Sequential Monte Carlo and a new visual tracker

Thomas Schön
Uppsala University
Sweden

Max Planck institute for intelligent systems
Tübingen, Germany, October 15, 2019.
Application – indoor localization using the magnetic field (I/II)

**Aim:** Compute the **position** using variations in the ambient magnetic field and the motion of the person (acceleration and angular velocities). All of this observed using sensors in a standard smartphone.

![Principle of magnetic terrain navigation. Here a pre-generated magnetic map is overlaid on top of a picture of the space.](image)

First we need a map, which we build using a tailored Gaussian process.

www.youtube.com/watch?v=enlMiUqPVJo


Show movie!

Aim: To provide intuition for the key mechanisms underlying sequential Monte Carlo (SMC), hint at a few ways in which SMC fits into the machine learning toolbox and show a new tracker.

Outline:

1. Introductory example
2. SMC for dynamical systems
3. SMC is a general method
4. Deep probabilistic regression
Representing a nonlinear dynamical systems

The state space model is a **Markov** chain that makes use of a **latent** variable representation to describe dynamical phenomena.

Consists of the unobserved (state) process \( \{x_t\}_{t \geq 0} \) modelling the dynamics and the observed process \( \{y_t\}_{t \geq 1} \) modelling the relationship between the measurements and the unobserved state process:

\[
x_t = f(x_{t-1}, \theta) + v_t,
\]
\[
y_t = g(x_t, \theta) + e_t.
\]
The full probabilistic model is given by

\[ p(x_{0:T}, \theta, y_{1:T}) = \prod_{t=1}^{T} p(y_t | x_t, \theta) \prod_{t=1}^{T} p(x_t | x_{t-1}, \theta) p(x_0 | \theta) p(\theta) \]

The nonlinear filtering problem involves the measurement update

\[ p(x_t | y_{1:t}) = \frac{p(y_t | x_t) p(x_t | y_{1:t-1})}{p(y_t | y_{1:t-1})} \]

and the time update

\[ p(x_t | y_{1:t-1}) = \int p(x_t | x_{t-1}) p(x_{t-1} | y_{1:t-1}) \, dx_{t-1}. \]
Sequential Monte Carlo (SMC)

The need for approximate methods (such as SMC) is tightly coupled to the intractability of the integrals above.

SMC provide approximate solutions to integration problems where there is a sequential structure present.

The particle filter approximates $p(x_t | y_{1:t})$ for

$$x_t = f(x_{t-1}) + v_t,$$
$$y_t = g(x_t) + e_t,$$

by maintaining an empirical distribution made up of $N$ samples (particles) $\{x_t^i\}_{i=1}^N$ and the corresponding weights $\{w_t^i\}_{i=1}^N$

$$
\widehat{p}(x_t | y_{1:t}) = \frac{\sum_{i=1}^N w_t^i}{\sum_{l=1}^N w_t^l} \delta_{x_t^i}(x_t).
$$
SMC – in words

1. **Propagation**: Sample a new successor state and append it to the earlier.

2. **Weighting**: The weights corrects for the discrepancy between the proposal distribution and the target distribution.

3. **Resampling**: Focus the computation on the promising parts of the state space by randomly pruning particles, while still preserving the asymptotic guarantees of importance sampling.
Sequential Monte Carlo (SMC) – abstract

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

(Abstract) problem formulation: Sample from a sequence of probability distributions $\{\pi_t(x_{0:t})\}_{t\geq 1}$ defined on a sequence of spaces of increasing dimension, where

$$\pi_t(x_{0:t}) = \frac{\widetilde{\pi}_t(x_{0:t})}{Z_t},$$

such that $\widetilde{\pi}_t(x_t) : \mathcal{X}^t \rightarrow \mathbb{R}^+$ is known point-wise and $Z_t = \int \pi(x_{0:t})dx_{0:t}$ is often computationally challenging.

SMC methods are a class of sampling-based algorithms capable of:

1. Approximating $\pi(x)$ and compute integrals $\int \varphi(x)\pi(x)dx$.
2. Approximating the normalizing constant $Z$ (unbiased).

**Important question:** How general is this formulation?
SMC is actually more general than we first thought

The sequence of target distributions \( \{ \pi_t(x_{1:t}) \}_{t=1}^n \) can be constructed in many different ways.

The most basic construction arises from chain-structured graphs, such as the state space model.

\[
\begin{align*}
\pi_t(x_{1:t}) &= \quad \tilde{\pi}_t(x_{1:t}) \\
\frac{p(x_{1:t} \mid y_{1:t})}{p(y_{1:t})} &= \frac{p(x_{1:t}, y_{1:t})}{p(y_{1:t})} \\
Z_t &= \int p(x_{1:t}) \, dx_{1:t}
\end{align*}
\]
SMC can be used for general graphical models

SMC methods are used to approximate a sequence of probability distributions on a sequence of spaces of increasing dimension.

**Key idea:**
1. Introduce a sequential decomposition of any probabilistic graphical model.
2. Each subgraph induces an intermediate target dist.
3. Apply SMC to the sequence of intermediate target dist.

SMC also provides an unbiased estimate of the normalization constant!

Going from classical SMC to D&C-SMC

The computational graph of classic SMC is a sequence (chain)

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{sequence_graph.png}
\caption{Computational flow of D&C-SMC. Each node corresponds to a target distribution.}
\end{figure}

D&C-SMC generalize the classical SMC framework from sequences to trees.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{tree_graph.png}
\caption{Tree decomposition of the target distribution.}
\end{figure}

Approximate Bayesian inference – blending

Deterministic methods
- Message passing
- Laplace’s method

Monte Carlo methods
- Variational inf.
- Markov chain Monte Carlo
- Sequential Monte Carlo

Figure 3: The trace plots of the first 50 steps using PMH0 (black), PMH1 (red) and PMH2 (blue). The dotted lines show the true parameters of the LGSS model. The gray contours show the log-posterior.
### Deterministic methods:

- **Good:** Accurate and rapid inference
- **Bad:** Results in biases that are hard to quantify

### Monte Carlo methods:

- **Good:** Asymptotic consistency, lots of theory available
- **Bad:** Can suffer from a high computational cost

Examples of freedom in the SMC algorithm that opens up for blending:

- The **proposal** distributions can be defined in many ways.
- The **intermediate target** distributions can be defined in many ways.

Leads to very interesting and useful algorithms.
Deep probabilistic regression
Commonly used deep regression approaches

**Regression:** Based on training data \( \{x_n, y_n\}_{n=1}^{N} \) learn a model that is capable of predicting the continuous \( y_n \) based on the input \( x_n \).

1. **Direct regression** Train the DNN to predict the outputs \( y = f_\theta(x) \) by minimizing a loss \( \ell(f_\theta(x_n), y_n) \).
   - Ex. \( L^2 \) loss, \( p(y \mid x, \theta) = \mathcal{N}(y \mid f_\theta(x), \sigma^2) \).

2. **Probabilistic regression** Let \( p(y \mid x, \theta) = p(y \mid \phi_\theta(x)) \), where the parameters \( \phi \) of a given family of probability distributions \( p(y \mid \phi) \) are provided by a DNN.
   - Ex. \( p(y \mid \phi) = \mathcal{N}(y \mid \mu_\theta(x), \sigma^2_\theta(x)) \), where the DNN outputs \( f_\theta(x) = \phi_\theta(x) = [\mu_\theta(x), \log \sigma^2_\theta(x)]^T \).

3. **Confidence-based regression** Use the DNN to predict a confidence score \( f_\theta(x, y) \in \mathbb{R} \) and let \( y^* = \arg \max_y f_\theta(x, y) \).

4. **Regression-by-classification** Discretize the output space \( \mathcal{Y} \) into a finite set of \( C \) classes and use standard classification techniques.
Our (simple and very general) construction

A general regression method with a clear probabilistic interpretation in the sense that we learn a model \( p(y \mid x, \theta) \) without requiring \( p(y \mid x, \theta) \) to belong to a particular family of distributions.

Let the DNN be a function \( f_\theta : \mathcal{X} \times \mathcal{Y} \to \mathbb{R} \) that maps an input-output pair \( \{x_n, y_n\} \) to a scalar value \( f_\theta(x_n, y_n) \in \mathbb{R} \).

Define the resulting (flexible) probabilistic model as

\[
p(y \mid x, \theta) = \frac{e^{f_\theta(x,y)}}{Z(x, \theta)}, \quad Z(x, \theta) = \int e^{f_\theta(x,y)} dy
\]
Learning flexible deep conditional target densities

1D toy illustration showing that we can learn multi-modal and asymmetric distributions, i.e. our model is flexible.

We train by maximizing the log-likelihood:

$$\max_{\theta} \sum_{n=1}^{N} \log p(y_n | x_n, \theta) = \max_{\theta} \sum_{n=1}^{N} - \log \left( \int e^{f_{\theta}(x_n, y)} dy \right) + f_{\theta}(x_n, y_n)$$

**Challenge:** Requires the normalization constant to be evaluated...

**Solution:** Monte Carlo! (via a simple importance sampling construction)
Experiments

Good results on four different computer vision (regression) problems:
1. Object detection, 2. Age estimation, 3. Head-pose estimation and
4. **Visual tracking**.

**Task (visual tracking):** Estimate a bounding box of a target object in
every frame of a video. The target object is defined by a given box in the
first video frame.

Show Movie!

Fredrik K. Gustafsson, Martin Danelljan, Goutam Bhat and TS. **DCTD: deep conditional target densities for accurate regression.**

SMC provide approximate solutions to integration problems where there is a sequential structure present.

- SMC is more general than we first thought.
- SMC can indeed be computationally challenging, but it comes with rather well-developed analysis and guarantees.
- There is still a lot of freedom waiting to be exploited.
- Constructed a practical deep flexible model for regression.

Forthcoming SMC introduction written with an ML audience in mind.