This is note covers ideas for using a particle filter for the `dynamichazard` package in R. See `https://cran.r-project.org/web/packages/dynamichazard/vignettes/ddhazard.pdf` for more information of the package. At this point, the estimation method includes the following (crude) approximations: an Extended Kalman filters, unscented Kalman filter and a mode approximations. You can run `dynamichazard::ddhazard_app()` after installing the package to see the performance and computation time of the methods.

# 1 Method

## Model

The model is:

$$\begin{aligned} y_{it} &\sim \mathrm{P}\left(y_{it} \middle| \theta_{it}\right) \\ \boldsymbol{\theta}_t &= \mathbf{X}_t \boldsymbol{\alpha}_t \\ \boldsymbol{\alpha}_t &= \mathbf{F}\boldsymbol{\alpha}_{t-1} + \mathbf{R}\boldsymbol{\eta}_t \quad \boldsymbol{\eta}_t \sim N\left(\mathbf{0}, \mathbf{Q}\right) \\ & \qquad\qquad\quad \boldsymbol{\alpha}_0 \sim N\left(\boldsymbol{a}_0, \mathbf{Q}_0\right) \end{aligned} \tag{1}$$

where we denote the conditional densities as $\boldsymbol{y}_t \sim g\left(\cdot \middle| \boldsymbol{\alpha}_t\right)$ and $\boldsymbol{\alpha}_t \sim f\left(\cdot \middle| \boldsymbol{\alpha}_{t-1}\right)$. An alternative here could be to add noise to $\boldsymbol{\theta}_t$ and use the methods in Andrieu and Doucet (2002).

We are in a survival analysis setting where the simplest model has an indicator of death of individual $i$ in time $t$ such that $y_{it} \in \{0, 1\}$ where $\theta_{it}$ is the linear predictor where we use the logistic function as the link function. For each $t$ we a have risk set given by $R_t \subseteq \{1, 2, \dots, n\}$. The observed outcomes are given by $\boldsymbol{y}_t = \{y_{it}\}_{i \in R_t}$. $\mathbf{X}_t$ is the matrix of the covariates and $\boldsymbol{\alpha}_t \in \mathbb{R}^p$ is the time-varying coefficients. The problems we are looking at have $n >> p$ (e.g. $n = 100000$ and $p = 20$).

I am planning to use a first order random walk for $\boldsymbol{\alpha}_t$ so I could drop $\mathbf{F}$. Further, I will need to estimate $\mathbf{Q}, \boldsymbol{\alpha}_0$ and fix $\mathbf{Q}_0$. To do so, I will use a particle filter and smoother to get smoothed estimates of $\boldsymbol{\alpha}_1, \dots, \boldsymbol{\alpha}_d$ and use an EM-algorithm. See Kantas et al. (2015) for a review of parameter estimation in state-space models with particle filters.

One choice of smoother is the generalized two-filter smoothing in Fearnhead et al. (2010) and Briers et al. (2010). I have included the $\mathcal{O}\left(N\right)$ smoother in Fearnhead et al. (2010) shown in algorithm 1. I will use this throughout the rest of this note.

## Considerations

Algorithm 1 requires that we specify the following importance densities and re-sampling weights (optimal values are given as the right hand side):

$$
\begin{aligned}
q\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t\right) &= \mathrm{P}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t\right) \\
\beta_t^{(j)} &\propto \mathrm{P}\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}\right) w_{t-1}^{(j)} \\
\widetilde{q}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right) \widetilde{\beta}_t^{(k)} &\approx \gamma_t\left(\boldsymbol{\alpha}_t\right) \mathrm{P}\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right) \mathrm{P}\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)} \middle| \boldsymbol{\alpha}_t\right) \frac{\widetilde{w}_{t+1}^{(k)}}{\gamma_{t+1}\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right)} \\
\widetilde{q}\left(\widehat{\alpha}_t^{(i)} \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right) &= \mathrm{P}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right)
\end{aligned} \tag{2}
$$

Further, we need to define a backwards filter distribution approximation:

$$
\widetilde{p}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{t:T}\right) \propto \gamma_t\left(\boldsymbol{\alpha}_t\right) \mathrm{P}\left(\boldsymbol{y}_{t:T} \middle| \boldsymbol{\alpha}_t\right) \tag{3}
$$

with an artificial prior distribution $\gamma_t\left(\boldsymbol{\alpha}_t\right)$. See Briers et al. (2010) for how to select $\gamma_t$ and examples hereof. Fearnhead et al. (2010) also provides examples.

Given the models of interest we have that:

- Evaluating $\mathrm{P}\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right)$ is an expensive operation as $n >> p$ and it has a $\mathcal{O}\left(np\right)$ computational cost. In general, any $\mathcal{O}\left(n^l\right)$ with $l \geq 1$ is going to take considerable time if it needed in any steps of the algorithms.
- Evaluating $f\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}\right)$ and $f\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t+1}\right)$ is cheap and can be done in closed form and sampling from this distribution can be done in closed form.
- The second example in Fearnhead et al. (2010) is close to the model here though with $n = 1$ and $p = 2$.

**Forward filter (algorithm 2**

Let $\mathcal{N}\left(\cdot \middle| \cdot, \cdot\right)$ denote a (multivariate) normal distribution. Then we can select the proposal density as:

$$
q\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t\right) = \mathcal{N}\left(\boldsymbol{\alpha}_t \middle| \mathbf{F}\boldsymbol{\alpha}_{t-1}^{(j)}, \mathbf{Q}\right) \tag{4}
$$

which we can sample from in $\mathcal{O}\left(Np^2\right)$ time if we have a pre-computed Cholesky decomposition. This is often called the *bootstrap filter*. Another option is to use normal approximation of $\boldsymbol{y}_t$:

$$
\begin{aligned}
g\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right) &\simeq \widetilde{g}_t\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right) \\
&= \mathcal{N}\left(\mathbf{X}_t\boldsymbol{\alpha}_t \middle| \mathbf{X}_t\mathbf{F}\bar{\boldsymbol{\alpha}}_{t-1} - \mathbf{G}_t\left(\mathbf{F}\bar{\boldsymbol{\alpha}}_{t-1}\right)^{-1} \boldsymbol{g}_t\left(\mathbf{F}\bar{\boldsymbol{\alpha}}_{t-1}\right), -\mathbf{G}_t\left(\mathbf{F}\bar{\boldsymbol{\alpha}}_{t-1}\right)^{-1}\right)
\end{aligned} \tag{5}
$$

where:

$$
\begin{aligned}
\mathbf{g}_t\left(\boldsymbol{\alpha}\right) &= \left\{ \frac{\partial \log \mathrm{P}\left(y_{it}|\theta_{it}\right)}{\partial \theta_{it}}\bigg|_{\theta_{it}=\boldsymbol{x}_{it}^\top\boldsymbol{\alpha}} \right\}_{i \in R_t} \\
\mathbf{G}_t\left(\boldsymbol{\alpha}\right) &= \mathrm{diag}\left( \left\{ \frac{\partial^2 \log \mathrm{P}\left(y_{it}|\theta_{it}\right)}{\partial \theta_{it}^2}\bigg|_{\theta_{it}=\boldsymbol{x}_{it}^\top\boldsymbol{\alpha}} \right\}_{i \in R_t} \right)
\end{aligned} \tag{6}
$$

to get:

$$q\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t\right) \propto \mathcal{N}\left(\boldsymbol{\alpha}_t \middle| \mathbf{F}\boldsymbol{\alpha}_{t-1}^{(j)}, \mathbf{Q}\right) \widetilde{g}\left(\boldsymbol{y}_t \middle| \mathbf{F}\boldsymbol{\alpha}_{t-1}^{(j)}\right) \tag{7}$$

where we can analytically integrate out $\widetilde{g}$ to get:

$$\begin{aligned} q\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t\right) &= \mathcal{N}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t\right) \\ \boldsymbol{\Sigma}_t^{-1} &= \mathbf{X}_t^\top \left(-\mathbf{G}_t\left(\mathbf{F}\bar{\boldsymbol{\alpha}}_{t-1}\right)\right)\mathbf{X}_t + \mathbf{Q}^{-1} \\ \boldsymbol{\mu}_t &= \boldsymbol{\Sigma}_t\left(\mathbf{Q}^{-1}\mathbf{F}\boldsymbol{\alpha}_{t-1}^{(j)} + \mathbf{X}_t^\top\left(-\mathbf{G}_t\left(\mathbf{F}\bar{\boldsymbol{\alpha}}_{t-1}\right)\right)\left(\mathbf{X}_t\mathbf{F}\bar{\boldsymbol{\alpha}}_{t-1} - \mathbf{G}_t\left(\mathbf{F}\bar{\boldsymbol{\alpha}}_{t-1}\right)^{-1}\boldsymbol{g}_t\left(\mathbf{F}\bar{\boldsymbol{\alpha}}_{t-1}\right)\right)\right) \\ &= \boldsymbol{\Sigma}_t\left(\mathbf{Q}^{-1}\mathbf{F}\boldsymbol{\alpha}_{t-1}^{(j)} + \mathbf{X}_t^\top\left(\left(-\mathbf{G}_t\left(\mathbf{F}\bar{\boldsymbol{\alpha}}_{t-1}\right)\right)\mathbf{X}_t\mathbf{F}\bar{\boldsymbol{\alpha}}_{t-1} + \boldsymbol{g}_t\left(\mathbf{F}\bar{\boldsymbol{\alpha}}_{t-1}\right)\right)\right) \end{aligned} \tag{8}$$

We can select $\bar{\boldsymbol{\alpha}}_{t-1}$ as the weighted mean given the cloud at time $t-1$. This is similar to the second order random walk example in Fearnhead et al. (2010) though they use the mode. The downside is an $\mathcal{O}\left(np^2 + p^3\right)$ computational cost though independent of the number of particles. The total cost of sampling is $\mathcal{O}\left(np^2 + p^3 + Np^2\right)$. Another option is to set $\bar{\boldsymbol{\alpha}}_{t-1} = \boldsymbol{\alpha}_{t-1}^{(j)}$. This will improve the Taylor expansion but yields an $\mathcal{O}\left(N\left(np^2 + p^3\right)\right)$ computational cost.

Next, we have the re-sampling weights. A simple solution is not to use an auxiliary particle filter and set:

$$\beta_t^{(j)} \propto w_{t-1}^{(j)} \tag{9}$$

which has an $\mathcal{O}(N)$ cost of sampling. Another options is to set:

$$\begin{aligned} \beta_t^{(j)} &\propto \mathrm{P}\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}\right) w_{t-1}^{(j)} \\ &= w_{t-1}^{(j)} \int_{\boldsymbol{\alpha}_t} \mathcal{N}\left(\boldsymbol{\alpha}_t \middle| \mathbf{F}\boldsymbol{\alpha}_{t-1}^{(j)}, \mathbf{Q}\right) g\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right) \partial\boldsymbol{\alpha}_t \\ &\approx w_{t-1}^{(j)} \int_{\boldsymbol{\alpha}_t} \mathcal{N}\left(\boldsymbol{\alpha}_t \middle| \mathbf{F}\boldsymbol{\alpha}_{t-1}^{(j)}, \mathbf{Q}\right) \widetilde{g}_t\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right) \partial\boldsymbol{\alpha}_t \\ &\approx \frac{w_{t-1}^{(j)} \mathcal{N}\left(\boldsymbol{\mu}_t \middle| \mathbf{F}\boldsymbol{\alpha}_{t-1}^{(j)}, \mathbf{Q}\right) \widetilde{g}_t\left(\boldsymbol{y}_t \middle| \boldsymbol{\mu}_t\right)}{q\left(\boldsymbol{\mu}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t\right)} \end{aligned} \tag{10}$$

which is an $\mathcal{O}\left(Np^2\right)$ computational cost assuming that we are using (19) already.

**Backward filter (algorithm 3**

We need to specify $\gamma_t(\boldsymbol{\alpha}_t)$. Briers et al. (2010, page 69 and 70) provides recommendation on the selection. This leads to:

$$\begin{aligned} \gamma_t(\boldsymbol{\alpha}_t) &= \mathcal{N}\left(\boldsymbol{\alpha}_t \middle| \overleftarrow{\boldsymbol{m}}_t, \overleftarrow{\mathbf{P}}_t\right) \\ \overleftarrow{\boldsymbol{m}}_t &= \mathbf{F}^t\boldsymbol{a}_0 \\ \overleftarrow{\mathbf{P}}_t &= \begin{cases} \mathbf{Q}_0 & t = 0 \\ \mathbf{F}\mathbf{P}_{t-1}\mathbf{F}^\top + \mathbf{Q} & t > 0 \end{cases} \end{aligned} \tag{11}$$

such that:

$$\begin{aligned}
\mathrm{P}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t+1}\right) &= \mathcal{N}\left(\boldsymbol{\alpha}_t \middle| \overleftarrow{\boldsymbol{a}}_t, \overleftarrow{\mathbf{S}}_t\right) \\
\overleftarrow{\mathbf{S}}_t &= \overleftarrow{\mathbf{P}}_t \mathbf{F}^\top \overleftarrow{\mathbf{P}}_{t+1}^{-1} \mathbf{Q} \left(\mathbf{F}^\top\right)^{-1} \\
\overleftarrow{\boldsymbol{a}}_t &= \overleftarrow{\mathbf{P}}_t \mathbf{F}^\top \overleftarrow{\mathbf{P}}_{t+1}^{-1} \boldsymbol{\alpha}_{t+1} + \overleftarrow{\mathbf{S}}_t \overleftarrow{\mathbf{P}}_t^{-1} \overleftarrow{\boldsymbol{m}}_t
\end{aligned} \tag{12}$$

This simplifies for the first order random walk to:

$$\begin{aligned}
\overleftarrow{\boldsymbol{m}}_t &= \boldsymbol{a}_0 \\
\overleftarrow{\mathbf{P}}_t &= t\mathbf{Q} + \mathbf{Q}_0 \\
\overleftarrow{\mathbf{S}}_t &= (t\mathbf{Q} + \mathbf{Q}_0)\left((t+1)\mathbf{Q} + \mathbf{Q}_0\right)^{-1} \mathbf{Q} \\
\overleftarrow{\boldsymbol{a}}_t &= (t\mathbf{Q} + \mathbf{Q}_0)\left((t+1)\mathbf{Q} + \mathbf{Q}_0\right)^{-1} \boldsymbol{\alpha}_{t+1} + \overleftarrow{\boldsymbol{\Sigma}}_t \overleftarrow{\mathbf{P}}_t^{-1} \overleftarrow{\boldsymbol{m}}_t
\end{aligned} \tag{13}$$

Further, setting $\mathbf{Q}_0 = \mathbf{Q}$ (only in the artificial prior where we may alter $\gamma_0$ – see Briers et al. (2010, page 70)) gives us:

$$\begin{aligned}
\overleftarrow{\boldsymbol{\Sigma}}_t &= \mathbf{Q}^0 \\
\overleftarrow{\boldsymbol{a}}_t &= \mathbf{Q}^{-1}\boldsymbol{\alpha}_{t+1} + \frac{1}{t+1}\mathbf{Q}^{-1}\boldsymbol{a}_0
\end{aligned} \tag{14}$$

which makes no sense ( if I am not mistaken) but this is what they write as the intermediate results in the appendix of Fearnhead et al. (2010). Their end results though make sense. This yields the following backward version of equation (19):

$$\begin{aligned}
\widetilde{q}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right) &= \mathcal{N}\left(\boldsymbol{\alpha}_t \middle| \overleftarrow{\boldsymbol{\mu}}_t, \overleftarrow{\boldsymbol{\Sigma}}_t\right) \\
\overleftarrow{\boldsymbol{\Sigma}}_t^{-1} &= \mathbf{P}_t^{-1} + \mathbf{X}_t^\top \left(-\mathbf{G}_t\left(\mathbf{F}^{-1}\bar{\boldsymbol{\alpha}}_{t+1}\right)\right)\mathbf{X}_t + \mathbf{F}^{-1}\mathbf{Q}^{-1}\left(\mathbf{F}^{-1}\right)^\top \\
\overleftarrow{\boldsymbol{\mu}}_t &= \overleftarrow{\boldsymbol{\Sigma}}_t\left(\overleftarrow{\mathbf{P}}_t^{-1}\overleftarrow{\boldsymbol{m}}_t + \mathbf{F}^{-1}\mathbf{Q}^{-1}\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)} + \mathbf{X}_t^\top\left(\left(-\mathbf{G}_t\left(\mathbf{F}^{-1}\bar{\boldsymbol{\alpha}}_{t+1}\right)\right)\mathbf{X}_t\mathbf{F}^{-1}\bar{\boldsymbol{\alpha}}_{t+1} + \boldsymbol{g}_t\left(\mathbf{F}^{-1}\bar{\boldsymbol{\alpha}}_{t+1}\right)\right)\right)
\end{aligned} \tag{15}$$

which is (/should be) an approximation of:

$$\begin{aligned}
\widetilde{q}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right) &\propto g\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right) f\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)} \middle| \boldsymbol{\alpha}_t\right) \frac{\gamma_t\left(\boldsymbol{\alpha}_t\right)}{\gamma_{t+1}\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right)} \\
&\approx \widetilde{g}\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right) \mathcal{N}\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)} \middle| \mathbf{F}\boldsymbol{\alpha}_t, \mathbf{Q}\right) \frac{\mathcal{N}\left(\boldsymbol{\alpha}_t \middle| \overleftarrow{\boldsymbol{m}}_t, \overleftarrow{\mathbf{P}}_t\right)}{\mathcal{N}\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)} \middle| \overleftarrow{\boldsymbol{m}}_{t+1}, \overleftarrow{\mathbf{P}}_{t+1}\right)}
\end{aligned} \tag{16}$$

The weights are computed similarly to the forward filter using the backward proposal density if an equivelent version of we use an auxiliary particle filter. We can also use a bootstrap like filter with the importance density:

$$q\left(\boldsymbol{\alpha}_t \middle| \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}, \boldsymbol{y}_t\right) = \mathcal{N}\left(\boldsymbol{\alpha}_t \middle| \mathbf{F}^{-1}\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}, \mathbf{Q}\right) \tag{17}$$

**Combining / smoothing (algorithm 1**

We need to specify the importance density $\widetilde{q}\left(\cdot \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right)$ (see Fearnhead et al. (2010, page 453)). Again, we can use a similar idea to those in the appendix of Fearnhead et al. (2010) and choose:

$$\widetilde{q}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right) \propto \mathrm{P}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}\right) \frac{\mathrm{P}\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)} \middle| \boldsymbol{\alpha}_t\right)}{\gamma_{t+1}\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right)} \propto x \tag{18}$$

$$\begin{aligned}
\widetilde{q}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right) &= \mathcal{N}\left(\boldsymbol{\alpha}_t \middle| \overleftrightarrow{\boldsymbol{\mu}}_t, \overleftrightarrow{\boldsymbol{\Sigma}}_t\right) \\
\overleftrightarrow{\boldsymbol{\Sigma}}_t^{-1} &= \mathbf{Q}^{-1} + \mathbf{X}_t^{\top}\left(-\mathbf{G}_t\left(\bar{\boldsymbol{\alpha}}_t\right)\right)\mathbf{X}_t + \mathbf{F}^{-1}\mathbf{Q}^{-1}\left(\mathbf{F}^{-1}\right)^{\top} \\
\overleftrightarrow{\boldsymbol{\mu}}_t &= \overleftrightarrow{\boldsymbol{\Sigma}}_t\left(\mathbf{Q}^{-1}\boldsymbol{\alpha}_{t-1}^{(j)} + \mathbf{F}^{-1}\mathbf{Q}^{-1}\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)} + \mathbf{X}_t^{\top}\left(\left(-\mathbf{G}_t\left(\bar{\boldsymbol{\alpha}}_t\right)\right)\mathbf{X}_t\bar{\boldsymbol{\alpha}}_t + \boldsymbol{g}_t\left(\bar{\boldsymbol{\alpha}}_t\right)\right)\right)
\end{aligned} \tag{19}$$

where $\bar{\boldsymbol{\alpha}}_t$ can be a combined mean given the cloud means at time $t-1$ and $t+1$ or a mean for each of the two drawn particles in the $(j,k)$ pairs. This is to approximate:

$$\widetilde{q}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right) \propto \widetilde{g}\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right) f\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}\right) \frac{f\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)} \middle| \boldsymbol{\alpha}_t\right)}{\gamma_{t+1}\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right)} \tag{20}$$

We can also use a bootstrap like filter by sampling from:

$$\begin{aligned}
\widetilde{q}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right) &= \mathcal{N}\left(\boldsymbol{\alpha}_t \middle| \widetilde{\boldsymbol{m}}, \widetilde{\mathbf{S}}\right) \\
\widetilde{\mathbf{S}} &= \left(\mathbf{Q}^{-1} + \mathbf{F}^{-1}\mathbf{Q}^{-1}\left(\mathbf{F}^{-1}\right)^{\top}\right)^{-1} \\
\widetilde{\boldsymbol{m}} &= \widetilde{\mathbf{S}}\left(\mathbf{Q}^{-1}\mathbf{F}\boldsymbol{\alpha}_{t-1}^{(j)} + \mathbf{F}^{-1}\mathbf{Q}^{-1}\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right)
\end{aligned} \tag{21}$$

TODO: add what to do on time 0 and $d$.

**Algorithm 1** $\mathcal{O}(N)$ two filter smoother using the method in Fearnhead et al. (2010).

**Input:**

$\mathbf{Q}, \mathbf{Q}_0, \boldsymbol{a}_0, \mathbf{X}_1, \ldots, \mathbf{X}_d, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_d, R_1, \ldots, R_d, \mathbf{S}$

Let $\boldsymbol{\alpha}_t^{(i)}$ denote particle $i$ at time $t$, $w_t^{(i)}$ denote the weight of the particle and $\beta_t^{(i)}$ denote the re-sampling weight.

Importance density which optimally is (see Fearnhead et al. (2010, page 453)):

$$\widetilde{q}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right) = \mathrm{P}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right) \tag{22}$$

1: **procedure** FILTER FORWARD

2: Run a forward particle filter to get a particle swarm $\left\{\boldsymbol{\alpha}_t^{(j)}, w_t^{(j)}, \beta_t^{(j)}\right\}_{j=1,\ldots,N}$ approximating $\mathrm{P}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{1:t}\right)$ for $t = 0, 1, \ldots, d$. See algorithm 2.

3: **procedure** FILTER BACKWARDS

4: Run a similar backward filter to get $\left\{\widetilde{\boldsymbol{\alpha}}_t^{(k)}, \widetilde{w}_t^{(k)}, \widetilde{\beta}_t^{(k)}\right\}_{k=1,\ldots,N}$ approximating $\mathrm{P}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{t:d}\right)$ for $t = d, d-1, \ldots, 2$. See algorithm 3.

5: **procedure** SMOOTH (COMBINE)

6: **for** $t = 1, \ldots, d-1$ **do**

*Re-sample:*

7: $i = 1, 2, \ldots, N_s$ pairs of $(j_i, k_i)$ where each component is independently sampled using re-sampling weights $\beta_t^{(j)}$ and $\widetilde{\beta}_t^{(k)}$

*Propagate:*

8: Sample particles $\widehat{\boldsymbol{\alpha}}_t^{(i)}$ from importance density $\widetilde{q}\left(\cdot \middle| \boldsymbol{\alpha}_{t-1}^{(j_i)}, \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k_i)}\right)$.

*Re-weight:*

9: Assign each particle weights:

$$\widehat{w}_t^{(i)} \propto \frac{\mathrm{P}\left(\widehat{\boldsymbol{\alpha}}_t^{(i)} \middle| \boldsymbol{\alpha}_{t-1}^{(j_i)}\right) \mathrm{P}\left(\boldsymbol{y}_t \middle| \widehat{\boldsymbol{\alpha}}_t^{(i)}\right) \mathrm{P}\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k_i)} \middle| \widehat{\boldsymbol{\alpha}}_t^{(i)}\right) w_{t-1}^{(j_i)} \widetilde{w}_{t+1}^{(k_i)}}{\widetilde{q}\left(\widehat{\boldsymbol{\alpha}}_t^{(i)} \middle| \boldsymbol{\alpha}_{t-1}^{(j_i)}, \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k_i)}\right) \beta_t^{(j_i)} \widetilde{\beta}_t^{(k_i)} \gamma_{t+1}\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k_i)}\right)} \tag{23}$$

**Algorithm 2** Forward filter due to Pitt and Shephard (1999). You can compare with Doucet and Johansen (2009, page 20 and 25). The version and notation below is from Fearnhead et al. (2010, page 449).

**Input:**

1: Importance density and specification of weights are optimally given by:

$$\begin{gathered} q\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t\right) = \mathrm{P}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t\right) \\ \beta_t^{(j)} \propto \mathrm{P}\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}\right) w_{t-1}^{(j)} \end{gathered} \tag{24}$$

2: **for** $t = 1, \ldots, d$ **do**

3: **procedure** RESAMPLE

4: Compute re-sampling weights $\beta_t^{(j)}$ and re-sample according to $\beta_t^{(j)}$ to get indices $j_1, \ldots j_N$.

5: **procedure** PROPAGATE

6: Sample new particles $\boldsymbol{\alpha}_t^{(i)}$ using importance density $q\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j_i)}, \boldsymbol{y}_t\right)$.

7: **procedure** RE-WEIGHT

8: Re-weight particles using:

$$w_t^{(i)} \propto \frac{\mathrm{P}\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t^{(i)}\right) \mathrm{P}\left(\boldsymbol{\alpha}_t^{(i)} \middle| \boldsymbol{\alpha}_{t-1}^{(j_i)}\right) w_{t-1}^{(j_i)}}{q\left(\boldsymbol{\alpha}_t^{(i)} \middle| \boldsymbol{\alpha}_{t-1}^{(j_i)}, \boldsymbol{y}_t\right) \beta_t^{(j_i)}} \tag{25}$$

**Algorithm 3** Backwards filter.

**Input:**

A backwards filter distribution approximation:

$$\widetilde{p}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{t:T}\right) \propto \gamma_t\left(\boldsymbol{\alpha}_t\right) \mathrm{P}\left(\boldsymbol{y}_{t:T} \middle| \boldsymbol{\alpha}_t\right) \tag{26}$$

with an artificial prior distribution $\gamma_t\left(\boldsymbol{\alpha}_t\right)$. This is introduced as $\mathrm{P}\left(\boldsymbol{y}_{t:T} \middle| \boldsymbol{\alpha}_t\right)$ is not a density function in $\boldsymbol{\alpha}_t$.

Importance density and specification of weights (Fearnhead et al. (2010, page 451 – maybe look in the example in the appendix)):

$$\widetilde{q}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right) \widetilde{\beta}_t^{(k)} \approx \gamma_t\left(\boldsymbol{\alpha}_t\right) \mathrm{P}\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right) \mathrm{P}\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)} \middle| \boldsymbol{\alpha}_t\right) \frac{\widetilde{w}_{t+1}^{(k)}}{\gamma_{t+1}\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right)} \tag{27}$$

where I figure that we want (the first follows from Briers et al. (2010, page 74)):

$$\begin{aligned} \widetilde{q}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right) &\propto \mathrm{P}\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right) \mathrm{P}\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)} \middle| \boldsymbol{\alpha}_t\right) \frac{\gamma_t(\boldsymbol{\alpha}_t)}{\gamma_{t+1}\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right)} \\ \widetilde{\beta}_t^{(k)} &\propto \widetilde{p}\left(\boldsymbol{y}_t \middle| \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right) \widetilde{w}_{t+1}^{(k)} \end{aligned} \tag{28}$$

1: **for** $t = d-1, \ldots, 1$ **do**

2: **procedure** RESAMPLE

3: Compute re-sampling weights $\widetilde{\beta}_t^{(k)}$ and re-sample according to $\widetilde{\beta}_t^{(k)}$ to get indices $k_1, \ldots k_N$.

4: **procedure** PROPAGATE

5: Sample new particles $\boldsymbol{\alpha}_t^{(i)}$ using importance density $\widetilde{q}\left(\boldsymbol{\alpha}_t \middle| \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k_i)}, \boldsymbol{y}_t\right)$.

6: **procedure** RE-WEIGHT

7: Re-weight particles using (see Briers et al. (2010, page 72) and add the ratios of weights and re-sampling weights):

$$\widetilde{w}_t^{(i)} \propto \frac{\mathrm{P}\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t^{(i)}\right) \mathrm{P}\left(\boldsymbol{\alpha}_{t+1}^{(k_i)} \middle| \boldsymbol{\alpha}_t^{(i)}\right) w_{t+1}^{(k_i)} \gamma_t\left(\boldsymbol{\alpha}_t^{(i)}\right)}{q\left(\boldsymbol{\alpha}_t^{(i)} \middle| \boldsymbol{\alpha}_{t+1}^{(k_i)}, \boldsymbol{y}_t\right) \beta_t^{(k_i)} \gamma_{t+1}\left(\boldsymbol{\alpha}_{t+1}^{(k_i)}\right)} \tag{29}$$

$$
\begin{aligned}
\widetilde{\mathbf{F}} &= \begin{pmatrix} \mathbf{R} & \mathbf{0} & \mathbf{0} \\ \mathbf{FR} & \mathbf{R} & \mathbf{0} \\ \mathbf{F}^2\mathbf{R} & \mathbf{FR} & \mathbf{R} \end{pmatrix} \\
\widetilde{\mathbf{Q}} &= \widetilde{\mathbf{F}} \begin{pmatrix} \mathbf{Q} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{Q} \end{pmatrix} \widetilde{\mathbf{F}}^\top \\
\widetilde{\boldsymbol{a}}_0 &= \begin{pmatrix} \mathrm{E}\left(\boldsymbol{a}_{t-1}\right) \\ \mathbf{F}\mathrm{E}\left(\boldsymbol{a}_{t-1}\right) \\ \mathbf{F}^2\mathrm{E}\left(\boldsymbol{a}_{t-1}\right) \end{pmatrix} = \begin{pmatrix} \mathbf{F}^{-2}\mathrm{E}\left(\boldsymbol{a}_{t+1}\right) \\ \mathbf{F}^{-1}\mathrm{E}\left(\boldsymbol{a}_{t+1}\right) \\ \mathrm{E}\left(\boldsymbol{a}_{t+1}\right) \end{pmatrix} \\
\begin{pmatrix} \boldsymbol{\alpha}_{t-1} \\ \boldsymbol{\alpha}_t \\ \boldsymbol{\alpha}_{t+1} \end{pmatrix} &\sim \mathcal{N}\left(\widetilde{\boldsymbol{a}}_0, \widetilde{\mathbf{Q}}\right)
\end{aligned}
\tag{30}
$$

$$
\begin{aligned}
&\mathrm{E}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{1:(t-1)}, \boldsymbol{\alpha}_{(t+1):d}\right) \\
&\qquad = \left(\widetilde{\boldsymbol{a}}_0\right)_2 + \mathrm{E}\left(\left(\widetilde{\boldsymbol{a}}_0\right)_2, \left(\widetilde{\boldsymbol{a}}_0\right)_{\{1,3\}}\right) \mathrm{Var}\left(\left(\left(\widetilde{\boldsymbol{a}}_0\right)_{1,3}\right)\right)^{-1} \left(\begin{pmatrix} \boldsymbol{\alpha}_{t-1} \\ \boldsymbol{\alpha}_{t+1} \end{pmatrix} - \left(\widetilde{\boldsymbol{a}}_0\right)_{\{1,3\}}\right)
\end{aligned}
\tag{31}
$$

# 2 Literature review

**Doucet and Johansen (2009)**

- Do re-sample and likely use *Systematic Resampling* from Kitagawa (1996). If you use it, then read up on Douc and Cappé (2005).
- Use threshold on *effective sample size* to deside when to re-sample.
- Use *Auxiliary Particle Filtering*. That is, use the information from the $t+1$ outcome when re-computing the weights (*sample before re-sampling*). See Carpenter et al. (1999), Pitt and Shephard (2001) and Pitt and Shephard (1999).
- Likely use MCMC Moves (re-sample moves) or block sampling limit degeneracy problem. See text for references.
- Rao-Blackwellised Particle Filtering: always exploit when you can have closed form solutions like when parts of the problem is (conditional) multivariate Gaussian. See the text and Andrieu and Doucet (2002) for examples.
- Two filter smother is likely the way to go. It may be better '*the support of the smoothed estimate differs substantially from that of the filtering estimate*'. The cost is $\mathcal{O}\left(N^2T\right)$ but can be reduced to $\mathcal{O}\left(NT\right)$ using the methods in Fearnhead et al. (2010) and Briers et al. (2005).

**Andrieu and Doucet (2002)**

Covers a RaoBlackwellized particle filtering where the state is linear and Gaussian and observations are (potentially) non-linear and non-Gaussian.

- The filter exploits closed form solutions from Gaussian part.
- It is not an auxiliary particle filter so w likely want to change this.
- Has a low memory requirement as we do not need to store a smooth covariance matrix for each particle.
- Mentions a MCMC move approach for sampling to deal with degeneracy. See De Jong (1997).

**Fearnhead et al. (2010)**

Covers a two filter smoother:

- Run an *auxiliary particle filtering* forward and backward filter.
- The backwards filter uses samples from '*marginal smoothing densities direct than re-weight those drawn from another distribution*'. This solves degeneracy problems like those that occurs for a second order random walk. They use that the forward and backwards sets of particle clouds (or swarms) are independent. Then we sample in the combination step using particles from $t-1$ from the forward filter and particles from $t+1$ from the backwards filter. The computational cost in $\mathcal{O}\left(N^2T\right)$.
- Get an $\mathcal{O}\left(NT\right)$ filter by sampling a forward particle and backward particle pair at each time $t$ while combining.
- Shows examples which are fully or partially linear Gaussian models. Derivations are in the appendix.
- Points out in the discussion that we likely want more particles in smoothing/combination step than in the forward and backwards filters.

# References


Christophe Andrieu and Arnaud Doucet. Particle filtering for partially observed gaussian state space models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 64(4):827–836, 2002.

Mark Briers, Arnaud Doucet, and Sumeetpal S Singh. Sequential auxiliary particle belief propagation. In *Information Fusion, 2005 8th International Conference on*, volume 1, pages 8–pp. IEEE, 2005.

Mark Briers, Arnaud Doucet, and Simon Maskell. Smoothing algorithms for state–space models. *Annals of the Institute of Statistical Mathematics*, 62(1): 61–89, 2010.

James Carpenter, Peter Clifford, and Paul Fearnhead. Improved particle filter for nonlinear problems. *IEE Proceedings-Radar, Sonar and Navigation*, 146 (1):2–7, 1999.

Piet De Jong. The scan sampler for time series models. *Biometrika*, 84(4): 929–937, 1997.

Randal Douc and Olivier Cappé. Comparison of resampling schemes for particle filtering. In *Image and Signal Processing and Analysis, 2005. ISPA 2005. Proceedings of the 4th International Symposium on*, pages 64–69. IEEE, 2005.

Arnaud Doucet and Adam M Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of nonlinear filtering*, 12(656-704): 3, 2009.

Paul Fearnhead, David Wyncoll, and Jonathan Tawn. A sequential smoothing algorithm with linear computational cost. *Biometrika*, 97(2):447–464, 2010.

Nikolas Kantas, Arnaud Doucet, Sumeetpal S Singh, Jan Maciejowski, Nicolas Chopin, et al. On particle methods for parameter estimation in state-space models. *Statistical science*, 30(3):328–351, 2015.

Genshiro Kitagawa. Monte carlo filter and smoother for non-gaussian nonlinear state space models. *Journal of computational and graphical statistics*, 5(1): 1–25, 1996.

Michael K Pitt and Neil Shephard. Filtering via simulation: Auxiliary particle filters. *Journal of the American statistical association*, 94(446):590–599, 1999.

Michael K Pitt and Neil Shephard. Auxiliary variable based particle filters. In *Sequential Monte Carlo methods in practice*, pages 273–293. Springer, 2001.