Smoothing methods for particle filters in tracking applications

Joel Nulsen[1], Paul Baxter[2] and Trevor Wood[2,*]

Abstract – **In the use of particle filters to estimate a target's location, smoothing can be used to refine state estimation using future data. In this paper we present two established smoothing algorithms, the Sequential Fixed-Lag Smoother (SFLS) and the Backwards Simulation Particle Smoother (BS-PS). While SFLS can run in real time (although with a fixed delay), BS-PS can be excessively computationally expensive. We propose a novel combination of BS-PS and SFLS which requires significantly less computation in exchange for a small reduction in accuracy. The performance and complexity of all four methods, including the particle filter with no smoothing and the combined smoother, are compared both on simulated truth data and on real data with known waypoints for an indoor tracking system using a combination of inertial sensor data and a building map.**

I. INTRODUCTION

In tracking problems where noisy measurements are available, or fusion of disparate incomplete data sources is required, discrete time Bayesian filtering techniques are often applied to give an estimation of the target state in the form of a probability distribution. In the filtering problem, this estimation is based on all measurements collected up to the current time. In sufficiently simple situations where the relevant equations are linear and all noise can be assumed to be white and Gaussian, analytic methods such as the Kalman filter exist to give optimal solutions – see for example Chapter 4 of [1].

However, many real-world problems have complex state spaces and nonlinear process equations which render such analytic approaches intractable. In these instances, sequential Monte Carlo methodologies known as particle filters are often used. Particle filters attempt to approximate the desired probability distribution using a set of N particles, each with an associated importance weight.

The state of particle i at time step k is written $\boldsymbol{x}_k^{(i)}$, for $i = 1, \dots, N$ and $k = 1, \dots, T$. At each time step, the process noise $\boldsymbol{v}_k^{(i)}$ is sampled from some known distribution (typically Gaussian) and the particle is propagated according to the forwards model

$$\boldsymbol{x}_{k+1}^{(i)} = \boldsymbol{f}\left(\boldsymbol{x}_k^{(i)}, \boldsymbol{v}_k^{(i)}\right), \tag{1}$$

where $\boldsymbol{f}$ is in general some nonlinear function taking values in the state space. The importance weights $w_k^{(i)}$ are typically assigned based on the observations made up to time k. The exact way in which this is done depends on the particular filtering algorithm being used, but they must satisfy the normalisation condition $\sum_{i=1}^{N} w_k^{(i)} = 1$. The approximation to the target state estimation distribution at time k is then given by $\sum_{i=1}^{N} w_k^{(i)}\, \delta\left(\boldsymbol{x}_k^{(i)}\right)$, with δ the Dirac delta function.

Most particle filters also employ resampling to maintain particle diversity when too many particles are given zero (or almost zero) weight. The filter draws N new particles from the old particle sates, with the probability of choosing a particular state given by its weight.

In this paper we focus on the smoothing problem of estimation based on future observations. Both the Sequential Fixed-Lag Smoother and the Backwards Simulation Particle Smoother work in conjunction with a particle filter by using the particle states at future times to select a subset (or equivalently re-weight) the particles at the present time step – they do not use the future observations directly. Because of this, their performance is heavily dependent on that of the underlying particle filter. Improvements in performance most often arise in situations where particles become erroneous, clearly diverging from the target state, and are subsequently discarded/de-weighted by the particle filter. Smoothing techniques can allow such particles to be eliminated sooner (SFLS) or even altogether (BS-PS).

Sections II and III present the SFLS and BS-PS algorithms along with notes on their performance and complexity. Section IV proposes and outlines a novel combination of these two algorithms. Section V describes the experimental setups used in detail, and finally Sections VI and VII present the results of these experiments in terms of accuracy and computational cost respectively. A brief discussion of suitability of the techniques discussed for various tracking applications along is given in Section VIII.

[1] Centre for Mathematical Sciences, University of Cambridge, Wilberforce Road, Cambridge, CB3 0WA
[2] Cambridge Consultants, Science Park, Milton Road, Cambridge, CB4 0DW
* Corresponding Author – trevor.wood@cambridgeconsultants.com

II. SEQUENTIAL FIXED-LAG SMOOTHER

A. Algorithm and notes

There are several SFLS algorithms in existence; a simple version is presented here, along with a brief statistical justification.

In the particle filter framework, a particle $\boldsymbol{x}_{k+1}^{(j)}$ can be said to be "descended" from a unique particle $\boldsymbol{x}_k^{(i)}$ if it was produced either by progressing $\boldsymbol{x}_k^{(i)}$ forwards according to (1), or by resampling and drawing the state occupied by $\boldsymbol{x}_k^{(i)}$. Following this thread backwards in time, we can define the "ancestor" of $\boldsymbol{x}_{k+1}^{(j)}$ at time t for all $0 \leq t \leq k$ inductively in the obvious way.

An SFLS has a fixed delay between observations and estimates; suppose the delay is L time steps. Then when the particle filter reaches step $k+L$, our smoothing algorithm updates the weights of particles at step k according to

$$w_{k|k+L}^{(i)} = \sum_{j \leftarrow i} w_{k+L}^{(j)}, \tag{2}$$

where the notation $j \leftarrow i$ may be read as "j descended from i", and $w_{k|k+L}^{(i)}$ is the importance weight of particle i at time k conditioned on observations up to time $k+L$. In other words, the updated weight of a particle at time k is determined by the number and weights of its descendants at time $k+L$.

In particular, those particles at time k with no descendants at time $k+L$ are downweighted to zero, and can be discarded, in effect, early. This is most pronounced in situations where the particle filter de-weights a large proportion of particles at each time step, and is very important in the combination of SFLS and BS-PS given in Section IV. At the same time, "fertile" particles which have many descendants after L time steps are given higher weights.

As mentioned earlier, SFLS can run at the same speed as the particle filter, but results are only available after a fixed time delay equivalent to the lag L. It also has a fixed memory requirement; a list of ancestors for each particle for the last L time steps has to be stored as a cyclic buffer. However, increasing the lag tends to give better results [2], so choosing the length of lag to be used is a balance of accuracy versus computational cost and result availability.

B. Justification of algorithm

For the standard "bootstrap" particle filter with resampling at every step and observations $\boldsymbol{z}_k$, the importance weights are [3]

$$w_k^{(i)} \propto p\left(\boldsymbol{z}_k \middle| \boldsymbol{x}_k^{(i)}\right), \tag{3}$$

where $p\left(\boldsymbol{z}_k \middle| \boldsymbol{x}_k^{(i)}\right)$ is the probability of observing $\boldsymbol{z}_k$ conditioned on the target state being $\boldsymbol{x}_k^{(i)}$. We seek the updated weights $w_{k|k+L}^{i}$ in terms of the w_{k+L}^{j}. The Chapman-Kolmogorov equations give (integrating over all intermediate states)

$$w_{k|k+L}^{(i)} \propto p\left(\boldsymbol{z}_{k+L} \middle| \boldsymbol{x}_k^{(i)}\right) \tag{4}$$

$$= \int \mathrm{d}\boldsymbol{x}_{k+1} \dots d\boldsymbol{x}_{k+L}\, p\left(\boldsymbol{z}_{k+L} | \boldsymbol{x}_{k+L}\right) p(\boldsymbol{x}_{k+L} | \boldsymbol{x}_{k+L-1}) \dots p\left(\boldsymbol{x}_{k+1} \middle| \boldsymbol{x}_k^{(i)}\right), \tag{5}$$

where $p\left(\boldsymbol{z}_{k+L} \middle| \boldsymbol{x}_k^{(i)}\right)$ is the probability of observing $\boldsymbol{z}_{k+L}$ at time $k+L$ conditioned on the target being in state $\boldsymbol{x}_k^{(i)}$ at time k, and $p(\boldsymbol{x}_{k+1} | \boldsymbol{x}_k)$ is the probability of propagating a particle in state $\boldsymbol{x}_k$ at time k to a state $\boldsymbol{x}_{k+1}$ at time $k+1$. We then make use of the operator approximation

$$\int \mathrm{d}\boldsymbol{x}_{k+1} \dots d\boldsymbol{x}_{k+L}\, p(\boldsymbol{x}_{k+L} | \boldsymbol{x}_{k+L-1}) \dots p\left(\boldsymbol{x}_{k+1} \middle| \boldsymbol{x}_k^{(i)}\right) \approx \sum_{j \leftarrow i} \tag{6}$$

The dummy labels on the LHS are all the intermediate states between times k and $k+L$, and the corresponding index on the RHS sums over the descendants of particle $\boldsymbol{x}_k^{(i)}$. This becomes exact in the limit $N \to \infty$. Substituting into (5) gives

$$p\left(\boldsymbol{z}_{k+L} \middle| \boldsymbol{x}_k^{(i)}\right) \approx \sum_{j \leftarrow i} p\left(\boldsymbol{z}_{k+L} \middle| \boldsymbol{x}_{k+L}^{(j)}\right) \tag{7}$$

$$\propto \sum_{j \leftarrow i} w_{k+L}^{j} \tag{8}$$

as required.

III. BACKWARDS SIMULATION PARTICLE SMOOTHER

A. Algorithm

Whereas the SFLS can effectively only see the future L time steps ahead, the backwards simulation particle smoother instead works with the complete output of the particle filter, at all time steps; such techniques are known as global smoothing. This has some immediately evident drawbacks. For example, the amount of data that needs to be stored is potentially very large, proportional to the dimensionality of the state space, the total number of time steps T, and the number of particles N used by the underlying particle filter. As we shall see, the computational complexity scales similarly. In addition, BS-PS can only be implemented after the particle filter has finished running, severely restricting the availability of results.

The BS-PS algorithm, as derived on page 167 of [1], produces any number S of trajectories through the state space, a single one of which we shall denote by $\{\tilde{\boldsymbol{x}}_k\}_{k=1}^{T}$. The process for generating such a path is as follows:

1. Choose $\tilde{\boldsymbol{x}}_T = \boldsymbol{x}_T^{(i)}$ with probability $w_T^{(i)}$
2. For $k = T-1, \ldots, 1$
   a. Compute new weights by

$$w_{k|k+1}^{(i)} \propto w_k^{(i)} p\left(\tilde{\boldsymbol{x}}_{k+1} \middle| \boldsymbol{x}_k^{(i)}\right) \tag{9}$$

   b. Choose $\tilde{\boldsymbol{x}}_k = \boldsymbol{x}_k^{(i)}$ with probability $w_{k|k+1}^{(i)}$

Scaling the weights according to the forwards stepping probability in (9) helps to ensure that the trajectories are continuous paths through the state space – each backwards step will be individually reasonable. Further, particles with zero weight will not be included in the trajectory, so erroneous particles as described in Section I will be ignored.

B. Complexity

The calculation of individual stepping probabilities may be relatively simple. For example, in many particle filter applications, the forwards model (1) takes the slightly simpler form

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k) + \boldsymbol{v}_k, \tag{10}$$

and the process noise is assumed to be white Gaussian, i.e. $\boldsymbol{v}_k \sim \mathcal{N}(\boldsymbol{0}, Q)$ for a given covariance matrix Q. Denoting the probability of an event A by $\mathbb{P}(A)$, we have

$$\begin{aligned} p\left(\tilde{\boldsymbol{x}}_{k+1} \middle| \boldsymbol{x}_k^{(i)}\right) &= \mathbb{P}\left(\boldsymbol{v}_k = \tilde{\boldsymbol{x}}_{k+1} - \boldsymbol{f}\left(\boldsymbol{x}_k^{(i)}\right)\right) \\ &= \mathcal{N}\left(\tilde{\boldsymbol{x}}_{k+1} - \boldsymbol{f}\left(\boldsymbol{x}_k^{(i)}\right); \boldsymbol{0}, Q\right) \end{aligned} \tag{11}$$

where $\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \Sigma)$ is the multivariate normal probability density function with mean $\boldsymbol{\mu}$ and covariance matrix Σ evaluated at the point $\boldsymbol{x}$.

However, as a whole BS-PS is computationally expensive. This is due to the fact that the above probability must be evaluated once at each time step for all N particles, for each backwards trajectory. Indeed, [1] gives the computational cost of the BS-PS algorithm as $O(STN)$. In theory, the complexity may be reduced using rejection sampling to generate the trajectory [4]. However, it was found that this approach can be significantly slower, e.g. in cases where the stepping probabilities $p\left(\tilde{\boldsymbol{x}}_{k+1} \middle| \boldsymbol{x}_k^{(i)}\right)$ are almost all either zero or extremely small, as in many applications.

The accuracy of the backwards smoother is much more dependent on the number of particles used by the underlying particle filter than it is on the number of trajectories produced [5]. As such, it is reasonable to take $S \ll N$, particularly if N is required to be very large. As a result, in many implementations it is the large value of N that is the primary cause of computational cost, rather than a large value of S.

While BS-PS was found to be an expensive algorithm for our example of the indoor tracker (see Section VIII), taking a long period of time to generate a usable number of trajectories, this may not always be the case. The high rate of particle de-weighting in the indoor tracker requires N to be very large, whence arises the cost of the algorithm. In applications where the particle filter does not require so many particles, BS-PS may be reasonably efficient.

The linear dependence of the complexity on T is, for the most part, unavoidable. Dividing a particularly long data set into chunks may allow for parallel processing, but the total computation required is the same. Further, BS-PS does not outperform the particle filter when $k \lesssim T$, since at these times it does not have much future data with which to work. Thus such a division of the data set would somewhat compromise the performance of the smoother.

It should also be noted that, while potentially very computationally costly, the input of particles at all time steps to the BS-PS algorithm does result in significantly increased performance, especially compared to a particle filter with no smoothing. Section IV presents a method for using BS-PS quickly in cases where N is required to be very large.

C. Path vs. state

There is a subtle qualitative difference between the output of the particle filter or SFLS as opposed to BS-PS. The former gives state estimations for each time step, from which one can extract an estimate of the path taken. By contrast, BS-PS gives multiple estimations for the path taken, from which a state estimate at each time step can be formed. This is relevant when it comes to comparing accuracy; there may be time steps where SFLS gives a better state estimation than BS-PS, but trajectories produced by BS-PS tend to better reflect the target path as a whole.

IV. COMBINATION OF SFLS AND BS-PS

As mentioned in Section II, SFLS discards particles with no descendants after the lag period L. In some cases this reduction in particle number can be very significant. For the simulated data experiment in Section VI.A, the particle filter used $N = 50{,}000$ particles at each time step, but the SFLS typically only kept between $N_k = 700$ and $N_k = 2{,}000$ of these (note that the number of particles discarded varies between time steps, hence the k suffix).

Turning to BS-PS, we find that while performance is typically higher than both the plain particle filter and the SFLS, it can be prohibitively computationally expensive. This expense arises from the typically quite large number of particles required by the filter. The question then naturally arises: what if both techniques were applied? That is, what would be the effect of running the particle filter with the SFLS (recall this does not slow down the process) and then using the remaining particles to generate backwards trajectories from BS-PS?

Depending on the proportion of particles discarded by the SFLS, this technique may speed up BS-PS very significantly. The above reduction in particle number resulted in an increase of speed by a factor of 50, as detailed in Section VIII. We find, however, that this combination has a slightly reduced accuracy in comparison to BS-PS running on the particle filter with no other smoothing.

V. EXPERIMENTAL SETUP

A. Indoor tracker

The performances and computational complexities of the methods presented were tested using a particle filter for tracking the location of a target indoors. Here, a particle state consists of x and y position coordinates, an orientation bias and a step length bias. While it is only the position that is estimated by the tracker, the hidden bias parameters feature in the update of a particle's position, and have noise added to them at each step.

For the most part, measurements take the form of step headings and times, but where possible the tracker can also use GPS measurements to improve location estimation – typically this only occurs when the target is outdoors. Information is also provided by making use of the building map along with an assumption that the user does not move through walls. Given an assumed average step length, these measurements can be used to generate a dead reckoning path. However, due to variation in step length and noise in the heading measurements, such paths tend to gradually bear less and less resemblance to the true path as time goes on. It was for this reason that a particle filter was used for the indoor tracking problem; it makes allowances for noise, and enables the effective use of map data to give much-improved results.

The high rate of de-weighting referred to in Section III.B comes from the fact that the particle filter assigns zero weight to any particles which cross a wall as their position is updated; in a narrow corridor, for instance, this can happen very often.

B. Experiment with simulated data

The first experiment used simulated truth data for a path around a building. Process noise was then added to the path, and finally noisy measurements were obtained, including GPS positions and times when the path was outside the building. White Gaussian noise was added to these measurements at each time step, with

standard deviations given by baseline values multiplied by a scale factor. The baseline values were 0.15 radians for step orientation, 0.05 seconds for step time, 0.25 metres for GPS position, and 0.005 seconds for GPS time.

C. *Experiment with real data*

The indoor tracker particle filter was also tested using real inertial and GPS data, gathered by a unit which could be attached to a person's belt (see right). Similarly to the simulated data above, the raw data was processed to give a list of step bearings and times. While the unit is capable of receiving GPS signals while outside, for the purposes of this experiment this functionality was switched off.

In order to be able to measure accuracy, four waypoints were chosen around the path. At each waypoint, a button on the unit was pressed, and the time of this button press was stored. The particle states at these times were then recorded, from which the error was calculated.

D. *Error measures*

The weighted RMSE's for the particle filter and the SFLS at time k are given by

$$E_k = \sqrt{\sum_{i=1}^{N} w_k^{(i)} \left| \boldsymbol{x}_k^{(i)} - \boldsymbol{y}_k \right|^2} \quad \text{and} \quad E_k = \sqrt{\sum_{i=1}^{N_k} w_{k|k+L}^{(i)} \left| \boldsymbol{x}_k^{(i)} - \boldsymbol{y}_k \right|^2} \tag{7}$$

respectively, where $\boldsymbol{y}_k$ is the known position of the target at time k. The trajectories produced by BS-PS do not have weights associated with them, so the standard measure

$$E_k = \sqrt{\sum_{i=1}^{S} \frac{1}{S} \left| \tilde{\boldsymbol{x}}_k^{(i)} - \boldsymbol{y}_k \right|^2} \tag{8}$$

was used instead, where S is the number of trajectories.

Another error measure used was cut-off weighted RMSE: if a particle's RMSE was above a certain threshold, in this case chosen to be 15m, it was deemed to be simply wrong, and the error was recorded as being exactly the threshold.

VI. RESULTS

A. *Simulated data*

First we investigate performance for a single level of measurement noise (scale factor 2). All four methods (particle filter, SFLS, BS-PS and BS-PS on SFL smoothed data) were run 20 times, with the particle filter using 50,000 particles at each time step, SLFS using a lag of 30 time steps, and the BS-PS routines both producing 100 trajectories each time.

Figure 1 shows the average weighted RMSE at each time step, for a single noise level. Possibly the most striking features of this figure are the large peaks between 200 and 300 seconds in the error for the particle filter, and (less pronounced) for SFLS. These represent a period when a large number of particles typically became erroneous, and diverged significantly from the target path. In both cases, the error was subsequently corrected, although the SFLS tended to correct it sooner and adjust the particle weights to improve accuracy further. Indeed, the amount of time by which the SFLS corrected the error sooner was almost precisely the length of the lag used. By contrast, the peaks are absent for both ordinary BS-PS and BS-PS on SFL smoothed data (combined smoother); since the erroneous particles were later discarded, they did not feature in the trajectories that BS-PS produced.

We also note that BS-PS and the combined smoother have very similar levels of RMSE, and that they both consistently outperform the particle filter and the SFL smoother, though not by a great amount for the first half of the data set. It appears that it is precisely in those situations where particles diverge, only to re-converge on the correct state later, that backwards simulation has the greatest impact on accuracy.

Figure 3 gives more of an overview of the results. In order to prevent the peak in the particle filter error from skewing the results unreasonably, the cut-off weighted RMSE was averaged over the time period in Figure 1 for each run, and this is displayed in a box and whisker plot. Error measures which were deemed to be anomalous (outside $\sim 2.7\sigma$ assuming a normal distribution) are shown separately as crosses. We see clearly that smoothing has improved the estimation of the target state. BS-PS has very significantly improved the accuracy of the estimation, though slightly less so when running in combination with SFLS.

Finally, Figure 2 shows how weighted RMSE varies as measurement noise is increased; this was averaged over 5 runs for each noise level (scale factors 0.1, 0.2, 0.5, 1, 2 and 5), with the BS-PS producing 50 trajectories each time due to computational constraints. The results show that the conclusions about relative performance of the four algorithms hold for all but the highest noise level. At the highest noise level, the standard deviation for RMSE is the highest, so the relative increase in error for BS-PS is likely to be an artefact of the small number of tests. Figure 2 also serves to illustrate the dependency of the smoothing algorithms on the particle filter; as noise is increased, the particle filter performs worse, and then so too do all of the smoothing algorithms.

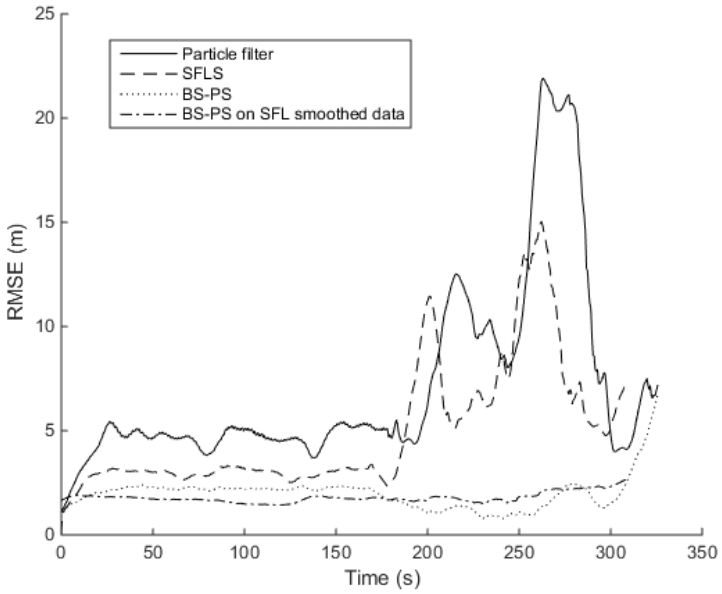


Figure 1: average weighted RMSE over the course of the data set

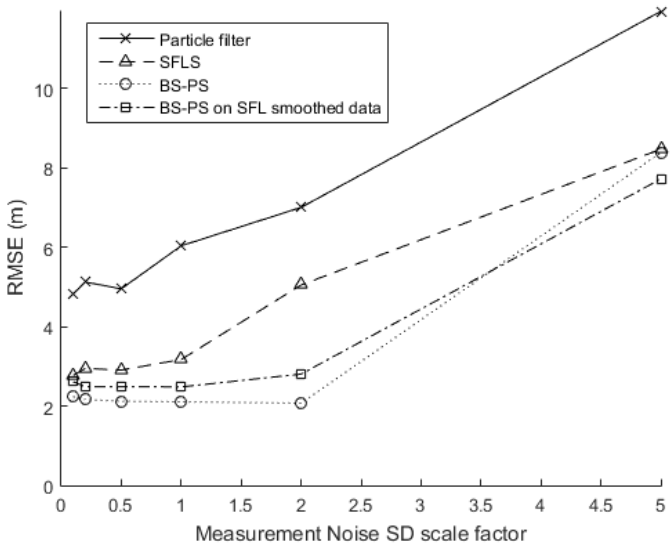


Figure 2: average weighted RMSE with measurement noise level

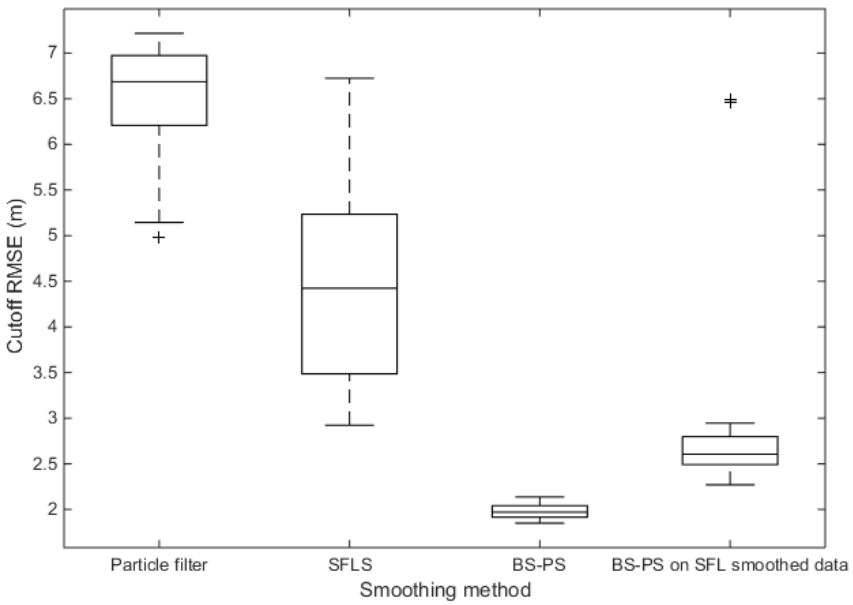


Figure 3: cut-off weighted RMSE for each run

B. Real data

Figure 4 shows the average weighted RMSE at the four predetermined waypoints along the path taken. In order to eliminate the effects of randomness as much as possible, 5 data sets for the same route were collected, and all four methods were run 5 times on each data set. Once again the particle filter used 50,000 particles and the SFLS had a lag of 30 time steps, and the BS-PS routines here produced 50 trajectories each time.

While the results are distinctly mixed, we see that smoothing in general does significantly improve accuracy. The unsmoothed particle filter gives the worst or second worst performance at all waypoints. The only absolutely consistent relationship in accuracy is between the particle filter and SFLS; here smoothing always improves accuracy, although only marginally at waypoint 2. It is also worth noting that the combined smoothing method outperforms ordinary BS-PS at waypoints 1, 3 and 4.

Figure 5 shows the positions of the particles at waypoint 4 in a single run for all four methods, as well as the true position of the waypoint (red circle). It clearly illustrates the ability of smoothing to refine state estimation through the elimination of erroneous particles, as well as the ability of BS-PS to further refine estimations given by SFLS. Note, however, that while it certainly tightens the estimation of SFLS, it does not necessarily improve it; it is most effective in situations when SFLS is incapable of reducing the estimation to a single particle cloud by itself.

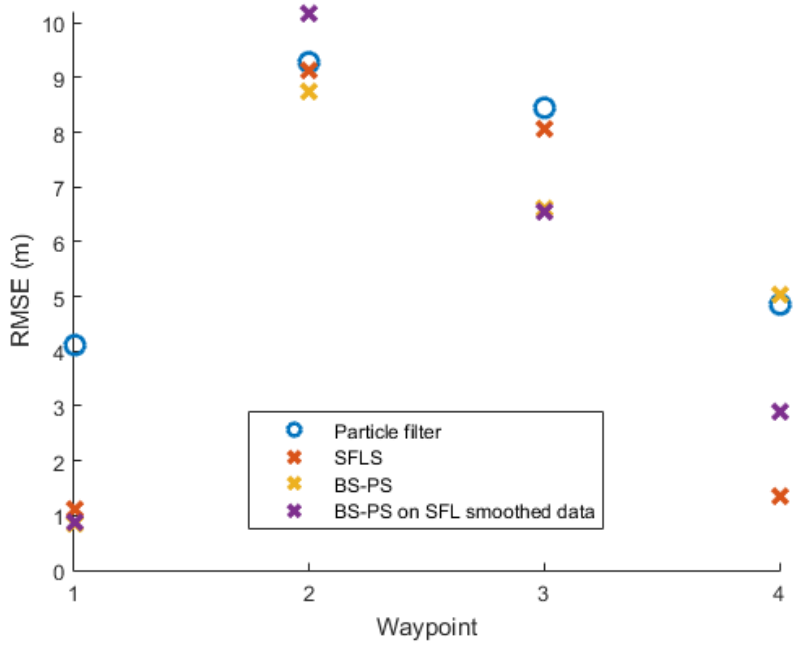


Figure 4: average weighted RMSE at each waypoint

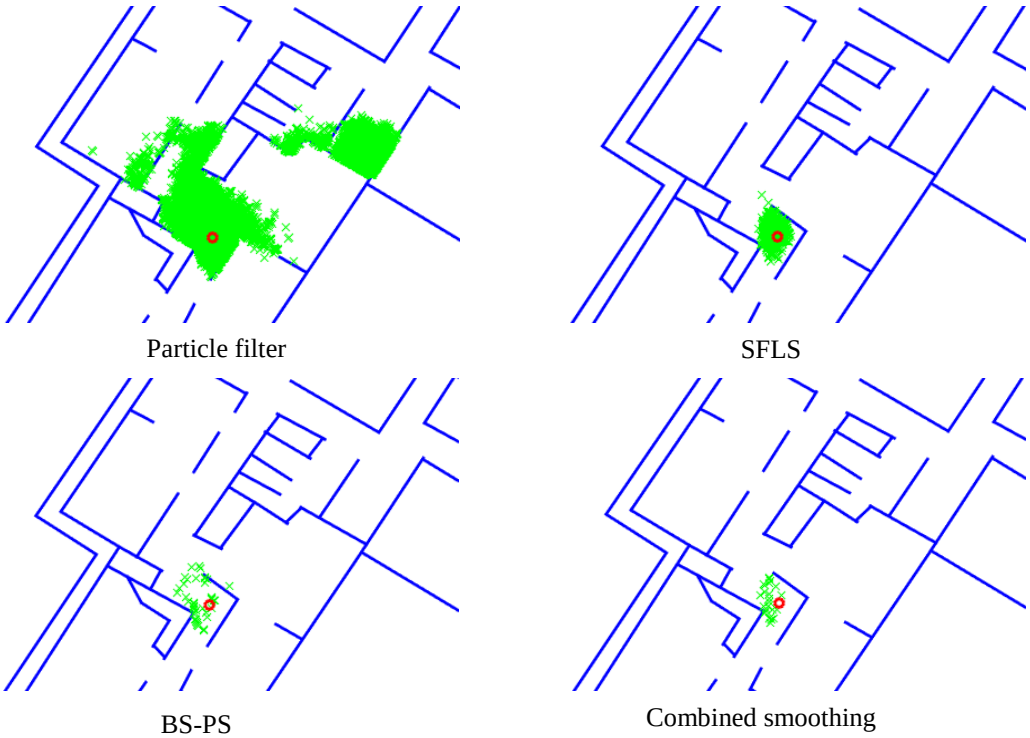


Figure 5: particle positions and true location at waypoint 4

VII. COMPUTATIONAL COMPLEXITY

Both the particle filter and SFLS are capable of following data in real time, though with a fixed delay in the case of the SFLS. In both of the experiments presented in Section V, however, data was collected in advance rather than streamed into the filter. The real time capabilities of these methods instead manifested themselves as the processing time being significantly less than the length of the data set itself. Here we focus instead on the computational cost of implementing BS-PS.

Processing was done using an Intel® i7-4770 3.40GHz CPU.

A. Simulated data

The simulated data set was approximately 600 seconds long (although this is only an estimate based on how long each step would take), which the particle filter and SFLS could process in 150 seconds.

To produce 100 trajectories for this data, the BS-PS routine running on the plain particle filter took an average of ~45 minutes (2,700 seconds).

However, BS-PS running on the SFL smoothed data, which has slightly improved performance compared to BS-PS on the plain particle filter, took an average of just 54 seconds, running in less than one tenth of the length of the data set. The factor 50 reduction in time taken is consistent with what one might expect given that SFLS typically only kept $700 < N_k < 2{,}000$ particles at each time step. The memory requirements (purely for running BS-PS) were also much reduced, from 872MB to 30MB.

B. Real data

The real data set was roughly half the length of the simulated one, lasting 308 seconds. 50 BS-PS trajectories took an average of ~15 minutes (900 seconds) to calculate using the plain particle filter output. By contrast, running this on the SFL smoothed data took an average of just 9 seconds. This further reduction in runtime, now by a factor of 100, came about primarily because SFLS kept even fewer particles for the real data than it did for the simulated data, typically in the range $300 < N_k < 1{,}200$ out of the total 50,000. The memory required was reduced from 443MB to 7.5MB.

VIII. CONCLUSION

A. Comparison of Results for Different Techniques

In this paper, we presented two different smoothing techniques for particle filters. Sequential fixed-lag smoothing provides an efficient way of improving state estimation, but can be outperformed by the more costly backwards simulation particle smoother.

We found that a very cost-effective way to utilise the increased performance of BS-PS in situations where the particle filter requires a large number of particles is to smooth the output of the particle filter using SFLS first, in order to reduce the number of particles, before proceeding. In our example of the indoor tracker, the reduction in processing time required was dramatic, and accuracy attained by BS-PS was also slightly improved in the real data example. The table below presents a summary of the results for both the simulated (SD scale 2) and real data.

The results also make it clear, however, that no single one of SFLS and BS-PS outperforms the other in a truly consistent manner. This raises the possibility of further research into the precise nature of the situations in which each smoothing algorithm is the optimal choice.

		Particle filter		SFLS		BS-PS		Combined smoothing	
		Simulated	Real	Simulated	Real	Simulated	Real	Simulated	Real
RMSE (m)	Mean	7.16	6.67	4.96	4.91	1.98	5.31	3.27	5.12
	95 percentile	24.1	9.45	15.8	9.26	3.04	13.29	4.16	10.72
Results availability		Real time		Fixed delay		After post-processing		After post-processing	
Processing time (s)		150	90	150	90	2850	990	204	99
Additional memory		N/A	N/A	72MB	72MB	872MB	443MB	102MB	79MB

B. Relevance of Techniques for Different Applications

As a tracking algorithm, particle filtering techniques are suitable for a range of applications beyond the indoor tracking example presented here. Tracking of aircraft, ships, submarines, road vehicles and people using tracking technologies such as radar, sonar and video may all benefit from particle filter approaches when the measurement is noisy and non-linear in the system state variables. The more interesting question is when SFLS, BS-PS or combined smoothing should be preferred to a standard particle filter in real applications.

In scenarios where tracking in "up-to-the-second" real-time is necessary, none of the smoothing approaches here will be suitable as they inherently require future information to improve the tracking estimate.

Both BS-PS and combined smoothing are post-processing approaches which require the whole data set to be available before processing begins. These techniques are more appropriate for scenarios where the problem is to reconstruct the path that a target took. Relevant scenarios might include post-analysis of the path of an uncooperative target, post-analysis of a military exercise or retrospective analysis of traffic patterns (including patterns of flow through a building or along a flight path). It would of course be possible to bring these approaches closer to real-time by periodically running the full algorithm on the data set collected up to a particular point in time. However, this would lead to a computational burden beyond that described in the table above.

SFLS is relevant for scenarios in-between real-time and post-processing by giving results at a fixed delay. This would be relevant for scenarios where decisions based on the tracker output do not need to be taken instantly and where a more accurate estimate at a slight delay is better than a less accurate estimate now. The results using SFLS reported above were obtained by introducing only a 30 second time delay, which might be acceptable in some operational scenarios.

REFERENCES

[1] Simo Särkkä, 2013. "Bayesian Filtering and Smoothing," Cambridge University Press.

[2] Dan Simon, 2006. Chapter 9.3, "Optimal State Estimation," Wiley-Blackwell.

[3] B. Ristic, S. Arulampalam and N. Gordon, 2004. Chapter 3.5, "Beyond the Kalman Filter," Artech House.

[4] R. Douc, A. Garivier, E. Moulines and J. Olsson, 2011. Page 9, "Sequential Monte Carlo smoothing for general state space hidden Markov models," *Annals of Applied Probability*, **21**(6), 2109-2145.

[5] F. Lindsten and T. B. Schön, 2013. Chapter 3.1, "Backward Simulation Methods for Monte Carlo Statistical Inference," Now Publishers.

[6] E. Taghavi, F. Lindsten, L. Svensson, and T. B. Schön, 2013. "Adaptive stopping for fast particle smoothing," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Vancouver, Canada.

BAYESIAN STATISTICS 6, pp. 000--000
J. M. Bernardo, J. O. Berger, A. P. Dawid and A. F. M. Smith (Eds.)

Fixed-Lag Smoothing using Sequential Importance Sampling[1]

TIM C. CLAPP and SIMON J. GODSILL
Cambridge University Department of Engineering, UK

SUMMARY

In this paper we present methods for fixed-lag smoothing using Sequential Importance sampling (SIS) for state space models with unknown parameters. Sequential processing using Monte Carlo simulation is an area of growing interest for many engineering and statistical applications where data arrive point by point rather than in a batch. The methods presented here are related to the particle filtering ideas seen in Gordon *et al.* (1993), Liu and Chen (1995), Berzuini *et al.* (1997), Pitt and Shephard (1998) and Doucet *et al.* (1998). Techniques for fixed-lag simulation using either the filtering density or the smoothing density are developed. In addition we describe methods for regenerating parameters of the state-space model by sampling. We are concerned in particular with problems in Digital Communication systems where off-line or batch-based methods, such as Markov chain Monte Carlo (MCMC), are not well suited. The new techniques are demonstrated by application to a standard digital communications model and the performance of the various methods is compared.

Keywords: FIXED-LAG SMOOTHING, SAMPLING IMPORTANCE RESAMPLING (SIS), SEQUENTIAL MONTE CARLO, BLIND DECONVOLUTION, IMPORTANCE SAMPLING, PARTICLE FILTERS.

1. INTRODUCTION

The topic of Bayesian filtering using Monte Carlo techniques is a rapidly growing research area in many branches of science, statistics and engineering, since a sequential approach often provides a very natural way of processing large amounts of data that arrive point by point rather than in a batch. Monte Carlo methods have been shown to give dramatic improvements in performance for non-linear and non-Gaussian state-space models when compared with classical sequential techniques such as the extended Kalman filter (Anderson and Moore, 1979) and its variants.

The methods in this paper are developed from the particle filtering ideas presented in Gordon *et al.* (1993), Liu and Chen (1995), Berzuini *et al.* (1997) and Pitt and Shephard (1998). Particle filters work by representing the initial probability distribution by a set of points or *particles* drawn from that distribution. To update the samples from this probability distribution with the arrival of new data, weights are assigned to the particles according to their *importance* to the new distribution. If the variance of these weights becomes large, then only a small number of particles are contributing towards the representation of the probability distributions. We may *rejuvenate* by resampling the particles with probability according to their weights and reset weights to equal values. Clearly the probability distribution is only accurately described with a sufficiently high number of particles.

While the filtering problem is certainly the primary interest in many applications such as target tracking, in other applications such as digital communications or noise reduction for

[1]This work is supported by the Engineering and Physical Sciences Research Council.

speech signals, a small amount of delay, or latency, is allowable in the estimation of certain quantities. Then it will be profitable to perform fixed-lag smoothing rather than filtering (see e.g. Heller and Jacobs (1971) for a communications example). In this paper we address the problem of fixed-lag smoothing for state space models with fixed parameters. The methods developed are motivated by the structure of simple digital communications models which allow for very efficient simulation of the fixed-lag state vector.

We will assume a state-space form for the model of the data:

$$\mathbf{x}_t = f(\mathbf{x}_{t-1}, \boldsymbol{\theta}, \mathbf{b}_t)$$
$$\mathbf{y}_t = g(\mathbf{x}_t, \boldsymbol{\theta}, \mathbf{v}_t)$$

where $\mathbf{x}_t$ is the current state vector, $\mathbf{y}_t$ is the observation vector, $\boldsymbol{\theta}$ is a vector of unknown parameters and $\mathbf{v}_t$ is observation noise. In the communications model used later to illustrate the methods, the input sequence $\mathbf{b}_t$ is a discrete-valued sequence of transmitted symbols and hence $\mathbf{x}_t$ is a discrete Markov model. However, the techniques developed are also applicable when $\{\mathbf{x}_t\}$ and $\{\mathbf{y}_t\}$ form a linear Gaussian state-space model conditional upon $\boldsymbol{\theta}$.

In the standard Monte Carlo filtering problem the aim is to simulate the state $\mathbf{x}_t$ from the filtering density, $p(\mathbf{x}_t|\mathbf{y}_{0:t})$, where $\mathbf{y}_{0:t}$ denotes the observation vectors from time 0 to t, inclusive. In the Monte Carlo fixed-lag problem we require a simulation from the smoothing density at lag p, that is $p(\mathbf{x}_{t-p}|\mathbf{y}_{0:t})$. For the model above there are two major technical challenges involved in this. The first is to handle the parameters $\boldsymbol{\theta}$, which are not straightforwardly incorporated into the basic frameworks of Gordon *et al.* (1993) or Pitt and Shephard (1998), since they require $\boldsymbol{\theta}$ to be dynamic rather than fixed. Berzuini *et al.* (1997), Liu and Chen (1995), and Liu and Chen (1998) give some insights into how the problem might be solved but do not give a complete solution for the general case. The second problem is the fixed-lag issue, which was also discussed by Pitt and Shephard in an early version of their 1998 paper. We provide some further insight into the solution of these challenging issues by considering state space models for which the following conditions apply:

- We can sample from $p(\boldsymbol{\theta} \mid \mathbf{x}, \mathbf{y})$, either directly or by some other method such as MCMC or rejection sampling.
- $\{\mathbf{x}_t\}$ is a discrete-valued Markov model and we can evaluate $p(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta})$. Alternatively, $\{\mathbf{x}_t\}$ and $\{\mathbf{y}_t\}$ form a linear Gaussian state-space model conditional upon $\boldsymbol{\theta}$.

Our methods are illustrated by an application from the field of digital communication systems. Each input data point is drawn from a finite set of symbols. We represent the transmission medium as a fixed filter with a finite impulse response (FIR), hence a discrete state-space system is formed. Conventional Markov chain Monte Carlo (MCMC) techniques such as the Gibbs sampler are unsuitable for this task because they can only perform processing on a batch of data. Data arrives sequentially, so it is advantageous to process it in this way. In addition, many communication systems are interactive, so there is a maximum level of latency that can be tolerated before a symbol is decoded. However, by allowing a delay of around $4-5$ times the channel length, significant gains in performance can be obtained (Heller and Jacobs, 1971).

The paper is organised as follows: an introduction to importance sampling and the technique of sequential importance sampling is described in section 2. Section 3 describes two methods for achieving fixed-lag simulation using the filtering density—the differences arise from whether the parameters are marginalised analytically from the filtering density, the

position of the filtering density tracked and the importance function used. Section 4 describes a method for fixed-lag simulation by updating the smoothing density. A comparison of the algorithms in the form of simulations on a digital communications system is presented in section 5.

2. SEQUENTIAL IMPORTANCE SAMPLING (SIS)

2.1. *Importance Sampling*

Suppose we are interested in the expectation of some function, $E\{f(\mathbf{x})\}$, but we cannot generate random draws from $p(\mathbf{x})$ directly. If $\pi(\mathbf{x})$ is a normalised density then we may write:

$$E\{f(\mathbf{x})\} = \int \frac{f(\mathbf{x})p(\mathbf{x})}{\pi(\mathbf{x})}\pi(\mathbf{x})\, dx$$

which may be estimated by drawing N samples, $\mathbf{x}^{(i)}$, from $\pi(\mathbf{x})$ and evaluating the expression:

$$E\{f(\mathbf{x})\} \approx \sum_{i=1}^{N} f(\mathbf{x}^{(i)}) \frac{w^{(i)}}{\sum_{j=1}^{N} w^{(j)}} \tag{1}$$

where $w^{(i)} = \frac{p(\mathbf{x}^{(i)})}{\pi(\mathbf{x}^{(i)})}$. The density $\pi(.)$ is called the importance function and the $w^{(i)}$ are the importance weights.

2.2. *Sequential Importance Sampling*

This section describes the basic technique of SIS in which the state distribution does not depend upon parameters, $\boldsymbol{\theta}$.

Suppose we have N samples, $\mathbf{x}_{0:t}^{(i)}$, from the probability distribution of interest at time t, $p(\mathbf{x}_{0:t} \,|\, \mathbf{y}_{0:t})$. With the arrival of a new data point $\mathbf{y}_{t+1}$, we would like to update this distribution to $p(\mathbf{x}_{0:t+1} \,|\, \mathbf{y}_{0:t+1})$ without modifying the simulated past trajectories, $\mathbf{x}_{0:t}^{(i)}$. Normally we can evaluate the likelihood $p(\mathbf{y}_t \,|\, \mathbf{x}_t)$ and the prior distribution $p(\mathbf{x}_{t+1} \,|\, \mathbf{x}_t)$. We may update by using importance sampling sequentially, exploiting Bayesian updating at each time step as follows:

$$p(\mathbf{x}_{0:t+1} \,|\, \mathbf{y}_{0:t+1}) = \frac{p(\mathbf{y}_{t+1}, \mathbf{x}_{t+1} \,|\, \mathbf{x}_t)}{p(\mathbf{y}_{t+1} \,|\, \mathbf{y}_{0:t})} \times p(\mathbf{x}_{0:t} \,|\, \mathbf{y}_{0:t}) \tag{2}$$

More generally, if an importance function of the form:

$$\begin{aligned}\pi(\mathbf{x}_{0:t+1} \,|\, \mathbf{y}_{0:t+1}) &= \pi(\mathbf{x}_{t+1} \,|\, \mathbf{x}_{0:t}, \mathbf{y}_{0:t+1})\, \pi(\mathbf{x}_{0:t} \,|\, \mathbf{y}_{0:t}) \\ &= \pi(\mathbf{x}_0 \,|\, \mathbf{y}_0) \prod_{k=1}^{t+1} \pi(\mathbf{x}_{k+1} \,|\, \mathbf{x}_{0:k}, \mathbf{y}_{0:k+1})\end{aligned}$$

is chosen, the importance weights may be evaluated recursively (Doucet *et al.* , 1998). Typically one cannot calculate the normalising term $p(\mathbf{y}_{t+1} \,|\, \mathbf{y}_{0:t})$; however this is not required since it is independent of the state trajectory. The unnormalised incremental importance weights may then be calculated:

$$w_{t+1}^{(i)} = \frac{p(\mathbf{y}_{t+1}, \mathbf{x}_{t+1} \,|\, \mathbf{x}_t^{(i)})}{\pi(\mathbf{x}_{0:t+1} \,|\, \mathbf{x}_{0:t}^{(i)}, \mathbf{y}_{0:t+1})} \times w_t^{(i)}$$

In choosing a suitable importance function, any factorisation of the joint density, $p(\mathbf{y}_{t+1}, \mathbf{x}_{t+1} \mid \mathbf{x}_t)$, may be exploited so as to simplify this expression.

Statistics of interest may then be obtained from equation 1. If the variance of the weights becomes too large, we may resample the trajectories with the probability of each trajectory being selected equal to the corresponding weight. A useful measure of whether to resample is the effective sample size, described by Liu and Chen (1995), which tells us how many particles are actually contributing significantly to the distribution.

3. FIXED-LAG SIMULATION USING THE FILTERING DENSITY

The standard SIS technique may be adapted to achieve fixed-lag smoothing, whilst updating the filtering density. We consider two ways of achieving this: standard SIS (with parameters marginalised) in which the state history is stored back to a lag of p, and a new technique in which the states and parameters are both drawn sequentially.

3.1. *SIS and Sequential Imputations*

The simplest method for fixed-lag simulation is to update the filtering density as each data point arrives as in standard SIS, i.e. track the evolution of $p(\mathbf{x}_{0:t} \mid \mathbf{y}_{0:t})$, storing the state history of each particle back to a lag of p states. For this type of updating we may use any of the standard importance distributions ranging from the prior distribution, $p(\mathbf{x}_{t+1} \mid \mathbf{x}_t)$, as described by Handschin and Mayne (1969) and Gordon *et al.* (1993) to the optimal importance function, $p(\mathbf{x}_{t+1} \mid \mathbf{y}_{t+1}, \mathbf{x}_t)$, described by Zaritskii *et al.* (1975) and Liu and Chen (1995) (under the name of sequential imputations). For a review of these methods see Doucet *et al.* (1998). This direct approach requires analytic marginalisation of the parameters $\boldsymbol{\theta}$, which is in fact possible for the communications model we shall consider, but will not be possible for many other models of practical interest.

The optimal method for filtering the data without lookahead is the method of *sequential imputations* as described by Liu and Chen (1995) (see Doucet *et al.* (1998) for a discussion of this). At each step only a draw of the state from $p(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \mathbf{y}_{t+1})$ is required. The incremental importance weight is given by $p(\mathbf{y}_{t+1} \mid \mathbf{x}_t)$, and updates of the mean and variance of the marginal distribution of the parameters, $\boldsymbol{\theta}$, are obtained by recursive relationships derived from the matrix inversion lemma in the conditionally Gaussian parameter case.

Given the stored histories of each particle back to time $t-p$ it is then straightforward, using the importance weights from time t, to make an estimate of the smoothing density, from which estimates such as the MAP estimate may be extracted (we refer to this method as 'sequential imputations with decision step'). The performance of the fixed-lag MAP approach is compared in our simulations with the standard sequential imputations methods in which the joint MAP state sequence is estimated after the arrival of all the data.

3.2. *Lagged time filtering density*

As an alternative, we may attempt to draw the fixed-lag state and the parameter jointly, a procedure with more generality than sequential imputations, since it will not require a marginalisation of the parameters. To simplify notation, we will shift our time-base forward by p samples. Hence at time t we will draw fixed-lag samples of $\mathbf{x}_t$ using information from $\mathbf{y}_{0:t+p}$.

The joint prediction density may be factorised as follows:

$$p(\mathbf{y}_{t+1}, \mathbf{x}_{t+1}, \boldsymbol{\theta} \mid \mathbf{x}_{0:t}, \mathbf{y}_{0:t}) = p(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \boldsymbol{\theta}, \mathbf{y}_{t+1})\, p(\mathbf{y}_{t+1} \mid \mathbf{x}_t, \boldsymbol{\theta})\, p(\boldsymbol{\theta} \mid \mathbf{x}_{0:t}, \mathbf{y}_{0:t})$$

By using this factorisation, new draws for the parameters can be made at each time without having to draw the states from the prior density, $p(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \mathbf{y}_t)$, as suggested by Liu and Chen

(1998). It is possible to sample directly from the first term; however we have chosen instead to use the importance function:

$$\pi(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \boldsymbol{\theta}, \mathbf{y}_{t+1:t+p+1}) = p(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \boldsymbol{\theta}, \mathbf{y}_{t+1:t+p+1}) \tag{3}$$

with the hope of using the additional data to bias the draw towards states that will be useful at later iterations. This may be accomplished by using forward filtering backward sampling techniques described by Carter and Kohn (1994), Doucet and Duvaut (1996) and Clapp and Godsill (1997), drawing from $p(\mathbf{x}_{t+1:t+p+1} \mid \mathbf{x}_t, \boldsymbol{\theta}, \mathbf{y}_{t+1:t+p+1})$ and discarding the unwanted imputed values $\mathbf{x}_{t+2:t+p+1}$. This introduces the importance weight:

$$w_{t+1} = \frac{p(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \boldsymbol{\theta}, \mathbf{y}_{t+1})}{\pi(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \boldsymbol{\theta}, \mathbf{y}_{t+1:t+p+1})}\, w_t \tag{4}$$

$$= \frac{p(\mathbf{y}_{t+1} \mid \mathbf{x}_{t+1}, \boldsymbol{\theta})\, p(\mathbf{y}_{t+1:t+p+1} \mid \mathbf{x}_t, \boldsymbol{\theta})}{p(\mathbf{y}_{t+1} \mid \mathbf{x}_t, \boldsymbol{\theta})\, p(\mathbf{y}_{t+1:t+p+1} \mid \mathbf{x}_{t+1}, \boldsymbol{\theta})}\, w_t \tag{5}$$

The first term of both numerator and denominator are easy to compute. For calculation of the second terms, note the following factorisation:

$$p(\mathbf{y}_{t:t-p+1} \mid \boldsymbol{\theta}, \mathbf{x}_{t-p}) = p(\mathbf{y}_t \mid \boldsymbol{\theta}, \mathbf{y}_{t-1:t-p+1}, \mathbf{x}_{t-p}) \times p(\mathbf{y}_{t-1} \mid \boldsymbol{\theta}, \mathbf{y}_{t-2:t-p+1}, \mathbf{x}_{t-p}) \ldots p(\mathbf{y}_{t-p+1} \mid \boldsymbol{\theta}, \mathbf{x}_{t-p}) \tag{6}$$

Each of these terms may be expressed as:

$$p(\mathbf{y}_{t-k} \mid \boldsymbol{\theta}, \mathbf{y}_{t-k-1:t-p+1}, \mathbf{x}_{t-p}) = \sum_{\mathbf{x}_{t-k}} p(\mathbf{y}_{t-k} \mid \boldsymbol{\theta}, \mathbf{x}_{t-k}, \mathbf{x}_{t-p})\, p(\mathbf{x}_{t-k} \mid \boldsymbol{\theta}, y_{t-p+1:t-k-1}, \mathbf{x}_{t-p}) \tag{7}$$

The first term is the likelihood and the second is the prediction density, both of which are calculated during the forward pass when making the draw from equation 3. Although a separate forward pass is needed for the calculation of $p(\mathbf{y}_{t+1:t+p+1} \mid \mathbf{x}_{t+1}, \boldsymbol{\theta})$, this is not required when calculating $p(\mathbf{y}_{t+1:t+p+1} \mid \mathbf{x}_t, \boldsymbol{\theta})$.

Note that in this case the draw for the parameters does not include any information from the extra received data, $\mathbf{y}_{t+1:t+p+1}$. However, in the case of fixed parameters, the prior rapidly becomes quite strong so any additional information in these states does not affect the posterior for $\boldsymbol{\theta}$ very much.

3.2.1. *Method*

Let us assume we have available at time t, N samples, $\mathbf{x}_t^{(j)}$, from the distribution $p(\mathbf{x}_t \mid \mathbf{y}_{0:t})$,[2] possibly with associated weights $w_t^{(j)}$. At time $t+1$ the additional observation $\mathbf{y}_{t+p+1}$ is available. These samples may be updated to the distribution $p(\mathbf{x}_{t+1} \mid \mathbf{y}_{0:t+1})$ as follows:

1. For each j:
   (a) Draw $\boldsymbol{\theta}^{(j)}$ from $p(\boldsymbol{\theta} \mid \mathbf{x}_{0:t}^{(j)}, \mathbf{y}_{0:t})$.
   (b) Draw $\mathbf{x}_{t+1}^{(j)}$ from equation 3.

[2] Initial samples could be created by any batch-based MCMC method.

(c) Calculate importance weights from equation 5.

2. If weights have a high variance (or equivalently, the effective sample size is too low), rejuvenate by resampling with probability proportional to the weights and reset the weights to equal values.

4. FIXED-LAG SIMULATION USING THE SMOOTHING DENSITY

Rather than using the filtering density, $p(\mathbf{x}_{0:t} \mid \mathbf{y}_{0:t})$, the smoothing density, $p(\mathbf{x}_{0:t-p} \mid \mathbf{y}_{0:t})$, may be used to achieve fixed-lag smoothing. Ideologically this is a better approach, although it introduces some practical problems.

The fixed-lag smoothing distribution is given by:

$$\begin{aligned} p(\mathbf{x}_{0:t-p+1} \mid \mathbf{y}_{0:t+1}) &= \frac{p(\mathbf{y}_{t+1} \mid \mathbf{y}_{0:t}, \mathbf{x}_{0:t-p+1})\, p(\mathbf{x}_{0:t-p+1} \mid \mathbf{y}_{0:t})}{p(\mathbf{y}_{t+1} \mid \mathbf{y}_{0:t})} \\ &= \frac{p(\mathbf{y}_{t+1} \mid \mathbf{y}_{0:t}, \mathbf{x}_{t-p+1})\, p(\mathbf{x}_{t-p+1} \mid \mathbf{y}_{t-p+1:t}, \mathbf{x}_{0:t-p})\, p(\mathbf{x}_{0:t-p} \mid \mathbf{y}_{0:t})}{p(\mathbf{y}_{t+1} \mid \mathbf{y}_{0:t})} \end{aligned} \tag{8}$$

Let us define the importance sampling distribution in the usual way:

$$\begin{aligned} \pi(\mathbf{x}_{0:t-p+1} \mid \mathbf{y}_{0:t+1}) &= \pi(\mathbf{x}_0 \mid \mathbf{y}_{0:p}) \prod_{k=1}^{t-p+1} \pi\left(\mathbf{x}_k \mid \mathbf{x}_{0:k-1}, \mathbf{y}_{0:k+p}\right) \\ &= \pi(\mathbf{x}_{t-p+1} \mid \mathbf{x}_{0:t-p}, \mathbf{y}_{0:t+1})\, \pi(\mathbf{x}_{0:t-p} \mid \mathbf{y}_{0:t}) \end{aligned}$$

then the unnormalised importance weight satisfies:

$$\begin{aligned} w(\mathbf{x}_{0:t+1}) &= \frac{p(\mathbf{x}_{0:t-p+1} \mid \mathbf{y}_{0:t+1})}{\pi(\mathbf{x}_{0:t-p+1} \mid \mathbf{y}_{0:t+1})} \\ &= w(\mathbf{x}_{0:t})\, \frac{p(\mathbf{y}_{t+1} \mid \mathbf{y}_{t-p:t}, \mathbf{x}_{0:t-p+1})\, p(\mathbf{x}_{t-p+1} \mid \mathbf{y}_{t-p+1:t}, \mathbf{x}_{0:t-p})}{\pi(\mathbf{x}_{t-p+1} \mid \mathbf{x}_{0:t-p}, \mathbf{y}_{0:t+1})\, p(\mathbf{y}_{t+1} \mid \mathbf{y}_{0:t})} \end{aligned} \tag{9}$$

For a general distribution, the term $p(\mathbf{y}_{t+1} \mid \mathbf{y}_{t-p:t}, \mathbf{x}_{0:t-p+1})$ cannot be evaluated. In some models we may assume that for a sufficiently large value of the lag, p, the term $p(\mathbf{y}_{t+1} \mid \mathbf{y}_{t-p:t}, \mathbf{x}_{0:t-p+1})$ will be approximately constant for all values of $\mathbf{x}_{0:t-p+1}$.[3] This is saying simply that future observations are independent of states from the distant past.

We now consider generating samples from the joint distribution $p(\mathbf{x}, \boldsymbol{\theta} \mid \mathbf{y})$ by using the following factorisation:

$$p(\mathbf{x}_{t-p+1}, \boldsymbol{\theta} \mid \mathbf{y}_{0:t}, \mathbf{x}_{0:t-p}) = p(\mathbf{x}_{t-p+1} \mid \mathbf{x}_{0:t-p}, \mathbf{y}_{t-p+1:t}, \boldsymbol{\theta})\, p(\boldsymbol{\theta} \mid \mathbf{y}_{0:t}, \mathbf{x}_{0:t-p}) \tag{10}$$

Note that we can easily sample from the distribution:

$$p(\mathbf{x}_{t-p+1}, \mathbf{x}_{t-p+2:t} \mid \mathbf{x}_{0:t-p}, \mathbf{y}_{t-p+1:t}, \boldsymbol{\theta}) \tag{11}$$

[3] A similar approximation is made in the communications literature concerning trace-back in a Viterbi equaliser Proakis (1995). After a fixed delay the paths stored in the Viterbi algorithm are assumed to have joined, or if not the most likely path is chosen at this time.

as before. The imputed values, $\widetilde{\mathbf{x}}_{t-p+2:t}$, are retained for use in sampling $\boldsymbol{\theta}$ in the next iteration. We cannot easily sample from $p(\boldsymbol{\theta} \mid \mathbf{y}_{0:t}, \mathbf{x}_{0:t-p})$; however we may sample from the importance distribution:

$$\pi(\boldsymbol{\theta} \mid \mathbf{y}_{0:t}, \mathbf{x}_{0:t-p}) = p(\boldsymbol{\theta} \mid \mathbf{y}_{0:t}, \mathbf{x}_{0:t-p}, \widetilde{\mathbf{x}}_{t-p+1:t}) \tag{12}$$

This introduces an incremental importance weight:

$$\begin{aligned} w_{t+1} &= \frac{p(\boldsymbol{\theta} \mid \mathbf{y}_{0:t}, \mathbf{x}_{0:t-p})}{p(\boldsymbol{\theta} \mid \mathbf{y}_{0:t}, \mathbf{x}_{0:t-p}, \widetilde{\mathbf{x}}_{t-p+1:t})} \, w_t \\ &= \frac{\sum_{\widetilde{\mathbf{x}}_{t-p+1:t}} p(\boldsymbol{\theta} \mid \mathbf{y}_{0:t}, \mathbf{x}_{0:t-p}, \widetilde{\mathbf{x}}_{t-p+1:t})}{p(\boldsymbol{\theta} \mid \mathbf{y}_{0:t}, \mathbf{x}_{0:t-p}, \widetilde{\mathbf{x}}_{t-p+1:t})} \, w_t \end{aligned} \tag{13}$$

At first glance it may appear that the numerator of equation 13 is expensive to calculate, however all the terms will be calculated when sampling from equation 11; only a small number of additional calculations are required.

4.1. *Method*

Let us assume we have available at time t, N samples, $\mathbf{x}_{t-p}^{(j)}$, from the distribution $p(\mathbf{x}_{t-p}^{(j)} \mid \mathbf{y}_{0:t})$, possibly with associated weights $w^{(j)}$. At time $t+1$ the additional observation $\mathbf{y}_{t+1}$ is available. These samples may be updated to the distribution $p(\mathbf{x}_{t-p+1} \mid \mathbf{y}_{0:t+1})$ as follows:

1. For each j:
   (a) Draw $\boldsymbol{\theta}^{(j)}$ from equation 12.
   (b) Draw $\mathbf{x}_{t-p+1}^{(j)}$, $\widetilde{\mathbf{x}}_{t-p+2:t}^{(j)}$ from equation 11 and draw the additional $\widetilde{\mathbf{x}}_{t+1}^{(j)}$ from $p(\mathbf{x}_{t+1} | \mathbf{y}_{t+1}, \widetilde{\mathbf{x}}_t^{(j)}, \boldsymbol{\theta})$.
   (c) Calculate importance weights from equation 13.
2. If weights have a high variance (or equivalently, the effective sample size is too low), rejuvenate by resampling with probability proportional to the weights and reset the weights to equal values.

5. SIMULATIONS

5.1. *Problem Formulation*

Most digital communications systems transmit a signal $\{b_t\}$ where the value of each of the b_t are taken from a finite alphabet of q symbols. This is transmitted over a channel which may introduce distortion and noise. The channel model used here is an FIR filter with additive Gaussian noise:

$$y_t = \sum_{i=0}^{n-1} b_{t-i} h_i + v_t \tag{14}$$

$$b_t \in \{S_0, \ldots, S_{q-1}\} \tag{15}$$

$$v_t \overset{i.i.d.}{\sim} N(0, \sigma_v^2) \tag{16}$$

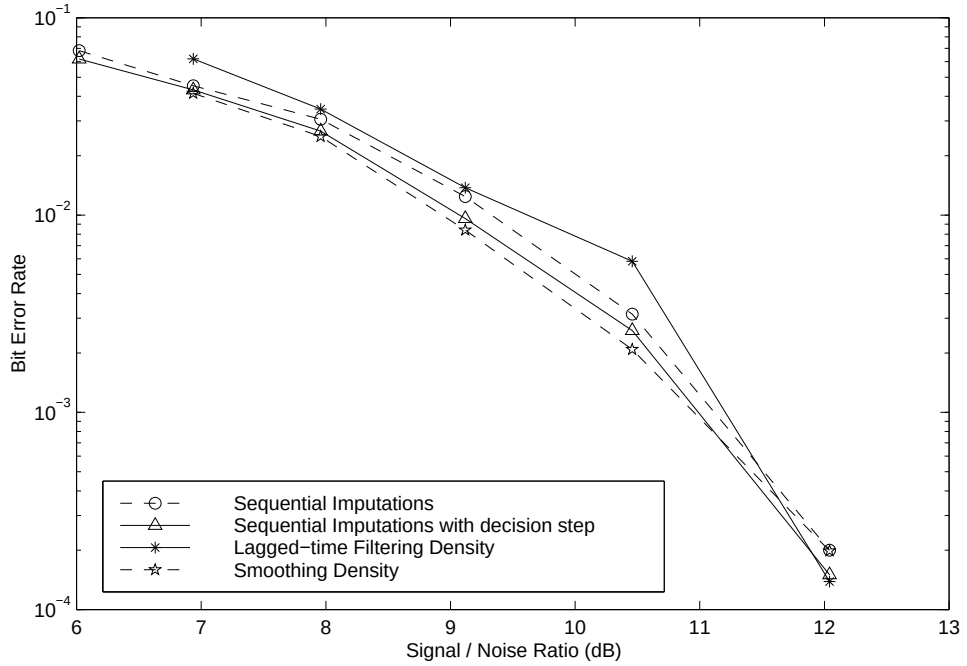


Figure 1. *Graph of Bit Error Rate against Signal-to-Noise ratio in simulations using the different SIS algorithms.*

where $\{y_t\}$ is the observed signal, $\{b_t\}$ is the transmitted signal, S_i a symbol from the alphabet, $\mathbf{h} = \{h_0, \ldots, h_{n-1}\}$ is the channel filter, n is the number of filter taps and t is an integer.

This may be represented in an equivalent state-space form:

$$\left\{ \begin{array}{rcl} \mathbf{x}_t & = & \left[\begin{array}{c|c} \mathbf{0^T} & 0 \\ \hline \mathrm{I} & \mathbf{0} \end{array}\right] \mathbf{x}_{t-1} + \begin{bmatrix} 1 \\ \mathbf{0} \end{bmatrix} b_t \\ y_t & = & \mathbf{h^T x}_t + v_t \end{array} \right. \tag{17}$$

where $\mathbf{x}_t = [b_t, b_{t-1}, \ldots, b_{t-n+1}]^{\mathbf{T}}$ is the state (a vector of length n) and $\mathbf{0}$ is the null vector of length $n-1$. A collection of states, $\mathbf{x}_{1:t}$, can be exactly determined by $\{b_1, \ldots, b_t\}$ and some initial conditions, given equation 17.

5.2. *Results*

To compare methods we tried them on communications channels simulated directly from equation 14 using a binary amplitude modulation scheme where bits are encoded as ± 1. All methods require an estimate of the initial density: this was obtained using a Gibbs sampler (see Clapp and Godsill (1997) for details) which was run over a frame of length 200 samples. The channel was chosen to be $\mathbf{h} = [0.766, 0.575, 0.287]^{\mathbf{T}}$ and the lag $p = 15$, five times the channel length. In all cases 151 particle streams were chosen. This is an order of magnitude lower than many implementations of the particle filter, however it is sufficient for our purposes since we can produce a large number of points close to the posterior modes due to the efficient algorithms employed and the comparative simplicity of the discrete state-space system. For the case of the current time filtering density (sequential imputations) algorithm and the update of the smoothing density, rejuvenation was carried out when the effective sample size fell below 135. The method using the lagged time filtering density produced a much wider variance in the weights, so resampling took place at each step. The input bit at time t was determined by

summing over all states at time t which would have that bit set (this corresponds to a marginal MAP estimate).

The performance of the various algorithms are shown in figure 1. Differential encoding was not employed, but the bits were inverted when the phase ambiguity was incorrectly resolved (choosing $\mathbf{h} = [-0.766, -0.575, -0.287]^{\mathbf{T}}$ is an equally probable solution). In a practical system, either differential encoding or a small number of known bits are required. Note that the error bounds on the points at very high signal to noise ratios are quite large due to a very low number of errors being detected. This could account for the curves swapping over.

5.3. *Comparison of Algorithms*

All algorithms performed well. For comparison the performance of a Viterbi equaliser using 20 known symbols in a frame of 200, the channel estimate being obtained by the least mean squared error, lies somewhere between the dashed circle and the solid triangle curves. The Viterbi algorithm performs maximum likelihood sequence estimation given the channel parameters and is commonly used in practical applications e.g. GSM/ETSI mobile phones. Errors typically occurred in bursts: the burst could last up to 4 symbols (not necessarily all would be in error), though 2 was most common. Rejuvenation occurred most iterations on the sequential imputations algorithm, although the gap increased to about every 10 iterations with less noise present. When using the smoothing density, typical values for the time gap between rejuvenation varied between 150 and 700 as noise levels decreased.

The comparatively poor performance of the lagged-time filtering density algorithm is perhaps due to this algorithm essentially behaving as a directed filter rather than a smoothing algorithm. All of our other algorithms give a genuinely smoothed output; however it is trivial to add a delayed decision step in the same way as for SIS.

It is interesting to note that using the sequential imputations algorithm with the decision step actually performs better than retaining all the past states data until the end and then making a decision. The likely reason for this is because the states drawn are not necessarily good ones and the frequent rejuvenation is required. This repeated resampling leads to degeneracy in the earlier states.

Computationally the lagged draw of a state (or equivalently a joint draw of the states) is the most expensive part of the algorithms. As a result sequential imputations is by far the fastest method, and using the lagged-time filtering density was slowest (by almost a factor of 2) due to the extra pass through the lagged data required for calculation of the importance weights.

6. CONCLUSION

In this paper we have explored some of the issues involved in performing sequential simulations for state space models with unknown parameters, presenting algorithms for filtering and fixed-lag smoothing which can be applied to many of the models current in engineering time series. The utility of the algorithms has been demonstrated by application to a basic digital communications model. There are as yet, however, several unresolved technical issues in this challenging area, and we hope the methods presented here will give some further insight into solving the sequential simulation problem in all its generality.

REFERENCES

Anderson, B. D. O. and Moore, J. B. (1979). *Optimal Filtering.* Englewood Cliffs, NJ: Prentice-Hall.

Berzuini, C., Best, N. G., Gilks, W. R. and Larizza, C. (1997). Dynamic conditional independence models and Markov chain Monte Carlo methods. *J. Amer. Statist. Assoc.* **92**(440), 1403--1412.

Carter, C. K. and Kohn, R. (1994). Gibbs sampling for state space models. *Biometrika* **81**(3), 541--553.

Clapp, T. and Godsill, S. J. (1997). Bayesian Blind Deconvolution for Mobile Communications. *Proceedings of IEE Colloqium on Adaptive Signal Processing for Mobile Communication Systems*, 9/1--9/6. Savoy Place, London. Reference No: 1997/383.

Doucet, A. and Godsill (1998). On Sequential Monte Carlo Sampling Methods for Bayesian Filtering. Submitted for publication. Available as Technical Report CUED/F-INFENG/TR. 310, Cambridge University Department of Engineering.

Doucet, A. and Duvaut, P. (1996). Fully Bayesian Analysis of Hidden Markov models. *Proceedings of EUSIPCO.*

Gelman, A. and Carlin, J. B. and Stern, H. S. and Rubin, D. B. (1995). *Bayesian Data Analysis.* London: Chapman and Hall.

Gordon, N., Salmond, D. and Ewing, C. (1995). Bayesian State Estimation for Tracking and Guidance using the Bootstrap Filter. *Journal of Guidance Control and Dynamics* **18**(6), 1434--1443.

Gordon, N., Salmond, D. and Smith, A. F. M. (1993). Non-linear / Non-Gaussian Bayesian State Estimation. *IEE Proceedings of Radar and Signal Processing, Pt.F*, **140**(2), 107--113.

Handschin, J. and Mayne, D. (1969). Monte Carlo Techniques to Estimate the Conditional Expectation in Multi-stage Non-Linear Filtering. *Int. J. Cont.*, volume 9, 547--559.

Heller, J. A. and Jacobs, I. M. (1971). Viterbi Decoding for Satellite and Space Communication. *IEEE Trans. Communcation Technology* **COM-19**, 835--848.

Kong, A., Liu, J. and Wong, W. H. (1994). Sequential Imputations and Bayesian missing data problems. *J. Amer. Statist. Assoc.* **89**(425), 278--288.

Liu, J. and Chen, R. (1995). Blind Deconvolution via Sequential Imputations. *J. Amer. Statist. Assoc.* **90**(430), 567--576.

Liu, J. S. and Chen, R. (1998). Sequential Monte Carlo methods for dynamical systems, *J. Amer. Statist. Assoc.* **93**.

Pitt, M. and Shephard, N. (1998). Filtering via Simulation: Auxiliary Particle Filters, *J. Amer. Statist. Assoc.* (to appear).

Proakis, J. G. (1995). *Digital Communications*. Electrical engineering series. New York: McGraw-Hill, 3rd edition.

Ross, S. M. (1997) *Simulation*. Statistical modelling and decision science series. New York: Academic Press, 2nd edition.

Smith, A. F. M. and Gelfand, A. E. (1992). Bayesian Statistics without Tears: A Sampling-Resampling Perspective. *Amer. Statist.* **46**(2), 84--88.

Zaritskii, V., Svetnik, V. and Shimelevich, L. (1975). Monte Carlo Technique in Problems of Optimal Data Processing. *Auto. Remo. Cont.* **12**, 95--103.

The Annals of Applied Probability
2011, Vol. 21, No. 6, 2109–2145
DOI: 10.1214/10-AAP735

SEQUENTIAL MONTE CARLO SMOOTHING FOR GENERAL STATE SPACE HIDDEN MARKOV MODELS[1]

By Randal Douc, Aurélien Garivier, Eric Moulines and Jimmy Olsson

Institut Télécom/Télécom SudParis, CNRS UMR 5157, CNRS/Télécom ParisTech, CNRS UMR 5141, Institut Télécom/Télécom ParisTech, CNRS UMR 5141 and Lund University

Computing smoothing distributions, the distributions of one or more states conditional on past, present, and future observations is a recurring problem when operating on general hidden Markov models. The aim of this paper is to provide a foundation of particle-based approximation of such distributions and to analyze, in a common unifying framework, different schemes producing such approximations. In this setting, general convergence results, including exponential deviation inequalities and central limit theorems, are established. In particular, time uniform bounds on the marginal smoothing error are obtained under appropriate mixing conditions on the transition kernel of the latent chain. In addition, we propose an algorithm approximating the joint smoothing distribution at a cost that grows only linearly with the number of particles.

1. Introduction. Statistical inference in general state space hidden Markov models (HMM) involves computation of the *posterior distribution* of a set $X_{s:s'} \stackrel{\text{def}}{=} [X_s, \ldots, X_{s'}]$ of state variables conditional on a record $Y_{0:T} = y_{0:t}$ of observations. This distribution will, in the following, be denoted by $\phi_{s:s'|T}$ where the dependence of this measure on the observed values $y_{0:T}$ is implicit. The posterior distribution can be expressed in closed-form only in very specific cases, principally, when the state space model is linear and Gaussian or when the state space of the hidden Markov chain is a finite set. In the vast majority of cases, nonlinearity or non-Gaussianity render analytic solutions intractable [3, 26, 33, 36].

Received July 2009; revised March 2010.
[1]Supported in part by the French National Research Agency, under the program ANR-07 ROBO 0002 CFLAM.
AMS 2000 subject classifications. Primary 60G10, 60K35; secondary 60G18.
Key words and phrases. Sequential Monte Carlo methods, particle filter, smoothing, hidden Markov models.

This limitation has led to an increase of interest in alternative computational strategies handling more general state and measurement equations without constraining a priori the behavior of the posterior distributions. Among these, *sequential Monte Carlo* (SMC) *methods* play a central role. SMC methods—in which the *sequential importance sampling* and *sampling importance resampling* methods proposed by [23] and [35], respectively, are combined—refer to a class of algorithms approximating a *sequence of probability distributions*, defined on a *sequence of probability spaces*, by updating recursively a set of random *particles* with associated nonnegative *importance weights*. The SMC methodology has emerged as a key tool for approximating state posterior distribution flows in general state space models; see [9, 10, 12] for general introductions as well as theoretical results for SMC methods and [17, 31, 33] for applications of SMC within a variety of scientific fields.

The recursive formulas generating the *filter distributions* ϕ_T (short-hand notation for $\phi_{T:T|T}$) and the *joint smoothing distributions* $\phi_{0:T|T}$ are closely related; thus, executing the standard SMC scheme in the filtering mode provides, as a by-product, approximations of the joint smoothing distributions. More specifically, the branches of the genealogical tree associated with the historical evolution of the filtering particles up to time step T form, when combined with the corresponding importance weights of these filtering particles, a weighted sample approximating the joint smoothing distribution $\phi_{0:T|T}$; see [9], Section 3.4, for details. From these paths, one may readily obtain a weighted sample targeting the fixed lag or fixed interval smoothing distribution by extracting the required subsequence of states while retaining the weights. This appealingly simple scheme can be used successfully for estimating the joint smoothing distribution for small values of T or any marginal smoothing distribution $\phi_{s|T}$, with $s \leq T$, when s and T are close; however, when T is large and $s \ll T$, the associated particle approximations are inaccurate since the genealogical tree degenerates gradually as the interacting particle system evolves [20, 21].

In this article, we thus give attention to more sophisticated approaches and consider instead the *forward filtering backward smoothing* (FFBSm) *algorithm* and the *forward filtering backward simulation* (FFBSi) *sampler*. These algorithms share some similarities with the Baum–Welch algorithm for finite state space models and the Kalman filter-based smoother and simulation smoother for linear Gaussian state space models [8]. In the FFBSm algorithm, the particle weights obtained when approximating the filter distributions in a forward filtering pass are modified in a backward pass; see [18, 24, 27]. The FFBSi algorithm simulates, conditionally independently given the particles and particle weights produced in a similar forward filtering pass, state trajectories being approximately distributed according to the joint smoothing distribution; see [21].

The computational complexity of the FFBSm algorithm when used for estimating marginal fixed interval smoothing distributions or of the original formulation of the FFBSi sampler grows (in most situations) as the square of the number N of particles multiplied by the time horizon T. To alleviate this potentially very large computational cost, some methods using intricate data structures for storing the particles have been developed; see, for example, [28]. These algorithms have a complexity of order $O(N\log(N))$ and are thus amenable to practical applications; however, this reduction in complexity comes at the cost of introducing some level of approximation.

In this paper, a modification of the original FFBSi algorithm is presented. The proposed scheme has a complexity that grows only *linearly* in N and does not involve any numerical approximation techniques. This algorithm may be seen as an alternative to a recent proposal by [20] which is based on the so-called *two-filter algorithm* [2].

The smoothing weights computed in the backward pass of the FFBSm algorithm at a given time instant s (or the law of the FFBSi algorithm) are statistically dependent on all forward filtering pass particles and weights computed before and after this time instant. This intricate dependence structure makes the analysis of the resulting particle approximation challenging; up to our best knowledge, only a single consistency result is available in [21], but its proof is plagued by a (subtle) mistake that seems difficult to correct. Therefore, very little is known about the convergence of the schemes under consideration, and the second purpose of this paper is to fill this gap.[2] In this contribution, we focus first on finite time horizon approximations. Given a finite time horizon T, we derive *exponential deviation inequalities* stating that the probability of obtaining, when replacing $\phi_{s:T|T}$ by the corresponding FFBSm or FFBSi estimator, a Monte Carlo error exceeding a given $\varepsilon > 0$ is bounded by a quantity of order $O(\exp(-cN\varepsilon^2))$ where c is positive constant depending on T as well as the target function under consideration. The obtained inequalities, which are presented in Theorem 5 (FFBSm) and Corollary 6 (FFBSi), hold for any given number N of particles and are obtained by combining a novel backward error decomposition with an adaptation of the Hoeffding inequality to statistics expressed as ratios of random variables. We then consider the asymptotic (as the number N of particles tends to infinity) regime and establish a central limit theorem (CLT) with rate $\sqrt{N}$ and with an explicit expression of the asymptotic variance; see

[2]Since the first version of this paper has been released, an article [11] has been published. This work, developed completely independently from ours, complement the results presented in this manuscript. In particular, this paper presents a functional central limit theorems as well as nonasymptotic variance bounds. Additionally, this work shows how the forward filtering backward smoothing estimates of additive functionals can be computed using a forward only recursion.

Theorem 8. The proof of our CLT relies on a technique, developed gradually in [6, 15, 30], which is based on a CLT for triangular arrays of dependent random variables; however, since we are required to take the complex dependence structure of the smoothing weights into account, our proof is significantly more involved than in the standard filtering framework considered in the mentioned works.

The second part of the paper is devoted to time uniform results, and we here study the behavior of the particle-based marginal smoothing distribution approximations as the time horizon T tends to infinity. In this setting, we first establish, under the assumption that the Markov transition kernel M of the latent signal is strongly mixing (Assumption 4), time uniform deviation bounds of the type described above which hold for any particle population size N and where the constant c is *independent* of T; see Theorem 11. This result may seem surprising, and the nonobvious reason for its validity stems from the fact that the underlying Markov chain forgets, when evolving conditionally on the observations, its initial conditions in the forward *as well as* the backward directions. Finally, we prove (see Theorem 12), under the same uniform mixing assumption, that the asymptotic variance of the CLT for the particle-based marginal smoothing distribution approximations remains bounded as T tends to infinity. The uniform mixing assumption in Assumption 4 points typically to applications where the state space of the latent signal is compact; nevertheless, in the light of recent results on filtering stability [14, 29] one may expect the geometrical contraction of the backward kernel to hold for a significantly larger class of nonuniformly mixing models (see [14] for examples from, e.g., financial economics). But even though the geometrical mixing rate is supposed to be constant in this more general case, applying the mentioned results will yield a bound of contraction containing a multiplicative constant depending highly on the initial distributions as well as the observation record under consideration. Since there are currently no available results describing this dependence, applying such bounds to the instrumental decomposition used in the proof of Theorem 5 seems technically involved. Recently, [39] managed to derive *qualitative* time average convergence results for standard (bootstrap-type) particle filters under a mild tightness assumption being satisfied also in the noncompact case when the hidden chain is geometrically ergodic. Even though this technique does not (on the contrary to our approach) supply a rate of convergence, it could possibly be adopted to our framework in order to establish time average convergence of the particle-based marginal smoothing distribution approximations in a noncompact setting.

The paper is organized as follows. In Section 2, the FFBSm algorithm and the FFBSi sampler are introduced. An exponential deviation inequality for the fixed interval joint smoothing distribution is derived in Section 3.1, and a CLT is established in Section 3.2. In Section 4, time uniform exponential

bounds on the error of the FFBSm marginal smoothing distribution estimator are computed under the mentioned mixing condition on the kernel M. Finally, under the same mixing condition, an explicit bound on the asymptotic variance of the marginal smoothing distribution estimator is derived in Section 4.2.

Notation and definitions. For any sequence $\{a_n\}_{n\geq 0}$ and any pair of integers $0 \leq m \leq n$, we denote $a_{m:n} \stackrel{\text{def}}{=} (a_m, \ldots, a_n)$. We assume in the following that all random variables are defined on a common probability space $(\Omega, \mathcal{F}, \mathbb{P})$. The sets $\mathbb{X}$ and $\mathbb{Y}$ are supposed to be Polish spaces and we denote by $\mathcal{B}(\mathbb{X})$ and $\mathcal{B}(\mathbb{Y})$ the associated Borel σ-algebras. $\mathcal{F}_{\mathrm{b}}(\mathbb{X})$ denotes the set of all bounded $\mathcal{B}(\mathbb{X})/\mathcal{B}(\mathbb{R})$-measurable functions from $\mathbb{X}$ to $\mathbb{R}$. For any measure ζ on $(\mathbb{X}, \mathcal{B}(\mathbb{X}))$ and any ζ-integrable function f, we set $\zeta(f) \stackrel{\text{def}}{=} \int_{\mathbb{X}} f(x)\zeta(dx)$. Two measures ζ and ζ' are said to be *proportional* (written $\zeta \propto \zeta'$) if they differ only by a normalization constant.

A kernel V from $(\mathbb{X}, \mathcal{B}(\mathbb{X}))$ to $(\mathbb{Y}, \mathcal{B}(\mathbb{Y}))$ is a mapping from $\mathbb{X} \times \mathcal{B}(\mathbb{Y})$ into $[0,1]$ such that, for each $A \in \mathcal{B}(\mathbb{Y})$, $x \mapsto V(x, A)$ is a nonnegative, bounded, and measurable function on $\mathbb{X}$, and, for each $x \in \mathbb{X}$, $A \mapsto V(x, A)$ is a measure on $\mathcal{B}(\mathbb{Y})$. For $f \in \mathcal{F}_{\mathrm{b}}(\mathbb{X})$ and $x \in \mathbb{X}$, denote by $V(x, f) \stackrel{\text{def}}{=} \int V(x, dx')f(x')$; we will sometimes also use the abridged notation $Vf(x)$ instead of $V(x, f)$. For a measure ν on $(\mathbb{X}, \mathcal{B}(\mathbb{X}))$, we denote by νV the measure on $(\mathbb{Y}, \mathcal{B}(\mathbb{Y}))$ defined by, for any $A \in \mathcal{B}(\mathbb{Y})$, $\nu V(A) \stackrel{\text{def}}{=} \int_{\mathbb{X}} V(x, A)\nu(dx)$.

Consider now a possibly nonlinear state space model, where the *state process* $\{X_t\}_{t\geq 0}$ is a Markov chain on the state space $(\mathbb{X}, \mathcal{B}(\mathbb{X}))$. Even though t is not necessarily a temporal index, we will often refer to this index as "time." We denote by χ and M the initial distribution and transition kernel, respectively, of this process. The state process is assumed to be hidden but partially observed through the *observations* $\{Y_t\}_{t\geq 0}$ which are $\mathbb{Y}$-valued random variables being conditionally independent given the latent state sequence $\{X_t\}_{t\geq 0}$; in addition, there exists a σ-finite measure λ on $(\mathbb{Y}, \mathcal{B}(\mathbb{Y}))$ and a nonnegative transition density function g on $\mathbb{X} \times \mathbb{Y}$ such that $\mathbb{P}[Y_t \in A | X_t] = \int_A g(X_t, y)\lambda(dy)$ for all $A \in \mathcal{B}(\mathbb{Y})$. The mapping $x \mapsto g(x, y)$ is referred to as the *likelihood function* of the state given an observed value $y \in \mathbb{Y}$. The kernel M as well as the transition density g are supposed to be known. In the setting of this paper, we assume that we have access to a record of arbitrary but fixed observations $y_{0:T} \stackrel{\text{def}}{=} [y_0, \ldots, y_T]$, and our main task is to estimate the posterior distribution of (different subsets of) the state vector $X_{0:T}$ given these observations. For any $t \geq 0$, we denote by $g_t(x) \stackrel{\text{def}}{=} g(x, y_t)$ (where the dependence on y_t is implicit) the likelihood function of the state X_t given the observation y_t.

For simplicity, we consider a *fully dominated* state space model for which there exists a σ-finite measure ν on $(\mathbb{X}, \mathcal{B}(\mathbb{X}))$ such that, for all $x \in \mathbb{X}$, $M(x, \cdot)$

has a transition probability density $m(x,\cdot)$ with respect to ν. For notational simplicity, $\nu(dx)$ will sometimes be replaced by dx.

For any initial distribution χ on $(\mathbb{X},\mathcal{B}(\mathbb{X}))$ and any $0\le s\le s'\le T$, denote by $\phi_{s:s'|T}$ the posterior distribution of the state vector $X_{s:s'}$ given the observations $y_{0:T}$. For lucidity, the dependence of $\phi_{s:s'|T}$ on the initial distribution χ is omitted. Assuming that $\int\cdots\int\chi(dx_0)\prod_{u=1}^{T}g_{u-1}(x_{u-1})M(x_{u-1},dx_u)g_T(x_T)>0$, this distribution may be expressed as, for all $h\in\mathcal{F}_{\mathrm{b}}(\mathbb{X}^{s'-s+1})$,

$$\phi_{s:s'|T}(h)=\frac{\int\cdots\int\chi(dx_0)\prod_{u=1}^{T}g_{u-1}(x_{u-1})M(x_{u-1},dx_u)g_T(x_T)h(x_{s:s'})}{\int\cdots\int\chi(dx_0)\prod_{v=1}^{T}g_{v-1}(x_{v-1})M(x_{v-1},dx_v)g_T(x_T)}.$$

In the expression above, the dependence on the observation sequence is implicit. If $s=s'$, we use $\phi_{s|T}$ (the marginal smoothing distribution at time s) as shorthand for $\phi_{s:s|T}$. If $s=s'=T$, we denote by $\phi_s\overset{\mathrm{def}}{=}\phi_{s|s}$ the filtering distribution at time s.

2. Algorithms. Conditionally on the observations $y_{0:T}$, the state sequence $\{X_s\}_{s\ge0}$ is a time inhomogeneous Markov chain. This property remains true in the *time-reversed* direction. Denote by B_η the so-called *backward kernel* given by, for any probability measure η on $(\mathbb{X},\mathcal{B}(\mathbb{X}))$,

$$\mathrm{B}_\eta(x,h)\overset{\mathrm{def}}{=}\frac{\int\eta(dx')m(x',x)h(x')}{\int\eta(dx')m(x',x)},\qquad h\in\mathcal{F}_{\mathrm{b}}(\mathbb{X}). \tag{1}$$

The posterior distribution $\phi_{s:T|T}$ may be expressed as, for any integers $T>0$, $s\in\{0,\dots,T-1\}$ and any $h\in\mathcal{F}_{\mathrm{b}}(\mathbb{X}^{T-s+1})$,

$$\phi_{s:T|T}(h)=\int\cdots\int\phi_T(dx_T)\mathrm{B}_{\phi_{T-1}}(x_T,dx_{T-1})\cdots\mathrm{B}_{\phi_s}(x_{s+1},dx_s)h(x_{s:T}). \tag{2}$$

Therefore, the joint smoothing distribution may be computed recursively, backward in time, according to

$$\phi_{s:T|T}(h)=\int\cdots\int\mathrm{B}_{\phi_s}(x_{s+1},dx_s)\phi_{s+1:T|T}(dx_{s+1:T})h(x_{s:T}). \tag{3}$$

2.1. *The forward filtering backward smoothing algorithm.* As mentioned in the Introduction, the method proposed by [18, 24] for approximating the smoothing distribution is a two pass procedure. In the forward pass, particle approximations ϕ_s^N of the filter distributions ϕ_s are computed recursively for all time steps from $s=0$ up to $s=T$. The filter distribution flow $\{\phi_s\}_{s\ge0}$ satisfies the forward recursion

$$\phi_s(h)=\frac{\gamma_s(h)}{\gamma_s(\mathbf{1})}\qquad\text{where }\gamma_0(h)=\chi(g_0h),\gamma_s(h)\overset{\mathrm{def}}{=}\gamma_{s-1}M(g_sh),s\ge1, \tag{4}$$

for $h \in \mathcal{F}_{\mathrm{b}}(\mathbb{X})$, with $\mathbf{1}$ being the unity function $x \mapsto 1$ on $\mathbb{X}$. In terms of SMC, each filter distribution ϕ_s is approximated by means of a set of particles $\{\xi_s^i\}_{i=1}^N$ and associated importance weights $\{\omega_s^i\}_{i=1}^N$ according to

$$\phi_s^N(h) \stackrel{\mathrm{def}}{=} \frac{\gamma_s^N(h)}{\gamma_s^N(\mathbf{1})} \qquad \text{where } \gamma_s^N(h) \stackrel{\mathrm{def}}{=} N^{-1} \sum_{i=1}^N \omega_s^i h(\xi_s^i). \tag{5}$$

Having produced, using methods described in Section 2.4 below, a sequence of such weighted samples $\{(\xi_t^i, \omega_t^i)\}_{i=1}^N$, $1 \le t \le T$, an approximation of the smoothing distribution is constructed in a backward pass by replacing, in (2), the filtering distribution by its particle approximation. This yields

$$\phi_{s:T|T}^N(h) \stackrel{\mathrm{def}}{=} \int \cdots \int \phi_T^N(dx_T) \mathrm{B}_{\phi_{T-1}^N}(x_T, dx_{T-1}) \cdots \mathrm{B}_{\phi_s^N}(x_{s+1}, dx_s) h(x_{s:T}) \tag{6}$$

for any $h \in \mathcal{F}_{\mathrm{b}}(\mathbb{X}^{T-s+1})$. The approximation above can be computed recursively in the backward direction according to

$$\phi_{s:T|T}^N(h) = \int \cdots \int \mathrm{B}_{\phi_s^N}(x_{s+1}, dx_s) \phi_{s+1:T|T}^N(dx_{s+1:T}) h(x_{s:T}). \tag{7}$$

Now, by definition,

$$\mathrm{B}_{\phi_s^N}(x, h) = \sum_{i=1}^N \frac{\omega_s^i m(\xi_s^i, x)}{\sum_{\ell=1}^N \omega_s^\ell m(\xi_s^\ell, x)} h(\xi_s^i), \qquad h \in \mathcal{F}_{\mathrm{b}}(\mathbb{X}),$$

and inserting this expression into (6) gives

$$\phi_{s:T|T}^N(h) = \sum_{i_s=1}^N \cdots \sum_{i_T=1}^N \left(\prod_{u=s+1}^T \frac{\omega_{u-1}^{i_{u-1}} m(\xi_{u-1}^{i_{u-1}}, \xi_u^{i_u})}{\sum_{\ell=1}^N \omega_{u-1}^\ell m(\xi_{u-1}^\ell, \xi_u^{i_u})} \right) \frac{\omega_T^{i_T}}{\Omega_T} h(\xi_s^{i_s}, \ldots, \xi_T^{i_T}), \tag{8}$$

of $\phi_{s:T|T}(h)$, where $h \in \mathcal{F}_{\mathrm{b}}(\mathbb{X}^{T-s-1})$ and

$$\Omega_t \stackrel{\mathrm{def}}{=} \sum_{i=1}^N \omega_t^i. \tag{9}$$

The estimator $\phi_{s:T|T}^N$ is impractical since the cardinality of its support grows exponentially with the number $T - s$ of time steps; nevertheless, it plays a key role in the theoretical developments that follow. A more practical approximation of this quantity will be defined in the next section. When the dimension of the input space is moderate, the computational cost of evaluating the estimator can be reduced to $O(N \log N)$ by using the *fast multipole method* as suggested in [28]; note, however, that this method involves approximations that introduce some bias. On the other hand, in certain specific scenarios, such as discrete Markov chains with sparse transition matrices over large state spaces, the complexity can even be reduced to $O(NT)$ without any truncation; see [1].

2.2. *The forward filtering backward simulation algorithm.* The estimator (8) may be understood alternatively by noting that the normalized smoothing weights define a probability distribution on the set $\{1,\dots,N\}^{T-s}$ of trajectories associated with an inhomogeneous Markov chain. Indeed, consider, for $t \in \{0,\dots,T-1\}$, the Markov transition matrix $\{\Lambda_t^N(i,j)\}_{i,j=1}^N$ given by

$$\Lambda_t^N(i,j) = \frac{\omega_t^j m(\xi_t^j, \xi_{t+1}^i)}{\sum_{\ell=1}^N \omega_t^\ell m(\xi_t^\ell, \xi_{t+1}^i)}, \qquad (i,j) \in \{1,\dots,N\}^2. \tag{10}$$

For $1 \le t \le T$, denote by

$$\mathcal{F}_t^N \stackrel{\text{def}}{=} \sigma\{Y_{0:T}, (\xi_s^i, \omega_s^i); 0 \le s \le t, 1 \le i \le N\} \tag{11}$$

the σ-algebra generated by the observations from time 0 to time T as well as the particles and importance weights produced in the forward pass up to time t. The transition probabilities defined in (10) induce an inhomogeneous Markov chain $\{J_u\}_{u=0}^T$ evolving backward in time as follows. At time T, the random index J_T is drawn from the set $\{1,\dots,N\}$ such that J_T takes the value i with a probability proportional to ω_T^i. At time $t \le T-1$ and given that the index J_{t+1} was drawn at time step $t+1$, the index J_t is drawn from the set $\{1,\dots,N\}$ such that J_t takes the value j with probability $\Lambda_t^N(J_t, j)$. The joint distribution of $J_{0:T}$ is therefore given by, for $j_{0:T} \in \{1,\dots,N\}^{T+1}$,

$$\mathbb{P}[J_{0:T} = j_{0:T} | \mathcal{F}_T^N] = \frac{\omega_T^{j_T}}{\Omega_T} \Lambda_T^N(J_T, j_{T-1}) \cdots \Lambda_0^N(j_1, j_0). \tag{12}$$

Thus, and this is a key observation, the FFBS estimator (8) of the joint smoothing distribution may be written as the conditional expectation

$$\phi_{0:T|T}^N(h) = \mathbb{E}[h(\xi_0^{J_0}, \dots, \xi_T^{J_T}) | \mathcal{F}_T^N], \qquad h \in \mathcal{F}_{\mathrm{b}}(\mathbb{X}^{T+1}). \tag{13}$$

We may therefore construct an unbiased estimator of the FFBS estimator by drawing, conditionally independently given $\mathcal{F}_T^N$, N paths of $\{J_{0:T}^\ell\}_{\ell=1}^N$ of the inhomogeneous Markov chain introduced above and then forming the (practical) estimator

$$\tilde{\phi}_{0:T|T}^N(h) = N^{-1} \sum_{\ell=1}^N h(\xi_0^{J_0^\ell}, \dots, \xi_T^{J_T^\ell}), \qquad h \in \mathcal{F}_{\mathrm{b}}(\mathbb{X}^{T+1}). \tag{14}$$

This practical estimator was introduced in [21] (Algorithm 1, page 158). For ease of notation, we have here simulated N replicates of the backward, index-valued Markov chain, but it would of course also be possible to sample a number of paths that is either larger or smaller than N. The estimator $\phi_{0:T|T}^N$ may be seen as a Rao–Blackwellized version of $\tilde{\phi}_{0:T|T}^N$. The variance

of the latter is increased, but the gain in computational complexity is significant. The associated algorithm is referred in the sequel to as the forward filtering backward simulation (FFBSi) algorithm. In Section 4, forgetting properties of the inhomogeneous backward chain will play a key role when establishing time uniform stability properties of the proposed smoothing algorithm.

The computational complexity for sampling a single path of $J_{0:T}$ is $O(NT)$; therefore, the overall computational effort spent when estimating $\tilde{\phi}^N_{0:T|T}$ using the FFBSi sampler is $O(N^2T)$. Following [28], this complexity can be reduced further to $O(N\log(N)T)$ by means of the fast multipole method; however, here again computational work is gained at the cost of introducing additional approximations.

2.3. *A fast version of the forward filtering backward simulation algorithm.* We are now ready to describe one of the main contributions of this paper, namely a novel version of the FFBSi algorithm that can be proved to reach linear computational complexity under appropriate assumptions. At the end of the filtering phase of the FFBSi algorithm, all weighted particle samples $\{(\xi^i_s,\omega^i_s)\}^N_{i=1}$, $0\le s\le T$, are available, and it remains to sample efficiently index paths $\{J^\ell_{0:T}\}^N_{\ell=1}$ under the distribution (12). When the transition kernel m is bounded from above in the sense that $m(x,x')\le\sigma_+$ for all $(x,x')\in\mathbb{X}\times\mathbb{X}$, the paths can be simulated recursively backward in time using the following accept–reject procedure. As in the standard FFBSi algorithm, the recursion is initiated by sampling $J^1_T,\ldots,J^N_T$ multinomially with probabilities proportional to $\{\omega^i_T\}^N_{i=1}$. For $s\in\{0,\ldots,T\}$, let $\mathcal{G}^N_s$ the smallest σ-field containing $\mathcal{F}^N_T$ and $\sigma(J^\ell_t:1\le l\le N,t\ge s)$; then in order to draw J^ℓ_s conditionally on $\mathcal{G}^N_{s+1}$, we draw, first, an index proposal I^ℓ_s taking the value $i\in\{1,\ldots,N\}$ with a probability proportional to ω^i_t and, second, an independent uniform random variable U^ℓ_s on $[0,1]$. Then we set $J^\ell_s=I^\ell_s$ if $U^\ell_s\le m(\xi^{I^\ell_s}_s,\xi^{J^\ell_{s+1}}_{s+1})/\sigma_+$; otherwise, we reject the proposed index and make another trial. To create samples of size $n\in\{1,\ldots,N\}$ from a multinomial distribution on a set of N elements at lines 1 and 6, Algorithm 1 relies on an efficient procedure described in Appendix B.1 that requires $O(n(1+\log(1+N/n)))$ elementary operations; see Proposition 14. Using this technique, the computational complexity of Algorithm 1 can be upper-bounded as follows.

For the bootstrap particle filter as well as the fully adapted auxiliary particle filter (see Section 2.4 for precise descriptions of these SMC filters), it is possible to derive an asymptotic expression for the number of simulations required at line 8 of Algorithm 1 even if the kernel m is not bounded from below. The following result is obtained using theory derived in the coming section.

Algorithm 1 FFBSi-smoothing

sample $J_T^1, \dots, J_T^N$ multinomially with probabilities proportional to $\{\omega_T^i\}_{i=1}^N$
for s from $T-1$ down to 0 **do**
  $L \leftarrow (1, \dots, N)$
  while L is not empty **do**
    $n \leftarrow$ size(L)
    sample $I_1, \dots, I_n$ multinomially with probabilities proportional to $\{\omega_s^i\}_{i=1}^N$
    sample $U_1, \dots, U_n$ independently and uniformly over $[0,1]$
    $nL \leftarrow \varnothing$
    for k from 1 to n **do**
      if $U_k \leq m(\xi_s^{I(k)}, \xi_{s+1}^{J_{s+1}^{L(k)}})/\sigma_+$ **then**
        $J_s^{L(k)} \leftarrow I_k$
      else
        $nL \leftarrow nL \cup \{L(k)\}$
      end if
    end for
    $L \leftarrow nL$
  end while
end for

PROPOSITION 1. *Assume that the transition kernel is bounded from above, $m(x,x') \leq \sigma_+$ for all $(x,x') \in \mathbb{X} \times \mathbb{X}$. At each iteration $s \in \{0, \dots, T-1\}$, let Z_s^N be the number of simulations required in the accept–reject procedure of Algorithm 1.*

- *For the bootstrap auxiliary filter, Z_s^N/N converges in probability to*

$$\alpha(s) \stackrel{\text{def}}{=} \sigma_+ \phi_{s|s-1}(g_s) \frac{\int \cdots \int dx_{s+1} \prod_{u=s+2}^T \int m(x_{u-1}, dx_u) g_u(x_u)}{\int \cdots \int \phi_{s|s-1}(dx_s) g_s(x_s) \prod_{u=s+1}^T m(x_{u-1}, dx_u) g_u(x_u)}$$

  as N goes to infinity.
- *In the fully adapted case, Z_s^N/N converges in probability to*

$$\beta(s) \stackrel{\text{def}}{=} \sigma_+ \frac{\int \cdots \int dx_{s+1} \prod_{u=s+2}^T \int m(x_{u-1}, dx_u) g_u(x_u)}{\int \cdots \int \phi_s(dx_s) g_s(x_s) \prod_{u=s+1}^T m(x_{u-1}, dx_u) g_u(x_u)}$$

  as N goes to infinity.

A sufficient condition for ensuring finiteness of $\alpha(s)$ and $\beta(s)$ is that $\int g_u(x_u)\, dx_u < \infty$ for all $u \geq 0$.

If the transition kernel satisfies stronger mixing conditions, it is possible to derive an upper-bound on the computational complexity of the FFBSi for any auxiliary particle filter, that is, the total number of computations (and not only the total number of simulations). Note that this result is not limited to the bootstrap and the fully adapted cases.

PROPOSITION 2. *Assume that the transition kernel is bounded from below and above, that is,* $\sigma_- \leq m(x,x') \leq \sigma_+$ *for all* $(x,x') \in \mathbb{X}\times\mathbb{X}$. *Let* $C(N,T)$ *denote the number of elementary operations required in Algorithm 1. Then, there exists a constant* K *such that such that* $\mathbb{E}[C(N,T)] \leq KNT\sigma_+/\sigma_-$.

The proofs of Propositions 1 and 2 involve theory developed in the coming section and are postponed to Section 5.

Before concluding this section on reduced complexity, let us mention that efficient smoothing strategies have been considered by [19] using quasi-Monte Carlo methods. The smoother (restricted to be one-dimensional) presented in this work has a complexity that grows quadraticly in the number of particles N; nevertheless, since the variance of the same decays as $O(N^{-2})$ (or faster) thanks to the use of quasi-random numbers, the method is equivalent to methods with complexity growing linearly in N [since the standard Monte Carlo variance is $O(N^{-1})$]. This solution is of course attractive; we are however not aware of extensions of this approach to multiple dimensions.

2.4. *Auxiliary particle filters.* It remains to describe in detail how to produce sequentially the weighted samples $\{(\xi_s^i,\omega_s^i)\}_{i=1}^N$, $0\leq s\leq T$, which can be done in several different ways (see [3, 17, 31] and the references therein). Still, most algorithms may be formulated within the unifying framework of the *auxiliary particle filter* described in the following. Let $\{\xi_0^i\}_{i=1}^N$ be i.i.d. random variables such that $\xi_0^i \sim \rho_0$ and set $\omega_0^i \stackrel{\mathrm{def}}{=} d\chi/d\rho_0(\xi_0^i)g_0(\xi_0^i)$. The weighted sample $\{(\xi_0^i,\omega_0^i)\}_{i=1}^N$ then targets the initial filter ϕ_0 in the sense that $\phi_0^N(h)$ estimates $\phi_0(h)$ for $h\in\mathcal{F}_{\mathrm{b}}(\mathbb{X})$. In order to describe the sequential structure of the auxiliary particle filter, we proceed inductively and assume that we have at hand a weighted sample $\{(\xi_{s-1}^i,\omega_{s-1}^i)\}_{i=1}^N$ targeting ϕ_{s-1} in the same sense. Next, we aim at simulating new particles from the target $\phi_s^{N,\mathrm{t}}$ defined as

$$\phi_s^{N,\mathrm{t}}(h) = \frac{\gamma_{s-1}^N M(g_s h)}{\gamma_{s-1}^N M(g_s)}, \qquad h\in\mathcal{F}_{\mathrm{b}}(\mathbb{X}), \tag{15}$$

in order to produce an updated particle sample approximating the subsequent filter ϕ_s. Following [32], this may be done by considering the *auxiliary* target distribution

$$\phi_s^{N,\mathrm{a}}(i,h) \stackrel{\mathrm{def}}{=} \frac{\omega_{s-1}^i M(\xi_{s-1}^i, g_s h)}{\sum_{\ell=1}^N \omega_{s-1}^\ell M(\xi_{s-1}^\ell, g_s h)}, \qquad h\in\mathcal{F}_{\mathrm{b}}(\mathbb{X}), \tag{16}$$

on the product space $\{1,\dots,N\}\times\mathbb{X}$ equipped with the product σ-algebra $\mathcal{P}(\{1,\dots,N\})\otimes\mathcal{B}(\mathbb{X})$. By construction, $\phi_s^{N,\mathrm{t}}$ is the marginal distribution of $\phi_s^{N,\mathrm{a}}$ with respect to the particle index. Therefore, we may approximate the target distribution $\phi_s^{N,\mathrm{t}}$ on $(\mathbb{X},\mathcal{B}(\mathbb{X}))$ by simulating from the auxiliary distribution and then discarding the indices. More specifically, we first simulate pairs $\{(I_s^i,\xi_s^i)\}_{i=1}^N$ of indices and particles from the instrumental distribution

$$\pi_{s|s}(i,h)\propto\omega_{s-1}^i\vartheta_s(\xi_{s-1}^i)P_s(\xi_{s-1}^i,h),\qquad h\in\mathcal{F}_{\mathrm{b}}(\mathbb{X}),\tag{17}$$

on the product space $\{1,\dots,N\}\times\mathbb{X}$, where $\{\vartheta_s(\xi_{s-1}^i)\}_{i=1}^N$ are so-called *adjustment multiplier weights* and P_s is a Markovian *proposal* transition kernel. In the sequel, we assume for simplicity that $P_s(x,\cdot)$ has, for any $x\in\mathbb{X}$, a density $p_s(x,\cdot)$ with respect to the reference measure ν. For each draw (I_s^i,ξ_s^i), $i=1,\dots,N$, we compute the importance weight

$$\omega_s^i\stackrel{\mathrm{def}}{=}\frac{m(\xi_{s-1}^{I_s^i},\xi_s^i)g_s(\xi_s^i)}{\vartheta_s(\xi_{s-1}^{I_s^i})p_s(\xi_{s-1}^{I_s^i},\xi_s^i)},\tag{18}$$

such that $\omega_s^i\propto d\phi_s^{N,\mathrm{a}}/d\pi_{s|s}(I_s^i,\xi_s^i)$, and associate it to the corresponding particle position ξ_s^i. Finally, the indices $\{I_s^i\}_{i=1}^N$ are discarded whereupon $\{(\xi_s^i,\omega_s^i)\}_{i=1}^N$ is taken as an approximation of ϕ_s. The simplest choice, yielding to the so-called *bootstrap particle filter algorithm* proposed by [22], consists of setting, for all $x\in\mathbb{X}$, $\vartheta_s(x)\equiv1$ and $p_s(x,\cdot)\equiv m(x,\cdot)$. A more appealing—but often computationally costly—choice consists of using the adjustment weights $\vartheta_s(x)\equiv\vartheta_s^\star(x)\stackrel{\mathrm{def}}{=}\int m(x,x')g_s(x')\,dx'$, $x\in\mathbb{X}$, and the proposal transition density

$$p_s^\star(x,x')\stackrel{\mathrm{def}}{=}\frac{m(x,x')g_s(x')}{\vartheta_s^\star(x)},\qquad(x,x')\in\mathbb{X}\times\mathbb{X}.$$

In this case, the auxiliary particle filter is referred to as *fully adapted*. Other choices are discussed in [16] and [7].

3. Convergence of the FFBS and FFBSi algorithms. In this section, the convergence of the FFBS and FFBSi algorithms are studied. For these two algorithms, nonasymptotic Hoeffding-type deviation inequalities and CLTs are obtained. We also introduce a decomposition, serving as a basis for most results obtained in this paper, of the error $\phi_{0:T|T}^N-\phi_{0:T|T}$ and some technical conditions under which the results are derived.

For any function $f:\mathbb{X}^d\to\mathbb{R}$, we define by $|f|_\infty\stackrel{\mathrm{def}}{=}\sup_{x\in\mathbb{X}^d}|f(x)|$ and $\operatorname{osc}(f)\stackrel{\mathrm{def}}{=}\sup_{(x,x')\in\mathbb{X}^d\times\mathbb{X}^d}|f(x)-f(x')|$ the supremum and oscillator norms, respectively. Denote $\bar{\mathbb{N}}\stackrel{\mathrm{def}}{=}\mathbb{N}\cup\{\infty\}$ and consider the following assumptions where T is the time horizon which can be either a finite integer or infinity.

ASSUMPTION 1. For all $0 \le t \le T$, $g_t(\cdot) > 0$ and $\sup_{0\le t\le T} |g_t|_\infty < \infty$.

Define for $t \ge 0$ the importance weight functions

$$\omega_0(x) \stackrel{\text{def}}{=} \frac{d\chi}{d\rho_0}(x) g_0(x) \quad \text{and} \quad \omega_t(x,x') \stackrel{\text{def}}{=} \frac{m(x,x') g_t(x')}{\vartheta_t(x) p_t(x,x')}, \qquad t \ge 1. \tag{19}$$

ASSUMPTION 2. $\sup_{1\le t\le T} |\vartheta_t|_\infty < \infty$ and $\sup_{0\le t\le T} |\omega_t|_\infty < \infty$.

The latter assumption is rather mild; it holds in particular under Assumption 1 for the bootstrap filter ($p_t = m$ and $\vartheta_t \equiv 1$) and is automatically fulfilled in the fully adapted case ($\omega_t \equiv 1$).

The coming proofs are based on a decomposition of the joint smoothing distribution that we introduce below. For $0 \le t < T$ and $h \in \mathcal{F}_{\mathrm{b}}(\mathbb{X}^{T+1})$, define the kernel $L_{t,T} : \mathbb{X}^{t+1} \times \mathcal{B}(\mathbb{X})^{\otimes T+1} \to [0,1]$ by

$$L_{t,T}(x_{0:t}, h) \stackrel{\text{def}}{=} \int \cdots \int \left(\prod_{u=t+1}^{T} M(x_{u-1}, dx_u) g_u(x_u) \right) h(x_{0:T}) \tag{20}$$

and set $L_{T,T}(x_{0:T}, h) \stackrel{\text{def}}{=} h(x_{0:T})$. By construction, for every $t \in \{0, \ldots, T\}$, the joint smoothing distribution may be expressed as

$$\phi_{0:T|T}(h) = \frac{\phi_{0:t|t}[L_{t,T}(\cdot, h)]}{\phi_{0:t|t}[L_{t,T}(\cdot, \mathbf{1})]}. \tag{21}$$

This expression extends the classical forward–backward decomposition to the joint smoothing distribution; here $L_{t,T}(\cdot, h)$ plays the role of the so-called backward variable. This suggests to decompose the error $\phi^N_{0:T|T}(h) - \phi_{0:T|T}(h)$ as the following telescoping sum:

$$\begin{aligned} \phi^N_{0:T|T}(h) - \phi_{0:T|T}(h) &= \frac{\phi^N_0[L_{0,T}(\cdot, h)]}{\phi^N_0[L_{0,T}(\cdot, \mathbf{1})]} - \frac{\phi_0[L_{0,T}(\cdot, h)]}{\phi_0[L_{0,T}(\cdot, \mathbf{1})]} \\ &\quad + \sum_{t=1}^{T} \left\{ \frac{\phi^N_{0:t|t}[L_{t,T}(\cdot, h)]}{\phi^N_{0:t|t}[L_{t,T}(\cdot, \mathbf{1})]} - \frac{\phi^N_{0:t-1|t-1}[L_{t-1,T}(\cdot, h)]}{\phi^N_{0:t-1|t-1}[L_{t-1,T}(\cdot, \mathbf{1})]} \right\}. \end{aligned} \tag{22}$$

The first term on RHS of the decomposition above can be easily dealt with since ϕ^N_0 is a weighted empirical distribution associated to i.i.d. random variables.

To cope with the terms in the sum of the RHS in (22), we introduce some kernels (depending on the *past* particles) that stress the dependence with respect to the *current* particules. More precisely, $\phi^N_{0:t|t}[L_{t,T}(\cdot, h)]$ is expressed as

$$\phi^N_{0:t|t}[L_{t,T}(\cdot, h)] = \phi^N_t[\mathcal{L}^N_{t,T}(\cdot, h)] = \frac{\gamma^N_t[\mathcal{L}^N_{t,T}(\cdot, h)]}{\gamma^N_t(\mathbf{1})}, \tag{23}$$

where the random kernels $\mathcal{L}_{t,T}^N : \mathbb{X} \times \mathcal{B}(\mathbb{X})^{\otimes(T+1)} \to [0,1]$ are defined by: for all $0 < t \le T$, and $x_t \in \mathbb{X}$,

$$\mathcal{L}_{t,T}^N(x_t, h) \stackrel{\text{def}}{=} \int \cdots \int \mathrm{B}_{\phi_{t-1}^N}(x_t, dx_{t-1}) \cdots \mathrm{B}_{\phi_0^N}(x_1, dx_0) L_{t,T}(x_{0:t}, h), \tag{24}$$

and

$$\mathcal{L}_{0,T}^N(x, h) \stackrel{\text{def}}{=} L_{0,T}(x, h). \tag{25}$$

We stress that the kernels $\mathcal{L}_{t,T}^N$ depend on the particles and weights $(\xi_s^i, \omega_s^i)_{i=1}^N$, $0 \le s \le t-1$, through the particle approximations $\phi_{t-1}^N, \ldots, \phi_0^N$ of the filter distributions. When proving the CLT for the FFBS algorithm, it will be crucial to establish that for any $h \in \mathcal{F}_{\mathrm{b}}(\mathbb{X}^{T+1})$, $\mathcal{L}_{t,T}^N(\cdot, h)$ converges (see Lemma 7 below), as the number N of particles tends to infinity, to a deterministic function $\mathcal{L}_{t,T}(\cdot, h)$ given by

$$\mathcal{L}_{t,T}(x_t, h) \stackrel{\text{def}}{=} \int \cdots \int \mathrm{B}_{\phi_{t-1}}(x_t, dx_{t-1}) \cdots \mathrm{B}_{\phi_0}(x_1, dx_0) L_{t,T}(x_{0:t}, h). \tag{26}$$

In the sequel, the case $h = \mathbf{1}$ will be of particular importance; in that case, $L_{t,T}(x_{0:t}, \mathbf{1})$ does not depend on $x_{0:t-1}$, yielding

$$\mathcal{L}_{t,T}^N(x_t, \mathbf{1}) = \mathcal{L}_{t,T}(x_t, \mathbf{1}) = L_{t,T}(x_{0:t}, \mathbf{1}) \tag{27}$$

for all $x_{0:t} \in \mathbb{X}^{t+1}$. Using these functions, the difference appearing in the sum in (22) may then be rewritten as

$$\begin{aligned} &\frac{\phi_{0:t|t}^N[L_{t,T}(\cdot, h)]}{\phi_{0:t|t}^N[L_{t,T}(\cdot, \mathbf{1})]} - \frac{\phi_{0:t-1|t-1}^N[L_{t-1,T}(\cdot, h)]}{\phi_{0:t-1|t-1}^N[L_{t-1,T}(\cdot, \mathbf{1})]} \\ &\quad = \frac{1}{\gamma_t^N[\mathcal{L}_{t,T}^N(\cdot, \mathbf{1})]} \left(\gamma_t^N[\mathcal{L}_{t,T}^N(\cdot, h)] - \frac{\phi_{t-1}^N[\mathcal{L}_{t-1,T}^N(\cdot, h)]}{\phi_{t-1}^N[\mathcal{L}_{t-1,T}^N(\cdot, \mathbf{1})]} \gamma_t^N[\mathcal{L}_{t,T}^N(\cdot, \mathbf{1})] \right) \\ &\quad = \frac{N^{-1} \sum_{\ell=1}^N \omega_t^\ell G_{t,T}^N(\xi_t^\ell, h)}{N^{-1} \sum_{\ell=1}^N \omega_t^\ell \mathcal{L}_{t,T}(\xi_t^\ell, \mathbf{1})}, \end{aligned} \tag{28}$$

where the kernel $G_{t,T}^N : \mathbb{X} \times \mathcal{B}(\mathbb{X})^{T+1} \to [0,1]$ is defined by, for $x \in \mathbb{X}$,

$$G_{t,T}^N(x, h) \stackrel{\text{def}}{=} \mathcal{L}_{t,T}^N(x, h) - \frac{\phi_{t-1}^N[\mathcal{L}_{t-1,T}^N(\cdot, h)]}{\phi_{t-1}^N[\mathcal{L}_{t-1,T}^N(\cdot, \mathbf{1})]} \mathcal{L}_{t,T}^N(x, \mathbf{1}). \tag{29}$$

Similarly to $\mathcal{L}_{t,T}^N(\cdot, h)$, the functions $G_{t,T}^N(\cdot, h)$ depend on the past particles; it will however be shown (see Lemma 7 below) that $G_{t,T}^N(\cdot, h)$ converges to the deterministic function given by, for $x \in \mathbb{X}$,

$$G_{t,T}(x, h) \stackrel{\text{def}}{=} \mathcal{L}_{t,T}(x, h - \phi_{0:T|T}(h)). \tag{30}$$

The key property of this decomposition is stated in the following lemma.

LEMMA 3. *Assume that Assumptions 1–2 hold for some $T < \infty$. Then, for any $0 \le t \le T$, the variables $\{\omega_t^\ell G_{t,T}^N(\xi_t^\ell, h)\}_{\ell=1}^N$ are, conditionally on the σ-field $\mathcal{F}_{t-1}^N$, i.i.d. with zero mean. Moreover, there exists a constant C (that may depend on t and T) such that, for all $N \ge 1$, $\ell \in \{1, \dots, N\}$, and $h \in \mathcal{F}_{\mathrm{b}}(\mathbb{X}^{T+1})$,*

$$|\omega_t^\ell G_{t,T}^N(\xi_t^\ell, h)| \le |\omega_t|_\infty |G_{t,T}^N(\xi_t^\ell, h)| \le C \operatorname{osc}(h).$$

PROOF. By construction, all pairs of particles and weights of the weighted sample $\{(\xi_t^\ell, \omega_t^\ell)\}_{\ell=1}^N$ are i.i.d. conditionally on the σ-field $\mathcal{F}_{t-1}^N$. This implies immediately that the variables $\{\omega_t^\ell G_{t,T}^N(\xi_t^\ell, h)\}_{\ell=1}^N$ are also i.i.d. conditionally on the same σ-field $\mathcal{F}_{t-1}^N$. We now show that $\mathbb{E}[\omega_t^1 G_{t,T}^N(\xi_t^1, h)|\mathcal{F}_{t-1}^N] = 0$. Using the definition of $G_{t,T}^N$ and the fact that $\phi_{t-1}^N[\mathcal{L}_{t-1,T}^N(\cdot, h)]$ and $\phi_{t-1}^N[\mathcal{L}_{t-1,T}^N(\cdot, \mathbf{1})]$ are $\mathcal{F}_{t-1}^N$-measurable, we have

$$\begin{aligned}
&\mathbb{E}[\omega_t^1 G_{t,T}^N(\xi_t^1, h)|\mathcal{F}_{t-1}^N] \\
&\quad = \mathbb{E}[\omega_t^1 \mathcal{L}_{t,T}^N(x, h)|\mathcal{F}_{t-1}^N] - \frac{\phi_{t-1}^N[\mathcal{L}_{t-1,T}^N(\cdot, h)]}{\phi_{t-1}^N[\mathcal{L}_{t-1,T}^N(\cdot, \mathbf{1})]} \mathbb{E}[\omega_t^1 \mathcal{L}_{t,T}^N(x, \mathbf{1})|\mathcal{F}_{t-1}^N],
\end{aligned}$$

which is equal to zero provided that the relation

$$\mathbb{E}[\omega_t^1 \mathcal{L}_{t,T}^N(\xi_t^1, h)|\mathcal{F}_{t-1}^N] = \frac{\phi_{t-1}^N[\mathcal{L}_{t-1,T}^N(\cdot, h)]}{\phi_{t-1}^N(\vartheta_t)} \tag{31}$$

holds for any $h \in \mathcal{F}_{\mathrm{b}}(\mathbb{X})$. We now turn to the proof of (31). Note that for any $f \in \mathcal{F}_{\mathrm{b}}(\mathbb{X})$,

$$\begin{aligned}
\mathbb{E}[\omega_t^1 f(\xi_t^1)|\mathcal{F}_{t-1}^N] &= \frac{\sum_{\ell=1}^N \omega_{t-1}^\ell \int M(\xi_{t-1}^\ell, dx) g_t(x) f(x)}{\sum_{\ell=1}^N \omega_{t-1}^\ell \vartheta_t(\xi_{t-1}^\ell)} \\
&= \frac{\phi_{t-1}^N[M(\cdot, g_t f)]}{\phi_{t-1}^N(\vartheta_t)}.
\end{aligned} \tag{32}$$

It turns out that (31) is a consequence of (32) with $f(\cdot) = \mathcal{L}_{t,T}^N(\cdot, h)$, but since $\mathcal{L}_{t-1,T}^N(\cdot, h)$ is in general different from $M(\cdot, g_t \mathcal{L}_{t,T}^N(\cdot, h))$, we have to prove directly that

$$\phi_{t-1}^N[\mathcal{L}_{t-1,T}^N(\cdot, h)] = \phi_{t-1}^N[M(\cdot, g_t \mathcal{L}_{t,T}^N(\cdot, h))]. \tag{33}$$

Write

$$\begin{aligned}
&\phi_{t-1}^N[M(\cdot, g_t \mathcal{L}_{t,T}^N(\cdot, h))] \\
&\quad = \Omega_t^{-1} \sum_{\ell=1}^N \omega_{t-1}^\ell \int \cdots \int m(\xi_{t-1}^\ell, x_t) g_t(x_t) \left(\prod_{u=1}^t \mathrm{B}_{\phi_{u-1}^N}(x_u, dx_{u-1}) \right) \\
&\qquad \times L_{t,T}(x_{0:t}, h)\, dx_t.
\end{aligned} \tag{34}$$

To simplify the expression in the RHS, we will use the two following equalities:

$$\left(\sum_{\ell=1}^{N}\omega_{t-1}^{\ell}m(\xi_{t-1}^{\ell},x_t)\right)\mathrm{B}_{\phi_{t-1}^N}(x_t,dx_{t-1})=\sum_{\ell=1}^{N}\omega_{t-1}^{\ell}m(x_{t-1},x_t)\delta_{\xi_{t-1}^{\ell}}(dx_{t-1}), \tag{35}$$

$$\int M(x_{t-1},dx_t)g_t(x_t)L_{t,T}(x_{0:t},h)=L_{t-1,T}(x_{0:t-1},h). \tag{36}$$

The first relation is derived directly from the definition (1) of the backward kernel, the second is a recursive expression of $L_{t,T}$ which is straightforward from the definition (20). Now, (35) and (36) allow for writing

$$\begin{aligned}
&\sum_{\ell=1}^{N}\omega_{t-1}^{\ell}\int\cdots\int m(\xi_{t-1}^{\ell},x_t)g_t(x_t)\prod_{u=1}^{t}\mathrm{B}_{\phi_{u-1}^N}(x_u,dx_{u-1})L_{t,T}(x_{0:t},h)\,dx_t\\
&\quad=\sum_{\ell=1}^{N}\omega_{t-1}^{\ell}\int\cdots\int M(x_{t-1},dx_t)g_t(x_t)\delta_{\xi_{t-1}^{\ell}}(dx_{t-1})\\
&\qquad\qquad\times\prod_{u=1}^{t-1}\mathrm{B}_{\phi_{u-1}^N}(x_u,dx_{u-1})L_{t,T}(x_{0:t},h)\\
&\quad=\sum_{\ell=1}^{N}\omega_{t-1}^{\ell}\int\cdots\int\delta_{\xi_{t-1}^{\ell}}(dx_{t-1})\prod_{u=1}^{t-1}\mathrm{B}_{\phi_{u-1}^N}(x_u,dx_{u-1})L_{t-1,T}(x_{0:t-1},h)\\
&\quad=\sum_{\ell=1}^{N}\omega_{t-1}^{\ell}\mathcal{L}_{t-1}^{N}(\xi_{t-1}^{\ell},h).
\end{aligned}$$

By plugging this expression into (34), we obtain (33) from which (31) follows via (32). Finally, $\mathbb{E}[\omega_t^1 G_{t,T}^N(\xi_t^1,h)|\mathcal{F}_{t-1}^N]=0$. It remains to check that the random variable $\omega_t^1 G_{t,T}^N(\xi_t^1,h)$ is bounded. But this is immediate since

$$\begin{aligned}
|\omega_t^1 G_{t,T}^N(\xi_t^1,h)|&=|\omega_t|_\infty\left|\mathcal{L}_{t,T}^N(\cdot,h)-\frac{\phi_{t-1}^N[\mathcal{L}_{t-1,T}^N(\cdot,h)]}{\phi_{t-1}^N[\mathcal{L}_{t-1,T}^N(\cdot,\mathbf{1})]}\mathcal{L}_{t,T}(\cdot,\mathbf{1})\right|_\infty\\
&\le 2|\omega_t|_\infty|\mathcal{L}_{t,T}^N(\cdot,\mathbf{1})|_\infty\operatorname{osc}(h)\le 2|\omega_t|_\infty|L_{t,T}(\cdot,\mathbf{1})|_\infty\operatorname{osc}(h).
\end{aligned} \tag{37}$$

□

3.1. *Exponential deviation inequality.* We first establish a nonasymptotic deviation inequality. Considering (28), we are led to prove a Hoeffding inequality for ratios. For this purpose, we use the following elementary lemma which will play a key role in the sequel. The proof is postponed to Appendix A.

LEMMA 4. *Assume that a_N, b_N and b are random variables defined on the same probability space such that there exist positive constants β, B, C and M satisfying:*

(I) $|a_N/b_N| \le M$, $\mathbb{P}$-*a.s. and* $b \ge \beta$, $\mathbb{P}$-*a.s.*,
(II) *for all* $\varepsilon > 0$ *and all* $N \ge 1$, $\mathbb{P}[|b_N - b| > \varepsilon] \le Be^{-CN\varepsilon^2}$,
(III) *for all* $\varepsilon > 0$ *and all* $N \ge 1$, $\mathbb{P}[|a_N| > \varepsilon] \le Be^{-CN(\varepsilon/M)^2}$.

Then

$$\mathbb{P}\left(\left|\frac{a_N}{b_N}\right| > \varepsilon\right) \le B\exp\left(-CN\left(\frac{\varepsilon\beta}{2M}\right)^2\right).$$

THEOREM 5. *Assume that Assumptions 1–2 hold for some $T < \infty$. Then, there exist constants $0 < B$ and $C < \infty$ (depending on T) such that for all N, $\varepsilon > 0$, and all measurable functions $h \in \mathcal{F}_{\mathrm{b}}(\mathbb{X}^{T+1})$,*

$$\mathbb{P}[|\phi^N_{0:T|T}(h) - \phi_{0:T|T}(h)| \ge \varepsilon] \le Be^{-CN\varepsilon^2/\operatorname{osc}^2(h)}. \tag{38}$$

In addition,

$$N^{-1}\sum_{\ell=1}^{N} \omega_t^\ell \mathcal{L}_{t,T}(\xi_t^\ell, \mathbf{1}) \xrightarrow[N\to\infty]{\mathrm{P}} \frac{\phi_{t-1}[\mathcal{L}_{t-1,T}(\cdot, \mathbf{1})]}{\phi_{t-1}(\vartheta_t)}. \tag{39}$$

REMARK 1. As a by-product, Theorem 5 provides an exponential inequality for the particle approximation of the filter. For any $h \in \mathcal{F}_{\mathrm{b}}(\mathbb{X})$, define the function $h_{0:T} : \mathbb{X}^{T+1} \to \mathbb{R}$ by $h_{0:T}(x_{0:T}) = h(x_T)$. By construction, $\phi_{0:T|T}(h_{0:T}) = \phi_T(h)$ and $\phi^N_{0:T|T}(h_{0:T}) = \phi^N_T(h)$. With this notation, equation (38) may be rewritten as

$$\mathbb{P}[|\phi^N_T(h) - \phi_T(h)| \ge \varepsilon] \le Be^{-CN\varepsilon^2/\operatorname{osc}^2(h)}.$$

An inequality of this form was first obtained by [12] (see also [9], Chapter 7).

PROOF. We prove (38) by induction on T using the decomposition (22). Assume that (38) holds at time $T-1$, for $\phi^N_{0:T-1|T-1}(h)$. Let $h \in \mathcal{F}_{\mathrm{b}}(\mathbb{X}^{T+1})$ and assume without loss of generality that $\phi_{0:T|T}(h) = 0$. Then (21) implies that $\phi_0[L_{0,T}(\cdot, h)] = 0$ and the first term of the decomposition (22) thus becomes

$$\frac{\phi^N_0[L_{0,T}(\cdot, h)]}{\phi^N_0[L_{0,T}(\cdot, \mathbf{1})]} = \frac{N^{-1}\sum_{i=0}^{N} \frac{d\chi}{d\rho_0}(\xi_0^i) g_0(\xi_0^i) L_{0,T}(\xi_0^i, h)}{N^{-1}\sum_{\ell=0}^{N} \frac{d\chi}{d\rho_0}(\xi_0^\ell) g_0(\xi_0^\ell) L_{0,T}(\xi_0^\ell, \mathbf{1})}, \tag{40}$$

where $\{\xi_0^i\}_{i=1}^N$ are i.i.d. random variables with distribution ρ_0. We obtain an exponential inequality for (40) by applying Lemma 4 with

$$\begin{cases} a_N = N^{-1}\sum_{i=0}^{N} \dfrac{d\chi}{d\rho_0}(\xi_0^i) g_0(\xi_0^i) L_{0,T}(\xi_0^i, h), \\ b_N = N^{-1}\sum_{i=0}^{N} \dfrac{d\chi}{d\rho_0}(\xi_0^i) g_0(\xi_0^i) L_{0,T}(\xi_0^i, \mathbf{1}), \\ b = \beta = \chi[g_0(\cdot) L_{0,T}(\cdot, \mathbf{1})]. \end{cases}$$

Condition (I) is trivially satisfied and conditions (II) and (III) follow from the Hoeffding inequality for i.i.d. variables.

By (22) and (28), it is now enough to establish an exponential inequality for

$$\frac{\phi_{0:t|t}^N[L_{t,T}(\cdot, h)]}{\phi_{0:t|t}^N[L_{t,T}(\cdot, \mathbf{1})]} - \frac{\phi_{0:t-1|t-1}^N[L_{t-1,T}(\cdot, h)]}{\phi_{0:t-1|t-1}^N[L_{t-1,T}(\cdot, \mathbf{1})]} = \frac{N^{-1}\sum_{\ell=1}^N \omega_t^\ell G_{t,T}^N(\xi_t^\ell, h)}{N^{-1}\sum_{\ell=1}^N \omega_t^\ell \mathcal{L}_{t,T}(\xi_t^\ell, \mathbf{1})}, \tag{41}$$

where $0 < t \le T$. For that purpose, we use again Lemma 4 with

$$\begin{cases} a_N = N^{-1}\sum_{\ell=1}^{N} \omega_t^\ell G_{t,T}^N(\xi_t^\ell, h), \\ b_N = N^{-1}\sum_{\ell=1}^{N} \omega_t^\ell \mathcal{L}_{t,T}(\xi_t^\ell, \mathbf{1}), \\ b = \beta = \dfrac{\phi_{t-1}[\mathcal{L}_{t-1,T}(\cdot, \mathbf{1})]}{\phi_{t-1}(\vartheta_t)}. \end{cases} \tag{42}$$

By considering the LHS of (41), $|a_N/b_N| \le 2|h|_\infty$, verifying condition (I) in Lemma 4. By Lemma 3, Hoeffding's inequality implies that there exist constants B and C such that for all N, $\varepsilon > 0$, and all measurable function $h \in \mathcal{F}_{\mathrm{b}}(\mathbb{X}^{T+1})$,

$$\begin{aligned} &\mathbb{P}\left[\left| N^{-1}\sum_{\ell=1}^{N} \omega_t^\ell G_{t,T}^N(\xi_t^\ell, h)\right| \ge \varepsilon\right] \\ &\quad = \mathbb{E}\left[\mathbb{P}\left[\left| N^{-1}\sum_{\ell=1}^{N} \omega_t^\ell G_{t,T}^N(\xi_t^\ell, h)\right| \ge \varepsilon \middle| \mathcal{F}_{t-1}^N\right]\right] \le Be^{-CN\varepsilon^2/\operatorname{osc}^2(h)}, \end{aligned}$$

verifying condition (III) in Lemma 4. It remains to verify condition (II). Since the pairs of particles and weights of the weighted sample $\{(\xi_t^\ell, \omega_t^\ell)\}_{\ell=1}^N$ are i.i.d. conditionally on $\mathcal{F}_{t-1}^N$, Hoeffding's inequality implies that

$$\mathbb{P}\left[\left| b_N - \mathbb{E}\left[N^{-1}\sum_{\ell=1}^{N} \omega_t^\ell \mathcal{L}_{t,T}(\xi_t^\ell, \mathbf{1}) \middle| \mathcal{F}_{t-1}^N\right]\right| \ge \varepsilon\right] \le Be^{-CN\varepsilon^2}. \tag{43}$$

Moreover, by (32), (27), and the definition (20), we have

$$\begin{aligned}(44)\qquad &\mathbb{E}\left[N^{-1}\sum_{\ell=1}^{N}\omega_t^\ell \mathcal{L}_{t,T}(\xi_t^\ell,\mathbf{1})\Big|\mathcal{F}_{t-1}^N\right]-b\\ &\quad=\frac{\phi_{t-1}^N[\mathcal{L}_{t-1,T}(\cdot,\mathbf{1})]}{\phi_{t-1}^N(\vartheta_t)}-\frac{\phi_{t-1}[\mathcal{L}_{t-1,T}(\cdot,\mathbf{1})]}{\phi_{t-1}(\vartheta_t)}=\frac{\phi_{t-1}^N(H)}{\phi_{t-1}^N(\vartheta_t)},\end{aligned}$$

with $H(\cdot)\stackrel{\mathrm{def}}{=}\mathcal{L}_{t-1,T}(\cdot,\mathbf{1})-\phi_{t-1}[\mathcal{L}_{t-1,T}(\cdot,\mathbf{1})]\vartheta_t(\cdot)/\phi_{t-1}(\vartheta_t)$. To obtain an exponential deviation inequality for (44), we apply again Lemma 4 with

$$\begin{cases} a'_N=\phi_{t-1}^N(H),\\ b'_N=\phi_{t-1}^N(\vartheta_t),\\ b'=\beta'=\phi_{t-1}(\vartheta_t).\end{cases}$$

By using the inequality

$$\begin{aligned}&\mathcal{L}_{t-1,T}(x_{t-1},\mathbf{1})\\ &\quad=\vartheta_t(x_{t-1})\int\frac{m(x_{t-1},x_t)g_t(x_t)}{\vartheta_t(x_{t-1})p_t(x_{t-1},x_t)}p_t(x_{t-1},x_t)\mathcal{L}_{t,T}(x_t,\mathbf{1})\,dx_t\\ &\quad\le\vartheta_t(x_{t-1})|\omega_t|_\infty|\mathcal{L}_{t,T}(\cdot,\mathbf{1})|_\infty,\end{aligned}$$

we obtain the bound $|\phi_{t-1}^N(H)/\phi_{t-1}^N(\vartheta_t)|\le 2|\omega_t|_\infty|\mathcal{L}_{t,T}(\cdot,\mathbf{1})|_\infty$ which verifies condition (I). Now, since $t-1\le T-1$ and $\phi_{t-1}(H)=0$, the induction assumption implies that conditions (II) and (III) are satisfied for $|b'_N-b'|$ and $|a'_N|$. Hence, Lemma 4 shows that

$$(45)\qquad \mathbb{P}\left[\left|\mathbb{E}\left[N^{-1}\sum_{\ell=1}^{N}\omega_t^\ell\mathcal{L}_{t,T}(\xi_t^\ell,\mathbf{1})\Big|\mathcal{F}_{t-1}^N\right]-b\right|>\varepsilon\right]\le Be^{-CN\varepsilon^2}.$$

Finally, (43) and (45) ensure that condition (II) in Lemma 4 is satisfied and an exponential deviation inequality for (41) follows. The proof of (38) is complete. The last statement (39) of the theorem is a consequence of (43) and (45). □

The exponential inequality of Theorem 5 may be more or less immediately extended to the FFBSi estimator.

COROLLARY 6. *Under the assumptions of Theorem 5 there exist constants $0<B$ and $C<\infty$ (depending on T) such that for all N, $\varepsilon>0$, and all measurable functions h,*

$$(46)\qquad \mathbb{P}[|\tilde{\phi}_{0:T|T}^N(h)-\phi_{0:T|T}(h)|\ge\varepsilon]\le Be^{-CN\varepsilon^2/\operatorname{osc}^2(h)},$$

where $\tilde{\phi}_{0:T|T}^N(h)$ is defined in (14).

PROOF. Using (13) and the definition of $\tilde{\phi}^N_{s:T|T}(h)$, we may write

$$\tilde{\phi}^N_{0:T|T}(h) - \phi^N_{0:T|T}(h) = N^{-1}\sum_{\ell=1}^{N}[h(\xi_0^{J_0^\ell},\ldots,\xi_T^{J_T^\ell}) - \mathbb{E}[h(\xi_0^{J_0},\ldots,\xi_T^{J_T})|\mathcal{F}_T^N]],$$

which implies (46) by the Hoeffding inequality and (38). □

3.2. *Asymptotic normality.* We now extend the theoretical analysis of the forward-filtering backward-smoothing estimator (6) to a CLT. Consider the following mild assumption on the proposal distribution.

ASSUMPTION 3. $|m|_\infty < \infty$ and $\sup_{0\le t\le T}|p_t|_\infty < \infty$.

CLTs for interacting particle models have been established in [9, 12, 15]; the application to these results to auxiliary particle filters is presented in [25] and [16], Theorem 3.2. Here, we base our proof on techniques developed in [15] (extending [6] and [30]). As noted in the previous section, it turns out crucial that $G^N_{t,T}(\cdot,h)$ converges to a deterministic function as $N\to\infty$. This convergence is stated in the following lemma.

LEMMA 7. *Assume Assumptions 1–3. Then, for any $h\in\mathcal{F}_{\mathrm{b}}(\mathbb{X})$ and $x\in\mathbb{X}$,*

$$\lim_{N\to\infty}\mathcal{L}^N_{t,T}(x,h) = \mathcal{L}_{t,T}(x,h), \qquad \mathbb{P}\text{-a.s.},$$

$$\lim_{N\to\infty}G^N_{t,T}(x,h) = G_{t,T}(x,h), \qquad \mathbb{P}\text{-a.s.},$$

where $\mathcal{L}^N_{t,T}$, $\mathcal{L}_{t,T}$, $G^N_{t,T}$ and $G_{t,T}$ are defined in (24), (26), (29) and (30). Moreover, there exists a constant C (that may depend on t and T) such that for all $N\ge 1$, $\ell\in\{1,\ldots,N\}$, and $h\in\mathcal{F}_{\mathrm{b}}(\mathbb{X})$,

$$|\omega_t^\ell G_{t,T}(\xi_t^\ell,h)| \le |\omega_t|_\infty |G_{t,T}(\xi_t^\ell,h)| \le C\operatorname{osc}(h), \qquad \mathbb{P}\text{-a.s.}$$

PROOF OF LEMMA 7. Let $h\in\mathcal{F}_{\mathrm{b}}(\mathbb{X})$ and $x_t\in\mathbb{X}$. By plugging (1) with $\eta=\phi^N_{t-1}$ into the definition (24) of $\mathcal{L}^N_{t,T}(x_t,h)$, we obtain immediately

$$\begin{aligned}\mathcal{L}^N_{t,T}(x_t,h) &= \frac{\int\cdots\int \phi^N_{t-1}(dx_{t-1})\prod_{u=0}^{t-2}\mathrm{B}_{\phi^N_u}(x_{u+1},dx_u)m(x_{t-1},x_t)L_{t,T}(x_{0:t},h)}{\int\phi^N_{t-1}(dx_{t-1})m(x_{t-1},x_t)}\\ &= \frac{\phi^N_{0:t-1|t-1}[H([\cdot,x_t])]}{\phi^N_{t-1}[m(\cdot,x_t)]} \qquad \text{with } H(x_{0:t}) \overset{\text{def}}{=} m(x_{t-1},x_t)L_{t,T}(x_{0:t},h).\end{aligned}$$

The convergence of $\mathcal{L}_{t,T}^N(\cdot,h)$ follows from Theorem 5. The proof of the convergence of $G_{t,T}^N(\cdot,h)$ follows the same lines. Finally, the final statement of the lemma is derived from Lemma 3 and the almost sure convergence of $G_{t,T}^N(\cdot,h)$ to $G_{t,T}(\cdot,h)$. $\square$

Now, we may state the CLT with an asymptotic variance given by a finite sum of terms involving the limiting kernel $G_{t,T}$.

THEOREM 8. *Assume Assumptions 1–3. Then, for any* $h \in \mathcal{F}_{\mathrm{b}}(\mathbb{X}^{T+1})$,

$$\sqrt{N}(\phi_{0:T|T}^N(h) - \phi_{0:T|T}(h)) \xrightarrow{\mathcal{D}} \mathcal{N}(0, \Gamma_{0:T|T}[h]) \tag{47}$$

with

$$\Gamma_{0:T|T}[h] \stackrel{\mathrm{def}}{=} \frac{\rho_0[\omega_0^2(\cdot)G_{0,T}^2(\cdot,h)]}{\rho_0^2[\omega_0(\cdot)\mathcal{L}_{0,T}(\cdot,\mathbf{1})]} + \sum_{t=1}^{T} \frac{\phi_{t-1}[\upsilon_{t,T}(\cdot,h)]\phi_{t-1}(\vartheta_t)}{\phi_{t-1}^2[\mathcal{L}_{t-1,T}(\cdot,\mathbf{1})]}, \tag{48}$$

$$\upsilon_{t,T}(\cdot,h) \stackrel{\mathrm{def}}{=} \vartheta_t(\cdot) \int P_t(\cdot,dx)\omega_t^2(\cdot,x)G_{t,T}^2(x,h). \tag{49}$$

PROOF. Without loss of generality, we assume that $\phi_{0:T|T}(h) = 0$. We show that $\sqrt{N}\phi_{0:T|T}^N(h)$ may be expressed as

$$\sqrt{N}\phi_{0:T|T}^N(h) = \sum_{t=0}^{T} \frac{V_{t,T}^N(h)}{W_{t,T}^N}, \tag{50}$$

where the sequence of random vectors $[V_{0,T}^N(h), \ldots, V_{T,T}^N(h)]$ is asymptotically normal and $[W_{0,T}^N, \ldots, W_{T,T}^N]$ converge in probability to a deterministic vector. The proof of (47) then follows from Slutsky's lemma. Actually, the decomposition (50) follows immediately from the backward decomposition (22) by setting, for $t \in \{1, \ldots, T\}$,

$$V_{0,T}^N(h) \stackrel{\mathrm{def}}{=} N^{-1/2} \sum_{\ell=1}^{N} \frac{d\chi}{d\rho_0}(\xi_0^\ell) g_0(\xi_0^\ell) G_{0,T}(\xi_0^\ell, h),$$

$$V_{t,T}^N(h) \stackrel{\mathrm{def}}{=} N^{-1/2} \sum_{\ell=1}^{N} \omega_t^\ell G_{t,T}^N(\xi_t^\ell, h),$$

$$W_{0,T}^N \stackrel{\mathrm{def}}{=} N^{-1} \sum_{\ell=1}^{N} \frac{d\chi}{d\rho_0}(\xi_0^\ell) g_0(\xi_0^\ell) \mathcal{L}_{0,T}(\xi_0^\ell, \mathbf{1}),$$

$$W_{t,T}^N \stackrel{\mathrm{def}}{=} N^{-1} \sum_{\ell=1}^{N} \omega_t^\ell \mathcal{L}_{t,T}(\xi_t^\ell, \mathbf{1}).$$

The convergence

$$W_{0,T}^N \xrightarrow{\mathrm{P}}_{N\to\infty} \chi[g_0(\cdot)\mathcal{L}_{0,T}(\cdot,\mathbf{1})],$$
$$W_{t,T}^N \xrightarrow{\mathrm{P}}_{N\to\infty} \frac{\phi_{t-1}[\mathcal{L}_{t-1,T}(\cdot,\mathbf{1})]}{\phi_{t-1}(\vartheta_t)}$$

of $[W_{0,T}^N,\dots,W_{T,T}^N]$ to a deterministic vector is established immediately using (39) and noting that the initial particles $(\xi_0^i)_{i=1}^N$ are i.i.d. We devote the rest of the proof to showing that the sequence of random vectors $[V_{0,T}^N(h),\dots,V_{T,T}^N(h)]$ is asymptotically normal. Proceeding recursively in time, we prove by induction over $t\in\{0,\dots,T\}$ (starting with $t=0$) that $[V_{0,T}^N(h),\dots,V_{t,T}^N(h)]$ is asymptotically normal. More precisely, using the Cramér–Wold device, it is enough to show that for all scalars $(\alpha_0,\dots,\alpha_t)\in\mathbb{R}^{t+1}$,

$$\sum_{r=0}^{t}\alpha_r V_{r,T}^N(h) \xrightarrow{\mathcal{D}}_{N\to\infty} \mathcal{N}\Bigg(0,\sum_{r=0}^{t}\alpha_r^2\sigma_{r,T}^2[h]\Bigg), \tag{51}$$

where, for $r\ge 1$,

$$\sigma_{0,T}^2[h] \stackrel{\mathrm{def}}{=} \rho_0[\omega_0^2 G_{0,T}^2(\cdot,h)], \qquad \sigma_{t,T}^2[h] \stackrel{\mathrm{def}}{=} \frac{\phi_{t-1}[\upsilon_{t,T}(\cdot,h)]}{\phi_{t-1}(\vartheta_t)}.$$

The case $t=0$ is elementary since the initial particles $\{\xi_0^i\}_{i=1}^N$ are i.i.d. Assume now that (51) holds for some $t-1\le T$; for all scalars $(\alpha_1,\dots,\alpha_{t-1})\in\mathbb{R}^{t-1}$,

$$\sum_{r=s}^{t-1}\alpha_r V_{r,T}^N(h) \xrightarrow{\mathcal{D}}_{N\to\infty} \mathcal{N}\Bigg(0,\sum_{r=s}^{t-1}\alpha_r^2\sigma_{r,T}^2[h]\Bigg). \tag{52}$$

The sequence of random variable $V_{t,T}^N(h)$ may be expressed as an additive function of a triangular array of random variables,

$$V_{t,T}^N(h)=\sum_{\ell=1}^{N}U_{N,\ell}, \qquad U_{N,\ell}\stackrel{\mathrm{def}}{=}\omega_t^\ell G_{t,T}^N(\xi_t^\ell,h)/\sqrt{N},$$

where $G_{t,T}^N(x,h)$ is defined in (29). Lemma 3 implies that $\mathbb{E}[V_{t,T}^N(h)|\mathcal{F}_{t-1}^N]=0$, yielding

$$\mathbb{E}\Bigg[\sum_{r=0}^{t}\alpha_r V_{r,T}^N(h)\Bigg|\mathcal{F}_{t-1}^N\Bigg]=\sum_{r=0}^{t-1}\alpha_r V_{r,T}^N(h)\xrightarrow{\mathcal{D}}_{N\to\infty}\mathcal{N}\Bigg(0,\sum_{r=1}^{t-1}\alpha_r^2\sigma_{r,T}^2[h]\Bigg),$$

where the last limit follows by the induction assumption hypothesis (52). By [15], Theorem A.3, page 2360, as the random variables $\{U_{N,\ell}\}_{\ell=1}^N$ are centered and conditionally independent given $\mathcal{F}_{t-1}^N$, (51) holds provided that

the asymptotic smallness condition

$$\sum_{\ell=1}^{N} \mathbb{E}[U_{N,\ell}^2 \mathbb{1}_{\{|U_{N,\ell}|\geq\varepsilon\}} | \mathcal{F}_{t-1}^N] \xrightarrow{\mathrm{P}}_{N\to\infty} 0 \tag{53}$$

holds for any $\varepsilon > 0$ and that the conditional variance converges:

$$\sum_{\ell=1}^{N} \mathbb{E}[U_{N,\ell}^2 | \mathcal{F}_{t-1}^N] \xrightarrow{\mathrm{P}}_{N\to\infty} \sigma_{t,T}^2[h]. \tag{54}$$

Lemma 3 implies that $|U_{N,\ell}| \leq C \operatorname{osc}(h)/\sqrt{N}$, verifying immediately the asymptotic smallness condition (53). To conclude the proof, we thus only need to establish the convergence (54) of the asymptotic variance. Via Lemma 3 and straightforward computations, we conclude that

$$\begin{aligned}
\sum_{\ell=1}^{N} \mathbb{E}[U_{N,\ell}^2 | \mathcal{F}_{t-1}^N] &= \mathbb{E}[(\omega_t^1 G_{t,T}^N(\xi_t^1, h))^2 | \mathcal{F}_{t-1}^N] \\
&= \int \sum_{\ell=1}^{N} \frac{\omega_{t-1}^\ell \vartheta_t(\xi_{t-1}^\ell) P_t(\xi_{t-1}^\ell, dx)}{\sum_{j=1}^N \omega_{t-1}^j \vartheta_t(\xi_{t-1}^j)} (\omega_t(\xi_{t-1}^\ell, x) G_{t,T}^N(x,h))^2 \\
&= \left(\frac{\Omega_{t-1}}{\sum_{j=1}^N \omega_{t-1}^j \vartheta_t(\xi_{t-1}^j)} \right) \left(\frac{1}{\Omega_{t-1}} \sum_{\ell=1}^{N} \omega_{t-1}^\ell \upsilon_{t,T}^N(\xi_{t-1}^\ell, h) \right) \\
&= \frac{\phi_{t-1}^N[\upsilon_{t,T}^N(\cdot, h)]}{\phi_{t-1}^N(\vartheta_t)},
\end{aligned} \tag{55}$$

where Ω_t is defined in (9) and

$$\upsilon_{t,T}^N(\cdot, h) \stackrel{\mathrm{def}}{=} \vartheta_t(\cdot) \int P_t(\cdot, dx) \omega_t^2(\cdot, x) [G_{t,T}^N(x,h)]^2.$$

The denominator in on RHS of (55) converges evidently in probability to $\phi_{t-1}(\vartheta_t)$ by Theorem 5. The numerator is more complex since $\upsilon_{t,T}^N$ depends on $G_{t,T}^N$ whose definition involves all the approximations $\phi_{t-1}^N, \ldots, \phi_0^N$ of the past filters. To obtain its convergence, note that, by Theorem 5, $\phi_{t-1}^N(\upsilon_{t,T}(\cdot, h)) \xrightarrow{\mathrm{P}} \phi_{t-1}(\upsilon_{t,T}(\cdot, h))$ as N tends to infinity; hence, it only remains to prove that

$$\phi_{t-1}^N[\upsilon_{t,T}^N(\cdot, h) - \upsilon_{t,T}(\cdot, h)] \xrightarrow{\mathrm{P}}_{N\to\infty} 0. \tag{56}$$

For that purpose, introduce the following notation: for all $x \in \mathbb{X}$,

$$\begin{aligned}
A_N(x) &\stackrel{\mathrm{def}}{=} \phi_{t-1}^N[\vartheta_t(\cdot) p_t(\cdot, x) \omega_t^2(\cdot, x) | (G_{t,T}^N(x,h))^2 - G_{t,T}^2(x,h)|], \\
B_N(x) &\stackrel{\mathrm{def}}{=} \phi_{t-1}^N[\vartheta_t(\cdot) p_t(\cdot, x)].
\end{aligned}$$

Applying Fubini's theorem,

$$\lim_{N\to\infty} \mathbb{E}\left[\int A_N(x)\,dx\right] = \lim_{N\to\infty} \int \mathbb{E}[A_N(x)]\,dx = 0, \tag{57}$$

where the last equality is due to the generalized Lebesgue convergence theorem [34], Proposition 18, page 270, with $f_N(x) = \mathbb{E}[A_N(x)]$ and $g_N(x) = 2C\,\mathrm{osc}(h)\mathbb{E}[B_N(x)]$ provided that the following conditions hold:

(i) for any $x \in \mathbb{X}$, $\mathbb{E}[A_N(x)] \le 2C^2\,\mathrm{osc}^2(h)\mathbb{E}[B_N(x)]$,
(ii) for any $x \in \mathbb{X}$, $\lim_{N\to\infty} \mathbb{E}[A_N(x)] = 0$, $\mathbb{P}$-a.s.,
(iii) $\lim_{N\to\infty} \int \mathbb{E}[B_N(x)]\,dx = \int \lim_{N\to\infty} \mathbb{E}[B_N(x)]\,dx$.

Proof of (i). The bound follows directly from Lemmas 7 and 3.

Proof of (ii). Using again Lemmas 7 and 3, for any $x \in \mathbb{X}$,

$$A_N(x) \le 2C^2|\vartheta_t|_\infty |p_t|_\infty\,\mathrm{osc}^2(h),$$

$$\limsup_{N\to\infty} A_N(x) \le |\vartheta_t p_t \omega_t^2|_\infty \limsup_{N\to\infty} |(G_{t,T}^N(x,h))^2 - G_{t,T}^2(x,h)| = 0, \qquad \mathbb{P}\text{-a.s.}$$

These two inequalities combined with $A_N(x) \ge 0$ allow for applying the Lebesgue dominated convergence theorem, verifying condition (ii).

Proof of (iii). We have

$$\begin{aligned}
\lim_{N\to\infty} \int \mathbb{E}[B_N(x)]\,dx &\overset{(a)}{=} \lim_{N\to\infty} \mathbb{E}\left[\phi_{t-1}^N\left(\vartheta_t(\cdot)\int p_t(\cdot,x)\,dx\right)\right] \\
&\overset{(b)}{=} \phi_{t-1}(\vartheta_t) \overset{(c)}{=} \int \phi_{t-1}(\vartheta_t(\cdot)p_t(\cdot,x))\,dx \\
&\overset{(d)}{=} \int \lim_{N\to\infty} \mathbb{E}[\phi_{t-1}^N(\vartheta_t(\cdot)p_t(\cdot,x))]\,dx \\
&= \int \lim_{N\to\infty} \mathbb{E}[B_N(x)]\,dx,
\end{aligned}$$

where (a) and (c) are consequences of Fubini's theorem and (b) and (d) follows from the L^1-convergence of $\phi_t^N(h)$ to $\phi_t(h)$ (see Theorem 5) with $h(\cdot) = \vartheta_t(\cdot)$ and $h(\cdot) = \vartheta_t(\cdot)p_t(\cdot,x)$.

Thus, (57) holds, yielding that $\int A_N(x)\,dx \xrightarrow{\mathrm{P}} 0$ as N tends to infinity. This in turn implies (56) via the inequality

$$|\phi_{t-1}^N[\upsilon_{t,T}^N(\cdot,h) - \upsilon_{t,T}(\cdot,h)]| \le \int A_N(x)\,dx.$$

This establishes (51) and therefore completes the proof. □

The weak convergence of $\sqrt{N}(\phi_{0:T|T}^N(h) - \phi_{0:T|T}(h))$ for the FFBS algorithm implies more or less immediately the one of $\sqrt{N}(\tilde{\phi}_{0:T|T}^N(h) - \phi_{0:T|T}(h))$ for the FFBSi algorithm.

COROLLARY 9. *Under the assumptions of Theorem 8,*

$$\begin{aligned}&\sqrt{N}(\tilde{\phi}^N_{0:T|T}(h)-\phi_{0:T|T}(h))\\&\quad\xrightarrow{\mathcal{D}}\mathcal{N}(0,\phi^2_{0:T|T}[h-\phi_{0:T|T}(h)]+\Gamma_{0:T|T}[h-\phi_{0:T|T}(h)]).\end{aligned}\tag{58}$$

PROOF. Using (13) and the definition of $\tilde{\phi}^N_{0:T|T}(h)$, we may write

$$\begin{aligned}&\sqrt{N}(\tilde{\phi}^N_{0:T|T}(h)-\phi_{0:T|T}(h))\\&\quad=N^{-1/2}\sum_{\ell=1}^N[h(\xi_0^{J_0^\ell},\ldots,\xi_T^{J_T^\ell})-\mathbb{E}[h(\xi_0^{J_0},\ldots,\xi_T^{J_T})|\mathcal{F}_T^N]]\\&\qquad+\sqrt{N}(\phi^N_{0:T|T}(h)-\phi_{0:T|T}(h)).\end{aligned}$$

Note that since $\{J^\ell_{0:T}\}_{\ell=1}^N$ are i.i.d. conditional on $\mathcal{F}_T^N$, (58) follows from (47) and direct application of [15], Theorem A.3, page 2360, by noting that

$$\begin{aligned}&N^{-1}\sum_{\ell=1}^N\mathbb{E}[\{h(\xi_0^{J_0^\ell},\ldots,\xi_T^{J_T^\ell})-\mathbb{E}[h(\xi_0^{J_0},\ldots,\xi_T^{J_T})|\mathcal{F}_T^N]\}^2|\mathcal{F}_T^N]\\&\quad=(\phi^N_{0:T|T}[h-\phi^N_{0:T|T}(h)])^2\xrightarrow{\mathrm{P}}(\phi_{0:T|T}[h-\phi_{0:T|T}(h)])^2.\end{aligned}$$

□

4. Time uniform bounds. Most often, it is not required to compute the joint smoothing distribution but rather the marginal smoothing distributions $\phi_{s|T}$. Considering (8) for a function h that depends on the component x_s only, we obtain particle approximations of the marginal smoothing distributions by associating the set $\{\xi_s^j\}_{j=1}^N$ of particles with weights obtained by marginalizing the joint smoothing weights according to

$$\omega^{i_s}_{s|T}=\sum_{i_{s+1}=1}^N\cdots\sum_{i_T=1}^N\prod_{u=s+1}^t\frac{\omega_{u-1}^{i_{u-1}}m(\xi_{u-1}^{i_{u-1}},\xi_u^{i_u})}{\sum_{\ell=1}^N\omega_{u-1}^\ell m(\xi_{u-1}^\ell,\xi_u^{i_u})}\frac{\omega_T^{i_T}}{\Omega_T}.$$

It is easily seen that these marginal weights may be recursively updated backward in time as

$$\omega^i_{s|T}=\sum_{j=1}^N\frac{\omega_s^i m(\xi_s^i,\xi_{s+1}^j)}{\sum_{\ell=1}^N\omega_s^\ell m(\xi_s^\ell,\xi_{s+1}^j)}\omega^j_{s+1|T}.\tag{59}$$

In this section, we study the long-term behavior of the marginal fixed-interval smoothing distribution estimator. For that purpose, it is required to impose a type of mixing condition on the Markov transition kernel; see [5] and the references therein. For simplicity, we consider elementary but strong

conditions which are similar to the ones used in [9], Chapter 7.4, or [3], Chapter 4; these conditions, which points to applications where the state space $\mathbb{X}$ is compact, can be relaxed, but at the expense of many technical difficulties [4, 37, 38, 40].

ASSUMPTION 4. There exist two constants $0 < \sigma_- \le \sigma_+ < \infty$, such that, for any $(x, x') \in \mathbb{X} \times \mathbb{X}$,

$$\sigma_- \le m(x, x') \le \sigma_+. \tag{60}$$

In addition, there exists a constant $c_- > 0$ such that, $\int \chi(dx_0) g_0(x_0) \ge c_-$ and for all $t \ge 1$,

$$\inf_{x \in \mathbb{X}} \int M(x, dx') g_t(x') \ge c_- > 0. \tag{61}$$

Assumption 4 implies that $\nu(\mathbb{X}) < \infty$; in the sequel, we will consider without loss of generality that $\nu(\mathbb{X}) = 1$. Note also that, under Assumption 4, the average number of simulations required in the accept–reject mechanism per sample of the FFBSi algorithm is bounded by σ_+/σ_-.

The goal of this section consists in establishing, under the assumptions mentioned above, that the FFBS approximation of the *marginal* fixed interval smoothing probability satisfies an exponential deviation inequality with constants that are uniform in time and, under the same assumptions, that the variance of the CLT is uniformly bounded in time.

For obtaining these results, we will need upper-bounds on $G^N_{t,T}$ and $G_{t,T}$ that are more precise than the ones stated in Lemmas 3 and 7. For any function $h \in \mathcal{F}_{\mathrm{b}}(\mathbb{X})$ and $s \le T$, define the extension $\Pi_{s,T} h \in \mathcal{F}_{\mathrm{b}}(\mathbb{X}^{T+1})$ of h to $\mathbb{X}^{T+1}$ by

$$\Pi_{s,T} h(x_{0:T}) \stackrel{\mathrm{def}}{=} h(x_s), \qquad x_{0:T} \in \mathbb{X}^{T+1}. \tag{62}$$

LEMMA 10. *Assume that Assumptions 1–4 hold with $T = \infty$. Let $s \le T$. Then, for all t,T, $N \ge 1$, and $h \in \mathcal{F}_{\mathrm{b}}(\mathbb{X})$,*

$$|G^N_{t,T}(\cdot, \Pi_{s,T} h)|_\infty \le \rho^{|t-s|} \operatorname{osc}(h) |\mathcal{L}_{t,T}(\cdot, \mathbf{1})|_\infty, \tag{63}$$

where $\mathcal{L}_{t,T}$ is defined in (26) and

$$\rho = 1 - \frac{\sigma_-}{\sigma_+}. \tag{64}$$

Moreover, for all t, $T \ge 1$, and $h \in \mathcal{F}_{\mathrm{b}}(\mathbb{X})$,

$$|G_{t,T}(\cdot, \Pi_{s,T} h)|_\infty \le \rho^{|t-s|} \operatorname{osc}(h) |\mathcal{L}_{t,T}(\cdot, \mathbf{1})|_\infty. \tag{65}$$

PROOF. Using (27) and (29),

$$\frac{G^N_{t,T}(x,\Pi_{s,T}h)}{\mathcal{L}_{t,T}(x,\mathbf{1})} = \frac{\mathcal{L}^N_{t,T}(x,\Pi_{s,T}h)}{\mathcal{L}^N_{t,T}(x,\mathbf{1})} - \frac{\phi^N_{t-1}[\mathcal{L}^N_{t-1,T}(\cdot,\Pi_{s,T}h)]}{\phi^N_{t-1}[\mathcal{L}^N_{t-1,T}(\cdot,\mathbf{1})]}. \tag{66}$$

To prove (63), we will rewrite (66) and obtain an exponential bound by either using ergodicity properties of the "a posteriori" chain (when $t \le s$), or by using ergodicity properties of the backward kernel (when $t > s$).

Assume first that $t \le s$. The quantity $L_{t,T}(x_{0:t},\Pi_{s,T}h)$ does not depend on $x_{0:t-1}$ so that by (24) and definition (20) of $L_{t,T}$,

$$\begin{aligned}\mathcal{L}^N_{t,T}(x_t,\Pi_{s,T}h) &= L_{t,T}(x_{0:t},\Pi_{s,T}h)\\ &= \int\cdots\int\left(\prod_{u=t+1}^{T} M(x_{u-1},dx_u)g_u(x_u)\right)h(x_s)\\ &= \mathcal{L}_{t,T}(x_t,\Pi_{s,T}h).\end{aligned} \tag{67}$$

Now, by construction, for any $t \le s$,

$$\mathcal{L}_{t-1,T}(x_{t-1},\Pi_{s,T}h) = \int M(x_{t-1},dx_t)g_t(x_t)\mathcal{L}_{t,T}(x_t,\Pi_{s,T}h). \tag{68}$$

The relations (66), (67) and (68) imply that

$$\frac{G^N_{t,T}(x,\Pi_{s,T}h)}{\mathcal{L}_{t,T}(x,\mathbf{1})} = \frac{\mu[\mathcal{L}_{t,T}(\cdot,\Pi_{s,T}h)]}{\mu[\mathcal{L}_{t,T}(\cdot,\mathbf{1})]} - \frac{\mu'[\mathcal{L}_{t,T}(\cdot,\Pi_{s,T}h)]}{\mu'[\mathcal{L}_{t,T}(\cdot,\mathbf{1})]}, \tag{69}$$

where $\mu \stackrel{\text{def}}{=} \delta_x$ and μ' is the nonnegative finite measure defined by

$$\mu'(A) \stackrel{\text{def}}{=} \iint \phi^N_{t-1}(dx_{t-1})M(x_{t-1},dx_t)g_t(x_t)\mathbf{1}_A(x_t), \qquad A \in \mathcal{B}(\mathbb{X}).$$

Now, for any finite measure μ on $(\mathbb{X},\mathcal{B}(\mathbb{X}))$, the quantity

$$\begin{aligned}&\frac{\mu[\mathcal{L}_{t,T}(\cdot,\Pi_{s,T}h)]}{\mu[\mathcal{L}_{t,T}(\cdot,\mathbf{1})]}\\ &\quad= \frac{\int\cdots\int \mu(dx_t)\prod_{u=t+1}^{T} M(x_{u-1},dx_u)g_u(x_u)h(x_s)}{\int\cdots\int \mu(dx_t)\prod_{u=t+1}^{T} M(x_{u-1},dx_u)g_u(x_u)}\\ &\quad= \frac{\int\cdots\int \mu(dx_t)\prod_{u=t+1}^{s} M(x_{u-1},dx_u)g_u(x_u)h(x_s)\mathcal{L}_{s,T}(x_s,\mathbf{1})}{\int\cdots\int \mu(dx_t)\prod_{u=t+1}^{s} M(x_{u-1},dx_u)g_u(x_u)\mathcal{L}_{s,T}(x_s,\mathbf{1})}\end{aligned}$$

may be seen as the expectation of $h(X_s)$ conditionally on $Y_{t:T}$, where X_t is distributed according to $A \mapsto \mu(A)/\mu(\mathbb{X})$. Under the strong mixing condition (Assumption 4), it is shown in [12] (see also [9]) that, for any $t \le s \le T$, any

finite measure μ and μ' on $(\mathbb{X}, \mathcal{B}(\mathbb{X}))$, any function $h \in \mathcal{F}_{\mathrm{b}}(\mathbb{X})$, that

$$\begin{aligned}\Bigg| &\frac{\int \cdots \int \mu(dx_t) \prod_{u=t+1}^{s} M(x_{u-1}, dx_u) g_u(x_u) h(x_s) \mathcal{L}_{s,T}(x_s, \mathbf{1})}{\int \cdots \int \mu(dx_t) \prod_{u=t+1}^{s} M(x_{u-1}, dx_u) g_u(x_u) \mathcal{L}_{s,T}(x_s, \mathbf{1})} \\ &- \frac{\int \cdots \int \mu'(dx_t) \prod_{u=t+1}^{s} M(x_{u-1}, dx_u) g_u(x_u) h(x_s) \mathcal{L}_{s,T}(x_s, \mathbf{1})}{\int \cdots \int \mu'(dx_t) \prod_{u=t+1}^{s} M(x_{u-1}, dx_u) g_u(x_u) \mathcal{L}_{s,T}(x_s, \mathbf{1})} \Bigg| \\ &\leq \rho^{s-t} \operatorname{osc}(h),\end{aligned}$$

where ρ is defined in (64). This shows (63) when t is smaller than s.

Consider now the case $s < t \leq T$. By definition,

$$\begin{aligned}\mathcal{L}_{t,T}^N(x_t, \Pi_{s,T} h) &= \int \cdots \int L_{t,T}(x_{0:t}, \Pi_{s,T} h) \prod_{u=s+1}^{t} \mathrm{B}_{\phi_{u-1}^N}(x_u, dx_{u-1}) \\ &= \int \cdots \int \mathcal{L}_{t,T}(x_t, \mathbf{1}) \prod_{u=s+1}^{t} \mathrm{B}_{\phi_{u-1}^N}(x_u, dx_{u-1}) h(x_s),\end{aligned} \tag{70}$$

where the last expression is obtained from the following equality, valid for $s < t$:

$$\begin{aligned}L_{t,T}(x_{0:t}, \Pi_{s,T} h) &= h(x_s) \int \cdots \int \prod_{u=t+1}^{T} M(x_{u-1}, dx_u) g_u(x_u) \\ &= h(x_s) \mathcal{L}_{t,T}(x_t, \mathbf{1}).\end{aligned}$$

Moreover, combining (33) and (70),

$$\begin{aligned}&\phi_{t-1}^N [\mathcal{L}_{t-1,T}^N(\cdot, \Pi_{s,T} h)] \\ &\quad = \int \cdots \int \phi_{t-1}^N(du_{t-1}) M(u_{t-1}, dx_t) g_t(x_t) \mathcal{L}_{t,T}^N(x_t, \Pi_{s,T} h) \\ &\quad = \int \cdots \int \phi_{t-1}^N(du_{t-1}) M(u_{t-1}, dx_t) g_t(x_t) \mathcal{L}_{t,T}(x_t, \mathbf{1}) \\ &\qquad\qquad \times \prod_{u=s+1}^{t} \mathrm{B}_{\phi_{u-1}^N}(x_u, dx_{u-1}) h(x_s).\end{aligned}$$

By plugging this expression and (70) into (66), we obtain

$$\frac{G_{t,T}^N(x, \Pi_{s,T} h)}{\mathcal{L}_{t,T}(x, \mathbf{1})} = \int \cdots \int \left\{ \frac{\mu(dx_t)}{\mu(\mathbb{X})} - \frac{\mu'(dx_t)}{\mu'(\mathbb{X})} \right\} \prod_{u=s+1}^{t} \mathrm{B}_{\phi_{u-1}^N}(x_u, dx_{u-1}) h(x_s),$$

with $\mu(dx_t) = \delta_x(dx_t)\mathcal{L}_{t,T}(x_t, \mathbf{1})$ and μ' being the nonnegative measure defined by

$$\mu'(A) = \int \phi_{t-1}^N[m(\cdot, x_t)]g_t(x_t)\mathcal{L}_{t,T}(x_t, \mathbf{1})\mathbf{1}_A(x_t)\,dx_t.$$

Under the uniform ergodicity condition (Assumption 4) it holds, for any probability measure η on $(\mathbb{X}, \mathcal{B}(\mathbb{X}))$, and any $A \in \mathcal{B}(\mathbb{X})$,

$$\mathrm{B}_\eta(x, A) = \frac{\int_A \eta(dx')m(x', x)}{\int \eta(dx')m(x', x)} \geq \frac{\sigma_-}{\sigma_+}\eta(A);$$

thus, the transition kernel B_η is uniformly Doeblin with minorizing constant σ_-/σ_+ and the proof of (63) for $s < t \leq T$ follows. The last statement of the Lemma follows from (63) and the almost-sure convergence

$$\lim_{N\to\infty} G_{t,T}^N(x, h) = G_{t,T}(x, h), \qquad \mathbb{P}\text{-a.s.},$$

for all $x \in \mathbb{X}$, which was established in Lemma 7. $\square$

4.1. *A time uniform exponential deviation inequality.* Under the strong mixing Assumption 4, a time uniform deviation inequality for the *marginal smoothing* approximation can be derived using the exponentially decreasing bound on the quantity $G_{t,T}^N$ obtained in Lemma 10.

THEOREM 11. *Assume Assumptions 1–4 hold with $T = \infty$. Then, there exist constants $0 \leq B$, $C < \infty$ such that for all integers N, s, T, with $s \leq T$, and for all $\varepsilon > 0$,*

$$\mathbb{P}[|\phi_{s|T}^N(h) - \phi_{s|T}(h)| \geq \varepsilon] \leq Be^{-CN\varepsilon^2/\operatorname{osc}^2(h)}, \tag{71}$$

$$\mathbb{P}[|\tilde{\phi}_{s|T}^N(h) - \phi_{s|T}(h)| \geq \varepsilon] \leq Be^{-CN\varepsilon^2/\operatorname{osc}^2(h)}, \tag{72}$$

where $\phi_{s|T}^N(h)$ and $\tilde{\phi}_{s|T}^N(h)$ are defined in (6) and (14).

Letting $s = T$ in Theorem 11 provides, as a special case, the (already known) time uniform deviation inequality for the *filter* approximation; however, the novelty of the bounds obtained here is that these confirm the stability of the FFBSm and FFBSi marginal smoothing approximations also when s is fixed and T tends to infinity (see [9] for further discussion).

PROOF OF THEOREM 11. Combining (27) with the definition (20) and Assumption 4 yields, for all $x \in \mathbb{X}$,

$$\frac{\sigma_-}{\sigma_+} \leq \frac{\mathcal{L}_{t,T}(x, \mathbf{1})}{|\mathcal{L}_{t,T}(\cdot, \mathbf{1})|_\infty} \leq 1. \tag{73}$$

Let $h \in \mathcal{F}_b(\mathbb{X}^{T+1})$ and assume without loss of generality that $\phi_{0:T|T}(h) = 0$. Then, (21) implies that $\phi_0[L_{0,T}(\cdot, h)] = 0$ and the first term of the decomposition (22) thus becomes

$$\frac{\phi_0^N[L_{0,T}(\cdot,h)]}{\phi_0^N[L_{0,T}(\cdot,\mathbf{1})]} = \frac{N^{-1}\sum_{i=0}^N \frac{d\chi}{d\rho_0}(\xi_0^i) g_0(\xi_0^i) L_{0,T}(\xi_0^i, h)}{N^{-1}\sum_{\ell=0}^N \frac{d\chi}{d\rho_0}(\xi_0^\ell) g_0(\xi_0^\ell) L_{0,T}(\xi_0^\ell, \mathbf{1})}, \tag{74}$$

where $(\xi_0^\ell)_{\ell=1}^N$ are i.i.d. random variables with distribution ρ_0. Noting that $L_{0,T} = \mathcal{L}_{0,T}$ we obtain an exponential deviation inequality for (74) by applying Lemma 4 with

$$\begin{cases} a_N = N^{-1}\displaystyle\sum_{i=0}^N \frac{d\chi}{d\rho_0}(\xi_0^i) g_0(\xi_0^i) \mathcal{L}_{0,T}(\xi_0^i, h)/|\mathcal{L}_{0,T}(\cdot,h)|_\infty, \\ b_N = N^{-1}\displaystyle\sum_{i=0}^N \frac{d\chi}{d\rho_0}(\xi_0^i) g_0(\xi_0^i) \mathcal{L}_{0,T}(\xi_0^i, \mathbf{1})/|\mathcal{L}_{0,T}(\cdot,h)|_\infty, \\ b = \chi[g_0(\cdot)\mathcal{L}_{0,T}(\cdot,\mathbf{1})]/|\mathcal{L}_{0,T}(\cdot,h)|_\infty, \\ \beta = \chi(g_0)\sigma_-/\sigma_+. \end{cases}$$

Here, condition (I) is trivially satisfied and conditions (II) and (III) follow from the Hoeffding inequality for i.i.d. variables.

According to (22) and (28), it is now required, for any $1 \le t \le T$, to derive an exponential inequality for

$$A_{t,T}^N \stackrel{\text{def}}{=} \frac{N^{-1}\sum_{\ell=1}^N \omega_t^\ell G_{t,T}^N(\xi_t^\ell, \Pi_{s,T} h)}{N^{-1}\sum_{\ell=1}^N \omega_t^\ell \mathcal{L}_{t,T}(\xi_t^\ell, \mathbf{1})}.$$

Note first that, using (73), we have

$$|A_{t,T}^N| \le \left(\frac{\sigma_+}{\sigma_-}\right) \frac{N^{-1}\sum_{\ell=1}^N \omega_t^\ell G_{t,T}^N(\xi_t^\ell, \Pi_{s,T} h)/|\mathcal{L}_{t,T}(\cdot,\mathbf{1})|_\infty}{N^{-1}\sum_{\ell=1}^N \omega_t^\ell}.$$

We use again Lemma 4 with

$$\begin{cases} a_N = N^{-1}\displaystyle\sum_{\ell=1}^N \omega_t^\ell G_{t,T}^N(\xi_t^\ell, \Pi_{s,T} h)/|\mathcal{L}_{t,T}(\cdot,\mathbf{1})|_\infty, \\ b_N = N^{-1}\displaystyle\sum_{\ell=1}^N \omega_t^\ell, \\ b = \mathbb{E}[\omega_t^1 | \mathcal{F}_{t-1}^N] = \phi_{t-1}^N[M(\cdot, g_t)]/\phi_{t-1}^N(\vartheta_t), \\ \beta = c_-/|\vartheta_t|_\infty. \end{cases}$$

Assumption 4 shows that $b \ge \beta$ and Lemma 10 shows that $|a_N/b_N| \le M \stackrel{\text{def}}{=} \rho^{|t-s|}\operatorname{osc}(h)$, where ρ is defined in (64). Therefore, condition (I) of Lemma 4

is satisfied and the Hoeffding inequality gives

$$\mathbb{P}[|b_N - b| \geq \varepsilon] \leq \mathbb{E}\left[\mathbb{P}\left[\left|N^{-1}\sum_{\ell=1}^{N}(\omega_t^\ell - \mathbb{E}[\omega_t^1|\mathcal{F}_{t-1}^N])\right| \geq \varepsilon \middle| \mathcal{F}_{t-1}^N\right]\right]$$
$$\leq 2\exp(-2N\varepsilon^2/|\omega_t|_\infty^2),$$

establishing condition (II) in Lemma 4. Finally, Lemma 10 and the Hoeffding inequality imply that

$$\mathbb{P}[|a_N| \geq \varepsilon] \leq \mathbb{E}\left[\mathbb{P}\left[\left|N^{-1}\sum_{\ell=1}^{N}\omega_t^\ell G_{t,T}^N(\xi_t^\ell, \Pi_{s,T}h)/|\mathcal{L}_{t,T}(\cdot,\mathbf{1})|_\infty\right| \geq \varepsilon \middle| \mathcal{F}_{t-1}^N\right]\right]$$
$$\leq 2\exp\left(-2\frac{N\varepsilon^2}{|\omega_t|_\infty^2\rho^{2|t-s|}\operatorname{osc}^2(h)}\right) = 2\exp\left(-2\frac{N\varepsilon^2}{|\omega_t|_\infty^2 M^2}\right).$$

Lemma 4 therefore yields

$$\mathbb{P}\left(\left|\frac{a_N}{b_N}\right| \geq \varepsilon\right) \leq 2\exp\left(-\frac{N\varepsilon^2 c_-^2}{2\operatorname{osc}^2(h)\rho^{2|t-s|}|\omega_t|_\infty^2|\vartheta_t|_\infty^2}\right),$$

so that

$$\mathbb{P}(|A_{t,T}^N| \geq \varepsilon) \leq 2\exp\left(-\frac{N\varepsilon^2 c_-^2\sigma_-^2}{2\operatorname{osc}^2(h)\rho^{2|t-s|}|\omega_t|_\infty^2|\vartheta_t|_\infty^2\sigma_+^2}\right).$$

A time uniform exponential deviation inequality for $\sum_{t=1}^{T} A_{t,T}$ then follows from Lemma 13 and the proof is complete. □

4.2. *A time uniform bound on the variance of the marginal smoothing distribution.* Analogous to the result obtained in the previous section, a time uniform bound on the asymptotic variance in the CLT for the *marginal smoothing* approximations can, again under the strong mixing Assumption 4, be easily obtained from the exponentially decreasing bound on $G_{t,T}$ stated and proved in Lemma 10 for the quantity.

THEOREM 12. *Assume Assumptions 1–4 hold with $T = \infty$. Then, for all $s \leq T$,*

$$\Gamma_{0:T|T}[\Pi_{s,T}h] \leq \left(\frac{\sigma_+}{\sigma_-}\left(1 \vee \sup_{t\geq 1}|\vartheta_t|_\infty\right)\sup_{t\geq 0}|\omega_t|_\infty \operatorname{osc}(h)\right)^2 \frac{1+\rho^2}{1-\rho^2},$$

where $\Gamma_{0:T|T}$ is defined in (48).

In accordance with the results of the previous section, letting $s = T$ in the previous theorem provides a time uniform bound on the asymptotic

variance for the *filter* approximation; nevertheless, as mentioned previously, the situation of interest for us is when s is fixed and T goes to infinity.

PROOF OF THEOREM 12. Combining (73) and (65) with $\rho_0(\omega_0) = 1$ yields

$$\frac{\rho_0(\omega_0^2(\cdot)G_{0,T}^2(\cdot, \Pi_{s,T}h))}{\rho_0^2[\omega_0(\cdot)\mathcal{L}_{0,T}(\cdot, \mathbf{1})]} \leq \left(\frac{\sigma_+}{\sigma_-}|\omega_0|_\infty \operatorname{osc}(h)\rho^s\right)^2.$$

Moreover, by inserting, for any $0 < t \leq T$, the bound obtained in (65) into the expression (49) of $v_{t,T}$ we obtain

$$\frac{\phi_{t-1}(v_{t,T}(\cdot, \Pi_{s,T}h)))\phi_{t-1}(\vartheta_t)}{\phi_{t-1}^2[\mathcal{L}_{t-1,T}(\cdot, \mathbf{1})]} \leq \left(\frac{\sigma_+}{\sigma_-}|\vartheta_t|_\infty|\omega_t|_\infty \operatorname{osc}(h)\rho^{|t-s|}\right)^2.$$

Finally, plugging the two bounds above into (48) gives

$$\Gamma_{0:T|T}[\Pi_{s,T}h] \leq \left(\frac{\sigma_+}{\sigma_-}\left(1 \vee \sup_{t\geq 1}|\vartheta_t|_\infty\right)\sup_{t\geq 0}|\omega_t|_\infty \operatorname{osc}(h)\right)^2\left(\sum_{t=0}^{\infty}\rho^{2|t-s|}\right),$$

which completes the proof. □

5. Proofs of Propositions 1 and 2. Having at hand the theory established in the previous sections, we are now ready to present the proofs of Propositions 1 and 2.

PROOF OF PROPOSITION 1. The average number of simulations required to sample J_s^ℓ conditionally on $\mathcal{G}_{s+1}^N$ is $\sigma_+\Omega_s/\sum_{u=1}^N \omega_s^u m(\xi_s^u, \xi_{s+1}^{J_{s+1}^\ell})$. Hence, the number of simulations Z_s^N required to sample $\{J_s^\ell\}_{\ell=1}^N$ has conditional expectation

$$\mathbb{E}[Z_s^N|\mathcal{G}_{s+1}^N] = \sum_{\ell=1}^N \frac{\sigma_+\Omega_s}{\sum_{i=1}^N \omega_s^i m(\xi_s^i, \xi_{s+1}^{J_{s+1}^\ell})}.$$

We denote $\omega_{s|T}^i \overset{\text{def}}{=} \mathbb{P}[J_s^1 = i|\mathcal{F}_T^N]$ and $\omega_{s:s+1|T}^{\ell i} \overset{\text{def}}{=} \mathbb{P}[J_s^1 = \ell, J_{s+1}^1 = i|\mathcal{F}_T^N]$ and write

$$\begin{aligned}
\mathbb{E}[Z_s^N|\mathcal{F}_T^N] &= \sum_{i=1}^N \omega_{s+1|T}^i \frac{\sigma_+\Omega_s}{\sum_{j=1}^N \omega_s^j m(\xi_s^j, \xi_{s+1}^i)} \\
&= \sigma_+\Omega_s \sum_{i=1}^N \sum_{\ell=1}^N \frac{\omega_{s+1|T}^i \omega_s^\ell m(\xi_s^\ell, \xi_{s+1}^i)}{\sum_{j=1}^N \omega_s^j m(\xi_s^j, \xi_{s+1}^i)} \times \frac{1}{\omega_s^\ell m(\xi_s^\ell, \xi_{s+1}^i)} \\
&= \sigma_+\Omega_s \sum_{i=1}^N \sum_{\ell=1}^N \omega_{s:s+1|T}^{\ell i} \frac{1}{\omega_s^\ell m(\xi_s^\ell, \xi_{s+1}^i)}.
\end{aligned}$$

For the bootstrap particle filter, $\omega_s^\ell \equiv g_s(\xi_s^\ell)$; Theorem 5 then implies that $\Omega_s/N \xrightarrow{\mathrm{P}}_{N\to\infty} \phi_{s|s-1}(g_s)$ and

$$\sum_{i=1}^N \sum_{\ell=1}^N \omega_{s:s+1|T}^{\ell i} \frac{1}{\omega_s^\ell m(\xi_s^\ell, \xi_{s+1}^i)} \xrightarrow{\mathrm{P}}_{N\to\infty} \iint \phi_{s:s+1|T}(dx_{s:s+1}) \frac{1}{g_s(x_s)m(x_s,x_{s+1})}.$$

Besides,

$$\begin{aligned}
&\iint \phi_{s:s+1|T}(dx_{s:s+1}) \frac{1}{g_s(x_s)m(x_s,x_{s+1})} \\
&\quad = \int \cdots \int \phi_{s|s-1}(dx_s) \frac{g_s(x_s)M(x_s,dx_{s+1})}{g_s(x_s)m(x_s,x_{s+1})} g_{s+1}(x_{s+1}) \\
&\qquad\qquad \times \prod_{u=s+2}^T M(x_{u-1},dx_u) g_u(x_u) \\
&\qquad \Big/ \int \cdots \int \phi_{s|s-1}(dx_s) g_s(x_s) \prod_{u=s+1}^T M(x_{u-1},dx_u) g_u(x_u) \\
&\quad = \frac{\int \cdots \int dx_{s+1} \prod_{u=s+2}^T \int M(x_{u-1},dx_u) g_u(x_u)}{\int \cdots \int \phi_{s|s-1}(dx_s) g_s(x_s) \prod_{u=s+1}^T M(x_{u-1},dx_u) g_u(x_u)}.
\end{aligned}$$

Similarly, in the fully adapted case we have $\omega_s^i \equiv 1$ for all $i \in \{1,\dots,N\}$; thus, $\Omega_s = N$ and

$$\begin{aligned}
&\sum_{i=1}^N \sum_{\ell=1}^N \omega_{s:s+1|T}^{\ell i} \frac{1}{\omega_s^\ell m(\xi_s^\ell, \xi_{s+1}^i)} \\
&\quad \xrightarrow{\mathrm{P}}_{N\to\infty} \iint \phi_{s:s+1|T}(dx_{s:s+1}) \frac{1}{m(x_s,x_{s+1})} \\
&\quad = \frac{\int \cdots \int g_{s+1}(x_{s+1})\, dx_{s+1} \prod_{u=s+2}^T \int M(x_{u-1},dx_u) g_u(x_u)}{\int \cdots \int \phi_s(dx_s) \prod_{u=s+1}^T M(x_{u-1},dx_u) g_u(x_u)}.
\end{aligned}$$

In both cases, the numerator can be bounded from above by

$$\frac{\sigma_+^{T-s-1} \prod_{u=s+1}^T \int g_u(x_u)\, dx_u}{\int \cdots \int \phi_{s|s-1}(dx_s) g_s(x_s) \prod_{u=s+1}^T M(x_{u-1},dx_u) g_u(x_u)}$$

if $\int g_u(x_u)\, dx_u < \infty$ for all $u \geq 0$. $\square$

PROOF OF PROPOSITION 2. Fix a time step s of the algorithm and denote by C_s the number of elementary operations required for this step. For $k \in \{1, \dots, n\}$, let T_s^k be the number of times that k appears in list L at time s in the 'while' loop. Let also $N_s^u \stackrel{\text{def}}{=} \sum_{k=1}^N \mathbb{1}_{\{T_s^k \geq u\}}$ be the size of L (i.e., the value of n at line 6) after u iterations of the 'while' loop, with $N_s^0 \stackrel{\text{def}}{=} N$. Then, using Proposition 14 there exists a constant C such that

$$C_s \leq C \sum_{u=0}^{\infty} N_s^u \left(1 + \log\left(1 + \frac{N}{N_s^u}\right)\right).$$

As $n \to n(1 + \log(1 + N/n))$ is a concave, increasing function, it holds by Jensen's inequality that

$$\mathbb{E}[C_s] \leq C \sum_{u=0}^{\infty} \mathbb{E}[N_s^u] \left(1 + \log\left(1 + \frac{N}{\mathbb{E}[N_s^u]}\right)\right).$$

Besides,

$$\mathbb{E}[N_s^u] = \sum_{k=1}^{N} \mathbb{P}(T_s^k \geq u) \leq N \left(1 - \frac{\sigma_-}{\sigma_+}\right)^u$$

as σ_-/σ_+ is a lower bound on the acceptation probability. Thus,

$$\mathbb{E}[C_s] \leq CN \sum_{u=0}^{\infty} \left(1 - \frac{\sigma_-}{\sigma_+}\right)^u \left(1 + \log\left(1 + \frac{1}{(1 - \sigma_-/\sigma_+)^u}\right)\right) \leq \frac{KN\sigma_+}{\sigma_-}. \quad \square$$

APPENDIX A: PROOF OF LEMMA 4

Write

$$\left|\frac{a_N}{b_N}\right| \leq b^{-1} \left|\frac{a_N}{b_N}\right| |b - b_N| + b^{-1}|a_N| \leq \beta^{-1} M |b - b_N| + \beta^{-1}|a_N|, \qquad \mathbb{P}\text{-a.s.}$$

Thus,

$$\left\{\left|\frac{a_N}{b_N}\right| \geq \varepsilon\right\} \subseteq \left\{|b - b_N| \geq \frac{\varepsilon\beta}{2M}\right\} \cup \left\{|a_N| \geq \frac{\varepsilon\beta}{2}\right\},$$

from which the proof follows.

APPENDIX B: TECHNICAL RESULTS

LEMMA 13. *Let $\{Y_{n,i}\}_{i=1}^n$ be a triangular array of random variables such that there exist constants $B > 0$, $C > 0$, and ρ with $0 < \rho < 1$ satisfying, for all n, $i \in \{1, \dots, n\}$, and $\varepsilon > 0$,*

$$\mathbb{P}(|Y_{n,i}| \geq \varepsilon) \leq B \exp(-C\varepsilon^2 \rho^{-2i}).$$

Then, there exist constants $\bar{B} > 0$ and $\bar{C} > 0$ such that, for any n and $\varepsilon > 0$,

$$\mathbb{P}\left(\left|\sum_{i=1}^{n} Y_{n,i}\right| \geq \varepsilon\right) \leq \bar{B} e^{-\bar{C}\varepsilon^2}.$$

PROOF. Set $S \stackrel{\text{def}}{=} \sum_{\ell=1}^{\infty} \sqrt{\ell}\rho^{\ell}$; one easily concludes that

$$\mathbb{P}\left(\left|\sum_{i=1}^{n} Y_{n,i}\right| \geq \varepsilon\right) \leq \sum_{i=1}^{n} \mathbb{P}(|Y_{n,i}| \geq \varepsilon S^{-1}\sqrt{i}\rho^{i}) \leq B \sum_{i=1}^{n} \exp(-CS^{-1}\varepsilon^2 i).$$

Set $\varepsilon_0 > 0$. The proof follows by noting that, for any $\varepsilon \geq \varepsilon_0$,

$$\sum_{i=1}^{n} \exp(-CS^{-1}\varepsilon^2 i) \leq \exp(CS^{-1}\varepsilon_0^2)\exp(-CS^{-1}\varepsilon^2)/(1 - \exp(CS^{-1}\varepsilon_0^2)).$$

□

B.1. Description of the sampling procedure. In this section, we describe and analyze an efficient multinomial sampling procedure, detailed in Algorithm 2. Given a probability distribution $(p_1, \ldots, p_N)$ on the set $\{1, \ldots, N\}$, it returns a sample of size n of that distribution. Compared to the procedure described in Section 7.4.1 in [3], its main virtue is to be efficient for both large and small samples sizes: if $n = 1$, the complexity is $O(\log(N))$, while if $n = N$, the complexity is $O(N)$.

PROPOSITION 14. *The number of elementary operations required by Algorithm 2 is $O(n + n\log(1 + N/n))$.*

PROOF. The order statistics at line 5 and the permutation at line 6 can be sampled using $O(n)$ operations; see [13], Chapter V and XIII. For each value of k between 1 and n, denote by G_k the number of times lines 11–13 are executed. Observe that line 18 is executed the same number of times, and thus the number of elementary operations required by call to Algorithm 2 is $O(n + \sum_{k=1}^{n} G_k)$. But the value of l is increased during iteration k by at least $2^{G_k} - 1$, and as the final value of l is at most equal to N, it holds that

$$\sum_{k=1}^{n} 2^{G_k} \leq N + n.$$

By convexity,

$$\exp\left(\frac{\log(2)}{n}\sum_{k=1}^{n} G_k\right) \leq \frac{1}{n}\sum_{k=1}^{n} 2^{G_k} \leq 1 + \frac{N}{n},$$

which implies that

$$\sum_{k=1}^{n} G_k \leq n\log\left(1 + \frac{N}{n}\right)/\log(2).$$

□

Algorithm 2 Multinomial sampling

$q_1 \leftarrow p_1$
for k from 1 to N **do**
    $q_k \leftarrow q_{k-1} + p_k$
end for
sample an order statistics $U_{(1)}, \ldots, U_{(n)}$ of an i.i.d. uniform distribution
uniformly sample a permutation σ on $\{1, \ldots, n\}$
$l \leftarrow 0, r \leftarrow 1$
for k from 1 to n **do**
    $d \leftarrow 1$
    while $U_{(k)} \geq q_r$ **do**
        $l \leftarrow r$
        $r \leftarrow \min(r + 2^d, N)$
        $d \leftarrow d + 1$
    end while
    while $r - l > 1$ **do**
        $m \leftarrow \lfloor (l + r)/2 \rfloor$
        if $U_{(k)} \geq q_m$ **then**
            $l \leftarrow m$
        else
            $r \leftarrow m$
        end if
    end while
    $I_{\sigma(k)} \leftarrow r$
end for

REFERENCES

[1] Barembruch, S., Garivier, A. and Moulines, E. (2009). On approximate maximum-likelihood methods for blind identification: How to cope with the curse of dimensionality. *IEEE Trans. Signal Process.* **57** 4247–4259. MR2683174

[2] Briers, M., Doucet, A. and Maskell, S. (2010). Smoothing algorithms for state-space models. *Ann. Inst. Statist. Math.* **62** 61–89. MR2577439

[3] Cappé, O., Moulines, E. and Rydén, T. (2005). *Inference in Hidden Markov Models*. Springer, New York. MR2159833

[4] Chigansky, P. and Liptser, R. (2004). Stability of nonlinear filters in nonmixing case. *Ann. Appl. Probab.* **14** 2038–2056. MR2099662

[5] Chigansky, P., Liptser, R. and van Handel, R. (2011). Intrinsic methods in filter stability. In *The Oxford Handbook of Nonlinear Filtering*. Oxford University Press, Oxford.

[6] Chopin, N. (2004). Central limit theorem for sequential Monte Carlo methods and its application to Bayesian inference. *Ann. Statist.* **32** 2385–2411. MR2153989

[7] Cornebise, J., Moulines, É. and Olsson, J. (2008). Adaptive methods for sequential importance sampling with application to state space models. *Statist. Comput.* **18** 461–480. MR2461889

[8] DE JONG, P. and SHEPHARD, N. (1995). The simulation smoother for time series models. *Biometrika* **82** 339–350. MR1354233

[9] DEL MORAL, P. (2004). *Feynman–Kac Formulae: Genealogical and Interacting Particle Systems with Applications*. Springer, New York. MR2044973

[10] DEL MORAL, P. and DOUCET, A. (2009). Particle methods: An introduction with applications. Available at http://hal.inria.fr/docs/00/40/39/17/PDF/RR-6991.pdf.

[11] DEL MORAL, P., DOUCET, A. and SINGH, S. (2010). A backward particle interpretation of Feynman–Kac formulae. *ESAIM M2AN* **44** 947–975.

[12] DEL MORAL, P. and MICLO, L. (2000). Branching and interacting particle systems approximations of Feynman–Kac formulae with applications to non-linear filtering. In *Séminaire de Probabilités, XXXIV. Lecture Notes in Math.* **1729** 1–145. Springer, Berlin. MR1768060

[13] DEVROYE, L. (1986). *Nonuniform Random Variate Generation*. Springer, New York. MR0836973

[14] DOUC, R., FORT, G., MOULINES, E. and PRIOURET, P. (2009). Forgetting the initial distribution for hidden Markov models. *Stochastic Process. Appl.* **119** 1235–1256. MR2508572

[15] DOUC, R. and MOULINES, E. (2008). Limit theorems for weighted samples with applications to sequential Monte Carlo methods. *Ann. Statist.* **36** 2344–2376. MR2458190

[16] DOUC, R., MOULINES, É. and OLSSON, J. (2009). Optimality of the auxiliary particle filter. *Probab. Math. Statist.* **29** 1–28. MR2552996

[17] DOUCET, A., DE FREITAS, N. and GORDON, N. (eds.) (2001). *Sequential Monte Carlo Methods in Practice*. Springer, New York. MR1847783

[18] DOUCET, A., GODSILL, S. and ANDRIEU, C. (2000). On sequential Monte-Carlo sampling methods for Bayesian filtering. *Statist. Comput.* **10** 197–208.

[19] FEARNHEAD, P. (2005). Using random quasi-Monte-Carlo within particle filters, with application to financial time series. *J. Comput. Graph. Statist.* **14** 751–769. MR2211365

[20] FEARNHEAD, P., WYNCOLL, D. and TAWN, J. (2010). A sequential smoothing algorithm with linear computational cost. *Biometrika* **97** 447–464. MR2650750

[21] GODSILL, S. J., DOUCET, A. and WEST, M. (2004). Monte Carlo smoothing for non-linear time series. *J. Amer. Statist. Assoc.* **50** 438–449.

[22] GORDON, N., SALMOND, D. and SMITH, A. F. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proc. F, Radar Signal Process.* **140** 107–113.

[23] HANDSCHIN, J. E. and MAYNE, D. Q. (1969). Monte Carlo techniques to estimate the conditional expectation in multi-stage non-linear filtering. *Internat. J. Control (1)* **9** 547–559. MR0246490

[24] HÜRZELER, M. and KÜNSCH, H. R. (1998). Monte Carlo approximations for general state-space models. *J. Comput. Graph. Statist.* **7** 175–193. MR1649366

[25] JOHANSEN, A. M. and DOUCET, A. (2008). A note on auxiliary particle filters. *Statist. Probab. Lett.* **78** 1498–1504. MR2528342

[26] KAILATH, T., SAYED, A. and HASSIBI, B. (2000). *Linear Estimation*. Prentice Hall, New York.

[27] KITAGAWA, G. (1996). Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *J. Comput. Graph. Statist.* **5** 1–25. MR1380850

[28] Klaas, M., Briers, M., De Freitas, N., Doucet, A., Maskell, S. and Lang, D. (2006). Fast particle smoothing: If I had a million particles. In *23rd Int. Conf. Machine Learning (ICML)*. Pittsburgh, Pennsylvania.

[29] Kleptsyna, M. L. and Veretennikov, A. Y. (2008). On discrete time ergodic filters with wrong initial data. *Probab. Theory Related Fields* **141** 411–444. MR2391160

[30] Künsch, H. R. (2005). Recursive Monte Carlo filters: Algorithms and theoretical analysis. *Ann. Statist.* **33** 1983–2021. MR2211077

[31] Liu, J. S. (2001). *Monte Carlo Strategies in Scientific Computing*. Springer, New York. MR1842342

[32] Pitt, M. K. and Shephard, N. (1999). Filtering via simulation: Auxiliary particle filters. *J. Amer. Statist. Assoc.* **94** 590–599. MR1702328

[33] Ristic, B., Arulampalam, M. and Gordon, A. (2004). *Beyond Kalman Filters: Particle Filters for Target Tracking*. Artech House, Norwood, MA.

[34] Royden, H. L. (1988). *Real Analysis*, 3rd ed. Macmillan Co., New York. MR1013117

[35] Rubin, D. B. (1987). A noniterative sampling/importance resampling alternative to the data augmentation algorithm for creating a few imputations when the fraction of missing information is modest: The SIR algorithm (discussion of Tanner and Wong). *J. Amer. Statist. Assoc.* **82** 543–546.

[36] van Handel, R. (2007). Filtering, stability, and robustness. Ph.D. thesis, CalTech.

[37] Van Handel, R. (2008). Discrete time nonlinear filters with informative observations are stable. *Electron. Commun. Probab.* **13** 562–575. MR2461532

[38] van Handel, R. (2009). Uniform observability of hidden Markov models and filter stability for unstable signals. *Ann. Appl. Probab.* **19** 1172–1199.

[39] van Handel, R. (2009). The stability of conditional Markov processes and Markov chains in random environments. *Ann. Probab.* **37** 1876–1925. MR2561436

[40] van Handel, R. (2009). Uniform time average consistency of Monte Carlo particle filters. *Stochastic Process. Appl.* **119** 3835–3861. MR2552307

R. Douc
Télécom SudParis
9 rue Charles Fourier
91011 Evry-Cedex
France
E-mail: randal.douc@it-sudparis.eu

A. Garivier
E. Moulines
Télécom ParisTech
46 rue Barrault
75634 Paris, Cedex 13
France
E-mail: aurelien.garivier@telecom-paristech.fr
eric.moulines@telecom-paristech.fr

J. Olsson
Center for Mathematical Sciences
Lund University
Box 118
SE-22100 Lund
Sweden
E-mail: jimmy@maths.lth.se

Efficient particle-based online smoothing in general hidden Markov models: the PaRIS algorithm

Jimmy Olsson* **and Johan Westerborn**

Department of Mathematics
KTH Royal Institute of Technology
SE-100 44 Stockholm, Sweden
e-mail: jimmyol@kth.se, johawes@kth.se

Abstract: This paper presents a novel algorithm, the particle-based, rapid incremental smoother (PaRIS), for efficient online approximation of smoothed expectations of additive state functionals in general hidden Markov models. The algorithm, which has a linear computational complexity under weak assumptions and very limited memory requirements, is furnished with a number of convergence results, including a central limit theorem. An interesting feature of PaRIS, which samples on-the-fly from the retrospective dynamics induced by the particle filter, is that it requires two or more backward draws per particle in order to cope with degeneracy of the sampled trajectories and to stay numerically stable in the long run with an asymptotic variance that grows only linearly with time.

Primary 62M09; secondary 62F12.
Keywords and phrases: central limit theorem, general hidden Markov models, Hoeffding-type inequality, online estimation, particle filter, particle path degeneracy, sequential Monte Carlo, smoothing.

1. Introduction

This paper deals with the problem of state estimation in *general state-space hidden Markov models* (HMMs) using *sequential Monte Carlo* (SMC) *methods* (also known as *particle filters*), and presents a novel online algorithm for the computation of smoothed expectations of *additive state functionals* in models of this sort. The algorithm, which copes with the well-known problem of *particle ancestral path degeneracy* at a computational complexity that is only *linear* in the number of particles, is provided with a rigorous theoretical analysis establishing its convergence and long-term stability as well as a simulation study illustrating its computational efficiency.

Given measurable spaces $(\mathsf{X}, \mathcal{X})$ and $(\mathsf{Y}, \mathcal{Y})$, an HMM is a bivariate stochastic process $\{(X_t, Y_t)\}_{t\in\mathbb{N}}$ (where t will often be referred to as "time" without being necessarily a temporal index) taking its values in the product space $(\mathsf{X} \times \mathsf{Y}, \mathcal{X} \otimes \mathcal{Y})$, where the X-valued marginal process $\{X_t\}_{t\in\mathbb{N}}$ is a *Markov chain* (often referred to as the *state sequence*) which is only partially observed through the Y-valued *observation process* $\{Y_t\}_{t\in\mathbb{N}}$. Conditionally on the unobserved state sequence $\{X_t\}_{t\in\mathbb{N}}$, the observations are assumed to be independent and such that the conditional distribution of each Y_t depends on the corresponding state X_t only. HMMs are nowadays used within a large variety of scientific and engineering disciplines such as econometrics [6], speech recognition [35], and computational biology [27] (the more than 360 references in [2] for the period 1989–2000 gives an idea of the applicability of these models).

*This work is supported by the Swedish Research Council, Grant 2011-5577.

Any kind of statistical inference in HMMs involves typically the computation of conditional distributions of unobserved states given observations. Of particular interest are the sequences of *filter distributions*, i.e., the conditional distributions of X_t given $Y_{0:t} := (Y_0, \ldots, Y_t)$ (this will be our generic notation for vectors), and *smoothing distributions*, i.e., the joint conditional distributions of $X_{0:t}$ given $Y_{0:t}$, for $t \in \mathbb{N}$. We will denote these distributions by ϕ_t and $\phi_{0:t}$, respectively (precise definitions of these measures are given in Section 2.2). In this paper we are focusing on the problem of computing, recursively in time, smoothed expectations

$$\phi_{0:t} h_t = \int h_t(x_{0:t})\, \phi_{0:t}(\mathrm{d}x_{0:t}) \quad (t \in \mathbb{N}), \tag{1.1}$$

for additive functionals h_t of form

$$h_t(x_{0:t}) := \sum_{\ell=0}^{t-1} \tilde{h}_\ell(x_{\ell:\ell+1}) \quad (x_{0:t} \in \mathsf{X}^{t+1}). \tag{1.2}$$

Expectations of the form (1.1) appear naturally in the context of parameter estimation using the *maximum-likelihood method*, e.g., when computing the *score-function* (the gradient of the log-likelihood function) via the *Fisher identity* or when computing the *intermediate quantity* of the *expectation-maximization* (EM) *algorithm*. Of particular relevance is the situation where the HMM belongs to an *exponential family*. We refer to [5, Sections 10 and 11] for a comprehensive treatment of these matters. Moreover, *online* implementations of EM (see, e.g., [29, 3]) require typically such smoothed expectations to be computed in an online fashion. In the case of *marginal smoothing* the interest lies in computing conditional expectations of some state $X_{\hat{s}}$ given $Y_{0:t}$ for $t \geq \hat{s}$, which can be cast into our framework by letting, in (1.2), $\tilde{h}_\ell = 0$ for $\ell \neq \hat{s}$ and $\tilde{h}_{\hat{s}}(x_{\hat{s}:\hat{s}+1}) = \tilde{h}_{\hat{s}}(x_{\hat{s}})$. Nevertheless, since exact computation of smoothed expectations is possible only in the cases of linear Gaussian HMMs or HMMs with finite state space, we are in general referred to finding approximations of these quantities, and the present paper focuses on the use of SMC-based techniques for this task. A particle filter approximates the flow $\{\phi_t\}_{t\in\mathbb{N}}$ of filter distributions by a sequence of occupation measures associated with samples $\{(\xi_t^i, \omega_t^i)\}_{i=1}^N$, $t \in \mathbb{N}$, of random draws, *particles* (the ξ_t^i's), with associated non-negative importance weights (the ω_t^i's). Particle filters revolve around two operations: a *selection step* duplicating/discarding particles with large/small importance weights, respectively, and a *mutation step* evolving randomly the selected particles in the state space. The first and most basic implementation, the so-called *bootstrap particle filter* [21] (see also [25]), propagates, in the mutation step, the particles according to the dynamics of the hidden Markov chain and selects the same multinomially according to importance weights proportional to the local likelihood of each particle given the current observation. This scheme imposes a dynamics of the particle cloud that resembles closely that of the filter distribution flow. Due to its very strong potential to solve nonlinear/non-Gaussian filtering problems, SMC methods have been subject to extensive research during the last two decades, resulting in a broad range of developments and variations of the original scheme; see, e.g., [15, 5, 4, 17] and the references therein.

Interestingly, the particle filter provides, as a by-product, approximations also of the joint smoothing distributions in the sense that for each $t \in \mathbb{N}$, the occupation measure associated with the *ancestral lines* of the particles $\{\xi_t^i\}_{i=1}^N$ forms, when the lines are assigned the corresponding weights $\{\omega_t^i\}_{i=1}^N$, an estimate of $\phi_{0:t}$. Unfortunately, this *Poor man's smoother* (using the terminology of [14]) has a major flaw in that resampling systematically the particles leads to significant depletion of the trajectories and the existence of a random time before which all the ancestor paths coincide.

In fact, [24] established, in the case of a compact state space X, a bound on the expected hight of the "crown" of the ancestral tree, i.e., the expected time distance from the last generation back to the most recent common ancestor, which is proportional to $N\log(N)$ and *uniform* in time. Thus, the ratio of the length of the "crown" to that of the "trunk" tends to zero when time increases, implying that the Monte Carlo approximation obtained through this naive approach will, for long observation records, be based on practically a *single draw*, leading to a depleted estimator with a variance that grows quadratically with time.

1.1. Previous work

In the case of additive state functionals it is possible to cope partly with the degeneracy problem described above by means of a *fixed-lag smoothing* technique [26, 30, 32]. This approach avoids the particle path degeneracy by "localizing" the smoothing of a certain state around observations that are only significantly statistically dependent of the state in question and discarding remote and weakly influential observations, i.e., subsequent observations located at a time distance from the state exceeding a lag chosen by the user. The method is expected to work well if the mixing properties of the model allow the lag to be smaller than the length of the "crown" of the ancestral tree. Still, such truncation introduces a mixing-dependent bias, and designing the size of the lag is thus a non-trivial task.

A completely different way of approaching the problem goes via the so-called *forward-filtering backward-smoothing decomposition*, which is based on the fact that the latent process still satisfies the Markov property when evolving backward in time and conditionally on the observations. Consequently, each smoothing measure $\phi_{0:t}$ can be represented as the joint law of this inhomogeneous backward chain with initial distribution given by the corresponding filter ϕ_t. Since the transition kernels of the backward chain depend on the filter distributions, which may be estimated efficiently by a particle filter, a particle-based approximation of the smoothing distribution can thus be naturally obtained by running, in a prefatory filtering pass, the particle filter up to time t (if $\phi_{0:t}$ is the distribution of interest) and, in a backward pass, forming particle-based estimates of the backward kernels (and consequently the smoothing distribution) by modifying the particle weights computed in the forward pass. This scheme, which avoids completely the path degeneracy problem at the cost of a rather significant computational complexity, is referred to as the *forward-filtering backward smoothing* (FFBSm) *algorithm* [16, 23, 25]. As an alternative, the *forward-filtering backward simulation* (FFBSi) *algorithm* [20] generates, in order to reduce the computational overhead of FFBSm, trajectories being approximately distributed according to the smoothing distribution by *simulating* transitions according to the backward dynamics induced by the particle filter approximations produced by the forward pass; as a consequence, FFBSm can be viewed as a Rao-Blackwellized version of FFBSi. These two algorithms correspond directly to the *Rauch-Tung-Striebel smoother* [36] for linear Gaussian HMMs or the *Baum-Welch algorithm* [1] for HMMs with finite state space. FFBSm and FFBSi were analyzed theoretically in [11] (see also [9]), which provides exponential concentration inequalities and well as central limit theorems (CLTs) for these algorithms. Since each backward draw of FFBSi requires a normalizing constant with N terms to be computed, the overall complexity of the algorithm is $\mathcal{O}(N^2)$. Under the mild assumption that the transition density of the latent chain is uniformly bounded, this complexity can be reduced to $\mathcal{O}(N)$ by means of simple accept-reject approach. The latter technique, which was found in [11], will play a key role also in the development of the present paper. Since the Markov transition kernels of the backward chain depend on the filter distributions, the FFBSm and FFBSi algorithms require in general *batch mode*

processing of the observations. This is the case also for the $\mathcal{O}(N)$ smoother proposed in [19], which is based on the *two-filter representation* of each marginal smoothing distribution.

When the objective consists in *online* smoothing of additive state functionals (1.2), *recursive* approximation of the forward-filtering backward-smoothing decomposition can be achieved by introducing the auxiliary statistics

$$\mathbf{T}_t h_t(x_t) = \mathbb{E}[h_t(X_{0:t}) \mid X_t = x_t, Y_{0:t}] \quad (t \in \mathbb{N}, x_t \in \mathsf{X}),$$

where $\mathbb{E}$ denotes expectation associated with the law of the canonical version of the HMM (more precisely, in the previous expression $\mathbf{T}_t$ is a normalized transition kernel which will be defined in Section 2.2). This auxiliary statistic can be updated online according to

$$\mathbf{T}_{t+1} h_{t+1}(x_{t+1}) = \mathbb{E}[\mathbf{T}_t h_t(X_t) + \tilde{h}_t(X_t, x_{t+1}) \mid X_{t+1} = x_{t+1}, Y_{0:t}] \quad (t \in \mathbb{N}, x_{t+1} \in \mathsf{X}); \tag{1.3}$$

see [29, 3, 9]. In this recursive formula, the expectation is taken under the backward kernel describing the conditional distribution of X_t given X_{t+1} and $Y_{0:t}$. On the basis of the auxiliary statistics, each smoothed additive functional may be computed as

$$\phi_{0:t} h_t = \int \mathbf{T}_t h_t(x_t)\, \phi_t(\mathrm{d}x_t) \quad (t \in \mathbb{N}).$$

Following [9], a particle representation of the recursion (1.3) is naturally formed using the estimates of the retrospective dynamics provided by the FFBSm algorithm. Interestingly, this yields a procedure that estimates, as new observations become available, the smoothing distribution flow in a forward-only manner while avoiding completely any problems of particle path degeneracy. However, since the method requires the normalizations of the backward kernels to be computed for each forward particle, the overall complexity of this algorithm is again $\mathcal{O}(N^2)$, which is unrealistic for large particle sample sizes.

1.2. Our approach

Our novel algorithm, which we will refer to as the *particle-based, rapid incremental smoother* (PaRIS), is, similarly to the forward-only implementation of FFBSm proposed in [9], based on (1.3) and can be viewed as an adaptation of the FFBSi algorithm to this recursion. It also shares some similarities with the ancestor sampling approach within the framework of particle Gibbs sampling [28]. Appealingly, we are able to adopt the accept-technique proposed by [11], yielding a fast algorithm with $\mathcal{O}(N)$ complexity. PaRIS differs from the forward-only implementation of FFBSm in the way the update (1.3) of the auxiliary function is implemented; more specifically, instead of computing each subsequent auxiliary statistic as the expected sum of the previous statistic and the incremental term under the retrospective dynamics induced by the particle filter, PaRIS *simulates* $\tilde{N}$ such sums using the backward kernel and updates each statistic by taking the *sample mean* of these draws. Thus, as for the FFBSi algorithm, forward-only FFBSm can be viewed as a Rao-Blackwellization of PaRIS. Interestingly, the design of the sample size $\tilde{N}$ is ultimately critical, as the naive choice $\tilde{N} = 1$ leads to a degeneracy phenomenon that resembles closely that of the Poor man's smoother and, consequently, a variance that grows quadratically with t; on the other hand, for all $\tilde{N} \geq 2$ the algorithm stays numerically stable in the long run with a *linearly* increasing variance. The main objective of the present paper is to investigate theoretically this phase transition by, first,

deriving, via a non-asymptotic Hoeffding-type inequality, the asymptotic (as N tends to infinity) variance of the Monte Carlo estimates produced by the algorithm (which is highly nontrivial due to the complex dependence structures induced by the backward simulation) and, second, verifying that this asymptotic variance is, for any $\tilde{N} \geq 2$, of order $\mathcal{O}(t)$ and $\mathcal{O}(1)$ in the cases of joint smoothing and marginal smoothing, respectively.The authors are not aware of any similar analysis in the SMC literature. The stability results are obtained under strong mixing assumptions that are standard in the literature of SMC analysis (see, e.g., [10, 8, 5]). Also the numerical performance of algorithm is investigated in a simulation study, comprising a linear Gaussian state space model (for which any quantity of interest may be computed exactly using the Rauch-Tung-Striebel smoother) and a stochastic volatility model.

We finally point out that the PaRIS algorithm was outlined by us in the conference note [33] without any theoretical support; in the present paper we are able to confirm, through a rigorous theoretical analysis, the conjectures made in the note in question concerning the stability properties of the algorithm.

To sum up, the smoothing algorithm we propose

- is computationally very efficient and easy to implement,
- does not suffer from particle lineage degeneracy,
- allows the observed data of the HMM to be processed online with minimal memory requirements, and
- is furnished with rigorous theoretical results describing the convergence and numeric stability of the same.

1.3. Outline

After having introduced some kernel notation, HMMs, and the smoothing problem in Section 2, we describe carefully, in Section 2.3, particle filters, FFBSm (and its forward-only implementation), and FFBSi. Section 3.1 contains the derivation of our novel algorithm as well as some discussion of the choice of the design parameter $\tilde{N}$. Our theoretical results are presented in Section 3.2, including a Hoeffding-type inequality (Theorem 1) and a CLT (Theorem 3). Section 3.2.3 is devoted to the numerical stability of PaRIS in the case $\tilde{N} \geq 2$, and Theorem 8 and Theorem 9 provide variance bounds in the cases of joint and marginal smoothing, respectively. In Section 4 we test numerically the algorithm and some conclusions are drawn in Section 5. Finally, Appendix A and Appendix B provide all proofs and some technical results, respectively.

2. Preliminiaries

2.1. Notation

Before going into the details concerning HMMs and particle filters we introduce some notation. For any measurable space $(\mathsf{X}, \mathcal{X})$, where $\mathcal{X}$ is a countably generated σ-algebra, we denote by $\mathsf{F}(\mathcal{X})$ the set of bounded $\mathcal{X}/\mathcal{B}(\mathbb{R})$-measurable functions on X. For any $h \in \mathsf{F}(\mathcal{X})$, we let $\|h\|_\infty := \sup_{x \in \mathsf{X}} |h(x)|$ and $\operatorname{osc}(h) := \sup_{(x,x') \in \mathsf{X}^2} |h(x) - h(x')|$ denote the sup and oscillator norms of h, respectively. Let $\mathsf{M}(\mathcal{X})$ be the set of σ-finite measures on $(\mathsf{X}, \mathcal{X})$ and $\mathsf{M}_1(\mathcal{X}) \subset \mathsf{M}(\mathcal{X})$ the probability measures. Given $n \in \mathbb{N}$, we will denote product sets and product σ-fields by $\mathsf{X}^n := \mathsf{X} \times \cdots \times \mathsf{X}$ and $\mathcal{X}^n := \mathcal{X} \otimes \cdots \otimes \mathcal{X}$ (n times), respectively. For real numbers and integers we define the sets $\mathbb{R}_+ := [0, \infty)$, $\mathbb{R}_+^* := (0, \infty)$,

$\mathbb{N} := \{0, 1, 2, \ldots\}$, and $\mathbb{N}^* := \{1, 2, 3, \ldots\}$. For any quantities $\{a_\ell\}_{\ell=m}^n$we denote vectors as $a_{m:n} := (a_m, \ldots, a_n)$ and for any $(m, n) \in \mathbb{N}^2$ such that $m \leq n$ we denote $[\![m, n]\!] := \{m, m+1, \ldots, n\}$. The cardinality of a set S is denoted by $\#\mathsf{S}$.

An unnormalized transition kernel $\mathbf{K}$ from $(\mathsf{X}, \mathcal{X})$ to $(\mathsf{Y}, \mathcal{Y})$ induces two integral operators, one acting on functions and the other on measures. More specifically, let $h \in \mathsf{F}(\mathcal{X} \otimes \mathcal{Y})$ and $\nu \in \mathsf{M}(\mathcal{X})$, and define the measurable function

$$\mathbf{K}h : \mathsf{X} \ni x \mapsto \int h(x, y)\, \mathbf{K}(x, \mathrm{d}y),$$

and the measure

$$\nu\mathbf{K} : \mathcal{Y} \ni \mathsf{A} \mapsto \int \mathbf{K}(x, \mathsf{A})\, \nu(\mathrm{d}x),$$

whenever these quantities are well-defined. Moreover, let $\mathbf{K}$ be defined as above and let $\mathbf{L}$ be another unnormalized transition kernel from $(\mathsf{Y}, \mathcal{Y})$ to a third measurable space $(\mathsf{Z}, \mathcal{Z})$; we then define two different *products* of $\mathbf{K}$ and $\mathbf{L}$, namely

$$\mathbf{KL} : \mathsf{X} \times \mathcal{Z} \ni (x, \mathsf{A}) \mapsto \int \mathbf{K}(x, \mathrm{d}y)\, \mathbf{L}(y, \mathsf{A})$$

and

$$\mathbf{K} \otimes \mathbf{L} : \mathsf{X} \times (\mathcal{Y} \otimes \mathcal{Z}) \ni (x, \mathsf{A}) \mapsto \int \mathbb{1}_{\mathsf{A}}(y, z)\, \mathbf{K}(x, \mathrm{d}y)\, \mathbf{L}(y, \mathrm{d}z),$$

whenever these are well-defined. Note that the previous products form new transition kernels from $(\mathsf{X}, \mathcal{X})$ to $(\mathsf{Z}, \mathcal{Z})$ and from $(\mathsf{X}, \mathcal{X})$ to $(\mathsf{Y} \times \mathsf{Z}, \mathcal{Y} \otimes \mathcal{Z})$, respectively. We also define the $\otimes$-product of a kernel $\mathbf{K}$ and a measure $\nu \in \mathsf{M}(\mathcal{X})$ as the new measure

$$\nu \otimes \mathbf{K} : \mathcal{X} \otimes \mathcal{Y} \ni \mathsf{A} \mapsto \int \mathbb{1}_{\mathsf{A}}(x, y)\, \mathbf{K}(x, \mathrm{d}y)\, \nu(\mathrm{d}x).$$

The concept of *reverse kernels* will be of importance in the coming developments. For a kernel $\mathbf{K}$ from $(\mathsf{X}, \mathcal{X})$ to $(\mathsf{Y}, \mathcal{Y})$ and a probability measure $\eta \in \mathsf{M}_1(\mathcal{X})$, the reverse kernel $\overleftarrow{\mathbf{K}}_\eta$ associated with $(\eta, \mathbf{K})$ is a transition kernel from $(\mathsf{Y}, \mathcal{Y})$ to $(\mathsf{X}, \mathcal{X})$ satisfying, for all $h \in \mathsf{F}(\mathcal{X} \otimes \mathcal{Y})$,

$$(\eta \otimes \mathbf{K})h = (\eta\mathbf{K}) \otimes \overleftarrow{\mathbf{K}}_\eta h.$$

A reverse kernel does not always exist; however if $\mathbf{K}$ has a transition density κ with respect to some reference measure in $\mathsf{M}(\mathcal{Y})$, then $\overleftarrow{\mathbf{K}}_\eta$ exists and is given by

$$\overleftarrow{\mathbf{K}}_\eta h(x) := \frac{\int h(\tilde{x})\kappa(\tilde{x}, x)\, \eta(\mathrm{d}\tilde{x})}{\int \kappa(\tilde{x}, x)\, \eta(\mathrm{d}\tilde{x})} \quad (h \in \mathsf{F}(\mathcal{X}), x \in \mathsf{X}) \tag{2.1}$$

(see [5, Section 2.1] for details).

Finally, for any kernel $\mathbf{K}$ and any bounded measurable function h we write $\mathbf{K}^2 h := (\mathbf{K}h)^2$ and $\mathbf{K}h^2 := \mathbf{K}(h^2)$. Similar notation will be used for measures.

2.2. Hidden Markov models

Let $(\mathsf{X}, \mathcal{X})$ and $(\mathsf{Y}, \mathcal{Y})$ be some measurable spaces, $\mathbf{Q} : \mathsf{X} \times \mathcal{X} \to [0,1]$ and $\mathbf{G} : \mathsf{X} \times \mathcal{Y} \to [0,1]$ some Markov transition kernels, and $\chi \in \mathsf{M}_1(\mathcal{X})$. We define an HMM as the canonical version of the bivariate Markov chain $\{(X_t, Y_t)\}_{t\in\mathbb{N}}$ having transition kernel

$$\mathsf{X} \times \mathsf{Y} \times (\mathcal{X} \otimes \mathcal{Y}) : ((x,y), \mathsf{A}) \mapsto \mathbf{Q} \otimes \mathbf{G}(x, \mathsf{A}). \tag{2.2}$$

and initial distribution $\chi \otimes \mathbf{G}$. The state process $\{X_t\}_{t\in\mathbb{N}}$ is assumed to be only partially observed through the observations process $\{Y_t\}_{t\in\mathbb{N}}$. The dynamics (2.2) implies that (we refer to [5, Section 2.2] for details)

(i) the state sequence $\{X_t\}_{t\in\mathbb{N}}$ is a Markov chain with transition kernel $\mathbf{Q}$ and initial distribution χ,
(ii) the observations are, conditionally on the states, independent and such that the conditional distribution of each Y_t depends on the corresponding X_t only and is given by the *emission distribution* $\mathbf{G}(X_t, \cdot)$.

We will throughout the paper assume that $\mathbf{G}$ admits a density g (referred to as the *emission density*) with respect to some reference measure $\nu \in \mathsf{M}(\mathcal{Y})$, i.e.,

$$\mathbf{G}h(x) = \int h(y) g(x,y)\, \nu(\mathrm{d}y) \quad (x \in \mathsf{X}, h \in \mathsf{F}(\mathcal{Y})).$$

In the following we assume that we are given a distinguished sequence $\{y\}_{t\in\mathbb{N}}$ of observations of $\{Y_t\}_{t\in\mathbb{N}}$, and will in general omit the dependence on these observations from the notation. Thus, define $g_t(x) := g(x, y_t)$, $x \in \mathsf{X}$. For any $(s, s', t) \in \mathbb{N}^3$ such that $0 \le s \le s' \le t$ we denote by $\phi_{s:s'|t}$ the conditional distribution (posterior) of $X_{s:s'}$ given the observations $Y_{0:t} = y_{0:t}$. This distribution may be expressed as

$$\phi_{s:s'|t} h = \frac{\int \cdots \int h(x_{s:s'}) g_0(x_0)\, \chi(\mathrm{d}x_0) \prod_{\ell=0}^{t-1} g_{\ell+1}(x_{\ell+1})\, \mathbf{Q}(x_\ell, \mathrm{d}x_{\ell+1})}{\int \cdots \int g_0(x_0)\, \chi(\mathrm{d}x_0) \prod_{\ell=0}^{t-1} g_{\ell+1}(x_{\ell+1})\, \mathbf{Q}(x_\ell, \mathrm{d}x_{\ell+1})} \quad (h \in \mathsf{F}(\mathcal{X}^{s'-s+1})) \tag{2.3}$$

(assuming that the denominator is non-zero). If $s = s' = t$, we let ϕ_t be shorthand for $\phi_{t|t}$, i.e., the filter distribution at time t. If $s = 0$ and $s' = t$, then $\phi_{0:t|t}$ is the joint smoothing distribution. For $t \in \mathbb{N}$, define the unnormalized transition kernels

$$\mathbf{L}_t h(x) := \mathbf{Q}(g_{t+1} h)(x) \quad (x \in \mathsf{X}, h \in \mathsf{F}(\mathcal{X})),$$

with the convention that $\mathbf{L}_s \mathbf{L}_t \equiv \mathrm{id}$ whenever $s > t$. In addition, we let $\mathbf{L}_{-1}$ be the Boltzmann multiplicative operator associated with g_0, i.e., $\mathbf{L}_{-1} h(x) \equiv g_0(x) h(x)$ for all $h \in \mathsf{F}(\mathcal{X})$ and $x \in \mathsf{X}$. By combining this notation with (2.3) we may express each filter distribution as

$$\phi_t = \frac{\chi \mathbf{L}_{-1} \cdots \mathbf{L}_{t-1}}{\chi \mathbf{L}_{-1} \cdots \mathbf{L}_{t-1} \mathbb{1}_{\mathsf{X}}} \quad (t \in \mathbb{N}),$$

which implies immediately the *filter recursion*

$$\phi_{t+1} = \frac{\phi_t \mathbf{L}_t}{\phi_t \mathbf{L}_t \mathbb{1}_{\mathsf{X}}} \quad (t \in \mathbb{N}). \tag{2.4}$$

In the following we will often deal with sums and products of functions with possibly different arguments. Since these functions will be defined on products of X, we will, when needed, with a slight abuse of notation, let subscripts define the domain and the values of such sums and products. For instance, $f_t\tilde{f}_t : \mathsf{X} \ni x_t \mapsto f_t(x_t)\tilde{f}_t(x_t)$ while $f_t + \tilde{f}_{t+1} : \mathsf{X}^2 \ni (x_t, x_{t+1}) \mapsto f_t(x_t) + \tilde{f}_{t+1}(x_{t+1})$.

We will for simplicity assume that the HMM is *fully dominated*, i.e., that also $\mathbf{Q}$ admits a transition density q with respect to some reference measure $\mu \in \mathsf{M}(\mathcal{X})$. In this case the reverse kernel $\overleftarrow{\mathbf{Q}}_\eta$ of $\mathbf{Q}$ with respect to any $\eta \in \mathsf{M}_1(\mathcal{X})$ is well-defined and specified by (2.1) (with $\kappa = q$). It may be shown (see, e.g., [5, Proposition 3.3.6]) that the state process has still the Markov property when evolving conditionally on $Y_{0:t} = y_{0:t}$ in the time-reversed direction; moreover, the distribution of X_s given X_{s+1} and $Y_{0:t} = y_{0:t}$ is, for any $s \leq t$, given by $\overleftarrow{\mathbf{Q}}_{\phi_s}$, which is referred to as the *backward kernel at time* s. Consequently, we may express each joint smoothing distribution $\phi_{0:t|t}$ as

$$\phi_{0:t|t} = \phi_t \mathbf{T}_t, \tag{2.5}$$

where we have defined the kernels

$$\mathbf{T}_t := \begin{cases} \overleftarrow{\mathbf{Q}}_{\phi_{t-1}} \otimes \overleftarrow{\mathbf{Q}}_{\phi_{t-2}} \otimes \cdots \otimes \overleftarrow{\mathbf{Q}}_{\phi_0} & \text{for } t \in \mathbb{N}^*, \\ \mathrm{id} & \text{for } t = 0. \end{cases}$$

As discussed in the introduction, the aim of this paper is, given a sequence $\{\tilde{h}_t\}_{t\in\mathbb{N}}$ of terms, to estimate the sequence $\{\phi_{0:t|t}h_t\}_{t\in\mathbb{N}}$, where each h_t is given by (1.2). By convention, $h_0 \equiv 0$ (implying, e.g., that $\mathbf{T}_0 h_0 = 0$). Using (2.5), each quantity of interest may be expressed as $\phi_t \mathbf{T}_t h_t = \phi_{0:t|t} h_t$. In addition, note that $\{\mathbf{T}_t h_t\}_{t\in\mathbb{N}}$ may be expressed recursively as

$$\mathbf{T}_{t+1} h_{t+1} = \overleftarrow{\mathbf{Q}}_{\phi_t}(\mathbf{T}_t h_t + \tilde{h}_t), \tag{2.6}$$

a formula that will play a key role in the coming developments.

Finally, define, for $(s,t) \in \mathbb{N}^2$ such that $s \leq t$, the *retro-prospective kernels*

$$\begin{aligned} \mathbf{D}_{s,t} h(x_s) &:= \iint h(x_{0:t})\, \mathbf{T}_s(x_s, \mathrm{d}x_{0:s-1})\, \mathbf{L}_s \cdots \mathbf{L}_{t-1}(x_s, \mathrm{d}x_{s+1:t}) \\ \tilde{\mathbf{D}}_{s,t} h(x_s) &:= \mathbf{D}_{s,t}(h - \phi_{0:t|t} h)(x_s) \end{aligned} \qquad (x_s \in \mathsf{X}, h \in \mathsf{F}(\mathcal{X}^{t+1}))$$

operating simultaneously in the backward and forward directions. Note that the only difference between $\mathbf{D}_{s,t}$ and $\tilde{\mathbf{D}}_{s,t}$ is that the latter is centralized around the joint smoothing distribution.

2.3. Particle-based smoothing in HMMs

2.3.1. The bootstrap particle filter

In the following we assume that all random variables are defined on a common probability space $(\Omega, \mathcal{F}, \mathbb{P})$. The bootstrap particle filter updates sequentially in time a set of particles and associated weights in order to approximate the filter distribution flow $\{\phi_t\}_{t\in\mathbb{N}}$ given the sequence $\{y_t\}_{t\in\mathbb{N}}$ of observations. Assume that we have at hand a particle sample $\{(\omega_t^i, \xi_t^i)\}_{i=1}^N$ approximating the filter distribution ϕ_t in the sense that for all $h \in \mathsf{F}(\mathcal{X})$,

$$\phi_t^N h = \sum_{i=1}^N \frac{\omega_t^i}{\Omega_t} h(\xi_t^i) \overset{N\to\infty}{\rightharpoonup} \phi_t h, \tag{2.7}$$

where $\Omega_s := \sum_{i=1}^N \omega_t^i$ denotes the weight sum. To form a weighted particle sample $\{(\omega_{t+1}^i, \xi_{t+1}^i)\}_{i=1}^N$ targeting the subsequent filter ϕ_{t+1} we simply the plug the approximation ϕ_t^N into the filter recursion (2.4), yielding the approximation of ϕ_{t+1} by a mixture distribution proportional to $\sum_{i=1}^N \omega_t^i \mathbf{L}_t(\xi_t^i, \cdot)$, and aim at updating the particle cloud by sampling from this mixture. However, since $\mathbf{L}_t$ is generally intractable, we augment the space by the index i and apply importance sampling from the extended distribution proportional to $\omega_t^i \mathbf{L}_t(\xi_t^i, \cdot)$ using the distribution proportional to $\omega_t^i \mathbf{Q}(\xi_t^i, \cdot)$ as instrumental distribution. This yields a sampling schedule comprising two operations: selection and mutation. In the selection step, a set $\{I_{t+1}^i\}_{i=1}^N$ of indices are drawn multinomially according to probabilities proportional to $\{\omega_t^i\}_{i=1}^N$. After this, the mutation step propagates the particles forward according to the dynamics of the state process and assigns the mutated particles importance weights given by the emission density, i.e., for all $i \in [\![1, N]\!]$,

$$\xi_{t+1}^i \sim \mathbf{Q}(\xi_t^{I_{t+1}^i}, \cdot),$$
$$\omega_{t+1}^i = g_{t+1}(\xi_{t+1}^i).$$

The algorithm, which is the standard bootstrap particle filter presented in [21], is initialized by drawing $\{\xi_0^i\}_{i=1}^N \sim \chi^{\otimes N}$ and letting $\omega_0^i = g_0(\xi_0^i)$ for all $i \in [\![1, N]\!]$. In this basic scheme, which is summarized in Algorithm 1, the information provided by the most current observation y_{t+1} enters the algorithm via the importance weights only. However, instead of moving the particles "blindly" according to the latent dynamics $\mathbf{Q}$, it is, in order to direct the particle swarm toward regions of the state space with large posterior probability, possible to increase the influence of the last observation on the mutation moves as well as the selection mechanism step via the framework of *auxiliary particle filters* [34]. Even though all the results of the present paper can be extended straightforwardly to auxiliary particle filters, we have chosen to limit the presentation to bootstrap-type particle filters only for clarity.

Algorithm 1 Bootstrap particle filter

Require: A weighted particle sample $\{(\xi_t^i, \omega_t^i)\}_{i=1}^N$ targeting ϕ_t.
1: **for** $i = 1 \to N$ **do**
2: $I_{t+1}^i \sim \mathsf{Pr}(\{\omega_t^\ell\}_{\ell=1}^N)$;
3: draw $\xi_{t+1}^i \sim \mathbf{Q}(\xi_t^{I_{t+1}^i}, \cdot)$;
4: set $\omega_{t+1}^i \leftarrow g_{t+1}(\xi_{t+1}^i)$;
5: **end for**
6: **return** $\{(\xi_t^i, \omega_t^i)\}_{i=1}^N$.

In the following we will express Algorithm 1 in a compact form by writing

$$\text{“}\{(\xi_{t+1}^i, \omega_{t+1}^i)\}_{i=1}^N \leftarrow \mathsf{PF}\left(\{(\xi_t^i, \omega_t^i)\}_{i=1}^N\right)\text{”}.$$

2.3.2. Forward-filtering backward-smoothing (FFBSm)

As discussed in the introduction, the bootstrap filter may also be used for smoothing, as the weighted occupation measures associated with the genealogical trees of the particle samples generated by the algorithm form consistent estimates of the joint smoothing distributions. A way of detouring the particle path degeneracy of this Poor man's smoother goes via the backward decomposition (2.5),

granted that we are able to approximate each kernel $\overleftarrow{\mathbf{Q}}_{\phi_s}$, $s \in \mathbb{N}$. However, considering instead the reverse kernel associated with the particle filter ϕ_s^N, yields, via (2.1), the particle approximations

$$\overleftarrow{\mathbf{Q}}_{\phi_s^N} h(x) = \sum_{i=1}^N \frac{\omega_s^i q(\xi_s^i, x)}{\sum_{\ell=1}^N \omega_s^\ell q(\xi_s^\ell, x)} h(\xi_s^i) \quad (x \in \mathsf{X}, h \in \mathsf{F}(\mathcal{X})). \tag{2.8}$$

FFBSm consists in simply inserting these approximations into (2.5), i.e., approximating, for $h \in \mathsf{F}(\mathcal{X}^{t+1})$, $\phi_{0:t|t} h$ by

$$\phi_{0:t|t}^N h := \sum_{i_0=1}^N \cdots \sum_{i_t=1}^N \left(\prod_{s=0}^{t-1} \frac{\omega_s^{i_s} q(\xi_s^{i_s}, \xi_{s+1}^{i_{s+1}})}{\sum_{\ell=1}^N \omega_s^\ell q(\xi_s^\ell, \xi_{s+1}^{i_{s+1}})} \right) \frac{\omega_t^{i_t}}{\Omega_t} h(\xi_0^{i_0}, \ldots, \xi_t^{i_t}). \tag{2.9}$$

For general objective functions h, this occupation measure is impractical as the cardinality of its support grows geometrically fast with time. In the case where the objective function h is of additive form (1.2) the computational complexity is still quadratic, since computation of the normalizing constants $\sum_{\ell=1}^N \omega_s^\ell q(\xi_s^\ell, \xi_{s+1}^i)$ is required for all $i \in [\![1, N]\!]$ and $s \in [\![0, t-1]\!]$. Consequently, FFBSm a computationally intensive approach.

2.3.3. Forward-only implementation of FFBSm

Appealingly, as noted by [9], in the case of additive state functionals the sequence $\{\phi_t^N h_t\}_{t \in \mathbb{N}}$ can be computed *on-the-fly* as t increases on the basis of the recursion (2.6). More specifically, plugging the estimates (2.8) into the recursion in question yields particle approximations $\{\tilde{\tau}_t^i\}_{i=1}^N$ of the statistics $\{\mathbf{T}_t h_t(\xi_t^i)\}_{i=1}^N$ evaluated at the particle locations. After initializing $\tilde{\tau}_0^i = 0$ for all $i \in [\![1, N]\!]$, these approximations may, when new observations become available, be updated by first evolving the particle filter sample one step and then setting

$$\tilde{\tau}_{t+1}^i = \sum_{j=1}^N \frac{\omega_t^j q(\xi_t^j, \xi_{t+1}^i)}{\sum_{\ell=1}^N \omega_t^\ell q(\xi_t^\ell, \xi_{t+1}^i)} \{\tilde{\tau}_t^j + \tilde{h}_t(\xi_t^j, \xi_{t+1}^i)\} \quad (t \in \mathbb{N}), \tag{2.10}$$

yielding, via (2.5), the estimate

$$\phi_{0:t|t}^N h_t = \sum_{i=1}^N \frac{\omega_t^i}{\Omega_t} \tilde{\tau}_t^i,$$

of $\phi_{0:t|t} h_t$. Besides allowing for online processing of the data, the algorithm has also the appealing property that only the current statistics $\{\tilde{\tau}_t^i\}_{i=1}^N$ and particle sample $\{(\xi_t^i, \omega_t^i)\}_{i=1}^N$ need to be stored in the memory. Still, the complexity of the scheme is $\mathcal{O}(N^2)$ due to the computation of the normalizing constants of the backward kernel induced by the particle filter.

2.3.4. Forward-filtering backward-simulation (FFBSi)

In order to remedy the high computational complexity of FFBSm, FFBSi generates trajectories on the *index space* $[\![1, N]\!]^{t+1}$ by simulating repeatedly a time-reversed, inhomogeneous Markov chain $\{\tilde{J}_s\}_{s=0}^t$ with transition probabilities

$$\mathbf{\Lambda}_s^N(i, j) := \frac{\omega_s^j q(\xi_s^j, \xi_{s+1}^i)}{\sum_{\ell=1}^N \omega_t^\ell q(\xi_s^\ell, \xi_{s+1}^i)} \quad ((s, i, j) \in [\![0, t-1]\!] \times [\![1, N]\!]^2) \tag{2.11}$$

and initial distribution (i.e., distribution at time t) $\mathsf{Pr}(\{\omega_t^j\}_{j=1}^N)$. Given $\{\tilde{J}_s\}_{s=0}^t$, an approximate draw from the joint smoothing distribution is formed by the random vector $(\xi_0^{\tilde{J}_0}, \ldots, \xi_t^{\tilde{J}_t})$. Consequently, the uniformly weighted occupation measure associated with a set of conditionally independent such draws provides a finite-dimensional approximation of the smoothing distribution $\phi_{0:t|t}$; see [20]. In this basic formulation of FFBSi, the backward sampling pass requires the normalizing constants of the particle-based backward kernels to be computed, and hence the algorithm suffers from a quadratic complexity. On the other hand, on the contrary to FFBSm, this complexity is the same for *all* types of objective functions (whereas FFBSm has quadratic complexity only when applied to additive state functionals). However, following [11] it is, under the assumption that there exists $\bar{\varepsilon} \in \mathbb{R}_+^*$ such that $q(x, x') \leq \bar{\varepsilon}$ for all $(x, x') \in \mathsf{X}^2$ (an assumption that is satisfied for most models of interest), possible to reduce the computational complexity of FFBSi by simulating the approximate backward kernel using the following accept-reject technique. In order to sample from $\mathsf{Pr}(\{\mathbf{\Lambda}_s^N(i,j)\}_{j=1}^N)$ for given $s \in [\![0, t-1]\!]$ and $i \in [\![1, N]\!]$, a candidate J^* drawn from the proposal distribution $\mathsf{Pr}(\{\omega_s^j\}_{j=1}^N)$ is accepted with probability $q(\xi_s^{J^*}, \xi_{s+1}^i)/\bar{\varepsilon}$. The procedure repeated until acceptance; see Algorithm 3 for an efficient way of implementing this approach. Under the additional assumption that the transition density is bounded also from below (see Assumption 2 below) it can be shown (see [11, Proposition 2]) that the computational complexity of this accept-reject-based FFBSi algorithm is indeed *linear* (i.e., $\mathcal{O}(N)$).

3. Main results

Requiring separate forward and backward processing of the data, the standard design of FFBSi is not useful in online applications. We hence propose a novel algorithm which can be viewed as a hybrid between the forward-only implementation of the FFBSm algorithm and the FFBSi algorithm. In order to gain computational effort, it the replaces, in the spirit of FFBSi, exact computation of (2.10) by a Monte Carlo estimate. The algorithm, which is presented in the next section, is furnished with rigorous theoretical results concerning its convergence and numerical stability in Section 3.2.

3.1. The particle-based, rapid incremental smoother (PaRIS)

Given estimates $\{\tau_t^i\}_{i=1}^N$ of the auxiliary statistics $\{\mathbf{T}_t h_t(\xi_t^i)\}_{i=1}^N$ and a particle sample $\{(\xi_t^i, \omega_t^i)\}_{i=1}^N$ targeting the filter ϕ_t, the algorithm updates the estimated auxiliary statistics by, first, propagating the particle cloud one step, yielding $\{(\xi_{t+1}^i, \omega_{t+1}^i)\}_{i=1}^N$, second, drawing, for each $i \in [\![1, N]\!]$, conditionally independent and identically distributed indices $\{J_{t+1}^{(i,j)}\}_{j=1}^{\tilde{N}}$, where $\tilde{N} \in \mathbb{N}^*$ is some given sample size referred to as the *precision parameter*, according to

$$\{J_{t+1}^{(i,j)}\}_{j=1}^{\tilde{N}} \sim \mathsf{Pr}(\{\mathbf{\Lambda}_t^N(i,\ell)\}_{\ell=1}^{\tilde{N}})^{\otimes \tilde{N}} \quad (i \in [\![1, N]\!]),$$

where the transition probabilities $\mathbf{\Lambda}_t^N$ are defined in (2.11), and, third, letting

$$\tau_{t+1}^i = \tilde{N}^{-1} \sum_{j=1}^{\tilde{N}} \left(\tau_t^{J_{t+1}^{(i,j)}} + \tilde{h}_t(\xi_t^{J_{t+1}^{(i,j)}}, \xi_{t+1}^i) \right) \quad (i \in [\![1, N]\!]).$$

Using the updated statistics $\{\tau_{t+1}^i\}_{i=1}^{\tilde{N}}$, an estimate of $\phi_{0:t+1|t+1} h_{t+1} = \phi_{t+1} \mathbf{T}_{t+1} h_{t+1}$ is obtained as $\sum_{i=1}^N \omega_{t+1}^i \tau_{t+1}^i / \Omega_{t+1}$. As for FFBSm, the algorithm is initialized by setting $\tau_0^i = 0$ for $i \in [\![1, N]\!]$.

The resulting smoother, which is summarized in Algorithm 2, allows for online processing with constant memory requirements, as it requires only the current particle cloud and estimated auxiliary statistics to be stored at each iteration. In addition, applying, in Step (4), the accept-reject technique described in the previous section yields, for a given $\tilde{N}$, an algorithm with *linear complexity*.

Algorithm 2 Particle-based, rapid incremental smoother (PaRIS)

Require: Particles sample $\{(\xi_t^i, \omega_t^i)\}_{i=1}^N$ targeting ϕ_t and estimated auxiliary statistics $\{\tau_t^i\}_{i=1}^N$.
1: run $\{(\xi_{t+1}^i, \omega_{t+1}^i)\}_{i=1}^N \leftarrow \mathsf{PF}(\{(\xi_t^i, \omega_t^i)\}_{i=1}^N)$;
2: **for** $i = 1 \to N$ **do**
3: **for** $j = 1 \to \tilde{N}$ **do**
4: draw $J_{t+1}^{(i,j)} \sim \mathsf{Pr}(\{\omega_t^j q(\xi_t^\ell, \xi_{t+1}^i)\}_{\ell=1}^N)$;
5: **end for**
6: Set $\tau_{t+1}^i \leftarrow \tilde{N}^{-1} \sum_{j=1}^{\tilde{N}} \left(\tau_t^{J_{t+1}^{(i,j)}} + \tilde{h}_t(\xi_t^{J_{t+1}^{(i,j)}}, \xi_{t+1}^i) \right)$;
7: **end for**
8: **return** $\{\tau_{t+1}^i\}_{i=1}^N$ and $\{(\xi_{t+1}^i, \omega_{t+1}^i)\}_{i=1}^N$.

In the PaRIS scheme, the precision parameter $\tilde{N}$ has to be set by the user. As shown in Section 3.2, the algorithm is asymptotically consistent (as the particle sample size N tends to infinity) *for any fixed* $\tilde{N} \in \mathbb{N}^*$ (i.e., the precision parameter does not need to be increased with N in order to guarantee consistency). Increasing the precision parameter increases the accuracy of the algorithm at the cost of additional computational complexity. Importantly, there is a *significant qualitative difference between the cases* $\tilde{N} = 1$ *and* $\tilde{N} \geq 2$, and it turns out that the latter is required to keep PaRIS numerically stable. This will be clear from the theoretical bounds on the asymptotic variance obtained in Section 3.2 as well as from the numerical experiments in Section 4.

In order to understand the fundamental difference between the cases $\tilde{N} = 1$ and $\tilde{N} \geq 2$ we may use the backward indices to connect the particles of different generations. Hence, let, for all $t \in \mathbb{N}$ and $i \in [\![1, N]\!]$, $\mathcal{J}_{t,t}^{(i,\varnothing)} := i$ and, for all $s \in [\![0, t-1]\!]$ and $j_{s:t-1} \in [\![1, \tilde{N}]\!]^{t-s}$,

$$\mathcal{J}_{s,t}^{(i,j_{s:t-1})} := J_{s+1}^{(\mathcal{J}_{s+1,t}^{(i,j_{s+1:t-1})}, j_s)}$$

and let us write "$\xi_s^\ell \dashleftarrow \xi_t^i$" if there exists a sequence $j_{s:t-1}$ of indices such that $\ell = \mathcal{J}_{s,t}^{(i,j_{s:t-1})}$. Note that the support of the PaRIS estimator at time t is given by $\mathsf{S}_t := \prod_{s=0}^t \mathsf{A}_{s,t} \subset \mathsf{X}^{t+1}$, where $\mathsf{A}_{s,t} := \cup_{i=1}^N \{\xi_s^\ell : \xi_s^\ell \dashleftarrow \xi_t^i\} \subset \{\xi_s^\ell\}_{\ell=1}^N$ (so that, since $\xi_t^i \dashleftarrow \xi_t^i$ for all $i \in [\![1, N]\!]$, $\mathsf{A}_{t,t} = \{\xi_t^\ell\}_{\ell=1}^N$). When $\tilde{N} = 1$, the sequence $\{\#\mathsf{A}_{s,t}\}_{t=s}^\infty$ is non-decreasing, and $\#\mathsf{A}_{s,t} = 1 \Rightarrow \#\mathsf{A}_{u,t} = 1$ for all $u \in [\![0, s]\!]$. This implies a degeneracy phenomenon that resembles closely that of the Poor man's smoother. On the contrary, in the case $\tilde{N} \geq 2$ it may well occur that $\#\mathsf{A}_{s,t} > \#\mathsf{A}_{s,t+1}$, also when $\#\mathsf{A}_{s,t+1} = 1$. The previous is, for $N = 3$ and $t = 4$, illustrated graphically in Figure 1, where columns of nodes represent particle clouds at different time steps (with time increasing rightward) and arrows indicate connections through the relation $\dashleftarrow$. Black-colored particles are included in the support S_4 of the final estimator, while gray-colored ones are inactive. As clear from Figure 1(a), setting $\tilde{N} = 1$ depletes quickly the support of the estimator, leading to a numerically unstable algorithm. Figure 1(b) shows the same configuration as in (a), but with one additional backward sample (i.e., $\tilde{N} = 2$). In this case, the sequence $\{\#\mathsf{A}_{s,4}\}_{s=0}^4$ is no longer non-decreasing, and a high degree of depletion at some time points (such as $s = 2$) has merely local effect of the support of the estimator. In the coming sections, the fact that PaRIS stays numerically stable for any fixed $\tilde{N} \geq 2$ is established theoretically as well as through simulations.

(a) $\tilde{N} = 1$ (b) $\tilde{N} = 2$

FIG 1. *Genealogical traces corresponding to backward simulation in the PaRIS algorithm. Columns of nodes refer to different particle populations (with $N = 3$) at different time points (with time increasing rightward) and arrows indicate connections through the relation* $\leftarrow\!\!-$. *Black-colored particles are included in the support S_4 of the final estimator, while gray-colored ones are inactive.*

3.2. Theoretical results

The coming convergence analysis is driven by the following assumption.

Assumption 1.

(i) For all $t \in \mathbb{N}$, $g_t \in \mathsf{F}(\mathcal{X})$ and $g_t(x) > 0$, $x \in \mathsf{X}$,
(ii) $q \in \mathsf{F}(\mathcal{X}^2)$.

Assumption 1(i) implies finiteness and positiveness of the particle weights; the boundedness of the transition density q implied by Assumption 1(ii) allows, besides certain technical arguments (formalized in Lemma 14) based on the generalized Lebesgue theorem, the accept-reject sampling technique discussed in Section 2.3.4 to be used.

In turns out to be necessary to establish the convergence of PaRIS for a slightly more general *affine modification* of the additive state functional (1.2) under consideration. More specifically, we will verify that for all $t \in \mathbb{N}$ and $(f_t, \tilde{f}_t) \in \mathsf{F}(\mathcal{X})^2$,

$$\sum_{i=1}^{N} \frac{\omega_t^i}{\Omega_t} \{\tau_t^i f_t(\xi_t^i) + \tilde{f}_t(\xi_t^i)\} \overset{N\to\infty}{\rightrightarrows} \phi_t(\mathbf{T}_t h_t f_t + \tilde{f}_t), \tag{3.1}$$

where $\{\tau_t^i\}_{i=1}^N$ and $\{(\xi_t^i, \omega_t^i)\}_{i=1}^N$ are the output of Algorithm 2, in the senses of exponential concentration, weak convergence, and L_p error. The analogous results for the original additive state functional are then obtained as corollaries by simply applying (3.1) with $f_t \equiv \mathbb{1}_{\mathsf{X}}$ and $\tilde{f}_t \equiv \mathbb{1}_{\mathsf{X}^c}$. Our proofs, which are presented in Appendix A, are based on single-step analyses of the scheme and rely on techniques developed in [11] and [12]. Nevertheless, the analysis of PaRIS is, especially in the case of weak convergence, highly non-trivial due to the complex dependence between the ancestral lineages of the particles induced by the backward sampling approach (on the contrary to standard FFBSi, where the backward trajectories are conditionally independent; see the previous section).

3.2.1. Hoeffding-type inequalities

Besides being a result of independent interest, the following exponential concentration inequality for finite sample sizes N plays an instrumental role in the proof of the CLT in the next section. For reasons that will be clear in the proof of Theorem 3, the bound is established for the unnormalized (i) as well as normalized (ii) estimator.

Theorem 1. *Let Assumption 1 hold. Then for all $t \in \mathbb{N}$, $(f_t, \tilde{f}_t) \in \mathsf{F}(\mathcal{X})^2$, and $\tilde{N} \in \mathbb{N}^*$ there exist constants $(c_t, \tilde{c}_t) \in (\mathbb{R}_+^*)^2$ (depending on h_t, $\tilde{N}$, f_t, and $\tilde{f}_t$) such that for all $N \in \mathbb{N}^*$ and all $\varepsilon \in \mathbb{R}_+^*$,*

$$(i)\ \mathbb{P}\left(\left|\frac{1}{N}\sum_{i=1}^N \omega_t^i\{\tau_t^i f_t(\xi_t^i) + \tilde{f}_t(\xi_t^i)\} - \phi_{t-1}\mathbf{L}_{t-1}(\mathbf{T}_t h_t f_t + \tilde{f}_t)\right| \geq \varepsilon\right) \leq c_t \exp(-\tilde{c}_t N\varepsilon^2),$$

$$(ii)\ \mathbb{P}\left(\left|\sum_{i=1}^N \frac{\omega_t^i}{\Omega_t}\{\tau_t^i f_t(\xi_t^i) + \tilde{f}_t(\xi_t^i) - \phi_t(\mathbf{T}_t h_t f_t + \tilde{f}_t)\}\right| \geq \varepsilon\right) \leq c_t \exp(-\tilde{c}_t N\varepsilon^2)$$

(with the convention $\phi_{-1} \equiv \chi$).

The following is an immediate consequence of Theorem 1.

Corollary 2. *Let Assumption 1 hold. Then for all $t \in \mathbb{N}$ and $\tilde{N} \in \mathbb{N}^*$ there exist constants $(c_t, \tilde{c}_t) \in (\mathbb{R}_+^*)^2$ (depending on h_t and $\tilde{N}$) such that for all $N \in \mathbb{N}^*$ and all $\varepsilon \in \mathbb{R}_+^*$,*

$$\mathbb{P}\left(\left|\sum_{i=1}^N \frac{\omega_t^i}{\Omega_t}(\tau_t^i - \phi_t \mathbf{T}_t h_t)\right| \geq \varepsilon\right) \leq c_t \exp(-\tilde{c}_t N\varepsilon^2).$$

3.2.2. Central limit theorems and asymptotic L_p error

Theorem 3. *Let Assumption 1 hold. Then for all $t \in \mathbb{N}$, $(f_t, \tilde{f}_t) \in \mathsf{F}(\mathcal{X})^2$, and $\tilde{N} \in \mathbb{N}^*$, as $N \to \infty$,*

$$\sqrt{N}\sum_{i=1}^N \frac{\omega_t^i}{\Omega_t}\{\tau_t^i f_t(\xi_t^i) + \tilde{f}_t(\xi_t^i) - \phi_t(\mathbf{T}_t h_t f_t + \tilde{f}_t)\} \xrightarrow{\mathcal{D}} \sigma_t\langle f_t, \tilde{f}_t\rangle(h) Z,$$

where Z has standard Gaussian distribution and

$$\begin{aligned}\sigma_t^2\langle f_t, \tilde{f}_t\rangle(h_t) &:= \tilde{\sigma}_t^2\langle f_t, \tilde{f}_t\rangle(h_t) \\ &+ \sum_{s=0}^{t-1}\sum_{\ell=0}^{s} \tilde{N}^{\ell-(s+1)} \frac{\phi_\ell \mathbf{L}_\ell\{\overleftarrow{\mathbf{Q}}_{\phi_\ell}(\mathbf{T}_\ell h_\ell + \tilde{h}_\ell - \mathbf{T}_{\ell+1}h_{\ell+1})^2 \mathbf{L}_{\ell+1}\cdots\mathbf{L}_s(g_{s+1}\{\mathbf{L}_{s+1}\cdots\mathbf{L}_{t-1}f_t\}^2)\}}{(\phi_\ell\mathbf{L}_\ell\cdots\mathbf{L}_{s-1}\mathbb{1}_{\mathsf{X}})(\phi_s\mathbf{L}_s\cdots\mathbf{L}_{t-1}\mathbb{1}_{\mathsf{X}})^2}\end{aligned} \tag{3.2}$$

with

$$\tilde{\sigma}_t^2\langle f_t, \tilde{f}_t\rangle(h_t) := \sum_{s=0}^{t-1} \frac{\phi_s \mathbf{L}_s\{g_{s+1}\tilde{\mathbf{D}}_{s+1,t}^2(h_t f_t + \tilde{f}_t)\}}{(\phi_s\mathbf{L}_s\cdots\mathbf{L}_{t-1}\mathbb{1}_{\mathsf{X}})^2}$$

being the asymptotic variance of the FFBSm algorithm (where, by convention, $\mathbf{L}_m\mathbf{L}_n = \mathrm{id}$ if $m > n$).

Remark 4. *Since for all $s \in [\![0, t-1]\!]$ and $\ell \in [\![0, s]\!]$, $\ell - (s+1) \leq -1$, it holds, in (3.2), that*

$$\lim_{\tilde{N}\to\infty} \sigma_t^2\langle f_t, \tilde{f}_t\rangle(h_t) = \tilde{\sigma}_t^2\langle f_t, \tilde{f}_t\rangle(h_t),$$

i.e., for large $\tilde{N}$ the asymptotic variance of PaRIS tends to that of the FFBSm algorithm. This is in line with our expectations, as the forward-only version of FFBSm can be viewed as a Rao-Blackwellization of PaRIS.

Again, the following is an immediate consequence of Theorem 3.

Corollary 5. *Let Assumption 1 hold. Then for all $t \in \mathbb{N}$ and $\tilde{N} \in \mathbb{N}^*$, as $N \to \infty$,*

$$\sqrt{N} \sum_{i=1}^{N} \frac{\omega_t^i}{\Omega_t} (\tau_t^i - \phi_t \mathbf{T}_t h_t) \xrightarrow{\mathcal{D}} \sigma_t(h_t) Z,$$

where Z has standard Gaussian distribution and

$$\sigma_t^2(h_t) := \tilde{\sigma}_t^2(h_t) + \sum_{s=0}^{t-1} \sum_{\ell=0}^{s} \tilde{N}^{\ell-(s+1)} \frac{\phi_\ell \mathbf{L}_\ell \{\overleftarrow{\mathbf{Q}}_{\phi_\ell} (\mathbf{T}_\ell h_\ell + \tilde{h}_\ell - \mathbf{T}_{\ell+1} h_{\ell+1})^2 \mathbf{L}_{\ell+1} \cdots \mathbf{L}_s (g_{s+1} \{\mathbf{L}_{s+1} \cdots \mathbf{L}_{t-1} \mathbb{1}_{\mathsf{X}}\}^2\}}{(\phi_\ell \mathbf{L}_\ell \cdots \mathbf{L}_{s-1} \mathbb{1}_{\mathsf{X}})(\phi_s \mathbf{L}_s \cdots \mathbf{L}_{t-1} \mathbb{1}_{\mathsf{X}})^2}. \tag{3.3}$$

with

$$\tilde{\sigma}_t^2(h_t) := \sum_{s=0}^{t-1} \frac{\phi_s \mathbf{L}_s (g_{s+1} \tilde{\mathbf{D}}_{s+1,t}^2 h_t)}{(\phi_s \mathbf{L}_s \cdots \mathbf{L}_{t-1} \mathbb{1}_{\mathsf{X}})^2}$$

being the asymptotic variance of the FFBSm algorithm.

By following identically the lines of the proof of [13, Theorem 8], we may use Corollary 2 and Corollary 5 for deriving also the asymptotic L_p error of the estimates produced by the algorithm.

Corollary 6. *Let Assumption 1 hold. Then for all $p \in \mathbb{R}_+^*$, $t \in \mathbb{N}$, and $\tilde{N} \in \mathbb{N}^*$,*

$$\lim_{N \to \infty} \sqrt{N} \left\| \sum_{i=1}^{N} \frac{\omega_t^i}{\Omega_t} (\tau_t^i - \phi_t \mathbf{T}_t h_t) \right\|_{\mathsf{L}_p} = \sqrt{2} \sigma_t(h_t) \left(\frac{\Gamma\{(p+1)/2\}}{\sqrt{2\pi}} \right)^{1/p},$$

where $\sigma_t^2(h_t)$ is given in (3.3).

3.2.3. Time uniform asymptotic variance bounds

In the present section we establish the long-term numerical stability of the PaRIS algorithm by bounding the asymptotic variance (3.3) (and hence, by Corollary 6, the asymptotic L_p error) using mixing-based arguments. We will treat separately joint smoothing and marginal smoothing, and derive, for precision parameters $\tilde{N} \geq 2$, $\mathcal{O}(t)$ and $\mathcal{O}(1)$ bounds, respectively, on the asymptotic variances in these cases. Since such time dependence is the best possible for SMC error bounds on the path and marginal spaces, these results confirm the conjecture that the algorithm stays numerically stable for precision parameters of this sort. Similar results for the FFBSm and FFBSi algorithms were obtained in [11, 18]. The analysis will be carried through under the following *strong mixing* assumption, which is standard in the literature of SMC analysis (see [10] and, e.g., [8, 5, 7, 13] for refinements) and points to applications where the state space X is a compact set.

Assumption 2.

(i) There exist constants $0 < \underline{\varepsilon} < \bar{\varepsilon} < \infty$ such that for all $(x, \tilde{x}) \in \mathsf{X}^2$,

$$\underline{\varepsilon} \leq q(x, \tilde{x}) \leq \bar{\varepsilon}.$$

(ii) There exist constants $0 < \underline{\delta} < \bar{\delta} < \infty$ such that for all $t \in \mathbb{N}$, $\|g_t\|_\infty \leq \bar{\delta}$ and $\underline{\delta} \leq \mathbf{L}_t \mathbb{1}_{\mathsf{X}}(x)$, $x \in \mathsf{X}$.

Joint smoothing

The following assumption implies that the additive functional under consideration grows at most linearly with time, which is a minimal requirement for obtaining an $\mathcal{O}(t)$ asymptotic variance.

Assumption 3. *There exists* $|\tilde{h}|_\infty \in \mathbb{R}_+^*$ *such that for all* $s \in \mathbb{N}$, $\operatorname{osc}(\tilde{h}_s) \leq |\tilde{h}|_\infty$.

As an auxiliary result, we provide an $\mathcal{O}(t)$ bound on the asymptotic variance of the FFBSm algorithm; see [18] for a similar result on the L_p error for finite particle sample sizes.

Proposition 7. *Let Assumption 2 and Assumption 3 hold. Then*

$$\limsup_{t\to\infty} \frac{1}{t}\tilde{\sigma}_t^2(h_t) \leq |\tilde{h}|_\infty^2 \frac{4\bar{\delta}}{\underline{\delta}(1-\varrho)^4}.$$

In the light of (7) it suffices to bound the second term of (3.3) by a quantity of order $\mathcal{O}(t)$. This yields the following result, where, interestingly, the incremental asymptotic variance caused by the backward simulation is *inversely proportional to the precision parameter* $\tilde{N}$. This is well in line with the theory of *random weight SMC methods*, in which, in similarity to our algorithm, intractable quantities (the importance weights) are replaced by random and unbiased estimates of the same (see [32, 31]).

Theorem 8. *Let Assumption 2 and Assumption 3 hold. Then for all* $\tilde{N} \geq 2$,

$$\limsup_{t\to\infty} \frac{1}{t}\sigma_t^2(h_t) \leq |\tilde{h}|_\infty^2 \frac{\bar{\delta}}{\underline{\delta}(1-\varrho)^4}\left(4 + \frac{(4\bar{\delta}\varrho^2 + 1 - \varrho)^2}{(\tilde{N}-1)(1-\varrho)}\right),$$

where $\sigma_t^2(h_t)$ *is defined in* (3.3).

Marginal smoothing

We turn to marginal smoothing, i.e., the situation when all terms of the additive functional are zero but a single one. For such a particular objective function we able to construct a time uniform bound on the *unnormalized* asymptotic variance of the same form as before, with one term representing the FFBSm asymptotic variance (see [11, Theorem 12]) and one additional term being inversely proportional to the precision parameter and representing the loss of accuracy introduced by backward sampling.

Assumption 4. *The additive functional has the following form. For some* $\hat{s} \in \mathbb{N}$,

$$\tilde{h}_s(x_{s:s+1}) = \begin{cases} 0 & \text{for } s \neq \hat{s}, \\ \tilde{h}_{\hat{s}}(x_{\hat{s}}) & \text{for } s = \hat{s}. \end{cases}$$

Theorem 9. *Let Assumption 2 and Assumption 4 hold. Then for all* $t \in \mathbb{N}$ *and* $\tilde{N} \geq 2$,

$$\sigma_t^2(h_t) \leq \operatorname{osc}^2(\tilde{h}_{\hat{s}})\frac{\bar{\delta}}{(1-\varrho)^3}\left(\bar{\delta}\frac{1+\varrho^2}{1+\varrho} + 4\frac{1}{\underline{\delta}(\tilde{N}-1)\{(1-\varrho^2)\wedge(1/2)\}}\right),$$

where $\sigma_t^2(h_t)$ *is defined in* (3.3).

3.2.4. Computational complexity

We conclude this section with some comments on the complexity of the algorithm. Under Assumption 1(ii), we may cast the accept-reject technique proposed in [11, Algorithm 1] into the framework of PaRIS. A pseudo-code describing the resulting scheme is provided by Algorithm 3 in Section B.2. For a given $t \in \mathbb{N}$, we denote by $C_t\langle N, \tilde{N}\rangle$ the (random) number of elementary operations needed for executing the PaRIS algorithm parameterized by $(N, \tilde{N}) \in (\mathbb{N}^*)^2$ from time zero to time t. Note that $C_t\langle N, \tilde{N}\rangle$ is strongly data dependent, as the observations $\{y_s\}_{s=0}^t$ effect, via the particle weights, the acceptance probabilities at the different time steps. Still, under the strong mixing assumption above it is possible to bound uniformly this random variable. The following result is an immediate consequence of [11, Proposition 2].

Theorem 10. *Let Assumption 2 hold. Then there exists a constant $c \in \mathbb{R}_+^*$ such that $\mathbb{E}[C_t\langle N, \tilde{N}\rangle] \leq ctN\tilde{N}/(1-\varrho)$ for all $t \in \mathbb{N}$.*

Thus, the expected number of trials grows linearly with time, the number of particles, and the precision parameter, showing the importance of keeping the latter at a minimum. On the other hand, since the variance bound derived in Proposition 7 (and Theorem 9) is inversely proportional to $\tilde{N}$, using an excessively large precision parameter will not pay off in terms of variance reduction (as the variance term controlled by the precision parameter will be negligible beside the variance corresponding to FFBSm). *We hence advocate keeping $\tilde{N}$ at a highly moderate value*, and will return to this matter in connection to the numerical illustrations of the next section.

4. Simulations

An exhaustive study of the numerical aspects of PaRIS is beyond the scope of the present paper; nevertheless, we benchmark the algorithm on two different models, namely

- a linear Gaussian state-space model (for which all quantities of interest can be computed exactly for comparison) and
- a stochastic volatility model [22].

4.1. Linear Gaussian state-space model

We first consider the linear Gaussian state-space model

$$\begin{aligned} X_{t+1} &= aX_t + \sigma_\varepsilon \varepsilon_{t+1} \\ Y_t &= bX_t + \sigma_\zeta \zeta_t \end{aligned} \quad (t \in \mathbb{N}), \tag{4.1}$$

where $\mathsf{Y} = \mathsf{X} = \mathbb{R}$ and $\{\varepsilon_t\}_{t\in\mathbb{N}^*}$ and $\{\zeta_t\}_{t\in\mathbb{N}}$ are sequences of mutually independent standard normally distributed random variables. The parameters $(a, b) \in \mathbb{R}^2$ and $(\sigma_\varepsilon, \sigma_\zeta) \in (\mathbb{R}_+^*)^2$ are considered to be known. We aim at computing smoothed expectations of the sufficient statistics

$$h_t^{(1)}(x_{0:t}) := \sum_{s=0}^t x_s, \quad h_t^{(2)}(x_{0:t}) := \sum_{s=0}^t x_s^2, \quad h_t^{(3)}(x_{0:t}) = \sum_{s=0}^{t-1} x_s x_{s+1} \quad (x_{0:t} \in \mathsf{X}^{t+1}) \tag{4.2}$$

under the dynamics governed by the parameter vector $(a, b, \sigma_\varepsilon, \sigma_\zeta) = (.7, 1, .2, 1)$, and assume for simplicity that the model is well-specified. For this model, the *disturbance smoother* (see, e.g., [5,

Algorithm 5.2.15]) provides the exact values of the smoothed sufficient statistics, and we compared these values with approximations obtained using PaRIS as well as the forward-only implementation of FFBSm. With our implementation, parameterizing PaRIS and FFBSm with $(N, \tilde{N}) = (150, 2)$ and $N = 50$, respectively, resulted in very similar computational times for the two algorithms, with PaRIS being slightly faster (recall that FFBSm has a quadratic complexity). As clear from the box plots (based on *time-normalized* estimates) displayed in Figure 2, PaRIS outperforms clearly FFBSm as the former exhibits lower variance as well as smaller bias for equal computational time.

As a measure of numerical performance, we define *efficiency* as inverse sample variance over computational time. Figure 3 reports the efficiencies by which the PaRIS and forward-only FFBSm algorithms estimate $\phi_{0:t|t} h_t^{(1)}$ using each $N = 500$ particles. As evident from the plot, PaRIS exhibits a higher efficiency uniformly over all time points. The variance estimates were based on 50 replicates.

In order to examine the dependence of the performance of PaRIS on the design of the precision parameter $\tilde{N}$, we produced estimates of $\phi_{0:t|t} h_t^{(1)}$ for $t \in [\![0, 1000]\!]$ using the algorithm for each of the precision parameters $\tilde{N} \in \{1, 2, 3, 4, 10, 30\}$. All these estimators were computed on the basis of the same forward particles, so also an additional FFBSm-based estimator. This experiment was, in order to estimate the variances of the (seven) different estimators, replicated 100 times for the same fixed sequence of observations. Figure 4, displaying estimated variance as a function of time, shows a momentous difference between the cases $\tilde{N} = 1$ and $\tilde{N} > 1$ (note the difference in y-axis scale between the two graphs); the graphs in the top ($\tilde{N} = 1$) and bottom ($\tilde{N} > 1$) figures exhibit variance growths that appear to be close to quadratic and linear, respectively, which is well in accordance with the theory. Increasing the precision parameter $\tilde{N}$ from 2 to 4 implies some decrease of variance, while increasing the same from 4 to 30 has only marginal effect on the accuracy of the estimator (the difference between the variances corresponding to $\tilde{N} = 10$ and $\tilde{N} = 30$ is close to indistinguishable). This is perfectly in line with the theoretical results obtained in Section 3, where the second term of the variance bound in Theorem 8 is inversely proportional to the precision parameter. Finally, ratios of variances of estimators associated with different $\tilde{N}$ are displayed in Figure 5, which shows a linearly increasing ratio of the variances associated with $\tilde{N} = 1$ and $\tilde{N} = 2$ and a close to constant ratio of the variances associated with $\tilde{N} = 2$ and $\tilde{N} = 3$.

Finally, in order to illustrate our algorithm's capacity of coping with particle path degeneracy, we report, in Figure 6, the ratios $\#\mathsf{S}_t / N^{t+1}$, $t \in [\![0, 1000]\!]$, where $\#\mathsf{S}_t$ is the cardinality of the support of the PaRIS algorithm at time t (in the notation of Section 3.1), for the precision parameters $\tilde{N} \in \{1, 2, 3, 10, 30\}$. Here $N = 100$, and again the estimators associated with different precision parameters were based on the same forward particles. The 95% confidence bounds displayed the same plot were obtained on the basis of 100 replicates of observation record. Judging by these confidence bounds, the dependence of the copiousness of the support on the observations is fairly robust. Interestingly, for $\tilde{N} = 1$ the sequence of ratios tends quickly to zero, while letting $\tilde{N} > 1$ stabilizes completely the support of the estimator. Already $\tilde{N} = 2$ yields a support that involves, on the average and in the long run, more than 50% of all forward particles. Again, increasing the precision parameter has some effect for moderate values of the same, say, up to $\tilde{N} = 10$, while increasing the parameter further from 10 to 30 (which implies a significant increase of computational overhead) effects only marginally the cardinality of the support. Also this observation is perfectly in line with the theory presented in Section 3, consolidating our apprehension that only a modest value of $\tilde{N}$ is required as long as $\tilde{N} \geq 2$.

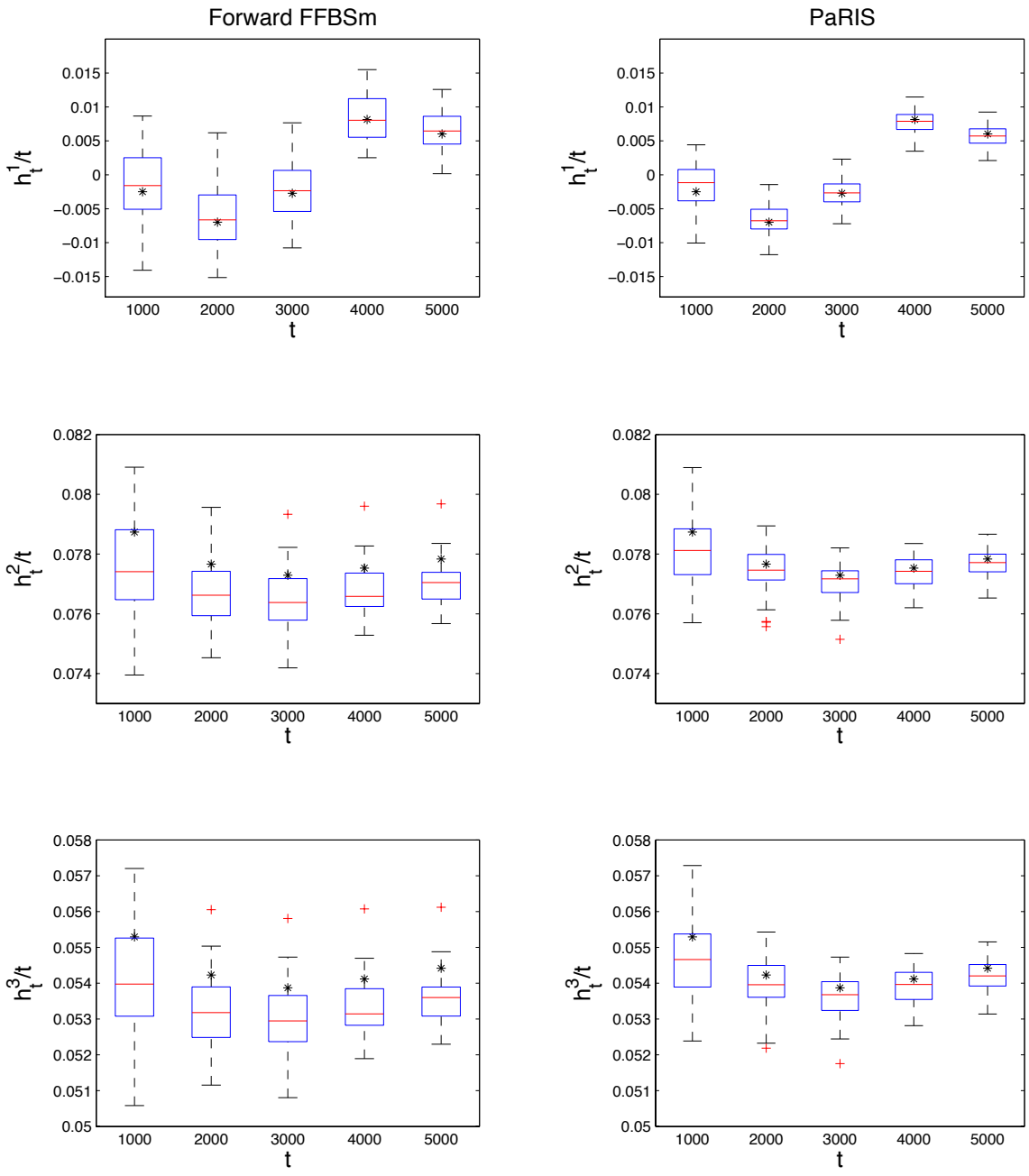


FIG 2. *Box plots of estimates of smoothed sufficient statistics* (4.2) *for the linear Gaussian model* (4.1) *produced by PaRIS (right column) and the forward-only version of FFBSm (left column) using* $(N, \tilde{N}) = (150, 2)$ *and* $N = 50$*, respectively (yielding close to identical computational times). The boxes are based on 50 replicates of the estimates for the same fixed observation sequence and asterisks indicate exact values obtained with the disturbance smoother.*

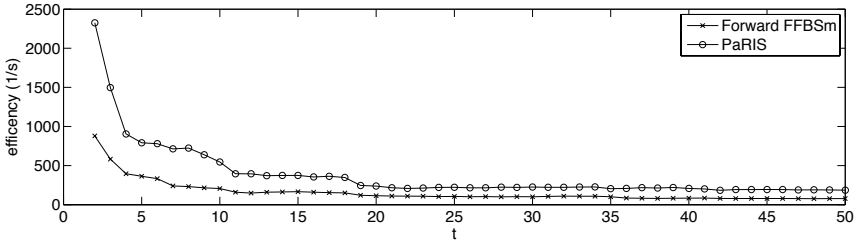


FIG 3. *Estimated efficiencies for the PaRIS and forward-only FFBSm algorithms using each $N = 500$ particles. (The first time step is removed from the plot due to very high efficiencies for both algorithms.)*

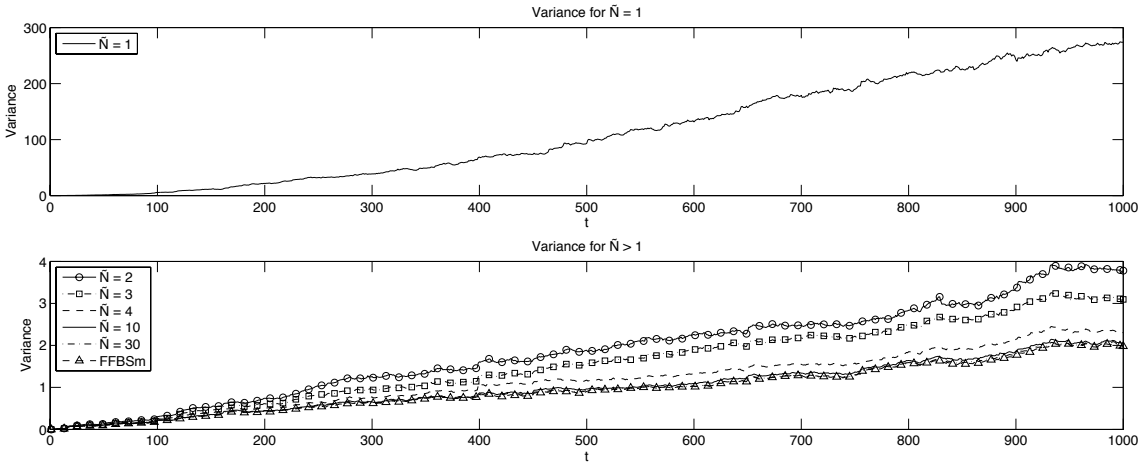


FIG 4. *Estimated variances of PaRIS estimators for $\tilde{N} = 1$ (top graph) and $\tilde{N} \in \{2, 3, 4, 10, 30\}$ (bottom graph) at different time steps $t \in [\![0, 1000]\!]$. The bottom graph includes variance estimates of the forward-only FFBSm estimator. The variance estimates are based on 100 replicates.*

4.2. Stochastic volatility model

For the sake of completeness we also consider a nonlinear model, namely the standard stochastic volatility model

$$\begin{aligned} X_{t+1} &= \phi X_t + \sigma_\varepsilon \varepsilon_{t+1} \\ Y_t &= \beta \exp(X_t/2)\zeta_t \end{aligned} \quad (t \in \mathbb{N}), \tag{4.3}$$

where $\mathsf{X} = \mathsf{Y} = \mathbb{R}$ and $\{\varepsilon_t\}_{t\in\mathbb{N}^*}$ and $\{\zeta_t\}_{t\in\mathbb{N}}$ are as in the previous example. We assume that the model parameters $\phi \in \mathbb{R}$ and $(\sigma, \beta) \in (\mathbb{R}_+^*)^2$ are known and that the model is well-specified. Our aim is to compute, using again PaRIS and the forward-only implementation of FFBSm, smoothed expectations of the sufficient statistics

$$h_t^{(1)}(x_{0:t}) := \sum_{s=0}^{t} x_s^2, \quad h_t^{(2)}(x_{0:t}) := \sum_{s=0}^{s-1} x_s x_{s+1} \quad (x_{0:t} \in \mathsf{X}^{t+1}) \tag{4.4}$$

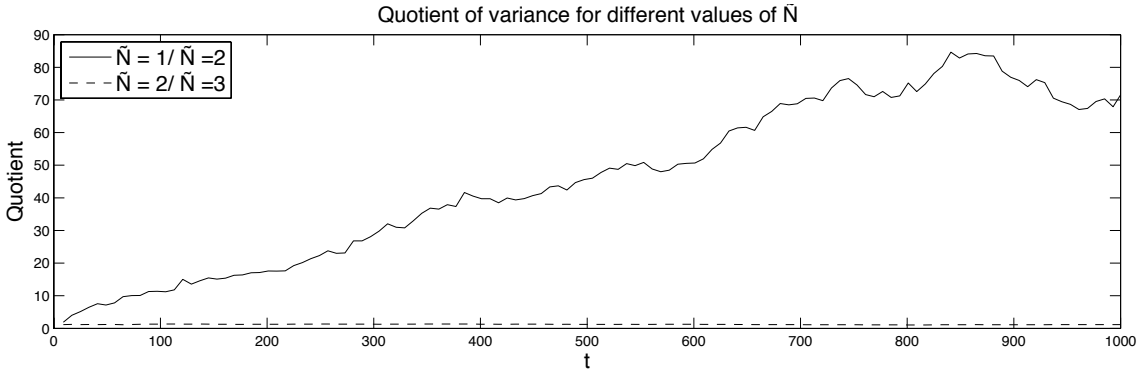


FIG 5. *Ratios of variances of the estimators associated with* $\tilde{N} = 1$ *and* $\tilde{N} = 2$ *(solid line) and* $\tilde{N} = 2$ *and* $\tilde{N} = 3$ *(dashed line).*

for a model parameterized by $(\phi, \sigma, \beta) = (.975, .16, .63)$. In this case, both algorithms used $N = 250$ particles and the precision parameter of PaRIS was set to $\tilde{N} = 2$. Figure 7 shows box plots based on 100 replicates of estimates of $\phi_{0:t|t} h_t^{(i)}/t$, for $i \in \{1, 2\}$ and $t \in \{2{,}000, 4{,}000, 6{,}000, 8{,}000, 10{,}000\}$, obtained using these methods. Even though the variance and the bias of the estimates produced by the two algorithms are comparable, PaRIS was now *5 times faster* than the FFBSm algorithm.

4.3. Some comments on the implementation

When applying accept-reject-based backward sampling (Algorithm 3), some acceptance probabilities will be small due to the random support of the particle-based backward kernel. In order to avoid getting stuck, it may be convenient to equip the algorithm with a threshold for the number of trials used at each accept-reject operation; when the threshold is reached, accept-reject sampling is cancelled and replaced by a draw from original distribution (recall that we are just using accept-reject sampling in order to reduce the computational work). Figure 8 displays computational time as a function of the size of this threshold for the linear Gaussian model and $N = 250$ particles. Interestingly, the graph has a minimum for the threshold value 14, and using this value we run the algorithm and counted the number of trials at any accept-reject sampling operation. The outcome is presented in the histogram plot to the right, from which it is clear that the majority of the particles are accepted after just a few trials (moreover, an index is most commonly accepted at once). In addition, at only 3.55% of the occasions, the number of trials exceeded the threshold. Needless to say, the optimal threshold depends on the model as well as the number of particles (when the number of particles is small, a too high threshold may have significant negative effect on the computational efficiency; on the contrary, when the number of particles is large, the performance of the algorithm is relatively robust vis-à-vis the design of the threshold). Further simulations not presented here indicate however that a threshold value around $\sqrt{N}$ could be a rule of thumb.

5. Conclusions

We have presented a novel algorithm, the particle-based, rapid incremental smoother, PaRIS, for computationally efficient online smoothing of additive state functionals in general HMMs. The

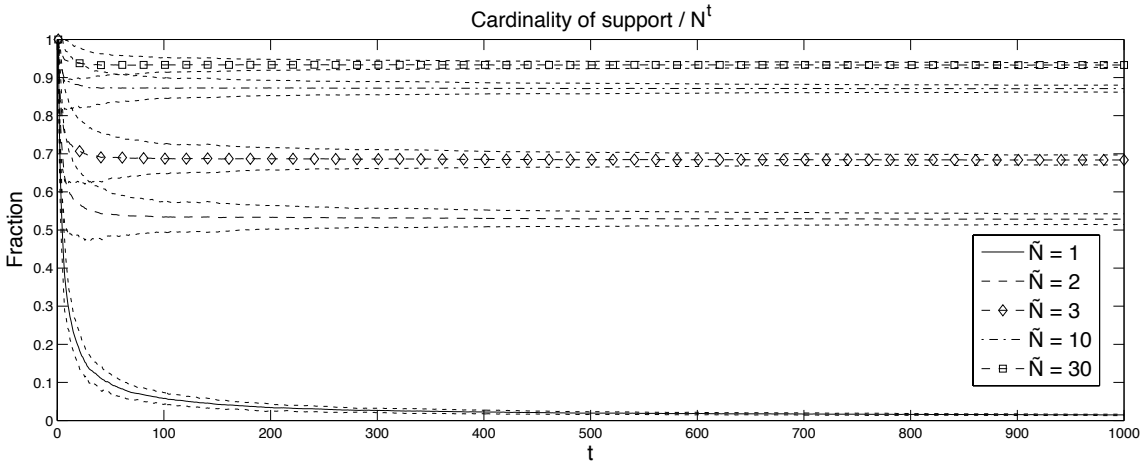


FIG 6. *Plot the ratios* $\#\mathsf{S}_t/N^{t+1}$, $t \in [\![0, 1000]\!]$, *for the precision parameters* $\tilde{N} \in \{1, 2, 3, 10, 30\}$ *and* $N = 100$. *The* 95% *confidence bounds are obtained on the basis of* 100 *replicates of the observation record.*

algorithm, which is based on a backward decomposition of the smoothing distribution which can be implemented recursively for objective functions of additive type, can be viewed as a hybrid between the forward-only implementation of FFBSm and the FFBSi algorithm; more specifically, forward-only FFBSm may be viewed as a Rao-Blackwellized version of PaRIS. The algorithm is furnished with a number of convergence results, where the main result is a CLT of PaRIS's Monte Carlo output at the rate $\sqrt{N}$. The analysis of PaRIS is considerably more involved than that of the FFBSi algorithm due to the complex dependence structure introduced by the retrospective simulation (on the contrary to FFBSi, where the trajectories are conditionally independent given the particles generated in the forward pass). Interestingly, the design of the precision parameter, i.e., the number of Monte Carlo simulations used for approximating the backward decomposition, turns out to be critical, since using a single backward draw yields a degeneracy phenomenon that resembles closely that of the Poor man's smoother. However, as established theoretically as well as through simulations, using at lest *two* such draws stabilizes completely the support of the estimator. For $\tilde{N} \geq 2$ we are able to derive $\mathcal{O}(1+1/(\tilde{N}-1))$ and $\mathcal{O}(t\{1+1/(\tilde{N}-1)\})$ bounds on the asymptotic variance in the cases of marginal and joint smoothing, respectively, and since the second term of these bounds is inversely proportional to the precision parameter, we suggest this parameter to be kept at a moderate value in order to gain computational speed. As known to the authors, this is the first analysis ever of this kind.

The algorithm we propose has a linear complexity in the number of particles while the forward-only implementation of FFBSm has a quadratic complexity, and a numerical comparison between the two shows clearly that PaRIS achieves the same accuracy as FFBSm at a considerably lower computational cost. In addition, similarly to forward-only FFBSm, our smoother has limited and constant memory requirements, as it needs only the current particle sample and a set of estimated auxiliary statistics to be stored at each iteration.

Smoothing of additive state functionals is a key ingredient of most—frequentistic or Bayesian—online parameter estimation techniques for HMMs. Since these applications are most often characterized by strict computational requirements, PaRIS can be naturally cast into any such framework.

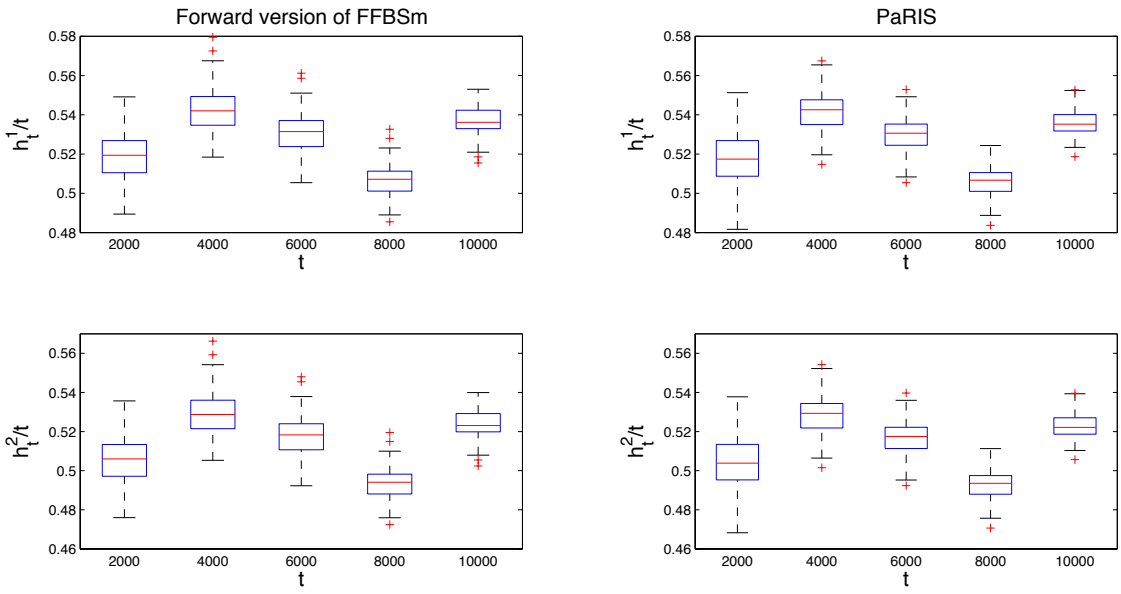


FIG 7. *Box plots of estimates of smoothed sufficient statistics* (4.4) *for the stochastic volatility model* (4.3) *produced by PaRIS (right column) and the forward-only version of FFBSm (left column) using* $(N, \tilde{N}) = (250, 2)$ *and* $N = 250$*, respectively. With this parameterization, PaRIS was 5 times faster than the FFBSm algorithm. The boxes are based on 100 replicates of the estimates for the same fixed observation sequence.*

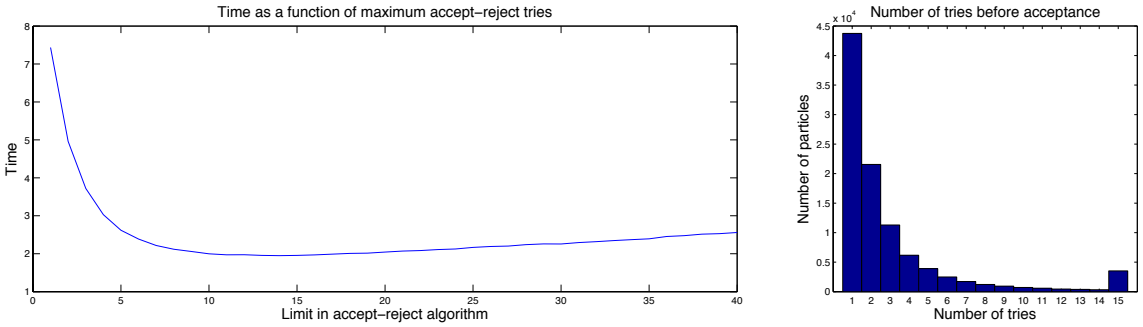


FIG 8. *Computational time as a function of the size of the accept-reject threshold for the linear Gaussian model and* $N = 250$ *particles (left panel). The histogram to the right displays the number of trials needed before acceptance at any accept-reject sampling operation in algorithm when the threshold is* 14 *(corresponding to minimal computational time); the bar at* 15 *represents* 3.55% *of the occasions.*

Appendix A: Proofs

A.1. Two prefatory lemmas

Lemma 11. *For all $t \in \mathbb{N}$ and $(f_{t+1}, \tilde{f}_{t+1}) \in \mathsf{F}(\mathcal{X})^2$ it holds that*

$$\phi_{t+1}(\mathbf{T}_{t+1}h_{t+1}f_{t+1} + \tilde{f}_{t+1}) = \frac{\phi_t\{\mathbf{T}_t h_t \mathbf{L}_t f_{t+1} + \mathbf{L}_t(\tilde{h}_t f_{t+1} + \tilde{f}_{t+1})\}}{\phi_t \mathbf{L}_t \mathbb{1}_{\mathsf{X}}}.$$

Proof. By combining the definitions of $\mathbf{T}_t$, $\mathbf{L}_t$ and using reversibility,

$$\begin{aligned}
\phi_t \mathbf{L}_t(\mathbf{T}_{t+1}h_{t+1}f_{t+1}) &= \phi_t \mathbf{Q}\{\overleftarrow{\mathbf{Q}}_{\phi_t}(\mathbf{T}_t h_t + \tilde{h}_t) g_{t+1} f_{t+1}\} \\
&= \phi_t \mathbf{Q} \otimes \overleftarrow{\mathbf{Q}}_{\phi_t}\{(\mathbf{T}_t h_t + \tilde{h}_t) g_{t+1} f_{t+1}\} \\
&= \phi_t \otimes \mathbf{Q}\{(\mathbf{T}_t h_t + \tilde{h}_t) g_{t+1} f_{t+1}\} \\
&= \phi_t\{\mathbf{T}_t h_t \mathbf{L}_t f_{t+1} + \mathbf{L}_t(\tilde{h}_t f_{t+1})\}.
\end{aligned}$$

Now the statement of the lemma follows by dividing both sides of the previous equation by $\phi_t \mathbf{L}_t \mathbb{1}_{\mathsf{X}}$ and using the identity $\phi_{t+1} = \phi_t \mathbf{L}_t / \phi_t \mathbf{L}_t \mathbb{1}_{\mathsf{X}}$. □

Lemma 12. *For all $t \in \mathbb{N}$, $(f_{t+1}, \tilde{f}_{t+1}) \in \mathsf{F}(\mathcal{X})^2$, and $(N, \tilde{N}) \in (\mathbb{N}^*)^2$ the random variables $\{\omega_{t+1}^i(\tau_{t+1}^i f_{t+1}(\xi_{t+1}^i) + \tilde{f}_{t+1}(\xi_{t+1}^i))\}_{i=1}^N$ are, conditionally on $\tilde{\mathcal{F}}_{t-1}^N$, i.i.d. with expectation*

$$\mathbb{E}\left[\omega_{t+1}^1\{\tau_{t+1}^1 f_{t+1}(\xi_{t+1}^1) + \tilde{f}_{t+1}(\xi_{t+1}^1)\} \mid \tilde{\mathcal{F}}_t^N\right] = \sum_{i=1}^N \frac{\omega_t^i}{\Omega_t}\{\tau_t^i \mathbf{L}_t f_{t+1}(\xi_t^i) + \mathbf{L}_t(\tilde{h}_t f_{t+1} + \tilde{f}_{t+1})(\xi_t^i)\}. \quad \text{(A.1)}$$

Proof. The multinomial selection procedure implies that the particles $\{\xi_{t+1}^i\}_{i=1}^N$ are i.i.d. conditionally on $\tilde{\mathcal{F}}_t^N$. Hence, since also the backward indices $\{J_{t+1}^{(i,j)}\}_{j=1}^{\tilde{N}}$ are i.i.d. conditionally on the particle ξ_t^i and the σ-field $\tilde{\mathcal{F}}_t^N$, we conclude that $\{\omega_{t+1}^i(\tau_{t+1}^i f_{t+1}(\xi_{t+1}^i) + \tilde{f}_{t+1}(\xi_{t+1}^i))\}_{i=1}^N$ are i.i.d. conditionally on $\tilde{\mathcal{F}}_t^N$.

In order to compute the common conditional expectation we decompose the same according to

$$\begin{aligned}
&\mathbb{E}\left[\omega_{t+1}^1\{\tau_{t+1}^1 f_{t+1}(\xi_{t+1}^1) + \tilde{f}_{t+1}(\xi_{t+1}^1)\} \mid \tilde{\mathcal{F}}_t^N\right] \\
&\qquad = \mathbb{E}\left[\omega_{t+1}^1 \tau_{t+1}^1 f_{t+1}(\xi_{t+1}^1) \mid \tilde{\mathcal{F}}_t^N\right] + \mathbb{E}\left[\omega_{t+1}^1 \tilde{f}_{t+1}(\xi_{t+1}^1) \mid \tilde{\mathcal{F}}_t^N\right],
\end{aligned}$$

where, by the tower property,

$$\begin{aligned}
&\mathbb{E}\left[\omega_{t+1}^1 \tau_{t+1}^1 f_{t+1}(\xi_{t+1}^1) \mid \tilde{\mathcal{F}}_t^N\right] \\
&= \mathbb{E}\left[\omega_{t+1}^1 f_{t+1}(\xi_{t+1}^1) \mathbb{E}\left[\tau_{t+1}^1 \mid \tilde{\mathcal{F}}_t^N \vee \mathcal{F}_{t+1}^N\right] \mid \tilde{\mathcal{F}}_t^N\right] \\
&= \mathbb{E}\left[\omega_{t+1}^1 f_{t+1}(\xi_{t+1}^1) \sum_{\ell=1}^N \frac{\omega_t^\ell q(\xi_t^\ell, \xi_{t+1}^1)}{\sum_{\ell'=1}^N \omega_t^{\ell'} q(\xi_t^{\ell'}, \xi_{t+1}^1)}\{\tau_t^\ell + \tilde{h}_t(\xi_t^\ell, \xi_{t+1}^1)\} \mid \tilde{\mathcal{F}}_t^N\right] \\
&= \sum_{i=1}^N \frac{\omega_t^i}{\Omega_t} \int q(\xi_t^i, x) g_{t+1}(x) f_{t+1}(x) \sum_{\ell=1}^N \frac{\omega_t^\ell q(\xi_t^\ell, x)}{\sum_{\ell'=1}^N \omega_t^{\ell'} q(\xi_t^{\ell'}, x)}\{\tau_t^\ell + \tilde{h}_t(\xi_t^\ell, x)\}\, \mu(\mathrm{d}x) \\
&= \sum_{\ell=1}^N \frac{\omega_t^\ell}{\Omega_t}\{\tau_t^\ell \mathbf{L}_t f_{t+1}(\xi_t^\ell) + \mathbf{L}_t(\tilde{h}_t f_{t+1})(\xi_t^\ell)\}.
\end{aligned}$$

We conclude the proof by noting that

$$\mathbb{E}\left[\omega_{t+1}^1 \tilde{f}_{t+1}(\xi_{t+1}^1) \mid \tilde{\mathcal{F}}_t^N\right] = \sum_{\ell=1}^N \frac{\omega_t^\ell}{\Omega_t} \mathbf{Q}(g_{t+1}\tilde{f}_{t+1})(\xi_t^\ell) = \sum_{\ell=1}^N \frac{\omega_t^\ell}{\Omega_t} \mathbf{L}_t \tilde{f}_{t+1}(\xi_t^\ell).$$

□

A.2. Proof of Theorem 1

We proceed by induction and assume that the claim of the theorem holds for $t \in \mathbb{N}$. To establish (i) for $t+1$, write, using Lemma 11,

$$\begin{aligned}
&\frac{1}{N}\sum_{i=1}^N \omega_{t+1}^i \{\tau_{t+1}^i f_{t+1}(\xi_{t+1}^i) + \tilde{f}_{t+1}(\xi_{t+1}^i)\} - \phi_t \mathbf{L}_t(\mathbf{T}_{t+1} h_{t+1} f_{t+1} + \tilde{f}_{t+1}) \\
&\quad = \frac{1}{N}\sum_{i=1}^N \omega_{t+1}^i \{\tau_{t+1}^i f_{t+1}(\xi_{t+1}^i) + \tilde{f}_{t+1}(\xi_{t+1}^i)\} - \mathbb{E}\left[\omega_{t+1}^1 \{\tau_{t+1}^1 f_{t+1}(\xi_{t+1}^1) + \tilde{f}_{t+1}(\xi_{t+1}^1)\} \mid \tilde{\mathcal{F}}_t^N\right] \\
&\qquad + \sum_{i=1}^N \frac{\omega_t^i}{\Omega_t} \{\tau_t^i \mathbf{L}_t f_{t+1}(\xi_t^i) + \mathbf{L}_t(\tilde{h}_t f_{t+1} + \tilde{f}_{t+1})(\xi_t^i)\} - \phi_t\{\mathbf{T}_t h_t \mathbf{L}_t f_{t+1} + \mathbf{L}_t(\tilde{h}_t f_{t+1} + \tilde{f}_{t+1})\}.
\end{aligned}$$

Since the functions $\mathbf{L}_t f_{t+1}$ and $\mathbf{L}_t(\tilde{h}_t f_{t+1} + \tilde{f}_{t+1})$ belong to $\mathsf{F}(\mathcal{X})$, the induction hypothesis (ii) implies that for all $\varepsilon \in \mathbb{R}_+^*$,

$$\mathbb{P}\left(\left|\sum_{i=1}^N \frac{\omega_t^i}{\Omega_t} \{\tau_t^i \mathbf{L}_t f_{t+1}(\xi_t^i) + \mathbf{L}_t(\tilde{h}_t f_{t+1} + \tilde{f}_{t+1})(\xi_t^i)\} - \phi_t\{\mathbf{T}_t h_t \mathbf{L}_t f_{t+1} + \mathbf{L}_t(\tilde{h}_t f_{t+1} + \tilde{f}_{t+1})\}\right| \geq \varepsilon\right) \leq c_t \exp(-\tilde{c}_t N \varepsilon^2).$$

In addition, by Lemma 12, $\{\omega_{t+1}^i(\tau_{t+1}^i f_{t+1}(\xi_{t+1}^i) + \tilde{f}_{t+1}(\xi_{t+1}^i))\}_{i=1}^N$ are conditionally i.i.d. given $\tilde{\mathcal{F}}_t^N$; thus, since for all $i \in [\![1, N]\!]$,

$$|\omega_{t+1}^i \{\tau_{t+1}^i f_{t+1}(\xi_{t+1}^i) + \tilde{f}_{t+1}(\xi_{t+1}^i)\}| \leq \|g_{t+1}\|_\infty (\|h_{t+1}\|_\infty \|f_{t+1}\|_\infty + \|\tilde{f}_{t+1}\|_\infty) < \infty,$$

the conditional Hoeffding inequality provides constants $(d, \tilde{d}) \in (\mathbb{R}_+^*)^2$ such that

$$\mathbb{P}\left(\left|\frac{1}{N}\sum_{i=1}^N \omega_{t+1}^i \{\tau_{t+1}^i f_{t+1}(\xi_{t+1}^i) + \tilde{f}_{t+1}(\xi_{t+1}^i)\} - \mathbb{E}\left[\omega_{t+1}^1 \{\tau_{t+1}^1 f_{t+1}(\xi_{t+1}^1) + \tilde{f}_{t+1}(\xi_{t+1}^1)\} \mid \tilde{\mathcal{F}}_t^N\right]\right| \geq \varepsilon\right) \leq d \exp(-\tilde{d} N \varepsilon^2).$$

This establishes (i).

The inequality (ii) for the self-normalized estimator is an immediate consequence of (i) and the generalized Hoeffding inequality in [11, Lemma 4].

Finally, we conclude the proof by checking that the result is straightforwardly true for the base case $t = 0$, since $\mathbf{T}_0 h_0 = 0$, $\tau_0^i = 0$ for all $i \in [\![1, N]\!]$, and the weighted sample $\{(\xi_0^i, \omega_0^i)\}_{i=1}^N$ (targeting ϕ_0) is generated by standard importance sampling.

A.3. Proof of Theorem 3

We proceed by induction and suppose that the claim of the theorem holds true for some $t \in \mathbb{N}$. Thus, pick $(f_{t+1}, \tilde{f}_{t+1}) \in \mathsf{F}(\mathcal{X})^2$ and assume first that $\phi_{t+1}(\mathbf{T}_{t+1} h_{t+1} f_{t+1} + \tilde{f}_{t+1}) = 0$. Write

$$\begin{aligned}\sqrt{N}\sum_{i=1}^{N}\frac{\omega_t^i}{\Omega_t}\{\tau_{t+1}^i f_{t+1}(\xi_{t+1}^i) + \tilde{f}_{t+1}(\xi_{t+1}^i)\} &= N\Omega_t^{-1}\frac{1}{\sqrt{N}}\sum_{i=1}^{N}\Bigg(\omega_{t+1}^i\{\tau_{t+1}^i f_{t+1}(\xi_{t+1}^i) + \tilde{f}_{t+1}(\xi_{t+1}^i)\} \\ &\quad - \sum_{\ell=1}^{N}\frac{\omega_t^\ell}{\Omega_t}\{\tau_t^\ell \mathbf{L}_t f_{t+1}(\xi_{t+1}^\ell) + \mathbf{L}_t(\tilde{h}_t f_{t+1} + \tilde{f}_{t+1})(\xi_{t+1}^\ell)\}\Bigg) \\ &\quad + N\Omega_t^{-1}\sqrt{N}\sum_{\ell=1}^{N}\frac{\omega_t^\ell}{\Omega_t}\{\tau_t^\ell \mathbf{L}_t f_{t+1}(\xi_{t+1}^\ell) + \mathbf{L}_t(\tilde{h}_t f_{t+1} + \tilde{f}_{t+1})(\xi_{t+1}^\ell)\},\end{aligned}$$

where, by Theorem 1, $N\Omega_t^{-1}$ tends to $(\phi_t \mathbf{L}_t \mathbb{1}_{\mathsf{X}})^{-1}$ in probability. In order to establish the weak convergence of the first term, we will apply Theorem 16 to the triangular array

$$v_N^i := \frac{1}{\tilde{N}\sqrt{N}}\sum_{j=1}^{\tilde{N}}\tilde{v}_N(J_{t+1}^{(i,j)}, \xi_{t+1}^i) \quad (i \in [\![1, N]\!], N \in \mathbb{N}^*),$$

where

$$\begin{aligned}\tilde{v}_N(j, x) &:= g_{t+1}(x)\{(\tau_t^j + \tilde{h}_t(\xi_t^j, x))f_{t+1}(x) + \tilde{f}_{t+1}(x)\} \\ &\quad - \sum_{\ell=1}^{N}\frac{\omega_t^\ell}{\Omega_t}\{\tau_t^\ell \mathbf{L}_t f_{t+1}(\xi_t^\ell) + \mathbf{L}_t(\tilde{h}_t f_{t+1} + \tilde{f}_{t+1})(\xi_t^\ell)\} \quad (x \in \mathsf{X}, j \in [\![1, \tilde{N}]\!], N \in \mathbb{N}^*), \quad \text{(A.2)}\end{aligned}$$

furnished with the filtration $\{\tilde{\mathcal{F}}_t^N\}_{N \in \mathbb{N}^*}$. Note that for all $i \in [\![1, N]\!]$, $\mathbb{E}[v_N^i \mid \tilde{\mathcal{F}}_t^N] = 0$ (by Lemma 12) and $|v_N^i| \leq 2\|g_{t+1}\|_\infty(\|h_{t+1}\|_\infty\|f_{t+1}\|_\infty + \|\tilde{f}_{t+1}\|_\infty)/\sqrt{N}$. To check the condition (**B1**) in Theorem 16, write, using, first, that $\{v_N^i\}_{i=1}$ are conditionally i.i.d. given $\tilde{\mathcal{F}}_t^N$ and, second, that the backward indices $\{J_{t+1}^{(i,j)}\}_{j=1}^{\tilde{N}}$ are, for all $i \in [\![1, N]\!]$, i.i.d. conditionally on $\tilde{\mathcal{F}}_t^N$ and ξ_{t+1}^i,

$$\begin{aligned}\sum_{i=1}^{N}\mathbb{E}\left[(v_N^i)^2 \mid \tilde{\mathcal{F}}_t^N\right] &= \tilde{N}^{-2}\mathbb{E}\left[\left(\sum_{j=1}^{\tilde{N}}\tilde{v}_N(J_{t+1}^{(1,j)}, \xi_{t+1}^1)\right)^2 \mid \tilde{\mathcal{F}}_t^N\right] \\ &= \tilde{N}^{-1}\mathbb{E}\left[\mathbb{E}\left[\tilde{v}_N^2(J_{t+1}^{(1,1)}, \xi_{t+1}^1) \mid \tilde{\mathcal{F}}_t^N \vee \mathcal{F}_{t+1}^N\right] \mid \tilde{\mathcal{F}}_t^N\right] \quad \text{(A.3)} \\ &\quad + \tilde{N}^{-1}(\tilde{N} - 1)\mathbb{E}\left[\mathbb{E}^2\left[\tilde{v}_N(J_{t+1}^{(1,1)}, \xi_{t+1}^1) \mid \tilde{\mathcal{F}}_t^N \vee \mathcal{F}_{t+1}^N\right] \mid \tilde{\mathcal{F}}_t^N\right]. \quad \text{(A.4)}\end{aligned}$$

We treat separately the two terms (A.3) and (A.4). Concerning (A.3),

$$\begin{aligned}\mathbb{E}\left[\mathbb{E}\left[\tilde{v}_N^2(J_{t+1}^{(1,1)},\xi_{t+1}^1) \mid \tilde{\mathcal{F}}_t^N \vee \mathcal{F}_{t+1}^N\right] \mid \tilde{\mathcal{F}}_t^N\right] &= \mathbb{E}\left[\sum_{\ell=1}^N \tilde{v}_N^2(\ell,\xi_{t+1}^1)\frac{\omega_t^\ell q(\xi_t^\ell,\xi_{t+1}^1)}{\sum_{\ell'=1}^N \omega_t^{\ell'} q(\xi_t^{\ell'},\xi_{t+1}^1)} \mid \tilde{\mathcal{F}}_t^N\right] \\ &= \sum_{i=1}^N \frac{\omega_t^i}{\Omega_t}\int q(\xi_t^i,x)\sum_{\ell=1}^N \tilde{v}_N^2(\ell,x)\frac{\omega_t^\ell q(\xi_t^\ell,x)}{\sum_{\ell'=1}^N \omega_t^{\ell'} q(\xi_t^{\ell'},x)}\,\mu(\mathrm{d}x) \\ &= \sum_{\ell=1}^N \frac{\omega_t^\ell}{\Omega_t}\int q(\xi_t^\ell,x)\tilde{v}_N^2(\ell,x)\,\mu(\mathrm{d}x).\end{aligned}$$

Now, using the definition (A.2),

$$\begin{aligned}\sum_{\ell=1}^N \frac{\omega_t^\ell}{\Omega_t}\int q(\xi_t^\ell,x)\tilde{v}_N^2(\ell,x)\,\mu(\mathrm{d}x) &= \sum_{\ell=1}^N \frac{\omega_t^\ell}{\Omega_t}(\tau_t^\ell)^2\mathbf{L}_t(g_{t+1}f_{t+1}^2)(\xi_t^\ell) \\ &+ \sum_{\ell=1}^N \frac{\omega_t^\ell}{\Omega_t}\left(\tau_t^\ell 2\mathbf{L}_t\{g_{t+1}f_{t+1}(\tilde{h}_t f_{t+1}+\tilde{f}_{t+1})\}(\xi_t^\ell) + \mathbf{L}_t\{g_{t+1}(\tilde{h}_t f_{t+1}+\tilde{f}_{t+1})^2\}(\xi_t^\ell)\right) \\ &- \left(\sum_{\ell=1}^N \frac{\omega_t^\ell}{\Omega_t}\{\tau_t^\ell \mathbf{L}_t f_{t+1}(\xi_t^\ell) + \mathbf{L}_t(\tilde{h}_t f_{t+1}+\tilde{f}_{t+1})(\xi_t^\ell)\}\right)^2.\end{aligned}$$

In the previous expression, by Lemma 13,

$$\sum_{\ell=1}^N \frac{\omega_t^\ell}{\Omega_t}(\tau_t^\ell)^2\mathbf{L}_t(g_{t+1}f_{t+1}^2)(\xi_t^\ell) \xrightarrow{\mathbb{P}} \phi_t\{\mathbf{T}_t^2 h_t \mathbf{L}_t(g_{t+1}f_{t+1}^2)\} + \eta_t\{\mathbf{L}_t(g_{t+1}f_{t+1}^2)\},$$

where the functional η_t is defined in (A.9), and, by Corollary 2 and Lemma 11 (recalling that $\phi_t\mathbf{L}_t(\mathbf{T}_{t+1}f_{t+1}+\tilde{f}_{t+1})=0$ by assumption),

$$\begin{aligned}&\sum_{\ell=1}^N \frac{\omega_t^\ell}{\Omega_t}\left(\tau_t^\ell 2\mathbf{L}_t\{g_{t+1}f_{t+1}(\tilde{h}_t f_{t+1}+\tilde{f}_{t+1})\}(\xi_t^\ell) + \mathbf{L}_t\{g_{t+1}(\tilde{h}_t f_{t+1}+\tilde{f}_{t+1})^2\}(\xi_t^\ell)\right) \\ &\qquad \xrightarrow{\mathbb{P}} 2\phi_t(\mathbf{T}_t h_t \mathbf{L}_t\{g_{t+1}f_{t+1}(\tilde{h}_t f_{t+1}+\tilde{f}_{t+1})\}) + \phi_t\mathbf{L}_t\{g_{t+1}(\tilde{h}_t f_{t+1}+\tilde{f}_{t+1})^2\}, \\ &\sum_{\ell=1}^N \frac{\omega_t^\ell}{\Omega_t}\{\tau_t^\ell \mathbf{L}_t f_{t+1}(\xi_t^\ell) + \mathbf{L}_t(\tilde{h}_t f_{t+1}+\tilde{f}_{t+1})(\xi_t^\ell)\} \xrightarrow{\mathbb{P}} 0. \qquad \text{(A.5)}\end{aligned}$$

We hence conclude that

$$\begin{aligned}\text{(A.3)} \xrightarrow{\mathbb{P}} \tilde{N}^{-1}\Big(&\phi_t\{\mathbf{T}_t^2 h_t \mathbf{L}_t(g_{t+1}f_{t+1}^2)\} + \eta_t\{\mathbf{L}_t(g_{t+1}f_{t+1}^2)\} \\ &+2\phi_t(\mathbf{T}_t h_t \mathbf{L}_t\{g_{t+1}f_{t+1}(\tilde{h}_t f_{t+1}+\tilde{f}_{t+1})\}) + \phi_t\mathbf{L}_t\{g_{t+1}(\tilde{h}_t f_{t+1}+\tilde{f}_{t+1})^2\}\Big) \\ &= \tilde{N}^{-1}\left(\phi_t\mathbf{L}_t(g_{t+1}\{(\mathbf{T}_t h_t+\tilde{h}_t)f_{t+1}+\tilde{f}_{t+1}\}^2) + \eta_t\{\mathbf{L}_t(g_{t+1}f_{t+1}^2)\}\right). \qquad \text{(A.6)}\end{aligned}$$

We turn to (A.4) and write

$$\begin{aligned}
&\mathbb{E}\left[\mathbb{E}^2\left[\tilde{v}_N(J_{t+1}^{(1,1)}, \xi_{t+1}^1) \mid \tilde{\mathcal{F}}_t^N \vee \mathcal{F}_{t+1}^N\right] \mid \tilde{\mathcal{F}}_t^N\right] \\
&\qquad = \mathbb{E}\left[\left(\sum_{\ell=1}^N \tilde{v}_N(\ell, \xi_{t+1}^1) \frac{\omega_t^\ell q(\xi_t^\ell, \xi_{t+1}^1)}{\sum_{\ell'=1}^N \omega_t^{\ell'} q(\xi_t^{\ell'}, \xi_{t+1}^1)}\right)^2 \mid \tilde{\mathcal{F}}_t^N\right] \\
&\qquad = \sum_{i=1}^N \frac{\omega_t^i}{\Omega_t} \int q(\xi_t^i, x) \left(\sum_{\ell=1}^N \tilde{v}_N(\ell, x) \frac{\omega_t^\ell q(\xi_t^\ell, x)}{\sum_{\ell'=1}^N \omega_t^{\ell'} q(\xi_t^{\ell'}, x)}\right)^2 \mu(\mathrm{d}x),
\end{aligned}$$

where we note that the right hand side can, by (A.2), be written as $\phi_t^N \mathbf{Q} \varphi_N$, with

$$\begin{aligned}
\varphi_N(x) := \Bigg(& g_{t+1}(x) f_{t+1}(x) \sum_{\ell=1}^N \frac{\omega_t^\ell q(\xi_t^\ell, x)}{\sum_{\ell'=1}^N \omega_t^{\ell'} q(\xi_t^{\ell'}, x)} \{\tau_t^\ell + \tilde{h}_t(\xi_t^\ell, x)\} + g_{t+1}(x) \tilde{f}_{t+1}(x) \\
& - \sum_{\ell'=1}^N \frac{\omega_t^{\ell'}}{\Omega_t} \{\tau_t^{\ell'} \mathbf{L}_t f_{t+1}(\xi_t^{\ell'}) + \mathbf{L}_t(\tilde{h}_t f_{t+1} + \tilde{f}_{t+1})(\xi_t^{\ell'})\} \Bigg)^2 \qquad (x \in \mathsf{X}).
\end{aligned}$$

Note that $\|\varphi_N\|_\infty \le 4\|g_{t+1}\|_\infty^2 (\|h_{t+1}\|_\infty \|f_{t+1}\|_\infty + \|\tilde{f}_{t+1}\|_\infty)^2$ for all $N \in \mathbb{N}$. Moreover, by Corollary 2 (and, in particular, the implication (A.5)) it holds, for all $x \in \mathsf{X}$, $\mathbb{P}$-a.s.,

$$\begin{aligned}
\varphi_N(x) &\to g_{t+1}^2(x) \left(f_{t+1}(x) \frac{\int q(\tilde{x}, x) \{\mathbf{T}_t h_t(\tilde{x}) + \tilde{h}_t(\tilde{x}, x)\} \phi_t(\mathrm{d}\tilde{x})}{\int q(\tilde{x}, x) \phi_t(\mathrm{d}\tilde{x})} + \tilde{f}_{t+1}(x) \right)^2 \\
&= g_{t+1}^2(x) \{f_{t+1}(x) \overleftarrow{\mathbf{Q}}_{\phi_t}(\mathbf{T}_t h_t + \tilde{h}_t)(x) + \tilde{f}_{t+1}(x)\}^2 \\
&= g_{t+1}^2(x) \{f_{t+1}(x) \mathbf{T}_{t+1} h_{t+1}(x) + \tilde{f}_{t+1}(x)\}^2.
\end{aligned}$$

Thus, under Assumption 1 we may apply Lemma 14, yielding

$$\phi_t^N \mathbf{Q} \varphi_N \xrightarrow{\mathbb{P}} \phi_t \mathbf{L}_t \{g_{t+1}(f_{t+1} \mathbf{T}_{t+1} h_{t+1} + \tilde{f}_{t+1})^2\},$$

and we may hence conclude that

$$(\text{A.4}) \xrightarrow{\mathbb{P}} \tilde{N}^{-1}(\tilde{N} - 1) \phi_t \mathbf{L}_t \{g_{t+1}(f_{t+1} \mathbf{T}_{t+1} h_{t+1} + \tilde{f}_{t+1})^2\}.$$

Finally, by combining this limit with (A.6) we obtain, using the identity

$$\begin{aligned}
&\phi_t \mathbf{L}_t(g_{t+1}\{(\mathbf{T}_t h_t + \tilde{h}_t) f_{t+1} + \tilde{f}_{t+1}\}^2) - \phi_t \mathbf{L}_t \{g_{t+1}(f_{t+1} \mathbf{T}_{t+1} h_{t+1} + \tilde{f}_{t+1})^2\} \\
&\quad = \phi_t \otimes \mathbf{Q}(g_{t+1}^2\{(\mathbf{T}_t h_t + \tilde{h}_t) f_{t+1} + \tilde{f}_{t+1}\}^2 - g_{t+1}^2\{f_{t+1} \mathbf{T}_{t+1} h_{t+1} + \tilde{f}_{t+1}\}^2) \\
&\quad = \phi_t \mathbf{Q} \otimes \overleftarrow{\mathbf{Q}}_{\phi_t}(g_{t+1}^2\{(\mathbf{T}_t h_t + \tilde{h}_t) f_{t+1} + \tilde{f}_{t+1}\}^2 - g_{t+1}^2\{f_{t+1} \mathbf{T}_{t+1} h_{t+1} + \tilde{f}_{t+1}\}^2) \\
&\quad = \phi_t \mathbf{Q} \otimes \overleftarrow{\mathbf{Q}}_{\phi_t}\{g_{t+1}^2 f_{t+1}^2 (\mathbf{T}_t h_t + \tilde{h}_t - \mathbf{T}_{t+1} h_{t+1})^2\} \\
&\quad = \phi_t \mathbf{L}_t \overleftarrow{\mathbf{Q}}_{\phi_t}\{g_{t+1} f_{t+1}^2 (\mathbf{T}_t h_t + \tilde{h}_t - \mathbf{T}_{t+1} h_{t+1})^2\},
\end{aligned}$$

the convergence

$$\sum_{i=1}^{N} \mathbb{E}\left[(\upsilon_N^i)^2 \mid \tilde{\mathcal{F}}_t^N\right] \xrightarrow{\mathbb{P}} \phi_t \mathbf{L}_t\{g_{t+1}(f_{t+1}\mathbf{T}_{t+1}h_{t+1} + \tilde{f}_{t+1})^2\} + \tilde{N}^{-1}\left(\phi_t \mathbf{L}_t \overleftarrow{\mathbf{Q}}_{\phi_t}\{g_{t+1}f_{t+1}^2(\mathbf{T}_t h_t + \tilde{h}_t - \mathbf{T}_{t+1}h_{t+1})^2\} + \eta_t\{\mathbf{L}_t(g_{t+1}f_{t+1}^2)\}\right), \quad \text{(A.7)}$$

which verifies (**B1**). In order to check also the condition (**B2**), write, for $\varepsilon \in \mathbb{R}_+^*$,

$$\sum_{i=1}^{N} \mathbb{E}\left[(\upsilon_N^i)^2 \mathbb{1}_{\{|\upsilon_N^i| \geq \varepsilon\}} \mid \tilde{\mathcal{F}}_t^N\right] \leq 4\|g_{t+1}\|_\infty^2(\|h_{t+1}\|_\infty \|f_{t+1}\|_\infty + \|\tilde{f}_{t+1}\|_\infty)^2 \times \mathbb{1}_{\{2\|g_{t+1}\|_\infty(\|h_{t+1}\|_\infty\|f_{t+1}\|_\infty + \|\tilde{f}_{t+1}\|_\infty) \geq \varepsilon\sqrt{N}\}},$$

where the indicator function on the right hand side is zero for N large enough. This shows the condition (**B2**). Hence, for general $(f_{t+1}, \tilde{f}_{t+1})$ (by just replacing $\tilde{f}_{t+1}$ by $\tilde{f}_{t+1} - \phi_{t+1}(\mathbf{T}_{t+1}h_{t+1}f_{t+1} + \tilde{f}_{t+1})$), by Theorem 16, [37, Lemma A.5], and Slutsky's lemma,

$$\sqrt{N}\sum_{i=1}^{N} \frac{\omega_{t+1}^i}{\Omega_{t+1}}\{\tau_{t+1}^i f_{t+1}(\xi_{t+1}^i) + \tilde{f}_{t+1}(\xi_{t+1}^i) - \phi_{t+1}(\mathbf{T}_{t+1}h_{t+1}f_{t+1} + \tilde{f}_{t+1})\} \xrightarrow{\mathcal{D}} \sigma_{t+1}\langle f_{t+1}, \tilde{f}_{t+1}\rangle(h) Z,$$

where Z is a standard Gaussian variable and

$$\sigma_{t+1}^2\langle f_{t+1}, \tilde{f}_{t+1}\rangle(h_{t+1}) := \frac{\phi_t \mathbf{L}_t(g_{t+1}\{f_{t+1}\mathbf{T}_{t+1}h_{t+1} + \tilde{f}_{t+1} - \phi_{t+1}(\mathbf{T}_{t+1}h_{t+1}f_{t+1} + \tilde{f}_{t+1})\}^2)}{(\phi_t \mathbf{L}_t \mathbb{1}_{\mathsf{X}})^2} + \sum_{\ell=0}^{t} \tilde{N}^{\ell-(t+1)} \frac{\phi_\ell \mathbf{L}_\ell\{\overleftarrow{\mathbf{Q}}_{\phi_\ell}(\mathbf{T}_\ell h_\ell + \tilde{h}_\ell - \mathbf{T}_{\ell+1}h_{\ell+1})^2 \mathbf{L}_{\ell+1}\cdots\mathbf{L}_t(g_{t+1}f_{t+1}^2)\}}{(\phi_\ell \mathbf{L}_\ell \cdots \mathbf{L}_{t-1}\mathbb{1}_{\mathsf{X}})(\phi_t \mathbf{L}_t \mathbb{1}_{\mathsf{X}})^2} + \frac{\sigma_t^2(\mathbf{L}_t f_{t+1}, \mathbf{L}_t\{\tilde{h}_{t+1}f_{t+1} + \tilde{f}_{t+1} - \phi_{t+1}(\mathbf{T}_{t+1}h_{t+1}f_{t+1} + \tilde{f}_{t+1})\})}{(\phi_t \mathbf{L}_t \mathbb{1}_{\mathsf{X}})^2}. \quad \text{(A.8)}$$

We now apply the induction hypothesis to the last term. For this purpose, note that, by Lemma 11,

$$h_t \mathbf{L}_t f_{t+1} + \mathbf{L}_t\{\tilde{h}_{t+1}f_{t+1} + \tilde{f}_{t+1} - \phi_{t+1}(\mathbf{T}_{t+1}h_{t+1}f_{t+1} + \tilde{f}_{t+1})\} - \phi_t(h_t \mathbf{L}_t f_{t+1} + \mathbf{L}_t\{\tilde{h}_{t+1}f_{t+1} + \tilde{f}_{t+1} - \phi_{t+1}(\mathbf{T}_{t+1}h_{t+1}f_{t+1} + \tilde{f}_{t+1})\}) = \mathbf{L}_t\{h_{t+1}f_{t+1} + \tilde{f}_{t+1} - \phi_{t+1}(\mathbf{T}_{t+1}h_{t+1}f_{t+1} + \tilde{f}_{t+1})\},$$

yielding, for all $\mathbb{N} \ni s < t$,

$$\begin{aligned}\tilde{\mathbf{D}}_{s+1,t}(h_t \mathbf{L}_t f_{t+1} + \mathbf{L}_t\{\tilde{h}_{t+1}f_{t+1} + \tilde{f}_{t+1} - \phi_{t+1}(\mathbf{T}_{t+1}h_{t+1}f_{t+1} + \tilde{f}_{t+1})\}) &= \mathbf{D}_{s+1,t}\mathbf{L}_t\{h_{t+1}f_{t+1} + \tilde{f}_{t+1} - \phi_{t+1}(\mathbf{T}_{t+1}h_{t+1}f_{t+1} + \tilde{f}_{t+1})\} \\ &= \tilde{\mathbf{D}}_{s+1,t+1}(h_{t+1}f_{t+1} + \tilde{f}_{t+1}).\end{aligned}$$

We may hence conclude that

$$\frac{\sigma_t^2(\mathbf{L}_t f_{t+1}, \mathbf{L}_t\{\tilde{h}_{t+1} f_{t+1} + \tilde{f}_{t+1} - \phi_{t+1}(\mathbf{T}_{t+1} h_{t+1} f_{t+1} + \tilde{f}_{t+1})\})}{(\phi_t \mathbf{L}_t \mathbb{1}_{\mathsf{X}})^2} = \sum_{s=0}^{t-1} \frac{\phi_s \mathbf{L}_s\{g_{s+1}\tilde{\mathbf{D}}_{s+1,t+1}^2(h_{t+1} f_{t+1} + \tilde{f}_{t+1})\}}{(\phi_s \mathbf{L}_s \cdots \mathbf{L}_t \mathbb{1}_{\mathsf{X}})^2} + \sum_{s=0}^{t-1}\sum_{\ell=0}^{s} \tilde{N}^{\ell-(s+1)} \frac{\phi_\ell \mathbf{L}_\ell\{\overleftarrow{\mathbf{Q}}_{\phi_\ell}(\mathbf{T}_\ell h_\ell + \tilde{h}_\ell - \mathbf{T}_{\ell+1} h_{\ell+1})^2 \mathbf{L}_{\ell+1} \cdots \mathbf{L}_s(g_{s+1}\{\mathbf{L}_{s+1} \cdots \mathbf{L}_t f_{t+1}\}^2)\}}{(\phi_\ell \mathbf{L}_\ell \cdots \mathbf{L}_{s-1}\mathbb{1}_{\mathsf{X}})(\phi_s \mathbf{L}_s \cdots \mathbf{L}_t \mathbb{1}_{\mathsf{X}})^2}.$$

Finally, we complete the induction step by noting that

$$\frac{\phi_t \mathbf{L}_t(g_{t+1}\{f_{t+1}\mathbf{T}_{t+1} h_{t+1} + \tilde{f}_{t+1} - \phi_{t+1}(\mathbf{T}_{t+1} h_{t+1} f_{t+1} + \tilde{f}_{t+1})\}^2)}{(\phi_t \mathbf{L}_t \mathbb{1}_{\mathsf{X}})^2} = \frac{\phi_t \mathbf{L}_t\{g_{t+1}\tilde{\mathbf{D}}_{t+1,t+1}^2(h_{t+1} f_{t+1} + \tilde{f}_{t+1})\}}{(\phi_t \mathbf{L}_t \mathbb{1}_{\mathsf{X}})^2}.$$

It remains to check the base case; however, letting, in (A.8), $t = 0$ and $\sigma_0^2 \equiv 0$ (as $\mathbf{T}_0 h_0 = 0$ and $\tau_0^i = 0$ for all $i \in [\![1, N]\!]$) yields

$$\sigma_1^2\langle f_1, \tilde{f}_1\rangle(h_1) = \frac{\phi_0 \mathbf{L}_0(g_1\{f_1 \mathbf{T}_1 h_1 + \tilde{f}_1 - \phi_1(\mathbf{T}_1 h_1 f_1 + \tilde{f}_1)\}^2)}{(\phi_0 \mathbf{L}_0 \mathbb{1}_{\mathsf{X}})^2} + \tilde{N}^{-1}\frac{\phi_0 \mathbf{L}_0\{\overleftarrow{\mathbf{Q}}_{\phi_0}(\tilde{h}_0 - \mathbf{T}_1 h_1)^2 g_1 f_1^2\}}{(\phi_0 \mathbf{L}_0 \mathbb{1}_{\mathsf{X}})^2} = \frac{\phi_0 \mathbf{L}_0\{g_1 \tilde{\mathbf{D}}_{1,1}(h_1 f_1 + \tilde{f}_1)^2\}}{(\phi_0 \mathbf{L}_0 \mathbb{1}_{\mathsf{X}})^2} + \tilde{N}^{-1}\frac{\phi_0 \mathbf{L}_0\{\overleftarrow{\mathbf{Q}}_{\phi_0}(\tilde{h}_0 - \mathbf{T}_1 h_1)^2 g_1 f_1^2\}}{(\phi_0 \mathbf{L}_0 \mathbb{1}_{\mathsf{X}})^2}$$

which is, under the standard convention that $\mathbf{L}_m \mathbf{L}_n = \mathrm{id}$ if $m > n$, in agreement with (3.2). This completes the proof.

Lemma 13. *Let Assumption 1 hold. Then for all $t \in \mathbb{N}$, $f_t \in \mathsf{F}(\mathcal{X})$, and $\tilde{N} \in \mathbb{N}^*$,*

$$\sum_{i=1}^{N} \frac{\omega_t^i}{\Omega_t}(\tau_t^i)^2 f_t(\xi_t^i) \xrightarrow{\mathbb{P}} \phi_t(\mathbf{T}_t^2 h_t f_t) + \eta_t(f_t),$$

where

$$\eta_t(f_t) := \sum_{\ell=0}^{t-1} \tilde{N}^{\ell-t} \frac{\phi_\ell \mathbf{L}_\ell\{\overleftarrow{\mathbf{Q}}_{\phi_\ell}(\mathbf{T}_\ell h_\ell + \tilde{h}_\ell - \mathbf{T}_{\ell+1} h_{\ell+1})^2 \mathbf{L}_{\ell+1} \cdots \mathbf{L}_{t-1} f_t\}}{\phi_\ell \mathbf{L}_\ell \cdots \mathbf{L}_{t-1}\mathbb{1}_{\mathsf{X}}}. \tag{A.9}$$

Proof. Again, we proceed by induction. First, the base case $t = 0$ is trivially true since $\mathbf{T}_0 h_0 = 0$, $\tau_0^i = 0$ for all $i \in [\![1, N]\!]$, and $\sum_{\ell=0}^{-1} = 0$ by convention. We now assume that the claim of the lemma holds true for some $t \in \mathbb{N}$. Since Corollary 2 implies that $N^{-1}\Omega_{t+1} \xrightarrow{\mathbb{P}} \phi_t \mathbf{L}_t \mathbb{1}_{\mathsf{X}}$ it is enough to study the convergence of $N^{-1}\sum_{i=1}^{N} \omega_{t+1}^i(\tau_{t+1}^i)^2 f_{t+1}(\xi_{t+1}^i)$. For this purpose we will apply Theorem 15 to the triangular array

$$\upsilon_N^i := N^{-1}\omega_{t+1}^i(\tau_{t+1}^i)^2 f_{t+1}(\xi_{t+1}^i) \quad (i \in [\![1, N]\!], N \in \mathbb{N}^*)$$

furnished with the filtration $\{\tilde{\mathcal{F}}_t^N\}_{N\in\mathbb{N}^*}$. Note that $|\upsilon_N^i| \leq \|g_{t+1}\|_\infty \|h_{t+1}\|_\infty^2 \|f_{t+1}\|_\infty / N$ for all $i \in [\![1, N]\!]$ and $N \in \mathbb{N}^*$. In addition, using, first, that $\{\upsilon_N^i\}_{i=1}^N$ are conditionally i.i.d given $\tilde{\mathcal{F}}_t^N$ and,

second, that for all $i \in [\![1, N]\!]$, the backward indices $\{J_{t+1}^{(i,j)}\}_{j=1}^{\tilde{N}}$ are conditionally i.i.d. given $\tilde{\mathcal{F}}_t^N$ and ξ_{t+1}^i,

$$\begin{aligned}
&\sum_{i=1}^N \mathbb{E}\left[\upsilon_N^i \mid \tilde{\mathcal{F}}_t^N\right] \\
&= \mathbb{E}\left[\omega_{t+1}^1 (\tau_{t+1}^1)^2 f_{t+1}(\xi_{t+1}^1) \mid \tilde{\mathcal{F}}_t^N\right] \\
&= \tilde{N}^{-1}\mathbb{E}\left[\omega_{t+1}^1 f_{t+1}(\xi_{t+1}^1)\mathbb{E}\left[\left(\tau_t^{J_{t+1}^{(1,1)}} + \tilde{h}_t(\xi_t^{J_{t+1}^{(1,1)}}, \xi_{t+1}^1)\right)^2 \mid \tilde{\mathcal{F}}_t^N \vee \mathcal{F}_{t+1}^N\right] \mid \tilde{\mathcal{F}}_t^N\right] \qquad \text{(A.10)} \\
&\quad + \tilde{N}^{-1}(\tilde{N}-1)\mathbb{E}\left[\omega_{t+1}^1 f_{t+1}(\xi_{t+1}^1)\mathbb{E}^2\left[\tau_t^{J_{t+1}^{(1,1)}} + \tilde{h}_t(\xi_t^{J_{t+1}^{(1,1)}}, \xi_{t+1}^1) \mid \tilde{\mathcal{F}}_t^N \vee \mathcal{F}_{t+1}^N\right] \mid \tilde{\mathcal{F}}_t^N\right]. \qquad \text{(A.11)}
\end{aligned}$$

We treat separately the two terms (A.10) and (A.11). First,

$$\begin{aligned}
\text{(A.10)} &= \tilde{N}^{-1}\mathbb{E}\left[\omega_{t+1}^1 f_{t+1}(\xi_{t+1}^1)\sum_{\ell=1}^N \frac{\omega_t^\ell q(\xi_t^\ell, \xi_{t+1}^1)}{\sum_{\ell'=1}^N \omega_t^{\ell'} q(\xi_t^{\ell'}, \xi_{t+1}^1)}\{\tau_t^\ell + \tilde{h}_t(\xi_t^\ell, \xi_{t+1}^1)\}^2 \mid \tilde{\mathcal{F}}_t^N\right] \\
&= \tilde{N}^{-1}\sum_{i=1}^N \frac{\omega_t^i}{\Omega_t}\int q(\xi_t^i, x) g_{t+1}(x) f_{t+1}(x) \sum_{\ell=1}^N \frac{\omega_t^\ell q(\xi_t^\ell, x)}{\sum_{\ell'=1}^N \omega_t^{\ell'} q(\xi_t^{\ell'}, x)}\{\tau_t^\ell + \tilde{h}_t(\xi_t^\ell, x)\}^2 \,\mu(\mathrm{d}x) \\
&= \tilde{N}^{-1}\sum_{\ell=1}^N \frac{\omega_t^\ell}{\Omega_t}\int q(\xi_t^\ell, x)\, g_{t+1}(x) f_{t+1}(x)\{\tau_t^\ell + \tilde{h}_t(\xi_t^\ell, x)\}^2 \,\mu(\mathrm{d}x).
\end{aligned}$$

Using Corollary 2 and the induction hypothesis we obtain the limits

$$\begin{aligned}
&\sum_{\ell=1}^N \frac{\omega_t^\ell}{\Omega_t}(\tau_t^\ell)^2 \mathbf{L}_t(\xi_t^\ell, f_{t+1}) \xrightarrow{\mathbb{P}} \phi_t(\mathbf{T}_t^2 h_t \mathbf{L}_t f_{t+1}) + \eta_t(\mathbf{L}_t f_{t+1}), \\
&\sum_{\ell=1}^N \frac{\omega_t^\ell}{\Omega_t}\tau_t^\ell \mathbf{L}_t(\xi_t^\ell, \tilde{h}_t f_{t+1}) \xrightarrow{\mathbb{P}} \phi_t\{\mathbf{T}_t h_t \mathbf{L}_t(\tilde{h}_t f_{t+1})\}, \\
&\sum_{\ell=1}^N \frac{\omega_t^\ell}{\Omega_t}\mathbf{L}_t(\xi_t^\ell, \tilde{h}_t^2 f_{t+1}) \xrightarrow{\mathbb{P}} \phi_t \mathbf{L}_t(\tilde{h}_t^2 f_{t+1}),
\end{aligned}$$

which yield

$$\begin{aligned}
\text{(A.10)} \xrightarrow{\mathbb{P}} \tilde{N}^{-1}&\left(\phi_t(\mathbf{T}_t^2 h_t \mathbf{L}_t f_{t+1}) + \eta_t(\mathbf{L}_t f_{t+1}) + 2\phi_t\{\mathbf{T}_t h_t \mathbf{L}_t(\tilde{h}_t f_{t+1})\} + \phi_t \mathbf{L}_t(\tilde{h}_t^2 f_{t+1})\right) \\
&= \tilde{N}^{-1}\left(\phi_t \mathbf{L}_t\{(\mathbf{T}_t h_t + \tilde{h}_t)^2 f_{t+1}\} + \eta_t(\mathbf{L}_t f_{t+1})\right).
\end{aligned}$$

We turn to the second term (A.11) and equate the same with $\tilde{N}^{-1}(\tilde{N}-1)\phi_t^N \mathbf{Q}\varphi_N$, where

$$\varphi_N(x) := g_{t+1}(x) f_{t+1}(x)\left(\sum_{\ell=1}^N \frac{\omega_t^\ell q(\xi_t^\ell, x)}{\sum_{\ell'=1}^N \omega_t^{\ell'} q(\xi_t^{\ell'}, x)}\{\tau_t^\ell + \tilde{h}_t(\xi_t^\ell, x)\}\right)^2 \quad (x \in \mathsf{X}).$$

Since, $\|\varphi_N\|_\infty \leq \|g_{t+1}\|_\infty \|f_{t+1}\|_\infty \|h_{t+1}\|_\infty$ for all $N \in \mathbb{N}$, and, by Corollary 2, for all $x \in \mathsf{X}$, $\mathbb{P}$-a.s.,

$$\begin{aligned}\varphi_N(x) &\to g_{t+1}(x) f_{t+1}(x) \left(\frac{\int q(\tilde{x}, x)\{\mathbf{T}_t h_t(\tilde{x}) + \tilde{h}_t(\tilde{x}, x)\}\, \phi_t(\mathrm{d}\tilde{x})}{\int q(\tilde{x}, x)\, \phi_t(\mathrm{d}\tilde{x})} \right)^2 \\ &= g_{t+1}(x) f_{t+1}(x) \overleftarrow{\mathbf{Q}}^2_{\phi_t}(\mathbf{T}_t h_t + \tilde{h}_t)(x) \\ &= g_{t+1}(x) f_{t+1}(x) \mathbf{T}^2_{t+1} h_{t+1}(x),\end{aligned}$$

we may, under Assumption 1, apply Lemma 14, yielding

$$\phi_t^N \mathbf{Q} \varphi_N \xrightarrow{\mathbb{P}} \phi_t \mathbf{Q}(g_{t+1} f_{t+1} \mathbf{T}^2_{t+1} h_{t+1}) = \phi_t \mathbf{L}_t(\mathbf{T}^2_{t+1} h_{t+1} f_{t+1}).$$

Consequently,

$$\begin{aligned}\sum_{i=1}^N \mathbb{E}\left[\upsilon_N^i \mid \tilde{\mathcal{F}}_t^N \right] &\xrightarrow{\mathbb{P}} \phi_t \mathbf{L}_t(\mathbf{T}^2_{t+1} h_{t+1} f_{t+1}) \\ &\quad + \tilde{N}^{-1} \left(\phi_t \mathbf{L}_t\{(\mathbf{T}_t h_t + \tilde{h}_t)^2 f_{t+1}\} - \phi_t \mathbf{L}_t(\mathbf{T}^2_{t+1} h_{t+1} f_{t+1}) + \eta_t(\mathbf{L}_t f_{t+1}) \right). \quad \text{(A.12)}\end{aligned}$$

In order to show that $\sum_{i=1}^N \upsilon_N^i$ has the same limit (A.12) in probability we use Theorem 15. Condition (**A1**) is easily checked by reusing (A.12) with f_{t+1} replaced by $|f_{t+1}|$. In order to check (**A2**) we simply note that for all $\varepsilon \in \mathbb{R}_+^*$,

$$\sum_{i=1}^N \mathbb{E}\left[|\upsilon_N^i| \mathbb{1}_{\{|\upsilon_N^i| \geq \varepsilon\}} \mid \tilde{\mathcal{F}}_t^N \right] \leq \|g_{t+1}\|_\infty \|h_{t+1}\|_\infty^2 \|f_{t+1}\|_\infty \mathbb{1}_{\{\|g_{t+1}\|_\infty \|h_{t+1}\|_\infty^2 \|f_{t+1}\|_\infty \geq \varepsilon N\}},$$

where the right hand side is zero for N large enough. Thus, Theorem 15 applies and since, by reversibility,

$$\begin{aligned}&\phi_t \mathbf{L}_t\{(\mathbf{T}_t h_t + \tilde{h}_t)^2 f_{t+1}\} - \phi_t \mathbf{L}_t(\mathbf{T}^2_{t+1} h_{t+1} f_{t+1}) \\ &\quad = \phi_t \otimes \mathbf{Q}(\{(\mathbf{T}_t h_t + \tilde{h}_t)^2 - \mathbf{T}^2_{t+1} h_{t+1}\} g_{t+1} f_{t+1}) \\ &\quad = \phi_t \mathbf{Q} \otimes \overleftarrow{\mathbf{Q}}_{\phi_t}\{(\mathbf{T}_t h_t + \tilde{h}_t - \mathbf{T}_{t+1} h_{t+1})^2 g_{t+1} f_{t+1}\} \\ &\quad = \phi_t \mathbf{L}_t\{\overleftarrow{\mathbf{Q}}_{\phi_t}(\mathbf{T}_t h_t + \tilde{h}_t - \mathbf{T}_{t+1} h_{t+1})^2 f_{t+1}\},\end{aligned}$$

Slutsky's lemma implies

$$\begin{aligned}\sum_{i=1}^N \frac{\omega_{t+1}^i}{\Omega_{t+1}} (\tau_{t+1}^i)^2 f_{t+1}(\xi_{t+1}^i) = N \Omega_{t+1}^{-1} \sum_{i=1}^N \upsilon_N^i &\xrightarrow{\mathbb{P}} \phi_{t+1}(\mathbf{T}^2_{t+1} h_{t+1} f_{t+1}) \\ &+ \frac{1}{\tilde{N}(\phi_t \mathbf{L}_t \mathbb{1}_{\mathsf{X}})} \left(\phi_t \mathbf{L}_t\{\overleftarrow{\mathbf{Q}}_{\phi_t}(\mathbf{T}_t h_t + \tilde{h}_t - \mathbf{T}_{t+1} h_{t+1})^2 f_{t+1}\} + \eta_t(\mathbf{L}_t f_{t+1}) \right).\end{aligned}$$

We may now conclude the proof by noting, using the induction hypothesis, the identity

$$(\phi_\ell \mathbf{L}_\ell \cdots \mathbf{L}_{t-1} \mathbb{1}_{\mathsf{X}})(\phi_t \mathbf{L}_t \mathbb{1}_{\mathsf{X}}) = \phi_\ell \mathbf{L}_\ell \cdots \mathbf{L}_t \mathbb{1}_{\mathsf{X}},$$

and the convention $\mathbf{L}_{t+1}\mathbf{L}_t = \mathrm{id}$, that

$$\frac{1}{\tilde{N}(\phi_t \mathbf{L}_t \mathbb{1}_{\mathsf{X}})} \left(\phi_t \mathbf{L}_t \{\overleftarrow{\mathbf{Q}}_{\phi_t}(\mathbf{T}_t h_t + \tilde{h}_t - \mathbf{T}_{t+1} h_{t+1})^2 f_{t+1}\} + \eta_t(\mathbf{L}_t f_{t+1})\right)$$
$$= \sum_{\ell=0}^{t} \tilde{N}^{\ell-(t+1)} \frac{\phi_\ell \mathbf{L}_\ell \{\overleftarrow{\mathbf{Q}}_{\phi_\ell}(\mathbf{T}_\ell h_\ell + \tilde{h}_\ell - \mathbf{T}_{\ell+1} h_{\ell+1})^2 \mathbf{L}_{\ell+1} \cdots \mathbf{L}_t f_{t+1}\}}{\phi_\ell \mathbf{L}_\ell \cdots \mathbf{L}_t \mathbb{1}_{\mathsf{X}}}.$$

□

The following lemma formalizes an argument used in the proof of [11, Theorem 8].

Lemma 14. *Let Assumption 1 hold. Let $\mathbf{\Psi}$ be a possibly unnormalized transition kernel on $(\mathsf{X}, \mathcal{X})$ having transition density $\psi \in \mathsf{F}(\mathcal{X}^2)$ with respect to some reference measure λ. Moreover, let $\{\varphi_N\}_{N\in\mathbb{N}^*}$ be a sequence of functions in $\mathsf{F}(\mathcal{X})$ for which*

(i) there exists $\varphi \in \mathsf{F}(\mathcal{X})$ such that for all $x \in \mathsf{X}$, $\varphi_N(x) \to \varphi(x)$, $\mathbb{P}$-a.s., and
(ii) there exists $|\varphi|_\infty \in \mathbb{R}_+^*$ such that $\|\varphi_N\|_\infty \le |\varphi|_\infty$ for all $N \in \mathbb{N}^*$.

Then for all $t \in \mathbb{N}$, $\phi_t^N \mathbf{\Psi} \varphi_N \xrightarrow{\mathbb{P}} \phi_t \mathbf{\Psi} \varphi$.

Proof. Since, by Corollary 2, $\phi_t^N \mathbf{\Psi} \varphi \xrightarrow{\mathbb{P}} \phi_t \mathbf{\Psi} \varphi$, it is enough to establish that

$$\phi_t^N \mathbf{\Psi} \varphi_N \xrightarrow{\mathbb{P}} \phi_t^N \mathbf{\Psi} \varphi.$$

For this purpose, set

$$a_N(x) := |\varphi_N(x) - \varphi(x)| \int \psi(\tilde{x}, x)\, \phi_t^N(\mathrm{d}\tilde{x}),$$
$$\tilde{a}_N(x) := \int \psi(\tilde{x}, x)\, \phi_t^N(\mathrm{d}\tilde{x}) \qquad (N \in \mathbb{N}^*, x \in \mathsf{X}).$$

Since $|\phi_t^N \mathbf{\Psi} \varphi_N - \phi_t^N \mathbf{\Psi} \varphi| \le \lambda a_N$ it is, by Markov's inequality, enough to show that $\mathbb{E}[\lambda a_N]$ tends to zero as N tends to infinity. However, by Fubini's theorem,

$$\lim_{N\to\infty} \mathbb{E}[\lambda a_N] = \lim_{N\to\infty} \int \mathbb{E}[a_N(x)]\, \lambda(\mathrm{d}x) = 0,$$

where the last equality is a consequence of the generalized Lebesgue dominated convergence theorem provided that

(i) $\lim_{N\to\infty} \mathbb{E}[a_N(x)] = 0$ for all $x \in \mathsf{X}$,
(ii) there exists $c \in \mathbb{R}_+^*$ such that $\mathbb{E}[a_N(x)] \le c\, \mathbb{E}[\tilde{a}_N(x)]$ for all $x \in \mathsf{X}$,
(iii) $\lim_{N\to\infty} \int \mathbb{E}[\tilde{a}_N(x)]\, \lambda(\mathrm{d}x) = \int \lim_{N\to\infty} \mathbb{E}[\tilde{a}_N(x)]\, \lambda(\mathrm{d}x)$.

Here (i) is implied by Corollary 2 and, as $|a_N(x)| \le \|\psi\|_\infty(\|\varphi\|_\infty + |\varphi|_\infty)$ for all $x \in \mathsf{X}$, the standard dominated convergence theorem. Moreover, (ii) is satisfied with $c = \|\varphi\|_\infty + |\varphi|_\infty$. Finally, to check (iii), notice that

$$\lim_{N\to\infty} \int \mathbb{E}[\tilde{a}_N(x)]\, \lambda(\mathrm{d}x) \overset{(a)}{=} \lim_{N\to\infty} \mathbb{E}[\phi_t^N \mathbf{\Psi} \mathbb{1}_{\mathsf{X}}] \overset{(b)}{=} \phi_t \mathbf{\Psi} \mathbb{1}_{\mathsf{X}} \overset{(c)}{=} \iint \psi(\tilde{x}, x)\, \phi_t(\mathrm{d}\tilde{x})\, \lambda(\mathrm{d}x)$$
$$\overset{(d)}{=} \int \lim_{N\to\infty} \mathbb{E}[\tilde{a}_N(x)]\, \lambda(\mathrm{d}x),$$

where (a) and (c) follow by Fubini's theorem and (b) and (d) are obtained from Corollary 2 and the standard dominated convergence theorem (as $\mathbf{\Psi}\mathbb{1}_{\mathsf{X}} \in \mathsf{F}(\mathcal{X})$ and $\psi \in \mathsf{F}(\mathcal{X}^2)$ by assumption). This completes the proof. □

A.4. Proof of Proposition 7

By [18, Lemma 1],

$$\|\tilde{\mathbf{D}}_{s+1,t} h_t\|_\infty \leq \|\mathbf{L}_{s+1} \dots \mathbf{L}_t \mathbb{1}_{\mathsf{X}}\|_\infty \sum_{\ell=0}^{t-1} \varrho^{\max\{s-\ell+2, \ell-s-3, 0\}} \operatorname{osc}(\tilde{h}_\ell), \tag{A.13}$$

and, consequently,

$$\sum_{s=0}^{t-1} \frac{\phi_s \mathbf{L}_s (g_{s+1} \tilde{\mathbf{D}}_{s+1,t}^2 h_t)}{(\phi_s \mathbf{L}_s \cdots \mathbf{L}_{t-1} \mathbb{1}_{\mathsf{X}})^2} \leq |\tilde{h}|_\infty^2 \sum_{s=0}^{t-1} \frac{(\phi_s \mathbf{L}_s g_{s+1}) \|\mathbf{L}_{s+1} \dots \mathbf{L}_t \mathbb{1}_{\mathsf{X}}\|_\infty^2}{(\phi_s \mathbf{L}_s \cdots \mathbf{L}_{t-1} \mathbb{1}_{\mathsf{X}})^2} \left(\sum_{\ell=0}^{t-1} \varrho^{\max\{s-\ell+2, \ell-s-3, 0\}} \right)^2 .$$

Now, under Assumption 2, for all $x \in \mathsf{X}$,

$$\underline{\varepsilon} \mu(g_{s+2} \mathbf{L}_{s+2} \cdots \mathbf{L}_t \mathbb{1}_{\mathsf{X}}) \leq \mathbf{L}_{s+1} \cdots \mathbf{L}_t \mathbb{1}_{\mathsf{X}}(x) \leq \bar{\varepsilon} \mu(g_{s+2} \mathbf{L}_{s+2} \dots \mathbf{L}_t \mathbb{1}_{\mathsf{X}}) \tag{A.14}$$

implying that

$$\frac{(\phi_s \mathbf{L}_s g_{s+1}) \|\mathbf{L}_{s+1} \cdots \mathbf{L}_t \mathbb{1}_{\mathsf{X}}\|_\infty^2}{(\phi_s \mathbf{L}_s \cdots \mathbf{L}_{t-1} \mathbb{1}_{\mathsf{X}})^2} = \frac{(\phi_{s+1} g_{s+1}) \|\mathbf{L}_{s+1} \cdots \mathbf{L}_t \mathbb{1}_{\mathsf{X}}\|_\infty^2}{(\phi_s \mathbf{L}_s \mathbb{1}_{\mathsf{X}})(\phi_{s+1} \mathbf{L}_{s+1} \cdots \mathbf{L}_{t-1} \mathbb{1}_{\mathsf{X}})^2} \leq \frac{\bar{\delta}}{\underline{\delta}} \left(\frac{\bar{\varepsilon}}{\underline{\varepsilon}} \right)^2 = \frac{\bar{\delta}}{\underline{\delta}(1-\varrho)^2}.$$

Moreover, as

$$\begin{aligned} \sum_{s=0}^{t-1} \left(\sum_{\ell=0}^{t-1} \varrho^{\max\{s-\ell+2, \ell-s-3, 0\}} \right)^2 &= \sum_{s=0}^{t-4} \left(\frac{2 - \varrho^{s+3} - \varrho^{t-s-3}}{1-\varrho} \right)^2 \\ &\quad + \sum_{s=t-3}^{t-1} \left(\rho^{s+2} \frac{1-\rho^{-t}}{1-\rho^{-1}} \right) = \frac{4t}{(1-\varrho)^2} + o(t), \end{aligned}$$

we conclude that

$$\limsup_{t \to \infty} \frac{1}{t} \sum_{s=0}^{t-1} \frac{\phi_s \mathbf{L}_s (g_{s+1} \tilde{\mathbf{D}}_{s+1,t}^2 h_t)}{(\phi_s \mathbf{L}_s \cdots \mathbf{L}_{t-1} \mathbb{1}_{\mathsf{X}})^2} \leq |\tilde{h}|_\infty^2 \frac{4\bar{\delta}}{\underline{\delta}(1-\varrho)^4}.$$

A.5. Proof of Theorem 8

The first term of $\sigma_t^2(h_t)$ is the asymptotic variance of the FFBSm algorithm, which is, by Proposition 7, bounded by $4|\tilde{h}|_\infty^2 \bar{\delta}/\{\underline{\delta}(1-\varrho)^4\}$. To treat the second term, we bound, using (A.14),

$$\begin{aligned} \frac{\mathbf{L}_{\ell+1} \cdots \mathbf{L}_s (g_{s+1} \{\mathbf{L}_{s+1} \cdots \mathbf{L}_{t-1} \mathbb{1}_{\mathsf{X}}\}^2)}{(\phi_{\ell+1} \mathbf{L}_{\ell+1} \cdots \mathbf{L}_{s-1} \mathbb{1}_{\mathsf{X}})(\phi_s \mathbf{L}_s \cdots \mathbf{L}_{t-1} \mathbb{1}_{\mathsf{X}})^2} &\leq \frac{\|\mathbf{L}_{\ell+1} \cdots \mathbf{L}_{s-1} \mathbb{1}_{\mathsf{X}}\|_\infty \bar{\delta} \|\mathbf{L}_{s+1} \cdots \mathbf{L}_{t-1} \mathbb{1}_{\mathsf{X}}\|_\infty^2}{(\phi_{\ell+1} \mathbf{L}_{\ell+1} \cdots \mathbf{L}_{s-1} \mathbb{1}_{\mathsf{X}})(\phi_s \mathbf{L}_s \mathbb{1}_{\mathsf{X}})(\phi_{s+1} \mathbf{L}_{s+1} \cdots \mathbf{L}_{t-1} \mathbb{1}_{\mathsf{X}})^2} \\ &\leq \frac{\bar{\delta}}{\underline{\delta}(1-\varrho)^3}. \end{aligned} \tag{A.15}$$

Moreover, since $\mathbf{T}_{\ell+1}h_{\ell+1} = \overleftarrow{\mathbf{Q}}_{\phi_\ell}(\mathbf{T}_\ell h_\ell + \tilde{h}_\ell)$ and $\mathbf{T}_\ell h_\ell = \mathbf{D}_{\ell,\ell}h_\ell$, we obtain, by reusing (A.13),

$$\|\mathbf{T}_\ell h_\ell + \tilde{h}_\ell - \mathbf{T}_{\ell+1}h_{\ell+1}\|_\infty \leq \operatorname{osc}(\mathbf{T}_\ell h_\ell) + \operatorname{osc}(\tilde{h}_\ell) \leq 4\|\tilde{\mathbf{D}}_{\ell,\ell}h_\ell\|_\infty + \operatorname{osc}(\tilde{h}_\ell) \leq |\tilde{h}|_\infty \left(\frac{4\bar{\delta}\varrho^2}{1-\varrho} + 1\right).$$

Thus,

$$\frac{1}{t}\sum_{s=0}^{t-1}\sum_{\ell=0}^{s} \tilde{N}^{\ell-(s+1)} \frac{\phi_{\ell+1}\{\overleftarrow{\mathbf{Q}}_{\phi_\ell}(\mathbf{T}_\ell h_\ell + \tilde{h}_\ell - \mathbf{T}_{\ell+1}h_{\ell+1})^2 \mathbf{L}_{\ell+1}\cdots\mathbf{L}_s(g_{s+1}\{\mathbf{L}_{s+1}\cdots\mathbf{L}_{t-1}f_t\}^2)\}}{(\phi_{\ell+1}\mathbf{L}_{\ell+1}\cdots\mathbf{L}_{s-1}\mathbb{1}_{\mathsf{X}})(\phi_s\mathbf{L}_s\cdots\mathbf{L}_{t-1}\mathbb{1}_{\mathsf{X}})^2}$$
$$\leq |\tilde{h}|_\infty^2 \frac{\bar{\delta}}{\underline{\delta}(1-\varrho)^3}\left(\frac{4\bar{\delta}\varrho^2}{1-\varrho}+1\right)^2 \frac{1}{t}\sum_{s=0}^{t-1}\sum_{\ell=0}^{s}\tilde{N}^{\ell-(s+1)},$$

and since

$$\lim_{s\to\infty}\sum_{\ell=0}^{s}\tilde{N}^{\ell-(s+1)} = (\tilde{N}-1)^{-1}$$

we may conclude the proof by taking the Cesàro mean.

A.6. Proof of Theorem 9

In the case of marginal smoothing, [11, Theorem 12] provides, for $t \geq \hat{s}$ (since the variance vanishes for $t < \hat{s}$, the result holds trivially true in this case), the time uniform bound

$$\tilde{\sigma}_t^2(h_t) \leq \operatorname{osc}^2(\tilde{h}_{\hat{s}})\frac{\bar{\delta}^2(1+\varrho^2)}{(1+\varrho)(1-\varrho)^3}$$

and hence, since all terms are zero except $\tilde{h}_{\hat{s}}$, it is enough to bound the quantity

$$\sum_{s=\hat{s}}^{t-1}\sum_{\ell=\hat{s}}^{s} \tilde{N}^{\ell-(s+1)} \frac{\phi_{\ell+1}\{\overleftarrow{\mathbf{Q}}_{\phi_\ell}(\mathbf{T}_\ell h_\ell + \tilde{h}_\ell - \mathbf{T}_{\ell+1}h_{\ell+1})^2 \mathbf{L}_{\ell+1}\cdots\mathbf{L}_s(g_{s+1}\{\mathbf{L}_{s+1}\cdots\mathbf{L}_{t-1}f_t\}^2)\}}{(\phi_{\ell+1}\mathbf{L}_{\ell+1}\cdots\mathbf{L}_{s-1}\mathbb{1}_{\mathsf{X}})(\phi_s\mathbf{L}_s\cdots\mathbf{L}_{t-1}\mathbb{1}_{\mathsf{X}})^2} \tag{A.16}$$

(where $\tilde{h}_\ell = 0$ for $\ell > \hat{s}$). In addition, by [11, Lemma 10], for all $\ell \geq \hat{s}$,

$$\|\tilde{\mathbf{D}}_{\ell,\ell}h_\ell\|_\infty \leq \varrho^{\ell-\hat{s}}\operatorname{osc}(\tilde{h}_{\hat{s}}),$$

yielding

$$\|\mathbf{T}_\ell h_\ell + \tilde{h}_\ell - \mathbf{T}_{\ell+1}h_{\ell+1}\|_\infty \leq 2\varrho^{\ell-\hat{s}}\operatorname{osc}(\tilde{h}_{\hat{s}}).$$

By combining this with (A.15) we obtain, via standard operations on geometric sums,

$$\text{(A.16)} \leq 4\operatorname{osc}^2(\tilde{h}_{\hat{s}})\frac{\bar{\delta}}{\underline{\delta}(1-\varrho)^3}\sum_{s=\hat{s}}^{t-1}\sum_{\ell=\hat{s}}^{s}\tilde{N}^{\ell-(s+1)}\varrho^{2(\ell-\hat{s})}$$
$$= 4\operatorname{osc}^2(\tilde{h}_{\hat{s}})\frac{\bar{\delta}}{\underline{\delta}(1-\varrho)^3} \times \begin{cases} \frac{1}{(1-\tilde{N}\varrho^2)}\left(\frac{1-\tilde{N}^{-(t-\hat{s})}}{\tilde{N}-1} - \varrho^2\frac{1-\varrho^{2(t-\hat{s})}}{1-\varrho^2}\right) & \text{if } \tilde{N}\varrho^2 \neq 1, \\ \frac{1}{\tilde{N}-1}\left(2-\tilde{N}^{-(t-\hat{s})} - (t-\hat{s})\tilde{N}^{-(t-\hat{s})+1} - \tilde{N}^{-(t-\hat{s})}\right) & \text{if } \tilde{N}\varrho^2 = 1, \end{cases}$$

and hence, letting t tend to infinity,

$$(\text{A.16}) \leq 4\,\mathrm{osc}^2(\tilde{h}_{\hat{s}})\frac{\bar{\delta}}{\underline{\delta}(1-\varrho)^3} \times \begin{cases} \dfrac{1}{(\tilde{N}-1)(1-\varrho^2)} & \text{if } \tilde{N}\varrho^2 \neq 1, \\ \dfrac{2}{\tilde{N}-1} & \text{if } \tilde{N}\varrho^2 = 1, \end{cases}$$

which concludes the proof.

Appendix B: Technical results

B.1. Conditional limit theorems for triangular arrays of dependent random variables

We first recall two results, obtained in [12] (but reformulated slightly here for our purposes), which are essential for the developments of the present paper.

Theorem 15 ([12])**.** *Let $(\Omega, \mathcal{A}, \{\mathcal{F}_N\}_{N\in\mathbb{N}^*}, \mathbb{P})$ be a filtered probability space. In addition, let $\{\upsilon_N^i\}_{i=1}^N$, $N \in \mathbb{N}^*$, be a triangular array of random variables on $(\Omega, \mathcal{A}, \mathbb{P})$ such that for all $N \in \mathbb{N}^*$, the variables $\{\upsilon_N^i\}_{i=1}^N$ are conditionally independent given $\mathcal{F}_N$ with $\mathbb{E}[|\upsilon_N^i| \mid \mathcal{F}_N] < \infty$, $\mathbb{P}$-a.s., for all $i \in [\![1, N]\!]$. Moreover, assume that*

(A1) $$\lim_{\lambda\to\infty} \sup_{N\in\mathbb{N}^*} \mathbb{P}\left(\sum_{i=1}^N \mathbb{E}[|\upsilon_N^i| \mid \mathcal{F}_N] \geq \lambda\right) = 0.$$

(A2) *For all $\varepsilon > 0$, as $N \to \infty$,*

$$\sum_{i=1}^N \mathbb{E}\left[|\upsilon_N^i| \mathbb{1}_{\{|\upsilon_N^i|\geq\varepsilon\}} \mid \mathcal{F}_N\right] \xrightarrow{\mathbb{P}} 0.$$

Then, as $N \to \infty$,

$$\max_{m\in[\![1,N]\!]} \left|\sum_{i=1}^m \upsilon_N^i - \sum_{i=1}^m \mathbb{E}\left[\upsilon_N^i \mid \mathcal{F}_N\right]\right| \xrightarrow{\mathbb{P}} 0.$$

Theorem 16 ([12])**.** *Let the assumptions of Theorem 15 hold with $\mathbb{E}[(\upsilon_N^i)^2 \mid \mathcal{F}_N] < \infty$, $\mathbb{P}$-a.s., for all $i \in [\![1, N]\!]$, and **(A1)** and **(A2)** replaced by*

(B1) *For some constant $\varsigma^2 > 0$, as $N \to \infty$,*

$$\sum_{i=1}^N \left(\mathbb{E}[(\upsilon_N^i)^2 \mid \mathcal{F}_N] - \mathbb{E}^2\left[\upsilon_N^i \mid \mathcal{F}_N\right]\right) \xrightarrow{\mathbb{P}} \varsigma^2.$$

(B2) *For all $\varepsilon > 0$, as $N \to \infty$,*

$$\sum_{i=1}^N \mathbb{E}\left[(\upsilon_N^i)^2 \mathbb{1}_{\{|\upsilon_N^i|\geq\varepsilon\}} \mid \mathcal{F}_N\right] \xrightarrow{\mathbb{P}} 0.$$

Then, for all $u \in \mathbb{R}$, as $N \to \infty$,

$$\mathbb{E}\left[\exp\left(\mathrm{i}\,u\sum_{i=1}^N \left\{\upsilon_N^i - \mathbb{E}[\upsilon_N^i \mid \mathcal{F}_N]\right\}\right) \mid \mathcal{F}_N\right] \xrightarrow{\mathbb{P}} \exp\left(-u^2\varsigma^2/2\right).$$

B.2. An accept-reject-based algorithm for backward sampling

Given two subsequent particle samples $\{(\xi_{s-1}^i, \omega_{s-1}^i)\}_{i=1}^N$and $\{(\xi_s^i, \omega_s^i)\}_{i=1}^N$, the following algorithm, which is a trivial adjustment of [11, Algorithm 1], simulates the full set $\{J_s^{(i,j)} : (i,j) \in [\![1, N]\!] \times [\![1, \tilde{N}]\!]\}$ of backward indices required for one iteration of PaRIS. The algorithm requires Assumption 1(ii) to hold true.

Algorithm 3 Accept-reject-based backward sampling

Require: Particle samples $\{(\xi_{s-1}^i, \omega_{s-1}^i)\}_{i=1}^N$and $\{(\xi_s^i, \omega_s^i)\}_{i=1}^N$.

for $j = 1 \to \tilde{N}$ **do**
  set $\mathsf{L} \leftarrow [\![1, N]\!]$;
  while $\mathsf{L} \neq \varnothing$ **do**
    set $n \leftarrow \#\mathsf{L}$;
    draw $(I_1, \dots, I_N) \sim \mathsf{Pr}(\{\omega_{s-1}^i\}_{i=1}^N)^{\otimes N}$;
    draw $(U_1, \dots, U_N) \sim \mathsf{U}(0,1)^{\otimes N}$;
    set $\mathsf{L}_n \leftarrow \varnothing$;
    for $k = 1 \to n$ **do**
      if $U_k \leq q(\xi_{s-1}^{I_k}, \xi_s^{\mathsf{L}(k)})/\bar{\varepsilon}$ **then**
        set $J_s^{(\mathsf{L}(k),j)} \leftarrow I_k$;
      else
        set $\mathsf{L}_n \leftarrow \mathsf{L}_n \cup \{\mathsf{L}(k)\}$;
      end if
    end for
    set $\mathsf{L} \leftarrow \mathsf{L}_n$;
  end while
end for
return $\{J_s^{(i,j)} : (i,j) \in [\![1, N]\!] \times [\![1, \tilde{N}]\!]\}$

References

[1] L. E. Baum, T. P. Petrie, G. Soules, and N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *Ann. Math. Statist.*, 41(1):164–171, 1970.

[2] O. Cappé. Ten years of HMMs (online bibliography 1989–2000), March 2001.

[3] O. Cappé. Online EM algorithm for hidden Markov models. *Journal of Computational and Graphical Statistics*, 20(3):728–749, 2011.

[4] O. Cappé, S. J. Godsill, and E. Moulines. An overview of existing methods and recent advances in sequential Monte Carlo. *IEEE Proceedings*, 95(5):899–924, 2007.

[5] O. Cappé, E. Moulines, and T. Rydén. *Inference in Hidden Markov Models*. Springer, 2005.

[6] S. Chib, F. Nadari, and N. Shephard. Markov chain Monte Carlo methods for stochastic volatility models. *J. Econometrics*, 108:281–316, 2002.

[7] D. Crisan and K. Heine. Stability of the discrete time filter in terms of the tails of noise distributions. *J. Lond. Math. Soc. (2)*, 78(2):441–458, 2008.

[8] P. Del Moral. *Feynman-Kac Formulae. Genealogical and Interacting Particle Systems with Applications*. Springer, 2004.

[9] P. Del Moral, A. Doucet, and S. Singh. A Backward Particle Interpretation of Feynman-Kac Formulae. *ESAIM M2AN*, 44(5):947–975, 2010.

[10] P. Del Moral and A. Guionnet. On the stability of interacting processes with applications to filtering and genetic algorithms. *Annales de l'Institut Henri Poincaré*, 37:155–194, 2001.
[11] R. Douc, A. Garivier, E. Moulines, and J. Olsson. Sequential Monte Carlo smoothing for general state space hidden Markov models. *Ann. Appl. Probab.*, 21(6):2109–2145, 2011.
[12] R. Douc and E. Moulines. Limit theorems for weighted samples with applications to sequential Monte Carlo methods. *Ann. Statist.*, 36(5):2344–2376, 2008.
[13] R. Douc, E. Moulines, and J. Olsson. Long-term stability of sequential Monte Carlo methods under verifiable conditions. *Ann. Appl. Probab.*, 24(5):1767–1802, 2014.
[14] R. Douc, E. Moulines, and D. Stoffer. *Nonlinear Time Series: Theory, Methods and Applications with R Examples*. Chapman & Hall/CRC Texts in Statistical Science, 2014.
[15] A. Doucet, N. De Freitas, and N. Gordon, editors. *Sequential Monte Carlo Methods in Practice*. Springer, New York, 2001.
[16] A. Doucet, S. Godsill, and C. Andrieu. On sequential Monte-Carlo sampling methods for Bayesian filtering. *Stat. Comput.*, 10:197–208, 2000.
[17] A. Doucet and A. M. Johansen. A tutorial on particle filtering and smoothing: fifteen years later. *Oxford handbook of nonlinear filtering*, 2009.
[18] C. Dubarry and S. Le Corff. Non-asymptotic deviation inequalities for smoothed additive functionals in nonlinear state-space models. *Bernoulli*, 19(5B):2222–2249, 2013.
[19] P. Fearnhead, D. Wyncoll, and J. Tawn. A sequential smoothing algorithm with linear computational cost. *Biometrika*, 97(2):447–464, 2010.
[20] S. J. Godsill, A. Doucet, and M. West. Monte Carlo smoothing for non-linear time series. *J. Am. Statist. Assoc.*, 50:438–449, 2004.
[21] N. Gordon, D. Salmond, and A. F. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proc. F, Radar Signal Process.*, 140:107–113, 1993.
[22] J. Hull and A. White. The pricing of options on assets with stochastic volatilities. *J. Finance*, 42:281–300, 1987.
[23] M. Hürzeler and H. R. Künsch. Monte Carlo approximations for general state-space models. *J. Comput. Graph. Statist.*, 7:175–193, 1998.
[24] P. E. Jacob, L. M. Murray, and S. Rubenthaler. Path storage in the particle filter. *Statistics and Computing*, pages 1–10, 2013.
[25] G. Kitagawa. Monte-Carlo filter and smoother for non-Gaussian nonlinear state space models. *J. Comput. Graph. Statist.*, 1:1–25, 1996.
[26] G. Kitagawa and S. Sato. Monte Carlo smoothing and self-organising state-space model. In *Sequential Monte Carlo methods in practice*, Stat. Eng. Inf. Sci., pages 177–195. Springer, New York, 2001.
[27] T. Koski. *Hidden Markov Models for Bioinformatics*. Kluwer, 2001.
[28] F. Lindsten and T. B. Schön. Backward simulation methods for Monte Carlo statistical inference. *Foundations and Trends in Machine Learning*, 6(1):1–143, 2013.
[29] G. Mongillo and S. Denève. Online learning with hidden Markov models. *Neural Computation*, 20(7):1706–1716, 2008.
[30] J. Olsson, O. Cappé, R. Douc, and E. Moulines. Sequential Monte Carlo smoothing with application to parameter estimation in non-linear state space models. *Bernoulli*, 14(1):155–179, 2008. arXiv:math.ST/0609514.
[31] J. Olsson and J. Ströjby. Convergence of random weight particle filters. Technical report, Lund University, 2010.
[32] J. Olsson and J. Ströjby. Particle-based likelihood inference in partially observed diffusion

processes using generalised Poisson estimators. *Electron. J. Statist.*, 5:1090–1122, 2011.

[33] J. Olsson and J. Westerborn. Efficient particle-based online smoothing in general hidden Markov models. In *IEEE 2014 International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2014)*, 2014.

[34] M. K. Pitt and N. Shephard. Filtering via simulation: Auxiliary particle filters. *J. Am. Statist. Assoc.*, 94(446):590–599, 1999.

[35] L. R. Rabiner and B-H. Juang. *Fundamentals of Speech Recognition*. Prentice-Hall, 1993.

[36] H. Rauch, F. Tung, and C. Striebel. Maximum likelihood estimates of linear dynamic systems. *AIAA Journal*, 3(8):1445–1450, 1965.

[37] C. Vergé, P. Del Moral, E. Moulines, and J. Olsson. Convergence properties of weighted particle islands with application to the double bootstrap algorithm. Preprint, 2014.

Online Particle Smoothing with Application to Map-matching

Samuel Duffield Sumeetpal S. Singh
University of Cambridge

August 4, 2021

Abstract

We introduce a novel method for online smoothing in state-space models that utilises a fixed-lag approximation to overcome the well known issue of path degeneracy. Unlike classical fixed-lag techniques that only approximate certain marginals, we introduce an online resampling algorithm, called *particle stitching*, that converts these marginal samples into a full posterior approximation. We demonstrate the utility of our method in the context of map-matching, the task of inferring a vehicle's trajectory given a road network and noisy GPS observations. We develop a new state-space model for the difficult task of map-matching on dense, urban road networks.

1 Introduction

Widely used in modelling dynamical systems and time series, a state-space model is fully defined by the following distributions for the hidden $\{x_t\}_{t=0}^{\infty}$ and observed process $\{y_t\}_{t=0}^{\infty}$

$$\begin{aligned} &p(x_0), && \\ &p(x_t|x_{t-1}), && t = 1, 2, 3, \dots \\ &p(y_t|x_t), && t = 0, 1, 2, \dots \end{aligned}$$

The statistical inference goal of *smoothing* is the task of approximating the joint smoothing distribution (or posterior distribution)

$$p(x_{0:T}|y_{0:T}) \propto p(x_0|y_0) \prod_{t=1}^{T} p(x_t|x_{t-1}) p(y_t|x_t), \tag{1}$$

where $x_{0:T} = (x_0, \dots, x_T)$ are latent states to be inferred and $y_{0:T} = (y_0, \dots, y_T)$ are given observations.

For *online smoothing*, we have the additional requirement of being able to quickly and accurately update an approximation of $p(x_{0:T-1}|y_{0:T-1})$ to approximate $p(x_{0:T}|y_{0:T})$ in light of receiving a new observation y_T.

A popular and powerful approach to inference in generic state-space models is that of particle filtering/smoothing (or sequential Monte Carlo), see [1] for a recent, thorough review. Particle smoothers approximate the joint smoothing distribution with a collection of weighted (or unweighted) particles

$$p(x_{0:T}|y_{0:T}) \approx \sum_{i=1}^{N} w_T^{(i)} \delta_{x_{0:T}} \left(x_{0:T}^{(i)} \right),$$

as opposed to particle filters which are only asked to approximate the filtering marginals $p(x_T|y_{0:T})$. Existing online particle smoothing approximations either degenerate as the length of the state-space model increases [2, 3] or only target the smoothing marginals $p(x_t|y_{0:T})$ [4] rather than the joint smoothing distribution. More recently, efficient techniques have been developed to produce online approximations to expectations with respect to $p(x_{t-1}, x_t|y_{0:T})$ [5, 6]. However, these are integrand specific whereas a particle approximation to the joint smoothing distribution is significantly more useful as mathematical expectations can be calculated over a variety of functions defined over full trajectories, thus providing the user with complete flexibility.

In summary, our motivation is to develop an algorithm that simultaneously satisfies the following requirements

S. Duffield is supported by the UK Engineering and Physical Sciences Research Council (EPSRC) doctoral training award

Contact: sddd2@cam.ac.uk, sss40@cam.ac.uk

(R1) **Joint Smoothing**: The algorithm efficiently approximates $p(x_{0:T}|y_{0:T})$ rather than only marginals.

(R2) **Online**: Our approximation can be quickly updated as it receives new observations.

(R3) **Non-degenerate**: The algorithm avoids the path degeneracy of classical particle filters.

The application of particle smoothing to urban map-matching is particularly motivating. The combination of dense road networks (with frequent intersections) and noisy GPS observations leads to uncertainty over the location and route of vehicles. This is a compelling application of online joint smoothing where it is desirable to represent the route with a diverse collection of particles, where each particles describes a plausible vehicle trajectory.

The contribution of this work is both methodological and applied.

- We develop a new online resampling method called *particle stitching* that converts marginal fixed-lag samples, which are non-degenerate, into a full posterior approximation (1). In doing so we jointly satisfy (R1)-(R3) using particle smoothing.
- We present a new state-space model for map-matching in dense, urban road networks. We demonstrate the benefits of uncertainty quantification over popular optimisation-based approaches and show the performance of offline particle smoothing can be matched by the online particle smoothers introduced here.

The rest of the paper is structured as follows. In Section 2 we discuss related online and offline algorithms for particle smoothing, as summarised in Table 1 alongside those introduced in this paper. Section 3 describes how to combine blocked samples in a way that is invariant for a fixed-lag joint smoothing distribution and Section 4 introduces two efficient online methods for generating these blocked samples. Section 5 describes the problem of map-matching and Section 6 demonstrates numerically the benefits of uncertainty quantification as well as the performance of the introduced online particle smoothers and their sensitivity to key parameters. In Section 7 we conclude and discuss some potential extensions.

2 Background and Related Work

2.1 Particle Filtering and Path Degeneracy

A classical particle filter runs a single forward pass, updating particles at every observation. Each update consists of three steps: an optional *resample* step, a *propagation* step and a *weighting* step. These steps are described in Alg. 1.

The resampling operation converts a weighted sample into an unweighted sample that likely contains duplicates $\left\{x^{(i)}, w^{(i)}\right\}_{i=1}^{N} \to \left\{x^{(i)}, \frac{1}{N}\right\}_{i=1}^{N}$. The most common resampling method is multinomial sampling (with replacement)

$$\begin{aligned} \text{Sample} \quad & a^{(i)} \sim \text{Categorical}\left(\left\{w^{(i)}\right\}_{i=1}^{N}\right), \quad i = 1, \ldots, N, \\ \text{Set} \quad & x^{(i)} \leftarrow x^{(a^{(i)})} \quad w^{(i)} \leftarrow \frac{1}{N}, \qquad i = 1, \ldots, N, \end{aligned}$$

where $\text{Categorical}(\{w^{(i)}\}_{i=1}^{N}) = \sum_{i=1}^{N} w^{(i)}\delta(i)$ simply draws an index i from the index set $\{1, \ldots, N\}$ with probabilities $w^{(i)}$.

Although most commonly used only to approximate the filtering marginals $p(x_T|y_{0:T})$, the particle filter update as described in Alg. 1 does provide an asymptotically unbiased approximation to the full smoothing distribution $p(x_{0:T}|y_{0:T})$.

The reason that a classical particle filter is almost never used to approximate the smoothing distribution is due to *path degeneracy*. Path degeneracy occurs in state-space models with large T. For repeated particle filter updates, the early coordinates of particles (e.g. $x_0^{(i)}$) will only be altered in the resampling step. Resampling can only decrease particle diversity (the number of distinct particles) and therefore as T increases, and resampling has occurred with suitable frequency, the particle approximation of $p(x_0|y_{0:T})$ will eventually collapse to just a single particle duplicated N times.

Algorithm 1 Particle Filter

0: Input weighted sample $\{x_{0:T-1}^{(i)}, w_{T-1}^{(i)}\}_{i=1}^{N}$ approximating $p(x_{0:T-1}|y_{0:T-1})$ and new observation y_T.

1: Resample (optional)

$$\left\{x_{0:T-1}^{(i)}, w_{T-1}^{(i)}\right\}_{i=1}^{N} \rightarrow \left\{x_{0:T-1}^{(i)}, \frac{1}{N}\right\}_{i=1}^{N}$$

2: Propagate by sampling from some proposal distribution

$$x_T^{(i)} \sim q\left(x_T \left| x_{T-1}^{(i)}, y_T\right.\right), \qquad i = 1, \ldots, N.$$

Append to particle $x_{0:T}^{(i)} = \left(x_{0:T-1}^{(i)}, x_T^{(i)}\right)$.

3: Weight and normalise

$$w_T^{(i)} \propto \frac{p\left(x_T^{(i)} \left| x_{T-1}^{(i)}\right.\right) p\left(y_T \left| \tilde{x}_T^{(i)}\right.\right)}{q\left(x_T^{(i)} \left| x_{T-1}^{(i)}, y_T\right.\right)} w_{T-1}^{(i)} \qquad i = 1, \ldots, N.$$

4: Output weighted sample $\{x_{0:T}^{(i)}, w_T^{(i)}\}_{i=1}^{N}$ approximating $p(x_{0:T}|y_{0:T})$

Due to the optional nature of the resampling step, adaptive schemes are often applied, with a popular choice to be to only resample if the *effective sample size*, $\mathrm{ESS}(\{w^{(i)}\}_{i=1}^{N}) = 1/\sum_{i=1}^{N} w^{(i)\,2}$, falls below some threshold. In addition to criteria such as the effective sample size that ensure resampling is only applied when necessary, a variety of techniques have been developed to mitigate the effects of path degeneracy. These include adaptively increasing the number of particles [7] as well as more sophisticated resampling schemes [8–10] which aim to only resample particles with negligible weights. These approaches can improve performance over multinomial resampling however still suffer the collapse in the approximation of $p(x_0|y_{0:T})$ for T sufficiently large.

The merits of resampling are well known, in particular resampling is vital in ensuring a diverse particle approximation to the filtering marginals $p(x_T|y_{0:T})$ - often the particle filter's primary task. Indeed, a particle filter without resampling is simply importance sampling on an a space whose dimension increases with every new observation. As a result, the divergence between importance and target distribution increases and the number of particles must increase exponentially [11] - the addition of resampling means a stable approximation to the filtering marginals is maintained.

2.2 Marginal Fixed-Lag

An alternative approach in mitigating the path degeneracy induced by repeated resampling is to simply stop resampling the early coordinates of particles, as proposed in [4]. That is, replace the resample step (Alg. 1, step 1) with the scheme described in Alg. 2.

Algorithm 2 Marginal Fixed-lag Resampling (for $T > L$)

1: Fix $\{x_{0:T-L-1}^{(i)}\}_{i=1}^{N}$

2: Resample only recent coordinates

$$\left\{x_{T-L:T}^{(i)}, w_T^{(i)}\right\}_{i=1}^{N} \rightarrow \left\{x_{T-L:T}^{(i)}, \frac{1}{N}\right\}_{i=1}^{N}$$

Stitch arbitrarily $x_{0:T}^{(i)} = (x_{0:T-L-1}^{(i)}, x_{T-L:T}^{(i)})$.

The justification for freezing $x_{0:T-L-1}^{(i)}$ is that after a certain lag L the smoothing distribution of early coordinates become (approximately) independent of new observations

$$p(x_{0:t}|y_{0:T}) \approx p(x_{0:t}|y_{0:\min(t+L,T)}), \tag{2}$$

	Joint Smoothing	Online	Path Degeneracy	Fixed-lag Approx.	Complexity
Particle Filter	✓	✓	✓		N
Marginal Fixed-lag [4]		✓	For large L	✓	N
Forward Filtering-Backward Smoothing [12]					N^2
Forward Filtering-Backward Simulation [13]	✓				N^2 N with RS†
Forward Smoothing for Additive Functionals [5]		✓			N^2
PaRIS [6]		✓			N with RS†
Block Sampling [14]	✓	✓	✓		LN
Online Particle Smoother (Alg. 4)	✓	✓	For large L	✓	N^2 N with RS†
Online Particle Smoother with Backward Simulation (Alg. 5)	✓	✓		✓	LN^2 LN with RS†

Table 1: Comparison of particle smoothing algorithms, for number of particles N and fixed-lag parameter L.
†For these algorithms the rejection sampling technique of [15] can be applied to obtain linear complexity when a bound for the transition density $p(x_t|x_{t-1})$ is available.

this represents a *fixed-lag approximation*.

The major issue with the fixed-lag resampling scheme described in Alg. 2 is that early and recent coordinates of particles are *arbitrarily stitched* together and therefore only provide a particle approximation to the *fixed-lag marginal* smoothing distribution:

$$p^L_{\text{marg}}(x_{0:T}|y_{0:T}) = \prod_{t=0}^{T-L-1} p(x_t|y_{0:t+L})p(x_{T-L:T}|x_{T-L-1}, y_{T-L:T}). \tag{3}$$

As such we lose the ability to take expectations over the joint distribution of early coordinates $x_{0:T-L-1}$.

A more useful particle approximation targets the *fixed-lag joint* smoothing distribution

$$p^L(x_{0:T}|y_{0:T}) = p(x_0|y_{0:L}) \prod_{t=1}^{T-L-1} p(x_t|x_{t-1}, y_{t:t+L})p(x_{T-L:T}|x_{T-L-1}, y_{T-L:T}), \tag{4}$$

which permits expectations over full trajectories $x_{0:T}$.

2.3 Offline Smoothing

A popular method for approximating the smoothing distribution $p(x_{0:T}|y_{0:T})$ is that of *forward filtering-backward smoothing* [12]. This method stores the output of each particle filter update $\{x_t^{(i)}\}_{i=1}^N \sim p(x_t|y_{0:t})$ and runs a full backward pass that updates the weights based on the backward decomposition

$$p(x_{t-1}|x_t, y_{0:t-1}) = \frac{p(x_t|x_{t-1})p(x_{t-1}|y_{0:t-1})}{p(x_t|y_{0:t-1})}, \qquad t = T, \dots, 1. \tag{5}$$

Similarly, *forward filtering-backward simulation* (FFBSi) [13] runs a full backward pass based on the same decomposition, but samples a new ancestor from the particle cloud (according to (5)) rather than only updating the weights and thus targets the joint smoothing distribution. Additionally, [15] showed that rejection sampling can significantly reduce the complexity of this pass in the case that $p(x_t|x_{t-1})$ has a tractable upper bound. An implementation of backward simulation is described in, Alg. 6.

Both of these algorithms require a full backward pass in light of every new observation and therefore are not suitable for online smoothing.

2.4 Online Smoothing

It was noted in [5] that the forward filtering-backward smoothing algorithm can be implemented in a single forward pass in the case of *additive functionals*, i.e. expectations of the form

$$\mathbb{E}_{p(x_{0:T}|y_{0:T})}\left[\sum_{t=0}^{T-1} g_t(x_t, x_{t+1})\right]. \tag{6}$$

Computing such expectations is useful for calibrating the state-space model parameters, see [3,16,17]. The PaRIS algorithm [6] combines this technique with rejection sampling to implement an efficient and cheap version of forward filtering-backward simulation in a single forward pass (for additive functionals).

Although it permits the implementation of these online algorithms, the requirement of additive functionals is very restrictive. Additionally, the forward only technique is *function specific*, i.e. it directly updates an approximation to the expected value of the additive functional. Our approach induces a controllable bias through the fixed-lag approximation but can approximate any expectation $\mathbb{E}_{p(x_{0:T}|y_{0:T})}[f(x_{0:T})]$ over the joint smoothing distribution and is therefore significantly more general than the marginal or additive functional approaches.

Block sampling [14] is an online method that targets the joint smoothing distribution. Every time a new observation is received, the block sampling scheme discards the most recent coordinates (within lag L) and re-proposes from an enlarged proposal distribution based on many more recent observations. Although this scheme does make use of a fixed-lag parameter, the weights and resampling still act on the full joint smoothing distribution. By moving a larger proportion of the trajectories, block sampling can (when combined with adaptive resampling schemes) mitigate but not avoid path degeneracy as resampling still takes place over full trajectories. Our proposed method builds on the block sampling approach by proposing blocks with a single coordinate overlap and then resampling in a way that targets the fixed-lag joint smoothing distribution (4).

3 Fixed-lag Stitching

In this section, we first detail the technique underlying backward simulation and then describe how the same technique can be applied in a forward implementation under a fixed-lag approximation.

3.1 Backward Simulation

In a single iteration of backward simulation, we have samples

$$\left\{\tilde{x}_{t-1}^{(i)}, \tilde{w}_{t-1}^{(i)}\right\}_{i=1}^{N} \text{ approximating } p(x_{t-1}|y_{0:t-1}),$$
$$\left\{x_t^{(j)}\right\}_{j=1}^{N} \text{ approximating } p(x_t|y_{0:T}),$$

but desire samples from the joint

$$\left\{(x_{t-1}^{(j)}, x_t^{(j)})\right\}_{j=1}^{N} \text{ approximating } p(x_{t-1}, x_t|y_{0:T}).$$

Here, and in the next sections, we have used tildes to represent the particles from which we look to resample - particles without tildes remain fixed. Seeing as $\{x_t^{(j)}\}_{j=1}^N$ already have the correct marginals this amounts to sampling

$$x_{t-1}^{(j)} \sim p(x_{t-1}|x_t^{(j)}, y_{0:t-1}) \qquad \text{for } j = 1, \ldots, N.$$

We cannot sample from $p(x_{t-1}|x_t^{(j)}, y_{0:t-1})$ directly, so we consider the decomposition

$$p(x_{t-1}|x_t^{(j)}, y_{0:t-1}) = \frac{p(x_t^{(j)}|x_{t-1})p(x_{t-1}|y_{0:t-1})}{\int p(x_t^{(j)}|x_{t-1})p(x_{t-1}|y_{0:t-1})dx_{t-1}}.$$

and then use $p(x_{t-1}|y_{0:t-1})$ to form an importance sampling approximation. Specifically, we use the particle filter's empirical approximation in the numerator and denominator to obtain the self-normalised weights [5, 13]

$$w_{t-1}^{(i\leftarrow j)} = \frac{\tilde{w}_{t-1}^{(i)} p(x_t^{(j)}|\tilde{x}_{t-1}^{(i)})}{\sum_{k=1}^{N} \tilde{w}_{t-1}^{(k)} p(x_t^{(j)}|\tilde{x}_{t-1}^{(k)})}.$$

where $\sum_{k=1}^{N} \tilde{w}_{t-1}^{(k)} p(x_t^{(j)}|\tilde{x}_{t-1}^{(k)})$ is an asymptotically unbiased approximation of $p(x_t^{(j)}|y_{0:t-1})$ and now $\sum_{i=1}^{N} w_{t-1}^{(i\leftarrow j)} = 1$.

Sampling from the empirical distribution $\sum_{i=1}^{N} w_{t-1}^{(i\leftarrow j)} \delta_{x_{t-1}}(\tilde{x}_{t-1}^{(i)})$ directly for each j is the $O(N^2)$ backward simulation technique [13], which when iterated for $t = T, \ldots, 1$ is asymptotically unbiased for the joint smoothing distribution $p(x_{0:T}|y_{0:T})$.

3.2 Fixed-lag Forward Simulation - Intractable

In this section we aim to approximate the fixed-lag joint smoothing distribution defined in (4) during the forward pass of particle filtering thus obtaining an online algorithm. In the setting of fixed-lag forward simulation we have

$$\left\{x_{t-1}^{(i)}\right\}_{i=1}^{N} \text{ approximating } p(x_{t-1}|y_{0:T-1}),$$
$$\left\{\tilde{x}_t^{(j)}, \tilde{w}_t^{(j)}\right\}_{j=1}^{N} \text{ approximating } p(x_t|y_{0:T}),$$

where $t = T - L$. Recall the fixed-lag approximation uses all past, present and at most L future observations to infer x_{t-1}, which implies $p(x_{t-1}|y_{0:T-1}) \approx p(x_{t-1}|y_{0:T})$.

We desire unweighted samples from the joint

$$\left\{\left(x_{t-1}^{(i)}, x_t^{(i)}\right)\right\}_{i=1}^{N} \text{ approximating } p^L(x_{t-1}, x_t|y_{0:T}),$$

where $p^L(x_{t-1}, x_t|y_{0:T}) = p(x_{t-1}|y_{0:T-1})p(x_t|x_{t-1}, y_{t:T})$. As $\left\{x_{t-1}^{(i)}\right\}_{i=1}^{N}$ have the correct marginal, obtaining the desired joint samples amounts to sampling

$$x_t^{(i)} \sim p(x_t|x_{t-1}^{(i)}, y_{t:T}) \qquad \text{for } i = 1, \ldots, N.$$

We cannot sample this conditional density directly and (similarly to backward simulation) employ importance sampling to approximate it.

In order to use the empirical approximation of $p(x_t|y_{0:T})$ as the sampling density (for an online implementation) we need to write

$$p(x_t|x_{t-1}^{(i)}, y_{t:T}) = \frac{h_t(x_{t-1}^{(i)}, x_t)p(x_t|y_{0:T})}{\int h_t(x_{t-1}^{(i)}, x_t)p(x_t|y_{0:T})dx_t},$$

for some suitably defined non-negative and integrable function $h_t(x_{t-1}^{(i)}, x_t)$. We then use the set of samples $\left\{\tilde{x}_t^{(j)}, \tilde{w}_t^{(j)}\right\}_{j=1}^{N}$ in the numerator and also to approximate the denominator to obtain a set of normalised weights

$$w_t^{(i\to j)} = \frac{\tilde{w}_t^{(j)} h_t(x_{t-1}^{(i)}, \tilde{x}_t^{(j)})}{\sum_{k=1}^{N} \tilde{w}_t^{(k)} h_t(x_{t-1}^{(i)}, \tilde{x}_t^{(k)})}.$$

To find h_t,

$$\begin{aligned} p(x_t|x_{t-1}^{(i)}, y_{t:T}) &= p(x_t|x_{t-1}^{(i)}, y_{0:T}) \\ &= \frac{p(y_{t:T}|x_t)p(x_t|x_{t-1}^{(i)})}{p(y_{t:T}|x_{t-1}^{(i)})}, \\ &= \frac{p(y_{t:T}|x_t)p(x_t|x_{t-1}^{(i)})}{p(y_{t:T}|x_{t-1}^{(i)})p(x_t|y_{0:T})} p(x_t|y_{0:T}). \end{aligned}$$

where in the last line we have multiplied and divided by $p(x_t|y_{0:T})$. A further application of Bayes' theorem to $p(x_t|y_{0:T})$ gives us

$$p(x_t|x_{t-1}^{(i)}, y_{t:T}) = \frac{p(y_{t:T}|y_{0:t-1})p(x_t|x_{t-1}^{(i)})}{p(y_{t:T}|x_{t-1}^{(i)})p(x_t|y_{0:t-1})} p(x_t|y_{0:T}).$$

Since $p(x_t|x_{t-1}^{(i)}, y_{t:T})$ integrates to 1, h_t is clearly

$$h_t(x_{t-1}^{(i)}, x_t) = \frac{p(x_t|x_{t-1}^{(i)})}{p(x_t|y_{0:t-1})}$$

which is where we stop as the density $p(x_t|y_{0:t-1})$ is not tractable (note that we could replace $p(x_t|y_{0:t-1})$ with an empirical approximation using marginal filtering particles but the resulting algorithm would have a prohibitive $O(N^3)$ complexity).

3.3 Fixed-lag Forward Simulation - Tractable

In the previous section, the $h_t(x_{t-1}^{(i)}, x_t)$ was intractable. We address this problem by defining a new conditional density

$$\pi(x_t, x_{t-1}|x_{t-1}^{(i)}, y_{0:T}) = p(x_t|x_{t-1}^{(i)}, y_{t:T})\lambda(x_{t-1}|x_{t-1}^{(i)}, x_t)$$

which trivially admits $p(x_t|x_{t-1}^{(i)}, y_{t:T})$ as the marginal. The potential dependency of λ on $y_{0:T}$ is implicit. The conditional density λ is to be chosen and we show how to make this choice so that when using $p(x_{t-1}, x_t|y_{0:T})$ within an importance sampling approximation of π, the weight is now tractable. In keeping with the notation of the previous section,

$$\left\{(\tilde{x}_{t-1}^{(j)}, \tilde{x}_t^{(j)}), \tilde{w}_t^{(j)}\right\}_{j=1}^N \text{ approximates } p(x_{t-1}, x_t|y_{0:T}),$$

where we have used tildes to represent the particles from which we look to resample - non-tilde particles are fixed. We remark that this idea of resolving the intractability of weights via sampling a higher dimensional density is inspired by the work of [14] for blocked resampling of the path-space particle approximations.

Proposition 1. *Let* $\lambda(x_{t-1}|x_{t-1}^{(i)}, x_t) = p(x_{t-1}|y_{0:t-1})$ *and* $H(x_{t-1}^{(i)}, x_{t-1}, x_t) = p(x_t|x_{t-1}^{(i)})/p(x_t|x_{t-1})$ *then*

$$\pi(x_t, x_{t-1}|x_{t-1}^{(i)}, y_{0:T}) = \frac{H(x_{t-1}^{(i)}, x_{t-1}, x_t)p(x_t, x_{t-1}|y_{0:T})}{\int H(x_{t-1}^{(i)}, x_{t-1}, x_t)p(x_t, x_{t-1}|y_{0:T})dx_{t-1:t}}.$$

The self-normalised approximation of $p(x_t|x_{t-1}^{(i)}, y_{0:T})$ *is* $\left\{\tilde{x}_t^{(j)}, \tilde{w}_t^{(i\to j)}\right\}$ *where*

$$w_t^{(i\to j)} = \tilde{w}_t^{(j)} \frac{p(\tilde{x}_t^{(j)}|x_{t-1}^{(i)})}{p(\tilde{x}_t^{(j)}|\tilde{x}_{t-1}^{(j)})} \left(\sum_{k=1}^N \tilde{w}_t^{(k)} \frac{p(\tilde{x}_t^{(k)}|x_{t-1}^{(i)})}{p(\tilde{x}_t^{(k)}|\tilde{x}_{t-1}^{(k)})}\right)^{-1}. \tag{7}$$

The interpretation of this proposition for forward smoothing is as follows: we can now sample from this approximation of π directly. Discarding the sampled x_{t-1} leaves samples $(x_{t-1}^{(i)}, x_t^{(i)})$ from the desired joint $p^L(x_{t-1}, x_t|y_{0:T})$. Repeating this for each i results in an $O(N^2)$ algorithm that is asymptotically unbiased for the fixed-lag joint distribution.

Proof. Expanding the density π yields

$$\begin{aligned}\pi(x_t, x_{t-1}|x_{t-1}^{(i)}, y_{0:T}) &= p(x_t|x_{t-1}^{(i)}, y_{t:T})\lambda(x_{t-1}|x_{t-1}^{(i)}, x_t), \\ &= \frac{p(y_{t:T}|x_t)p(x_t|x_{t-1}^{(i)})}{p(y_{t:T}|x_{t-1}^{(i)})}\lambda(x_{t-1}|x_{t-1}^{(i)}, x_t).\end{aligned}$$

Dividing by the sampling distribution $p(x_{t-1}, x_t|y_{0:T}) = p(x_{t-1}|y_{0:T})p(x_t|x_{t-1}, y_{t:T})$ gives

$$\pi(x_t, x_{t-1}|x_{t-1}^{(i)}, y_{0:T})/p(x_{t-1}, x_t|y_{0:T}) = \frac{p(y_{t:T}|x_t)p(x_t|x_{t-1}^{(i)})}{p(y_{t:T}|x_{t-1}^{(i)})p(x_{t-1}, x_t|y_{0:T})}\lambda(x_{t-1}|x_{t-1}^{(i)}, x_t).$$

Bayes' theorem on $p(x_{t-1}, x_t|y_{0:T})$ gives

$$\pi(x_t, x_{t-1}|x_{t-1}^{(i)}, y_{0:T})/p(x_{t-1}, x_t|y_{0:T}) = \frac{p(x_t|x_{t-1}^{(i)})p(y_{t:T}|y_{0:t-1})}{p(x_t|x_{t-1})p(y_{t:T}|x_{t-1}^{(i)})p(x_{t-1}|y_{0:t-1})}\lambda(x_{t-1}|x_{t-1}^{(i)}, x_t).$$

We now observe that the choice of $\lambda(x_{t-1}|x_{t-1}^{(i)}, x_t,) = p(x_{t-1}|y_{0:t-1})$ will make all terms involving (x_{t-1}, x_t) tractable.

$$\pi(x_t, x_{t-1}|x_{t-1}^{(i)}, y_{0:T}) = \frac{p(x_t|x_{t-1}^{(i)})p(y_{t:T}|y_{0:t-1})}{p(x_t|x_{t-1})p(y_{t:T}|x_{t-1}^{(i)})} p(x_{t-1}, x_t|y_{0:T}).$$

□

Algorithm 3 Fixed-lag Particle Stitching

Input unweighted sample $\{x_{0:T-L-1}^{(i)}\}_{i=1}^N$, weighted sample $\{\tilde{x}_{T-L-1:T}^{(j)}, \tilde{w}_T^{(j)}\}_{j=1}^N$, bound $\rho \geq p(x_{T-L}|x_{T-L-1})$ and the maximum number of rejections to attempt R.
Calculate the non-interacting stitching weights and normalise in j

$$\hat{w}_T^{(j)} \propto \frac{1}{p(\tilde{x}_{T-L}^{(j)}|\tilde{x}_{T-L-1}^{(j)})} \tilde{w}_T^{(j)} \qquad j = 1, \ldots, N.$$

for $i = 1, \ldots, N$ **do**
  for $r = 1, \ldots, R$ **do**
    Sample $c^* \sim$ Categorical $\left(\left\{\hat{w}_T^{(j)}\right\}_{j=1}^N\right)$
    Sample $u \sim U(0, 1)$
    if $u < p(\tilde{x}_{T-L}^{(c^*)}|x_{T-L-1}^{(i)})/\rho$ **then**
      Accept c^*
      break
    end if
  end for
  if a sample c^* was accepted **then**
    Set $x_{T-L:T}^{(i)} = \tilde{x}_{T-L:T}^{(c^*)}$
  else
    Calculate the stitching weights and normalise in j

$$w_T^{(i\to j)} \propto \frac{p(\tilde{x}_{T-L}^{(j)}|x_{T-L-1}^{(i)})}{p(\tilde{x}_{T-L}^{(j)}|\tilde{x}_{T-L-1}^{(j)})} \tilde{w}_T^{(j)} \qquad j = 1, \ldots, N.$$

    Sample $c_i \sim$ Categorical $\left(\left\{w_T^{(i\to j)}\right\}_{j=1}^N\right)$
    Set $x_{T-L:T}^{(i)} = \tilde{x}_{T-L:T}^{(c_i)}$
  end if
end for
Output unweighted sample $\left\{x_{0:T}^{(i)}\right\}_{i=1}^N$ approximating $p^L(x_{0:T}|y_{0:T})$.

In (7), we have described above an empirical approximation that stitches together particles from $p(x_{T-L-1}|y_{0:T-1})$ with those from $p(x_{T-L-1:T-L}|y_{0:T})$. By the conditional independence structure of state-space models this is equivalent to stitching together blocks from $p^L(x_{0:T-L-1}|y_{0:T-1})$ with those from $p(x_{T-L-1:T}|y_{0:T})$ - assuming we can sample from $p(x_{T-L-1:T}|y_{0:T})$.

Sampling from $p(x_{T-L-1:T}|y_{0:T})$ is not directly possible for non-trivial state-space models. In Section 4 we describe two efficient methods for block sampling by recycling previously generated particles (Alg. 4 and Alg. 5).

The fixed-lag stitching described provides a method to, in principal, approximate the full smoothing distribution $p(x_{0:T}|y_{0:T})$. It does so by using very separate tools to the forward-only techniques in [5,6], these techniques do not require any stitching and directly update an expectation over additive functionals. Our method generates a collection of particles approximating the joint smoothing distribution and is therefore far more general. We have utilised the block sampling framework from [14], in particular through the balancing distribution λ. However, the fixed-lag stitching is novel and effectively uses a fixed-lag approximation to implement a forward implementation of the technique underlying backward simulation [13]. This is in contrast to [14], where particles are reweighted and resampled over full trajectories and therefore still suffer from path degeneracy.

3.4 Rejection Sampling

Sampling from (7) can be done directly at a computational complexity of $O(N^2)$. However, when a bound for the transition density is available

$$\rho \geq p(x_{T-L}|x_{T-L-1}) \quad \forall x_{T-L-1}, x_{T-L}, \tag{8}$$

we can utilise the rejection sampling approach of [15] to avoid calculating the normalisation constants and bring the computational complexity down to $O(N)$. In practice, the rejection sampling is not always faster than the direct one. A pragmatic approach is the hybrid described in [18] where for each particle, up to $R < N$ samples are proposed to the rejection sampler, if all are subsequently rejected the direct scheme is applied, thus setting $R = 0$ recovers the direct scheme. This hybrid algorithm is described in Alg. 3.

4 Sampling from $p(x_{T-L-1:T}|y_{0:T})$

We now describe two methods for sampling coordinates $\tilde{x}_{T-L-1:T}$ in a way that is asymptotically unbiased for $p(x_{T-L-1:T}|y_{0:T})$, and can therefore be plugged into the fixed-lag stitching procedure.

4.1 Particle Filter

Recall the online setting where we have unweighted particles $\{x^{(i)}_{0:T-1}\}_{i=1}^N$ approximating $p^L(x_{0:T-1}|y_{0:T-1})$ and receive a new observation y_T.

Our first method is based on the fact that the particle approximation provided by a classical particle filter is asymptotically unbiased for the full joint smoothing distribution $p(x_{0:T}|y_{0:T})$. Although this approximation deteriorates due to path degeneracy it may still be sufficient for sampling the recent coordinates $x_{T-L-1:T}$.

Thus, we propose applying the classical particle filter proposal and weighting steps to $\{x^{(i)}_{0:T-1}\}_{i=1}^N$, generating weighted particles $\{x^{(i)}_{0:T}, \tilde{w}^{(i)}_T\}_{i=1}^N$, before splitting the trajectories

$$\left\{x^{(i)}_{0:T}, \tilde{w}^{(i)}_T\right\}_{i=1}^N \rightarrow \left\{x^{(i)}_{0:T-L-1}\right\}_{i=1}^N \text{ and } \left\{\tilde{x}^{(i)}_{T-L-1:T}, \tilde{w}^{(i)}_T\right\}_{i=1}^N,$$

where $\tilde{x}^{(i)}_{T-L-1:T} = x^{(i)}_{T-L-1:T}$. Under the fixed-lag approximation, $x_{0:T-L-1}$ is conditionally independent of y_T and therefore the new weights need not apply to these earlier coordinates. Whereas the $\{\tilde{x}^{(i)}_{T-L-1:T}, \tilde{w}^{(i)}_T\}_{i=1}^N$ are asymptotically unbiased for the desired sampling distribution $p(x_{T-L-1:T}|y_{0:T})$ and can therefore be plugged into the stitching procedure, Alg. 3. The coordinates x_{T-L-1} are duplicated to provide the overlap required for stitching.

Algorithm 4 Online Particle Smoother (for $T > L$)

0: Input unweighted smoothing sample $\{x^{(i)}_{0:T-1}\}_{i=1}^N$, new observation y_T and fixed-lag L.
1: Fix $\{x^{(i)}_{0:T-L-1}\}_{i=1}^N$
2: Execute particle filter propagate and weight steps, Alg. 1: steps 2:3

Generate $\{\tilde{x}^{(j)}_{T-L-1:T}, \tilde{w}^{(j)}_T\}_{j=1}^N$ from $\{x^{(j)}_{T-L-1:T-1}\}_{j=1}^N$ and y_T,

forming a weighted sample approximating $p(x_{T-L-1:T}|y_{0:T})$.
3: Stitch together

$$\{x^{(i)}_{0:T-L-1}\}_{i=1}^N \text{ and } \{\tilde{x}^{(j)}_{T-L-1:T}, \tilde{w}^{(j)}_T\}_{j=1}^N \rightarrow \{x^{(i)}_{0:T}\}_{i=1}^N.$$

using Alg. 3.
4: Output unweighted sample $\{x^{(i)}_{0:T}\}_{i=1}^N$ approximating $p^L(x_{0:T}|y_{0:T})$

The algorithm is described in Alg. 4, and ends up being a relatively simple modification to a classical particle filter where the resampling step is compulsory and altered to include the stitching probabilities in the weights (7).

When the transition bound (8) is available the complexity of the update remains $O(N)$ or $O(N^2)$ when the bound is unavailable.

4.2 Partial Backward Simulation

If the lag parameter L is chosen to be too large, the above mechanism will still suffer from path degeneracy in the same way a particle filter does. To remedy this we propose a partial run of backward simulation (Alg. 6) at each time step to rejuvenate the trajectories $x_{T-L-1:T}$. This technique is considered in [19] for generating samples from $p(x_{T-L:T}|y_{0:T})$ without the subsequent stitching.

This has the additional requirement of storing the marginal approximations $\{\tilde{x}_t^{(k)}, \tilde{w}_t^{(k)}\}_{k=1}^N$ for $t = T-L-1, \ldots, T$ from the particle filter, but permits the use of adaptive resampling and completely avoids path degeneracy.

The resulting algorithm, Alg. 5, has a complexity of $O(LN)$ per update if the transition bound (8) is available otherwise $O(LN^2)$.

Algorithm 5 Online Particle Smoother with Backward Simulation (for $T > L$)

0: Input unweighted smoothing sample $\{x_{0:T-1}^{(i)}\}_{i=1}^N$, weighted marginal filtering samples $\{\tilde{x}_t^{(k)}, \tilde{w}_t^{(k)}\}_{k=1}^N$ for $t = T-L-1, \ldots, T-1$, new observation y_T, and fixed-lag L.

1: Fix $\{x_{0:T-L-1}^{(i)}\}_{i=1}^N$

2: Execute particle filter update, Alg. 1, to generate the new marginal filtering sample

$$\text{Generate } \{\tilde{x}_T^{(k)}, \tilde{w}_T^{(k)}\}_{k=1}^N \quad \text{from } \{\tilde{x}_{T-1}^{(k)}, \tilde{w}_{T-1}^{(k)}\}_{k=1}^N \text{ and } y_T.$$

3: Run partial backward simulation, Alg. 6, on the weighted filtering samples

$$\{\tilde{x}_t^{(k)}, \tilde{w}_t^{(k)}\}_{k=1}^N, \text{ for } t = T, \ldots, T-L-1 \quad \rightarrow \{\tilde{x}_{T-L-1:T}^{(j)}\}_{j=1}^N.$$

forming an unweighted sample approximating $p(x_{T-L-1:T}|y_{0:T})$.

4: Stitch together

$$\{x_{0:T-L-1}^{(i)}\}_{i=1}^N \text{ and } \{\tilde{x}_{T-L-1:T}^{(j)}, \tfrac{1}{N}\}_{j=1}^N \rightarrow \{x_{0:T}^{(i)}\}_{i=1}^N.$$

using Alg. 3.

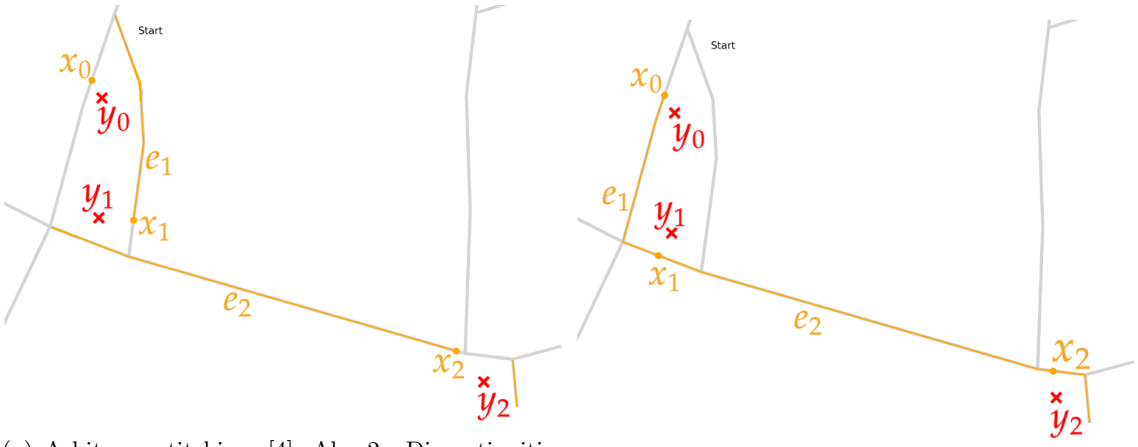
5: Output unweighted sample $\{x_{0:T}^{(i)}\}_{i=1}^N$ approximating $p^L(x_{0:T}|y_{0:T})$ and weighted filtering samples $\{\tilde{x}_T^{(k)}, \tilde{w}_T^{(k)}\}_{k=1}^N$, $t = T-L, \ldots, T$.

Both techniques to sample from $p(x_{T-L-1:T}|y_{0:T})$ utilise the output of a particle filter. Indeed, they are also applicable to alternative filtering techniques such as auxiliary particle filters [20] and the backward simulation approach is also applicable to filters that discard historic trajectories such as the marginal particle filter [21].

5 Map-Matching

Map-matching is the task of inferring the true trajectory of a vehicle given noisy GPS observations and a map of the road network. A road network is defined as a graph within $\mathbb{R}^2$, where intersections are represented by nodes (vertices) and roads (assumed to be single lanes and one-way) are represented by edges with a two-way road being represented by two edges. Some collections of nodes and edges are depicted in Figure 1, 3 and 5, with map-matched vehicle routes overlaid. The recorded GPS observations (see Figure 1) may lie outside the road whereas valid vehicle positions/trajectories to be inferred must strictly lie on the road. Following [22, 23] for urban road networks we infer the vehicle's position along an edge but not it's width within the road.

Applications of map-matching are wide-ranging and thus a general purpose algorithm is highly desirable. Existing probabilistic approaches to map-matching have mostly adopted the approach of [22],



(a) Arbitrary stitching, [4], Alg. 2. Discontinuities where the vehicle jumps, e.g. $e(x_0) \neq e_1^o$.

(b) Fixed-lag stitching, Alg. 3

Figure 1: Comparison of fixed-lag techniques for map-matching. Each image displays a single trajectory from the resulting particle approximation. Both techniques produce plausible marginals but arbitrary stitching fails to produce a continuous trajectory.

where each observation is snapped to the nearest point on any and all edges that fall within a truncation distance. These points then form a discrete hidden Markov model, on which the Viterbi algorithm can produce a single route of high probability. In contrast, our approach provides uncertainty quantification through a collection of particles, with each particle representing a possible route. Previous applications of particle filtering to map-matching [24,25] fail to tackle the problem of path degeneracy, with the exception of [23] who introduce the use of FFBSi for offline map-matching. Our formulation is similar to that of [23] but differs through the inclusion of a term in the transition density adapted from [22] that penalises non-direct routes (which is vital in dense urban road networks), as well as the use of the optimal proposal density (that takes into account the new observation) rather than simply the bootstrap proposal which will perform poorly for small GPS noise or dense road networks.

Traditional fixed-lag smoothing [4], Alg. 2 where particles are arbitrarily stitched together is not appropriate for map-matching, as seen in Figure 1. As these methods target the product of smoothing marginals in (3), arbitrary stitching cannot produce a continuous trajectory and therefore does not permit expectations over multiple observation times.

In the rest of this section, we describe our state-space model for map-matching, the optimal proposal distribution and the induced weights in the context of fixed-lag stitching. In the next section we present some results on synthetic and real data. An open source python package providing easy offline and online map-matching is provided [1] alongside code for all the simulations to follow.

5.1 Model Variables

We define a state-space model for a single vehicle's trajectory with the variables

- $e_t \subset \mathbb{N}$, a finite ordered set of edge labels that define a connected path. Each edge label corresponds to a unique one-way section of road. The variable e_t details the edges traversed (and in which order) between observation times $t-1$ and t, including the choices made at encountered intersections (nodes).
- $x_t \in \mathbb{R}^2$ the position of the vehicle at observation time. The variable x_t defines a Cartesian coordinate restricted to lie on the road network, specifically x_t lies on the final edge of the finite ordered set of edge labels e_t, for $t > 0$.
- $y_t \in \mathbb{R}^2$ noisy observation of the vehicle's position x_t, not restricted to the road network.

Note here the change of notation from the previous section, now x_t refers only to vehicle position and the full latent states are $x_0, (e_1, x_1), (e_2, x_2), \ldots, (e_T, x_T)$ as depicted in Figure 2.

We denote e_t^o for the first edge in the ordered set e_t and e_t^* for the final edge, thus $e_t = (e_t^o, \ldots, e_t^*)$. We use the notation $e(x_t)$ to denote the edge label on which x_t lies, so $e(x_{t-1}) = e_t^o$ and $e(x_t) = e_t^*$.

[1] `https://github.com/SamDuffield/bmm`

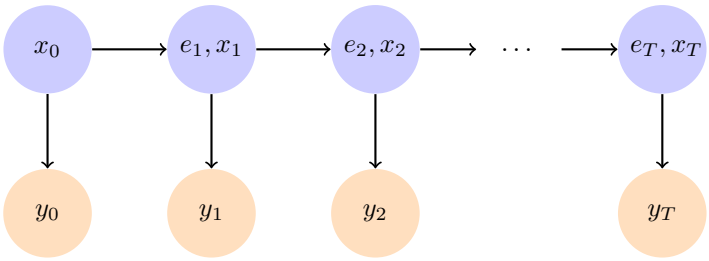


Figure 2: Conditional independence structure of the state-space model for map-matching.

5.2 Model Distributions

Our transition density can be written as

$$p(e_t, x_t|x_{t-1}) = \frac{\gamma(\|x_t - x_{t-1}\|_{e_t}) \exp(-\beta|\|x_t - x_{t-1}\|_{e_t} - \|x_t - x_{t-1}\||)}{Z(x_{t-1})}, \tag{9}$$

with normalising constant

$$Z(x_{t-1}) = \sum_{e_t} \int_{x_t} \gamma(\|x_t - x_{t-1}\|_{e_t}) \exp(-\beta|\|x_t - x_{t-1}\|_{e_t} - \|x_t - x_{t-1}\||) \mathrm{d}x_t. \tag{10}$$

The summation in (10) is taken over all possible series of edges starting at $e(x_{t-1})$. $\|x_t - x_{t-1}\|_{e_t}$ is the distance travelled between x_{t-1} and x_t along the series e_t (restricted to the road network) whereas $\|x_t - x_{t-1}\|$ is the *great circle* distance (not restricted to the road network).

Thus the following distributions fully define our state-space model:

- $\gamma(\|x_t - x_{t-1}\|_{e_t})$. Prior on distance travelled between observations - some simple analytical distribution on $\mathbb{R}^+$, penalising lengthy routes. We assume an exponential distribution with probability mass at 0 to represent the possibility of the vehicle remaining stationary (due to traffic lights, heavy traffic etc)

$$\gamma(d) = p^0 \mathbb{I}(d = 0) + (1-p^0)\mathbb{I}(d > 0)\lambda e^{-\lambda d}. \tag{11}$$

- $\exp(-\beta|\|x_t - x_{t-1}\|_{e_t} - \|x_t - x_{t-1}\||)$ adapted from [22], penalising non-direct (or winding) routes. Non-direct routes with lots of curvature will have a high discrepancy between the road distance travelled and great circle distance and thus will have a low probability under this term, reflecting a driver's preference to take short, direct routes where possible.

- $p(y_t|x_t) = \mathcal{N}(y_t|x_t, \sigma^2_{\text{GPS}} I_2)$. Isotropic Gaussian observation noise.

- We set $p(x_0)$ to be uniform on the road network. I.e. no prior information on the start of the trajectory other than constricting it to the road network (as with all inferred positions).

To make $p(x_0|y_0)$ tractable we define the initial observation density to be a truncated Gaussian:

$$p(y_0|x_0) \propto \mathcal{N}(y_0|x_0, \sigma^2_{\text{GPS}} I_2)\, \mathbb{I}(||y_0 - x_0|| < r_{\text{GPS}}),$$

giving

$$p(x_0|y_0) \propto \mathcal{N}(x_0|y_0, \sigma^2_{\text{GPS}} I_2)\, \mathbb{I}(||y_0 - x_0|| < r_{\text{GPS}}),$$

where x_0 is restricted to the road network but y_0 is not. In our simulations we set $r_{\text{GPS}} = 5\sigma_{\text{GPS}}$.

5.3 Optimal Proposal

The (locally) optimal proposal [12] for particle filtering combines the transition density (9) and the newly received observation y_T:

$$q^{\text{opt}}(x_T, e_T|x_{T-1}, y_T) \propto p(e_T, x_T|x_{T-1}) p(y_T|x_T). \tag{12}$$

The standard reweighting step of the particle filter update (Alg. 1) then becomes

$$w_T^{\text{opt}\ (i)} \propto p(y_T|x_{T-1}^{(i)}),$$

where

$$p(y_T|x_{T-1}) = \sum_{e_T} \int_{x_T} p(e_T, x_T|x_{T-1}) p(y_T|x_T) \, \mathrm{d}x_T, \tag{13}$$

is the normalisation constant of (12).

Neither sampling from the optimal proposal (12), nor evaluating the subsequent weights (13), nor evaluating the normalising constant of the transition density (10) are immediately tractable as we do not have closed form expressions for the edge geometries.

Instead, we opt to approximate the required integrals numerically by discretising the edges up to some maximal possible distance travelled $d_{\max}$. This numerical integration can be implemented efficiently across particles by caching route searches and likelihood evaluations, as many particles will typically lie on the same or adjacent edges.

5.4 Fixed-lag Stitching

In the context of the fixed-lag stitching described in Section 3, we propose stitching together each

$$(x^{(i)}_{0:T-L-1}, e^{(i)}_{1:T-L-1}) \text{ with a sample from } \left\{(\tilde{x}^{(j)}_{T-L-1:T}, \tilde{e}^{(j)}_{T-L:T})\right\}_{j=1}^{N}.$$

Thus the adjusted weights become

$$w_T^{(i\to j)} \propto \frac{p(\tilde{e}^{(j)}_{T-L}, \tilde{x}^{(j)}_{T-L}|x^{(i)}_{T-L-1})}{p(\tilde{e}^{(j)}_{T-L}, \tilde{x}^{(j)}_{T-L}|\tilde{x}^{(j)}_{T-L-1})} w_T^{(j)} \mathbb{I}\left(e(x^{(i)}_{T-L-1}) = \tilde{e}^{o\,(j)}_{T-L}\right).$$

Similarly, for the rejection sampling we get non-interacting weights

$$\hat{w}_T^{(j)} \propto \frac{w_T^{(j)}}{p(\tilde{e}^{(j)}_{T-L}, \tilde{x}^{(j)}_{T-L}|\tilde{x}^{(j)}_{T-L-1})},$$

and we accept a sample if $e(x^{(i)}_{T-L-1}) = \tilde{e}^{o\,(j)}_{T-L}$ and

$$\begin{aligned} u <& \gamma\left(\|\tilde{x}^{(j)}_{T-L} - x^{(i)}_{T-L-1}\|_{\tilde{e}^{(j)}_{T-L}}\right) \\ & \exp\left(-\beta\left|\|\tilde{x}^{(j)}_{T-L} - x^{(i)}_{T-L-1}\|_{\tilde{e}^{(j)}_{T-L}} - \|\tilde{x}^{(j)}_{T-L} - x^{(i)}_{T-L-1}\|\right|\right) \\ & \rho^{-1}, \end{aligned}$$

where $u \sim U(0,1)$ and we have a bound $\rho > \gamma(d)$ for any d. The availability of this bound depends on the choice of distribution for $\gamma(\cdot)$, in the case of (11) a bound is available: $\rho = \max((1-p^0)\lambda, p^0)$.

6 Simulations

We tuned the model hyperparameters using offline gradient expectation-maximisation [16] (running offline FFBSi for the E-step) over 20 routes from the Porto taxi dataset [26] where observations are 15 seconds apart, resulting in values of $p^0 = 0.14$, $\lambda = 0.07/15$, $\beta = 0.05$ and $\sigma_{\mathrm{GPS}} = 5.2$, all edges are discretised to a resolution of 1m.

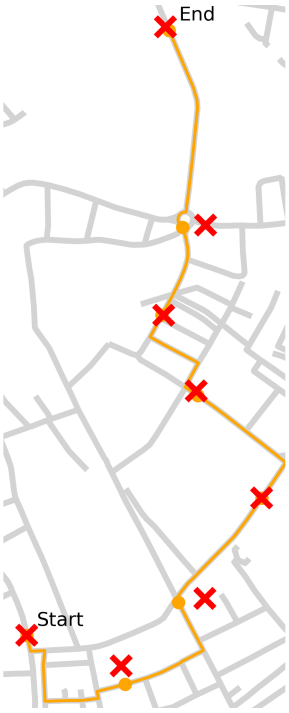
The true map-matching posterior is analytically intractable and instead we use the approach of [23] as our *gold standard* for benchmarking: offline forward filtering-backward simulation with a large $N = 1000$, i.e. Alg. 1 to generate the filtering marginals and Alg. 6 for the smoothing particles.

6.1 Synthetic Data

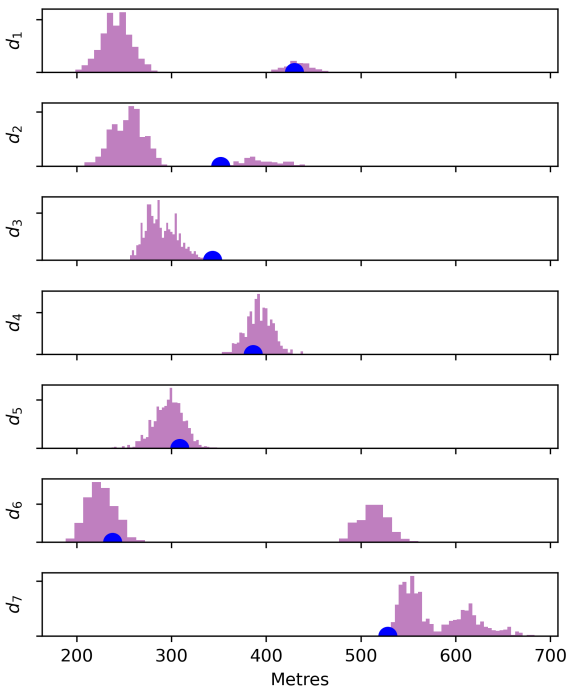
In Figure 3 we demonstrate the benefits of uncertainty quantification for offline map-matching by comparing our particle based gold standard (offline FFBSi) which is the approach of [23] against the popular optimisation approach of [22] on synthetic observations for a trip between the Cambridge Engineering department and the Fort St George pub.

Although the optimisation based approach finds a plausible route (point estimate) it misses out on others that are equally plausible and thus valuable information is lost, this is particularly prevalent when inferring the distances the vehicle travelled, Figure 3c. There is significant uncertainty in both

(a) Gold standard, offline FFBSi with $N = 1000$. All trajectories overlaid.

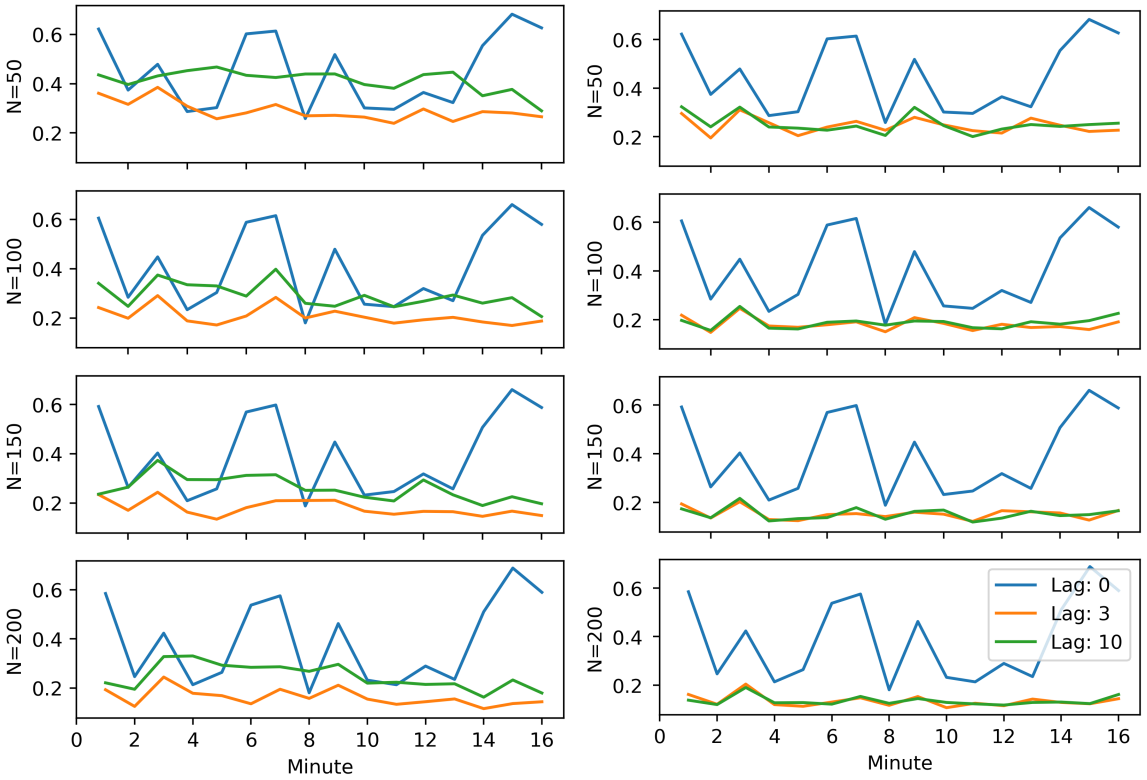


(b) Viterbi map-matching [22].



(c) Histograms represent $p(\|x_t - x_{t-1}\|_{e_t} | y_{0:T})$ from FFBSi. Spots represent distances inferred using Viterbi map-matching.

Figure 3: Offline particle smoothing vs Viterbi map-matching for synthetic trip across Cambridge.



(a) Online Particle Smoother

(b) with Backward Simulation.

Figure 4: Total variation distance from gold standard (offline FFBSi) for posterior over cumulative distance travelled in each minute of a 16 minute taxi route (observation every 15 seconds) [26]. Simulations averaged over 20 random seeds.

the edges traversed and the distances travelled that is captured by FFBSi but not by the Viterbi algorithm.

All particles generated by FFBSi represent a plausible route - more direct routes are preferred but not overly so - this provides evidence to suggest the model is well suited to difficult dense urban road networks and that the optimal proposal is efficiently generating high probability particles.

In the problem considered here and in [22, 23] there is no variability in the position of the vehicle along the width of the road - arguably more suitable for dense, urban networks. A less constrained inference problem in e.g. [27] would also infer the position along the width of the road. The model can then be formulated with Gaussian noise allowing Kalman filter or Rao-Blackwellisation techniques (that are not applicable to our state-space model) can be used to accelerate inference.

6.2 Real Data

We now explore the effects of the algorithmic parameters in the online particle smoothers (sample size, lag parameter and number of rejections) for map-matching a route from the Porto taxi dataset [26].

In Figure 4, we analyse the posterior distribution for the cumulative (road) distance travelled by the taxi each minute. As observations are received every 15 seconds this amounts to expectations averaged over blocks of the full smoothing distribution and thus marginal approximations (3) would be insufficient. For each minute of the trip, we compare particle approximations by calculating the total variation distance over the empirical distributions. This total variation distance is calculated by binning the distance travelled variable, then the empirical distributions are defined over a discrete space and the total variation distance is tractable

$$\mathrm{TV}\left(\{d_1^{(i)}\}_{i=1}^{N_1}, \{d_2^{(j)}\}_{j=1}^{N_2}\right) = \frac{1}{2}\sum_{[a,b]\in\mathcal{D}}\left|\frac{\sum_{i=1}^{N_1}\mathbb{I}[d_1^{(i)}\in[a,b]]}{N_1} - \frac{\sum_{j=1}^{N_2}\mathbb{I}[d_2^{(j)}\in[a,b]]}{N_2}\right|,$$

where $\mathcal{D}$ represents $[0,\infty)$ discretised into 5 metre width bins and the empirical distances are assumed to be unweighted (although easily adjusted to include weights).

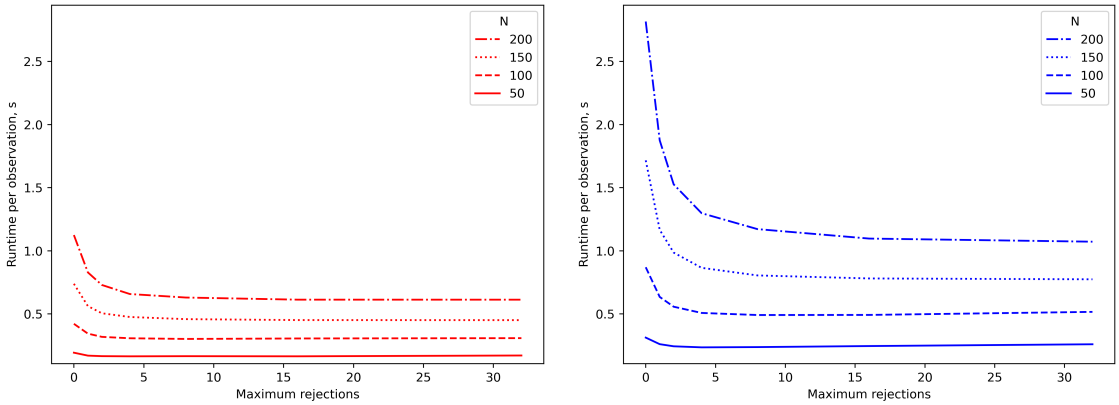
(a) Gold standard, offline FFBSi with $N = 1000$.

(b) Online particle smoother with $L = 0$ and $N = 200$. Poor approximation due to small lag.

(c) Online particle smoother, $L = 10$ and $N = 200$. Evidence of path degeneracy due to large lag.

(d) Online particle smoother with backward simulation, $L = 10$ and $N = 200$.

Figure 5: The start of a route from [26] with various particle approximations.



(a) Online Particle Smoother

(b) with Backward Simulation.

Figure 6: Algorithmic runtime vs number of rejections tolerated for $L = 3$. Runtimes averaged over 65 observations and 20 random seeds.

Figure 5 depicts the varying algorithmic performance on the start of the route.

We initially observe that setting $L = 0$ performs very poorly for all sample sizes. The overly small lag parameter results in a large deviation between the true smoothing distribution (1) and the joint fixed-lag smoothing distribution (4), Figure 5b.

The online particle smoother suffers from path degeneracy for the large lag $L = 10$ as observed by a loss of particle diversity (compared to the backward simulation techniques) in Figure 5c. It does however perform well for $L = 3$.

The addition of backward simulation completely avoids the issue of path degeneracy, but induces a bias by targeting $p^L(x_{0:T}|y_{0:T})$ rather than $p^L(x_{0:T}|y_{0:T})$, this bias is controllable through the choice of the lag parameter, L. We observe that increasing the lag parameter from $L = 3$ to $L = 10$ does little to improve performance and as such can posit that the distributions (1) and (4) are suitably close for $L = 3$.

We have not compared numerically to the marginal or additive functional based smoothers described in section 2 (such as PaRIS [6]) as by definition (6) they are not suitable to expectations over multiple observation times as analysed in Figure 4 and depicted in Figure 1.

A classical particle filter can be recovered by setting $L = \infty$ in the online particle smoother and would suffer path degeneracy to an even greater extent than the online particle smoother with $L = 10$.

Finally, in Figure 6 we compare the effect on algorithmic runtime from increasing the maximum number of rejections, R, attempted in the hybrid stitching scheme Alg. 3, as well as backward simulation if applicable. Recall that setting $R = 0$ recovers the full $O(N^2)$ scheme. For a suitably large number of rejections the runtimes of both algorithms can be seen to increase linearly in N.

7 Discussion

In this work, we have developed techniques to efficiently approximate the full joint smoothing distribution or rather the fixed-lag approximation to it in an online setting, this is highly desirable as it permits the online estimation of a range functions that are defined over full trajectories or any subset thereof. The online particle smoother (Alg. 4) comes at the same computational complexity as a classical particle filter, whereas the inclusion of backward simulation (Alg. 5) negates the issue of path degeneracy for more difficult models where a large lag parameter is required.

We formulated a state-space model that is specifically designed for urban map-matching and demonstrated the value of particle based uncertainty quantification versus established optimisation based approaches. We then showed that the performance of gold standard offline smoothing FFBSi can be obtained with the online particle smoothers.

The choice of the lag parameter L determines the distance between the distributions $p^L(x_{0:T}|y_{0:T})$ and $p(x_{0:T}|y_{0:T})$, naturally we desire this to be small and therefore L large. The question of how large is a difficult one as it is dependent on the mixing of the state-space model, as discussed in [6].

In practice, the lag parameter can be tuned through preliminary runs on offline or simulated data. This can be done by analysing the sensitivity of smoothing expectations of interest (model-specific)

to the value of the lag parameter L and comparing against the true underlying values (in the case of simulated data) or the equivalent expectation from FFBSi. This is investigated for map-matching in Figure 4. The tuning of L is perhaps easier for the online particle smoother with backward simulation. The backward simulation completely avoids the issue of path degeneracy and thus error only arises from Monte Carlo variance and the difference between $p^L(x_{0:T}|y_{0:T})$ and $p(x_{0:T}|y_{0:T})$.

An interesting extension would be to investigate the use of a variable lag parameter which is chosen dynamically, as achieved for a function specific version of the marginal fixed-lag particle filter in [28].

It would be desirable to obtain theoretical results bounding the error induced by the introduced online particle smoothers. We leave this for future work as we anticipate the analysis to be somewhat intricate - combining the work on central limit theorems for particle smoothing such as [5,15] and the bias induced by a fixed-lag approximation [2]. As well as quantifying the impact of using the tractable weights with overlapping coordinate from 3.3 as opposed to the optimal but intractable weights in 3.2.

The realisation of a low-probability transition or observation in the true underlying process can cause degeneracy either at stitching time or in the filtering weights. It is possible to reintroduce particle diversity by applying an MCMC kernel after stitching as in resample-move particle filters [29] or particle rejuvenation [30]. The MCMC kernel is applied independently to each particle and must be invariant for the full smoothing distribution $p(x_{0:T}|y_{0:T})$. As we are only concerned with increasing particle diversity rather than taking ergodic averages we need only propose moving a subset of the trajectories, whether that be at stitching time or the latest observation time.

Algorithm 6 Backward Simulation

Input weighted marginal filtering samples $\{\tilde{x}_t^{(k)}, \tilde{w}_t^{(k)}\}_{k=1}^N$ for $t = 0, \ldots, T$, bound $\rho \geq p(x_t|x_{t-1})$ and the maximum number of rejections to attempt R.
Resample

$$\left\{\tilde{x}_T^{(k)}, \tilde{w}_T^{(k)}\right\}_{k=1}^N \rightarrow \left\{x_T^{(i)}\right\}_{i=1}^N$$

for $t = T-1, \ldots, 0$ **do**
  for $i = 1, \ldots, N$ **do**
    for $r = 1, \ldots, R$ **do**
      Sample $c^* \sim$ Categorical $\left(\left\{\tilde{w}_t^{(k)}\right\}_{k=1}^N\right)$
      Sample $u \sim U(0,1)$
      if $u < p(x_{t+1}^{(i)}|\tilde{x}_t^{(c^*)})/\rho$ **then**
        Accept c^*
        break
      end if
    end for
    if a sample c^* was accepted **then**
      Set $x_t^{(i)} = \tilde{x}_t^{(c^*)}$
    else
      Calculate interacting weights and normalise in k

$$w_t^{(k \leftarrow i)} \propto p(x_{t+1}^{(i)}|\tilde{x}_t^{(k)})\tilde{w}_t^{(k)} \qquad k = 1, \ldots, N.$$

      Sample $c_i \sim$ Categorical $\left(\left\{w_t^{(k \leftarrow i)}\right\}_{k=1}^N\right)$ and set $x_t^{(i)} = \tilde{x}_t^{(c_i)}$
    end if
  end for
end for
Output unweighted sample $\left\{x_{0:T}^{(i)}\right\}_{i=1}^N$ approximating $p(x_{0:T}|y_{0:T})$.

References

[1] N. Chopin and O. Papaspiliopoulos, *An Introduction to Sequential Monte Carlo*, ser. Springer Series in Statistics. Springer International Publishing, 2020.

[2] J. Olsson, O. Cappé, R. Douc, and Éric Moulines, "Sequential Monte Carlo smoothing with application to parameter estimation in nonlinear state space models," *Bernoulli*, vol. 14, no. 1, pp. 155 – 179, 2008.

[3] G. Poyiadjis, A. Doucet, and S. S. Singh, "Particle approximations of the score and observed information matrix in state space models with application to parameter estimation," *Biometrika*, vol. 98, no. 1, pp. 65–80, 02 2011.

[4] G. Kitagawa and S. Sato, *Monte Carlo Smoothing and Self-Organising State-Space Model*. New York, NY: Springer New York, 2001, pp. 177–195.

[5] P. Del Moral, A. Doucet, and S. S. Singh, "A backward particle interpretation of Feynman-Kac formulae," *ESAIM: M2AN*, vol. 44, no. 5, pp. 947–975, 2010.

[6] J. Olsson and J. Westerborn, "Efficient particle-based online smoothing in general hidden Markov models: The PaRIS algorithm," *Bernoulli*, vol. 23, no. 3, pp. 1951–1996, 08 2017.

[7] V. Elvira, J. Míguez, and P. M. Djurić, "Adapting the number of particles in sequential Monte Carlo methods through an online scheme for convergence assessment," *IEEE Transactions on Signal Processing*, vol. 65, no. 7, pp. 1781–1794, 2017.

[8] R. Douc and O. Cappe, "Comparison of resampling schemes for particle filtering," in *ISPA 2005. Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis, 2005.*, 2005, pp. 64–69.

[9] T. Li, M. Bolic, and P. M. Djuric, "Resampling methods for particle filtering: Classification, implementation, and strategies," *IEEE Signal Processing Magazine*, vol. 32, no. 3, pp. 70–86, 2015.

[10] A. Koppel, A. S. Bedi, B. M. Sadler, and V. Elvira, "Nearly consistent finite particle estimates in streaming importance sampling," 2021.

[11] S. Chatterjee and P. Diaconis, "The sample size required in importance sampling," *Ann. Appl. Probab.*, vol. 28, no. 2, pp. 1099–1135, 04 2018.

[12] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, Jul. 2000.

[13] S. J. Godsill, A. Doucet, and M. West, "Monte Carlo smoothing for nonlinear time series," *Journal of the American Statistical Association*, vol. 99, no. 465, pp. 156–168, 2004.

[14] A. Doucet, M. Briers, and S. Senecal, "Efficient block sampling strategies for sequential Monte Carlo methods," *Journal of Computational and Graphical Statistics*, vol. 15, no. 3, pp. 693–711, 2006.

[15] R. Douc, A. Garivier, E. Moulines, and J. Olsson, "Sequential Monte Carlo smoothing for general state space hidden Markov models," *The Annals of Applied Probability*, vol. 21, no. 6, pp. 2109–2145, 2011.

[16] N. Kantas, A. Doucet, S. S. Singh, J. Maciejowski, and N. Chopin, "On particle methods for parameter estimation in state-space models," *Statist. Sci.*, vol. 30, no. 3, pp. 328–351, 08 2015.

[17] P. Del Moral, A. Doucet, and S. S. Singh, "Uniform stability of a particle approximation of the optimal filter derivative," *SIAM Journal on Control and Optimization*, vol. 53, no. 3, pp. 1278–1304, 2015.

[18] E. Taghavi, F. Lindsten, L. Svensson, and T. B. Schön, "Adaptive stopping for fast particle smoothing," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 6293–6297.

[19] T. Clapp and S. Godsill, "Fixed-lag smoothing using sequential importance sampling," 1999.

[20] M. K. Pitt and N. Shephard, "Filtering via simulation: Auxiliary particle filters," *Journal of the American Statistical Association*, vol. 94, no. 446, pp. 590–599, 1999.

[21] M. Klaas, N. d. Freitas, and A. Doucet, "Toward practical n^2 Monte Carlo: The marginal particle filter," in *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*, ser. UAI'05. Arlington, Virginia, USA: AUAI Press, 2005, p. 308–315.

[22] P. Newson and J. Krumm, "Hidden Markov map matching through noise and sparseness," in *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, ser. GIS '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 336–343.

[23] M. Roth, F. Gustafsson, and U. Orguner, "On-road trajectory generation from gps data: A particle filtering/smoothing application," in *2012 15th International Conference on Information Fusion*, 2012, pp. 779–786.

[24] P. Davidson, J. Collin, and J. Takala, "Application of particle filters to a map-matching algorithm," *Gyroscopy and Navigation*, vol. 2, no. 4, p. 285, Oct 2011.

[25] K. Kempinska, T. Davies, and J. Shawe-Taylor, "Probabilistic map-matching using particle filters," *arXiv e-prints*, p. arXiv:1611.09706, Nov. 2016.

[26] L. Moreira-Matias, J. Gama, M. Ferreira, J. Moreira, and L. Damas, "Predicting taxi-passenger demand using streaming data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, pp. 1393–1402, 09 2013.

[27] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P. . Nordlund, "Particle filters for positioning, navigation, and tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 425–437, 2002.

[28] J. Alenlöv and J. Olsson, "Particle-based adaptive-lag online marginal smoothing in general state-space models," *IEEE Transactions on Signal Processing*, vol. 67, no. 21, pp. 5571–5582, 2019.

[29] W. R. Gilks and C. Berzuini, "Following a moving target—Monte Carlo inference for dynamic Bayesian models," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 63, no. 1, pp. 127–146, 2001.

[30] F. Lindsten, P. Bunch, S. S. Singh, and T. B. Schön, "Particle ancestor sampling for near-degenerate or intractable state transition models," 2015.

Particle-Based Adaptive-Lag Online Marginal Smoothing in General State-Space Models

Johan Alenlöv and Jimmy Olsson

***Abstract*—We present a novel algorithm, an adaptive-lag smoother, approximating efficiently, in an online fashion, sequences of expectations under the marginal smoothing distributions in general state-space models. The algorithm evolves recursively a bank of estimators, one for each marginal, in resemblance with the so-called particle-based, rapid incremental smoother (PaRIS). Each estimator is propagated until a stopping criterion, measuring the fluctuations of the estimates, is met. The presented algorithm is furnished with theoretical results describing its asymptotic limit and memory usage.**

***Index Terms*—Sequential Monte Carlo methods, state-space models, marginal smoothing, PaRIS, particle filters, state estimation.**

I. Introduction

STATE-SPACE models (SSMs), also known as *hidden Markov models* (HMMs), are fundamental in many scientific and engineering disciplines. Incorporating unobservable, Markovian states, these models are adjustable enough to model a variety of complex, real-world time series in, e.g., econometrics [1], speech recognition [2], and target tracking [3]. In this paper we focus on online state reconstruction in SSMs; more specifically, our goal is to estimate, on-the-fly as new data appear, expectations under the marginal posteriors of the different states given the data.

More precisely, an SSM is a bivariate model consisting of an observable process $\{Y_t\}_{t\in\mathbb{N}}$, referred to as the *observation process*, and an unobservable Markov chain $\{X_t\}_{t\in\mathbb{N}}$, known as the *state process*, taking on values in some general measurable spaces $(\mathsf{Y},\mathcal{Y})$ and $(\mathsf{X},\mathcal{X})$, respectively. Throughout the paper the subindex t will often be referred to as "time" without being necessarily a temporal index.

When operating on SSMs one is often interested in calculating expectations under the conditional distribution of one or several states conditioned upon a subset of some given stream $\{y_t\}_{t\in\mathbb{N}}$ of observations. For any $(s, s,' t) \in \mathbb{N}^3$ such that $s \le s' \le t$ we denote by $\phi_{s:s'|t}$ the conditional distribution of $X_{s:s'} = (X_s, \ldots, X_{s'})$ (our notation for vectors) given $Y_{0:t} = y_{0:t}$ (a precise definition is given in Section II). Some special distributions of interest are the *filter distributions* $\phi_t := \phi_{t:t|t}$, the *joint smoothing distributions* $\phi_{0:t|t}$, and, finally, when $s = s'$, the *marginal smoothing distributions* $\phi_{s|t}$.

For any probability measure μ and real-valued measurable function h, let $\mu h := \int h(x)\,\mu(\mathrm{d}x)$ denote the Lebesgue integral of h with respect to μ (whenever this is well defined). Given an observation stream $\{y_t\}_{t\in\mathbb{N}}$ and a sequence $\{h_t\}_{t\in\mathbb{N}}$ of real-valued functions, the present paper focuses on online calculation of the vector flow

$$(\phi_{0|t}h_0, \phi_{1|t}h_1, \ldots, \phi_{s|t}h_s, \ldots, \phi_{t-1|t}h_{t-1}, \phi_t h_t), \quad t \in \mathbb{N}, \tag{1}$$

as t increases. Note that a significant challenge with this problem is that *all* elements of the vector (1) change with t. Moreover, in the problem formulation, the word "online" means, first, that the observation sequence is processed in a single sweep as new observations appear and, second, that the computational cost of updating the approximations one step, i.e., from t to $t+1$, is uniformly bounded in t. Hence, the aggregated computational cost and memory requirements of the algorithm should grow at most linearly with t.

Remark 1: In the literature (see, e.g., [4]), the problem of computing $\phi_{s|t}$ offline for a fixed t and all $s \le t$ is typically referred to as *fixed-interval smoothing*. On the other hand, the problem of computing $\phi_{s|t}$ online for a fixed s and increasing t is typically referred to as *fixed-point smoothing*. Our aim of computing the whole vector (1) online could hence be viewed as a combination of these two problems.

A. Previous Work

Exact smoothing is possible only in models with finite state space and for linear Gaussian SSMs using the *forward-backward smoother* [5] and the *disturbance smoother* [6] (sometimes referenced to as the *Kalman smoother*), respectively. When dealing with general, possibly nonlinear SSMs, current methods take mainly two different approaches. The first approach relies, as in the *extended Kalman filter* and the *unscented Kalman filter* [7], [8], on linearisation of the model and Kalman filtering. These methods work well if the model is almost linear Gaussian, but will introduce significant errors in the presence of highly nonlinear/non-Gaussian model components.

The second approach relies on Monte Carlo simulation, preferably in the form of *sequential Monte Carlo* (SMC) *methods*, or, *particle filters*. Particle filters propagate recursively a sample of random simulations, so-called *particles*, mimicking

Manuscript received March 18, 2019; revised June 23, 2019; accepted July 23, 2019. Date of publication September 12, 2019; date of current version October 3, 2019. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Vasanthan Raghavan. The work of J. Olsson was supported by the Swedish Research Council under Grant 2018-05230. *(Corresponding author: Johan Alenlöv.)*

J. Alenlöv is with the Department of Information Technology, Uppsala University, 752 36 Uppsala, Sweden (e-mail: johan.alenlov@it.uu.se).

J. Olsson is with the Department of Mathematics, KTH Royal Institute of Technology, 114 28 Stockholm, Sweden (e-mail: jimmyol@kth.se).

Digital Object Identifier 10.1109/TSP.2019.2941066

typically, as in the *bootstrap particle filter*, the latent state (or, prior) dynamics. At each time step, the particles are associated with importance weights compensating for the discrepancy between the prior dynamics and the conditional (posterior) dynamics of the states given the observations. By duplicating and killing, through resampling, particles with high and low importance, respectively, the particles are directed towards regions of the state space with high posterior probability.

At each time step, the weighted empirical measure associated with the particles serves as an approximation of the filter distribution at the time step in question. Moreover, it can be shown that the empirical measure formed by the ancestral lines of the particles provides an approximation also of the joint smoothing distribution. However, since the particles are resampled repeatedly, these trajectories collapse eventually, and in the long run such a Monte Carlo approximation will rely on more or less a single path. Thus, the statistician is referred to alternative techniques such as the *forward-filtering backward-smoothing* (FFBSm) [9], *forward-filtering backward-simulation* (FFBSi) [10] or *joint backward simulation Rao-Blackwellized particle filter* [11] algorithms, which approximate the so-called *backward decomposition* of the joint smoothing distribution. The *two-filter algorithm* [4], [12]–[14] combines two separate particle filters, one in the forward direction and one in the reverse direction. In the *adaptive path integral smoother* [15], smoothing is performed by solving a control problem. A popular class of methods are the *Markov chain Monte Carlo* (MCMC) *methods* simulating a Markov chain admitting the joint smoothing distribution as invariant distribution. Recently, [16] combined successfully particle methods and MCMC into *particle Markov chain Monte Carlo* (PMCMC) *methods* to construct such Markov chains. All of these methods, in their most basic forms, require more than one pass of the data, either a forward and backward pass or that the estimates needs to be iterated until convergence is reached. Thus, the online processing setting of the present paper invalidates these methods.

As known to us, there is only one existing approach that applies to the problem (1) without violating the online criteria set up by us, namely the particle-based *fixed-lag smoother* [17], [18]. This is a genealogical tracing-based particle method that copes with particle path degeneracy by means of truncation. The truncation implies a bias that may be controlled using forgetting arguments (see [18] for an analysis). A problem with this method, which will be explained in some detail in Section IV-B, is that the truncation lag is a design parameter that needs to be set a priori by the user. This is nontrivial, as the optimal choice of the lag depends on the mixing properties of the model.

B. Our Contribution

In the present paper we introduce a novel algorithm that provides an approximate solution to the research question outlined in the previous section. The algorithm relies on a recursive version of the backward decomposition which has been employed previously for forward smoothing of additive state functionals [19], [20]. The same decomposition is used in the *particle-based, rapid incremental smoother* (PaRIS), proposed by the authors in [21] and analysed theoretically in [22], which performs online smoothing of additive state functionals with constant memory and computational complexity demands that grow only linearly with the number of particles (compared to the quadratic growth of other algorithms [20]).

The method proposed in the present paper is driven by the same sampling technique as the PaRIS and could be likened to computing one PaRIS estimate per marginal in (1). All these PaRIS estimates are formed on the basis of the output of the same underlying particle filter. However, since our aim is to compute, on-the-fly, *all* the marginal expectations in (1), we need, in order to avoid computational overload, to stop updating the estimate of a given marginal expectation once some criterion tells us to do so. This leads to an *adaptive-lag approach*. A sensible such stopping criterion will be designed using the forgetting properties of the model.

Our contribution is presented in two steps: first, we consider an ideal algorithm, applicable in the context of linear Gaussian models and requiring closed-form computation of all quantities of interest; after this, the general—possible nonlinear/non-Gaussian—case is dealt with using particle-based approximations. By using results derived in [22], we are able to analyse theoretically the asymptotic properties and memory demands of our algorithm in order to place the same on more solid ground.

Finally, we illustrate numerically the performance of the proposed algorithm and compare the same to that of the marginal fixed-lag smoother. As shown by the simulations, our adaptive-lag approach avoids completely the complex bias-variance trade-off that is an unavoidable ingredient of optimal fixed-lag design. Moreover, it is shown that the marginal adaptive-lag smoother is on par with the optimally tuned fixed-lag smoother in terms of variance.

C. Outline

In Section II we introduce formally SSMs and the backward decomposition. Section III introduces the ideal algorithm, providing a conceptual understanding of our approach without involving particle approximations. In Section IV we turn to the nonlinear/non-Gaussian case and replace intractable quantities by particle-based estimates. Theoretical results are presented in Section V and in Section VI the algorithm is benchmarked numerically on two models. Our conclusions are presented in Section VII and, finally, Section A contains some proofs.

II. Preliminaries

In the following, let $\mathbb{N}$ and $\mathbb{N}^*$ denote the sets of nonnegative and positive natural numbers, respectively. For any bounded measurable function h we let $\|h\|_\infty := \sup_{x\in\mathsf{X}} |h(x)|$ denote the supremum norm of h.

In the following we assume that all random elements are well-defined on a common probability space $(\Omega, \mathcal{F}, \mathbb{P})$. Let $(\mathsf{X}, \mathcal{X})$ and $(\mathsf{Y}, \mathcal{Y})$ be some measurable spaces, $\mathbf{Q} : \mathsf{X} \times \mathcal{X} \to [0,1]$ and $\mathbf{G} : \mathsf{X} \times \mathcal{Y} \to [0,1]$ some Markov transition kernels, and χ a probability measure on $\mathcal{X}$. An SSM is a bivariate Markov chain $\{(X_t, Y_t)\}_{t\in\mathbb{N}}$ on $\mathsf{X} \times \mathsf{Y}$ such that $X_{t+1}, Y_{t+1} \mid X_t, Y_t \sim \mathbf{Q}(X_t, \mathrm{d}x_{t+1})\,\mathbf{G}(x_{t+1}, \mathrm{d}y_{t+1})$ and $X_0, Y_0 \sim \chi(\mathrm{d}x_0)\,\mathbf{G}(x_0, \mathrm{d}y_0)$. It is assumed that only $\{Y_t\}_{t\in\mathbb{N}}$ is observable. Using this definition, it is easily shown that

i) the unobservable state sequence $\{X_t\}_{t\in\mathbb{N}}$ is a Markov chain with transition kernel $\mathbf{Q}$ and initial distribution χ.
ii) the variables of the observable process $\{Y_t\}_{t\in\mathbb{N}}$ are, conditionally on the states, independent and such that the conditional distribution of each Y_t depends on the corresponding X_t only.

Throughout this paper we will assume that the model is *fully dominated*, i.e., that $\mathbf{Q}$ and $\mathbf{G}$ admit transition densities q and g, respectively, with respect to some reference measures. We will in the following assume that we are given a fixed stream $\{y_t\}_{t\in\mathbb{N}}$ of observations. For ease of notation, let for all $t \in \mathbb{N}$, $g_t(x) := g(x, y_t)$, $x \in \mathsf{X}$. The joint smoothing distribution at time t, i.e., the law of $X_{0:t}$ conditionally to $Y_{0:t} = y_{0:t}$, is

$$\phi_{0:t|t}(\mathrm{d}x_{0:t}) := \frac{\chi(\mathrm{d}x_0)g_0(x_0)\prod_{s=1}^{t}\mathbf{Q}(x_{s-1},\mathrm{d}x_s)g_s(x_s)}{\int\cdots\int\chi(\mathrm{d}x'_0)g_0(x'_0)\prod_{s=1}^{t}\mathbf{Q}(x'_{s-1},\mathrm{d}x'_s)g_s(x'_s)},$$

and all posteriors $\phi_{s:s'|t}$ of interest (including the filter ϕ_t) are obtained as marginals of $\phi_{0:t|t}$.

In an SSM, the time-reversed state process is still Markov when evolving conditionally to $Y_{0:t}$. As a consequence, the joint smoothing distributions can be expressed differently using the so-called backward decomposition. Indeed, using Bayes' formula and the Markov property of the bivariate process $\{(X_t, Y_t)\}_{t\in\mathbb{N}}$ it is checked straightforwardly that the distribution of X_s given $X_{s+1} = x_{s+1}$ and $Y_{0:s} = y_{0:s}$ is given by

$$\overleftarrow{\mathbf{Q}}_{\phi_s}(x_{s+1},\mathrm{d}x_s) := \frac{q(x_s,x_{s+1})\phi_s(\mathrm{d}x_s)}{\int q(x'_s,x_{s+1})\phi_s(\mathrm{d}x'_s)}, \quad s\in\mathbb{N} \tag{2}$$

(see, e.g., [23, Prop. 3.3.6]). Using (2), the joint-smoothing distribution at time t may be expressed as

$$\phi_{0:t|t}(\mathrm{d}x_{0:t}) = \phi_t(\mathrm{d}x_t)\prod_{s=0}^{t-1}\overleftarrow{\mathbf{Q}}_{\phi_s}(x_{s+1},\mathrm{d}x_s)$$

This decomposition is instrumental in many smoothing procedures [10], [20], [22], [24].

III. Online Marginal Smoothing

Recall that our aim is to estimate the vectors (1) in an online manner as t increases. For the moment, consider estimation of some expectation $\phi_{s|t}h_s$ for some given marginal $\phi_{s|t}$. Under suitable ergodicity conditions (to be specified later), we may expect observations of the distant future to have limited effect on the posterior of some state X_s. Consequently, we may expect $\phi_{s|t}h_s$ to converge to some fixed point $\phi_{s|\infty}h_s$ as t increases; see [23, Sec. 4.3]. Thus, allowing for a negligible bias, we may update $\phi_{s|t}h_s$ only as long as the sequence $\{\phi_{s|t}h_s\}_{t=s}^{\infty}$ exhibits discernible fluctuations, i.e., until, say, $t = s_\varepsilon$, and approximate $\phi_{s|t}h_s$ by $\phi_{s|s_\varepsilon}h_s$ for all $t \geq s_\varepsilon$. Here ε is an algorithmic parameter regulating the stopping criterion. This idea is explored in the following.

A. An Ideal Algorithm

Let $(s, t) \in \mathbb{N}$ be such that $s \leq t$ and consider the marginal expectation $\phi_{s|t}h_s = \mathbb{E}[h_s(X_s) \mid Y_{0:t} = y_{0:t}]$. By the tower property,

$$\mathbb{E}\left[h_s(X_s) \mid Y_{0:t} = y_{0:t}\right] = \mathbb{E}\left[T_{s|t}(X_t) \mid Y_{0:t} = y_{0:t}\right], \tag{3}$$

where

$$T_{s|t}(x_t) := \mathbb{E}[h_s(X_s) \mid Y_{0:t-1} = y_{0:t-1}, X_t = x_t], \quad x_t \in \mathsf{X}, \tag{4}$$

is a statistic appearing frequently in the literature on smoothing; see for instance [20], [22], [24]. Appealingly, reapplying the tower property, the statistics $\{T_{s|t}\}_{t=s}^{\infty}$ can be expressed recursively (see [19], [25], [26]) through

$$\begin{aligned} T_{s|t+1}(x_{t+1}) &= \mathbb{E}[T_{s|t}(X_t) \mid Y_{0:t} = y_{0:t}, X_{t+1} = x_{t+1}] \\ &= \int T_{s|t}(x_t)\,\overleftarrow{\mathbf{Q}}_{\phi_t}(x_{t+1},\mathrm{d}x_t) \\ &= \int T_{s|t}(x_t)\frac{q(x_t,x_{t+1})\,\phi_t(\mathrm{d}x_t)}{\int q(x'_t,x_{t+1})\,\phi_t(\mathrm{d}x'_t)}, \quad x_{t+1}\in\mathsf{X}. \end{aligned} \tag{5}$$

The recursion is initialised by setting

$$T_{s|s}(x_s) := h_s(x_s), \quad x_s \in \mathsf{X},$$

and for completeness we define

$$T_{s|u} \equiv 0 \text{ for } u < s.$$

By (3), $\phi_{s|t}h_s = \phi_t T_{s|t}$, and the target can hence be calculated by applying the filter ϕ_t to the function $T_{s|t}$.

Now the question arises when to stop updating the quantity of interest; indeed, since we are interested in computing the full vector (1) but opposed to letting the computational complexity of the algorithm increase with time, we are forced to terminate updating when the fluctuations of the sequence $\{\phi_t T_{s|t}\}_{t=s}^{\infty}$ have ceased. In the present paper we will stop updating at the time point s_ε for which the variance of $T_{s|t}$ under the filter ϕ_t falls below some given threshold $\varepsilon > 0$ for the first time. After that, we output $\phi_{s_\varepsilon} T_{s|s_\varepsilon}$ as our estimate of $\phi_t T_{s|t}$ for all $t \geq s_\varepsilon$. This choice can be clearly motivated by (5), from which it is clear that once $T_{s|t}$ is close to constant in the support of ϕ_t, then also $T_{s|t+1}$ is close to constant everywhere.

The algorithm can be summarized as follows.

- Initialise by letting $\mathsf{S} \leftarrow \emptyset$. The set S will keep track of our active estimators. In addition, set the tolerance ε.
- For $t \leftarrow 0, 1, 2, 3, \ldots$
  - for each $s \in \mathsf{S}$, calculate $T_{s|t}(x_t)$ using (5);
  - let $\mathsf{S} \leftarrow \mathsf{S} \cup \{t\}$, i.e. activate an estimator at time t and set $T_{t|t} \leftarrow h_t$;
  - for each $s \in \mathsf{S}$, calculate the variance $\mathbb{V}_{\phi_t}[T_{s|t}(X_t)]$; if it is smaller than ε, let $\mathsf{S} \leftarrow \mathsf{S} \setminus \{s\}$ and output $\phi_t T_{s|t}$.

B. Example: Linear Gaussian SSMs

As mentioned previously, exact computation of the filter and joint-smoothing distributions is possible only for a few specific models. Here we present a Kalman filter-based version in the case of linear Gaussian SSMs.

In the linear Gaussian SSMs, an n_x-dimensional autoregressive state process is partially observed through n_y-dimensional observations. The model is specified by the equations

$$X_{t+1} = AX_t + U_{t+1},$$
$$Y_t = BX_t + V_t,$$

where $A \in \mathbb{R}^{n_x \times n_x}$ and $B \in \mathbb{R}^{n_y \times n_x}$ and $\{U_t\}_{t\in\mathbb{N}}$ and $\{V_t\}_{t\in\mathbb{N}}$ are sequences of mutually independent Gaussian noises with zero mean and covariance matrices $\Sigma_U \in \mathbb{R}^{n_x \times n_x}$ and $\Sigma_V \in \mathbb{R}^{n_y \times n_y}$, respectively. All matrices are assumed to be pre-specified. Given the sequence $\{y_t\}_{t\in\mathbb{N}}$ of observations we wish to estimate (1) in the case of *affine* objective functions $h_s(x) = \alpha_s^\intercal x + \beta_s$, where $\alpha_s \in \mathbb{R}^{n_x}$ and $\beta_s \in \mathbb{R}$ are pre-specified.

In this model each filter ϕ_t is Gaussian, and the *Kalman filter* propagates its mean μ_t and covariance matrix Σ_t recursively through time. We calculate the backward kernel (2), which is proportional to the filter at time s times the transition density of the latent Markov chain. Since both these distributions are Gaussian with known mean and covariance matrices, it is easy to show that also the distribution of X_t conditioned on $X_{t+1} = x_{t+1}$ and $Y_{0:t} = y_{0:t}$ is Gaussian with mean $\mu_{t|t+1}(x_{t+1}) = \Sigma_{t|t+1}(A^\intercal\Sigma_U^{-1}x_{t+1} + \Sigma_t^{-1}\mu_t)$ and covariance matrix $\Sigma_{t|t+1} = (A^\intercal\Sigma_U^{-1}A + \Sigma_t^{-1})^{-1}$. We may hence write down a specific updating procedure for the functions $\{T_{s|t}\}_{t=s}^{\infty}$ in this case:

- Initialisation: for $t = s$, let $T_{s|s}(x_s) = h_s(x_s) = \alpha_s^\intercal x_s + \beta_s$, $\alpha_{s|s} = \alpha_s$, and $\beta_{s|s} = \beta_s$.
- Proceeding recursively, assume that $T_{s|t}(x_t)$ is of affine form $T_{s|t}(x_t) = \alpha_{s|t}^\intercal x_t + \beta_{s|t}$ for given $\alpha_{s|t}$ and $\beta_{s|t}$; then also $T_{s|t+1}(x_{t+1}) = \alpha_{s|t+1}^\intercal x_{t+1} + \beta_{s|t+1}$ is of affine form, where

$$\alpha_{s|t+1}^\intercal = \alpha_{s|t}^\intercal\Sigma_{t|t+1}A^\intercal\Sigma_U^{-1},$$
$$\beta_{s|t+1} = \alpha_{s|t}^\intercal\Sigma_{t|t+1}A^\intercal\Sigma_t^{-1}\mu_t + \beta_{s|t}.$$

The last step of the algorithm—consisting in checking whether the variance of the function above is small enough—is now easily performed by calculating $\mathbb{V}_{\phi_t}[T_{s|t}(X_t)] = \alpha_{s|t}^\intercal\Sigma_t\alpha_{s|t}$ and comparing this with some pre-specified threshold ε. This completes all the steps needed for executing the algorithm, and we refer to Section VI for a numerical illustration.

Even though the previous method provides an exact implementation of the algorithm, it is limited to a single class of models and specific target functions. To move beyond this simplified setting we need to rely on approximations, and this will be discussed in the next section.

IV. Particle-Based Online Marginal Smoothing

As mentioned previously, exact computation of the filter distributions—and hence the backward kernels—is possible only in a few specific cases. In the general case we will approximate these distributions using particle filters, which are recalled in the following.

A. Particle Filters

A particle filter propagates recursively a set of particles with associated weights in order to approximate the filter distribution flow $\{\phi_t\}_{t\in\mathbb{N}}$ given the sequence $\{y_t\}_{t\in\mathbb{N}}$.

Algorithm 1: Bootstrap Particle Filter.

Require: A weighted sample $\{(\xi_t^i, \omega_t^i)\}_{i=1}^N$ targeting the filter distribution ϕ_t
1: **for** $i = 1 \to N$ **do**
2: draw $I_{t+1}^i \sim \mathsf{Pr}(\{\omega_t^\ell\}_{\ell=1}^N)$;
3: draw $\xi_{t+1}^i \sim \mathbf{Q}(\xi_t^{I_{t+1}^i}, \cdot)$;
4: set $\omega_{t+1}^i = g_{t+1}(\xi_{t+1}^i)$;
5: **end for**
6: **return** $\{(\xi_{t+1}^i, \omega_{t+1}^i)\}_{i=1}^N$

We describe recursively the most basic particle filter—the bootstrap filter [27]—and assume that we have at hand a sample $\{(\xi_t^i, \omega_t^i)\}_{i=1}^N$ of particles (the ξ_t^i) and associated weights (the ω_t^i) targeting the filter distribution ϕ_t in the sense that for all ϕ_t-integrable functions f,

$$\sum_{i=1}^N \frac{\omega_t^i}{\Omega_t} f(\xi_t^i) \backsimeq \phi_t f,$$

where $\Omega_t := \sum_{i=1}^N \omega_t^i$ denotes the total weight and "$\backsimeq$" means that the estimator on the left hand side is *consistent*, i.e., converges in probability to the right hand side, as $N \to \infty$. To form a new weighted sample $\{(\omega_{t+1}^i, \xi_{t+1}^i)\}_{i=1}^N$ targeting the subsequent filter distribution ϕ_{t+1}, a two-step procedure is applied. First, the particles are resampled by drawing randomly a set of N independent indices $\{I_{t+1}^i\}_{i=1}^N$ from the categorical distribution induced by the probabilities proportional to the weights $\{\omega_t^i\}_{i=1}^N$, an operation denoted by $I_{t+1}^i \sim \mathsf{Pr}(\{\omega_t^i\}_{i=1}^N)$, $i \in \{1, \ldots, N\}$.

Second, the resampled particles are moved conditionally independently according to the dynamics of the state process, i.e., for all $i \in \{1, \ldots, N\}$,

$$\xi_{t+1}^i \sim \mathbf{Q}\left(\xi_t^{I_{t+1}^i}, \cdot\right).$$

Finally, new importance weights are computed according to

$$\omega_{t+1}^i = g_{t+1}\left(\xi_{t+1}^i\right)$$

for all $i \in \{1, \ldots, N\}$.

Initialisation is carried through by drawing $\xi_0^i \sim \chi$ and setting $\omega_0^i = g_0(\xi_0^i)$ for all $i \in \{1, \ldots, N\}$.

We summarise the procedure in Algorithm 1, and denote by "$\{(\xi_{t+1}^i, \omega_{t+1}^i)\}_{i=1}^N \leftarrow \mathsf{PF}(\{(\xi_t^i, \omega_t^i)\}_{i=1}^N, y_{t+1})$" one application of Algorithm 1. (By convention we let $\{(\xi_0^i, \omega_0^i)\}_{i=1}^N \leftarrow \mathsf{PF}(\{(\xi_{-1}^i, \omega_{-1}^i)\}_{i=1}^N, y_0)$ denote the initial step.)

So far we have only considered estimation of the filter distributions. The rest of this section will be devoted to particle approximation of the marginal smoothing distributions.

For all $i \in \{1, \ldots, N\}$, define recursively genealogical indices $\{G_{s|t}^i\}_{s=0}^t$ by $G_{t|t}^i = i$ and $G_{s-1|t}^i = I_s^{G_{s|t}^i}$. The set $\{(\xi_0^{G_{0|t}^i}, \ldots, \xi_t^{G_{t|t}^i})\}_{i=1}^N$ is often referred to as the *genealogical tree* of the particles, and it is easy to show that the genealogical tree may, together with the importance weights $\{\omega_t^i\}_{i=1}^N$, be used for estimating the joint-smoothing distribution. In particular,

$$\sum_{i=1}^N \frac{\omega_t^i}{\Omega_t} h_s\left(\xi_s^{G_{s|t}^i}\right) \backsimeq \phi_{s|t} h_s, \tag{6}$$

which means a given marginal smoothing distribution $\phi_{s|t}$ can be approximated using the weighted sample formed by the *time s ancestors* of $\{\xi_t^i\}_{i=1}^N$. We refer to this estimator as the *Poor man's smoother*. A well-known problem with the Poor man's smoother is that the repeated resampling operations of the particle filter always deplete the genealogical tree in the long run; thus, sooner or later, for some $s < t$ and some i_0, $G_{u|t}^{i_0} = G_{u|t}^j$ for all $j \in \{1, \ldots, N\}$ and all $u \in \{0, \ldots, s\}$, implying that the estimates of $\phi_{u|t}h_u, u \in \{0, \ldots, s\}$, will be based on only a single particle path. The work [28] establishes, under assumptions requiring typically the state space to be a compact set, an $\mathcal{O}(N\log(N))$ bound on the expected distance from the last generation to the most recent common ancestor which is *uniform* in time. Thus, the number of active, unique particles in the estimator (6) tends to one as t increases, leading to a depleted and impractical estimator.

B. Fixed-Lag Smoothing

To remedy the problem of particle lineage degeneracy, a fixed-lag smoother [17] can be used. The idea is to approximate the marginal smoothing distribution $\phi_{s|t}$ by the distribution $\phi_{s|s_\Delta(t)}$, where $s_\Delta(t) = (s + \Delta) \wedge t$ for some pre-specified lag $\Delta \in \mathbb{N}^*$. In this case, using the notation above, we get the biased approximation

$$\phi_{s|t}h_s \approx \sum_{i=1}^N \frac{\omega_{s_\Delta(t)}^i}{\Omega_{s_\Delta(t)}} h_s\left(\xi_s^{G_{s|s_\Delta(t)}^i}\right),$$

where $G_{s|s_\Delta(t)}^i$ is defined as above.

The approach requires suitable design of the lag Δ, and we face here a classical bias-variance tradeoff: if Δ is too small, then the forgetting of the model has not kicked in, and the discrepancy between the distributions is going to be large (leading to high bias); on the other hand, if Δ is too large, then, by path degeneracy, the estimate will be depleted (leading to high variance). The optimal choice of lag depends on the mixing of the SSM, and [18] proposes an optimal choice of Δ as $\lceil c\log(t)\rceil$, where the constant c depends on the mixing. This is problematic since it is hard to calculate, and even estimate, the mixing of an SSM. Thus, designing properly the lag is indeed a non-trivial task. The lag-based particle estimator that we propose in the next section relies again on forgetting-based arguments, but adapts the lag in a completely automatic manner.

C. The Adaptive-Lag Smoother

We present here a particle-based version of the ideal algorithm in Section III-A, which can be thought of as an adaptive-lag smoother. In this algorithm we employ novel techniques for updating particle estimates of the functions $T_{s|t}$, and the adaptive lag equals the number of steps that each function is updated before the stopping criterion triggers truncation. The truncation depends heavily on the mixing of the model, but is now determined in an adaptive manner.

The critical step in the algorithm presented in Section III-A is the need of estimating and updating the statistics $T_{s|t}$ through the recursion (5). We proceed by induction and assume that we have at hand a set $\{\tilde{\tau}_{s|t}^i\}_{i=1}^N$ of estimates of $\{T_{s|t}(\xi_t^i)\}_{i=1}^N$. Proceeding as in [20], these estimates are updated to estimates $\{\tilde{\tau}_{s|t+1}^i\}_{i=1}^N$ of $\{T_{s|t+1}(\xi_{t+1}^i)\}_{i=1}^N$ by replacing, in (5), ϕ_t by a particle approximation, yielding

$$\tilde{\tau}_{s|t+1}^i = \sum_{j=1}^N \frac{\omega_t^j q(\xi_t^j, \xi_{t+1}^i)}{\sum_{\ell=1}^N \omega_t^\ell q(\xi_t^\ell, \xi_{t+1}^i)} \tilde{\tau}_{s|t}^j, \quad i \in \{1, \ldots, N\}, \quad (7)$$

where the ratio is a particle approximation of the backward kernel (2), and the particle approximation $\sum_{i=1}^N \omega_t^i \tilde{\tau}_{s|t}^i / \Omega_t \backsimeq \phi_{s|t}h_s$. Casting the recursion (7) into the ideal algorithm in Section III yields the following procedure:

- Initialise by letting $\mathsf{S} \leftarrow \emptyset$ and setting the tolerance ε.
- For $t \leftarrow 0, 1, 2, 3, \ldots$
  - run $\{(\xi_t^i, \omega_t^i)\}_{i=1}^N \leftarrow \mathsf{PF}(\{(\xi_{t-1}^i, \omega_{t-1}^i)\}_{i=1}^N, y_t)$;
  - for each $s \in \mathsf{S}$ and $i \in \{1, \ldots, N\}$, calculate $\tilde{\tau}_{s|t}^i$ using (7);
  - let $\mathsf{S} \leftarrow \mathsf{S} \cup \{t\}$ and $\tilde{\tau}_{t|t}^i \leftarrow h_t(\xi_t^i)$ for all $i \in \{1, \ldots, N\}$.
  - for each $s \in \mathsf{S}$, if

$$\sum_{i=1}^N \frac{\omega_t^i}{\Omega_t}\left(\tilde{\tau}_{s|t}^i - \sum_{j=1}^N \frac{\omega_t^j}{\Omega_t}\tilde{\tau}_{s|t}^j\right)^2 < \varepsilon,$$

    then let $\mathsf{S} \leftarrow \mathsf{S} \setminus \{s\}$ and output $\sum_{i=1}^N \omega_t^i \tilde{\tau}_{s|t}^i / \Omega_t$ as an estimate of $\phi_{s|t'}h_s$ for all $t' \geq t$.

A drawback with the updating formula (7) is that it requires a sum of N terms to be computed for each particle, which yields an overall $\mathcal{O}(N^2)$ computational complexity. Needless to say, this is impractical when N is large.

To reduce the computational burden, we proceed as in the PaRIS [22, Alg. 2] and replace the right hand side of (7), which can be interpreted as an expectation, by a Monte Carlo estimate. More precisely, assuming that we have at hand a set $\{\tau_{s|t}^i\}_{i=1}^N$ of estimates of $\{T_{s|t}(\xi_t^i)\}_{i=1}^N$, we replace (7) by the empirical average

$$\tau_{s|t+1}^i = \frac{1}{\tilde{N}} \sum_{j=1}^{\tilde{N}} \tau_{s|t}^{J_t^{(i,j)}}, \quad (8)$$

where $\{J_t^{(ij)}\}_{j=1}^{\tilde{N}}$ are conditionally independent draws from $\mathsf{Pr}(\{\omega_t^\ell q(\xi_t^\ell, \xi_{t+1}^i)\}_{\ell=1}^N)$ and $\tilde{N} \in \mathbb{N}^*$ is a precision parameter. Such draws can most often be produced at low computational cost using rejection sampling. Indeed, assume that the transition density q is uniformly bounded by some constant $\bar{\varepsilon}$, i.e., $q(x, x') \leq \bar{\varepsilon}$ for all $(x, x') \in \mathsf{X}^2$; then, following [24], a draw J from $\mathsf{Pr}(\{\omega_t^\ell q(\xi_t^\ell, \xi_{t+1}^i)\}_{\ell=1}^N)$ can be produced by repeating the following steps until acceptance:

1) draw $J \sim \mathsf{Pr}(\{\omega_t^i\}_{i=1}^N)$;
2) accept J with probability $q(\xi_t^J, \xi_{t+1}^i)/\bar{\varepsilon}$.

Importantly, under the mixing assumptions given in Section V it is possible to show that the expected number of trials required for sampling all the indices $\{J_t^{(ij)}\}_{j=1}^{\tilde{N}}$ is linear in $\tilde{N}$; see [22, Thm. 10]. This yields an $\mathcal{O}(N\tilde{N})$ algorithm, and as we will see below, $\tilde{N}$ can be kept at a very low value (say, $\tilde{N} = 2$).

(a) $\tilde{N} = 1$ (b) $\tilde{N} = 2$

Fig. 1. Genealogical traces corresponding to backward simulation in the adaptive-lag smoothing algorithm. Columns of nodes refer to different particle populations (with $N = 3$) at different time points (with time increasing rightward) and arrows indicate connections through the backward draws in the algorithm. The striped particles are in the final estimator for the marginal smoothing distribution when $s = 0$ and $t = 4$, black-colored particles are in the support of the historical trajectories of the final estimator, while gray-colored ones are inactive.

The variance of $T_{s|t}$ under ϕ_t is estimated using

$$\sigma_{s|t}^{2,N} := \sum_{i=1}^{N} \frac{\omega_t^i}{\Omega_t} \left(\tau_{s|t}^i - \sum_{\ell=1}^{N} \frac{\omega_t^\ell}{\Omega_t} \tau_{s|t}^\ell \right)^2,$$

and the updating procedure is stopped if $\sigma_{s|t}^{2,N} < \varepsilon$, where ε is some pre-chosen tolerance. Letting

$$s_\varepsilon^N(t) := \min \left\{ u \geq s : \sigma_{s|u}^{2,N} < \varepsilon \right\} \wedge t, \tag{9}$$

we return the estimator

$$\phi_{s|t}^{N,\epsilon} h_s := \sum_{i=1}^{N} \frac{\omega_{s_\varepsilon^N(t)}^i}{\Omega_{s_\varepsilon^N(t)}} \tau_{s|s_\varepsilon^N(t)}^i$$

of $\phi_{s|t} h_s$. The algorithm is presented in detail in Algorithm 2.

Remark 2: In Algorithm 2 we let, on Line 7, the underlying particles be propagated by means of the standard bootstrap particle filter for simplicity. However, from a methodological point of view, running the proposed algorithm requires only access to a sequence of consistent samples—with or without weights—targeting the filter distribution flow, and it is, in practice, of subordinate importance how these samples are produced. This is relevant for, e.g., high-dimensional scenarios or scenarios with highly informative observations, where alternative particle filters, such as *feedback particle filters* [29] or *auxiliary particle filters* [30], may improve estimation significantly.

D. Designing Algorithmic Parameters

In Algorithm 2, the parameters ε and $\tilde{N}$ are set by the user. As main results, [22] establishes that the PaRIS is (i) consistent for all fixed $\tilde{N} \in \mathbb{N}^*$ and (ii) numerically stable only if $\tilde{N} \geq 2$. The latter is illustrated by Fig. 1, from which it is clear that using $\tilde{N} = 1$ leads to a degeneracy phenomenon that is reminiscent of the degeneracy of the genealogical tree. As it is clear from the same figure, this phenomenon is avoided in the case $\tilde{N} = 2$. This distinction between the cases $\tilde{N} = 1$ and $\tilde{N} \geq 2$ is also present in the central limit theorem in [22, Thm. 8], where, in the marginal smoothing case, a time uniform $\mathcal{O}(1 + 1/\tilde{N})$ bound on the asymptotic variance is obtainable only in the case $\tilde{N} \geq 2$. As suggested by this bound, there is no gain in using a too large precision $\tilde{N}$, and typically $\tilde{N} = 2$ provides a satisfactory accuracy in simulations.

When it concerns the tolerance ε, using a smaller tolerance implies a larger lag, implying in turn more accurate estimates (and conversely). However, a larger lag requires a larger bank of active estimators, increasing in turn the computational time and memory requirement. This trade-off is studied in more detail in Section V and Section VI.

Algorithm 2: Adaptive-Lag Smoother.

1: set $\mathsf{S} \leftarrow \{0\}$;
2: run $\{(\xi_0^i, \omega_0^i)\}_{i=1}^N \leftarrow \mathsf{PF}(\{(\xi_{-1}^i, \omega_{-1}^i)\}_{i=1}^N, y_0)$;
3: **for** $i = 1 \to N$ **do**
4: set $\tau_{0|0}^i \leftarrow \tilde{h}_0(\xi_0^i)$;
5: **end for**
6: **for** $t \leftarrow 1, 2, 3, \ldots$ **do**
7: run $\{(\xi_t^i, \omega_t^i)\}_{i=1}^N \leftarrow \mathsf{PF}(\{(\xi_{t-1}^i, \omega_{t-1}^i)\}_{i=1}^N, y_t)$;
8: **for** $i = 1 \to N$ **do**
9: **for** $j = 1 \to \tilde{N}$ **do**
10: draw $J_t^{(ij)} \sim \mathsf{Pr}(\{\omega_{t-1}^\ell q(\xi_{t-1}^\ell, \xi_t^i)\}_{\ell=1}^N)$;
11: **end for**
12: **for** $s \in \mathsf{S}$ **do**
13: set $\tau_{s|t}^i \leftarrow \tilde{N}^{-1} \sum_{j=1}^{\tilde{N}} \tau_{s|t-1}^{J_t^{(i,j)}}$;
14: **end for**
15: set $\tau_{t|t}^i \leftarrow h_t(\xi_t^i)$;
16: **end for**
17: set $\mathsf{S} \leftarrow \mathsf{S} \cup \{t\}$;
18: **for** $s \in \mathsf{S}$ **do**
19: **if** $\sigma_{s|t}^{2,N} < \varepsilon$ **then**
20: set $\phi_{s|t'}^{N,\epsilon} h_s \leftarrow \sum_{i=1}^N \omega_t^i \tau_{s|t}^i / \Omega_t$ for all $t' \geq t$;
21: set $\mathsf{S} \leftarrow \mathsf{S} \setminus \{s\}$.
22: **end if**
23: **end for**
24: **end for**

V. Theory

A. Convergence of the Sample Variance Criterion

We start off the theoretical analysis by studying the asymptotics (as $N \to \infty$) of the sample variances $\sigma_{s|t}^{2,N}$.

The analysis will be carried through under the following assumptions.

Assumption 1: For all $t \in \mathbb{N}$, $\|g_t\|_\infty < \infty$. Moreover, there exists a constant $|h|_\infty < \infty$ such that $\mathrm{osc}(h_t) < |h|_\infty$ for all $t \in \mathbb{N}$.

Note that the first part of Assumption 1 implies finiteness of the particle weights. The following auxiliary result establishes

the convergence of each sample variance criterion to a deterministic limit.

Lemma 3: Let Assumption 1 hold. Then for all $(s,t) \in \mathbb{N}^2$ such that $s \leq t$, it holds, as $N \to \infty$,

$$\sum_{i=1}^{N} \frac{\omega_t^i}{\Omega_t} \left(\tau_{s|t}^i - \sum_{\ell=1}^{N} \frac{\omega_t^\ell}{\Omega_t} \tau_{s|t}^\ell \right)^2 \xrightarrow{\mathbb{P}} \sigma_{s|t}^{2,\infty},$$

where

$$\sigma_{s|t}^{2,\infty} := \phi_t \left\{ (T_{s|t} - \phi_t T_{s|t})^2 \right\} + \eta_{s,t}$$

with $\eta_{s,t}$ being defined in Section A, Eqn. (11).

Second, we bound the limiting variance criterion. This calls for the following *strong mixing assumptions*.

Assumption 2:

(i) There exist $0 < \underline{\varepsilon} < \bar{\varepsilon} < \infty$ such that $\underline{\varepsilon} < q(x,x') < \bar{\varepsilon}$ for all $(x,x') \in \mathsf{X}^2$.

(ii) There exist $0 < \underline{\delta} < \bar{\delta} < \infty$ such that for all $t \in \mathbb{N}$, $\underline{\delta} < g_t(x) < \bar{\delta}$ for all $x \in \mathsf{X}$.

Under Assumption 2(i), define $\varrho := 1 - \underline{\varepsilon}/\bar{\varepsilon}$.

Assumptions similar to Assumption 2 appear frequently in the literature (see, e.g., [31]) and require typically the state space of the hidden chain to be a compact set.

Theorem 4: Under Assumptions 1 and 2, it holds, for all $(s,t) \in \mathbb{N}^2$ such that $s \leq t$ and all $\tilde{N} \in \mathbb{N}^*$,

$$\sigma_{s|t}^{2,\infty} \leq |h|_\infty^2 \begin{cases} c_1 \varrho^{2(t-s)} + c_2 \tilde{N}^{-(t-s)} & \text{if } \tilde{N}\varrho^2 \neq 1, \\ \varrho^{2(t-s)} + c_3 (t-s) \tilde{N}^{-(t-s)} & \text{if } \tilde{N}\varrho^2 = 1, \end{cases} \quad (10)$$

where the constants c_1, c_2 and c_3 are independent of s and t.

The first term in the bound (10) is related to the mixing of the SSM, and since $\varrho \in (0,1)$ this term tends to zero geometrically fast as t grows. The second term is related to the Monte Carlo error induced by the PaRISian update. Here we clearly see that it is required that $\tilde{N} \geq 2$ in order for this term to vanish as t increases. In that case the desired limit $\sigma_{s|t}^{2,\infty} \to 0$ holds as $t \to \infty$.

B. Convergence of the Estimator

In order to derive the asymptotic limit of our estimator we introduce the following notation. Let $s_\varepsilon(t) := \min\{u \geq s : \sigma_{s|u}^{2,\infty} < \varepsilon\} \wedge t$, which can be understood as the limit of $s_\varepsilon^N(t)$ (defined in (9)) as N tends to infinity. In addition, let $\phi_{s|t}^\epsilon := \phi_{s|s_\varepsilon(t)}$ and notice that this measure differs slightly from the adaptive-lag approximation delivered by the ideal algorithm in Section III-A, since the limiting variance $\sigma_{s|t}^{2,\infty}$ is not equal to $\mathbb{V}_{\phi_t}[T_{s|t}(X_t)]$; recall that the former also has an additional term $\eta_{s,t}$ corresponding to the Monte Carlo approximation of the backward kernel.

As expected and as established by the following theorem, $\phi_{s|t}^\epsilon$ is indeed the asymptotic limit of the proposed estimator.

Theorem 5: Let Assumptions 1 and 2 hold. Then for all $(s,t) \in \mathbb{N}^2$ such that $s \leq t$ and all bounded measurable functions h_s, as $N \to \infty$,

$$\phi_{s|t}^{N,\epsilon} h_s \xrightarrow{\mathbb{P}} \phi_{s|t}^\epsilon h_s.$$

C. Bound on Asymptotic Memory Requirement

Finally, we show that the memory requirement of the algorithm stays, in the asymptotic regime, uniformly bounded in t, which was a requirement in the problem statement. Asymptotically, the estimate of $\phi_{s|t} h_s$ is still under construction at time t if $\sigma_{s|t}^{2,\infty} \geq \varepsilon$ for all $t \in \{s, \ldots, t\}$. Thus, let

$$\mathfrak{A}_t := \sum_{s=0}^{t} \prod_{u=s}^{t} \mathbb{1}_{\left\{\sigma_{s|u}^{2,\infty} \geq \varepsilon\right\}}$$

be the number of active adaptive-lag estimators at time t in the asymptotic regime.

Theorem 6: Under Assumption 2, for all $\tilde{N} \geq 2$,

$$\mathfrak{A}_t \leq \frac{\log(\varepsilon / \{|h|_\infty^2 d(\varrho, \tilde{N})\})}{\log(\varrho^2 \vee \tilde{N}^{-1})},$$

where $d(\varrho, \tilde{N}) > 0$ depends on ϱ and $\tilde{N}$ only.

We remark that the bound in Theorem 6 is uniform in time, implying a uniformly bounded memory requirement of the algorithm, at least in the asymptotic regime. Moreover, we note that the bound in Theorem 6 has an $\mathcal{O}(-\log \varepsilon)$ term.

VI. Simulations

We benchmark the algorithm on two different models. First we consider a linear Gaussian SSM, which enables computation of the exact distributions using the disturbance smoother Kalman smoother (see, e.g., [23, sec. 5.2.4]). The second model is the now classical stochastic volatility model proposed in [32].

A. Linear Gaussian SSM

Consider a linear Gaussian SSM given by the following equations:

$$X_{t+1} = aX_t + \sigma_U U_{t+1},$$
$$Y_t = bX_t + \sigma_V V_t,$$

where $\{U_t\}_{t \geq 0}$ and $\{V_t\}_{t \geq 0}$ are independent sequences of mutually independent standard Gaussian noise variables. Initially, $X_0 \sim \mathcal{N}(0, \sigma_V^2/(1-a^2))$. We consider smoothed means, corresponding to the objective functions $h_s = \mathrm{id}$ for all $s \in \mathbb{N}$. We simulate a data record comprising 201 observations $y_{0:200}$ from the the model parameterised by $(a, b, \sigma_U, \sigma_V) = (.95, .5, .5, 2)$. Using tolerances $\varepsilon \in \{.5, .2, .1, 10^{-3}\}$ and $(N, \tilde{N}) = (400, 2)$ we performed 100 independent runs of the algorithm with the same input data. In addition, the Kalman version of the ideal algorithm (as presented in Section III-B) was run with the same tolerances as the particle version.

First, the estimates produced by these two algorithms are compared to exact values computed offline using the disturbance smoother. The outcome is displayed in Fig. 2, from which it is clear that the estimates improve with decreasing tolerance. For $\varepsilon = .5$, the estimates exhibit clear fluctuations around the true values, while for $\varepsilon = 10^{-3}$ the estimates follow closely the true quantities. The top panel of Fig. 3 reports time-averaged mean squared errors (MSEs) (with respect to the exact posterior means delivered by the disturbance smoother) for different values of ε. As expected, the MSE decreases monotonously with ε. However, since decreasing tolerance ε comes at the cost of computational work, we also introduce a measure of *efficiency*, defined as the product of reciprocal MSE and reciprocal CPU runtime. The bottom panel of Fig. 3 reports efficiency for different values of ε. Since the bias of the algorithm is more or less eliminated

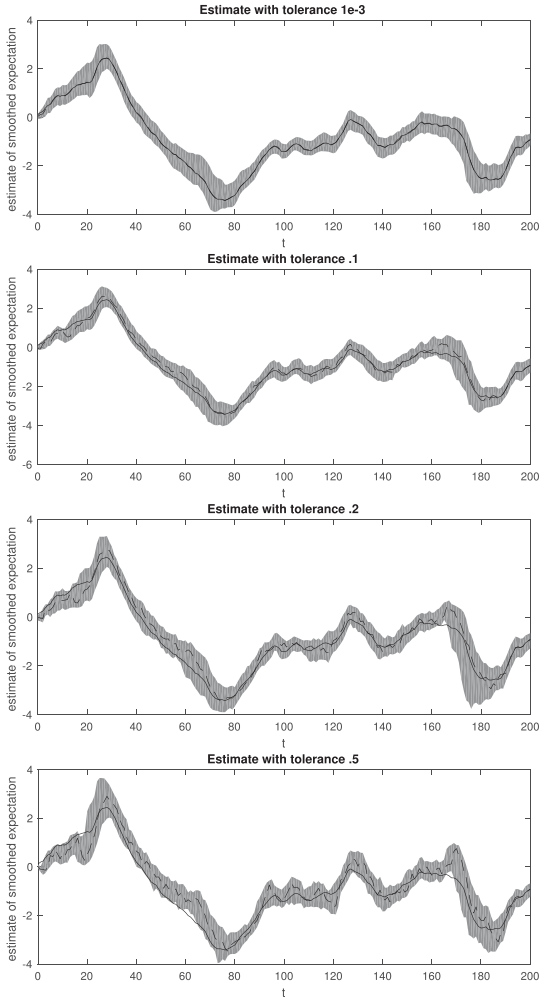


Fig. 2. Smoothed means for the linear Gaussian model. In each plot, the gray zone is the range of the particle-based estimates, while the black line indicates the exact smoothed values computed offline using the disturbance smoother. The dashed line is the Kalman version of our algorithm with the same tolerances as the particle-based method. When $\varepsilon = 10^{-3}$ the Kalman filter-based estimates and the exact values are more or less indistinguishable.

for all sufficiently small tolerances ε, there is an optimal value around $\varepsilon = .5 \times 10^{-3}$ for which the efficiency is maximal.

Finally, Fig. 4 displays the initial variance criterion $\sigma_{0|t}^{2,N}$ for different values of t, and the plot is well in line with the exponentially decreasing bound provided by Theorem 4. Finally, we study the truncation lags $s_\varepsilon^N(t) - s$ determined by the algorithm for different tolerances. The average lags across the 100 runs are displayed in Fig. 4, from which it is evident that decreasing ε leads, in accordance with Theorem 6, on the average to larger lags. The slope in the end of the plot indicates non-truncation.

B. Stochastic Volatility Model

The second model of consideration is the stochastic volatility model

$$X_{t+1} = \phi X_t + \sigma U_{t+1},$$
$$Y_t = \beta \exp(X_t/2) V_t,$$

where $\{U_t\}_{t\in\mathbb{N}}$ and $\{V_t\}_{t\in\mathbb{N}}$ are independent sequences of mutually independent standard Gaussian noise variables and

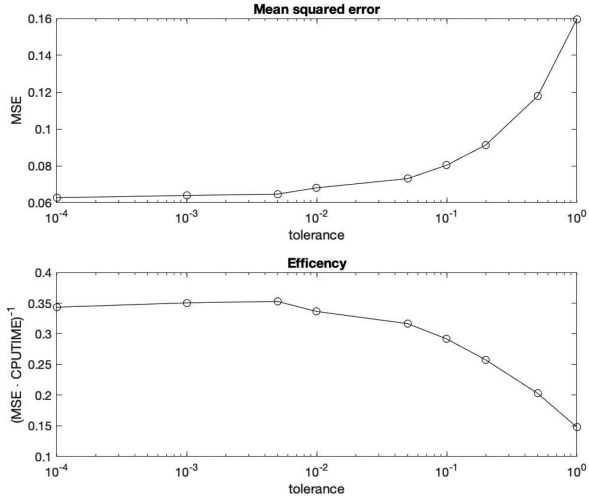


Fig. 3. Top panel: Time-averaged MSE for different tolerances ε. MSE decreases monotonously with ε. Bottom panel: Efficiency, defined as the product of reciprocal MSE and reciprocal CPU runtime, against tolerance ε. Maximal efficiency is reached for $\varepsilon = .5 \cdot 10^{-2}$.

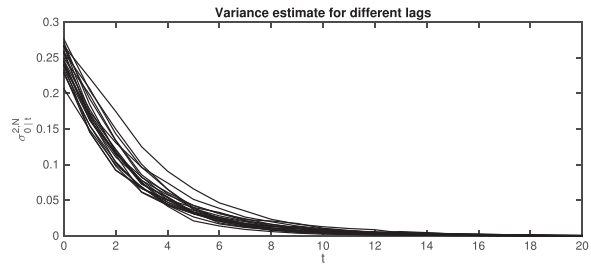


Fig. 4. Linear Gaussian model: the initial variance criterion $\sigma_{0|t}^{2,N}$ for different values of t.

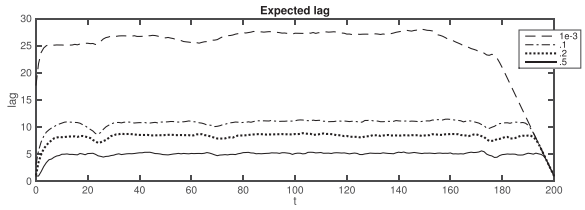


Fig. 5. Linear Gaussian model: average lag against time t.

$X_0 \sim \mathcal{N}(0, \sigma^2/(1-\phi^2))$. In this nonlinear SSM, $\{Y_t\}_{t\in\mathbb{N}}$ can be thought of as the log-returns of a stock while $\{X_t\}_{t\in\mathbb{N}}$ can then be thought of as the unobserved log-volatility of the observed returns.

As before, we simulate 201 observations from the model indexed by $(\phi, \sigma, \beta) = (.98, \sqrt{.1}, \sqrt{.7})$. We employ the adaptive-lag smoother targeting again the mean of the marginal smoothing distribution, i.e., $h_s = \mathrm{id}$ for all $s \in \mathbb{N}$. Algorithm 2 is run with the tolerances $\varepsilon \in \{.5, .1, 10^{-3}\}$ and sample sizes $(N, \tilde{N}) = (400, 2)$. The algorithm is executed 200 times for each value of ε, and all runs are based on the same data input. Since exact computation is infeasible for this nonlinear model, we use, as reference, proxies for the true posterior means obtained as the averages of 10 independent replicates of the full PaRIS (i.e., without stopping criterion) with $(N, \tilde{N}) = (2000, 2)$.

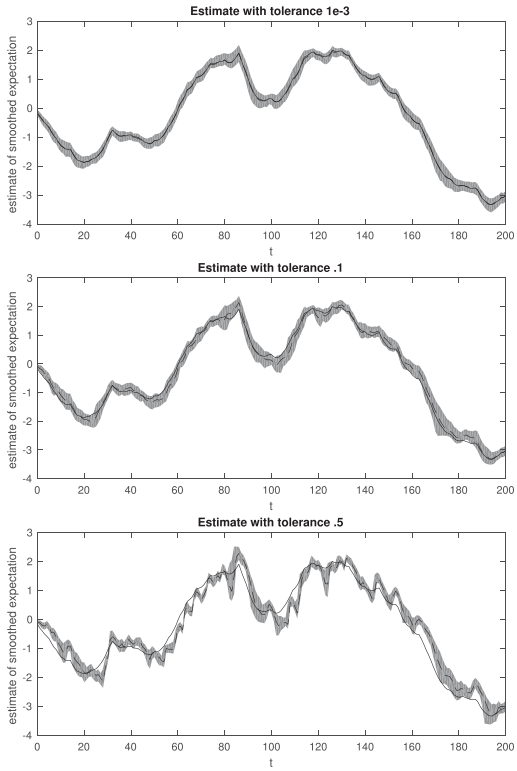


Fig. 6. Stochastic volatility model: smoothed means for different tolerances (cf. Fig. 2).

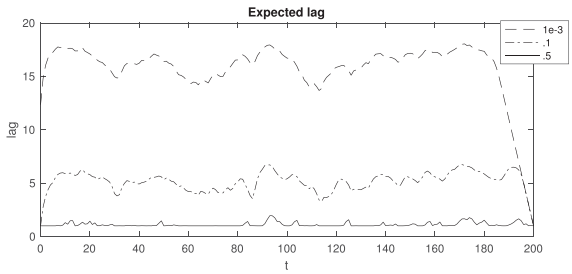


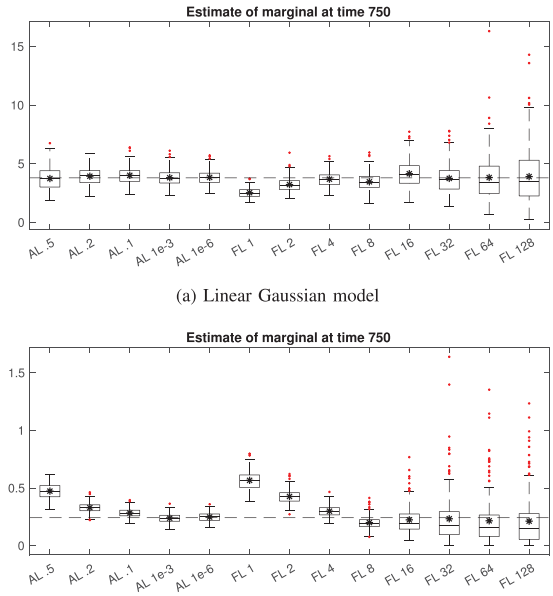
Fig. 7. Stochastic volatility model: average lags for different tolerances.

The outcome is reported in Fig. 6. For $\varepsilon = .5$ the marginal smoothing estimates of the adaptive-lag smoother deviate significantly from the ground truth of the PaRIS algorithm for most time-steps. Decreasing the tolerance to $\varepsilon = .1$ yields clearly improved—but still undesirably volatile—estimates. However, decreasing the tolerance even further to $\varepsilon = 10^{-3}$ leads to a plot where the PaRIS estimates are firmly in the area of the estimates produced by the adaptive-lag smoother, and taking the mean over all the adaptive-lag estimates yields values indistinguishable from the estimates delivered by the PaRIS.

Finally, Fig. 7 provides the average lags at different time steps, and obviously the nonlinear components of the model leads to a higher degree of adaptation compared to the linear Gaussian model.

C. Comparison With Fixed-Lag Smoothing

We end the numerical illustrations with a comparison of the adaptive-lag smoother to the fixed-lag smoother. As mentioned earlier, in the fixed-lag smoother a lag has to be set a priori. The determination of the optimal lag requires the user to make a complex bias-variance tradeoff, governed by the mixing properties of the model as well as the coalescence properties of the particle paths. It is hence impossible to determine the optimal lag without first carrying through a preparatory simulation study. We simulate new data from both the linear Gaussian and the stochastic volatility models using the same parameters as above and compare the adaptive-lag approach with $\varepsilon \in \{.5, .2, .1, 10^{-3}, 10^{-6}\}$ and $(N, \tilde{N}) = (400, 2)$ to the fixed-lag approach with lags $\Delta \in \{1, 2, 4, 8, 16, 32, 64, 128\}$ and $N = 400$. For both models we simulate 1001 observations and aim at estimating $\mathbb{E}[X_s^2 \mid Y_{0:1000}]$. We perform 200 independent runs of all algorithms. The results are displayed in Fig. 8, where we have chosen $s = 750$ for illustration. From the plot it is clear that too small choices of the lag Δ lead to significant bias, while too large choices increase excessively the variance. The optimal choice of Δ can be seen to be somewhere around $\Delta = 8$ and $\Delta = 16$ for both models in this example. On the other hand, from the same plot it is clear that the tolerance ε does not obey the same bias-variance-tradeoff. When ε is simply small enough, the adaptive-lag smoother provides estimates having a variance being on par with that of the optimally tuned fixed-lag smoother. Moreover, importantly, decreasing excessively to $\varepsilon = 10^{-6}$ does not, by the numerical stability of the PaRIS-type smoothing estimates, imply additional variance.



(a) Linear Gaussian model

(b) Stochastic volatility model

Fig. 8. Comparison of the adaptive-lag smoother and the fixed-lag smoother for the linear Gaussian model (a) and stochastic volatility model (b). The black stars indicate the sample mean for each algorithm and the dashed line indicates a ground truth calculated exactly by the Kalman smoother in (a) and with the PaRIS algorithm in (b).

VII. Conclusion

In this paper we have presented a novel algorithm—an adaptive-lag marginal smoother—for online computation of

marginal-smoothing expectations in general SSMs. We are not aware of any other algorithm in the literature solving satisfactorily this challenging problem.

The proposed algorithm differs from the standard particle-based fixed-lag smoother [17], [18] (in our view, the only competitor of the proposed method) in essentially two ways: first, the estimates produced by the algorithm do not at all suffer from particle path degeneracy; second, the lag is designed adaptively by the algorithm via a variance criterion. Remarkably, since the PaRIS-type updates stabilize completely the support of the estimator as long as $\tilde{N} \geq 2$, the user does not risk afflicting the produced estimates with undesired variance by using an unnecessarily small tolerance ε and, consequently, an unnecessarily large average lag. This is the main advantage of the proposed algorithm over particle-based fixed-lag smoothing.

We remark that the proposed algorithm is, just like the PaRIS, *function specific* in the sense that it is adjusted to the given sequence $\{h_t\}_{t\in\mathbb{N}}$ of objective functions and outputs directly a sequence of estimated expectations rather than a sequence of weighted samples targeting the marginal smoothing distributions of interest. Thus, solving the problem (1) for *different* objective function sequences requires different function-specific updates (8) to be run in parallel. Still, nothing prevents these updates to be based on the same underlying particle filter and even the same backward draws $J_t^{(i,j)}$.

Like all existing particle-based forward-filtering backward-smoothing-type algorithms, the proposed procedure relies on particle approximation of the backward kernels. Such approximation requires typically the SSM to be fully dominated. Even though many SSMs used in practice are indeed fully dominated, extending the methodology beyond this class of models is an important direction of research. Moreover, in our work we linearize the computational complexity of the backward sampling operation—the computational bottleneck of the algorithm—using the accept-reject approach proposed in [24]. However, in practice, especially when the state space X is high dimensional, the upper bound $\bar{\varepsilon}$ may be very large, implying only a limited computational gain. In such cases, the adaptive-stopping approach suggested in [33] may come in useful. Finally, as in all backward-sampling-type smoothing algorithms, problems may occur for models and observation sequences for which the filter and smoothing distributions have significantly different support; we refer to [15] for a discussion.

When it concerns the theoretical developments, a result that is missing in the present paper is a rigorous theoretical analysis of the estimator's bias and the dependence of the bias on the tolerance ε. Since the estimator is driven by ergodicity arguments, such an analysis requires control of the forgetting of the backward chain. However, this is still an open—and presumably very complex—problem, and the question is thus left as future research.

Needless to say, the strong mixing assumptions (see Assumption 2) driving the theoretical analysis of our estimator are restrictive. Still, these are standard in the literature, and only a few—technically involved—works analyze the time-uniform convergence of particle filters under weaker, verifiable assumptions that can be checked also for nonlinear SSMs with non-compact state space; see e.g. [34], [35]. For the particle-based backward algorithm, [36] provides a stochastic stability analysis of the FFBSm algorithm under assumptions that point to applications with non-compact state spaces, and a similar approach may be applicable to our estimator.

The convergence results presented in Section V hold only asymptotically as $N \to \infty$, and to furnish the estimator with error bounds for finite N would be appealing. Looking at the PaRIS algorithm, it is indeed possible to derive an exponential concentration inequality for fixed N (see [22, Corollary 2]), and we thus expect such a Hoeffding-type bound to hold also for the estimator proposed in the present paper. However, to derive a *time uniform* exponential concentration inequality for these algorithms appears to be considerably more complicated, as Hoeffding's inequality, which would be the first tool to try out in the proof, seems to be too crude for this purpose. Hence, we also have to leave this as future work.

APPENDIX
PROOFS OF THEOREMS

We preface the proofs by some additional notation.

For all $t \in \mathbb{N}$, define the unnormalised transition kernels

$$\mathbf{L}_t(x_t, \mathrm{d}x_{t+1}) := g_{t+1}(x_{t+1})\, \mathbf{Q}(x_t, \mathrm{d}x_{t+1}),$$

with the convention that $\mathbf{L}_s\mathbf{L}_t \equiv \mathrm{id}$ when $s > t$. In addition, we let $\mathbf{L}_{-1}(x, \mathrm{d}x_0) := g_0(x_0)\, \delta_x(\mathrm{d}x_0)$. Moreover, we may express each joint smoothing distribution $\phi_{0:t|t}$ as $\phi_{0:t|t} = \phi_t \mathbf{T}_t$, where we have defined the kernels

$$\mathbf{T}_t(x_t, \mathrm{d}x_{0:t-1}) := \begin{cases} \prod_{s=0}^{t-1} \overleftarrow{\mathbf{Q}}_{\phi_s}(x_{s+1}, \mathrm{d}x_s) & \text{for } t \in \mathbb{N}^*, \\ \mathrm{id} & \text{for } t = 0. \end{cases}$$

Notice that $T_{s|t}(x_s)$ in (4) can be expressed as $T_{s|t}(x_s) = \mathbf{T}_t h_s(x_s)$. Finally, define the operator

$$\mathbf{D}_t : h \mapsto \mathbf{T}_t(h - \phi_{0:t|t} h),$$

acting on the space of bounded measurable functions.

Proof of Lemma 3: By [22, Lemma 13] it holds that

$$\sum_{i=1}^{N} \frac{\omega_t^i}{\Omega_t} (\tau_{s|t}^i)^2 \xrightarrow{\mathbb{P}} \phi_t(\mathbf{T}_t^2 h_s) + \eta_t,$$

where

$$\eta_t := \sum_{\ell=s}^{t-1} \tilde{N}^{\ell-t} \frac{\phi_\ell \mathbf{L}_\ell \{ \overleftarrow{\mathbf{Q}}_{\phi_\ell} (\mathbf{T}_\ell h_s - \mathbf{T}_{\ell+1} h_s)^2 \mathbf{L}_{\ell+1} \cdots \mathbf{L}_{t-1} \mathbb{1}_{\mathsf{X}} \}}{\phi_\ell \mathbf{L}_\ell \cdots \mathbf{L}_{t-1} \mathbb{1}_{\mathsf{X}}}. \tag{11}$$

In addition, from [22, Theorem 1] we get that

$$\left(\phi_{s|t}^N h_s\right)^2 \xrightarrow{\mathbb{P}} {\phi_t}^2(\mathbf{T}_t h_s),$$

and combining the previous two limits yields, as $N \to \infty$,

$$\begin{aligned} \sigma_{s|t}^{2,N} &= \sum_{i=1}^{N} \frac{\omega_t^i}{\Omega_t} (\tau_{s|t}^i)^2 - \left(\sum_{i=1}^{N} \frac{\omega_t^i}{\Omega_t} \tau_{s|t}^i \right)^2 \\ &\xrightarrow{\mathbb{P}} \sigma_{s|t}^{2,\infty} := \phi_t(\mathbf{T}_t^2 h_s) - {\phi_t}^2(\mathbf{T}_t h_s) + \eta_t \\ &= \phi_t \left\{ (\mathbf{T}_t h_s - \phi_t \mathbf{T}_t h_s)^2 \right\} + \eta_t. \end{aligned} \tag{12}$$

■

Proof of Theorem 4: We begin by noticing that

$$\phi_t\left\{(\mathbf{T}_t h_s - \phi_t \mathbf{T}_t h_s)^2\right\} = \phi_t\left(\mathbf{D}_t^2 h_s\right).$$

Thus, by [24, Lemma 10] it holds, for all $t \geq s$,

$$\|\mathbf{D}_t h_s\|_\infty \leq \varrho^{t-s}|h|_\infty,$$

giving us the bound

$$\|\phi_t(\mathbf{D}_t^2 h_s)\|_\infty \leq \varrho^{2(t-s)}|h|_\infty^2$$

on the first term of (12). In order to bound the second term of (12), i.e. η_t, we note that, [24, Lemma 10] yields that

$$\|\mathbf{T}_\ell h_s - \mathbf{T}_{\ell+1} h_s\|_\infty \leq 2\varrho^{\ell-s}|h|_\infty.$$

In addition, under Assumption 2, for all $x \in \mathsf{X}$,

$$\begin{aligned}&\underline{\varepsilon}\mu(g_{\ell+2}\mathbf{L}_{\ell+2}\cdots\mathbf{L}_{t-1}\mathbb{1}_{\mathsf{X}})\\ &\quad\leq \mathbf{L}_{\ell+1}\cdots\mathbf{L}_{t-1}\mathbb{1}_{\mathsf{X}}(x)\\ &\quad\leq \bar{\varepsilon}\mu(g_{\ell+2}\mathbf{L}_{\ell+2}\cdots\mathbf{L}_{t-1}\mathbb{1}_{\mathsf{X}}).\end{aligned}$$

Combining the previous two bounds allows η_t to be bounded; indeed, proceed like

$$\begin{aligned}&\frac{\phi_\ell \mathbf{L}_\ell\{\overleftarrow{\mathbf{Q}}_{\phi_\ell}(\mathbf{T}_\ell h_s - \mathbf{T}_{\ell+1}h_s)^2\mathbf{L}_{\ell+1}\cdots\mathbf{L}_{t-1}\mathbb{1}_{\mathsf{X}}\}}{\phi_\ell \mathbf{L}_\ell\cdots\mathbf{L}_{t-1}\mathbb{1}_{\mathsf{X}}}\\ &\quad\leq 4\varrho^{2(\ell-s)}|h|_\infty^2\tfrac{\bar{\varepsilon}}{\underline{\varepsilon}} = \varrho^{2(t-s)}\tfrac{4|h|_\infty^2}{(1-\varrho)}.\end{aligned}$$

Assuming now that $\tilde{N}\varrho^2 \neq 1$ we may bound η_t using that

$$\begin{aligned}\eta_t &\leq \frac{4|h|_\infty^2}{(1-\varrho)}\sum_{\ell=s}^{t-1}\tilde{N}^{\ell-t}\varrho^{2(\ell-s)}\\ &= \frac{4|h|_\infty^2}{(1-\varrho)}\tilde{N}^{-t}\varrho^{-2s}\sum_{\ell=s}^{t-1}(\tilde{N}\varrho^2)^\ell\\ &= \frac{4|h|_\infty^2}{(1-\varrho)(1-\tilde{N}\varrho^2)}\left(\tilde{N}^{-(t-s)} - \varrho^{2(t-s)}\right). \qquad (13)\end{aligned}$$

On the other hand, if $\tilde{N}\varrho^2 = 1$, (13) yields

$$\eta_t \leq \tfrac{4|h|_\infty^2}{(1-\varrho)}\tilde{N}^{-(t-s)}(t-s).$$

The previous argument may be summarised as

$$\sigma_{s|t}^{2,\infty} \leq |h|_\infty^2\begin{cases}c_1\varrho^{2(t-s)} + c_2\tilde{N}^{-(t-s)} & \text{if } \tilde{N}\varrho^2 \neq 1,\\ \varrho^{2(t-s)} + c_3(t-s)\tilde{N}^{-(t-s)} & \text{if } \tilde{N}\varrho^2 = 1,\end{cases}$$

where

$$c_1 := 1 - \tfrac{4}{(1-\varrho)(1-\tilde{N}\varrho^2)},\ \ c_2 := \tfrac{4}{(1-\varrho)(1-\tilde{N}\varrho^2)},\ \ c_3 := \tfrac{4}{(1-\varrho)}.$$

■

Proof of Theorem 5: Write

$$\begin{aligned}\phi_{s|t}^{N,\epsilon}h_s &= \sum_{u=s+1}^{t}\mathbb{1}_{\{s_\varepsilon^N(t)=u\}}\phi_{s|u}^N h_s\\ &= \sum_{u=s+1}^{t}\phi_{s|u}^N h_s \mathbb{1}_{\left\{\sigma_{s|u}^{2,N}<\varepsilon\right\}}\prod_{\ell=s+1}^{u-1}\mathbb{1}_{\left\{\sigma_{s|\ell}^{2,N}\geq\varepsilon\right\}}\\ &\quad+\phi_{s|t}^N h_s\prod_{\ell=s+1}^{t}\mathbb{1}_{\left\{\sigma_{s|\ell}^{2,N}\geq\varepsilon\right\}}.\end{aligned}$$

By Lemma 3 and Theorem 4 it holds that $\sigma_{s|t}^{2,N} \xrightarrow{\mathbb{P}} \sigma_{s|t}^{2,\infty}$ as $N \to \infty$. This yields, by Slutsky's theorem, as $N \to \infty$,

$$\begin{aligned}\phi_{s|t}^{N,\epsilon}h_s \xrightarrow{\mathbb{P}} &\sum_{u=s+1}^{t}\phi_{s|u}h_s\mathbb{1}_{\left\{\sigma_{s|u}^{2,\infty}<\varepsilon\right\}}\prod_{\ell=s+1}^{u-1}\mathbb{1}_{\left\{\sigma_{s|\ell}^{2,\infty}\geq\varepsilon\right\}}\\ &+\phi_{s|t}h_s\prod_{\ell=s+1}^{t}\mathbb{1}_{\left\{\sigma_{s|\ell}^{2,\infty}\geq\varepsilon\right\}} = \sum_{u=s+1}^{t}\mathbb{1}_{\{u=s_\varepsilon(t)\}}\phi_{s|u}h_s.\end{aligned}$$

■

Proof of Theorem 6: Note that

$$\mathfrak{A}_t = \sum_{s=0}^{t}\prod_{u=s}^{t}\mathbb{1}_{\left\{\sigma_{s|u}^{2,\infty}\geq\varepsilon\right\}} \leq \sum_{s=0}^{t}\mathbb{1}_{\left\{\sigma_{s|t}^{2,\infty}\geq\varepsilon\right\}}.$$

Now, we consider separately the two cases $\tilde{N}\varrho^2 \neq 1$ and $\tilde{N}\varrho^2 = 1$.

First, assume that $\tilde{N}\varrho^2 \neq 1$.

By Theorem 4, $\sigma_{s|t}^{2,\infty} \geq \varepsilon$ implies that $|h|_\infty^2 c_1\varrho^{2(t-s)} + |h|_\infty^2 c_2\tilde{N}^{-(t-s)} \geq \varepsilon$. To proceed further, consider the two subcases (i): $\tilde{N}^{-1} > \varrho^2$ and (ii): $\tilde{N}^{-1} < \varrho^2$.

(i) In this case, $c_1 < 0$; thus, if

$$|h|_\infty^2 c_1\varrho^{2(t-s)} + |h|_\infty^2 c_2\tilde{N}^{-(t-s)} \geq \varepsilon,$$

then

$$|h|_\infty^2 c_2\tilde{N}^{-(t-s)} \geq \varepsilon.$$

Consequently, an upper limit on $t-s$ is given by

$$t - s \leq \tfrac{\log(\varepsilon/\{|h|_\infty^2 c_2\})}{\log\tilde{N}^{-1}}.$$

Thus, $\mathfrak{A}_t$ may be bounded according to

$$\begin{aligned}\mathfrak{A}_t &\leq \sum_{s=0}^{t}\mathbb{1}\left\{t-s \leq \tfrac{\log(\varepsilon/\{|h|_\infty^2 c_2\})}{\log\tilde{N}^{-1}}\right\}\\ &\leq \tfrac{\log(\varepsilon/\{|h|_\infty^2 c_2\})}{\log\tilde{N}^{-1}}.\end{aligned}$$

(ii) In this case, $c_2 < 0$; thus, as previously,

$$t - s \leq \tfrac{\log(\varepsilon/\{|h|_\infty^2 c_1\})}{\log\varrho^2},$$

yielding the bound

$$\mathfrak{A}_t \leq \tfrac{\log(\varepsilon/\{|h|_\infty^2 c_1\})}{\log\varrho^2}.$$

We now assume that $\tilde{N}\varrho^2 = 1$. In this case, $\sigma_{s|t}^{2,\infty} \geq \varepsilon$ implies that $|h|_\infty^2\tilde{N}^{-(t-s)}\{1 + c_3(t-s)\} \geq \varepsilon$ as c_3 is always positive and $t \geq s$.

Thus,

$$t - s \leq \tfrac{\log(\varepsilon/\{|h|_\infty^2(1+c_3)\})}{\log\tilde{N}^{-1}}.$$

providing the bound

$$\mathfrak{A}_t \leq \tfrac{\log(\varepsilon/\{|h|_\infty^2(1+c_3)\})}{\log\tilde{N}^{-1}}.$$

Combining these results gives us the final bound

$$\mathfrak{A}_t \leq \tfrac{\log(\varepsilon/\{|h|_\infty^2 d(\varrho,\tilde{N})\})}{\log(\varrho^2\vee\tilde{N}^{-1})}$$

for a constant $d(\varrho, \tilde{N})$ depending on the constants c_1, c_2, and c_3 as well as the values of ϱ^2 and $\tilde{N}$. ■

ACKNOWLEDGMENT

The authors would like to thank the anonymous referees for insightful comments that improved the presentation of the paper.

REFERENCES

[1] S. Chib, F. Nadari, and N. Shephard, "Markov chain Monte Carlo methods for stochastic volatility models," *J. Econometrics*, vol. 108, pp. 281–316, 2002.

[2] L. R. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1993.

[3] L. Mihaylova, D. Angelova, S. Honary, D. R. Bull, C. N. Canagarajah, and B. Ristic, "Mobility tracking in cellular networks using particle filtering," *IEEE Trans. Wireless Commun.*, vol. 6, no. 10, pp. 3589–3599, Oct. 2007.

[4] M. Briers, A. Doucet, and S. Maskell, "Smoothing algorithms for state-space models," *Ann. Inst. Statist. Math.*, vol. 62, no. 1, pp. 61–89, 2010.

[5] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–285, Feb. 1989.

[6] P. De Jong, "A cross validation filter for time series models," *Biometrika*, vol. 75, pp. 594–600, 1988.

[7] B. D. O. Anderson and J. B. Moore, *Optimal Filtering*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1979.

[8] S. J. Julier and J. K. Uhlmann, "A new extension of the Kalman filter to nonlinear systems," in *Proc. 11th Int. Symp. Aerosp./Defense Sens., Simul. Controls*, 1997, pp. 182–193.

[9] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte-Carlo sampling methods for Bayesian filtering," *Statist. Comput.*, vol. 10, pp. 197–208, 2000.

[10] S. J. Godsill, A. Doucet, and M. West, "Monte Carlo smoothing for nonlinear time series," *J. Amer. Statist. Assoc.*, vol. 50, pp. 438–449, 2004.

[11] F. Lindsten, P. Bunch, S. J. Godsill, and T. B. Schön, "Rao-Blackwellized particle smoothers for mixed linear/nonlinear state-space models," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, May 2013, pp. 6288–6292.

[12] G. Kitagawa, "Monte-Carlo filter and smoother for non-Gaussian nonlinear state space models," *J. Comput. Graph. Statist.*, vol. 1, pp. 1–25, 1996.

[13] P. Fearnhead, D. Wyncoll, and J. Tawn, "A sequential smoothing algorithm with linear computational cost," *Biometrika*, vol. 97, no. 2, pp. 447–464, 2010.

[14] T. N. M. Nguyen, S. Le Corff, and E. Moulines, "On the two-filter approximations of marginal smoothing distributions in general state-space models," *Advances Appl. Probability*, vol. 50, no. 1, p. 154–177, 2017.

[15] H.-C. Ruiz and H. J. Kappen, "Particle smoothing for hidden diffusion processes: Adaptive path integral smoother," *IEEE Trans. Signal Process.*, vol. 65, no. 12, pp. 3191–3203, Jun. 2017.

[16] C. Andrieu, A. Doucet, and R. Holenstein, "Particle Markov chain Monte Carlo methods," *J. Roy. Statist. Soc. Series B Statist. Method.*, vol. 72, no. 3, pp. 269–342, 2010.

[17] G. Kitagawa and S. Sato, "Monte Carlo smoothing and self-organising state-space model," in *Sequential Monte Carlo Methods Practice* (Statistics for Engineering and Information Science). New York, NY, USA: Springer, 2001, pp. 177–195.

[18] J. Olsson, O. Cappé, R. Douc, and E. Moulines, "Sequential Monte Carlo smoothing with application to parameter estimation in non-linear state space models," *Bernoulli*, vol. 14, no. 1, pp. 155–179, 2008.

[19] O. Cappé, "Online EM algorithm for hidden Markov models," *J. Comput. Graph. Statist.*, vol. 20, no. 3, pp. 728–749, 2011.

[20] P. Del Moral, A. Doucet, and S. Singh, "Forward smoothing using sequential Monte Carlo," Cambridge Univ., Tech. Rep. CUED/F-INFENG/TR 638, 2010.

[21] J. Olsson and J. Westerborn, "Efficient particle-based online smoothing in general hidden Markov models," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2014, pp. 8003–8007.

[22] J. Olsson and J. Westerborn, "Efficient particle-based online smoothing in general hidden Markov models: The PaRIS algorithm," *Bernoulli*, vol. 23, no. 3, pp. 1951–1996, 2017.

[23] O. Cappé, E. Moulines, and T. Rydén, *Inference in Hidden Markov Models*. Berlin, Germany: Springer, 2005.

[24] R. Douc, A. Garivier, E. Moulines, and J. Olsson, "Sequential Monte Carlo smoothing for general state space hidden Markov models," *Ann. Appl. Probab.*, vol. 21, no. 6, pp. 2109–2145, 2011.

[25] G. Mongillo and S. Denève, "Online learning with hidden Markov models," *Neural Comput.*, vol. 20, no. 7, pp. 1706–1716, 2008.

[26] P. Del Moral, A. Doucet, and S. Singh, "A Backward Particle Interpretation of Feynman-Kac Formulae," *ESAIM Math. Modelling, Numer. Anal.*, vol. 44, no. 5, pp. 947–975, 2010.

[27] N. Gordon, D. Salmond, and A. F. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEE Proc. F, Radar Signal Process.*, vol. 140, no. 2, pp. 107–113, Apr. 1993.

[28] P. E. Jacob, L. M. Murray, and S. Rubenthaler, "Path storage in the particle filter," *Statist. Comput.*, vol. 25, no. 2, pp. 487–496, 2015.

[29] T. Yang, P. G. Mehta, and S. P. Meyn, "Feedback particle filter," *IEEE Trans. Autom. Control*, vol. 58, no. 10, pp. 2465–2480, Oct. 2013.

[30] M. K. Pitt and N. Shephard, "Filtering via simulation: Auxiliary particle filters," *J. Amer. Statist. Assoc.*, vol. 94, no. 446, pp. 590–599, 1999.

[31] P. Del Moral, *Feynman-Kac Formulae. Genealogical and Interacting Particle Systems with Applications*. Berlin, Germany: Springer, 2004.

[32] J. Hull and A. White, "The pricing of options on assets with stochastic volatilities," *J. Finance*, vol. 42, pp. 281–300, 1987.

[33] E. Taghavi, F. Lindsten, L. Svensson, and T. Schön, "Adaptive stopping for fast particle smoothing," in *Proc. 38th Int. Conf. Acoust., Speech, Signal Process.*, 2013, pp. 6293–6297.

[34] R. van Handel, "Uniform time average consistency of Monte Carlo particle filters," *Stochastic Processes Their Appl.*, vol. 119, no. 11, pp. 3835–3861, 2009.

[35] R. Douc, E. Moulines, and J. Olsson, "Long-term stability of sequential Monte Carlo methods under verifiable conditions," *Ann. Appl. Probab.*, vol. 24, no. 5, pp. 1767–1802, 2014.

[36] A. Jasra, "On the behaviour of the backward interpretation of Feynman-Kac formulae under verifiable conditions," *J. Appl. Probability*, vol. 52, no. 2, pp. 339–359, 2015.

Johan Alenlöv received the M.Sc. degree in engineering mathematics from Lund University, Lund, Sweden, in 2012, and the Ph.D. degree in applied and computational mathematics from KTH Royal Institute of Technology, Stockholm, Sweden, in 2017.

Since 2018, he has been holding a postdoctoral position with the Division of Systems and Control, Department of Information Technology, Uppsala University, Uppsala, Sweden. His research interest includes inference in general state-space models using sequential Monte Carlo methods.

Jimmy Olsson received the M.Sc. degree in engineering physics and the Ph.D. degree in mathematical statistics from Lund University, Lund, Sweden, in 2002 and 2007, respectively. Following a postdoctoral position with Tlcom ParisTech, Paris, France, he was with the Centre for Mathematical Sciences, Lund University, during 2008 and 2013. He is currently an Associate Professor with the Department of Mathematics, KTH Royal Institute of Technology, Stockholm, Sweden. His research interests include inference in stochastic processes, computational statistics, especially sequential Monte Carlo and Markov chain Monte Carlo methods, and applied probability.

A Tutorial on Particle Filtering and Smoothing: Fifteen years later

Arnaud Doucet
The Institute of Statistical Mathematics,
4-6-7 Minami-Azabu, Minato-ku,
Tokyo 106-8569, Japan.
Email: Arnaud@ism.ac.jp

Adam M. Johansen
Department of Statistics,
University of Warwick,
Coventry, CV4 7AL, UK
Email: A.M.Johansen@warwick.ac.uk

First Version 1.0 – April 2008
This Version 1.1 – December 2008 with typographical corrections March 2012

Abstract

Optimal estimation problems for non-linear non-Gaussian state-space models do not typically admit analytic solutions. Since their introduction in 1993, particle filtering methods have become a very popular class of algorithms to solve these estimation problems numerically in an online manner, i.e. recursively as observations become available, and are now routinely used in fields as diverse as computer vision, econometrics, robotics and navigation. The objective of this tutorial is to provide a complete, up-to-date survey of this field as of 2008. Basic and advanced particle methods for filtering as well as smoothing are presented.

Keywords: Central Limit Theorem, Filtering, Hidden Markov Models, Markov chain Monte Carlo, Particle methods, Resampling, Sequential Monte Carlo, Smoothing, State-Space models.

1 Introduction

The general state space hidden Markov models, which are summarised in section 2.1, provide an extremely flexible framework for modelling time series. The great descriptive power of these models comes at the expense of intractability: it is impossible to obtain analytic solutions to the inference problems of interest with the exception of a small number of particularly simple cases. The "particle" methods described by this tutorial are a broad and popular class of Monte Carlo algorithms which have been developed over the past fifteen years to provide approximate solutions to these intractable inference problems.

1.1 Preliminary remarks

Since their introduction in 1993 [22], particle filters have become a very popular class of numerical methods for the solution of optimal estimation problems in non-linear non-Gaussian scenarios. In comparison with standard approximation methods, such as the popular Extended Kalman Filter, the principal advantage of particle methods is that they do not rely on any local linearisation technique or any crude functional approximation. The price that must be paid for this flexibility is computational: these methods are computationally expensive. However, thanks to the availability of ever-increasing computational power, these methods are already used in real-time applications appearing in fields as diverse as chemical engineering, computer vision, financial econometrics, target tracking and robotics. Moreover, even in scenarios in which there are no real-time constraints, these methods can be a powerful alternative to Markov chain Monte Carlo (MCMC) algorithms — alternatively, they can be used to design very efficient MCMC schemes.

As a result of the popularity of particle methods, a few tutorials have already been published on the subject [3, 8, 18, 29]. The most popular, [3], dates back to 2002 and, like the edited volume [16] from 2001, it is now somewhat outdated. This tutorial differs from previously published tutorials in two ways. First, the obvious: it is, as of December 2008, the most recent tutorial on the subject and so it has been possible to include some very recent material on advanced particle methods for filtering and smoothing. Second, more importantly, this tutorial was not intended to resemble a cookbook. To this end, all of the algorithms are presented within a simple, unified framework. In particular, we show that essentially all basic and advanced methods for particle filtering can be reinterpreted as some special instances of a single generic Sequential Monte Carlo (SMC) algorithm. In our opinion, this framework is not only elegant but allows the development of a better intuitive and theoretical understanding of particle methods. It also shows that essentially any particle filter can be implemented using a simple computational framework such as that provided by [24]. Absolute beginners might benefit from reading [17], which provides an elementary introduction to the field, before the present tutorial.

1.2 Organisation of the tutorial

The rest of this paper is organised as follows. In Section 2, we present hidden Markov models and the associated Bayesian recursions for the filtering and smoothing distributions. In Section 3, we introduce a generic SMC algorithm which provides weighted samples from any sequence of probability distributions. In Section 4, we show how all the (basic and advanced) particle filtering methods developed in the literature can be interpreted as special instances of the generic SMC algorithm presented in Section 3. Section 5 is devoted to particle smoothing and we mention some open problems in Section 6.

2 Bayesian Inference in Hidden Markov Models

2.1 Hidden Markov Models and Inference Aims

Consider an $\mathcal{X}-$valued discrete-time Markov process $\{X_n\}_{n\geq 1}$ such that

$$X_1 \sim \mu(x_1) \text{ and } X_n|(X_{n-1}=x_{n-1}) \sim f(x_n|x_{n-1}) \tag{1}$$

where "$\sim$" means distributed according to, $\mu(x)$ is a probability density function and $f(x|x')$ denotes the probability density associated with moving from x' to x. All the densities are with respect to a dominating measure that we will denote, with abuse of notation, dx. We are interested in estimating $\{X_n\}_{n\geq 1}$ but only have access to the $\mathcal{Y}-$valued process $\{Y_n\}_{n\geq 1}$. We assume that, given $\{X_n\}_{n\geq 1}$, the observations $\{Y_n\}_{n\geq 1}$ are statistically independent and their marginal densities (with respect to a dominating measure dy_n) are given by

$$Y_n|(X_n=x_n) \sim g(y_n|x_n). \tag{2}$$

For the sake of simplicity, we have considered only the homogeneous case here; that is, the transition and observation densities are independent of the time index n. The extension to the inhomogeneous case is straightforward. It is assumed throughout that any model parameters are known.

Models compatible with (1)-(2) are known as hidden Markov models (HMM) or general state-space models (SSM). This class includes many models of interest. The following examples provide an illustration of several simple problems which can be dealt with within this framework. More complicated scenarios can also be considered.

Example 1 - *Finite State-Space HMM.* In this case, we have $\mathcal{X}=\{1,...,K\}$ so

$$\Pr(X_1=k)=\mu(k), \ \Pr(X_n=k|X_{n-1}=l)=f(k|l).$$

The observations follow an arbitrary model of the form (2). This type of model is extremely general and examples can be found in areas such as genetics in which they can describe imperfectly observed genetic sequences, signal processing, and computer science in which they can describe, amongst many other things, arbitrary finite-state machines.

Example 2 - *Linear Gaussian model.* Here, $\mathcal{X}=\mathbb{R}^{n_x}$, $\mathcal{Y}=\mathbb{R}^{n_y}$, $X_1 \sim \mathcal{N}(0,\Sigma)$ and

$$\begin{aligned} X_n &= AX_{n-1}+BV_n, \\ Y_n &= CX_n+DW_n \end{aligned}$$

where $V_n \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0,I_{n_v})$, $W_n \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0,I_{n_w})$ and A,B,C,D are matrices of appropriate dimensions. Note that $\mathcal{N}(m,\Sigma)$ denotes a Gaussian distribution of mean m and variance-covariance matrix Σ, whereas $\mathcal{N}(x;m,\Sigma)$ denotes the Gaussian density of argument x and similar statistics. In this case $\mu(x)=\mathcal{N}(x;0,\Sigma)$, $f(x'|x)=\mathcal{N}\left(x';Ax,BB^{\mathrm{T}}\right)$ and $g(y|x)=\mathcal{N}\left(y;Cx,DD^{\mathrm{T}}\right)$. As inference is analytically tractable for this model, it has been extremely widely used for problems such as target tracking and signal processing.

Example 3 - *Switching Linear Gaussian model.* We have $\mathcal{X}=\mathcal{U}\times\mathcal{Z}$ with $\mathcal{U}=\{1,...,K\}$ and $\mathcal{Z}=\mathbb{R}^{n_z}$. Here $X_n=(U_n,Z_n)$ where $\{U_n\}$ is a finite state-space Markov chain such that $\Pr(U_1=k)=\mu_U(k)$, $\Pr(U_n=k|U_{n-1}=l)=f_U(k|l)$ and conditional upon $\{U_n\}$ we have a linear Gaussian model with $Z_1|U_1 \sim \mathcal{N}(0,\Sigma_{U_1})$ and

$$\begin{aligned} Z_n &= A_{U_n}Z_{n-1}+B_{U_n}V_n, \\ Y_n &= C_{U_n}Z_n+D_{U_n}W_n \end{aligned}$$

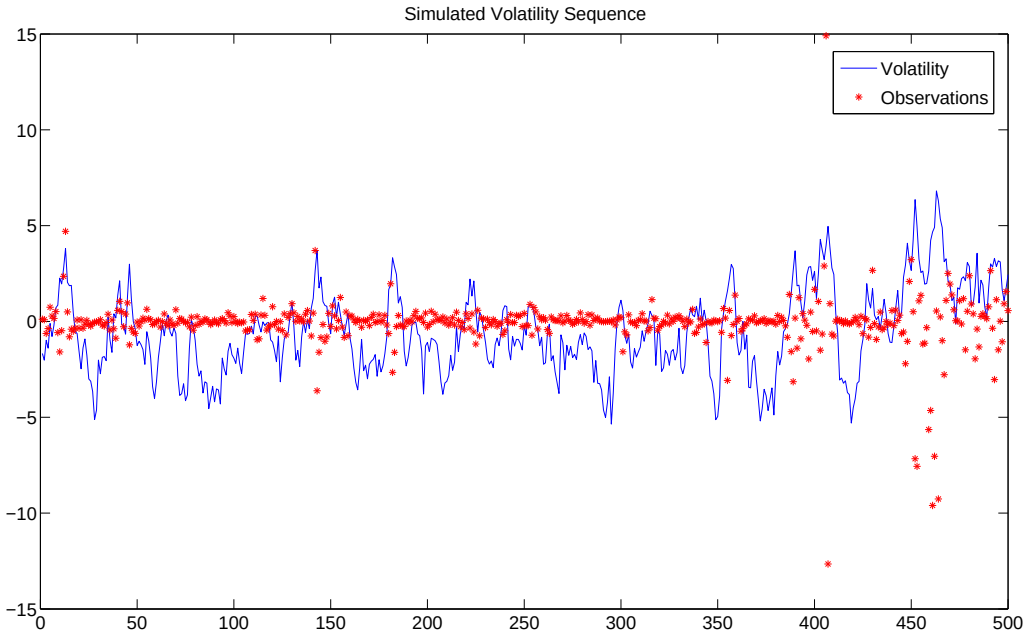


Figure 1: A simulation of the stochastic volatility model described in example 4.

where $V_n \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I_{n_v})$, $W_n \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I_{n_w})$ and $\{A_k, B_k, C_k, D_k; k = 1, ..., K\}$ are matrices of appropriate dimensions. In this case we have $\mu(x) = \mu(u, z) = \mu_U(u)\mathcal{N}(z; 0, \Sigma_u)$, $f(x'|x) = f((u', z')|(u, z)) = f_U(u'|u)\mathcal{N}\left(z'; A_{u'}z, B_{u'}B_{u'}^{\mathrm{T}}\right)$ and $g(y|x) = g(y|(u, z)) = \mathcal{N}\left(y; C_u z, D_u D_u^{\mathrm{T}}\right)$. This type of model provides a generalisation of that described in example 2 with only a slight increase in complexity.

Example 4 - *Stochastic Volatility model.* We have $\mathcal{X} = \mathcal{Y} = \mathbb{R}$, $X_1 \sim \mathcal{N}\left(0, \frac{\sigma^2}{1-\alpha^2}\right)$ and

$$\begin{aligned} X_n &= \alpha X_{n-1} + \sigma V_n, \\ Y_n &= \beta \exp(X_n/2) W_n \end{aligned}$$

where $V_n \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1)$ and $W_n \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1)$. In this case we have $\mu(x) = \mathcal{N}\left(x; 0, \frac{\sigma^2}{1-\alpha^2}\right)$, $f(x'|x) = \mathcal{N}\left(x'; \alpha x, \sigma^2\right)$ and $g(y|x) = \mathcal{N}\left(y; 0, \beta^2 \exp(x)\right)$. Note that this choice of initial distribution ensures that the marginal distribution of X_n is also $\mu(x)$ for all n. This type of model, and its generalisations, have been very widely used in various areas of economics and mathematical finance: inferring and predicting underlying volatility from observed price or rate data is an important problem. Figure 1 shows a short section of data simulated from such a model with parameter values $\alpha = 0.91, \sigma = 1.0$ and $\beta = 0.5$ which will be used below to illustrate the behaviour of some simple algorithms.

Equations (1)-(2) define a Bayesian model in which (1) defines the prior distribution of the process of interest $\{X_n\}_{n \geq 1}$ and (2) defines the likelihood function; that is:

$$p(x_{1:n}) = \mu(x_1) \prod_{k=2}^{n} f(x_k|x_{k-1}) \tag{3}$$

and

$$p(y_{1:n}|x_{1:n}) = \prod_{k=1}^{n} g(y_k|x_k), \tag{4}$$

where, for any sequence $\{z_n\}_{n\geq 1}$, and any $i \leq j$, $z_{i:j} := (z_i, z_{i+1}, ..., z_j)$.

In such a Bayesian context, inference about $X_{1:n}$ given a realisation of the observations $Y_{1:n} = y_{1:n}$ relies upon the posterior distribution

$$p(x_{1:n}|y_{1:n}) = \frac{p(x_{1:n}, y_{1:n})}{p(y_{1:n})}, \tag{5}$$

where

$$p(x_{1:n}, y_{1:n}) = p(x_{1:n})\, p(y_{1:n}|x_{1:n}), \tag{6}$$

$$\text{and } p(y_{1:n}) = \int p(x_{1:n}, y_{1:n})\, dx_{1:n}. \tag{7}$$

For the finite state-space HMM model discussed in Example 1, the integrals correspond to finite sums and all these (discrete) probability distributions can be computed exactly. For the linear Gaussian model discussed in Example 2, it is easy to check that $p(x_{1:n}|y_{1:n})$ is a Gaussian distribution whose mean and covariance can be computed using Kalman techniques; see [1], for example. However, for most non-linear non-Gaussian models, it is not possible to compute these distributions in closed-form and we need to employ numerical methods. Particle methods are a set of flexible and powerful simulation-based methods which provide samples approximately distributed according to posterior distributions of the form $p(x_{1:n}|y_{1:n})$ and facilitate the approximate calculation of $p(y_{1:n})$. Such methods are a subset of the class of methods known as Sequential Monte Carlo (SMC) methods.

In this tutorial, we will review various particle methods to address the following problems:

• *Filtering and Marginal likelihood computation*: Assume that we are interested in the sequential approximation of the distributions $\{p(x_{1:n}|y_{1:n})\}_{n\geq 1}$ and marginal likelihoods $\{p(y_{1:n})\}_{n\geq 1}$. That is, we wish to approximate $p(x_1|y_1)$ and $p(y_1)$ at the first time instance, $p(x_{1:2}|y_{1:2})$ and $p(y_{1:2})$ at the second time instance and so on. We will refer to this problem as the optimal filtering problem. This is slightly at variance with the usage in much of the literature which reserves the term for the estimation of the marginal distributions $\{p(x_n|y_{1:n})\}_{n\geq 1}$ rather than the joint distributions $\{p(x_{1:n}|y_{1:n})\}_{n\geq 1}$.

We will describe basic and advanced particle filtering methods to address this problem including auxiliary particle filtering, particle filtering with MCMC moves, block sampling strategies and Rao-Blackwellised particle filters.

• *Smoothing*: Consider attempting to sample from a joint distribution $p(x_{1:T}|y_{1:T})$ and approximating the associated marginals $\{p(x_n|y_{1:T})\}$ where $n = 1, ..., T$. Particle filtering techniques can be used to solve this problem but perform poorly when T is large for reasons detailed in this tutorial. We will describe several particle smoothing methods to address this problem. Essentially, these methods rely on the particle implementation of the forward filtering-backward smoothing formula or of a generalised version of the two-filter smoothing formula.

2.2 Filtering and Marginal Likelihood

The first area of interest, and that to which the vast majority of the literature on particle methods has been dedicated from the outset, is the problem of filtering: characterising the distribution of the state of the hidden Markov model at the present time, given the information provided by all of the observations received up to the present time. This can be thought of as a "tracking" problem: keeping track of the current "location" of the system given noisy observations — and, indeed, this is an extremely popular area

of application for these methods. The term is sometimes also used to refer to the practice of estimating the full trajectory of the state sequence up to the present time given the observations received up to this time.

We recall that, following (1)-(2), the posterior distribution $p(x_{1:n}|y_{1:n})$ is defined by (5) — the prior is defined in (3) and the likelihood in (4). The unnormalised posterior distribution $p(x_{1:n}, y_{1:n})$ given in (5) satisfies

$$p(x_{1:n}, y_{1:n}) = p(x_{1:n-1}, y_{1:n-1}) f(x_n|x_{n-1}) g(y_n|x_n). \tag{8}$$

Consequently, the posterior $p(x_{1:n}|y_{1:n})$ satisfies the following recursion

$$p(x_{1:n}|y_{1:n}) = p(x_{1:n-1}|y_{1:n-1}) \frac{f(x_n|x_{n-1}) g(y_n|x_n)}{p(y_n|y_{1:n-1})}, \tag{9}$$

where

$$p(y_n|y_{1:n-1}) = \int p(x_{n-1}|y_{1:n-1}) f(x_n|x_{n-1}) g(y_n|x_n) dx_{n-1:n} \tag{10}$$

In the literature, the recursion satisfied by the marginal distribution $p(x_n|y_{1:n})$ is often presented. It is straightforward to check (by integrating out $x_{1:n-1}$ in (9)) that we have

$$p(x_n|y_{1:n}) = \frac{g(y_n|x_n) p(x_n|y_{1:n-1})}{p(y_n|y_{1:n-1})}, \tag{11}$$

where

$$p(x_n|y_{1:n-1}) = \int f(x_n|x_{n-1}) p(x_{n-1}|y_{1:n-1}) dx_{n-1}. \tag{12}$$

Equation (12) is known as the prediction step and (11) is known as the updating step. However, most particle filtering methods rely on a numerical approximation of recursion (9) and not of (11)-(12).

If we can compute $\{p(x_{1:n}|y_{1:n})\}$ and thus $\{p(x_n|y_{1:n})\}$ sequentially, then the quantity $p(y_{1:n})$, which is known as the marginal likelihood, can also clearly be evaluated recursively using

$$p(y_{1:n}) = p(y_1) \prod_{k=2}^{n} p(y_k|y_{1:k-1}) \tag{13}$$

where $p(y_k|y_{1:k-1})$ is of the form (10).

2.3 Smoothing

One problem, which is closely related to filtering, but computationally more challenging for reasons which will become apparent later, is known as smoothing. Whereas filtering corresponds to estimating the distribution of the current state of an HMM based upon the observations received up until the current time, smoothing corresponds to estimating the distribution of the state at a particular time given all of the observations up to some *later* time. The trajectory estimates obtained by such methods, as a result of the additional information available, tend to be smoother than those obtained by filtering. It is intuitive that if estimates of the state at time n are not required instantly, then better estimation performance is likely to be obtained by taking advantage of a few later observations. Designing efficient sequential algorithms for the solution of this problem is not quite as straightforward as it might seem, but a number of effective strategies have been developed and are described below.

More formally: assume that we have access to the data $y_{1:T}$, and wish to compute the marginal distributions $\{p(x_n|y_{1:T})\}$ where $n = 1, ..., T$ or to sample from $p(x_{1:T}|y_{1:T})$. In principle, the marginals $\{p(x_n|y_{1:T})\}$ could be obtained directly by considering the joint distribution $p(x_{1:T}|y_{1:T})$ and integrating out the variables $(x_{1:n-1}, x_{n+1:T})$. Extending this reasoning in the context of particle methods, one can simply use the identity $p(x_n|y_{1:T}) = \int p(x_{1:T}|y_{1:T}) dx_{1:n-1} dx_{n+1:T}$ and take the same approach which is used in particle filtering:

use Monte Carlo algorithms to obtain an approximate characterisation of the joint distribution and then use the associated marginal distribution to approximate the distributions of interest. Unfortunately, as is detailed below, when $n \ll T$ this strategy is doomed to failure: the marginal distribution $p(x_n|y_{1:n})$ occupies a privileged role within the particle filter framework as it is, in some sense, better characterised than any of the other marginal distributions.

For this reason, it is necessary to develop more sophisticated strategies in order to obtain good smoothing algorithms. There has been much progress in this direction over the past decade. Below, we present two alternative recursions that will prove useful when numerical approximations are required. The key to the success of these recursions is that they rely upon only the marginal filtering distributions $\{p(x_n|y_{1:n})\}$.

2.3.1 Forward-Backward Recursions

The following decomposition of the joint distribution $p(x_{1:T}|y_{1:T})$

$$\begin{aligned} p(x_{1:T}|y_{1:T}) &= p(x_T|y_{1:T}) \prod_{n=1}^{T-1} p(x_n|x_{n+1}, y_{1:T}) \\ &= p(x_T|y_{1:T}) \prod_{n=1}^{T-1} p(x_n|x_{n+1}, y_{1:n}), \end{aligned} \tag{14}$$

shows that, conditional on $y_{1:T}$, $\{X_n\}$ is an inhomogeneous Markov process.

Eq. (14) suggests the following algorithm to sample from $p(x_{1:T}|y_{1:T})$. First compute and store the marginal distributions $\{p(x_n|y_{1:n})\}$ for $n = 1, ..., T$. Then sample $X_T \sim p(x_T|y_{1:T})$ and for $n = T-1, T-2, ..., 1$, sample $X_n \sim p(x_n|X_{n+1}, y_{1:n})$ where

$$p(x_n|x_{n+1}, y_{1:n}) = \frac{f(x_{n+1}|x_n)\, p(x_n|y_{1:n})}{p(x_{n+1}|y_{1:n})}$$

It also follows, by integrating out $(x_{1:n-1}, x_{n+1:T})$ in Eq. (14), that

$$p(x_n|y_{1:T}) = p(x_n|y_{1:n}) \int \frac{f(x_{n+1}|x_n)}{p(x_{n+1}|y_{1:n})} p(x_{n+1}|y_{1:T})\, dx_{n+1}. \tag{15}$$

So to compute $\{p(x_n|y_{1:T})\}$, we simply modify the backward pass and, instead of sampling from $p(x_n|x_{n+1}, y_{1:n})$, we compute $p(x_n|y_{1:T})$ using (15).

2.3.2 Generalised Two-Filter Formula

The two-filter formula is a well-established alternative to the forward-filtering backward-smoothing technique to compute the marginal distributions $\{p(x_n|y_{1:T})\}$ [4]. It relies on the following identity

$$p(x_n|y_{1:T}) = \frac{p(x_n|y_{1:n-1})\, p(y_{n:T}|x_n)}{p(y_{n:T}|y_{1:n-1})},$$

where the so-called backward information filter is initialised at time $n = T$ by $p(y_T|x_T) = g(y_T|x_T)$ and satisfies

$$\begin{aligned} p(y_{n:T}|x_n) &= \int \prod_{k=n+1}^{T} f(x_k|x_{k-1}) \prod_{k=n}^{T} g(y_k|x_k)\, dx_{n+1:T} \\ &= g(y_n|x_n) \int f(x_{n+1}|x_n)\, p(y_{n+1:T}|x_{n+1})\, dx_{n+1}. \end{aligned} \tag{16}$$

The backward information filter is not a probability density in argument x_n and it is even possible that $\int p(y_{n:T}|x_n)\,dx_n = \infty$. Although this is not an issue when $p(y_{n:T}|x_n)$ can be computed exactly, it does preclude the direct use of SMC methods to estimate this integral. To address this problem, a generalised version of the two-filter formula was proposed in [5]. It relies on the introduction of a set of artificial probability distributions $\{\widetilde{p}_n(x_n)\}$ and the joint distributions

$$\widetilde{p}_n(x_{n:T}|y_{n:T}) \propto \widetilde{p}_n(x_n) \prod_{k=n+1}^{T} f(x_k|x_{k-1}) \prod_{k=n}^{T} g(y_k|x_k), \tag{17}$$

which are constructed such that their marginal distributions, $\widetilde{p}_n(x_n|y_{n:T}) \propto \widetilde{p}_n(x_n)\,p(y_{n:T}|x_n)$, are simply "integrable versions" of the backward information filter. It is easy to establish the generalised two-filter formula

$$p(x_1|y_{1:T}) \propto \frac{\mu(x_1)\,\widetilde{p}(x_1|y_{1:T})}{\widetilde{p}_1(x_1)},\ p(x_n|y_{1:T}) \propto \frac{p(x_n|y_{1:n-1})\widetilde{p}(x_n|y_{n:T})}{\widetilde{p}_n(x_n)} \tag{18}$$

which is valid whenever the support of $\widetilde{p}_n(x_n)$ includes the support of the prior $p_n(x_n)$; that is

$$p_n(x_n) = \int \mu(x_1) \prod_{k=2}^{n} f(x_k|x_{k-1})\,dx_{1:n-1} > 0 \Rightarrow \widetilde{p}_n(x_n) > 0.$$

The generalised two-filter smoother for $\{p(x_n|y_{n:T})\}$ proceeds as follows. Using the standard forward recursion, we can compute and store the marginal distributions $\{p(x_n|y_{1:n-1})\}$. Using a backward recursion, we compute and store $\{\widetilde{p}(x_n|y_{n:T})\}$. Then for any $n = 1, ..., T$ we can combine $p(x_n|y_{1:n-1})$ and $\widetilde{p}(x_n|y_{n:T})$ to obtain $p(x_n|y_{1:T})$.

In [4], this identity is discussed in the particular case where $\widetilde{p}_n(x_n) = p_n(x_n)$. However, when computing $\{\widetilde{p}(x_n|y_{n:T})\}$ using SMC, it is necessary to be able to compute $\widetilde{p}_n(x_n)$ exactly hence this rules out the choice $\widetilde{p}_n(x_n) = p_n(x_n)$ for most non-linear non-Gaussian models. In practice, we should select a heavy-tailed approximation of $p_n(x_n)$ for $\widetilde{p}_n(x_n)$ in such settings. It is also possible to use the generalised-two filter formula to sample from $p(x_{1:T}|y_{1:T})$; see [5] for details.

2.4 Summary

Bayesian inference in non-linear non-Gaussian dynamic models relies on the sequence of posterior distributions $\{p(x_{1:n}|y_{1:n})\}$ and its marginals. Except in simple problems such as Examples 1 and 2, it is not possible to compute these distributions in closed-form. In some scenarios, it might be possible to obtain reasonable performance by employing functional approximations of these distributions. Here, we will discuss only Monte Carlo approximations of these distributions; that is numerical schemes in which the distributions of interest are approximated by a large collection of N random samples termed particles. The main advantage of such methods is that under weak assumptions they provide asymptotically (i.e. as $N \to \infty$) consistent estimates of the target distributions of interest. It is also noteworthy that these techniques can be applied to problems of moderately-high dimension in which traditional numerical integration might be expected to perform poorly.

3 Sequential Monte Carlo Methods

Over the past fifteen years, particle methods for filtering and smoothing have been the most common examples of SMC algorithms. Indeed, it has become traditional to present particle filtering and SMC as being the same thing in much of the literature. Here, we wish to emphasise that SMC actually encompasses a broader range of algorithms — and by doing so we are able to show that many more advanced techniques for

approximate filtering and smoothing can be described using precisely the same framework and terminology as the basic algorithm.

SMC methods are a general class of Monte Carlo methods that sample sequentially from a sequence of target probability densities $\{\pi_n(x_{1:n})\}$ of increasing dimension where each distribution $\pi_n(x_{1:n})$ is defined on the product space $\mathcal{X}^n$. Writing

$$\pi_n(x_{1:n}) = \frac{\gamma_n(x_{1:n})}{Z_n} \tag{19}$$

we require only that $\gamma_n : \mathcal{X}^n \to \mathbb{R}^+$ is known pointwise; the normalising constant

$$Z_n = \int \gamma_n(x_{1:n})\, dx_{1:n} \tag{20}$$

might be unknown. SMC provide an approximation of $\pi_1(x_1)$ and an estimate of Z_1 at time 1 then an approximation of $\pi_2(x_{1:2})$ and an estimate of Z_2 at time 2 and so on.

For example, in the context of filtering, we could have $\gamma_n(x_{1:n}) = p(x_{1:n}, y_{1:n})$, $Z_n = p(y_{1:n})$ so $\pi_n(x_{1:n}) = p(x_{1:n}|y_{1:n})$. However, we emphasise that this is just one particular choice of target distributions. Not only can SMC methods be used outside the filtering context but, more importantly for this tutorial, some advanced particle filtering and smoothing methods discussed below do not rely on this sequence of target distributions. Consequently, we believe that understanding the main principles behind generic SMC methods is essential to the development of a proper understanding of particle filtering and smoothing methods.

We start this section with a very basic review of Monte Carlo methods and Importance Sampling (IS). We then present the Sequential Importance Sampling (SIS) method, point out the limitations of this method and show how resampling techniques can be used to partially mitigate them. Having introduced the basic particle filter as an SMC method, we show how various advanced techniques which have been developed over the past fifteen years can themselves be interpreted within the same formalism as SMC algorithms associated with sequences of distributions which may not coincide with the filtering distributions. These alternative sequences of target distributions are either constructed such that they admit the distributions $\{p(x_{1:n}|y_{1:n})\}$ as marginal distributions, or an importance sampling correction is necessary to ensure the consistency of estimates.

3.1 Basics of Monte Carlo Methods

Initially, consider approximating a generic probability density $\pi_n(x_{1:n})$ for some fixed n. If we sample N independent random variables, $X_{1:n}^i \sim \pi_n(x_{1:n})$ for $i = 1, ..., N$, then the Monte Carlo method approximates $\pi_n(x_{1:n})$ by the empirical measure[1]

$$\widehat{\pi}_n(x_{1:n}) = \frac{1}{N}\sum_{i=1}^{N} \delta_{X_{1:n}^i}(x_{1:n}),$$

where $\delta_{x_0}(x)$ denotes the Dirac delta mass located at x_0. Based on this approximation, it is possible to approximate any marginal, say $\pi_n(x_k)$, easily using

$$\widehat{\pi}_n(x_k) = \frac{1}{N}\sum_{i=1}^{N} \delta_{X_k^i}(x_k),$$

and the expectation of any test function $\varphi_n : \mathcal{X}^n \to \mathbb{R}$ given by

$$I_n(\varphi_n) := \int \varphi_n(x_{1:n})\, \pi_n(x_{1:n})\, dx_{1:n},$$

[1] We persist with the abusive use of density notation in the interests of simplicity and accessibility; the alternations required to obtain a rigorous formulation are obvious.

is estimated by

$$I_n^{\mathrm{MC}}(\varphi_n) := \int \varphi_n(x_{1:n})\,\widehat{\pi}_n(x_{1:n})\,dx_{1:n} = \frac{1}{N}\sum_{i=1}^{N}\varphi_n\left(X_{1:n}^i\right).$$

It is easy to check that this estimate is unbiased and that its variance is given by

$$\mathbb{V}\left[I_n^{\mathrm{MC}}(\varphi_n)\right] = \frac{1}{N}\left(\int \varphi_n^2(x_{1:n})\,\pi_n(x_{1:n})\,dx_{1:n} - I_n^2(\varphi_n)\right).$$

The main advantage of Monte Carlo methods over standard approximation techniques is that the variance of the approximation error decreases at a rate of $\mathcal{O}(1/N)$ regardless of the dimension of the space $\mathcal{X}^n$. However, there are at least two main problems with this basic Monte Carlo approach:

• *Problem 1*: If $\pi_n(x_{1:n})$ is a complex high-dimensional probability distribution, then we cannot sample from it.

• *Problem 2*: Even if we knew how to sample exactly from $\pi_n(x_{1:n})$, the computational complexity of such a sampling scheme is typically at least linear in the number of variables n. So an algorithm sampling exactly from $\pi_n(x_{1:n})$, sequentially for each value of n, would have a computational complexity increasing at least linearly with n.

3.2 Importance Sampling

We are going to address *Problem 1* using the IS method. This is a fundamental Monte Carlo method and the basis of all the algorithms developed later on. IS relies on the introduction of an *importance density*[2] $q_n(x_{1:n})$ such that

$$\pi_n(x_{1:n}) > 0 \Rightarrow q_n(x_{1:n}) > 0.$$

In this case, we have from (19)-(20) the following IS identities

$$\pi_n(x_{1:n}) = \frac{w_n(x_{1:n})\,q_n(x_{1:n})}{Z_n}, \tag{21}$$

$$Z_n = \int w_n(x_{1:n})\,q_n(x_{1:n})\,dx_{1:n} \tag{22}$$

where $w_n(x_{1:n})$ is the *unnormalised weight* function

$$w_n(x_{1:n}) = \frac{\gamma_n(x_{1:n})}{q_n(x_{1:n})}.$$

In particular, we can select an importance density $q_n(x_{1:n})$ from which it is easy to draw samples; e.g. a multivariate Gaussian. Assume we draw N independent samples $X_{1:n}^i \sim q_n(x_{1:n})$ then by inserting the Monte Carlo approximation of $q_n(x_{1:n})$ — that is the empirical measure of the samples $X_{1:n}^i$ — into (21)–(22) we obtain

$$\widehat{\pi}_n(x_{1:n}) = \sum_{i=1}^{N} W_n^i \delta_{X_{1:n}^i}(x_{1:n}), \tag{23}$$

$$\widehat{Z}_n = \frac{1}{N}\sum_{i=1}^{N} w_n\left(X_{1:n}^i\right) \tag{24}$$

where

$$W_n^i = \frac{w_n\left(X_{1:n}^i\right)}{\sum_{j=1}^{N} w_n\left(X_{1:n}^j\right)}. \tag{25}$$

[2]Some authors use the terms *proposal density* or *instrumental density* interchangeably.

Compared to standard Monte Carlo, IS provides an (unbiased) estimate of the normalising constant with relative variance

$$\frac{\mathbb{V}_{\mathrm{IS}}\left[\widehat{Z}_n\right]}{Z_n^2} = \frac{1}{N}\left(\int \frac{\pi_n^2\left(x_{1:n}\right)}{q_n\left(x_{1:n}\right)} dx_{1:n} - 1\right). \tag{26}$$

If we are interested in computing $I_n\left(\varphi_n\right)$, we can also use the estimate

$$I_n^{\mathrm{IS}}\left(\varphi_n\right) = \int \varphi_n\left(x_{1:n}\right) \widehat{\pi}_n\left(x_{1:n}\right) dx_{1:n} = \sum_{i=1}^{N} W_n^i \varphi_n\left(X_{1:n}^i\right).$$

Unlike $I_n^{\mathrm{MC}}\left(\varphi_n\right)$, this estimate is biased for finite N. However, it is consistent and it is easy to check that its asymptotic bias is given by

$$\lim_{N\to\infty} N\left(I_n^{\mathrm{IS}}\left(\varphi_n\right) - I_n\left(\varphi_n\right)\right) = -\int \frac{\pi_n^2\left(x_{1:n}\right)}{q_n\left(x_{1:n}\right)}\left(\varphi_n\left(x_{1:n}\right) - I_n\left(\varphi_n\right)\right) dx_{1:n}.$$

When the normalising constant is known analytically, we can calculate an unbiased importance sampling estimate — however, this generally has higher variance and this is not typically the case in the situations in which we are interested.

Furthermore, $I_n^{\mathrm{IS}}(\varphi)$ satisfies a Central Limit Theorem (CLT) with asymptotic variance

$$\frac{1}{N}\int \frac{\pi_n^2\left(x_{1:n}\right)}{q_n\left(x_{1:n}\right)}\left(\varphi_n\left(x_{1:n}\right) - I_n\left(\varphi_n\right)\right)^2 dx_{1:n}. \tag{27}$$

The bias being $\mathcal{O}(1/N)$ and the variance $\mathcal{O}(1/N)$, the mean-squared error given by the *squared* bias plus the variance is asymptotically dominated by the variance term.

For a given test function, $\varphi_n\left(x_{1:n}\right)$, it is easy to establish the importance distribution minimising the asymptotic variance of $I_n^{\mathrm{IS}}(\varphi_n)$. However, such a result is of minimal interest in a filtering context as this distribution depends on $\varphi_n\left(x_{1:n}\right)$ and we are typically interested in the expectations of several test functions. Moreover, even if we were interested in a single test function, say $\varphi_n\left(x_{1:n}\right) = x_n$, then selecting the optimal importance distribution at time n would have detrimental effects when we will try to obtain a sequential version of the algorithms (the optimal distribution for estimating $\varphi_{n-1}(x_{1:n-1})$ will almost certainly not be — even similar to — the marginal distribution of $x_{1:n-1}$ in the optimal distribution for estimating $\varphi_n(x_{1:n})$ and this will prove to be problematic).

A more appropriate approach in this context is to attempt to select the $q_n\left(x_{1:n}\right)$ which minimises the variance of the importance weights (or, equivalently, the variance of $\widehat{Z}_n$). Clearly, this variance is minimised for $q_n\left(x_{1:n}\right) = \pi_n\left(x_{1:n}\right)$. We cannot select $q_n\left(x_{1:n}\right) = \pi_n\left(x_{1:n}\right)$ as this is the reason we used IS in the first place. However, this simple result indicates that we should aim at selecting an IS distribution which is close as possible to the target. Also, although it is possible to construct samplers for which the variance is finite without satisfying this condition, it is advisable to select $q_n\left(x_{1:n}\right)$ so that $w_n\left(x_{1:n}\right) < C_n < \infty$.

3.3 Sequential Importance Sampling

We are now going to present an algorithm that admits a fixed computational complexity at each time step in important scenarios and thus addresses *Problem 2*. This solution involves selecting an importance distribution which has the following structure

$$\begin{aligned} q_n\left(x_{1:n}\right) &= q_{n-1}\left(x_{1:n-1}\right) q_n\left(x_n \middle| x_{1:n-1}\right) \\ &= q_1\left(x_1\right) \prod_{k=2}^{n} q_k\left(x_k \middle| x_{1:k-1}\right). \end{aligned} \tag{28}$$

Practically, this means that to obtain particles $X_{1:n}^i \sim q_n(x_{1:n})$ at time n, we sample $X_1^i \sim q_1(x_1)$ at time 1 then $X_k^i \sim q_k\left(x_k | X_{1:k-1}^i\right)$ at time k for $k = 2, ..., n$. The associated unnormalised weights can be computed recursively using the decomposition

$$\begin{aligned} w_n(x_{1:n}) &= \frac{\gamma_n(x_{1:n})}{q_n(x_{1:n})} \\ &= \frac{\gamma_{n-1}(x_{1:n-1})}{q_{n-1}(x_{1:n-1})} \frac{\gamma_n(x_{1:n})}{\gamma_{n-1}(x_{1:n-1}) q_n(x_n | x_{1:n-1})} \end{aligned} \tag{29}$$

which can be written in the form

$$\begin{aligned} w_n(x_{1:n}) &= w_{n-1}(x_{1:n-1}) \cdot \alpha_n(x_{1:n}) \\ &= w_1(x_1) \prod_{k=2}^{n} \alpha_k(x_{1:k}) \end{aligned}$$

where the *incremental importance weight* function $\alpha_n(x_{1:n})$ is given by

$$\alpha_n(x_{1:n}) = \frac{\gamma_n(x_{1:n})}{\gamma_{n-1}(x_{1:n-1}) q_n(x_n | x_{1:n-1})}. \tag{30}$$

The SIS algorithm proceeds as follows, with each step carried out for $i = 1, \ldots, N$:

Sequential Importance Sampling

At time $n = 1$

- Sample $X_1^i \sim q_1(x_1)$.
- Compute the weights $w_1\left(X_1^i\right)$ and $W_1^i \propto w_1\left(X_1^i\right)$.

At time $n \geq 2$

- Sample $X_n^i \sim q_n(x_n | X_{1:n-1}^i)$.
- Compute the weights

$$\begin{aligned} w_n\left(X_{1:n}^i\right) &= w_{n-1}\left(X_{1:n-1}^i\right) \cdot \alpha_n\left(X_{1:n}^i\right), \\ W_n^i &\propto w_n\left(X_{1:n}^i\right). \end{aligned}$$

At any time, n, we obtain the estimates $\widehat{\pi}_n(x_{1:n})$ (Eq. 23) and $\widehat{Z}_n$ (Eq. 24) of $\pi_n(x_{1:n})$ and Z_n, respectively. It is straightforward to check that a consistent estimate of Z_n/Z_{n-1} is also provided by the same set of samples:

$$\widehat{\frac{Z_n}{Z_{n-1}}} = \sum_{i=1}^{N} W_{n-1}^i \alpha_n\left(X_{1:n}^i\right).$$

This estimator is motivated by the fact that

$$\int \alpha_n(x_{1:n}) \pi_{n-1}(x_{1:n-1}) q_n(x_n | x_{1:n-1}) dx_{1:n} = \int \frac{\gamma_n(x_{1:n}) \pi_{n-1}(x_{1:n-1}) q_n(x_n | x_{1:n-1})}{\gamma_{n-1}(x_{1:n-1}) q_n(x_n | x_{1:n-1})} dx_{1:n} = \frac{Z_n}{Z_{n-1}}.$$

In this sequential framework, it would seem that the only freedom the user has at time n is the choice of $q_n\left(x_n \middle| x_{1:n-1}\right)$[3]. A sensible strategy consists of selecting it so as to minimise the variance of $w_n\left(x_{1:n}\right)$. It is straightforward to check that this is achieved by selecting

$$q_n^{\mathrm{opt}}\left(x_n \middle| x_{1:n-1}\right) = \pi_n\left(x_n \middle| x_{1:n-1}\right)$$

as in this case the variance of $w_n\left(x_{1:n}\right)$ conditional upon $x_{1:n-1}$ is zero and the associated incremental weight is given by

$$\alpha_n^{\mathrm{opt}}\left(x_{1:n}\right) = \frac{\gamma_n\left(x_{1:n-1}\right)}{\gamma_{n-1}\left(x_{1:n-1}\right)} = \frac{\int \gamma_n\left(x_{1:n}\right) dx_n}{\gamma_{n-1}\left(x_{1:n-1}\right)}.$$

Note that it is not always possible to sample from $\pi_n\left(x_n \middle| x_{1:n-1}\right)$. Nor is it always possible to compute $\alpha_n^{\mathrm{opt}}\left(x_{1:n}\right)$. In these cases, one should employ an approximation of $q_n^{\mathrm{opt}}\left(x_n \middle| x_{1:n-1}\right)$ for $q_n\left(x_n \middle| x_{1:n-1}\right)$.

In those scenarios in which the time required to sample from $q_n\left(x_n \middle| x_{1:n-1}\right)$ and to compute $\alpha_n\left(x_{1:n}\right)$ is independent of n (and this is, indeed, the case if q_n is chosen sensibly and one is concerned with a problem such as filtering), it appears that we have provided a solution for *Problem 2*. However, it is important to be aware that the methodology presented here suffers from severe drawbacks. Even for standard IS, the variance of the resulting estimates increases exponentially with n (as is illustrated below; see also [28]). As SIS is nothing but a special version of IS in which we restrict ourselves to an importance distribution of the form (28) it suffers from the same problem. We demonstrate this using a very simple toy example.

Example. Consider the case where $\mathcal{X} = \mathbb{R}$ and

$$\begin{aligned} \pi_n\left(x_{1:n}\right) &= \prod_{k=1}^{n} \pi_n\left(x_k\right) = \prod_{k=1}^{n} \mathcal{N}\left(x_k; 0, 1\right), \\ \gamma_n\left(x_{1:n}\right) &= \prod_{k=1}^{n} \exp\left(-\frac{x_k^2}{2}\right), \\ Z_n &= \left(2\pi\right)^{n/2}. \end{aligned} \tag{31}$$

We select an importance distribution

$$q_n\left(x_{1:n}\right) = \prod_{k=1}^{n} q_k\left(x_k\right) = \prod_{k=1}^{n} \mathcal{N}\left(x_k; 0, \sigma^2\right).$$

In this case, we have $\mathbb{V}_{\mathrm{IS}}\left[\widehat{Z}_n\right] < \infty$ only for $\sigma^2 > \frac{1}{2}$ and

$$\frac{\mathbb{V}_{\mathrm{IS}}\left[\widehat{Z}_n\right]}{Z_n^2} = \frac{1}{N}\left[\left(\frac{\sigma^4}{2\sigma^2 - 1}\right)^{n/2} - 1\right].$$

It can easily be checked that $\frac{\sigma^4}{2\sigma^2-1} > 1$ for any $\frac{1}{2} < \sigma^2 \neq 1$: the variance increases exponentially with n even in this simple case. For example, if we select $\sigma^2 = 1.2$ then we have a reasonably good importance distribution as $q_k\left(x_k\right) \approx \pi_n\left(x_k\right)$ but $N\frac{\mathbb{V}_{\mathrm{IS}}\left[\widehat{Z}_n\right]}{Z_n^2} \approx \left(1.103\right)^{n/2}$ which is approximately equal to 1.9×10^{21} for $n = 1000$! We would need to use $N \approx 2 \times 10^{23}$ particles to obtain a relative variance $\frac{\mathbb{V}_{\mathrm{IS}}\left[\widehat{Z}_n\right]}{Z_n^2} = 0.01$. This is clearly impracticable.

[3]However, as we will see later, the key to many advanced SMC methods is the introduction of a sequence of target distributions which differ from the original target distributions.

3.4 Resampling

We have seen that IS — and thus SIS — provides estimates whose variance increases, typically exponentially, with n. Resampling techniques are a key ingredient of SMC methods which (partially) solve this problem in some important scenarios.

Resampling is a very intuitive idea which has major practical and theoretical benefits. Consider first an IS approximation $\widehat{\pi}_n(x_{1:n})$ of the target distribution $\pi_n(x_{1:n})$. This approximation is based on weighted samples from $q_n(x_{1:n})$ and does not provide samples approximately distributed according to $\pi_n(x_{1:n})$. To obtain approximate samples from $\pi_n(x_{1:n})$, we can simply sample from its IS approximation $\widehat{\pi}_n(x_{1:n})$; that is we select $X_{1:n}^i$ with probability W_n^i. This operation is called resampling as it corresponds to sampling from an approximation $\widehat{\pi}_n(x_{1:n})$ which was itself obtained by sampling. If we are interested in obtaining N samples from $\widehat{\pi}_n(x_{1:n})$, then we can simply resample N times from $\widehat{\pi}_n(x_{1:n})$. This is equivalent to associating a number of offspring N_n^i with each particle $X_{1:n}^i$ in such a way that $N_n^{1:N} = \left(N_n^1, ..., N_n^N\right)$ follow a multinomial distribution with parameter vector $\left(N, W_n^{1:N}\right)$ and associating a weight of $1/N$ with each offspring. We approximate $\widehat{\pi}_n(x_{1:n})$ by the resampled empirical measure

$$\overline{\pi}_n(x_{1:n}) = \sum_{i=1}^{N} \frac{N_n^i}{N} \delta_{X_{1:n}^i}(x_{1:n}) \tag{32}$$

where $\mathbb{E}\left[N_n^i \middle| W_n^{1:N}\right] = NW_n^i$. Hence $\overline{\pi}_n(x_{1:n})$ is an unbiased approximation of $\widehat{\pi}_n(x_{1:n})$.

Improved unbiased resampling schemes have been proposed in the literature. These are methods of selecting N_n^i such that the unbiasedness property is preserved, and such that $\mathbb{V}\left[N_n^i \middle| W_n^{1:N}\right]$ is smaller than that obtained via the multinomial resampling scheme described above. To summarize, the three most popular algorithms found in the literature are, in descending order of popularity/efficiency:

Systematic Resampling Sample $U_1 \sim \mathcal{U}\left[0, \frac{1}{N}\right]$ and define $U_i = U_1 + \frac{i-1}{N}$ for $i = 2, ..., N$, then set $N_n^i = \left|\left\{U_j : \sum_{k=1}^{i-1} W_n^k \leq U_j \leq \sum_{k=1}^{i} W_n^k\right\}\right|$ with the convention $\sum_{k=1}^{0} := 0$. It is straightforward to establish that this approach is unbiased.

Residual Resampling Set $\widetilde{N}_n^i = \lfloor NW_n^i \rfloor$, sample $\overline{N}_n^{1:N}$ from a multinomial of parameters $\left(N, \overline{W}_n^{1:N}\right)$ where $\overline{W}_n^i \propto W_n^i - N^{-1}\widetilde{N}_n^i$ then set $N_n^i = \widetilde{N}_n^i + \overline{N}_n^i$. This is very closely related to breaking the empirical CDF up into N components and then sampling once from each of those components: the stratified resampling approach of [7].

Multinomial Resampling Sample $N_n^{1:N}$ from a multinomial of parameters $\left(N, W_n^{1:N}\right)$.

Note that it is possible to sample efficiently from a multinomial distribution in $\mathcal{O}(N)$ operations. However, the systematic resampling algorithm introduced in [25] is the most widely-used algorithm in the literature as it is extremely easy to implement and outperforms other resampling schemes in most scenarios (although this is not guaranteed in general [13]).

Resampling allows us to obtain samples distributed approximately according to $\pi_n(x_{1:n})$, but it should be clear that if we are interested in estimating $I_n(\varphi_n)$ then we will obtain an estimate with lower variance using $\widehat{\pi}_n(x_{1:n})$ than that which we would have obtained by using $\overline{\pi}_n(x_{1:n})$. By resampling we indeed add some extra "noise"— as shown by [9]. However, an important advantage of resampling is that it allows us to remove of particles with low weights with a high probability. In the sequential framework in which we are interested, this is extremely useful as we do not want to carry forward particles with low weights and we want to focus our computational efforts on regions of high probability mass. Clearly, there is always the possibility than a particle having a low weight at time n could have an high weight at time $n+1$, in which case resampling could be wasteful. It is straightforward to consider artificial problems for which this

is the case. However, we will show that in the estimation problems we are looking at the resampling step is provably beneficial. Intuitively, resampling can be seen to provide stability in the future at the cost of an increase in the immediate Monte Carlo variance. This concept will be made more precise in section 3.6.

3.5 A Generic Sequential Monte Carlo Algorithm

SMC methods are a combination of SIS and resampling. At time 1, we compute the IS approximation $\widehat{\pi}_1(x_1)$ of $\pi_1(x_1)$ which is a weighted collection of particles $\left\{W_1^i, X_1^i\right\}$. Then we use a resampling step to eliminate (with high probability) those particles with low weights and multiply those with high weights. We denote by $\left\{\frac{1}{N}, \overline{X}_1^i\right\}$ the collection of equally-weighted resampled particles. Remember that each original particle X_1^i has N_1^i offspring so there exist N_1^i distinct indexes $j_1 \neq j_2 \neq \cdots \neq j_{N_1^i}$ such that $\overline{X}_1^{j_1} = \overline{X}_1^{j_2} = \cdots = \overline{X}_1^{j_{N_1^i}} = X_1^i$. After the resampling step, we follow the SIS strategy and sample $X_2^i \sim q_2(x_2|\,\overline{X}_1^i)$. Thus $\left(\overline{X}_1^i, X_2^i\right)$ is approximately distributed according to $\pi_1(x_1)\, q_2(x_2|\,x_1)$. Hence the corresponding importance weights in this case are simply equal to the incremental weights $\alpha_2(x_{1:2})$. We then resample the particles with respect to these normalised weights and so on. To summarise, the algorithm proceeds as follows (this algorithm is sometimes referred to as Sequential Importance Resampling (SIR) or Sequential Importance Sampling and Resampling (SIS/R)).

Sequential Monte Carlo

At time $n = 1$

- Sample $X_1^i \sim q_1(x_1)$.
- Compute the weights $w_1\left(X_1^i\right)$ and $W_1^i \propto w_1\left(X_1^i\right)$.
- Resample $\left\{W_1^i, X_1^i\right\}$ to obtain N equally-weighted particles $\left\{\frac{1}{N}, \overline{X}_1^i\right\}$.

At time $n \geq 2$

- Sample $X_n^i \sim q_n(x_n|\,\overline{X}_{1:n-1}^i)$ and set $X_{1:n}^i \leftarrow \left(\overline{X}_{1:n-1}^i, X_n^i\right)$.
- Compute the weights $\alpha_n\left(X_{1:n}^i\right)$ and $W_n^i \propto \alpha_n\left(X_{1:n}^i\right)$.
- Resample $\left\{W_n^i, X_{1:n}^i\right\}$ to obtain N new equally-weighted particles $\left\{\frac{1}{N}, \overline{X}_{1:n}^i\right\}$.

At any time n, this algorithm provides two approximations of $\pi_n(x_{1:n})$. We obtain

$$\widehat{\pi}_n(x_{1:n}) = \sum_{i=1}^{N} W_n^i \delta_{X_{1:n}^i}(x_{1:n}) \tag{33}$$

after the sampling step and

$$\overline{\pi}_n(x_{1:n}) = \frac{1}{N} \sum_{i=1}^{N} \delta_{\overline{X}_{1:n}^i}(x_{1:n}) \tag{34}$$

after the resampling step. The approximation (33) is to be preferred to (34). We also obtain an approximation of Z_n/Z_{n-1} through

$$\widehat{\frac{Z_n}{Z_{n-1}}} = \frac{1}{N}\sum_{i=1}^{N} \alpha_n\left(X_{1:n}^i\right).$$

As we have already mentioned, resampling has the effect of removing particles with low weights and multiplying particles with high weights. However, this is at the cost of immediately introducing some additional variance. If particles have unnormalised weights with a small variance then the resampling step might be unnecessary. Consequently, in practice, it is more sensible to resample only when the variance of the unnormalised weights is superior to a pre-specified threshold. This is often assessed by looking at the variability of the weights using the so-called Effective Sample Size (ESS) criterion [30, pp. 35-36], which is given (at time n) by

$$ESS = \left(\sum_{i=1}^{N}\left(W_n^i\right)^2\right)^{-1}.$$

Its interpretation is that in a simple IS setting, inference based on the N weighted samples is approximately equivalent (in terms of estimator variance) to inference based on ESS perfect samples from the target distribution. The ESS takes values between 1 and N and we resample only when it is below a threshold N_T; typically $N_T = N/2$. Alternative criteria can be used such as the entropy of the weights $\left\{W_n^i\right\}$ which achieves its maximum value when $W_n^i = \frac{1}{N}$. In this case, we resample when the entropy is below a given threshold.

Sequential Monte Carlo with Adaptive Resampling

At time $n = 1$

- Sample $X_1^i \sim q_1(x_1)$.
- Compute the weights $w_1\left(X_1^i\right)$ and $W_1^i \propto w_1\left(X_1^i\right)$.
- If resampling criterion satisfied then resample $\left\{W_1^i, X_1^i\right\}$ to obtain N equally weighted particles $\left\{\frac{1}{N}, \overline{X}_1^i\right\}$ and set $\left\{\overline{W}_1^i, \overline{X}_1^i\right\} \leftarrow \left\{\frac{1}{N}, \overline{X}_1^i\right\}$, otherwise set $\left\{\overline{W}_1^i, \overline{X}_1^i\right\} \leftarrow \left\{W_1^i, X_1^i\right\}$.

At time $n \geq 2$

- Sample $X_n^i \sim q_n(x_n|\,\overline{X}_{1:n-1}^i)$ and set $X_{1:n}^i \leftarrow \left(\overline{X}_{1:n-1}^i, X_n^i\right)$.
- Compute the weights $\alpha_n\left(X_{1:n}^i\right)$ and $W_n^i \propto \overline{W}_{n-1}^i \alpha_n\left(X_{1:n}^i\right)$.
- If resampling criterion satisfied, then resample $\left\{W_n^i, X_{1:n}^i\right\}$ to obtain N equally weighted particles $\left\{\frac{1}{N}, \overline{X}_{1:n}^i\right\}$ and set $\left\{\overline{W}_n^i, \overline{X}_n^i\right\} \leftarrow \left\{\frac{1}{N}, \overline{X}_n^i\right\}$, otherwise set $\left\{\overline{W}_n^i, \overline{X}_n^i\right\} \leftarrow \left\{W_n^i, X_n^i\right\}$.

In this context too we have two approximations of $\pi_n(x_{1:n})$

$$\widehat{\pi}_n(x_{1:n}) = \sum_{i=1}^{N} W_n^i \delta_{X_{1:n}^i}(x_{1:n}), \tag{35}$$

$$\overline{\pi}_n(x_{1:n}) = \sum_{i=1}^{N} \overline{W}_n^i \delta_{\overline{X}_{1:n}^i}(x_{1:n})$$

which are equal if no resampling step is used at time n. We may also estimate Z_n/Z_{n-1} through

$$\widehat{\frac{Z_n}{Z_{n-1}}} = \sum_{i=1}^{N} \overline{W}_{n-1}^i \alpha_n\left(X_{1:n}^i\right). \tag{36}$$

SMC methods involve systems of particles which interact (via the resampling mechanism) and, consequently, obtaining convergence results is a much more difficult task than it is for SIS where standard results (iid asymptotics) apply. However, there are numerous sharp convergence results available for SMC; see [10] for an introduction to the subject and the monograph of Del Moral [11] for a complete treatment of the subject. An explicit treatment of the case in which resampling is performed adaptively is provided by [12].

The presence or absence of degeneracy is the factor which most often determines whether an SMC algorithm works in practice. However strong the convergence results available for limitingly large samples may be, we cannot expect good performance if the *finite* sample which is actually used is degenerate. Indeed, some degree of degeneracy is inevitable in all but trivial cases: if SMC algorithms are used for sufficiently many time steps every resampling step reduces the number of unique values representing X_1, for example. For this reason, any SMC algorithm which relies upon the distribution of full paths $x_{1:n}$ will fail for large enough n for any finite sample size, N, in spite of the asymptotic justification. It is intuitive that one should endeavour to employ algorithms which do not depend upon the full path of the samples, but only upon the distribution of some finite component $x_{n-L:n}$ for some fixed L which is independent of n. Furthermore, ergodicity (a tendency for the future to be essentially independent of the distant past) of the underlying system will prevent the accumulation of errors over time. These concepts are precisely characterised by existing convergence results, some of the most important of which are summarised and interpreted in section 3.6.

Although sample degeneracy emerges as a consequence of resampling, it is really a manifestation of a deeper problem — one which resampling actually mitigates. It is inherently impossible to accurately represent a distribution on a space of arbitrarily high dimension with a sample of fixed, finite size. Sample impoverishment is a term which is often used to describe the situation in which very few different particles have significant weight. This problem has much the same effect as sample degeneracy and occurs, in the absence of resampling, as the inevitable consequence of multiplying together incremental importance weights from a large number of time steps. It is, of course, not possible to circumvent either problem by increasing the number of samples at every iteration to maintain a constant effective sample size as this would lead to an exponential growth in the number of samples required. This sheds some light on the resampling mechanism: it "resets the system" in such a way that its representation of final time marginals remains well behaved at the expense of further diminishing the quality of the path-samples. By focusing on the fixed-dimensional final time marginals in this way, it allows us to circumvent the problem of increasing dimensionality.

3.6 Convergence Results for Sequential Monte Carlo Methods

Here, we briefly discuss selected convergence results for SMC. We focus on the CLT as it allows us to clearly understand the benefits of the resampling step and why it "works". If multinomial resampling is used at every iteration[4], then the associated SMC estimates of $\widehat{Z}_n/Z_n$ and $I_n(\varphi_n)$ satisfy a CLT and their respective

[4]Similar expressions can be established when a lower variance resampling strategy such as residual resampling is used and when resampling is performed adaptively [12]. The results presented here are sufficient to guide the design of particular

asymptotic variances are given by

$$\frac{1}{N}\left(\int \frac{\pi_n^2(x_1)}{q_1(x_1)}dx_1 - 1 + \sum_{k=2}^{n}\int \frac{\pi_n^2(x_{1:k})}{\pi_{k-1}(x_{1:k-1})\,q_k(x_k|\,x_{1:k-1})}dx_{k-1:k} - 1\right) \tag{37}$$

and

$$\begin{array}{l}\int \frac{\pi_n^2(x_1)}{q_1(x_1)}\left(\int \varphi_n(x_{1:n})\,\pi_n(x_{2:n}|\,x_1)\,dx_{2:n} - I_n(\varphi_n)\right)^2 dx_1 \\ +\sum_{k=2}^{n-1}\int \frac{\pi_n^2(x_{1:k})}{\pi_{k-1}(x_{1:k-1})q_k(x_k|x_{1:k-1})}\left(\int \varphi_n(x_{1:n})\,\pi_n(x_{k+1:n}|\,x_{1:k})\,dx_{k+1:n} - I_n(\varphi_n)\right)^2 dx_{1:k} \\ +\int \frac{\pi_n^2(x_{1:n})}{\pi_{n-1}(x_{1:n-1})q_n(x_n|x_{1:n-1})}\left(\varphi_n(x_{1:n}) - I_n(\varphi_n)\right)^2 dx_{1:n}.\end{array} \tag{38}$$

A short and elegant proof of this result is given in [11, Chapter 9]; see also [9]. These expression are very informative. They show that the resampling step has the effect of "resetting" the particle system whenever it is applied. Comparing (26) to (37), we see that the SMC variance expression has replaced the importance distribution $q_n(x_{1:n})$ in the SIS variance with the importance distributions $\pi_{k-1}(x_{1:k-1})\,q_k(x_k|\,x_{1:k-1})$ obtained after the resampling step at time $k-1$. Moreover, we will show that in important scenarios the variances of SMC estimates are orders of magnitude smaller than the variances of SIS estimates.

Let us first revisit the toy example discussed in section 3.3.

Example (continued). In this case, it follows from (37) that the asymptotic variance is finite only when $\sigma^2 > \frac{1}{2}$ and

$$\begin{aligned}\frac{\mathbb{V}_{\mathrm{SMC}}\left[\widehat{Z}_n\right]}{Z_n^2} &\approx \frac{1}{N}\left[\int \frac{\pi_n^2(x_1)}{q_1(x_1)}dx_1 - 1 + \sum_{k=2}^{n}\int \frac{\pi_n^2(x_k)}{q_k(x_k)}dx_k - 1\right] \\ &= \frac{n}{N}\left[\left(\frac{\sigma^4}{2\sigma^2-1}\right)^{1/2} - 1\right]\end{aligned}$$

compared to

$$\frac{\mathbb{V}_{\mathrm{IS}}\left[\widehat{Z}_n\right]}{Z_n^2} = \frac{1}{N}\left[\left(\frac{\sigma^4}{2\sigma^2-1}\right)^{n/2} - 1\right].$$

The asymptotic variance of the SMC estimate increases linearly with n in contrast to the exponential growth of the IS variance. For example, if we select $\sigma^2 = 1.2$ then we have a reasonably good importance distribution as $q_k(x_k) \approx \pi_n(x_k)$. In this case, we saw that it is necessary to employ $N \approx 2 \times 10^{23}$ particles in order to obtain $\frac{\mathbb{V}_{\mathrm{IS}}\left[\widehat{Z}_n\right]}{Z_n^2} = 10^{-2}$ for $n = 1000$. Whereas to obtain the same performance, $\frac{\mathbb{V}_{\mathrm{SMC}}\left[\widehat{Z}_n\right]}{Z_n^2} = 10^{-2}$, SMC requires the use of just $N \approx 10^4$ particles: an improvement by 19 orders of magnitude.

This scenario is overly favourable to SMC as the target (31) factorises. However, generally speaking, the major advantage of SMC over IS is that it allows us to exploit the forgetting properties of the model under study as illustrated by the following example.

Example. Consider the following more realistic scenario where

$$\gamma_n(x_{1:n}) = \mu(x_1)\prod_{k=2}^{n} M_k(x_k|\,x_{k-1})\prod_{k=1}^{n} G_k(x_k)$$

with μ a probability distribution, M_k a Markov transition kernel and G_k a positive "potential" function. Essentially, filtering corresponds to this model with $M_k(x_k|\,x_{k-1}) = f(x_k|\,x_{k-1})$ and the time inhomogeneous potential function $G_k(x_k) = g(y_k|\,x_k)$. In this case, $\pi_k(x_k|\,x_{1:k-1}) = \pi_k(x_k|\,x_{k-1})$ and we would

algorithms and the additional complexity involved in considering more general scenarios serves largely to produce substantially more complex expressions which obscure the important points.

typically select an importance distribution $q_k(x_k|x_{1:k-1})$ with the same Markov property $q_k(x_k|x_{1:k-1}) = q_k(x_k|x_{k-1})$. It follows that (37) is equal to

$$\frac{1}{N}\left(\int \frac{\pi_1^2(x_1)}{q_1(x_1)}dx_1 - 1 + \sum_{k=2}^{n}\int \frac{\pi_n^2(x_{k-1:k})}{\pi_{k-1}(x_{k-1})\,q_k(x_k|x_{k-1})}dx_{k-1:k} - 1\right)$$

and (38), for $\varphi_n(x_{1:n}) = \varphi(x_n)$, equals:

$$\begin{array}{l}\int \frac{\pi_1^2(x_1)}{q_1(x_1)}\left(\int \varphi(x_n)\,\pi_n(x_n|x_1)\,dx_{2:n} - I_n(\varphi)\right)^2 dx_1 \\ + \sum_{k=2}^{n-1}\int \frac{\pi_n^2(x_{k-1:k})}{\pi_{k-1}(x_{k-1})q_k(x_k|x_{k-1})}\left(\int \varphi(x_n)\,\pi_n(x_n|x_k)\,dx_k - I_n(\varphi)\right)^2 dx_{k-1:k} \\ + \int \frac{\pi_n^2(x_{n-1:n})}{\pi_{n-1}(x_{n-1})q_n(x_n|x_{n-1})}\left(\varphi(x_n) - I_n(\varphi)\right)^2 dx_{n-1:n},\end{array}$$

where we use the notation $I_n(\varphi)$ for $I_n(\varphi_n)$. In many realistic scenarios, the model associated with $\pi_n(x_{1:n})$ has some sort of ergodic properties; i.e. $\forall x_k, x_k' \in \mathcal{X}\ \pi_n(x_n|x_k) \approx \pi_n(x_n|x_k')$ for large enough $n-k$. In layman's terms, at time n what happened at time k is irrelevant if $n-k$ is large enough. Moreover, this often happens exponentially fast; that is for any (x_k, x_k')

$$\frac{1}{2}\int |\pi_n(x_n|x_k) - \pi_n(x_n|x_k')|\,dx_n \leq \beta^{n-k}$$

for some $\beta < 1$. This property can be used to establish that for bounded functions $\varphi \leq \|\varphi\|$

$$\left|\int \varphi(x_n)\,\pi_n(x_n|x_k)\,dx_n - I(\varphi)\right| \leq \beta^{n-k}\|\varphi\|$$

and under weak additional assumptions we have

$$\frac{\pi_n^2(x_{k-1:k})}{\pi_{k-1}(x_{k-1})\,q_k(x_k|x_{k-1})} \leq A$$

for a finite constant A. Hence it follows that

$$\frac{\mathbb{V}_{\mathrm{SMC}}\left[\widehat{Z}_n\right]}{Z_n^2} \leq \frac{C\cdot n}{N},$$

$$\mathbb{V}_{\mathrm{SMC}}\left[\widehat{I}_n(\varphi)\right] \leq \frac{D}{N}.$$

for some finite constants C, D that are independent of n. These constants typically increase polynomially/exponentially with the dimension of the state-space $\mathcal{X}$ and decrease as $\beta \to 0$.

3.7 Summary

We have presented a generic SMC algorithm which approximates $\{\pi_n(x_{1:n})\}$ and $\{Z_n\}$ sequentially in time.

- Wherever it is possible to sample from $q_n(x_n|x_{1:n-1})$ and evaluate $\alpha_n(x_{1:n})$ in a time independent of n, this leads to an algorithm whose computational complexity does not increase with n.
- For any k, there exists $n > k$ such that the SMC approximation of $\pi_n(x_{1:k})$ consists of a single particle because of the successive resampling steps. It is thus impossible to get a "good" SMC approximation of the joint distributions $\{\pi_n(x_{1:n})\}$ when n is too large. This can easily be seen in practice, by monitoring the number of distinct particles approximating $\pi_n(x_1)$.

- However, under mixing conditions, this SMC algorithm is able to provide estimates of marginal distributions of the form $\pi_n(x_{n-L+1:n})$ and estimates of Z_n/Z_{n-1} whose asymptotic variance is uniformly bounded with n. This property is crucial and explains why SMC methods "work" in many realistic scenarios.

- Practically, one should keep in mind that the variance of SMC estimates can only expected to be reasonable if the variance of the incremental weights is small. In particular, this requires that we can only expect to obtain good performance if $\pi_n(x_{1:n-1}) \approx \pi_{n-1}(x_{1:n-1})$ and $q_n(x_n|x_{1:n-1}) \approx \pi_n(x_n|x_{1:n-1})$; that is if the successive distributions we want to approximate do not differ much one from each other and the importance distribution is a reasonable approximation of the "optimal" importance distribution. However, if successive distributions differ significantly, it is often possible to design an artificial sequence of distributions to "bridge" this transition [21, 31].

4 Particle Filtering

Remember that in the filtering context, we want to be able to compute a numerical approximation of the distribution $\{p(x_{1:n}|y_{1:n})\}_{n\geq 1}$ *sequentially* in time. A direct application of the SMC methods described earlier to the sequence of target distributions $\pi_n(x_{1:n}) = p(x_{1:n}|y_{1:n})$ yields a popular class of particle filters. More elaborate sequences of target and proposal distributions yield various more advanced algorithms. For ease of presentation, we present algorithms in which we resample at each time step. However, in practice we recommend only resampling when the ESS is below a threshold and employing the systematic resampling scheme.

4.1 SMC for Filtering

First, consider the simplest case in which $\gamma_n(x_{1:n}) = p(x_{1:n}, y_{1:n})$ is chosen, yielding $\pi_n(x_{1:n}) = p(x_{1:n}|y_{1:n})$ and $Z_n = p(y_{1:n})$. Practically, it is only necessary to select the importance distribution $q_n(x_n|x_{1:n-1})$. We have seen that in order to minimise the variance of the importance weights at time n, we should select $q_n^{\text{opt}}(x_n|x_{1:n-1}) = \pi_n(x_n|x_{1:n-1})$ where

$$\begin{aligned}\pi_n(x_n|x_{1:n-1}) &= p(x_n|y_n, x_{n-1}) \\ &= \frac{g(y_n|x_n) f(x_n|x_{n-1})}{p(y_n|x_{n-1})},\end{aligned} \tag{39}$$

and the associated incremental importance weight is $\alpha_n(x_{1:n}) = p(y_n|x_{n-1})$. In many scenarios, it is not possible to sample from this distribution but we should aim to approximate it. In any case, it shows that we should use an importance distribution of the form

$$q_n(x_n|x_{1:n-1}) = q(x_n|y_n, x_{n-1}) \tag{40}$$

and that there is nothing to be gained from building importance distributions depending also upon $(y_{1:n-1}, x_{1:n-2})$ — although, at least in principle, in some settings there may be advantages to using information from subsequent observations if they are available. Combining (29), (30) and (40), the incremental weight is given by

$$\alpha_n(x_{1:n}) = \alpha_n(x_{n-1:n}) = \frac{g(y_n|x_n) f(x_n|x_{n-1})}{q(x_n|y_n, x_{n-1})}.$$

The algorithm can thus be summarised as follows.

SIR/SMC for Filtering

At time $n = 1$

- Sample $X_1^i \sim q(x_1|y_1)$.
- Compute the weights $w_1(X_1^i) = \frac{\mu(X_1^i) g(y_1|X_1^i)}{q(X_1^i|y_1)}$ and $W_1^i \propto w_1(X_1^i)$.
- Resample $\{W_1^i, X_1^i\}$ to obtain N equally-weighted particles $\left\{\frac{1}{N}, \overline{X}_1^i\right\}$.

At time $n \geq 2$

- Sample $X_n^i \sim q(x_n|y_n, \overline{X}_{n-1}^i)$ and set $X_{1:n}^i \leftarrow \left(\overline{X}_{1:n-1}^i, X_n^i\right)$.
- Compute the weights $\alpha_n(X_{n-1:n}^i) = \frac{g(y_n|X_n^i) f(X_n^i|X_{n-1}^i)}{q(X_n^i|y_n, X_{n-1}^i)}$ and $W_n^i \propto \alpha_n(X_{n-1:n}^i)$.
- Resample $\{W_n^i, X_{1:n}^i\}$ to obtain N new equally-weighted particles $\left\{\frac{1}{N}, \overline{X}_{1:n}^i\right\}$.

We obtain at time n

$$\widehat{p}(x_{1:n}|y_{1:n}) = \sum_{i=1}^{N} W_n^i \delta_{X_{1:n}^i}(x_{1:n}),$$

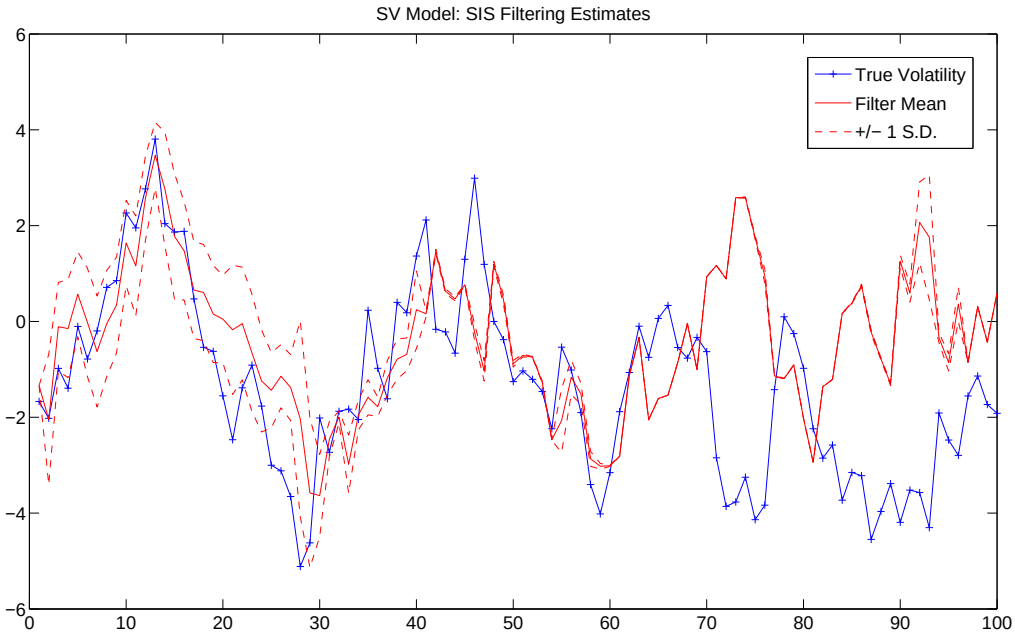


Figure 2: Filtering estimates obtained for the stochastic volatility model using SIS. At each time the mean and standard deviation of x_n conditional upon $y_{1:n}$ is estimated using the particle set. It initially performs reasonably, but the approximation eventually collapses: the estimated mean begins to diverge from the truth and the estimate of the standard deviation is inaccurately low.

$$\widehat{p}\left(y_n \middle| y_{1:n-1}\right) = \sum_{i=1}^{N} W_{n-1}^i \alpha_n \left(X_{n-1:n}^i\right).$$

However, if we are interested only in approximating the marginal distributions $\{p\left(x_n \middle| y_{1:n}\right)\}$ and $\{p\left(y_{1:n}\right)\}$ then we need to store only the terminal-value particles $\left\{X_{n-1:n}^i\right\}$ to be able to compute the weights: the algorithm's storage requirements do not increase over time.

Many techniques have been proposed to design "efficient" importance distributions $q\left(x_n \middle| y_n, x_{n-1}\right)$ which approximate $p\left(x_n \middle| y_n, x_{n-1}\right)$. In particular the use of standard suboptimal filtering techniques such as the Extended Kalman Filter or the Unscented Kalman Filter to obtain importance distributions is very popular in the literature [14, 37]. The use of local optimisation techniques to design $q\left(x_n \middle| y_n, x_{n-1}\right)$ centred around the mode of $p\left(x_n \middle| y_n, x_{n-1}\right)$ has also been advocated [33, 34].

4.1.1 Example: Stochastic Volatility

Returning to example 4 and the simulated data shown in figure 1, we are able to illustrate the performance of SMC algorithms with and without resampling steps in a filtering context.

An SIS algorithm (corresponding to the above SMC algorithm without a resampling step) with $N = 1000$ particles, in which the conditional prior is employed as a proposal distribution (leading to an algorithm in which the particle weights are proportional to the likelihood function) produces the output shown in figure 2. Specifically, at each iteration, n, of the algorithm the conditional expectation and standard deviation of x_n, given $y_{1:n}$ is obtained. It can be seen that the performance is initially good, but after a few iterations the estimate of the mean becomes inaccurate, and the estimated standard deviation shrinks to a very small

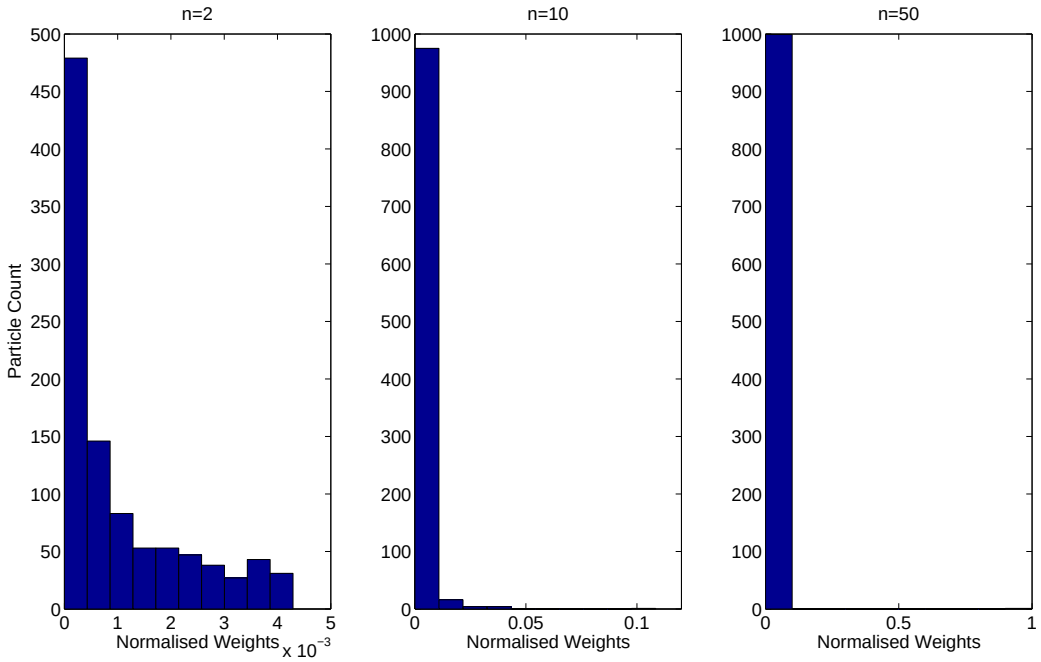


Figure 3: Empirical distributions of the particle weights obtained with the SIS algorithm for the stochastic volatility model at iterations 2, 10 and 50. Although the algorithm is reasonably initialised, by iteration 10 only a few tens of particles have significant weight and by iteration 50 a single particle is dominant.

value. This standard deviation is an estimate of the standard deviation of the conditional posterior obtained via the particle filter: it is not a measure of the standard deviation of the estimator. Such an estimate can be obtained by considering several independent particle filters run on the same data and would illustrate the high variability of estimates obtained by a poorly-designed algorithm such as this one. In practice, approximations such as the effective sample size are often used as surrogates to characterise the uncertainty of the filter estimates but these perform well only if the filter is providing a reasonable approximation of the conditional distributions of interest. Figure 3 supports the theory that the failure of the algorithm after a few iterations is due to weight degeneracy, showing that the number of particles with significant weight falls rapidly.

The SIR algorithm described above was also applied to this problem with the same proposal distribution and number of particles as were employed in the SIS case. For simplicity, multinomial resampling was applied at every iteration. Qualitatively, the same features would be observed if a more sophisticated algorithm were employed, or adaptive resampling were used although these approaches would lessen the severity of the path-degeneracy problem. Figure 4 shows the distribution of particle weights for this algorithm. Notice that unlike the SIS algorithm shown previously, there are many particles with significant weight at all three time points. It is important to note that while this is encouraging it is not evidence that the algorithm is performing well: it provides no information about the path-space distribution and, in fact, it is easy to construct poorly-performing algorithms which appear to have a good distribution of particle weights (for instance, consider a scenario in which the target is relatively flat in its tails but sharply concentrated about a mode; if the proposal has very little mass in the vicinity of the mode then it is likely that a collection of very similar importance weights will be obtained — but the sample thus obtained does not characterise the distribution of interest well). Figure 5 shows that the algorithm does indeed produce a reasonable estimate and plausible credible interval. And, as we expect, a problem does arise when we consider the smoothing distributions $p(x_n|y_{1:500})$ as shown in figure 6: the estimate and credible interval is unreliable for $n \ll 500$. This is due to the degeneracy caused at the beginning of the path by repeated resampling. In contrast the

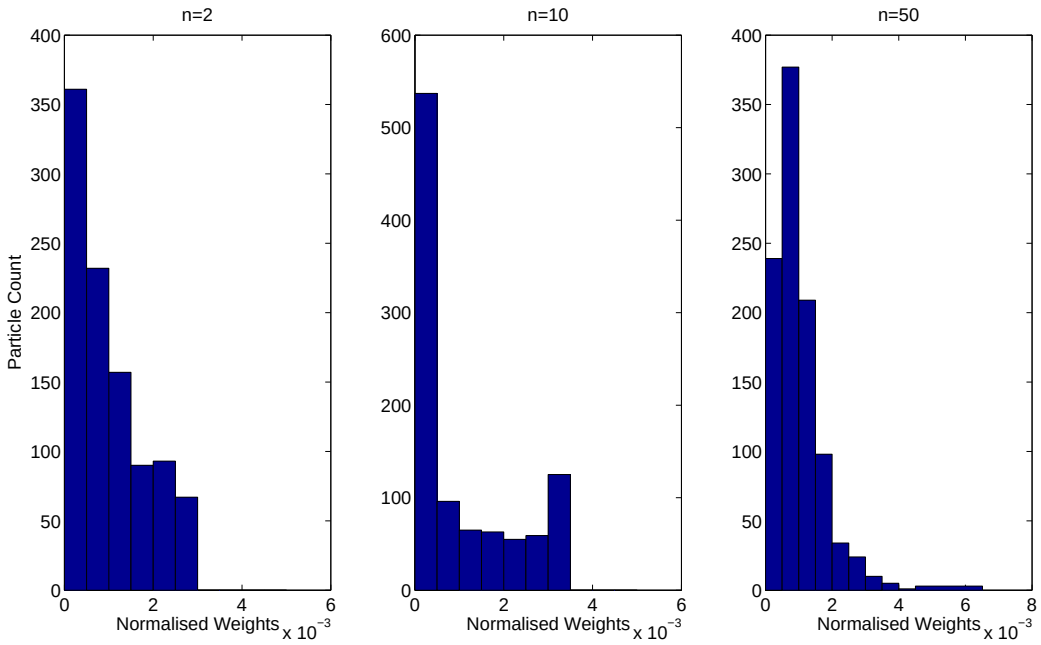


Figure 4: Empirical distribution of particle weights for an SIR algorithm applied to the stochastic volatility model. Notice that there is no evidence of weight degeneracy in contrast to the SIS case. This comes at the cost of reducing the quality of the path-space representation.

smoothed estimate for $n \approx 500$ (not shown) is reasonable.

4.2 Auxiliary Particle Filtering

As was discussed above, the optimal proposal distribution (in the sense of minimising the variance of importance weights) when performing standard particle filtering is $q(x_n|y_n, x_{n-1}) = p(x_n|y_n, x_{n-1})$. Indeed, $\alpha_n(x_{n-1:n})$ is independent of x_n in this case so it is possible to interchange the order of the sampling and resampling steps. Intuitively, this yields a better approximation of the distribution as it provides a greater number of distinct particles to approximate the target. This is an example of a general principle: resampling, if it is to be applied in a particular iteration, should be performed before, rather than after, any operation that doesn't influence the importance weights in order to minimise the loss of information.

It is clear that if importance weights are independent of the new state and the proposal distribution corresponds to the marginal distribution of the proposed states then weighting, resampling and then sampling corresponds to a reweighting to correct for the discrepancy between the old and new marginal distribution of the earlier states, resampling to produce an unweighted sample and then generation of the new state from its conditional distribution. This intuition can easily be formalised.

However, in general, the incremental importance weights do depend upon the new states and this straightforward change of order becomes impossible. In a sense, this interchange of sampling and resampling produces an algorithm in which information from the next observation is used to determine which particles should survive resampling at a given time (to see this, consider weighting and resampling occurring as the very last step of the iteration before the current one, rather than as the first step of that iteration). It is desirable to find methods for making use of this future information in a more general setting, so that we can obtain the same advantage in situations in which it is not possible to make use of the optimal proposal distribution.

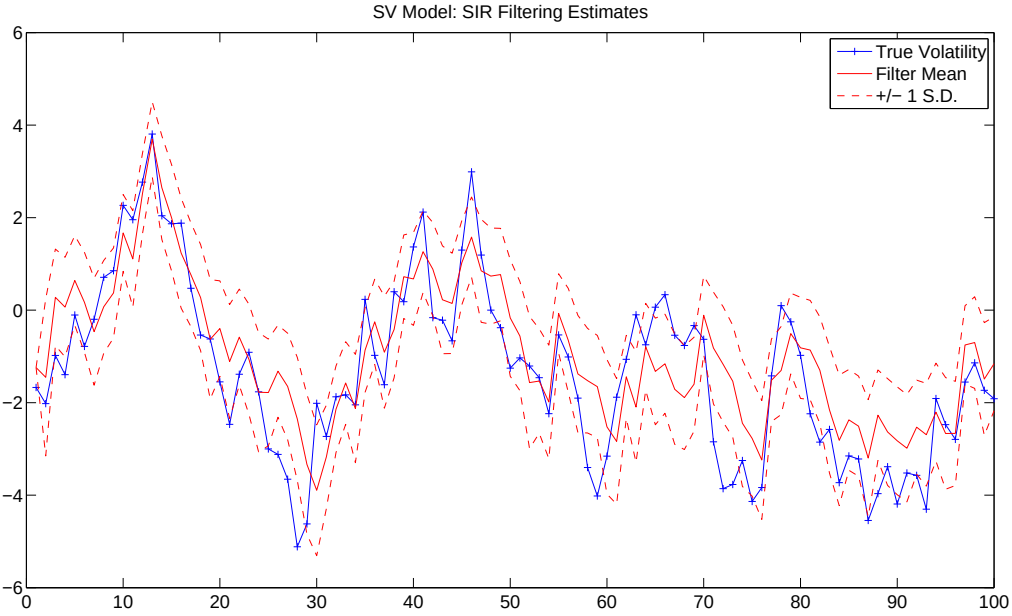


Figure 5: SIR Filtering estimates for the SV model.

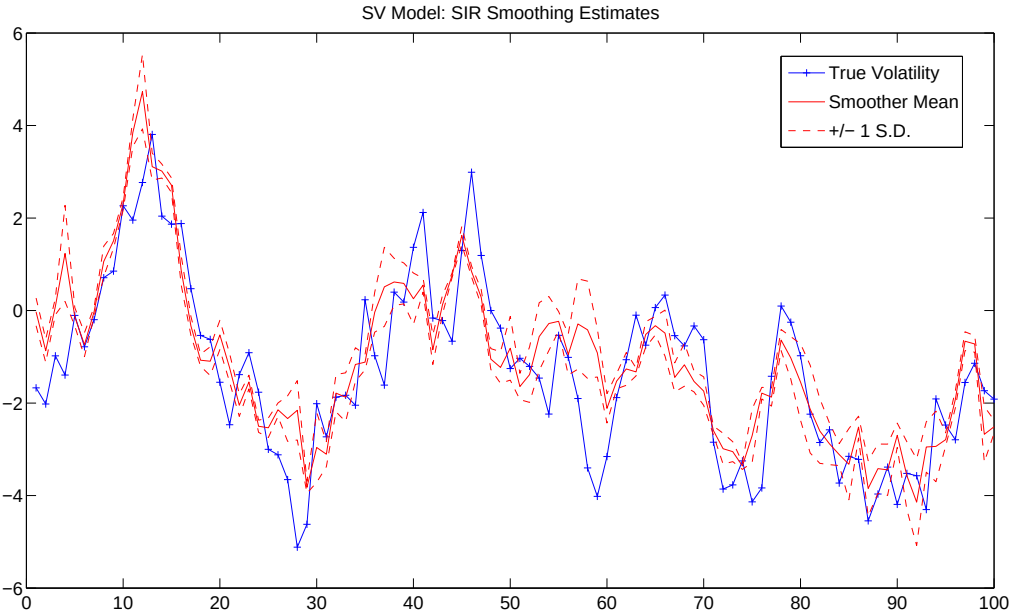


Figure 6: SIR Smoothing estimates for the SV model.

The Auxiliary Particle Filter (APF) is an alternative algorithm which does essentially this. It was originally introduced in [33] using auxiliary variables — hence its name. Several improvements were proposed to reduce its variance [7, 34]. We present here the version of the APF presented in [7] which only includes one resampling step at each time instance. It has long been realised that, experimentally, this version outperforms the original two stage resampling algorithm proposed in [33] and is widely used; see [7] for a comparison of both approaches. The APF is a look ahead method where at time n we try to predict which samples will be in regions of high probability masses at time $n+1$.

It was shown in [23] that the APF can be reinterpreted as a *standard* SMC algorithm applied to the following sequence of target distributions

$$\gamma_n(x_{1:n}) = p(x_{1:n}, y_{1:n})\,\widetilde{p}(y_{n+1}|x_n) \tag{41}$$

with $\widetilde{p}(y_{n+1}|x_n)$ chosen as an approximation of the predictive likelihood $p(y_{n+1}|x_n)$ if it is not known analytically. It follows that $\pi_n(x_{1:n})$ is an approximation of $p(x_{1:n}|y_{1:n+1})$ denoted $\widetilde{p}(x_{1:n}|y_{1:n+1})$ given by

$$\pi_n(x_{1:n}) = \widetilde{p}(x_{1:n}|y_{1:n+1}) \propto p(x_{1:n}|y_{1:n})\,\widetilde{p}(y_{n+1}|x_n) \tag{42}$$

In the APF we also use an importance distribution $q_n(x_n|x_{1:n-1})$ of the form (40) which is typically an approximation of (39). Note that (39) is different from $\pi_n(x_n|x_{1:n-1})$ in this scenario. Even if we could sample from $\pi_n(x_n|x_{1:n-1})$, one should remember that in this case the object of inference is not $\pi_n(x_{1:n}) = \widetilde{p}(x_{1:n}|y_{1:n+1})$ but $p(x_{1:n}|y_{1:n})$. The associated incremental weight is given by

$$\begin{aligned}\alpha_n(x_{n-1:n}) &= \frac{\gamma_n(x_{1:n})}{\gamma_{n-1}(x_{1:n-1})\,q_n(x_n|x_{1:n-1})} \\ &= \frac{g(y_n|x_n)\,f(x_n|x_{n-1})\,\widetilde{p}(y_{n+1}|x_n)}{\widetilde{p}(y_n|x_{n-1})\,q(x_n|y_n, x_{n-1})}.\end{aligned}$$

To summarize, the APF proceeds as follows.

Auxiliary Particle Filtering

At time $n = 1$

- Sample $X_1^i \sim q(x_1|y_1)$.
- Compute the weights $w_1\left(X_1^i\right) = \frac{\mu\left(X_1^i\right) g\left(y_1|X_1^i\right)\widetilde{p}\left(y_2|X_1^i\right)}{q\left(X_1^i|y_1\right)}$ and $W_1^i \propto w_1\left(X_1^i\right)$.
- Resample $\left\{W_1^i, X_1^i\right\}$ to obtain N equally-weighted particles $\left\{\frac{1}{N}, \overline{X}_1^i\right\}$.

At time $n \geq 2$

- Sample $X_n^i \sim q(x_n|y_n, \overline{X}_{n-1}^i)$ and set $X_{1:n}^i \leftarrow \left(\overline{X}_{1:n-1}^i, X_n^i\right)$.
- Compute the weights $\alpha_n\left(X_{n-1:n}^i\right) = \frac{g\left(y_n|X_n^i\right) f\left(X_n^i|X_{n-1}^i\right)\widetilde{p}\left(y_{n+1}|X_n^i\right)}{\widetilde{p}\left(y_n|X_{n-1}^i\right) q\left(X_n^i|y_n, X_{n-1}^i\right)}$ and $W_n^i \propto \alpha_n\left(X_{n-1:n}^i\right)$.
- Resample $\left\{W_n^i, X_{1:n}^i\right\}$ to obtain N new equally-weighted particles $\left\{\frac{1}{N}, \overline{X}_{1:n}^i\right\}$.

Keeping in mind that this algorithm does not approximate the distributions $\{p(x_{1:n}|y_{1:n})\}$ but the distributions $\{\widetilde{p}(x_{1:n}|y_{1:n+1})\}$, we use IS to obtain an approximation of $p(x_{1:n}|y_{1:n})$ with

$$\pi_{n-1}(x_{1:n-1})\,q_n(x_n|x_{1:n-1}) = \widetilde{p}(x_{1:n-1}|y_{1:n})\,q(x_n|y_n,x_{n-1})$$

as the importance distribution. A Monte Carlo approximation of this importance distribution is obtained after the sampling step in the APF and the associated unnormalised importance weights are given by

$$\widetilde{w}_n(x_{n-1:n}) = \frac{p(x_{1:n},y_{1:n})}{\gamma_{n-1}(x_{1:n-1})\,q_n(x_n|x_{1:n-1})} = \frac{g(y_n|x_n)\,f(x_n|x_{n-1})}{\widetilde{p}(y_n|x_{n-1})\,q(x_n|y_n,x_{n-1})}. \tag{43}$$

It follows that we obtain

$$\begin{aligned}
\widehat{p}(x_{1:n}|y_{1:n}) &= \sum_{i=1}^{N} \widetilde{W}_n^i \delta_{X_{1:n}^i}(x_{1:n}), \\
\widehat{p}(y_n|y_{1:n-1}) &= \left(\frac{1}{N}\sum_{i=1}^{N} \widetilde{w}_n\left(X_{n-1:n}^i\right)\right)\left(\sum_{i=1}^{N} \widetilde{W}_{n-1}^i \widetilde{p}\left(y_n|X_{n-1}^i\right)\right)
\end{aligned}$$

where $\widetilde{W}_n^i \propto \widetilde{w}_n\left(X_{n-1:n}^i\right)$ or $\widetilde{W}_n^i \propto W_{n-1}^i \widetilde{w}_n\left(X_{n-1:n}^i\right)$ if resampling was not performed at the end of the previous iteration. Selecting $q_n(x_n|x_{1:n-1}) = p(x_n|y_n,x_{n-1})$ and $\widetilde{p}(y_n|x_{n-1}) = p(y_n|x_{n-1})$, when it is possible to do so, leads to the so-called "perfect adaptation" case [33]. In this case, the APF takes a particularly simple form as $\alpha_n(x_{n-1:n}) = p(y_n|x_{n-1})$ and $\widetilde{w}_n(x_{n-1:n}) = 1$. This is similar to the algorithm discussed in the previous subsection where the order of the sampling and resampling steps is interchanged.

This simple reinterpretation of the APF shows us several things:

- We should select a distribution $\widetilde{p}(x_{1:n}|y_{1:n})$ with thicker tails than $p(x_{1:n}|y_{1:n})$.
- Setting $\widetilde{p}(y_n|x_{n-1}) = g(y_n|\mu(x_{n-1}))$ where μ denotes some point estimate is perhaps unwise as this will not generally satisfy that requirement.
- We should use an approximation of the predictive likelihood which is compatible with the model we are using in the sense that it encodes at least the same degree of uncertainty as the exact model.

These observations follows from the fact that $\widetilde{p}(x_{1:n}|y_{1:n})$ is used as an importance distribution to estimate $p(x_{1:n}|y_{1:n})$ and this is the usual method to ensure that the estimator variance remains finite. Thus $\widetilde{p}(y_n|x_{n-1})$ should be more diffuse than $p(y_n|x_{n-1})$.

It has been suggested in the literature to set $\widetilde{p}(y_n|x_{n-1}) = g(y_n|\mu(x_{n-1}))$ where $\mu(x_{n-1})$ corresponds to the mode, mean or median of $f(x_n|x_{n-1})$. However, this simple approximation will often yield an importance weight function (43) which is not upper bounded on $\mathcal{X}\times\mathcal{X}$ and could lead to estimates with a large/infinite variance.

An alternative approach, selecting an approximation $\widetilde{p}(y_n,x_n|x_{n-1}) = \widetilde{p}(y_n|x_{n-1})\,q(x_n|y_n,x_{n-1})$ of the distribution $p(y_n,x_n|x_{n-1}) = p(y_n|x_{n-1})\,p(x_n|y_n,x_{n-1}) = g(y_n|x_n)\,f(x_n|x_{n-1})$ such that the ratio (43) is upper bounded on $\mathcal{X}\times\mathcal{X}$ and such that it is possible to compute $\widetilde{p}(y_n|x_{n-1})$ pointwise and to sample from $q(x_n|y_n,x_{n-1})$, should be preferred.

4.3 Limitations of Particle Filters

The algorithms described earlier suffer from several limitations. It is important to emphasise at this point that, even if the optimal importance distribution $p(x_n|y_n,x_{n-1})$ can be used, this does not guarantee that

the SMC algorithms will be efficient. Indeed, if the variance of $p(y_n|x_{n-1})$ is high, then the variance of the resulting approximation will be high. Consequently, it will be necessary to resample very frequently and the particle approximation $\widehat{p}(x_{1:n}|y_{1:n})$ of the joint distribution $p(x_{1:n}|y_{1:n})$ will be unreliable. In particular, for $k \ll n$ the marginal distribution $\widehat{p}(x_{1:k}|y_{1:n})$ will only be approximated by a few if not a single unique particle because the algorithm will have resampled many times between times k and n. One major problem with the approaches discussed above is that only the variables $\{X_n^i\}$ are sampled at time n but the path values $\{X_{1:n-1}^i\}$ remain fixed. An obvious way to improve upon these algorithms would involve not only sampling $\{X_n^i\}$ at time n, but also modifying the values of the paths over a fixed lag $\{X_{n-L+1:n-1}^i\}$ for $L > 1$ in light of the new observation y_n; L being fixed or upper bounded to ensure that we have a sequential algorithm (i.e. one whose computational cost and storage requirements are uniformly bounded over time). The following two sections describe two approaches to limit this degeneracy problem.

4.4 Resample-Move

This degeneracy problem has historically been addressed most often using the Resample-Move algorithm [20]. Like Markov Chain Monte Carlo (MCMC), it relies upon Markov kernels with appropriate invariant distributions. Whilst MCMC uses such kernels to generate collections of correlated samples, the Resample-Move algorithm uses them within an SMC algorithm as a principled way to "jitter" the particle locations and thus to reduce degeneracy. A Markov kernel $K_n(x'_{1:n}|x_{1:n})$ of invariant distribution $p(x_{1:n}|y_{1:n})$ is a Markov transition kernel with the property that

$$\int p(x_{1:n}|y_{1:n}) K_n(x'_{1:n}|x_{1:n}) dx_{1:n} = p(x'_{1:n}|y_{1:n}).$$

For such a kernel, if $X_{1:n} \sim p(x_{1:n}|y_{1:n})$ and $X'_{1:n}|X_{1:n} \sim K(x_{1:n}|X_{1:n})$ then $X'_{1:n}$ is still marginally distributed according to $p(x_{1:n}|y_{1:n})$. Even if $X_{1:n}$ is not distributed according to $p(x_{1:n}|y_{1:n})$ then, after an application of the MCMC kernel, $X'_{1:n}$ can only have a distribution closer than that of $X_{1:n}$ (in total variation norm) to $p(x_{1:n}|y_{1:n})$. A Markov kernel is said to be ergodic if iterative application of that kernel generates samples whose distribution converges towards $p(x_{1:n}|y_{1:n})$ irrespective of the distribution of the initial state.

It is easy to construct a Markov kernel with a specified invariant distribution. Indeed, this is the basis of MCMC — for details see [35] and references therein. For example, we could consider the following kernel, based upon the Gibbs sampler: set $x'_{1:n-L} = x_{1:n-L}$ then sample x'_{n-L+1} from $p\left(x_{n-L+1}|y_{1:n}, x'_{1:n-L}, x_{n-L+2:n}\right)$, sample x'_{n-L+2} from $p\left(x_{n-L+2}|y_{1:n}, x'_{1:n-L+1}, x_{n-L+3:n}\right)$ and so on until we sample x'_n from $p\left(x_n|y_{1:n}, x'_{1:n-1}\right)$; that is

$$K_n(x'_{1:n}|x_{1:n}) = \delta_{x_{1:n-L}}\left(x'_{1:n-L}\right) \prod_{k=n-L+1}^{n} p\left(x'_k|y_{1:n}, x'_{1:k-1}, x_{k+1:n}\right)$$

and we write, with a slight abuse of notation, the non-degenerate component of the MCMC kernel $K_n\left(x'_{n-L+1:n}|x_{1:n}\right)$. It is straightforward to verify that this kernel is $p(x_{1:n}|y_{1:n})$-invariant.

If it is not possible to sample from $p\left(x'_k|y_{1:n}, x'_{1:k-1}, x_{k+1:n}\right) = p\left(x'_k|y_k, x'_{k-1}, x_{k+1}\right)$, we can instead employ a Metropolis-Hastings (MH) strategy and sample a candidate according to some proposal $q\left(x'_k|y_k, x'_{k-1}, x_{k:k+1}\right)$ and accept it with the usual MH acceptance probability

$$\begin{aligned}
&\min\left(1, \frac{p(x'_{1:k}, x_{k+1:n}|y_{1:n})\, q\left(x_k|y_k, x'_{k-1}, x'_k, x_{k+1}\right)}{p\left(x'_{1:k-1}, x_{k+1:n}|y_{1:n}\right) q\left(x'_k|y_k, x'_{k-1}, x_{k:k+1}\right)}\right) \\
&= \min\left(1, \frac{g(y_k|x'_k) f(x_{k+1}|x'_k) f\left(x'_k|x'_{k-1}\right) q\left(x_k|y_k, x'_{k-1}, x'_k, x_{k+1}\right)}{g(y_k|x_k) f(x_{k+1}|x_k) f\left(x_k|x'_{k-1}\right) q\left(x'_k|y_k, x'_{k-1}, x_{k:k+1}\right)}\right).
\end{aligned}$$

It is clear that these kernels can be ergodic only if $L = n$ and *all* of the components of $x_{1:n}$ are updated. However, in our context we will typically not use ergodic kernels as this would require sampling an increasing

number of variables at each time step. In order to obtain truly online algorithms, we restrict ourselves to updating the variables $X_{n-L+1:n}$ for some fixed or bounded L.

The algorithm proceeds as follows, with K_n denoting a Markov kernel of invariant distribution $p(x_{1:n}|y_{1:n})$.

SMC Filtering with MCMC Moves

At time $n = 1$

- Sample $X_1^i \sim q(x_1|\, y_1)$.
- Compute the weights $w_1\left(X_1^i\right) = \frac{\mu\left(X_1^i\right) g\left(\, y_1|X_1^i\right)}{q\left(\, X_1^i|y_1\right)}$ and $W_1^i \propto w_1\left(X_1^i\right)$.
- Resample $\left\{W_1^i, X_1^i\right\}$ to obtain N equally-weighted particles $\left\{\frac{1}{N}, \overline{X}_1^i\right\}$.
- Sample $X_1'^i \sim K_1(x_1|\, \overline{X}_1^i)$.

At time $1 < n < L$

- Sample $X_n^i \sim q(x_n|\, y_n, X_{n-1}'^i)$ and set $X_{1:n}^i \leftarrow \left(X_{1:n-1}'^i, X_n^i\right)$.
- Compute the weights $\alpha_n\left(X_{n-1:n}^i\right) = \frac{g\left(\, y_n|X_n^i\right) f\left(\, X_n^i|X_{n-1}^i\right)}{q\left(\, X_n^i|y_n, X_{n-1}^i\right)}$ and $W_n^i \propto \alpha_n\left(X_{n-1:n}^i\right)$.
- Resample $\left\{W_n^i, X_{1:n}^i\right\}$ to obtain N equally-weighted particles $\left\{\frac{1}{N}, \overline{X}_{1:n}^i\right\}$.
- Sample $X_{1:n}'^i \sim K_n(x_{1:n}|\, \overline{X}_{1:n}^i)$.

At time $n \geq L$

- Sample $X_n^i \sim q(x_n|\, y_n, X_{n-1}'^i)$ and set $X_{1:n}^i \leftarrow \left(X_{1:n-1}'^i, X_n^i\right)$.
- Compute the weights $\alpha_n\left(X_{n-1:n}^i\right) = \frac{g\left(\, y_n|X_n^i\right) f\left(\, X_n^i|X_{n-1}^i\right)}{q\left(\, X_n^i|y_n, X_{n-1}^i\right)}$ and $W_n^i \propto \alpha_n\left(X_{n-1:n}^i\right)$.
- Resample $\left\{W_n^i, X_{1:n}^i\right\}$ to obtain N new equally-weighted particles $\left\{\frac{1}{N}, \overline{X}_{1:n}^i\right\}$.
- Sample $X_{n-L+1:n}'^i \sim K_n(x_{n-L+1:n}|\, \overline{X}_{1:n}^i)$ and set $X_{1:n}'^i \leftarrow \left(\overline{X}_{1:n-L}^i, X_{n-L+1:n}'^i\right)$.

The following premise, which [35] describes as "generalised importance sampling", could be used to justify inserting MCMC transitions into an SMC algorithm after the sampling step. Given a target distribution π, an instrumental distribution μ and a π-invariant Markov kernel, K, the following generalisation of the IS identity holds:

$$\int \pi(y)\varphi(y)dy \quad = \quad \iint \mu(x)K(\, y|\, x)\frac{\pi(y)L(x|\, y)}{\mu(x)K(\, y|\, x)}\varphi(y)dxdy$$

for any Markov kernel L. This approach corresponds to importance sampling on an enlarged space using $\mu(x)K(y|\, x)$ as the proposal distribution for a target $\pi(y)L(x|\, y)$ and then estimating a function $\varphi'(x, y) =$

$\varphi(y)$. In particular, for the time-reversal kernel associated with K

$$L(x|\,y) = \frac{\pi(x)K(y|\,x)}{\pi\,(y)},$$

we have the importance weight

$$\frac{\pi(y)L(x|\,y)}{\mu(x)K(y|\,x)} = \frac{\pi\,(x)}{\mu(x)}.$$

This interpretation of such an approach illustrates its deficiency: the importance weights depend only upon the location before the MCMC move while the sample depends upon the location after the move. Even if the kernel was perfectly mixing, leading to a collection of iid samples from the target distribution, some of these samples would be eliminated and some replicated in the resampling step. Resampling before an MCMC step will always lead to greater sample diversity than performing the steps in the other order (and this algorithm can be justified directly by the invariance property).

Based on this reinterpretation of MCMC moves within IS, it is possible to reformulate this algorithm as a specific application of the generic SMC algorithm discussed in Section 3. To simply notation we write $q_n\left(x_n|\,x_{n-1}\right)$ for $q\left(x_n|\,y_n, x_{n-1}\right)$. To clarify our argument, it is necessary to add a superscript to the variables; e.g. X_k^p corresponds to the p^{th} time the random variable X_k is sampled; in this and the following section, this superscript *does not* denote the particle index. Using such notation, this algorithm is the generic SMC algorithm associated to the following sequence of target distributions

$$\begin{aligned}
&\pi_n\left(x_1^{1:L+1}, \ldots, x_{n-L+1}^{1:L+1}, x_{n-L}^{1:L}, ..., x_n^{1:2}\right) \\
&= p\left(x_1^{L+1}, \ldots., x_{n-L+1}^{L+1}, x_{n-L}^{L}, ..., x_n^2 \middle|\, y_{1:n}\right) L_n\left(x_n^1, x_{n-1}^2, ..., x_{n-L+1}^L \middle|\, x_n^2, x_{n-1}^3, ..., x_{n-L+1}^{L+1}\right) \\
&\times \cdots \times L_2\left(x_1^2, x_2^1 \middle|\, x_1^3, x_2^2\right) L_1\left(x_1^1 \middle|\, x_1^2\right)
\end{aligned}$$

where L_n is the time-reversal kernel associated with K_n whereas, if no resampling is used[5], a path up to time n is sampled according to

$$\begin{aligned}
&q_n\left(x_1^{1:L+1}, \ldots, x_{n-L+1}^{1:L+1}, x_{n-L}^{1:L}, ..., x_n^{1:2}\right) \\
&= q_1\left(x_1^1\right) K_1\left(x_1^2 \middle|\, x_1^1\right) q_2\left(x_2^1 \middle|\, x_1^2\right) K_2\left(x_1^3, x_2^2 \middle|\, x_1^2, x_2^1\right) \\
&\times \cdots \times q_n\left(x_n^1 \middle|\, x_{n-1}^2\right) K_n\left(x_{n-L+1}^{L+1}, \ldots, x_{n-1}^3, x_n^2 \middle|\, x_{1:n-L}^{L+1}, x_{n-L+1}^L, \ldots, x_{n-1}^2, x_n^1\right).
\end{aligned}$$

This sequence of target distributions admits the filtering distributions of interest as marginals. The clear theoretical advantage of using MCMC moves is that the use of even non-ergodic MCMC kernels $\{K_n\}$ can only improve the mixing properties of $\{\pi_n\}$ compared to the "natural" sequence of filtering distributions; this explains why these algorithms outperform a standard particle filter for a given number of particles.

Finally, we note that the incorporation of MCMC moves to improve sample diversity is an idea which is appealing in its simplicity and which can easily be incorporated into any of the algorithms described here.

4.5 Block Sampling

The Resample-Move method discussed in the previously section suffers from a major drawback. Although it does allow us to reintroduce some diversity among the set of particles after the resampling step over a lag of length L, the importance weights have the same expression as for the standard particle filter. So this strategy does not significantly decrease the number of resampling steps compared to a standard approach. It can partially mitigate the problem associated with resampling, but it does not prevent these resampling steps in the first place.

[5]Once again, similar expressions can be obtained in the presence of resampling and the technique remains valid.

An alternative *block sampling* approach has recently been proposed in [15]. This approach goes further than the Resample-Move method, which aims to sample only the component x_n at time n in regions of high probability mass and then to uses MCMC moves to rejuvenate $x_{n-L+1:n}$ after a resampling step. The block sampling algorithm attempts to directly sample the components $x_{n-L+1:n}$ at time n; the previously-sampled values of the components $x_{n-L+1:n-1}$ sampled are simply discarded. In this case, it can easily be shown that the optimal importance distribution (that which minimises the variance of the importance weights at time n) is:

$$p\left(x_{n-L+1:n}\middle|\, y_{n-L+1:n}, x_{n-L}\right) = \frac{p\left(x_{n-L:n}, y_{n-L+1:n}\right)}{p\left(y_{n-L+1:n}\middle|\, x_{n-L}\right)} \tag{44}$$

where

$$p\left(y_{n-L+1:n}\middle|\, x_{n-L}\right) = \int \prod_{k=n-L+1}^{n} f\left(x_k\middle|\, x_{k-1}\right) \cdot g\left(y_k\middle|\, x_k\right) dx_{n-L+1:n}. \tag{45}$$

As in the standard case (corresponding to $L = 1$), it is typically impossible to sample from (44) and/or to compute (45). So in practice we need to design an importance distribution $q\left(x_{n-L+1:n}\middle|\, y_{n-L+1:n}, x_{n-L}\right)$ approximating $p\left(x_{n-L+1:n}\middle|\, y_{n-L+1:n}, x_{n-L}\right)$. Henceforth, we consider the case where $L > 1$.

The algorithm proceeds as follows.

SMC Block Sampling for Filtering

At time $n = 1$

- Sample $X_1^i \sim q(x_1|\, y_1)$.
- Compute the weights $w_1\left(X_1^i\right) = \frac{\mu\left(X_1^i\right) g\left(y_1|X_1^i\right)}{q\left(X_1^i|y_1\right)}$ and $W_1^i \propto w_1\left(X_1^i\right)$.
- Resample $\left\{W_1^i, X_1^i\right\}$ to obtain N equally-weighted particles $\left\{\frac{1}{N}, \overline{X}_1^i\right\}$.

At time $1 < n < L$

- Sample $X_{1:n}^i \sim q(x_{1:n}|\, y_{1:n})$.
- Compute the weights $\alpha_n\left(X_{1:n}^i\right) = \frac{p\left(X_{1:n}^i, y_{1:n}\right)}{q\left(X_{1:n}^i\middle|y_{1:n}\right)}$ and $W_n^i \propto \alpha_n\left(X_{n-1:n}^i\right)$.
- Resample $\left\{W_n^i, X_{1:n}^i\right\}$ to obtain N equally-weighted particles $\left\{\frac{1}{N}, \overline{X}_{1:n}^i\right\}$.

At time $n \geq L$

- Sample $X_{n-L+1:n}^i \sim q(x_{n-L+1:n}|y_{n-L+1:n}, \overline{X}_{1:n-1}^i)$.
- Compute the weights

$$w_n\left(\overline{X}_{n-L:n-1}^i, X_{n-L+1:n}^i\right) = \frac{p\left(\overline{X}_{1:n-L}^i, X_{n-L+1:n}^i, y_{1:n}\right) q(\overline{X}_{n-L+1:n-1}^i|y_{n-L+1:n-1}, \overline{X}_{n-L}^i)}{p\left(\overline{X}_{1:n-1}^i, y_{1:n-1}\right) q(X_{n-L+1:n}^i|y_{n-L+1:n}, \overline{X}_{n-L}^i)} \tag{46}$$

and $W_n^i \propto w_n\left(\overline{X}_{n-L:n-1}^i, X_{n-L+1:n}^i\right)$.

• Resample $\left\{W_n^i, \overline{X}_{1:n-L}^i, X_{n-L+1:n}^i\right\}$ to obtain N new equally weighted particles $\left\{\frac{1}{N}, \overline{X}_{1:n}^i\right\}$.

When the optimal IS distribution is used $q\left(x_{n-L+1:n}\middle|\, y_{n-L+1:n}, x_{n-L}\right) = p\left(x_{n-L+1:n}\middle|\, y_{n-L+1:n}, x_{n-L}\right)$, we obtain

$$\begin{aligned} w_n\left(\overline{x}_{1:n-1}, x_{n-L+1:n}\right) &= \frac{p\left(\overline{x}_{1:n-L}, x_{n-L+1:n}, y_{1:n}\right) p(\overline{x}_{n-L+1:n-1}|y_{n-L+1:n-1}, \overline{x}_{n-L})}{p\left(\overline{x}_{1:n-1}, y_{1:n-1}\right) p\left(x_{n-L+1:n}\middle|\, y_{n-L+1:n}, \overline{x}_{n-L}\right)} \\ &= p\left(y_n\middle|\, y_{n-L+1:n}, \overline{x}_{n-L}\right). \end{aligned}$$

This optimal weight has a variance which typically decreases exponentially fast with L (under mixing assumptions). Hence, in the context of adaptive resampling, this strategy dramatically reduces the number of resampling steps. In practice, we cannot generally compute this optimal weight and thus use (46) with an approximation of $p\left(x_{n-L+1:n}\middle|\, y_{n-L+1:n}, x_{n-L}\right)$ for $q\left(x_{n-L+1:n}\middle|\, y_{n-L+1:n}, x_{n-L}\right)$. When a good approximation is available, the variance of (46) can be reduced significantly compared to the standard case where $L = 1$.

Again, this algorithm is a specific application of the generic SMC algorithm discussed in Section 3. To simply notation we write $q_n\left(x_{n-L+1:n}\middle|\, x_{n-L}\right)$ for $q\left(x_{n-L+1:n}\middle|\, y_{n-L+1:n}, x_{n-L}\right)$. To clarify our argument, it is again necessary to add a superscript to the variables; e.g. X_k^p corresponds to the p^{th} time the random variable X_k is sampled; remember that in the present section, this superscript *does not* denote the particle index. Using this notation, the algorithm corresponds to the generic SMC algorithm associated with the following sequence of target distributions

$$\begin{aligned} &\pi_n\left(x_1^{1:L}, \ldots, x_{n-L+1}^{1:L}, x_{n-L+2}^{1:L-1}, \ldots, x_n^1\right) \\ &= p\left(x_{1:n-L+1}^L, x_{n-L+2}^{L-1}, \ldots, x_n^1\middle|\, y_{1:n}\right) q_{n-1}\left(x_{n-L+1}^{L-1}, \ldots, x_{n-1}^1\middle|\, x_{n-L}^L\right) \\ &\times \cdots q_2\left(x_1^2, x_2^1\right) q_1\left(x_1^1\right). \end{aligned}$$

where, if no resampling is used, a path is sampled according to

$$\begin{aligned} &q_n\left(x_1^{1:L}, \ldots, x_{n-L+1}^{1:L}, x_{n-L+2}^{1:L-1}, ..., x_n^1\right) \\ &= q_1\left(x_1^1\right) q_2\left(x_1^2, x_2^1\right) \times \cdots \times q_n\left(x_{n-L+1}^L, ..., x_n^1\middle|\, x_{n-L}^L\right). \end{aligned}$$

The sequence of distributions admits the filtering distributions of interest as marginals. The mixing properties of $\{\pi_n\}$ are also improved compared to the 'natural' sequence of filtering distributions; this is the theoretical explanation of the better performance obtained by these algorithms for a given number of particles.

4.6 Rao-Blackwellised Particle Filtering

Let us start by quoting Trotter [36]: "A good Monte Carlo is a dead Monte Carlo". Trotter specialised in Monte Carlo methods and did not advocate that we should not use them, but that we should avoid them whenever possible. In particular, whenever an integral can be calculated analytically doing so should be preferred to the use of Monte Carlo techniques.

Assume, for example, that one is interested in sampling from $\pi(x)$ with $x = (u, z) \in \mathcal{U} \times \mathcal{Z}$ and

$$\pi(x) = \frac{\gamma(u, z)}{Z} = \pi(u)\, \pi\left(z\middle|\, u\right)$$

where $\pi(u) = Z^{-1}\gamma(u)$ and

$$\pi\left(z\middle|\, u\right) = \frac{\gamma(u, z)}{\gamma(u)}$$

admits a closed-form expression; e.g. a multivariate Gaussian. Then if we are interested in approximating $\pi(x)$ and computing Z, we only need to perform a MC approximation $\pi(u)$ and $Z = \int \gamma(u)\, du$ on the space $\mathcal{U}$ instead of $\mathcal{U} \times \mathcal{Z}$. We give two classes of important models where this simple idea can be used successfully.

4.6.1 Conditionally linear Gaussian models

Consider $\mathcal{X} = \mathcal{U} \times \mathcal{Z}$ with $\mathcal{Z} = \mathbb{R}^{n_z}$. Here $X_n = (U_n, Z_n)$ where $\{U_n\}$ is a unobserved Markov process such that $U_1 \sim \mu_U(u_1)$, $U_n | U_{n-1} \sim f_U(u_n | U_{n-1})$ and conditional upon $\{U_n\}$ we have a linear Gaussian model with $Z_1 | U_1 \sim \mathcal{N}(0, \Sigma_{U_1})$ and

$$\begin{aligned} Z_n &= A_{U_n} Z_{n-1} + B_{U_n} V_n, \\ Y_n &= C_{U_n} Z_n + D_{U_n} W_n \end{aligned}$$

where $V_n \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I_{n_v})$, $W_n \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I_{n_w})$ and for any $u \in \mathcal{U}$ $\{A_u, B_u, C_u, D_u\}$ are matrices of appropriate dimensions. In this case we have $\mu(x) = \mu(u, z) = \mu_U(u)\mathcal{N}(x; 0, \Sigma_z)$, $f(x' | x) = f((u', z') | (u, z)) = f_U(u' | u)\mathcal{N}\left(z'; A_{u'} z, B_{u'} B_{u'}^{\mathrm{T}}\right)$ and $g(y | x) = g(y | (u, z)) = \mathcal{N}\left(y; C_u z, D_u D_u^{\mathrm{T}}\right)$. The switching state-space model discussed in *Example* 3 corresponds to the case where $\{U_n\}$ is a finite state-space Markov process.

We are interested in estimating

$$p(u_{1:n}, z_{1:n} | y_{1:n}) = p(u_{1:n} | y_{1:n})\, p(z_{1:n} | y_{1:n}, u_{1:n}).$$

Conditional upon $\{U_n\}$, we have a standard linear Gaussian model $\{Z_n\}$, $\{Y_n\}$. Hence $p(z_{1:n} | y_{1:n}, u_{1:n})$ is a Gaussian distribution whose statistics can be computed using Kalman techniques; e.g. the marginal $p(z_n | y_{1:n}, u_{1:n})$ is a Gaussian distribution whose mean and covariance can be computed using the Kalman filter. It follows that we only need to use particle methods to approximate

$$\begin{aligned} \gamma_n(u_{1:n}) &= p(u_{1:n}, y_{1:n}) \\ &= p(u_{1:n})\, p(y_{1:n} | u_{1:n}) \end{aligned}$$

where $p(u_{1:n})$ follows from the Markov assumption on $\{U_n\}$ and $p(y_{1:n} | u_{1:n})$ is a marginal likelihood which can be computed through the Kalman filter. In this case, we have

$$\begin{aligned} q^{\mathrm{opt}}(u_n | y_{1:n}, u_{1:n-1}) &= p(u_n | y_{1:n}, u_{1:n-1}) \\ &= \frac{p(y_n | y_{1:n-1}, u_{1:n})\, f_U(u_n | u_{n-1})}{p(y_n | y_{1:n-1}, u_{1:n-1})}. \end{aligned}$$

The standard SMC algorithm associated with $\gamma_n(u_{1:n})$ and the sequence of IS distributions $q(u_n | y_{1:n}, u_{1:n-1})$ proceeds as follows.

SMC for Filtering in Conditionally Linear Gaussian Models

At time $n = 1$

- Sample $U_1^i \sim q(u_1 | y_1)$.
- Compute the weights $w_1\left(U_1^i\right) = \frac{\mu_U\left(U_1^i\right) p\left(y_1 | U_1^i\right)}{q\left(U_1^i | y_1\right)}$ and $W_1^i \propto w_1\left(U_1^i\right)$.
- Resample $\left\{W_1^i, U_1^i\right\}$ to obtain N equally-weighted particles $\left\{\frac{1}{N}, \overline{U}_1^i\right\}$.

At time $n \geq 2$

- Sample $U_n^i \sim q\left(u_n | y_{1:n}, \overline{U}_{1:n-1}^i\right)$ and set $U_{1:n}^i \leftarrow \left(\overline{U}_{1:n-1}^i, U_n^i\right)$.

- Compute the weights $\alpha_n\left(U_{1:n}^i\right) = \frac{p\left(y_n|y_{1:n-1},U_{1:n}^i\right)f_U\left(U_n^i|U_{n-1}^i\right)}{q\left(U_n^i|y_{1:n},U_{1:n-1}^i\right)}$ and $W_n^i \propto \alpha_n\left(U_{1:n}^i\right)$.

- Resample $\left\{W_n^i, U_{1:n}^i\right\}$ to obtain N new equally-weighted particles $\left\{\frac{1}{N}, \overline{U}_{1:n}^i\right\}$.

This algorithm provides also two approximations of $p\left(u_{1:n}|y_{1:n}\right)$ given by

$$\widehat{p}\left(u_{1:n}|y_{1:n}\right) = \sum_{i=1}^{N} W_n^i \delta_{U_{1:n}^i}\left(u_{1:n}\right),$$
$$\overline{p}\left(u_{1:n}|y_{1:n}\right) = \frac{1}{N}\sum_{i=1}^{N} \delta_{\overline{U}_{1:n}^i}\left(u_{1:n}\right)$$

and

$$\widehat{p}\left(y_n|y_{1:n-1}\right) = \frac{1}{N}\sum_{i=1}^{N} \alpha_n\left(U_{1:n}^i\right).$$

At first glance, it seems that this algorithm cannot be implemented as it requires storing paths $\left\{U_{1:n}^i\right\}$ of increasing dimension so as to allow the computation of $p\left(y_n|y_{1:n-1},u_{1:n}\right)$ and sampling from $q\left(u_n|y_{1:n},u_{1:n-1}\right)$. The key is to realise that $p\left(y_n|y_{1:n-1},u_{1:n}\right)$ is a Gaussian distribution of mean $y_{n|n-1}\left(u_{1:n}\right)$ and covariance $S_{n|n-1}\left(u_{1:n}\right)$ which can be computed using the Kalman filter. Similarly, given that the optimal IS distribution only depends on the path $u_{1:n}$ through $p\left(y_n|y_{1:n-1},u_{1:n}\right)$, it is sensible to build an importance distribution $q\left(u_n|y_{1:n},u_{1:n-1}\right)$ which only depends on $u_{1:n}$ through $p\left(y_n|y_{1:n-1},u_{1:n}\right)$. Hence in practice, we do not need to store $\left\{U_{1:n}^i\right\}$ but only $\left\{U_{n-1:n}^i\right\}$ and the Kalman filter statistics associated with $\left\{U_{1:n}^i\right\}$. The resulting particle filter is a bank of interacting Kalman filters where each Kalman filter is used to compute the marginal likelihood term $p\left(y_{1:n}|u_{1:n}\right)$.

4.6.2 Partially observed linear Gaussian models

The same idea can be applied to the class of partially observed linear Gaussian models [2]. Consider $\mathcal{X} = \mathcal{U} \times \mathcal{Z}$ with $\mathcal{Z} = \mathbb{R}^{n_z}$. Here $X_n = (U_n, Z_n)$ with $Z_1 \sim \mathcal{N}\left(0, \Sigma_1\right)$

$$Z_n = AZ_{n-1} + BV_n,$$
$$U_n = CZ_n + DW_n$$

where $V_n \overset{\text{i.i.d.}}{\sim} \mathcal{N}\left(0, I_{n_v}\right)$, $W_n \overset{\text{i.i.d.}}{\sim} \mathcal{N}\left(0, I_{n_w}\right)$; see [2] for generalisations. We make the additional assumption that

$$g\left(y_n|x_n\right) = g\left(y_n|\left(u_n,z_n\right)\right) = g\left(y_n|u_n\right).$$

In this case, we are interested in estimating

$$\begin{aligned} p\left(u_{1:n},z_{1:n}|y_{1:n}\right) &= p\left(u_{1:n}|y_{1:n}\right)p\left(z_{1:n}|y_{1:n},u_{1:n}\right) \\ &= p\left(u_{1:n}|y_{1:n}\right)p\left(z_{1:n}|u_{1:n}\right) \end{aligned}$$

where $p\left(z_{1:n}|u_{1:n}\right)$ is a multivariate Gaussian distribution whose statistics can be computed using a Kalman filter associated with the linear model $\{U_n, Z_n\}$. It follows that we only need to use particle methods to approximate

$$\begin{aligned} \gamma_n\left(u_{1:n}\right) &= p\left(u_{1:n},y_{1:n}\right) \\ &= p\left(u_{1:n}\right)p\left(y_{1:n}|u_{1:n}\right) \end{aligned}$$

where $p(y_{1:n}|u_{1:n}) = \prod_{k=1}^{n} g(y_k|u_k)$ and $p(u_{1:n})$ is the marginal Gaussian prior of $\{U_n\}$ which corresponds to the 'marginal' likelihood term which can be computed using the Kalman filter associated with $\{U_n, Z_n\}$. The resulting particle filter is also an interacting bank of Kalman filters but the Kalman filters are here used to compute the marginal prior $p(u_{1:n})$.

5 Particle Smoothing

We have seen previously that SMC methods can provide an approximation of the sequence of distributions $\{p(x_{1:n}|y_{1:n})\}$. Consequently, sampling from a joint distribution $p(x_{1:T}|y_{1:T})$ and approximating the marginals $\{p(x_n|y_{1:T})\}$ for $n = 1, ..., T$ is straightforward. We just run an SMC algorithm up to time T and sample from/marginalise our SMC approximation $\widehat{p}(x_{1:T}|y_{1:T})$. However, we have seen that this approach is bound to be inefficient when T is large as the successive resampling steps lead to particle degeneracy: $\widehat{p}(x_{1:n}|y_{1:T})$ is approximated by a single unique particle for $n \ll T$. In this section, we discuss various alternative schemes which do not suffer from these problems. The first method relies on an simple fixed-lag approximation whereas the other algorithms rely on the forward-backward recursions presented in Section 2.3.

5.1 Fixed-lag Approximation

The fixed-lag approximation is the simplest approach. It was proposed in [26]. It relies on the fact that, for hidden Markov models with "good" forgetting properties, we have

$$p(x_{1:n}|y_{1:T}) \approx p\left(x_{1:n}|y_{1:\min(n+\Delta,T)}\right) \tag{47}$$

for Δ large enough; that is observations collected at times $k > n+\Delta$ do not bring any additional information about $x_{1:n}$. This suggests a very simple scheme — simply don't update the estimate at time k after time $k = n + \Delta$. Indeed, in practice we just do not resample the components $X_{1:n}^i$ of the particles $X_{1:k}^i$ at times $k > n + \Delta$. This algorithm is trivial to implement but the main practical problem is that we typically do not know Δ. Hence we need to replace Δ with an estimate of it denoted L. If we select $L < \Delta$, then $p\left(x_{1:n}|y_{1:\min(n+L,T)}\right)$ is a poor approximation of $p(x_{1:n}|y_{1:T})$. If we select a large values of L to ensure that $L \geq \Delta$ then the degeneracy problem remains substantial. Unfortunately automatic selection of L is difficult (and, of course, for some poorly-mixing models Δ is so large that this approach is impractical). Experiments on various models have shown that good performance were achieved with $L \approx 20 - 50$. Note that such fixed-lag SMC schemes do not converge asymptotically (i.e. as $N \to \infty$) towards the true smoothing distributions because we do not have $p(x_{1:n}|y_{1:T}) = p\left(x_{1:n}|y_{1:\min(n+L,T)}\right)$. However, the bias might be negligible and can be upper bounded under mixing conditions [8]. It should also be noted that this method does not provide an approximation of the joint $p(x_{1:T}|y_{1:T})$ — it approximates only the marginal distributions.

5.2 Forward Filtering-Backward Smoothing

We have seen previously that it is possible to sample from $p(x_{1:T}|y_{1:T})$ and compute the marginals $\{p(x_n|y_{1:T})\}$ using forward-backward formulæ. It is possible to obtain an SMC approximation of the forward filtering-backward sampling procedure directly by noting that for

$$\widehat{p}(x_n|y_{1:n}) = \sum_{i=1}^{N} W_n^i \delta_{X_n^i}(x_n)$$

we have

$$\widehat{p}\left(x_n \middle| X_{n+1}, y_{1:n}\right) = \frac{f\left(X_{n+1} \middle| x_n\right) \widehat{p}\left(x_n \middle| y_{1:n}\right)}{\int f\left(X_{n+1} \middle| x_n\right) \widehat{p}\left(x_n \middle| y_{1:n}\right) dx_n} = \sum_{i=1}^{N} \frac{W_n^i f\left(X_{n+1} | X_n^i\right) \delta_{X_n^i}\left(x_n\right)}{\sum_{j=1}^{N} W_n^j f\left(X_{n+1} | X_n^j\right)}.$$

Consequently, the following algorithm generates a sample approximately distributed according to $p\left(x_{1:T} \middle| y_{1:T}\right)$: first sample $X_T \sim \widehat{p}\left(x_T \middle| y_{1:T}\right)$ and for $n = T-1, T-2, ..., 1$, sample $X_n \sim \widehat{p}\left(x_n \middle| X_{n+1}, y_{1:n}\right)$.

Similarly, we can also provide an SMC approximation of the forward filtering-backward smoothing procedure by direct means. If we denote by

$$\widehat{p}(x_n | y_{1:T}) = \sum_{i=1}^{N} W_{n|T}^i \delta_{X_n^i}(x_n) \tag{48}$$

the particle approximation of $p(x_n | y_{1:T})$ then, by inserting (48) into (15), we obtain

$$\begin{aligned} \widehat{p}(x_n | y_{1:T}) &= \sum_{i=1}^{N} W_n^i \left[\sum_{j=1}^{N} W_{n+1|T}^j \frac{f\left(X_{n+1}^j | X_n^i\right)}{\left[\sum_{l=1}^{N} W_n^l f\left(X_{n+1}^j | X_n^l\right)\right]} \right] \delta_{X_n^i}(x_n) \\ &:= \sum_{i=1}^{N} W_{n|T}^i \delta_{X_n^i}(x_n). \end{aligned} \tag{49}$$

The forward filtering-backward sampling approach requires $\mathcal{O}(NT)$ operations to sample one path approximately distributed according to $p\left(x_{1:T} \middle| y_{1:T}\right)$ whereas the forward filtering-backward smoothing algorithm requires $\mathcal{O}\left(N^2 T\right)$ operations to approximate $\{p\left(x_n \middle| y_{1:T}\right)\}$. Consequently, these algorithms are only useful for very long time series in which sample degeneracy prevents computationally-cheaper, crude methods from working.

5.3 Generalised Two-filter Formula

To obtain an SMC approximation of the generalised two-filter formula (18), we need to approximate the backward filter $\{\widetilde{p}(x_n | y_{n:T})\}$ and to combine the forward filter and the backward filter in a sensible way. To obtain an approximation of $\{\widetilde{p}(x_n | y_{n:T})\}$, we simply use an SMC algorithm which targets the sequence of distributions $\{\widetilde{p}(x_{n:T} | y_{n:T})\}$ defined in (17). We run the SMC algorithm "backward in time" to approximate $\widetilde{p}(x_T | y_T)$ then $\widetilde{p}(x_{T-1:T} | y_{T-1:T})$ and so on. We obtain an SMC approximation denoted

$$\widehat{\widetilde{p}}(x_{n:T} | y_{n:T}) = \sum_{i=1}^{N} \widetilde{W}_n^i \delta_{X_{n:T}^i}\left(x_{n:T}\right).$$

To combine the forward filter and the backward filter, we obviously cannot multiply the SMC approximations of both $p(x_n | y_{1:n-1})$ and $\widetilde{p}(x_{n:T} | y_{n:T})$ directly, so we first rewrite Eq. (18) as

$$p\left(x_n \middle| y_{1:T}\right) \propto \frac{\int f\left(x_n \middle| x_{n-1}\right) p\left(x_{n-1} \middle| y_{1:n-1}\right) dx_{n-1} \cdot \widetilde{p}(x_n | y_{n:T})}{\widetilde{p}_n\left(x_n\right)}.$$

By plugging the SMC approximations of $p\left(x_{n-1} \middle| y_{1:n-1}\right)$ and $\widetilde{p}(x_n | y_{n:T})$ in this equation, we obtain

$$\widehat{p}(x_n | y_{1:T}) = \sum_{i=1}^{N} W_{n|T}^i \delta_{\widetilde{X}_n^i}(x_n)$$

where

$$W_{n|T}^j \propto \widetilde{W}_n^j \sum_{i=1}^N W_{n-1}^i \frac{f\left(\widetilde{X}_n^j | X_{n-1}^i\right)}{\widetilde{p}_n\left(\widetilde{X}_n^j\right)}. \tag{50}$$

Like the SMC implementation of the forward-backward smoothing algorithm, this approach has a computational complexity $\mathcal{O}(N^2T)$. However fast computational methods have been developed to address this problem [27]. Moreover it is possible to reduce this computational complexity to $\mathcal{O}(NT)$ by using rejection sampling to sample from $p\left(x_n | y_{1:T}\right)$ using $p\left(x_{n-1} | y_{1:n-1}\right)$ and $\widetilde{p}(x_n | y_{n:T})$ as proposal distributions. More recently, an importance sampling type approach has also been proposed in [19] to reduce the computational complexity to $O\left(NT\right)$; see [6] for a related idea developed in the context of belief propagation. Compared to the forward-backward formula, it might be expected to substantially outperform that algorithm in any situation in which the support of the smoothed estimate differs substantially from that of the filtering estimate. That is, in those situations in which observations obtained at time $k > n$ provide a significant amount of information about the state at time n[6]. The improvement arises from the fact that the SMC implementation of the forward-backward smoother simply reweights a sample set which targets $p(x_{1:n}|y_{1:n})$ to account for the information provided by $y_{n+1:k}$ whereas the two filter approach uses a different sample set with locations appropriate to the smoothing distributions.

6 Summary

We have provided a review of particle methods for filtering, marginal likelihood computation and smoothing. Having introduced a simple SMC algorithm, we have shown how essentially all of the particle-based methods introduced in the literature to solve these problems can be interpreted as a combination of two operations: sampling and resampling. By considering an appropriate sequence of target distributions, defined on appropriate spaces, it is possible to interpret all of these algorithms as particular cases of the general algorithm.

This interpretation has two primary advantages:

1. The standard algorithm may be viewed as a particle interpretation of a Feynman-Kac model (see [11]) and hence strong, sharp theoretical results can be applied to all algorithms within this common formulation.
2. By considering all algorithms within the same framework is is possible to develop a common understanding and intuition allowing meaningful comparisons to be drawn and sensible implementation decisions to be made.

Although much progress has been made over the past fifteen years, and the algorithms described above provide good estimates in many complex, realistic settings, there remain a number of limitations and open problems:

- As with any scheme for numerical integration, be it deterministic or stochastic, there exist problems which exist on sufficiently high-dimensional spaces and involving sufficiently complex distributions that it is not possible to obtain an accurate characterisation in a reasonable amount of time.

[6] This situation occurs, for example, whenever observations are only weakly informative and the state evolution involves relatively little stochastic variability. An illustrative example provided in [5] shows that this technique can dramatically outperform all of the other approaches detailed above.

- In many settings of interest the likelihood and state transition density are only known up to a vector of unknown parameters. It is typically of interest to estimate this vector of parameters at the same time as (or in preference to) the vector of states. Formally, such situations can be described as directly as a state space model with a degenerate transition kernel. However, this degeneracy prevents the algorithms described above from working in practice.
- Several algorithms intended specifically for parameter estimation have been developed in recent years. Unfortunately, space constraints prevent us from discussing these methods within the current tutorial. Although progress has been made, this is a difficult problem and it cannot be considered to have been solved in full generality. These issues will be discussed in further depth in an extended version of this tutorial which is currently in preparation.

It should also be mentioned that the SMC algorithm presented in this tutorial can be adapted to perform much more general Monte Carlo simulation: it is not restricted to problems of the filtering type, or even to problems with a sequential character. It has recently been established that it is also possible to employ SMC within MCMC and to obtain a variety of related algorithms.

References

[1] Anderson, B. D. O. and Moore, J. B. (1979) *Optimal Filtering*, Prentice Hall, Englewood Cliffs, New Jersey.

[2] Andrieu, C. and Doucet, A. (2002) Particle filtering for partially observed Gaussian state space models. *Journal of the Royal Statistical Society* B, **6**4(4), 827-836.

[3] Arulampalam, S., Maskell, S., Gordon, N. and Clapp, T. (2002) A tutorial on particle filters for on-line nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, **50**(2):174–188.

[4] Bresler, Y. (1986) Two-filter formula for discrete-time non-linear Bayesian smoothing. *International Journal of Control*, **43**(2), 629–641.

[5] Briers, M., Doucet, A. and Maskell, S. (2008) Smoothing algorithms for state-space models, *Annals of the Institute of Statistical Mathematics*, to appear.

[6] Briers, M., Doucet, A. and Singh, S.S. (2005) Sequential auxiliary particle belief propagation, *Proceedings Conference Fusion*.

[7] Carpenter, J., Clifford, P. and Fearnhead, P. (1999) An improved particle filter for non-linear problems. *IEE proceedings — Radar, Sonar and Navigation*, **146**, 2–7.

[8] Cappé, O. , Godsill, S. J. and Moulines, E. (2007) An overview of existing methods and recent advances in sequential Monte Carlo. *IEEE Proceedings*, **95**(5):899–924.

[9] Chopin, N. (2004) Central limit theorem for sequential Monte Carlo and its application to Bayesian inference. *Ann. Statist.*, **32**, 2385-2411.

[10] Crisan, D. and Doucet, A. (2002) A survey of convergence results on particle filtering for practitioners. *IEEE Transactions on Signal Processing*, **50**(3), 736–746.

[11] Del Moral, P. (2004) *Feynman-Kac Formulae: Genealogical and Interacting Particle Systems with Applications*. Series: Probability and Applications, Springer-Verlag, New York.

[12] Del Moral, P., Doucet, A. and Jasra, A. (2008) On adaptive resampling procedures for sequential Monte Carlo methods. Technical report. INRIA.

[13] Douc, R., Cappé, O. and Moulines, E. (2005) Comparison of resampling schemes for particle filtering. In 4th International Symposium on Image and Signal Processing and Analysis (ISPA).

[14] Doucet, A., Godsill, S. J. and Andrieu, C. (2000) On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, **10**, 197–208.

[15] Doucet, A., Briers, M. and Senecal, S. (2006) Efficient block sampling strategies for sequential Monte Carlo. *Journal of Computational and Graphical Statistics*, **15**(3), 693–711.

[16] Doucet, A., de Freitas, N. and Gordon, N.J. (eds.) (2001) *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, New York.

[17] Doucet, A., de Freitas, N. and Gordon, N.J. (2001) An introduction to sequential Monte Carlo methods. in [16], 3–13.

[18] Fearnhead, P. (2008) Computational methods for complex stochastic systems: A review of some alternatives to MCMC. *Statistics and Computing*, **18**, 151–171.

[19] Fearnhead, P., Wyncoll, D. and Tawn, J. (2008) A sequential smoothing algorithm with linear computational cost. Technical report, Department of Mathematics and Statistics, Lancaster University.

[20] Gilks, W.R. and Berzuini, C. (2001). Following a moving target - Monte Carlo inference for dynamic Bayesian models. *Journal of the Royal Statistical Society* B, **63**, 127–146.

[21] Godsill, S.J. and Clapp, T. (2001) Improvement strategies for Monte Carlo particle filters, in [16], 139-158.

[22] Gordon N.J., Salmond D.J. and Smith A.F.M. (1993) Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE-Proceedings-F*, **140**, 107–113.

[23] Johansen, A.M. and Doucet, A. (2008) A note on auxiliary particle filters. *Statistics and Probability Letters*, **78**(12), 1498–1504.

[24] Johansen, A.M. (2008) SMCTC: Sequential Monte Carlo in C++. Technical Report 08-16. University of Bristol, Department of Statistics.

[25] Kitagawa, G. (1996). Monte Carlo filter and smoother for non-Gaussian non-linear state space models. *Journal of Computational and Graphical Statistics*, **5**, 1-25.

[26] Kitagawa, G. and Sato, S. (2001) Monte Carlo smoothing and self-organizing state-space model, in [16], 177–195.

[27] Klaas, M. et al. (2005) Fast particle smoothing: if I had a million particles, in *Proceedings International Conference on Machine Learning*.

[28] Kong, A, Liu, J.S. and Wong, W. H. (1994) Sequential imputations and Bayesian missing data problems. *Journal of the American Statistical Association*, **89**, 1032–1044.

[29] Künsch, H. R. (2001) State-space and hidden Markov models. in *Complex Stochastic Systems* (eds. O. E. Barndorff-Nielsen, D. R. Cox and C. Kluppelberg), CRC Press, 109–173.

[30] Liu, J. S. (2001) *Monte Carlo Strategies in Scientific Computing*. Springer-Verlag, New York.

[31] Neal, R. M. (2001) Annealed importance sampling. *Statistics and Computing*, **11**, 125–139.

[32] Olsson, J., Cappé, O., Douc., R. and Moulines, E. (2008) Sequential Monte Carlo smoothing with application to parameter estimation in non-linear state space models. *Bernoulli*, **14**(1):155-179.

[33] Pitt, M. K. and Shephard, N. (1999) Filtering via simulation: auxiliary particle filter. *Journal of the American Statistical Association*, **94**, 590–599.

[34] Pitt, M. K. and Shephard, N. (2001) Auxiliary variable based particle filters, in [16], 271–293.

[35] Robert, C. P. and Casella, G. (2004) *Monte Carlo Statistical Methods*. 2nd edn., Springer-Verlag, New York.

[36] Trotter, H. F., and Tukey, J. W. (1956) Conditional Monte Carlo for normal samples, in *Symposium on Monte Carlo Methods* (ed. H. A. Meyer) Wiley, New York, 64–79.

[37] van der Merwe, R., Doucet, A., De Freitas, N. and Wan, E. (2000) The unscented particle filter. in *Proceedings of Neural and Information Processing Systems*.

IET Signal Processing

Research Article

IET Journals
The Institution of Engineering and Technology

Particle filter based on one-step smoothing with adaptive iteration

ISSN 1751-9675
Received on 5th April 2016
Revised 22nd December 2016
Accepted on 31st January 2017
E-First on 3rd May 2017
doi: 10.1049/iet-spr.2016.0194
www.ietdl.org

Zhibin Yan[1], *Yanhua Yuan*[1,2] ✉

[1]*Center for Mathematics and Interdisciplinary Sciences, Harbin Institute of Technology, Harbin 150001, People's Republic of China*
[2]*College of Science, Heilongjiang University of Science and Technology, Harbin, 150022, People's Republic of China*
✉ *E-mail: yuanhua_69@sina.com*

Abstract: A new one-step particle smoother is explicitly given in the form of proper weighted samples. It is employed iteratively to improve the importance sampling in particle filtering through incorporating the current measurement information into the a priori distribution. An adaptive iteration strategy is proposed to accelerate the running, which introduces a parameter into the weight increment to adjust the iteration process. Then, new particle filtering method can be constructed through combining the one-step smoothing and the adaptive iteration strategy.

1 Introduction

Since Gordon *et al.* [1] proposed the classical particle filter (PF), i.e., bootstrap filter, filtering algorithms of such kind have been widely investigated in recent years for their ability to deal with systems without the linear and/or Gaussian assumptions [2–5]. In practical application, such algorithms often suffer from two obstacles: one is the particle degeneracy, and the other is the expensive computational cost when a large number of particles are required to improve the estimation precision [6, 7].

We know that the performance of PF heavily depends on the choice of trial distribution ([7–10], and also see Remark 1). The smaller the difference between the trial and target distribution, the better the weighted particles drawn from the trial distribution approximate the target distribution [10, 11]. For non-linear filtering problem, the filtering distribution as target distribution is difficult for direct sampling. Many efforts have been devoted to design an appropriate trial distribution for improving the performance of PF. In [1], the authors adopted the roughening and prior editing technique. In [12–17], the iterative extended Kalman PF (IEK-PF) was proposed to decrease the difference between the trial and target distribution. The predicting stage of IEK-PF is the same as in the bootstrap filter, but in the updating stage, each of predicted particles is used as an initial value to input into iterative extended Kalman filter (IEKF). Then the final output of IEKF is weighted as the filtering result of IEK-PF. The main idea of IEK-PF is to modify the trial distribution by IEKF, which uses the previous filtering result as the nominal value of EKF at the current time. In [18], IEKF was also used in the updating stage, while an adaptive annealing parameter was designed to accelerate the speed of estimation. In [6], a hybrid filter combining finite impulse response filter and PF was proposed to localise the mobile robot quickly and accurately based on wireless sensor networks. In [19], an adaptive strategy of variable sampling size based on Kullback–Leibler distance was used to improve the sampling efficiency. A good trial distribution was designed in [20] through employing particle Markov chain Monte Carlo method, and the particle flow technique was used to improve the performance of non-linear filtering in [21]. The variational Bayesian inference [22, 23], which combines the variational idea and Bayesian inference, can be used to estimate the state and unknown model parameters of the system.

In this paper, we explicitly give a new one-step particle smoother in the form of proper weighted samples, which is, as conditional distribution, the optimal estimation of the one-step past state based on the measurement information up to the current step. In the original particle filtering, the trial distribution does not contain the current measurement information, which appears in, however, the filtering distribution. All improving methods are to incorporate the current measurement information into the trial in some manner. We use the obtained new one-step particle smoother to improve the trial to contain the current measurement information in some optimal manner. To accelerate the running of the algorithm, we suggest an adaptive iteration strategy which uses a parameter to adjust the iteration process. Then, we can construct new PFs combining the one-step smoothing and the adaptive iteration strategy. These two techniques help to overcome the two obstacles aforementioned as shown in the simulation study on indoor localisation of mobile robot.

This paper is organised as follows. In Section 2, a convenient checking for proper weighted samples is given, and a one-step particle smoother is proposed in the form of proper weighted samples. In Section 3, the one-step smoother is improved through adaptive iteration to incorporate the current measurement to the trial distribution for particle filtering. In Section 4, an indoor localisation problem of mobile robot is studied using our proposed particle filtering method. Finally, the conclusion is drawn in Section 5.

2 One-step particle smoother

2.1 Weak checking for proper weighted samples

The concept of proper weighted samples has been proposed in [11, 24], which formulate the PF method into the uniform framework of sequential Monte Carlo theory. To verify whether a set of weighted samples is proper to a given distribution, by definition, one needs to check (2) for all measurable functions. As a preparation to study one-step particle smoother, we give a checking method in Proposition 1, which is weak in the sense that we only need to check the equation for indicating measurable functions. Also, we use it to interpret the importance sampling in Proposition 2.

Proposition 1: A set of weighted random samples $(\boldsymbol{x}^{(j)}, w^{(j)})$, $j = 1, \ldots, m$, is proper with respect to a distribution π, if and only if for any measurable set A, the following equation:

$$\lim_{m \to \infty} \frac{\sum_{j=1}^{m} \chi_A(\boldsymbol{x}^{(j)}) w^{(j)}}{\sum_{j=1}^{m} w^{(j)}} = E_{\pi}[\chi_A(\boldsymbol{X})] \tag{1}$$

holds with probability 1, where $\chi_A(\cdot)$ is the indicating measurable function of A.

Proof: Since $\chi_A(\cdot)$ is measurable, according to the definition of proper weighted random samples, the 'only if' part holds obviously. For the 'if' part, we arbitrarily take an integrable measurable function $h(\cdot)$ to prove

$$\lim_{m\to\infty}\frac{\sum_{j=1}^{m}h(\boldsymbol{x}^{(j)})w^{(j)}}{\sum_{j=1}^{m}w^{(j)}}=E_{\pi}[h(\boldsymbol{X})]\,. \tag{2}$$

We can take a sequence of simple functions $h_n(\cdot), n=1,2,\ldots$, which satisfies $\lim_{n\to\infty}h_n(\boldsymbol{x})=h(\boldsymbol{x})$, and $|h_n(\boldsymbol{x})|\le h(\boldsymbol{x})$. Simple function is, by definition, linear combination of a finite number of indicating measurable functions, and (2) is linear about h. Therefore, that (2) holds for simple function h_n comes from that (1) holds for any indicating measurable function. Lastly, taking $n\to\infty$, the result follows from Lebesgue's dominated convergence theorem [25]. □

Proposition 2: Given two distributions $\pi(\boldsymbol{x})$ and $g(\boldsymbol{x})$, suppose that the samples $\boldsymbol{x}^{(j)}$ are drawn from $g(\boldsymbol{x})$, and the weights $w^{(j)}$ are defined by

$$w^{(j)}=\frac{\pi(\boldsymbol{x}^{(j)})}{g(\boldsymbol{x}^{(j)})},\quad \text{for}\quad j=1,2,\ldots,m\,. \tag{3}$$

Then the set of weighted samples $(\boldsymbol{x}^{(j)},w^{(j)})$ is proper with respect to $\pi(\boldsymbol{x})$.

Proof: For any measurable set A, by the law of large number we have

$$\begin{aligned}\lim_{m\to\infty}\frac{\sum_{j=1}^{m}\chi_A(\boldsymbol{x}^{(j)})w^{(j)}}{\sum_{j=1}^{m}w^{(j)}}&=\frac{\lim_{m\to\infty}(1/m)\sum_{j=1}^{m}\chi_A(\boldsymbol{x}^{(j)})w^{(j)}}{\lim_{m\to\infty}(1/m)\sum_{j=1}^{m}w^{(j)}}\\&=\frac{E_g[\chi_A(\boldsymbol{X})w(\boldsymbol{X})]}{E_g[w(\boldsymbol{X})]}\end{aligned}$$

holds with probability 1. Since

$$\begin{aligned}\frac{E_g[\chi_A(\boldsymbol{X})w(\boldsymbol{X})]}{E_g[w(\boldsymbol{X})]}&=\frac{\int\chi_A(\boldsymbol{x})w(\boldsymbol{x})g(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}}{\int w(\boldsymbol{x})g(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}}\\&=\frac{\int\chi_A(\boldsymbol{x})(\pi(\boldsymbol{x})/g(\boldsymbol{x}))g(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}}{\int(\pi(\boldsymbol{x})/g(\boldsymbol{x}))g(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}}\\&=\frac{\int\chi_A(\boldsymbol{x})\pi(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}}{\int\pi(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}}\\&=E_{\pi}[\chi_A(\boldsymbol{X})]=\pi(\boldsymbol{X}\in A),\end{aligned}$$

The result holds from Proposition 1. □

Remark 1: From this proposition, we can see that the probability distribution $\pi(\boldsymbol{x})$ can be approximated by that determined by probability mass function $p(\boldsymbol{x}^{(j)})=\tilde{w}_m^{(j)}, j=1,2,\ldots,m$, where $\tilde{w}_m^{(j)}=w^{(j)}/\sum_{j=1}^{m}w^{(j)}$. This interprets the importance sampling. For convenience, the distribution $\pi(\boldsymbol{x})$ appearing in Proposition 2 is called target distribution, and $g(\boldsymbol{x})$ trial distribution.

2.2 One-step smoothing in weighted samples

The state-space model of dynamic system contains two parts:

- State equation

$$\boldsymbol{x}_k=\boldsymbol{f}_k(\boldsymbol{x}_{k-1},\boldsymbol{\xi}_{k-1}), \tag{4}$$

  which determines the one-step transition probability density $q_k(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$.

- Observation equation

$$\boldsymbol{y}_k=\boldsymbol{h}_k(\boldsymbol{x}_k,\boldsymbol{v}_k), \tag{5}$$

  which determines the measurement probability density $\phi_k(\boldsymbol{y}_k|\boldsymbol{x}_k)$.

The probability distributions of $x_0,\boldsymbol{\xi}_k$, and $\boldsymbol{v}_k$ are known, the functions $\boldsymbol{f}_k$ and $\boldsymbol{h}_k$ are deterministic, and $x_0,\boldsymbol{\xi}_k$, and $\boldsymbol{v}_k$ are independent. In various applications, we want to estimate the state when the observations $\boldsymbol{y}_k$ are received sequentially.

According to probability theory, the filtering problem can be formulated into recursively computing the distribution $\pi_k(\boldsymbol{x}_k)=p(\boldsymbol{x}_k|\boldsymbol{y}_1,\boldsymbol{y}_2,\ldots,\boldsymbol{y}_k)$ at time $k, k=1,2,\ldots$, by

$$\pi_k(\boldsymbol{x}_k)\propto\int q_k(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})\phi_k(\boldsymbol{y}_k|\boldsymbol{x}_k)\pi_{k-1}(\boldsymbol{x}_{k-1})\mathrm{d}\boldsymbol{x}_{k-1}\,. \tag{6}$$

By representing a distribution with its weighted samples, particle filtering recursively generates a set of proper weighted samples for $\pi_k(\boldsymbol{x}_k)$ from a given set of proper weighted samples for $\pi_{k-1}(\boldsymbol{x}_{k-1})$ [1].

Theorem 1: Suppose that the set of weighted samples $(\boldsymbol{x}_{k-1}^{(j)},w_{k-1}^{(j)})$ is proper with respect to $\pi_{k-1}(\boldsymbol{x}_{k-1})$. If $\boldsymbol{x}_k^{(j)}$ is randomly sampled from the transition distribution $q_k(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}^{(j)})$, then the set of weighted samples $\{(\boldsymbol{x}_k^{(j)},\boldsymbol{x}_{k-1}^{(j)},w_{k-1}^{(j)})\}_{j=1}^{m}$ is proper with respect to the joint distribution $p(\boldsymbol{x}_k,\boldsymbol{x}_{k-1}|\boldsymbol{y}_1,\ldots,\boldsymbol{y}_{k-1})$.

Proof: Suppose the samples $\boldsymbol{x}_{k-1}^{(j)}$ are from distribution $g(\boldsymbol{x}_{k-1})$, and A, B are two measurable sets of the state space. Since $\boldsymbol{x}_k^{(j)}$ is a sample from the transition distribution $q_k(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}^{(j)})$, then $(\boldsymbol{x}_k^{(j)},\boldsymbol{x}_{k-1}^{(j)})$ is a sample from $g(\boldsymbol{x}_{k-1})q_k(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}^{(j)})$. Then we have

$$\begin{aligned}&\lim_{m\to\infty}\frac{\sum_{j=1}^{m}\chi_{AB}(\boldsymbol{x}_k^{(j)},\boldsymbol{x}_{k-1}^{(j)})w_{k-1}^{(j)}}{\sum_{j=1}^{m}w_{k-1}^{(j)}}\\&=\frac{\int\chi_{AB}(\boldsymbol{x}_k,\boldsymbol{x}_{k-1})w_{k-1}(\boldsymbol{x}_{k-1})g(\boldsymbol{x}_{k-1})q_k(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})\,\mathrm{d}\boldsymbol{x}_k\,\mathrm{d}\boldsymbol{x}_{k-1}}{\int w_{k-1}(\boldsymbol{x}_{k-1})g(\boldsymbol{x}_{k-1})q_k(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})\,\mathrm{d}\boldsymbol{x}_k\,\mathrm{d}\boldsymbol{x}_{k-1}}\\&=\frac{\int\chi_{AB}(\boldsymbol{x}_k,\boldsymbol{x}_{k-1})q_k(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})\pi_{k-1}(\boldsymbol{x}_{k-1})\,\mathrm{d}\boldsymbol{x}_k\,\mathrm{d}\boldsymbol{x}_{k-1}}{\int q_k(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})\pi_{k-1}(\boldsymbol{x}_{k-1})\,\mathrm{d}\boldsymbol{x}_k\,\mathrm{d}\boldsymbol{x}_{k-1}}\\&=E_{p(\boldsymbol{x}_k,\boldsymbol{x}_{k-1}|\boldsymbol{y}_1,\ldots,\boldsymbol{y}_{k-1})}[\chi_{AB}(\boldsymbol{X}_k,\boldsymbol{X}_{k-1})],\end{aligned}$$

where $w_{k-1}(\boldsymbol{x}_{k-1})=\pi_{k-1}(\boldsymbol{x}_{k-1})/g(\boldsymbol{x}_{k-1})$ and $p(\boldsymbol{x}_k,\boldsymbol{x}_{k-1}|\boldsymbol{y}_1,\ldots,\boldsymbol{y}_{k-1})=q_k(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})\pi_{k-1}(\boldsymbol{x}_{k-1})$. □

Theorem 2: Suppose that the set of weighted samples $(\boldsymbol{x}_k^{(j)},\boldsymbol{x}_{k-1}^{(j)},w_{k-1}^{(j)})$ is proper with respect to $p(\boldsymbol{x}_k,\boldsymbol{x}_{k-1}|\boldsymbol{y}_1,\ldots,\boldsymbol{y}_{k-1})$ and define

$$w_k^{(j)}=\phi_k(\boldsymbol{y}_k|\boldsymbol{x}_k^{(j)})w_{k-1}^{(j)}\,. \tag{7}$$

Then the set of weighted samples $(\boldsymbol{x}_k^{(j)},\boldsymbol{x}_{k-1}^{(j)},w_k^{(j)})$ is proper with respect to $p(\boldsymbol{x}_k,\boldsymbol{x}_{k-1}|\boldsymbol{y}_1,\ldots,\boldsymbol{y}_k)$.

Proof: The samples $(\boldsymbol{x}_k^{(j)},\boldsymbol{x}_{k-1}^{(j)})$ are from $g(\boldsymbol{x}_{k-1})q_k(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$, and

$$w_{k-1}^{(j)}=\frac{p(\boldsymbol{x}_{k-1}^{(j)}|\boldsymbol{y}_1,\ldots,\boldsymbol{y}_{k-1})}{g(\boldsymbol{x}_{k-1}^{(j)})}\,.$$

By Bayes formula, we have

$$p(\boldsymbol{x}_k^{(j)}, \boldsymbol{x}_{k-1}^{(j)} | \boldsymbol{y}_1, \ldots, \boldsymbol{y}_k)$$
$$= \frac{\phi_k(\boldsymbol{y}_k | \boldsymbol{x}_k^{(j)}) q_k(\boldsymbol{x}_k^{(j)} | \boldsymbol{x}_{k-1}^{(j)}) p(\boldsymbol{x}_{k-1}^{(j)} | \boldsymbol{y}_1, \ldots, \boldsymbol{y}_{k-1})}{p(\boldsymbol{y}_k | \boldsymbol{y}_1, \ldots, \boldsymbol{y}_k)}.$$

Since

$$\frac{p(\boldsymbol{x}_k^{(j)}, \boldsymbol{x}_{k-1}^{(j)} | \boldsymbol{y}_1, \ldots, \boldsymbol{y}_k)}{g(\boldsymbol{x}_{k-1}^{(j)}) q_k(\boldsymbol{x}_k^{(j)} | \boldsymbol{x}_{k-1}^{(j)})} = \frac{\phi_k(\boldsymbol{y}_k | \boldsymbol{x}_k^{(j)}) w_{k-1}^{(j)}}{p(\boldsymbol{y}_k | \boldsymbol{y}_1, \ldots, \boldsymbol{y}_k)} \propto w_k^{(j)}$$

The result holds by Proposition 2. □

From this theorem, we have the following corollary immediately.

Corollary 1: The set of weighted samples $\{(\boldsymbol{x}_k^{(j)}, w_k^{(j)})\}_{j=1}^m$ is proper with respect to $\pi_k(\boldsymbol{x}_k)$, and the set of weighted samples $\{(\boldsymbol{x}_{k-1}^{(j)}, w_k^{(j)})\}_{j=1}^m$ is proper with respect to $p(\boldsymbol{x}_{k-1} | \boldsymbol{y}_1, \ldots, \boldsymbol{y}_k)$.

Remark 2: Through Theorems 1, 2, and Corollary 1, we obtain the weighted samples of $\pi_k(\boldsymbol{x}_k)$ from the weighted samples of $\pi_{k-1}(\boldsymbol{x}_{k-1})$. Furthermore, the weighted samples of $p(\boldsymbol{x}_{k-1} | \boldsymbol{y}_1, \ldots, \boldsymbol{y}_k)$ are also obtained, which solves the one-step smoothing problem using sampling method. In this way we give a one-step particle smoother.

Theorem 3: Suppose that the set of weighted samples $\{(\boldsymbol{x}_{k-1}^{(j)}, w_k^{(j)})\}_{j=1}^m$ is proper with respect to $p(\boldsymbol{x}_{k-1} | \boldsymbol{y}_1, \ldots, \boldsymbol{y}_k)$. If $\boldsymbol{x}_k^{*(j)}$ is randomly sampled from the transition distribution $q_k(\boldsymbol{x}_k | \boldsymbol{x}_{k-1}^{(j)})$, then $\{(\boldsymbol{x}_k^{*(j)}, w_k^{(j)})\}_{j=1}^m$ is proper with respect to the following distribution on $\boldsymbol{x}_k$:

$$g_k(\boldsymbol{x}_k, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_k) = \int q_k(\boldsymbol{x}_k | \boldsymbol{x}_{k-1}) p(\boldsymbol{x}_{k-1} | \boldsymbol{y}_1, \ldots, \boldsymbol{y}_k) \, \mathrm{d}\boldsymbol{x}_{k-1}. \quad (8)$$

Proof: As in Theorem 1, we can prove that $\{(\boldsymbol{x}_{k-1}^{(j)}, \boldsymbol{x}_k^{*(j)}, w_k^{(j)})\}_{j=1}^m$ is proper with respect to the joint distribution $q_k(\boldsymbol{x}_k | \boldsymbol{x}_{k-1}) p(\boldsymbol{x}_{k-1} | \boldsymbol{y}_1, \ldots, \boldsymbol{y}_k)$ on $(\boldsymbol{x}_{k-1}, \boldsymbol{x}_k)$. Then the result follows from the fact that the defined $g_k(\boldsymbol{x}_k, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_k)$ in (8) is the marginal distribution of $\boldsymbol{x}_k$ to the joint. □

Remark 3: We adopt $g_k(\boldsymbol{x}_k, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_k)$ defined in (8) as trial distribution to construct weighted samples of the filtering distribution $\pi_k(\boldsymbol{x}_k)$, and call this as PF based on one-step smoothing. Now we give some justification for it. According to the terms of Bayes theory, relative to the newly received measurement information $\boldsymbol{y}_k$ at time step k, the filtering distribution $\pi_k(\boldsymbol{x}_k)$ is posterior and $p(\boldsymbol{x}_k | \boldsymbol{y}_1, \ldots, \boldsymbol{y}_{k-1})$ is a priori. From the viewpoint of importance sampling, the classical particle filtering (bootstrap filtering) amounts to take $\pi_k(\boldsymbol{x}_k)$ as target distribution and $p(\boldsymbol{x}_k | \boldsymbol{y}_1, \ldots, \boldsymbol{y}_{k-1})$ as trial distribution. Under this viewpoint, the performance of the algorithm, including whether there is particle degeneracy and whether there is a need for a very large sample size, depends heavily on the distance (the degree of difference) between the distributions $\pi_k(\boldsymbol{x}_k)$ and $p(\boldsymbol{x}_k | \boldsymbol{y}_1, \ldots, \boldsymbol{y}_{k-1})$. It is very natural that if we incorporate the newly received measurement information $\boldsymbol{y}_k$ into the trial in some manner, instead of only using the a priori itself as the trial, then the efficiency of the algorithm may be significantly improved.

3 Improving one-step smoothing by iteration with adaptive parameter

Although the PF based on one-step smoothing should behave well theoretically, the problem of sample degeneracy and computing efficiency still occurs in practice. In this section, we suggest an iteration method and an adaptive strategy to adjust this iteration.

3.1 Improving one-step smoothing by iteration

In this subsection, we suggest an iteration scheme to improve the one-step smoothing so as to further utilise the information $\boldsymbol{y}_k$ in hope of obtaining a better trial distribution. Suppose that we are in time step k. Then the one-step smoothing scheme with iterations is shown as follows:

- Iteration initialisation. Sample $(\boldsymbol{x}_{k-1}^{(j)}, w_k^{(j)})$ from the one-step particle smoother described by Corollary 1, and denote them by $(\boldsymbol{x}_{k-1,0}^{(j)}, w_{k,0}^{(j)})$ as the initial values of the iteration, where $j = 1, 2, \ldots, m$.
- Draw a sample $\boldsymbol{x}_{k,i}^{(j)}$ from $q_k(\boldsymbol{x}_k | \boldsymbol{x}_{k-1,i-1}^{(j)})$.
- Compute $w_{k,i}^{(j)} \propto \phi_k(\boldsymbol{y}_k | \boldsymbol{x}_{k,i}^{(j)}) w_{k,i-1}^{(j)}$.
- Denote $(\boldsymbol{x}_{k-1,i}^{(j)}, w_{k,i}^{(j)}) := (\boldsymbol{x}_{k-1,i-1}^{(j)}, w_{k,i}^{(j)})$.
- Stop the iteration if a prescribed stopping condition is satisfied.

We see that the bigger $\phi_k(\boldsymbol{y}_k | \boldsymbol{x}_{k,i}^{(j)})$, the bigger $w_{k,i}^{(j)}$ which is the weight of $\boldsymbol{x}_{k-1,i-1}^{(j)}$. Denote the distribution obtained after the ith iteration as $p_i(\boldsymbol{x}_{k-1} | \boldsymbol{y}_1, \ldots, \boldsymbol{y}_k)$. Then going along with iterations, the significant region of $p_i(\boldsymbol{x}_{k-1} | \boldsymbol{y}_1, \ldots, \boldsymbol{y}_k)$ is gradually modified so that for any sample from it, the state $\boldsymbol{x}_k$ propagated by this sample will fall into the high likelihood region of $\phi_k(\boldsymbol{y}_k | \boldsymbol{x}_k)$ with more odds. In this way, the aim of obtaining a better trial distribution will be achieved [1].

Intuitively speaking, when the distance between the filtering distribution and the trial distribution determined by the iterations is sufficiently small, we can stop the iterations. Concretely, we adopt the following stopping rule:

- Select a small positive number ρ.
- Compute $d_{ki} = \| \boldsymbol{y}_k - \hat{\boldsymbol{z}}_{ki} \|$, where

$$\hat{\boldsymbol{z}}_{ki} = E_{p_v}[\boldsymbol{h}_k(\boldsymbol{x}_{k,i}, \boldsymbol{v}_k) | \boldsymbol{x}_{k,i} = \hat{x}_{k,i}]$$
$$= \int \boldsymbol{h}_k(\hat{x}_{k,i}, \boldsymbol{v}) p(\boldsymbol{v}) \, \mathrm{d}\boldsymbol{v} \quad \text{and}$$
$$\hat{x}_{k,i} = \frac{\sum_{j=1}^m \boldsymbol{x}_{k,i}^{(j)} w_{k,i}^{(j)}}{\sum_{j=1}^m w_{k,i}^{(j)}}$$

with other notations as in (5).

- Stop the iterations, if

$$d_{ki} \le \rho. \quad (9)$$

This stopping rule stems from the recognition that if the obtained trial distribution is near to the filtering distribution, then the predicted measurement based on it will be near to the real measurement.

Remark 4: When the iteration stops at step i, the weighted samples $(\boldsymbol{x}_{k,i}^{(j)}, w_{k,i}^{(j)})$ represent the filtering distribution $\pi_k(\boldsymbol{x}_k)$. We call this as PF based on one-step smoothing with iteration.

3.2 Adaptive iteration strategy

In practice, the iterations suggested above may stop slowly. To make the algorithm become practically usable, we use an adaptive strategy [18] to accelerate the running of the algorithm. For convenience we call $\phi_k(\boldsymbol{y}_k | \boldsymbol{x}_{k,i}^{(j)})$ the weight increment, which appears in the iteration scheme in the last subsection. The adaptive strategy is to associate the weight increment with a parameter to adaptively adjust the iteration process. How to design such a

parameterised weight increment $\phi_k(\boldsymbol{y}_k|\boldsymbol{x}_{k,i}^{(j)}, \beta_{k,i})$ depends on the concrete form of the observation (5), and we take

$$\boldsymbol{h}_k(\boldsymbol{x}_k, \boldsymbol{v}_k) = \boldsymbol{h}_k(\boldsymbol{x}_k) + \boldsymbol{v}_k$$

to illustrate, where v_k has normal distribution $\mathcal{N}(0, \boldsymbol{R}_k)$. Denote

$$d_{kij} = \sqrt{(\boldsymbol{y}_k - \boldsymbol{z}_{kij})^T \boldsymbol{R}_k^{-1} (\boldsymbol{y}_k - \boldsymbol{z}_{kij})},$$

where $\boldsymbol{z}_{kij} = \boldsymbol{h}_k(\boldsymbol{x}_{k,i}^{(j)})$. We take

$$\phi_k(\boldsymbol{y}_k|\boldsymbol{x}_{k,i}^{(j)}, \beta_{k,i}) = \exp\left\{-\beta_{k,i} d_{kij}^2\right\}.$$

Let γ be a given tiny positive number less than 1. Then, at iteration step i in time step k, the process of determining $\phi_k(\boldsymbol{y}_k|\boldsymbol{x}_{k,i}^{(j)}, \beta_{k,i})$ can be described as follows:

- Compute $d_{kij}, j = 1, 2, \ldots, m$.
- Find $d_{kil} = \min_j \{d_{kij}\}$ and $d_{kiL} = \max_j \{d_{kij}\}$.
- Select

$$\ln \alpha_{k,i} \in \left(0, \frac{d_{kil}^2 - d_{kiL}^2}{d_{kil}^2 + d_{kiL}^2} \ln \gamma\right).$$

- Take

$$\beta_{k,i} \in \left(\frac{-\ln \gamma + \ln \alpha_{k,i}}{d_{kiL}^2}, \frac{-\ln \gamma - \ln \alpha_{k,i}}{d_{kil}^2}\right).$$

- Compute $\phi_k(\boldsymbol{y}_k|\boldsymbol{x}_{k,i}^{(j)}, \beta_{k,i})$ and substitute it for $\phi_k(\boldsymbol{y}_k|\boldsymbol{x}_{k,i}^{(j)})$ appearing in the one-step smoothing scheme with iterations in the subsection 3.1.

Remark 5: The way of taking parameters $\alpha_{k,i}$ and $\beta_{k,i}$ is indeed to solve the following two inequalities:

$$\exp\{-\beta_{k,i} d_{kil}^2\} > \alpha_{k,i}\gamma, \tag{10}$$

$$\exp\{-\beta_{k,i} d_{kiL}^2\} < \frac{\gamma}{\alpha_{k,i}}. \tag{11}$$

The parameter $\beta_{k,i}$ is intended to be an adjusting factor which adaptively adjusts the distance between the real measurement and the predicted measurement based on the sample, and $\alpha_{k,i}$ is designed in order that the two bounds appearing in the inequalities can adaptively vary with the time step k and iteration step i.

4 Simulations

Consider an indoor localisation system based on wireless sensor networks for tracking position of mobile robot [6]. The state of the robot is described by

$$\begin{aligned} x_k &= x_{k-1} + u_{1k}\cos\left(\theta_{k-1} + \frac{1}{2}u_{2k}\right) + w_{1k}, \\ y_k &= y_{k-1} + u_{1k}\sin\left(\theta_{k-1} + \frac{1}{2}u_{2k}\right) + w_{2k}, \\ \theta_k &= \theta_{k-1} + u_{2k} + w_{3k}, \end{aligned}$$

where x_k and y_k are the coordinates on a two-dimensional plane relative to an external coordinate frame, θ_k is the heading angle at time k, and $u_k = [u_{1k}, u_{2k}]^T$ is control signal. The observation is described by

$$\begin{aligned} z_{1k} &= \frac{1}{c}(d_{1k} - d_{2k}) + v_{1k}, \quad z_{2k} = \frac{1}{c}(d_{1k} - d_{3k}) + v_{2k}, \\ z_{3k} &= \frac{1}{c}(d_{1k} - d_{4k}) + v_{3k}, \quad z_{4k} = \theta_k + v_{4k}, \end{aligned}$$

where $c = 299792458$ m/s is the speed of light, $d_{ik} = \sqrt{(x_k - a_i)^2 + (y_k - b_i)^2}$, $i = 1, 2, 3, 4$ represent the distances between the mobile robot and each of the receivers, where (a_i, b_i) are coordinates of the fixed receivers. The zero-mean Gaussian noises $\xi_k = [\boldsymbol{\xi}_{1k}, \boldsymbol{\xi}_{2k}, \boldsymbol{\xi}_{3k}]^T$ and $\boldsymbol{v}_k = [v_{1k}, v_{2k}, v_{3k}, v_{4k}]^T$ have covariance matrices $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k$, respectively. See [6] for details.

In general, small amplitude of measurement noise often leads to particle impoverishment, and then influences the performance of particle filtering [6]. For illustrating the performance of PF based on one-step smoothing with adaptive iteration (AIPF), simulations are performed under two different conditions: the measurement noises of the general and the small amplitude, and in each case using two algorithms: particle filtering based on sequential importance sampling with resampling and roughening (SIRPF, see [1, 7, 9] for details) and the AIPF proposed in this paper, respectively.

In the following simulations, the four receivers are installed at points $(0, 0)$, $(0, 20)$, $(20, 0)$, and $(20, 20)$, all in metres, the control signal u_k is taken to be $[0.1, 1]^T$, the covariance matrices of the process noise are $\boldsymbol{Q}_k = \text{diag}(0.1^2, 0.1^2, 1)$. In following simulations, take the sample size $m = 300$, the parameter $\rho = 6$ for the stopping rule.

4.1 Case 1: The general measurement noise amplitude

We take the measurement noise covariance matrix $\boldsymbol{R}_k = \text{diag}(0.5^2, 0.5^2, 0.5^2, c^2)/c^2$ as the so-called general measurement noise amplitude, and set the initial state $[x_0, y_0, \theta_0]^T = [10, 5, 0]^T$ for generating a state trajectory which simulates and then is used to represent the actual one.

The tracking result of SIRPF is shown in Fig. 1*a*, where the samples representing the filtering estimation at initial time step are taken from normal distribution with expectation $[10, 5, 0]^T$ and covariance matrix $\text{diag}(0.1, 0.1, 1)$. Figs. 1*b–d* show the tracking results of AIPF for the samples representing the initial filtering being taken from normal distribution with expectation $[10, 5, 0]^T$ and covariance matrix $\text{diag}(1, 1, 1)$, normal distribution with expectation $[10, 5, 0]^T$ and covariance matrix $\text{diag}(5, 5, 10)$, and uniform distribution over the square region $[0, 20] \times [0, 20]$, respectively. The initial filtering estimation is based on the a priori information about the initial state before any observation information arrives. For given system and observation equation and system and observation noise, the behaviour of a filtering algorithm relies on the initial filtering estimation. We set the above simulation environment in order to test the sensitivity of our algorithm AIPF relative to the information about the initial state.

Fig. 2*a* shows that at the first 20 time steps the position errors of SIRPF and AIPF are similar. Unlike AIPF, the error of SIRPF increases quickly later. Fig. 2*b* shows that the position errors of AIPF in the three different estimations for initial state. Here, at each time step, the position error is defined as the distance between the estimated and the real positions. Fig. 3 shows the orientation errors, which is defined as the difference of the estimated and the real orientation angles.

We see that SIRPF fails to track although the initial estimation error is set smaller, which has the small covariance matrix $\text{diag}(0.1, 0.1, 1)$. The AIPF can track the position of the mobile robot steadily and is insensitive to the information about the initial state.

Comparing with [6], we use a small sample size $m = 300$ reaching similar tracking effect in [6] which uses a sample size $m = 10000$.

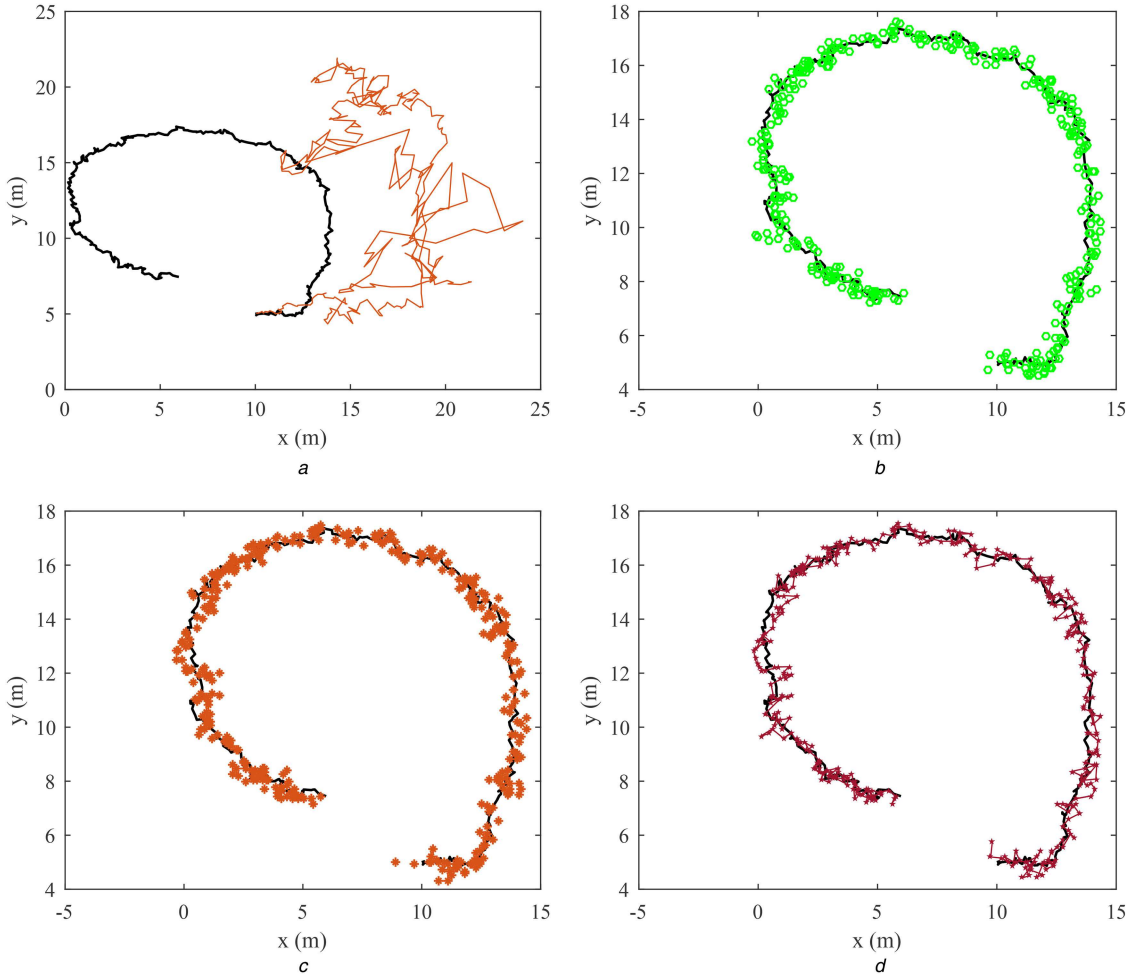


Fig. 1 *Real and estimated position trajectories in Case 1, where the real trajectory is shown by the black solid line*
(a) Estimated position trajectory of SIRPF, under that the initial filtering distribution is normal distribution with expectation [10,5,0]T and covariance matrix *diag* (0.1,0.1,1), ***(b)*** Estimated position trajectory of AIPF, in which the initial filtering distribution is normal distribution with expectation [10,5,0]T and covariance matrix *diag*(1,1,1), and denote this AIPF as AIPF1, ***(c)*** Estimated position trajectory by using AIPF, in which the initial filtering distribution is normal distribution with expectation [10,5,0]T and covariance matrix *diag*(5,5,10), and denote this AIPF as AIPF2, ***(d)*** Estimated position trajectory by using AIPF, where the initial filtering distribution is uniform distribution over square region [0,20]× [0,20], and denote this AIPF as AIPF3.

4.2 Case 2: The small measurement noise amplitude

According to [6], in this small measurement noise situation, localisation by methods of the particle filtering type fails frequently. For simulations, we take $\boldsymbol{R}_k = \mathrm{diag}(0.05^2, 0.05^2, 0.05^2, 0.1^2c^2)/c^2$, which is 1% of that in Case 1. Other conditions for generating the actual trajectory and tracking simulations are the same as in Case 1. The results are shown in Figs. 4–6. We see again that SIRPF fails to track and AIPF tracks well. Furthermore, because of the less measurement noise, AIPF exhibits less steady-state error in Case 2 than in Case 1; compared Figs. 5 and 2.

Compared with the localising result shown in Fig. 6*a* in [6], we find that the significant tracking error at the initial stage appearing there does not occur in our result. This is because the one-step smoothing procedure has the effect to improve the past estimation by the new observation.

5 Conclusion

The proposed one-step particle smoother helps to improve the efficiency of importance sampling in particle filtering. The adaptive strategy of introducing parameterised weight increment can accelerate the computing. It is a future research direction to incorporate some existed strategies, such as roughening and re-sampling, into PF based on one-step smoothing with adaptive iteration.

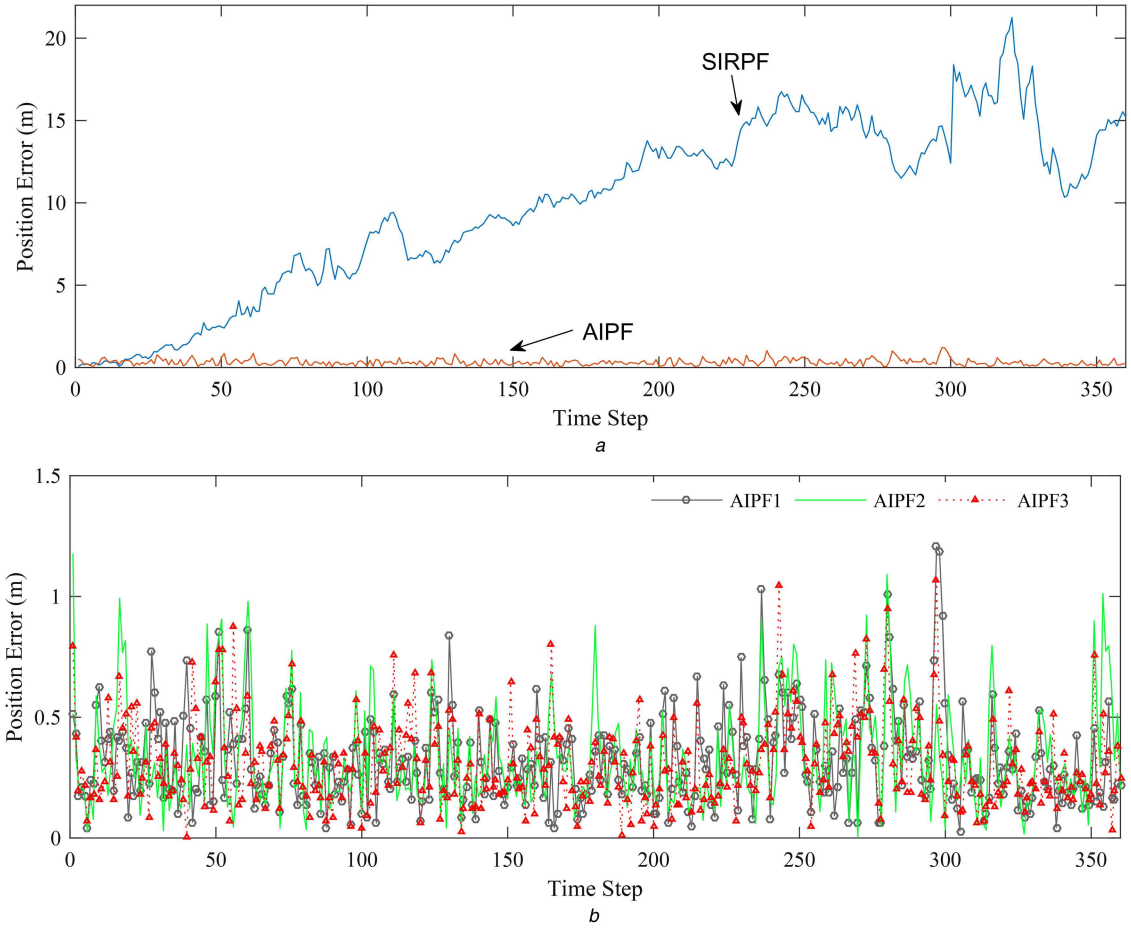


Fig. 2 *Position errors in Case 1*

(a) Position error of SIRPF and AIPF under the same initial filtering distribution, ***(b)*** Position error of AIPF1, AIPF2 and AIPF3

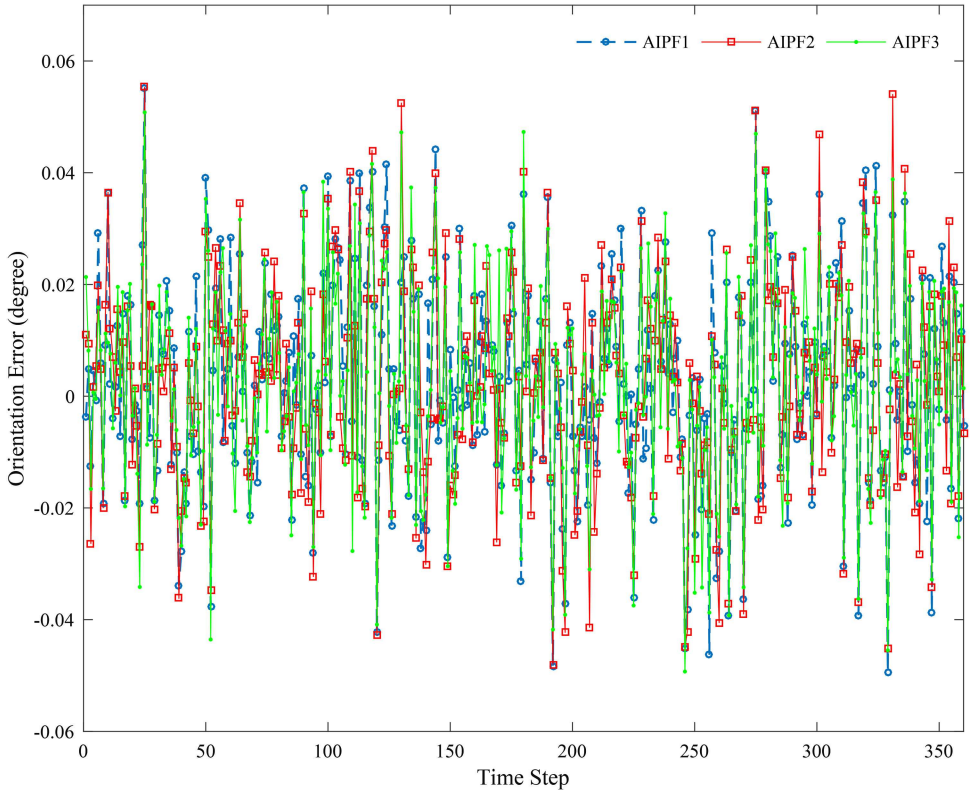


Fig. 3 *Orientation errors of AIPFs in Case 1*

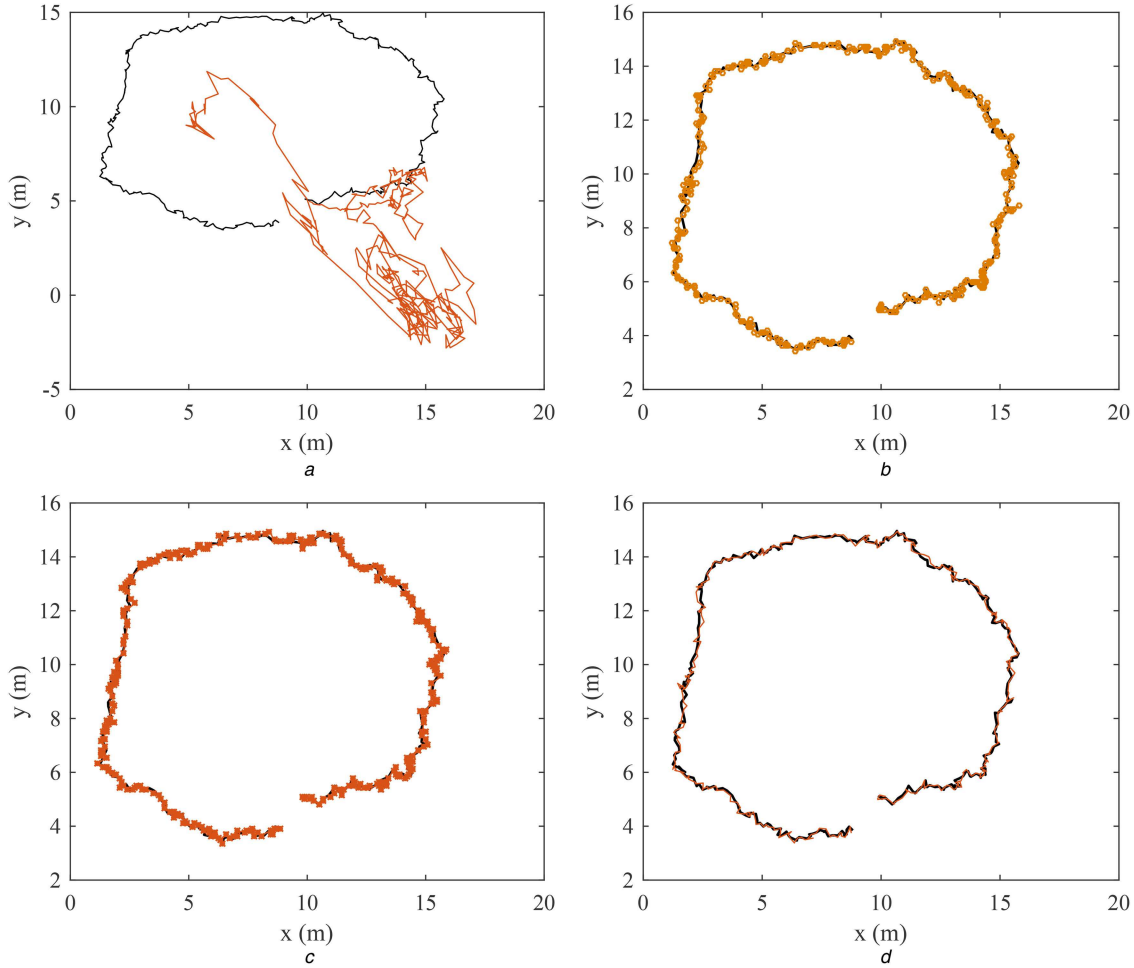


Fig. 4 *Real and estimated position trajectories in Case 2, where the real trajectory is shown by the black solid line*
(a) Estimated position trajectory of SIRPF, ***(b)*** Estimated position trajectory of AIPF1, *(c)* Estimated position trajectory of AIPF2, *(d)* Estimated position trajectory of AIPF3

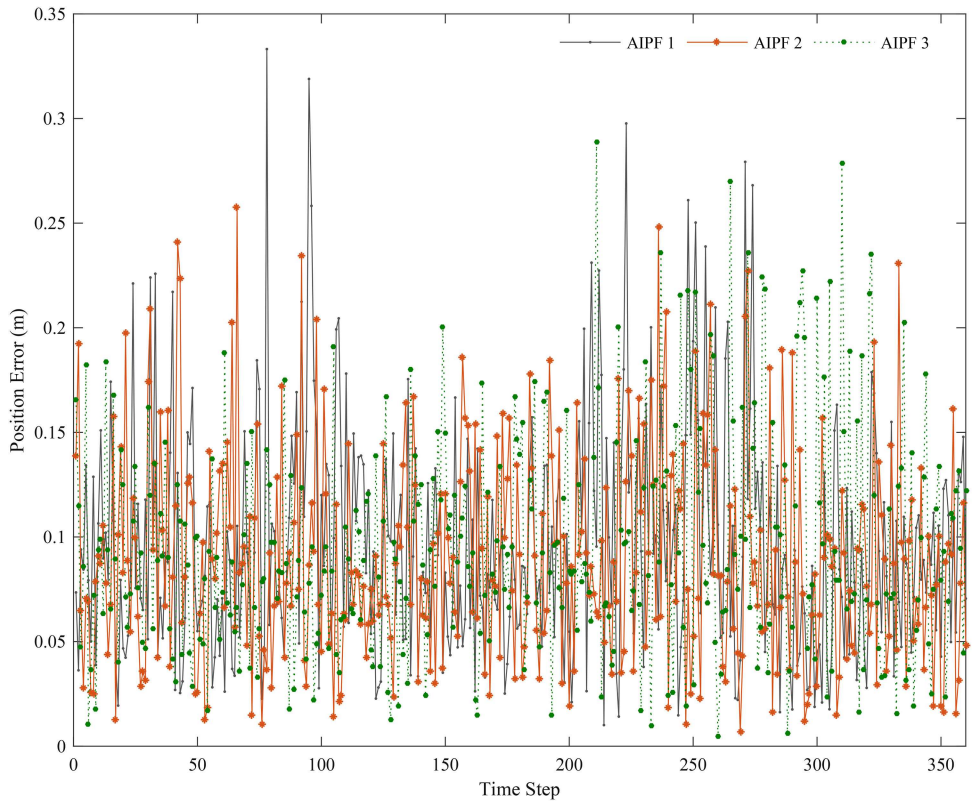


Fig. 5 *Position errors of AIPFs in Case 2*

IET Signal Process., 2017, Vol. 11 Iss. 5, pp. 596-603

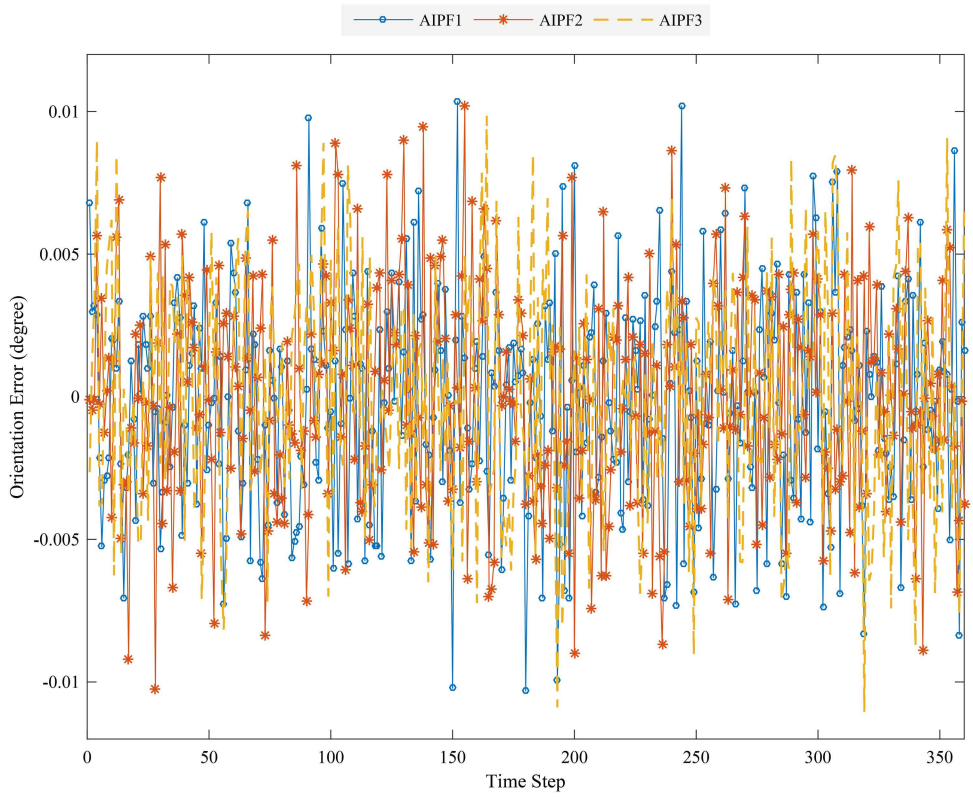


Fig. 6 *Orientation errors of AIPFs in Case 2*

6 References

[1] Gordon, N., Salmond, D., Smith, A.F.: 'Novel approach to nonlinear/non-Gaussian Bayesian state estimation', *IEE Proc. F-Radar Signal Process.*, 1993, **140**, (5), pp. 107–113

[2] Kim, J., Tandal, M., Menon, P.K.: 'Particle filter for ballistic target tracking with glint noise', *J. Guid. Control Dyn.*, 2010, **33**, (6), pp. 1918–1921

[3] Carmi, A., Oshman, Y.: 'Adaptive particle filtering for spacecraft attitude estimation from vector observations', *J. Guid. Control Dyn.*, 2009, **32**, (1), pp. 232–241

[4] Carmi, A., Oshman, Y.: 'Fast particle filtering for attitude and angular-rate estimation from vector observation', *J. Guid. Control Dyn.*, 2009, **32**, (1), pp. 70–78

[5] McKenna, S.J., Nait-Charif, H.: 'Tracking human motion using auxiliary particle filters and iterated likelihood weighting', *Image Vis. Comput.*, 2007, **25**, pp. 852–862

[6] Par, J.M., Ahn, C.K., Lim, M.T.: 'Improving reliability of particle filter-based localization in wireless sensor networks via hybrid particle/FIR filtering', *IEEE Trans. Ind. Inf.*, 2015, **11**, (5), pp. 1089–1098

[7] Doucet, A., Johansen, A.: 'A tutorial on particle filters and smoothing: fifteen years later', in Crisan, D., Rozovsky, B. (ED.): '*The Oxford University handbook of nonlinear filtering*' (Oxford University Press, 2011), pp. 656–704

[8] Doucet, A., Godsill, S., Andrieu, C.: 'On sequential Monte Carlo sampling methods for Bayesian filtering', *Stat. Comput.*, 2000, **10**, pp. 192–208

[9] Arulampalam, M.S., Maskell, S., Gordon, N.J., *et al.*: 'A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking', *IEEE Trans. Signal Process.*, 2002, **50**, pp. 174–188

[10] Pitt, M.K., Shephard, N.: 'Filtering via simulation: auxiliary particle filters', *J. Am. Stat. Assoc.*, 1999, **94**, (446), pp. 590–599

[11] Liu, J.S., Chen, R.: 'Sequential Monte Carlo methods for dynamic systems', *J. Am. Stat. Assoc.*, 1998, **93**, (443), pp. 1032–1044

[12] Zuo, J.Y., Jia, Y.: 'Particle filter guided by iterated extended Kalman filter'. 2013 13th Int. Conf. on Control, Automation and Systems, Gwangju, Korea, October 2013, pp. 1605–1609

[13] Xu, X., Zhao, N., Dong, H.: 'The iterated extended Kalman particle filter for speech enhancement'. Ninth Int. Conf. on Signal Processing, Beijing, China, October 2008, pp. 104–107

[14] Zhou, D., Ravey, A., Gao, F., *et al.*: 'Online estimation of state of charge of Li-ion battery using an iterated extended Kalman particle filter'. 2015 IEEE Transportation Electrification Conf. and Expo (ITEC), Dearborn, USA, June 2015, pp. 1–5

[15] Zhang, J., Ji, H., Xue, Q.: 'A new approach based on particle filter for target tracking with gline noise'. 2009 IEEE Int. Conf. on Systems, Man, and Cybernetics, San Antonio, USA, October 2009, pp. 4791–4795

[16] Long, K.K., Hanh, D.N., Tuan, D.H.: 'Improving iterated extended Kalman filter for non-Gaussian noise environments'. The Sixth Int. Forum on Strategic Technology, Harbin, China, August 2011, pp. 1114–1117

[17] Bell, B.M., Cathey, F.W.: 'The iterated Kalman filter update as a Gauss-Newton method', *IEEE Trans. Autom. Control*, 1993, **38**, (2), pp. 294–297

[18] Zuo, J.Y., Zhang, Y.N., Lian, W.: 'Adaptive iterated particle filter', *Electron. Lett.*, 2013, **49**, (12), pp. 742–744

[19] Fox, D.: 'Adapting the sampling size in particle filterings through KLD-sampling', *Int. J. Robot. Res.*, 2003, **22**, (12), pp. 985–1003

[20] Andrieu, C., Doucet, A., Holenstein, R.: 'Particle Markov chain Monte Carlo methods', *J. R. Stat. Soc. B*, 2010, **72**, (2), pp. 1–33

[21] Daum, F., Huang, J.: 'Generalized particle flow for nonlinear filters'. Processing of SPIE, 2010, vol. **7698**, p. 76980I-1–12

[22] Smidl, V., Quinn, A.: 'Variational Bayesian filtering', *IEEE Trans. Signal Process.*, 2008, **56**, (10), pp. 5020–5030

[23] Sarkka, S., Nummenmaa, A.: 'Recursive noise adaptive Kalman filtering by variational Bayesian approximations', *IEEE Trans. Autom. Control*, 2009, **54**, (3), pp. 596–600

[24] Liu, J.S.: '*Monte Carlo strategies in scientific computing*' (Springer-Verlag Press, New York, 2001, 1st edn.), pp. 23–104

[25] Rudin, W.: '*Real and complex analysis*' (McGraw-Hill Education, New York, 1987, 3rd edn.), p. 26

Nonlin. Processes Geophys., 21, 633–643, 2014
www.nonlin-processes-geophys.net/21/633/2014/
doi:10.5194/npg-21-633-2014

Nonlinear Processes in Geophysics
Open Access

Monte Carlo fixed-lag smoothing in state-space models

A. Cuzol[1] **and E. Mémin**[2]

[1]University of Bretagne-Sud, UMR 6205, LMBA, 56000 Vannes, France
[2]INRIA Rennes-Bretagne Atlantique, Rennes, France

Correspondence to: A. Cuzol (anne.cuzol@univ-ubs.fr)

Received: 16 April 2013 – Revised: 13 March 2014 – Accepted: 14 March 2014 – Published: 28 May 2014

Abstract. This paper presents an algorithm for Monte Carlo fixed-lag smoothing in state-space models defined by a diffusion process observed through noisy discrete-time measurements. Based on a particle approximation of the filtering and smoothing distributions, the method relies on a simulation technique of conditioned diffusions. The proposed sequential smoother can be applied to general nonlinear and multidimensional models, like the ones used in environmental applications. The smoothing of a turbulent flow in a high-dimensional context is given as a practical example.

1 Introduction

The framework of this paper concerns state-space models described by general diffusions of the form

$$\mathrm{d}\boldsymbol{x}(t) = f(\boldsymbol{x}(t))\mathrm{d}t + \sigma(\boldsymbol{x}(t))\mathrm{d}\boldsymbol{B}(t), \quad (1)$$

which are partially observed through noisy measurements at discrete times. Such models can describe many dynamical phenomena in the environmental sciences and physics, but also in finance or engineering applications. The main motivation of this work concerns environmental applications, where nonlinearity and high-dimensionality arise. Indeed, environmental models and data describe nonlinear phenomena over large domains, with high spatial resolution. The continuous dynamical model (Eq. 1) is defined from a priori physical laws, while observations are supplied by sensors (satellite data for instance) and can appear with very low time frequency. As an example, in the application presented in the last part of this paper, the dimension of the state and observations is of the order of many thousands, and the model is described by the nonlinear Navier–Stokes equation. Filtering and smoothing in such state-space models aim at coupling model and observations, which is called data assimilation. The goal of the filtering is to estimate the system state distribution knowing past and present observations. This allows us for instance to give proper initial conditions to forecast the future state of a system characterizing atmospheric or oceanographic flows. On the other hand, the smoothing aims at estimating the state distribution using past and future observations, and this retrospective state estimation allows us to analyze a spatio–temporal phenomenon over a given time period, for climatology studies for instance. Applications of data assimilation are numerous and interest is growing in environmental sciences with the increase in available data. However, it is still a challenge to develop filtering and smoothing methods that can be used within a general nonlinear and high-dimensional context.

Monte Carlo sequential methods, contrary to standard Kalman filters, are able to deal with the filtering problem in nonlinear state-space models. The particle filtering (Del Moral et al., 2001; Doucet et al., 2000) solves the whole filtering equations through Monte Carlo approximations of the state distribution. On the other hand, ensemble Kalman methods (Evensen, 2003) take into account in some way the nonlinearities in the system, but are based on a Gaussian assumption. For high-dimensional systems, ensemble Kalman methods are preferred in practice to particle filters (Stroud et al., 2010; Van Leeuwen, 2009) since they reach a better performance for a limited number of particles. In order to keep this advantage while alleviating the Gaussian assumption, both methods are combined in Papadakis et al. (2010), leading to a particle filter that can be applied to high-dimensional systems. We will use this technique for the filtering step in the high-dimensional application presented in Sect. 5.

The aim of this paper is to propose a new smoothing method. Within the particle filter framework, the smoothing can be computed backward, reweighting past particles

using present observations (Briers et al., 2010; Godsill et al., 2004). There is however one main difficulty for continuous models of type Eq. (1). As a matter of fact, it is necessary to know the transition density of the process between observation times, which is not available for general diffusions. This transition density can be approximated through Monte Carlo simulations, as proposed by Durham and Gallant (2002) to solve inference problems for diffusion processes. However, these approximations are based on Brownian bridge (or modified versions of it) simulations that do not take into account the drift part of the model. For nonlinear and high-dimensional models with a drift term that dominates, such approximations will be inefficient. It is also possible to obtain an unbiased estimate of the transition density (see Beskos et al., 2006), but this approach is not adapted to a multi-dimensional context. As a matter of fact, the use of this technique in a multivariate setting imposes constraints on the diffusion drift (in particular, the drift function has to be of gradient type). In parallel, within the framework of ensemble Kalman methods, Evensen and van Leeuwen (2000) have proposed estimating backward the smoothing distribution in a recursive way, based on existing filtering trajectories; Stroud et al. (2010) presented and applied an ensemble Kalman smoothing method, relying on a linearization of the system dynamics.

All previously mentioned smoothing methods require us to perform specific assumptions or simplifications in order to deal with general nonlinear models of type Eq. (1) in a high-dimensional context. To the best of our knowledge, it remains a challenging problem to develop smoothing methods that can be used in this general setting. In this paper, we deal with this issue sequentially each time a new observation is available, by smoothing the hidden state from this new observation time up to the previous one. This approach, called fixed-lag smoothing, then constitutes a partial answer to the global smoothing problem that would take into account all available observations. Nevertheless, it is reasonable to assume that the distribution of the hidden state depends on future observations through the next observation only, as soon as the time step between measurements is long (which is typically the case in the environmental applications that motivate this work). Under this assumption, a new observation will impact the distribution of the hidden process up to the previous observation only. This point of view justifies the use of a fixed-lag smoothing in our setting as a reasonable approximation of the global smoothing problem.

Such a fixed-lag smoothing may be directly obtained from the particle filtering result, reweighting past trajectories. However, this smoothing will fail in two cases: when the number of particles is too small compared to the size of the system, or when existing trajectories do not correspond to plausible trajectories of the dynamical model. Unfortunately, these two situations have to be faced when smoothing in a high-dimensional state-space model. Firstly, the number of particles has to be reduced for computational reasons. Secondly, particle filters that have been proposed in this high-dimensional context require us to correct trajectories towards the observations (Papadakis et al., 2010; Van Leeuwen and Ades, 2013). This implies that filtering states are consistent at observation times, but also that filtering trajectories may not be plausible realizations of the underlying physical model. In that case, a smoothing based on existing trajectories will fail. Note that these remarks are not only valid for the fixed-lag smoothing, but also for previously mentioned global techniques relying on existing trajectories. In particular, a genealogical smoothing based on ancestral particle lines (Del Moral, 2004) will be deficient in a high-dimensional setting, since many trajectories will share only a few ancestral lines.

In contrast, our method does not rely on existing particles only. It is built on a conditional simulation technique of diffusions proposed by Delyon and Hu (2006) that provides new state trajectories at hidden times between observations. This simulation technique is adapted to a multivariate context where the drift dominates, contrary to techniques based on Brownian bridge sampling (Durham and Gallant, 2002). Moreover, it does not require constraining assumptions for multivariate models, contrary to other techniques based on the exact simulation of diffusions (Beskos and Roberts, 2005; Beskos et al., 2006). The proposed smoothing method can then be applied to high-dimensional systems. Finally, it does not require model linearization nor Gaussian hypotheses, and so is able to deal with general nonlinear models.

The remainder of the paper is organized as follows. Section 2 briefly describes sequential Monte Carlo filtering methods in state-space models, and presents the fixed-lag smoothing problem. Section 3 presents the conditional simulation technique of diffusions of Delyon and Hu (2006), and details the construction of the proposed Monte Carlo estimate of smoothing distributions. The method is then experimented on a one-dimensional example in Sect. 4. Finally, the method is applied in Sect. 5 to a practical nonlinear and high-dimensional case, similar to the problems that have to be faced in environmental applications. A discussion is given in Sect. 6.

2 Monte Carlo filtering and smoothing in state-space models

In this section we recall briefly the particle filtering and smoothing methods for models of type Eq. (1), where the hidden state vector $\boldsymbol{x} \in \mathbb{R}^n$ is observed through the observation vector $\boldsymbol{y} \in \mathbb{R}^m$ at discrete times $\{t_1, t_2, \ldots\}$:

$$\boldsymbol{y}(t_k) = g(\boldsymbol{x}(t_k)) + \boldsymbol{\gamma}_{t_k}. \tag{2}$$

The drift function f and observation operator g can be nonlinear. The dynamical model uncertainty is described by an n-dimensional Brownian motion with covariance $\boldsymbol{\Sigma} =$

$\sigma(\boldsymbol{x}(t))\sigma(\boldsymbol{x}(t))^T$. The functions f, g and σ are assumed to be known, as well as the law of the observation noise $\boldsymbol{\gamma}_{t_k}$.

In particular, we present the standard particle filter and the weighted ensemble Kalman filter, which can be used to face the filtering problem in high-dimensional systems.

2.1 Particle-based filtering methods

Filtering aims at estimating recursively the distribution $p(\boldsymbol{x}_{t_1:t_k}|\boldsymbol{y}_{t_1:t_k})$ (and in particular its marginal distribution $p(\boldsymbol{x}_{t_k}|\boldsymbol{y}_{t_1:t_k})$) at each observation time t_k. This filtering problem can be solved with a Monte Carlo sequential approach, called particle filtering (Del Moral et al., 2001; Doucet et al., 2000). The method relies on a Monte Carlo approximation of the filtering distribution over a set of weighted trajectories $\{\boldsymbol{x}^{(i)}_{t_1:t_k}\}_{i=1:N}$ (called particles):

$$\hat{p}(\boldsymbol{x}_{t_1:t_k}|\boldsymbol{y}_{t_1:t_k}) = \sum_{i=1}^{N} w^{(i)}_{t_k} \delta_{\boldsymbol{x}^{(i)}_{t_1:t_k}}(\boldsymbol{x}_{t_1:t_k}), \tag{3}$$

whose marginal distribution at time t_k is written as

$$\hat{p}(\boldsymbol{x}_{t_k}|\boldsymbol{y}_{t_1:t_k}) = \sum_{i=1}^{N} w^{(i)}_{t_k} \delta_{\boldsymbol{x}^{(i)}_{t_k}}(\boldsymbol{x}_{t_k}). \tag{4}$$

Particle filters rely on a sequential importance sampling scheme that recursively samples particles, and updates their weights at observation times. The weights correspond to the ratio between the target distribution and the importance sampling distribution $\pi(\boldsymbol{x}_{t_k}|\boldsymbol{x}_{t_0:t_{k-1}}, \boldsymbol{y}_{t_1:t_k})$. They are recursively computed as follows:

$$w^{(i)}_{t_k} \propto w^{(i)}_{t_{k-1}} \frac{p(\boldsymbol{y}_{t_k}|\boldsymbol{x}^{(i)}_{t_k}) p(\boldsymbol{x}^{(i)}_{t_k}|\boldsymbol{x}^{(i)}_{t_{k-1}})}{\pi(\boldsymbol{x}^{(i)}_{t_k}|\boldsymbol{x}^{(i)}_{t_0:t_{k-1}}, \boldsymbol{y}_{t_1:t_k})}. \tag{5}$$

In practice, a resampling procedure is added in order to avoid degeneracy. This procedure duplicates trajectories with large weights and removes small weighted trajectories.

2.1.1 Standard particle filter

When the proposal distribution π is set to the prior (i.e., $\pi(\boldsymbol{x}_{t_k}|\boldsymbol{x}_{t_0:t_{k-1}}, \boldsymbol{y}_{t_1:t_k}) = p(\boldsymbol{x}_{t_k}|\boldsymbol{x}_{t_{k-1}})$), the weight updating rule (Eq. 5) is simplified to the computation of the data likelihood $p(\boldsymbol{y}_{t_k}|\boldsymbol{x}^{(i)}_{t_k})$. This particular instance of the particle filter is called the *Bootstrap filter* or sequential importance resampling (SIR) filter (Gordon et al., 1993). Due to its simplicity it is the most commonly used particle filter. It is however a very inefficient distribution for high-dimensional space as it does not take into account the current observation and depends only weakly on the past data through the filtering distribution estimated at the previous instant. This distribution requires a great number of particles to explore meaningful areas of the state space.

2.1.2 Weighted ensemble Kalman filter (WEnKF)

One way to incorporate observation within the proposal distribution efficiently consists in relying on the ensemble Kalman filtering mechanism to define this distribution. This is the idea proposed in the WEnKF technique (Papadakis et al., 2010). In the WEnKF approach the importance sampling is taken as a Gaussian approximation of $p(\boldsymbol{x}_{t_k}|\boldsymbol{x}_{t_{k-1}}, \boldsymbol{y}_{t_k})$. This approach is close to the technique proposed in Van Leeuwen (2010). A variation of a similar technique based on a deterministic square-root formulation is also described in Beyou et al. (2013). In order to make the estimation of the filtering distribution exact (up to the sampling), each member of the ensemble must be weighted at each observation instant t_k with appropriate weights $w^{(i)}_{t_k}$, defined from Eq. (5). Therefore, the weighted ensemble Kalman filter (WEnKF) procedure can be simply summarized by Algorithm 1.

Algorithm 1 The WEnKF algorithm

For each $t_k = t_1, t_2, \ldots$:

- Start from particle set $\{\boldsymbol{x}^{(i)}_{t_{k-1}}, i = 1, \ldots, N\}$ and observation $\boldsymbol{y}_{t_k}$
- Obtain particle set $\{\boldsymbol{x}^{(i)}_{t_k}, i = 1, \ldots, N\}$ from:
  - **EnKF step:** Get $\boldsymbol{x}^{(i)}_{t_k}$, $i = 1, \ldots, N$, from the assimilation of $\boldsymbol{y}_{t_k}$ with an EnKF procedure;
  - **Computation of weights:** $w^{(i)}_{t_k} \propto w^{(i)}_{t_{k-1}} \frac{p(\boldsymbol{y}_{t_k}|\boldsymbol{x}^{(i)}_{t_k}) p(\boldsymbol{x}^{(i)}_{t_k}|\boldsymbol{x}^{(i)}_{t_{k-1}})}{p(\boldsymbol{x}^{(i)}_{t_k}|\boldsymbol{x}^{(i)}_{t_{k-1}}, \boldsymbol{y}^{(i)}_{t_k})}$;
  - **Resampling:** For $j = 1, \ldots, N$, sample with replacement index $I(j)$ from discrete probability $\{w^{(i)}_{t_k}, i = 1, \ldots, N\}$ over $\{1, \ldots, N\}$ and set $\boldsymbol{x}^{(j)}_{t_k} = \boldsymbol{x}^{I(j)}_{t_k}$. Set $w^{(i)}_{t_k} = \frac{1}{N} \quad \forall i = 1, \ldots, N$.

Note that particle-based filtering techniques update the filtering distribution at observation times only. However, after the estimate $\hat{p}(\boldsymbol{x}_{t_k}|\boldsymbol{y}_{t_1:t_k})$ has been updated at observation time t_k, the filtering distribution can be predicted in order to have a continuous estimation of $\hat{p}(\boldsymbol{x}_t|\boldsymbol{y}_{t_1:t_k})$ for all $t \in]t_k, t_{k+1}[$ until the next observation time

$$\hat{p}(\boldsymbol{x}_t|\boldsymbol{y}_{t_1:t_k}) = \sum_{i=1}^{N} w^{(i)}_{t_k} \delta_{\boldsymbol{x}^{(i)}_t}(\boldsymbol{x}_t), \tag{6}$$

where, for all $i = 1, \ldots, N$, the state $\boldsymbol{x}^{(i)}_t$ is sampled from Eq. (1), starting from $\boldsymbol{x}^{(i)}_{t_k}$.

2.2 Fixed-lag smoothing problem

Contrary to the filtering approach that uses past and present observations, the smoothing in state-space models aims at estimating $p(\boldsymbol{x}_t|\boldsymbol{y}_{t_1:t_{\text{end}}})$ for all $t \in [t_1, t_{\text{end}}]$, using all past and future observations over a given time period. As raised in the introduction, existing smoothing methods do not apply directly to a general nonlinear model of type Eq. (1) in a high-dimensional context, since assumptions have to be made that may not be realistic. Instead of solving the global smoothing, we will concentrate in the rest of the paper on a fixed-lag smoothing, which constitutes a partial answer to the global smoothing problem.

The objective of the fixed-lag smoothing will be to replace the predictive distribution (Eq. 6) by its smoothed version $p(\boldsymbol{x}_t|\boldsymbol{y}_{t_1:t_{k+1}})\ \forall t \in]t_k, t_{k+1}]$ sequentially each time a new observation $\boldsymbol{y}_{t_{k+1}}$ arrives. This will allow us to reduce the temporal discontinuities inherent in the filtering technique that successively predicts the distribution of the state between observations, and updates this distribution at observation times.

To achieve this, by construction of the particle filter that weights entire trajectories (see Eq. 3), it is known (see for instance Doucet et al., 2000) that the fixed-lag smoothing distribution $\hat{p}(\boldsymbol{x}_t|\boldsymbol{y}_{t_1:t_{k+1}})$ can be directly obtained from the marginal at time t of $\hat{p}(\boldsymbol{x}_{t_1:t_{k+1}}|\boldsymbol{y}_{t_1:t_{k+1}})$. The empirical smoothing distribution is then given by

$$\hat{p}(\boldsymbol{x}_t|\boldsymbol{y}_{t_1:t_{k+1}}) = \sum_{i=1}^{N} w_{t_{k+1}}^{(i)} \delta_{\boldsymbol{x}_t^{(i)}}(\boldsymbol{x}_t) \quad \forall t \in]t_k, t_{k+1}]. \tag{7}$$

However, this approximation is simply a reweighting of past existing particle trajectories, and relies on the support of the filtering distribution at time t_k. If the number of particles is too small with respect to the state dimension, the support may be greatly reduced by the correction step (assigning small weights to all particles except a few), leading in practice to a bad estimation of $p(\boldsymbol{x}_t|\boldsymbol{y}_{t_1:t_{k+1}})$. Moreover, if particle trajectories have been forced towards observations during the filtering step (like in the WEnKF procedure), a smoothing based on those particles will fail because it will not be able to correct discontinuities. Consequently, since we are interested in smoothing techniques that are efficient in a high-dimensional context, this direct smoothing technique can not be used in its basic form and has to be improved.

In the following, we propose to use a conditional simulation technique of diffusions that will enable the sampling of new smoothed trajectories between times t_k and t_{k+1}. The approximation of the smoothing distribution (Eq. 7) at each hidden time will then be improved. The conditional simulation technique is presented in the next section, before the resulting smoothing procedure we propose.

3 Fixed-lag smoothing with conditional simulation

The smoothing method we propose is based on a conditional simulation technique that is presented in Sect. 3.1. We then develop in Sect. 3.2 how this technique can be used to improve the estimation of the smoothing distribution (Eq. 7).

3.1 Conditional simulation

Conditional simulation of a diffusion aims at sampling trajectories from a given process

$$\mathrm{d}\boldsymbol{x}(t) = f(\boldsymbol{x}(t))\mathrm{d}t + \sigma(\boldsymbol{x}(t))\mathrm{d}\boldsymbol{B}(t) \tag{8}$$

between two times $t = 0$ and $t = T$, with the constraints $\boldsymbol{x}(0) = \boldsymbol{u}$ and $\boldsymbol{x}(T) = \boldsymbol{v}$. This simulation problem is treated by Delyon and Hu (2006), where the authors show how to obtain the law of the constrained process from a Girsanov theorem. In practice, the proposed algorithms consist in simulating trajectories according to another diffusion process, which is built to respect the constraints and is easy to simulate from. The conditional distribution of the constrained process (Eq. 8) is absolutely continuous with respect to the distribution of the auxiliary process, with explicitly given density. For instance, in the case where the drift is bounded (a similar algorithm is proposed in Delyon and Hu (2006) for the unbounded case) and with σ invertible, the algorithm is based on the simulation of trajectories from the following process:

$$\mathrm{d}\tilde{\boldsymbol{x}}(t) = \left(f(\tilde{\boldsymbol{x}}(t)) - \frac{\tilde{\boldsymbol{x}}(t) - \boldsymbol{v}}{T - t} \right) \mathrm{d}t + \sigma(\tilde{\boldsymbol{x}}(t))\mathrm{d}\boldsymbol{B}(t), \tag{9}$$

with initial condition $\tilde{\boldsymbol{x}}(0) = \boldsymbol{u}$. Note that Lemma 4 in Delyon and Hu (2006) deals with the existence of a unique solution for this equation. This process is a simple modification of Eq. (8), where a deterministic part is added to the drift. It is then easy to simulate unconditional trajectories from this process, and all simulated trajectories will satisfy $\tilde{\boldsymbol{x}}(T) = \boldsymbol{v}$ by construction. For simplicity we will assume in the following that σ is independent of $\boldsymbol{x}(t)$ (note however that this is not an assumption in Delyon and Hu (2006)). The law of the conditioned process is given by

$$\mathbb{E}[h(\boldsymbol{x})|\boldsymbol{x}(0) = \boldsymbol{u}, \boldsymbol{x}(T) = \boldsymbol{v}] = \mathbb{E}\left[h(\tilde{\boldsymbol{x}})\alpha(\tilde{\boldsymbol{x}})\right], \tag{10}$$

for all measurable functions h, where

$$\alpha(\tilde{\boldsymbol{x}}) = \exp\left(-\int_0^T \frac{(\tilde{\boldsymbol{x}}(t) - \boldsymbol{v})^T \boldsymbol{\Sigma}^{-1} f(\tilde{\boldsymbol{x}}(t))}{T - t} \mathrm{d}t \right) \tag{11}$$

is the density coming from the Girsanov theorem (see Delyon and Hu, 2006), with $\boldsymbol{\Sigma} = \sigma(\tilde{\boldsymbol{x}}(t))\sigma(\tilde{\boldsymbol{x}}(t))^T$.

Let us note that the presence of the drift part of model (Eq. 8) in the auxiliary process (Eq. 9) is crucial to making the simulation efficient. The same process had initially been

proposed by Clark (1990) to solve the conditional simulation problem. On the other hand, standard Brownian bridges that could be used as auxiliary processes (Durham and Gallant, 2002) lead in practice to poor approximations of the original constrained diffusion in our high-dimensional setting, since Brownian bridge trajectories are too far away from trajectories of Eq. (8).

In the following, the conditional marginal of interest $p(\boldsymbol{x}_t|\boldsymbol{x}(0)=\boldsymbol{u},\boldsymbol{x}(T)=\boldsymbol{v})$ will then be approximated as follows:

$$\hat{p}(\boldsymbol{x}_t|\boldsymbol{x}(0)=\boldsymbol{u},\boldsymbol{x}(T)=\boldsymbol{v})=\sum_{j=1}^{M}\alpha(\tilde{\boldsymbol{x}}^{(j)})\delta_{\tilde{\boldsymbol{x}}_t^{(j)}}(\boldsymbol{x}_t) \quad \forall t\in[0,T], \tag{12}$$

where the M trajectories $\{\tilde{\boldsymbol{x}}_t^{(j)}\}_{j=1:M}$ are simulated from Eq. (9) with $\tilde{\boldsymbol{x}}_0^{(j)}=\boldsymbol{u}$ for all $j=1,\ldots,M$.

3.2 Proposed fixed-lag smoothing method

We show in the following how the conditional simulation technique can be used to improve the estimation of the local smoothing distribution $p(\boldsymbol{x}_t|\boldsymbol{y}_{t_1:t_{k+1}})$ for all $t\in]t_k,t_{k+1}]$.

We first note that this distribution can be decomposed as

$$\begin{aligned}p(\boldsymbol{x}_t|\boldsymbol{y}_{t_1:t_{k+1}})&=\int p(\boldsymbol{x}_t,\boldsymbol{x}_{t_k},\boldsymbol{x}_{t_{k+1}}|\boldsymbol{y}_{t_1:t_{k+1}})\mathrm{d}\boldsymbol{x}_{t_k}\mathrm{d}\boldsymbol{x}_{t_{k+1}}\\&=\int p(\boldsymbol{x}_{t_k},\boldsymbol{x}_{t_{k+1}}|\boldsymbol{y}_{t_1:t_{k+1}})p(\boldsymbol{x}_t|\boldsymbol{x}_{t_k},\boldsymbol{x}_{t_{k+1}},\boldsymbol{y}_{t_1:t_{k+1}})\\&\quad\mathrm{d}\boldsymbol{x}_{t_k}\mathrm{d}\boldsymbol{x}_{t_{k+1}}.\end{aligned} \tag{13}$$

Then, from the state-space model properties, we obtain

$$\begin{aligned}p(\boldsymbol{x}_t|\boldsymbol{y}_{t_1:t_{k+1}})&=\int p(\boldsymbol{x}_{t_k},\boldsymbol{x}_{t_{k+1}}|\boldsymbol{y}_{t_1:t_{k+1}})p(\boldsymbol{x}_t|\boldsymbol{x}_{t_k},\boldsymbol{x}_{t_{k+1}})\\&\quad\mathrm{d}\boldsymbol{x}_{t_k}\mathrm{d}\boldsymbol{x}_{t_{k+1}}.\end{aligned} \tag{14}$$

Moreover, from the particle filter Monte Carlo approximation described by Eq. (3), the joint law $p(\boldsymbol{x}_{t_k},\boldsymbol{x}_{t_{k+1}}|\boldsymbol{y}_{t_1:t_{k+1}})$ can be replaced by

$$\hat{p}(\boldsymbol{x}_{t_k},\boldsymbol{x}_{t_{k+1}}|\boldsymbol{y}_{t_1:t_{k+1}})=\sum_{i=1}^{N}w_{t_{k+1}}^{(i)}\delta_{(\boldsymbol{x}_{t_{k+1}}^{(i)},\boldsymbol{x}_{t_k}^{(i)})}(\boldsymbol{x}_{t_{k+1}},\boldsymbol{x}_{t_k}), \tag{15}$$

where $w_{t_{k+1}}^{(i)}$ are the particle filter importance weights.

Plugging Eq. (15) into Eq. (14) then leads to the following approximation for the fixed-lag smoothing distribution:

$$\hat{p}(\boldsymbol{x}_t|\boldsymbol{y}_{t_1:t_{k+1}})=\sum_{i=1}^{N}w_{t_{k+1}}^{(i)}p(\boldsymbol{x}_t|\boldsymbol{x}_{t_k}^{(i)},\boldsymbol{x}_{t_{k+1}}^{(i)}). \tag{16}$$

The conditional distribution $p(\boldsymbol{x}_t|\boldsymbol{x}_{t_k}^{(i)},\boldsymbol{x}_{t_{k+1}}^{(i)})$ can be estimated using Eq. (12) for each pair of initial and end points $\boldsymbol{x}_{t_k}^{(i)}$ and $\boldsymbol{x}_{t_{k+1}}^{(i)}$:

$$\hat{p}(\boldsymbol{x}_t|\boldsymbol{x}_{t_k}^{(i)},\boldsymbol{x}_{t_{k+1}}^{(i)})=\sum_{j=1}^{M}\alpha(\tilde{\boldsymbol{x}}^{(i)(j)})\delta_{\tilde{\boldsymbol{x}}_t^{(i)(j)}}(\boldsymbol{x}_t), \tag{17}$$

where each $\tilde{\boldsymbol{x}}_t^{(i)(j)}$ is sampled from Eq. (9) with initial constraint $\tilde{\boldsymbol{x}}_{t_k}^{(i)(j)}=\boldsymbol{x}_{t_k}^{(i)}$ and final constraint $\boldsymbol{x}_{t_{k+1}}^{(i)}$.

The estimation of the smoothing distribution of interest is finally written as

$$\hat{p}(\boldsymbol{x}_t|\boldsymbol{y}_{t_1:t_{k+1}})=\sum_{i=1}^{N}w_{t_{k+1}}^{(i)}\sum_{j=1}^{M}\alpha(\tilde{\boldsymbol{x}}^{(i)(j)})\delta_{\tilde{\boldsymbol{x}}_t^{(i)(j)}}(\boldsymbol{x}_t), \quad \forall t\in]t_k,t_{k+1}]. \tag{18}$$

The algorithm we propose to compute the fixed-lag smoothing distribution on a given time interval $[t_k,t_{k+1}]$ is therefore the following:

Algorithm 2 Fixed-lag conditional smoothing

For each $t_k=t_1,t_2,\ldots$:

- Store $\{\boldsymbol{x}_{t_k}^{(i)}\}_{i=1:N}$ and compute $\{\boldsymbol{x}_{t_{k+1}}^{(i)}\}_{i=1:N}$ and associated weights $\{w_{t_{k+1}}^{(i)}\}_{i=1:N}$ from a particle filter algorithm;
- For each pair $\{\boldsymbol{x}_{t_k}^{(i)},\boldsymbol{x}_{t_{k+1}}^{(i)}\}$, $i=1,\ldots,N$:
  - Simulate M conditional trajectories $\{\tilde{\boldsymbol{x}}_t^{(i)(j)}\}_{j=1:M}$ for $t\in[t_k,t_{k+1}]$ from Eq. (9) with an Euler scheme, with the constraints $\tilde{\boldsymbol{x}}_{t_k}^{(i)(j)}=\boldsymbol{x}_{t_k}^{(i)}$ and $\tilde{\boldsymbol{x}}_{t_{k+1}}^{(i)(j)}=\boldsymbol{x}_{t_{k+1}}^{(i)}$,
  - Compute weights $\alpha(\tilde{\boldsymbol{x}}^{(i)(j)})$ from Eq. (11) for all $j=1,\ldots,M$, with final constraint $\boldsymbol{x}_{t_{k+1}}^{(i)}$;
- Compute $\hat{p}(\boldsymbol{x}_t|\boldsymbol{y}_{t_1:t_{k+1}})=\sum_{i=1}^{N}w_{t_{k+1}}^{(i)}\sum_{j=1}^{M}\alpha(\tilde{\boldsymbol{x}}^{(i)(j)})\delta_{\tilde{\boldsymbol{x}}^{(i)(j)}}(\boldsymbol{x}_t)$ for all $t\in]t_k,t_{k+1}]$.

4 One-dimensional simulation study

In this section, the smoothing method is first experimented on a one-dimensional state space model. Since the proposed approach relies on a preliminary particle filtering step, filtering results are first presented in Sect. 4.2 (either considering a standard particle filter or the WEnKF). The results obtained with the standard fixed-lag smoothing method are then shown in Sect. 4.3. Finally, Sect. 4.4 presents the smoothing results obtained with the proposed technique.

4.1 State-space model

The one-dimensional state-space model of interest is a sine diffusion, partially observed with noise (used as an illustration by Fearnhead et al. (2008) for a particle filtering method):

$$\mathrm{d}x(t) = \sin(x(t))\mathrm{d}t+\sigma_x\mathrm{d}B(t), \tag{19}$$

$$y_{t_k} = x_{t_k}+\gamma_{t_k}, \tag{20}$$

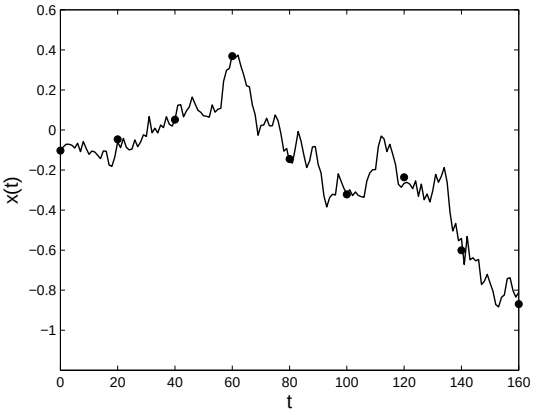


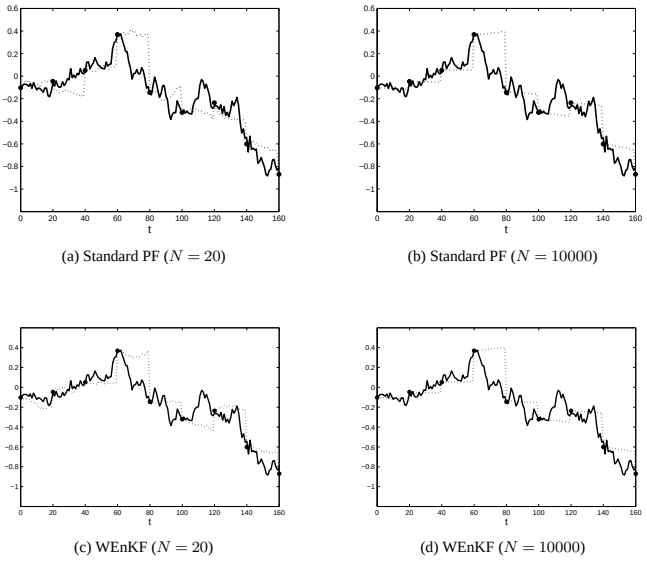
Figure 1. Simulated sine diffusion trajectory $x(t)$ and partial observations $y(t_k)$ (dots) with $t_k - t_{k-1} = 20\Delta t$.

where $\sigma_x^2 = 0.5$ and $\gamma_{t_k} \sim \mathcal{N}(0, \sigma_y)$ with $\sigma_y^2 = 0.01$. One trajectory of the process is first simulated from Eq. (19) with an Euler-type discretization scheme of time step $\Delta t = 0.005$. This trajectory will constitute the hidden process, observed through y_{t_k} generated according to Eq. (20) at every time step t_k, with $t_k - t_{k-1} = 20\Delta t$. The trajectory is plotted in Fig. 1, together with the corresponding discrete observations at times t_k.

4.2 Particle filtering results

The filtering results are presented for the standard particle filter (denoted PF in the following) and the weighted ensemble Kalman filter (WEnKF). Two situations are shown, with a reduced ($N = 20$) and high number ($N = 10\,000$) of particles. The case with a high number of particles is shown as the reference for comparison purpose; note however that this ideal situation is not reachable in a high-dimensional context, since the number of particles has to be reduced for evident computational cost reasons.

The results for the two configurations are presented in Fig. 2, where the dotted lines represent the filtering mean estimates. The filtering distribution $p(x_{t_k}|y_{t_1:t_k})$ is estimated at each observation time t_k using Eq. (4), and predicted between observation times from Eq. (6). The mean is then estimated from weighted particles as $\sum_{i=1}^{N} w_{t_k}^{(i)} x_t^{(i)}$, for all $t \in [t_k, t_{k+1}[$. Figure 2a–b shows that the standard particle filter results diverge from the reference solution between observation times, for low or high number of particles. As a matter of fact, when no observation is available, the state distribution is predicted from the dynamics only, so that particles trajectories are not guided towards the next observation. At observation times t_k, high weights are given to particles that are close to the observation, so that the estimated mean suddenly gets closer to the solution. These discontinuities between measurement times can also be observed in the WEnKF results (Fig. 2c–d), because particle trajectories are



(a) Standard PF ($N = 20$) (b) Standard PF ($N = 10000$)

(c) WEnKF ($N = 20$) (d) WEnKF ($N = 10000$)

Figure 2. Standard PF and WEnKF results. Thick line: hidden diffusion; dots: partial observations; dotted line: estimated filtering mean.

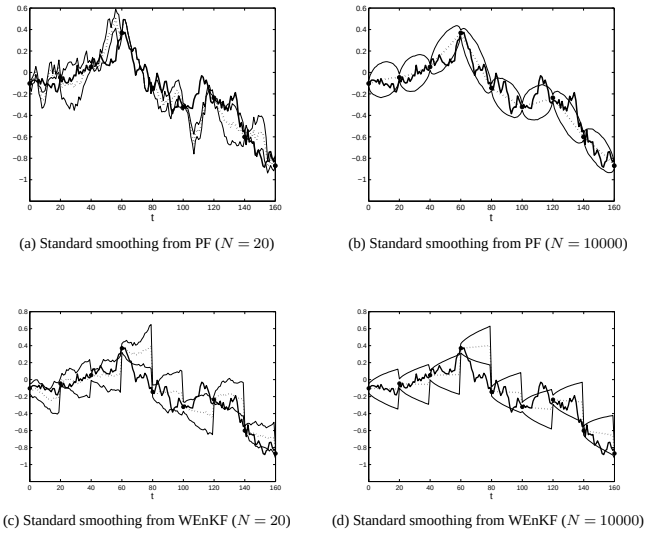
brutally corrected with the EnKF step at observation times. A smoothing will aim at reducing these temporal discontinuities, while providing dynamically consistent solutions.

4.3 Standard fixed-lag smoothing results

From the particle filtering results, we now present the results obtained with the direct particle smoothing procedure described in Sect. 2.2. This procedure relies on existing trajectories. The smoothing distribution $\hat{p}(x_t|y_{t_1:t_{k+1}})$ is computed backward for all $t \in]t_k, t_{k+1}]$ using expression Eq. (7) each time a new observation $y_{t_{k+1}}$ becomes available. The smoothing mean is computed as $\sum_{i=1}^{N} w_{t_{k+1}}^{(i)} x_t^{(i)}$ for all $t \in]t_k, t_{k+1}]$, and the standard deviation is computed in the same way from the weighted particles.

It can be observed in Fig. 3a that the smoothing based on the standard particle filter is not efficient when the number of particles N is small: only a few particles are close to the observation at time t_k and have nonzero weights, implying that the smoothing distribution is poorly estimated (see for instance between observation times $t = 100$ and $t = 120$, where the smoothing distribution is artificially peaked but far from the hidden trajectory). The smoothing result obtained from the reference configuration $N = 10\,000$ is plotted in Fig. 3b. In that situation, since many trajectories have high weights at observation times, the estimation of backward smoothing distributions is improved and includes the hidden trajectory.

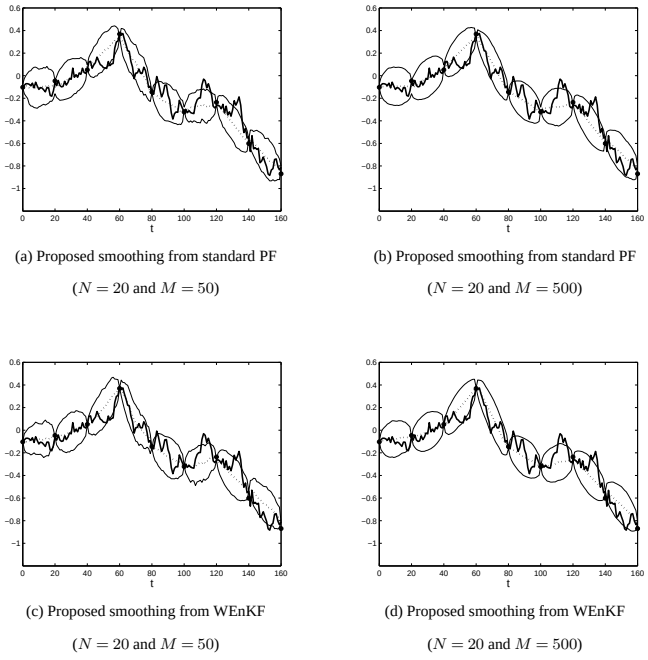
Moreover, Fig. 3c–d shows that the standard smoothing based on the WEnKF result fails for a low or high number of particles. As a matter of fact, particle trajectories are artificially corrected by the EnKF step at each observation time.



(a) Standard smoothing from PF ($N = 20$) (b) Standard smoothing from PF ($N = 10000$)

(c) Standard smoothing from WEnKF ($N = 20$) (d) Standard smoothing from WEnKF ($N = 10000$)

Figure 3. Standard smoothing from PF and WEnKF results. Thick line: hidden diffusion; dots: partial observations; dotted line: estimated filtering mean.



(a) Proposed smoothing from standard PF ($N = 20$ and $M = 50$) (b) Proposed smoothing from standard PF ($N = 20$ and $M = 500$)

(c) Proposed smoothing from WEnKF ($N = 20$ and $M = 50$) (d) Proposed smoothing from WEnKF ($N = 20$ and $M = 500$)

Figure 4. Proposed conditional smoothing result. Thick line: hidden diffusion; dots: partial observations; dotted line: estimated backward smoothing mean; thin line: estimated standard deviation.

Resulting trajectories are highly non-plausible. Even for a huge number of particles, a smoothing based on those existing trajectories is not able to reduce the induced time discontinuities.

4.4 Proposed smoothing results

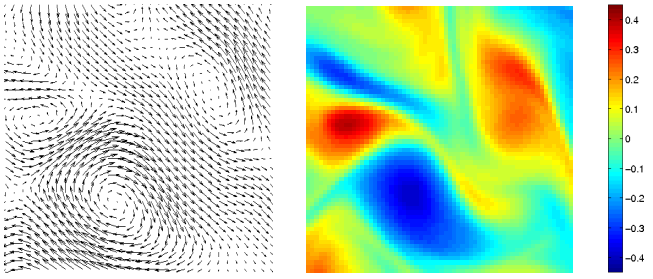
In this section, we show how the proposed method can improve the estimation of backward smoothing distributions when it is not adequate to rely on existing trajectories only. This is the case if the number of particles is too small, as demonstrated from the experiment presented in Fig. 3a, or if the existing trajectories do not correspond to plausible trajectories of the model (as shown for the WEnKF result in Fig. 3c–d).

Our smoothing is first applied using the filtering output of the standard particle filter with $N = 20$ particles. Figure 4a shows the result obtained with a sampling of $M = 50$ conditional trajectories between each pair $\{x_{t_k}^{(i)}, x_{t_{k+1}}^{(i)}\}$, $i = 1, \ldots, N$. The smoothing distribution $\hat{p}(x_t | y_{t_1:t_{k+1}})$ is computed from Eq. (18), so the smoothing mean is computed as $\sum_{i=1}^{N} w_{t_k}^{(i)} \sum_{j=1}^{M} \alpha(\tilde{x}^{(i)(j)}) \tilde{x}_t^{(i)(j)}$ for all $t \in]t_k, t_{k+1}]$, and similarly for the standard deviation. This result highlights the fact that since the proposed method creates new trajectories, it is able to correct the deficiencies of the standard smoothing approach presented in Fig. 3a when the initial number of filtering particles is too small. In Fig. 4b, the same experiment is presented using $M = 500$ conditional trajectories. In that case, the result is very similar to the reference particle smoothing result presented in Fig. 3b, obtained from a particle filter with $N = 10\,000$.

In parallel, the proposed smoothing has been tested using the output of the WEnKF filtering technique with $N = 20$ particles. Again, the smoothing is computed with $M = 50$ and $M = 500$ conditional trajectories, and the corresponding results are presented in Fig. 3c–d. Instead on relying on existing WEnKF trajectories that may not be plausible trajectories of the model (because of the EnKF correction step), the proposed method samples new trajectories between observation times. This leads to a good estimation of the smoothing distributions, contrary to the standard smoothing presented in Fig. 3c. Note that the smoothing results are very similar to the result obtained from the standard particle filter (Fig. 3a–b) because both filters have similar behavior at observation times.

5 Application to a high-dimensional assimilation problem

This section aims at illustrating the applicability of our method to a high-dimensional and nonlinear scenario, without extensive study at this stage. The method is applied to a turbulence assimilation problem, where the model of interest is of type Eq. (1). The goal is to recover temporal estimates of velocity/vorticity over a given spatial domain of size $n = 64 * 64$, from a sequence of noisy observations and a continuous a priori dynamical model based on a stochastic version of the Navier–Stokes equation. Within an environmental framework, a direct application would be the estimation of wind fields or sea surface currents from satellite data.



(a) (b)

Figure 5. State example. **(a)** Velocity field $\mathbf{w}_t$; **(b)** associated vorticity map $\boldsymbol{\xi}_t$.

5.1 State-space model

Let $\xi(\boldsymbol{x})$ denote the scalar vorticity at point $\boldsymbol{x} = (x, y)^T$, associated with the 2-D velocity $\boldsymbol{w}(\boldsymbol{x}) = (w_x(\boldsymbol{x}), w_y(\boldsymbol{x}))^T$ through $\xi(\boldsymbol{x}) = \frac{\partial w_y}{\partial x} - \frac{\partial w_x}{\partial y}$. Let $\boldsymbol{\xi} \in \mathbb{R}^n$ be the state vector describing the vorticity over an $n = 64 * 64$ square domain, and $\boldsymbol{w} \in \mathbb{R}^{2n}$ the associated velocity field over the domain. We will focus on incompressible flows such that the divergence of the velocity field is null. A stochastic version of the Navier–Stokes equation in its velocity–vorticity form can then be written as

$$\mathrm{d}\boldsymbol{\xi}_t = -\nabla\boldsymbol{\xi}_t \cdot \boldsymbol{w}_t \mathrm{d}t + \frac{1}{Re}\Delta\boldsymbol{\xi}_t \mathrm{d}t + \sigma \mathrm{d}\boldsymbol{B}_t, \tag{21}$$

where Re denotes the flow Reynolds number ($Re = 3000$). The uncertainty is modeled by a Brownian motion of size n, with covariance $\boldsymbol{\Sigma} = \sigma\sigma^T$, where $\sigma \in \mathbb{R}^n$. A velocity field example, generated from the model Eq. (21), is shown in Fig. 5a, together with the corresponding vorticity map (b).

We assume the hidden vorticity vector $\boldsymbol{\xi}$ is observed through noisy measurements $\boldsymbol{y}_{t_k}$ at discrete times t_k, where $t_k - t_{k-1} = 100\Delta t$, and $\Delta t = 0.1$ is the time step used to discretize Eq. (21). In our experimental setup, measurements correspond to PIV (particle image velocimetry) image sequences used in fluid mechanics applications. Note however that other kind of data can be used similarly within this state space model, like meteorological or oceanographic data for instance. The state and observation are related in our case through $\boldsymbol{y}_{t_k} = g(\boldsymbol{\xi}_{t_k}) + \boldsymbol{\gamma}_{t_k}$, where g is a nonlinear function linking the vorticity to the image data, and $\boldsymbol{\gamma}_{t_k}$ is a Gaussian noise, uncorrelated in time.

5.2 Implementation details

We recall that the smoothing relies first on a particle filter step. Due to the high dimensionality of the state vector, the use of a standard particle filter is not adapted to solve the filtering problem, as discussed by Snyder et al. (2008) or Van Leeuwen (2009). We then make use of the method presented by Papadakis et al. (2010), which combines the benefits of the ensemble Kalman filter, known to perform well in practice for high-dimensional systems (Stroud et al., 2010), and the particle filter (which solves theoretically the true filtering problem, without approximating the filtering distributions with Gaussian distributions). Since the method of Papadakis et al. (2010) is intrinsically a particle filter, it then leads at each observation time t_k to a set of particles and weights $\{\boldsymbol{\xi}^{(i)}_{t_1:t_k}, w_{t_k}\}_{i=1:N}$, as required by the algorithm proposed in Sect. 3.

The particle filter step requires simulations from the dynamical model (Eq. 21), and the conditional simulation step requires us to sample trajectories from its constrained version, which consists in a similar problem with modified drift (see process Eq. 9). The model is discretized in time with the time step $\Delta t = 0.1$; more information about the discretization scheme may be obtained in Papadakis et al. (2010). The random perturbations are assumed to be realizations of Gaussian random fields that are correlated in space with the Gaussian covariance function $\boldsymbol{\Sigma}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \eta \exp(-\frac{||\boldsymbol{x}_i - \boldsymbol{x}_j||^2}{\lambda})$, where $\eta = 0.01$ and $\lambda = 13$. In practice, the simulation of these perturbations is performed in Fourier space, with the method described in Evensen (2003).

Finally, the estimation of the smoothing distributions requires the computation of conditional trajectories weights, corresponding to Girsanov weights given by Eq. (11). After a Riemann sum approximation of the integral, the computation of weights requires the inversion of the matrix $\boldsymbol{\Sigma}$ of size (n, n), where $n = 64 * 64$ is the number of grid points. We choose to compute $\boldsymbol{\Sigma}^{-1}$ empirically using a singular value decomposition computed from the M realizations of the perturbation fields used for the constrained trajectories simulations. Let $\mathbf{Z}$ be the matrix of size (n, M) containing the M-centered fields of size $n = 64 * 64$; the SVD leads to $\mathbf{Z} = \mathbf{U}\mathbf{D}\mathbf{V}^T$, so that $\mathbf{Z}\mathbf{Z}^T = \mathbf{U}\mathbf{D}\mathbf{D}^T\mathbf{U}^T$. The inverse of the covariance matrix $\boldsymbol{\Sigma}^{-1}$ is finally computed as

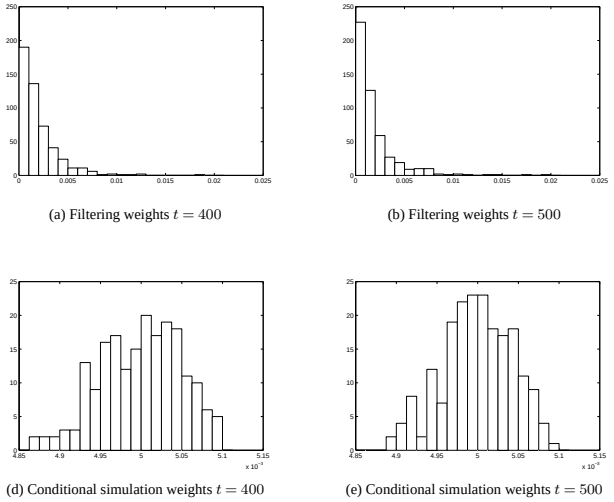
$$M(\mathbf{Z}\mathbf{Z}^T)^{-1} = M\mathbf{U}(\mathbf{D}\mathbf{D}^T)^{-1}\mathbf{U}^T, \tag{22}$$

which only requires the inversion of a diagonal. Note that more efficient procedures could be implemented in our case (homogeneous Gaussian covariance) since the covariance function is separable in the x and y directions. This means that the covariance matrix $\boldsymbol{\Sigma}$ can be written as the Kronecker product of smaller matrices and more easily inverted (Sun et al., 2012). However, the SVD inversion can be applied to any covariance structure, in particular it could deal with a non-homogeneous covariance matrix.

5.3 Results

In this section, we illustrate the capability of the proposed method to reduce the temporal discontinuities inherently introduced by the filtering in continuous–discrete state-space models.

The smoothing result relies on the output of the WEnKF filtering step, computed with $N = 500$ particles. Compared



(a) Filtering weights $t = 400$ (b) Filtering weights $t = 500$

(d) Conditional simulation weights $t = 400$ (e) Conditional simulation weights $t = 500$

Figure 6. Filtering and conditional simulation weights.

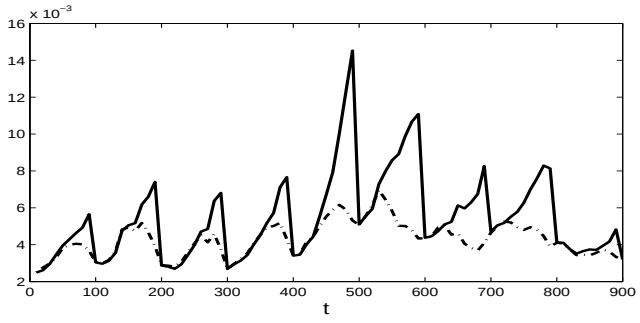


Figure 7. Full line: mean square error between ground truth vorticity and estimated filtering mean; dotted line: mean square error between ground truth vorticity and estimated backward smoothing mean.

to the size of the system, the number of particles is theoretically too small for the filter to be truly efficient. In practice, many filtering trajectories have close to zero weight at observation times. Histograms of filtering weights are given as illustration in Fig. 6a–b at two times $t = 400$ and $t = 500$. Note however that the filter is not degenerate and is able to provide results that get close to the hidden vorticity at measurement times. This can be observed in Fig. 7, where the mean square error is plotted with a full line, averaged at each time over the image domain of size $n = 64 * 64$. Since the ground truth vorticity sequence is known in our experimental setup, the mean square error is computed between the hidden vorticity and the estimated filtering mean, given by $\sum_{i=1}^{N} w_{t_k}^{(i)} \boldsymbol{\xi}_t^{(i)}$ for all $t \in [t_k, t_{k+1}[$. The correction steps lead to successive error decreases at observation times.

The proposed smoothing method has been applied with $M = 200$. Note that we take benefit from the fact that many filterings particles have close to zero weight. Indeed, the smoothing method relies in practice on a reduced number $\tilde{N}M$ of sampled conditional trajectories (with $\tilde{N} << N$), which makes the problem computationally tractable. In this experiment, we have retained around 5 % of initial filtering trajectories. The smoothing distribution $\hat{p}(\boldsymbol{\xi}_t | \boldsymbol{y}_{t_1:t_{k+1}})$ is computed for all $t \in]t_k, t_{k+1}]$ from Eq. (18), and its mean is computed as $\sum_{i=1}^{N} w_{t_{k+1}}^{(i)} \sum_{j=1}^{M} \alpha(\tilde{\boldsymbol{\xi}}^{(i)(j)}) \tilde{\boldsymbol{\xi}}_t^{(i)(j)}$. Histograms of conditional simulation weights $\alpha(\tilde{\boldsymbol{\xi}}^{(i)(j)})$ are given as an illustration in Fig. 6c–d for a given particle (i) at two times, $t = 400$ and $t = 500$.

The mean square error is computed between the true vorticity and the estimated smoothing mean, and plotted in Fig. 7 with a dotted line. As expected, the smoothing method reduces the error at hidden times between observations.

In addition, we present below a qualitative evaluation of the smoothing result for the same experiment, over a specific time interval.

The WEnKF result is first presented in Fig. 8 for the time interval $[400, 500]$ between two observations, where estimated mean vorticity maps are computed as $\sum_{i=1}^{N} w_{400}^{(i)} \boldsymbol{\xi}_t^{(i)}$ for all $t \in [400, 500[$, and as $\sum_{i=1}^{N} w_{500}^{(i)} \boldsymbol{\xi}_t^{(i)}$ for $t = 500$. The temporal discontinuity between estimations can be observed when reaching observation time $t = 500$: the vorticity map is suddenly modified in order to fit the observations, introducing inconsistencies in the vorticity temporal trajectories. Note that the application of the standard particle smoothing (described in Sect. 2.2) will fail here, and not only because the number of particles is too small. As a matter of fact, we recall that the filtering trajectories have been computed from the method presented in Papadakis et al. (2010), which uses the ensemble Kalman filter step as an importance distribution in the particle filter algorithm. The ensemble Kalman filter consists of a prediction step from the dynamical model Eq. (21), and a correction step that shifts particles towards the observation. Because of this correction step, the sampled filtering trajectories between two observation times do not correspond to trajectories of the dynamical model. This implies that from such a particle filter, the standard smoothing based on existing trajectories will not be able to reduce the temporal discontinuities observed in Fig. 8. This can be observed in Fig. 9, where smoothed vorticity maps are computed as $\sum_{i=1}^{N} w_{400}^{(i)} \boldsymbol{\xi}_t^{(i)}$ for $t = 400$, and as $\sum_{i=1}^{N} w_{500}^{(i)} \boldsymbol{\xi}_t^{(i)}$ for all $t \in]400, 500]$. The discontinuity at time $t = 500$ is still present.

The result obtained with the proposed method is plotted in Fig. 10. Estimated mean vorticity maps are computed as $\sum_{i=1}^{N} w_{500}^{(i)} \sum_{j=1}^{M} \alpha(\tilde{\boldsymbol{\xi}}^{(i)(j)}) \tilde{\boldsymbol{\xi}}_t^{(i)(j)}$ for all $t \in [400, 500]$. Spatio–temporal vorticity trajectories are gradually modified until observation time $t = 500$, preserving the fluid flow properties. As a matter of fact, since the proposed method samples new trajectories from the law of the physical process (Eq. 21), the smoothed vorticity trajectories are by construction consistent with the a priori dynamical model. In order to sample the smoothed trajectories, the method relies on the model and on filtering marginals at observation times,

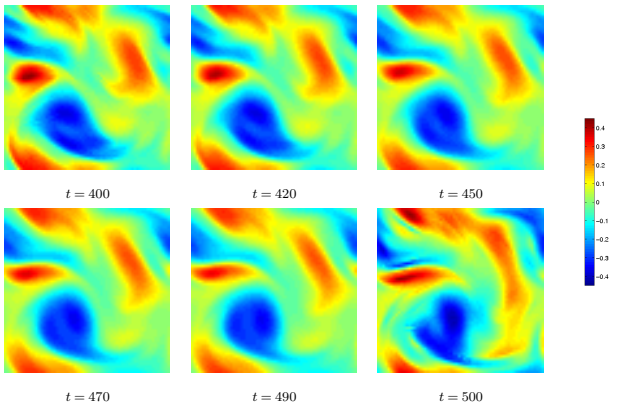


Figure 8. Filtering result with the method of Papadakis et al. (2010). Estimated mean vorticity maps for different times t between observation times $t = 400$ and $t = 500$.

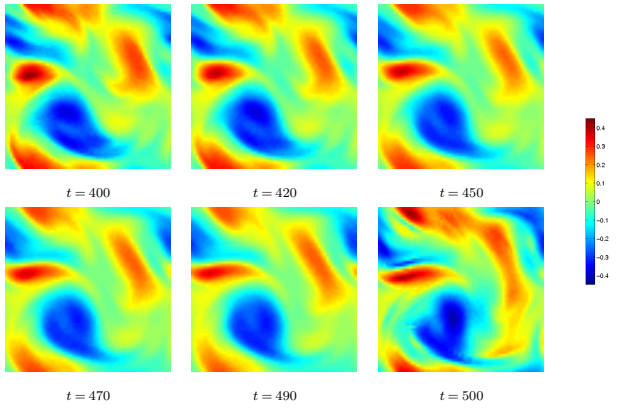


Figure 9. Standard particles smoothing result (see Sect. 2.2). Estimated mean vorticity maps for different times t between observation times $t = 400$ and $t = 500$.

but not on filtering trajectories at hidden times. It is then able to smooth the discontinuities inherent in the particle filtering technique we have used, contrary to the standard smoothing presented in Fig. 9.

6 Conclusion and discussion

In this paper we introduced a smoothing algorithm based on a conditional simulation technique of diffusions. The proposed smoothing is formulated as fixed-lag, in the sense that it is performed sequentially each time a new observation appears, in order to correct the state at hidden times up to the previous observation. Note that a decomposition similar to Eqs. (13) to (18) can be written from an integration up to a previous time t_{k-h}, with $h > 1$. This implies that the smoother can be formulated with a larger fixed lag, in order to correct the state backward not only up to the previous observation, but up to further measurement times. Yet, due to the successive resampling steps that have been performed in the filtering

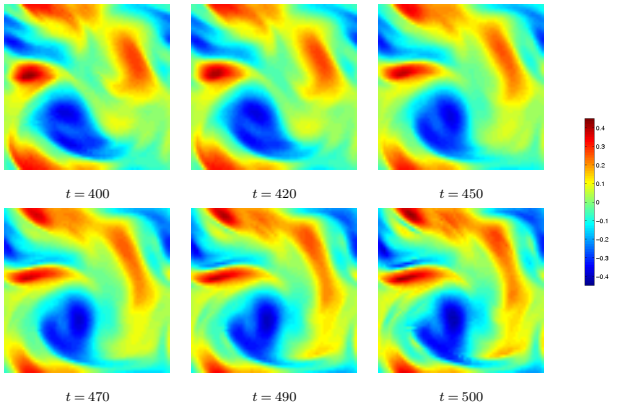


Figure 10. Smoothing result with the proposed method. Estimated mean vorticity maps for different times t between observation times $t = 400$ and $t = 500$.

steps before time t_k, there are in practice only a few distinct filtering trajectories at times t_{k-h} if h is large. Consequently, the estimation of the joint law in Eq. (15) will not be reliable anymore if h is too large.

We have shown the practical applicability of the method to a high-dimensional problem. Nevertheless, the algorithm remains costly since a second Monte Carlo step is added to the Monte Carlo nature of particle filter algorithms. Yet, from an algorithmic point of view, the sequential nature of the proposed technique allows the smoothing to be implemented with a similar structure as filtering methods (sequential sampling and weighting of model trajectories). It is then easy to couple this smoothing to an operational filtering system and benefit from parallelization strategies, for instance.

Since the proposed smoothing uses the filtering result as input, it relies on the success of the underlying particle filter. For high-dimensional systems, a standard particle filter is not adapted and it is necessary to use filtering techniques that guide particles towards observations. In this paper, we use the WEnKF algorithm. In practice, any efficient particle filtering technique with such a guiding can be used within our framework. Note however that the construction of such techniques remains an open area of research.

We plan to work on the application of the smoothing method to a real high-dimensional case (for the estimation of sea surface currents from satellite image sequences). However, such a work will imply numerous difficulties which are not related to the smoothing technique but to the definition of the state-space model: definition of a suitable physical model, good characterization of state noise structure and model parameters. Therefore, this will be part of a future work.

Edited by: P. J. van Leeuwen

References

Beskos, A. and Roberts, G. O.: Exact simulation of diffusions, The Annal. Appl. Probabil., 15, 2422–2444, 2005.

Beskos, A., Papaspiliopoulos, O., Roberts, G. O., and Fearnhead, P.: Exact and computationally efficient likelihood-based estimation for discretely observed diffusion processes (with discussion), J. Roy. Stat. Soc. B, 68, 333–382, 2006.

Beyou, S., Cuzol, A., Gorthi, S., and Mémin, E.: Weighted Ensemble Transform Kalman Filter for Image Assimilation, Tellus A, 65, 18803, doi:10.3402/tellusa.v65i0.18803, 2013.

Briers, M., Doucet, A., and Maskell, S.: Smoothing algorithms for state-space models, Ann. Insti. Stat. Mathe., 62, 61–89, 2010.

Clark, J.: The simulation of pinned diffusions, in: Proceedings of the 29th IEEE Conference on Decision and Control, 1418–1420, 1990.

Del Moral, P.: Feynman-Kac Formulae. Genealogical and Interacting Particle Systems with Applications, Springer, 2004.

Del Moral, P., Jacod, J., and Protter, P.: The Monte Carlo Method for filtering with discrete-time observations, Probabil. Theor. Relat. Fields, 120, 346–368, 2001.

Delyon, B. and Hu, Y.: Simulation of conditioned diffusions and applications to parameter estimation, Stochast. Process. Applic., 116, 1660–1675, 2006.

Doucet, A., Godsill, S., and Andrieu, C.: On sequential Monte Carlo sampling methods for Bayesian filtering, Stat. Comput., 10, 197–208, 2000.

Durham, G. and Gallant, A.: Numerical techniques for maximum likelihood estimation of continuous-time diffusion processes, J. Business Econom. Stat., 20, 297–316, 2002.

Evensen, G.: The ensemble Kalman filter: theoretical formulation and practical implementation, Ocean Dynam., 53, 343–367, 2003.

Evensen, G. and van Leeuwen, P.: An ensemble Kalman Smoother for nonlinear dynamics, Mon. Weather Rev., 128, 1852–1867, 2000.

Fearnhead, P., Papaspiliopoulos, O., and Roberts, G.: Particle filters for partially observed diffusions, J. Roy. Stat. Soc. B, 70, 755–777, 2008.

Godsill, S. J., Doucet, A., and West, M.: Monte Carlo smoothing for nonlinear time series, J. Am. Stat. Assoc., 99, 156–168, 2004.

Gordon, N., Salmond, D., and Smith, A.: Novel approach to nonlinear/non-Gaussian Bayesian state estimation, IEEE Proc. F, 140, 107–113, 1993.

Papadakis, N., Mémin, E., Cuzol, A., and Gengembre, N.: Data assimilation with the weighted ensemble Kalman filter, Tellus A: Dynam. Meteorol. Oceanogr., 62, 673–697, 2010.

Snyder, C., Bengtsson, T., Bickel, P., and Anderson, J.: Obstacles to high-dimensional particle filtering, Mon. Weather Rev., 136, 4629–4640, 2008.

Stroud, J. R., Stein, M. L., Lesht, B. M., Schwab, D. J., and Beletsky, D.: An ensemble Kalman filter and smoother for satellite data assimilation, J. Am. Stat. Assoc., 105, 978–990, 2010.

Sun, Y., Bo, L., and Genton, M. G.: Geostatistics for large datasets, in: Advances and challenges in space-time modelling of natural events, 55–77, Springer, 2012.

Van Leeuwen, P. J.: Particle filtering in Geophysical systems, Mon. Weather Rev., 137, 4089–4114, 2009.

Van Leeuwen, P. J.: Nonlinear data assimilation in geosciences: an extremely efficient particle filter, Q. J. Roy. Meteorol. Soc., 136, 1991–1999, 2010.

Van Leeuwen, P. J. and Ades, M.: Efficient fully nonlinear data assimilation for geophysical fluid dynamics, Comput. Geosci., 55, 16–27, 2013.

On-road Trajectory Generation from GPS Data: A Particle Filtering/Smoothing Application

Michael Roth, Fredrik Gustafsson and Umut Orguner

Michael Roth, Fredrik Gustafsson and Umut Orguner, On-road Trajectory Generation from GPS Data: A Particle Filtering/Smoothing Application, 2012, 15th International Conference on Information Fusion (FUSION).

Postprint available at: Linköping University Electronic Press
http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-88782

On-road Trajectory Generation from GPS Data: A Particle Filtering/Smoothing Application

Michael Roth and Fredrik Gustafsson
Dept. of Electrical Engineering
Linköping University
Linköping, Sweden
Email: roth@isy.liu.se, fredrik.gustafsson@liu.se

Umut Orguner
Dept. of Electrical and Electronics Engineering
Middle East Technical University
Ankara, Turkey
Email: umut@eee.metu.edu.tr

***Abstract*—Many studies in target localization and tracking use GPS measurements as ground truth. These GPS locations might be in conflict with computed estimates in applications where road network information is available (and employed in the estimation procedure). This paper proposes to use particle methods to generate on-road trajectories that can be used as improved ground truth for such road constrained estimation schemes. A bootstrap particle filter and three different particle smoothers are utilized to obtain kinematic target state estimates. The particle smoothers require important adjustments for their implementation in the resulting hybrid state space. The performances of the presented methods are compared on challenging real data obtained from an urban area. Although particle filters and smoothers can be applied to general localization problems, with arbitrary sensors, we concentrate on GPS measurements, motivated by an application in cellular network systems.**

I. Introduction

There are many target tracking and localization applications where the target is constrained to a known road network. We can here mention localization in cellular networks, tracking based on ground radar or aerial vision sensors, and various ground sensor network applications. Our motivation comes from a practical need for a system manufacturer of cellular network systems, where they currently employ manual drive tests to measure system performance and optimize network parameters. This is also where our field test data come from. A related need occurs in self-optimizing networks (SON), where user equipments are commanded to transmit position related data whenever they face a problem, such as a missed or dropped call, poor signal to noise ratio, severe multipath, late hand-over *etc*. If the user equipment is concluded to be road-bound, coverage over the road network can be optimized by adjusting antenna angles and system parameters. For SON, smoothing is both possible, and also required to get sufficient accuracy of the often inaccurate network measurements.

In all of these applications, one needs to compare estimates (which are constrained to the road) with ground truth position, which has to be more accurate than the estimates. Often GPS positions are employed as ground truth. However, GPS data might be in conflict with the road information. We therefore consider the problem of generating on-road trajectories from GPS measurements. Instead of projecting the GPS data onto the closest road segment, which is problematic at road intersections, we pursue a filtering/smoothing approach. An estimation scheme that includes road network information requires state-of-the-art algorithms. A concise overview of recent developments is given in [1]. Among the vast literature on road constrained tracking we furthermore highlight the book [2], which advocates the here applied particle methods, and an article [3] that showed how improved tracking results were obtained by considering the road network.

From an estimation perspective the main challenge is that the state space has discrete components: the target's road segment. Classical (Kalman) filtering approaches are often used in an interacting multiple model (IMM) framework (with the road segment as mode state) for such a problem. Further modifications are required to obtain computationally feasible algorithms. Particle methods in turn can be applied directly. We investigate a bootstrap particle filter [4] and a number of smoothing algorithms that are based on the filter output. The involved state space model, likelihood, and transition density are developed for the problem at hand.

The on-road trajectory generation problem is formulated in the next section. Section III introduces the required particle methods while Section IV covers the application in detail. Results are shown and discussed in Section V. Concluding remarks are given in Section VI.

II. Problem formulation

We want to obtain the kinematic state of a target that is known to travel on-road by processing available GPS measurements. In our setting, road segments are considered as curves in the x-y-plane and in general the given GPS measurements do not lie on these curves. On one hand this is a result of neglecting the road widths, on the other hand the GPS measurements might be erroneous (multipath effects in urban areas, atmospheric effects, timing errors). Figure 1 shows the considered GPS measurements with time stamps and a section of the road network.

An ad-hoc procedure to obtain on-road data would be to orthogonally project each GPS position onto the closest road segment. Apart from the task of finding the closest segment,

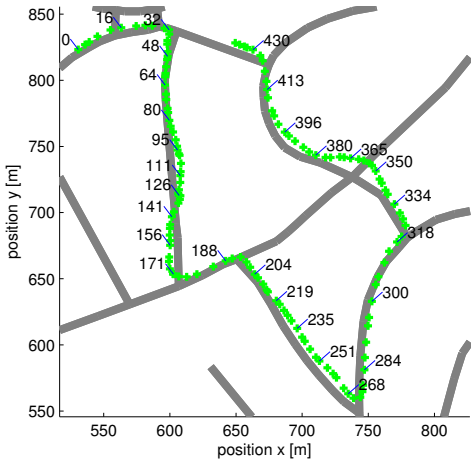


Figure 1. GPS trajectory and road network section, time stamps in seconds.

this can give erroneous results, depending on the road topology. A difficult bit can be seen in Figure 1 between 350 and 380 seconds. Clearly, a projection onto the closest segment would pick the wrong road in this example.

Alternatively, we consider a tracking problem for a target that is bound to move on-road [2]. The road segment is treated as state variable. Filtering and smoothing techniques can be used to obtain approximate probability density functions which again can be used to compute an estimate. More specifically, particle methods are chosen to cope with the hybrid state space. It is of interest to see if the inherent lookahead benefit of smoothing can yield improved performance over filtering in this off-line problem.

III. Particle filtering and smoothing theory

In this section we briefly introduce the required particle methods. We consider a discrete time state space model

$$x_{k+1} = f_k(x_k, v_k), \tag{1}$$
$$y_k = h_k(x_k, e_k). \tag{2}$$

Subsequent states x_{k+1} are related to current states x_k and process noise v_k by the (possibly nonlinear and possibly time-varying) function f_k. A measurement y_k is generated by x_k via the function h_k and measurement noise e_k. Both v_k and e_k are assumed to have known probability distributions. In case of additive noise e_k and a continuous measurement space, the likelihood $p(y_k|x_k)$ can be computed by evaluating the probability density function of e_k at $y_k - h(x_k)$.

A. Particle Filtering (PF)

An introduction to the subject can be found in e.g. several tutorial papers [5]–[7] or recent books [2], [8]. Gordon et al. [4] is often cited as first particle filter (PF) and indeed we will stick to the proposed method.

In the filtering problem one seeks the posterior distribution of a state x_k given all the measurements up to time step k: $p(x_k|y_{1:k})$. Although a conceptual solution to this problem is given using Bayes' law, the resulting posterior cannot be described by a finite number of parameters in general (except for the well known special cases). The PF therefore approximates the posterior by a set of samples (particles) and weights $\{x_k^{(i)}, w_k^{(i)}\}_{i=1}^N$. Actually, the algorithm provides a particle representation of the joint smoothing density $p(x_{0:k}|y_{1:k})$ that is sequentially computed based on the concept of importance sampling. At each step k, N samples $x_k^{(i)}$ are drawn from proposal distributions $q(x_k|x_{0:k-1}^{(i)}, y_{1:k})$ and appended to the existing trajectories to form $\{x_{0:k}^{(i)}\}_{i=1}^N$. The corresponding weights are then computed according to

$$w_k^{(i)} = \frac{p(x_k^{(i)}|x_{k-1}^{(i)})p(y_k|x_k^{(i)})}{q(x_k^{(i)}|x_{0:k-1}^{(i)}, y_{1:k})} \tag{3}$$

and subsequently normalized. In case the proposal is chosen as prior $p(x_k|x_{k-1}^{(i)})$, as in [4], each particle (trajectory) gets weighted by its likelihood only. Samples from the prior can be drawn by predicting the particle with an independent realization of v_k according to (1).

Problematic is that this sequential importance sampling eventually leads to so called weight degeneracy. That is, all but one weights will turn to zero. In [4] a crucial resampling step, in which particles are discarded or duplicated according to their weights, was introduced to counteract this problem. The degree of degeneracy can be assessed by the effective number of particles (approximately) given by

$$N_{\text{eff}} = \frac{1}{\sum_{i=1}^N (w^{(i)})^2}. \tag{4}$$

Resampling can be applied whenever N_{eff} falls below a threshold.

An approximate marginal filtering density can be obtained by extracting $\{x_k^{(i)}, w_k^{(i)}\}_{i=1}^N$ from $\{x_{0:k}^{(i)}, w_k^{(i)}\}_{i=1}^N$. The algorithm is initialized with appropriate particles and weights $\{x_0^{(i)}, w_0^{(i)}\}_{i=1}^N$.

B. Particle Smoothing

Particle smoothers have only recently been developed. There exist different variants of which three will be explained in this section. Further details can be found in the overview [6], the thesis [9], and the references below.

We seek the marginal smoothing density $p(x_k|y_{1:K})$ where $y_{1:K}$ is a batch of measurements and k ranges from 0 to K. Again, a conceptual solution can be derived but needs to be approximated by sets of particles. The presented algorithms rely on the output of a previously run PF and do not generate new particles. Therefore, it is necessary that regions of interest in the state space are explored thoroughly in the filtering

procedure. For bootstrap filters this means that the process noise should not be too small.

1) Fixed lag smoothing distribution from filter output (FL): In Section III-A we stated that the particle filter actually gives an approximation of the joint smoothing density $p(x_{0:k}|y_{1:k})$. In the same way as we extracted marginal densities at time k we could also obtain a particle representation of a fixed lag smoothing density $p(x_{k-L}|y_k)$ by pairing the corresponding weights and samples $\{x_{k-L}^{(i)}, w_k^{(i)}\}_{i=1}^N$. It should be noted that due to the resampling process many particles at step k share common ancestors. For this reason a depleted sampling representation is expected for large lags L.

2) Forward filter backward smoother (FFBSm): The method proposed in [10] targets the marginal smoothing distribution. It is sometimes referred to as forward filter backward smoother. The particles from the filtering step are kept but assigned new weights. Initialized with the filtering weights $\{w_K^{(i)}\}_{i=1}^N$ the following update is performed for $k = K-1$ to $k = 0$:

$$\bar{w}_k^{(i)} = \sum_{j=1}^{N} \bar{w}_{k+1}^{(j)} \frac{w_k^{(i)} p(x_{k+1}^{(j)}|x_k^{(i)})}{\sum_{l=1}^N w_k^{(l)} p(x_{k+1}^{(j)}|x_k^{(l)})}. \tag{5}$$

The computational complexity for smoothing a trajectory this way is of order KN^2 as the transition density needs to be evaluated for all pairings of particles at k and $k+1$.

3) Forward filter backward simulator (FFBSi): Another approach, suggested in [11], is to systematically pick M samples $\{\tilde{x}_k\}_{i=1}^M$ from $\{x_k^{(i)}\}_{i=1}^N$ backwards in time for each k and to append them to previously assembled trajectories $\{\tilde{x}_{k+1:K}\}_{i=1}^M$. These M "simulated" trajectories (hence the name) form an equally weighted particle approximation of the joint smoothing density. We present how a single trajectory is drawn. At K, one sample $\tilde{x}_K = x_K^{(i)}$ is chosen with probability $w_K^{(i)}$. For each k from $K-1$ to 0, N normalized weights are computed according to

$$\tilde{w}_k^{(i)} = \frac{w_k^{(i)} p(\tilde{x}_{k+1}|x_k^{(i)})}{\sum_{l=1}^N w_k^{(l)} p(\tilde{x}_{k+1}|x_k^{(l)})} \tag{6}$$

and again used to select $\tilde{x}_k = x_k^{(i)}$ with probability $\tilde{w}_k^{(i)}$. Implementing this procedure requires KMN evaluations of the transition density. We can hence use $N > M$ particles in the filter and subsequently select fewer particles for the smoother to decrease the number of operations. An even more efficient variant that scales as KN has been suggested in [12].

For algorithmic details of all presented smoothing algorithms and the fast FFBSi implementation [12], the reader is referred to [9].

C. Computing an estimate from a cloud of particles

In most applications a state estimate $\hat{x}_k$ and an indication of its confidence (e.g. a covariance matrix) are required. In a continuous state space the minimum mean squared error (MMSE) estimate is a popular choice. It is merely given by a weighted average of particles $\hat{x}_k = \frac{1}{N}\sum_{i=1}^N w_k^{(i)} x_k^{(i)}$. Alternatively, a maximum a posteriori (MAP) estimate is desired in a filtering application. This is not given by the particle with the highest weight. A method to compute MAP estimates from particles can be found in [13]. The suggested algorithm involves evaluation of the transition density and can be used to incorporate more information into the filtering process while keeping a simple bootstrap filter thats draws samples from the prior. Similar to a MAP estimate for filtering a maximum smoothing pdf estimate could be derived.

In a state space with discrete components MMSE estimates cannot be computed by simple averaging. Also, constraints on the continuous part of the state space are challenging to handle. We provide a solution for our specific problem in Section IV-G.

IV. Application to the on-road trajectory generation problem

We will next develop the required functions to apply the algorithms of the previous section.

A. Road network information

There are several ways how a road network can be described, as seen in e.g. [1], [14]. The choice of description in turn leads to a certain type of road coordinates.

We focus on a point wise description which represents the network by labeled way points (nodes) that are stored along their position and the labels of all connected nodes. (Example: node 4 has position $\mathsf{x} = 3$ and $\mathsf{y} = 204$ and is connected to nodes 3, 8, and 149.) The nodes are assumed to be connected by straight road segments and the segment lengths can be readily computed. Also one-way roads can be represented in this format. For the time being, we do not consider road widths, speed limits, stop signs and traffic lights, which among other information could be included in more informative road network data.

The description relates to the field of graph theory and node indices and segment lengths can be seen as a weighted directed graph [15]. Consequently we can obtain the shortest paths between all pairs of nodes and also find out about the number of intersections on these shortest paths. For each node pair i, j we store the distance on the shortest path and a factor f_{ij} that is computed according to following algorithm:

1: set $\mathsf{f}_{ij} = 1$
2: **for all** nodes l on the shortest path from i to j **do**
3: **if** node l is an intersection **then**
4: set n_l = number of nodes attached to l
5: $\mathsf{f}_{ij} = \mathsf{f}_{ij}/(\mathsf{n}_l - 1)$
6: **end if**
7: **end for**

Each factor f_{ij} is the probability to be at node j when, starting at node i, a distance as long as the shortest path between i and j is traveled. Here the assumption is made that all other connecting paths can be neglected and only one initial direction is considered. The factor relates to the motion model of Section IV-C and will be used in the evaluation of transition densities in Section IV-E.

We symbolically denote the entire road network by $\mathcal{R}$.

B. Road and global coordinates

Given the road network $\mathcal{R}$, we now turn to the state representation of a road bound target. Each segment of $\mathcal{R}$ can be identified by two node index integers n_1 and n_2. These form the discrete part of the state space.

Let d be the target position (a distance) on the current road segment, by convention measured from n_1. Obviously this distance has to obey $0 < \mathsf{d} < \mathsf{l}$ where l is the segment length. Let s be the speed on the current road segment. By convention a target with $\mathsf{s} > 0$ is traveling from n_1 towards n_2.

A kinematic state of an on-road target is given by

$$\begin{aligned} x_k &= \begin{pmatrix} \mathsf{d}_k, & \mathsf{s}_k, & \mathsf{n}_{1k}, & \mathsf{n}_{2k} \end{pmatrix}^T \\ &\triangleq \begin{pmatrix} \mathsf{l}_k - \mathsf{d}_k, & -\mathsf{s}_k, & \mathsf{n}_{2k}, & \mathsf{n}_{1k} \end{pmatrix}^T . \end{aligned} \tag{7}$$

The second row shows that the same state has two different representations.

We next relate the state to the GPS measurements which are given in a Cartesian coordinate frame. Let x and y denote the corresponding two dimensional target position, $\dot{\mathsf{x}}$ and $\dot{\mathsf{y}}$ Cartesian velocities. Using the road network information, each state (7) can be transformed to Cartesian coordinates by simple geometric operations. In particular, we denote the mapping of road bound to Cartesian position by $\mathsf{M}(\mathsf{d}, \mathsf{n}_1, \mathsf{n}_2, \mathcal{R})$ and obtain the following nonlinear measurement equation (2):

$$y_k = \begin{pmatrix} \mathsf{x}_k^{\mathsf{GPS}}, & \mathsf{y}_k^{\mathsf{GPS}} \end{pmatrix}^T = \underbrace{\mathsf{M}(\mathsf{d}_k, \mathsf{n}_{1k}, \mathsf{n}_{2k}, \mathcal{R})}_{h(x_k)} + e_k. \tag{8}$$

The distribution of the two dimensional noise e_k will be described in Section IV-D.

C. State transition function

In order to draw independent samples from the prior distribution $p(x_{k+1}|x_k)$ in a particle filter, we predict each particle with an independent process noise realization. Here we develop the required state transition function (1).

The two state components d_k and s_k are first treated without (segment length) constraints and subsequently adjusted if necessary. As the available measurements are sampled non-uniformly at times t_k we define $\mathsf{T}_k = \mathsf{t}_{k+1} - \mathsf{t}_k$ to be the sampling intervals. Of several available one-dimensional motion models [8], [16] we pick a (nearly) constant velocity type:

$$\begin{pmatrix} \overline{\mathsf{d}}_{k+1} \\ \mathsf{s}_{k+1} \end{pmatrix} = \begin{pmatrix} 1 & \mathsf{T}_k \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \mathsf{d}_k \\ \mathsf{s}_k \end{pmatrix} + \mathsf{v}_k. \tag{9}$$

The process noise v_k affects the target's maneuverability. We specify its covariance matrix

$$\mathsf{Q}_k = \operatorname{cov} \mathsf{v}_k = \begin{pmatrix} \mathsf{T}_k^3/3 & \mathsf{T}_k^2/2 \\ \mathsf{T}_k^2/2 & \mathsf{T}_k \end{pmatrix} \mathsf{q} \tag{10}$$

but do not restrict ourselves to the commonly chosen Gaussian distribution. The model (9) together with (10) can be derived by discretization of a continuous-time constant velocity model with white acceleration noise (power spectral density q) input [16]. The matrix (10) has full rank in contrast to its more often used alternative $[(\frac{\mathsf{T}_k^2}{2}, \mathsf{T}_k)\,\mathsf{q}\,(\frac{\mathsf{T}_k^2}{2}, \mathsf{T}_k)^T]$ and thus, any expressions involving its inverse, e.g. densities, can be evaluated.

The particle filtering framework merely requires that samples from the process noise can be generated. This in turn opens up for the use of wider tailed distributions than the Gaussian that might be beneficial in modeling a wider range of maneuvers, for instance Student's t-distribution [17]. Such process noise choices can be used to complement IMM approaches [3] and might reduce the number of required modes.

The linear update (9) might yield a $\overline{\mathsf{d}}_{k+1}$ that extends the current road segment. Using the road network information we project the excess distance onto a following segment [18] and alter the node indices accordingly. Of course there might be several candidates so whenever an intersection is involved we pick one of the alternatives with equal probability (without reconsidering the segments we came from). Also, road segments can be skipped in one prediction, especially for a dense road network with many short segments. An example is illustrated in Figure 2. Symbolically, we define the projection P and write

$$(\mathsf{d}_{k+1}, \mathsf{n}_{1k+1}, \mathsf{n}_{2k+1})^T = \mathsf{P}(\overline{\mathsf{d}}_{k+1}, \mathsf{n}_{1k}, \mathsf{n}_{2k}, \mathsf{c}_k, \mathcal{R}) \tag{11}$$

where c_k represents the random choices during the projection.

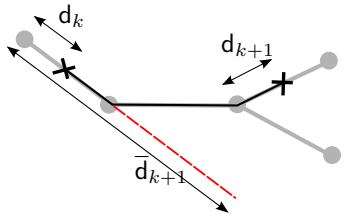


Figure 2. Prediction of on-road distances; here, a road segment is skipped.

D. Likelihood

The two dimensional real valued measurement noise e_k can be obtained from Equation (8). As it enters additively, the likelihood $p(y_k|x_k)$ can be computed by evaluating the (yet to

be determined) noise pdf. We will pursue a practical approach and base the likelihood choice on its function in the PF.

In the utilized bootstrap filter each particle is weighted by its likelihood. By inspecting Figure 1 it is reasonable to have a likelihood that 1) does not decay too fast with Euclidean distance from 0 (such that the filter works even though the measurement is far from the road), but also that 2) does not assign disproportionately high weights to particles that are close to measurements. The latter demand is to avoid being tricked by ambiguous situations as between 350 and 380 seconds. In summary, the likelihood should not have a high peak close to 0 but instead be "less informative".

Choices for $p(y_k|x_k)$ include 1) uniform on a circle around the origin, i.e. all particles in a defined vicinity of the measurement get assigned equal weights. The circle radius should be larger than the maximum distance between measurement and the "true road". Here, particles that are too far away from the measurement will be discarded immediately in the resampling step. 2) Gaussian with large covariance matrix. Here, each particle gets assigned an individual weight.

E. Transition density

In order apply the algorithms of Section III-B, we need to be able to evaluate transition densities $p(x_{k+1}|x_k)$. The key idea is to reverse the mapping (11) to reconstruct the intermediate $\overline{\mathsf{d}}_{k+1}$ from (9). With certain rearrangements we can then proceed as if x_{k+1} and x_k were on the same road segment.

Let us first consider the trivial case of x_{k+1} and x_k on the same road segment: For interchanged node indices ($\mathsf{n}_{1k+1} = \mathsf{n}_{2k}$) we need to alter one of the states in the same way as shown in (7). If $\mathsf{n}_{1k+1} = \mathsf{n}_{1k}$ we can simply derive the process noise term v_k from (9). Evaluating the pdf of v_k gives the desired transition density under the assumption that the target moved on the shortest path from x_k to x_{k+1}.

For the case of x_{k+1} and x_k on different road segments we again assume the target to have taken the shortest path. From the available states we know that it traveled on one of four different routes (via n_{1k} and n_{1k+1}, n_{1k} and n_{2k+1}, n_{2k} and n_{1k+1}, or n_{2k} and n_{2k+1}). Obviously, these paths will have different lengths which we can compute using the road network information. Let

$$\tilde{\mathsf{d}}_{k+1} = \overline{\mathsf{d}}_{k+1} - \mathsf{d}_k \tag{12}$$

be an increment in on-road distance obtained by the prediction step. Then the shortest path is the smallest increment among the candidates

$$\begin{aligned}
\tilde{\mathsf{d}}^1_{k+1} &= \mathsf{d}_{k+1} + \mathsf{D}(\mathsf{n}_{1k+1}, \mathsf{n}_{2k}, \mathcal{R}) + (\mathsf{l}_k - \mathsf{d}_k),\\
\tilde{\mathsf{d}}^2_{k+1} &= \mathsf{d}_{k+1} + \mathsf{D}(\mathsf{n}_{1k+1}, \mathsf{n}_{1k}, \mathcal{R}) + \mathsf{d}_k,\\
\tilde{\mathsf{d}}^3_{k+1} &= (\mathsf{l}_{k+1} - \mathsf{d}_{k+1}) + \mathsf{D}(\mathsf{n}_{2k+1}, \mathsf{n}_{2k}, \mathcal{R}) + (\mathsf{l}_k - \mathsf{d}_k),\\
\tilde{\mathsf{d}}^4_{k+1} &= (\mathsf{l}_{k+1} - \mathsf{d}_{k+1}) + \mathsf{D}(\mathsf{n}_{2k+1}, \mathsf{n}_{1k}, \mathcal{R}) + \mathsf{d}_k.
\end{aligned}$$

The utilized function D returns the distance of two nodes on the shortest connecting path (which is basically a look up from $\mathcal{R}$). For instance $\mathsf{D}(\mathsf{n}_{1k+1}, \mathsf{n}_{2k}, \mathcal{R})$ gives the distance on the shortest path from n_{2k} to n_{1k+1}. As each increment corresponds to one of the four alternatives, we can (with careful rearrangements) find $\overline{\mathsf{d}}_{k+1}$ and s_{k+1}. Care needs to be taken with the sign of s_{k+1}. With the recovered quantities we can again evaluate the pdf of v_k from (9).

For different road segments $p(x_{k+1}|x_k)$ is not necessarily given by the pdf of v_k because there could have been intersections on the shortest path. Therefore we multiply it by the correcting factor $\mathsf{f} \leq 1$ (Section IV-A) that accounts for the random choices (Section IV-C) on the assumed shortest path.

F. Filter and smoother initialization

The filtering algorithm depends on its initial conditions, that is the set of particles at $k = 0$. For the on-road trajectory generation problem we suggest to manually select appropriate on-road states $x_0^{(i)}$ (especially the road segment), based on the corresponding GPS measurement. Similarly, we can replace the smoother initial particles (approximation of $p(x_K|y_{1:K})$) by a manually selected set. Here, it might be useful to spread out the particles a bit further such that the evaluated $p(x_K^{(j)}|x_{K-1}^{(i)})$ differs from zero for sufficiently many pairs of particles.

G. Computing an estimate

At this stage we should remind ourselves of the actual aim of on-road trajectory generation. We want to obtain accurate kinematic states that can be used as ground truth for other tracking algorithms. The desired trajectory should be given in a global coordinate frame instead of road coordinates. Of course any state x in road coordinates can be transformed to global coordinates by simple geometric relations. We introduce $\xi = \begin{pmatrix}\mathsf{x} & \mathsf{y} & \dot{\mathsf{x}} & \dot{\mathsf{y}}\end{pmatrix}^T$ as such a transformed x and proceed by considering estimates that can be obtained from weighted particles $\{\xi^{(i)}, w^{(i)}\}_{i=1}^N$.

The classical MMSE estimate given by $\hat{\xi}_k^{\mathsf{MMSE}} = \sum_i w_k^{(i)} \xi^{(i)}$ is generally off-road and hence not applicable to our problem. However, if the estimate is chosen as minimizer of some cost function among all particles, that is $\xi^{(i)}$ with

$$i = \arg\min_i C(\xi^{(i)}), \tag{13}$$

the resulting estimate will be on-road. It can be computed by evaluating the cost function at all particles. Minimization of

$$C(\xi) = \sum_j w^{(j)} (\xi - \xi^{(j)})^T (\xi - \xi^{(j)}) \tag{14}$$

yields the true expected value for many particles in the absence of road constraints. With road constraints present, $\xi^{(i)}$ found by (13) provides a meaningful on-road expected value estimate and can be seen as constrained extension of

the standard MMSE scheme. We denote the obtained estimate by $\hat{\xi}^{\mathsf{CMMSE}}$. Moreover, the cost function in (13) can be chosen arbitrarily and facilitates computation of alternative estimates, for instance based on $C(\xi) = \sum_j w^{(j)} \|\xi - \xi^{(j)}\|$.

V. Results and Discussion

A. Road network and trajectory analysis

The considered road network consists of 830 nodes which are connected by 890 road segments. It covers an area of about two square kilometers.

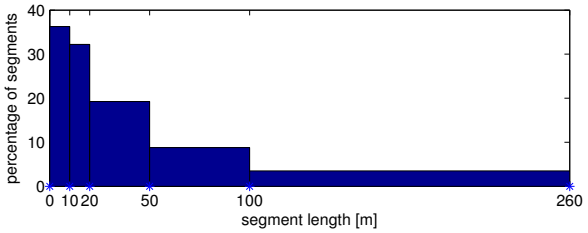


Figure 3. Road network data: normalized segment length histogram.

Figure 3 shows a normalized histogram of the segment lengths. It can be seen that about 36 per cent are less than 10 meters, about 32 per cent are between 10 and 20 meters in length. These short segments are likely to be skipped in case of a target that travels at higher speeds or large intervals between consecutive measurements. On the other end of the scale we see that more than 3 per cent of the segments are longer than 100 meters. These few long segments, however, amount to more than 4 kilometers of road. Turning to the amount of intersections in the road network we analyze the number of adjacent segments to each node. About 15 per cent of the nodes are intersections with 3 (12 per cent) and 4 (3 per cent) attached roads. A considerably large percentage of the nodes (5 per cent) are dead ends. This can be explained by boundary effects of the regarded network sector. Appropriate reactions to such a dead end road must be accounted for in the motion model to avoid strange behavior while filtering. The majority of nodes (80 per cent) has two road segments attached.

We next analyze the provided GPS trajectory which consists of $K = 140$ measurements that have been taken from a larger data set. From Figure 1 we saw already that it deviates from the road network. Figure 4 contains two normalized histograms of which the first displays the Euclidean distances between consecutive measurements. It can be seen that the increments are rather short with a mean of about 5 meters. The second histogram shows the times between consecutive sampling instants which range from 2.6 to 3.9 seconds. The target speed is low with a mean value of 2 meters per second.

B. Filtering and smoothing results

The filtering and smoothing algorithms of Section III have been implemented and tested with different settings. We

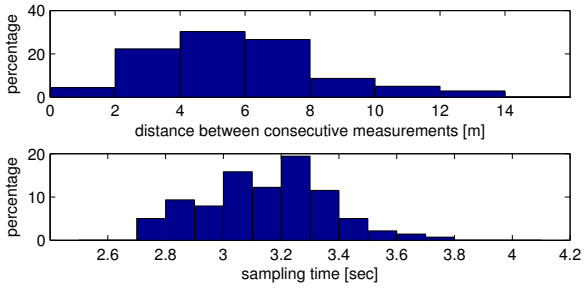


Figure 4. GPS trajectory: normalized histograms of distances between consecutive measurements, sampling times.

present results that were obtained with $N = 500$ and $M = 100$ particles. Resampling in the PF was carried out whenever $N_{\mathsf{eff}} < \frac{1}{2}N$. The likelihood was chosen to be uniform on a circle with 25 meter radius. The driving noise parameter of (10) was set to $\mathsf{q} = 0.1$, and the corresponding v were chosen from a Student's t distribution [17] with degrees of freedom parameter $\nu = 3$. The fixed lag (FL) results have been obtained from the PF output with $L = 3$.

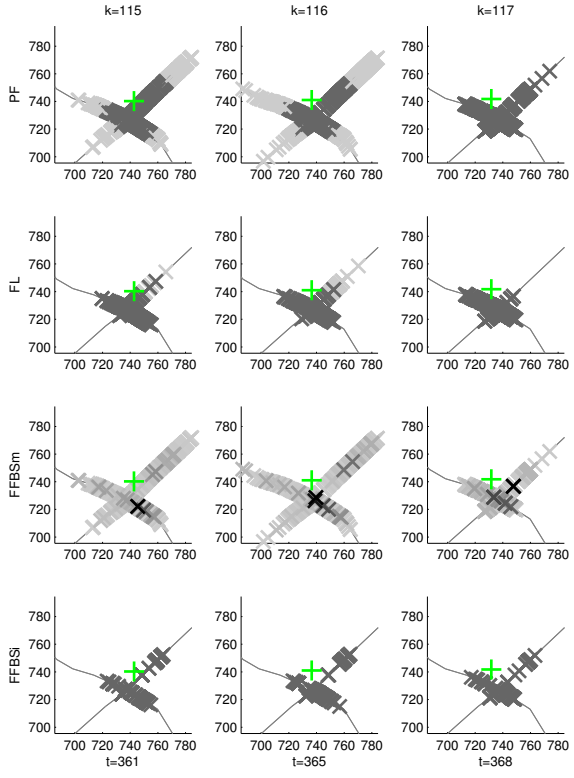


Figure 5. Particle clouds for the applied algorithms at a challenging road map sector; GPS measurement marked by green +; x and y in meter, times t in seconds.

The subplots in Figure 5 illustrate clouds of particles for three consecutive time steps at a challenging road map sector, see

also Figure 1. Only the position components $x_k^{(i)}$ and $y_k^{(i)}$ are displayed as gray crosses. The color intensity corresponds to the particle weights. In all plots the GPS measurement is illustrated with a green +-marker, and clearly off-road. Time steps and corresponding times are provided for each column. The first row shows PF samples. As a result of the uniform likelihood, all particles in the 25 meter circle around the measurement have the same weight. In the last scan of the first row all samples have the same weight so resampling must have taken place. It can furthermore be seen how particles spread out on the correct and wrong road. The second row displays fixed lag smoothing (FL) samples which appear more focused around the measurement. Even for a small lag $L = 3$ fewer particles are found on wrong roads. The third row presents similar scans for the FFBSm algorithm. Relatively few samples are assigned significant weights (the darkest crosses), among them some on the wrong segment (e.g. third column). This can be explained by the fact that the algorithm computes the weight not only based on the illustrated position but also on the particle speed. FFBSi is illustrated in the last row. Here, only $M = 100$ samples are used. Although all particles carry the same weight, some of them are duplicates. Formed clusters in each scan reveal multi modality in the provided smoothing density.

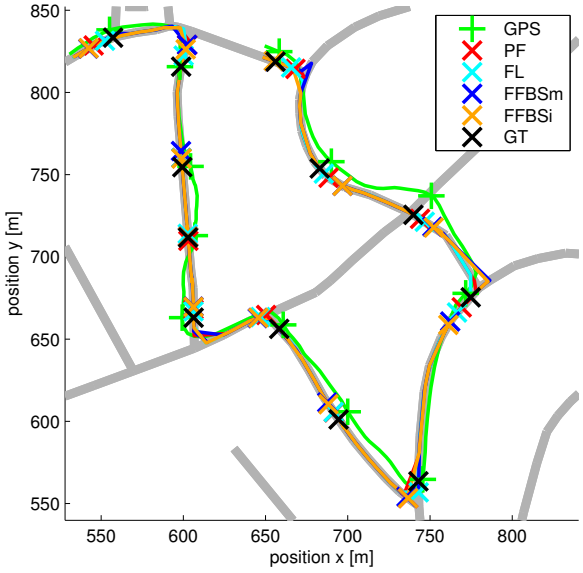


Figure 6. On-road trajectories and point estimates for a number of steps.

Position estimates were computed using the CMMSE scheme of Section IV-G. In Figure 6 they are plotted as solid lines, obviously aligned to the road map except for few cut corners. For selected k the estimates, along the GPS measurement and a manually projected position (ground truth GT), are given as markers. All algorithms managed to resolve the ambiguous situation that was highlighted in Section II. FFBSm and FFBSi briefly pick the wrong road segment in the rightmost turn. Also PF is subject to such erratic estimates, in other runs even to a larger extend. The manually projected data are generally ahead of all estimates and closest to them is the FL smoother. FFBSm and FFBSi lie often close to one another. The manually projected data (GT) were used to compute RMS

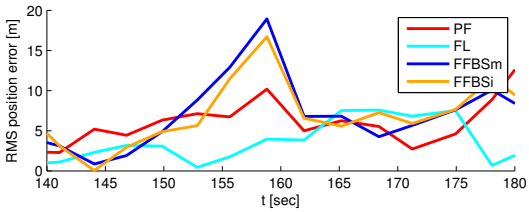


Figure 7. RMS position error computed from manually projected position data.

position errors which are illustrated in Figure 7. Averaged (over the entire trajectory) RMS position errors (in meter) are PF 7.9, FL 6.4, FFBSm 11.7, FFBSi 11.3.

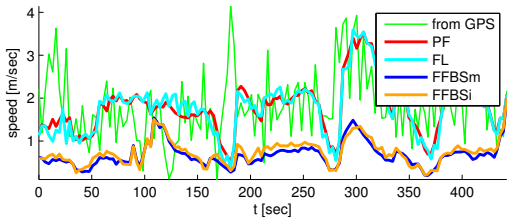


Figure 8. Speed computed from velocity estimates and differenced GPS data.

Velocity estimates ($\dot{x}$ and $\dot{y}$) are not constrained to the road map and can hence be computed as weighted particle means. Figure 8 displays the associated speed $\sqrt{\dot{x}^2 + \dot{y}^2}$ over time, including a signal that has been obtained by differencing the GPS data. The GPS speed is noise corrupted and thus likely to be higher than how fast the target actually traveled. PF and FL closely follow the GPS signal – their velocity is overestimated due to noise effects. FFBSm and FFBSi both provide slower velocity estimates which appear more realistic.

Motivated by the velocity results, we investigate FFBSm and FFBSi further and turn to computational aspects. The configuration $N = 500$ led to largely increased computation times for FFBSm (KN^2 operations). A lower N could be used, but this resulted in a depleted particle representation as too few weights turned out to be of significant size. Even for $N = 500$ the effective number of particles (4) is low for FFBSm, as illustrated in Figure 9. Furthermore, numerical problems occurred during FFBSm runs whenever the denominator in (5) turned out to be zero. Even for only few such occurrences the algorithm diverged. An adjustment in the likelihood and transition density implementations could circumvent these issues. Also, the use of wider tailed distributions for v reduced such effects. Turning to the FFBSi, an advantage is that the number of backwards particles M can be flexibly adjusted to reduce computational load (KNM operations, about KN in its fast implementation). For $M = 100$, the FFBSi computations were

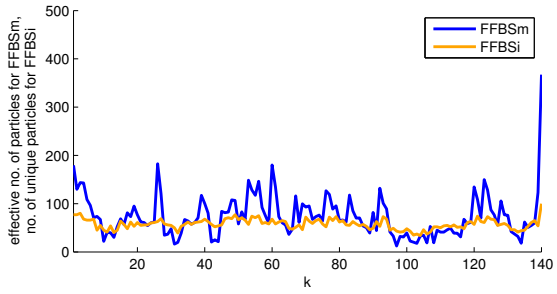


Figure 9. Effective number of particles for FFBSm, $N = 500$; number of unique particles for FFBSi, $N = 500$, $M = 100$.

much quicker than those of FFBSm and the performance as good, even in terms of particle diversity. A quantity that gives an indication of the diversity of unweighted samples in FFBSi is the number of unique particles and also illustrated in Figure 9.

The experiments show that the lookahead provided by smoothing can help resolve ambiguous situations. On average, however, no larger improvements of the PF output was achieved by smoothing. This is because PF already predicts its samples based on the motion model and inclusion of the transition densities provides little extra information for simple scenarios as for instance motion on a straight road. When compared to manually projected position data, both FFBSm and FFBSi are outperformed by a simple FL algorithm that, compared to PF, comes at no extra computational load. The manually projected data are, however, not necessarily the true trajectory that has been actually traveled – the error performance should thus be interpreted with care. If a smoothed velocity estimate is desired, the advanced smoothers present a more realistic picture as PF and FL tend to overestimate the target speed. From a computational point of view, FFBSi is to be preferred over FFBSm.

VI. Conclusion

We have shown how particle methods can be applied to obtain accurate position estimates comparable to ground truth data. We developed an advanced on-road motion model that can easily be extended to account for complex road network topologies. Furthermore, a way how the corresponding state transition density can be evaluated has been shown. GPS data were processed using a particle filter and smoothing algorithms were subsequently applied to the filter output. In ambiguous road topologies smoothing yields improved results. A fixed lag smoother was shown to be sufficient for position estimation, but overestimates the target speed. Improved velocity estimates were obtained by employing advanced smoothing algorithms (FFBSm and FFBSi). Among the two schemes, the particle forward filter backward simulator (FFBSi) seems to be the better choice justified by its computational complexity and flexibility (M not necessarily equal to N). The algorithms were successfully applied to real data.

Acknowledgment

This work has been supported by the MC Impulse (Monte Carlo based Innovative Management and Processing for an Unrivaled Leap in Sensor Exploitation) project, a European Commission FP7 Marie Curie Initial Training Network.

References

[1] F. Gustafsson, U. Orguner, T. B. Schon, P. Skoglar, and R. Karlsson, "Navigation and tracking of road-bound vehicles," in *Handbook of Intelligent Vehicles*, A. Eskandarian, Ed. Springer, Mar. 2012.

[2] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Artech House, Jan. 2004.

[3] U. Orguner, T. B. Schon, and F. Gustafsson, "Improved target tracking with road network information," in *2009 IEEE Aerospace conference*. IEEE, Mar. 2009, pp. 1–11.

[4] N. J. Gordon, D. J. Salmond, and A. F. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *Radar and Signal Processing, IEE Proceedings F*, vol. 140, no. 2, pp. 107–113, Apr. 1993.

[5] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, Feb. 2002.

[6] O. Cappe, S. J. Godsill, and E. Moulines, "An overview of existing methods and recent advances in sequential Monte Carlo," *Proceedings of the IEEE*, vol. 95, no. 5, pp. 899–924, May 2007.

[7] F. Gustafsson, "Particle filter theory and practice with positioning applications," *Aerospace and Electronic Systems Magazine, IEEE*, vol. 25, no. 7, pp. 53–82, 2010.

[8] ——, *Statistical Sensor Fusion*. Studentlitteratur AB, Mar. 2010.

[9] F. Lindsten, "Rao-Blackwellised particle methods for inference and identification," Licentiate Thesis no. 1480, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, Jun. 2011.

[10] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, 2000.

[11] S. J. Godsill, A. Doucet, and M. West, "Monte Carlo smoothing for nonlinear time series," *Journal of the American Statistical Association*, vol. 99, no. 465, pp. 156–168, 2004.

[12] R. Douc, "Sequential Monte Carlo smoothing for general state space hidden markov models," *The Annals of Applied Probability*, vol. 21, no. 6, pp. 2109–2145, Dec. 2011.

[13] H. Driessen and Y. Boers, "MAP estimation in particle filter tracking," in *2008 IET Seminar on Target Tracking and Data Fusion: Algorithms and Applications*. IET, Apr. 2008, pp. 41–45.

[14] M. Ulmke and W. Koch, "Road-map assisted ground moving target tracking," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 42, no. 4, pp. 1264–1274, Oct. 2006.

[15] A. Bondy and U. Murty, *Graph Theory*, 3rd ed. Springer, Aug. 2008.

[16] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*, 1st ed. Wiley-Interscience, Jun. 2001.

[17] S. Kotz and S. Nadarajah, *Multivariate t Distributions and their Applications*. Cambridge University Press, Feb. 2004.

[18] D. Streller, "Road map assisted ground target tracking," in *2008 11th International Conference on Information Fusion*. IEEE, Jul. 2008.

The one step fixed-lag particle smoother as a strategy to improve the prediction step of particle filtering

Samuel Nyobe, Fabien F. Campillo, Serge Moto, Vivien Rossi

The one step fixed-lag particle smoother as a strategy to improve the prediction step of particle filtering

S. Nyobe*, F. Campillo†, S. Moto‡, V. Rossi§

December 3, 2021

Abstract

Sequential Monte Carlo methods have been a major breakthrough in the field of numerical signal processing for stochastic dynamical state-space systems with partial and noisy observations. However, these methods still present certain weaknesses. One of the most fundamental is the degeneracy of the filter due to the impoverishment of the particles: the prediction step allows the particles to explore the state-space and can lead to the impoverishment of the particles if this exploration is poorly conducted or when it conflicts with the following observation that will be used in the evaluation of the likelihood of each particle. In this article, in order to improve this last step within the framework of the classic bootstrap particle filter, we propose a simple approximation of the one step fixed-lag smoother. At each time iteration, we propose to perform additional simulations during the prediction step in order to improve the likelihood of the selected particles.

Keywords: particle filter, bootstrap particle filter, one step fixed-lag particle smoother, prediction step, extended Kalman filter, unscented Kalman filter.

1 Introduction

Since the 1980s, sequential Monte Carlo (SMC) methods, also called particle filter (PF) methods [11, 6, 2, 7, 4], have met with vast success and incredible

*MIBA research Unity, Faculty of Science, University of Yaoundé 1, P.O. Box 812 Yaoundé, Cameroon, UMI 209, UMMISCO-Cameroon, IRD, Sorbonne University, P.O. Box 93143 Bondy, France Cedex, samuel.nyobe@facsciences-uy1.cm

†Inria, MathNeuro Team, Montpellier, France, fabien.campillo@inria.fr

‡MIBA research Unity, Faculty of Science, University of Yaoundé 1, P.O. Box 812 Yaoundé, Cameroon, UMI 209, UMMISCO-Cameroon, IRD, Sorbonne University, P.O. Box 93143 Bondy, France Cedex, serge-constant.moto@facsciences-uy1.cm

§RU Forets and Societies, CIRAD, Yaoundé, Cameroon, Computer Engineering Dpt, National Advanced School of Engineering, University of Yaoundé 1, P.O. Box 8390, Yaoundé, Cameroon vivien.rossi@cirad.fr

expansion in the context of problems of filtering for hidden Markov models (HMM), also called nonlinear filtering for state-space models with partial and noisy observations. The success of these methods is due to their ability to take into account, in a numerically realistic and efficient way, the non-linearity of the dynamics and the non-Gaussianity of the underlying conditional distributions.

The exact (i.e., non approximated) dynamics of these HMMs takes the form of a sequential Bayes formula represented in Eqs. (3)-(4) also called Bayesian filter. In the case of linear models with Gaussian additive noises, the Kalman filter makes it possible to calculate the exact solution. Otherwise, except for few particular models, it is not possible to calculate the exact solution, it is necessary to calculate approximations. The first approximation methods, the extended Kalman filter (EKF) [1] and variants, consisted in linearizing the models then in applying the Kalman filter. But in the case of strongly nonlinear models, the EKF often diverges.

Since the 1980s, Monte Carlo methods have become very effective alternatives. Monte Carlo methods are now recognized as a powerful tool in estimation of Bayesian filter in nonlinear/non-Gaussian hidden Markov models.

Among them, the PF techniques rely on an online importance sampling approximation of the sequential Bayes formula (3)-(4). Indeed, at each iteration, the PF builds of set of particles i.e. an independently and identically distributed sample from an approximation of the theoretical solution of the Bayesian filter.

An iteration of the FP classically takes place in two steps: the prediction step, which explores the state space by moving the particles according to the equation of state, followed by the correction step consisting on the one hand in weighting the particles according to their correspondence with the new observation, i.e. according to their likelihood, and on the other hand in resampling the particles according to these weights. These two steps can be understood respectively as mutation and selection steps of genetic algorithms.

The first really efficient PF algorithm, namely the bootstrap particle filter (BPF) or sampling-importance-resampling filter proposed in 1993 [11], follows exactly these two steps.

The resampling step is essential, indeed without it we observe an impoverishment of the particles in a few iterations: a few particles, even only one, will concentrate all the likelihood; all the other particles thus becoming useless, the filter then loses the "track" of the current state.

However, despite this resampling step particle filters often suffer from another type of particle degeneration [2, 18, 7, 22, 16]. This occurs when the particles explore areas of the state space that are not in correspondence with the new observation. This is due to the fact that in its classical version, especially in the case of BPF, the prediction step propagates the particles in the state space without taking into account the next observation. At the time of the weighting via the likelihood, a very large part of the particles are associated with an almost zero weight, this phenomenon is aggravated by the machine epsilon. In the worst case, all the particles can be associated with a zero weight, the filter has then totally lost the track of the state evolution.

To overcome this problem it is relevant to take into account the observations

in the prediction step consisting in exploring the state space. One can consult the review articles [9, 10] concerning the possibilities of improvement of particle filters.

The present paper aims to tackle the problem of degeneracy. We propose a new PF algorithm, called predictive bootstrap particle smoother (PBPS), to further improve the prediction step but keeping the simplicity of the BF. The principle of the PBPS is to take into account the current and the next observations for the prediction and correction steps. At each iteration during the prediction step, we perform additional predictions for assessing particle likelihood with the next observation. It can be seen as a simple approximation of the one step fixed-lag smoother. The additional predictions contribute to the correction step in order to improve the likelihood of the selected particles with the current and next observation.

We assess the performance of the PBPS by comparisons on simulation studies with the extended Kalman filter (EKF) [1], the unscented Kalman filter (UKF) [21, 13], and the bootstrap particle filter (BPF) [11]. The performance criterium is the accuracy of the estimates versus the computation time. For simulation studies we consider two nonlinear models: a classical one-dimensional one and a four-dimensional bearings-only tracking one.

2 Problem statement

2.1 The state space model

We consider a Markovian state-space model with state process $(X_k)_{k\geq 0}$ taking values in $\mathbb{R}^n$ and observation process $(Y_k)_{k\geq 1}$ taking values in $\mathbb{R}^d$. We suppose that conditionally on $(X_k)_{k\geq 0}$, the observations Y_k are independent.

The ingredients of the state-space model are:

$$
\begin{aligned}
q_k(x|x') &\stackrel{\text{def}}{=} p_{X_k|X_{k-1}=x'}(x)\,, && \text{(state transition kernel)}\\
\psi_k(x|y) &\stackrel{\text{def}}{=} p_{Y_k|X_k=x}(y)\,, && \text{(local likelihood function)}\\
\mu_0(x) &\stackrel{\text{def}}{=} p_{X_0}(x)\,, && \text{(initial distribution)}
\end{aligned}
$$

for any $x, x' \in \mathbb{R}^n$, $y \in \mathbb{R}^d$.

These ingredients can be made explicit by considering for example the following state space model:

$$X_k = f_{k-1}(X_{k-1}) + g_{k-1}(X_{k-1})\, W_{k-1}\,, \tag{1}$$

$$Y_k = h_k(X_k) + V_k\,, \tag{2}$$

for $1 \leq k \leq K$, where X_k (resp. Y_k, W_k, V_k) takes values in $\mathbb{R}^n$ (resp. $\mathbb{R}^d$, $\mathbb{R}^m$, $\mathbb{R}^d$), f_k (resp. g_k, h_k) is a differentiable and at most linear growth, uniformly in k, from $\mathbb{R}^n$ to $\mathbb{R}^n$ (resp. $\mathbb{R}^d$, $\mathbb{R}^{n\times m}$, $\mathbb{R}^d$), g_k being also bounded. Random

sequences W_k and V_k are independently and identically distributed centered white Gaussian noises; W_k, V_k, X_0 being independent. In this case:

$$q_k(x|x') = \frac{\exp\Big(-\frac{1}{2}\big[x - f_{k-1}(x')\big]^* \big[g_k(x')\, Q_{k-1}\, g_k(x')^*\big]^{-1} \big[x - f_{k-1}(x')\big]\Big)}{\sqrt{(2\,\pi)^n\ \det Q_{k-1}}}$$

and

$$\psi_k(x|y) \propto \exp\Big(-\frac{1}{2}\big[y - h_k(x)\big]^* R_k^{-1} \big[y - h_k(x)\big]\Big)$$

(ψ_k has to be known up to a multiplicative constant).

2.2 Nonlinear filtering

Nonlinear filtering aims at determining the conditional distribution η_k of the current state X_k given the past observations $Y_1, \ldots, Y_k$, namely:

$$\eta_k(x) \stackrel{\text{def}}{=} p_{X_k|Y_{1:k}=y_{1:k}}(x)\,,\ x \in \mathbb{R}^n$$

for any $k \geq 1$ and $y_{1:k} \in (\mathbb{R}^d)^k$, here we use the notation:

$$\text{“}Y_{1:k}\text{” for } (Y_1, \ldots, Y_k)$$

(e.g. $Y_{1:k} = y_{1:k}$ means $Y_\ell = y_\ell$ for all $\ell = 1, \ldots, k$).

The nonlinear filter allows to determine η_k from η_{k-1} using the classical two-step recursive Bayes formula:

Prediction step. The predicted distribution $\eta_{k^-}(x) \stackrel{\text{def}}{=} p_{X_k|Y_{1:k-1}=y_{1:k-1}}(x)$ of X_k given $Y_{1:k-1} = y_{1:k-1}$ is given by:

$$\eta_{k^-}(x) = \int_{\mathbb{R}^n} q_k(x|x')\ \eta_{k-1}(x')\ \mathrm{d}x'\,, \quad x \in \mathbb{R}^n\,. \tag{3}$$

Correction step. The new observation $Y_k = y_k$ allows to update the predicted distribution in order to obtain η_k according to the Bayes formula:

$$\eta_k(x) = \frac{\psi_k(x|y_k)\,\eta_{k^-}(x)}{\int_{\mathbb{R}^n} \psi_k(x'|y_k)\,\eta_{k^-}(x')\,\mathrm{d}x'}\,, \quad x \in \mathbb{R}^n\,. \tag{4}$$

Note that in the correction step (4) , η_k is proportional to the product of η_{k-1} and the local likelihood function, that is:

$$\eta_k(x) \propto \psi_k(x|y_k)\,\eta_{k^-}(x)\,.$$

2.3 Nonlinear smoothing

To get $\bar{\eta}_{k-1}(x)$ the conditional distribution of X_{k-1} given $Y_{0:k} = y_{0:k}$, we consider the extended state vector $\mathbf{X}_k = (X_k, X_{k-1})$, it's a Markov with transition:

$$\begin{aligned} Q_k(x', x''|x_{k-1}, x_{k-2}) &= p_{X_k, X_{k-1}|X_{k-1}=x_{k-1}, X_{k-2}=x_{k-2}}(x', x'') \\ &= p_{X_k|X_{k-1}=x'', X_{k-1}=x_{k-1}, X_{k-2}=x_{k-2}}(x')\; p_{X_{k-1}|X_{k-1}=x_{k-1}, X_{k-2}=x_{k-2}}(x'') \\ &= p_{X_k|X_{k-1}=x_{k-1}, X_{k-2}=x_{k-2}}(x')\; \delta_{x_{k-1}}(x'') \\ &= p_{X_k|X_{k-1}=x_{k-1}}(x')\; \delta_{x_{k-1}}(x'') \\ &= q_k(x'|x_{k-1})\; \delta_{x_{k-1}}(x'')\,. \end{aligned} \tag{5}$$

The conditional distribution $\boldsymbol{\eta}_k(x', x'')$ of $\mathbf{X}_k = (X_k, X_{k-1})$ given $Y_{0:k} = y_{0:k}$, we apply the previous filter formula:

$$\begin{aligned} \boldsymbol{\eta}_{k^-}(x', x'') &= \iint Q_k(x', x''|x_{k-1}, x_{k-2})\, \boldsymbol{\eta}_{k-1}(x_{k-1}, x_{k-2})\, \mathrm{d}x_{k-1}\, \mathrm{d}x_{k-2} \\ &= \iint q_k(x'|x_{k-1})\, \delta_{x_{k-1}}(x'')\, \boldsymbol{\eta}_{k-1}(x_{k-1}, x_{k-2})\, \mathrm{d}x_{k-1}\, \mathrm{d}x_{k-2} \\ &= \iint q_k(x'|x_{k-1})\, \delta_{x_{k-1}}(x'')\, \boldsymbol{\eta}_{k-1}(x'', x_{k-2})\, \mathrm{d}x_{k-1}\, \mathrm{d}x_{k-2}\,, \end{aligned} \tag{6}$$

and the distribution of Y_k given $(X_k = x_k, X_{k-1} = x_{k-1})$ is the distribution of Y_k given $X_k = x_k$, hence:

$$\boldsymbol{\eta}_k(x', x'') \propto \psi_k(x'|y_k)\, \boldsymbol{\eta}_{k^-}(x', x'')\,, \tag{7}$$

which the x''-marginal distribution gives the conditional distribution of X_{k-1} given $Y_{0:k} = y_{0:k}$.

2.4 Approximations

The main difficulty encountered by the nonlinear filter (3)-(4) lies in the two integrations. These integrations can be solved explicitly in only in the linear/Gaussian case, leading to the Kalman filter, and in a very few other specific nonlinear/non-Gaussian cases. In the latter cases, the optimal filter can be solved *explicitly* in the form of a finite dimensional filter; hence in the vast majority of cases, it is necessary to use approximation techniques [4]. Among the approximation techniques, we will consider the extended Kalman filter (EKF) [1], the unscented Kalman filter (UKF) [21, 13] and particle filter techniques, also called sequential Monte Carlo techniques [11, 6], see [8] for a recent overview.

Concerning the particle filters, as we will see, it is important to notice that on the one hand we do not need to know the analytical expressions of the state transition kernel and of the initial distribution, we just need to be able to sample (efficiently) from them; on the other hand, we do need the analytical expression of the local likelihood function (up to a multiplicative constant), indeed, for a given y, we need to compute $\psi_k(x|y)$ for a very large number of x values.

3 Particle approximations

The particle approximation η_k^N of η_k is a Monte Carlo empirical approximation of the form:

$$\eta_k^N(x) = \sum_{i=1}^N \omega_k^i \, \delta_{\xi_k^i}(x) \,, \; x \in \mathbb{R}^n \,.$$

composed of N particles ξ_k^i in $\mathbb{R}^n$ and weights ω_k^i, the weights are positive and sum to one [6]. Ideally the particles are sampled from η_k and the importance weight are all equal to $1/N$. *For the sake of notation simplicity, we now omit the superscript N in η_k^N.*

3.1 The bootstrap particle filter

The bootstrap particle filter (BPF) filter is a classical sequential importance resampling method: suppose we have a good approximation $\eta_{k-1} = \frac{1}{N} \sum_{i=1}^N \delta_{\xi_{k-1}^i}$ of η_{k-1}. We can apply the prediction step (3) to η_{k-1} and get:

$$\int_{\mathbb{R}^n} q_k(x|x') \; \eta_{k-1}(x') \, \mathrm{d}x' = \sum_{i=1}^N \omega_{k-1}^i \; q_k(x|\xi_{k-1}^i)$$

and then apply the correction step (4) and get:

$$\sum_{i=1}^N \omega_{k-1}^i \, \psi_k(x|y_k) \, q_k(x|\xi_{k-1}^i) \; \Big/ \int_{\mathbb{R}^n} \sum_{i=1}^N \omega_{k-1}^i \, \psi_k(x'|y_k) \, q_k(x'|\xi_{k-1}^i) \, \mathrm{d}x' \,.$$

Both the two last expressions are mixture of the densities $q_k(\,\cdot\,|\xi_{k-1}^i)$ and therefor not of the particle type. The bootstrap particle filter (BPF) proposed by [11] is the simplest method to propose a particle approximation. For the prediction step we use the sampling technique:

$$\xi_{k^-}^i \sim q_k(\,\cdot\,|\xi_{k-1}^i) \,, \; i = 1 : N \quad \text{(independently)} \tag{8}$$

which is:

$$\xi_{k^-}^i = f_{k-1}(\xi_{k-1}^i) + g_{k-1}(\xi_{k-1}^i) \, w^i$$

where w^i are i.i.d. $N(0, R_{k-1})$ samples. Then we let

$$\eta_{k^-}(x) \stackrel{\text{def}}{=} \sum_{i=1}^N \omega_{k-1}^i \, \delta_{\xi_{k^-}^i}(x) \,. \tag{9}$$

Through the correction step (4), this approximation η_{k^-} gives $\sum_{i=1}^N \omega_k^i \, \delta_{\xi_{k^-}^i}$ where:

$$\omega_k^i \stackrel{\text{def}}{=} \frac{\psi_k(\xi_{k^-}^i|y_k) \, \omega_{k^-}^i}{\sum_{j=1}^N \psi_k(\xi_{k^-}^j|y_k) \, \omega_{k^-}^j} \tag{10}$$

are the updated weight taking account of the new observation y_k through the likelihood function ψ_{k,y_k}. This correction step *must* be completed by a resampling of the particles $\xi_{k^-}^i$ according to the importance weights ω_k^i:

$$\xi_k^i \sim \sum_{j=1}^{N} \omega_k^j \, \delta_{\xi_{k^-}^j} \,, \; i = 1:N \quad \text{(independently)} \tag{11}$$

leading the to the particle approximation:

$$\eta_k(x) \stackrel{\text{def}}{=} \sum_{i=1}^{N} \omega_k^i \, \delta_{\xi_k^i}(x) \,, \text{ with } \omega_k^i = \frac{1}{N} \,. \tag{12}$$

Algorithm 1 gives a summary of the BPF, in this version a resampling is performed at each time step. The multinomial resampling step (11), the basic idea proposed in [11], could be deeply improved, there are several resampling techniques, see [5, 15] for more details.

1: $\xi_0^i \overset{\text{iid}}{\sim} \mu_0(\,\cdot\,) \,, \; i = 1:N$ `# initialization`
2: **return** $\xi_0^{1:N}$
3: **for** $k = 1:K$ **do**
4: $\quad \xi_{k^-}^i \sim q_k(\,\cdot\,|\xi_{k-1}^i) \,, \; i = 1:N$ `# evolution of particles`
5: $\quad \omega_k^i \leftarrow \psi_k(\xi_{k^-}^i|y_k) \,, \; i = 1:N$ `# likelihood`
6: $\quad \omega_k^i \leftarrow \omega_k^i / \sum_{j=1}^N \omega_k^j \,, \; i = 1:N$ `# reweighting`
7: $\quad \xi_k^i \overset{\text{iid}}{\sim} \sum_{j=1}^N \omega_k^j \, \delta_{\xi_{k^-}^j} \,, \; i = 1:N$ `# resampling`
8: $\quad$ **return** $\xi_k^{1:N}$
9: **end for**

Algorithm 1: Bootstrap particle filter (BPF).

It is not necessary to resample the particles at each time iteration like in Algorithm 1. However, this resampling step should be done regularly in terms of time iterations to avoid degeneracy of the weights [6].

We will look at another degeneracy problem. Suppose that in step (10), the local likelihood values $\psi_k(\xi_{k^-}^i|y_k)$ associated with the predicted particles $\xi_{k^-}^i$ are all very small, or even equal to zero due to rounding in floating point arithmetic. This normalization step (10) is then impossible. This problem occurs when the filter "loses track" of the true state X_k, i.e. the likelihood of the predicted particules $\xi_{k^-}^i$ w.r.t. the observation y_k are all negligible, in this case the observation y_k appears as an outlier. This occurs especially when the period of time between two successive instants of observation is very large compared to the dynamics of the state process. Strategies to the overcome that weakness encompass the iterated extended Kalman particle filter [17] or the iterated unscented Kalman particle filter [12]; see [10] and [16] for reviews of the subject.

3.2 The predictive bootstrap particle smoother

The purpose of the predictive bootstrap filter (PBPS) is to improve the correction step (10) of the BPF by using both $Y_k = y_k$ and $Y_{k+1} = y_{k+1}$ at each iteration k. In a way, the PBPS can be seen as an approximation of the distribution of X_k given $Y_{1:k+1} = y_{1:k+1}$, the one step fixed-lag smoother presented in Section 2.3.

In the proposed algorithm, the prediction step consists first in propagating the N particles of η_{k-1} to time k to get η_{k^-}; second, for each one of these N particles propagating, one-step-ahead at time $k+1$, an offspring particle. Then the correction step consists in updating the weights of the N particles of η_{k^-} according to their likelihood with y_k and the likelihood of their one-step ahead offspring particles with y_{k+1}.

The iteration $k-1 \to k$ of the filter is more precisely:

Prediction step. Like for the BPF we sample N particles and compute N normalized likelihood weights:

$$\tilde{\xi}_k^i \sim q_k(\,\cdot\,|\xi_{k-1}^i)\,, \qquad \tilde{\omega}_k^i \propto \psi_k(\xi_{k^-}^i|y_k)\,\omega_{k^-}^i\,, \qquad i = 1:N\,.$$

Again we will resample the particles $\tilde{\xi}_k^{1:N}$, but instead of doing so according to the weights $\tilde{\omega}_k^{1:N}$, we will first modify the latter ones. For each index i, we generate a one-step-ahead offspring particles and compute its weight:

$$\tilde{\xi}_{k+1}^i \sim \tilde{q}_{k+1}(\,\cdot\,|\tilde{\xi}_k^i)\,, \qquad \tilde{\omega}_{k+1}^i \stackrel{\text{def}}{=} \psi_{k+1}(\tilde{\xi}_{k+1}^i|y_{k+1}).$$

Note that the likelihood weights $\tilde{\omega}_{k+1}^i$ depend on the next observation y_{k+1}. See later for the choice of one-step-ahead sampler $\tilde{q}_{k+1}$.

Correction step. We compute the weights at time k according to Eq. (7):

$$\omega_k^i \stackrel{\text{def}}{=} \tilde{\omega}_k^i\;\tilde{\omega}_{k+1}^i \qquad \text{for } i = 1:N. \tag{13}$$

Then, N particles $\tilde{\xi}_k^{1:N}$ are resampled according to the weights $\omega_k^{1:N}$.

The PBPS is depicted in Algorithm 2.

Choice of one-step-ahead sampler $\tilde{q}_{k+1}$. For the special case of the system (1)-(2) we can choose a simpler "deterministic sampler":

$$\tilde{\xi}_{k+1}^i = f_k(\tilde{\xi}_k^i)$$

which corresponds to the one-step-ahead mean:

$$\begin{aligned}\tilde{\xi}_{k+1}^i &= \mathbb{E}\big(X_{k+1}\big|X_k = \tilde{\xi}_k^i\big)\\ &= \mathbb{E}\big(f_k(X_k) + g_k(X_k)\,W_k\big|X_k = \tilde{\xi}_k^i\big)\\ &= f_k(\tilde{\xi}_k^i) + g_k(\tilde{\xi}_k^i)\,\underbrace{\mathbb{E}(W_k|X_k = \tilde{\xi}_k^i)}_{=0} = f_k(\tilde{\xi}_k^i)\end{aligned}$$

This choice greatly reduces the computation burden and it is enough, as we will see, in linear state equation. For highly nonlinear state equation, we can choose $\tilde{q}_{k+1} = q_{k+1}$, which corresponds to simulate the state dynamic.

$\xi_0^i \overset{\text{iid}}{\sim} \mu(\,\cdot\,),\ i = 1:N$ `# initialization`

for $k = 1:K$ **do**

  for $i = 1:N$ **do**

    $\tilde{\xi}_k^i \sim q_k(\,\cdot\,|\xi_{k-1}^i)$ `# particles propagation`

    $\tilde{\omega}_k^i \leftarrow \psi_k(\tilde{\xi}_k^i|y_k)$

    $\tilde{\xi}_{k+1}^i = f_k(\tilde{\xi}_k^i)$ `# offspring particles generating`

    $\tilde{\omega}_{k+1}^i \leftarrow \psi_{k+1}(\tilde{\xi}_{k+1}^i|y_{k+1})$ `# offspring particles weighting`

    $\omega_k^i \leftarrow \tilde{\omega}_k^i\,\tilde{\omega}_{k+1}^i$ `# particles weighting`

  end for

  $\omega_k^i \leftarrow \omega_k^i / \sum_{i'=1}^N \omega_k^{i'}\ ,\ i = 1:N$ `# weights normalization`

  $\xi_k^i \overset{\text{iid}}{\approx} \sum_{i'=1}^N \omega_k^{i'}\,\delta_{\tilde{\xi}_k^{i'}}\ ,\ i = 1:N$ `# particles resampling`

  return $\xi_k^{1:N}$

end for

Algorithm 2: predictive bootstrap particle smoother (PBPS).

4 Simulation studies

We compare numerically the PBPS to other filters on two space-state models described below. The filters performance is assessed trough Monte Carlo scheme by a criteria based on mean squared error between the state trajectory and the filter estimates. We assess the performance versus computational time by testing several numbers of particles.

4.1 Space-state models simulated

First case study: one-dimensional model

We consider the following one-dimensional nonlinear model [11, 14, 6]:

$$\begin{aligned} X_k &= \frac{1}{2}\,X_{k-1} + \frac{25\,X_{k-1}}{1+X_{k-1}^2} + 8\cos\big(1.2\,(k-1)\big) + W_{k-1}\,, \\ Y_k &= \frac{X_k^2}{20} + V_k\,, \end{aligned} \tag{14}$$

with $1 \leq k \leq K$ $(K = 50)$, $X_0 \sim \mathcal{N}(0,1)$; $W_k \overset{\text{iid}}{\sim} \mathcal{N}(0,3^2)$, $V_k \overset{\text{iid}}{\sim} \mathcal{N}(0,1)$; X_0, $(W_k)_{k\geq 1}$ and $(V_k)_{k\geq 1}$ mutually independent. Note that the state process X_k is observed only through X_k^2 so the filters have difficulties to determine whether X_k is positive or negative, especially since the state process X_k regularly changes of sign. Thus this model is regularly used as benchmark for testing filters, as filters may easily lose track of X_k.

Second case study: a four-dimensional bearings-only tracking model

We consider the following four-dimensional model [11, 19, 3]:

$$\begin{aligned} X_k &= \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} X_{k-1} + \sigma_W \begin{pmatrix} 0.5 & 0 \\ 0 & 0.5 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} W_{k-1}\,, \\ Y_k &\sim \text{wrapped Cauchy}\Big(\arctan\Big(\frac{X_k[1]}{X_k[2]}\Big), \rho\Big), \end{aligned} \tag{15}$$

with $1 \leq k \leq K = 20$, $X_0 \sim \mathcal{N}(\bar{X}_0, P_0)$ with:

$$\bar{X}_0 = \big(-0.05, 0.2, 0.001, -0.055\big)^*\,, \quad P_0 = 0.01\,\mathrm{diag}\big(0.5^2, 0.3^2, 0.005^2, 0.01^2\big)\,,$$

$\sigma_W = 0.001$, $W_k \overset{\text{iid}}{\sim} \mathcal{N}(0, I_{2\times 2})$, $\rho = 1 - 0.005^2$, X_0 and $(W_k)_{k\geq 1}$ mutually independent; conditionally on $(X_k)_{k\geq 0}$, $(Y_k)_{k\geq 1}$ and $(W_k)_{k\geq 1}$ are independent.

The state equation corresponds to a target moving on a plane. The state vector is:

$$X_k = (X_k[1], X_k[2], X_k[3], X_k[4])^* = (x_1, x_2, \dot{x}_1, \dot{x}_2)^*\,,$$

where (x_1, x_2) are the Cartesian coordinates of the target in the plane and $(\dot{x}_1, \dot{x}_2)$ are the corresponding velocities. The observer is located at the origin of the plane and accessed only to the azimuth angle $\beta = \arctan(x_1/x_2) \in [-\pi, \pi]$ corrupted by noise.

The conditional probability density function of the measured angle Y_k given the state X_k is assumed to be a wrapped Cauchy distribution with concentration parameter ρ [19]:

$$p_{Y_k|X_k=x}(y) = \frac{1}{2\,\pi}\,\frac{1-\rho^2}{1+\rho^2-2\,\rho\,\cos\Big(y - \arctan(\frac{x[1]}{x[2]}))\Big)}\,, \quad -\pi \leq y < \pi\,, \tag{16}$$

where $\rho \in [0, 1]$ is the mean resultant length. Note that the state dynamics is linear and Gaussian, but the observation dynamics is nonlinear and non-Gaussian.

4.2 Performance criteria

For both case studies, we compare the filters performances through a Monte Carlo scheme. We simulate S independent trajectories $(X^{(s)}_{0:K}, Y^{(s)}_{1:K})_{s=1:S}$ of the state-space models. For each simulation s, we ran R times each particle filter $\mathcal{F} \in \{\text{BPF, PBPS}\}$ and we compute the root mean squared error at time k:

$$\mathrm{RMSE}_k(\mathcal{F}) \overset{\text{def}}{=} \frac{1}{S}\sum_{s=1}^{S}\sqrt{\frac{1}{R}\sum_{r=1}^{R}\big|\hat{X}_k^{\mathcal{F}(s,r)} - X_k^{(s)}\big|^2}\,, \tag{17}$$

where:

$$\hat{X}_k^{\mathcal{F}(s,r)} \overset{\text{def}}{=} \int_{\mathbb{R}^n} x\,\eta_k^{\mathcal{F}(s,r)}(x)\,\mathrm{d}x = \frac{1}{N}\sum_{i=1}^{N}\xi_k^{\mathcal{F}(s,r),i}$$

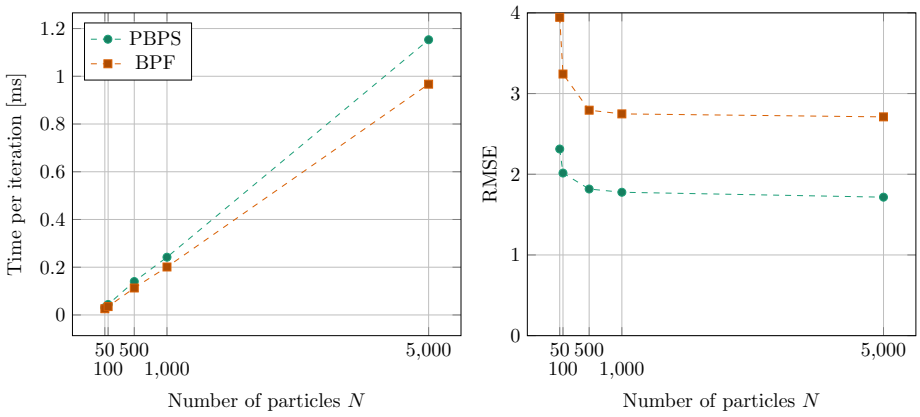


Figure 1: One-dimensional case study (14) — We compare the performances of the PBPS and the BPF with 5 different values of N: 50, 100, 500, 1,000, and 5,000. We plot the average computation time per iteration (left) and RMSE (right) with $S = 100$ and $R = 40$. We can see that PBPS is clearly superior to BPF. For example, the PBPS with 50 particles is significantly faster than the BPF with 5000 particles while giving better results.

is the numerical approximation of $\hat{X}_k^{(s)} = \mathbb{E}(X_k^{(s)}|Y_{1:k}^{(s)})$ by the filter $\mathcal{F}$. We also compute the global root mean squared error:

$$\mathrm{RMSE}(\mathcal{F}) \stackrel{\text{def}}{=} \frac{1}{K+1} \sum_{k=0}^{K} \mathrm{RMSE}_k(\mathcal{F})\,. \tag{18}$$

We proceed to the systematic resampling technique [5, 15] in all particle filters. To fairly compare the different particle filters, we compute the RMSE and the mean computational times of an iteration on the same simulation of states and observations trajectories.

All implementations are done in R language [20] using a 2.10 GHz core i3 intel running Linux Mint. 19.1 with a 8 Go RAM.

4.3 Simulation results

One-dimensional case study (14)

First we compare the BPF and the PBPS. In Fig. 1, we plot the average computation time per iteration (left) and the RMSE (right) with $S = 100$, $R = 40$ and for different values of N: 50, 100, 500, 1000 and 5000.

Compared to the BPF, and with the same number of particules, the PBPS requires a little more computation time (Fig. 1 left) but is significantly more accurate (Fig. 1 right).

For instance, the PBPS with $N = 50$ particles is more accurate than the BPF with $N = 5000$ particles while being significantly faster. Similarly, the PBPS with $N = 1000$ particles is 5 times faster than the BPF with $N = 5000$ particles, while being 30% better. The strategy proposed by the PBPS is very advantageous here, in particular due to the very nonlinear nature of this model for both the state equation and the observation equation.

Next, in Fig. 2, we simulate one trajectory of the state/observation process and we compare the BPF and the PBPS (with $N = 1000$) with the extended Kalman filter (EKF) and une unscented Kalman filter (UKF).

For each filter $\mathcal{F} \in \{\text{PBPS, BPF, EKF, UKF}\}$, we plot the true state trajectory $k \to X_k$, the approximation $k \to \hat{X}_k^{\mathcal{F}}$, and associated 95% confidence region. The PBPS approximation appears to have a smaller error $k \to |\hat{X}_k^{\mathcal{F}} - X_k|$ and a smaller confidence region than the other filters.

Due to the nature of the system, it is not surprising that the EKF has a very poor quality behavior. In particular, the EKF has a lot of difficulty to take into account the passages of the state variable by 0: once it has "chosen" to go to one side or the other of 0, it no longer has the possibility of changing sides even in the case where the sign of the state X_k is different. Let us note that the UKF, even if it is a little less precise compared to particle filters, allows to take into account the non-linearities of the system and is much faster than all the other filters.

Finally, in Fig. 3, we compare $k \to \text{RMSE}_k(\mathcal{F})$ for $\mathcal{F} \in \{\text{PBPS, BPF, EKF, UKF}\}$ (with $N = 1,000$). Note that for EKF and UKF, the summation on replicas r is useless. Once again, it can be seen that the PBPS filter behaves significantly better than the others. As said before the EKF filter behaves very poorly on this example, unlike the UKF which is closer to the behavior of particle filters. We note again the very bad behavior of the EKF and the relatively good behavior of the UKF, the latter remains less accurate than the BPF and the PBPS but requires much less computing time.

Four-dimensional case study (15)

Classically, in this case of study the trajectory of the target is not simulated according to the equation (15) but according to a uniform rectilinear motion or according to a uniform motion with some course changes. On the other hand, the filters follow the model (15): there is thus a mismatch between the simulation and the filter. This will allow us to test the robustness of the filters which is an essential property of the domain.

We run four simulations: one where the target stays in a straight line and one where the target changes course; and in each case the filter uses a value of σ_W small ($\sigma_W = 0.001$) and a value of σ_W large ($\sigma_W = 0.01$). The first value of 0.001 corresponds better to the target motion while the second value 0.01 is too large compared to the target trajectory.

In the case of Fig. 4 the target follows a uniform rectilinear motion and the

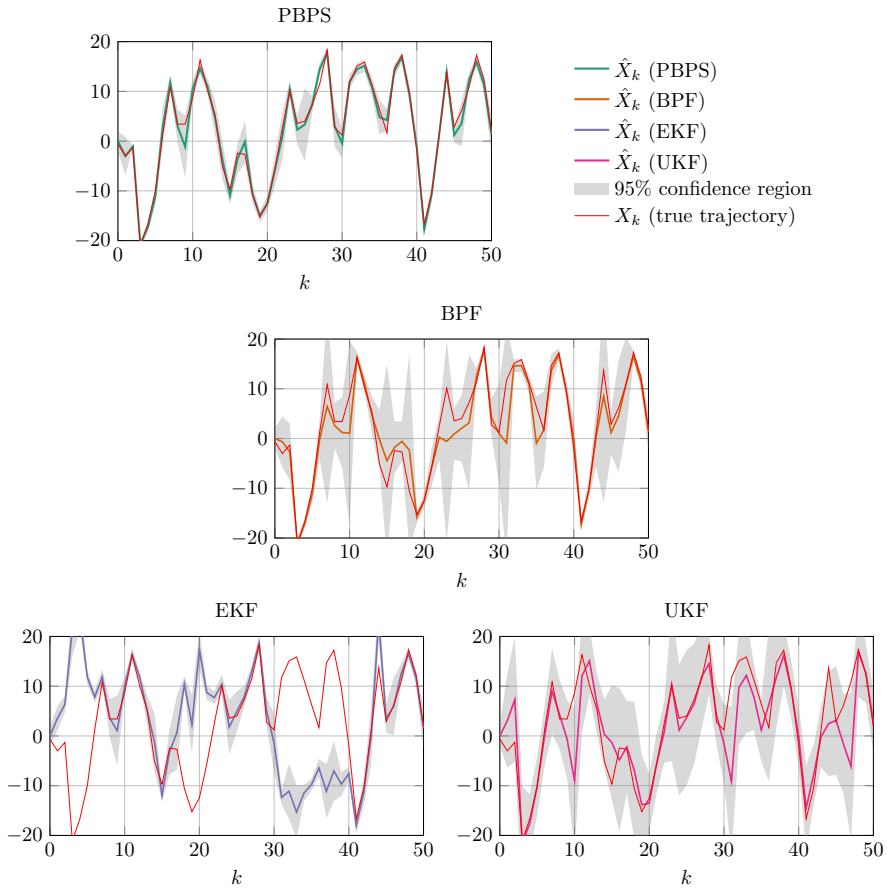


Figure 2: One-dimensional case study (14) — For each filter, with $N = 1000$, we plot the true state trajectory $k \to X_k$ (red), the approximation $k \to \hat{X}_k$, and the associated 95% confidence region (grey area). The PBPS approximation appears to have a smaller error $k \to \hat{X}_k - X_k$ and a smaller confidence region than the other filters.

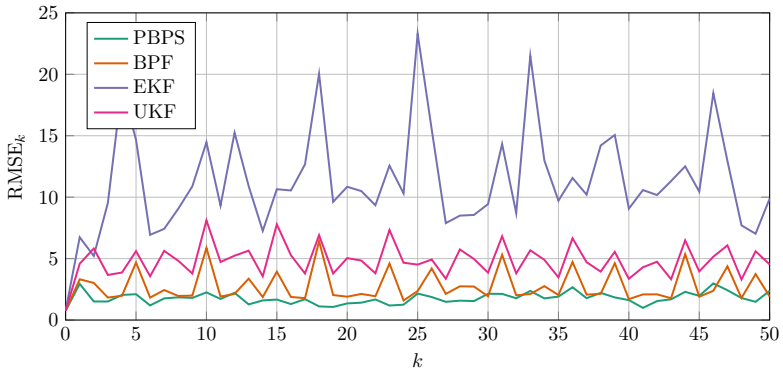


Figure 3: One-dimensional case study (14) — We compare $k \to \text{RMSE}_k$ ($S = 100$ and $R = 40$), for the PBPS ($N = 1000$), BPF ($N = 1000$), EKF, and the UKF (for the two latter filters, the summation on replicas r is useless).

filter assumes a small value of σ_W (0.001). The computation time per iteration according to the number of particles is presented in Fig. 4 (bottom left); the RMSE also according to the number of particles is presented in Fig. 4 (bottom right). The PBPS filter is slightly more computationally time consuming than the BPF filter; the accuracies are equivalent.

In Fig. 4 (top), we present an example of simulation of the trajectory and of the two filters: in addition to the real trajectory, we plot the trajectories estimated by each of the two filters, to which we associate the 95% uncertainty ellipses corresponding to the empirical covariance matrices of the position (x_1, x_2). These ellipses are very stretched in the direction of the line of sight represented by lines coming from the position $(0, 0)$ of the observer.

In the case of Fig. 5 the target follows a uniform rectilinear motion and the filter assumes a large value of σ_W (0.01). The computation time per iteration is equivalent to the previous case. On the other hand, in terms of RMSE, the PBPS performs much better than the BPF.

In Fig. 5 (top), we see that the BPF loses the target track while the PBPS is much more robust. Indeed, here the value $\sigma_W = 0.01$ used in the filter is too large compared to the real target track and the PBPS reacts much better to this model mismatch.

In the case of Fig. 6 the target follows a uniform rectilinear motion except for a single change of course and the filter assumes a small value of $sigma_W$ (0.001). The computation time per iteration is equivalent to the previous cases. In terms of RMSE, the two filters are equivalent.

In Fig. 6 (top), we see that both filters lose the target track. Indeed, the value $\sigma_W = 0.001$ is too small to account for the change in heading. A classical

idea is to increase this value in order to allow the filter to maintain the track. This is what we do in the next example.

In the case of Fig. 7 the target follows a uniform rectilinear motion except for a single change of course and the filter assumes a large value of σ_W (0.01). The computation time per iteration is equivalent to the previous cases. On the other hand, in terms of RMSE the PBPS performs much better than the BPF.

In Fig. 7 (top), we see that the BPF loses the target track while the PBPS is clearly more robust.

In conclusion, for a slightly longer computation time, the PBPS filter is clearly more robust than the BPF in terms of assumptions on the target trajectory, which is particularly sought after in this type of application.

5 Discussion and conclusion

For the first example presented, at an equivalent level of accuracy and computation time, the PBPS requires significantly fewer particles. With the same number of particles, there is a strong reduction of the empirical variance of the error, this is also due to the fact that the PBPS is a smoother which uses one more observation than the BPF which is a filter; if this is done with a slight increase of the computation time per iteration for the same number of particles, the PBPS is still much more efficient than the BPF since it requires much less particles for an equivalent level of accuracy. For the second example, in the case where the BPF works well, the PBPS has an equivalent level of accuracy for a slightly higher computation time. On the other hand, in cases where the BPF loses the target trajectory, the PBPS does not lose it and has a much higher accuracy for a slightly longer computation time.

When the equation of state is strongly nonlinear, as in Example 1, PBPS is significantly more accurate than BPF with half as many particles. For example 2, it is easy to adjust the variance of the noise on the dynamics of the PBPS so that it supports a change in the course of the target. Whereas for the BPF we were not able to find a variance value that would allow it to withstand a change in course. So the example 2 illustrates the greater robustness of the SPBF compared to the BPF when a nonlinearity occurs on the state dynamics.

We introduced an improved version of the BPF, the *predictive bootstrap particle smoother* (PBPS), for nonlinear state-space models, which presents a markedly improved behavior. Although the PBPS relied on an observation horizon of one-step forward, for systems with nonlinear state dynamics it produced more accurate estimates than other filters for a significantly lower computation time. Beyond the specific interest of PBPS, our results demonstrated that it is possible to build particle filtering algorithms based on step forward horizons with competitive computation time.

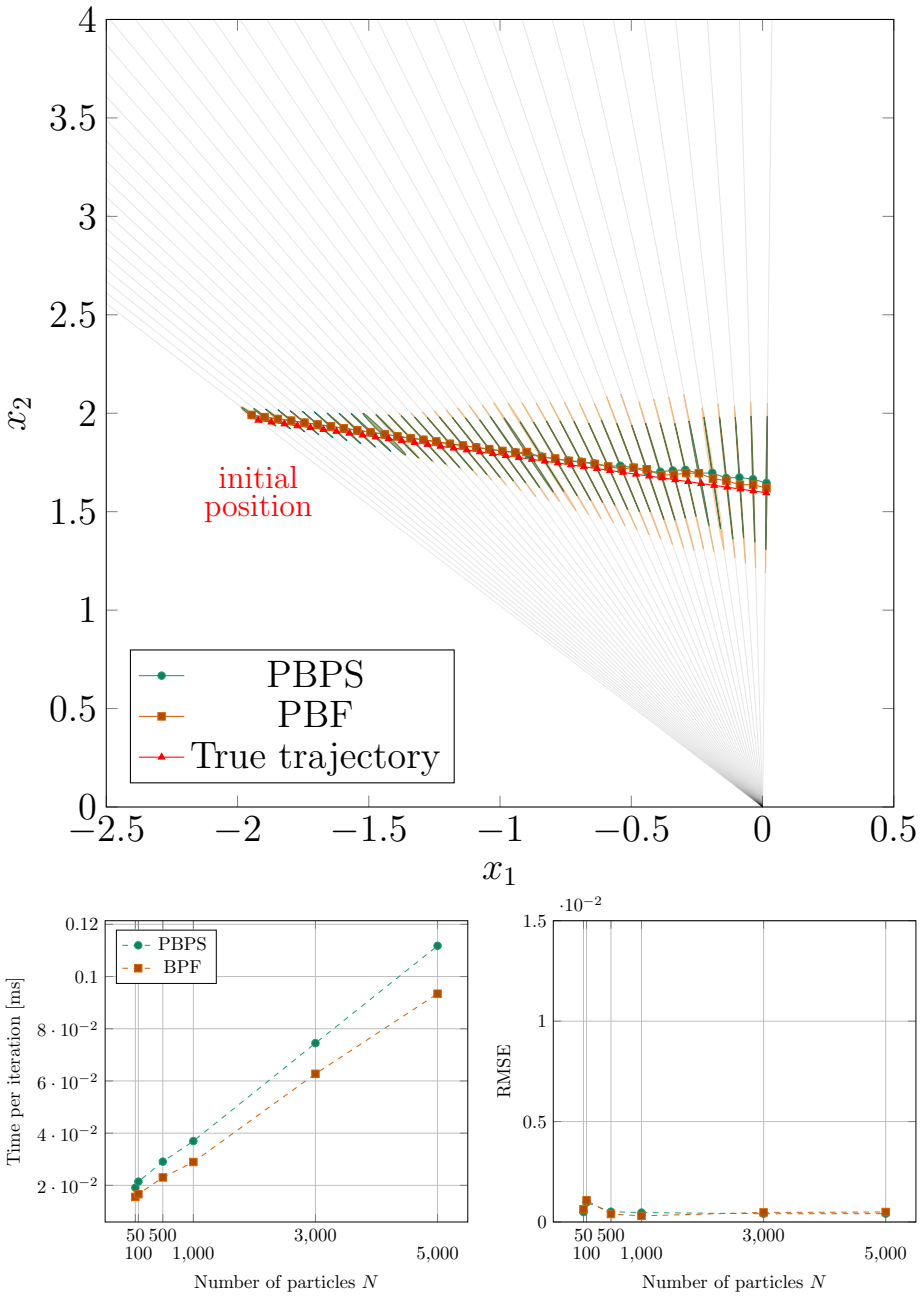


Figure 4: Four-dimensional case study (15) — The target follows a uniform rectilinear motion and the filter assumes a small value of $\sigma_W = 0.001$.

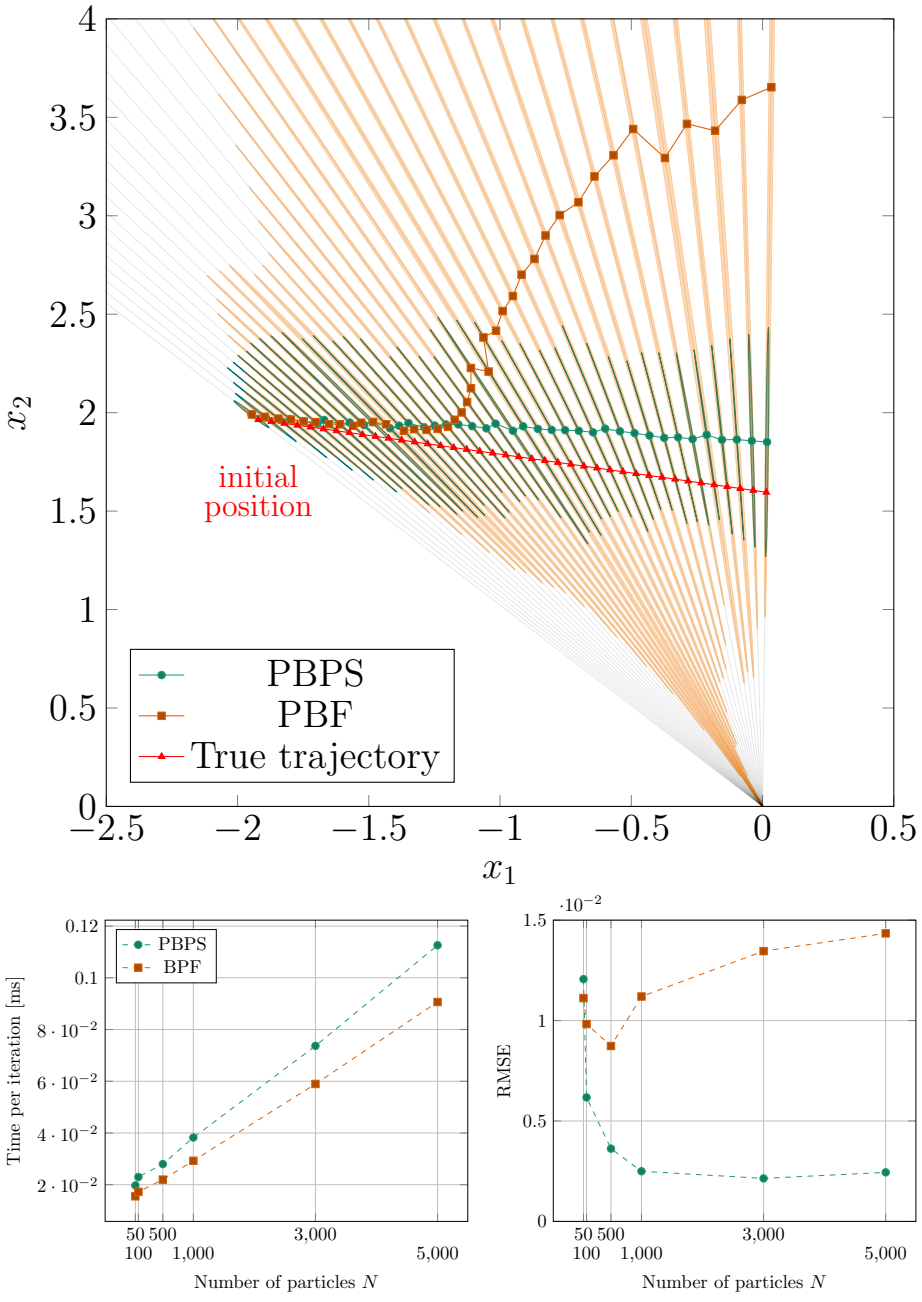


Figure 5: Four-dimensional case study (15) — The target follows a uniform rectilinear motion and the filter assumes a large value of $\sigma_W = 0.01$.

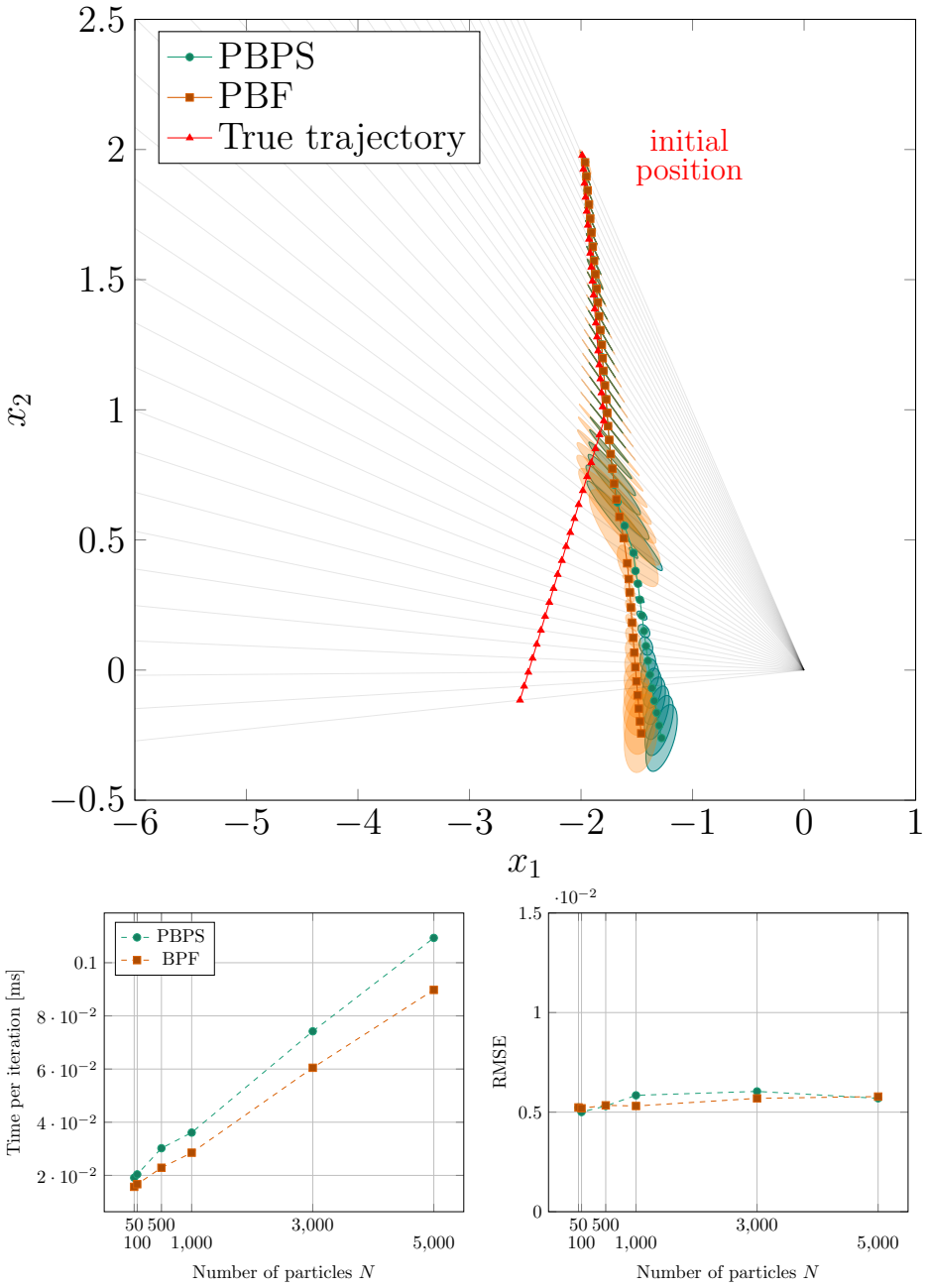


Figure 6: Four-dimensional case study (15) — The target follows a uniform rectilinear motion with a single change in the course and the filter assumes a small value of $\sigma_W = 0.001$.

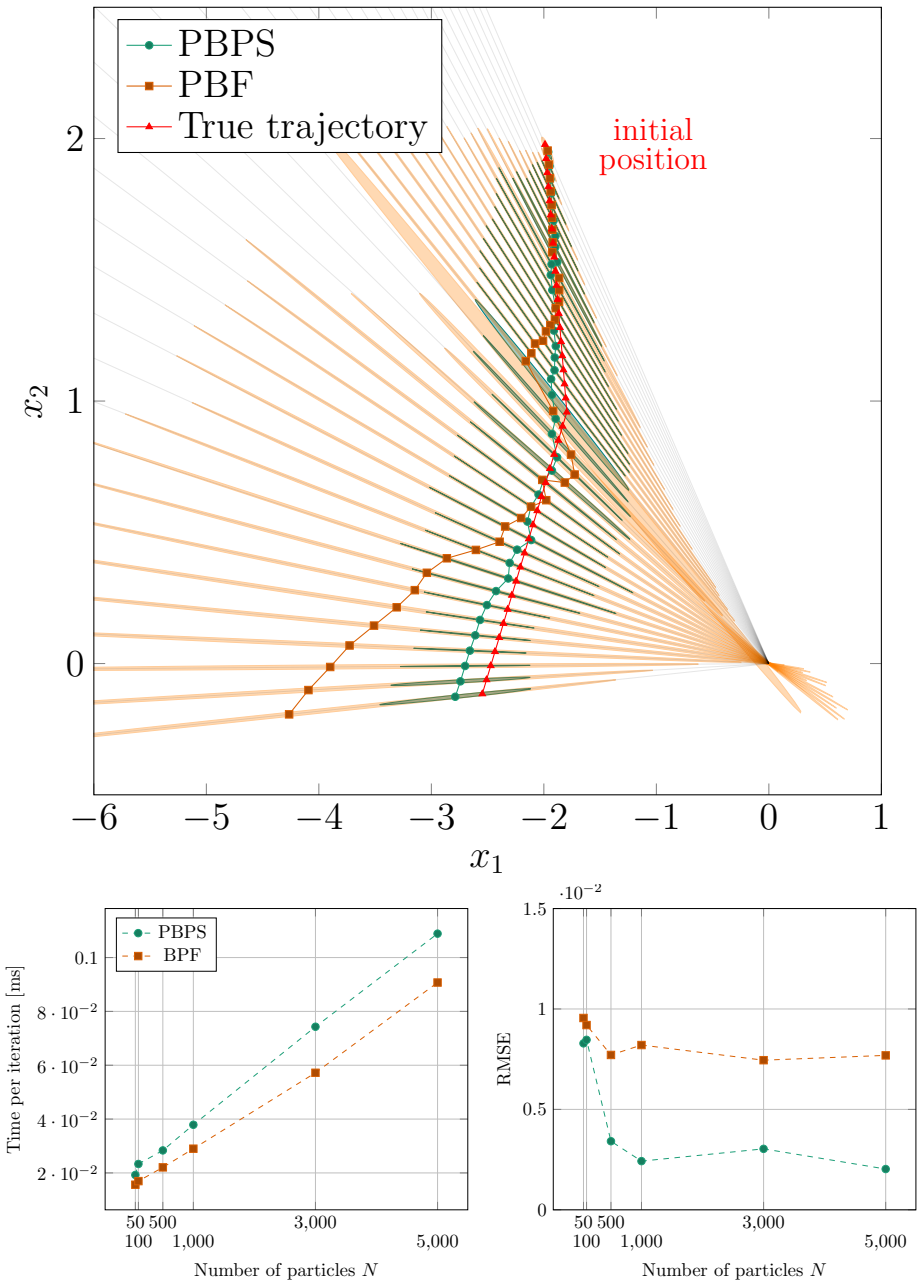


Figure 7: Four-dimensional case study (15) — The target follows a uniform rectilinear motion with a single change in the course and the filter assumes a large value of $\sigma_W = 0.01$.

References

[1] B. D. Anderson and J. B. Moore. *Optimal Filtering*. Prentice-Hall, 1979.

[2] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002.

[3] F. Campillo and V. Rossi. Convolution particle filter for parameter estimation in general state-space models. *IEEE Transactions on Aerospace and Electronic Systems*, 45(3):1063–1072, 2009.

[4] D. Crisan and B. Rozovskii, editors. *The Oxford Handbook of Nonlinear Filtering*. Oxford University Press, 2011.

[5] R. Douc and O. Cappé. Comparison of resampling schemes for particle filtering. In *Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis, ISPA*, 2005.

[6] A. Doucet, N. de Freitas, and N. J. Gordon, editors. *Sequential Monte Carlo Methods in Practice*. Springer New York, 2001.

[7] A. Doucet and A. M. Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. In *[4]*, 2011.

[8] A. Doucet and A. Lee. Sequential Monte Carlo methods. In M. Maathuis, M. Drton, S. Lauritzen, and M. Wainwright, editors, *Handbook of Graphical Models*. Chapman & Hall/CRC, 2018.

[9] S. Godsill. Particle filtering: the first 25 years and beyond. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7760–7764. IEEE, 2019.

[10] S. J. Godsill and T. Clapp. Improvement strategies for Monte Carlo particle filters. In *[6]*, pages 139–158, 2001.

[11] N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non–Gaussian Bayesian state estimation. *IEEE Proceedings F Radar and Signal Processing*, 140(2):107–113, 1993.

[12] W. Guo, C. Han, and M. Lei. Improved unscented particle filter for nonlinear Bayesian estimation. In *10th International Conference on Information Fusion*, 2007.

[13] S. J. Julier and J. K. Uhlmann. Unscented filtering and nonlinear estimation. *IEEE Review*, 92(3):401–422, 2004.

[14] G. Kitagawa. Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, 5(1):1–25, 1996.

[15] T. Li, M. Bolic, and P. M. Djuric. Resampling methods for particle filtering: Classification, implementation, and strategies. *IEEE Signal Processing Magazine*, 32(3):70–86, 2015.

[16] T. Li, S. Sun, T. P. Sattar, and J. M. Corchado. Fight sample degeneracy and impoverishment in particle filters: A review of intelligent approaches. *Expert Systems with Applications*, 41(8):3944–3954, 2014.

[17] L. Liang-Qun, J. Hong-Bing, and L. Jun-Hui. The iterated extended Kalman particle filter. In *IEEE International Symposium on Communications and Information Technology*, volume 2, pages 1213–1216, 2005.

[18] S. Park, J. P. Hwang, E. Kim, and H.-J. Kang. A new evolutionary particle filter for the prevention of sample impoverishment. *IEEE Transactions on Evolutionary Computation*, 13(4):801–809, 2009.

[19] M. K. Pitt and N. Shephard. Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, 94(446):590–599, 1999.

[20] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2021.

[21] E. A. Wan and R. van der Merwe. The unscented Kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium*, pages 153–158, 2000.

[22] J. Zuo. Dynamic resampling for alleviating sample impoverishment of particle filter. *IET Radar, Sonar Navigation*, 7(9):968–977, 2013.