Particle Filters for Nonlinear State Space Models

Hans-Peter Höllwirth

Supervisor:
Dr. Christian Brownlees

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

Abstract

This paper empirically studies a novel particle filter method proposed by Brownlees and Kristensen (2017) for parameter estimation of nonlinear state space models. The particle filter, named *Importance Sampling Particle Filter*, is tested and compared to other established particle filters on two variations of a local level model. Inspections of the log-likelihood plots with respect to model parameters, as well as Monte Carlo maximum likelihood estimations establish the correctness of the new method. Finally, the novel particle filter is successfully applied to a nonlinear state space models, the hierarchical dynamic Poisson model.

Acknowledgement

I am grateful to my supervisor, Dr. Christian Brownlees, who guided and supported me throughout the completion of this project while giving me the room to explore my own ideas.

Contents

1 Introduction **4**

2 State Space Models **5**
- 2.1 Local Level Model 5
- 2.2 Latent State Inference 6
- 2.3 Parameter Inference 6

3 Filtering **8**
- 3.1 Kalman Filter 8
- 3.2 SIR Particle Filter 11
- 3.3 CSIR Particle Filter 14
- 3.4 Importance Sampling Particle Filter 16

4 Evaluation **22**
- 4.1 Method Comparison 22
- 4.2 Parameter Inference 22

5 Illustration **25**
- 5.1 Trivariate Local Level Model 25
- 5.2 Hierarchical Dynamic Poisson Model 29

6 Conclusion **33**

1 Introduction

Particle filtering has become an established technique to solve state space models. Theory exists since the 1950s and Kalman filter was proposed in 1960, however, particle filtering then was computationally too expensive. They work online to approximate the marginal distribution of the latent process as observations become available. Importance sampling is used at each time point to approximate the distribution with a set of discrete values, known as particles, each with a corresponding weight.

Particle filtering are commonly used for two types of inference problems: Inferring the latent state from observations, and estimating the state space model parameters. Established particle filter methods are limited with regards to their inference ability to particular state space models. The Kalman filter method (Kalman, 1960) is restricted to linear Gaussian models for which the method uses the fact that the prediction and filtering densities can be solved analytically. For nonlinear models, these densities can only be approximated via sampling. The SIR algorithm

For this reason, Brownlees and Kristensen (2017) proposed a new method called *Importance Sampling Particle Filter* which works so and so. The main purpose of this project was to implement and test the new method and to compare the method with other established particle filter algorithms.

The report is structured as follows. I explain the basic concept of state space models In *Chapter 2*. I introduce the univariate local level model which I use in following chapters to illustrate filter outputs. In *Chapter 3* I then introduce several filter methods, including the Kalman filter, the Sequential Importance Resampling (SIR) algorithm, the related Continuous Sequential Importance Resampling (CSIR) algorithm, as well as the novel particle filter called Importance Sampling Particle Filter. Next, I compare these methods and use Monte Carlo tests to compare their inference performance in *Chapter 4*. Finally, I illustrate the filter methods on two different state space models: A trivariate local level model and a nonlinear hierarchical dynamic Poisson model. I present the results in *Chapter 5*.

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 associated 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 the 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 some value x_1. The transition and measurement density of the local level model are simple to deduce:

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

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

$$p(x_t|\mathcal{I}_t, \boldsymbol{\theta}) \propto p(y_t|x_t, \boldsymbol{\theta}) p(x_t|\mathcal{I}_{t-1}, \boldsymbol{\theta})$$

where the first term is the measurement density for observation y_t, and the second term is the prediction density from the equation before.

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:

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

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 the latent process $p(x_{0:t}|y_{0:t})$ are 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 of $\boldsymbol{x}_{t-1}$).

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

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

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

   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.

Algorithm

Algorithm 1 (Local Level) Kalman filter

procedure KALMANFILTER($\mathcal{I}_T, \boldsymbol{\theta}, \boldsymbol{\mu}_1, \Sigma_1$)
  $\boldsymbol{\mu}_{1|0} \leftarrow \boldsymbol{\mu}_1$ ▷ initialization
  $\Sigma_{1|0} \leftarrow \Sigma_1$
  for t `in` $1:T$ **do**
    $\boldsymbol{v}_t = \boldsymbol{y}_t - \boldsymbol{\mu}_{t|t-1}$
    $K_t = \Sigma_{t|t-1}(\Sigma_{t|t-1} + \Sigma_\epsilon)^{-1}$
    $\boldsymbol{\mu}_{t|t} = \boldsymbol{\mu}_{t|t-1} + K_t \boldsymbol{v}_t$ ▷ update step
    $\Sigma_{t|t} = \Sigma_{t|t-1}(1 - K_t)$
    $\boldsymbol{\mu}_{t+1|t} \leftarrow \boldsymbol{\mu}_{t|t}$ ▷ prediction step
    $\Sigma_{t+1|t} \leftarrow \Sigma_{t|t} + \Sigma_\eta$
  return $\{\boldsymbol{\mu}_{t|t}\}$, $\{\Sigma_{t|t}\}$

Algorithm 1 describes the recursive update of the prediction and filtering density. To start the recursion, we need an initial state density with $\boldsymbol{\mu}_1$ and Σ_1. In our 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.

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 90% confidence level (right plot).

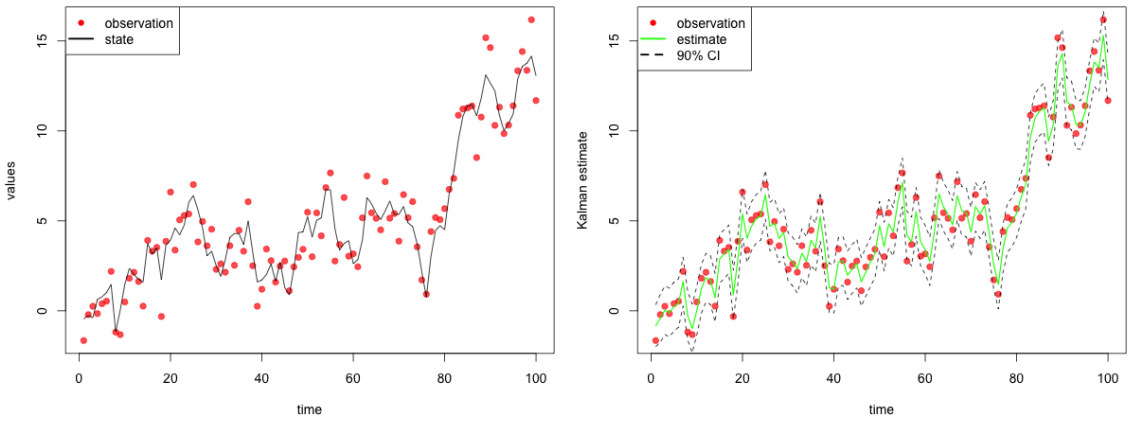

Figure 3.1: Kalman estimate of univariate local level realization

Likelihood evaluation

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

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

Its elements F_t and $\boldsymbol{v}_t$ 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:

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

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

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

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

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

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 Resampling Particle Filter

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

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

Sample quantiles can be used to construct confidence intervals for these estimates. Throughout this paper, I report the 0.05 and 0.95 quantiles to form the 90% confidence interval.

Example: Local Level Model

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

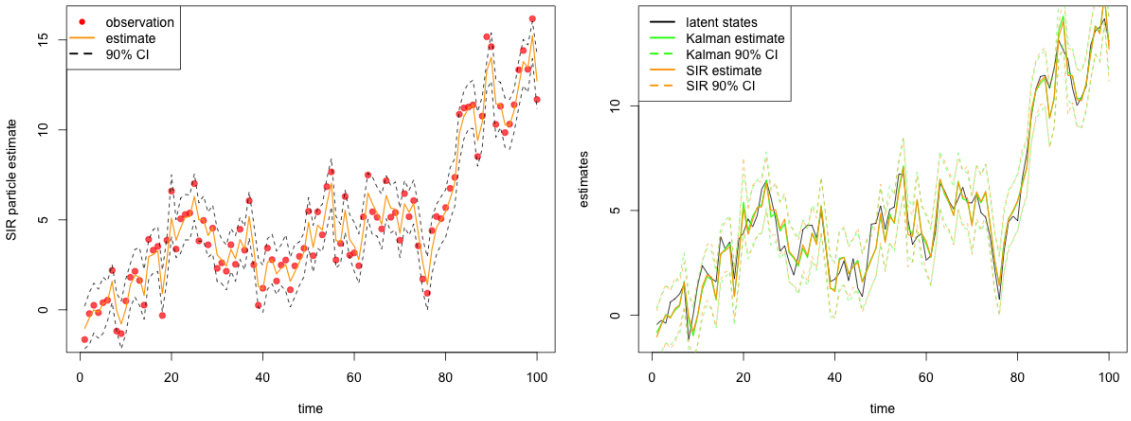

Figure 3.2: SIR particle estimate of univariate local level realization

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:

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

$$\hat{p}(y_t|\mathcal{I}_{t-1}, \boldsymbol{\theta}) \approx \frac{1}{P} \sum_{i=1}^{P} p(y_t|x^i_{t|t-1}, \boldsymbol{\theta}) = \frac{1}{P} \sum_{i=1}^{P} \tilde{w}^i_t$$

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

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

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 we want to use the algorithm for parameter inference where the optimizer might get stuck in a local optimum. Thus, for the purpose of parameter inference, other particle filters such as *Continuous Sequential Importance Resampling (CSIR)* are usually preferred.

3.3 CSIR Particle Filter

The *Continuous Sequential Importance Resampling (CSIR)* particle filter by Malik and Pitt (2011) adapts the filtering step in the SIR algorithm in a way that smoothes the (log-)likelihood with respect to the unknown parameters $\boldsymbol{\theta}$. In particular, it replaces the multinomial sampling step (that uses the importance weights as respective probabilities) by an alternative sampling step which is described in *Algorithm 3*. The prediction step and the computation of the importance weights are the same for the CSIR particle filter as the steps described in *Algorithm 2*.

Algorithm

The algorithm takes P predictive particles and their associated, normalized importance weights as input. Instead of bootstrapping from that set of particles with their associated weight as probability, the algorithm sorts the particles in ascending order, forms its (discrete) cumulative distribution function (cdf), and then interpolates this cdf to obtain a continuous cumulative distribution function. The interpolation lets the continuous cdf pass through the mid-point of each step in the discrete cdf. The algorithm then samples filtering particles from the inverted, smooth cumulative distribution function. The details are described in *Algorithm 3*.

Algorithm 3 Continuous Sequential Importance Resampling Step

procedure CSIR($\{x^i_{t|t-1}\}_{i=1}^P, \{w^i_t\}_{i=1}^P$)
  $u \leftarrow$ sort(Uniform(P, 0, 1))
  $\{(\hat{x}^i_{t|t-1}, \hat{w}^i_t)\}_{i=1}^P \leftarrow$ sort($\{(x^i_{t|t-1}, w^i_t)\}_{i=1}^P$ by $x_{t|t-1}$) ▷ sort particles
  $cw_1 \leftarrow 0$
  for i `in` $1:P$ **do**
    $cw_{i+1} \leftarrow cw_i + \hat{w}^i_t$ ▷ cumulative weights
  $j \leftarrow 1$
  for i `in` $1:P$ **do** ▷ continuous resampling
    while $cw_i < u_j$ `and` $u_j \leq cw_{i+1}$ `and` $j < P$ **do**
      $x^j_{t|t} \leftarrow (x^{i+1}_{t|t-1} - x^i_{t|t-1})/(cw_{i+1} - cw_i) \times (u_j - cw_i)$
      if $j < P$ **then**
        $j \leftarrow j + 1$
      else
        break
  return $\{x^i_{t|t}\}_{i=1}^P$

Note that the sorting step of the particles in the CSIR algorithm prevents this method to be used for multivariate particles. In other words, the CSIR algorithm is only applicable for univariate state space models.

Example: Local Level Model

Figure 3.3 compares the log-likelihood function of the SIR with the CSIR algorithm for a realization of a local level model with respect to σ^2_η.

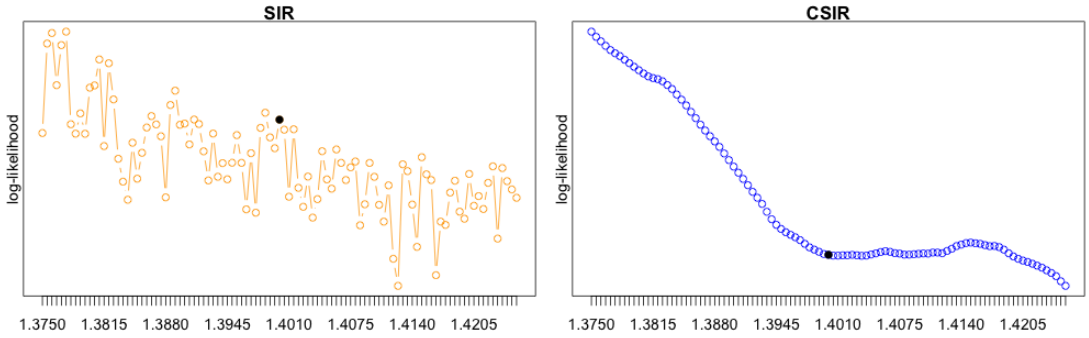

Figure 3.3: Log-likelihood as a function of σ^2_η with SIR and CSIR filter

The graphs show the measured log-likelihood around the true parameter $\sigma_\eta^2 = 1.4$ (highlighted with a black dot) with small intervals of 0.0005. The plots illustrate how the CSIR method smoothes the log-likelihood function which in the case of the SIR algorithm is cluttered with many local optima and so makes the method unsuited for parameter inference. Note though that the log-likelihood function of the CSIR method is not a concave function either.

3.4 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 with parameter vector $\tilde{\boldsymbol{\theta}}$ (independent of $\boldsymbol{\theta}$) to compute the log-likelihood with respect to $\boldsymbol{\theta}$ via recursive importance sampling. Let $\{\{\tilde{\boldsymbol{x}}_{t|t}^i\}_{i=1}^P\}_{t=1}^T$ and $\{\{\tilde{\boldsymbol{x}}_{t|t-1}^i\}_{i=1}^P\}_{t=1}^T$ be the sets of filtering and predictive particles generated by the auxiliary, misspecified particle filter and which approximate $\tilde{x}_{0:T}$. We can then use these particles to approximate the likelihood of parameter vector $\boldsymbol{\theta}$:

$$
\begin{aligned}
p(y_t|\mathcal{I}_{t-1}, \boldsymbol{\theta}) &= \int p(y_t|\tilde{x}_t, \boldsymbol{\theta}) p(\tilde{x}_t|\mathcal{I}_{t-1}, \boldsymbol{\theta}) d\tilde{x}_t \\
&= \int p(y_t|\tilde{x}_t, \boldsymbol{\theta}) \left[\frac{p(\tilde{x}_t|\mathcal{I}_{t-1}, \boldsymbol{\theta})}{p(\tilde{x}_t|\mathcal{I}_{t-1}, \tilde{\boldsymbol{\theta}})}\right] p(\tilde{x}_t|\mathcal{I}_{t-1}, \tilde{\boldsymbol{\theta}}) d\tilde{x}_t \\
\hat{p}(y_t|\mathcal{I}_{t-1}, \boldsymbol{\theta}) &\approx \frac{1}{P}\sum_{i=1}^P p(y_t|\tilde{x}_{t|t-1}^i, \boldsymbol{\theta}) \left[\frac{p(\tilde{x}_{t|t-1}^i|\mathcal{I}_{t-1}, \boldsymbol{\theta})}{p(\tilde{x}_{t|t-1}^i|\mathcal{I}_{t-1}, \tilde{\boldsymbol{\theta}})}\right] \\
&= \frac{1}{P}\sum_{i=1}^P p(y_t|\tilde{x}_{t|t-1}^i, \boldsymbol{\theta}) is_{t|t-1}^i
\end{aligned}
$$

where $is_{t|t-1}^i$ is called the *predictive importance weight* of particle $\tilde{x}_{t|t-1}^i$. The predictive importance weights can be computed in a recursive fashion:

1. The **predictive importance weight** $is_{t|t-1}^i$ is computed for each propagated particle $\tilde{x}_{t|t-1}^i$, using the transition density functions with parameter vectors $\boldsymbol{\theta}$ and $\tilde{\boldsymbol{\theta}}$,

and the *filtering importance weight* $is^i_{t-1|t-1}$ of the filtering particle $\tilde{x}^i_{t-1|t-1}$:

$$\begin{aligned} is^i_{t|t-1} &= \frac{p(\tilde{x}^i_{t|t-1}|\mathcal{I}_{t-1}, \boldsymbol{\theta})}{p(\tilde{x}^i_{t|t-1}|\mathcal{I}_{t-1}, \tilde{\boldsymbol{\theta}})} \\ &= \left[\frac{p(\tilde{x}^i_{t|t-1}|\tilde{x}^i_{t-1|t-1}, \boldsymbol{\theta})}{p(\tilde{x}^i_{t|t-1}|\tilde{x}^i_{t-1|t-1}, \tilde{\boldsymbol{\theta}})}\right]\left[\frac{p(\tilde{x}^i_{t-1|t-1}|\mathcal{I}_{t-1}, \boldsymbol{\theta})}{p(\tilde{x}^i_{t-1|t-1}|\mathcal{I}_{t-1}, \tilde{\boldsymbol{\theta}})}\right] \\ &= \left[\frac{p(\tilde{x}^i_{t|t-1}|\tilde{x}^i_{t-1|t-1}, \boldsymbol{\theta})}{p(\tilde{x}^i_{t|t-1}|\tilde{x}^i_{t-1|t-1}, \tilde{\boldsymbol{\theta}})}\right] is^i_{t-1|t-1} \end{aligned}$$

2. The **filtering importance weight** $is^i_{t|t}$ is computed for each filtering particle $\tilde{x}^i_{t|t}$, using the measurement density functions with parameter vectors $\boldsymbol{\theta}$ and $\tilde{\boldsymbol{\theta}}$, and the predictive importance weight $is^i_{t|t-1}$ of the predictive particle $\tilde{x}^i_{t|t-1}$:

$$\begin{aligned} is^i_{t|t} &= \frac{p(\tilde{x}^i_{t|t}|\mathcal{I}_t, \boldsymbol{\theta})}{p(\tilde{x}^i_{t|t}|\mathcal{I}_t, \tilde{\boldsymbol{\theta}})} \\ &= \left(\frac{p(y_t|\tilde{x}^i_{t|t}, \boldsymbol{\theta})p(\tilde{x}^i_{t|t}|\mathcal{I}_{t-1}, \boldsymbol{\theta})}{p(y_t|\mathcal{I}_{t-1}, \boldsymbol{\theta})}\right) / \left(\frac{p(y_t|\tilde{x}^i_{t|t}, \tilde{\boldsymbol{\theta}})p(\tilde{x}^i_{t|t}|\mathcal{I}_{t-1}, \tilde{\boldsymbol{\theta}})}{p(y_t|\mathcal{I}_{t-1}, \tilde{\boldsymbol{\theta}})}\right) \\ &= \left[\frac{p(y_t|\tilde{x}^i_{t|t}, \boldsymbol{\theta})}{p(y_t|\tilde{x}^i_{t|t}, \tilde{\boldsymbol{\theta}})}\right]\left[\frac{p(y_t|\mathcal{I}_{t-1}, \tilde{\boldsymbol{\theta}})}{p(y_t|\mathcal{I}_{t-1}, \boldsymbol{\theta})}\right]\left[\frac{p(\tilde{x}^i_{t|t}|\mathcal{I}_{t-1}, \boldsymbol{\theta})}{p(\tilde{x}^i_{t|t}|\mathcal{I}_{t-1}, \tilde{\boldsymbol{\theta}})}\right] \\ &= \left[\frac{p(y_t|\tilde{x}^i_{t|t}, \boldsymbol{\theta})}{p(y_t|\tilde{x}^i_{t|t}, \tilde{\boldsymbol{\theta}})}\right]\left[\frac{p(y_t|\mathcal{I}_{t-1}, \tilde{\boldsymbol{\theta}})}{p(y_t|\mathcal{I}_{t-1}, \boldsymbol{\theta})}\right] is^j_{t|t-1} \end{aligned}$$

   where j is such that $\tilde{x}^i_{t|t} = \tilde{x}^j_{t|t-1}$. Note that the likelihood term $p(y_t|\mathcal{I}_{t-1}, \boldsymbol{\theta})$ has been defined before and that the auxiliary likelihood term $p(y_t|\mathcal{I}_{t-1}, \tilde{\boldsymbol{\theta}})$ is simply the likelihood with the all predictive importance weights $is^i_{t|t-1} = 1$.

The recursion is started by evaluating $is^i_{1|0}$ with initial densities $p(\tilde{x}^i_{1|0}, \boldsymbol{\theta})$ and $p(\tilde{x}^i_{1|0}, \tilde{\boldsymbol{\theta}})$ which are assumed to be known here.

Algorithm

Algorithm 4 sketches the method. The algorithm takes the observations $\mathcal{I}_T$, the predictive and filter particles of the misspecified model with parameter vector $\tilde{\boldsymbol{\theta}}$, the auxiliary parameter vector $\tilde{\boldsymbol{\theta}}$ itself, and the parameter vector $\boldsymbol{\theta}$ for which we want to estimate the log-likelihood as input. We can use the sequential importance sampling (SIR) algorithm to generate the predictive and filter particles with any sensible choice of parameter values

in $\tilde{\boldsymbol{\theta}}$. Note that the algorithm does not work with particles produced from the continuous sequential importance sampling (CSIR) algorithm.

Algorithm 4 Importance Sampling Particle Filter

procedure ISPARTICLEFILTER($\mathcal{I}_T, \{\{\tilde{\boldsymbol{x}}^i_{t|t}\}_{i=1}^P\}_{t=1}^T, \{\{\tilde{\boldsymbol{x}}^i_{t|t-1}\}_{i=1}^P\}_{t=1}^T, \boldsymbol{\theta}, \tilde{\boldsymbol{\theta}}$)
  $loglik \leftarrow 0$
  for i `in` $1:P$ **do**
    $is^i_{1|0} \leftarrow p(\tilde{x}_{1|0}|\boldsymbol{\theta})/p(\tilde{x}_{1|0}|\tilde{\boldsymbol{\theta}})$ ▷ initialize predictive importance weights
  for t `in` $1:T$ **do**
    $w_t \leftarrow \tilde{w}_t \leftarrow 0$
    for i `in` $1:P$ **do** ▷ estimate likelihoods
      $w_t \leftarrow \tilde{w} + p(y_t|\tilde{x}^i_{t|t-1}, \boldsymbol{\theta}) is^i_{t|t-1}/P$
      $\tilde{w}_t \leftarrow \widehat{\tilde{w}} + p(y_t|\tilde{x}^i_{t|t-1}, \tilde{\boldsymbol{\theta}})/P$
    $loglik \leftarrow loglik + \log w_t$
    for i `in` $1:P$ **do** ▷ update filtering importance weights
      $f \leftarrow p(y_t|\tilde{x}^i_{t|t}, \boldsymbol{\theta})$
      $\tilde{f} \leftarrow p(y_t|\tilde{x}^i_{t|t}, \tilde{\boldsymbol{\theta}})$
      $is^i_{t|t} \leftarrow (\tilde{w}_t/w_t)(f/\tilde{f}) is^j_{t|t-1}$ ▷ where j is such that $\tilde{x}^i_{t|t} = \tilde{x}^j_{t|t-1}$
    if $t < T$ **then**
      for i `in` $1:P$ **do** ▷ update predictive importance weights
        $p \leftarrow p(\tilde{x}_{t+1|t}|\tilde{x}_{t|t}, \boldsymbol{\theta})$
        $\tilde{p} \leftarrow p(\tilde{x}_{t+1|t}|\tilde{x}_{t|t}, \tilde{\boldsymbol{\theta}})$
        $is^i_{t+1|t} \leftarrow (p/\tilde{p}) is^i_{t|t}$
  return $loglik$

Importantly, the algorithm does not produce any estimates or predictions on the latent states. The only object of interest the algorithm returns is the (log-)likelihood of the model with $\boldsymbol{\theta}$. This allows us to use the algorithm for parameter inference where it smoothes the (log-)likelihood with respect to $\boldsymbol{\theta}$. The algorithm can not be used for latent state inference.

Example: Local Level Model

We consider the realization of an univariate local model with parameters plotted in *Figure 3.1* and take as input the filtering and prediction particles produced by the SIR algorithm whose latent state estimates have been plotted in *Figure 3.2*. Note again that the importance sampling particle filter does not produce estimates of the latent state.

Figure 3.4 shows the mean filtering and predictive importance weights when we run the importance sampling particle filter with the true parameters $\boldsymbol{\theta} = [\sigma_\eta^2 = 1.4, \sigma_\epsilon^2 = 1.0]^T$, using auxiliary parameters $\tilde{\boldsymbol{\theta}} = [\tilde{\sigma}_\eta^2 = 1.0, \tilde{\sigma}_\epsilon^2 = 1.0]^T$. Observe that the means stay in a rather narrow bandwidth around 1 and importantly do not diverge to zero or infinity at any point.

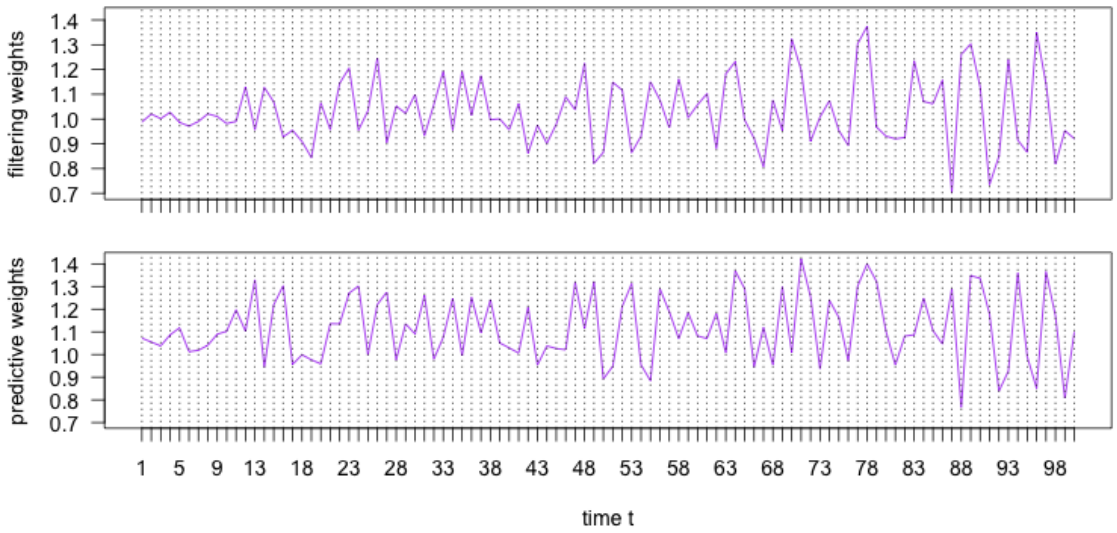

Figure 3.4: Importance weights with $P = 200$ particles

Likelihood evaluation

As explained earlier, the likelihood is directly evaluated by the importance sampling particle filter algorithm and is actually its main output. The algorithm approximates the log-likelihood via importance sampling:

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

Its main attractive feature is that the log-likelihood is smooth with respect to $\boldsymbol{\theta}$. That claim is supported by the plots in in *Figure 3.5* which compare the log-likelihood of different

methods for different values of σ_η^2 (while σ_ϵ^2 is kept at its true value 1.0): the Kalman filter (green), the SIR particle filter (orange), the CSIR particle filter (blue), and the importance sampling particle filter (magenta).

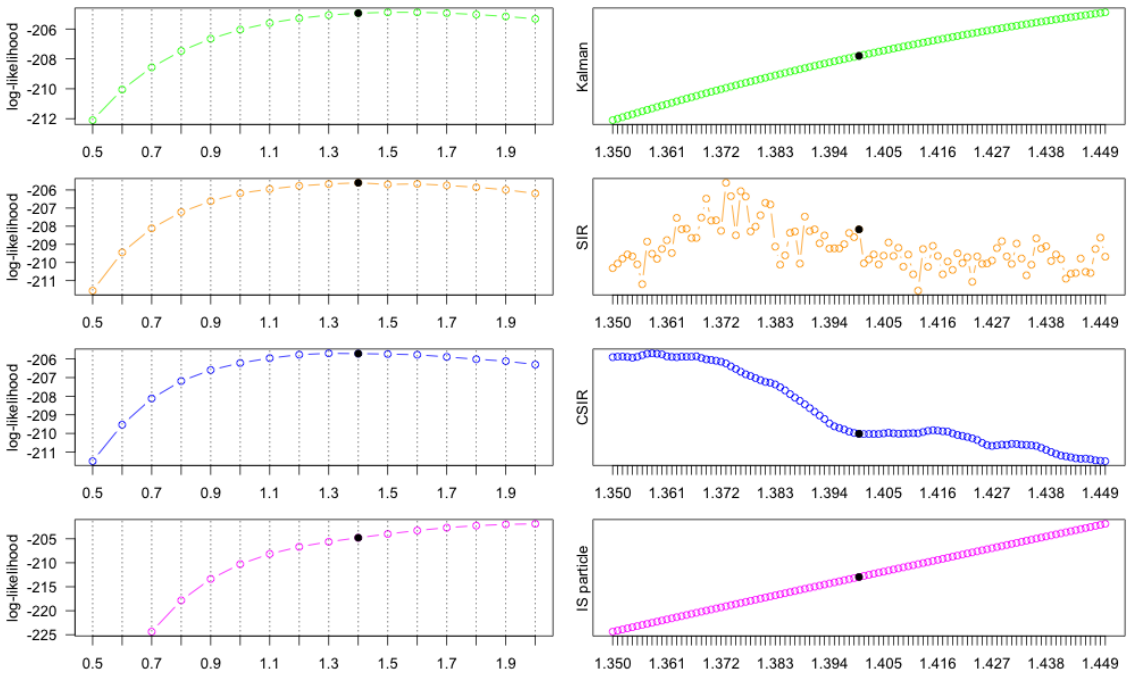

Figure 3.5: Log-likelihood as a function of σ_η^2 with different filters

The left plots show the log-likelihood for larger step sizes (0.1) of σ_η^2 (again, with the true parameter $\sigma_\eta^2 = 1.4$ highlighted with a black point). The function with respect to σ_η^2 appears to be smooth for all four filters. However, as the right plots show, the picture changes when we decrease the step size to 0.001 around the true σ_η^2. While both the Kalman and the IS particle filter still produce a smooth, concave log-likelihood curve, the log-likelihood plots of the SIR and CSIR particle filters no longer appear to be concave. The SIR particle filter in particular is cluttered with many local optima.

Finally, we want to understand how the choice of the auxiliary parameter values $\tilde{\boldsymbol{\theta}}$ impact the log-likelihood. In the simulation of the importance sampling particle filter in *Figure 3.5*, the auxiliary parameters $\tilde{\boldsymbol{\theta}} = [\tilde{\sigma}_\eta^2, \tilde{\sigma}_\epsilon^2]^T$ have been set to the true parameters $\sigma_\eta^2 = 1.4$ and $\sigma_\epsilon^2 = 1.0$. *Figure 3.6* shows how the choice of the auxiliary parameters impact the log-likelihood values of the IS particle filter and its shape with respect to $\boldsymbol{\theta}$.

The left plot shows the log-likelihood evaluated for the true parameters as a function of

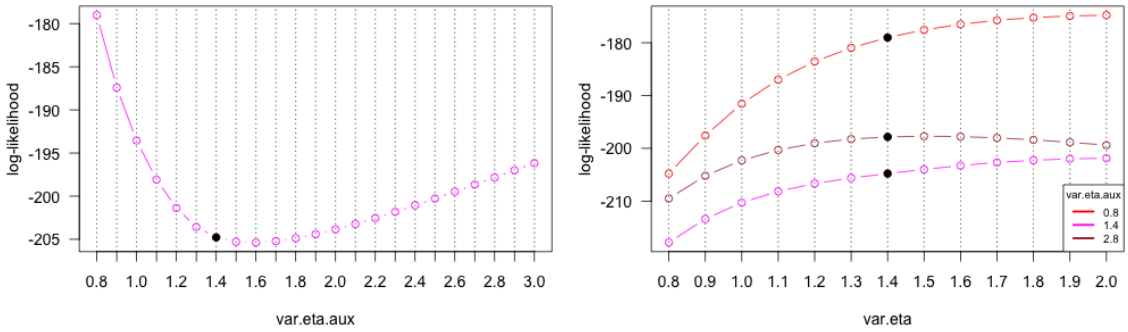

Figure 3.6: Impact of choice of $\tilde{\sigma}_\eta^2$ on log-likelihood of IS particle filter

the auxiliary parameter $\tilde{\sigma}_\eta^2$ (meanwhile, $\tilde{\sigma}_\epsilon^2$ is kept fixed at $\sigma_\epsilon^2 = 1.0$). The log-likelihood is smallest close to $\tilde{\sigma}_\eta^2 = \sigma_\eta^2 = 1.4$ and smoothly increases as $\tilde{\sigma}_\eta^2$ moves further away. The graph shows that the value of the log-likelihood of the IS particle filter depends on the choice of the auxiliary parameters.

The right graph in *Figure 3.6* plots the log-likelihood for the same filtering and predictive particles as a function of σ_η^2 for 3 choices of $\tilde{\sigma}_\eta^2$: 0.8, 1.4, and 2.8. We see that the shape of the curves are similar (in particular, concave), but do not peak at the same value of σ_η^2.

4 Evaluation

4.1 Method Comparison

Having introduced four different particle filters, it is important to understand which filter to select for a given model and question. *Table 4.1* summarizes for which models and inference types the presented filters can be potentially used.

Filter	Latent state	Parameter	Comment
Kalman	x	x	linear Gaussian models only
SIR	x		
CSIR	x	x	univariate models only
IS		x	

Table 4.1: Summary of inference with presented filters

The Kalman filter only works for linear Gaussian state space models. Kalman's latent state estimates statistically minimize the error and so is optimal with respect to many different statistical measures, eg. the mean squared error. The sequential importance resampling (SIR) particle filter works for latent state inference but usually performs poorly for parameter inference where the log-likelihood with respect to the parameters lacks smoothness. The continuous SIR method smoothes the log-likelihood with respect to the parameters and so works for both latent state and parameter inference. However, CSIR only applies to univariate models. Finally, the novel importance sampling (IS) particle filter solves parameter inference for any type of state space models. The method does not produce any filtering or prediction particles though and so cannot be used for latent state inference.

4.2 Parameter Inference

Next, I use Monte Carlo simulations to compare the different filters on how they perform with respect to parameter inference, measured by bias, standard error, and mean squared error (MSE). I test the performance on random realizations of the univariate local level model with $\sigma_\eta^2 = 1.4$ and $\sigma_\epsilon^2 = 1.0$ of several different lengths T. The CSIR method and the importance sampling (IS) particle filter are tested for different particle sizes P. Each

(T, P) combination is tested over 100 realizations.

For simplification, I assume $\sigma_\epsilon^2 = 1.0$ to be known and only estimate σ_η^2 via maximum likelihood estimation (MLE). I use box-constrained optimization with σ_η^2 being bounded in $[0.1, 5.0]$. The IS particle filter uses particles generated with $\tilde{\sigma}_\eta^2 = 1.0$.

T	Kalman	CSIR				IS			
		P20	P50	P200	P500	P20	P50	P200	P500
50	-0.053	0.311	0.133	-0.014	-0.033	-0.382	-0.342	-0.309	-0.294
100	-0.103	0.381	0.075	-0.052	-0.075	-0.376	-0.360	-0.313	-0.299
250	-0.089	0.421	0.125	-0.031	-0.064	-0.382	-0.333	-0.323	-0.306
500	-0.085	0.388	0.166	-0.032	-0.063	-0.389	-0.348	-0.338	-0.323

Table 4.2: Monte Carlo bias summary

Table 4.2 shows the bias results over 100 Monte Carlo simulations. First, note that the bias reduces as the number of particles P increases. The effect is a lot less pronounced for the importance sampling filter which in all scenarios underestimates the true parameter by about $0.4 - 0.3$. The bias is lowest for the CSIR filter with large Ps and the Kalman filter. The number of observations T has no apparent affect on the bias.

T	Kalman	CSIR				IS			
		P20	P50	P200	P500	P20	P50	P200	P500
50	0.057	0.084	0.068	0.063	0.060	0.021	0.025	0.028	0.033
100	0.036	0.062	0.046	0.038	0.038	0.015	0.016	0.018	0.019
250	0.023	0.042	0.031	0.027	0.024	0.008	0.009	0.009	0.011
500	0.017	0.032	0.027	0.019	0.018	0.006	0.007	0.006	0.008

Table 4.3: Monte Carlo standard error summary

Table 4.3 summarizes the standard error results over 100 Monte Carlo simulations. The standard error steadily decreases as the number of observations T increases. In the case of the CSIR filter, the standard error furthermore decreases as P is increased. The result for $P = 500$ closely matches the standard errors of the Kalman filter. In contrast, the number of particles P seems to have no affect on the standard error of the importance sampling (IS) filter. Interestingly, its standard errors are significantly lower than the Kalman filter standard errors.

Table 4.4 shows the mean squared error (MSE) results over 100 Monte Carlo simulations. The MSE is defined as the sum of the squared bias and squared standard error and therefore combines the effect of the observation size T and the number of particles P on

T	Kalman	CSIR				IS			
		P20	P50	P200	P500	P20	P50	P200	P500
50	0.322	0.798	0.477	0.394	0.353	0.190	0.177	0.172	0.191
100	0.141	0.532	0.217	0.145	0.151	0.162	0.155	0.129	0.125
250	0.059	0.351	0.112	0.071	0.063	0.153	0.119	0.112	0.106
500	0.035	0.251	0.100	0.036	0.036	0.156	0.126	0.117	0.112

Table 4.4: Monte Carlo MSE summary

the result. Observe that the MSE decreases as T grows. The MSE also tends to decrease as the number of particles P gets increased. The Kalman filter produced some unexpected high MSE values for low T values (50 an 100) but generally provides a lower bound to other particle filters for larger Ts. The CSIR results for $P = 500$ closely match the results of the Kalman filter. Interestingly, the IS particle filter produced the lowest MSE for small observation sizes (50 and 100).

The results are plotted in *Figure 4.1*, comparing the Kalman filter with the CSIR and importance sampling particle filter, both with $P = 500$ particles. The results with smaller particle sizes are also plotted (faded).

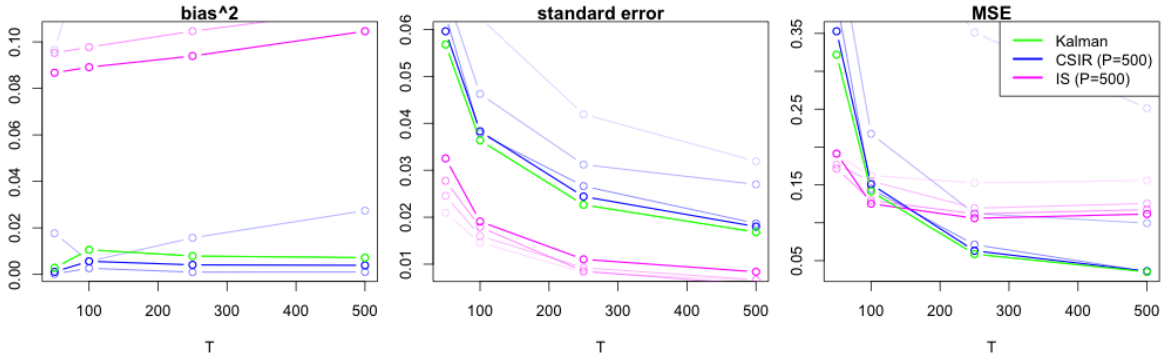

Figure 4.1: Monte Carlo results

The results of the Monte Carlo simulations support theoretic results which show that the Kalman filter minimizes the mean squared error of the estimated parameters. Thus, the Kalman filter is the preferred choice for linear Gaussian state space models. Particle filtering, on the other hand, suffers from propagation error, i.e. current approximations errors are related to past approximation errors. Thus, the number of particles P should be increased with the sample size T to guarantee a good degree of approximation.

5 Illustration

5.1 Trivariate Local Level Model

First, we test the latent state and parameter inference with different filters on a trivariate local level model with restricted covariance matrices to reduce the number of parameters.

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 $\boldsymbol{x}_1 \sim N(\boldsymbol{\mu}_1, \Sigma_1)$ 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 $\boldsymbol{\mu}_1$ and Σ_1 are assumed to be known.

Model Densities

The transition and measurement density of the trivariate local level from above can be easily derived from the model formulation:

$$\begin{aligned} p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{\theta}) &\sim N(\boldsymbol{x}_t, \sigma_\epsilon^2 I_3) \\ p(\boldsymbol{y}_t|\boldsymbol{x}_t, \boldsymbol{\theta}) &\sim N(\boldsymbol{x}_t, \Sigma_\eta) \end{aligned}$$

Realization

Figure 5.1 plots the states and observations for a realization of the trivariate local level model with length $T = 100$. The model parameters are

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

The initial state x_1 is drawn from a standard normal.

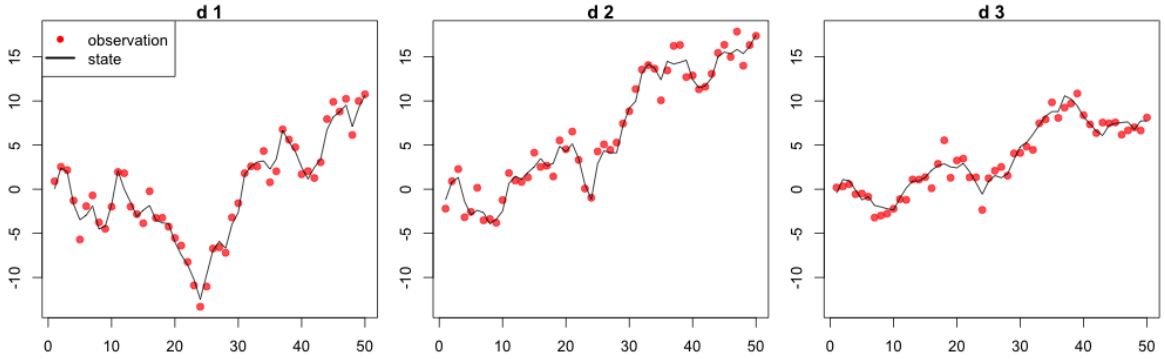

Figure 5.1: Realization of the model with $T = 50$

Latent State Inference

We estimate the latent states in each dimension with both the Kalman filter and the sequential importance resampling (SIR) method and plot the result in *Figure 5.2* and *Figure 5.3*, respectively. Note that the estimates of the two filters are almost identical because $P = 200$ was is sufficiently large for the number of observations $T = 50$.

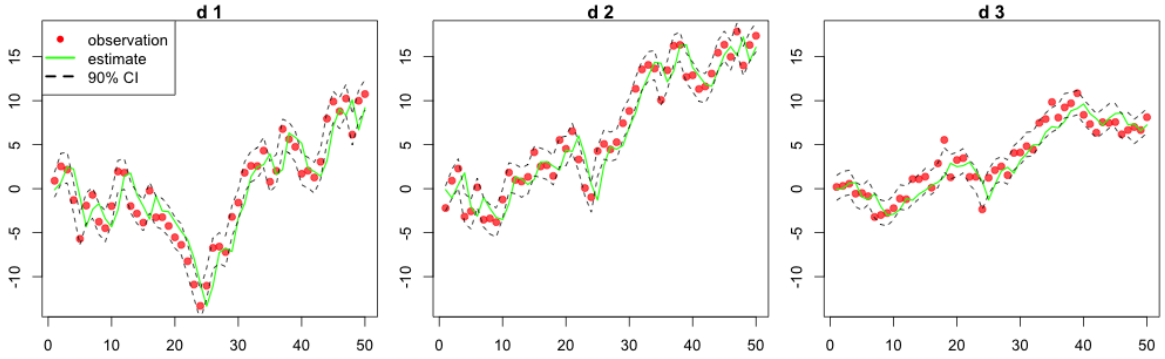

Figure 5.2: State estimates with Kalman filter

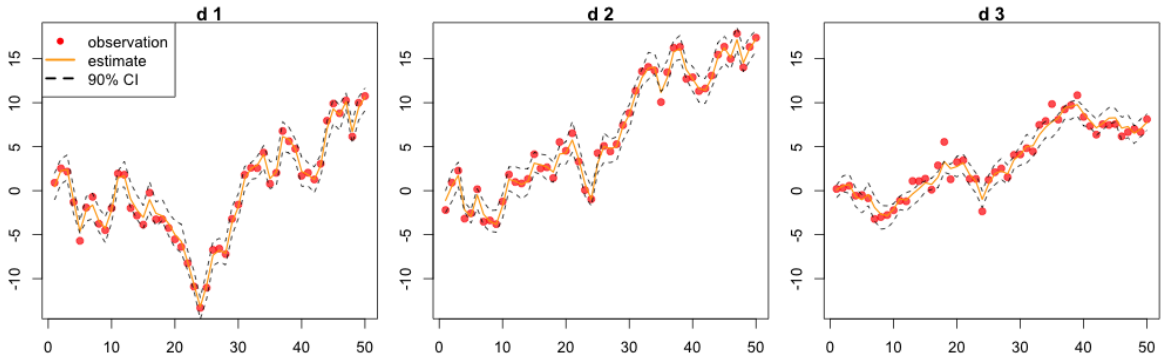

Figure 5.3: State estimates with SIR filter ($P = 200$)

Parameter Inference

Finally, we estimate the model parameters of the realization using maximum likelihood estimation with different filters. In particular, we consider the Kalman filter, the sequential importance resampling (SIR) method, and the novel importance sampling (IS) particle filter. We use box-constrained optimization with the variances $\sigma^2_{\eta 1}$, $\sigma^2_{\eta 2}$, and $\sigma^2_{\eta 3}$ bounded by $[0.1, 5.0]$ and ρ constrained to lie between $[-1.0, 1.0]$. We assume σ^2_ϵ to be fixed at 1.0. The result is summarized in *Table 5.1*.

	$\sigma^2_{\eta 1}$	$\sigma^2_{\eta 2}$	$\sigma^2_{\eta 3}$	ρ	true $\log \mathcal{L}$	MLE $\log \mathcal{L}$
True	4.20	2.80	0.90	0.70		
Kalman	4.96	3.10	1.01	0.73	-307.712	-307.459
SIR	2.46	2.32	1.15	0.47	-313.466	-339.068
IS	2.69	2.09	1.06	0.42		

Table 5.1: MLE parameters with different filters

Both the SIR and the IS particle filter have been tested with a rather small number of only $P = 50$ particles. The reason is that both algorithms rely on observing the multivariate normal density of each particle at each iteration which can considerably slow down the algorithms if P is large.

Note that the Kalman filter produced the best parameter estimates. The IS particle filter also gave reasonable estimates while the SIR estimates are far off the true parameter values. The reason for this bad performance is the lack of smoothness of the log-likelihood with respect to the parameters as discussed earlier which makes the SIR algorithm an inappropriate filter method for parameter inference. The argument is backed by the log-likelihood plots in *Figure 5.4* with respect to $\sigma^2_{\eta 1}$ and ρ. While the SIR plot with respect

to ρ still looks reasonable, the plot for $\sigma^2_{\eta 1}$ does not exhibit a concave shape.

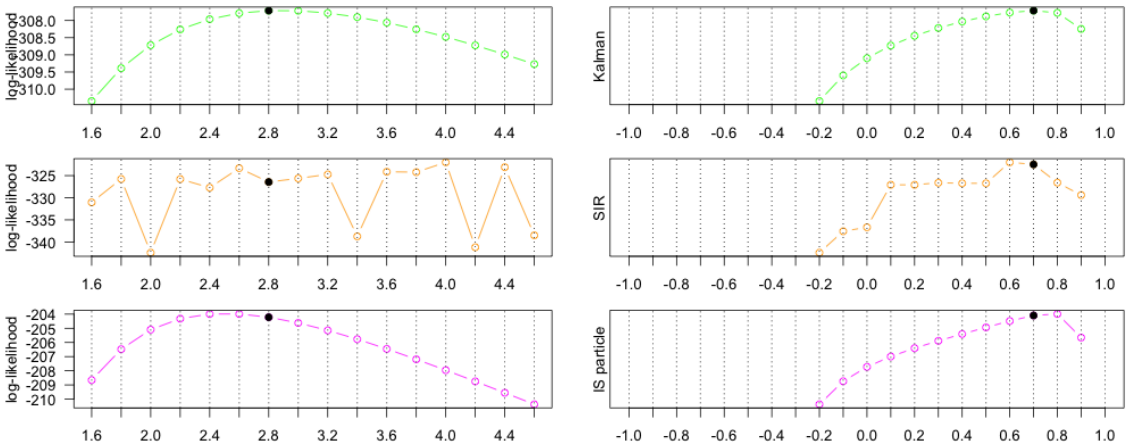

Figure 5.4: Log-likelihood as a function of $\sigma^2_{\eta 1}$ and ρ with different filters

5.2 Hierarchical Dynamic Poisson Model

Next, we test the particle filters on a non-linear model with non-Gaussian observations, called *hierarchical dynamic Poisson model*. The observations are modeled as a Poisson random variable and represent counts, i.e. the total number of occurrences during a fixed time period. The underlying latent state of an observation is modeled as the the average number of counts in that interval (i.e. the state is the parameter of the Poisson distribution). The state is assumed to be constructed from 3 components: a daily level, intra-daily periodic changes, and intra-daily fluctuations. Both the daily level and intra-daily fluctuations are are assumed to form an (independent) Markov chain. The model in this form might be well suited to model demand in various settings such as server access or patients in a hospital.

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.

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

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

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

The initial daily and intra-daily component is drawn from a normal with mean μ_1 and covariance Σ_1:

$$\log \lambda_1^{(D)}, \log \lambda_1^{(I)} \sim N(\mu_1, \Sigma_1)$$

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:

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

Again, the initial state parameters μ_1 and Σ_1 are assumed to be known.

Model Densities

The transition and measurement density of the hierarchical dynamic Poisson model can be described as a function of the state components:

$$\begin{aligned} p(\log \lambda_{m,n+1} | \log \lambda_m^{(D)}, \log \lambda_{m,n}^{(I)}, \log \lambda_n^{(P)}, \boldsymbol{\theta}) &\sim N(\mu_{m,n}, \sigma_{m,n}^2) \\ p(y_{m,n} | \log \lambda_m^{(D)}, \log \lambda_{m,n}^{(I)}, \log \lambda_n^{(P)}, \boldsymbol{\theta}) &\sim \text{Poisson}(\lambda_m^{(D)} \lambda_{m,n}^{(I)} \lambda_n^{(P)}) \end{aligned}$$

where

$$\begin{aligned} \mu_{m,n} &= \phi_0^{(D)} + \phi_1^{(D)} \log \lambda_m^{(D)} + \phi_1^{(I)} \log \lambda_{m,n}^{(I)} + \phi_1^{(P)} \sin(\pi(n-1)/M) \\ \sigma_{m,n}^2 &= \sigma_{(D)}^2 + \sigma_{(I)}^2 \end{aligned}$$

The transition and measurement density are used in the implementation of the SIR algorithm and the importance sampling particle filter.

Realization

The left graph in *Figure 5.5* 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

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

The initial daily and intra-daily state components were drawn from a standard normal. The right graph in *Figure 5.5* plots the state components $\log \lambda_m^{(D)}$, $\log \lambda_{m,n}^{(I)}$, and $\log \lambda_n^{(P)}$ of that realization.

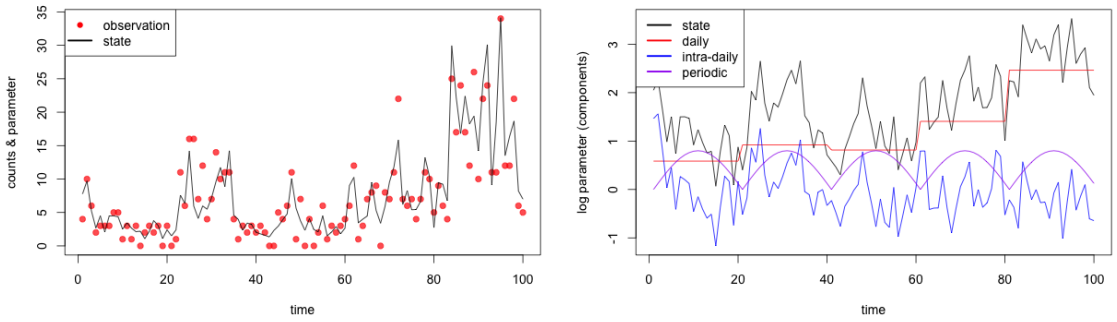

Figure 5.5: Realization of the model with $N = 5$ and $M = 20$

Latent State Inference

We estimate the latent states of the realization in *Figure 5.5* with the implementation of the sequential importance resampling (SIR) algorithm which uses the transition and measurement densities we derived before. The estimates of the latent states, together with the 90% confidence interval (estimated by the 5% and 95% quantile of the $P = 200$ particles) are plotted in *Figure 5.6*.

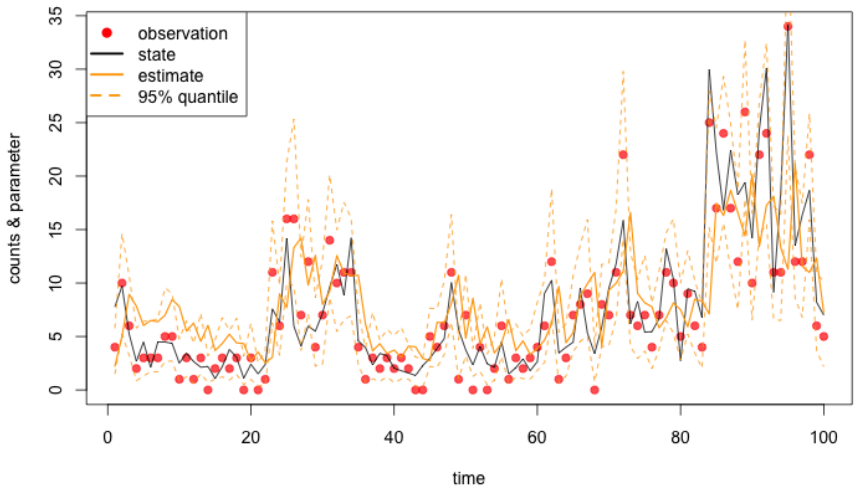

Figure 5.6: State estimates with SIR filter ($P = 200$)

Note that neither the Kalman filter nor the continuous SIR method can be used for this model. The Kalman filter cannot be used because the model is non-linear (and the observations are not Gaussian). The continuous SIR particle filter is not applicable because the state is a multivariate variable, consisting of the daily, intra-daily, and periodic component.

Parameter Inference

Finally, we estimate the model parameters of the realization using maximum likelihood estimation with different filters. In particular, we compare the sequential importance resampling (SIR) method with the novel importance sampling (IS) particle filter. We use box-constrained optimization with the parameters $\phi_0^{(D)}$, $\phi_1^{(D)}$, $\phi_1^{(I)}$, and $\phi_1^{(P)}$ bounded between $[0, 1]$ and the variances $\sigma^2_{(D)}$ and $\sigma^2_{(I)}$ constrained to lie between $[0.1, 5.0]$. The auxiliary parameter vector used for the IS particle filter is

$$\tilde{\boldsymbol{\theta}} = [\tilde{\phi}_0^{(D)} = 0.5, \tilde{\phi}_1^{(D)} = 0.5, \tilde{\sigma}^2_{(D)} = 1.0, \tilde{\phi}_1^{(I)} = 0.5, \tilde{\sigma}^2_{(I)} = 1.0, \tilde{\phi}_1^{(P)} = 1.0]^T$$

The result is summarized in *Table 5.2*.

	$\phi_0^{(D)}$	$\phi_1^{(D)}$	$\sigma^2_{(D)}$	$\phi_1^{(I)}$	$\sigma^2_{(I)}$	$\phi_1^{(P)}$	true $\log \mathcal{L}$	MLE $\log \mathcal{L}$
True	0.70	0.60	0.30	0.80	0.60	0.20		
SIR	0.76	0.56	0.85	0.48	0.83	1.13	-296.663	-310.851
IS	0.65	0.59	0.40	0.63	0.35	0.31	-273.295	-287.333

Table 5.2: MLE parameters with different filters

The table compares the estimated parameters with both methods to the true parameters. Note that the estimates of the IS particle filter are indeed closer to the true parameters. The lack of smoothness of the log-likelihood with respect to the parameters makes the SIR method likely to get stuck in a local optimum. The difference in the log-likelihood between the SIR algorithm and the IS particle filter is illustrated with respect to parameters $\phi_1^{(D)}$ and $\sigma^2_{(D)}$ in *Figure 5.7* and *Figure 5.8*, respectively. In both cases, the shape of the log-likelihood looks very similar when considering larger intervals of the parameter. But when we zoom in, we observe that the log-likelihood for the SIR method lacks smoothness.

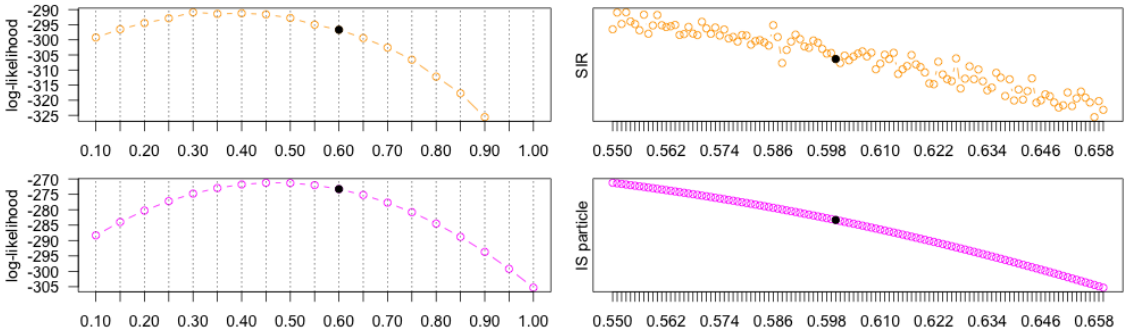

Figure 5.7: Log-likelihood as a function of $\phi_1^{(D)}$ with different filters

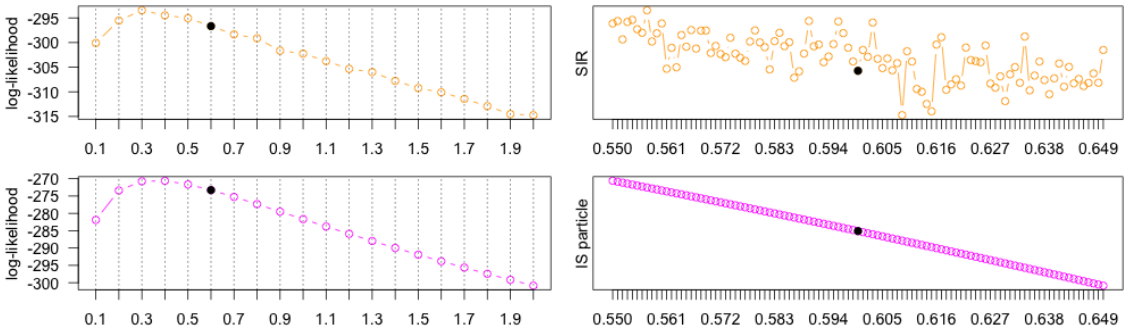

Figure 5.8: Log-likelihood as a function of $\sigma^2_{(D)}$ with different filters

6 Conclusion

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.

[5] S. Malik and M. K. Pitt. Particle filters for continuous likelihood evaluation and maximisation. *Journal of Econometrics*, 165(2):190–209, 2011.