Solving sequential inference problems using sequential Monte Carlo

"Approximate a sequence of probability distributions on a sequence of probability spaces of increasing dimension."

UPPSALA UNIVERSITET

Thomas Schön
Division of Systems and Control
Department of Information Technology
Uppsala University.

Email: thomas.schon@it.uu.se,
www: `user.it.uu.se/~thosc112`

Goal of the course

The **goal of this course** is to introduce the sequential Monte Carlo (SMC) method and to hint at its (surprisingly) general applicability.

SMC is introduced as a solution to the state inference problem in nonlinear dynamical systems, focusing on the particle filter.

After this course you should be able to derive your own SMC algorithms allowing you to solve inference problems using SMC.

SMC – (abstract) problem formulation

The distribution of interest, $\pi(x)$ is called **target distribution**.

Problem formulation: Sample sequentially from a sequence of target distributions $\{\pi_t(x_{1:t})\}_{t\geq 1}$ of increasing dimension, where

$$\pi_t(x_{1:t}) = \frac{\gamma_t(x_{1:t})}{Z_t},$$

such that $\gamma_t(x_t) : \mathsf{X}^t \to \mathbb{R}^+$ is known pointwise and $Z_t = \int \pi(x_{1:t})\mathrm{d}x_{1:t}$ is computationally challenging.

1. Approximate the normalizing constant Z_t.
2. Compute integrals $\int \varphi(x_t)\pi_t(x_t)\mathrm{d}x_t$.

Important question: How general is this formulation?

SMC – underlying idea (abstract)

Idea underlying SMC: At each time t SMC delivers a set of N weighted samples (particles) $\{w_t^i, x_t^i\}_{i=1}^N$, approximating the target distribution

$$\widehat{\pi}_t(\cdot) = \sum_{i=1}^{N} w_t^i \delta_{x_{1:t}^i}(\cdot).$$

This empirical distribution converge asymptotically ($N \to \infty$) to π_t for any function φ,

$$\sum_{i=1}^{N} w_t^i \varphi(x_{1:t}^i) \to \underbrace{\int \varphi(x_t)\pi_t(x_{1:t})\mathrm{d}x_t}_{\mathrm{E}_{\pi_t(x_{1:t})}[\varphi(x_{1:t})]}.$$

As a byproduct SMC also produce an **unbiased** estimate of the normalizing constants.

Model – data – inference algorithm

In solving problems we have to make assumptions and a **model** will to a large extent capture many of these assumptions.

A **model** is a compact and interpretable representation of the data that is observed.

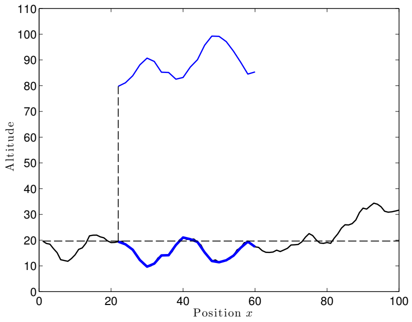
Using models to solve problems requires three key ingredients;

1. **Data:** Measurements from the system we are interested in.
2. **Model:** We use probabilistic models, allowing us to employ probability theory to represent and systematically work with the uncertainty that is inherent in most data.
3. **Inference algorithm:** The key topic of this course is SMC (introduced via the particle filter).

Particle filter – introductory example (I/III)

Consider a toy 1D localization problem.

Data



Model

Dynamic model:

$$x_{t+1} = x_t + u_t + v_t,$$

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.

The same idea has been used in many applications, see e.g.

Thomas Schön, Fredrik Gustafsson, and Per-Johan Nordlund. **Marginalized particle filters for mixed linear/nonlinear state-space models**. *IEEE Transactions on Signal Processing*, 53(7):2279-2289, July 2005.

UPPSALA UNIVERSITET

Particle filter – introductory example (II/III)

Task: Find the state x_t based on a set of measurements $y_{1:t} \triangleq \{y_1, \dots, y_t\}$ by computing the filter PDF $p(x_t \mid y_{1:t})$.

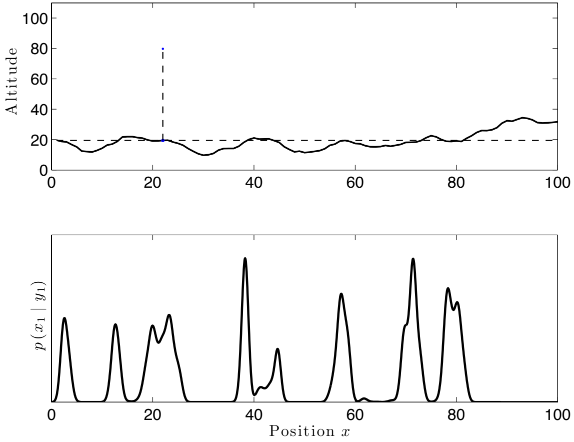
The particle filter (PF) maintains an approximation according to

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

that converge to the true filtering distribution as $N \to \infty$.

For intuition: Think of each particle as one simulation of the system state (in this example the horizontal position) and only keep the good ones.

Particle filter – introductory example (III/III)



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

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

Important message!

Given the computational tools we have today it is often rewarding to resist the linear Gaussian convenience!!

UPPSALA UNIVERSITET

Outline

The nonlinear SSM

A state space model (SSM) consists of a Markov process $\{x_t\}_{t\geq 1}$ that is indirectly observed via a measurement process $\{y_t\}_{t\geq 1}$,

$$
\begin{aligned}
x_{t+1} \mid x_t &\sim f_\theta(x_{t+1} \mid x_t, u_t), & \qquad x_{t+1} &= a_\theta(x_t, u_t) + v_{\theta,t},\\
y_t \mid x_t &\sim g_\theta(y_t \mid x_t, u_t), & y_t &= c_\theta(x_t, u_t) + e_{\theta,t},\\
x_1 &\sim \mu_\theta(x_1), & x_1 &\sim \mu_\theta(x_1),\\
(\theta &\sim p(\theta)), & (\theta &\sim p(\theta)),
\end{aligned}
$$

where $x_t \in \mathbb{R}^{n_x}$ denotes the state, $u_t \in \mathbb{R}^{n_u}$ denotes a known deterministic input signal, $y_t \in \mathbb{R}^{n_y}$ denotes the observed measurement and $\theta \in \Theta \subseteq \mathbb{R}^{n_\theta}$ denotes any unknown (static) parameters.

SSM as a probabilistic graphical model

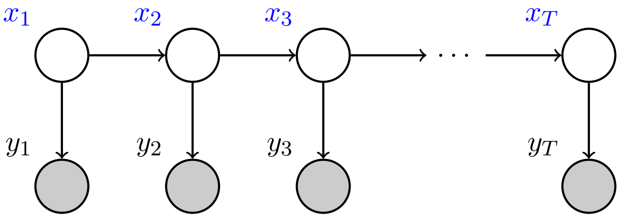


Figure: Graphical model for the SSM. Each stochastic variable is encoded using a node, where the nodes that are filled (gray) corresponds to variables that are observed and nodes that are not filled (white) are latent variables. The arrows pointing to a certain node encodes which variables the corresponding node are conditioned upon.

The SSM is an instance of a (directed) graphical model called **Bayesian network** or **belief network**.

The nonlinear SSM

State inference referes to the problem of finding information about the state(s) $x_{k:l}$ based on the available measurements $y_{1:t}$.

State inference in nonlinear SSMs is indeed one special case of the general problem of:

Sampling sequentially from a sequence of target distributions $\{\pi_t(x_{1:t})\}_{t \geq 1}$ *of increasing dimension, such that*

$$\pi_t(x_{1:t}) = \frac{\gamma_t(x_{1:t})}{Z_t},$$

where $\gamma_t(x_t) : \mathsf{X}^t \to \mathbb{R}^+$ *is known pointwise and* Z_t *is unknown.*

For example: $\pi_t(x_{1:t}) = p(x_{1:t} \mid y_{1:t})$, $\gamma_t(x_{1:t}) = p(x_{1:t}, y_{1:t})$, $Z_t = p(y_{1:t})$

Using SMC to infere SSMs

Recall: The sequence of target distributions $\{\pi(x_{1:t})\}_{t\geq 1}$ can be constructed in many different ways, explaining the generality and success of SMC.

The most basic construction arise from the SSM, where sequential structure of the target is inherent in the problem formulation.

$$
\begin{aligned}
\pi_1(x_1) &= p(x_1 \mid y_1), & Z_1 &= p(y_1), \\
\pi_2(x_{1:2}) &= p(x_{1:2} \mid y_{1:2}), & Z_2 &= p(y_{1:2}), \\
&\vdots & &\vdots \\
\pi_t(x_{1:t}) &= p(x_{1:t} \mid y_{1:t}), & Z_t &= p(y_{1:t}),
\end{aligned}
$$

UPPSALA UNIVERSITET

Our focus – the nonlinear filtering problem

State filtering problem: Recover information about the current state x_t based on the available measurements $y_{1:t}$, when

$$\begin{aligned} x_{t+1} \mid x_t &\sim f(x_{t+1} \mid x_t), \\ y_t \mid x_t &\sim g(y_t \mid x_t), \\ x_1 &\sim \mu(x_1). \end{aligned}$$

Strategy: Compute (an approximation of) the filtering PDF $p(x_t \mid y_{1:t})$.

Basics – probability

Let a and b be continuous random variables.

- Conditional probability:

$$p(a, b) = p(a \mid b)p(b)$$

- Marginalization (integrate out a variable)

$$p(a) = \int p(a, b)\mathrm{d}b$$

- Bayes' rule:

$$p(a \mid b) = \frac{p(b \mid a)p(a)}{p(b)}$$

The Markov property: $p(x_{t+1} \mid x_1, \dots, x_t) = p(x_{t+1} \mid x_t)$.

UPPSALA UNIVERSITET

The sequence of target distributions

The **measurement update**

$$p(x_t \mid y_{1:t}) = \frac{\overbrace{g(y_t \mid x_t)}^{\text{measurement}} \overbrace{p(x_t \mid y_{1:t-1})}^{\text{prediction pdf}}}{p(y_t \mid y_{1:t-1})},$$

and **time update**

$$p(x_t \mid y_{1:t-1}) = \int \underbrace{f(x_t \mid x_{t-1})}_{\text{dynamics}} \underbrace{p(x_{t-1} \mid y_{1:t-1})}_{\text{filtering pdf}} \mathrm{d}x_{t-1}.$$

Alternatively we can of course combine the two:

$$p(x_t \mid y_{1:t}) = \frac{g(y_t \mid x_t) \int f(x_t \mid x_{t-1}) p(x_{t-1} \mid y_{1:t-1}) \mathrm{d}x_{t-1}}{p(y_t \mid y_{1:t-1})}.$$

UPPSALA UNIVERSITET

Why do we need Monte Carlo methods?

In solving inference problems we are typically faced with various integration problems, which tend to be intractable and live in large dimensional spaces.

For example **expectation** arising in obtaining a point estimate. A commonly used point estimate is the conditional mean

$$\widehat{x}_{t\,|\,t} = \mathrm{E}\left[x_t \,|\, y_{1:t}\right] = \int x_t p(x_t \,|\, y_{1:t}) \mathsf{d}x_t.$$

Monte Carlo methods provides **computational solutions** where the distributions of interest are approximated by a large number of N random samples called particles.

Monte Carlo methods respects the model and the expressions we are trying to approximate.

UPPSALA UNIVERSITET

The Monte Carlo idea (I/II)

(Very) restrictive assumption: Assume that we have N samples $\{x^i\}_{i=1}^N$ from the target density $\pi(x)$,

$$\widehat{\pi}(x) = \sum_{i=1}^{N} \frac{1}{N} \delta_{x^i}(x)$$

Allows for the following approximation of the integral,

$$\mathrm{E}\left[\varphi(x)\right] = \int \varphi(x)\pi(x)\mathrm{d}x \approx \int \varphi(x) \sum_{i=1}^{N} \frac{1}{N} \delta_{x^i}(x)\mathrm{d}x = \frac{1}{N}\sum_{i=1}^{N} \varphi(x^i)$$

$$\text{"} \int + \delta \to \sum \text{"}$$

UPPSALA UNIVERSITET

The Monte Carlo idea (II/II)

The integral

$$I(\varphi(x)) \triangleq \mathrm{E}\left[\varphi(x)\right] = \int \varphi(x)\pi(x)\mathrm{d}x.$$

is approximated by

$$\widehat{I}_N(\varphi(x)) = \frac{1}{N}\sum_{i=1}^{N}\varphi(x^i).$$

The strong law of large numbers tells us that

$$\widehat{I}_N(\varphi(x)) \xrightarrow{\text{a.s.}} I(\varphi(x)), \qquad N \to \infty,$$

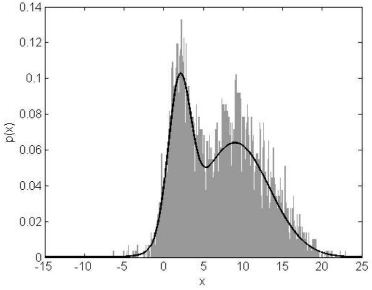
and the central limit theorem state that

$$\frac{\sqrt{N}\left(\widehat{I}_N(\varphi(x)) - I(\varphi(x))\right)}{\sigma_\varphi} \xrightarrow{\text{d}} \mathcal{N}\left(0,1\right), \qquad N \to \infty.$$

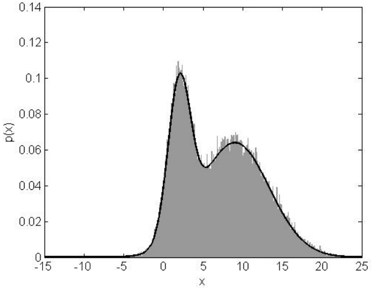
UPPSALA UNIVERSITET

The Monte Carlo idea – toy illustration

$$\pi(x) = 0.3\mathcal{N}(x \,|\, 2, 2) + 0.7\mathcal{N}(x \,|\, 9, 19)$$



5 000 samples



50 000 samples

Obvious problem: In general we are **not** able to directly sample from the density we are interested in.

Importance sampling – problem and idea

Importance sampling can be used to evaluate integrals of the form

$$I(\varphi(x)) = \mathrm{E}\left[\varphi(x)\right] = \int \varphi(x)\pi(x)\mathrm{d}x,$$

where it is hard to generate samples from the target density $\pi(x)$.

Note that:

$$\int \varphi(x)\pi(x)\mathrm{d}x = \int \varphi(x)\frac{\pi(x)}{q(x)}q(x)\mathrm{d}x.$$

Idea: Chose the **proposal** density $q(x)$ such that it is easy to generate samples from it and compensate for the mistmatch between the target and the proposal.

Importance sampling (I/II)

Problem: Generate samples distributed according to

$$\pi(x) = \frac{\gamma(x)}{Z}, \text{ where } Z = \int \gamma(x)\mathrm{d}x.$$

Equivalent formulation using a **proposal** density $q(x)$

$$\pi(x) = \frac{w(x)q(x)}{Z}, \text{ where } Z = \int w(x)q(x)\mathrm{d}x.$$

where the so-called importance weight is given by

$$w(x) = \frac{\gamma(x)}{q(x)}.$$

Importance sampling (II/II)

We are free to chose the proposal density as long as $\gamma(x) > 0 \Rightarrow q(x) > 0$.

1. Draw N samples $x^i \sim q(x), \;\; i = 1, \dots, N$.
2. Insert the Monte Carlo appr. $\widehat{q}(x) = \frac{1}{N}\sum_{i=1}^N \delta_{x^i}(x)$ into

$$\pi(x) = \frac{w(x)q(x)}{Z}, \text{ where } Z = \int w(x)q(x)\mathrm{d}x$$

results in

$$\widehat{\pi}(x) = \sum_{i=1}^N w^i \delta_{x^i}(x), \qquad \widehat{Z} = \frac{1}{N}\sum_{i=1}^N w(x^i),$$

where

$$w^i = \frac{w(x^i)}{\sum_{i=1}^N w(x^i)}, \qquad w(x^i) = \frac{\gamma(x^i)}{q(x^i)}.$$

Importance sampling (IS)

Algorithm 1 Importance sampler (IS)

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

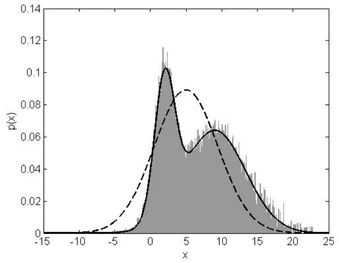
Each step is carried out for $i = 1, \dots, N$.

The convergence of the resulting approximation $\widehat{\pi}(x) = \sum_{i=1}^N w^i \delta_{x^i}(x)$ is since long well established.

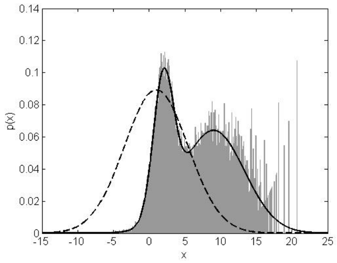
Sampling from a user-chosen proposal distribution q is corrected for by the weights, which **accounts for the discrepancy** between the proposal q and the target π.

UPPSALA UNIVERSITET

The importance of a good proposal density



$q_1(x) = \mathcal{N}(5, 20)$ (dashed curve)



$q_2(x) = \mathcal{N}(1, 20)$ (dashed curve)

$50\,000$ samples used in both simulations.

Lesson learned: It is important to be careful in selecting the proposal density.

Outline

1. Motivation and (a hopefully) intuitive introduction.
2. State inference in nonlinear state space models
3. Monte Carlo methods
   a) The idea
   b) Importance sampling
4. **Deriving a first particle filter (PF)**
5. Generic SMC sampler
6. Some of our current research activities (if there is time)

Using IS for our purposes

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

$$
\begin{aligned}
x_{t+1} \mid x_t &\sim f(x_{t+1} \mid x_t), \\
y_t \mid x_t &\sim g(y_t \mid x_t), \\
x_1 &\sim \mu(x_1).
\end{aligned}
$$

We have showed that the solution is

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

Relevant idea: Try to solve this using importance sampling!!

UPPSALA UNIVERSITET

Finding a proposal

Assume (in an "induction-like" fashion) that we at time $t-1$ have

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

allowing us to approximate the integral for $p(x_t \mid y_{1:t-1})$,

$$\begin{aligned}\widehat{p}(x_t \mid y_{1:t-1}) &= \int f(x_t \mid x_{t-1}) \sum_{i=1}^{N} w_{t-1}^{i} \delta_{x_{t-1}^{i}}(x_{t-1}) \mathrm{d}x_{t-1} \\ &= \sum_{i=1}^{N} w_{t-1}^{i} f(x_t \mid x_{t-1}^{i}).\end{aligned}$$

Idea: Use $\widehat{p}(x_t \mid y_{1:t-1})$ to guide the choice of proposal in an IS targeting the filtering PDF.

UPPSALA UNIVERSITET

Importance sampling reminder

```
Algorithm 2 Importance sampler
1. Sample $x^i \sim q(x)$.
2. Compute the weights $w(x^i) = \gamma(x^i)/q(x^i)$.
3. Normalize the weights $w^i = w(x^i)/\sum_{j=1}^N w(x^j)$.
```

Our proposal is

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

UPPSALA UNIVERSITET

Sampling from the proposal

Two step procedure to sample from mixture proposal $q(x_t \,|\, y_{1:t})$:

1. Select one of the components (**resampling**),

$$\mathbb{P}\left(\bar{x}_{t-1} = x_{t-1}^i \,\middle|\, \{x_{t-1}^j, w_{t-1}^j\}_{j=1}^N\right) = w_{t-1}^i.$$

2. Generate a sample from that component,

$$x_t \sim q(x_t \,|\, \bar{x}_{t-1}^i, y_t).$$

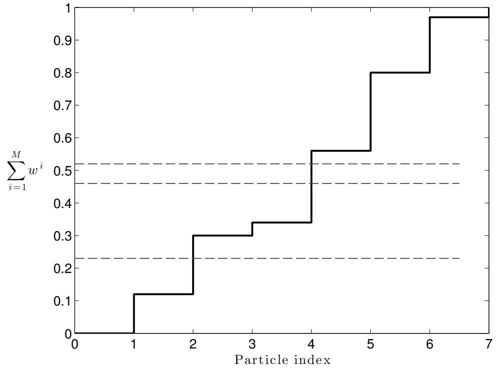
Repeat this N times.

Resampling (I/II)

Resampling is the procedure that (randomly) **turns a weighted** set of samples $\{x_{t-1}^i, w_{t-1}^i\}_{i=1}^N$ **into an unweighted** set of samples $\{\bar{x}_{t-1}^i, 1/N\}_{i=1}^N$ according to

$$\mathbb{P}\left(\bar{x}_{t-1} = x_{t-1}^i \,\middle|\, \{x_{t-1}^j, w_{t-1}^j\}_{j=1}^N\right) = w_{t-1}^i.$$

Resampling (II/II)



Illustrating how resampling works (using 7 particles).

1. Compute the cumulative sum of the weights.
2. Generate $u \sim \mathcal{U}[0, 1]$.

Three new samples are generated in the figure above, corresponding to sample $2, 4$ and 4.

Next step – computing the weights

```
Algorithm 3 Importance sampler
1. Sample $x^i \sim q(x)$.
2. Compute the weights $w(x^i) = \gamma(x^i)/q(x^i)$.
3. Normalize the weights $w^i = w(x^i)/\sum_{j=1}^N w(x^j)$.
```

Compute the weights

$$
\begin{aligned}
w_t(x_t^i) &= \frac{g(y_t \mid x_t)\widehat{p}(x_t \mid y_{1:t-1})}{q(x_t \mid y_{1:t})} \\
&= \frac{g(y_t \mid x_t) \sum_{j=1}^N w_{t-1}^j f(x_t \mid x_{t-1}^j)}{\sum_{j=1}^N w_{t-1}^j q(x_t \mid x_{t-1}^j)}
\end{aligned}
$$

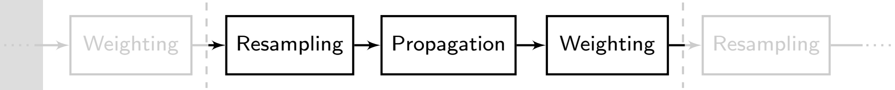
Computational complexity: $\mathcal{O}(N^2)$!

Result – A first particle filter

```
**Algorithm 4** Bootstrap particle filter (for $i = 1, \ldots, N$)
1. Initialization ($t = 1$):
  (a) Sample $x_1^i \sim \mu(x_1)$.
  (b) Compute the weights $\bar{w}_1^i = g(y_1 \mid x_1^i)$ and normalize,
      $w_1^i = \bar{w}_1^i / \sum_{j=1}^N \bar{w}_1^j$.
2. for $t = 2$ to $T$ do
  (a) Resample $\{x_{t-1}^i, w_{t-1}^i\}$ resulting in equally weighted particles
      $\{\bar{x}_{t-1}^i, 1/N\}$.
  (b) Propagate by sampling $x_t^i \sim f(x_t \mid \bar{x}_{t-1}^i)$.
  (c) Weight by computing $\bar{w}_t^i = g(y_t \mid x_t^i)$ and normalize
      $w_t^i = \bar{w}_t^i / \sum_{j=1}^N \bar{w}_t^j$.
```

UPPSALA UNIVERSITET

SMC structure



The structure is the same for all SMC algorithms. For the bootstrap PF we have,

Resampling: $\{x_{t-1}^i, w_{t-1}^i\}_{i=1}^N \to \{\bar{x}_{t-1}^i, 1/N\}_{i=1}^N$.

Propagation: $x_t^i \sim f(x_t \mid \bar{x}_{1:t-1}^i)$.

Weighting: $\bar{w}_t^i = W_t(x_t^i) = g(y_t \mid x_t^i)$ and normalize.

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

Important “design” considerations

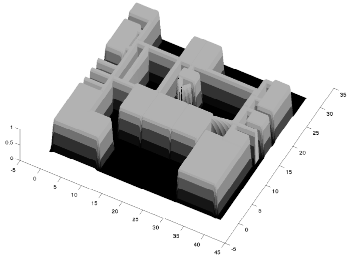
1. Adaptive resampling – only resample “when needed”.
2. Be careful when selecting the importance density.
3. Exploit analytically tractable sub-structures (Rao-Blackwellization).
4. ...

UPPSALA UNIVERSITET

Application – indoor localization (I/II)

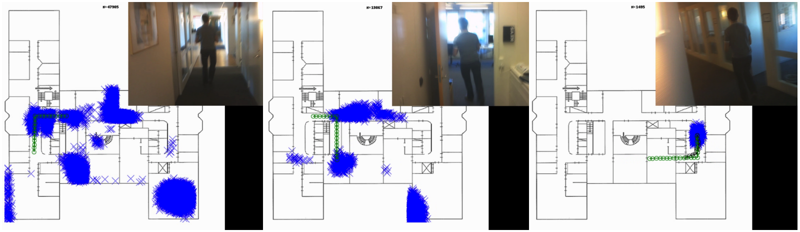
Aim: Compute the position of a person moving around indoors using sensors (inertial, magnetometer and radio) located in an ID badge and a map.

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



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

Application – indoor localization (II/II)



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.

Targeting the JSD instead

Our derivation of the PF is rather **non-standard**. The reason I like it is that it clearly shows why the resampling step is needed and where the need for the resampling step comes from.

The more **standard** way of deriving the PF is by **targeting the sequence of joint smoothing densities (JSD)** $\{p(x_{1:t} \mid y_{1:t})\}_{t\geq 1}$.

Enlightening derivation as well!! Shows that

SMC = SIS + Resampling

Problems?

Can you see any problems with the algorithm producing approximations of the JSD according to

$$p(x_{1:t} \mid y_{1:t}) = \sum_{i=1}^{N} w_t^i \delta_{x_{1:t}^i}(x_{1:t})$$

The resampling step remove particles with small weights and duplicate particles with large weights.

This results in **path degeneracy**, which we explain using a simple example.

Illustration of path degeneracy (I/II)

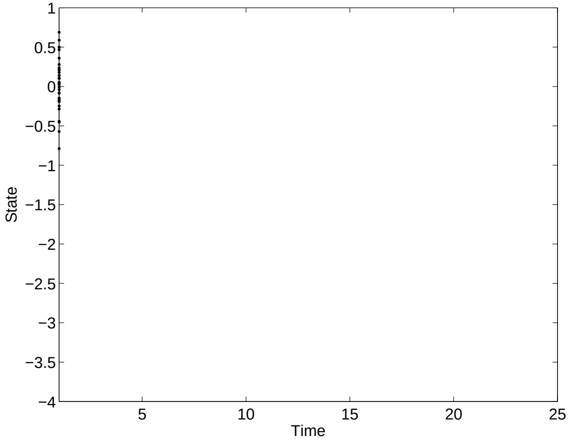
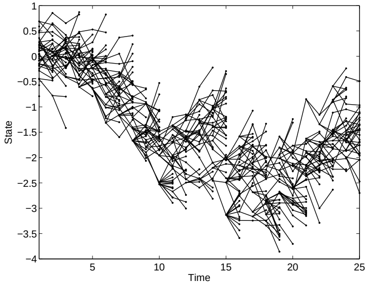
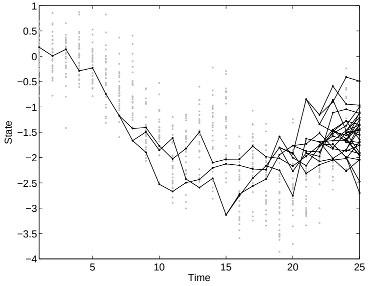


Illustration of path degeneracy (II/II)





Left plot: At each point in time all particles are plotted using a black dot and each particle is connected with its ancestor using a black line.

Right plot: The grey dots represents the $p(x_t \mid y_{1:t})$ at each point in time. The black lines shows the particle trajectories $\{x^i_{1:25}\}_{i=1}^{30}$ at time $t = 25$.

The right plot corresponds to the left plot with all trajectories that are not resampled removed (all particles are still visualized using gray dots).

UPPSALA UNIVERSITET

Mitigating the path degeneracy problem

This implies that if we are interested in the smoothing distribution

$$p(x_{1:T} \mid y_{1:T})$$

or some of its marginals we are **forced** to use different algorithms, which leads us to particle smoothers. **Backward simulation is key** here (and elsewhere!), for a self-contained tutorial, see

Fredrik Lindsten and Thomas B. Schön, **Backward simulation methods for Monte Carlo statistical inference**, *Foundations and Trends in Machine Learning*, 6(1):1-143, 2013.

Convergence results in one slide...

Let $\varphi : \mathsf{X} \mapsto \mathbb{R}$ be some test function of interest. The expectation

$$\mathrm{E}\left[\varphi(x_t) \mid y_{1:t}\right] = \int \varphi(x_t) p(x_t \mid y_{1:t}) \mathrm{d}x_t,$$

can be estimated by the particle filter

$$\widehat{\varphi}_t^N \triangleq \sum_{i=1}^{N} w_t^i \varphi(x_t^i).$$

The **CLT** governing the convergence of this estimator states

$$\sqrt{N}\left(\widehat{\varphi}_t^N - \mathrm{E}\left[\varphi(x_t) \mid y_{1:t}\right]\right) \xrightarrow{\mathrm{d}} \mathcal{N}(0, \sigma_t^2(\varphi)).$$

The **likelihood estimate** $\widehat{p}(y_{1:t}) = \prod_{s=1}^{t} \left\{ \frac{1}{N} \sum_{i=1}^{N} \bar{w}_s^i \right\}$ from the PF is **unbiased**, $\mathrm{E}_\psi\left[\widehat{p}(y_{1:t})\right] = p(y_{1:t})$ for any value of N and there are **CLTs available** as well.

Outline

1. Motivation and (a hopefully) intuitive introduction.
2. State inference in nonlinear state space modelsipho
3. Monte Carlo methods
   a) The idea
   b) Importance sampling
4. Deriving a first particle filter (PF)
5. **Generic SMC sampler**
6. Some of our current research activities (if there is time)

SMC – (abstract) problem formulation

The distribution of interest, $\pi(x)$ is called **target distribution**.

Problem formulation: Sample sequentially from a sequence of target distributions $\{\pi_t(x_{1:t})\}_{t\geq 1}$ of increasing dimension, where

$$\pi_t(x_{1:t}) = \frac{\gamma_t(x_{1:t})}{Z_t},$$

such that $\gamma_t(x_t) : \mathsf{X}^t \to \mathbb{R}^+$ is known pointwise and $Z_t = \int \pi(x_{1:t})\mathrm{d}x_{1:t}$ is computationally challenging.

So far we have seen that this formulation includes nonlinear SSMs, but the **important question** of the generality of formulation **remains**.

SMC

SMC is used to simulate from a sequence of probability distributions on a sequence of probability spaces of increasing dimension.

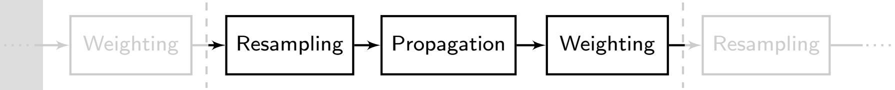
The target $\pi_t(x_{1:t})$ is a PDF on the **product space**

$$\mathsf{X}^t = \mathsf{X}_1 \times \mathsf{X}_2 \times \cdots \times \mathsf{X}_t.$$

SMC approximates the sequence of distributions $\pi_1, \pi_2, \dots, \pi_t$ using a set of N weighted particles,

$$\widehat{\pi}_t(\cdot) = \sum_{i=1}^{N} w_t^i \delta_{x_{1:t}^i}(\cdot).$$

SMC in words



Resampling: Focus the computation on the promising parts of the state space by pruning particles of low weight, while preserving the asymptotic guarantees of importance sampling.

Propagation: Sample a new successor state and append it to the earlier to form a sample from the t^{th} product space.

Weighting: The weights corrects for the discrepancy between the proposal q_t and the target π_t.

Doucet, A. and Johansen, A. M. (2011). **A tutorial on particle filtering and smoothing.** In Crisan, D. and Rozovskii, B., editors, *The Oxford Handbook of Nonlinear Filtering.* Oxford University Press.

Generic SMC sampler

```
Algorithm 5 Generic SMC sampler (for $i = 1, \dots, N$)
1. Initialization ($t = 1$):
    (a) Sample $x_1^i \sim q_1(x_1)$.
    (b) Compute the weights $\bar{w}_1^i = \gamma(x_1^i)/q_1(x_1^i)$ and normalize,
        $w_1^i = \bar{w}_1^i / \sum_{j=1}^N \bar{w}_1^j$.
2. for $t = 2$ to $T$ do
    (a) Resample $\{x_{1:t-1}^i, w_{t-1}^i\}$ resulting in equally weighted
        particles $\{\bar{x}_{1:t-1}^i, 1/N\}$.
    (b) Propagate by sampling $x_t^i \sim q_t(x_t \mid \bar{x}_{1:t-1}^i)$ and set
        $x_{1:t}^i = (\bar{x}_{1:t-1}^i, x_t^i)$.
    (c) Weight by computing $\bar{w}_t^i = \frac{\gamma_t(x_{1:t})}{\gamma_{t-1}(x_{1:t-1}) q_t(x_t \mid x_{1:t-1})}$ and
        normalize $w_t^i = \bar{w}_t^i / \sum_{j=1}^N \bar{w}_t^j$.
```

Generality of SMC

The sequence of target distributions $\{\pi_t(x_{1:t})\}_{t\geq 1}$ can be constructed in many different ways!

Two concrete examples:

1. When variables are **not** defined on product spaces, $\pi : \mathsf{X} \to \mathbb{R}^+$ we can introduce an artificial sequence of (auxiliary) distributions, where we are only interested in one of the marginals.

Del Moral, P., Doucet, A., and Jasra, A. (2006). **Sequential Monte Carlo samplers.** *Journal of the Royal Statistical Society: Series B*, 68(3):411-436.

2. Inference in probabilistic graphical models (PGM) is possible via such a sequence of auxiliary distributions. SMC provide consistent estimates and an unbiased estimate of the partition (normalization) constant (also for **loopy** PGMs!).

Christian A. Naesseth, Fredrik Lindsten and Thomas B. Schön, **Sequential Monte Carlo methods for graphical models**. *Advances in Neural Information Processing Systems (NIPS)* 27, Montreal, Canada, December, 2014.

Sequential Monte Carlo samplers

Suppose that the density of interest is defined over a space which is **not** a product space, say $\pi : \mathsf{X} \to \mathbb{R}^+$.

Key idea: Introduce auxiliary variables and transform this into a setup suitable for SMC using a sequence of auxiliary distributions. Typically we will only be interested in one of the marginals.

Sequential Monte Carlo samplers

Introduce a sequence of distributions

$$\pi_t(x_{1:t}) = \pi(x) \prod_{s=1}^{t-1} L_s(x_s \mid x_{s+1}),$$

defined on the product space

$$\mathsf{X}^t = \mathsf{X}_1 \times \mathsf{X}_2 \times \cdots \times \mathsf{X}_t.$$

L_s is a user-chosen backward kernel (e.g., an MCMC kernel).

$\pi_t(x_{1:t})$ admits $\pi(x)$ as a marginal **by construction**, effectively allowing it to be used as a surrogate for the actual target $\pi(x)$.

Using SMC within MCMC (PMCMC)

Particle MCMC (PMCMC) is a systematic way of combining SMC and MCMC.

Intuitively: SMC is used as a high-dimensional proposal mechanism on the space of state trajectories X^T.

A bit more precise (SSM special case): Construct a Markov chain with $p(\theta \mid y_{1:T})$ (or $p(\theta, x_{1:T} \mid y_{1:T})$) as its stationary distribution.

Christophe Andrieu, Arnaud Doucet and Roman Holenstein, **Particle Markov chain Monte Carlo methods**, *Journal of the Royal Statistical Society: Series B*, 72:269-342, 2010.

Fredrik Lindsten, Michael I. Jordan and Thomas B. Schön. **Particle Gibbs with ancestor sampling**. *Journal of Machine Learning Research (JMLR)*, 15:2145-2184, June 2014.

Goal of the course

The **goal of this course** is to introduce the sequential Monte Carlo (SMC) method and to hint at its (surprisingly) general applicability.

SMC is introduced as a solution to the state inference problem in nonlinear dynamical systems, focusing on the particle filter.

After this course you should be able to derive your own SMC algorithms allowing you to solve inference problems using SMC.

Some of our current research activities

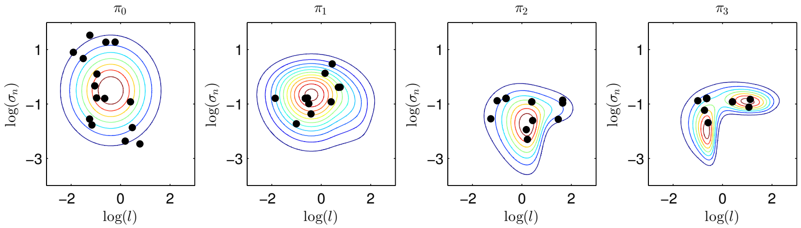
Joint work with (alphabetical order):

Christian A. Naesseth (Linköping University), **John Aston** (University of Cambridge), **Alexandre Bouchard-Côté** (University of British Columbia). **Johan Dahlin** (Linköping University), **Liang Dai** (Uppsala University), **Adam M. Johansen** (University of Warwick), **Michael I Jordan** (UC Berkeley), **Bonnie Kirkpatrick** (University of Miami), **Fredrik Lindsten** (University of Cambridge), **Andreas Svensson** (Uppsala University) and **Johan Wågberg** (Uppsala University).

Marginalizing hyperparameters in GPs

The hyperparameters encountered in the GP prior are often unknown, but they can still have a great influence on the posterior.

We offer a Bayesian approach, where the hyperparameters are **marginalized** (i.e. integrated out) using SMC.



Andreas Svensson, Johan Dahlin and Thomas B. Schön. **Marginalizing Gaussian process hyperparameters using sequential Monte Carlo**, *Preprint, arXiv:1502.01908*, February, 2015.

Nonlinear system identification

$$
\begin{aligned}
x_{t+1} \mid x_t &\sim f_\theta(x_{t+1} \mid x_t, u_t), \\
y_t \mid x_t &\sim g_\theta(y_t \mid x_t, u_t), \\
x_1 &\sim \mu_\theta(x_1), \\
(\theta &\sim p(\theta)).
\end{aligned}
\qquad
\begin{aligned}
x_{t+1} &= a_\theta(x_t, u_t) + v_{\theta,t}, \\
y_t &= c_\theta(x_t, u_t) + e_{\theta,t}, \\
x_1 &\sim \mu_\theta(x_1), \\
(\theta &\sim p(\theta)).
\end{aligned}
$$

Maximum likelihood

$$\widehat{\theta}_{\mathsf{ML}} = \arg\max_{\theta \in \Theta} p_\theta(y_{1:T}).$$

Bayesian

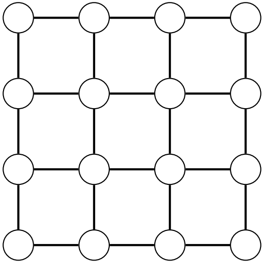
$$p(\theta \mid y_{1:T}) = \frac{p(y_{1:T} \mid \theta) p(\theta)}{p(y_{1:T})}.$$

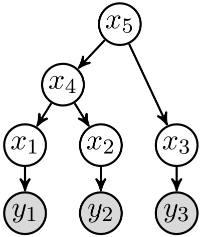
SMC provides a systematic way of exploring the state space.

Thomas B. Schön, Fredrik Lindsten, Johan Dahlin, Johan Wågberg, Christian A. Naesseth, Andreas Svensson and Liang Dai. **Sequential Monte Carlo methods for system identification**. *Preprint, arXiv:1503.06058*, March 2015.

Inference in probabilistic graphical models

Constructing an artificial sequence of intermediate target distributions for an SMC sampler is a powerful (and **quite possibly underutilized**) idea.





Christian A. Naesseth, Fredrik Lindsten and Thomas B. Schön, **Sequential Monte Carlo methods for graphical models**. *Advances in Neural Information Processing Systems (NIPS)* 27, Montreal, Canada, December, 2014.

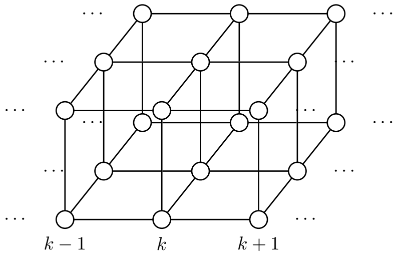
Fredrik Lindsten, Adam M. Johansen, Christian A. Naesseth, Bonnie Kirkpatrick, Thomas B. Schön, John Aston and Alexandre Bouchard-Côté. **Divide-and-Conquer with Sequential Monte Carlo**. *arXiv:1406.4993*, June 2014.

UPPSALA UNIVERSITET

SMC in high dimensions

The bootstrap PF suffers from weight collapse in high-dimensional settings.

This degeneracy can be reduced by using so-called **fully adapted** proposals.



We can mimic the efficient fully adapted proposals for arbitrary latent spaces and structures in high-dimensional models.

Approximations the proposal distribution and use a **nested coupling** of multiple SMC samplers and backward simulators.

Christian A. Naesseth, Fredrik Lindsten and Thomas B. Schön, **Nested sequential Monte Carlo**. *Preprint, arXiv:1502.02536*, February, 2015.

Conclusions

1. SMC approximates a sequence of probability distributions on a sequence of probability spaces of increasing dimension.
2. (Hopefully) conveyed the intuition underlying SMC.
3. SMC is applicable to **many** problems, not just SSMs via PF.

Exercises for the SMC module are available here,

`user.it.uu.se/~thosc112/courses.html`

Manuscript is also available (ask me for a draft if you want)

Thomas B. Schön and Fredrik Lindsten. **Learning of dynamical systems – Particle filters and Markov chain methods**, 2015.

Fast moving research area offering lots of opportunities!