Topics in Brain Computer Interfaces
CS295-7

Professor: **MICHAEL BLACK**

TA: **FRANK WOOD**

Spring 2005

Bayesian Inference through Particle Filtering

Homework Review

- Results?
- Questions?
- Causal vs. Generative?

Decoding Methods

Direct decoding methods:

$$\vec{x}_k = f(\vec{z}_k, \vec{z}_{k-1}, \ldots)$$

Simple linear regression method

$$x_k = \vec{f}_1^{\,T} \vec{Z}_{k:k-d}$$

$$y_k = \vec{f}_2^{\,T} \vec{Z}_{k:k-d}$$

Decoding Methods

Direct decoding methods:

$$\vec{x}_k = f(\vec{z}_k, \vec{z}_{k-1}, \ldots)$$

In contrast to generative encoding models:

$$\vec{z}_k = f(\vec{x}_k)$$

Need a sound way to exploit generative models for decoding.

Today's Strategy

- More mathematical than the previous classes.
- Group exploration and discovery.
- One topic with deeper level of understanding.
  – Particle Filtering
    - Review and explore recursive Bayesian estimation
    - Introduce SIS algorithm
    - Explore Monte Carlo integration
    - Examine SIS algorithm (if time permits)

Accounting for Uncertainty

Every real process has process and measurement noise

$$\vec{x}_k = \mathrm{f}_{\text{process}}(\vec{x}_{0:k-1}) + \text{noise}$$
$$\vec{z}_k = \mathrm{f}_{\text{observation}}(\vec{x}_{0:k}) + \text{noise}$$

A probabilistic process model accounts for process and measurement noise probabilistically. Noise appears as modeling uncertainty.

$$\vec{x}_k \mid \vec{x}_{k-1} \sim \hat{\mathrm{f}}_{\text{process}}(\vec{x}_{0:k-1}) + \text{uncertainty}$$
$$\vec{z}_k \mid \vec{x}_k \sim \hat{\mathrm{f}}_{\text{observation}}(\vec{x}_{0:k}) + \text{uncertainty}$$

Example: missile interceptor system. The missile propulsion system is noisy and radar observations are noisy. Even if we are given exact process and observation models our estimate of the missile's position may diverge if we don't account for uncertainty.

Recursive Bayesian Estimation

- Optimally integrates subsequent observations into process and observation models.

$$\vec{x}_k \mid \vec{x}_{k-1} \sim \hat{\mathrm{f}}_{\text{model}}(\vec{x}_{0:k-1}) + \text{uncertainty}$$

$$\vec{z}_k \mid \vec{x}_k \sim \hat{\mathrm{f}}_{\text{measurement}}(\vec{x}_{0:k}) + \text{uncertainty}$$

- Example
  - Biased coin.
  - Fair Bernoulli prior

Bayesian Inference

Likelihood
(evidence)

Posterior

$$p(x \mid z) = \frac{p(z \mid x)\, p(x)}{p(z)}$$

Prior (*a priori* – before the evidence)

a posteriori probability (after the evidence)

normalization constant (independent of mouth)

We ***infer*** system state from uncertain observations and our prior knowledge (model) of system state.

Notation and BCI Example

Observations

$$\vec{Z}_k \text{ or } \vec{z}_{1:k} = (\vec{z}_1, \vec{z}_2, \ldots, \vec{z}_{k-1})$$

$$\vec{z}_k = \begin{bmatrix} z_{1,k} \\ z_{2,k} \\ \vdots \\ z_{n,k} \end{bmatrix}$$

e.g. firing rates of all n cells at time k

System State

$$\vec{X}_k \text{or } \vec{x}_{1:k} = (\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_k)$$

$$\vec{x}_k = \begin{bmatrix} x_k \\ y_k \\ v_{x,k} \\ v_{y,k} \\ a_{x,k} \\ a_{y,k} \end{bmatrix}$$

e.g. hand *kinematics* at time k

Generative Model

Encoding:

linear, non-linear?

$$\vec{z}_k = f_{obs}(\vec{x}_{1:k}) + \vec{q}_k$$

noise (e.g. Normal or Poisson)

$$\vec{x}_k = f_p(\vec{x}_{1:k-1}) + \vec{w}_k$$

neural firing rate

***f()'s* Markov?**

state (e.g. hand position, velocity, acceleration)

Today's Goal

Build a probabilistic model of this real process and with it estimate the *posterior* distribution $p(\mathrm{x}_k \mid \mathrm{z}_{1:k})$ so that we can infer the most likely state $\underset{x_k}{\operatorname{argmax}}\, p(x_k \mid z_{1:k})$ or the expected state $\int x_k p(x_k \mid z_{1:k})$

How ???

$$p(\mathrm{x}_{k-1} \mid \mathrm{z}_{1:k-1}) \quad \Rightarrow \quad p(\mathrm{x}_k \mid \mathrm{z}_{1:k})$$

Recursion!

- How can we formulate this recursion ?
- How can we compute this recursion ?
- What assumptions must we make ?

Modeling

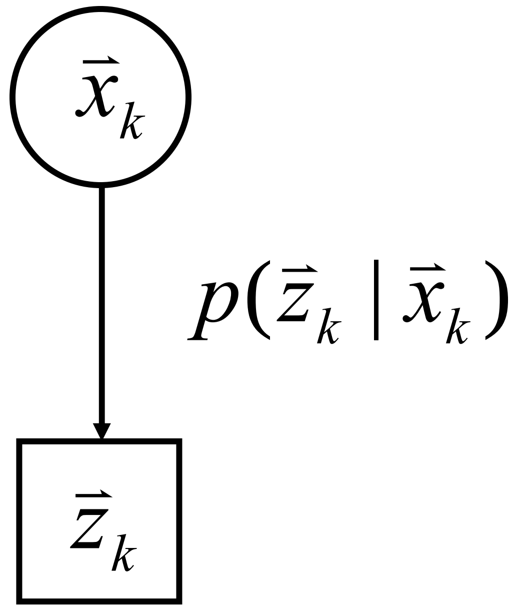
- An useful aside: Graphical models

Graphical models are a way of systematically diagramming the dependencies amongst groups of random variables. Graphical models can help elucidate assumptions and modeling choices that would otherwise be hard to visualize and understand.

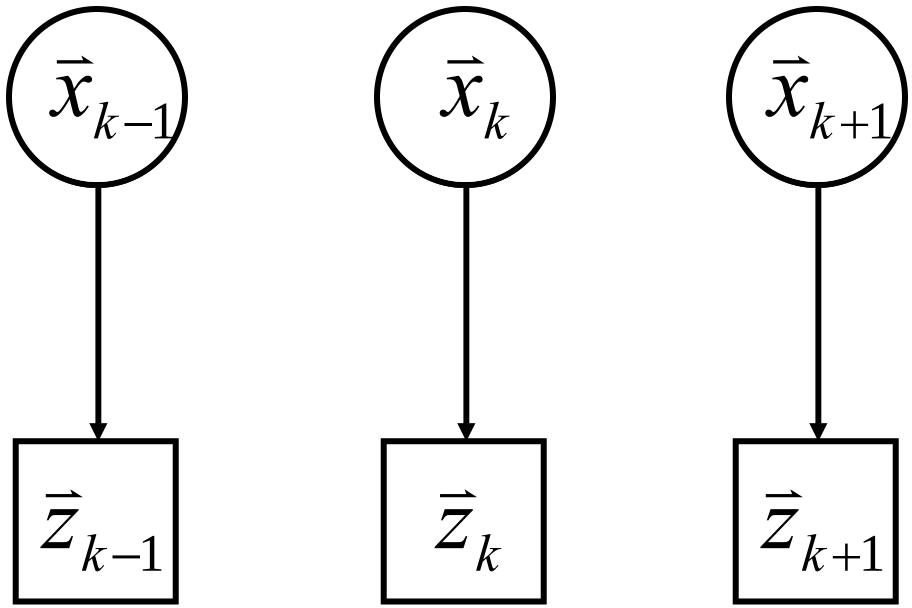
Using a graphical model will help us design our model!

Graphical Model

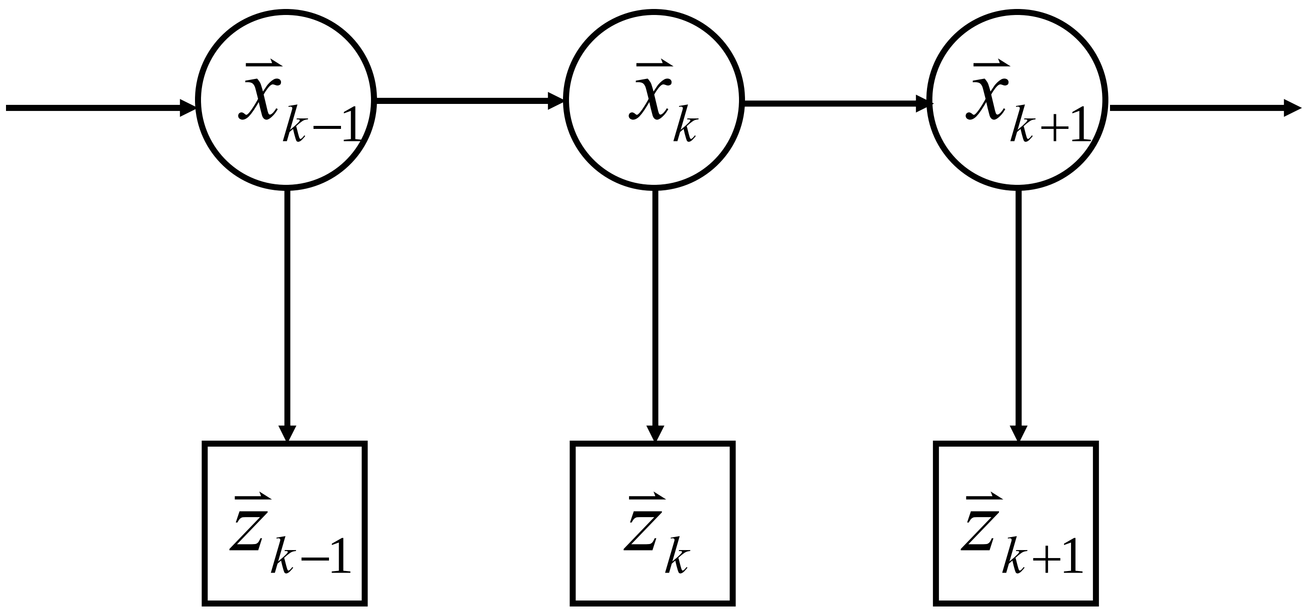
Generative model:



Graphical Model



Graphical Model



$$p(x_k \mid x_{k-1}, x_{k-2}, \ldots, x_1) = p(x_k \mid x_{k-1})$$

$$p(\vec{z}_k \mid \vec{z}_{1:k-1}, \vec{x}_k) = p(\vec{z}_k \mid \vec{x}_k) \qquad p(\vec{x}_k \mid \vec{z}_{1:k-1}, \vec{x}_{k-1}) = p(\vec{x}_k \mid \vec{x}_{k-1})$$

Summary

From these modeling choices all we have to choose is:

Encoding model

Likelihood model

$$p(\vec{z}_k \mid \vec{x}_k)$$

Temporal prior model

$$p(\vec{x}_k \mid \vec{x}_{k-1})$$

Initial distributions

$$p(\vec{x}_0), p(\vec{z}_0)$$

decoding

How to compute the posterior

$$p(\vec{x}_k \mid \vec{z}_{1:k}) = \kappa\, p(\vec{z}_k \mid \vec{x}_k) \int p(\vec{x}_k \mid \vec{x}_{k-1})\, p(\vec{x}_{k-1} \mid \vec{z}_{1:k-1})\, d\vec{x}_{k-1}$$

Linear Gaussian Generative Model

Observation Equation:

firing rate vector (zero mean, sqrt) $\begin{pmatrix} z_k^1 \\ z_k^2 \\ \vdots \\ z_k^{42} \end{pmatrix}$

system state vector (zero mean) $\begin{pmatrix} x_k \\ y_k \\ v_{x_k} \\ v_{y_k} \\ a_{x_k} \\ a_{y_k} \end{pmatrix}$

$$\vec{z}_k = H \vec{x}_k + \vec{q}_k$$

42 X 6 matrix

42 X 42 matrix

$$\vec{q}_k \sim N(0, Q)$$

$k = 0,1,2,\cdots$

System Equation:

$$\vec{x}_{k+1} = A \vec{x}_k + \vec{w}_k$$

6 X 6 matrix

6 X 6 matrix

$$\vec{w}_k \sim N(0, W)$$

$k = 0,1,2,\cdots$

Gaussian Assumption Clarified

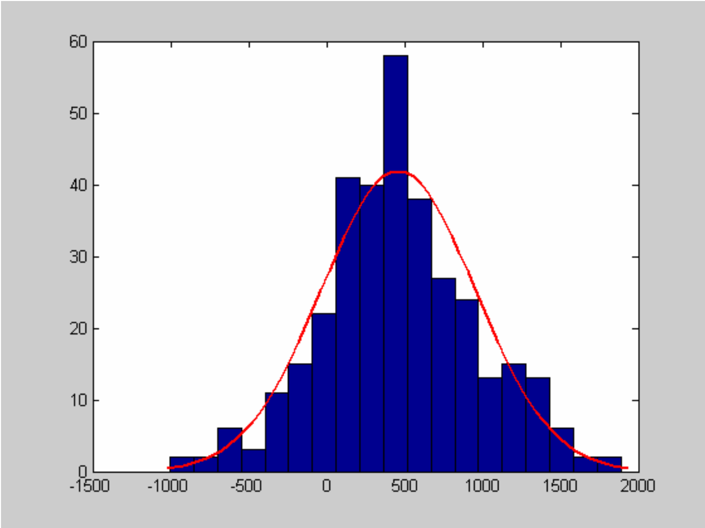
Gaussian distribution:

$$\vec{z}_k \sim N(H\vec{x}_k, Q)$$

$$\vec{z}_k - H\vec{x}_k = \vec{q}_k \sim N(0, Q)$$

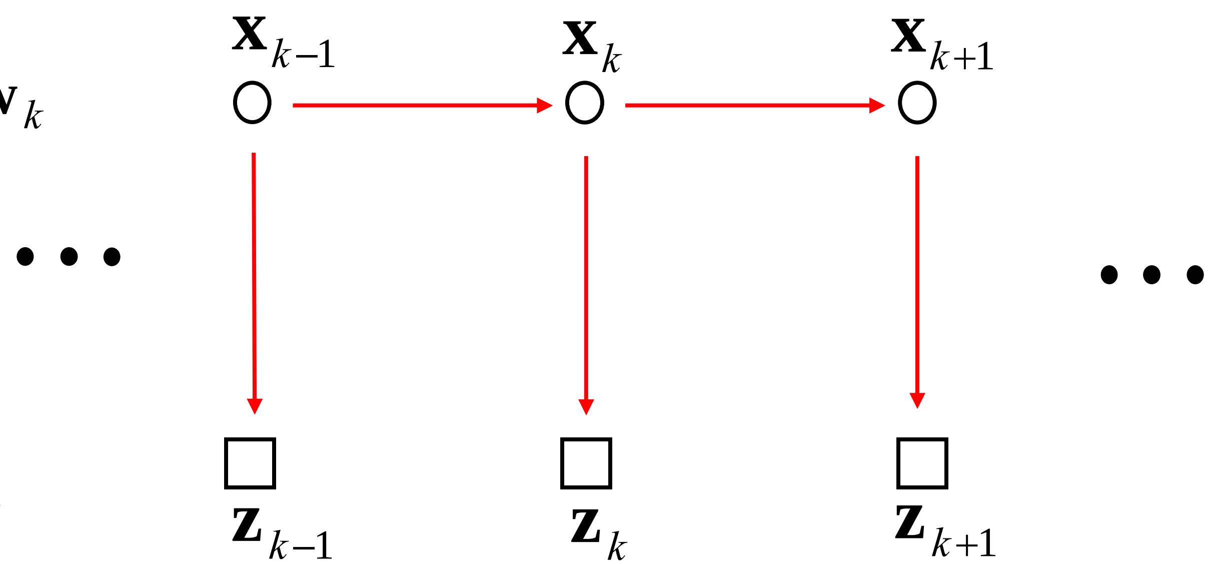
Recall:

$$p(x) = \frac{1}{\sqrt{2\pi\sigma}} \exp(-\frac{1}{2}(x-\mu)^2 / \sigma^2)$$



Graphical Model

$$\mathbf{x}_k = \mathbf{A}_k \mathbf{x}_{k-1} + \mathbf{w}_k$$



$$\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{q}_k$$

$$p(\mathbf{X}_M, \mathbf{Z}_M) = p(\mathbf{X}_M)\, p(\mathbf{Z}_M \mid \mathbf{X}_M)$$
$$= [p(\mathbf{x}_1) \prod_{k=2}^{M} p(\mathbf{x}_k | \mathbf{x}_{k-1})][\prod_{k=1}^{M} p(\mathbf{z}_k | \mathbf{x}_k)]$$

Break!

- When we come back quickly arrange yourselves in groups of 4 or 5. Uniformly distribute the applied math people and people who have taken CS143 (Vision) into these groups.
- Instead of straight-up lecture we are going to work through some derivations together to improve retention and facilitate understanding.
- If you don't have pencil and paper please get some.

Next step.

- Now we have a model – how do we do recursive Bayesian inference.
- I will present to you several relatively easy problems which I expect each group to solve in 5-10 minutes. When every group is finished I will select one group and ask for the person in that group who understood the problem the least to explain the solution to the class. The group is responsible for nominating this person and his or her ability to explain the solution.

Recursive Bayesian Inference

Likelihood (evidence)

Prior (*a priori* – before the evidence)

Posterior

$$p(\text{kinematics}_t \mid \text{firing}_{1:t}) = \frac{p(\text{firing}_t \mid \text{kinematics}_t)\, p(\text{kinematics}_t)}{p(\text{firing}_{1:t})}$$

a posteriori probability (after the evidence)

normalization constant (independent of kinematics)

We sequentially ***infer*** hand kinematics from uncertain evidence and our prior knowledge of how hands move.

Recursive Bayesian Estimation

- Update Stage
  - From the prediction stage you have a prior distribution over the system state at the current time k. After observing the process at time k you can update the posterior to reflect that new information.
- Prediction Stage
  - Given the posterior from a previous update stage and your system model you produce the next prior distribution.

Update Stage

$$p(\mathrm{x}_k | \mathrm{z}_{1:k})$$

Posterior

$$= \frac{p(\mathrm{x}_k, \mathrm{z}_{1:k})}{p(\mathrm{z}_{1:k})}$$

Bayes Rule

$$= \frac{p(z_k | \mathrm{x}_k, z_{1:k-1})\, p(\mathrm{x}_k, \mathrm{z}_{1:k-1})}{p(\mathrm{z}_k | \mathrm{z}_{1:k-1})\, p(\mathrm{z}_{1:k-1})}$$

Bayes Rule Again

Independence

$$= \frac{p(z_k | \mathrm{x}_k)\, p(\mathrm{x}_k | \mathrm{z}_{1:k-1})\, \cancel{p(\mathrm{z}_{1:k-1})}}{p(\mathrm{z}_k | \mathrm{z}_{1:k-1})\, \cancel{p(\mathrm{z}_{1:k-1})}}$$

Cancels

New Observation

$$= \frac{p(\boxed{z_k} | \mathrm{x}_k)\, \boxed{p(\mathrm{x}_k | \mathrm{z}_{1:k-1})}}{p(\mathrm{z}_k | \mathrm{z}_{1:k-1})}$$

Prior

Prediction Stage

$$\boxed{p(x_k | z_{1:k-1})} = \int p(x_k | x_{k-1}, z_{1:k-1})\, p(x_{k-1} | z_{1:k-1}) \partial x_{k-1}$$

$$= \int p(x_k | x_{k-1})\, \boxed{p(x_{k-1} | z_{1:k-1})} \partial x_{k-1}$$

Prior

Law of Total Probability

Independence

Posterior

$$\Pr(B \mid C) = \int \Pr(B \mid A, C) \Pr(A \mid C) \delta A$$

Phew!

- Let's drill this into our heads and actually run the Bayesian recursion to see how it starts and behaves.

The Bayesian Recursion

given $\mathrm{P}(\vec{x}_0)$ and $\mathrm{P}(\vec{z}_0)$ and the model $\mathrm{P}(\vec{z}_k | \vec{x}_k)$, $\mathrm{P}(\vec{x}_k | \vec{x}_{k-1})$

Run the Bayesian recursion to depth 2.

$$\mathrm{P}(\vec{x}_0 \mid \vec{z}_0) =$$

$$\mathrm{P}(\vec{x}_1 \mid \vec{z}_0) =$$

$$\mathrm{P}(\vec{x}_1 \mid \vec{z}_0, \vec{z}_1) =$$

$$\mathrm{P}(\vec{x}_2 \mid \vec{z}_0, \vec{z}_1) =$$

$$\mathrm{P}(\vec{x}_2 \mid \vec{z}_0, \vec{z}_1, \vec{z}_2) =$$

$$\vdots$$

Highlighting the Assumptions

$$p(\mathrm{x}_k | \mathrm{z}_{1:k})$$

$$= p(\mathrm{x}_k | \mathrm{z}_{1:k-1}, \mathrm{z}_k)$$

$$\propto p(\mathrm{z}_k | \mathrm{x}_k, \mathrm{z}_{1:k-1}) p(\mathrm{x}_k | \mathrm{z}_{1:k-1})$$

$$\propto p(\mathrm{z}_k | \mathrm{x}_k) p(\mathrm{x}_k | \mathrm{z}_{1:k-1})$$

$$\propto p(\mathrm{z}_k | \mathrm{x}_k) \int p(\mathrm{x}_k | \mathrm{x}_{k-1}, \mathrm{z}_{1:k-1}) p(\mathrm{x}_{k-1} | \mathrm{z}_{1:k-1}) \delta \mathrm{x}_{k-1}$$

$$\propto p(\mathrm{z}_k | \mathrm{x}_k) \int p(\mathrm{x}_k | \mathrm{x}_{k-1}) p(\mathrm{x}_{k-1} | \mathrm{z}_{1:k-1}) d\mathrm{x}_{k-1}$$

What's missing?

Bayes rule:
$p(a \mid b) = p(b \mid a) p(b) / p(a)$

Independence assumption:
$p(\mathbf{z}_k \mid \mathbf{x}_k, \mathbf{Z}_{k-1}) = p(\mathbf{z}_k \mid \mathbf{x}_k)$

Independence assumption:
$p(\mathrm{x}_k \mid \mathrm{x}_{k-1}, \mathrm{z}_{1:k-1}) = p(\mathrm{x}_k \mid \mathrm{x}_{k-1})$

Law of Total Probability:
$p(a \mid c) = \int p(a \mid b, c) p(b \mid c) db$

Bayesian Formulation

$$\mathrm{p}(x_k | z_{0:k}) = \kappa \; \mathrm{p}(z_k | x_k) \int \mathrm{p}(x_k | x_{k-1}) \mathrm{p}(x_{k-1} | z_{0:k-1}) dx_{k-1}$$

$\mathrm{p}(z_k|x_k)$: *likelihood*

$\mathrm{p}(x_k|x_{k-1})$: *temporal prior*

$\mathrm{p}(x_{k-1}|z_{1:k-1})$: posterior probability at previous time step

κ : normalizing term

General Model

- $p(\mathrm{x}_k \mid \mathrm{z}_{0:k})$ can be an arbitrary, non-Gaussian, multi-modal distribution.
- The recursive equation may have no explicit solution, but can usually be approximated numerically using Monte Carlo techniques such as **particle filtering**.
- However, if both the *likelihood* and *prior* are linear Gaussian, then the recursive equation has a closed form solution. This model, which we'll see next week, is known as the Kalman filter. **(Kalman, 1960)**

Particle Filtering

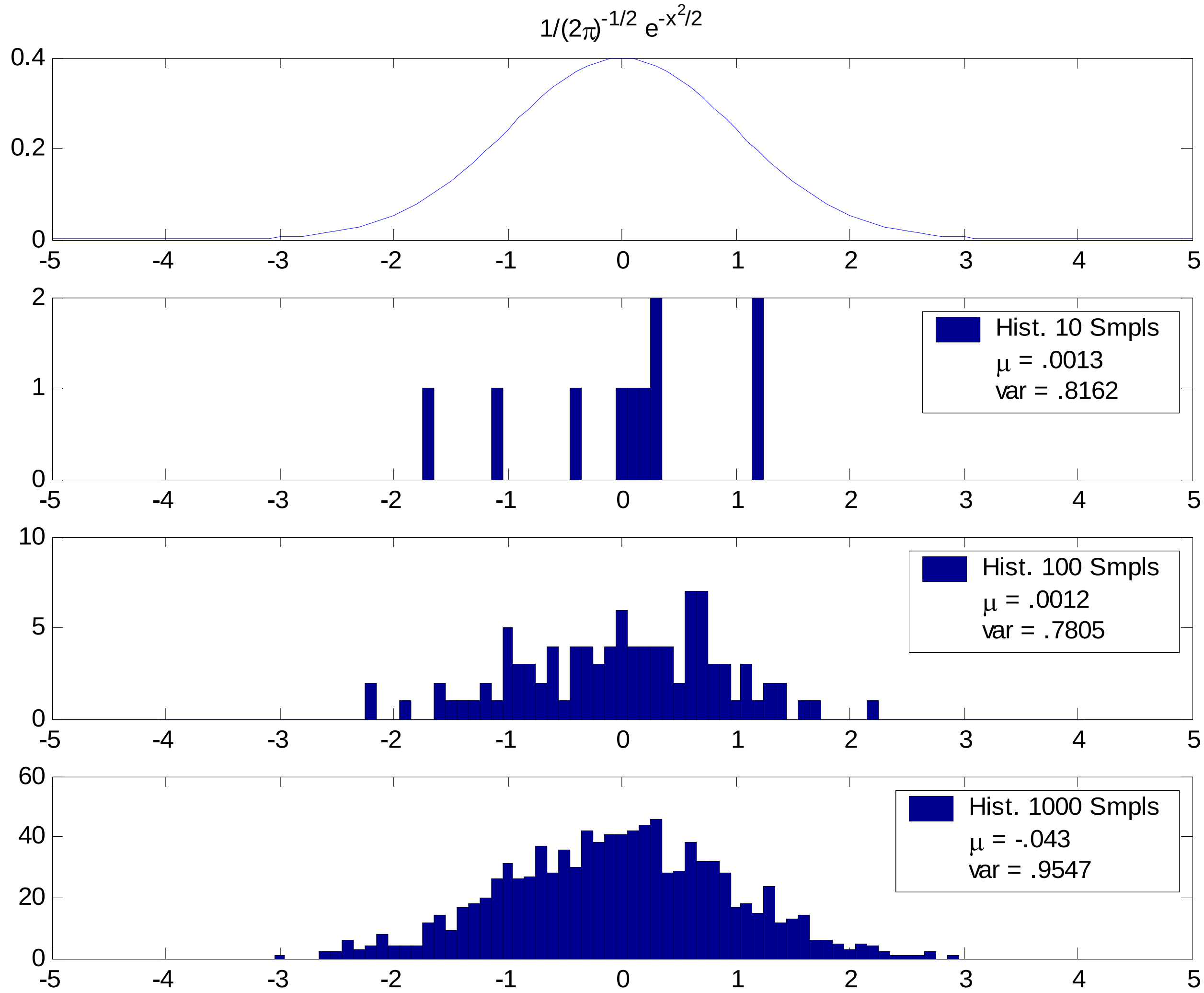
- "A technique for implementing a recursive Bayesian filter by Monte Carlo simulations" Arulampalam et. al.
- A set of samples (particles) and weights that represent the posterior distribution (a Random Measure) is maintained throughout the algorithm.
- It boils down to sampling, density representation by samples, and Monte Carlo integration.

Sampling

- Uniform
  - rand() Linear Congruential Generator
    - x(n) = a * x(n-1) + b mod M

      0.2311 0.6068 0.4860 0.8913 0.7621 0.4565 0.0185
- Normal
  - randn() Box-Mueller
    - x1,x2 ~ U(0,1) -> y1,y2 ~N(0,1)
      - y1 = sqrt(- 2 ln(x1)) cos(2 pi x2)
      - y2 = sqrt(- 2 ln(x1)) sin(2 pi x2)
- Binomial(p)
  - if(rand()<p)
- Higher Dimensional Distributions
  - Metropolis Hastings / Gibbs

Distribution Representation by Samples



Law of Large Numbers

- If we have *n* fair samples drawn from a distribution *P*

$$x_{1:n} \sim P(X)$$

- Then one version of the Law of Large Numbers says that the emperical average of a the value of a function over the samples converges to the expected value of the function as the number of samples grows.

$$\frac{1}{n}\sum_{i=1}^{n} f(x_i) \underset{n\to\infty}{\longrightarrow} \mathrm{E}_P\left[f(X)\right]$$

Why Does this Matter

- Particle Filtering represents distributions by samples.
- We need to either maximize or take an expected value of the posterior (both functions) with the posterior represented by samples.
- We need to sample from distributions to simulate trajectories, etc.

Monte Carlo Integration

Weight *Particle/Sample*

$$P_N(\partial x_k | z_{0:k}) = \frac{1}{N}\sum_{i=1}^{N} \boxed{w_k^i}\,\delta_{\boxed{x_k^i}}(\partial x_k)$$ *Given*

$$p(\mathrm{x}_k | \mathrm{z}_{0:k-1}) = \int p(\mathrm{x}_k | \mathrm{x}_{k-1})\, p(\mathrm{x}_{k-1} | \mathrm{z}_{0:k-1}) \partial \mathrm{x}_{k-1}$$ *Evaluate*

$$= \int p(\mathrm{x}_k | \mathrm{x}_{k-1}) \frac{1}{N}\sum_{i=1}^{N} w_{k-1}^i \delta_{x_{k-1}^i}(\partial x_{k-1}) \partial \mathrm{x}_{k-1}$$

$$= \frac{1}{N}\sum_{i=1}^{N} w_{k-1}^i \int p(\mathrm{x}_k | \mathrm{x}_{k-1}) \delta_{x_{k-1}^i}(\partial x_{k-1}) \partial \mathrm{x}_{k-1}$$

$$= \frac{1}{N}\sum_{i=1}^{N} w_{k-1}^i\, p(\mathrm{x}_k | x_{k-1}^i)$$

The Particle Filtering Story

- A bit hand wavy
  - Simplified from Arulampalam et al and DeFreitas et al
  - Largely overlooks the importance weights are maintained and updated
  - Doesn’t touch particle degeneration and replacement
- Posterior Representation by Samples
- Importance Sampling
- Weights in Importance Sampling
- Sampling from the Prediction Distribution
- Simple Particle Filter

Posterior Representation by Samples

- We use a set of random samples from the posterior distribution to represent the posterior. The we can use sample statistics to approximate expectations over the posterior.

Let $\mathcal{S} = \{\vec{\mathbf{x}}^{(j)}\}$ be a set of N fair samples from distribution $\mathcal{P}(\vec{\mathbf{x}})$, then for functions $f(\vec{\mathbf{x}})$

$$E_{\mathcal{S}}[f(\vec{\mathbf{x}})] \equiv \frac{1}{N}\sum_{j=1}^{N} f(\vec{\mathbf{x}}^{(j)}) \xrightarrow{N\to\infty} E_{\mathcal{P}}[f(\vec{\mathbf{x}})]$$

Problem: we need to update the samples such that they still accurately represent the posterior after the next observation.

Importance Sampling

Assume we have a weighted sample set $S_{k-1} = \left\{ x_{k-1}^{(i)}, z_{k-1}^{(i)} \right\} \text{for } 1 < i < N$

- the prediction distribution becomes a linear mixture model

From Monte Carlo integration over posterior at time k.

$$\mathrm{p}(x_k | z_{0:k-1}) = \frac{1}{N} \sum_{i=1}^{N} w_{k-1}^{i} \, \underline{\mathrm{p}(x_k | x_{k-1}^{i})}$$

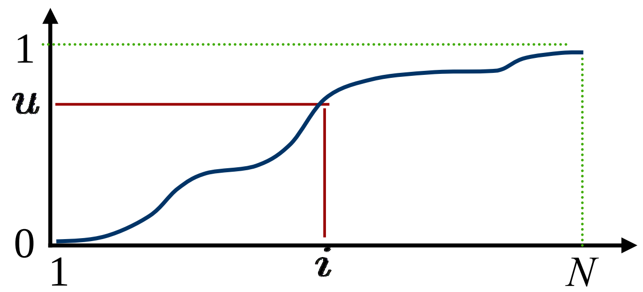
Importance weights

Specified as part of the model.

- Can sample from this mixture model by treating the weights as mixing probabilities

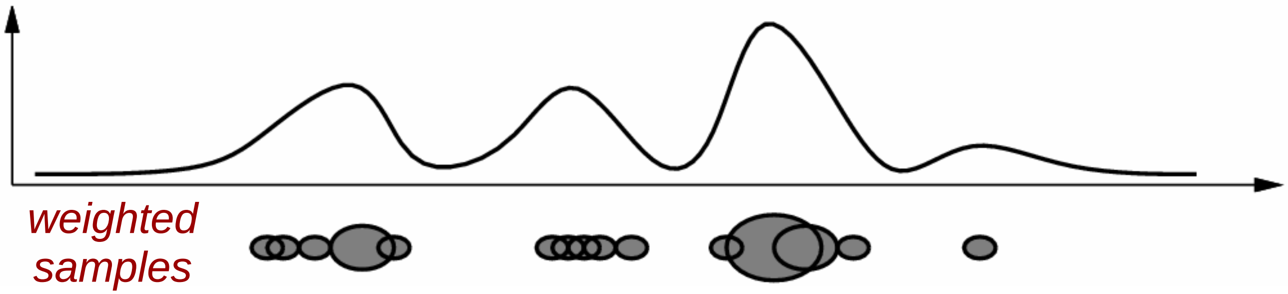
Cumulative distribution of weights

sample
$u \sim U(0, 1)$



and then sampling from the model proposal pdf $\mathrm{p}(x_k | x_{k-1}^{i})$

Weights in Importance Sampling



Weighted samples $\mathcal{S} = \{\vec{\mathbf{x}}^{(j)}, w^{(j)}\}$

- Draw samples $\vec{\mathbf{x}}^{(j)}$ from a *proposal distribution* $\mathcal{Q}(\vec{\mathbf{x}})$
- Find weights $w^{(j)}$ so that the linearly weighted sample statistics approximate expectations under the desired distribution $\mathcal{P}(\vec{\mathbf{x}})$

$$E_{\mathcal{S}}[f(\vec{\mathbf{x}})] \equiv \sum_{j=1}^{N} w^{(j)} f(\vec{\mathbf{x}}^{(j)}) \xrightarrow{N\to\infty} E_{\mathcal{Q}}[w(\vec{\mathbf{x}})\, f(\vec{\mathbf{x}})]$$

$$= \int w(\vec{\mathbf{x}})\, f(\vec{\mathbf{x}})\, \mathcal{Q}(\vec{\mathbf{x}})\, d\vec{\mathbf{x}}$$

$$= \int f(\vec{\mathbf{x}})\, \mathcal{P}(\vec{\mathbf{x}})\, d\vec{\mathbf{x}}$$

$$= E_{\mathcal{P}}[f(\vec{\mathbf{x}})]$$

i.e. $w(\vec{\mathbf{x}}) = \dfrac{\mathcal{P}(\vec{\mathbf{x}})}{\mathcal{Q}(\vec{\mathbf{x}})}$

Updating the Weights

The samples from the prediction distribution need to be re-weighted such that they still represent the posterior distribution well after a new observation:

$$p(x_k | z_{1:k}) = \frac{p(z_k | x_k)\, p(x_k | z_{0:k-1})}{p(z_k | z_{0:k-1})}$$

We have a sample representation for this.

This is our model likelihood

Goes away after re-normalization.

Major hand waving here! Inaccuracies abound!

$$= k\, l(z_k | x_k^j) \frac{1}{N} \sum_{i=1}^{N} w_{k-1}^j x_k^j$$

$$\Rightarrow w_k^j = k\, l(z_k | x_k^j) w_{k-1}^j$$

An algorithmic run-through

Simple particle filter:

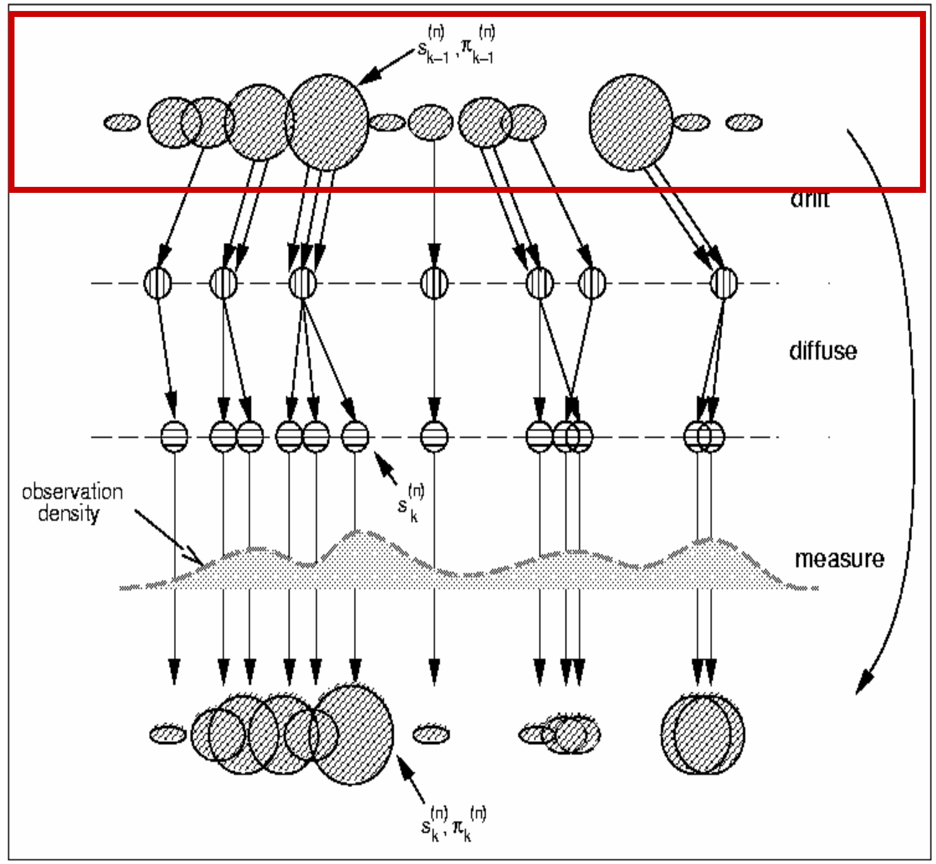
- draw samples from the prediction distribution $p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{z}}_{1:t-1})$
- weights are proportional to the ratio of posterior and prediction distributions, i.e. the normalized likelihood $c\, p(\vec{\mathbf{z}}_t \,|\, \vec{\mathbf{x}}_t)$

$$\underset{\text{posterior}}{p(\vec{\mathbf{x}}_{t-1} \,|\, \vec{\mathbf{z}}_{1:t-1})} \xrightarrow{\text{sample}} \underset{\text{temporal dynamics}}{p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{x}}_{t-1})} \xrightarrow{\text{sample}} \underset{\text{likelihood}}{p(\vec{\mathbf{z}}_t \,|\, \vec{\mathbf{x}}_t)} \xrightarrow{\text{normalize}} \underset{\text{posterior}}{p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{z}}_{1:t})}$$

[Gordon et al ’93; Isard & Blake ’98; Liu & Chen ’98, …]

Particle Filter

Posterior $p(\vec{x}_{k-1} \mid \vec{z}_{k-1})$

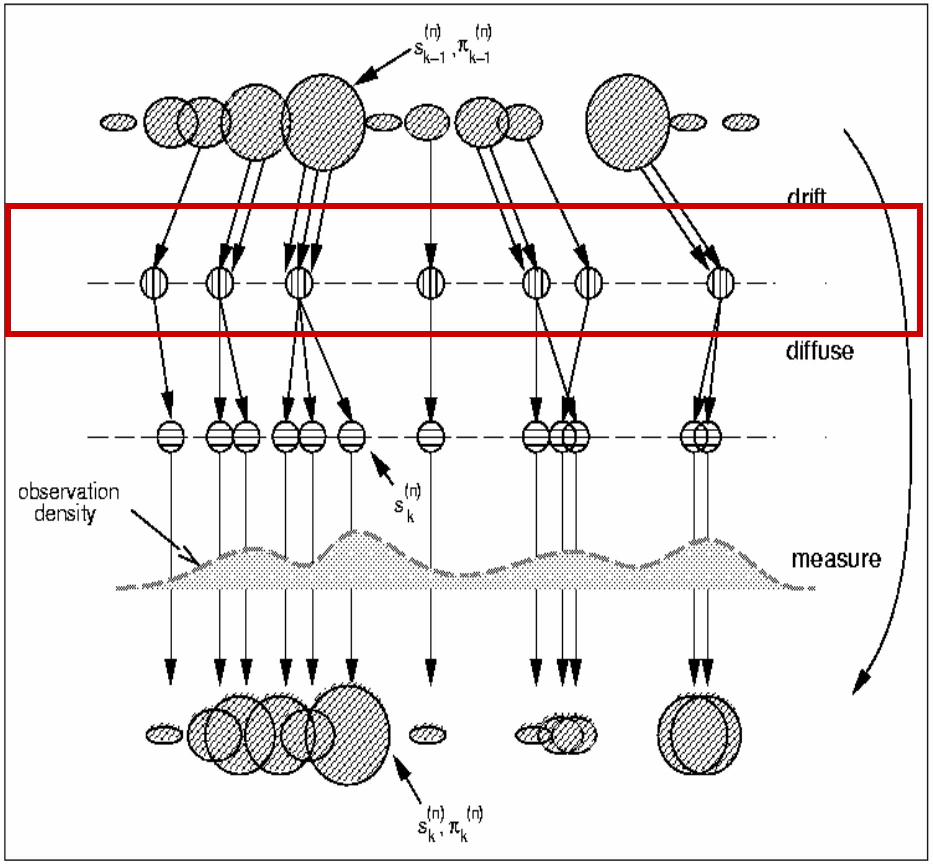


Isard & Blake ‘96

Particle Filter

Posterior $p(\vec{x}_{k-1} \mid \vec{z}_{1:k-1})$

sample



Isard & Blake '96

Particle Filter

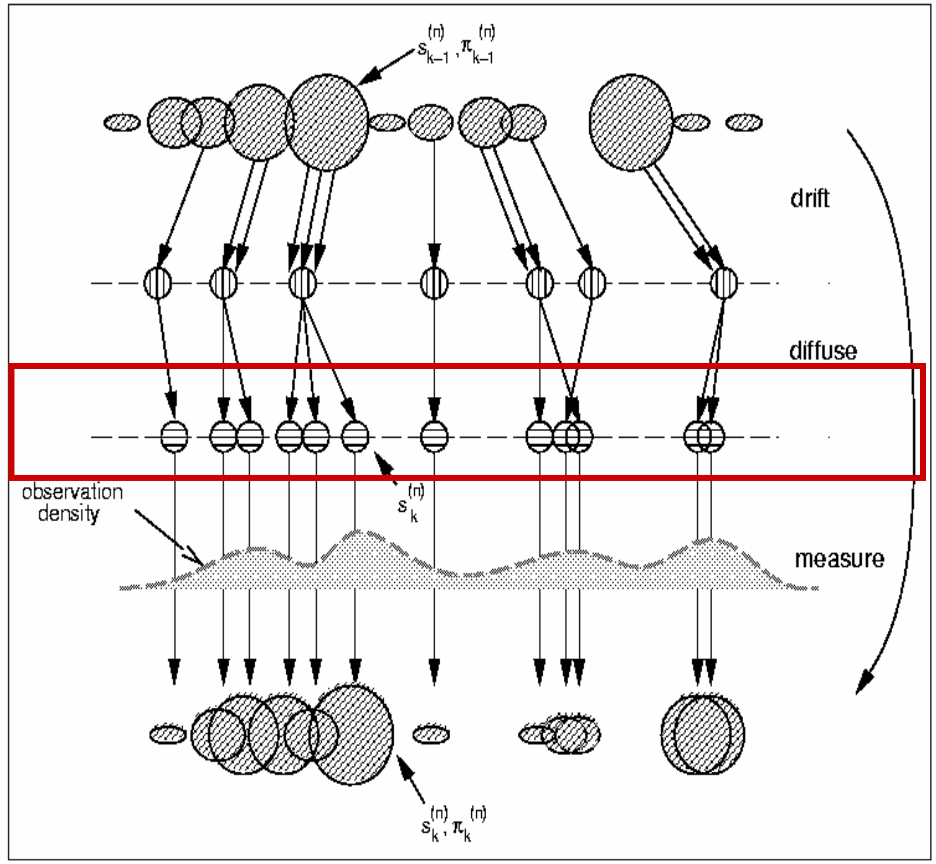
Posterior $p(\vec{x}_{k-1} | \vec{z}_{1:k-1})$

sample

Temporal dynamics

$p(x_k \mid x_{k-1})$

sample



Isard & Blake ‘96

Particle Filter

Posterior $p(\vec{x}_{k-1} \mid \vec{z}_{1:k-1})$

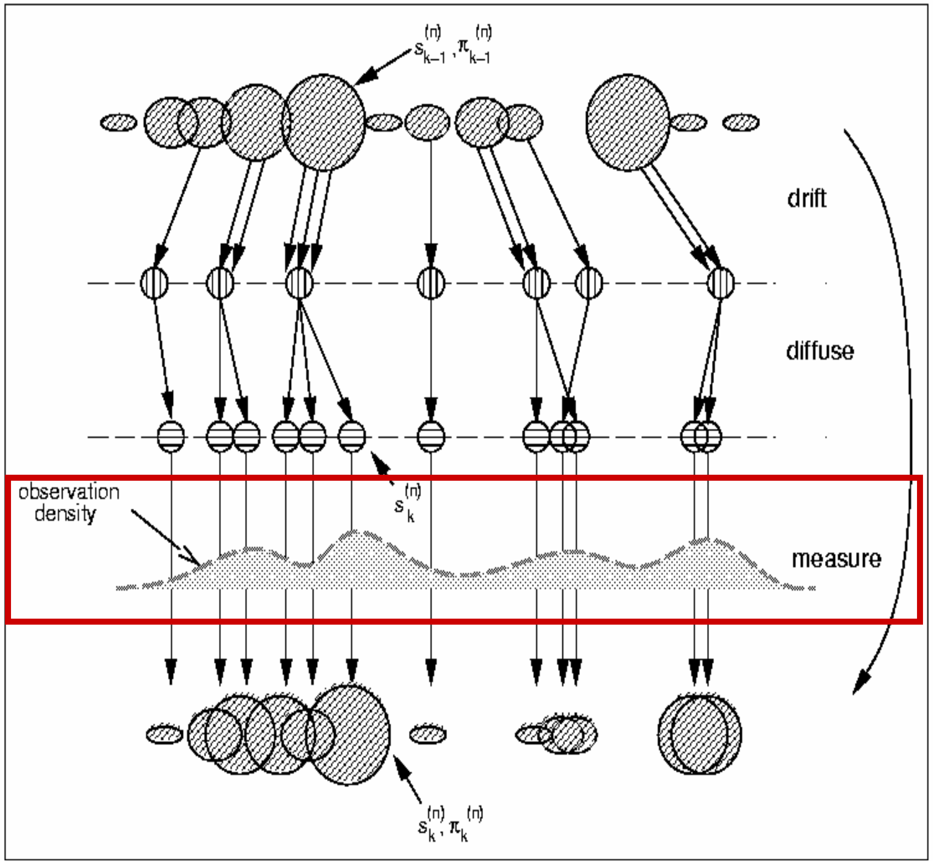
sample

Temporal dynamics

$p(x_k \mid x_{k-1})$

sample

Likelihood $p(\vec{z}_k \mid \vec{x}_k)$



Isard & Blake '96

Particle Filter

Posterior $p(\vec{x}_{k-1} \mid \vec{z}_{1:k-1})$

sample

Temporal dynamics

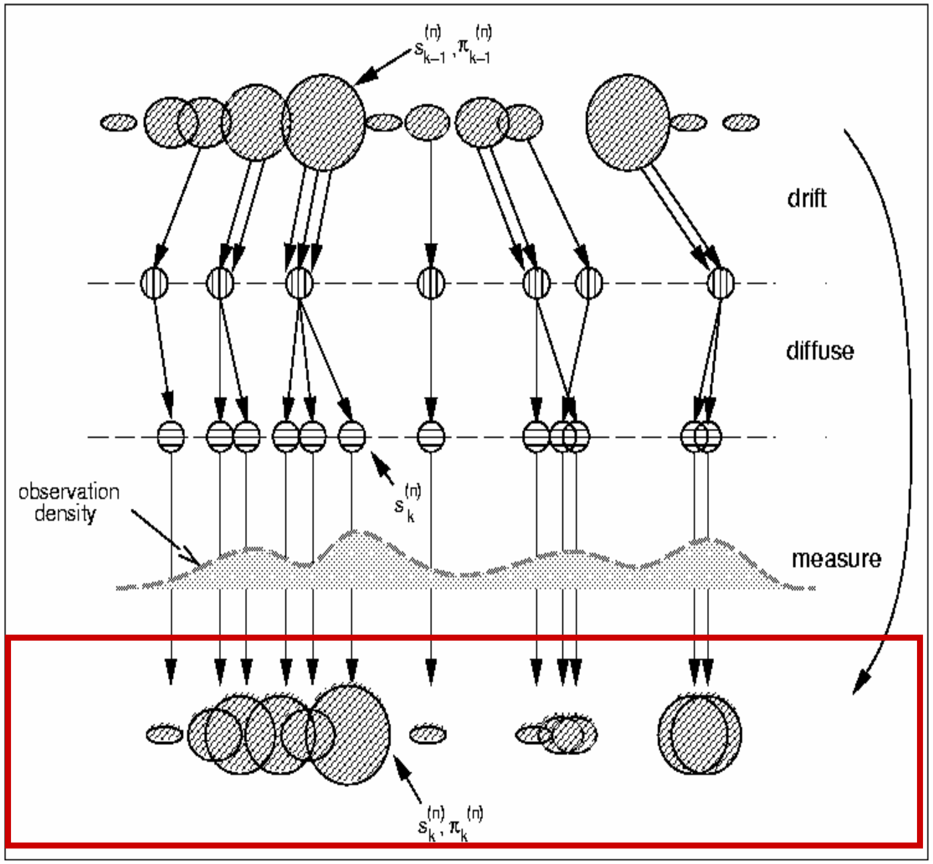
$p(x_k \mid x_{k-1})$

sample

Likelihood $p(\vec{z}_k \mid \vec{x}_k)$

normalize

Posterior $p(\vec{x}_k \mid \vec{z}_{1:k})$



Isard & Blake '96

Sampling

- Many sampling steps in particle filtering.
  - Samping from a Gaussian.
  - Sampling from a multinomial.
  - Sampling from a weighted mixture model.
- More general sampling techniques that we may get to later.
  - Metropolis-Hastings
  - Gibbs

Sampling from a Gaussian

Given

$$N(\mu, Q)$$

$$LL^T = Q$$

$$\vec{\tau} = [\tau_0, \tau_1, \tau_2 \ldots \tau_n]^T$$

Show

$$L\vec{\tau} + \vec{\mu} \sim N(\vec{\mu}, Q)$$

$$\mathrm{E}[L\tau] = L\,\mathrm{E}[\tau] = 0$$

$$\begin{aligned}\mathrm{Var}[L\vec{\tau}] &= \mathrm{E}[L\vec{\tau}\vec{\tau}^T L^T] \\ &= L\,\mathrm{E}[\vec{\tau}\vec{\tau}^T]L^T \\ &= Q\end{aligned}$$

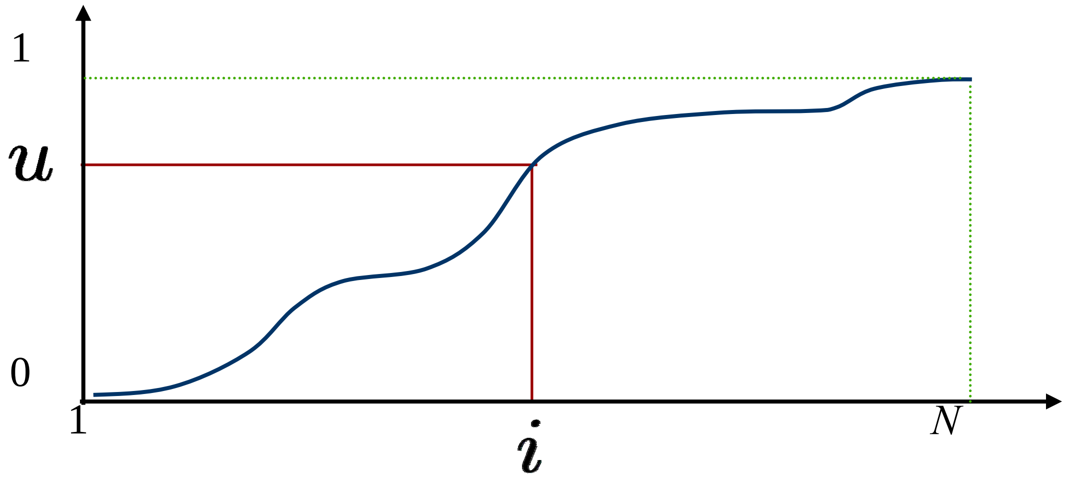
And explain how this fact can be used to sample from a Gaussian.

Sampling from a Multinomial

Given a weighted sample set $S = \{(\mathbf{x}^{(i)}, w^{(i)});\ i = 1 \ldots N\}$

Cumulative distribution of weights

sample
$u \sim U(0,1)$



Bayesian Inference

Infer (decode) behavior from firing.

p(behavior at k | firing up to k) =

$$p(\vec{x}_k \mid \vec{Z}_k) = \kappa \, p(\vec{z}_k \mid \vec{x}_k) \, p(\vec{x}_k \mid \vec{Z}_{k-1})$$

likelihood

prior

observation model

$$\vec{z}_k \sim \mathcal{N}(H \, \vec{x}_k, \, Q)$$

$$p(\vec{x}_k \mid \vec{Z}_{k-1}) = \int p(\vec{x}_k \mid \vec{x}_{k-1}) \, p(\vec{x}_{k-1} \mid \vec{Z}_{k-1}) \, d\vec{x}_{k-1}$$

Kalman Filter

$$\vec{x}_k \sim \mathcal{N}(A\vec{x}_{k-1}, W) \qquad \mathcal{N}(\hat{x}_{k-1}, P_{k-1})$$

system model

Kalman Filter

Likelihood

$$p(\vec{z}_{t-j} \mid \vec{x}_t)$$

observation model:

$$\vec{z}_t \sim \mathcal{N}(H_t \vec{x}_t, Q_t)$$

Temporal prior

$$p(\vec{x}_t \mid \vec{x}_{t-1})$$

system model:

$$\vec{x}_t \sim \mathcal{N}(A_t \vec{x}_{t-1}, W_t)$$

Posterior is also Gaussian

$$p(\vec{x}_t \mid \vec{Z}_t) = \kappa\, p(\vec{z}_t \mid \vec{x}_t) \int p(\vec{x}_t \mid \vec{x}_{t-1})\, p(\vec{x}_{t-1} \mid \vec{Z}_{t-1})\, d\vec{x}_{t-1}$$

Kalman filter.

Real-time, recursive, decoding.

KALMAN FILTER ALGORITHM

Time Update

Prior estimate

$$\hat{x}_k^- = A\hat{x}_{k-1}$$

Error covariance

$$P_k^- = AP_{k-1}A^T + W$$

Measurement Update

Posterior estimate

$$\hat{x}_k = \hat{x}_k^- + K_k(\vec{z}_k - H\hat{x}_k^-)$$

Error covariance

$$P_k = (I - K_k H)P_k^-$$

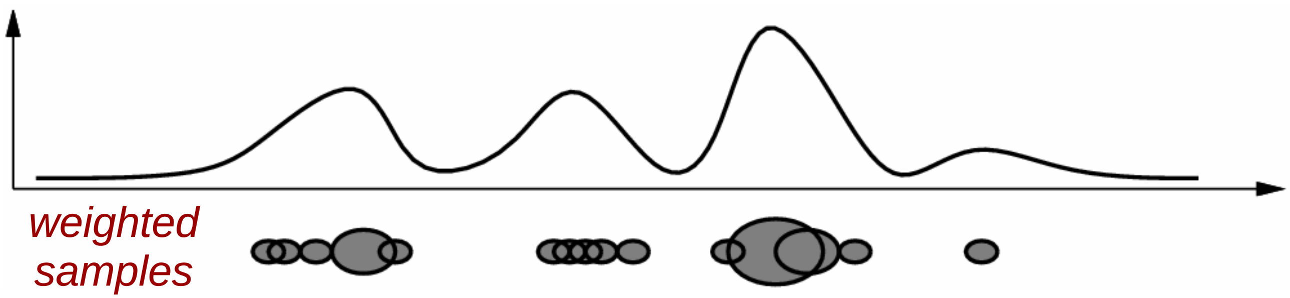
Kalman gain

$$K_k = P_k^- H^T (HP_k^- H^T + Q)^{-1}$$

Initial estimate of $\hat{x}_{k-1}$ and P_{k-1}

Welch and Bishop 2002

Factored Sampling



Weighted samples $S = \{(\mathbf{x}^{(i)}, w^{(i)}); i = 1 \ldots N\}$

Normalized likelihood:

$$w_t^{(n)} = \frac{p(\mathbf{I}_t | \mathbf{x}_t^{(n)})}{\sum_{i=1}^{N} p(\mathbf{I}_t | \mathbf{x}_t^{(i)})}$$