SMC Masterclass: Introduction to particle filters

Ömer Deniz Akyıldız[*,†]

[*]The Alan Turing Institute, London, UK.
[†]University of Cambridge, UK.
odakyildiz@turing.ac.uk

August 6, 2022

Abstract

Lecture notes for 2h long course I taught as a part of the Sequential Monte Carlo (SMC) Masterclass event held at University of Bristol, 30 March 2022-1 April 2022.

Contents

1 Introduction

In these notes, we aim at providing a comprehensive introduction to sequential Monte Carlo (SMC) methods for filtering and estimation problems.

Notation

We denote $x_{1:t} = (x_1, \ldots, x_t)$.

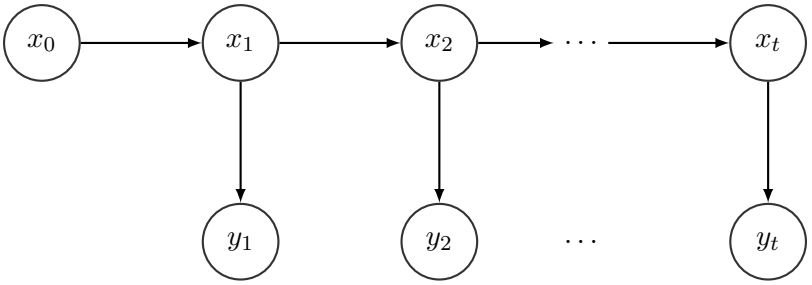

Figure 1: The conditional independence structure of a state-space model.

2 Background

In this section, we will introduce some essential tools to build the SMC methods we will introduce later.

2.1 State-space models

Consider a Markov process $(x_t)_{t\geq 0}$ defined on the measurable space $(\mathsf{X}, \mathcal{X})$ with $\mathsf{X} \subset \mathbb{R}^{d_x}$. This process denotes the signal of interest, e.g., the state of an object, the velocity field of a partial differential equation (PDE), hence the call it *the signal process*. Similarly, we define another sequence of random variables $(y_t)_{t\geq 1}$, defined on $\mathsf{Y} \subset \mathbb{R}^{d_y}$, to denote our observation sequence, or *the observation process*. This sequence denotes the observed data coming from the signal process and it can typically consist of noisy sensor measurements.

In order to formalise the probabilistic model, we define a state-space model (also called continuous-state hidden Markov model) is described by three distributions,

$$\begin{aligned} x_0 &\sim \mu(x_0) \\ x_t|x_{t-1} &\sim f(x_t|x_{t-1}), \\ y_t|x_t &\sim g(y_t|x_t), \end{aligned}$$

where μ is called the prior distribution, f is a Markov transition kernel defined on X, and g as the likelihood function. For convenience, we always assume the densities exist in this document but a general construction is possible.

2.2 The filtering problem

Given a sequence of observations, a typical problem is to estimate the conditional distributions of the signal process $(x_t)_{t\geq 0}$ given the observed data. We denote this distribution with $\pi_t(x_t|y_{1:t})$ which is called *the filtering distribution*. The problem of sequentially updating the sequence of filtering distributions $(\pi_t(x_t|y_{1:t}))_{t\geq 1}$ is called *the filtering problem*.

To introduce the idea intuitively, consider the scenario of tracking a target. We denote the states of the target with $(x_t)_{t\geq 0}$ which may include positions and velocities. We assume that the target moves in space w.r.t. f, i.e., the transition model of the target is given by $f(x_t|x_{t-1})$. Observations may consist of the locations of the target on $\mathbb{R}^2$ or power measurements with associated sensors (which may result in high-dimensional observations). At each time t, we receive a measurement vector y_t conditional on the true state of the system x_t. The likelihood of each observation is assumed to follow $g(y_t|x_t)$.

We now provide a simple recursion to demonstrate one possible solution to the filtering problem. Assume that we are given the distribution at time $t-1$ (to define our sequential

recursion) and would like to incorporate a recent observation y_t. One way to do so is to first perform *prediction*

$$\bar{\pi}_t(x_t|y_{1:t-1}) = \int f(x_t|x_{t-1})\pi_{t-1}(x_{t-1}|y_{1:t-1})\mathrm{d}x_{t-1}, \tag{1}$$

and obtain the predictive measure and then perform *update*

$$\pi_t(x_t|y_{1:t}) = \bar{\pi}_t(x_t|y_{1:t-1})\frac{g(y_t|x_t)}{p(y_t|y_{1:t-1})}, \tag{2}$$

where $p(y_t|y_{1:t-1}) = \int \bar{\pi}_t(x_t|y_{1:t-1})g(y_t|x_t)\mathrm{d}x_t$ is the incremental marginal likelihood.

Remark 1. We remark that the celebrated *Kalman filter* exactly implements recursions (1)–(2) in the case of[1]

$$\begin{aligned}\mu(x_0) &= \mathcal{N}(x_0; \mu_0, \Sigma_0),\\ f(x_t|x_{t-1}) &= \mathcal{N}(x_t; Ax_{t-1}, Q),\\ g(y_t|x_t) &= \mathcal{N}(y_t; Cx_t, R).\end{aligned}$$

For this Gaussian system, computing the integral (1) and the update (2) is analytically tractable, which results in Kalman filtering recursions of the mean and the covariance of the filtering distribution $\pi_t(x_t|y_{1:t})$. We skip the update rules of the Kalman filter, as our main aim is to focus on sequential Monte Carlo in this document.

Finally, we can move on to show how to update joint filtering distribution of the states $x_{0:t}$. To see this, note the recursion

$$\begin{aligned}\pi_t(x_{0:t}|y_{1:t}) &= \frac{\gamma(x_{0:t}, y_{1:t})}{p(y_{1:t})}\\ &= \frac{\gamma(x_{0:t-1}, y_{1:t-1})}{p(y_{1:t-1})}\frac{f(x_t|x_{t-1})g(y_t|x_t)}{p(y_t|y_{1:t-1})}\\ &= \pi_t(x_{0:t-1}|y_{1:t-1})\frac{f(x_t|x_{t-1})g(y_t|x_t)}{p(y_t|y_{1:t-1})}.\end{aligned}$$

This recursion will be behind the sequential Monte Carlo method we use for filtering in the next sections.

2.3 Importance sampling

Before we introduce the sequential Monte Carlo sampling for filtering, we introduce the basic importance sampling idea and its terminology. Assume that we aim at estimating expectations of a given density π, i.e., we would like to compute

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

We also assume that sampling from this density is not possible and we can only evaluate the *unnormalised* density $\gamma(x)$. One way to estimate this expectation is to sample from a proposal

[1]The variables (A, C, Q, R) can be time dependent, we consider the time-independent case for simplicity throughout.

measure q and rewrite the integral as

$$\begin{aligned}\pi(\varphi) &= \int \varphi(x)\pi(x)\mathrm{d}x, \\ &= \frac{\int \varphi(x)\frac{\gamma(x)}{q(x)}q(x)\mathrm{d}x}{\int \frac{\gamma(x)}{q(x)}q(x)\mathrm{d}x}, \\ &\approx \frac{\frac{1}{N}\sum_{i=1}^{N}\varphi(x^{(i)})\frac{\gamma(x^{(i)})}{q(x^{(i)})}}{\frac{1}{N}\sum_{i=1}^{N}\frac{\gamma(x^{(i)})}{q(x^{(i)})}}, \qquad x^{(i)} \sim q, \quad i = 1, \ldots, N. \end{aligned} \tag{3}$$

Let us know introduce the unnormalised weight function[2]

$$W(x) = \frac{\gamma(x)}{q(x)}. \tag{4}$$

With this, the Eq. (3) becomes

$$\begin{aligned}\pi^N(\varphi) &= \frac{\frac{1}{N}\sum_{i=1}^{N}\varphi(x^{(i)})W(x^{(i)})}{\frac{1}{N}\sum_{i=1}^{N}W(x^{(i)})}, \qquad x^{(i)} \sim q, \quad i = 1, \ldots, N, \\ &= \frac{\sum_{i=1}^{N}\varphi(x^{(i)})\mathsf{W}^{(i)}}{\sum_{i=1}^{N}\mathsf{W}^{(i)}}, \qquad x^{(i)} \sim q, \quad i = 1, \ldots, N,\end{aligned}$$

where $\mathsf{W}^{(i)} = W(x^{(i)})$ are called *the unnormalised weights*. Finally, we can obtain the estimator in a more convenient form,

$$\pi^N(\varphi) = \sum_{i=1}^{N}\mathsf{w}^{(i)}\varphi(x^{(i)}) = \pi^N(\varphi),$$

by introducing the *normalised importance weights*

$$\mathsf{w}^{(i)} = \frac{\mathsf{W}^{(i)}}{\sum_{i=1}^{N}\mathsf{W}^{(i)}}, \tag{5}$$

for $i = 1, \ldots, N$. We note that the particle approximation of π in this case is given as

$$\pi^N(\mathrm{d}x) = \sum_{i=1}^{N}\mathsf{w}^{(i)}\delta_{x^{(i)}}(\mathrm{d}x). \tag{6}$$

In the following section, we will derive the importance sampler aiming at building particle approximations of $\pi_t(x_{0:t}|y_{1:t})$ for a state-space model.

2.4 Importance sampling for state-space models: The emergence of the general particle filter

In this section, we simply derive an importance sampler for the joint filtering distribution $\pi_t(x_{0:t}|y_{1:t})$. We will see in the process that the particle filter is a special case of this conceptually simple importance sampler (defined just in many variables instead of one) and the infamous bootstrap particle filter is a further simplified case.

[2]More technically, these weights are the evaluations of the Radon-Nikodym derivative $W(x) = \frac{\mathrm{d}\gamma}{\mathrm{d}q}(x)$ (which, in this case, is just a ratio as we assume absolute continuity implicitly).

Let us assume that, in order to build an estimator of $\pi_t(x_{0:t}|y_{1:t})$, we have a proposal distribution over the entire path space $x_{0:t}$ denoted $q(x_{0:t})$. Note that, we also denote the unnormalised distribution of $x_{0:t}$ as $\gamma(x_{0:t}, y_{1:t})$ which is given as

$$\gamma(x_{0:t}, y_{1:t}) = \mu(x_0) \prod_{k=1}^{t} f(x_k|x_{k-1}) g(y_k|x_k). \tag{7}$$

This simply the joint distribution of all variables $(x_{0:t}, y_{1:t})$. Just as in the regular importance sampling case in eq. (4), we write

$$W_{0:t}(x_{0:t}) = \frac{\gamma(x_{0:t}, y_{1:t})}{q(x_{0:t})}.$$

Obviously, given samples from the proposal $x_{0:t}^{(i)} \sim q(x_{0:t})$, one can easily build the same weighted measure as in (6) on the path space by evaluating the weight $\mathsf{W}_{0:t}^{(i)} = W_{0:t}(x_{0:t}^{(i)})$ for $i = 1, \ldots, N$ and building a particle approximation

$$\pi^N(\mathrm{d}x_{0:t}) = \sum_{i=1}^{N} \mathsf{W}_{0:t}^{(i)} \delta_{x_{0:t}^{(i)}}(\mathrm{d}x_{0:t}).$$

However, this would be an undesirable scheme: We would need to store all variables in memory which is infeasible as t grows. Furthermore, with the arrival of a new observation y_{t+1}, this would have to be re-done, as this importance sampling procedure does not take into account the dynamic properties of the SSM. Therefore, implementing this sampler to build estimators sequentially is out of question.

Fortunately, we can design our proposal in certain ways so that this process can be done sequentially, starting from 0 to t. Furthermore, this would allow us to run the filter *online* and incorporate new observations. The clever choices of the proposal here lead to a variety of different *particle filters* as we shall see next. Let us consider a decomposition of the proposal

$$q(x_{0:t}) = q(x_0) \prod_{k=1}^{t} q(x_k|x_{1:k-1}).$$

Note that, based on this, we can build a recursion for the function $W(x_{0:t})$ by writing

$$\begin{aligned} W_{0:t}(x_{0:t}) &= \frac{\gamma(x_{0:t}, y_{1:t})}{q(x_{0:t})}, \\ &= \frac{\gamma(x_{0:t-1}, y_{1:t-1})}{q(x_{0:t-1})} \frac{f(x_t|x_{t-1}) g(y_t|x_t)}{q(x_t|x_{0:t-1})}, \\ &= W_{0:t-1}(x_{0:t-1}) \frac{f(x_t|x_{t-1}) g(y_t|x_t)}{q(x_t|x_{0:t-1})}, \\ &= W_{0:t-1}(x_{0:t-1}) W_t(x_{0:t}). \end{aligned} \tag{8}$$

That is, under this scenario, the weights can be computed *recursively* – given the weights of time $t-1$, one can evaluate $W_{0:t}(x_{0:t})$ and update the weights. However, this would not solve the infeasibility problem mentioned earlier, as the cost of evaluating using the whole path of samples is still out of question. Finally, to remedy this, we can further simplify our proposal

$$q(x_{0:t}) = q(x_0) \prod_{k=1}^{t} q(x_k|x_{k-1}).$$

by removing dependence to the past, essentially choosing a Markov process as a proposal. This allows us to obtain purely recursive weight computation

$$W_{0:t}(x_{0:t}) = \frac{\gamma(x_{0:t}, y_{1:t})}{q(x_{0:t})}, \tag{9}$$

$$= \frac{\gamma(x_{0:t-1}, y_{1:t-1})}{q(x_{0:t-1})} \frac{f(x_t|x_{t-1})g(y_t|x_t)}{q(x_t|x_{t-1})}, \tag{10}$$

$$= W_{0:t-1}(x_{0:t-1}) \frac{f(x_t|x_{t-1})g(y_t|x_t)}{q(x_t|x_{t-1})}, \tag{11}$$

$$= W_{0:t-1}(x_{0:t-1}) W_t(x_t, x_{t-1}), \tag{12}$$

using only the samples from time $t-1$ and time t. The advantage of this scheme is explicit in the notation: Note that the final weight function W_t only depends on (x_t, x_{t-1}), but not the whole past as in (8). The function $W_t(x_t, x_{t-1})$ is called the incremental weight function.

2.4.1 Sequential importance sampling

We can now see how the one-step update of this sampler works given a new observation. Assume that we have computed the unnormalised weights $\mathsf{W}_{1:t-1}^{(i)} = W(x_{0:t-1}^{(i)})$ recursively and obtained samples $x_{0:t-1}^{(i)}$. As we mentioned earlier, we only need the last sample $x_{t-1}^{(i)}$ to obtain the weight update given in (12). And also note that $\mathsf{W}_{1:t-1}^{(i)}$ for $i = 1, \ldots, N$ are just numbers, they do not need the storage of previous samples. Given this, we can now sample from the Markov proposal $x_t^{(i)} \sim q(x_t|x_{t-1}^{(i)})$ and compute the weights of the path sampler at time t as

$$\mathsf{W}_{1:t}^{(i)} = \mathsf{W}_{1:t-1}^{(i)} \times \mathsf{W}_t^{(i)},$$

where

$$\mathsf{W}_t^{(i)} = \frac{f(x_t^{(i)}|x_{t-1}^{(i)})g(y_t|x_t^{(i)})}{q(x_t^{(i)}|x_{t-1}^{(i)})}.$$

What we described in other words is that, given the samples $x_{t-1}^{(i)}$, we first perform sampling step

$$x_t^{(i)} \sim q(x_t|x_{t-1})$$

, and then compute

$$\mathsf{W}_t^{(i)} = \frac{f(x_t^{(i)}|x_{t-1}^{(i)})g(y_t|x_t^{(i)})}{q(x_t^{(i)}|x_{t-1}^{(i)})}.$$

and update

$$\mathsf{W}_{1:t}^{(i)} = \mathsf{W}_{1:t-1}^{(i)} \times \mathsf{W}_t^{(i)}.$$

These are unnormalised weights and we normalise them to obtain,

$$\mathsf{w}_{1:t}^{(i)} = \frac{\mathsf{W}_{1:t}^{(i)}}{\sum_{i=1}^{N} \mathsf{W}_{1:t}^{(i)}},$$

Algorithm 1 Sequential Importance Sampling (SIS)

1: Sample $x_0^{(i)} \sim q(x_0)$ for $i = 1, \ldots, N$.
2: **for** $t \geq 1$ **do**
3: Sample: $x_t^{(i)} \sim q(x_t | x_{t-1}^{(i)})$,
4: Compute weights:

$$\mathsf{W}_t^{(i)} = \frac{f(x_t^{(i)}|x_{t-1}^{(i)}) g(y_t|x_t^{(i)})}{q(x_t^{(i)}|x_{t-1}^{(i)})}.$$

and update

$$\mathsf{W}_{1:t}^{(i)} = \mathsf{W}_{1:t-1}^{(i)} \times \mathsf{W}_t^{(i)}.$$

Normalise weights,

$$\mathsf{w}_{1:t}^{(i)} = \frac{\mathsf{W}_{1:t}^{(i)}}{\sum_{i=1}^N \mathsf{W}_{1:t}^{(i)}}.$$

5: Report

$$\pi_t^N(\mathrm{d}x_{0:t}) = \sum_{i=1}^N \mathsf{w}_{1:t}^{(i)} \delta_{x_{0:t}^{(i)}}(\mathrm{d}x_{0:t}).$$

6: **end for**

which finally leads to the empirical measure,

$$\pi^N(\mathrm{d}x_{0:t}) = \sum_{i=1}^N \mathsf{w}_{1:t}^{(i)} \delta_{x_{0:t}^{(i)}}(\mathrm{d}x_{0:t}).$$

The full scheme is given in Algorithm 1. This method is called sequential importance sampling (SIS). This is not very popular in the literature due to the well known *weight degeneracy* problem. We next introduce a resampling step to this method and will obtain the first particle filter in this lecture.

2.4.2 Sequential importance sampling with resampling: The general particle filter

We finally describe the general particle filter by extending the above method with a resampling step employed after the weighting step. We will show in a practical session that the SIS method without resampling easily degenerates, i.e., after some time, only a single weight approximates to 1 and others to 0, rendering the method a point estimate. To keep the particle diversity, a resampling method is introduced in between weighting and sampling steps. This step does not introduce a systematic bias, although, it adds additional terms to the overall L_p error.

With the additional resampling step, the sequential importance sampling with resampling (SISR) takes the form given in Algorithm 2. We note that, effectively, resampling step sets $\mathsf{W}_{1:t-1}^{(i)} = 1/N$ for $i = 1, \ldots, N$. Therefore, we only need to compute the last incremental weight and weight our particles with the current weight. Also, note that the resampling step does introduce extra error but does not induce bias, since moments of π_t^N does not change.

Algorithm 2 Sequential Importance Sampling with Resampling (SISR)

1: Sample $x_0^{(i)} \sim q(x_0)$ for $i = 1, \ldots, N$.
2: **for** $t \geq 1$ **do**
3: Sample: $\bar{x}_t^{(i)} \sim q(x_t | x_{t-1}^{(i)})$,
4: Compute weights:

$$\mathsf{W}_t^{(i)} = \frac{f(\bar{x}_t^{(i)} | x_{t-1}^{(i)}) g(y_t | \bar{x}_t^{(i)})}{q(\bar{x}_t^{(i)} | x_{t-1}^{(i)})}.$$

Normalise weights,

$$\mathsf{w}_t^{(i)} = \frac{\mathsf{W}_t^{(i)}}{\sum_{i=1}^N \mathsf{W}_t^{(i)}}.$$

5: Report

$$\pi_t^N(\mathrm{d}x_t) = \sum_{i=1}^N \mathsf{w}_t^{(i)} \delta_{\bar{x}_t^{(i)}}(\mathrm{d}x_t).$$

6: Resample:

$$x_t^{(i)} \sim \sum_{i=1}^N \mathsf{w}_t^{(i)} \delta_{\bar{x}_t^{(i)}}(\mathrm{d}x_t).$$

7: **end for**

2.4.3 The bootstrap particle filter

In the general particle filter, the proposal $q(x_t|x_{t-1})$ is a design choice to be made and this depends on our specific knowledge of a good proposal for a given system. For example, one can incorporate future observations into this proposal in an ad-hoc or use the proposal choices like in the auxiliary particle filter (APF).

A generic choice exists, however, that is simply setting $q(x_t^{(i)}|x_{t-1}^{(i)}) = f(x_t^{(i)}|x_{t-1}^{(i)})$, i.e., using the transition density of the SSM under consideration as a proposal. The algorithm simplifies considerably in this case and the resulting method is called the bootstrap particle filter (BPF) which is given in Alg. 3. This algorithm has multiple appealing intuitive explanations beyond the derivation we provided based on importance sampling here. It can be most generally thought as an evolutionary method. To uncover some of this intuition, see Fig. 2.

To elaborate the interpretation, consider a set of particles $x_{t-1}^{(i)}$ representing the state of the system at time $t-1$. If our state-space transition model $f(x_t|x_{t-1})$ is well-specified (that is, if the underlying system we aim at tracking does indeed move according to f), then the first intuivite step we can do to predict where the state would be at time t would be to move particles according to f, that is sampling $\bar{x}_t^{(i)} \sim f(x_t|x_{t-1}^{(i)})$ which is the first step of the BPF. This gives us a predictive distribution which consists of $\bar{x}_t^{(i)}$ for $i = 1, \ldots, N$. The prediction step (naturally) does not require to observe the data point at y_t. Once we observe the data point y_t, we can then use this data point to evaluate a fitness measure for our particles. In other words, if a predictive particle $\bar{x}_t^{(i)}$ is a good fit to the observation, we would expect its likelihood $g(y_t|\bar{x}_t^{(i)})$

Algorithm 3 Bootstrap particle filter (BPF)

1: Sample $x_0^{(i)} \sim q(x_0)$ for $i = 1, \ldots, N$.
2: **for** $t \geq 1$ **do**
3: Sample: $\bar{x}_t^{(i)} \sim f(x_t | x_{t-1}^{(i)})$,
4: Compute weights:

$$\mathsf{W}_t^{(i)} = g(y_t | \bar{x}_t^{(i)}).$$

Normalise weights,

$$\mathsf{w}_t^{(i)} = \frac{\mathsf{W}_t^{(i)}}{\sum_{i=1}^N \mathsf{W}_t^{(i)}}.$$

5: Report

$$\pi_t^N(\mathrm{d}x_t) = \sum_{i=1}^N \mathsf{w}_t^{(i)} \delta_{\bar{x}_t^{(i)}}(\mathrm{d}x_t).$$

6: Resample:

$$x_t^{(i)} \sim \sum_{i=1}^N \mathsf{w}_t^{(i)} \delta_{\bar{x}_t^{(i)}}(\mathrm{d}x_t).$$

7: **end for**

to be high. Otherwise, this likelihood would be low. Thus, it intuitively makes sense to use our likelihood evaluations as "weights", that is to compute a measure of fitness for each particle. That is exactly what the BPF does at the second step by computing weights using the likelihood evaluations. The final step is then to use these relative weights to *resample* – a step that is used to refine the cloud of particles we have. Simply, the resampling step removes some of the particles with low weights (that are bad fits to the observation) and regenerates the particles with high weights.

The connection to evolutionary terms are clearer within this interpretation. The sampling step in the BPF can be seen as "mutation" that introduces changes to an individual particle according to some mutation mechanism (in our case, the dynamics). Then, weighting and resampling correspond to "selection" step, where individual particles are evaluated w.r.t. a fitness measure coming from the environment (defined by an observation) and individuals are reproduced in a random manner w.r.t. their fitness.

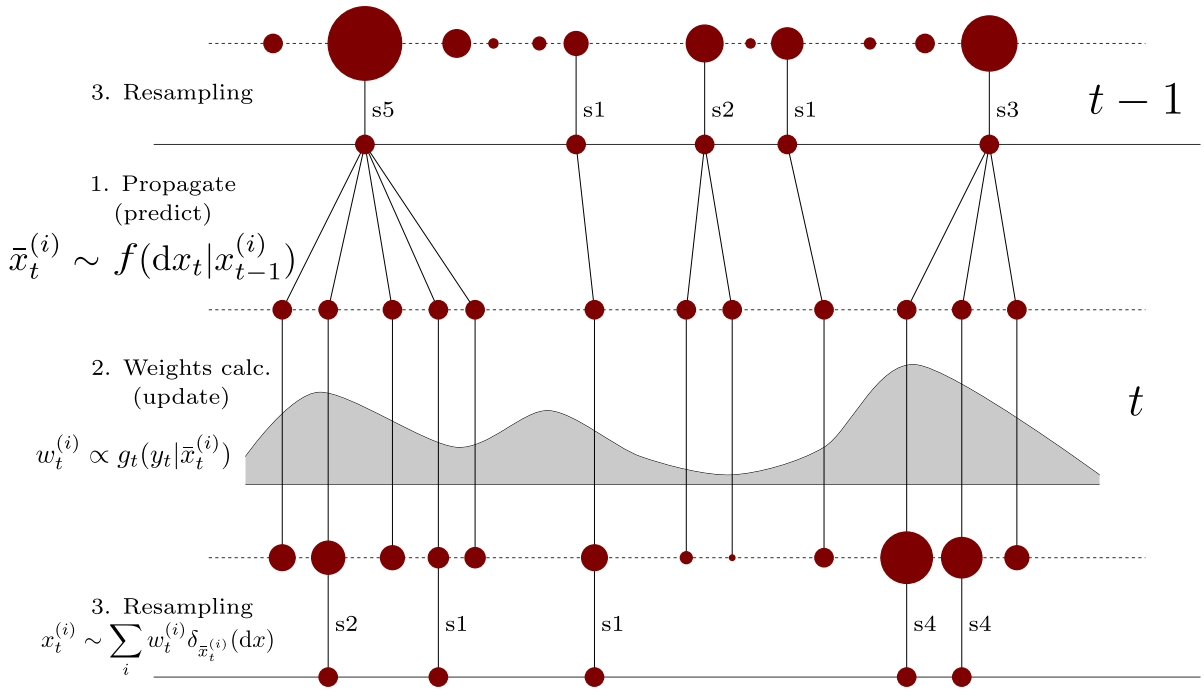

Figure 2: Intuitive model of BPF (Figure courtesy Victor Elvira).

References

Christophe Andrieu, Arnaud Doucet, and Roman Holenstein. Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72 (3):269–342, 2010.

Alexander Buchholz, Nicolas Chopin, and Pierre E Jacob. Adaptive tuning of Hamiltonian Monte Carlo within sequential Monte Carlo. *Bayesian Analysis*, 16(3):745–771, 2021.

Nicolas Chopin. A sequential particle filter method for static models. *Biometrika*, 89(3):539–552, 2002.

Nicolas Chopin. Central limit theorem for sequential Monte Carlo methods and its application to Bayesian inference. *The Annals of Statistics*, 32(6):2385–2411, 2004.

Nicolas Chopin and James Ridgway. Leave Pima Indians alone: binary regression as a benchmark for Bayesian computation. *Statistical Science*, 32(1):64–87, 2017.

Nicolas Chopin, Pierre E Jacob, and Omiros Papaspiliopoulos. Smc2: an efficient algorithm for sequential analysis of state space models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 75(3):397–426, 2013.

Chenguang Dai, Jeremy Heng, Pierre E Jacob, and Nick Whiteley. An invitation to sequential Monte Carlo samplers. *arXiv preprint arXiv:2007.11936*, 2020.

Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(3):411–436, 2006.

Simon J Godsill, Arnaud Doucet, and Mike West. Monte carlo smoothing for nonlinear time series. *Journal of the american statistical association*, 99(465):156–168, 2004.

Lawrence M Murray, Anthony Lee, and Pierre E Jacob. Parallel resampling in the particle filter. *Journal of Computational and Graphical Statistics*, 25(3):789–805, 2016.

Sinan Yildirim. *Maximum likelihood parameter estimation in time series models using sequential Monte Carlo*. PhD thesis, University of Cambridge, 2013.