Probabilistic Robotics Course

Particle Distributions Particle Filters

Giorgio Grisetti
grisetti@dis.uniroma1.it

Dept of Computer Control and Management Engineering
Sapienza University of Rome

Sampling from a Distribution

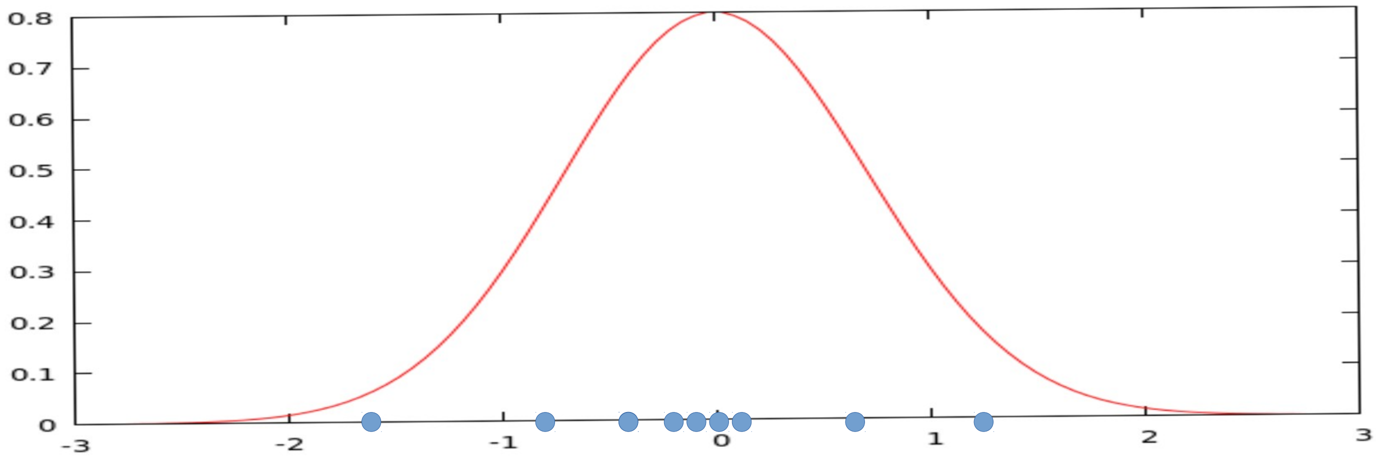
Sampling means generating a set of samples, given we know a distribution

$$x^{(i)} \sim p(x)$$

Most of the random number generators produce samples in from the uniform

$$y^{(i)} \sim U(0,1)$$

How can we generate samples from *p(x)?*



Generating Samples

We assume to have p(x) in closed form

$$P(x) = \int_{-\infty}^{x} p(x')dx'$$

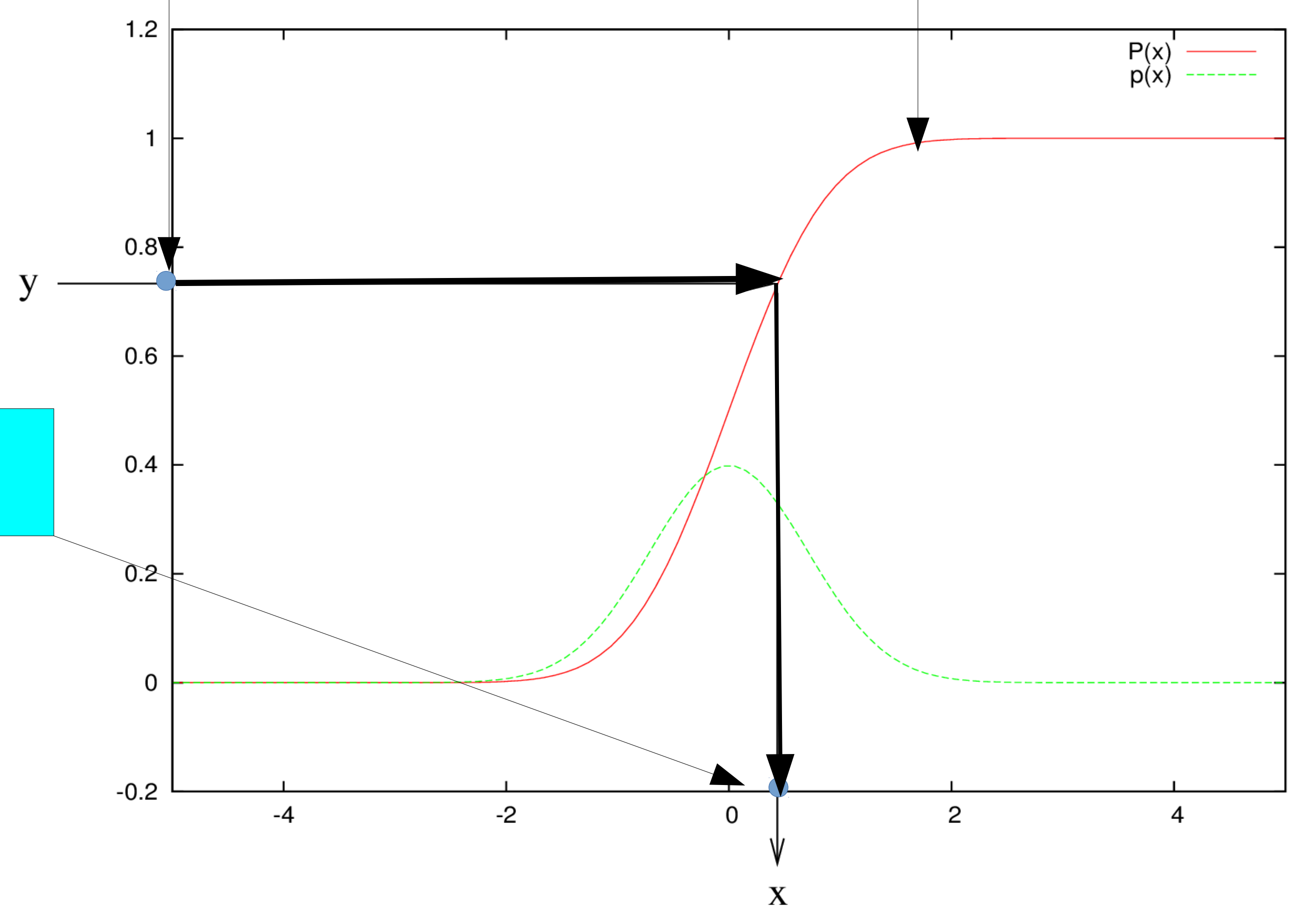
1.compute the cumulative distribution

$$y^{(i)} \sim U(0,1)$$

2. draw a sample from U(0,1)

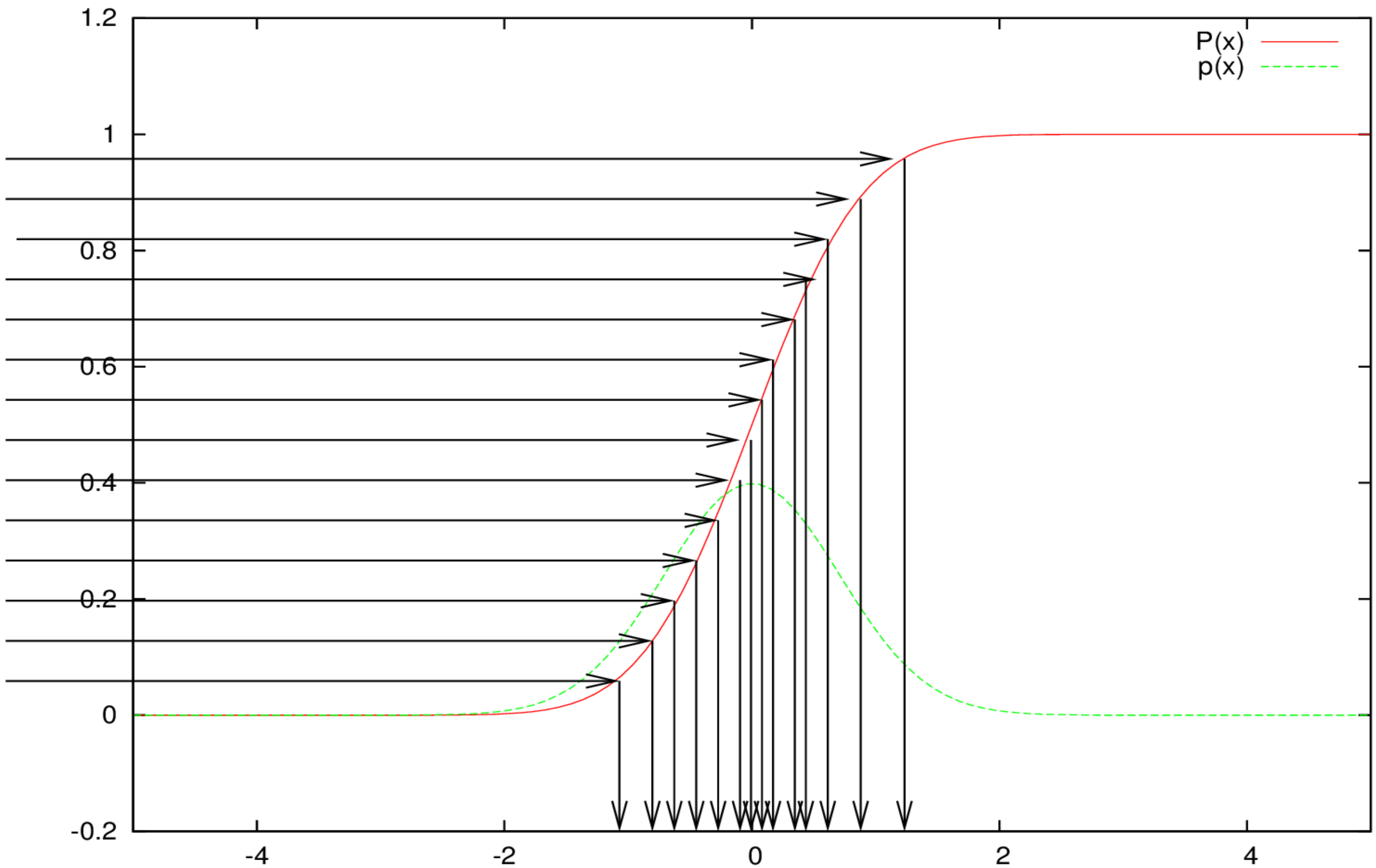
$$x^{(i)} = P^{-1}(y^{(i)})$$

3. compute the inverse of the cumulative at $y^{(i)}$



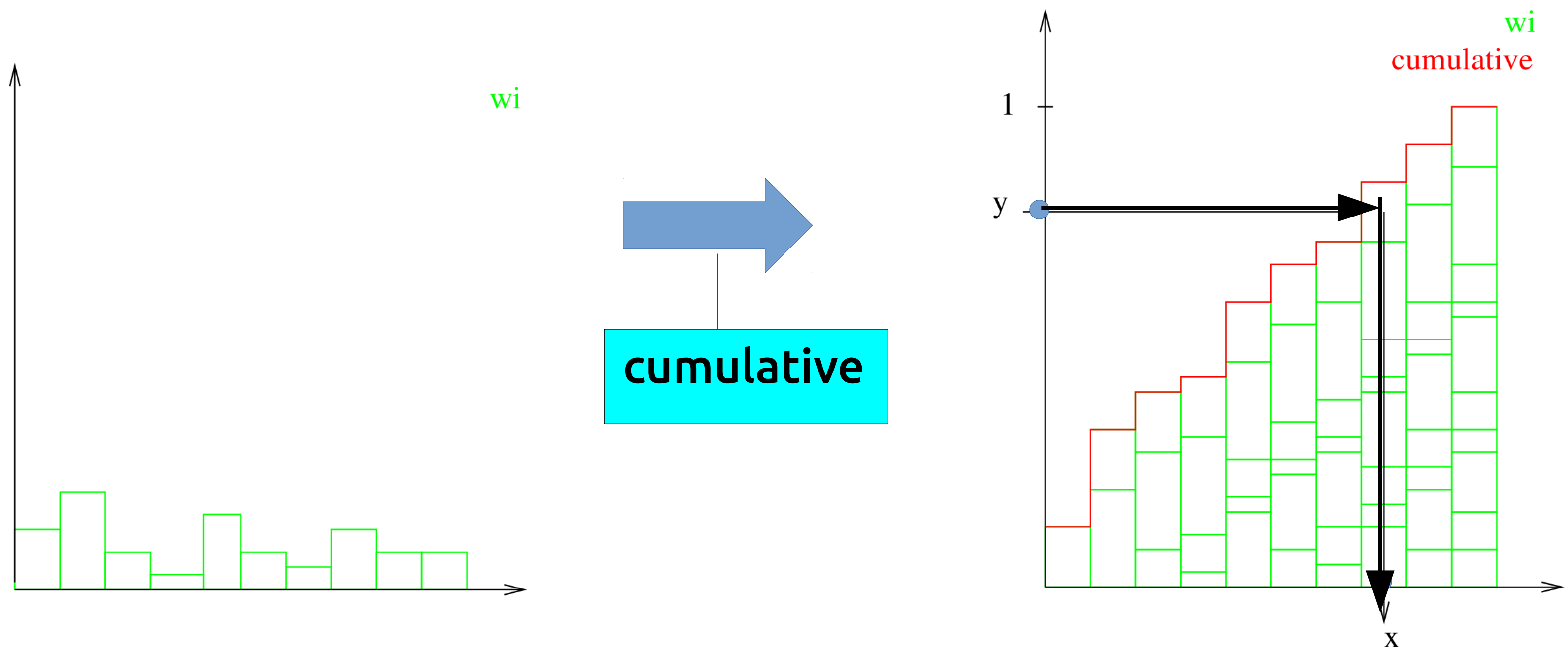
Generating Samples

Iterating this process generates denser samples where *p(x)* is higher



Discrete Case

If the distribution is discrete, we can do a similar process. The cumulative will look like a stair with uneven steps



Uniform Sampling

We will encounter the task of generating N samples from a discrete distribution.

Calling the random number generator N times might be expensive.

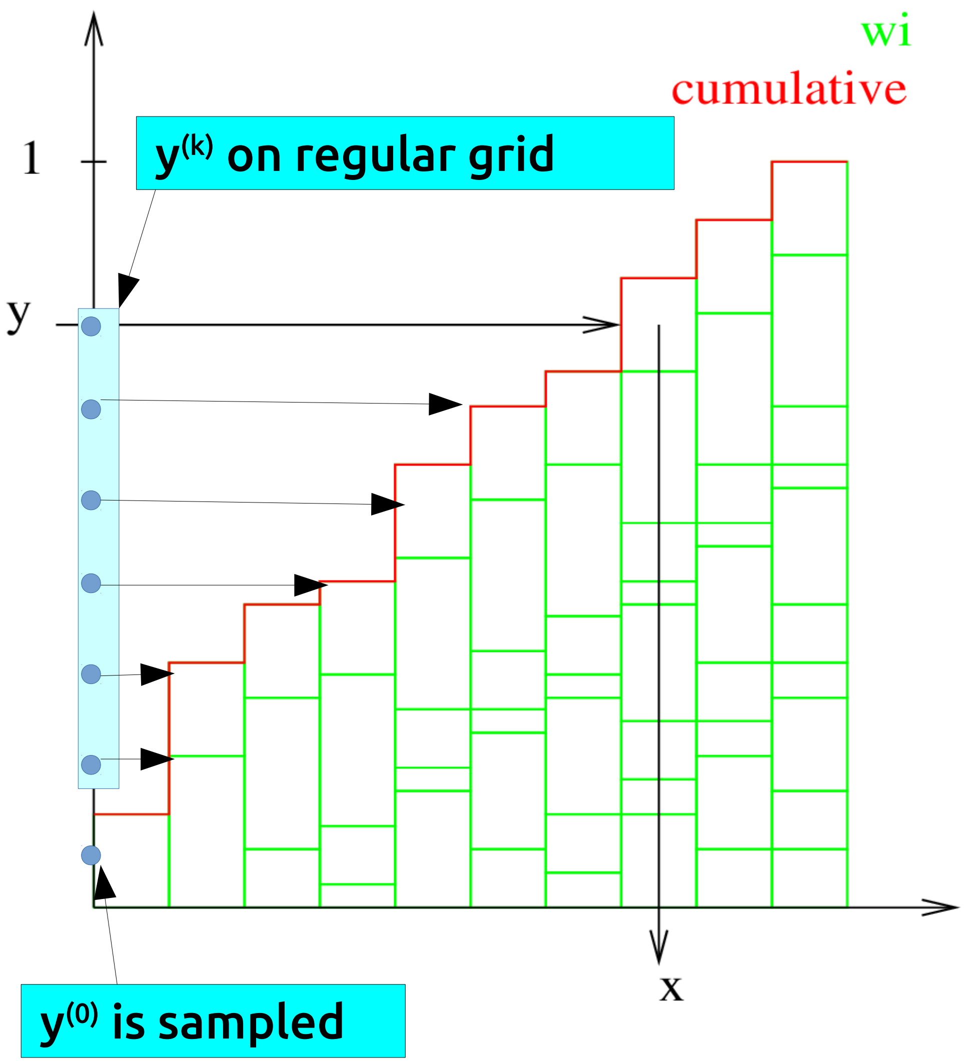
An alternative approach is uniform sampling

- sample a value *$y^{(1)}$* between 1 and 1/N

$$y^{(0)} \sim U(0, \frac{1}{N})$$

- pick the remaining $y^{(i)}$ samples in a regular grid

$$y^{(k)} = y^{(0)} + \frac{k}{N}$$



Uniform Sampling

Octave function

```
function sampled_indices=uniformSample(weights, num_desired_samples)
  %normalize the weights (if they are not normalized)
  normalizer=1./sum(weights);
  %resize the indices
  sampled_indices=zeros(num_desired_samples,1);
  step=1./num_desired_samples;

  y0=rand()*step;      %sample between 0 and 1/num_desired_samples
  yi=y0;               %value of the sample on the y space
  cumulative =0;       %this is our running cumulative distribution
  sample_index=1;      %the index of output where we write the sampled idx
  for (weight_index=1:size(weights,1))
      cumulative += normalizer*weights(weight_index); %update cumulative
      % fill with current_weight_index
      % until the cumulative does not become larger than yi
      while (cumulative>yi)
            sampled_indices(sample_index)=weight_index;
            sample_index++;
            yi+=step;
      endwhile
  endfor
endfunction
```

Importance Sampling

Sometimes we do not know the sampling distribution, so we cannot compute the inverse cumulative. In this case, we can generate **weighted** samples

1.sample from a known distribution $\pi(x)$ possibly close to $p(x)$

$$x^{(i)} \sim \pi(x)$$

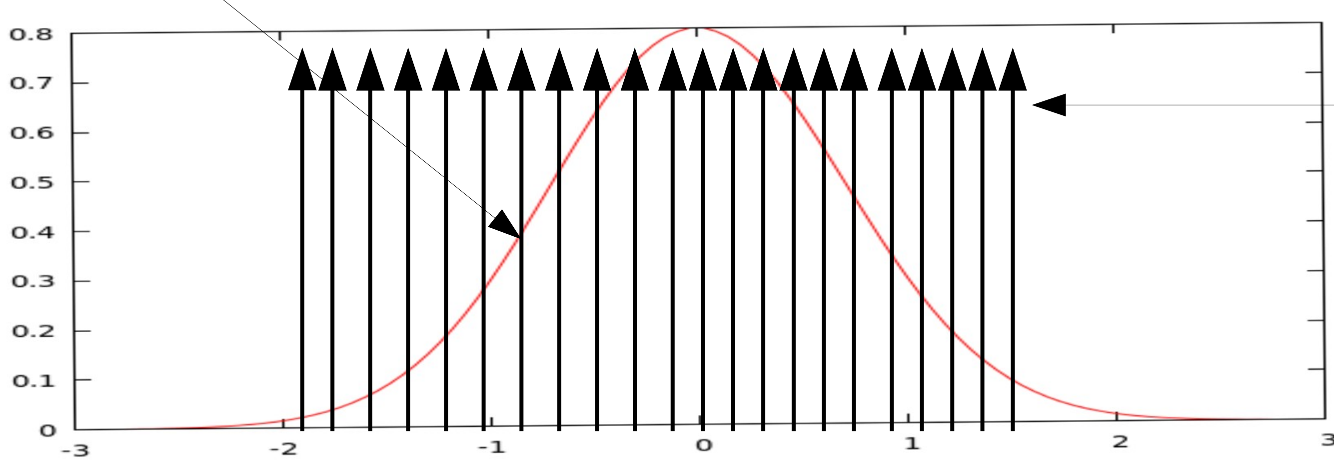
2.compute a weight by evaluating $\pi(x)$ and $p(x)$ at the sampled point

$$w^{(i)} = \frac{p(x^{(i)})}{\pi(x^{(i)})}$$

target distribution

proposal distribution

Gaussian (target)



samples generated by e.g. by a uniform (proposal)

Importance Sampling

Sometimes we do not know the sampling distribution, so we cannot compute the inverse cumulative. In this case, we can generate **weighted** samples

1.sample from a known distribution $\pi(x)$ possibly close to $p(x)$

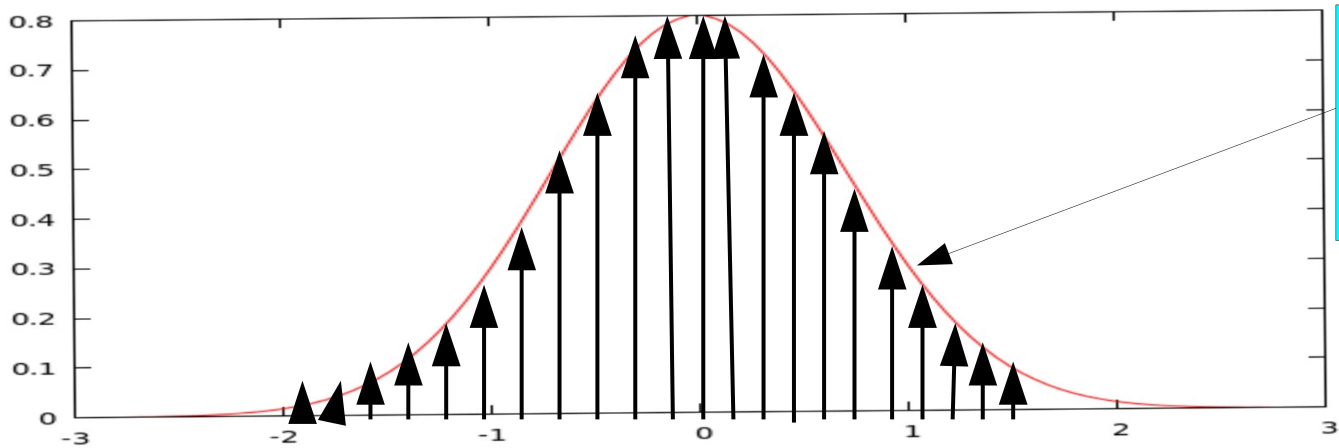
$$x^{(i)} \sim \pi(x)$$

2.compute a weight by evaluating $\pi(x)$ and $p(x)$ at the sampled point

$$w^{(i)} = \frac{p(x^{(i)})}{\pi(x^{(i)})}$$

target distribution

proposal distribution



weights recover the difference between target and proposal

Choice of Proposal

Care must be taken when choosing the proposal

- The proposal $\pi(x)$ should cover all the relevant portion of the target $p(x)$ otherwise some feasible samples might not be generated

$$p(x) > 0 \Rightarrow \pi(x) > 0$$

In the ideal case of sampling from the target distribution, the weights would be uniform

Resampling

If we want to turn a weighed sample set into an unweighed one, we need

- to repeat samples having high weights
- suppress samples with low weight.

This can be done

- by drawing a set of *indices* j from the normalized weights distribution such that

$$p(j) = \boxed{\tilde{w}^{(j)}} = \frac{w^{(j)}}{\sum_i w^{(i)}}$$

normalized weights

Repeating the samples according to the indices generated through the sampling procedure

Resampling

How to proceed?

1. sample a set of indices with probability proportional to the weights

input weighed sample set

$$\left\{\left\langle w^{(i)}, x^{(i)} \right\rangle\right\}$$

array of sample indices

$$\{j^{(i)}\} = \mathrm{uniformSample}(\{w^{(i)}\})$$

resulting samples

$$\tilde{x}^{(i)} = x^{(j(i))}$$

2. the new sample in position *i* is the old sample in position *j[i]*

Particle Densities

We can represent an approximation of a density function by a set of weighed samples

The “denser” the samples in a region, the higher will be the probability of that region

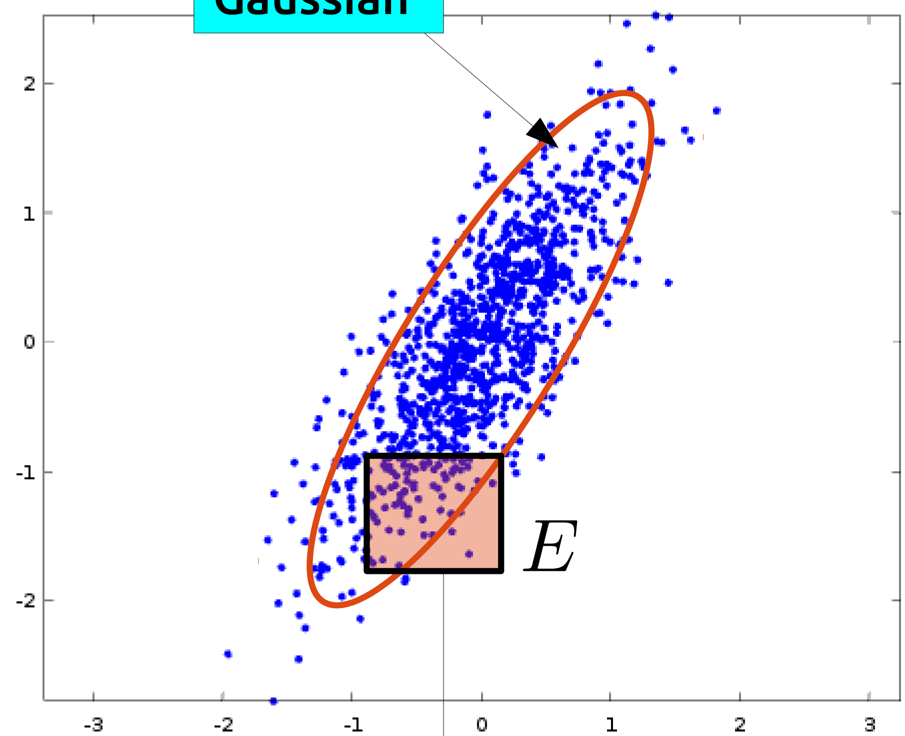
$$\mathbf{x}^{[i]} \sim p(\mathbf{x})$$

Dirac centered in $x^{(i)}$

$$p(\mathbf{x}) \simeq \sum_i w^{(i)} \delta(\mathbf{x} - \mathbf{x}^{(i)})$$

$$\int_E p(\mathbf{x}) d\mathbf{x} \simeq \sum_{\mathbf{x}^{[i]} \in E} w^{(i)}$$

Gaussian



The probability that x falls in a region E can be obtained by summing the weights in the region

Why Particles are Cool

Can represent arbitrary distributions

Easy to “visualize”

Easy to manipulate

Good for small state spaces

Transformation

Transformation is easy

Sampled density

$$p(\mathbf{x}_a) \simeq \sum_i w^{(i)} \delta(\mathbf{x}_a - \mathbf{x}_a^{(i)})$$

$$\mathbf{x}_b = \mathbf{f}(\mathbf{x}_a)$$

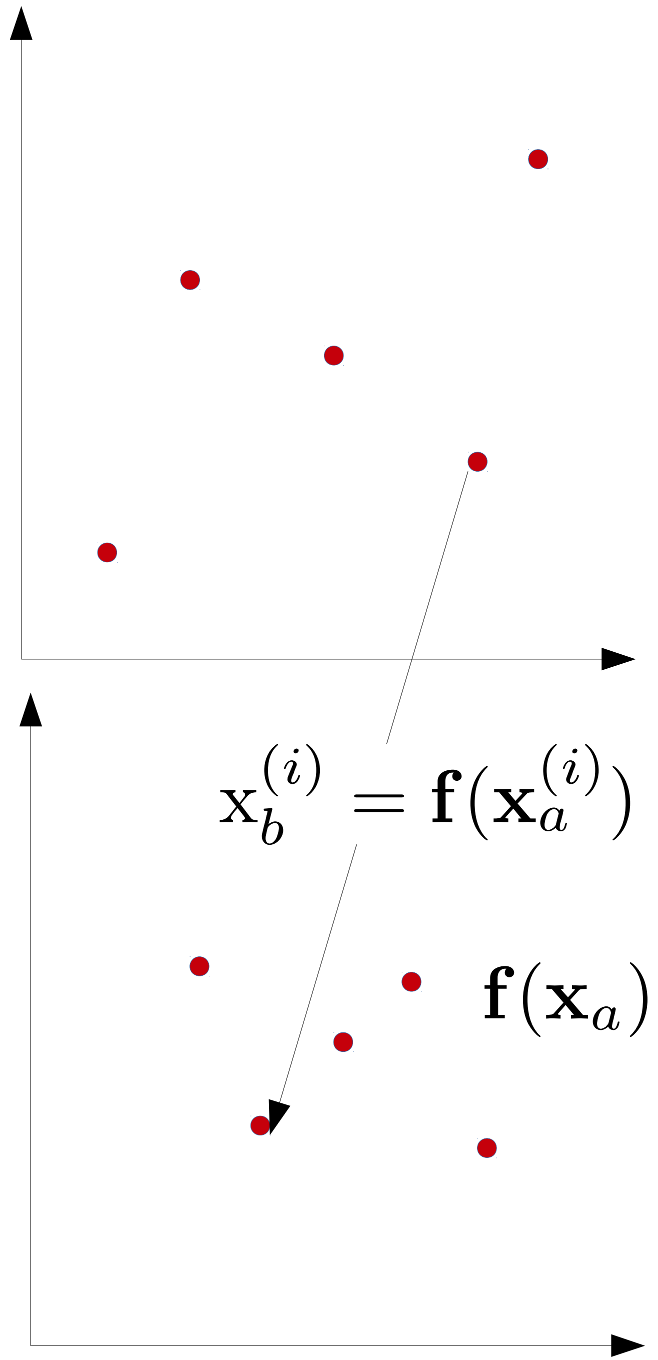
function of random variable

sampled density on x_b

$$p(\mathbf{x}_b) = \simeq \sum_i w^{(i)} \delta\left(\mathbf{x}_b - \mathbf{f}(\mathbf{x}_a^{(i)})\right)$$

$$\mathbf{x}_b^{(i)} = \mathbf{f}(\mathbf{x}_a^{(i)})$$

can be implemented by transforming each sample with f



Marginalization

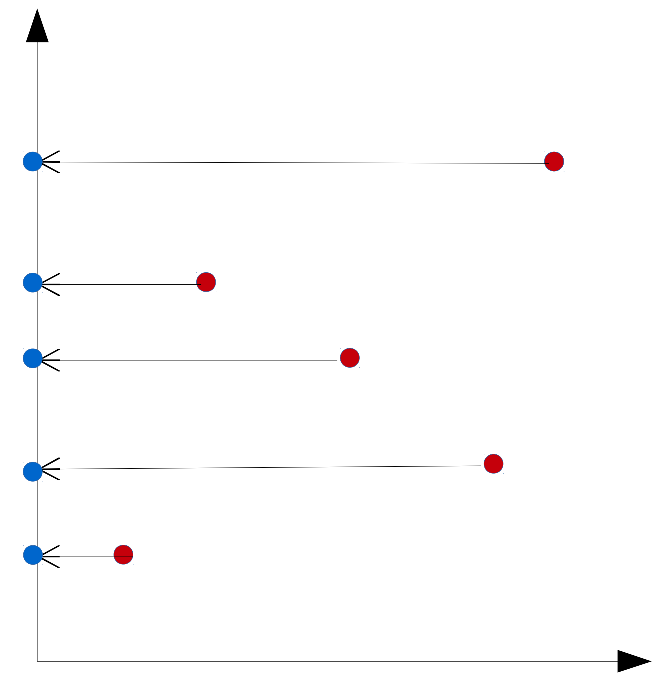
Marginalization just deletes from the sample set the coordinates of the marginalized component

Sampled density on x_a,x_b

$$p(\mathbf{x}_a, \mathbf{x}_b) \simeq \sum_i w^{(i)} \delta \left(\begin{pmatrix} \mathbf{x}_a \\ \mathbf{x}_b \end{pmatrix} - \begin{pmatrix} \mathbf{x}_a^{(i)} \\ \mathbf{x}_b^{(i)} \end{pmatrix} \right)$$

$$p(\mathbf{x}_a) = \int p(\mathbf{x}_a, \mathbf{x}_b) d\mathbf{x}_b$$

$$= \simeq \sum_i w^{(i)} \delta \left(\mathbf{x}_a - \mathbf{x}_a^{(i)} \right)$$



Chain Rule

$$p(\mathbf{x}_a) \simeq \sum_i w^{(i)} \delta(\mathbf{x}_a - \mathbf{x}_a^{(i)})$$

Sampled density on x_a

$$p(\mathbf{x}_b | \mathbf{x}_a)$$

Conditional on $x_b|x_a$

$$\mathbf{x}_b^{(i)} \sim p(\mathbf{x}_b | \mathbf{x}_a^{(i)})$$

1. Generate a sample from the conditional, for each sample in the conditioning

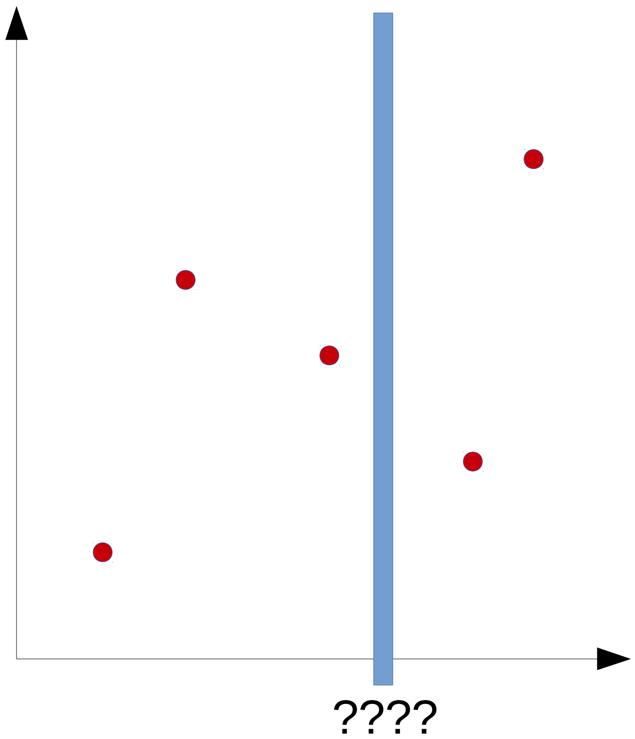
$$p(\mathbf{x}_a, \mathbf{x}_b) \simeq \sum_i w^{(i)} \delta \left(\begin{pmatrix} \mathbf{x}_a \\ \mathbf{x}_b \end{pmatrix} - \begin{pmatrix} \mathbf{x}_a^{(i)} \\ \mathbf{x}_b^{(i)} \end{pmatrix} \right)$$

2. Stack the samples to get a particle from the joint distribution

Conditioning

$$p(\mathbf{x}_a, \mathbf{x}_b) \simeq \sum_i w^{(i)} \delta \left(\begin{pmatrix} \mathbf{x}_a \\ \mathbf{x}_b \end{pmatrix} - \begin{pmatrix} \mathbf{x}_a^{(i)} \\ \mathbf{x}_b^{(i)} \end{pmatrix} \right)$$

$$p(\mathbf{x}_a | \mathbf{x}_b) = \frac{p(\mathbf{x}_a, \mathbf{x}_b)}{p(\mathbf{x}_b)}$$



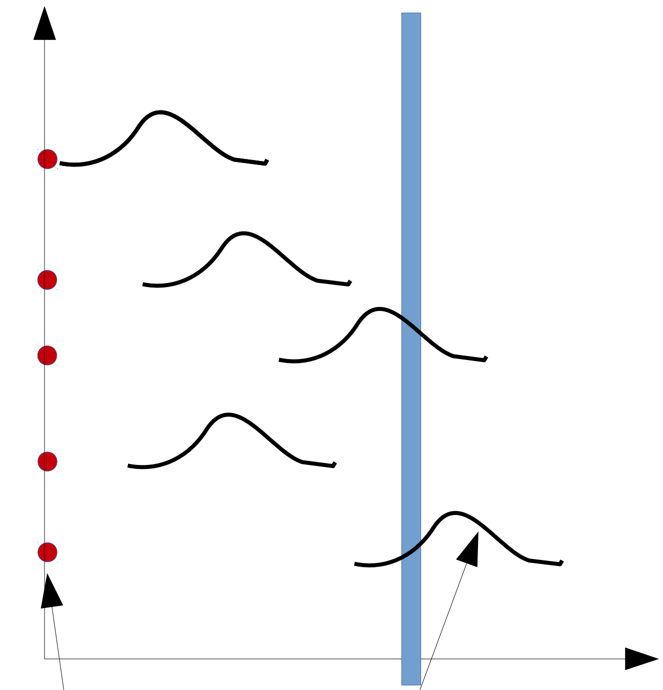
Not easy

Reason:

- Samples do not like to be sliced

Conditioning

Things would be different if we would have for each sample a conditional distribution on x_b



$$p(\mathbf{x}_a, \mathbf{x}_b) \simeq \sum_i w^{(i)} \delta\left(\mathbf{x}_a - \underbrace{\mathbf{x}_a^{(i)}}_{\text{sample}}\right) \underbrace{p(\mathbf{x}_b \mid \mathbf{x}_a^{(i)})}_{\text{conditional given the sample}}$$

Conditioning

$$p(\mathbf{x}_a, \mathbf{x}_b) \simeq \sum_i w^{(i)} \delta\left(\mathbf{x}_a - \mathbf{x}_a^{(i)}\right) p(\mathbf{x}_b \mid \mathbf{x}_a^{(i)})$$

mixture of conditional distributions for each sample

$$p(\mathbf{x}_a|\mathbf{x}_b) = \frac{p(\mathbf{x}_a, \mathbf{x}_b)}{p(\mathbf{x}_b)} = \frac{p(\mathbf{x}_a, \mathbf{x}_b)}{\int p(\mathbf{x}_a, \mathbf{x}_b) d\mathbf{x}_a}$$

expand the conditioning through chain rule and marginalization

$$\simeq \frac{\sum_i w^{(i)} \delta\left(\mathbf{x}_a - \mathbf{x}_a^{(i)}\right) p(\mathbf{x}_b \mid \mathbf{x}_a^{(i)})}{\int \left[\sum_i w^{(i)} \delta\left(\mathbf{x}_a - \mathbf{x}_a^{(i)}\right) p(\mathbf{x}_b \mid \mathbf{x}_a^{(i)})\right] d\mathbf{x}_a}$$

apply the mixture approximation

Flip sum and integral

$$= \frac{\sum_i w^{(i)} \delta\left(\mathbf{x}_a - \mathbf{x}_a^{(i)}\right) p(\mathbf{x}_b \mid \mathbf{x}_a^{(i)})}{\sum_i w^{(i)} p(\mathbf{x}_b \mid \mathbf{x}_a^{(i)}) \int \left[\delta\left(\mathbf{x}_a - \mathbf{x}_a^{(i)}\right)\right] d\mathbf{x}_a}$$

This is 1

Normalizer

$$= \frac{1}{\sum_i w^{(i)} p(\mathbf{x}_b \mid \mathbf{x}_a^{(i)})} \sum_i w^{(i)} \delta\left(\mathbf{x}_a - \mathbf{x}_a^{(i)}\right) p(\mathbf{x}_b \mid \mathbf{x}_a^{(i)})$$

Conditioning

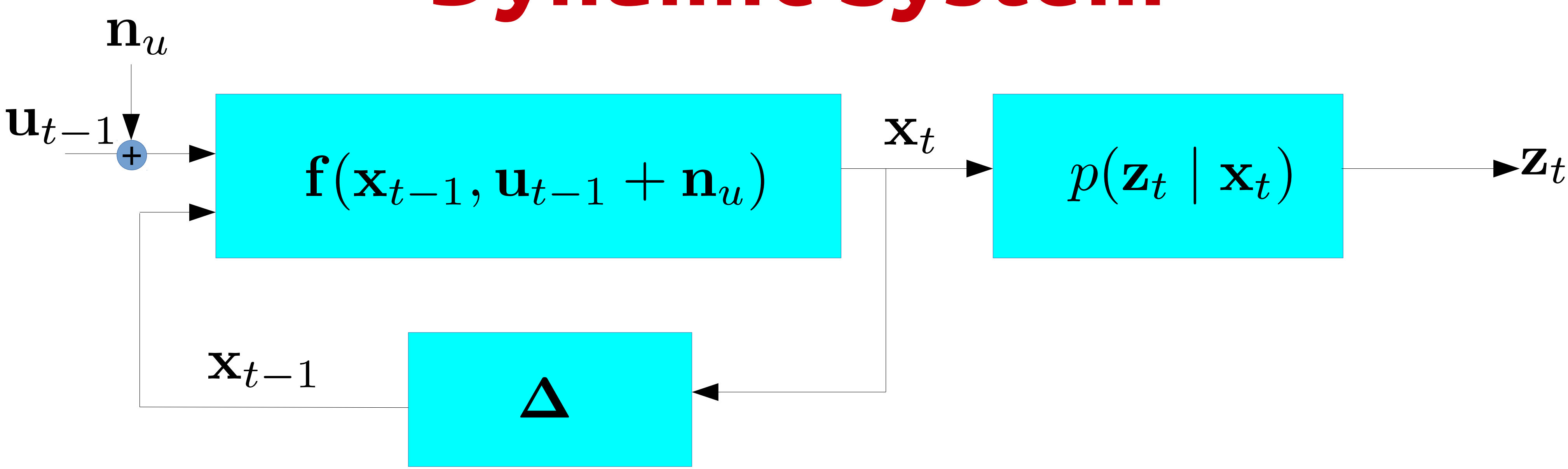
Note that conditioning only affects the weights

$$
\begin{aligned}
p(\mathbf{x}_a|\mathbf{x}_b) &\simeq \frac{1}{\sum_i w^{(i)} p(\mathbf{x}_b \mid \mathbf{x}_a^{(i)})} \sum_i w^{(i)} \delta\left(\mathbf{x}_a - \mathbf{x}_a^{(i)}\right) p(\mathbf{x}_b \mid \mathbf{x}_a^{(i)}) \\
&= \eta \sum_i w^{(i)} p(\mathbf{x}_b \mid \mathbf{x}_a^{(i)}) \delta\left(\mathbf{x}_a - \mathbf{x}_a^{(i)}\right)
\end{aligned}
$$

$$w_{a|b}^{(i)} \propto w_a^{(i)} p(\mathbf{x}_b \mid \mathbf{x}_a^{(i)})$$

To implement conditioning we need to multiply each weight by the conditional of the sample evaluated at the conditioning variable

Dynamic System



$\mathbf{x}_t = \mathbf{f}(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})$ — we know the transition function

$p(\mathbf{x}) \simeq \sum_i w^{(i)} \delta(\mathbf{x} - \mathbf{x}^{(i)})$ — state pdf is a set of weighed samples

$\mathbf{n}_u \sim p(\mathbf{n}_u)$ — additive noise distributed according to a $p(n_u)$ we can sample from

$p(\mathbf{z}_t \mid \mathbf{x}_t)$ — we can evaluate pointwise the observation model

Prediction

$$p(\mathbf{x}_{t-1|t-1}) \simeq \sum_i w^{(i)}_{t-1|t-1} \delta(\mathbf{x}_{t-1|t-1} - \mathbf{x}^{(i)}_{t-1|t-1})$$

prior

$$\mathbf{n}^{(i)}_u \sim p(\mathbf{n}_u)$$

1. generate I noise samples.
<x(i),n(i)> are samples from the joint distribution

$$p(\mathbf{x}_{t|t-1}) \simeq \sum_i w^{(i)}_{t-1|t-1} \delta(\mathbf{x}_{t|t-1} - \mathbf{f}(\mathbf{x}^{(i)}_{t-1|t-1}, \mathbf{u}_{t-1} + \mathbf{n}^{(i)}_u))$$

2. transform each sample with its noise trough f

$$\mathbf{x}^{(i)}_{t|t-1} = \mathbf{f}(\mathbf{x}^{(i)}_{t-1|t-1}, \mathbf{u}_{t-1} + \mathbf{n}^{(i)}_u)$$

$$w^{(i)}_{t|t-1} = w^{(i)}_{t-1|t-1}$$

samples of pdf after prediction

weights of pdf after prediction (unchanged)

Update

$$p(\mathbf{x}_{t|t-1}) \simeq \sum_i w_{t|t-1}^{(i)} \delta(\mathbf{x}_{t|t-1} - \mathbf{x}_{t|t-1}^{(i)})$$

prediction

$$p(\mathbf{x}_{t|t}) \simeq \sum_i w_{t-1|t-1}^{(i)} p(\mathbf{z}_t \mid \mathbf{x}^{(i)_t}) \delta(\mathbf{x}_{t|t-1} - \mathbf{x}_{t|t-1}^{(i)})$$

conditioned

$$w_{t|t}^{(i)} = w_{t|t-1}^{(i)} p(\mathbf{z}_t \mid \mathbf{x}_{t|t-1}^{(i)})$$

Weights after update. Multiply each weight of prediction by the likelihood of the measurement

$$\mathbf{x}_{t|t}^{(i)} = \mathbf{x}_{t|t-1}^{(i)}$$

Samples after update. Unchanged.

Resample a new generation to focus computation on likely regions of the state space

Particle Filter wrapup

Predict

$$\mathbf{n}_u^{(i)} \sim p(\mathbf{n}_u)$$

$$\mathbf{x}_{t|t-1}^{(i)} = \mathbf{f}(\mathbf{x}_{t-1|t-1}^{(i)}, \mathbf{u}_{t-1} + \mathbf{n}_u^{(i)})$$

$$w_{t|t-1}^{(i)} = w_{t-1|t-1}^{(i)}$$

1. generate I noise samples.

2. apply f to each sample+noise

Update

$$w_{t|t}^{(i)} = w_{t|t-1}^{(i)} p(\mathbf{z}_t \mid \mathbf{x}_{t|t-1}^{(i)})$$

$$\mathbf{x}_{t|t}^{(i)} = \mathbf{x}_{t|t-1}^{(i)}$$

3. multiply each weight by the conditional of the measurement evaluated at the weight

4. Resample a new generation to focus computation on likely regions of the state space