5

Sequential Monte Carlo methods

In Chapter 2 we introduced the filtering recursion for general state space models (Proposition 2.1). The recursive nature of the algorithm that, from the filtering distribution at time $t-1$ and the observation y_t computes the filtering distribution at time t, makes it ideally suited for a large class of applications in which inference must be made online, before the data collection ends. For those types of applications one must have, at any time, an up-to-date estimate of the current state of the system. Standard examples of such online types of applications include the following: tracking the position and speed of a moving aircraft observed through a radar; monitoring the location and characteristics of a storm based on satellite data; estimating the volatility of the prices of a group of stocks from tick-to-tick data. Unfortunately, for a general state space model the integrations in (2.7) cannot be carried out analytically. DLMs are a special case for which the Kalman filter gives a closed form solution to the filtering problem. However, even in this case, as soon as a DLM contains unknown parameters in its specification, the Kalman filter alone is not enough to compute the filtering distribution and, except in a few simple cases (see Section 4.3.1) one has to resort to numerical techniques.

For off-line, or batch, inference MCMC methods can be successfully employed for DLMs with unknown parameters, as explained in Chapter 4, and can be extended to nonlinear non-Gaussian models. However, they are of limited use for online inference because any time a new observation becomes available, a totally new Markov chain has to be simulated. In other words, in an MCMC approach, the output based on $t-1$ observations cannot be used to evaluate posterior distributions based on t observations. In this sense, unlike the Kalman filter, MCMC does not lend itself easily to a sequential usage.

Early attempts to sequentially update filtering distributions for nonlinear and non-Gaussian state space models were based on some form of linearization of the state and system equations and on Gaussian approximations (for details and references, see Cappé et al.; 2007). While useful for mildly nonlinear models, an approach of this type typically performs poorly for highly nonlinear

G. Petris et al., *Dynamic Linear Models with R*, Use R, DOI: b135794_5,

models. Furthermore, it does not solve the problem of sequentially estimating unknown parameters, even in the simple DLM case.

In this chapter we give an account of a relatively recent simulation approach, called Sequential Monte Carlo, that has proved very successful in online filtering applications of both DLMs with unknown parameters and general nonlinear non-Gaussian state space models. Sequential Monte Carlo provides an alternative set of simulation-based algorithms to approximate complicated posterior distributions. Although not limited to time series models, it has proved extremely successful when applied to DLMs and more general state space models—especially in those applications that require frequent updates of the posterior as new data are observed. Research in sequential Monte Carlo methods is currently very active and we will not try to give here an exhaustive review of the field. Instead, we limit ourselves to a general introduction and a more specific description of a few algorithms that can be easily implemented in the context of DLMs. For more information the interested reader can consult the books by Liu (2001), Doucet et al. (2001), Del Moral (2004), and Cappé et al. (2005). The article by Cappé et al. (2007) provides a current overview of the field.

5.1 The basic particle filter

Particle filtering, which is how sequential Monte Carlo is usually referred to in applications to state space models, is easier to understand when viewed as an extension of importance sampling. For this reason we open this section with a brief recall of importance sampling.

Suppose one is interested in evaluating the expected value

$$\mathrm{E}_\pi(f(X)) = \int f(x)\pi(x)\,\mathrm{d}x. \tag{5.1}$$

If g is an *importance density* having the property that $g(x) = 0$ implies $\pi(x) = 0$, then one can write

$$\mathrm{E}_\pi(f(X)) = \int f(x)\frac{\pi(x)}{g(x)}g(x)\,\mathrm{d}x = \mathrm{E}_g(f(X)w^\star(X)),$$

where $w^\star(x) = \pi(x)/g(x)$ is the so-called *importance function*. This suggests approximating the expected value of interest by generating a random sample of size N from g and computing

$$\frac{1}{N}\sum_{i=1}^{N} f(x^{(i)})w^\star(x^{(i)}) \approx \mathrm{E}_\pi(f(X)). \tag{5.2}$$

In Bayesian applications one can typically evaluate the target density only up to a normalizing factor, i.e., only $C \cdot \pi(x)$ can be computed, for an unknown

constant C. Unfortunately, this implies that also the importance function can only be evaluated up to the same factor C and (5.2) cannot be used directly. However, letting $\tilde{w}^{(i)} = Cw^\star(x^{(i)})$, if one takes $f(x) \equiv C$, then (5.2) yields

$$\frac{1}{N}\sum_{i=1}^{N} Cw^\star(x^{(i)}) = \frac{1}{N}\sum_{i=1}^{N} \tilde{w}^{(i)} \approx \mathrm{E}_\pi(C) = C. \tag{5.3}$$

Since the $\tilde{w}^{(i)}$'s are available, (5.3) provides a way of evaluating C. Moreover, for the purpose of evaluating (5.1) one does not need an explicit estimate of the constant C: in fact,

$$\begin{aligned}\mathrm{E}_\pi(f(X)) &\approx \frac{1}{N}\sum_{i=1}^{N} f(x^{(i)})w^\star(x^{(i)}) \\ &= \frac{\frac{1}{N}\sum_{i=1}^{N} f(x^{(i)})\tilde{w}^{(i)}}{C} \approx \frac{\sum_{i=1}^{N} f(x^{(i)})\tilde{w}^{(i)}}{\sum_{i=1}^{N}\tilde{w}^{(i)}} \\ &= \sum_{i=1}^{N} f(x^{(i)})w^{(i)},\end{aligned}$$

with $w^{(i)} = \tilde{w}^{(i)}/\sum_{j=1}^{N}\tilde{w}^{(j)}$. Note that: (1) the weights $w^{(i)}$ sum to one, and (2) the approximation $\mathrm{E}_\pi(f(X)) \approx \sum_{i=1}^{N} f(x^{(i)})w^{(i)}$ holds for every well-behaved function f. Therefore, the sample $x^{(1)}, \ldots, x^{(N)}$ with the associated weights $w^{(1)}, \ldots, w^{(N)}$ can be viewed as a discrete approximation of the target π. In other words, writing δ_x for the unit mass at x, and setting $\hat{\pi} = \sum_{i=1}^{N} w^{(i)}\delta_{x^{(i)}}$, one has $\pi \approx \hat{\pi}$.

In filtering applications, the target distribution changes every time a new observation is made, moving from $\pi(\theta_{0:t-1}|y_{1:t-1})$ to $\pi(\theta_{0:t}|y_{1:t})$. Note that the former is not a marginal distribution of the latter, even though $\theta_{0:t-1}$ are the first components of $\theta_{0:t}$. The problem then is how to efficiently update a discrete approximation of $\pi(\theta_{0:t-1}|y_{1:t-1})$ when the observation y_t becomes available, in order to obtain a discrete approximation of $\pi(\theta_{0:t}|y_{1:t})$. For every s, let us denote[1] by $\hat{\pi}_s(\theta_{0:s}|y_{1:s})$ the approximation of $\pi(\theta_{0:s}|y_{1:s})$. The updating process consists of two steps: for each point $\theta_{0:t-1}^{(i)}$ in the support of $\hat{\pi}_{t-1}$, (1) draw an additional component $\theta_t^{(i)}$ to obtain $\theta_{0:t}^{(i)}$ and, (2) update its weight $w_{t-1}^{(i)}$ to an appropriate $w_t^{(i)}$. The weighted points $(\theta_t^{(i)}, w_t^{(i)})$, $i = 1, \ldots, N$, provide the new discrete approximation $\hat{\pi}_t$. For every t, let g_t be the importance density used to generate $\theta_{0:t}$. Since at time t the observations $y_{1:t}$ are available, g_t may depend on them and we will write $g_t(\theta_{0:t}|y_{1:t})$ to make the dependence explicit. We assume that g_t can be expressed in the following form:

[1] We keep the index s in the notation $\hat{\pi}_s$ because approximations at different times can be in principle unrelated to one another, while the targets are all derived from the unique distribution of the process $\{\theta_i, y_j : i \geq 0, j \geq 1\}$.

$$g_t(\theta_{0:t}|y_{1:t}) = g_{t|t-1}(\theta_t|\theta_{0:t-1}, y_{1:t}) \cdot g_{t-1}(\theta_{0:t-1}|y_{1:t-1}).$$

This allows us to "grow" sequentially $\theta_{0:t}$ by combining $\theta_{0:t-1}$, drawn from g_{t-1} and available at time $t-1$, and θ_t, generated at time t from $g_{t|t-1}(\theta_t|\theta_{0:t-1}, y_{1:t})$. We will call the functions $g_{t|t-1}$ *importance transition densities*. Note that only the importance transition densities are needed to generate $\theta_{0:t}$. Suggestions about the selection of the importance density are provided at the end of the section. Let us consider next how to update the weights. One has, dropping the superscripts for notational simplicity:

$$\begin{aligned} w_t &\propto \frac{\pi(\theta_{0:t}|y_{1:t})}{g_t(\theta_{0:t}|y_{1:t})} \propto \frac{\pi(\theta_{0:t}, y_t|y_{1:t-1})}{g_t(\theta_{0:t}|y_{1:t})} \\ &\propto \frac{\pi(\theta_t, y_t|\theta_{0:t-1}, y_{1:t-1}) \cdot \pi(\theta_{0:t-1}|y_{1:t-1})}{g_{t|t-1}(\theta_t|\theta_{0:t-1}, y_{1:t}) \cdot g_{t-1}(\theta_{0:t-1}|y_{1:t-1})} \\ &\propto \frac{\pi(y_t|\theta_t) \cdot \pi(\theta_t|\theta_{t-1})}{g_{t|t-1}(\theta_t|\theta_{0:t-1}, y_{1:t})} \cdot w_{t-1}. \end{aligned}$$

Hence, for every i, after drawing $\theta_t^{(i)}$ from $g_{t|t-1}(\theta_t|\theta_{0:t-1}^{(i)}, y_{1:t})$, one can compute the unnormalized weight $\tilde{w}_t^{(i)}$ as

$$\tilde{w}_t^{(i)} = w_{t-1}^{(i)} \cdot \frac{\pi(y_t|\theta_t^{(i)}) \cdot \pi(\theta_t^{(i)}|\theta_{t-1}^{(i)})}{g_{t|t-1}(\theta_t^{(i)}|\theta_{0:t-1}^{(i)}, y_{1:t})}. \tag{5.4}$$

The fraction on the left-hand side of equation (5.4), or any quantity proportional[2] to it, is called the *incremental weight*. The final step in the updating process consists in scaling the unnormalized weights:

$$w_t^{(i)} = \frac{\tilde{w}_t^{(i)}}{\sum_{j=1}^N \tilde{w}_t^{(j)}}.$$

In practice it is often the case that, after a number of updates have been performed, a few points in the support of $\hat{\pi}_t$ have relatively large weights, while all the remaining have negligible weights. This clearly leads to a deterioration in the Monte Carlo approximation. To keep this phenomenon in control, a useful criterion to monitor over time is the *effective sample size*, defined as

$$N_{\mathit{eff}} = \left(\sum_{i=1}^N \left(w_t^{(i)}\right)^2 \right)^{-1},$$

which ranges between N (when all the weights are equal) and one (when one weight is equal to one). When N_{eff} falls below a threshold N_0, it is advisable

[2] The proportionality constant may depend on $y_{1:t}$, but should not depend on $\theta_t^{(i)}$ or $\theta_{0:t-1}^{(i)}$ for any i.

0. Initialize: draw $\theta_0^{(1)}, \ldots, \theta_0^{(N)}$ independently from $\pi(\theta_0)$ and set

$$w_0^{(i)} = N^{-1}, \qquad i = 1, \ldots, N.$$

1. For $t = 1, \ldots, T$:

1.1) For $i = 1, \ldots, N$:

- Draw $\theta_t^{(i)}$ from $g_{t|t-1}(\theta_t | \theta_{0:t-1}^{(i)}, y_{1:t})$ and set

$$\theta_{0:t}^{(i)} = (\theta_{0:t-1}^{(i)}, \theta_t^{(i)})$$

.

- Set

$$\tilde{w}_t^{(i)} = w_{t-1}^{(i)} \cdot \frac{\pi(\theta_t^{(i)}, y_t | \theta_{t-1}^{(i)})}{g_{t|t-1}(\theta_t^{(i)} | \theta_{0:t-1}^{(i)}, y_{1:t})}.$$

1.2) Normalize the weights:

$$w_t^{(i)} = \frac{\tilde{w}_t^{(i)}}{\sum_{j=1}^N \tilde{w}_t^{(j)}}.$$

1.3) Compute

$$N_{\mathit{eff}} = \left(\sum_{i=1}^N \big(w_t^{(i)}\big)^2 \right)^{-1}.$$

1.4) If $N_{\mathit{eff}} < N_0$, resample:

- Draw a sample of size N from the discrete distribution

$$\mathrm{P}\big(\theta_{0:t} = \theta_{0:t}^{(i)}\big) = w_t^{(i)}, \qquad i = 1, \ldots, N,$$

and relabel this sample

$$\theta_{0:t}^{(1)}, \ldots, \theta_{0:t}^{(N)}.$$

- Reset the weights: $w_t^{(i)} = N^{-1}$, $i = 1, \ldots, N$.

1.5) Set $\hat{\pi}_t = \sum_{i=1}^N w_t^{(i)} \delta_{\theta_{0:t}^{(i)}}$.

Algorithm 5.1: Summary of the particle filter algorithm

to perform a *resampling* step. This can be done in several different ways. The simplest, called multinomial resampling, consists of drawing a random sample of size N from $\hat{\pi}_t$ and using the sampled points, with equal weights, as the new discrete approximation of the target. The resampling step does not change the expected value of the approximating distribution $\hat{\pi}_t$, but it increases its Monte Carlo variance. In trying to keep the variance increase as small as possible, researchers have developed other resampling algorithms, more efficient than multinomial resampling in this respect. Of these, one of the most commonly used is residual resampling. It consists of creating, for

$i = 1, \ldots, N$, $\lfloor N w_t^{(i)} \rfloor$ copies of $\theta_{0:t}^{(i)}$ deterministically first, and then adding R_i copies of $\theta_{0:t}^{(i)}$, where $(R_1, \ldots, R_N)$ is a random vector having a multinomial distribution. The size and probability parameters are given by $N - M$ and $(\bar{w}^{(1)}, \ldots, \bar{w}^{(N)})$ respectively, where

$$M = \sum_{i=1}^{N} \lfloor N w_t^{(i)} \rfloor,$$
$$\bar{w}^{(i)} = \frac{N w_t^{(i)} - \lfloor N w_t^{(i)} \rfloor}{N - M}, \qquad i = 1, \ldots, N.$$

Algorithm 5.1 contains a summary of the basic particle filter. Let us stress once again the sequential character of the algorithm. Each pass of the outermost "for" loop represents the updating from $\hat{\pi}_{t-1}$ to $\hat{\pi}_t$ following the observation of the new data point y_t. Therefore, at any time $t \leq T$ one has a working approximation $\hat{\pi}_t$ of the current filtering distribution.

At time t, a discrete approximation of the filtering distribution $\pi(\theta_t | y_{0:t})$ is immediately obtained as a marginal distribution of $\hat{\pi}_t$. More specifically, if $\hat{\pi}_t = \sum_{i=1}^{N} w^{(i)} \delta_{\theta_{0:t}^{(i)}}$, we only need to discard the first t components of each path $\theta_{0:t}^{(i)}$, leaving only $\theta_t^{(i)}$, to obtain

$$\pi(\theta_t | y_{1:t}) \approx \sum_{i=1}^{N} w^{(i)} \delta_{\theta_t^{(i)}}.$$

As a matter of fact, particle filter is most frequently viewed, as the name itself suggests, as an algorithm to update sequentially the filtering distribution. Note that, as long as the transition densities $g_{t|t-1}$ are Markovian, the incremental weights in (5.4) only depend on $\theta_t^{(i)}$ and $\theta_{t-1}^{(i)}$, so that, if the user is only interested in the filtering distribution, the previous components of the path $\theta_{0:t}^{(i)}$ can be safely discarded. This clearly translates into substantial savings in terms of storage space. Another, more fundamental, reason to focus on the filtering distribution is that the discrete approximation provided by $\hat{\pi}_t$ is likely to be more accurate for the most recent components of $\theta_{0:t}$ than for the initial ones. To see why this is the case, consider, for a fixed $s < t$, that the $\theta_s^{(i)}$'s are generated at a time when only $y_{0:s}$ is available, so that they may well be far from the center of their smoothing distribution $\pi(\theta_s | y_{0:t})$, which is conditional on $t - s$ additional observations.

We conclude this section with practical guidelines to follow in the selection of the importance transition densities. In the context of DLMs, as well as for more general state space models, two are the most used importance transition densities. The first is $g_{t|t-1}(\theta_t | \theta_{0:t-1}, y_{1:t}) = \pi(\theta_t | \theta_{t-1})$, i.e., the actual transition density of the Markov chain of the states. It is clear that in

this way all the particles are drawn from the prior distribution of the states, without accounting for any information provided by the observations. The simulation of the particles and the calculation of the incremental weights are straightforward. However, most of the times the generated particles will fall in regions of low posterior density. The consequence will be an inaccurate discrete representation of the posterior density and a high Monte Carlo variance for the estimated posterior expected values. For these reasons we discourage the use of the prior as importance density. A more efficient approach, which accounts for the observations in the importance transition densities, consists of generating θ_t from its conditional distribution given θ_{t-1} and y_t. This distribution is sometimes referred to as the *optimal importance kernel*. In view of the conditional independence structure of the model, this is the same as the conditional distribution of θ_t given $\theta_{0:t-1}$ and $y_{1:t}$. Therefore, in this way one is generating θ_t from the target (conditional) distribution. However, since θ_{t-1} was not drawn from the current target, the particles $\theta_{0:t}^{(i)}$ are not draws from the target distribution[3] and the incremental importance weights need to be evaluated. Applying standard results about Normal models, it is easily seen that for a DLM the optimal importance kernel $g_{t|t-1}$ is a Normal density with mean and variance given by

$$\begin{aligned} \mathrm{E}(\theta_t|\theta_{t-1}, y_t) &= G_t\theta_{t-1} + W_t F_t' \Sigma_t^{-1}(y_t - F_t G_g \theta_{t-1}), \\ \mathrm{Var}(\theta_t|\theta_{t-1}, y_t) &= W_t - W_t F_t' \Sigma_t^{-1} F_t W_t, \end{aligned}$$

where $\Sigma_t = F_t W_t F_t' + V_t$. Note that for time-invariant DLMs the conditional variance above does not depend on t and can therefore be computed once and for all at the beginning of the process. The incremental weights, using this importance transition density, are proportional to the conditional density of y_t given $\theta_{t-1} = \theta_{t-1}^{(i)}$, i.e., to the $\mathcal{N}(F_t G_t \theta_{t-1}^{(i)}, \Sigma_t)$ density, evaluated at y_t.

5.1.1 A simple example

To illustrate the practical usage of the basic particle filter described in the previous section and to assess its accuracy, we present here a very simple example based on 100 observations simulated from a known DLM. The data are generated from a local level model with system variance $W = 1$, observation variance $V = 2$, and initial state distribution $\mathcal{N}(10, 9)$. We save the observations in `y`. Note the use of `dlmForecast` to simulate from a given model.

R code

```
> ### Generate data
> mod <- dlmModPoly(1, dV = 2, dW = 1, m0 = 10, C0 = 9)
```

[3] The reason for this apparent paradox is that the target distribution changes from time $t-1$ to time t. When one generates θ_{t-1}, the observation y_t is not used.

```
> n <- 100
> set.seed(23)
> simData <- dlmForecast(mod = mod, nAhead = n, sampleNew = 1)
> y <- simData$newObs[[1]]
```

In our implementation of the particle filter we set the number of particles to 1000, and we use the optimal importance kernel as importance transition density. As discussed in the previous section, for a DLM this density is easy to use both in terms of generating from it and for updating the particle weights. To keep things simple, instead of the more efficient residual resampling, we use plain multinomial resampling, setting the threshold for a resampling step to 500—that is, whenever the effective sample size drops below one half of the number of particles, we resample.

R code

```
> ### Basic Particle Filter - optimal importance density
> N <- 1000
> N_0 <- N / 2
> pfOut <- matrix(NA_real_, n + 1, N)
> wt <- matrix(NA_real_, n + 1, N)
> importanceSd <- sqrt(drop(W(mod) - W(mod)^2 /
+                       (W(mod) + V(mod))))
> predSd <- sqrt(drop(W(mod) + V(mod)))
> ## Initialize sampling from the prior
> pfOut[1, ] <- rnorm(N, mean = m0(mod), sd = sqrt(C0(mod)))
> wt[1, ] <- rep(1/N, N)
> for (it in 2 : (n + 1))
+ {
+     ## generate particles
+     means <- pfOut[it - 1, ] + W(mod) *
+         (y[it - 1] - pfOut[it - 1, ]) / (W(mod) + V(mod))
+     pfOut[it, ] <- rnorm(N, mean = means, sd = importanceSd)
+     ## update the weights
+     wt[it, ] <- dnorm(y[it - 1], mean = pfOut[it - 1, ],
+                       sd = predSd) * wt[it - 1, ]
+     wt[it, ] <- wt[it, ] / sum(wt[it, ])
+     ## resample, if needed
+     N.eff <- 1 / crossprod(wt[it, ])
+     if ( N.eff < N_0 )
+     {
+         ## multinomial resampling
+         index <- sample(N, N, replace = TRUE, prob = wt[it, ])
+         pfOut[it, ] <- pfOut[it, index]
+         wt[it, ] <- 1 / N
```

```
+       }
+ }
```

For a completely specified DLM the Kalman filter can be used to derive exact filtering means and variances. In Figure 5.1 we compare the exact fil-

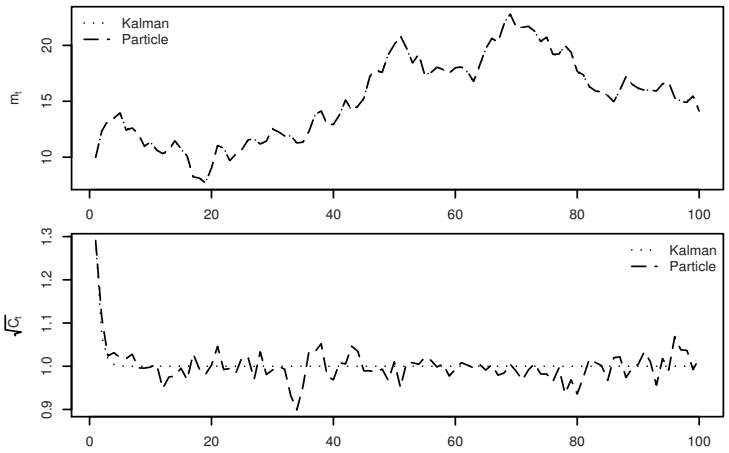

Fig. 5.1. Top: comparison of filtering state estimates computed with the Kalman filter and particle filter. Bottom: comparison of filtering standard deviations computed with the Kalman filter and particle filter

tering means and standard deviations, obtained using the Kalman filter, with the Monte Carlo approximations of the same quantities obtained using the particle filter algorithm. In terms of the filtering mean, the particle filter gives a very accurate approximation at any time (the two lines are barely distinguishable). The approximations to the filtering standard deviations are less precise, although reasonably close to the true values. The precision can be increased by increasing the number of particles in the simulation. The plots were obtained with the following code.

R code

```
> ## Compare exact filtering distribution with PF approximation
> modFilt <- dlmFilter(y, mod)
> thetaHatKF <- modFilt$m[-1]
> sdKF <- with(modFilt, sqrt(unlist(dlmSvd2var(U.C, D.C))))[-1]
> pfOut <- pfOut[-1, ]
> wt <- wt[-1, ]
> thetaHatPF <- sapply(1 : n, function(i)
+                      weighted.mean(pfOut[i, ], wt[i, ]))
> sdPF <- sapply(1 : n, function(i)
```

```
+                 sqrt(weighted.mean((pfOut[i, ] -
+                         thetaHatPF[i])^2, wt[i, ])))
> plot.ts(cbind(thetaHatKF, thetaHatPF),
+         plot.type = "s", lty = c("dotted", "longdash"),
+         xlab = "", ylab = expression(m[t]))
> legend("topleft", c("Kalman", "Particle"),
+        lty = c("dotted", "longdash"), bty = "n")
> plot.ts(cbind(sdKF, sdPF), plot.type = "s",
+         lty = c("dotted", "longdash"), xlab = "",
+         ylab = expression(sqrt(C[t])))
> legend("topright", c("Kalman", "Particle"),
+        lty = c("dotted", "longdash"), bty = "n")
```

5.2 Auxiliary particle filter

The particle filter described in the previous section applies to general state space models. However, its performance depends heavily on the specification of the importance transition densities. While for a DLM the optimal importance kernel can be obtained explicitely and its use typically provides fairly good approximations to the filtering distributions, for a general state space model this is not the case, and devising effective importance transition densities is a much harder problem. The *auxiliary particle filter* algorithm was proposed by Pitt and Shephard (1999) to overcome this difficulty. While not really needed for fully specified DLMs, an extension of the algorithm, due to Liu and West (2001), turns out to be very useful even in the DLM case when the model contains unknown parameters. For this reason we present Pitt and Shephard's auxiliary particle filter here, followed in the next section by Liu and West's extension to deal with unknown model parameters.

Suppose that at time $t-1$ a discrete approximation $\hat{\pi}_{t-1} = \sum_{i=1}^{N} w_{t-1}^{(i)} \delta_{\theta_{0:t-1}^{(i)}}$ to the joint smoothing distribution $\pi(\theta_{0:t-1}|y_{1:t-1})$ is available. The goal is to update the approximate smoothing distribution when a new data point is observed or, in other words, to obtain a discrete approximation $\hat{\pi}_t$ to the joint smoothing distribution at time t, $\pi(\theta_{0:t}|y_{1:t})$. We have:

$$\begin{aligned}
\pi(\theta_{0:t}|y_{1:t}) &\propto \pi(\theta_{0:t}, y_t|y_{1:t-1}) \\
&= \pi(y_t|\theta_{0:t}, y_{1:t-1}) \cdot \pi(\theta_t|\theta_{0:t-1}, y_{1:t-1}) \cdot \pi(\theta_{0:t-1}|y_{1:t-1}) \\
&= \pi(y_t|\theta_t) \cdot \pi(\theta_t|\theta_{t-1}) \cdot \pi(\theta_{0:t-1}|y_{1:t-1}) \\
&\approx \pi(y_t|\theta_t) \cdot \pi(\theta_t|\theta_{t-1}) \cdot \hat{\pi}_{t-1}(\theta_{0:t-1}) \\
&= \sum_{i=1}^{N} w_{t-1}^{(i)} \pi(y_t|\theta_t) \pi(\theta_t|\theta_{t-1}^{(i)}) \delta_{\theta_{0:t-1}^{(i)}}.
\end{aligned}$$

Note that the last expression is an unnormalized distribution for $\theta_{0:t}$, which is discrete in the first t components and continuous in the last, θ_t. This distribution, which approximates $\pi(\theta_{0:t}|y_{1:t})$, can be taken to be our target for an importance sampling step. The target being a mixture distribution, a standard approach to get rid of the summation is to introduce a latent variable I, taking values in $\{1, \ldots, N\}$, such that:

$$\begin{aligned} &\mathrm{P}(I = i) = w_{t-1}^{(i)}, \\ &\theta_{0:t}|I = i \sim C\pi(y_t|\theta_t)\pi(\theta_t|\theta_{t-1}^{(i)})\delta_{\theta_{0:t-1}^{(i)}}. \end{aligned}$$

Thus extended, the target becomes

$$\pi^{\mathrm{aux}}(\theta_{0:t}, i|y_{1:t}) \propto w_{t-1}^{(i)}\pi(y_t|\theta_t)\pi(\theta_t|\theta_{t-1}^{(i)})\delta_{\theta_{0:t-1}^{(i)}}.$$

The importance density suggested by Pitt and Shephard for this target is

$$g_t(\theta_{0:t}, i|y_{1:t}) \propto w_{t-1}^{(i)}\pi(y_t|\hat{\theta}_t^{(i)})\pi(\theta_t|\theta_{t-1}^{(i)})\delta_{\theta_{0:t-1}^{(i)}},$$

where $\hat{\theta}_t^{(i)}$ is a central value, such as the mean or the mode, of $\pi(\theta_t|\theta_{t-1} = \theta_{t-1}^{(i)})$. A sample from g_t is easily obtained by iterating, for $k = 1, \ldots, N$, the following two steps.

1. Draw a classification variable I_k, with
$$\mathrm{P}(I_k = i) \propto w_{t-1}^{(i)}\pi(y_t|\hat{\theta}_t^{(i)}), \qquad i = 1, \ldots, N.$$
2. Given $I_k = i$, draw
$$\theta_t^{(k)} \sim \pi(\theta_t|\theta_{t-1}^{(i)})$$
and set $\theta_{0:t}^{(k)} = (\theta_{0:t-1}^{(i)}, \theta_t^{(k)})$.

The importance weight of the kth draw from g_t is proportional to

$$\tilde{w}_t^{(k)} = \frac{w_{t-1}^{(I_k)}\pi(y_t|\theta_t^{(k)})\pi(\theta_t^{(k)}|\theta_{t-1}^{(k)})}{w_{t-1}^{(I_k)}\pi(y_t|\hat{\theta}_t^{(k)})\pi(\theta_t^{(k)}|\theta_{t-1}^{(k)})} = \frac{\pi(y_t|\theta_t^{(k)})}{\pi(y_t|\hat{\theta}_t^{(k)})}.$$

After normalizing the $\tilde{w}_t^{(k)}$'s and discarding the classification variables I_k's, we finally obtain the discrete approximation to the joint smoothing distribution at time t:

$$\hat{\pi}_t(\theta_{0:t}) = \sum_{i=1}^{N} w_t^{(i)}\delta_{\theta_{0:t}^{(i)}} \approx \pi(\theta_{0:t}|y_{1:t}).$$

As with the standard algorithm of Section 5.1, a resampling step is commonly applied in case the effective sample size drops below a specified threshold. A summary of the auxiliary particle filter is provided in Algorithm 5.2

0. Initialize: draw $\theta_0^{(1)}, \ldots, \theta_0^{(N)}$ independently from $\pi(\theta_0)$ and set

$$w_0^{(i)} = N^{-1}, \qquad i = 1, \ldots, N.$$

1. For $t = 1, \ldots, T$:

1.1) For $k = 1, \ldots, N$:

- Draw I_k, with $\mathrm{P}(I_k = i) \propto w_{t-1}^{(i)} \pi(y_t | \hat{\theta}_t^{(i)})$.
- Draw $\theta_t^{(k)}$ from $\pi(\theta_t | \theta_{t-1} = \theta_{t-1}^{(I_k)})$ and set

$$\theta_{0:t}^{(k)} = \big(\theta_{0:t-1}^{(I_k)}, \theta_t^{(k)}\big).$$

- Set

$$\tilde{w}_t^{(k)} = \frac{\pi(y_t | \theta_t^{(k)})}{\pi(y_t | \hat{\theta}_t^{(k)})}.$$

1.2) Normalize the weights:

$$w_t^{(i)} = \frac{\tilde{w}_t^{(i)}}{\sum_{j=1}^N \tilde{w}_t^{(j)}}.$$

1.3) Compute

$$N_{\mathit{eff}} = \left(\sum_{i=1}^N \big(w_t^{(i)}\big)^2 \right)^{-1}.$$

1.4) If $N_{\mathit{eff}} < N_0$, resample:

- Draw a sample of size N from the discrete distribution

$$\mathrm{P}\big(\theta_{0:t} = \theta_{0:t}^{(i)}\big) = w_t^{(i)}, \qquad i = 1, \ldots, N,$$

and relabel this sample

$$\theta_{0:t}^{(1)}, \ldots, \theta_{0:t}^{(N)}.$$

- Reset the weights: $w_t^{(i)} = N^{-1}$, $i = 1, \ldots, N$.

1.5) Set $\hat{\pi}_t = \sum_{i=1}^N w_t^{(i)} \delta_{\theta_{0:t}^{(i)}}$.

Algorithm 5.2: Summary of the auxiliary particle filter algorithm

The main advantage of the auxiliary particle filter over the simple direct algorithm described in the previous section consists in the fact that it allows to use the one-step prior distribution $\pi(\theta_t|\theta_{t-1})$ to draw θ_t without losing much efficiency. Loosely speaking, when drawing from g_t, the role of the first step is to preselect a conditioning θ_{t-1} that is likely to evolve into a highly plausible θ_t in light of the new observation y_t. In this way possible conflicts between prior—$\pi(\theta_t|\theta_{t-1})$—and likelihood—$\pi(y_t|\theta_t)$—are minimized. It should be emphasized that for a general state space model deriving and drawing from the optimal instrumental kernel is often unfeasible, unlike in the DLM case, while the

prior distribution is almost always available. Therefore, the ingenious use of the latter done by the auxiliary particle filter algorithm combines efficiency and simplicity.

5.3 Sequential Monte Carlo with unknown parameters

In real applications the model almost invariably contains unknown parameters that need to be estimated from the data. Denoting again by ψ the vector of unknown parameters, the target distribution at time t for a sequential Monte Carlo algorithm is therefore in this case $\pi(\theta_{0:t}, \psi|y_{1:t})$. As detailed in Section 4.4, a (weighted) sample from the forecast distributions can be easily obtained once a (weighted) sample from the joint posterior distribution is available. On the other hand, the filtering distribution and the posterior distribution of the parameter can be trivially obtained by marginalization. A simple-minded approach to sequential Monte Carlo for a model with an unknown parameter is to extend the state vector to include ψ as part of it, defining the trivial dynamics $\psi_t = \psi_{t-1}$ $(= \psi)$. In this way a relatively simple DLM typically becomes a nonlinear and nonnormal state space model. However, the most serious drawback is that, applying the general algorithm of Section 5.1 (or the auxiliary particle filter of Section 5.2), the values $\psi_t^{(i)}$, $i = 1, \ldots, N$, are those drawn at time $t = 0$, since there is no evolution for this fictitious state. In other words, $\psi_t^{(i)} = \psi_0^{(i)}$ for every i and t, so that the $\psi_t^{(i)}$'s, drawn from the prior distribution, are typically not representative of the posterior distribution at a later time $t > 0$. It is true that, as the particle filter algorithm is sequentially applied, the weights are adjusted to reflect the changes of the target distributions. However, this can only account for the relative weights: if the $\psi_t^{(i)}$'s happen to be all in the tails of the marginal target $\pi(\psi|y_{1:t})$, the discrete approximation provided by the algorithm will always be a poor one. There is, in view of the previous considerations, a need to "refresh" the sampled values of ψ in order to follow the evolution of the posterior distribution. This can be achieved by discarding the current values of ψ each time the target changes and generating new ones. Among the different available methods, probably the most commonly used is the one proposed by Liu and West (2001) and described below, which extends the auxiliary particle filter. Fearnhead (2002), Gilks and Berzuini (2001) and Storvik (2002) propose interesting alternative algorithms.

The idea of Liu and West essentially consists of constructing an approximate target distribution at time t that is continuous not only in θ_t, but also in ψ, so that using importance sampling one draws values of ψ from a continuous importance density, effectively forgetting about the values of ψ used in the discrete approximation at time $t-1$. Consider the discrete approximation available at time $t-1$:

$$\hat{\pi}_{t-1}(\theta_{0:t-1}, \psi) = \sum_{i=1}^{N} w_{t-1}^{(i)} \delta_{(\theta_{0:t-1}^{(i)}, \psi^{(i)})} \approx \pi(\theta_{0:t-1}, \psi | y_{0:t-1}).$$

Marginally,

$$\hat{\pi}_{t-1}(\psi) = \sum_{i=1}^{N} w_{t-1}^{(i)} \delta_{\psi^{(i)}} \approx \pi(\psi | y_{0:t-1}). \tag{5.5}$$

Liu and West suggest replacing each point mass $\delta_{\psi^{(i)}}$ with a Normal distribution, so that the resulting mixture becomes a continuous distribution. A naive way of doing so would be to replace $\delta_{\psi^{(i)}}$ with a Normal centered at $\psi^{(i)}$. However, while preserving the mean, this would increase the variance of the approximating distribution. To see that this is the case, let $\bar{\psi}$ and Σ be the mean vector and variance matrix of ψ under $\hat{\pi}_{t-1}$, and let

$$\tilde{\pi}_{t-1}(\psi) = \sum_{i=1}^{N} w_{t-1}^{(i)} \mathcal{N}(\psi; \psi^{(i)}, \Lambda).$$

Introducing a latent classification variable I for the component of the mixture an observation comes from, we have

$$\begin{aligned} \mathrm{E}(\psi) &= \mathrm{E}(\mathrm{E}(\psi|I)) = \mathrm{E}(\psi^{(I)}) \\ &= \sum_{i=1}^{N} w_{t-1}^{(i)} \psi^{(i)} = \bar{\psi}; \\ \mathrm{Var}(\psi) &= \mathrm{E}(\mathrm{Var}(\psi|I)) + \mathrm{Var}(\mathrm{E}(\psi|I)) \\ &= \mathrm{E}(\Lambda) + \mathrm{Var}(\psi^{(I)}) \\ &= \Lambda + \Sigma > \Sigma, \end{aligned}$$

where expected values and variances are with respect to $\tilde{\pi}_{t-1}$. However, by changing the definition of $\tilde{\pi}_{t-1}$ to

$$\tilde{\pi}_{t-1}(\psi) = \sum_{i=1}^{N} w_{t-1}^{(i)} \mathcal{N}(\psi; m^{(i)}, h^2 \Sigma),$$

with $m^{(i)} = a\psi^{(i)} + (1-a)\bar{\psi}$ for some a in $(0,1)$ and $a^2 + h^2 = 1$, we have

$$\begin{aligned} \mathrm{E}(\psi) &= \mathrm{E}(\mathrm{E}(\psi|I)) = \mathrm{E}(a\psi^{(I)} + (1-a)\bar{\psi}) \\ &= a\bar{\psi} + (1-a)\bar{\psi} = \bar{\psi}; \\ \mathrm{Var}(\psi) &= \mathrm{E}(\mathrm{Var}(\psi|I)) + \mathrm{Var}(\mathrm{E}(\psi|I)) \\ &= \mathrm{E}(h^2\Sigma) + \mathrm{Var}(a\psi^{(I)} + (1-a)\bar{\psi}) \\ &= h^2\Sigma + a^2\mathrm{Var}(\psi^{(I)}) = h^2\Sigma + a^2\Sigma = \Sigma. \end{aligned}$$

Thus, ψ has the same first and second moment under $\tilde{\pi}_{t-1}$ and $\hat{\pi}_{t-1}$. Albeit this is true for any a in $(0,1)$, in practice Liu and West recommend to set

$a = (3\delta - 1)/(2\delta)$ for a "discount factor" δ in $(0.95, 0.99)$, which corresponds to an a in $(0.974, 0.995)$. The very same idea can be applied even in the presence of $\theta_{0:t-1}$ to the discrete distribution $\hat{\pi}_{t-1}(\theta_{0:t-1}, \psi)$, leading to the extension of $\tilde{\pi}_{t-1}$ to a joint distribution for $\theta_{0:t-1}$ and ψ:

$$\tilde{\pi}_{t-1}(\theta_{0:t-1}, \psi) = \sum_{i=1}^{N} w_{t-1}^{(i)} \mathcal{N}(\psi; m^{(i)}, h^2 \Sigma) \delta_{\theta_{0:t-1}^{(i)}}.$$

Note that $\tilde{\pi}_{t-1}$ is discrete in $\theta_{0:t-1}$, but continuous in ψ. From this point onward, the method parallels the development of the auxiliary particle filter. After the new data point y_t is observed, the distribution of interest becomes

$$\begin{aligned}
\pi(\theta_{0:t}, \psi | y_{1:t}) &\propto \pi(\theta_{0:t}, \psi, y_t | y_{1:t-1}) \\
&= \pi(y_t | \theta_{0:t}, \psi, y_{1:t-1}) \cdot \pi(\theta_t | \theta_{0:t-1}, \psi, y_{1:t-1}) \cdot \pi(\theta_{0:t-1}, \psi | y_{1:t-1}) \\
&= \pi(y_t | \theta_t, \psi) \cdot \pi(\theta_t | \theta_{t-1}, \psi) \cdot \pi(\theta_{0:t-1}, \psi | y_{1:t-1}) \\
&\approx \pi(y_t | \theta_t, \psi) \cdot \pi(\theta_t | \theta_{t-1}, \psi) \cdot \tilde{\pi}_{t-1}(\theta_{0:t-1}, \psi) \\
&= \sum_{i=1}^{N} w_{t-1}^{(i)} \pi(y_t | \theta_t, \psi) \pi(\theta_t | \theta_{t-1}^{(i)}, \psi) \mathcal{N}(\psi; m^{(i)}, h^2 \Sigma) \delta_{\theta_{0:t-1}^{(i)}}.
\end{aligned}$$

Similarly to what we did in Section 5.2, we can introduce an auxiliary classification variable I such that:

$$\begin{aligned}
\mathrm{P}(I = i) &= w_{t-1}^{(i)}, \\
\theta_{0:t}, \psi | I = i &\sim C \pi(y_t | \theta_t, \psi) \pi(\theta_t | \theta_{t-1}^{(i)}, \psi) \mathcal{N}(\psi; m^{(i)}, h^2 \Sigma) \delta_{\theta_{0:t-1}^{(i)}}.
\end{aligned}$$

Note that the conditional distribution in the second line is continuous in θ_t and ψ, and discrete in $\theta_{0:t-1}$—in fact, degenerate on $\theta_{0:t-1}^{(i)}$. With the introduction of the random variable I, the auxiliary target distribution for the importance sampling update becomes

$$\pi^{\mathrm{aux}}(\theta_{0:t}, \psi, i | y_{1:t}) \propto w_{t-1}^{(i)} \pi(y_t | \theta_t, \psi) \pi(\theta_t | \theta_{t-1}^{(i)}, \psi) \mathcal{N}(\psi; m^{(i)}, h^2 \Sigma) \delta_{\theta_{0:t-1}^{(i)}}.$$

As an importance density, a convenient choice is

$$\begin{aligned}
g_t(\theta_{0:t}, \psi, i | y_{1:t}) \propto w_{t-1}^{(i)} \pi(y_t | \theta_t = \hat{\theta}_t^{(i)}, \psi = m^{(i)}) \pi(\theta_t | \theta_{t-1}^{(i)}, \psi) \\
\mathcal{N}(\psi; m^{(i)}, h^2 \Sigma) \delta_{\theta_{0:t-1}^{(i)}},
\end{aligned}$$

where $\hat{\theta}_t^{(i)}$ is a central value, such as the mean or the mode, of $\pi(\theta_t | \theta_{t-1} = \theta_{t-1}^{(i)}, \psi = m^{(i)})$. A sample from g_t can be obtained by iterating, for $k = 1, \ldots, N$, the following three steps.

1. Draw a classification variable I_k, with

$$\mathrm{P}(I_k = i) \propto w_{t-1}^{(i)} \pi(y_t | \theta_t = \hat{\theta}_t^{(i)}, \psi = m^{(i)}), \qquad i = 1, \ldots, N.$$

2. Given $I_k = i$, draw $\psi \sim \mathcal{N}(m^{(i)}, h^2\Sigma)$ and set $\psi^{(k)} = \psi$.
3. Given $I_k = i$ and $\psi = \psi^{(k)}$, draw

$$\theta_t^{(k)} \sim \pi(\theta_t | \theta_{t-1} = \theta_{t-1}^{(i)}, \psi = \psi^{(k)})$$

and set $\theta_{0:t}^{(k)} = (\theta_{0:t-1}^{(i)}, \theta_t^{(k)})$.

The importance weight of the kth draw from g_t is proportional to

$$\begin{aligned}\tilde{w}_t^{(k)} &= \frac{w_{t-1}^{(I_k)} \pi(y_t | \theta_t = \theta_t^{(k)}, \psi = \psi^{(k)}) \pi(\theta_t^{(k)} | \theta_{t-1}^{(k)}, \psi^{(k)}) \mathcal{N}(\psi^{(k)}; m^{(I_k)}, h^2\Sigma)}{w_{t-1}^{(I_k)} \pi(y_t | \theta_t = \hat{\theta}_t^{(I_k)}, \psi = m^{(I_k)}) \pi(\theta_t^{(k)} | \theta_{t-1}^{(k)}, \psi^{(k)}) \mathcal{N}(\psi^{(k)}; m^{(I_k)}, h^2\Sigma)} \\ &= \frac{\pi(y_t | \theta_t = \theta_t^{(k)}, \psi = \psi^{(k)})}{\pi(y_t | \theta_t = \hat{\theta}_t^{(I_k)}, \psi = m^{(I_k)})}.\end{aligned}$$

Renormalizing the weights, we obtain the approximate joint posterior distribution at time t

$$\hat{\pi}_t(\theta_{0:t}, \psi) = \sum_{i=1}^{N} w_t^{(i)} \delta_{(\theta_{0:t}^{(i)}, \psi^{(i)})} \approx \pi(\theta_{0:t}, \psi | y_{1:t}).$$

As was the case with the particle filter algorithms described in the previous sections, also in this case a resampling step can be applied whenever the effective sample size drops below a specified threshold. Algorithm 5.3 provides a convenient summary of the procedure.

Let us point out that, in order for the mixture of normals approximation of the posterior distribution at time $t-1$ to make sense, the parameter ψ has to be expressed in a form that is consistent with such a distribution—in particular, the support of a one-dimensional parameter must be the entire real line. For example, variances can be parametrized in terms of their log, probabilities in terms of their logit, and so on. In other words, and according to the suggestion by Liu and West, each parameter must be transformed so that the support of the distribution of the transformed parameter is the entire real line. A simpler alternative is to use a mixture of nonnormal distributions, appropriately selected so that their support is the same as that of the distribution of the parameter. For example, if a model parameter represents an unknown probability, and therefore its support is the interval $(0, 1)$, then one can consider approximating the discrete distribution obtained by the particle filter at time $t-1$ with a mixture of beta distributions instead of a mixture of normals, proceeding in all other respects as described above. Let us elaborate more on this simple example. Suppose ψ is an unknown parameter in $(0, 1)$. Denote by $\mu(\alpha, \beta)$ and $\sigma^2(\alpha, \beta)$ respectively the mean and variance of a beta distribution with parameters α and β. One can set, for each $i = 1, \ldots, N$,

$$\begin{aligned}\mu^{(i)} &= \mu(\alpha^{(i)}, \beta^{(i)}) = a\psi^{(i)} + (1-a)\bar{\psi}, \\ \sigma^{2(i)} &= \sigma^2(\alpha^{(i)}, \beta^{(i)}) = h^2\Sigma,\end{aligned} \tag{5.6}$$

0. Initialize: draw $(\theta_0^{(1)}, \psi^{(1)}), \ldots, (\theta_0^{(N)}, \psi^{(N)})$ independently from $\pi(\theta_0)\pi(\psi)$. Set $w_0^{(i)} = N^{-1}$, $i = 1, \ldots, N$, and

$$\hat{\pi}_0 = \sum_{i=1}^{N} w_0^{(i)} \delta_{(\theta_0^{(i)}, \psi^{(i)})}.$$

1. For $t = 1, \ldots, T$:

1.1) Compute $\bar{\psi} = \mathrm{E}_{\hat{\pi}_{t-1}}(\psi)$ and $\Sigma = \mathrm{Var}_{\hat{\pi}_{t-1}}(\psi)$. For $i = 1, \ldots, N$, set

$$m^{(i)} = a\psi^{(i)} + (1-a)\bar{\psi},$$
$$\hat{\theta}_t^{(i)} = \mathrm{E}(\theta_t | \theta_{t-1} = \theta_{t-1}^{(i)}, \psi = m^{(i)}).$$

1.2) For $k = 1, \ldots, N$:

- Draw I_k, with $\mathrm{P}(I_k = i) \propto w_{t-1}^{(i)} \pi(y_t | \theta_t = \hat{\theta}_t^{(i)}, \psi = m^{(i)})$.
- Draw $\psi^{(k)}$ from $\mathcal{N}(m^{(I_k)}, h^2\Sigma)$.
- Draw $\theta_t^{(k)}$ from $\pi(\theta_t | \theta_{t-1} = \theta_{t-1}^{(I_k)}, \psi = \psi^{(k)})$ and set

$$\theta_{0:t}^{(k)} = \big(\theta_{0:t-1}^{(I_k)}, \theta_t^{(k)}\big).$$

- Set

$$\tilde{w}_t^{(k)} = \frac{\pi(y_t | \theta_t = \theta_t^{(k)}, \psi = \psi^{(k)})}{\pi(y_t | \theta_t = \hat{\theta}_t^{(I_k)}, \psi = m^{(I_k)})}.$$

1.3) Normalize the weights:

$$w_t^{(i)} = \frac{\tilde{w}_t^{(i)}}{\sum_{j=1}^{N} \tilde{w}_t^{(j)}}.$$

1.4) Compute

$$N_{\mathit{eff}} = \left(\sum_{i=1}^{N} \big(w_t^{(i)}\big)^2 \right)^{-1}.$$

1.5) If $N_{\mathit{eff}} < N_0$, resample:

- Draw a sample of size N from the discrete distribution

$$\mathrm{P}\big((\theta_{0:t}, \psi) = (\theta_{0:t}^{(i)}, \psi^{(i)})\big) = w_t^{(i)}, \qquad i = 1, \ldots, N,$$

and relabel this sample

$$(\theta_{0:t}^{(1)}, \psi^{(1)}), \ldots, (\theta_{0:t}^{(N)}, \psi^{(N)}).$$

- Reset the weights: $w_t^{(i)} = N^{-1}$, $i = 1, \ldots, N$.

1.6) Set $\hat{\pi}_t = \sum_{i=1}^{N} w_t^{(i)} \delta_{(\theta_{0:t}^{(i)}, \psi^{(i)})}$.

Algorithm 5.3: Summary of Liu and West's algorithm

and solve for the pair $(\alpha^{(i)}, \beta^{(i)})$. The equations above can be explicitely solved for the parameters $\alpha^{(i)}$ and $\beta^{(i)}$, giving

$$\alpha^{(i)} = \frac{(\mu^{(i)})^2(1-\mu^{(i)})}{\sigma^{2(i)}} - \mu^{(i)},$$
$$\beta^{(i)} = \frac{\mu^{(i)}(1-\mu^{(i)})^2}{\sigma^{2(i)}} - (1-\mu^{(i)}).$$

It is straightforward to show that the mixture

$$\sum_{i=1}^{N} w_{t-1}^i \mathcal{B}(\psi; \alpha^{(i)}, \beta^{(i)}), \tag{5.7}$$

has the same mean and variance as (5.5), that is $\bar{\psi}$ and Σ.

On the same theme, consider the case of a positive unknown parameter ψ, such as a variance. Then we can consider a mixture of gamma distributions instead of a mixture on Normals. We can solve, for $i = 1, \ldots, N$ the system of equations (5.6) where $\alpha^{(i)}$ and $\beta^{(i)}$ are this time the parameters of a gamma distribution. The explicit solution is in this case

$$\alpha^{(i)} = \frac{(\mu^{(i)})^2}{\sigma^{2(i)}},$$
$$\beta^{(i)} = \frac{\mu^{(i)}}{\sigma^{2(i)}},$$

And the mixture

$$\sum_{i=1}^{N} w_{t-1}^i \mathcal{G}(\psi; \alpha^{(i)}, \beta^{(i)})$$

has mean $\bar{\psi}$ and variance Σ.

When the unknown parameter ψ is a vector it may not be easy to find a parametric family of multivariate distributions $f(\psi; \gamma)$ and proceed as we did in the examples above with the beta and gamma distributions, using a moment matching condition to come up with a continuous mixture that has the first mean and variance as the discrete particle approximation (5.5). When this is the case one can usually adopt the same moment matching approach marginally and consider a mixture of product densities. More specifically, consider a parameter $\psi = (\psi_1, \psi_2)$ and let

$$f(\psi; \gamma) = f_1(\psi_1; \gamma_1) f_2(\psi_2; \gamma_2), \qquad \gamma = (\gamma_1, \gamma_2),$$

where the parameter γ_j can be set in such a way that $f_j(\cdot|\gamma_j)$ has a specific mean and variance $(j = 1, 2)$. Let, with an obvious notation,

$$\bar{\psi} = \begin{bmatrix} \bar{\psi}_1 \\ \bar{\psi}_2 \end{bmatrix}, \qquad \Sigma = \begin{bmatrix} \Sigma_1 & \Sigma_{12} \\ \Sigma_{21} & \Sigma_2 \end{bmatrix}.$$

1.1) For $j = 1, 2$ and $i = 1, \ldots, N$:

- Compute $\bar{\psi}_j = \mathrm{E}_{\hat{\pi}_{t-1}}(\psi_j)$ and $\Sigma_j = \mathrm{Var}_{\hat{\pi}_{t-1}}(\psi_j)$. Set

$$\begin{aligned}
\mu_j^{(i)} &= a\psi_j^{(i)} + (1-a)\bar{\psi}_j, \\
\sigma_j^{(i)} &= h^2 \Sigma_j, \\
\mu^{(i)} &= (\mu_1^{(i)}, \mu_2^{(i)}), \\
\hat{\theta}_t^{(i)} &= \mathrm{E}(\theta_t | \theta_{t-1} = \theta_{t-1}^{(i)}, \psi = \mu^{(i)}).
\end{aligned}$$

- Solve for $\gamma_j^{(i)}$ the system of equations

$$\begin{aligned}
\mathrm{E}_{f_j(\cdot;\gamma_j^{(i)})}(\psi_j) &= \mu_j^{(i)}, \\
\mathrm{Var}_{f_j(\cdot;\gamma_j^{(i)})}(\psi_j) &= \sigma_j^{(i)}.
\end{aligned}$$

1.2) For $k = 1, \ldots, N$:

- Draw I_k, with $\mathrm{P}(I_k = i) \propto w_{t-1}^{(i)} \pi(y_t | \theta_t = \hat{\theta}_t^{(i)}, \psi = \mu^{(i)})$.
- For $j = 1, 2$, draw $\psi_j^{(k)}$ from $f_j(\cdot; \gamma_j^{(I_k)})$
- Draw $\theta_t^{(k)}$ from $\pi(\theta_t | \theta_{t-1} = \theta_{t-1}^{(I_k)}, \psi = \psi^{(k)})$ and set

$$\theta_{0:t}^{(k)} = \left(\theta_{0:t-1}^{(I_k)}, \theta_t^{(k)}\right).$$

- Set

$$\tilde{w}_t^{(k)} = \frac{\pi(y_t | \theta_t = \theta_t^{(k)}, \psi = \psi^{(k)})}{\pi(y_t | \theta_t = \hat{\theta}_t^{(I_k)}, \psi = m^{(I_k)})}.$$

Algorithm 5.4: Changes to Liu and West's algorithm when using product kernels

Then we can set, for $i = 1, \ldots, N$,

$$\begin{aligned}
\mu_j^{(i)} &= \int \psi_j f_j(\psi_j; \gamma_j^{(i)}) \, \mathrm{d}\psi_j = a\psi_j^{(i)} + (1-a)\bar{\psi}_j, \\
\sigma_j^{2(i)} &= \int (\psi_j - \mu_j^{(i)})^2 f_j(\psi_j; \gamma_j^{(i)}) \, \mathrm{d}\psi_j = h^2 \Sigma,
\end{aligned} \tag{5.8}$$

and solve for $\gamma_j^{(i)}$ $(j = 1, 2)$. These are the same equations as (5.6) for the marginal distributions of ψ_1 and ψ_2. Finally, we can consider the mixture

$$\sum_{i=1}^{N} w_{t-1}^{(i)} f_1(\psi_1; \gamma_1^{(i)}) f_2(\psi_2; \gamma_2^{(i)}). \tag{5.9}$$

This will have the same mean as (5.5), $\bar{\psi}$, and the same marginal variances, Σ_1 and Σ_2. As far as the covariance is concerned, a simple calculation shows that under the mixture distribution (5.9) ψ_1 and ψ_2 have covariance $a^2 \Sigma_{12}$. Since in the practical applications of Liu and West's method a is close to one,

it follows that $a^2\Sigma_{12} \approx \Sigma_{12}$. In summary, for a bivariate parameter ψ, the mixture (5.9) provides a countinuous approximation to (5.5) that matches first moments, marginal second moments, and covariances up to a factor $a^2 \approx 1$. Using this product kernel approximation instead of the mixture of normals originally proposed by Liu and West, parts 1.1 and 1.2 of Algorithm 5.3 have to be changed accordingly, as shown in Algorithm 5.4.

To conclude the discussion of mixtures of product kernels within Liu and West's approach, let us note that the two components ψ_1 and ψ_2 may also be multivariate. Furthermore, the technique described can be generalized in an obvious way to product kernels containing more than two factors.

5.3.1 A simple example with unknown parameters

As a simple application of particle filtering for models containing unknown parameters, we go back to the example discussed in Section 5.1.1, this time assuming that both the system and observation variances are unknown. Since we have two unknown positive parameters, we are going to use at any time t products of Gamma kernels in the mixture approximation to the posterior distribution of the parameters. In the notation of the previous section, we have $\psi_1 = V$, $\psi_2 = W$, and $f_j(\psi_j; \gamma_j)$ is a gamma density for $j = 1, 2$, where $\gamma_j = (\alpha_j, \beta_j)$ is the standard vector parameter of the gamma distribution (see Appendix A). We use the same data that we simulated in Section 5.1.1. We choose independent uniform priors on $(0, 10)$ for both V and W. By looking at a plot of the data, the upper limit 10 for the variances seems more than enough for the interval to contain the true value of the parameter. Within these boundaries, a uniform prior does not carry any particularly strong information about the unknown variances. The reason for not choosing a more spread out prior distribution is that the particle filter algorithm initially generates the particles from the prior and, if the prior puts little probability in regions having high likelihood, most of the particles will be discarded after just one or two steps. Note that we are not arguing for selecting a prior based on the particular numerical method that one uses to evaluate the posterior distribution. On the contrary, we think that in this case a uniform prior on a finite interval better represents our belief about the variances than, say, a prior with infinite variance. After all, after plotting the data, who will seriously consider the possibility of V being larger than 100? Or 1000?

R code

```
> ### PF with unknown parameters: Liu and West
> N <- 10000
> a <- 0.975
> set.seed(4521)
> pfOutTheta <- matrix(NA_real_, n + 1, N)
> pfOutV <- matrix(NA_real_, n + 1, N)
```

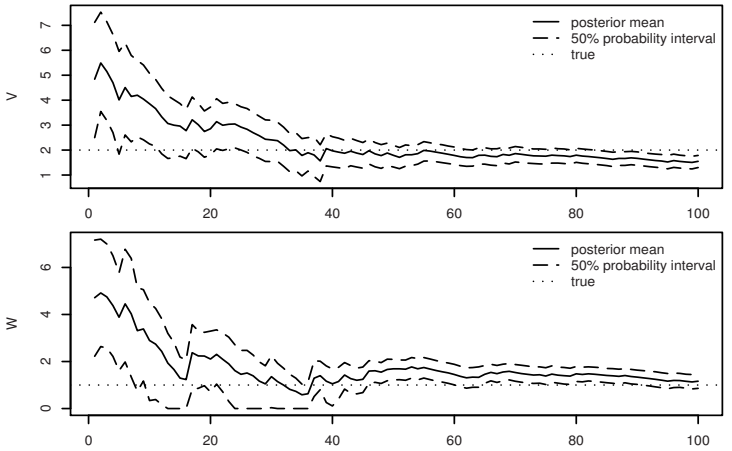

Fig. 5.2. Sequential estimation obtained via particle filter of V (top) and W (bottom)

```
> pfOutW <- matrix(NA_real_, n + 1, N)
> wt <- matrix(NA_real_, n + 1, N)
> ## Initialize sampling from the prior
> pfOutTheta[1, ] <- rnorm(N, mean = m0(mod),
+                          sd = sqrt(C0(mod)))
> pfOutV[1, ] <-  runif(N, 0, 10)
> pfOutW[1, ] <-  runif(N, 0, 10)
> wt[1, ] <- rep(1/N, N)
> for (it in 2 : (n + 1))
+ {
+     ## compute means and variances of the particle
+     ## cloud for V and W
+     meanV <- weighted.mean(pfOutV[it - 1, ], wt[it - 1, ])
+     meanW <- weighted.mean(pfOutW[it - 1, ], wt[it - 1, ])
+     varV <- weighted.mean((pfOutV[it - 1, ] - meanV)^2,
+                           wt[it - 1, ])
+     varW <- weighted.mean((pfOutW[it - 1, ] - meanW)^2,
+                           wt[it - 1, ])
+     ## compute the parameters of Gamma kernels
+     muV <- a * pfOutV[it - 1, ] + (1 - a) * meanV
+     sigma2V <- (1 - a^2) * varV
+     alphaV <- muV^2 / sigma2V
+     betaV <- muV / sigma2V
+     muW <- a * pfOutW[it - 1, ] + (1 - a) * meanW
+     sigma2W <- (1 - a^2) * varW
+     alphaW <- muW^2 / sigma2W
```

```
+     betaW <- muW / sigma2W
+     ## draw the auxiliary indicator variables
+     probs <- wt[it - 1,] * dnorm(y[it - 1], sd = sqrt(muV),
+                                  mean = pfOutTheta[it - 1, ])
+     auxInd <- sample(N, N, replace = TRUE, prob = probs)
+     ## draw the variances V and W
+     pfOutV[it, ] <- rgamma(N, shape = alphaV[auxInd],
+                            rate = betaV[auxInd])
+     pfOutW[it, ] <- rgamma(N, shape = alphaW[auxInd],
+                            rate = betaW[auxInd])
+     ## draw the state theta
+     pfOutTheta[it, ] <- rnorm(N, mean =
+                               pfOutTheta[it - 1, auxInd],
+                               sd = sqrt(pfOutW[it, ]))
+     ## compute the weights
+     wt[it, ] <- exp(dnorm(y[it - 1],
+                           mean = pfOutTheta[it, ],
+                           sd = sqrt(pfOutV[it, ]),
+                           log = TRUE) -
+                     dnorm(y[it - 1],
+                           mean = pfOutTheta[it - 1, auxInd],
+                           sd = sqrt(muV[auxInd]),
+                           log = TRUE))
+     wt[it, ] <- wt[it, ] / sum(wt[it, ])
+ }
```

5.4 Concluding remarks

We stressed in this chapter that the particle filter is very useful in online inference, where it can be used to recursively update a posterior distribution when the Kalman filter is not available because the model contains unknown parameters or otherwise—for example, the model is nonlinear. While this is certainly true, we would like to add a word of caution about the practical usage of particle filtering techniques in genuine sequential applications.

With very few exceptions that do not cover the samplers presented above, all the available asymptotic results hold when the time horizon T is fixed and the number of particles N is let to go to infinity. Furthermore, in order to obtain Monte Carlo approximations of similar quality for different time horizons T_1 and T_2, we need to use a number of particles proportional to T_i. The implication of these results is that if we start a particle filter with N particles, and we keep running it as new observations become available, the quality of the approximation will eventually deteriorate, making the particle approximation useless in the long run. The reason is that, even if we only use the particle

approximation of the marginal posterior at time t, $\hat{\pi}_t(\theta_t)$, we are effectively targeting the joint posterior distribution $\pi(\theta_{0:t}|y_{1:t})$, so we are trying to track a distribution in an increasingly large number of dimensions. Another intuitive way to explain the deterioration of the particle filter approximation over time is to consider that the approximation at time t is based on the approximation at time $t-1$, so that the errors accumulate.

A possible practical solution, for applications that require a sequential updating of a posterior destribution over an unbounded time horizon, is to run an MCMC sampler every T sampling intervals to draw a sample from the posterior distribution at that time—possibly based on the most recent kT data points only, with $k \gg 1$—and use this sample to start the particle filter updating scheme for the next T sampling intervals. In this way one can run the MCMC off-line, while at the same time keeping updating the posterior using a particle filter. For example, in tracking the domestic stock market, one can run an MCMC over the weekend, when the data flow stops, and use a particle filter with hourly data, say, to update the posterior during the working week.

A similar idea can be used to initialize the particle filter when the prior distribution is diffuse. In this case, as we previously noticed, starting the particle filter with a sample from the prior will produce particles that are going to be off-target after just one or two updating steps. Instead, one can run an MCMC based on a small initial stretch of data in order to start the particle filter from a fairly stable particle cloud.