Software practical — Uncertainty Quantification

Implementation of a Sequential Monte Carlo Particle Filter Library in C++

Winter semester 2019

Submitted by: **Nils Friess**

February 15, 2020

Supervisors: Prof. Robert Scheichl, Gianluca Detommaso

Heidelberg University

Introduction

{sec:intro}

In this report we present an implementation of a Particle Filter in C++. Before discussing the actual implementation we give the theoretical details of the method in this section. We start by giving a brief overview of the method before discussing the individual steps in more detail.

We remark here, that the naming in these methods is highly ambiguous and varies greatly from author to author. We use – with some exceptions – the naming and notation used in [2] and highlight parts where the naming differs from other publications.

Overview of a Particle Filter

A Particle Filter is a Sequential Monte Carlo (SMC) method[1] that is used to estimate the state of a system that changes over time using only noisy and/ or partial observations of the system's state. This will be done in a Bayesian framework where one attempts to construct the posterior probability density function (pdf) of the state based on the observations. We make the following assumptions:

- The model describing the evolution of the internal state in time is available in a probablistic form.
- The model that relates the observations to the internal state is available in a probabilistic form.
- The observations are only available sequentially, not as a batch (ie. we assume that we receive new measurments sequentially in time).

Due to the last assumption we aim at a recursive method that does neither require to store nor to reprocess all the previous information when a new observation becomes available. To formalise the first two assumptions we will use the notion of *hidden markov models*.

[1]Some authors use the terms *Particle Filter* and *SMC method* synonymously. Doucet and Johansen develop in [2] a framework in which Particle Filters are only one specific method in the much broader class of SMC methods. They argue that this distinction allows for a better understanding of these methods. In this report, we are only interested in the filtering problem and will introduce it without discussing the more general notion of SMC methods as given by Doucet and Johansen.

To that end, we consider an $\mathbb{R}^{d_x}$-valued discrete-time Markov process $\{X_n\}_{n\in\mathbb{N}_0}$ such that

$$X_0 \sim \mu(x_0) \qquad \text{and} \qquad X_n \mid (X_{n-1} = x_{n-1}) \sim f(x_n \mid x_{n-1}), \quad n \geq 1, \tag{1}$$ {eq:hmm:1}

where $f(x \mid x')$ denotes the probability density associated with moving from state x' to x. This process models the internal state that we try to estimate. However, only the $\mathbb{R}^{d_y}$-valued process $\{Y_n\}_{n\in\mathbb{N}}$ is available. We assume that these observations are statistically independent and that

$$Y_n \mid (X_n = x_n) \sim g(y_n \mid x_n). \tag{2}$$ {eq:hmm:2}

Our goal is now to estimate the state of the system at some time n which can now be stated as: we want to estimate the distribution $p(x_n \mid y_{1:n})$, where $y_{1:n} := (y_1, y_2, \dots, y_n)$. This is often referred to as the *filtering problem* or *tracking*.[2]

In a Bayesian framework this is achieved in two steps: prediction and update. By the Chapman-Kolmogorov equation we first obtain

$$\begin{aligned} p(x_{k+1} \mid y_{1:k}) &= \int p(x_{k+1} \mid x_k, y_{1:k}) p(x_k \mid y_{1:k}) \,\mathrm{d}x_k \\ &= \int f(x_{k+1} \mid x_k) p(x_k \mid y_{1:k}) \,\mathrm{d}x_k, \end{aligned}$$

where we used the Markov property of the system, i. e.

$$p(x_{k+1} \mid x_k, y_{1:k}) = p(x_{k+1} \mid x_k) = f(x_{k+1} \mid x_k).$$

Once a new observation y_{k+1} becomes available we can update our beliefs about the system's state using Bayes' theorem

$$p(x_{k+1} \mid y_{1:k+1}) = \frac{p(y_{k+1} \mid x_{k+1}) p(x_{k+1} \mid y_{1:k})}{p(y_{k+1} \mid y_{1:k})},$$

where the normalising constant

$$\begin{aligned} p(y_{k+1} \mid y_{1:k}) &= \int p(y_{k+1} \mid x_{k+1}) p(x_{k+1} \mid y_{1:k}) \,\mathrm{d}x_{k+1} \\ &= \int g(y_{k+1} \mid x_{k+1}) p(x_{k+1} \mid y_{1:k}) \,\mathrm{d}x_{k+1} \end{aligned}$$

[2]Note that Doucet and Johansen *do not* call this the filtering problem [2]. They reserve this term for the estimation of the joint distributions $p(x_{1:n} \mid y_{1:n})$. Since we are only concerned with estimating the marginal distribution $p(x_n \mid y_{1:n})$ we will still refer to this problem as filtering.

depends on the likelihood function defined by the model in (2). Putting these two steps together, we obtain a recursive formula using the previous filtered state of the system $p(x_k \mid y_{1:k})$ and a new observation y_{k+1} to compute the current state of the system.

Only in a restrictive set of cases can the arising integrals be computed in a closed-form (e. g. when f and g are linear and the posterior of the system is Gaussian [1, p. 175] or when the underlying state space of the Markov model is finite, cf. [2, Example 1]). In a more general nonlinear non-Gaussian setting, approximative methods such as particle filters are necessary.

Particle Filtering Methods

The central idea of Particle filters is to represent the posterior probability density function (pdf) as a weighted set of *particles*.

References

[1] M Sanjeev Arulampalam et al. “A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking”. In: *IEEE Transactions on signal processing* 50.2 (2002), pp. 174–188.

[2] Arnaud Doucet and Adam M. Johansen. “A Tutorial on Particle Filtering and Smoothing: Fiteen years later”. In: *The Oxford handbook of nonlinear filtering* (2011). Ed. by Dan Crisan and Boris Rozovskii, pp. 656–705.