Sequential Monte Carlo methods

Lecture 4 – The bootstrap particle filter

Fredrik Lindsten, Uppsala University

2017-08-24

Outline – Lecture 4

Aim: Derive our first sequential Monte Carlo method: the bootstrap particle filter.

Outline:

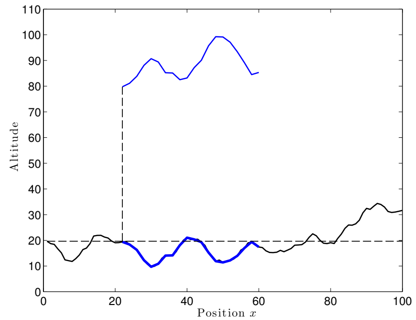
1. A (hopefully) intuitive preview
2. The bootstrap particle filter
3. Resampling
4. A toy example and a real world application

Particle filter preview

A (hopefully) intuitive preview (I/III)

Consider a toy 1D localization problem.

Data



Model

Dynamics:

$$X_{t+1} = X_t + u_t + V_t,$$

where X_t denotes position, u_t denotes velocity (known), $V_t \sim \mathcal{N}(0, 5)$ denotes an unknown disturbance.

Measurements:

$$Y_t = h(X_t) + E_t.$$

where $h(\cdot)$ denotes the world model (here the terrain height) and $E_t \sim \mathcal{N}(0, 1)$ denotes an unknown disturbance.

Task: Learn about the state X_t (position) based on the measurements $y_{1:t}$ by computing the filter density $p(x_t \mid y_{1:t})$.

A (hopefully) intuitive preview (II/III)

A (hopefully) intuitive preview (III/III)

Highlights two **key capabilities** of the PF:

1. Automatically handles an unknown and dynamically changing number of hypotheses (modes).
2. Works with nonlinear/non-Gaussian models.

The bootstrap particle filter

Nonlinear filtering problem

Recall that the nonlinear filtering problem amounts to computing the filter PDF $p(x_t \mid y_{1:t})$ when the model is given by

$$\begin{aligned} X_{t+1} \mid (X_t = x_t) &\sim p(x_{t+1} \mid x_t), \\ Y_t \mid (X_t = x_t) &\sim p(y_t \mid x_t), \\ X_0 &\sim p(x_0). \end{aligned}$$

We have shown that the solution is

$$\begin{aligned} p(x_t \mid y_{1:t}) &= \frac{p(y_t \mid x_t) p(x_t \mid y_{1:t-1})}{p(y_t \mid y_{1:t-1})}, \\ p(x_t \mid y_{1:t-1}) &= \int p(x_t \mid x_{t-1}) p(x_{t-1} \mid y_{1:t-1}) \mathrm{d}x_{t-1}. \end{aligned}$$

Basic idea: Try to approximate $p(x_t \mid y_{1:t})$ sequentially in time $t = 0, 1, \ldots$ using importance sampling!

Particle filter – representation

The **particle filter** approximates $p(x_t \mid y_{1:t})$ by maintaining an **empirical distribution** made up of N samples (particles) $\{x_t^i\}_{i=1}^N$ and corresponding importance weights $\{w_t^i\}_{i=1}^N$

$$\underbrace{\widehat{p}^N(x_t \mid y_{1:t})}_{\widehat{\pi}^N(x_t)} = \sum_{i=1}^N w_t^i \delta_{x_t^i}(x_t).$$

The particle filter provides a well-founded way of exploring the state space using random simulation.

Importance sampling reminder

Algorithm 1 Importance sampler

1. Sample $x^i \sim q(x)$.
2. Compute the weights $\widetilde{w}^i = \widetilde{\pi}(x^i)/q(x^i)$.
3. Normalize the weights $w^i = \widetilde{w}^i / \sum_{j=1}^N \widetilde{w}^j$.

Each step is carried out for $i = 1, \dots, N$.

Sampling from the proposal

We sample from the proposal

$$q(x_t \mid y_{1:t}) = \sum_{i=1}^N \nu_{t-1}^i q(x_t \mid x_{t-1}^i, y_t)$$

using a two step procedure:

1. Select one of the components

$$a_t^i \sim \mathcal{C}(\{\nu_{t-1}^j\}_{j=1}^N) \qquad \text{(categorical distribution)}$$

2. Generate a sample from the selected component,

$$x_t^i \sim q(x_t \mid x_{t-1}^{a_t^i}, y_t)$$

Repeat this N times, for $i = 1, \dots, N$.

Selecting the mixture components – resampling

The particle $\bar{x}_{t-1}^i = x_{t-1}^{a_t^i}$ is referred to as the **ancestor** of x_t^i, since x_t^i is generated conditionally on $\bar{x}_{t-1}^i$.

The variable $a_t^i \in \{1, \dots, N\}$ is referred to as the **ancestor index**, since it indexes the ancestor of particle x_t^i at time $t-1$.

Sampling the N ancestor indices

$$a_t^i \sim \mathcal{C}(\{\nu_{t-1}^j\}_{j=1}^N), \qquad i = 1, \dots, N$$

is referred to as **resampling**.

Resampling generates a new set of particles $\{\bar{x}_{t-1}^i\}_{i=1}^N$ by **sampling with replacement** from among $\{x_{t-1}^j\}_{j=1}^N$, according to some weights $\{\nu_{t-1}^j\}_{j=1}^N$.

Next step – computing the weights

Algorithm 2 Importance sampler

1. Sample $x^i \sim q(x)$.
2. **Compute the weights $\widetilde{w}^i = \widetilde{\pi}(x^i)/q(x^i)$.**
3. Normalize the weights $w^i = \widetilde{w}^i / \sum_{j=1}^N \widetilde{w}^j$.

Each step is carried out for $i = 1, \ldots, N$.

Result – A first particle filter

Algorithm 3 Bootstrap particle filter (for $i = 1, \ldots, N$)

1. **Initialization ($t = 0$):**
   (a) Sample $x_0^i \sim p(x_0)$.
   (b) Set initial weights: $w_0^i = 1/N$.
2. **for** $t = 1$ **to** T **do**
   (a) **Resample:** sample ancestor indices $a_t^i \sim \mathcal{C}(\{w_{t-1}^j\}_{j=1}^N)$.
   (b) **Propagate:** sample $x_t^i \sim p(x_t \mid x_{t-1}^{a_t^i})$.
   (c) **Weight:** compute $\widetilde{w}_t^i = p(y_t \mid x_t^i)$ and normalize $w_t^i = \widetilde{w}_t^i / \sum_{j=1}^N \widetilde{w}_t^j$.

SMC structure

Same structure for all SMC algorithms.

For the bootstrap PF, given $\{x_{t-1}^i, w_{t-1}^i\}_{i=1}^N$:

Resampling: $a_t^i \sim \mathcal{C}(\{w_{t-1}^j\}_{j=1}^N)$.

Propagation: $x_t^i \sim p(x_t \mid x_{t-1}^{a_t^i})$.

Weighting: $\widetilde{w}_t^i = p(y_t \mid x_t^i)$ and normalize.

The result is a new weighted set of particles $\{x_t^i, w_t^i\}_{i=1}^N$.

Intermediate approximations

Approximation of filtering distribution at time $t-1$:

$$\sum_{i=1}^N w_{t-1}^i \delta_{x_{t-1}^i}(x_{t-1}) \approx p(x_{t-1} \mid y_{1:t-1}).$$

For the **bootstrap particle filter**:

- After resampling: $\frac{1}{N}\sum_{i=1}^N \delta_{\bar{x}_{t-1}^i}(x_{t-1}) \approx p(x_{t-1} \mid y_{1:t-1})$.
- After propagation: $\frac{1}{N}\sum_{i=1}^N \delta_{x_t^i}(x_t) \approx p(x_t \mid y_{1:t-1})$.
- After weighting: $\sum_{i=1}^N w_t^i \delta_{x_t^i}(x_t) \approx p(x_t \mid y_{1:t})$.

Examples

An LG-SSM example (I/II)

Whenever you are working on a nonlinear inference method, always make sure that it solves the linear special case first!

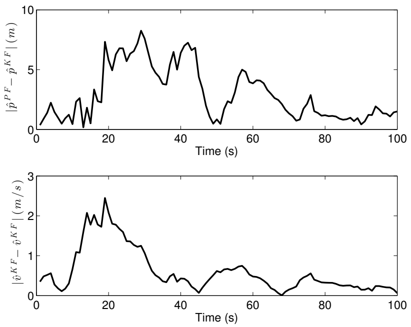
Consider the following LG-SSM (simple 1D positioning example)

$$\begin{pmatrix} X_t^{\text{pos}} \\ X_t^{\text{vel}} \\ X_t^{\text{acc}} \end{pmatrix} = \begin{pmatrix} 1 & T_s & T_s^2/2 \\ 0 & 1 & T_s \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X_{t-1}^{\text{pos}} \\ X_{t-1}^{\text{vel}} \\ X_{t-1}^{\text{acc}} \end{pmatrix} + \begin{pmatrix} T_s^3/6 \\ T_s^2/2 \\ T_s \end{pmatrix} V_t, \quad V_t \sim \mathcal{N}(0, Q),$$

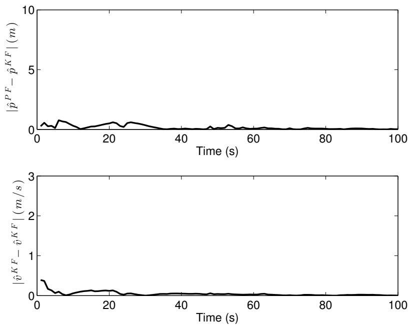
$$Y_t = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X_t^{\text{pos}} \\ X_t^{\text{vel}} \\ X_t^{\text{acc}} \end{pmatrix} + E_t, \qquad E_t \sim \mathcal{N}(0, R).$$

The Kalman filter provides the true filtering density, which implies that we can compare the PF to the truth in this case.

An LG-SSM example (II/II)



Using 200 particles.



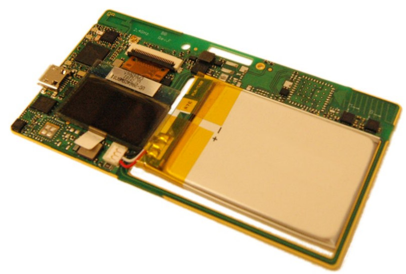
Using 20 000 particles.

The particle filter estimates converge as the number of particles tends to infinity (Lecture 5).

Nonlinear real-world application example

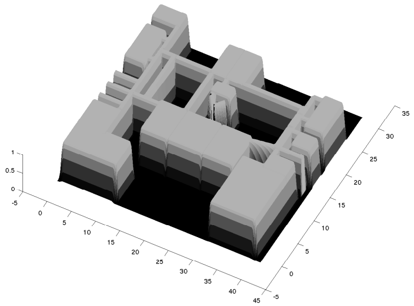
Aim: Compute the **position** of a person moving around indoors using sensors (inertial, magnetometer and radio) located in an ID badge, and a map.

The sensors (IMU and radio) and the DSP are mounted inside an ID badge.

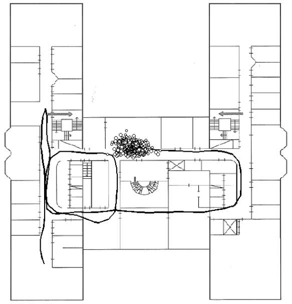


The inside of the ID badge.

Application – indoor localization (II/III)



"Likelihood model" for an office environment, the bright areas are rooms and corridors (i.e., walkable space).



An estimated trajectory and the particle cloud visualized at a particular instance.

Application – indoor localization (III/III)

Show movie

Johan Kihlberg, Simon Tegelid, Manon Kok and Thomas B. Schön. **Map aided indoor positioning using particle filters.** *Reglermöte (Swedish Control Conference)*, Linköping, Sweden, June 2014.

Use of random numbers in the particle filter

Random numbers are used to

1. **initialize**
2. **resample** and
3. **propagate**

the particles.

The weighting step does not require any new random numbers, it is just a function of already existing random numbers.

We can reason about and make use of the **joint probability distribution of these random variables**, from which the particle filter **generates one realization each time it is executed**.

A few concepts to summarize lecture 4

Bootstrap particle filter: A particle filter with a specific choice of proposals. Particles are simulated according to the dynamical model and weights are assigned according to the measurement likelihood.

Resampling: The procedure that generates a new set of particles $\{\bar{x}_{t-1}^i\}_{i=1}^N$ by sampling with replacement from among $\{x_{t-1}^j\}_{j=1}^N$, according to some weights $\{\nu_{t-1}^j\}_{j=1}^N$.

Ancestor indices: Random variable that are used to make the stochasticity of the resampling step explicit by keeping track of which particles that get resampled.