State Space Models

Hans-Peter Höllwirth

Master Project Report
Barcelona Graduate School of Economics
Master Degree in Data Science
2017

Contents

1 Introduction

2 State Space Models

State space models consist of two set of data:

1. A series of **latent states** $\{x_t\}_{t=1}^T$ (with $x_t \in \mathcal{X}$) that forms a Markov chain. Thus, x_t is independent of all past states but x_{t-1}.
2. A set of **observations** $\{y_t\}_{t=1}^T$ (with $y_t \in \mathcal{Y}$) where any observation y_t only depends on its latent state x_t. In other words, an observation is a noisy representation of its underlying state.

Note that if the state space $\mathcal{X}$ and the observation state $\mathcal{Y}$ are both discrete sets, the state space model reduces to a Hidden Markov Model.

The relation between the latent states and observations can be summarized by two probability distributions:

1. The **transition density** from the current state to a new state $p(x_{t+1}|x_t, \boldsymbol{\theta})$.
2. The **measurement density** for an observation given the latent state $p(y_t|x_t, \boldsymbol{\theta})$.

Here, $\boldsymbol{\theta} \in \Theta$ denotes the parameter vector of a state space model.

2.1 Local Level Model

Arguably, the simplest state space model is the (univariate) local level model. It has the following form:

$$\begin{aligned} \text{observation:} \quad & y_t = x_t + \epsilon_t, \quad \epsilon_t \sim N(0, \sigma_\epsilon^2) \\ \text{state:} \quad & x_{t+1} = x_t + \eta_t, \quad \eta_t \sim N(0, \sigma_\eta^2) \end{aligned}$$

with some initial state $x_1 \sim N(\mu_1, \Sigma_1)$. All noise elements, i.e. all ϵ_t's and η_t's, are both mutually independent and independent from the initial state x_1. Assuming that we know μ_1 and Σ_1, the model is fully specified by the following vector of parameters:

$$\boldsymbol{\theta} = [\sigma_\eta^2, \sigma_\epsilon^2]^T$$

Note that in the case of noise-free observations (i.e. $\sigma_\eta^2 = 0$), the model reduces to a pure random-walk. Likewise, if $\sigma_\epsilon^2 = 0$, the observations $\{y_t\}_{t=1}^T$ are a white noise representation of a some value x_1.
The transition and measurement density of the local level model are simple to deduce:

$$\begin{aligned} p(x_{t+1}|x_t, \boldsymbol{\theta}) &\quad \sim N(x_t, \sigma_\epsilon^2) \\ p(y_t|x_t, \boldsymbol{\theta}) &\quad \sim N(x_t, \sigma_\eta^2) \end{aligned}$$

2.2 Latent State Inference

Often, the main objective in state space models is to infer the latent state from observations. Let $\mathcal{I}_t$ denote the set of observed values up to time t:

$$\mathcal{I}_t = \{y_1, y_2, \ldots, y_t\}$$

Then information about the latent state x_t can be summarized by the following two probability distributions:

1. The **prediction density**, $p(x_t|\mathcal{I}_{t-1}, \boldsymbol{\theta})$, gives the probability of x_t given past observations $\mathcal{I}_{t-1}$.
2. The **filtering density**, $p(x_t|\mathcal{I}_t, \boldsymbol{\theta})$, gives the probability of x_t given the current and past observations $\mathcal{I}_t$.

The prediction and filtering densities are recursively related. Given the filtering density for state x_{t-1}, the prediction density for state x_t is

$$p(x_t|\mathcal{I}_{t-1}, \boldsymbol{\theta}) = \int p(x_t|x_{t-1}, \boldsymbol{\theta}) p(x_{t-1}|\mathcal{I}_{t-1}, \boldsymbol{\theta}) dx_{t-1}$$

where the first term in the integral is the transition density from x_{t-1} to x_t, and the second term is the filtering density from before. Likewise, given the prediction density for state x_t, the filtering density for x_t is

$$p(x_t|\mathcal{I}_t, \boldsymbol{\theta}) = \int p(x_t|x_{t-1}, \boldsymbol{\theta}) p(x_{t-1}|\mathcal{I}_{t-1}, \boldsymbol{\theta}) dx_{t-1}$$

2.3 Parameter Inference

Assuming a particular state space model, another common objective is is to infer the model parameters from observations. This is usually achieved via **maximum likelihood estimation**. The log-likelihood of the observations for a given parameter vector $\boldsymbol{\theta}$ is the product of the conditional densities of observations, given all previous observations:

$$\begin{aligned}\log \mathcal{L}(\boldsymbol{\theta}) &= \log \prod_{t=1}^{T} p(y_t|\mathcal{I}_{t-1}, \boldsymbol{\theta}) \\ &= \sum_{t=1}^{T} \log p(y_t|\mathcal{I}_{t-1}, \boldsymbol{\theta}) \\ &= \sum_{t=1}^{T} \log \int p(y_t|x_t, \boldsymbol{\theta}) p(x_t|\mathcal{I}_{t-1}, \boldsymbol{\theta}) dx_t\end{aligned} \tag{2.1}$$

The decomposition of the observation densities into measurement density and prediction density, however, makes the maximization problem analytically intractable for most state space models.

3 Filtering

The objective of filtering is to update our knowledge of the system each time a new observation y_t is brought in. That is, we want to find an estimate of the latent process $x_{0:t}$, given all observations $\mathcal{I}_t = y_{0:t}$ and the model parameters $\boldsymbol{\theta}$:

$$x_{0:t}|\mathcal{I}_t, \boldsymbol{\theta}$$

Note that the joint distribution of the latent process conditioned on the observations can be decomposed in a recursive form:

$$p(x_{0:t}|\mathcal{I}_t, \boldsymbol{\theta}) = \Big[\frac{p(y_t|x_t, \boldsymbol{\theta})p(x_t|x_{t-1}, \boldsymbol{\theta})}{p(y_t|\mathcal{I}_{t-1}, \boldsymbol{\theta})}\Big]p(x_{0:(t-1)}|\mathcal{I}_{t-1}, \boldsymbol{\theta})$$

This form allows for updating our knowledge of the system in an online fashion. This has significant computational advantages: We do not need to keep the whole time series in memory and we can simply update our knowledge once we observe a new y_t. Unfortunately, in many state space models the normalization term

$$p(y_t|\mathcal{I}_{t-1}, \boldsymbol{\theta}) = \int p(y_t|x_t, \boldsymbol{\theta})p(x_t|x_{t-1}, \boldsymbol{\theta})dx_t$$

is analytically intractable. One notable exception are linear Gaussian state space models such as the local level model.

3.1 Kalman Filter

State space models in which both the latent states $\{x_t\}_{t=1}^T$ and the observations $\{y_t\}_{t=1}^T$ have linear dependencies and are normally distributed, the joint distribution and of the latent process $p(x_{0:t}|y_{0:t})$ is also Gaussian (and so are the prediction and filtering density). The inference problem can be analytically solved by using standard results of multivariate Gaussian marginal and conditional distributions.

The *Kalman filter*, however, uses a more efficient way to infer the latent states. The filter recursively computes the Gaussian prediction density $p(x_t|\mathcal{I}_{t-1}, \boldsymbol{\theta}) = N(\mu_{t|t-1}, \Sigma_{t|t-1})$ and filtering density $p(x_t|\mathcal{I}_t, \boldsymbol{\theta}) = N(\mu_{t|t}, \Sigma_{t|t})$ by obtaining their respective mean and covariance. This method has the big advantage that it does not need all observations to be kept in memory and can easily update the system whenever a new observation is made. Consider the multivariate generalization of the univariate local level model from before:

$$\begin{aligned} \text{observation:} \quad & \boldsymbol{y}_t = \boldsymbol{x}_t + \boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim N(\mathbf{0}, \Sigma_\epsilon) \\ \text{state:} \quad & \boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \boldsymbol{\eta}_t, \quad \boldsymbol{\eta}_t \sim N(\mathbf{0}, \Sigma_\eta) \end{aligned}$$

For the case of this model, the mean and (co)variance of the prediction and filtering density are updated as follows:

1. The **prediction step** obtains the *a priori* estimate of $\boldsymbol{x}_t$, given $\mathcal{I}_{t-1}$ (using the *a posteriori* estimate with $t-1$).

$$\begin{aligned} \boldsymbol{\mu}_{t|t-1} &= \boldsymbol{\mu}_{t-1|t-1} \\ \Sigma_{t|t-1} &= \Sigma_{t-1|t-1} + \Sigma_\eta \end{aligned} \tag{3.1}$$

2. The **update step** combines a new observation $\boldsymbol{y}_t$ with the *a priori* estimate to obtain an improved *a posteriori* estimate.

$$\begin{aligned} \boldsymbol{\mu}_{t|t} &= \boldsymbol{\mu}_{t|t-1} + K_t \boldsymbol{v}_t \\ \Sigma_{t|t} &= \Sigma_{t|t-1}(1 - K_t) \end{aligned} \tag{3.2}$$

   where $\boldsymbol{v}_t = \boldsymbol{y}_t - \boldsymbol{\mu}_{t|t-1}$ denotes the difference between prediction and observation and $K_t = \Sigma_{t|t-1}(\Sigma_{t|t-1} + \Sigma_\epsilon)^{-1}$ denotes the *Kalman gain.* It determines how much the new observation affects the updated prediction.

Note that to start the recursion, we need an initial state density with $\boldsymbol{\mu}_1$ and Σ_1.

Algorithm

In the following version of the algorithm we assume $\boldsymbol{\mu}_1$ and Σ_1 to be known. There are various ways to initialize the algorithm when $\boldsymbol{\mu}_1$ and Σ_1 are unknown, however, these methods are beyond the scope of this project.

Algorithm 1 describes the recursive update of the prediction and filtering density. For our purpose, the algorithm returns the filtering densities. The sequence of filtering means $\{\boldsymbol{\mu}_{t|t}\}$ is the best possible estimation for the latent states $\{\boldsymbol{x}_t\}$. The sequence of covariances $\{\Sigma_{t|t}\}$ can be used to obtain confidence intervals for these estimates.

Note that the algorithm needs to invert the matrix $F_t = \Sigma_{t|t-1} + \Sigma_\epsilon$ at each iteration. For large dimensions of the states/observations, this can slow down this version of the Kalman filter significantly.

Example: Local Level Model

Figure 3.1 shows a realization of the univariate local level model with parameters $\sigma_\eta^2 = 1.4$ and $\sigma_\epsilon^2 = 1.0$ (left plot) and the Kalman prediction with a 95% confidence level (right plot).

Algorithm 1 (Local Level) Kalman filter

```
1: procedure KalmanFilter($\mathcal{I}_T, \boldsymbol{\theta}, \boldsymbol{\mu}_1, \Sigma_1$)
2:     $\boldsymbol{\mu}_{1|0} \leftarrow \boldsymbol{\mu}_1$                                    ▷ initialization
3:     $\Sigma_{1|0} \leftarrow \Sigma_1$
4:     for $t$ in $1:T$ do
5:         $\boldsymbol{v}_t = \boldsymbol{y}_t - \boldsymbol{\mu}_{t|t-1}$
6:         $K_t = \Sigma_{t|t-1}(\Sigma_{t|t-1} + \Sigma_\epsilon)^{-1}$
7:         $\boldsymbol{\mu}_{t|t} = \boldsymbol{\mu}_{t|t-1} + K_t\boldsymbol{v}_t$                ▷ update step
8:         $\Sigma_{t|t} = \Sigma_{t|t-1}(1 - K_t)$
9:         $\boldsymbol{\mu}_{t+1|t} \leftarrow \boldsymbol{\mu}_{t|t}$                           ▷ prediction step
10:        $\Sigma_{t+1|t} \leftarrow \Sigma_{t|t} + \Sigma_\eta$
11:    return $\{\boldsymbol{\mu}_{t|t}\}$, $\{\Sigma_{t|t}\}$
```

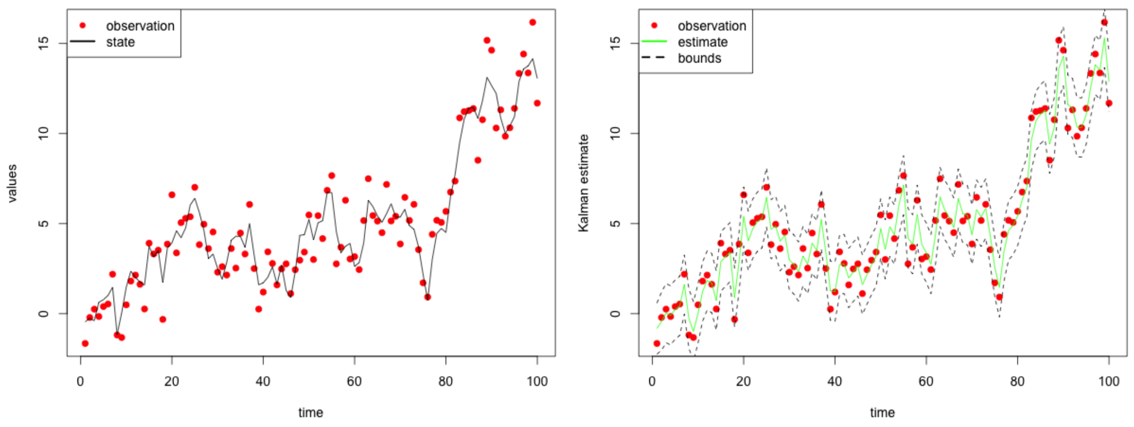

Figure 3.1: Kalman estimate of univariate local level realization

Likelihood evaluation

The log-likelihood function for the linear Gaussian space model has the following (*prediction error decomposition*) form:

$$\log \mathcal{L}(\mathcal{I}_T, \boldsymbol{\theta}) = -\frac{Td}{2}\log(2\pi) - \frac{1}{2}\sum_{t=1}^{T}(\log|F_t| + v_t^T F_t^{-1} v_t)$$

Its elements F_t and v_t are routinely calculated by the Kalman filter and so the log-likelihood can be directly evaluated in the Kalman function. Note that F_t might not be singular for all $t = 1, \ldots, T$ for particular $\boldsymbol{\theta}$ values. Setting $\log \mathcal{L}(\mathcal{I}_T, \boldsymbol{\theta}) = -\infty$ in this case suffices for the purpose of maximum likelihood estimation.

3.2 SIR Particle Filter

If the state space model is not linear and Gaussian, both the joint distribution $p(x_{0:t}|\mathcal{I}_t)$ and the marginal distribution $p(x_t|\mathcal{I}_t)$ are usually not analytically solvable due to the intractability of the normalization constant $p(y_t|\mathcal{I}_{t-1})$. In this case we can only resort to sampling techniques to approximate these distribution densities.

Particle filtering methods (which constitute a sub-class of *Sequential Monto Carlo* methods) approximate the prediction density $p(x_t|\mathcal{I}_{t-1}, \boldsymbol{\theta})$ and filtering density $p(x_t|\mathcal{I}_t, \boldsymbol{\theta})$ sequentially by using importance sampling techniques. Many different method variants exist, all of which involve two basic steps: simulating from the transition density $p(x_{t+1}|x_t, \boldsymbol{\theta})$ and evaluating the measurement density $p(y_t|x_t, \boldsymbol{\theta})$.

Sequential Importance Resampling (SIR)

One of the best-known particle filter methods is the *Sequential Importance Resampling (SIR)* algorithm by Gordon et al. (1993). Let P be the number of particles (= samples) per state. The algorithm recursively computes prediction and filtering particles:

1. The **prediction step** obtains a new prediction particle for each filtering particle by propagating the system, using the transition density:

$$x^i_{t|t-1} \sim p(x_t|x^i_{t-1|t-1}, \theta) \quad \text{for } i = 1, \dots, P$$

2. The **filtering step** (or update step) computes the importance weight w^i_t of each prediction particle

$$w^i_t = \frac{p(y_t|x^i_{t|t-1}, \theta)}{\sum_{j=1}^{P} p(y_t|x^j_{t|t-1}, \theta)} \quad \text{for } i = 1, \dots, P$$

   and then picks the filtering particles via multinomial sampling, using the computed importance weights as respective probabilities:

$$x^j_{t|t} \sim MN(w^1_t, \dots, w^P_t) \quad \text{for } j = 1, \dots, P$$

The algorithm's main characteristic is the resampling within the filtering step which removes particles with small weights with high probability while likely copying particles with high weights multiple times. While this step increases the immediate Monte Carlo variance, it gives better stability for future steps by reducing the risk of weight degeneracy (Doucet and Johansen, 2008).

Algorithm

In the following version of the algorithm we assume the initialization parameters $\boldsymbol{\delta}$ to be known. For example, in the case of the local level model, $\boldsymbol{\delta} = [\boldsymbol{\mu}_1, \Sigma_1]^T$.

Algorithm 2 Sequential Importance Sampling

```
 1: procedure SIR($\mathcal{I}_T, \boldsymbol{\theta}, \boldsymbol{\delta}, P$)
 2:     for $i$ in $1 : P$ do
 3:         $x^i_{0|0} \sim p(x_{0|0}|\boldsymbol{\delta})$                    ▷ initialization
 4:     for $t$ in $1 : T$ do
 5:         for $i$ in $1 : P$ do
 6:             $x^i_{t|t-1} \sim p(x_t|x^i_{t-1|t-1}, \boldsymbol{\theta})$          ▷ prediction step
 7:             $\tilde{w}^i_t \leftarrow p(y_t|x^i_{t|t-1}, \boldsymbol{\theta})$      ▷ compute importance weights
 8:         for $i$ in $1 : P$ do
 9:             $w^i_t \leftarrow \tilde{w}^j_t / \sum_{j=1}^P \tilde{w}^j_t$               ▷ normalize weights
10:         for $i$ in $1 : P$ do
11:             $x^i_{t|t} \sim MN(w^1_t, \ldots, w^P_t)$                ▷ filtering step
12:     return $\{\{\boldsymbol{x}^i_{t|t}\}_{i=1}^P\}_{t=1}^T$
```

Algorithm 2 describes the recursive computation of the prediction and filtering particles. For our purpose, the algorithm returns the filtering particles which estimate the filtering density $p(x_t|\mathcal{I}_t, \boldsymbol{\theta})$ (for other purposes it could return the prediction particles instead). An estimate of the latent states can be constructed by averaging over the particles for each time t with equal weights:

$$\hat{x}_t = \frac{1}{P}\sum_{i=1}^{P} x^i_{t|t}$$

Their standard deviation (or other sample quantiles) can be used to construct confidence intervals for these estimates.

The sequence of filtering means $\{\boldsymbol{\mu}_{t|t}\}$ is the best possible prediction for the latent states $\{\boldsymbol{x}_t\}$. The sequence of covariances $\{\Sigma_{t|t}\}$ can be used to obtain confidence intervals for these estimates.

Example: Local Level Model

In the case of the univariate local level model with $\boldsymbol{\theta} = [\sigma^2_\eta, \sigma^2_\epsilon]^T$, the prediction step draws from a normal transition density: $p(x_t|x^i_{t-1|t-1}, \boldsymbol{\theta}) = N(x_{t-1|t-1}, \sigma^2_\eta)$ and the importance weights are evaluated on a normal measurement density: $p(y_t|x^i_{t|t-1}, \boldsymbol{\theta}) = N(x_{t|t-1}, \sigma^2_\epsilon)$. The state initialization density is $p(x_{0|0}|\boldsymbol{\delta}) = N(\mu_1, \Sigma_1)$.

Figure 3.2 shows the SIR particle filter estimate with $P = 200$ particles for the same realization of the local level model that has been plotted in *Figure 3.1*. The left plot shows the estimated states together with the 0.05 and 0.95 sample quantiles. The right plot compares the SIR particle estimates with the Kalman estimates. Note that the estimates

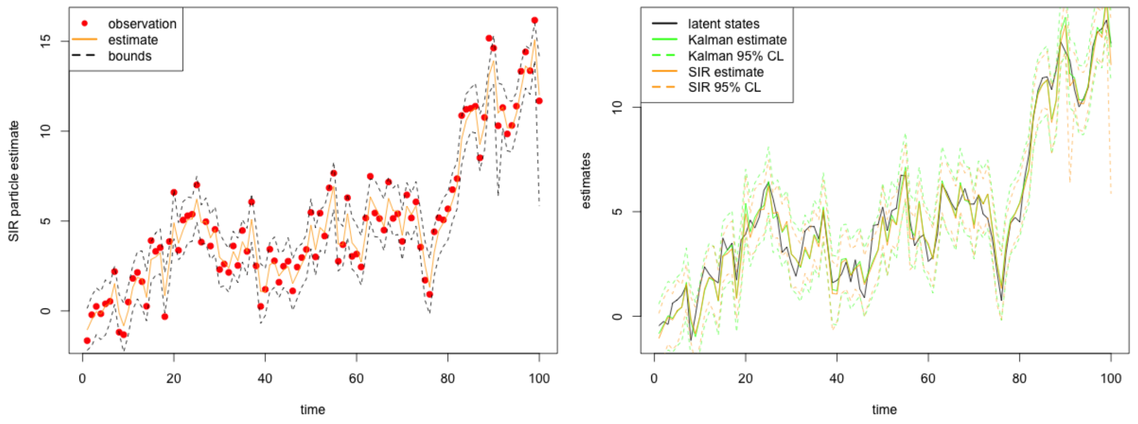

Figure 3.2: SIR particle estimate of univariate local level realization

of the Kalman filter and the SIR particle filter are almost identical. This is because the number of particles $P = 200$ was chosen to be sufficiently large. In general, the approximation error of particle filter methods goes 0 as $P \to \infty$.

Likelihood evaluation

The likelihood of a single observation given all previous observations $p(y_t|\mathcal{I}_{t-1}, \boldsymbol{\theta})$ can be estimated by simple Monte Carlo integration:

$$\begin{aligned} p(y_t|\mathcal{I}_{t-1}, \boldsymbol{\theta}) &= \int p(y_t|x_t, \boldsymbol{\theta}) p(x_t|\mathcal{I}_{t-1}, \boldsymbol{\theta}) dx_t \\ \hat{p}(y_t|\mathcal{I}_{t-1}, \boldsymbol{\theta}) &\approx \frac{1}{P} \sum_{i=1}^{P} p(y_t|x^i_{t|t-1}, \boldsymbol{\theta}) \\ &= \frac{1}{P} \sum_{i=1}^{P} \tilde{w}^i_t \end{aligned} \tag{3.3}$$

Thus, the likelihood can be easily constructed from the (unnormalized) importance weights in the particle filter. The estimated log-likelihood of the whole system, described the the series of observations $\mathcal{I}_T$, can then be defined as

$$\begin{aligned}\log \hat{\mathcal{L}}(\mathcal{I}_T, \boldsymbol{\theta}) &= \log \prod_{t=1}^{T} \hat{p}(y_t|\mathcal{I}_{t-1}, \boldsymbol{\theta}) \\ &= \sum_{t=1}^{T} \log \hat{p}(y_t|\mathcal{I}_{t-1}, \boldsymbol{\theta}) \\ &= \sum_{t=1}^{T} \log \frac{1}{P} \sum_{i=1}^{P} p(y_t|x^i_{t|t-1}, \boldsymbol{\theta})\end{aligned} \tag{3.4}$$

Note that for the sequential importance resampling algorithm, this log-likelihood estimator is not necessarily smooth with respect to $\boldsymbol{\theta}$. This can cause problems when trying to find the maximum like-likelihood. For this reason, *Continuous Sequential Importance Sampling (CSIR)* proves to be a better algorithm for parameter inference.

3.3 Importance Sampling Particle Filter

The SIR algorithm suffers from not being necessarily smooth with respect to the model parameters $\boldsymbol{\theta}$ which makes the method unfit for parameter inference. The alternative continuous SIR algorithm smoothes the log-likelihood with respect to $\boldsymbol{\theta}$, however, the algorithm only works in the univariate case. For this reason, Brownlees and Kristensen (2017) propose a novel particle filter method called *importance sampling particle filter* which smoothes the log-likelihood with respect to $\boldsymbol{\theta}$ and still works in the multivariate case.

The key idea of this new method is the following: Use an auxiliary, misspecified particle filter which does not depend on the model parameters $\boldsymbol{\theta}$ to compute the log-likelihood with respect to $\boldsymbol{\theta}$ via recursive importance sampling. Let $\tilde{p}(y_t|\tilde{x}_t)$ be the auxiliary measurement density, let $\tilde{p}(\tilde{x}_{t+1}|\tilde{x}_t)$ be the auxiliary transition density, and let $\tilde{p}(\tilde{x}_1)$ be the auxiliary initial density. Furthermore, let $\tilde{p}(\tilde{x}_t|\mathcal{I}_{t-1})$ and $\tilde{p}(\tilde{x}_t|\mathcal{I}_t)$ be the auxiliary prediction and filtering density, respectively (note that none of those densities depends on the model parameter $\boldsymbol{\theta}$).

We can use the samples from the misspecified, auxiliary prediction distribution

$$\begin{aligned}p(y_t|\mathcal{I}_{t-1}, \boldsymbol{\theta}) &= \int p(y_t|\tilde{x}_t, \boldsymbol{\theta}) p(\tilde{x}_t|\mathcal{I}_{t-1}, \boldsymbol{\theta}) d\tilde{x}_t \\ &= \int p(y_t|\tilde{x}_t, \boldsymbol{\theta}) \left[\frac{p(\tilde{x}_t|\mathcal{I}_{t-1}, \boldsymbol{\theta})}{p(\tilde{x}_t|\mathcal{I}_{t-1})}\right] p(\tilde{x}_t|\mathcal{I}_{t-1}) d\tilde{x}_t\end{aligned} \tag{3.5}$$

Algorithm

As input the algorithm takes the predictive and filter particles of a misspecified model with parameter vector $\tilde{\boldsymbol{\theta}}$.

Algorithm 3 Importance Sampling Particle Filter

```
1: procedure ISParticleFilter($\mathcal{I}_T, \{\{\tilde{\boldsymbol{x}}^i_{t|t}\}_{i=1}^P\}_{t=1}^T, \{\{\tilde{\boldsymbol{x}}^i_{t|t-1}\}_{i=1}^P\}_{t=1}^T, \boldsymbol{\theta}, \tilde{\boldsymbol{\theta}}$)
2:     for $i$ in $1:P$ do
3:         $is^i_{1|0} \leftarrow p(\tilde{x}_{1|0}|\boldsymbol{\theta})/p(\tilde{x}_{1|0}|\tilde{\boldsymbol{\theta}})$        ▷ initialize predictive importance weights
4:     for $t$ in $1:T$ do
5:         $w \leftarrow \tilde{w} \leftarrow 0$
6:         for $i$ in $1:P$ do        ▷ estimate likelihoods
7:             $w \leftarrow \tilde{w} + p(y_t|\tilde{x}^i_{t|t-1}, \boldsymbol{\theta}) is^i_{t|t-1}$
8:             $\tilde{w} \leftarrow \widehat{\tilde{w}} + p(y_t|\tilde{x}^i_{t|t-1}, \tilde{\boldsymbol{\theta}})$
9:         for $i$ in $1:P$ do        ▷ update filtering importance weights
10:            $f \leftarrow p(y_t|\tilde{x}^i_{t|t}, \boldsymbol{\theta})$
11:            $\tilde{f} \leftarrow p(y_t|\tilde{x}^i_{t|t}, \tilde{\boldsymbol{\theta}})$
12:            $is^i_{t|t} \leftarrow (\tilde{w}/w)(f/\tilde{f}) is^j_{t|t-1}$        ▷ where $j$ is such that $\tilde{x}^i_{t|t} = \tilde{x}^j_{t|t-1}$
13:        if $t < T$ then
14:            for $i$ in $1:P$ do        ▷ update predictive importance weights
15:                $p \leftarrow p(\tilde{x}_{t+1|t}|\tilde{x}_{t|t}, \boldsymbol{\theta})$
16:                $\tilde{p} \leftarrow p(\tilde{x}_{t+1|t}|\tilde{x}_{t|t}, \tilde{\boldsymbol{\theta}})$
17:                $is^i_{t+1|t} \leftarrow (p/\tilde{p}) is^i_{t|t}$
18:    return $\{\{\boldsymbol{x}^i_{t|t}\}_{i=1}^P\}_{t=1}^T$
```

Importantly, the algorithm does not produce latent state estimates and predictions. The only object of interest the algorithm returns is the likelihood of the model with $\boldsymbol{\theta}$ for parameter inference where this method has been shown to smooth the (log-)likelihood with respect to $\boldsymbol{\theta}$. The algorithm can not be used for latent state inference.

Likelihood evaluation

4 Illustration

4.1 Trivariate Local Level Model

The Model

Consider a time series of length T with each observation $\boldsymbol{y}_t = [y_{1t}, y_{2t}, y_{3t}]^T$ and each state $\boldsymbol{x}_t = [x_{1t}, x_{2t}, x_{3t}]^T$ being described by a 3-dimensional vector.

$$\begin{aligned} \text{observation:} \quad & \boldsymbol{y}_t = \boldsymbol{x}_t + \boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim N(\mathbf{0}, \sigma_\epsilon^2 I_3) \\ \text{state:} \quad & \boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \boldsymbol{\eta}_t, \quad \boldsymbol{\eta}_t \sim N(\mathbf{0}, \Sigma_\eta) \end{aligned}$$

with initial state $\boldsymbol{x}_1 \sim N(\boldsymbol{a}_1, P_1)$ and where we restrict the covariance matrix of the state disturbances, Σ_η, to the form

$$\Sigma_\eta = \begin{bmatrix} \sigma_{\eta 1}^2 & \rho\sigma_{\eta 1}\sigma_{\eta 2} & \rho\sigma_{\eta 1}\sigma_{\eta 3} \\ \rho\sigma_{\eta 1}\sigma_{\eta 2} & \sigma_{\eta 2}^2 & \rho\sigma_{\eta 2}\sigma_{\eta 3} \\ \rho\sigma_{\eta 1}\sigma_{\eta 3} & \rho\sigma_{\eta 2}\sigma_{\eta 3} & \sigma_{\eta 3}^2 \end{bmatrix}$$

Thus, Σ_η can be described by $\sigma_{\eta 1}^2$, $\sigma_{\eta 2}^2$, $\sigma_{\eta 3}^2 > 0$ and $\rho \in [0, 1]$. Furthermore, we assume for simplicity that the observation noise has the same variance in each dimension $\sigma_\epsilon^2 > 0$. Therefore, the model is fully specified by the following vector of parameters:

$$\boldsymbol{\theta} = [\rho, \sigma_{\eta 1}^2, \sigma_{\eta 2}^2, \sigma_{\eta 3}^2, \sigma_\epsilon^2]^T$$

The initial state parameters $\boldsymbol{a}_1$ and P_1 are assumed to be known.

Realization

Figure 4.1 plots the states and observations for a realization of the trivariate local level model with length $T = 100$. The model parameters are

$$\boldsymbol{\theta} = [\rho = 0.7, \sigma_{\eta 1}^2 = 4.2, \sigma_{\eta 2}^2 = 2.8, \sigma_{\eta 3}^2 = 0.9, \sigma_\epsilon^2 = 1.0]^T$$

The initial state x_1 is drawn from a standard normal.

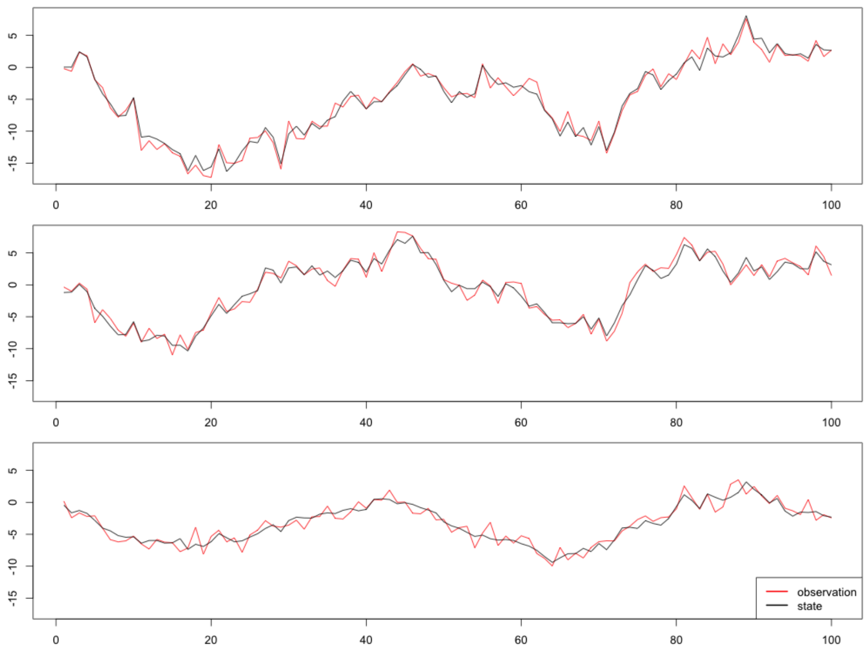

Figure 4.1: Realization of the model with $T = 100$

4.2 Hierarchical Dynamic Poisson Model

Explain the main idea and potential use cases.

The Model

Consider a time series over M days, each consisting of N intra-daily observations. Let m denote the day and n be the intraday index.

$$\begin{aligned}
\text{observation:} \quad y_{mn} &= \text{Poisson}(\lambda_{mn}) \\
\text{state:} \quad \log \lambda_{mn} &= \log \lambda_m^{(D)} + \log \lambda_{mn}^{(I)} + \log \lambda_n^{(P)}
\end{aligned}$$

where the state consists of a daily, an intra-daily, and a periodic component:

$$\begin{aligned}
\text{daily component:} \quad \log \lambda_{m+1}^{(D)} &= \phi_0^{(D)} + \phi_1^{(D)} \log \lambda_m^{(D)} + \eta_m^{(D)} & \eta_t \sim N(0, \sigma_{(D)}^2) \\
\text{intra-daily component:} \quad \log \lambda_{mn+1}^{(I)} &= \phi_1^{(I)} \log \lambda_{mn}^{(I)} + \eta_{mn}^{(I)} & \eta_{mn} \sim N(0, \sigma_{(I)}^2) \\
\text{periodic component:} \quad \log \lambda_n^{(P)} &= \phi_1^{(P)} \sin(\pi(n-1)/M)
\end{aligned}$$

The initial daily and intra-daily component is drawn from a normal with mean a_1 and covariance P_1:

$$\log \lambda_1^{(D)}, \log \lambda_1^{(I)} \sim N(a_1, P_1)$$

Note that both the daily and intra-daily component constitute an AR(1) model, with the mean of the intra-daily component $\phi_0^{(I)}$ set to 0. The model is fully specified by the following vector of parameters:

$$\boldsymbol{\theta} = [\phi_0^{(D)}, \phi_1^{(D)}, \sigma^2_{(D)}, \phi_1^{(I)}, \sigma^2_{(I)}, \phi_1^{(P)}]^T$$

Again, the initial state parameters a_1 and P_1 are assumed to be known.

Realization

Figure 4.2 plots the states and observations for a realization of the hierarchical dyanmic Poisson model over $N = 5$ days with $M = 20$ intra-daily observations. The model parameters are

$$\boldsymbol{\theta} = [\phi_0^{(D)} = 0.7, \phi_1^{(D)} = 0.6, \sigma^2_{(D)} = 0.6, \phi_1^{(I)} = 0.3, \sigma^2_{(I)} = 0.2, \phi_1^{(P)} = 0.8]^T$$

The initial daily and intra-daily state components were drawn from a standard normal.

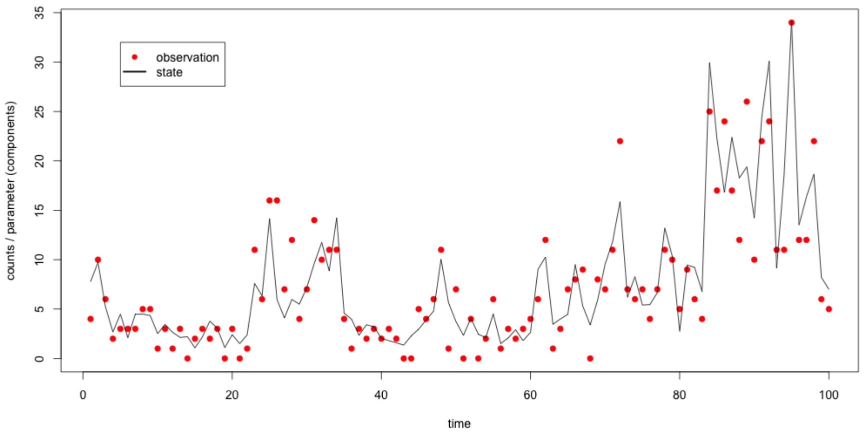

Figure 4.2: Realization of the model with $N = 5$ and $M = 20$

Densities

State transition and prediction density and how they are used in the particle filter

Maximum Likelihood Estimation

Show log-likelihood plots

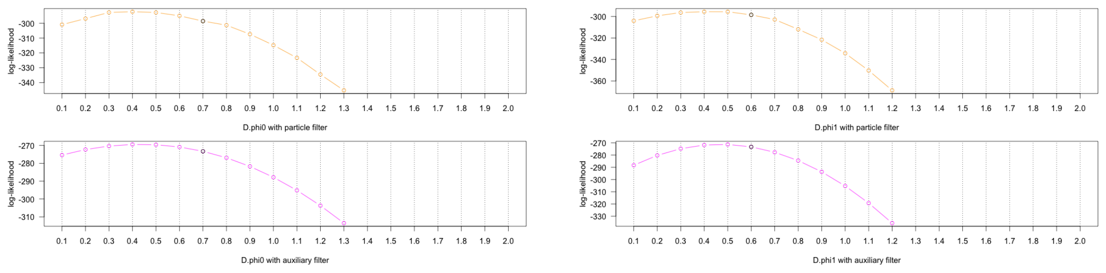

Figure 4.3: Belief convergence without misinformation after 300 and 2000 iterations

5 Evaluation

5.1 Method Comparison

Compare presented filters with respect to their potential use. Kalman filter only works for linear Gaussian state space models. SIR works for latent state inference but not well for parameter inference. CSIR works for both latent state and parameter inference, however, only in the univariate case. Importance sampling particle filter works for parameter inference but cannot be used for latent state inference. Present table with comment column

5.2 Parameter Inference

Compare bias, SE, and MSE for univiariate local level

Particle filtering suffers from propagation error. Current approximations errors are related to past approximation errors. The number of particles P should increase with the sample size T to guarantee a good degree of approximation.

6 Conclusion

by Etessami et al.[**?**]

Bibliography

[1] C. T. Brownlees and D. Kristensen. An importance sampling particle filtering algorithm for the estimation of dynamic latent variable models. *Working paper*, 2017.

[2] A. Doucet and A. M. Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. *Technical report*, 2008.

[3] N. Gordon, D. Salmond, and A. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *Radar and Signal Processing, IEE Proceedings F*, 140(2):107–113, 1993.

[4] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASMEJournal of Basic Engineering*, 82(Series D):35–45, 1960.