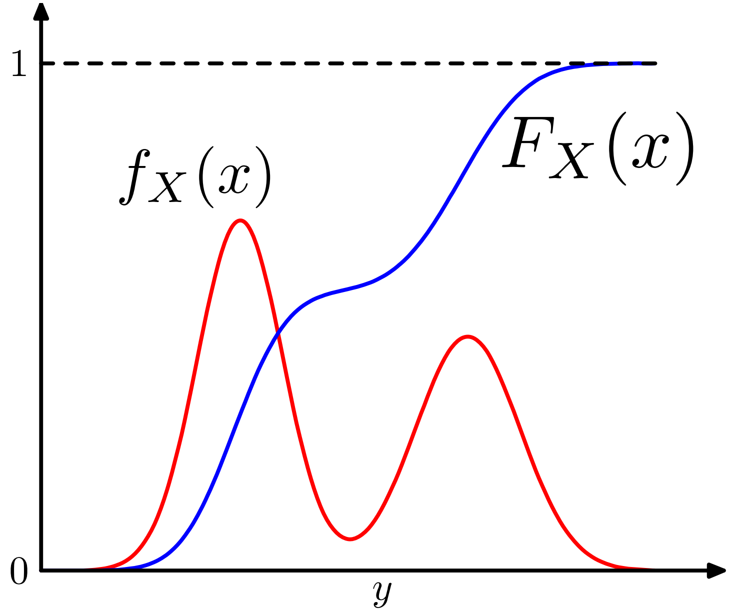
Basic Sampling (so far…)

- Uniform sampling (everything builds on this)
- Sampling from simple discrete distributions
  - Multinomial / categorical
  - Binomial / Bernoulli
  - Etc.
- Sampling for selected other distributions (e.g., Gaussian)
  - At least, Matlab knows how to do it.
- Ancestral sampling

Continuous Random Number Generation

- Input: Independent standard uniform variables $U_1, U_2, U_3, \ldots$
- 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!

Rejection Sampling

Assume

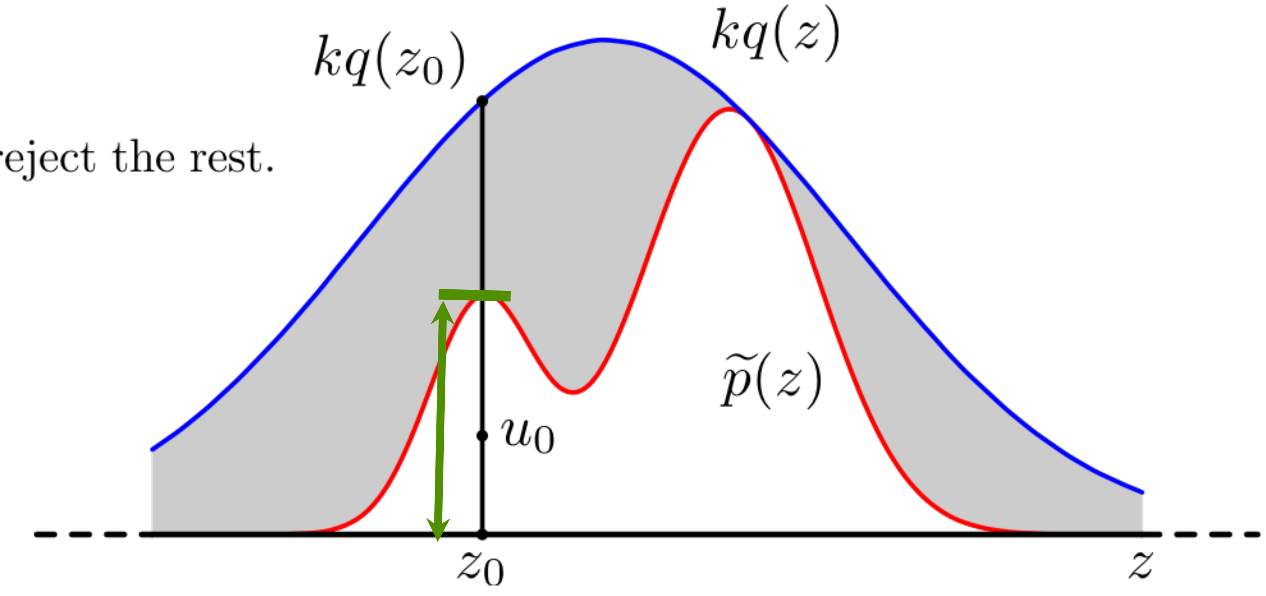
- Access to easy-to-sample distribution $q(z)$
- Constant *k* such that $\widetilde{p}(z) \leq k \cdot q(z)$

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

Algorithm

1) Sample $q(z)$

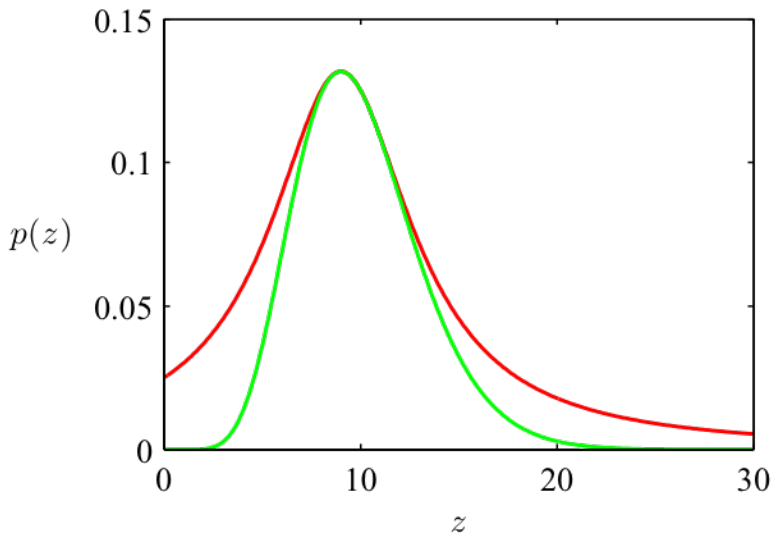
2) Keep samples in proportion to $\dfrac{\tilde{p}(z)}{k \bullet q(z)}$ and reject the rest.



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

$$\mu \triangleq \mathbb{E}[f] = \int f(x)p(x)\; dx \approx \frac{1}{L}\sum_{\ell=1}^{L} f(x^{(\ell)}) \triangleq \hat{f}_L$$

- Expectation estimated from *empirical distribution* of L samples:

$$\hat{p}_L(x) = \frac{1}{L}\sum_{\ell=1}^{L} \delta_{x^{(\ell)}}(x) \qquad\qquad x^{(\ell)} \sim p(x)$$

- The *Dirac delta* function is only well-defined within integrals:

$$\int_{\mathcal{X}} \delta_{\bar{x}}(x) f(x)\; dx = f(\bar{x}) \qquad\qquad \int_{A} \delta_{\bar{x}}(x)\; dx = \mathbb{I}(\bar{x} \in A)$$

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

$$\mathbb{E}[\hat{f}_L] = \frac{1}{L}\sum_{\ell=1}^{L} \mathbb{E}[f(x^{(\ell)})] = \mathbb{E}[f]$$

Monte Carlo Asymptotics

$$\mu \triangleq \mathbb{E}[f] = \int f(x)p(x)\,dx \approx \frac{1}{L}\sum_{\ell=1}^{L} f(x^{(\ell)}) \triangleq \hat{f}_L$$

- Estimator variance reduces at rate 1/L:

$$\mathrm{Var}[\hat{f}_L] = \frac{1}{L}\mathrm{Var}[f] = \frac{1}{L}\mathbb{E}[(f(x)-\mu)^2]$$

Independent of dimensionality of random variable X

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

$$\sqrt{L}\left(\hat{f}_L - \mu\right) \underset{L\to\infty}{\Longrightarrow} \mathcal{N}(0, \mathrm{Var}[f])$$

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

Weak Law

$$\lim_{L\to\infty} \Pr(|\hat{f}_L - \mu| < \epsilon) = 1, \text{ for any } \epsilon > 0$$

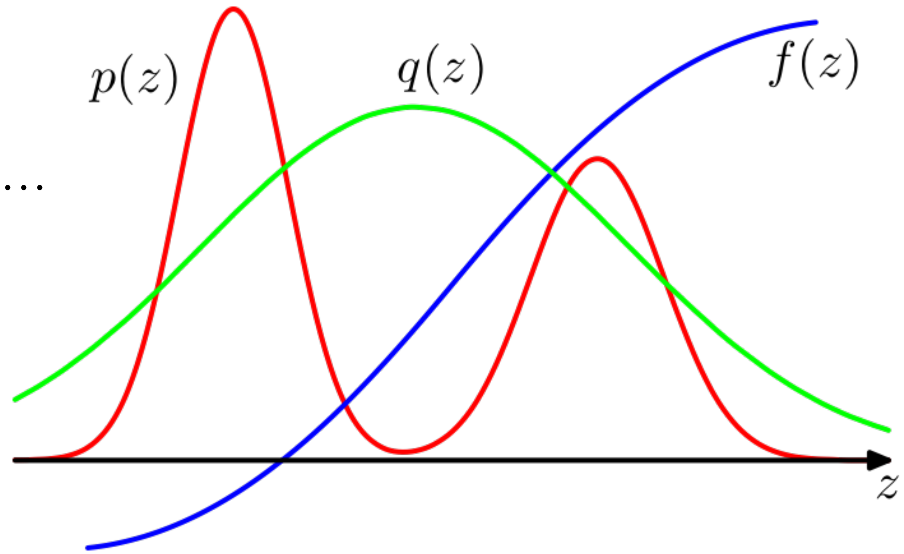
Strong Law

$$\Pr\left(\lim_{L\to\infty} \hat{f}_L = \mu\right) = 1$$

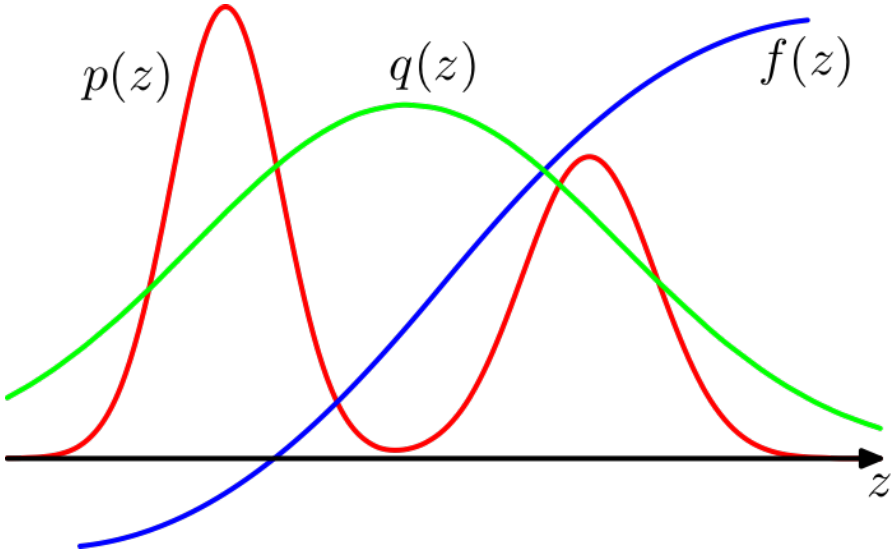
Importance Sampling

But we don't need to sample from *p(z)* to compute $\mathbf{E}_p[f]\ldots$

Assume we can sample some *q(z)*



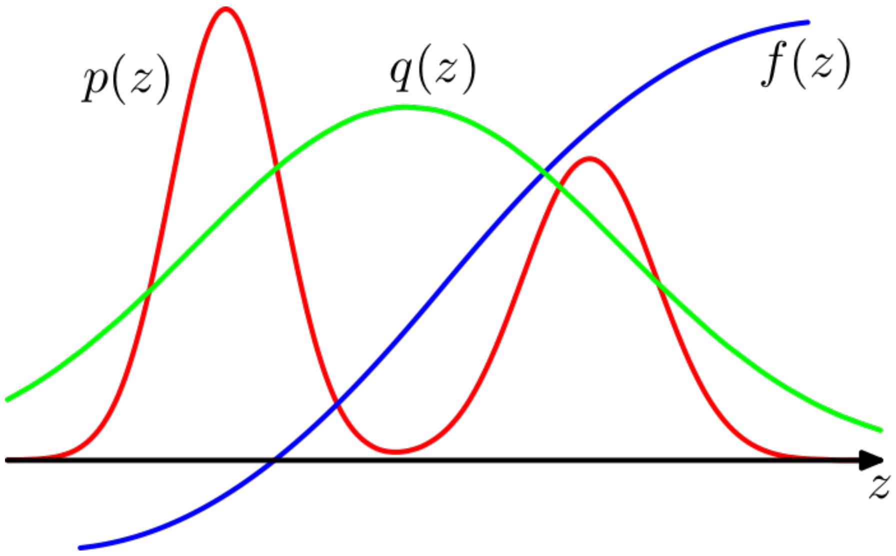
Importance Sampling



Rewrite $E_{p(z)}[f] = \int f(z) p(z) dz$

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

Importance Sampling



Rewrite $E_{p(z)}[f] = \int f(z)p(z)dz$

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

$$\cong \frac{1}{L}\sum_{l=1}^{L}\underbrace{\frac{p\left(z^{(l)}\right)}{q\left(z^{(l)}\right)}}_{\text{Requires that we can evaluate } p(z)} f\left(z^{(l)}\right)$$

where samples come from $q(z)$ (or $\tilde{q}(z)$)

Requires that we can evaluate *p(z)*

Importance Sampling (normalized)

$$p(z)=\frac{\tilde{p}(z)}{Z_p} \quad \text{and} \quad \mathrm{q}(z)=\frac{\tilde{q}(z)}{Z_q}$$

$$E_{p(z)}[f] \cong \frac{1}{L}\sum_{l=1}^{L}\frac{p\left(z^{(l)}\right)}{q\left(z^{(l)}\right)}f\left(z^{(l)}\right) \qquad \text{(samples from } q\left(z^{(l)}\right)\text{, equivalently, } \tilde{q}\left(z^{(l)}\right)\text{)}$$

Importance Sampling (normalized)

$$p(z) = \frac{\tilde{p}(z)}{Z_p} \quad \text{and} \quad \mathrm{q}(z) = \frac{\tilde{q}(z)}{Z_q}$$

$$E_{p(z)}[f] \cong \frac{1}{L}\sum_{l=1}^{L}\frac{p\left(z^{(l)}\right)}{q\left(z^{(l)}\right)} f\left(z^{(l)}\right) \qquad \text{(samples from } q\left(z^{(l)}\right)\text{, equivalently, } \tilde{q}\left(z^{(l)}\right)\text{)}$$

$$\cong \frac{Z_q}{Z_p}\frac{1}{L}\sum_{l=1}^{L}\frac{\tilde{p}\left(z^{(l)}\right)}{\tilde{q}\left(z^{(l)}\right)} f\left(z^{(l)}\right)$$

Importance Sampling (normalized)

$$p(z) = \frac{\tilde{p}(z)}{Z_p} \quad \text{and} \quad \mathrm{q}(z) = \frac{\tilde{q}(z)}{Z_q}$$

$$E_{p(z)}[f] \cong \frac{1}{L}\sum_{l=1}^{L}\frac{p(z^{(l)})}{q(z^{(l)})}f(z^{(l)}) \qquad \text{(samples from } q(z^{(l)}), \text{ equivalently}, \ \tilde{q}(z^{(l)}))$$

$$\cong \left(\frac{Z_q}{Z_p}\right)\frac{1}{L}\sum_{l=1}^{L}\frac{\tilde{p}(z^{(l)})}{\tilde{q}(z^{(l)})}f(z^{(l)})$$

$$= \left(\frac{Z_q}{Z_p}\right)\frac{1}{L}\sum_{l=1}^{L}\tilde{r}_l\, f(z^{(l)}) \qquad \text{(introducing } \tilde{r}_l = \frac{\tilde{p}(z^{(l)})}{\tilde{q}(z^{(l)})})$$

We call $\tilde{r}$ the *importance weights*

Importance Sampling (normalized)

$$Z_p = \int \tilde{p}(z)dz$$

$$\frac{Z_p}{Z_q} = \int \frac{\tilde{p}(z)}{Z_q}dz = \int \frac{\tilde{p}(z)}{\tilde{q}(z)}q(z)dz \qquad \text{(because } Z_q = \frac{\tilde{q}(z)}{q(z)}\text{)}$$

$$\cong \frac{1}{L}\sum_{l=1}^{L}\tilde{r}_l \qquad \text{(samples from } q\left(z^{(l)}\right)\text{, equivalently, } \tilde{q}\left(z^{(l)}\right)\text{)}$$

Importance Sampling (normalized)

$$E_{p(z)}[f] \cong \left(\frac{Z_q}{Z_p}\right)\frac{1}{L}\sum_{l=1}^{L}\tilde{r}_l f\left(z^{(l)}\right)$$ (samples coming from $\tilde{q}\left(z^{(l)}\right)$)

and $$\frac{Z_p}{Z_q} \cong \frac{1}{L}\sum_{l=1}^{L}\tilde{r}_l$$ (samples coming from $\tilde{q}\left(z^{(l)}\right)$)

Importance Sampling (normalized)

$$E_{p(z)}[f] \cong \left(\frac{Z_q}{Z_p}\right)\frac{1}{L}\sum_{l=1}^{L}\tilde{r}_l\, f\left(z^{(l)}\right)$$ (samples coming from $\tilde{q}\left(z^{(l)}\right)$)

and $$\frac{Z_p}{Z_q} \cong \frac{1}{L}\sum_{l=1}^{L}\tilde{r}_l$$ (samples coming from $\tilde{q}\left(z^{(l)}\right)$)

so $$E_{p(z)}[f] \cong \frac{\sum_{l=1}^{L}\tilde{r}_l\, f\left(z^{(l)}\right)}{\sum_{l=1}^{L}\tilde{r}_l}$$ (samples coming from $\tilde{q}\left(z^{(l)}\right)$)

where $$\tilde{r}_l = \frac{\tilde{p}\left(z^{(l)}\right)}{\tilde{q}\left(z^{(l)}\right)}$$

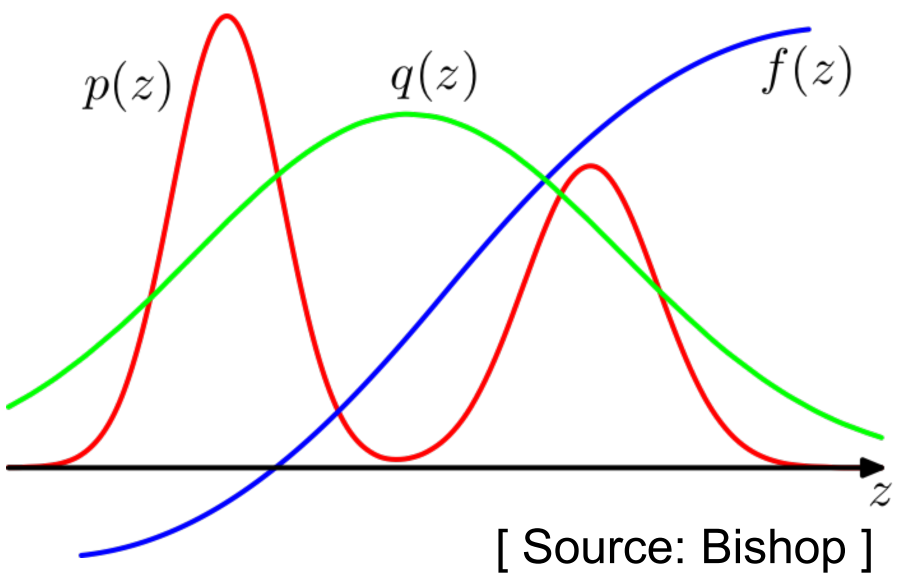
Importance Sampling On-A-Slide

1. Simulate from tractable distribution

$$\{z^{(l)}\}_{l=1}^{L} \sim q(z)$$

2. Compute importance weights & normalize

$$\widetilde{r}^{(l)} = \frac{\widetilde{p}(z^{(l)})}{q(z^{(l)})} \qquad r^{(l)} = \frac{\widetilde{r}^{(l)}}{\sum_{i=1}^{L} \widetilde{r}^{(i)}}$$

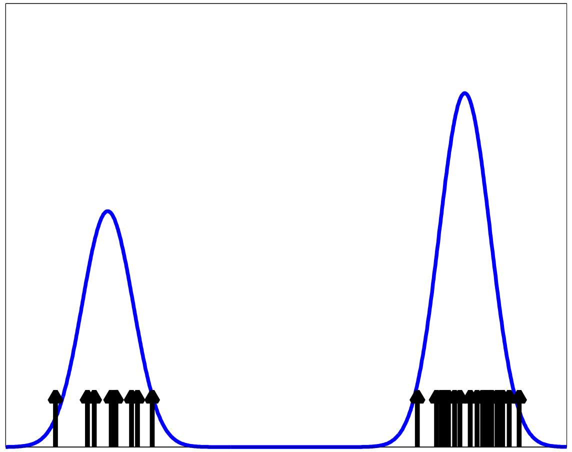


[Source: Bishop]

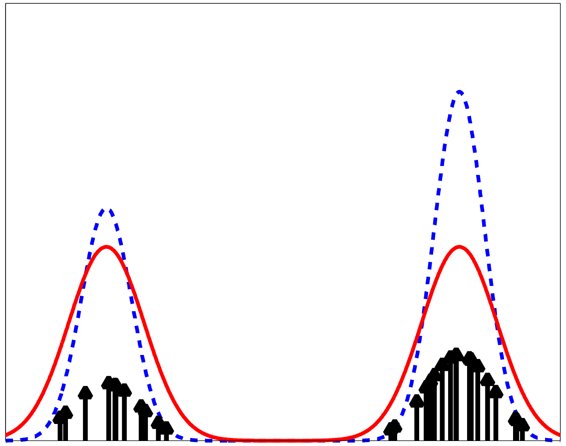
3. Compute importance-weighted expectation

$$\mathbf{E}_p[f(z)] \approx \sum_{l=1}^{L} r^{(l)} f(z^{(l)}) \equiv \hat{f}$$

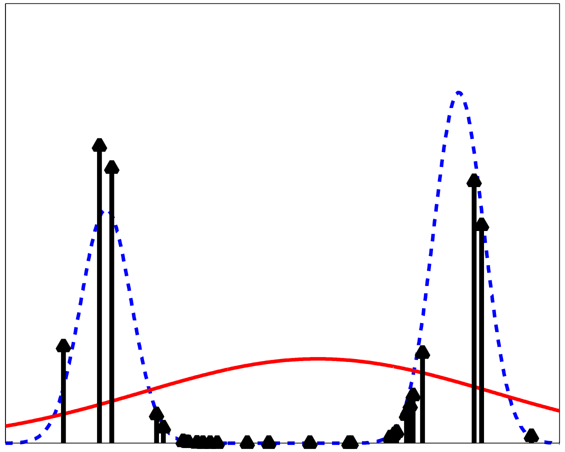
Selecting Proposal Distributions



Target Distribution



Good Proposal

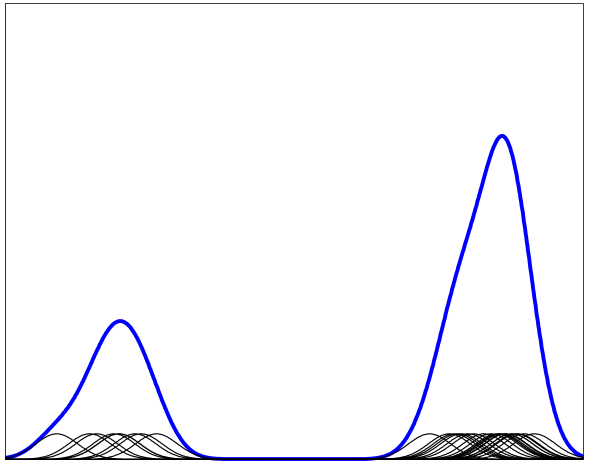


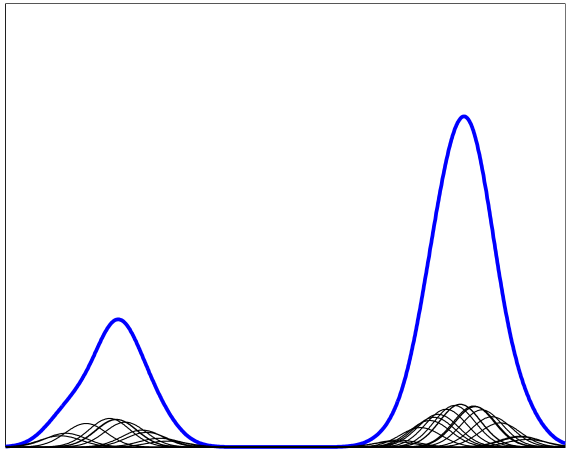
Poor Proposal

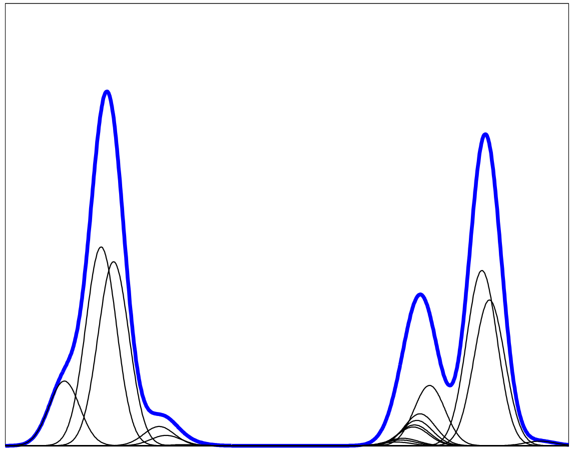
Kernel or Parzen window estimators interpolate to predict density:

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

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







Importance Sampling

Q: What is a good proposal distribution?

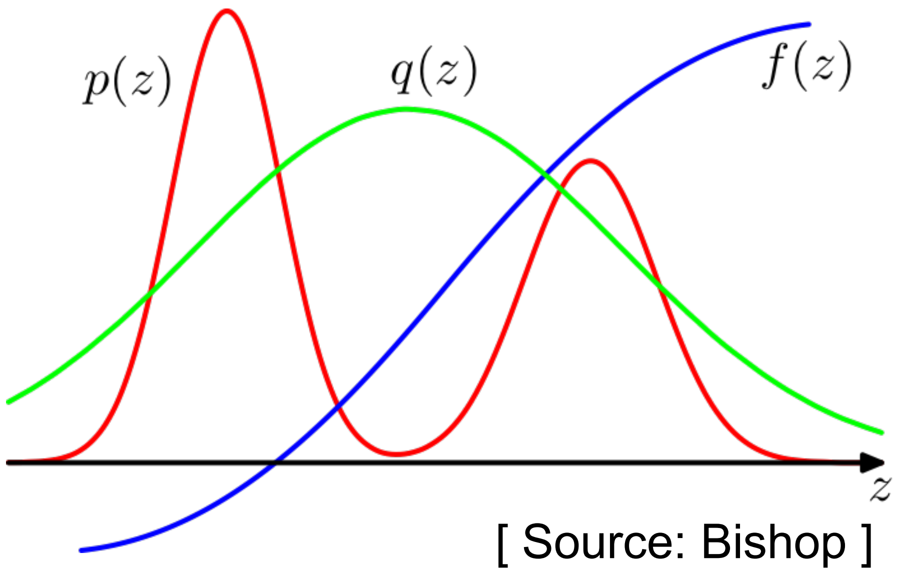
A: Minimize estimator variance

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

Minimum variance obtained when,

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

E.g. can do better than q=p



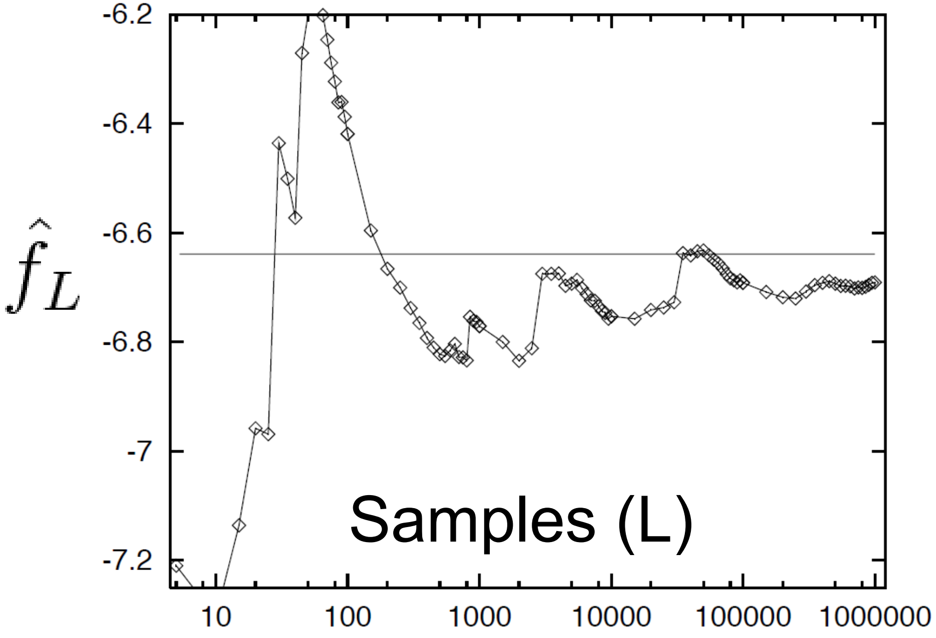
[Source: Bishop]

Estimator variance scales catastrophically with dimension:

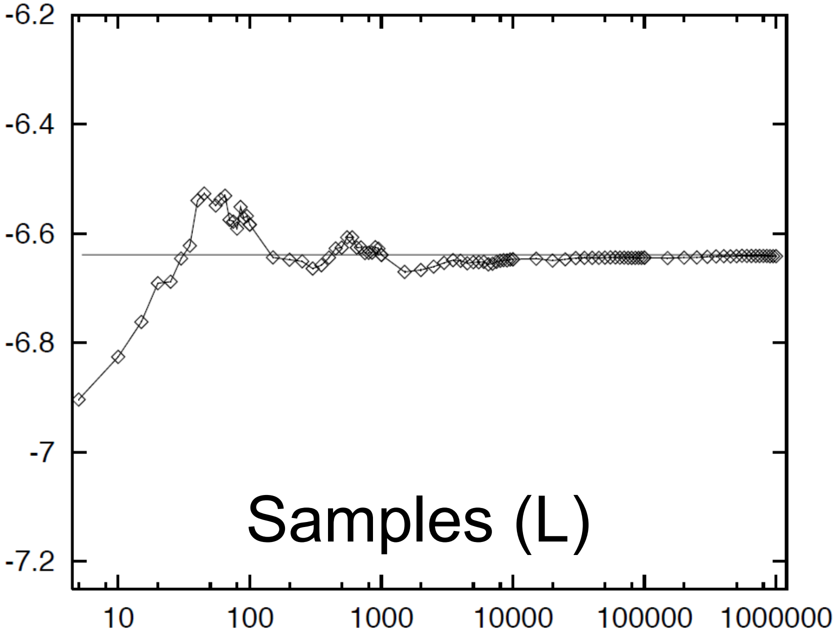
e.g. for N-dim. X and Gaussian q(x): $\mathrm{Var}_{q^*}(\hat{f}) = \exp(\sqrt{2N})$

Selecting Proposal Distributions

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



Gaussian Proposal



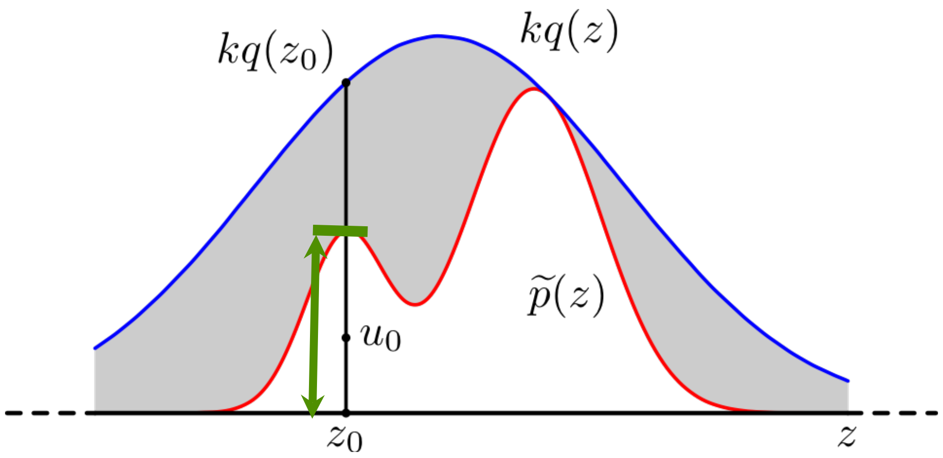
Cauchy (Student's-t) Proposal

Empirical variance of weights may not predict estimator variance!

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

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

Monte Carlo Methods Summary



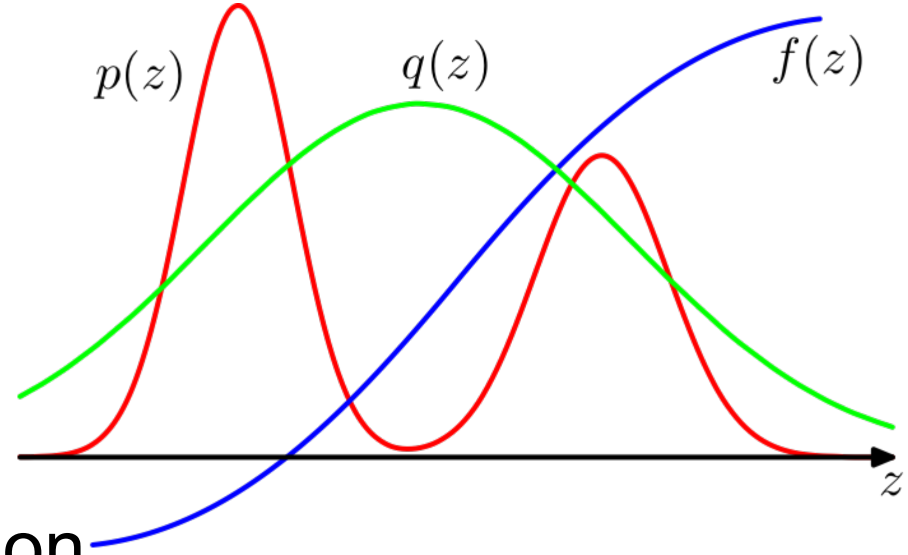
Rejection sampling

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

Pro: Efficient, easy to implement

Con: Acceptance rate evaporates as dimension increases

Importance Sampling



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

Pro: Efficient, easy to implement

Con: Variance grows exponentially in dimension

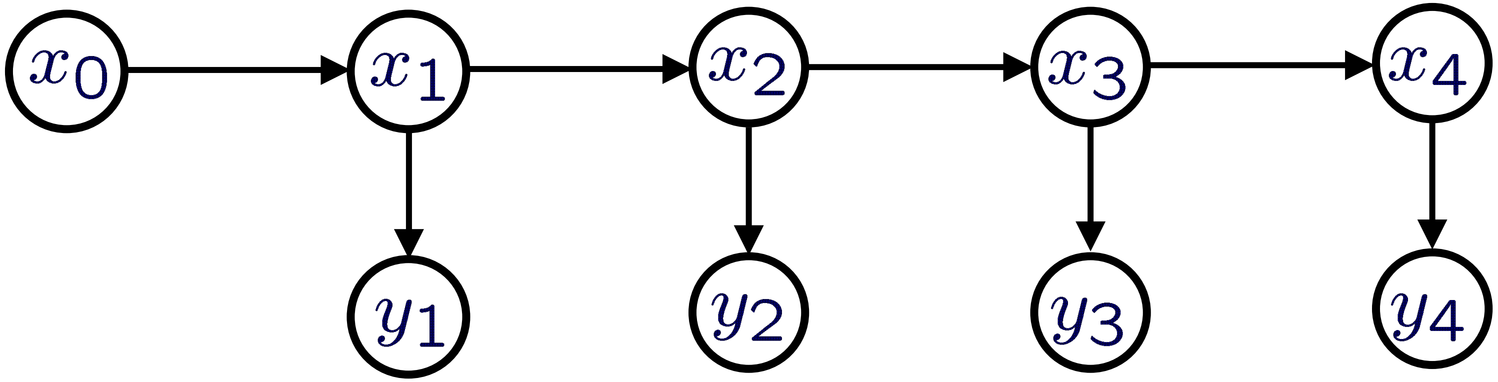
Outline

- 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



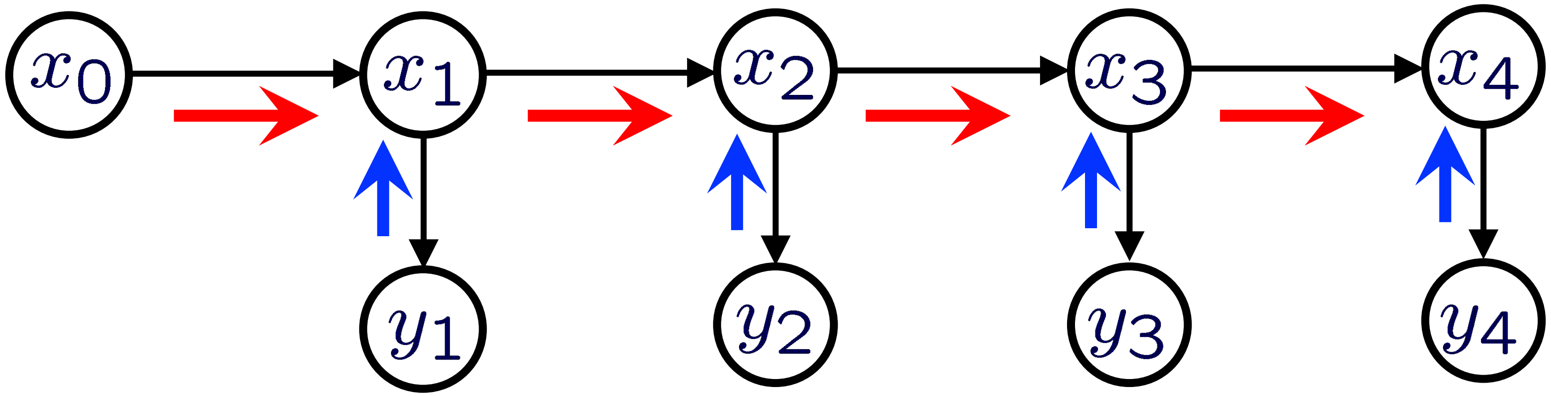
$x_t \in \mathbb{R}^d$

$y_t \in \mathbb{R}^k$

$$x_{t+1} = f(x_t, w_t) \qquad w_t \sim \mathcal{F}$$
$$y_t = g(x_t, v_t) \qquad v_t \sim \mathcal{G}$$

- 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



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

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

- Construct efficient proposal using Markov structure

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

$$q(x_t \mid x_{t-1}, y_t) \approx p(x_t \mid x_{t-1}, y)$$

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

But… weights will become degenerate, with most approaching zero

Particle Resampling

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

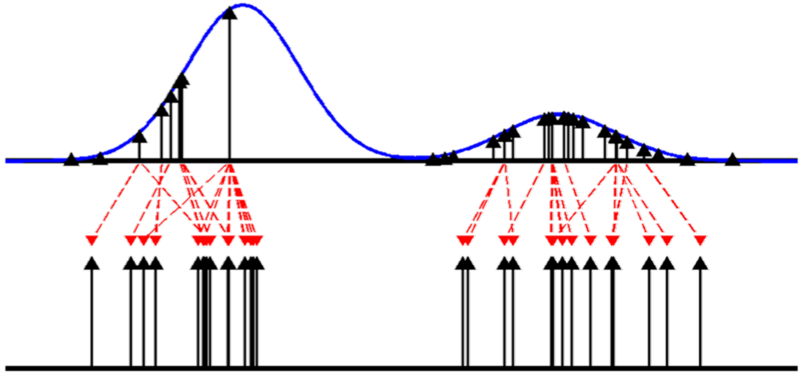
where $y_{\bar{t}} = \{y_1, \ldots, y_t\}$

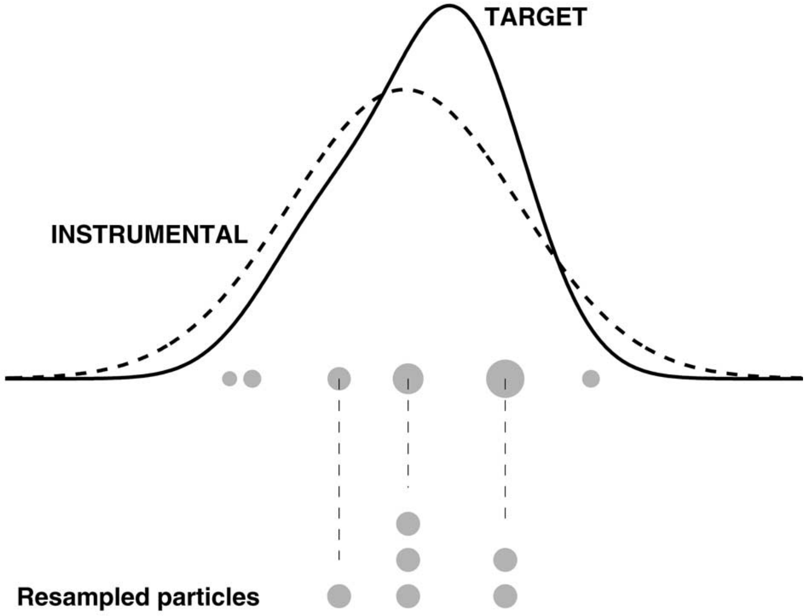
$$\bar{x}_t^{(\ell)} = x_t^{(j_\ell)}$$

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

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

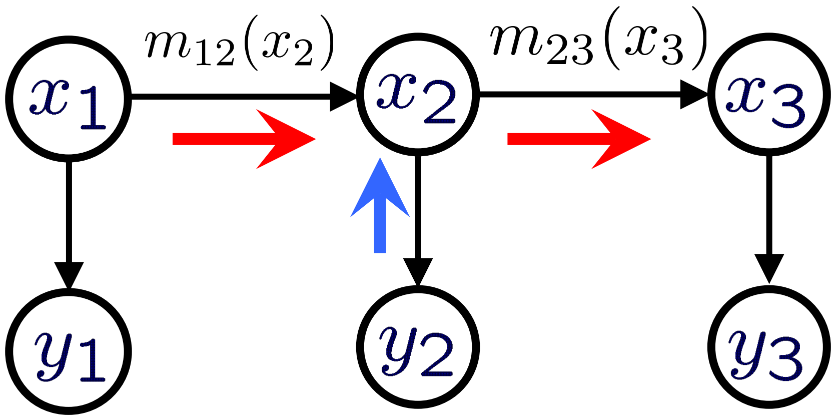
While remaining unbiased, resampling avoids degeneracies in which most weights go to zero





Particle Filtering Algorithms

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



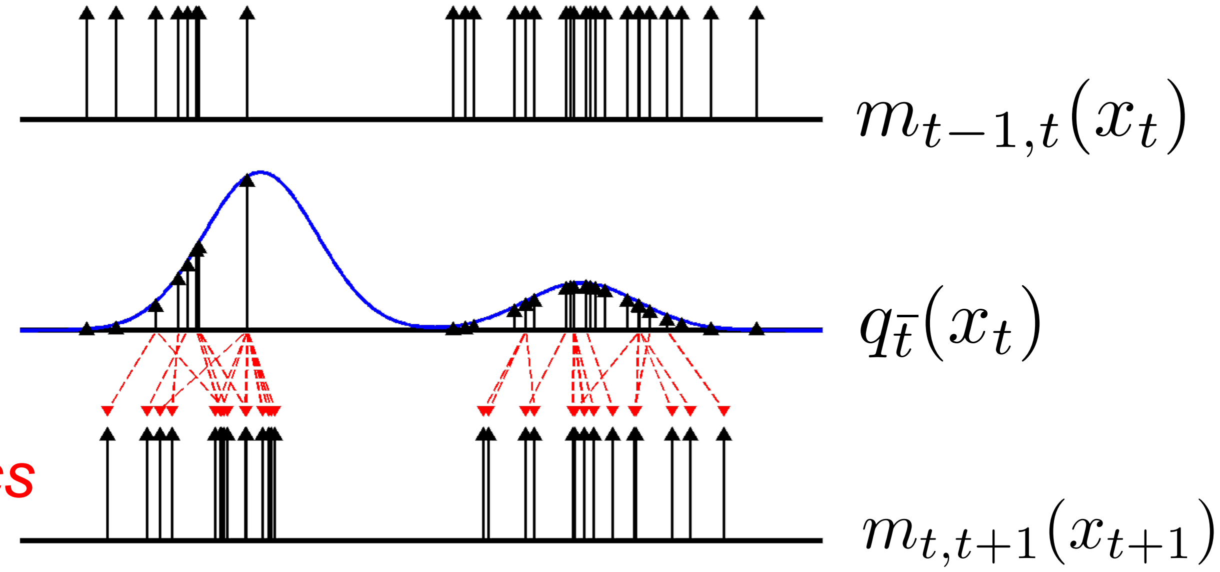
Sample-based density estimate

↓

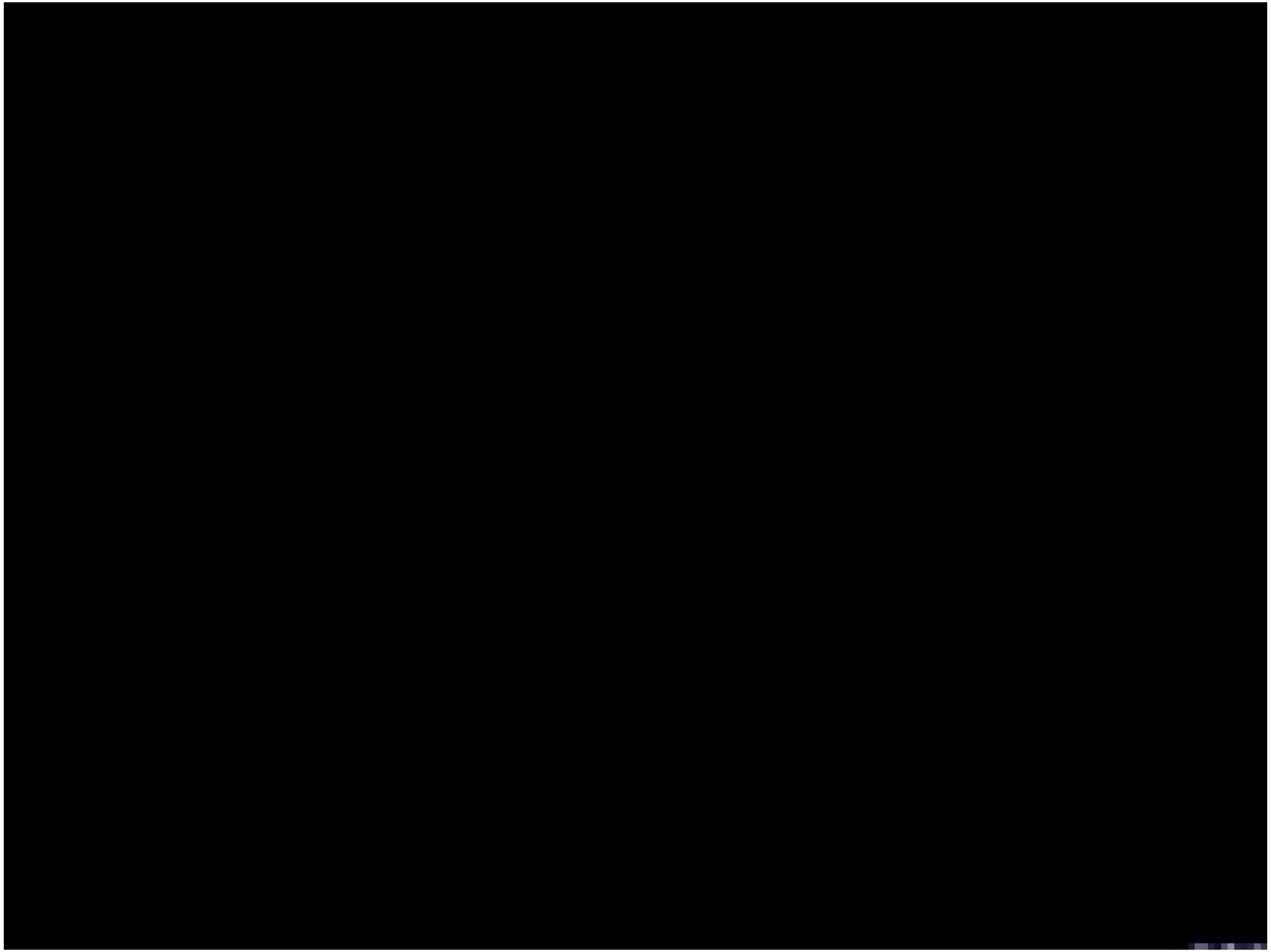
Weight by observation likelihood

↓

Resample & propagate by dynamics

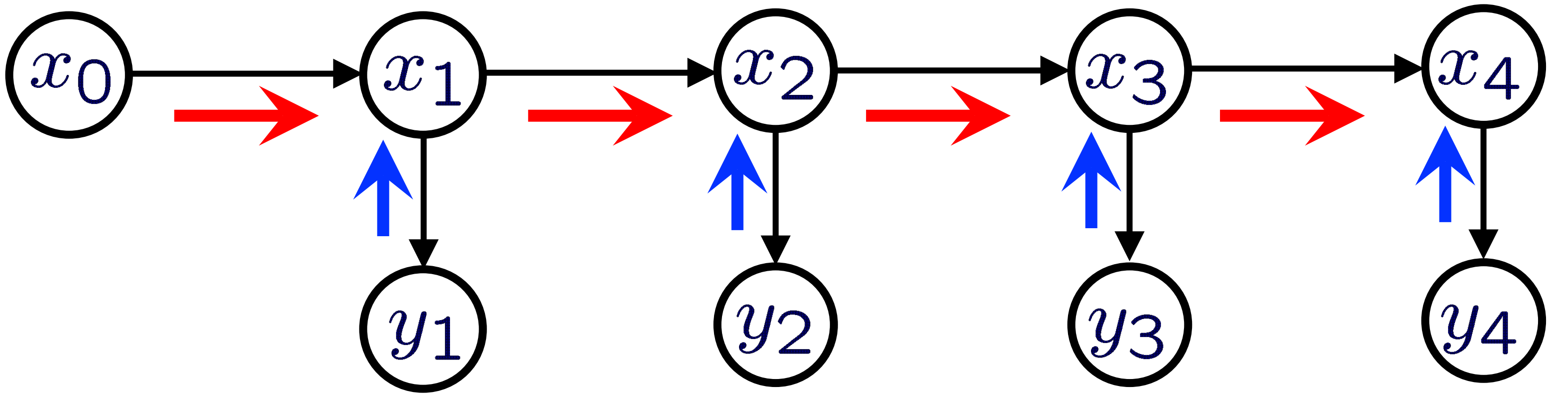


Particle Filters: The Movie



(M. Isard, 1996)

BP for State-Space Models



$$m_{t-1,t}(x_t) \propto p\left(x_t \mid y_{\overline{t-1}}\right) \qquad \text{where} \qquad y_{\bar{t}} = \{y_1, \ldots, y_t\}$$

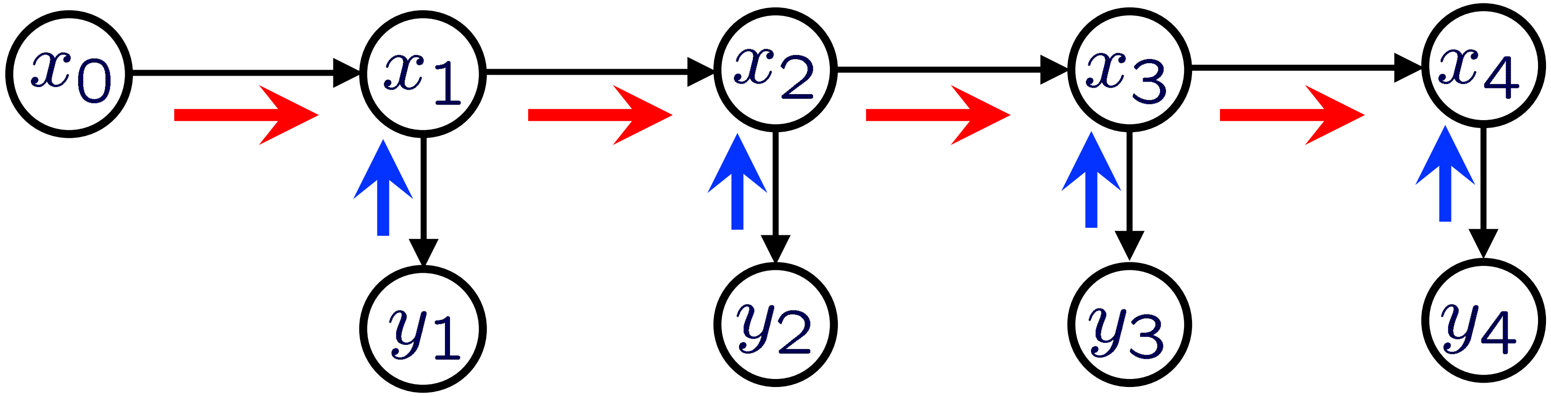
$$m_{t-1,t}(x_t)\, p(y_t \mid x_t) \propto p(x_t \mid y_{\bar{t}}) = q_{\bar{t}}(x_t)$$

Prediction (Integral/Sum step of BP):

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

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

Particle Filter: Measurement Update



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

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

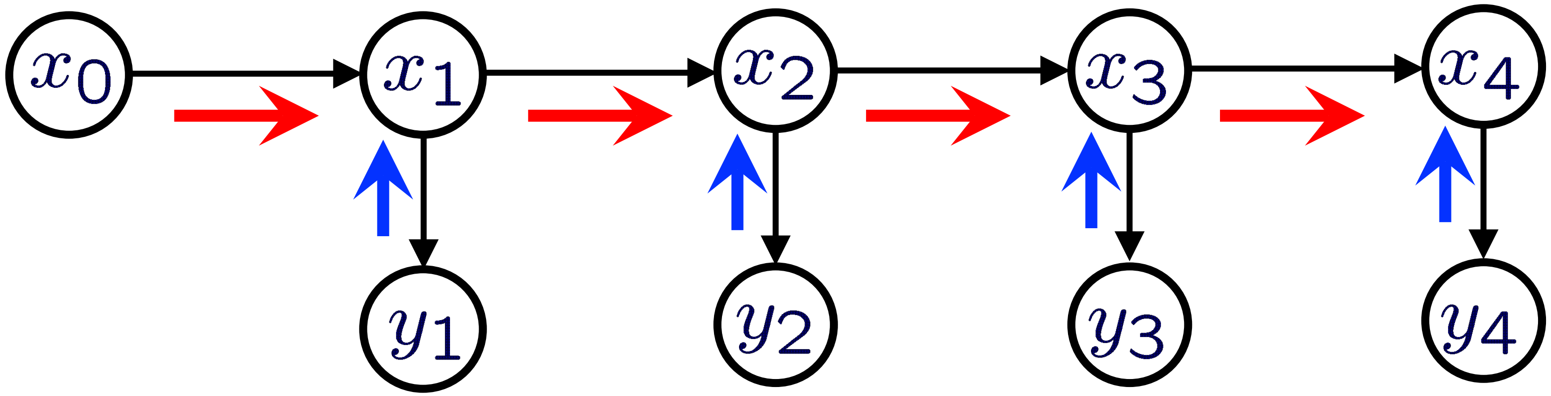
- **Bayes' Rule:** Posterior at particles proportional to prior times likelihood

$$q_{\bar{t}}(x_t) \propto m_{t-1,t}(x_t)\, p(y_t \mid x_t) \propto \sum_{\ell=1}^{L} w_{t-1,t}^{(\ell)}\, p(y_t \mid x_t^{(\ell)})\, \delta(x_t, x_t^{(\ell)})$$

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

*Variance of importance weights **increases** with each update*

Particle Filter: Sample Propagation



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

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

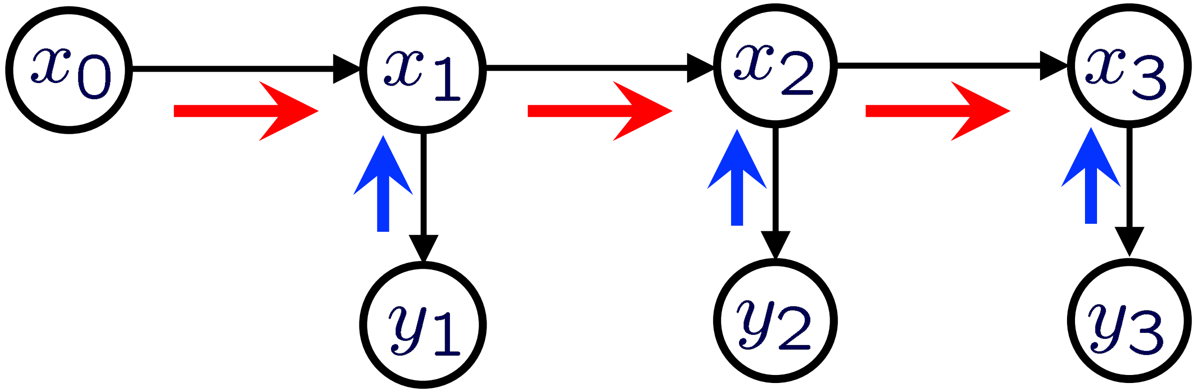
- **Prediction:** Sample next state conditioned on current particles

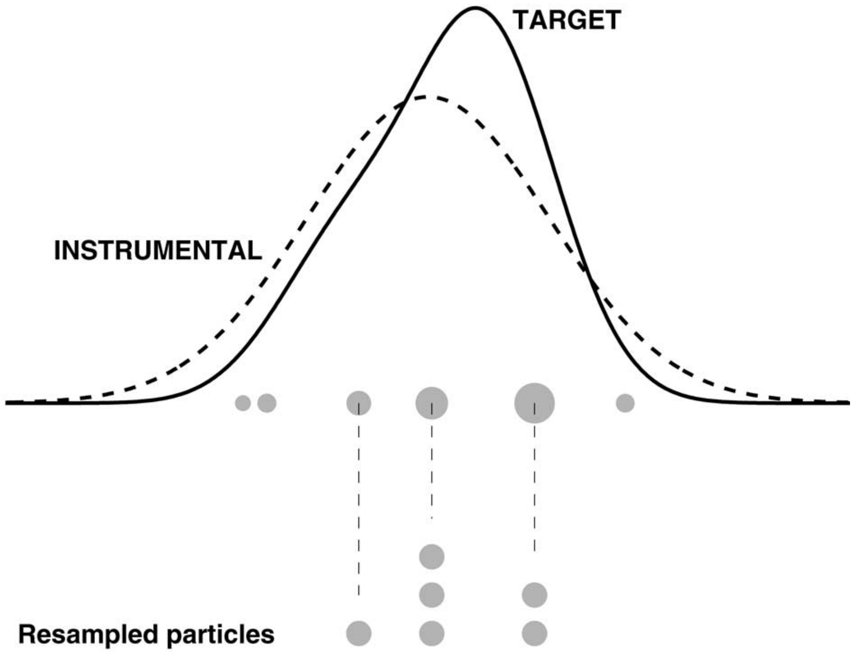
$$m_{t,t+1}(x_{t+1}) = \sum_{\ell=1}^{L} w_{t,t+1}^{(\ell)} \delta(x_{t+1}, x_{t+1}^{(\ell)})$$

$$x_{t+1}^{(\ell)} \sim p(x_{t+1} \mid x_t^{(\ell)})$$
$$w_{t,t+1}^{(\ell)} = w_t^{(\ell)}$$

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

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

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

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

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

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

Particle Filter: Resampling

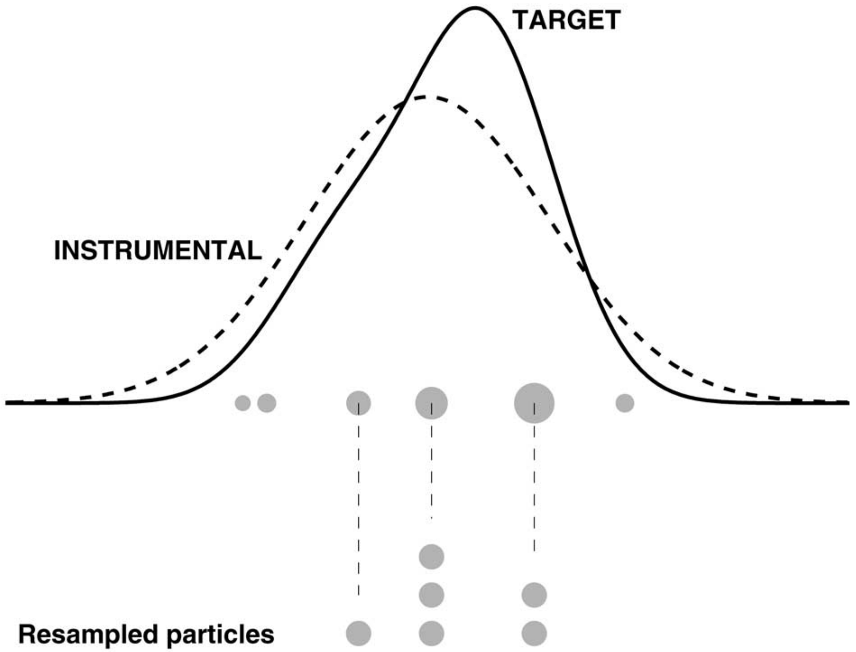
- **Effective Sample Size:**

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

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

- **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

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

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

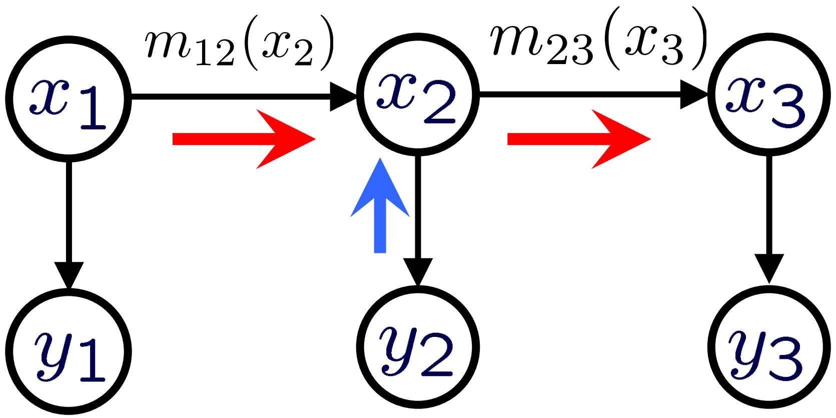
$$x_{t+1}^{(\ell)} \sim p(x_{t+1} \mid \tilde{x}_t^{(\ell)})$$

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

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

Particle Filtering Algorithms

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



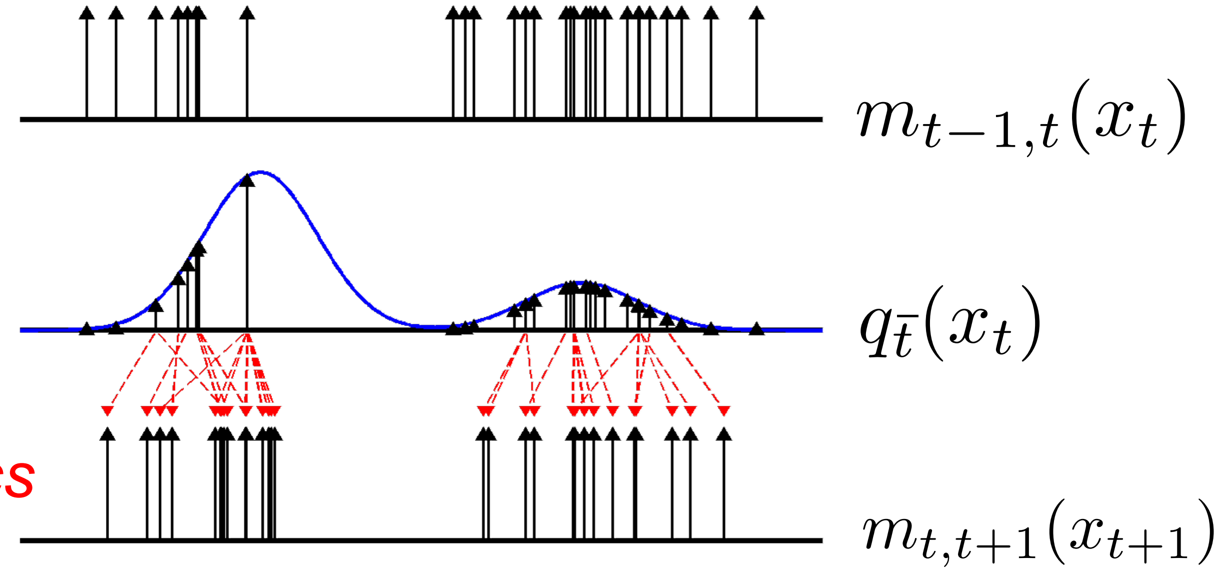
Sample-based density estimate

↓

Weight by observation likelihood

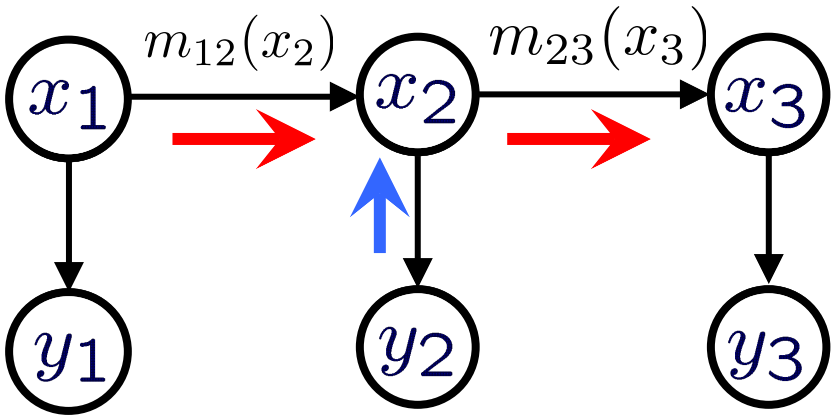
↓

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:

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

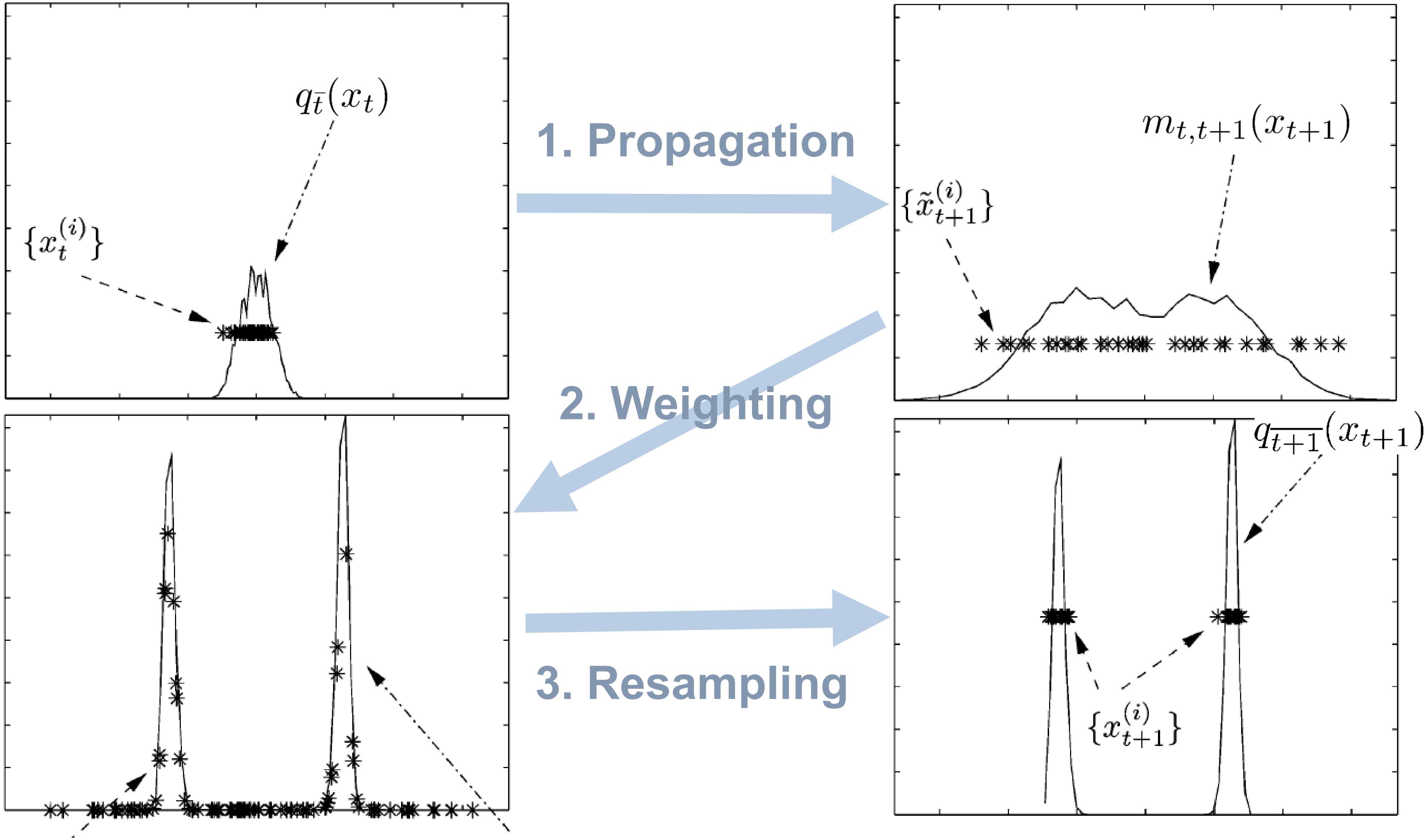
- Account for observation via importance weights:

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

- Sample from forward dynamics distribution of next state:

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

Bootstrap Particle Filter Summary

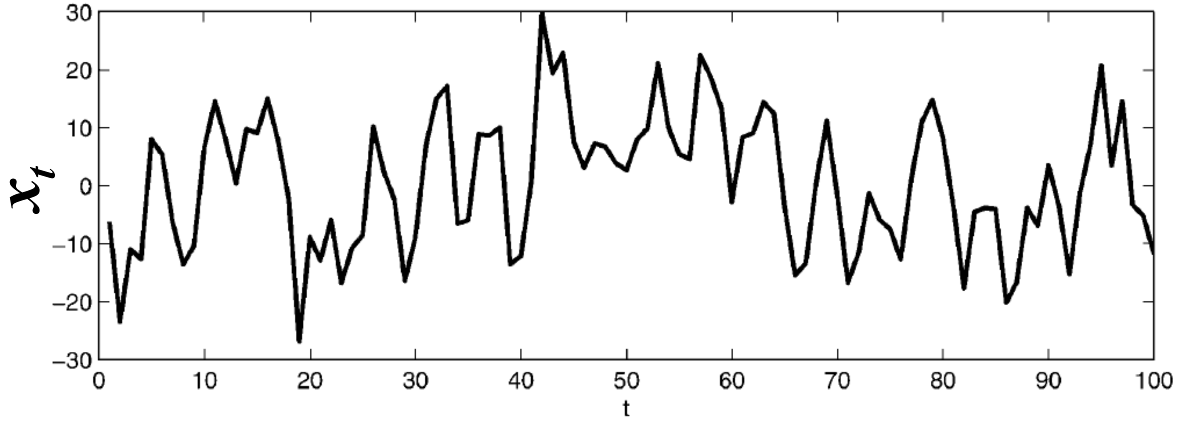


[Source: Cappe]

Toy Nonlinear Model

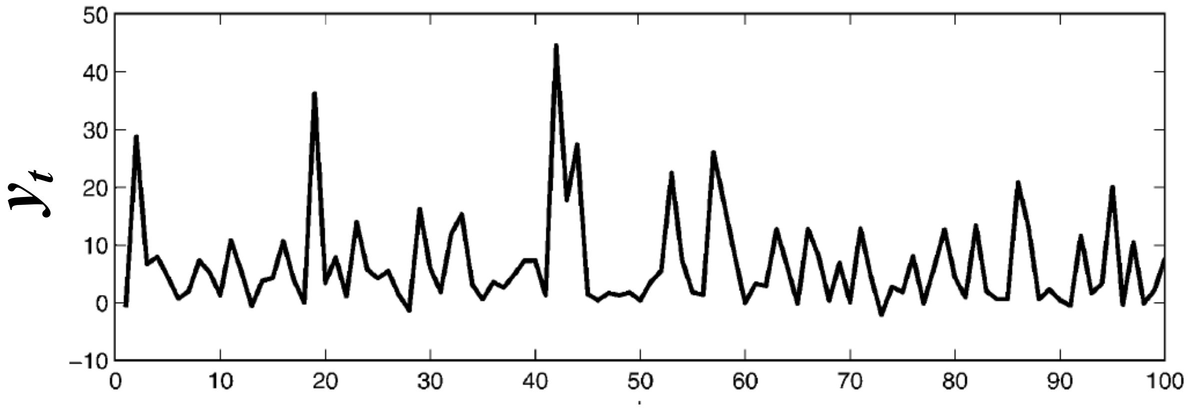
Nonlinear dynamics and observation model…

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



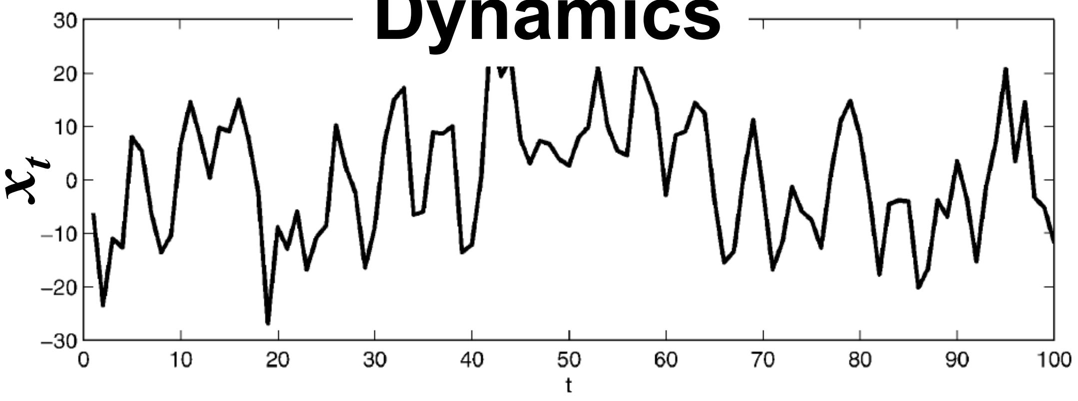
$$y_t = \frac{x_t^2}{20} + v_t$$

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

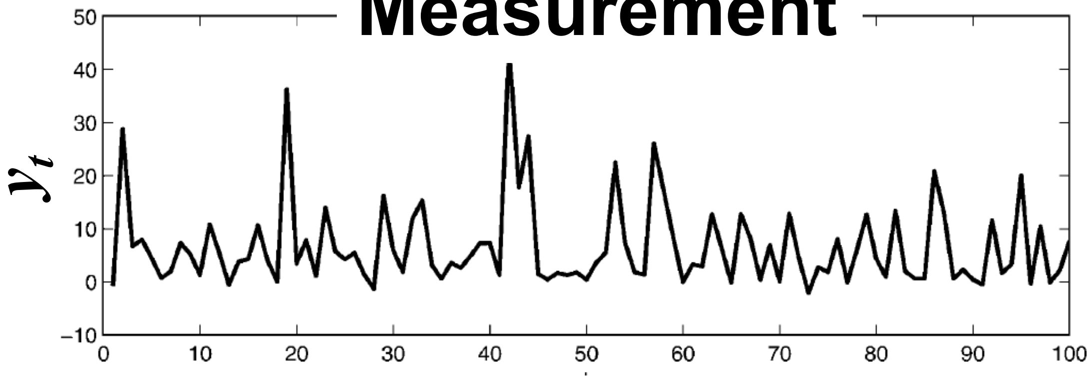
…filter equations lack closed form.

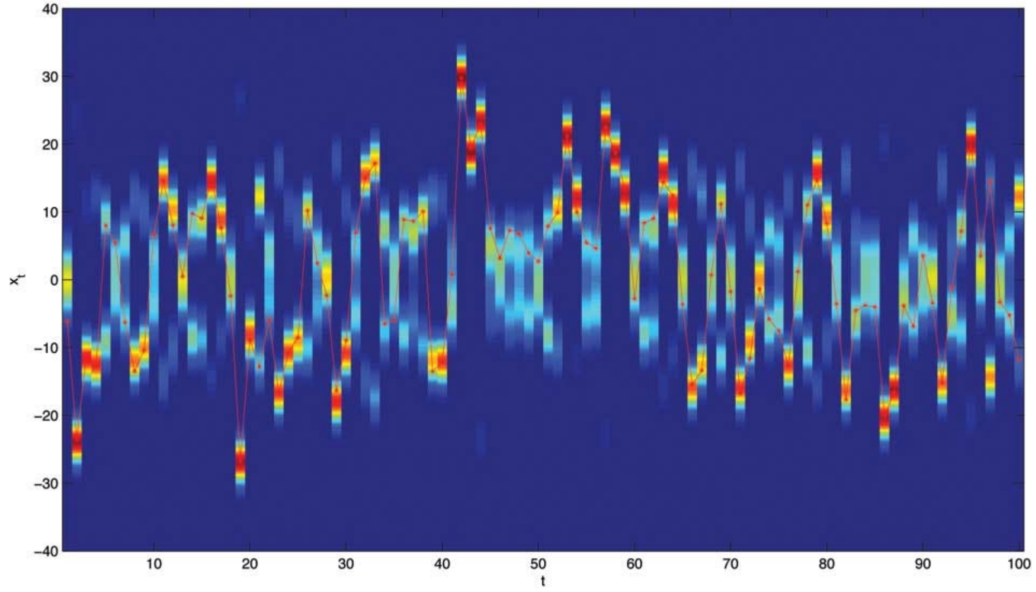
Toy Nonlinear Model

Dynamics

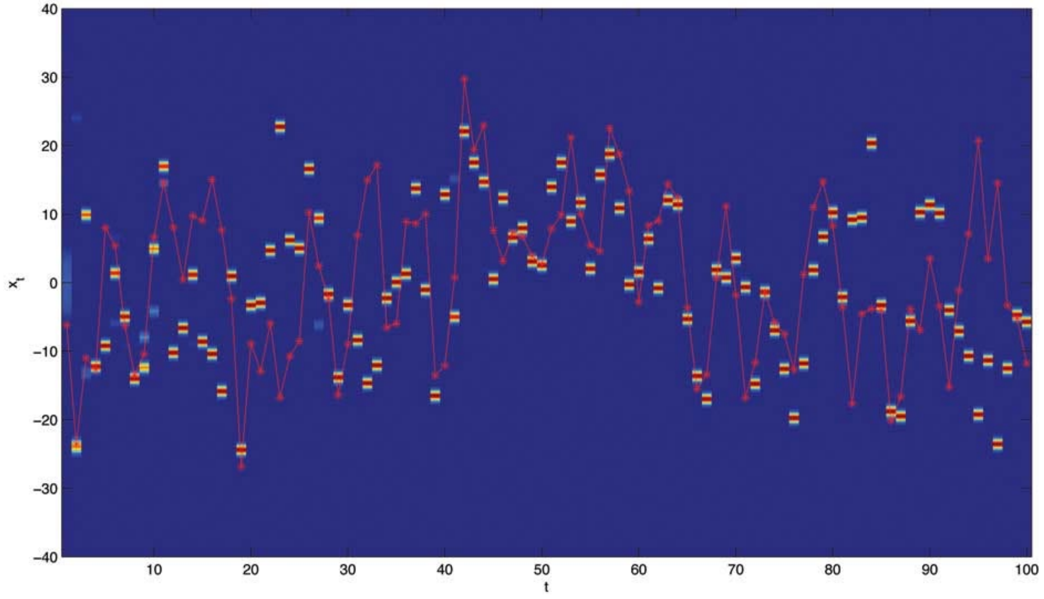


Measurement





Particle Filter Marginal KDEs

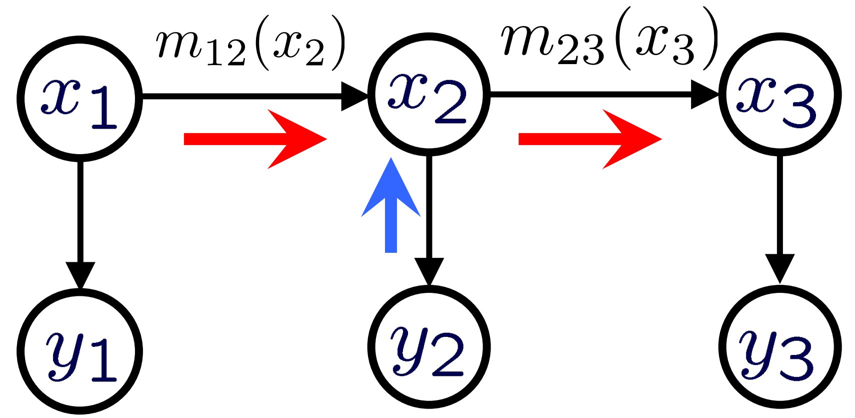


Full Sequence Importance Sampling

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:



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

- Sample from a *proposal distribution q*:

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

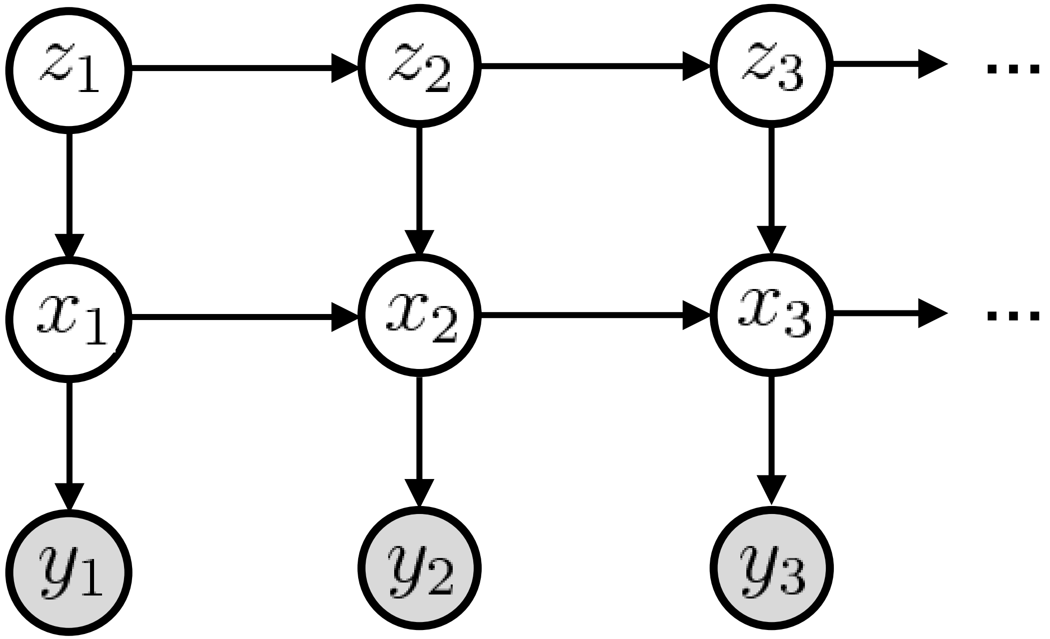
- Account for *observation and proposal* via importance weights:

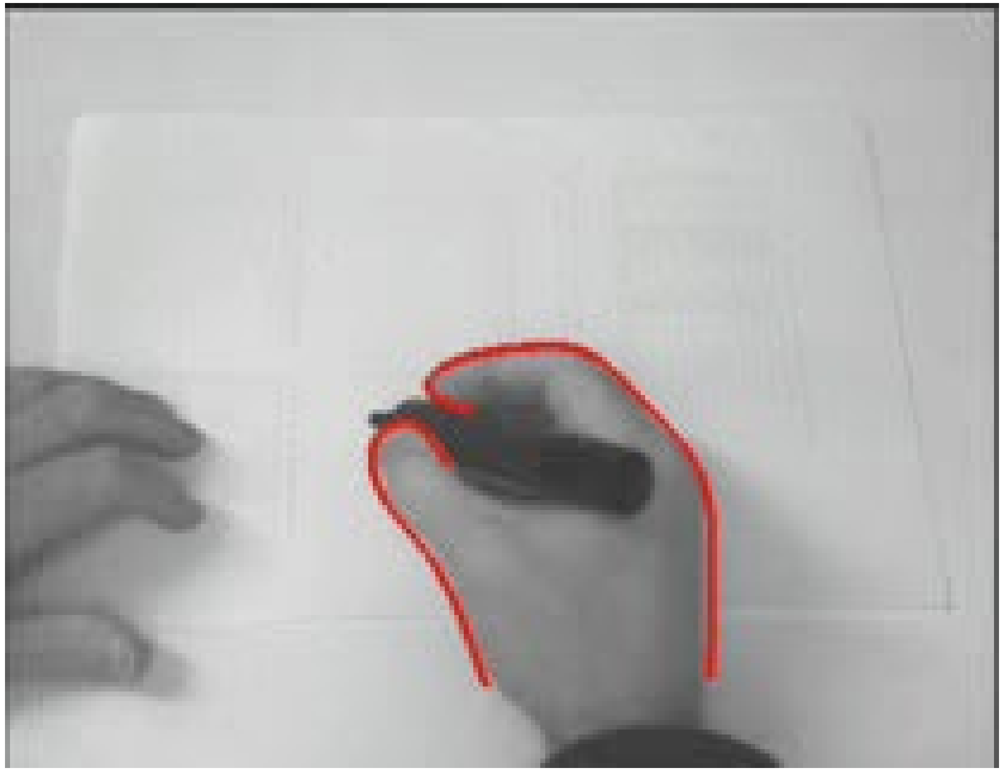
$$w_t^{(\ell)} \propto \frac{p(x_t^{(\ell)} \mid x_{t-1}^{(\ell)}) p(y_t \mid x_t^{(\ell)})}{q(x_t^{(\ell)} \mid x_{t-1}^{(\ell)}, y_t)}$$

- Resample to avoid particle degeneracy:

$$p(x_t \mid y_t, \ldots, y_1) \approx \sum_{\ell=1}^{L} \frac{1}{L} \delta_{x_t^{(\ell)}}(x_t) \qquad x_t^{(\ell)} \sim \sum_{m=1}^{L} w_t^{(m)} \delta_{x_t^{(m)}}(x_t)$$

Switching State-Space Model





Colors indicate 3 writing modes

[Video: Isard & Blake, ICCV 1998.]

Discrete switching state:

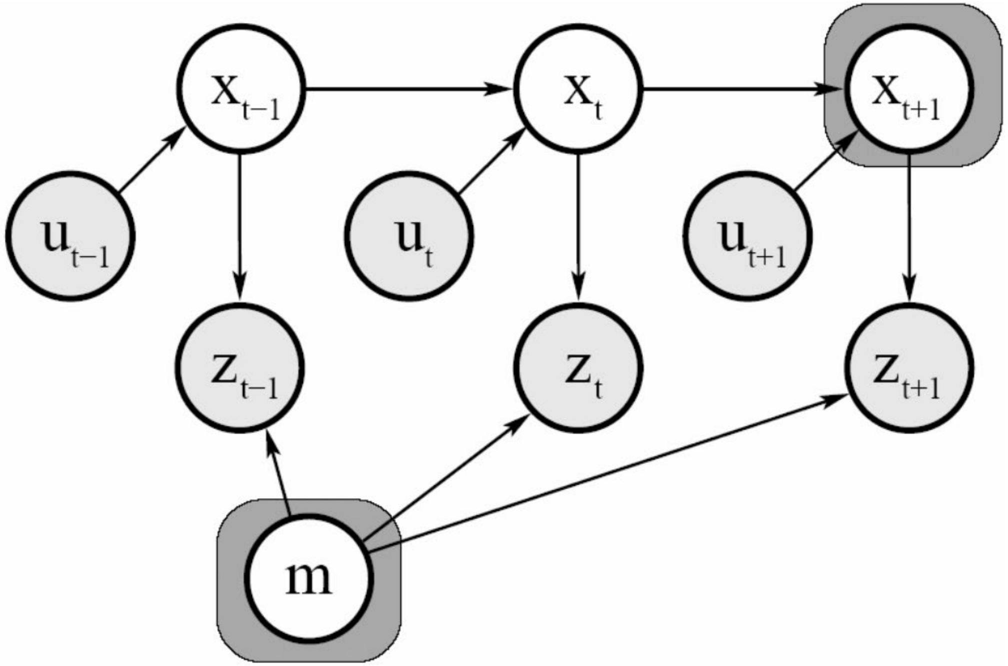
$$z_t \mid z_{t-1} \sim \mathrm{Cat}(\pi(z_{t-1}))$$ With stochastic transition matrix π

Switching state selects dynamics:

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

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)

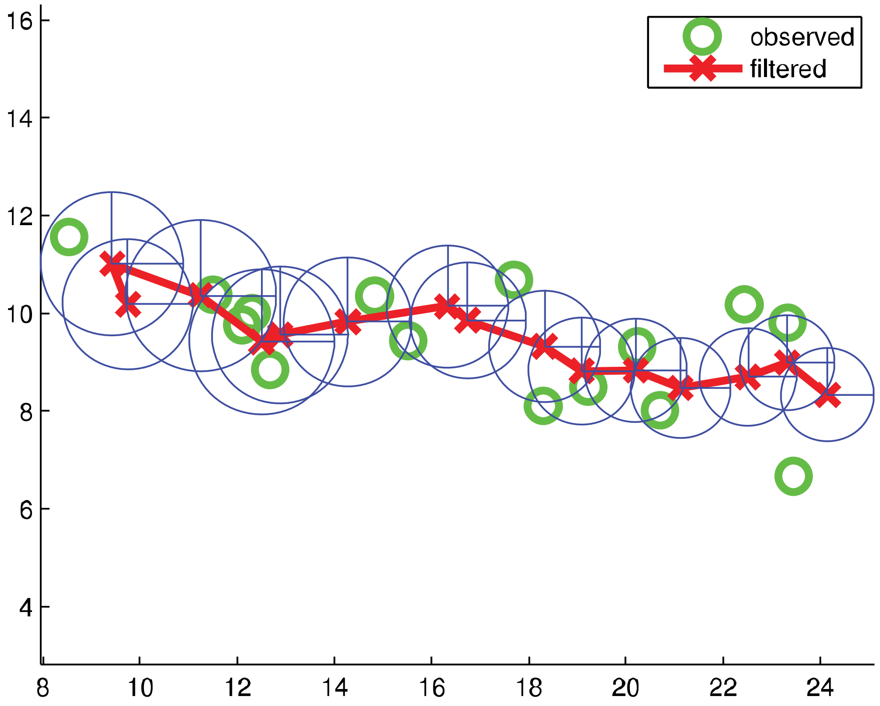
True trajectory (GPS)

Inferred trajectory & landmarks

Dynamical System Inference

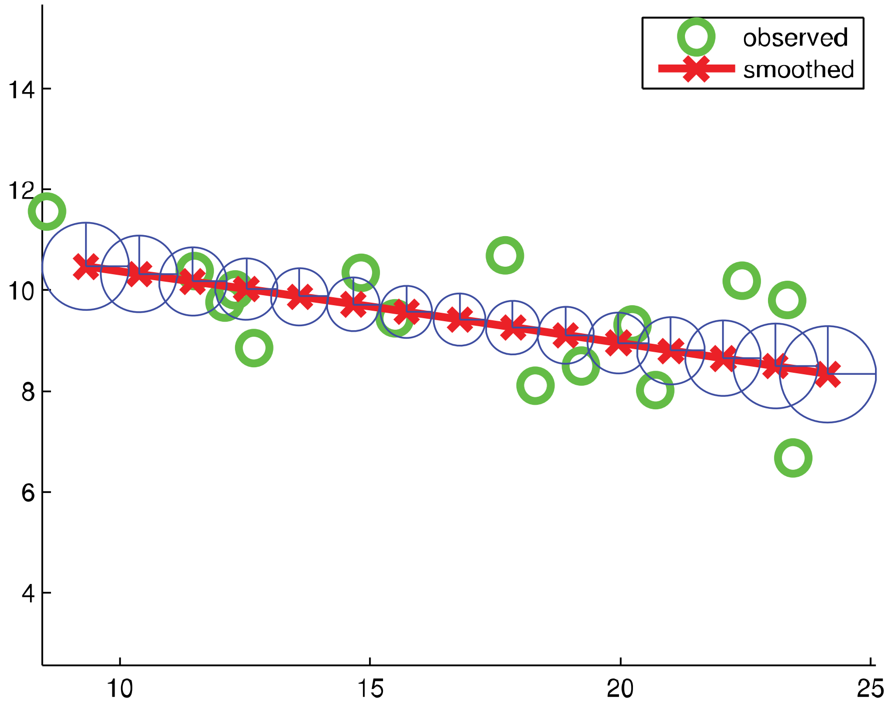
Define shorthand notation: $y_1^{t-1} \triangleq \{y_1, \ldots, y_{t-1}\}$

Filtering



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

Smoothing



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

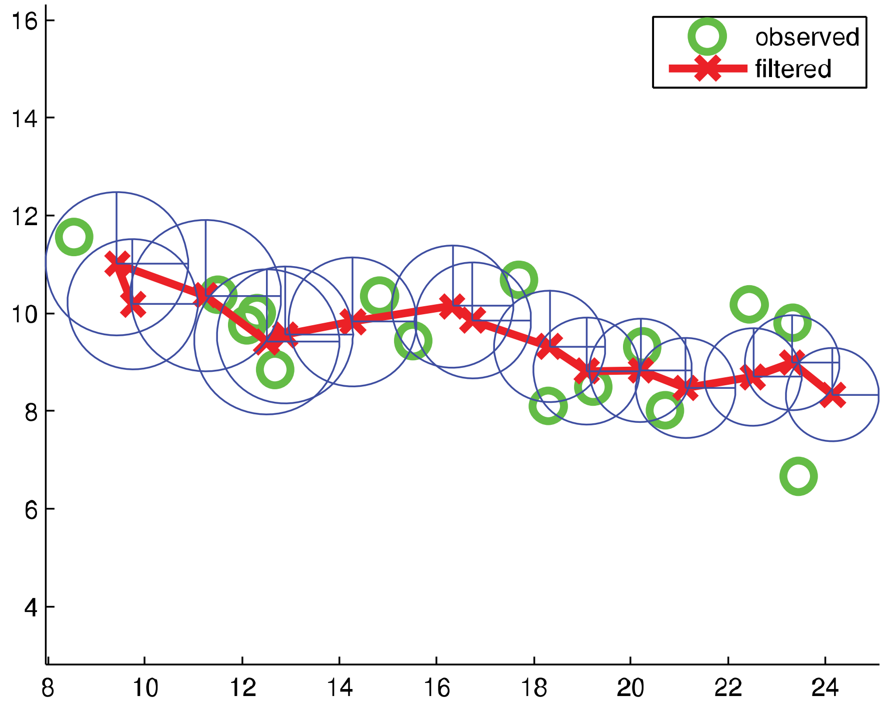
Administrivia

- HW4 Due Tonight
- HW5 Will be out next Monday 11/23
- Midterm grades by tomorrow

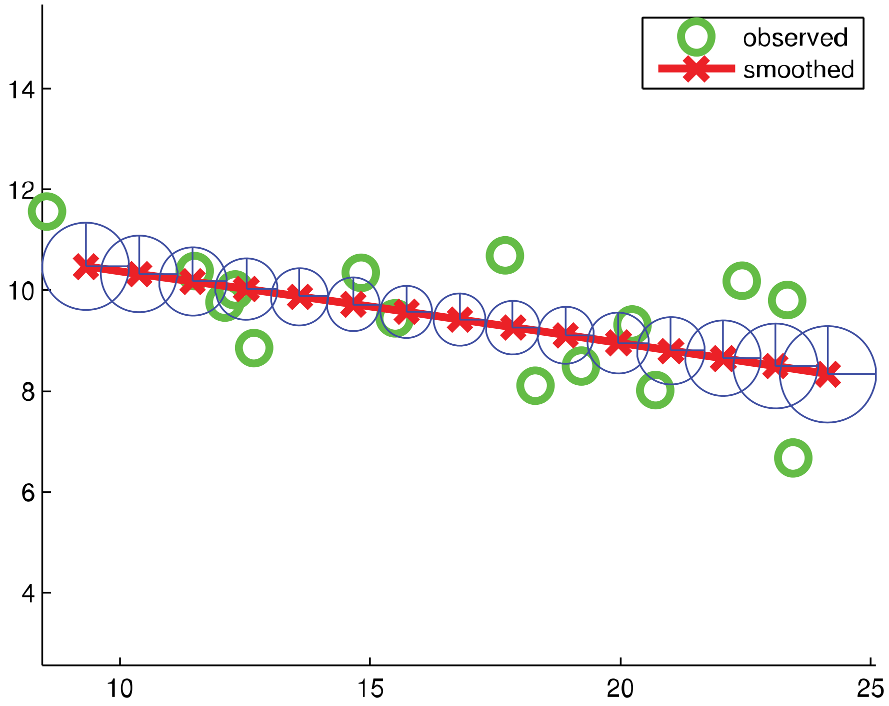
Dynamical System Inference

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

Filtering

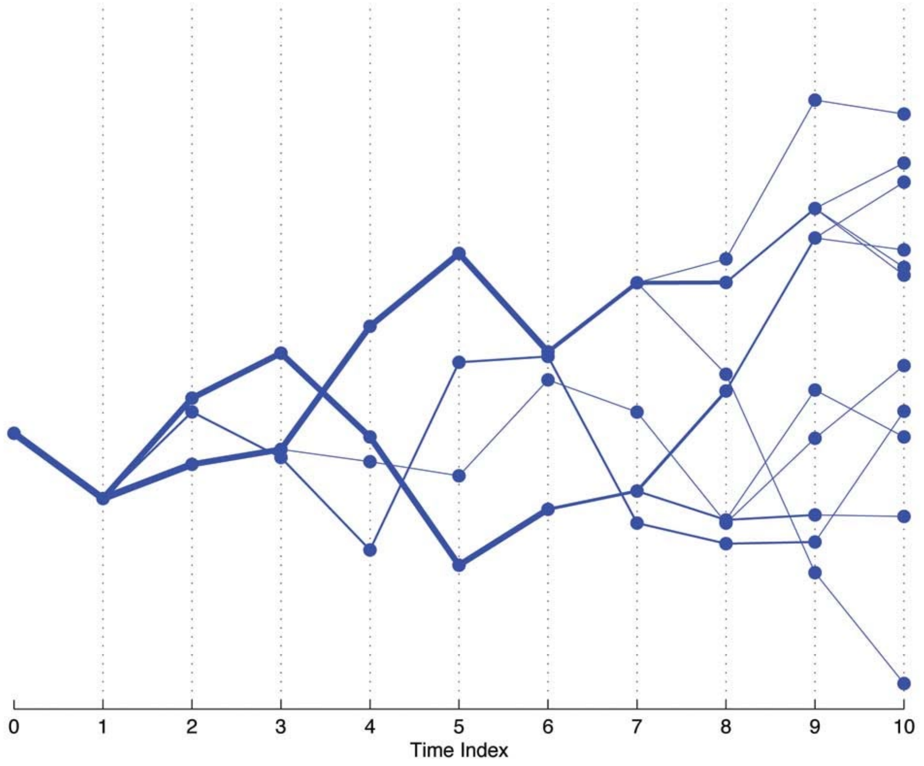


Smoothing



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

A Note On Smoothing



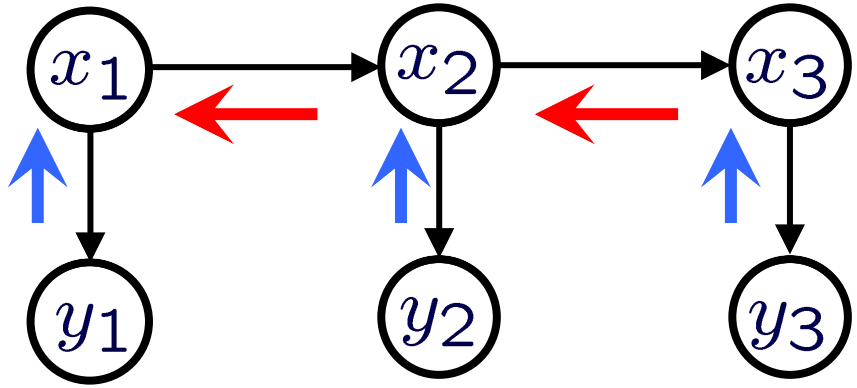
- ➢ Each resampling step discards states and they cannot subsequently restored
- ➢ Resampling introduces dependence across trajectories (common ancestors)
- ➢ Smoothed marginal estimates are generally poor
- ➢ Backwards simulation improves estimates of smoothed trajectories

Particle Filter Smoothing

Smoothing distribution factorizes as,

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

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



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

Filter distribution at time T

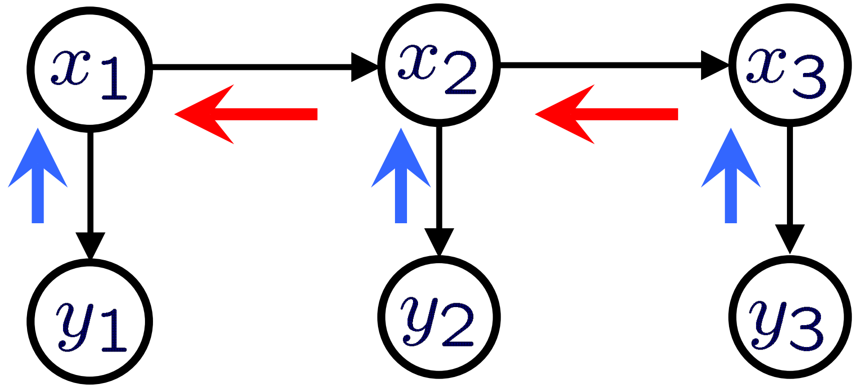
Suggests an algorithm to sample from $p(x_1^T \mid y_1^T)$:

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

Use resampling idea to sample from current particle trajectories in reverse

Particle Filter Smoothing

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



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

$$\propto p(x_{t+1} \mid x_t) p(x_t \mid y_1^t)$$

Forward pass sample-based filter marginal estimates:

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

Thus particle estimate of reverse prediction is:

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

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 $\{x_t^{(i)}, \omega_t^{(i)}\}_{1 \le i \le L}$

end for

Choose $\widetilde{x}_T = x_T^{(i)}$ with probability $\omega_t^{(i)}$.

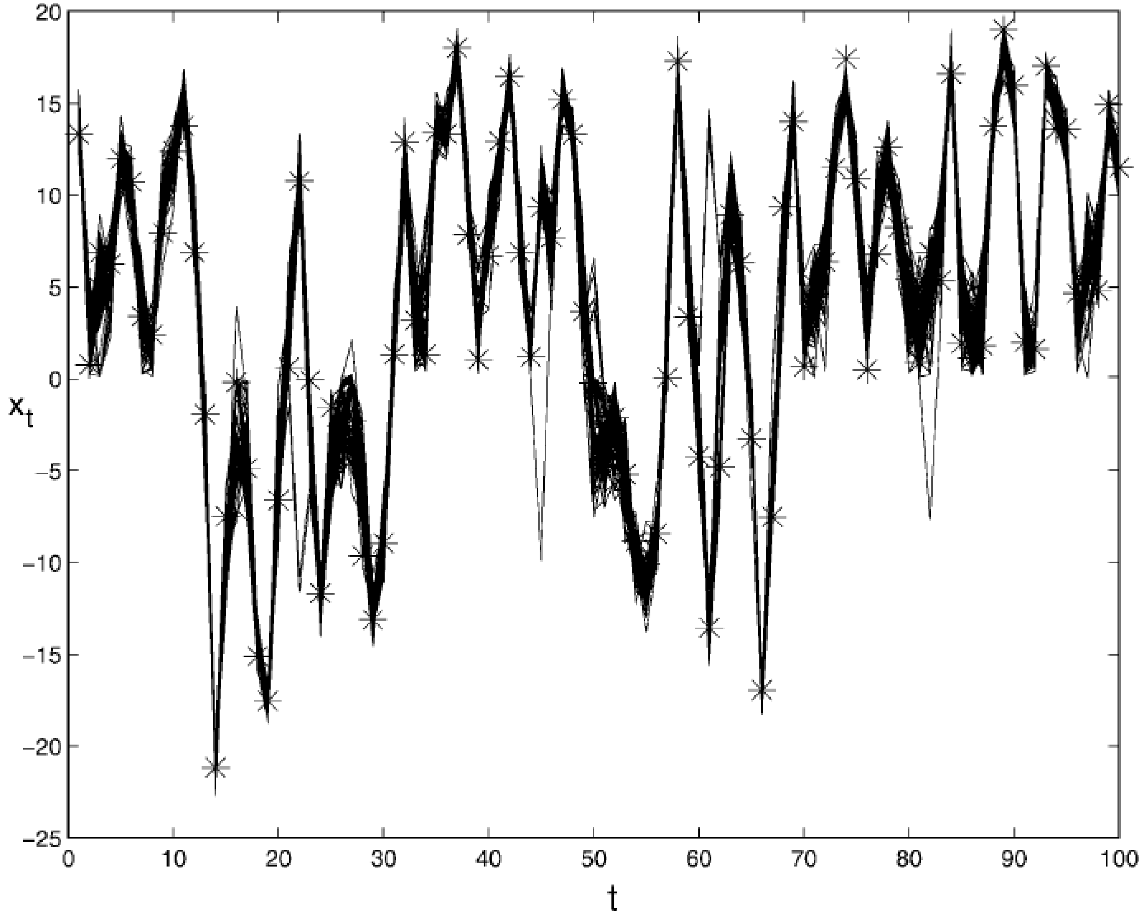
for $t = T - 1$ to 1 **do** ▷ Backward Pass Smoother

  Calculate $\rho_t^{(i)} \propto \omega_t^{(i)} p(\tilde{x}_{t+1} \mid x_t^{(i)})$ for $i = 1, \dots, L$ and normalize the modified weights.

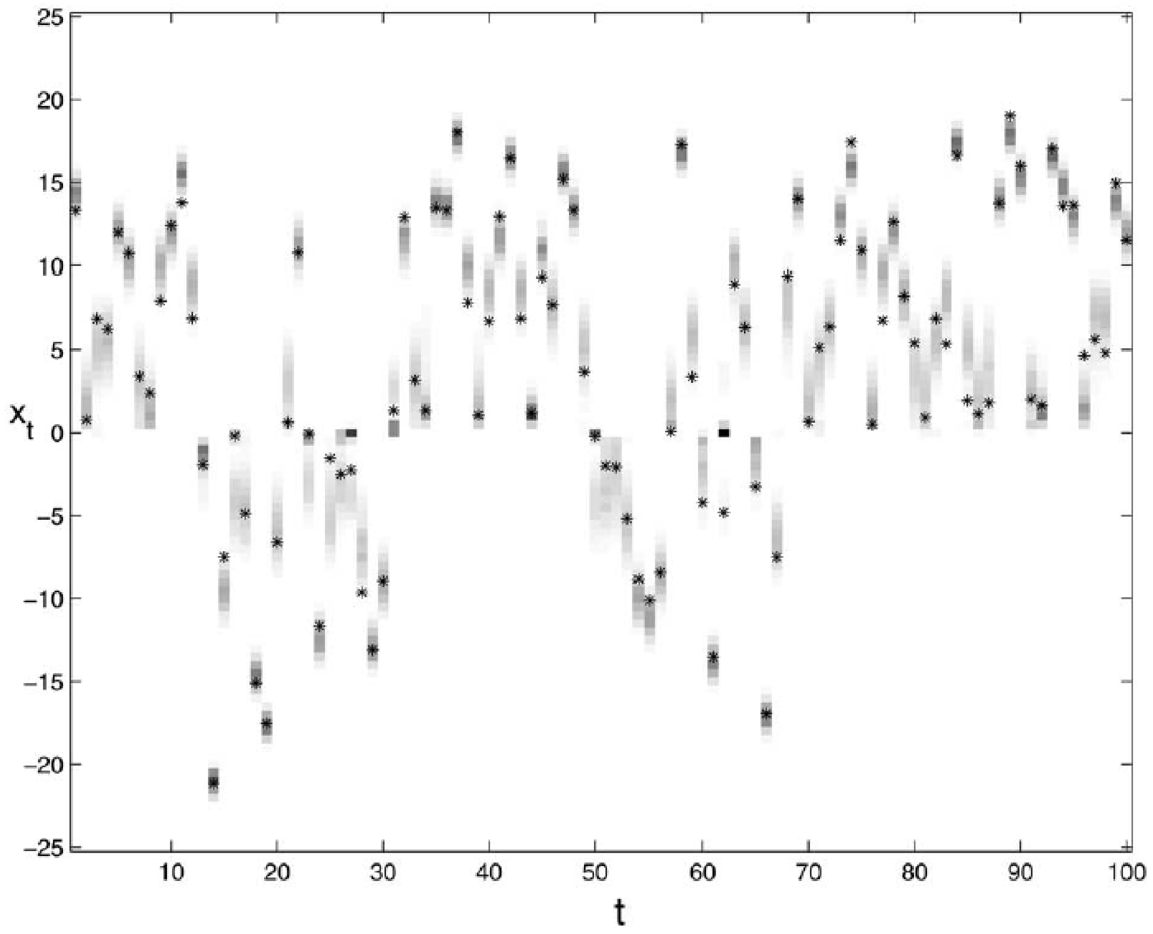
  Choose $\widetilde{x}_t = x_t^{(i)}$ with probability $\rho_t^{(i)}$.

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 $p(x_1^t \mid y_1^t)$ to sample trajectory prior to the resampling step
- Other smoothing topics:
  - Generalized two-filter smoothing
  - MC approximation of posterior marginals $p(x_t \mid y_1^T)$
- *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

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

- 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

See separate MCMC slides…

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)

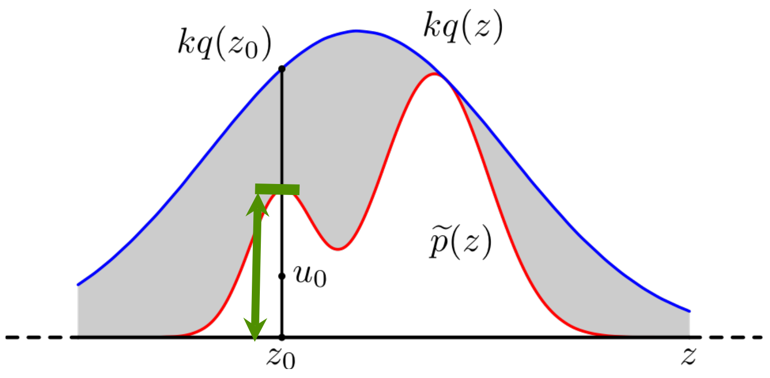
Monte Carlo Methods Summary

- In complex models we often have no other choice than to simulate realizations

- Rejection sampler choose proposal/constant s.t. $\widetilde{p}(z) \leq kq(z)$

1) Sample $q(z)$

2) Keep samples in proportion to $\dfrac{\tilde{p}(z)}{k \bullet q(z)}$ 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 $\{z^{(i)}\}_{i=1}^{L} \sim q$,

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

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

Proposal

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,

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

"Bootstrap" Particle Filter

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

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