Particle filters in the dynamichazard package

Benjamin Christoffersen

August 15, 2018

This is vignette covers the particle filter for the `dynamichazard` package in `R`. Some prior knowledge of particle filters is required. See Doucet and Johansen (2009) provides a tutorial on particle filters and Kantas et al. (2015) covers parameter estimation with particle filters. See also Cappé et al. (2005) for a general introduction to Hidden Markov models. This vignette relies heavily on Fearnhead et al. (2010).

1 Method

The model is

$$
\begin{aligned}
&y_{it} \sim \mathrm{P}\left(y_{it} \middle| \eta_{it}\right) \\
&\boldsymbol{\eta}_t = \mathbf{X}_t\mathbf{R}^+\boldsymbol{\alpha}_t + \boldsymbol{o}_t + \mathbf{Z}_t\boldsymbol{\omega} && && i = 1, \dots, n_t \\
&\boldsymbol{\alpha}_t = \mathbf{F}\boldsymbol{\alpha}_{t-1} + \mathbf{R}\boldsymbol{\epsilon}_t && \boldsymbol{\epsilon}_t \sim N\left(\mathbf{0}, \mathbf{Q}\right)' && t = 1, \dots, d \\
& && \boldsymbol{\alpha}_0 \sim N\left(\boldsymbol{a}_0, \mathbf{Q}_0\right)
\end{aligned} \tag{1}
$$

where I denote the conditional densities as $\boldsymbol{y}_t \sim g_t\left(\cdot \middle| \boldsymbol{\alpha}_t\right) = g\left(\cdot \middle| \mathbf{X}_t\mathbf{R}^+\boldsymbol{\alpha}_t + \mathbf{o}_t\right)$ and $\boldsymbol{\alpha}_t \sim f\left(\cdot \middle| \boldsymbol{\alpha}_{t-1}\right)$. We are in a survival analysis setting where the simplest model has an indicator of death of individual i in time t such that $y_{it} \in \{0, 1\}$, η_{it} is the linear predictor, and we use the logistic function as the link function. For each $t = 1, \dots, d$, we a have risk set given by $R_t \subseteq \{1, 2, \dots, n\}$. Further, we let $n_t = |R_t|$ and $n_{\max} = \max_{t=1,\dots d} = n_t$. The observed outcomes are denoted by $\boldsymbol{y}_t = \left\{y_{it}\right\}_{i \in R_t}$. $\mathbf{X}_t$ is the design matrix of the covariates and $\boldsymbol{\alpha}_t$ is the vector of time-varying coefficients. The $\mathbf{Z}_t$ is the design matrix for the fixed effects and $\boldsymbol{\omega}$ are the corresponding coefficients.

Superscript + denotes the Moore-Penrose inverse, the i'th row of $\mathbf{X}_t$ is $\boldsymbol{x}_{it}$, $\boldsymbol{x}_{it}, \boldsymbol{\epsilon}_t \in \mathbb{R}^r$, $\boldsymbol{\alpha}_t \in \mathbb{R}^p$, $\mathbf{F} \in \mathbb{R}^{p \times p}$ and is invertible, $\mathbf{Q} \in \mathbb{R}^{r \times r}$ is a positive definite matrix, $\boldsymbol{o}_t$ are know offsets, and $\mathbf{R} \in \{0, 1\}^{p \times r}$ with $p \geq r$ contains a subset of the columns of $\mathbf{I}_p$ identity matrix with no duplicate columns in $\mathbf{R}$. The latter implies that $\mathbf{R}^+ = \mathbf{R}^\top$ and $\mathbf{R}$ is left inverse (i.e., $\mathbf{R}^\top\mathbf{R} = \mathbf{I}_r$). $\mathbf{R}\mathbf{R}^\top$ is a $p \times p$ diagonal matrix with r diagonal entries with value 1 and $p - r$ with value zero. The problems we are looking at have $n_{\max} \gg p \geq r$ (e.g. $n_{\max} = 100000$ and $r = 5$). We will let

$$\boldsymbol{\xi}_t = \mathbf{R}^+\boldsymbol{\alpha}_t$$

I will use a particle filter and smoother to get smoothed estimates of $\boldsymbol{\alpha}_1, \dots, \boldsymbol{\alpha}_d$ given the outcomes $\boldsymbol{y}_{1:d} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \dots, \boldsymbol{y}_d\}$ and use an EM-algorithm to estimate $\mathbf{Q}$ and $\boldsymbol{a}_0$. One choice of smoother is the generalized two-filter smoothing in Fearnhead et al. (2010) and Briers et al. (2009). The rest of vignette is structured as follows: first I cover the particle filter and smoother. Then I cover the EM-algorithm and other miscellaneous topics. I will end with some points about the implementation.

Considerations

Algorithm 1 shows one of the generalized two-filter smoother from Fearnhead et al. (2010). It requires that we specify the following proposal distributions 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

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

with an artificial prior distribution $\gamma_t\left(\boldsymbol{\alpha}_t\right)$.

Given the models of interest we have that

- Evaluating $g_t\left(\boldsymbol{y_t} \middle| \boldsymbol{\alpha}_t\right)$ is an expensive operation as $n_{\max} \gg r$ and it has a $\mathcal{O}\left(n_{\max} r\right)$ computational cost. Any $\mathcal{O}\left(n_{\max}\right)$ operation is going to take considerable time.
- Evaluating $f\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}\right)$ is cheap, it is done in in closed form, and sampling from these distribution can be done in closed form.

We can also notice that the second example in Fearnhead et al. (2010) is close to the model here though with $n_{\max} = 1$. The following sections will closely follow the example shown in Fearnhead et al. (2010) and the appendix of the paper.

Forward filter (Algorithm 2)

This section will cover some options for Algorithm 2. Let $\mathcal{N}\left(\cdot \middle| \cdot, \cdot\right)$ denote a multivariate normal distribution. We can select the proposal density as

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

which we can sample from in $\mathcal{O}\left(Np^2\right)$ time if we have a pre-computed Cholesky decomposition of $\mathbf{Q}$. This is often called the *bootstrap filter*. Another option is to use normal approximation of $\boldsymbol{y}_t$'s conditional density for some value $\bar{\boldsymbol{\alpha}}_{t-1}$ and $\bar{\boldsymbol{\alpha}}_t = \mathbf{F}\bar{\boldsymbol{\alpha}}_{t-1}$

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

where

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

to get

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

Thus, we need to sample from

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

We can select $\bar{\boldsymbol{\alpha}}_{t-1}$ as the weighted mean given the particle cloud at time $t-1$ (that is, $\{\boldsymbol{\alpha}_{t-1}^{(1)}, \boldsymbol{\alpha}_{t-1}^{(2)}, \dots \boldsymbol{\alpha}_{t-1}^{(N)}\}$). Observing that $P\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}, \boldsymbol{y}_t\right)$ is log-concave, then following Doucet et al. (2000) we can set $\bar{\boldsymbol{\mu}}_t^{(0)} = \mathbf{F}\bar{\boldsymbol{\alpha}}_{t-1}$ and for $k = 1, \dots$

1. Set $\bar{\boldsymbol{\xi}}_t^{(k-1)} = \mathbf{R}^+\bar{\boldsymbol{\mu}}_t^{(k-1)}$ and let $\bar{\boldsymbol{\mu}}_t^{(k)} = \boldsymbol{\mu}_t\left(\bar{\boldsymbol{\alpha}}_{t-1}, \bar{\boldsymbol{\xi}}_t^{(k-1)}\right)$.
2. Stop if $\left|\bar{\boldsymbol{\mu}}_t^{(k)} - \bar{\boldsymbol{\mu}}_t^{(k-1)}\right| / \left|\bar{\boldsymbol{\mu}}_t^{(k-1)}\right| < \epsilon$. Otherwise set $k \leftarrow k+1$ and repeat 1.

for some small ϵ. The final functions $\boldsymbol{\mu}_t\left(\cdot, \bar{\boldsymbol{\xi}}_t^{(k)}\right)$ and $\boldsymbol{\Sigma}_t\left(\bar{\boldsymbol{\xi}}_t^{(k)}\right)$ defined in Equation (8) are then used as the proposal distributions. Further, $\boldsymbol{\mu}_t\left(\bar{\boldsymbol{\alpha}}_{t-1}, \bar{\boldsymbol{\xi}}_t^{(k)}\right)$ is the mode of $P\left(\boldsymbol{\alpha}_t \middle| \bar{\boldsymbol{\alpha}}_{t-1}, \boldsymbol{y}_t\right)$. Similar steps can be taken for the next proposal distributions. We will drop the argument in the functions in equations similar to Equation (8) and we will not go into details as the steps are very similar. See also Section 6.2. The downside is an $\mathcal{O}\left(n_{\max}p^2 + p^3\right)$ computational cost though independent of the number of particles, N. The total cost of sampling is $\mathcal{O}\left(n_{\max}p^2 + p^3 + Np^2\right)$.

Another option is to set $\bar{\boldsymbol{\alpha}}_{t-1} = \boldsymbol{\alpha}_{t-1}^{(j)}$ for each particle $j = 1, 2, \ldots, N$ in the particle cloud at time $t-1$. This is similar to the second order random walk example in Fearnhead et al. (2010). This will improve the Taylor expansion but yields an $\mathcal{O}\left(N\left(n_{\max}p^2 + p^3\right)\right)$ computational cost. The extra $Nn_{\max}p^2$ factor makes this much slower.

Next, we have the re-sampling weights. A simple solution is not to use an auxiliary particle filter as in the examples of Fearnhead et al. (2010) 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

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

where $q\left(\boldsymbol{\mu}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t\right)$ and $\boldsymbol{\mu}_t$ are from Equation (8) computed with $\boldsymbol{\alpha}_{t-1}^{(j)}$. The re-sampling weights will differ as we condition of different particle $\boldsymbol{\alpha}_{t-1}^{(j)}$. This comes at an $\mathcal{O}\left(Np^2\right)$ computational cost assuming that we are using (8) already in the sampling step. Otherwise it has the same computational cost as mentioned when I covered the proposal distribution since we have to make the Taylor approximation anyway.

Backward filter (Algorithm 3)

We need to specify the artificial prior $\gamma_t(\boldsymbol{\alpha}_t)$. Briers et al. (2009, page 69 and 70) provides recommendation on the selection. This leads to

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

Following Fearnhead et al. (2010) then we end with

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

which simplifies for the first order random walk to

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

Further, setting $\mathbf{Q}_0 = \mathbf{Q}$ (only in the artificial prior where we may alter γ_0 – see Briers et al., 2009, page 70) gives us

$$\begin{aligned}
\overleftarrow{\mathbf{S}}_t &= \frac{t+1}{t+2}\mathbf{Q} \\
\overleftarrow{\boldsymbol{a}}_t &= \frac{t+1}{t+2}\boldsymbol{\alpha}_{t+1} + \frac{1}{t+2}\boldsymbol{a}_0
\end{aligned} \tag{14}$$

We get the following backward version of Equation (8)

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

Notice that the dimension is now $p \geq r$. As an example, we can look at a second order uni-variate auto-regressive model

$$\mathbf{F} = \begin{pmatrix} \theta_1 & \theta_2 \\ 1 & 0 \end{pmatrix}, \quad \mathbf{R} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad \mathbf{Q} = \begin{pmatrix} \sigma^2 \end{pmatrix}, \quad \mathbf{Q}_0 = \begin{pmatrix} s_1^2 & s_{12} \\ s_{12} & s_2^2 \end{pmatrix}$$

here

$$\mathbf{F}^{\top}\mathbf{R}^{+\top}\mathbf{Q}^{-1}\mathbf{R}^{+}\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)} = \begin{pmatrix} \theta_1 \\ \theta_2 \end{pmatrix}\begin{pmatrix} \sigma^{-2} & 0 \end{pmatrix}\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)} = \begin{pmatrix} \sigma^{-2}\theta_1\widetilde{\xi}_{t+1}^{(k)} \\ \sigma^{-2}\theta_2\widetilde{\xi}_{t+1}^{(k)} \end{pmatrix}$$

We also find that

$$\mathbf{F}^{\top}\mathbf{R}^{+\top}\mathbf{Q}^{-1}\mathbf{R}^{+}\mathbf{F} = \mathbf{F}^{\top}\mathbf{R}\mathbf{Q}^{-1}\mathbf{R}^{\top}\mathbf{F} = \begin{pmatrix} \theta_1^2\sigma^{-2} & \theta_1\theta_2\sigma^{-2} \\ \theta_1\theta_2\sigma^{-2} & \theta_2^2\sigma^{-2} \end{pmatrix}$$

Further,

$$\overleftarrow{\mathbf{P}}_t = \mathbf{F}^t\mathbf{Q}_0\mathbf{F}^{t\top} + \sum_{i=1}^{t}\mathbf{F}^{(i-1)}\mathbf{R}\mathbf{Q}\mathbf{R}^{\top}\mathbf{F}^{(i-1)\top} = \mathbf{F}^t\mathbf{Q}_0\mathbf{F}^{t\top} + \sum_{i=1}^{t}(\mathbf{F}^{(i-1)})_{\cdot 1}\sigma^2(\mathbf{F}^{(i-1)\top})_{1\cdot}$$

where subscript $\mathbf{Z}_{k\cdot}$ is the k'th row vector and $\mathbf{Z}_{\cdot k}$ is the k'th column vector. Here, the first term has full rank if $\mathbf{Q}_0$ and $\mathbf{F}$ has full rank, is not sparse, and tends toward zero if the largest eigenvalue of $\mathbf{F}$ is less than one. Further the later terms has rank 1 and are not sparse either. The other terms and matrices

only have $r \times r$ non-zero entry due to multiplication with $\mathbf{R}$ or $\mathbf{R}^\top$ which is the first entry for vectors and the $(1,1)$ entry for matrices in the above example. We can use Equation (15) to approximate

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

Can we sample from $\mathcal{N}\left(\cdot \middle| \mathbf{R}\overleftarrow{\boldsymbol{\mu}}_t, \mathbf{R}\overleftarrow{\boldsymbol{\Sigma}}_t\mathbf{R}^\top\right)$ and set the remaining $p - r$ entries corresponding to the non-included identity matrix columns in $\mathbf{R}$ to $\overleftarrow{\boldsymbol{\mu}}_t$ for each particle $\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}$?

The re-sampling weights can be computed similarly to the forward filter using the backward transition density in Equation (12) in the numerator. We can also use a bootstrap like filter where the proposal distribution as in Equation (12).

Combining / smoothing (Algorithm 1)

We need to specify the proposal distribution $\widetilde{q}\left(\cdot \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right)$ (see Fearnhead et al. (2010, page 453)). Again, we can make a second order Taylor expansion as in Fearnhead et al. (2010) and choose

$$\begin{aligned}\widetilde{q}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right) &= \mathcal{N}\left(\boldsymbol{\alpha}_t \middle| \overleftrightarrow{\boldsymbol{\mu}}_t, \overleftrightarrow{\boldsymbol{\Sigma}}_t\right) \\ \overleftrightarrow{\boldsymbol{\Sigma}}_t^{-1} &= \mathbf{R}^{+\top}\mathbf{Q}^{-1}\mathbf{R}^{+} + \mathbf{R}\mathbf{X}_t^\top\left(-\mathbf{G}_t\left(\bar{\boldsymbol{\xi}}_t + \boldsymbol{o}_t\right)\right)\mathbf{X}_t\mathbf{R}^\top + \mathbf{F}^\top\mathbf{R}^{+\top}\mathbf{Q}^{-1}\mathbf{R}^{+}\mathbf{F} \\ \overleftrightarrow{\boldsymbol{\mu}}_t &= \overleftrightarrow{\boldsymbol{\Sigma}}_t\left(\mathbf{R}^{+\top}\mathbf{Q}^{-1}\mathbf{R}^{+}\mathbf{F}\boldsymbol{\alpha}_{t-1}^{(j)} + \mathbf{F}^\top\mathbf{R}^{+\top}\mathbf{Q}^{-1}\mathbf{R}^{+}\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)} + \mathbf{R}\mathbf{X}_t^\top\left(-\mathbf{G}_t\left(\bar{\boldsymbol{\xi}}_t + \boldsymbol{o}_t\right)\mathbf{X}_t\bar{\boldsymbol{\xi}}_t - \boldsymbol{o}_t + \boldsymbol{g}_t\left(\bar{\boldsymbol{\xi}}_t + \boldsymbol{o}_t\right)\right)\right)\end{aligned} \tag{17}$$

where $\bar{\boldsymbol{\xi}}_t$ 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_i, k_i) pairs. This is to approximate

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

Can we sample from $\mathcal{N}\left(\cdot \middle| \mathbf{R}\overleftrightarrow{\boldsymbol{\mu}}_t, \mathbf{R}\overleftrightarrow{\boldsymbol{\Sigma}}_t^{-1}\mathbf{R}^\top\right)$ and set just the remaining $p - r$ entries corresponding to the non-included identity matrix columns in $\mathbf{R}$ to $\overleftrightarrow{\boldsymbol{\mu}}_t$ for each particle pair or maybe $\boldsymbol{\alpha}_{t-1}^{(j)}$?

We can also use a bootstrap like filter by sampling from

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

$\overleftrightarrow{\boldsymbol{\Sigma}}_t$ may not have full rank. Take E.g, a third order uni-variate model. In this case

$$\mathbf{R}^{+\top}\mathbf{Q}^{-1}\mathbf{R}^{+} = \begin{pmatrix} \sigma^{-2} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix},$$

$$\mathbf{R}\mathbf{X}_t^{\top}\left(-\mathbf{G}_t\left(\bar{\boldsymbol{\xi}}_t\right)\right)\mathbf{X}_t\mathbf{R}^{\top} = \begin{pmatrix} z & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix},$$

$$\mathbf{F}\mathbf{R}^{+\top}\mathbf{Q}^{-1}\mathbf{R}^{+}\mathbf{F} = \sigma^{-2}\begin{pmatrix} \theta_1^2 & \theta_1\theta_2 & \theta_1\theta_3 \\ \theta_1\theta_2 & \theta_2^2 & \theta_2\theta_3 \\ \theta_1\theta_3 & \theta_2\theta_3 & \theta_3^2 \end{pmatrix}$$

The latter only has rank 1. I gather the current solution requires a full rank matrix...
I gather we have to extend the framework to sample pairs at e.g., time $t-1$ and $t+2$ and then sample state vectors at time t and $t + 1$ as mentioned in the discussion in of Fearnhead et al. (2010) (If I get their argument correctly).

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

Input:

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

Proposal distribution which optimally is (see Fearnhead et al. (2010, page 453))

$$\widetilde{q}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right) = \mathrm{P}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right) \tag{19}$$

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.

procedure FILTER FORWARD

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

procedure FILTER BACKWARDS

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

procedure SMOOTH (COMBINE)

for $t = 1, \ldots, N$ **do**

Re-sample

$i = 1, 2, \ldots, N_s$ 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

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

Re-weight

Assign each particle weights

$$\widehat{w}_t^{(i)} \propto \frac{f\left(\widehat{\boldsymbol{\alpha}}_t^{(i)} \middle| \boldsymbol{\alpha}_{t-1}^{(j_i)}\right) g_t\left(\boldsymbol{y}_t \middle| \widehat{\boldsymbol{\alpha}}_t^{(i)}\right) f\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k_i)} \middle| \widehat{\boldsymbol{\alpha}}_t^{(i)}\right) w_{t-1}^{(j_i)} \widetilde{w}_{t+1}^{(k_i)}}{\widetilde{q}\left(\widehat{\boldsymbol{\alpha}}_t^{(i)} \middle| \boldsymbol{\alpha}_{t-1}^{(j_i)}, \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k_i)}\right) \beta_t^{(j_i)} \widetilde{\beta}_t^{(k_i)} \gamma_{t+1}\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k_i)}\right)} \tag{20}$$

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:

Proposal distribution and specification of weights are optimally given by

$$\begin{aligned} q\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t\right) &= \mathrm{P}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}, \boldsymbol{y}_t\right) \\ \beta_t^{(j)} &\propto \mathrm{P}\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}\right) w_{t-1}^{(j)} \end{aligned} \tag{21}$$

1: Sample $\boldsymbol{\alpha}_0^{(1)}, \ldots, \boldsymbol{\alpha}_0^{(N_f)}$ particles from $\mathcal{N}\left(\cdot \middle| \boldsymbol{a}_0, \mathbf{Q}_0\right)$ and set the weights $w_0^{(1)}, \ldots, w_0^{(N_f)}$ to $1/N_f$.
2: **for** $t = 1, \ldots, d$ **do**
3: **procedure** RE-SAMPLE
4: Compute re-sampling weights $\beta_t^{(j)}$ and re-sample according to $\beta_t^{(j)}$ to get indices $j_1, \ldots j_N$. If we do not re-sample then set $\beta_t^{(j)} = 1$.
5: **procedure** PROPAGATE
6: Sample new particles $\boldsymbol{\alpha}_t^{(i)}$ using the proposal distribution $q\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j_i)}, \boldsymbol{y}_t\right)$.
7: **procedure** RE-WEIGHT
8: Re-weight particles using ($w_{t-1}^{(j_i)}/\beta_t^{(j_i)}$ is added due to the auxiliary particle filter)

$$w_t^{(i)} \propto \frac{g_t\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t^{(i)}\right) f\left(\boldsymbol{\alpha}_t^{(i)} \middle| \boldsymbol{\alpha}_{t-1}^{(j_i)}\right) w_{t-1}^{(j_i)}}{q\left(\boldsymbol{\alpha}_t^{(i)} \middle| \boldsymbol{\alpha}_{t-1}^{(j_i)}, \boldsymbol{y}_t\right) \beta_t^{(j_i)}} \tag{22}$$

Algorithm 3 Backwards filter. See Briers et al. (2009) and Fearnhead et al. (2010).

Input:

A backwards filter distribution approximation

$$\widetilde{p}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{t:d}\right) \propto \gamma_t\left(\boldsymbol{\alpha}_t\right) \mathrm{P}\left(\boldsymbol{y}_{t:d} \middle| \boldsymbol{\alpha}_t\right) \tag{23}$$

with an artificial prior distribution $\gamma_t\left(\boldsymbol{\alpha}_t\right)$.
Proposal distribution and specification of weights (Fearnhead et al. (2010, page 451 – look in the example in the appendix))

$$\widetilde{q}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right) \widetilde{\beta}_t^{(k)} \approx \gamma_t\left(\boldsymbol{\alpha}_t\right) \mathrm{P}\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right) \mathrm{P}\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)} \middle| \boldsymbol{\alpha}_t\right) \frac{\widetilde{w}_{t+1}^{(k)}}{\gamma_{t+1}\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right)} \tag{24}$$

where we want (see Briers et al. (2009, page 74))

$$\begin{aligned} \widetilde{q}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_t, \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right) &\propto \mathrm{P}\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right) \mathrm{P}\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)} \middle| \boldsymbol{\alpha}_t\right) \frac{\gamma_t(\boldsymbol{\alpha}_t)}{\gamma_{t+1}\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right)} \\ \widetilde{\beta}_t^{(k)} &\propto \widetilde{p}\left(\boldsymbol{y}_t \middle| \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k)}\right) \widetilde{w}_{t+1}^{(k)} \end{aligned} \tag{25}$$

1: Sample $\widetilde{\boldsymbol{\alpha}}_{d+1}^{(1)}, \ldots, \widetilde{\boldsymbol{\alpha}}_{d+1}^{(N_f)}$ particles from $\gamma_{d+1}(\cdot)$ and set the weights $\widetilde{w}_{d+1}^{(1)}, \ldots, w_{d+1}^{(N_f)}$ to $1/N_f$.
2: **for** $t = d, \ldots, 1$ **do**
3: **procedure** RE-SAMPLE
4: Compute re-sampling weights $\widetilde{\beta}_t^{(k)}$ and re-sample according to $\widetilde{\beta}_t^{(k)}$ to get indices $k_1, \ldots k_N$. If we do not re-sample then set $\widetilde{\beta}_t^{(k)} = 1$.
5: **procedure** PROPAGATE
6: Sample new particles $\widetilde{\boldsymbol{\alpha}}_t^{(i)}$ using the proposal distribution $\widetilde{q}\left(\boldsymbol{\alpha}_t \middle| \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k_i)}, \boldsymbol{y}_t\right)$.
7: **procedure** RE-WEIGHT
8: Re-weight particles using (see Briers et al. (2009, page 72) and $\widetilde{w}_{t+1}^{(k_i)} / \beta_t^{(k_i)}$ is added due to the auxiliary particle filter)

$$\widetilde{w}_t^{(i)} \propto \frac{g_t\left(\boldsymbol{y}_t \middle| \widetilde{\boldsymbol{\alpha}}_t^{(i)}\right) f\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k_i)} \middle| \widetilde{\boldsymbol{\alpha}}_t^{(i)}\right) \gamma_t\left(\widetilde{\boldsymbol{\alpha}}_t^{(i)}\right) \widetilde{w}_{t+1}^{(k_i)}}{q\left(\widetilde{\boldsymbol{\alpha}}_t^{(i)} \middle| \widetilde{\boldsymbol{\alpha}}_{t+1}^{(k_i)}, \boldsymbol{y}_t\right) \gamma_{t+1}\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(k_i)}\right) \beta_t^{(k_i)}} \tag{26}$$

2 Log likelihood evaluation

We can evaluate the log likelihood for a particular value of $\boldsymbol{\theta} = \{\mathbf{Q}, \mathbf{Q}_0, \boldsymbol{a}_0, \mathbf{F}\}$ as described in Doucet and Johansen (2009, page 5) and Malik and Pitt (2011, page 193) using the forward particle filter shown in Algorithm 2.

3 Parameter inference

In this section I first show an example of parameter estimation the first order random walk using EM-algorithm (Dempster et al., 1977). Then I cover the general vector auto-regression model and estimating the fixed effects. Lastly, I will turn to estimation of observed information matrix.

The formulas for parameter estimation for the first order random with the are particularly simple. We need to estimate $\mathbf{Q}$ and $\boldsymbol{a}_0$ elements of $\boldsymbol{\varphi} = \{\mathbf{Q}, \mathbf{Q}_0, \boldsymbol{a}_0\}$. We do this by running Algorithm 1 for the current $\boldsymbol{\varphi}$. We compute following to do so

$$
\begin{aligned}
\boldsymbol{t}_t^{(\boldsymbol{\varphi})} &= \int_{\boldsymbol{\alpha}_t} \boldsymbol{\alpha}_t \mathrm{P}_{\boldsymbol{\varphi}}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{1:d}\right) \partial \boldsymbol{\alpha}_t \approx \sum_{i=1}^{N_s} \widehat{\boldsymbol{\alpha}}_t^{(i)} \widehat{w}_t^{(i)} \\
\boldsymbol{T}_t^{(\boldsymbol{\varphi})} &= \int_{\mathbb{R}^{2p}} \left(\boldsymbol{\alpha}_t - \mathbf{F}\boldsymbol{\alpha}_{t-1}\right)\left(\boldsymbol{\alpha}_t - \mathbf{F}\boldsymbol{\alpha}_{t-1}\right)^\top \mathrm{P}_{\boldsymbol{\varphi}}\left(\boldsymbol{\alpha}_{(t-1):t} \middle| \boldsymbol{y}_{1:d}\right) \partial \boldsymbol{\alpha}_{(t-1):t} \qquad (27) \\
&\approx \sum_{i=1}^{N_s} \left(\widehat{\boldsymbol{\alpha}}_t^{(i)} - \mathbf{F}\boldsymbol{\alpha}_{t-1}^{(j_{it})}\right)\left(\widehat{\boldsymbol{\alpha}}_t^{(i)} - \mathbf{F}\boldsymbol{\alpha}_{t-1}^{(j_{it})}\right)^\top \widehat{w}_t^{(i)}
\end{aligned}
$$

where $\boldsymbol{\alpha}_{s:t} = \{\boldsymbol{\alpha}_s, \boldsymbol{\alpha}_{s+1}, \dots \boldsymbol{\alpha}_t\}$, we have extended the notation in Algorithm 1 such that superscript j_{it} is the index from forward cloud at time $t-1$ matching with i'th smoothed particle at time t, and the subscript in P denotes that it is the probability given the parameter $\boldsymbol{\varphi}$. The update of $\boldsymbol{a}_0$ and $\mathbf{Q}$ given the summary statistics is

$$
\boldsymbol{a}_0 = \boldsymbol{t}_0^{(\boldsymbol{\varphi})} \qquad \mathbf{Q} = \frac{1}{d-1} \sum_{t=2}^{d} \mathbf{R}^{+} \boldsymbol{T}_t^{(\boldsymbol{\varphi})} \mathbf{R}^{+\top} \qquad (28)
$$

We then repeat with the new $\boldsymbol{a}_0$ and $\mathbf{Q}$ for a given number of iterations or till a convergence criteria is satisfied. See Kantas et al. (2015), Del Moral et al. (2010) and Schön et al. (2011) for further details on parameter estimation with particle filters.

I do not think we can estimate $\boldsymbol{a}_0$ consistently so we likely just want to fix it at some value...

3.1 Vector auto-regression models

We start by defining the following matrices to cover estimation in general vector auto-regression models for the latent space variable

$$\begin{aligned}
\mathbf{N} &= \left(\widehat{\boldsymbol{\alpha}}_2^{(1)}, \widehat{\boldsymbol{\alpha}}_2^{(2)}, \dots, \widehat{\boldsymbol{\alpha}}_2^{(N_s)}, \widehat{\boldsymbol{\alpha}}_3^{(1)}, \dots, \widehat{\boldsymbol{\alpha}}_d^{(N_s)}\right)^\top \mathbf{R}^{+\top} \\
\mathbf{M} &= \left(\boldsymbol{\alpha}_1^{(j_{12})}, \boldsymbol{\alpha}_1^{(j_{22})}, \dots, \boldsymbol{\alpha}_1^{(j_{N_s 2})}, \boldsymbol{\alpha}_2^{(j_{13})}, \dots, \boldsymbol{\alpha}_{d-1}^{(j_{N_s d})}\right)^\top \\
\mathbf{W} &= \operatorname{diag}\left(\widehat{w}_2^{(1)}, \dots, \widehat{w}_2^{(N_s)}, \widehat{w}_3^{(1)}, \dots, \widehat{w}_d^{(N_s)}\right)
\end{aligned}$$

where $\operatorname{diag}(\cdot)$ is a diagonal matrix which diagonal elements are the argument. We implicitly let the above depend on the result of the E-step in a given iteration of the EM algorithm to ease the notation. The goal is to estimate $\mathbf{F}$ and $\mathbf{Q}$ in Equation (1). We can then show that the M-step maximizers are

$$\widehat{\mathbf{F}}^\top \mathbf{R}^{+\top} = \left(\mathbf{M}^\top \mathbf{W} \mathbf{M}\right)^{-1} \mathbf{M}^\top \mathbf{W} \mathbf{N} \tag{29}$$

$$\widehat{\mathbf{Q}} = \frac{1}{d-1} \left(\mathbf{N} - \mathbf{R}^+ \widehat{\mathbf{F}} \mathbf{M}\right)^\top \mathbf{W} \left(\mathbf{N} - \mathbf{R}^+ \widehat{\mathbf{F}} \mathbf{M}\right) \tag{30}$$

which is the typical vector auto-regression estimators with weights. Equation (29) and (30) can easily be computed in parallel using QR decompositions as in the `bam` in the `mgcv` package with a low memory footprint (see Wood et al., 2014). This is currently implemented. Though, the gains from a parallel implementation may be small as the computation here have a computation time which is independent of the number of observations. In other words, the other the computation is relatively much less demanding then the other parts.

3.2 Restricted vector auto-regression models

Suppose that we want to restrict some of the parameters of $\mathbf{F}$ and $\mathbf{Q}$. E.g., we can restrict the model to

$$\operatorname{vec}\left(\mathbf{R}^+ \mathbf{F}\right) = \mathbf{G}\boldsymbol{\theta} \qquad \mathbf{Q} = \mathbf{V}\mathbf{C}\mathbf{V}$$

$$\mathbf{V} = \begin{pmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \ddots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & \sigma_r \end{pmatrix} \qquad \mathbf{C} = \begin{pmatrix} 1 & \rho_{21} & \cdots & \rho_{r1} \\ \rho_{21} & 1 & \ddots & \rho_{r2} \\ \vdots & \ddots & \ddots & \vdots \\ \rho_{r1} & \dots & \rho_{r,r-1} & 1 \end{pmatrix}$$

$$\sigma_i = \exp(s_i) \qquad \rho_{ij} = \frac{2}{1 + \exp(-o_{ij})} - 1$$

with

$$\begin{aligned}
(s_1, s_2, \dots, s_r)^\top &= \mathbf{J}\boldsymbol{\psi} \\
(o_{21}, o_{31}, \dots, o_{r1}, o_{32}, \dots, o_{r,r-1})^\top &= \mathbf{K}\boldsymbol{\phi}
\end{aligned}$$

and where vec $(\cdot)$ is the vectorization function which stacks the the columns of a matrix from left to right. E.g.,

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$
$$\text{vec}\,(\mathbf{A}) = (a_{11}, a_{21}, a_{31}, a_{12}, a_{22}, a_{32}, a_{13}, a_{23}, a_{33})^\top$$

$\mathbf{G} \in \mathbb{R}^{rp \times g}$ is a known matrix with $g \leq rp$ and we assume that it has full column rank. Similarly, $\mathbf{J} \in \mathbb{R}^{r \times l}$ with $l \leq r$ and $\mathbf{K} \in \mathbb{R}^{r(r-1)/2 \times k}$ with $k \leq r(r-1)/2$. Both are known and have full column rank. We assume that $\mathbf{G}$ is such that $\mathbf{F}$ is non-singular for some $\boldsymbol{\theta}$. Similarly, we assume that $\mathbf{J}$ and $\mathbf{K}$ are such that $\mathbf{Q}$ is a positive definite matrix for some $\boldsymbol{\psi}$ and $\boldsymbol{\phi}$ pair. $\mathbf{V}$ is a diagonal matrix containing the standard deviations and $\mathbf{C}$ is the correlation matrix.

We cannot jointly maximize $\boldsymbol{\theta}$, $\boldsymbol{\psi}$, and $\boldsymbol{\phi}$ analytically but we can maximize $\boldsymbol{\theta}$ analytically conditional on $\boldsymbol{\psi}$ and $\boldsymbol{\phi}$. Hence, we can employ a Monte Carlo expectation conditional maximization algorithm in which we take two so-called conditional maximization steps (see Meng and Rubin, 1993, on the, non-Monte Carlo, expectation maximization algorithm). We need some more notation before we show the two conditional maximization steps. Let $H^{(r,p)}$ be the (r, p) commutation matrix so

$$H^{(r,p)}\text{vec}\left(\mathbf{R}^+\mathbf{F}\right) = \text{vec}\left(\left(\mathbf{R}^+\mathbf{F}\right)^\top\right) = \text{vec}\left(\mathbf{F}^\top\mathbf{R}^{+\top}\right)$$

and let superscript (i) denote the M-step estimates from the i'th iteration of the EM algorithm. Then the first conditional maximization step is

$$\boldsymbol{\theta}^{(i+1)} = \tilde{\mathbf{G}}^+\left(\mathbf{Q}^{(i)} \otimes \left(\mathbf{M}^\top\mathbf{W}\mathbf{M}\right)^{-1}\right)\tilde{\mathbf{G}}^{+\top}\tilde{\mathbf{G}}^\top\text{vec}\left(\mathbf{M}^\top\mathbf{W}\mathbf{N}\mathbf{Q}^{-(i)}\right) \tag{31}$$

where $\otimes$ is the Kronecker product and $\mathbf{Q}^{-(i)}$ is the inverse of $\mathbf{Q}^{(i)}$. Equation (31) is easily computed with the QR decomposition we compute for Equation (29). Having obtained the new $\boldsymbol{\theta}^{(i+1)}$ then we update the variable elements of $\mathbf{F}$ (those columns "picked out" by $\mathbf{R}$ in $\mathbf{R}^+\mathbf{F}$) and denote the new estimate $\widehat{\mathbf{F}}^{(i+1)}$. The second conditional maximization of $\boldsymbol{\psi}$ and $\boldsymbol{\phi}$ is

$$\mathbf{Z} = \left(\mathbf{N} - \mathbf{R}^+\widehat{\mathbf{F}}^{(i+1)}\mathbf{M}\right)^\top\mathbf{W}\left(\mathbf{N} - \mathbf{R}^+\widehat{\mathbf{F}}^{(i+1)}\mathbf{M}\right)$$
$$\boldsymbol{\psi}^{(i+1)}, \boldsymbol{\phi}^{(i+1)} = \underset{\boldsymbol{\psi},\boldsymbol{\phi}}{\arg\max} -(d-1)\log|\mathbf{Q}(\boldsymbol{\psi},\boldsymbol{\phi})| - \text{tr}\left(\mathbf{Q}(\boldsymbol{\psi},\boldsymbol{\phi})^{-1}\mathbf{Z}\right)$$

which can be done numerically. We have made $\mathbf{Q}$'s depends on $\boldsymbol{\psi}$ and $\boldsymbol{\phi}$ explicit to emphasize which factors are affected. $\mathbf{C}$ will not be a valid covariance for all $\boldsymbol{\phi} \in \mathbb{R}^k$ for some choices of $\mathbf{K}$. The invalid values are ruled out doing the numerical optimization. This completes the two conditional maximization steps. The next E-step is then performed using $\boldsymbol{\theta}^{(i+1)}$, $\boldsymbol{\psi}^{(i+1)}$, $\boldsymbol{\phi}^{(i+1)}$. Meng and Rubin (1993, see the discussion) comments that it may be beneficial to perform an E-step between each conditional maximization step when the E-step is relatively cheap. This is not the case here since all the above computation are independent of $n_{\max}$.

3.3 Estimating fixed effect coefficients

Next, we turn to estimating the fixed effects, $\boldsymbol{\omega}$, in Equation (1). Since each observation y_{it} is from an exponential family then it is easy to show that the M-step estimator amounts to generalized linear model with N_s observations for each y_{it} which differ only by an offset term and a weight. The offset term comes from the $\boldsymbol{x}_{it}^\top \mathbf{R}^+ \widehat{\boldsymbol{\alpha}}_j^{(t)}$ term in Equation (1) for each of the $j = 1, \dots, N_s$ smoothed particles. The corresponding offset terms are the smoothed weights, $\widehat{w}_j^{(t)}$. The problem can be solved in parallel using QR decompositions as in Section 3.1. This is what is done in the current implementation.

Currently, I only take one iteration of the iteratively re-weighted least squares. I gather I have to repeat till convergence though... This is however not nice computationally and the difference in the estimate from one M-step iteration to the next is very minor when you only take one iteratively re-weighted least square iteration...

3.4 Observed information matrix

Computing the observed information matrix requires an application of the missing information principle (Louis, 1982). However we cannot evaluate the quantities we need with the output from Algorithm 1 since we only have discrete approximation of the smoothed distribution of triplet of particles, $\boldsymbol{\alpha}_{t-1:t+1}$, but need an approximation for the entire path, $\boldsymbol{\alpha}_{1:d}$. One solution is to use so-called smoothing functionals. This is covered in e.g., (Cappé et al., 2005, section 8.3 and chapter 11) and Poyiadjis et al. (2011). The method in Cappé et al. (2005) only requires the forward particle filter output. It is though not implemented.

4 Other filter and smoother options

The $\mathcal{O}\left(N^2\right)$ two-filter smoother in Fearnhead et al. (2010) is going to be computationally expensive as an approximation is going to be needed for Equation (8) in the article. For instance, only the Taylor expansion approximation around a single point to approximate g would be feasible. The non-auxiliary version in Briers et al. (2009) is more feasible as it only requires evaluation of f in the smoothing part of the generalized two-filter smoother (see Equation (46) in the paper). Similar conclusions applies to the forward smoother in Del Moral et al. (2010) and the backward smoother as presented in Kantas et al. (2015). Both have a $\mathcal{O}\left(N^2\right)$ computational cost.

Despite the $\mathcal{O}\left(N^2\right)$ cost of the method in Briers et al. (2009) and Del Moral et al. (2010) they are still worthy candidates as the computational cost is independent of the number of observations, n. Further, the computational cost can be reduced to $\mathcal{O}\left(N \log(N)\right)$ with the approximations in Klaas et al. (2006).

The method in Malik and Pitt (2011, see particularly section 6.2 on page 203) can be used to do continuous likelihood evaluation. I am not sure how well these method scale with higher state dimension, p.

Kantas et al. (2015) show empirically that it may be worth just using a forward filter. However, the example is with a univariate outcome ($n = 1$ – not

to be confused with the number of periods d). The cost here of the forward filter is at least $\mathcal{O}(dNn_{\max}p)$. Every new particle yields an $\mathcal{O}(dn_{\max}p)$ cost which is expensive due to the large number of outcomes, n. Thus, the considerations are different and a $\mathcal{O}\left(dNn_{\max}p + N^2\right)$ method will not make a big difference unless N is large. Another alternative is to add noise to the parameters $\boldsymbol{\theta}$ at each time t and use the methods in Andrieu and Doucet (2002) or similar ideas to perform online estimation.

5 Briers et al. (2009)

The $\mathcal{O}\left(N^2\right)$ smother from Briers et al. (2009) is also implemented as it is feasible for a moderate number of particles (though, we can use the approximations in Kantas et al. (2015) to reduce the computational complexity). It is shown in Algorithm 4. The weights in Equation (35) comes from the generalized two-filter formula. To cover this filter, first define

$$\begin{aligned}
\widetilde{\mathrm{P}}\left(\boldsymbol{\alpha}_d \middle| \boldsymbol{y}_d\right) &= \frac{\gamma_d(\boldsymbol{\alpha}_d) g_d\left(\boldsymbol{y}_d \middle| \boldsymbol{\alpha}_d\right)}{\widetilde{\mathrm{P}}\left(\boldsymbol{y}_d\right)} \\
\widetilde{\mathrm{P}}\left(\boldsymbol{y}_d\right) &= \int \gamma_d(\boldsymbol{\alpha}_d) g_d\left(\boldsymbol{y}_d \middle| \boldsymbol{\alpha}_d\right) d\boldsymbol{\alpha}_d \\
\widetilde{\mathrm{P}}\left(\boldsymbol{\alpha}_{t:d} \middle| \boldsymbol{y}_{t:d}\right) &= \frac{\gamma_t(\boldsymbol{\alpha}_t) g_t\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right) \prod_{k=t+1}^{d} f\left(\boldsymbol{\alpha}_k \middle| \boldsymbol{\alpha}_{k-1}\right) g_k\left(\boldsymbol{y}_k \middle| \boldsymbol{\alpha}_k\right)}{\widetilde{\mathrm{P}}\left(\boldsymbol{y}_{t:d}\right)} \\
\widetilde{\mathrm{P}}\left(\boldsymbol{y}_{t:d}\right) &= \int \cdots \int \gamma_t(\boldsymbol{\alpha}_t) g_t\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right) \prod_{k=t+1}^{d} f\left(\boldsymbol{\alpha}_k \middle| \boldsymbol{\alpha}_{k-1}\right) g_k\left(\boldsymbol{y}_k \middle| \boldsymbol{\alpha}_k\right) d\boldsymbol{\alpha}_{t:d}
\end{aligned}$$

We can then find a backward recursion

$$\begin{aligned}
\widetilde{\mathrm{P}}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{(t+1):d}\right) &= \int \widetilde{\mathrm{P}}\left(\boldsymbol{\alpha}_{t+1} \middle| \boldsymbol{y}_{(t+1):d}\right) \frac{f\left(\boldsymbol{\alpha}_{t+1} \middle| \boldsymbol{\alpha}_t\right) \gamma_t(\boldsymbol{\alpha}_t)}{\gamma_{t+1}(\boldsymbol{\alpha}_{t+1})} d\boldsymbol{\alpha}_{t+1} \\
\widetilde{\mathrm{P}}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{t:d}\right) &= \frac{g_t\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right) \widetilde{\mathrm{P}}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{(t+1):d}\right)}{\int g_t\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right) \widetilde{\mathrm{P}}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{(t+1):d}\right) d\boldsymbol{\alpha}_t} \\
&\propto g_t\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right) \int \widetilde{\mathrm{P}}\left(\boldsymbol{\alpha}_{t+1} \middle| \boldsymbol{y}_{(t+1):d}\right) \frac{f\left(\boldsymbol{\alpha}_{t+1} \middle| \boldsymbol{\alpha}_t\right) \gamma_t(\boldsymbol{\alpha}_t)}{\gamma_{t+1}(\boldsymbol{\alpha}_{t+1})} d\boldsymbol{\alpha}_{t+1} \\
&\propto\!\!\propto \sum_{i=1}^{N} \widetilde{w}_t^{(i)} \delta\left(\boldsymbol{\alpha}_t - \widetilde{\boldsymbol{\alpha}}_t^{(i)}\right)
\end{aligned}$$

which gives us the backward particle filter in Algorithm (3) and $\delta(\cdot)$ is the Dirac Delta function. The final result we need is

$$\mathrm{P}\left(\boldsymbol{y}_{t:d} \middle| \boldsymbol{\alpha}_t\right) = \widetilde{\mathrm{P}}\left(\boldsymbol{y}_{t:d}\right) \frac{\widetilde{\mathrm{P}}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{t:d}\right)}{\gamma_t(\boldsymbol{\alpha}_t)}$$

Then we can generalize the two-filter formula in Kitagawa (1994) as follows

$$\begin{aligned}
\mathrm{P}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{1:d}\right) &= \frac{\mathrm{P}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{1:t-1}\right) \mathrm{P}\left(\boldsymbol{y}_{t:d} \middle| \boldsymbol{\alpha}_t\right)}{\mathrm{P}\left(\boldsymbol{y}_{t:d} \middle| \boldsymbol{y}_{1:t-1}\right)} \\
&\propto \mathrm{P}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{1:t-1}\right) \mathrm{P}\left(\boldsymbol{y}_{t:d} \middle| \boldsymbol{\alpha}_t\right) \\
&= \mathrm{P}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{1:t-1}\right) \widetilde{\mathrm{P}}\left(\boldsymbol{y}_{t:d}\right) \frac{\widetilde{\mathrm{P}}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{t:d}\right)}{\gamma_t(\boldsymbol{\alpha}_t)} \\
&\propto \mathrm{P}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{1:t-1}\right) \frac{\widetilde{\mathrm{P}}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{t:d}\right)}{\gamma_t(\boldsymbol{\alpha}_t)} \\
&= \widetilde{\mathrm{P}}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{t:d}\right) \frac{\left[\int \mathrm{P}\left(\boldsymbol{\alpha}_{t-1} \middle| \boldsymbol{y}_{1:t-1}\right) f\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}\right) \partial \boldsymbol{\alpha}_{t-1}\right]}{\gamma_t(\boldsymbol{\alpha}_t)} \\
&\propto\!\!\!\!\propto \sum_{i=1}^{N} \widetilde{w}_t^{(i)} \delta\left(\boldsymbol{\alpha}_t - \widetilde{\boldsymbol{\alpha}}_t^{(i)}\right) \frac{\left[\sum_{j=1}^{N} w_{t-1}^{(j)} f\left(\widetilde{\boldsymbol{\alpha}}_t^{(i)} \middle| \boldsymbol{\alpha}_{t-1}^{(j)}\right)\right]}{\gamma_t\left(\widetilde{\boldsymbol{\alpha}}_t^{(i)}\right)}
\end{aligned} \tag{32}$$

Similar arguments leads to

$$\begin{aligned}
&\mathrm{P}\left(\boldsymbol{\alpha}_{t-1:t} \middle| \boldsymbol{y}_{1:d}\right) \\
&\quad\propto \mathrm{P}\left(\boldsymbol{\alpha}_{t-1:t} \middle| \boldsymbol{y}_{1:t-1}\right) \mathrm{P}\left(\boldsymbol{y}_{t:d} \middle| \boldsymbol{\alpha}_{t-1:t}\right) \\
&\quad= f\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}\right) \mathrm{P}\left(\boldsymbol{\alpha}_{t-1} \middle| \boldsymbol{y}_{1:t-1}\right) \mathrm{P}\left(\boldsymbol{y}_{t:d} \middle| \boldsymbol{\alpha}_t\right) \\
&\quad\propto f\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}\right) \mathrm{P}\left(\boldsymbol{\alpha}_{t-1} \middle| \boldsymbol{y}_{1:t-1}\right) \frac{\widetilde{\mathrm{P}}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{t:d}\right)}{\gamma_t(\boldsymbol{\alpha}_t)} \\
&\quad\propto\!\!\!\!\propto \sum_{i=1}^{N} \sum_{j=1}^{N} \widetilde{w}_t^{(i)} \delta\left(\boldsymbol{\alpha}_t - \widetilde{\boldsymbol{\alpha}}_t^{(i)}\right) \frac{\left[\sum_{k=1}^{N} w_{t-1}^{(k)} f\left(\widetilde{\boldsymbol{\alpha}}_t^{(i)} \middle| \boldsymbol{\alpha}_{t-1}^{(k)}\right)\right]}{\gamma_t\left(\widetilde{\boldsymbol{\alpha}}_t^{(i)}\right)} \frac{w_{t-1}^{(j)} \delta\left(\boldsymbol{\alpha}_{t-1} - \boldsymbol{\alpha}_{t-1}^{(j)}\right) f\left(\widetilde{\boldsymbol{\alpha}}_t^{(i)} \middle| \boldsymbol{\alpha}_{t-1}^{(j)}\right)}{\left[\sum_{k=1}^{N} w_{t-1}^{(k)} f\left(\widetilde{\boldsymbol{\alpha}}_t^{(i)} \middle| \boldsymbol{\alpha}_{t-1}^{(k)}\right)\right]} \\
&\quad= \sum_{i=1}^{N} \sum_{j=1}^{N} \widehat{w}_t^{(i,j)} \delta\left(\boldsymbol{\alpha}_t - \widetilde{\boldsymbol{\alpha}}_t^{(i)}\right) \delta\left(\boldsymbol{\alpha}_{t-1} - \boldsymbol{\alpha}_{t-1}^{(j)}\right)
\end{aligned} \tag{33}$$

where

$$\widehat{w}_t^{(i,j)} = \widehat{w}_t^{(i)} \frac{w_{t-1}^{(j)} f\left(\widetilde{\boldsymbol{\alpha}}_t^{(i)} \middle| \boldsymbol{\alpha}_{t-1}^{(j)}\right)}{\left[\sum_{j=1}^{N} w_{t-1}^{(j)} f\left(\widetilde{\boldsymbol{\alpha}}_t^{(i)} \middle| \boldsymbol{\alpha}_{t-1}^{(j)}\right)\right]} \tag{34}$$

The above is what we need for the EM-algorithm.

Algorithm 4 $\mathcal{O}\left(N^2\right)$ generalized two-filter smoother using the method in Briers et al. (2009).

Input:

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

1: **procedure** FILTER FORWARD

2: Run a forward particle filter to get a particle clouds $\left\{\boldsymbol{\alpha}_t^{(j)}, w_t^{(j)}, \beta_{t+1}^{(j)}\right\}_{j=1,\ldots,N}$ 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 $\left\{\widetilde{\boldsymbol{\alpha}}_t^{(k)}, \widetilde{w}_t^{(k)}, \widetilde{\beta}_{t-1}^{(k)}\right\}_{k=1,\ldots,N}$ approximating $\mathrm{P}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{t:d}\right)$ for $t = d+1, d, d-1, \ldots, 1$. See Algorithm 3.

5: **procedure** SMOOTH (COMBINE)

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

7: Assign each backward filter particle a smoothing weight given by

$$\widehat{w}_t^{(i)} \propto \widetilde{w}_t^{(i)} \frac{\left[\sum_{j=1}^N w_{t-1}^{(j)} f\left(\widetilde{\boldsymbol{\alpha}}_t^{(i)} \middle| \boldsymbol{\alpha}_{t-1}^{(j)}\right)\right]}{\gamma_t\left(\widetilde{\boldsymbol{\alpha}}_t^{(i)}\right)} \tag{35}$$

We can now cover the generalized two-filter smother from Fearnhead et al. (2010). Similar to Equation (32) we find that

$$\begin{aligned}
\mathrm{P}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{1:d}\right) &\propto \mathrm{P}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{1:t-1}\right) \mathrm{P}\left(\boldsymbol{y}_{t:d} \middle| \boldsymbol{\alpha}_t\right) \\
&= \mathrm{P}\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{y}_{1:t-1}\right) g_t\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right) \mathrm{P}\left(\boldsymbol{y}_{t+1:d} \middle| \boldsymbol{\alpha}_t\right) \\
&= \int f\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}\right) \mathrm{P}\left(\boldsymbol{\alpha}_{t-1} \middle| \boldsymbol{y}_{1:t-1}\right) d\boldsymbol{\alpha}_{t-1} g_t\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right) \int f\left(\boldsymbol{\alpha}_{t+1} \middle| \boldsymbol{\alpha}_t\right) \mathrm{P}\left(\boldsymbol{y}_{t+1:d} \middle| \boldsymbol{\alpha}_{t+1}\right) d\boldsymbol{\alpha}_{t+1} \\
&\propto \int f\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}\right) \mathrm{P}\left(\boldsymbol{\alpha}_{t-1} \middle| \boldsymbol{y}_{1:t-1}\right) d\boldsymbol{\alpha}_{t-1} g_t\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right) \int f\left(\boldsymbol{\alpha}_{t+1} \middle| \boldsymbol{\alpha}_t\right) \frac{\widetilde{\mathrm{P}}\left(\boldsymbol{\alpha}_{t+1} \middle| \boldsymbol{y}_{t+1:d}\right)}{\gamma_{t+1}(\boldsymbol{\alpha}_{t+1})} d\boldsymbol{\alpha}_{t+1} \\
&\propto\!\!\propto \sum_{i=1}^N \sum_{j=1}^N f\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}^{(j)}\right) w_{t-1}^{(j)} g_t\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right) f\left(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(i)} \middle| \boldsymbol{\alpha}_t\right) \frac{\widetilde{w}_{t+1}^{(i)}}{\gamma_{t+1}(\widetilde{\boldsymbol{\alpha}}_{t+1}^{(i)})}
\end{aligned}$$

Thus, we can sample $\boldsymbol{\alpha}_t$ from a proposal distribution given the time $t-1$ forward filter particle, $\boldsymbol{\alpha}_{t-1}^{(j)}$, and time $t+1$ backward filter particle, $\widetilde{\boldsymbol{\alpha}}_{t+1}^{(i)}$, for all N^2 particle pairs. Alternatively, we can sample the $t-1$ and $t+1$ particles

independently which yields Algorithm 1. Further, we can find that

$$\begin{aligned}
\mathrm{P}\left(\boldsymbol{\alpha}_{t-1:t} \middle| \boldsymbol{y}_{1:d}\right) &= \mathrm{P}\left(\boldsymbol{\alpha}_{t-1:t} \middle| \boldsymbol{y}_{1:t-1}\right) g_t\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_{t-1:t}\right) \mathrm{P}\left(\boldsymbol{y}_{t+1:d} \middle| \boldsymbol{\alpha}_{t-1:t}\right) \\
&\propto f\left(\boldsymbol{\alpha}_t \middle| \boldsymbol{\alpha}_{t-1}\right) \mathrm{P}\left(\boldsymbol{\alpha}_{t-1} \middle| \boldsymbol{y}_{1:t-1}\right) g_t\left(\boldsymbol{y}_t \middle| \boldsymbol{\alpha}_t\right) \int f\left(\boldsymbol{\alpha}_{t+1} \middle| \boldsymbol{\alpha}_t\right) \frac{\widetilde{\mathrm{P}}\left(\boldsymbol{\alpha}_{t+1} \middle| \boldsymbol{y}_{t+1:d}\right)}{\gamma_{t+1}(\boldsymbol{\alpha}_{t+1})} d\boldsymbol{\alpha}_{t+1} \\
&\overset{\propto}{\sim} \sum_{i=1}^{N_s} \delta\left(\boldsymbol{\alpha}_t - \widehat{\boldsymbol{\alpha}}_t^{(i)}, \boldsymbol{\alpha}_t - \boldsymbol{\alpha}_{t-1}^{(j_i)}\right) f\left(\widehat{\boldsymbol{\alpha}}_t^{(i)} \middle| \boldsymbol{\alpha}_{t-1}^{(j_i)}\right) w_{t-1}^{(j_i)} g_t\left(\boldsymbol{y}_t \middle| \widehat{\boldsymbol{\alpha}}_t^{(i)}\right) \int f\left(\boldsymbol{\alpha}_{t+1} \middle| \widehat{\boldsymbol{\alpha}}_t^{(i)}\right) \frac{\widetilde{\mathrm{P}}\left(\boldsymbol{\alpha}_{t+1} \middle| \boldsymbol{y}_{t+1:d}\right)}{\gamma_{t+1}(\boldsymbol{\alpha}_{t+1})} d\boldsymbol{\alpha}_{t+1} \\
&\overset{\propto}{\sim} \sum_{i=1}^{N_s} \widehat{w}_t^{(i)} \delta\left(\boldsymbol{\alpha}_t - \widehat{\boldsymbol{\alpha}}_t^{(i)}, \boldsymbol{\alpha}_t - \boldsymbol{\alpha}_{t-1}^{(j_i)}\right)
\end{aligned}$$

where superscripts j_i are used as in Algorithm 1 implicitly dependent on t.

6 Implementation

The `PF_EM` method in the `dynamichazard` package contains an implementation of the above described method. You specify the number of particles by the `N_first`, `N_fw_n_bw` and `N_smooth` argument for respectively the N_f, N and N_s in the Algorithm 1-3. We may want more particle in the smoothing step, $N_s > N$, as pointed out in the discussion in Fearnhead et al. (2010, page 460 and 461). Further, selecting $N_f > N$ may be preferable to ensure coverage of the state space at time 0 and $d + 1$.

We do not need to sample the time 0 and $d + 1$ particles. Instead we can make a special proposal distribution for time 1 and time d. This is not implemented though...

The `method` argument specify how the filters are set up. The argument can take the following values

- `"bootstrap_filter"` for a bootstrap filter.
- `"PF_normal_approx_w_cloud_mean"` and `"AUX_normal_approx_w_cloud_mean"` for the Taylor approximation of the conditional density of $\boldsymbol{y}_t$ made around the weighted mean of the previous cloud. The `PF` and `AUX` prefix specifies whether or not the auxiliary version should be used.
- `"PF_normal_approx_w_particles"` and `"AUX_normal_approx_w_particles"` for the Taylor approximation of the conditional density of $\boldsymbol{y}_t$ made around the parent (or/and child) particle. The `PF_` and `AUX_` prefix specifies whether or not the auxiliary version should be used.

The smoother is selected with the `smoother` argument. `"Fearnhead_O_N"` gives the smoother in Algorithm 1 and `"Brier_O_N_square"` gives the smoother in Algorithm 4.

The *Systematic Resampling* (Kitagawa, 1996) is used in all re-sampling steps. See Douc and Cappé (2005) for a comparison of re-sampling methods. The rest of the arguments to `PF_EM` are similar to those of the `ddhazard` function.

6.1 Linear maps

The methods describe above involves many linear maps. These are implemented with `C++` abstract classes with specialized `map` member functions for particular problem to decrease the computation. An alternative would have been to use a sparse matrix implementation. As an example we have a mapping matrix $\mathbf{A}$ which `C++` abstract member on the main data object used in the package called `xyz`. E.g., this could $\mathbf{F}$ with name `state_trans`. Then the following operations are implemented

- `map()`: Returns $\mathbf{A}$.
- `map(const arma::vec &x, bool tranpose)`: Returns $\mathbf{A}^\top \boldsymbol{x}$ if `tranpose == true` and $\mathbf{A}\boldsymbol{x}$ otherwise.
- `map(const arma::mat &X, side s, bool tranpose)`: Let $\mathbf{B} = \mathbf{A}^\top$ if `tranpose == true` and otherwise $\mathbf{B} = \mathbf{A}$. Then the result is $\mathbf{B}\mathbf{X}\mathbf{B}^\top$ if `s == both`, $\mathbf{B}\mathbf{X}$ if `s == left`, and $\mathbf{X}\mathbf{B}^\top$ if `s == right`.

These are implemented for

`C++` member name	Matrix $\mathbf{A}$
`err_state`	$\mathbf{R}$
`err_state_inv`	$\mathbf{R}^+ = \mathbf{R}^\top$
`state_trans`	$\mathbf{F}$
`state_trans_err`	$\mathbf{R}^+\mathbf{F} = \mathbf{R}^\top\mathbf{F}$
`state_trans_inv`	$\mathbf{F}^{-1}$

Further, we will need function to compute terms

$$\overleftarrow{\mathbf{P}}_t\mathbf{F}^\top\overleftarrow{\mathbf{P}}_{t+1}^{-1}\boldsymbol{\alpha}_{t+1} + \overleftarrow{\mathbf{S}}_t\overleftarrow{\mathbf{P}}_t^{-1}\overleftarrow{\boldsymbol{m}}_t$$

for an arbitrary $\boldsymbol{\alpha}_{t+1}$, $\overleftarrow{\mathbf{S}}_t = \overleftarrow{\mathbf{P}}_t\mathbf{F}^\top\overleftarrow{\mathbf{P}}_{t+1}^{-1}\mathbf{R}\mathbf{Q}\mathbf{R}^\top\left(\mathbf{F}^\top\right)^{-1}$, $\overleftarrow{\mathbf{P}}_t^{-1}\overleftarrow{\boldsymbol{m}}_t$, and $\overleftarrow{\mathbf{P}}_t^{-1}$ for Equation (12) and (15). This is done with the methods `bw_mean(signed int, const arma::vec&)`, `bw_covar(signed int)`, `uncond_mean_term(signed int)`, and `uncond_covar_inv(signed int)`. The terms and factors that can will computed once using Equation (12) are computed and stored.

6.2 Proposal distribution

The proposal distributions in Equation (8), (15) and (17) can be done as follows. One input is an extra information matrix term which we denote by $\mathbf{B}$. This is $\mathbf{Q}^{-1}$ in Equation (8), $\mathbf{P}_t^{-1} + \mathbf{F}^\top\mathbf{R}^{+\top}\mathbf{Q}^{-1}\mathbf{R}^+\mathbf{F}$ in Equation (15), and $\mathbf{R}^{+\top}\mathbf{Q}^{-1}\mathbf{R}^+ + \mathbf{F}^\top\mathbf{R}^{+\top}\mathbf{Q}^{-1}\mathbf{R}^+\mathbf{F}$ in Equation (17). The other input is an extra mean term which we denote by $\boldsymbol{c}$. This is $\mathbf{Q}^{-1}\mathbf{R}^+\bar{\boldsymbol{\alpha}}_{t-1}$ in Equation (8), $\mathbf{P}_t^{-1}\boldsymbol{m}_t + \mathbf{F}^\top\mathbf{R}^{+\top}\mathbf{Q}^{-1}\mathbf{R}^+\bar{\boldsymbol{\alpha}}_{t+1}$ in Equation (15), and $\mathbf{F}^\top\mathbf{R}^{+\top}\mathbf{Q}^{-1}\mathbf{R}^+\bar{\boldsymbol{\alpha}}_{t+1} + \mathbf{R}\mathbf{Q}^{-1}\mathbf{R}^+\bar{\boldsymbol{\alpha}}_{t-1}$ for Equation (17). Then given an initial value $\bar{\boldsymbol{\mu}}_t^{(0)}$ for $k = 1, \dots$

1. Set $\bar{\boldsymbol{\xi}}_t^{(k-1)} = \mathbf{R}^+\bar{\boldsymbol{\mu}}_t^{(k-1)}$.

2. Compute

$$\begin{aligned}\boldsymbol{\Sigma}_t^{(k)-} &= \mathbf{R}\mathbf{X}_t^\top \left(-\mathbf{G}_t\left(\bar{\boldsymbol{\xi}}_t^{(k-1)}\mathbf{X}\mathbf{R}^\top + \boldsymbol{o}_t\right)\right) + \mathbf{B} \\ \bar{\boldsymbol{\mu}}_t^{(k)} &= \boldsymbol{\Sigma}_t^{(k)}\left(\boldsymbol{c} + \mathbf{R}\mathbf{X}_t^\top\left(-\mathbf{G}_t\left(\bar{\boldsymbol{\xi}}_t^{(k-1)} + \boldsymbol{o}_t\right)\mathbf{X}_t\bar{\boldsymbol{\xi}}_t^{(k-1)} - \boldsymbol{o}_t + \boldsymbol{g}_t\left(\bar{\boldsymbol{\xi}}_t^{(k-1)} + \boldsymbol{o}_t\right)\right)\right)\end{aligned}$$

3. Stop if $\left|\bar{\boldsymbol{\mu}}_t^{(k)} - \bar{\boldsymbol{\mu}}_t^{(k-1)}\right| / \left|\bar{\boldsymbol{\mu}}_t^{(k-1)}\right| < \epsilon$ and return $\boldsymbol{\Sigma}_t^{(k)}$ and $\boldsymbol{t} = \boldsymbol{\Sigma}_t^{(k)}\mathbf{R}\mathbf{X}_t^\top\left(-\mathbf{G}_t\left(\bar{\boldsymbol{\xi}}_t^{(k-1)} + \boldsymbol{o}_t\right)\mathbf{X}_t\bar{\boldsymbol{\xi}}_t^{(k-1)} - \boldsymbol{o}_t + \boldsymbol{g}_t\left(\bar{\boldsymbol{\xi}}_t^{(k-1)} + \boldsymbol{o}_t\right)\right)$. Otherwise set $k \leftarrow k+1$ and repeat 1.

The returned output it was we need to compute the proposal distribution. Equation (8) only the needs the $r \times r$ dimensional output since there is $p - r$ elements that are constant and where the multiplication by $\mathbf{R}$ and $\mathbf{R}^\top$ is not done. Thus, an `is_forward` flag is used to indicate whether the r dimensional output is returned in the function `taylor_normal_approx`.

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 Simon Maskell. Smoothing algorithms for state–space models. *Annals of the Institute of Statistical Mathematics*, 62 (1):61, Jun 2009. ISSN 1572-9052. doi: 10.1007/s10463-009-0236-2. URL `https://doi.org/10.1007/s10463-009-0236-2`.

Olivier Cappé, Eric Moulines, and Tobias Ryden. *Inference in Hidden Markov Models*. Springer-Verlag New York, 2005. ISBN 978-0-387-40264-2, 978-1-4419-2319-6.

Pierre Del Moral, Arnaud Doucet, and Sumeetpal Singh. Forward smoothing using sequential monte carlo. *arXiv preprint arXiv:1012.5390*, 2010.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

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.

Arnaud Doucet, Simon Godsill, and Christophe Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Statistics and Computing*, 10 (3):197–208, Jul 2000. ISSN 1573-1375. doi: 10.1023/A:1008935410038. URL `https://doi.org/10.1023/A:1008935410038`.

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. The two-filter formula for smoothing and an implementation of the gaussian-sum smoother. *Annals of the Institute of Statistical Mathematics*, 46(4):605–623, Dec 1994. ISSN 1572-9052. doi: 10.1007/BF00773470. URL `https://doi.org/10.1007/BF00773470`.

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.

Mike Klaas, Mark Briers, Nando De Freitas, Arnaud Doucet, Simon Maskell, and Dustin Lang. Fast particle smoothing: If i had a million particles. In *Proceedings of the 23rd international conference on Machine learning*, pages 481–488. ACM, 2006.

Thomas A. Louis. Finding the observed information matrix when using the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 44(2):226–233, 1982. ISSN 00359246. URL `http://www.jstor.org/stable/2345828`.

Sheheryar Malik and Michael K Pitt. Particle filters for continuous likelihood evaluation and maximisation. *Journal of Econometrics*, 165(2):190–209, 2011.

Xiao-Li Meng and Donald B. Rubin. Maximum likelihood estimation via the ecm algorithm: A general framework. *Biometrika*, 80(2):267–278, 1993. ISSN 00063444. URL `http://www.jstor.org/stable/2337198`.

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

George Poyiadjis, Arnaud Doucet, and Sumeetpal S. Singh. Particle approximations of the score and observed information matrix in state space models with application to parameter estimation. *Biometrika*, 98(1):65–80, 2011. ISSN 00063444. URL `http://www.jstor.org/stable/29777165`.

Thomas B Schön, Adrian Wills, and Brett Ninness. System identification of nonlinear state-space models. *Automatica*, 47(1):39–49, 2011.

Simon Wood, Yannig Goude, and Simon Shaw. Generalized additive models for large data sets. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 64(1):139–155, 2014. doi: 10.1111/rssc.12068. URL `https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/rssc.12068`.