Particle Filtering: Principle

The idea is to form a weighted particle presentation \((x^{(i)}, w^{(i)})\) of the posterior distribution:

\[
p(x) \approx \sum_{i} w^{(i)} \delta(x - x^{(i)}).
\]

Approximates Bayesian optimal filtering equations with importance sampling.

Particle filtering = Sequential importance sampling, with additional resampling step.
Monte Carlo Integration

In Bayesian inference we often want to compute posterior expectations of the form

$$E[g(x) \mid y_{1:T}] = \int g(x) \ p(x \mid y_{1:T}) \, dx$$

Monte Carlo: draw \( N \) independent random samples from \( x^{(i)} \sim p(x \mid y_{1:T}) \) and estimate the expectation as

$$E[g(x) \mid y_{1:T}] \approx \frac{1}{N} \sum_{i=1}^{N} g(x^{(i)})$$
In practice, we rarely can directly draw samples from the distribution $p(x \mid y_{1:T})$.

In importance sampling (IS), we draw samples from an importance distribution $x^{(i)} \sim \pi(x \mid y_{1:T})$ and compute weights $\tilde{w}^{(i)}$ such that

$$E[g(x) \mid y_{1:T}] \approx \sum_{i=1}^{N} \tilde{w}^{(i)} g(x^{(i)})$$
Importance sampling is based on the identity

\[
E[g(x) \mid y_{1:T}] = \int g(x) p(x \mid y_{1:T}) \, dx
\]

\[
= \int \left[ g(x) \frac{p(x \mid y_{1:T})}{\pi(x \mid y_{1:T})} \right] \pi(x \mid y_{1:T}) \, dx
\]

Thus we can form a Monte Carlo approximation as follows:

\[
E[g(x) \mid y_{1:T}] \approx \frac{1}{N} \sum_{i=1}^{N} \frac{p(x^{(i)} \mid y_{1:T})}{\pi(x^{(i)} \mid y_{1:T})} g(x^{(i)})
\]

That is, the importance weights can be defined as

\[
\tilde{w}^{(i)} = \frac{1}{N} \frac{p(x^{(i)} \mid y_{1:T})}{\pi(x^{(i)} \mid y_{1:T})}
\]
The problem is that we need to evaluate the normalization constant of $p(x^{(i)} | y_{1:T})$ – often not possible.

However, it turns out that we get a valid algorithm if we define unnormalized importance weights as

$$w^{*(i)} = \frac{p(y_{1:T} | x^{(i)}) p(x^{(i)})}{\pi(x^{(i)} | y_{1:T})}$$

and then normalize them:

$$w^{(i)} = \frac{w^{*(i)}}{\sum_j w^{*(j)}}$$

The (weight-normalized) importance sampling approximation is then

$$E[g(x) | y_{1:T}] \approx \sum_{i=1}^N w^{(i)} g(x^{(i)})$$
Importance Sampling: Algorithm

<table>
<thead>
<tr>
<th>Importance Sampling</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Draw $N$ samples from the importance distribution:</td>
</tr>
<tr>
<td>$x^{(i)} \sim \pi(x \mid y_{1:T}), \quad i = 1, \ldots, N.$</td>
</tr>
<tr>
<td>• Compute the unnormalized weights by</td>
</tr>
<tr>
<td>$w^*(i) = \frac{p(y_{1:T} \mid x^{(i)}) p(x^{(i)})}{\pi(x^{(i)} \mid y_{1:T})},$</td>
</tr>
<tr>
<td>and the normalized weights by</td>
</tr>
<tr>
<td>$w^{(i)} = \frac{w^<em>(i)}{\sum_{j=1}^{N} w^</em>(j)}.$</td>
</tr>
</tbody>
</table>
The approximation to the posterior expectation of $g(x)$ is then given as

$$E[g(x) \mid y_{1:T}] \approx \sum_{i=1}^{N} w^{(i)} g(x^{(i)}).$$

The posterior probability density approximation can then be formally written as

$$p(x \mid y_{1:T}) \approx \sum_{i=1}^{N} w^{(i)} \delta(x - x^{(i)}),$$

where $\delta(\cdot)$ is the Dirac delta function.

The efficiency depends on the choice of the importance distribution.
Sequential Importance Sampling (SIS) is concerned with models

\[ x_k \sim p(x_k \mid x_{k-1}) \]
\[ y_k \sim p(y_k \mid x_k) \]

The SIS algorithm uses a weighted set of *particles* 
\[ \{(w_k^{(i)}, x_k^{(i)}) : i = 1, \ldots, N\} \] such that

\[ E[g(x_k) \mid y_1:k] \approx \sum_{i=1}^{N} w_k^{(i)} g(x_k^{(i)}). \]

Or equivalently

\[ p(x_k \mid y_1:k) \approx \sum_{i=1}^{N} w_k^{(i)} \delta(x_k - x_k^{(i)}), \]

where \( \delta(\cdot) \) is the Dirac delta function.

Uses importance sampling sequentially.
Sequential Importance Sampling: Derivation [1/2]

- Let’s consider the **full posterior** distribution of states $x_{0:k}$ given the measurements $y_{1:k}$.
- We get the following **recursion** for the **posterior distribution**:

\[
p(x_{0:k} \mid y_{1:k}) \propto p(y_k \mid x_{0:k}, y_{1:k-1}) p(x_{0:k} \mid y_{1:k-1})
\]

\[= p(y_k \mid x_k) p(x_k \mid x_{0:k-1}, y_{1:k-1}) p(x_{0:k-1} \mid y_{1:k-1})
\]

\[= p(y_k \mid x_k) p(x_k \mid x_{k-1}) p(x_{0:k-1} \mid y_{1:k-1}).
\]

- We could now construct an **importance distribution** $x_{0:k}^{(i)} \sim \pi(x_{0:k} \mid y_{1:k})$ and compute the corresponding (normalized) **importance weights** as

\[
w^{(i)}_k \propto \frac{p(y_k \mid x^{(i)}_k) p(x^{(i)}_k \mid x^{(i)}_{k-1}) p(x^{(i)}_{0:k-1} \mid y_{1:k-1})}{\pi(x^{(i)}_{0:k} \mid y_{1:k})}.
\]
Let’s form the importance distribution recursively as follows:

\[
\pi(x_{0:k} | y_{1:k}) = \pi(x_k | x_{0:k-1}, y_{1:k}) \pi(x_{0:k-1} | y_{1:k-1})
\]

Expression for the importance weights can be written as

\[
w_k^{(i)} \propto \frac{p(y_k | x_k^{(i)}) p(x_k^{(i)} | x_{k-1}^{(i)})}{\pi(x_k^{(i)} | x_{0:k-1}, y_{1:k})} \frac{p(x_{0:k-1}^{(i)} | y_{1:k-1})}{\pi(x_{0:k-1}^{(i)} | y_{1:k-1})} \propto w_{k-1}^{(i)}
\]

Thus the weights satisfy the recursion

\[
w_k^{(i)} \propto \frac{p(y_k | x_k^{(i)}) p(x_k^{(i)} | x_{k-1}^{(i)})}{\pi(x_k^{(i)} | x_{0:k-1}^{(i)}, y_{1:k})} w_{k-1}^{(i)}
\]
Sequential Importance Sampling: Algorithm

Sequential Importance Sampling

- **Initialization:** Draw $N$ samples $x_0^{(i)}$ from the prior

\[ x_0^{(i)} \sim p(x_0) \]

and set $w_0^{(i)} = 1/N$.

- **Prediction:** Draw $N$ new samples $x_k^{(i)}$ from importance distributions

\[ x_k^{(i)} \sim \pi(x_k | x_0^{(i)} : k-1, y_1:k) \]

- **Update:** Calculate new weights according to

\[ w_k^{(i)} \propto w_{k-1}^{(i)} \frac{p(y_k | x_k^{(i)}) \ p(x_k^{(i)} | x_{k-1}^{(i)})}{\pi(x_k^{(i)} | x_0^{(i)} : k-1, y_1:k)} \]
The problem in SIS is that the algorithm is degenerate. It can be shown that the variance of the weights increases at every step. It means that we will always converge to single non-zero weight $w^{(i)} = 1$ and the rest being zero – not very useful algorithm. 

Solution: resampling!
Sequential Importance Resampling (SIR) algorithm adds the following resampling step to SIS algorithm:

- Interpret each weight $w_k^{(i)}$ as the probability of obtaining the sample index $i$ in the set $\{x_k^{(i)} \mid i = 1, \ldots, N\}$.
- Draw $N$ samples from that discrete distribution and replace the old sample set with this new one.
- Set all weights to the constant value $w_k^{(i)} = 1/N$.

There are many algorithms for implementing this — *stratified resampling* is optimal in terms of variance.
A simple way to do resampling is at every step – but every resampling operation increases variance.

We can also resample at, say, every $K$th step.

In adaptive resampling, we resample when the effective number of samples is too low (say, $N/10$):

$$n_{\text{eff}} \approx \frac{1}{\sum_{i=1}^{N} (w_k^{(i)})^2},$$

In theory, biased, but in practice works very well and is often used.
Sequential Importance Resampling: Algorithm

Sequential Importance Resampling

- Draw point $x_k^{(i)}$ from the importance distribution:

$$x_k^{(i)} \sim \pi(x_k \mid x_{0:k-1}^{(i)}, y_{1:k}), \quad i = 1, \ldots, N.$$ 

- Calculate new weights

$$w_k^{(i)} \propto w_{k-1}^{(i)} \frac{p(y_k \mid x_k^{(i)}) p(x_k^{(i)} \mid x_{k-1}^{(i)})}{\pi(x_k^{(i)} \mid x_{0:k-1}^{(i)}, y_{1:k})}, \quad i = 1, \ldots, N,$$

and normalize them to sum to unity.

- If the effective number of particles is too low, perform resampling.
In bootstrap filter we use the dynamic model as the importance distribution

\[ \pi(x_k^{(i)} \mid x_{0:k-1}, y_{1:k}) = p(x_k^{(i)} \mid x_{k-1}^{(i)}) \]

and resample at every step:

**Bootstrap Filter**

- Draw point \( x_k^{(i)} \) from the dynamic model:
  \[
  x_k^{(i)} \sim p(x_k \mid x_{k-1}^{(i)}), \quad i = 1, \ldots, N.
  \]

- Calculate new weights
  \[
  w_k^{(i)} \propto p(y_k \mid x_k^{(i)}), \quad i = 1, \ldots, N,
  \]

  and normalize them to sum to unity.

- Perform resampling.
The optimal importance distribution is

\[ \pi(x_k^{(i)} \mid x_{0:k-1}^{(i)}, y_{1:k}) = p(x_k^{(i)} \mid x_{k-1}^{(i)}, y_k) \]

Then the weight update reduces to

\[ w_k^{(i)} \propto w_{k-1}^{(i)} p(y_k \mid x_{k-1}^{(i)}), \quad i = 1, \ldots, N. \]

The optimal importance distribution can be used, for example, when the state space is finite.
We can also form a **Gaussian approximation** to the optimal importance distribution:

\[
p(x_{k}^{(i)} | x_{k-1}^{(i)}, y_k) \approx N(x_{k}^{(i)} | \tilde{m}_{k}^{(i)}, \tilde{P}_{k}^{(i)}).
\]

by using a single prediction and update steps of a Gaussian filter starting from a singular distribution at \(x_{k-1}^{(i)}\).

We can also replace above with the result of a Gaussian filter \(N(m_{k-1}^{(i)}, P_{k-1}^{(i)})\) started from a random initial mean.

A very common way seems to be to use the previous sample as the mean: \(N(x_{k-1}^{(i)}, P_{k-1}^{(i)})\).

A particle filter with UKF proposal has been given name **unscented particle filter (UPF)** — you can invent new PFs easily this way.
Rao-Blackwellized Particle Filter: Idea

- Rao-Blackwellized particle filtering (RBPF) is concerned with conditionally Gaussian models:

\[
p(x_k \mid x_{k-1}, \theta_{k-1}) = \mathcal{N}(x_k \mid A_{k-1}(\theta_{k-1}) x_{k-1}, Q_{k-1}(\theta_{k-1}))
\]
\[
p(y_k \mid x_k, \theta_k) = \mathcal{N}(y_k \mid H_k(\theta_k) x_k, R_k(\theta_k))
\]
\[
p(\theta_k \mid \theta_{k-1}) = \text{(any given form)},
\]

where
- \(x_k\) is the state
- \(y_k\) is the measurement
- \(\theta_k\) is an arbitrary latent variable

- **Given** the latent variables \(\theta_1: T\) the model is Gaussian.
- The RBPF uses SIR for the latent variables and computes the conditionally Gaussian part in closed form with Kalman filter.
The full posterior at step $k$ can be factored as

$$p(\theta_{0:k}, x_{0:k} \mid y_{1:k}) = p(x_{0:k} \mid \theta_{0:k}, y_{1:k}) p(\theta_{0:k} \mid y_{1:k})$$

The first term is Gaussian and computable with Kalman filter and RTS smoother.

For the second term we get the following recursion:

$$p(\theta_{0:k} \mid y_{1:k})$$

$$\propto p(y_k \mid \theta_{0:k}, y_{1:k-1}) p(\theta_{0:k} \mid y_{1:k-1})$$

$$= p(y_k \mid \theta_{0:k}, y_{1:k-1}) p(\theta_k \mid \theta_{0:k-1}, y_{1:k-1}) p(\theta_{0:k-1} \mid y_{1:k-1})$$

$$= p(y_k \mid \theta_{0:k}, y_{1:k-1}) p(\theta_k \mid \theta_{k-1}) p(\theta_{0:k-1} \mid y_{1:k-1})$$
Let’s take a look at the terms in

\[ p(y_k \mid \theta_{0:k}, y_{1:k-1}) p(\theta_k \mid \theta_{k-1}) p(\theta_{0:k-1} \mid y_{1:k-1}) \]

- The first term can be computed by running Kalman filter with fixed \( \theta_{0:k} \) over the measurement sequence.
- The second term is just the dynamic model.
- The third term is the posterior from the previous step.
We can form the importance distribution recursively:

\[
\pi(\theta_{0:k} \mid y_{1:k}) = \pi(\theta_k \mid \theta_{0:k-1}, y_{1:k}) \pi(\theta_{0:k-1} \mid y_{1:k-1})
\]

We then get the following recursion for the weights:

\[
w_{k}^{(i)} \propto \frac{p(y_k \mid \theta_{0:k-1}^{(i)}, y_{1:k-1}) p(\theta_k^{(i)} \mid \theta_{0:k-1}^{(i)})}{\pi(\theta_k^{(i)} \mid \theta_{0:k-1}, y_{1:k})} \pi(\theta_k \mid \theta_{0:k-1}, y_{1:k})
\]

Given the marginal posterior for \( \theta_{0:k} \) we can recover the Gaussian part \( x_{0:k} \) with Kalman filter and RTS smoother.

The optimal importance distribution takes the form

\[
p(\theta_k \mid y_{1:k}, \theta_{0:k-1}^{(i)}) \propto p(y_k \mid \theta_k, \theta_{0:k-1}^{(i)}) p(\theta_k \mid \theta_{0:k-1}^{(i)}, y_{1:k-1})
\]
Rao-Blackwellized Particle Filter: Algorithm [1/3]

**Rao-Blackwellized Particle Filter**

- Perform Kalman filter predictions for each of the Kalman filter means and covariances in the particles $i = 1, \ldots, N$ conditional on the previously drawn latent variable values $\theta_{k-1}^{(i)}$

\[
\begin{align*}
\mathbf{m}_k^{-(i)} &= \mathbf{A}_{k-1}(\theta_{k-1}^{(i)}) \mathbf{m}_{k-1}^{(i)} \\
\mathbf{P}_k^{-(i)} &= \mathbf{A}_{k-1}(\theta_{k-1}^{(i)}) \mathbf{P}_{k-1}^{(i)} \mathbf{A}_{k-1}^T(\theta_{k-1}^{(i)}) + \mathbf{Q}_{k-1}(\theta_{k-1}^{(i)}).
\end{align*}
\]

- Draw new latent variables $\theta_k^{(i)}$ for each particle in $i = 1, \ldots, N$ from the corresponding importance distributions

\[
\theta_k^{(i)} \sim \pi(\theta_k | \theta_{0:k-1}^{(i)}, y_{1:k}).
\]
Rao-Blackwellized Particle Filter (cont.)

- Calculate new weights as follows:

\[ w_k^{(i)} \propto w_{k-1}^{(i)} \frac{p(y_k \mid \theta_{0:k}^{(i)}, y_{1:k-1}) \, p(\theta_k^{(i)} \mid \theta_{k-1}^{(i)})}{\pi(\theta_k^{(i)} \mid \theta_{0:k-1}^{(i)}, y_{1:k})}, \]

where the likelihood term is the marginal measurement likelihood of the Kalman filter:

\[
p(y_k \mid \theta_{0:k}^{(i)}, y_{1:k-1}) = N\left( y_k \mid H_k(\theta_k^{(i)}) m_k^{(i)}, H_k(\theta_k^{(i)}) P_k^{(i)} H_k^T(\theta_k^{(i)}) + R_k(\theta_k^{(i)}) \right).
\]

- Then normalize the weights to sum to unity.
Perform Kalman filter updates for each of the particles conditional on the drawn latent variables $\theta^{(i)}_k$

- $v_k^{(i)} = y_k - H_k(\theta^{(i)}_k) m_k^-$
- $S_k^{(i)} = H_k(\theta^{(i)}_k) P_k^{(i)} H_k^T(\theta^{(i)}_k) + R_k(\theta^{(i)}_k)$
- $K_k^{(i)} = P_k^{(i)} H_k^T(\theta^{(i)}_k) S_k^{-1}$
- $m_k^{(i)} = m_k^- + K_k^{(i)} v_k^{(i)}$
- $P_k^{(i)} = P_k^{(i)} - K_k^{(i)} S_k^{(i)} [K_k^{(i)}]^T$.

If the effective number of particles is too low, perform resampling.
Rao-Blackwellized Particle Filter: Properties

- The Rao-Blackwellized particle filter produces a set of weighted samples \( \{ w_k^{(i)} , \theta_k^{(i)} , m_k^{(i)} , P_k^{(i)} : i = 1, \ldots, N \} \)
- The expectation of a function \( g(\cdot) \) can be approximated as
  \[
  E[g(x_k, \theta_k) \mid y_{1:k}] \approx \sum_{i=1}^{N} w_k^{(i)} \int g(x_k, \theta_k^{(i)}) N(x_k \mid m_k^{(i)}, P_k^{(i)}) \, dx_k.
  \]
- Approximation of the filtering distribution is
  \[
  p(x_k, \theta_k \mid y_{1:k}) \approx \sum_{i=1}^{N} w_k^{(i)} \delta(\theta_k - \theta_k^{(i)}) N(x_k \mid m_k^{(i)}, P_k^{(i)}).
  \]
- It is possible to do approximate Rao-Blackwellization by replacing the Kalman filter with a Gaussian filter.
Rao-Blackwellization can sometimes be used in models of the form

\[
\begin{align*}
    x_k &\sim p(x_k \mid x_{k-1}, \theta) \\
    y_k &\sim p(y_k \mid x_k, \theta) \\
    \theta &\sim p(\theta),
\end{align*}
\]

where vector \( \theta \) contains the unknown static parameters. Possible if the posterior distribution of parameters \( \theta \) depends only on some sufficient statistics \( T_k \):

\[
T_k = T_k(x_{1:k}, y_{1:k})
\]

We also need to have a recursion rule for the sufficient statistics.

Can be extended to time-varying parameters.
No restrictions in model – can be applied to non-Gaussian models, hierarchical models etc.

Global approximation.

Approaches the exact solution, when the number of samples goes to infinity.

In its basic form, very easy to implement.

Superset of other filtering methods – Kalman filter is a Rao-Blackwellized particle filter with one particle.
• **Computational requirements** much higher than of the Kalman filters.

• Problems with **nearly noise-free models**, especially with accurate dynamic models.

• Good importance distributions and efficient Rao-Blackwellized filters quite **tricky to implement**.

• Very hard to find **programming errors** (i.e., to debug).
Particle filters use weighted set of samples (particles) for approximating the filtering distributions.

Sequential importance resampling (SIR) is the general framework and bootstrap filter is a simple special case of it.

EKF, UKF and other Gaussian filters can be used for forming good importance distributions.

In Rao-Blackwellized particle filters a part of the state is sampled and part is integrated in closed form with Kalman filter.
The discretized pendulum model:

\[
\begin{pmatrix}
  x_{1k}^1 \\
  x_{2k}^2 \\
  x_{k}^1
\end{pmatrix}
= \begin{pmatrix}
  x_{k-1}^1 + x_{k-1}^2 \Delta t \\
  x_{k-1}^2 - g \sin(x_{k-1}^1) \Delta t \\
  f(x_{k-1})
\end{pmatrix} + \begin{pmatrix}
  0 \\
  q_{k-1}
\end{pmatrix}
\]

\[
y_k = \sin(x_{k}^1) + r_k, \\
= h(x_k)
\]

⇒ Matlab demonstration