Sequential Monte Carlo Methods in Practice

Eivind Lamo
with supervisor
Andreas Størksen Stordal

A thesis presented for the degree of
Master in Science
Statistics and Actuarial Science

Department of Mathematics
University of Bergen
Norway
November 2023

Abstract

With the continuous increase in computational power, sequential Monte Carlo methods have emerged as an efficient technique for estimating unknown data in a world consisting of nonlinearity and non-Gaussianity. In this thesis, we are building a theoretical foundation by the help of Bayesian statistics, that can be applied to numerous real-world problems. We are interested in solving the problem of estimating an unknown signal process given certain observations, where both processes are modelled as Markovian, nonlinear, non-Gaussian state-space models. In particular, we will try to estimate the unobserved volatility dynamics for the S&P 500 index using observed returns and a slight modification of Heston's stochastic volatility model. This will be done by the sequential importance resampling filter, which we will also combine with Markov chain Monte Carlo for parameter estimation. Our overall goal is to propose another alternative to Heston's model, by investigating how well the model responds to measuring volatility when including data from the financial crisis of 2007-2008.

Keywords: Bayesian inference; sequential Monte Carlo; volatility filtering; financial econometrics; particle Markov chain Monte Carlo

Acknowledgements

I first want to thank my supervisor Andreas Størksen Stordal, for accepting my request to supervise, helping me understand the whole concept making up this thesis, always being available for e-mail discussion regarding my buggy code and suggesting research articles outside his field.
I would also like to thank the Department of Mathematics, and particularly Associate Professor Yushu Li, for always taking her time when I would drop by her office during my time as a master's student, whether I had questions regarding courses or whatnot.
Additionally I want to thank my fellow students at the department making my time as a student enjoyable.
I want to thank my family for assisting me in the time after my surgery, making it possible for me to hand in this thesis.
Lastly, I especially want to thank my dear Evin for always visiting, supporting and not making me go mental whether I was stuck in the hospital or at home.

Eivind Lamo
Bergen, November 2023

Contents

List of Algorithms

Notation and Terminology

$\mathbb{R}^d$ – the d-dimensional Euclidean space.

$\mathcal{B}(\mathbb{R}^d)$ – the σ-algebra of Borel subsets of $\mathbb{R}^d$.

$B(\mathbb{R}^d)$ – the set of bounded $\mathcal{B}(\mathbb{R}^d)$-measurable functions defined on $\mathbb{R}^d$.

$C_b(\mathbb{R}^d)$ – the set of bounded continuous functions defined on $\mathbb{R}^d$.

$C_k(\mathbb{R}^d)$ – the set of compactly supported continuous functions defined on $\mathbb{R}^d$.

$\mathcal{M}_F(\mathbb{R}^d)$ – the set of finite measures over $\mathcal{B}(\mathbb{R}^d)$.

$\mathcal{P}(\mathbb{R}^d)$ – the set of probability measures over $\mathcal{B}(\mathbb{R}^d)$.

Markov Chains and Transition Kernels

Let $(\Omega, \mathcal{F}, P)$ be a probability space and $X = \{X_t\}_{t\in\mathbb{N}}$ be a stochastic process defined on said probability space, with values in $\mathbb{R}^{n_x}$. Let $\mathcal{F}_t^X$ be the σ-algebra generated by the process, i.e., $\mathcal{F}_t^X \triangleq \sigma(X_s, s \in [0,t])$. Then X is a *Markov chain* if, for all $t \in \mathbb{N}$ and $A \in \mathcal{B}(\mathbb{R}^{n_x})$

$$P(X_{t+1} \in A|\mathcal{F}_t^X) = P(X_{t+1} \in A|X_t). \tag{1}$$

The transition kernel of the Markov chain is the function $K_t(\cdot,\cdot)$ defined on $\mathbb{R}^{n_x} \times \mathcal{B}(\mathbb{R}^{n_x})$ such that, for all $t \in \mathbb{N}$ and $x \in \mathbb{R}^{n_x}$

$$K_t(x, A) = P(X_{t+1} \in A|X_t = x), \tag{2}$$

where K_t has the following properties:

- $K_t(x,\cdot)$ is a probability measure on $\mathbb{R}^{n_x}, \forall t \in \mathbb{N}$ and $x \in \mathbb{R}^{n_x}$.
- $K_t(\cdot, A) \in B(\mathbb{R}^{n_x}), \forall t \in \mathbb{N}$ and $A \in \mathcal{B}(\mathbb{R}^{n_x})$.

X has a distribution determined uniquely by its initial distribution and transition kernel, and we then denote q_t as the distribution of the random variable X_t, with,

$$q_t(A) \triangleq P(X_t \in A). \tag{3}$$

From (2) we can then show that q_t satisfies the recurrence formula $q_{t+1} = q_t K_t$, where $q_t K_t$ is the measure defined as

$$(q_t K_t)(A) \triangleq \int_{\mathbb{R}^{n_x}} K_t(x, A) q_t(dx). \tag{4}$$

We say that the transition kernel K_t satisfies the Feller property if, for all $t > 0$, the function $K_t f : \mathbb{R}^{n_x} \to \mathbb{R}$ defined as

$$K_t f(x) \triangleq \int_{\mathbb{R}^{n_x}} f(y) K_t(x, dy) \tag{5}$$

is continuous for every $f \in \mathcal{C}_b(\mathbb{R}^d)$. If K_t has the Feller property, then we have $K_t f \in \mathcal{C}_b(\mathbb{R}^d)$ for all $f \in \mathcal{C}_b(\mathbb{R}^d)$.

1 Introduction

Since all the way back in the 1940's, when Polish-American physicist Stanisław Ulam was working on nuclear weapon projects at the Los Alamos National Laboratory, Monte Carlo (MC) methods have been broadly used to estimate uncertainty that could not be explained or analytically evaluated by deterministic mathematical models. The use of MC methods have since then been highly useful, especially within optimization, numerical integration and sampling from probability distributions. These types of methods rely on repeated random sampling to obtain numerical results, in cases where an analytical solution might not be possible. In more recent time, there has been a rise of sequential Monte Carlo (SMC) methods, also commonly referred to as *particle filters*, along with Bayesian inference. SMC methods allow for sequential updating of the probabilities as more data become available. When these ideas were first introduced in the 1950's, they were largely overlooked and ignored, as the computational resources at the time were quite modest. These algorithms also had shortcomings, as particles would degenerate over time leading to poor particle diversity. Since Gordon et al. published their work introducing an SMC resampling algorithm in 1993, SMC methods have seen a dramatic increase in research activity. The *bootstrap filter* allowed for sequential updating of the probabilities followed by a resampling step, solving the degeneracy problem[1]. With computers continuously becoming faster and faster, SMC methods have experienced more and more real-life applications.

This thesis is meant to have an intuitive approach, by having a form that allows the reader to continuously immerse him or herself in the material. This is done by the use of the following structure. In Chapter 2 the problem at hand is introduced and formulated, and the model it is based on is specified. We also present some optimal solutions. In Chapter 3 we proceed by introducing the methods and the theoretical framework that can be applied for estimating the given problem. They are introduced in a manner that addresses the limitations of the preceding one. Several simple examples are provided along the way to assist the reader to better understand the theoretical concepts. In Chapter 4 we conduct a benchmark experiment, where we apply the main particle filter in this thesis to generated data by using a basic model. Moving on from here, we apply the method to real-life financial data, where the filtered values are unobserved. We also perform parameter estimation here, by combining two of our introduced methods. Finally, we discuss our results in Chapter 6 and conclude in Chapter 7. The main objective for this thesis, is to establish a well-specified volatility dynamic for further use in financial econometrics.

1.1 Introduction to Bayesian Filtering

When performing real-life data analysis, estimating unknown quantities given some observations is key. In these situations, we usually have some prior knowledge about what we are modelling. From this prior knowledge we can formulate Bayesian models. This means formulating a prior distribution for the unknown quantities and likelihood functions which relates these to our observations. The essence of it, is that we want to use randomness to better explain and analyze hidden states and underlying processes, using some relation between the observations and the underlying process. All inference performed on the unknown quantities stems from the well-known Bayes' theorem, given by

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}. \tag{6}$$

As we have already mentioned SMC methods, we will see how these can be useful in updating the posterior distribution as we get more data available, and thus we can perform inference on-line, sequentially. We will see how we iteratively update our estimates, based on both new observations and the prior knowledge. Common examples of this in real-life, are tracking an aircraft using radar measurements or estimating the volatility of financial instruments using stock market data. If we have data modelled by a linear Gaussian state-space model, we can apply the well-known *Kalman filter*, which allows us to derive an exact analytical solution. If data is modelled as a partially observed, finite state-space Markov chain, we can obtain an analytical solution by the *hidden Markov model* (HMM) *filter*. Unfortunately, our problems can often be too complex to use these methods, making them inefficient. Common problems with real data involve non-Gaussianity, high dimensionality and nonlinearity. Therefore, we can use SMC methods, which are very flexible, easy to implement and in general more applicable. Due to this, SMC methods are broadly used today, as we are not dependent on linearity or normal assumptions required by e.g., the Kalman filter. We are focusing here on how SMC can be applied in finance, more specifically how they can be used to estimate financial data that cannot be directly observed, by using observable data. The goal however remains the same, as we wish to estimate and predict uncertainty as precise as possible.

2 Problem and Panel Model

Our theoretical framework in this chapter and the next is build on the work by Doucet et al. (2001)[2] as well as Ristic et al. (2004)[3]. We are following the notation used in [2]. We begin by looking at two processes, where we have $[\{\mathbf{x}_t\}, \{\mathbf{y}_t\}]_{t\in\mathbb{N}}$ with $\mathbf{x}_t \in \mathcal{X}$ and $\mathbf{y}_t \in \mathcal{Y}$. Here $\mathbf{x}_t$ is our unobserved signal at time t with initial distribution $p(\mathbf{x}_0)$ and transition equation $p(\mathbf{x}_t|\mathbf{x}_{t-1})$, modelled as a *Markov process*. Our observed data is $\mathbf{y}_t$, with $\mathbf{y}_t$ being assumed conditionally independent given our unobserved process, giving it the marginal distribution $p(\mathbf{y}_t|\mathbf{x}_t)$. For simplicity, we restrict ourselves here to signals modelled as Markovian, nonlinear, non-Gaussian state-space models. Even though SMC can be applied to more general settings, this thesis will focus on the aforementioned case. Here $p(\cdot)$ denotes the probability function, with $p(\mathbf{x}_t)$ denoting discrete distributions and $p(d\mathbf{x}_t)$ continuous distributions. We can summarize our model in the following way

$$p(\mathbf{x}_0) \triangleq p(\mathbf{x}_0|\mathbf{y}_0), \tag{7}$$

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}), \text{ for } t \geq 1, \tag{8}$$

$$p(\mathbf{y}_t|\mathbf{x}_t), \text{ for } t \geq 1. \tag{9}$$

We then denote $\mathbf{x}_{0:t} \triangleq \{\mathbf{x}_0, ..., \mathbf{x}_t\}$ and $\mathbf{y}_{1:t} \triangleq \{\mathbf{y}_1, ..., \mathbf{y}_t\}$ respectively as the signal and the observations up to time t. Moreover, we wish to recursively estimate the *posterior distribution* $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$ and its associated features in time, including the marginal distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ and the expectations

$$I(f_t) = \mathbb{E}_{p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})}[f_t(\mathbf{x}_{0:t})] \triangleq \int f_t(\mathbf{x}_{0:t})p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})d\mathbf{x}_{0:t}. \tag{10}$$

The posterior density is then given by Bayes' theorem as in (6), at any time point t as

$$p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) = \frac{p(\mathbf{y}_{1:t}|\mathbf{x}_{0:t})p(\mathbf{x}_{0:t})}{\int p(\mathbf{y}_{1:t}|\mathbf{x}_{0:t})p(\mathbf{x}_{0:t})d\mathbf{x}_{0:t}}, \tag{11}$$

which we can use to obtain a recursive formula for the joint distribution by

$$p(\mathbf{x}_{0:t+1}|\mathbf{y}_{1:t+1}) = p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})\frac{p(\mathbf{y}_{t+1}|\mathbf{x}_{t+1})p(\mathbf{x}_{t+1}|\mathbf{x}_t)}{p(\mathbf{y}_{t+1}|\mathbf{y}_{1:t})}. \tag{12}$$

The marginal distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ also satisfies the following recursion.

$$\textit{Prediction: } p(\mathbf{x}_t|\mathbf{y}_{1:t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})d\mathbf{x}_{t-1}, \tag{13}$$

$$\textit{Updating: } p(\mathbf{x}_t|\mathbf{y}_{1:t}) = \frac{p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1})}{\int p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1})d\mathbf{x}_t}. \tag{14}$$

These equations may be problematic due to the calculation of complex, high-dimensional integrals, making them inefficient. Therefore, the recursive propagation of the posterior given in (13) and (14) is only a conceptual solution.

Implementing this would require the storage of the entire (non-Gaussian) density, which generally would be equivalent to an infinite dimensional vector. In some cases, as mentioned earlier, it *is* possible to obtain optimal algorithms for recursive Bayesian state estimation, as described in the next section.

2.1 Optimal Algorithms

For our panel model above, we can formulate optimal, finite-dimensional algorithms for estimating the states by recursive Bayesian estimation in the following cases:

1. In a linear-Gaussian case, the functional recursion of (13) and (14) becomes the Kalman filter.
2. If we have a discrete-valued state space, with a finite number of states, we can apply grid-based methods.
3. It is also possible to apply exact analytic solutions for certain subclasses of nonlinear problems, as discovered by Beneš[4] and Daum[5; 6].

We now give a brief explanation of two of these algorithms. For more on Beneš and Daum filters see [3].

2.1.1 The Kalman Filter

The Kalman filter is a robust tool employed for estimating and predicting system states amid uncertainties, playing a pivotal role in various applications, including target tracking, navigation, and control. When applying the Kalman filter, we assume that the posterior density is Gaussian at every time step, and therefore exactly and completely described by its mean and covariance. We have that if $p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})$ is Gaussian, it can be proven that $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ is also Gaussian, given the following assumptions:

- $\mathbf{v}_{t-1}$ and $\mathbf{w}_t$ are drawn from *Gaussian* densities of known parameters
- $\mathbf{f}_{t-1}(\mathbf{x}_{t-1}, \mathbf{v}_{t-1})$ is a known *linear* function of $\mathbf{x}_{t-1}$ and $\mathbf{v}_{t-1}$
- $\mathbf{h}_t(\mathbf{x}_t, \mathbf{w}_t)$ is a known *linear* function of $\mathbf{x}_t$ and $\mathbf{w}_t$.

A typical model satisfying these assumptions is

$$\mathbf{x}_t = \mathbf{F}_{t-1}\mathbf{x}_{t-1} + \mathbf{v}_{t-1}, \tag{15}$$

$$\mathbf{y}_t = \mathbf{H}_t\mathbf{x}_t + \mathbf{w}_t, \tag{16}$$

where $\mathbf{F}_{t-1}$ and $\mathbf{H}_t$ are known matrices defining the linear functions. The random Gaussian and mutually independent sequences $\mathbf{v}_{t-1}$ and $\mathbf{w}_t$ have covariances $\mathbf{Q}_{t-1}$ and $\mathbf{R}_t$ respectively.

The Kalman filter algorithm is then derived as a *generalized least squares* (GLS) method on the previous data with help from (13) and (14), and can be viewed as the following recursive relationship:

$$p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1}) = N(\mathbf{x}_{t-1}; \hat{\mathbf{x}}_{t-1|t-1}, \mathbf{P}_{t-1|t-1}) \tag{17}$$

$$p(\mathbf{x}_t|\mathbf{y}_{1:t-1}) = N(\mathbf{x}_t; \hat{\mathbf{x}}_{t|t-1}, \mathbf{P}_{t|t-1}) \tag{18}$$

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) = N(\mathbf{x}_t; \hat{\mathbf{x}}_{t|t}, \mathbf{P}_{t|t}) \tag{19}$$

where $N(\mathbf{x}; \boldsymbol{\mu}, \mathbf{P})$ is a Gaussian density with argument $\mathbf{x}$, mean $\boldsymbol{\mu}$ and covariance $\mathbf{P}$, meaning

$$N(\mathbf{x}; \boldsymbol{\mu}, \mathbf{P}) \triangleq \frac{1}{\sqrt{2\pi\mathbf{P}}} \exp\left\{ -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \mathbf{P}^{-1} (\mathbf{x}-\boldsymbol{\mu}) \right\}. \tag{20}$$

With $\mathbf{M}^T$ denoting the transpose of matrix $\mathbf{M}$, the means and covariances of the Kalman filter are computed as follows:

$$\hat{\mathbf{x}}_{t|t-1} = \mathbf{F}_{t-1}\hat{\mathbf{x}}_{t-1|t-1} \tag{21}$$

$$\mathbf{P}_{t|t-1} = \mathbf{Q}_{t-1} + \mathbf{F}_{t-1}\mathbf{P}_{t-1|t-1}\mathbf{F}_{t-1}^T \tag{22}$$

$$\hat{\mathbf{x}}_{t|t} = \hat{\mathbf{x}}_{t|t-1} + \mathbf{K}_t(\mathbf{y}_t - \mathbf{H}_t\hat{\mathbf{x}}_{t|t-1}) \tag{23}$$

$$\mathbf{P}_{t|t} = \mathbf{P}_{t|t-1} - \mathbf{K}_t\mathbf{S}_t\mathbf{K}_t^T \tag{24}$$

with

$$\mathbf{S}_t = \mathbf{H}_t\mathbf{P}_{t|t-1}\mathbf{H}_t^T + \mathbf{R}_t \tag{25}$$

being the covariance of the innovation term $\nu_t = \mathbf{y}_t - \mathbf{H}_t\hat{\mathbf{x}}_{t|t-1}$ and

$$\mathbf{K}_t = \mathbf{P}_{t|t-1}\mathbf{H}_t^T\mathbf{S}_t^{-1} \tag{26}$$

being the Kalman gain.
The Kalman filter thus recursively computes the mean and covariance of the posterior $p(\mathbf{x}_t|\mathbf{y}_{1:t})$. Given that the aforementioned assumptions hold, the optimal solution to the tracking problem is provided, and the implication is then that no other algorithm should be able to perform better in this linear Gaussian environment by the GLS method.

2.1.2 Grid-Based Methods

If we have a state space that is discrete and finite, grid-based methods provide the optimal recursion of the filtered density $p(\mathbf{x}_t|\mathbf{y}_{1:t})$. If we suppose that the state space at time $t-1$ consists of the discrete states $\{\mathbf{x}_{t-1}^{(i)}\}_{i=1}^{N}$ then for each state $\mathbf{x}_{t-1}^{(i)}$, we can let the conditional probability of that state be denoted $w_{t-1|t-1}^{(i)}$ given that we have the measurements up to time $t-1$. In other words, we have that $\mathrm{P}\{\mathbf{x}_{t-1} = \mathbf{x}_{t-1}^{(i)}|\mathbf{y}_{1:t-1}\} \triangleq w_{t-1|t-1}^{(i)}$. This yields the posterior density at $t-1$ as

$$p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1}) = \sum_{i=1}^{N} w_{t-1|t-1}^{(i)} \delta(\mathbf{x}_{t-1} - \mathbf{x}_{t-1}^{(i)}). \tag{27}$$

Here $\delta(\cdot)$ is the Dirac delta measure, defined as $\delta(\mathbf{x}-\mathbf{a}) = 0$ for $\mathbf{x} \neq \mathbf{a}$, and also $\int_{-\infty}^{\infty} \delta(\mathbf{x}-\mathbf{a})d\mathbf{x} = 1$. We can now substitute (27) into (13) and (14), which yields the prediction and update equations:

$$p(\mathbf{x}_t|\mathbf{y}_{1:t-1}) = \sum_{i=1}^{N} w_{t|t-1}^{(i)} \delta(\mathbf{x}_t - \mathbf{x}_t^{(i)}) \tag{28}$$

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) = \sum_{i=1}^{N} w_{t|t}^{(i)} \delta(\mathbf{x}_t - \mathbf{x}_t^{(i)}) \tag{29}$$

where

$$w_{t|t-1}^{(i)} \triangleq \sum_{j=1}^{N} w_{t-1|t-1}^{(j)} p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(j)}), \tag{30}$$

$$w_{t|t}^{(i)} \triangleq \frac{w_{t|t-1}^{(i)} p(\mathbf{y}_t|\mathbf{x}_t^{(i)})}{\sum_{j=1}^{N} w_{t|t-1}^{(j)} p(\mathbf{y}_t|\mathbf{x}_t^{(j)})}. \tag{31}$$

Here it is assumed that the transitional densities $p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(j)})$ and the likelihood functions $p(\mathbf{y}_t|\mathbf{x}_t^{(i)})$ are known, and again, the assumptions must hold true for this to be an optimal solution.

3 Monte Carlo Methods and Particle Filtering

To investigate the problems defined in Chapter 2, MC methods have been increasingly used, and then particularly MC integration methods. The advantage here, is that these methods are not subject to any linearity or Gaussianity constraints on the model. We will first show how one can approximate integrals given one has a sufficient number of samples from the required posterior distributions, and then explain the challenges regarding these methods. Moreover, we will explain how this has led to further development of other, more advanced methods, such as *importance sampling* (IS) and *sequential importance sampling* (SIS), as well as the *sequential importance resampling filter* (SIR). With the increase of computational power, these methods have been increasingly used in recent years.

3.1 Perfect Monte Carlo Sampling

Assuming we are able to simulate N independent and identically distributed (i.i.d.) random samples, or particles, $\{\mathbf{x}_{0:t}^{(i)}\}_{i=1}^{N}$, we can obtain an empirical estimate of the distribution according to (11) by

$$P_N(d\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) = \frac{1}{N}\sum_{i=1}^{N} \delta_{\mathbf{x}_{0:t}^{(i)}}(d\mathbf{x}_{0:t}), \tag{32}$$

where here $\delta_{\mathbf{x}_{0:t}^{(i)}}(d\mathbf{x}_{0:t})$ denotes the delta-Dirac mass located in $\mathbf{x}_{0:t}^{(i)}$. Perfect MC sampling thus refers to generating random samples from a probability distribution exactly, without any approximation. We can now obtain the estimate of $I(f_t)$ by

$$I_N(f_t) = \int f_t(\mathbf{x}_{0:t})P_N(d\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) = \frac{1}{N}\sum_{i=1}^{N} f_t(\mathbf{x}_{0:t}^{(i)}). \tag{33}$$

This provides an unbiased estimate, and from the strong law of large numbers we see that

$$I_N(f_t) \xrightarrow[N\to\infty]{a.s.} I(f_t), \tag{34}$$

with $\xrightarrow{a.s.}$ denoting almost sure convergence. Furthermore, if $\sigma_{f_t}^2 < \infty$ then the central limit theorem (CLT) holds as

$$\sqrt{N}[I_N(f_t) - I(f_t)] \xrightarrow[N\to\infty]{d} N(0, \sigma_{f_t}^2), \tag{35}$$

where $\xrightarrow{d}$ denotes convergence in distribution. The obvious benefit here is that we may easily estimate any quantity $I(f_t)$ from the set of particles, but the challenge is that in practice this is usually impossible to implement, as we cannot sample efficiently from the posterior. This is due to the fact that in the real world the posterior is likely multivariate and non-standard.

Example 1
We now illustrate a simple example of perfect MC sampling by the *inverse transform method*. Let X be a random variable that can be described by its CDF F_X. We now want to generate values of X so that it is distributed according to the distribution of the CDF.
In this example we use the CDF of the exponential distribution with $\lambda = 1$:

$$F(x) = 1 - e^{-x}. \tag{36}$$

We now want to solve $F(F^{-1}(u)) = u$ to perform an inversion, which yields

$$F^{-1}(u) = -(\log(1-u)). \tag{37}$$

Now we generate a random number u from the standard uniform distribution, i.e., $u \sim U(0,1)$, and we can estimate our distribution.

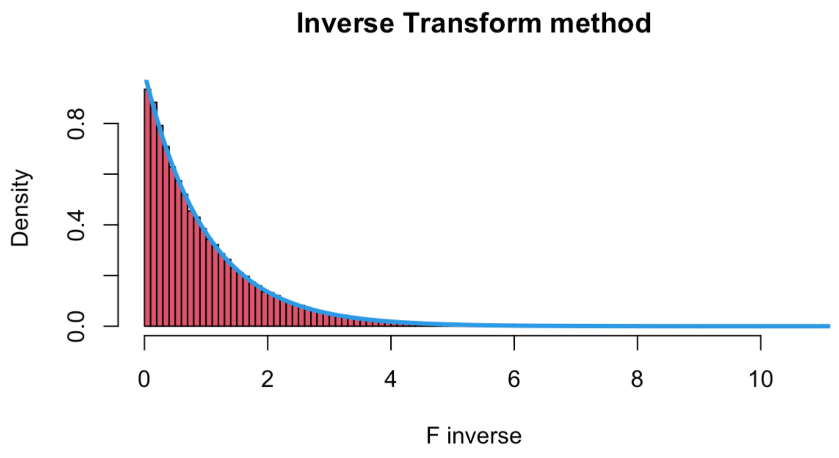

Figure 1: Simulating the exponential distribution with $N = 10^5$ values for u that we plug into $F^{-1}(u)$.

Our histogram here shows our estimation while the blue line is the true density, and we see how well it fits, as we get near perfect estimates. The difficulty as mentioned, is that in real life the inverse CDF is often impossible to calculate.

3.2 Importance Sampling

An alternative solution to the problem in the previous chapter is the importance sampling method[7]. In order to evaluate $I(f_t)$, importance sampling takes advantage of the identity

$$I(f_t) = \frac{\int f_t(\mathbf{x}_{0:t}) w(\mathbf{x}_{0:t}) \pi(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) d\mathbf{x}_{0:t}}{\int w(\mathbf{x}_{0:t}) \pi(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) d\mathbf{x}_{0:t}}, \tag{38}$$

where $\pi(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$ is the *importance sampling distribution*, or *proposal distribution*, and $w(\mathbf{x}_{0:t})$ is known as the *importance weight*,

$$w(\mathbf{x}_{0:t}) = \frac{p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})}{\pi(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})}. \tag{39}$$

This can be written as

$$\mathbb{E}_p[f_t] = \mathbb{E}_\pi\left[\frac{f_t(w_t)}{w_t}\right]. \tag{40}$$

It's necessary that the support of $\pi(\cdot)$ includes the support of $p(\cdot)$. If one now can simulate N i.i.d. particles $\{\mathbf{x}_{0:t}^{(i)}\}_{i=1}^N$ according to $\pi(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$, the MC estimate of $I(f_t)$ becomes

$$\hat{I}_N(f_t) = \frac{\frac{1}{N}\sum_{i=1}^N f_t(\mathbf{x}_{0:t}^{(i)})w(\mathbf{x}_{0:t}^{(i)})}{\frac{1}{N}\sum_{j=1}^N w(\mathbf{x}_{0:t}^{(j)})} = \sum_{i=1}^N f_t(\mathbf{x}_{0:t}^{(i)})\tilde{w}_t^{(i)}, \tag{41}$$

where the *normalized importance weights* $\tilde{w}_t^{(i)}$ are given by

$$\tilde{w}_t^{(i)} = \frac{w(\mathbf{x}_{0:t}^{(i)})}{\sum_{j=1}^N w(\mathbf{x}_{0:t}^{(j)})}. \tag{42}$$

The notation $\tilde{w}_t^{(i)}$ represents here the normalized weights, but in the literature these are sometimes referred to interchangeably. It is important to specify that when introducing resampling later, we *always* resample from the normalized weights.

Example 2
We illustrate IS by a simple one-dimensional example, without the use of particles. Say we want to estimate the following integral

$$\theta = f(x) = \int_0^\infty \frac{x^2}{\sqrt{2\pi}} e^{-x^2/2} dx, \tag{43}$$

with our importance function being

$$\pi(x) = \frac{2}{\sqrt{2\pi}} e^{-x^2/2}, \quad x > 0. \tag{44}$$

We see this is the positive standard normal distribution, thus we can easily sample from it. Let $X \sim N(0,1)$ then $Y = |X|$ follows the positive normal distribution, and we can perform importance sampling by

$$\hat{\theta} = \frac{1}{N}\sum_{i=1}^N \frac{f(\mathbf{y}_i)}{\pi(\mathbf{y}_i)}. \tag{45}$$

We see below how we get good estimates for the integral θ by performing importance sampling.

```
True Value IS Estimate
  0.500000    0.500169
```

Figure 2: IS with $N = 10^6$ samples from the positive standard normal distribution.

Importance sampling is a general MC integration method, but not appropriate for recursive estimation. This is because one must get all the data $\mathbf{y}_{1:t}$ before estimating $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$. Therefore, one must in general recompute the importance weights over the entire sequence each time new data $\mathbf{y}_{t+1}$ becomes available. This leads to high computational complexity, and we therefore present a solution for dealing with this by introducing sequential importance sampling.

3.3 Sequential Importance Sampling

We modify our IS-method so that the importance function $\pi(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$ at time t admits as marginal distribution at time $t-1$ the importance function $\pi(\mathbf{x}_{0:t-1}|\mathbf{y}_{1:t-1})$, this gives

$$\pi(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) = \pi(\mathbf{x}_{0:t-1}|\mathbf{y}_{1:t-1})\pi(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{y}_{1:t}), \tag{46}$$

which by iteration yields

$$\pi(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) = \pi(\mathbf{x}_0) \prod_{k=1}^{t} \pi(\mathbf{x}_k|\mathbf{x}_{0:k-1}, \mathbf{y}_{1:k}). \tag{47}$$

From this importance function we can recursively evaluate the importance weights from (42) in time, and we get

$$w_t^{(i)} \propto w_{t-1}^{(i)} \frac{p\big(\mathbf{y}_t|\mathbf{x}_t^{(i)}\big)p\big(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)}\big)}{\pi\big(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)}, \mathbf{y}_{1:t}\big)}. \tag{48}$$

We get an important particular case of this by adopting the prior distribution as importance distribution

$$\pi(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) = p(\mathbf{x}_{0:t}) = p(\mathbf{x}_0) \prod_{k=1}^{t} p(\mathbf{x}_k|\mathbf{x}_{k-1}). \tag{49}$$

Here, the importance weights satisfy $w_t^{(i)} \propto w_{t-1}^{(i)} p(\mathbf{y}_t|\mathbf{x}_t^{(i)})$. We will see when introducing the sequential importance resampling filter, that we restrict ourselves to the use of the prior distribution as importance sampling distribution. The *degeneracy problem* has revealed issues with the SIS method[8]. As the variance of the importance weights can only increase over time, this leads to a common problem with the SIS filter; degeneracy of particles. Simply explained, this means that after a certain number of recursive steps, only one particle will have a significant weight. SIS is nothing but a constrained version of importance sampling, and it is well known that importance sampling is usually inefficient in higher dimensions[9].

3.4 Selection of Importance Density

3.4.1 The Optimal Choice

It is critical to choose an importance density $\pi(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}, \mathbf{y}_t)$ that minimizes the variance of the importance weights when designing a particle filter. The optimal choice doing just this, has been shown by Doucet et al. (2000)[8] as

$$\begin{aligned}\pi(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}, \mathbf{y}_t)_{opt} &= p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}, \mathbf{y}_t) &(50)\\ &= \frac{p(\mathbf{y}_t|\mathbf{x}_t, \mathbf{x}_{t-1}^{(i)})p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})}{p(\mathbf{y}_t|\mathbf{x}_{t-1}^{(i)})}. &(51)\end{aligned}$$

We can here substitute (50) into (48) yielding

$$w_t^{(i)} \propto w_{t-1}^{(i)} p(\mathbf{y}_t|\mathbf{x}_{t-1}^{(i)}), \qquad (52)$$

meaning that the importance weights at time t can be computed before the particles are propagated to time t. To use the optimal importance function one must first sample from $p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}, \mathbf{y}_t)$ which generally is not straightforward, and second, evaluate

$$p(\mathbf{y}_t|\mathbf{x}_{t-1}^{(i)}) = \int p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})d\mathbf{x}_t \qquad (53)$$

up to a normalizing constant, which may be difficult. In some cases, it might be possible to use the optimal importance density, e.g., when $\mathbf{x}_t$ is a member of a finite set, or when $p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}, \mathbf{y}_t)$ is Gaussian [8].

Gaussian Optimal Importance Function

If we are considering a case where the state dynamics are nonlinear, the measurement equation linear, while all the random elements in the model are additive Gaussian, we get a system given by

$$\begin{aligned}\mathbf{x}_t &= \mathbf{f}_{t-1}(\mathbf{x}_{t-1}) + \mathbf{v}_{t-1}, &(54)\\ \mathbf{y}_t &= \mathbf{H}_t\mathbf{x}_t + \mathbf{w}_t, &(55)\end{aligned}$$

where we have the same definitions as in Chapter 2.1.1. For this particular case, we can show that both the optimal importance density and $p(\mathbf{y}_t|\mathbf{x}_{t-1})$ are

Gaussian, meaning:

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{y}_t) = N(\mathbf{x}_t; \mathbf{a}_t, \mathbf{\Sigma}_t) \tag{56}$$
$$p(\mathbf{y}_t|\mathbf{x}_{t-1}) = N(\mathbf{y}_t; \mathbf{b}_t, \mathbf{S}_t) \tag{57}$$

where

$$\mathbf{a}_t = \mathbf{f}_{t-1}(\mathbf{x}_{t-1}) + \mathbf{\Sigma}_t \mathbf{H}_t^T \mathbf{R}_t^{-1}(\mathbf{y}_t - \mathbf{b}_t) \tag{58}$$
$$\mathbf{\Sigma}_t = \mathbf{Q}_{t-1} - \mathbf{Q}_{t-1}\mathbf{H}_t^T \mathbf{S}_t^{-1} \mathbf{H}_t \mathbf{Q}_{t-1} \tag{59}$$
$$\mathbf{S}_t = \mathbf{H}_t \mathbf{Q}_{t-1} \mathbf{H}_t^T + \mathbf{R}_t \tag{60}$$
$$\mathbf{b}_t = \mathbf{H}_t \mathbf{f}_{t-1}(\mathbf{x}_{t-1}). \tag{61}$$

For proof see [3]. The described analytic evaluation of $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{y}_t)$ and $p(\mathbf{y}_t|\mathbf{x}_{t-1})$ is for most other cases difficult. In the next section, we will describe methods for approximating the optimal importance density, despite not all assumptions holding true.

3.4.2 Suboptimal Choices

The most common suboptimal choice is the transitional prior

$$\pi(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}, \mathbf{y}_t) = p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}). \tag{62}$$

If we have the same additive, zero-mean Gaussian process as in (54), this transitional prior is given by

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}) = N(\mathbf{x}_t; \mathbf{f}_{t-1}(\mathbf{x}_{t-1}^{(i)}), \mathbf{Q}_{t-1}). \tag{63}$$

We now substitute (62) into (48) yielding

$$w_t^{(i)} \propto w_{t-1}^{(i)} p(\mathbf{y}_t|\mathbf{x}_t^{(i)}). \tag{64}$$

As we stated when using the optimal importance function, the importance weights in (52) can be computed *before* the particles are propagated to time t. Equation (64) however, states that this is not possible with the transitional prior. If we are using $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ as the importance density, and it is a much broader distribution than the likelihood $p(\mathbf{y}_t|\mathbf{x}_t)$, then this means only a few particles will be given a high weight. Moreover, this causes the particles to degenerate rapidly making the filter rather useless. Methods for coping with this exist, like in the auxiliary particle filter in Chapter 3.6 [10], or by partitioned sampling [11]. Partitioned sampling is useful if the likelihood is very peaked, but cannot be factorized into a number of broader distributions, usually because each of the partitioned distributions are functions of some and not all of the states. Other examples where it is possible to construct suboptimal approximations to the optimal importance density are based on variations of the Kalman filter [8]. These methods use local linearization techniques with a Gaussian approximated importance density to $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{y}_t)$.

3.5 The Sequential Importance Resampling Filter

A challenge with SIS is that as t grows large, the distribution of the importance weights $w_t^{(i)}$ becomes more and more skewed, thus making it less suitable to represent the posterior distributions sufficiently. The assumptions required to use the SIR filter are very weak, as we must only know the state dynamics and measurement functions, as they are defined in Chapter 2.1.1. We must also be able to sample realizations from the process noise distribution and the prior, as well as the likelihood $p(\mathbf{y}_t|\mathbf{x}_t)$ needs to be available for point wise evaluation. We have the well-known technique of *bootstrapping*, sampling with replacement. The key idea of the bootstrap filter, or SIR filter in our terminology, is that we wish to focus on particles with higher importance weights $w_t^{(i)}$, by replacing the weighted empirical distribution $\hat{P}_N(d\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) = \sum_{i=1}^N \tilde{w}_t^{(i)} \delta_{\mathbf{x}_{0:t}^{(i)}}(d\mathbf{x}_{0:t})$ by the unweighted measure

$$P_N(d\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) = \frac{1}{N}\sum_{i=1}^{N} N_t^{(i)} \delta_{\mathbf{x}_{0:t}^{(i)}}(d\mathbf{x}_{0:t}). \tag{65}$$

Here $N_t^{(i)}$ is the number of offspring associated to particle $\mathbf{x}_{0:t}^{(i)}$ and chosen such that $P_N(d\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$ is close to $\hat{P}_N(d\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$ in the sense that, for any function f_t,

$$\int f_t(\mathbf{x}_{0:t}) P_N(d\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) \approx \int f_t(\mathbf{x}_{0:t}) \hat{P}_N(d\mathbf{x}_{0:t}|\mathbf{y}_{1:t}). \tag{66}$$

Resampling can be done in many ways, but in this thesis we will utilize the simple multinomial resampling scheme. What is important to note regarding the SIR filter, is as the resampling is done at every time index, we get $w_{t-1}^{(i)} = 1/N$ for all $i = 1, ..., N$. This means two things; first, there is no need to pass on the importance weights between each time step; second, the relationship in (64) simplifies to:

$$w_t^{(i)} \propto p(\mathbf{y}_t|\mathbf{x}_t^{(i)}). \tag{67}$$

We explain the SIR filter step-by-step on the next page.

Algorithm 1: SIR Filter Algorithm

```
Initialization, t = 0.
for $i = 1, ..., N$ do
    Sample $\mathbf{x}_0^{(i)} \sim p(\mathbf{x}_0)$ and set $t = 1$.
end
Importance Sampling Step
for $t = 1, ..., T$ do
    for $i = 1, ..., N$ do
        Sample $\tilde{\mathbf{x}}_t^{(i)} \sim p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})$ and set $\tilde{\mathbf{x}}_{0:t}^{(i)} = (\mathbf{x}_{0:t}^{(i)}, \tilde{\mathbf{x}}_t^{(i)})$.
        Evaluate the importance weights
            $w_t^{(i)} = p(\mathbf{y}_t|\tilde{\mathbf{x}}_t^{(i)}).$   (68)
    end
    Normalize the importance weights
        $\tilde{w}_t^{(i)} = \frac{w_t^{(i)}}{\sum_{j=i}^{N} w_t^{(j)}}.$   (69)
    Selection Step
    Resample with replacement $N$ particles $\{\mathbf{x}_{0:t}^{(i)}\}_{i=1}^{N}$ from the set
    $\{\tilde{\mathbf{x}}_{0:t}^{(i)}\}_{i=1}^{N}$ according to the normalized importance weights.
end
```

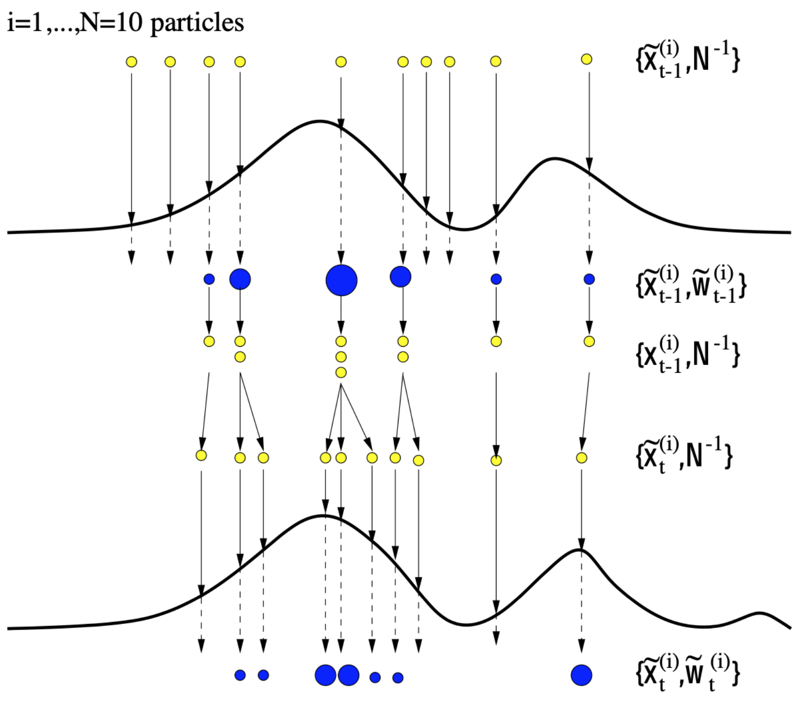
i=1,...,N=10 particles
$\{\tilde{x}_{t-1}^{(i)}, N^{-1}\}$
$\{\tilde{x}_{t-1}^{(i)}, \tilde{w}_{t-1}^{(i)}\}$
$\{x_{t-1}^{(i)}, N^{-1}\}$
$\{\tilde{x}_{t}^{(i)}, N^{-1}\}$
$\{\tilde{x}_{t}^{(i)}, \tilde{w}_{t}^{(i)}\}$

Figure 3: Visualization of the SIR filter with $N = 10$ particles. We see how we start with an unweighted measure at $t-1$, and how the weights are updated yielding the weighted measure.

Example 3
We now show a simple way to implement the SIR filter in practice. We apply the algorithm to the following nonlinear, non-Gaussian model. This is a classic illustration model, taken from Gordon et al. (1993)[1]. We have the two processes

$$x_t = \frac{1}{2}x_{t-1} + 25\frac{x_{t-1}}{1+x_{t-1}^2} + 8\cos(1.2t) + v_t \tag{70}$$

$$y_t = \frac{x_t^2}{20} + w_t, \tag{71}$$

where $x_1 \sim N(0, \sigma_1^2)$, v_t and w_t are mutually independent, normally distributed white noises, where $v_t \sim N(0, \sigma_v^2)$ and $w_t \sim N(0, \sigma_w^2)$ with $\sigma_1^2 = \sigma_v^2 = 10$ and $\sigma_w^2 = 1$.

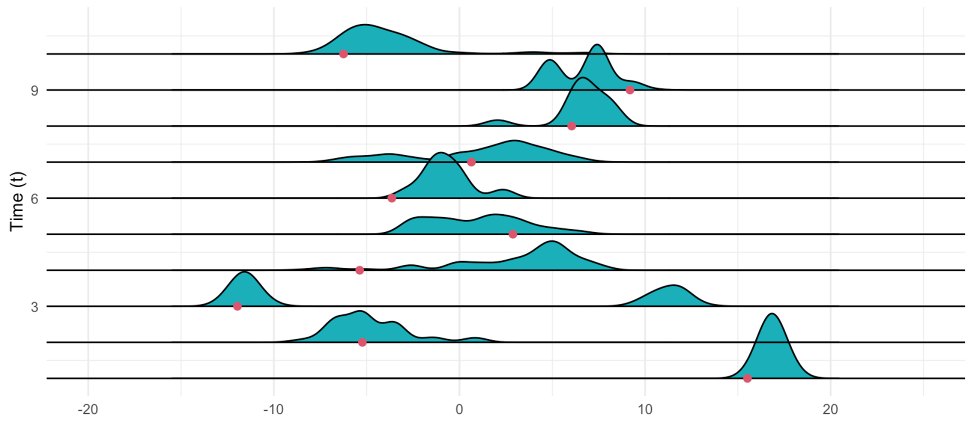

Figure 4: Estimated filtered densities at time points $t = \{10, 20, 30, ..., 100\}$ with $N = 1000$ particles. The red dots represent the true value of x_t at each time step.

When applying the SIR filter, the importance sampling density is independent of the measurement $\mathbf{y}_t$. Thus, the state space is explored without any knowledge of the observations. This can make the filter inefficient as it is sensitive to outliers. Diversity of particles is also a possible issue, as we resample at every iteration. However, the SIR filter has the advantage that the importance weights are easily evaluated and the importance density can be easily sampled.

3.6 Auxiliary SIR Filter

The auxiliary SIR (ASIR) filter was first introduced by Pitt and Shephard in 1999, as a variant of the SIR filter[11]. The basic idea was to perform the resampling step at time $t-1$ *before* the particles were propagated to time t. Hence, the ASIR filter tries to copy the sequence of steps carried out when we have the optimal importance density available. We can derive the ASIR filter by introducing the importance density $\pi(\mathbf{x}_t, i|\mathbf{y}_{1:t})$, sampling the pair $\{\mathbf{x}_t^{(j)}, i^{(j)}\}_{j=1}^N$, where $i^{(j)}$ refers to the index of the particle at $t-1$. We can apply Bayes' theorem to derive the following for the posterior

$$p(\mathbf{x}_t, i|\mathbf{y}_{1:t}) \propto p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t, i|\mathbf{y}_{1:t-1}) \tag{72}$$

$$= p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|i, \mathbf{y}_{1:t-1})p(i|\mathbf{y}_{1:t-1}) \tag{73}$$

$$= p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})w_{t-1}^{(i)}. \tag{74}$$

The filter then obtains a sample from the joint density $p(\mathbf{x}_t, i|\mathbf{y}_{1:t})$, and then omits the indices i in the pair $(\mathbf{x}_t, i)$ to produce a sample $\{\mathbf{x}_t^{(j)}\}_{j=1}^N$ from the marginalized density $p(\mathbf{x}_t|\mathbf{y}_{1:t})$.

We then define the importance density used to draw the sample $\{\mathbf{x}_t^{(j)}, i^{(j)}\}_{j=1}^N$ so that it satisfies

$$\pi(\mathbf{x}_t, i|\mathbf{y}_{1:t}) \propto p(\mathbf{y}_t|\boldsymbol{\mu}_t^{(i)})p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})w_{t-1}^{(i)} \tag{75}$$

where $\boldsymbol{\mu}_t^{(i)}$ is some characterization of $\mathbf{x}_t$ given $\mathbf{x}_{t-1}^{(i)}$. This could be the mean or a sample, where we would have $\boldsymbol{\mu}_t^{(i)} = \mathbb{E}[\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}]$ or $\boldsymbol{\mu}_t^{(i)} \sim p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})$, respectively. We can write

$$\pi(\mathbf{x}_t, i|\mathbf{y}_{1:t}) = \pi(i|\mathbf{y}_{1:t})\pi(\mathbf{x}_t|i, \mathbf{y}_{1:t}) \tag{76}$$

and define

$$\pi(\mathbf{x}_t, |i, \mathbf{y}_{1:t}) \triangleq p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}) \tag{77}$$

so that it follows from (75) that

$$\pi(i|\mathbf{y}_{1:t}) \propto p(\mathbf{y}_t|\boldsymbol{\mu}_t^{(i)})w_{t-1}^{(i)}. \tag{78}$$

Then we have according to (48), the sample $\{\mathbf{x}_t^{(j)}, i^{(j)}\}_{j=1}^N$ is assigned a weight proportional to the ratio of the right-hand side of (74) and (75) yielding

$$w_t^{(j)} \propto w_{t-1}^{(i^j)} \frac{p(\mathbf{y}_t|\mathbf{x}_t^{(j)})p(\mathbf{x}_t^{(j)}|\mathbf{x}_{t-1}^{(i^j)})}{\pi(\mathbf{x}_t^{(j)}, i^{(j)}|\mathbf{y}_{1:t})} = \frac{p(\mathbf{y}_t|\mathbf{x}_t^{(j)})}{p(\mathbf{y}_t|\boldsymbol{\mu}_t^{(i^j)})}. \tag{79}$$

Compared to the SIR filter, the ASIR filter presents an advantage in that it naturally generates points from the sample at $t-1$ which are most likely to be in the region of high likelihood when we condition on the current measurement. If we have small process noise such that $p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})$ is well characterized by $\boldsymbol{\mu}_t^{(i)}$ the ASIR filter is often less sensitive to outliers than the SIR filter, and we get more even weights $w_t^{(i)}$. If however we have large process noise, the ASIR filter will often work poorly. This is due to the fact that $p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})$ is not characterized by a single point in the state space, and the filter resamples based on poor approximation. This may even lead to the ASIR filter degrading the performance. We explain the algorithm on the next page.

Algorithm 2: Auxiliary SIR Filter Algorithm

```
Initialization $t = 0$.
for $i = 1, ..., N$ do
    Sample $\mathbf{x}_0^{(i)} \sim p(\mathbf{x}_0)$.
end
for $t = 1, ..., T$ do
    for $i = 1, ..., N$ do
        - Calculate $\boldsymbol{\mu}_t^{(i)}$.
        - Calculate the importance weights
            $$w_t^{(i)} = \pi(i|\mathbf{y}_{1:t}) \propto p(\mathbf{y}_t|\boldsymbol{\mu}_t^{(i)})w_{t-1}^{(i)}. \qquad (80)$$
    end
    Normalize the importance weights $\tilde{w}_t^{(i)} = \frac{w_t^{(i)}}{\sum_{j=1}^{N} w_t^{(j)}}$.
    Resample with replacement $N$ integers $\{i^{(j)}\}_{i=1}^{N}$ using
    probabilities from (80).
    for $j = 1, ..., N$ do
        Sample $\mathbf{x}_t^{(j)} \sim \pi(\mathbf{x}_t|i^{(j)}, \mathbf{y}_{1:t}) = p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i^j)})$.
        Calculate $w_t^{(j)} = \frac{p(\mathbf{y}_t|\mathbf{x}_t^{(j)})}{p(\mathbf{y}_t|\boldsymbol{\mu}_t^{(i^j)})}$ as in (79).
    end
    Normalize the weights $\tilde{w}_t^{(j)} = \frac{w_t^{(j)}}{\sum_{i=1}^{N} w_t^{(i)}}$.
end
```

We have now introduced the basis for SMC and particle filtering, and explained how these are used in different settings. As mentioned, the great challenge is often dealing with high dimensionality in statistical models. Markov chain Monte Carlo (MCMC) is a great tool for sampling from a probability distribution. It differs from SMC as it does not record a collection of multiple samples approximately distributed according to the posterior, but rather iteratively creates a single sample which is then added to a Markov chain. Thus, MCMC methods are not sequential, as they do not account for new data. This creates different areas of usage for the methods. SMC are great at estimating probability distributions on-line, while MCMC works great for parameter estimation. Parameters are fixed and do not change in time, which is why MCMC works well for this. By applying MCMC when estimating parameters, the possible problem of particle degeneration is avoided. However, an area where MCMC often performs worse, is when the distribution to be estimated is multimodal. This is because it can easily get stuck within one of the modes. We will therefore present a combination of the two methods in the next section, called particle Markov chain Monte Carlo (PMCMC).

3.7 Particle Markov Chain Monte Carlo

Combining the two methods SMC and MCMC has been done previously, like using MCMC kernels to build proposal densities for SMC algorithms[12]. With PMCMC we look to have a different approach, as the aim is to use SMC to design efficient high dimensional proposal distributions for MCMC algorithms. MCMC uses MC sampling to create a Markov chain, and thus one can estimate the target distribution by recording states from the chain. We will here first introduce the *Metropolis-Hastings algorithm* (MH), which is a well-known MCMC algorithm, that can also be applied to PMCMC. Then we demonstrate how MH can be combined with SMC methods. The following theoretical framework is largely based on the paper by Andrieu et al. (2010) [13].

3.7.1 The Metropolis-Hastings Algorithm

We begin by first introducing a basic and well-known MCMC algorithm. This algorithm is a random walk algorithm, designed to approximate a stationary distribution $p(\mathbf{X})$. The algorithm generates proposed states $\mathbf{X}^*$ from the proposal distribution $\pi(\mathbf{X}^*|\mathbf{X})$, where each state is evaluated according to the *acceptance ratio* α. We describe the algorithm step-by-step in algorithm 3. We can see how we either move to a new point, or stay at the same based on the relationship between our current point and the proposed point. The two essential ideas, are first, that if the new point is in a higher density region, we move with probability 1. Second, if the new point is in a lower density region, we still move with a non-zero probability. This makes us likely to stay in higher density regions, but also makes it clear how the algorithm is vulnerable to multimodal models. If we are trapped in a high density mode, one will usually need a very large step size to jump out of this. The burn-in period fixes some of these issues, but usually just if we start in a lower density region. However, this MC algorithm is still a very powerful and preferred method when it comes to sampling from a distribution with high dimensionality.

Algorithm 3: Metropolis-Hastings Algorithm

```
Initialization, $t = 0$.
Initialize the Markov chain with $\mathbf{X}_0$.
for $t = 1, ..., T + T_B$ do
    Generate a proposal $\mathbf{X}^*$ from the proposal distribution
     $\mathbf{X}^* \sim \pi(\mathbf{X}^*|\mathbf{X})$ where $\mathbf{X} = \mathbf{X}_t$.
    Compute acceptance ratio:
        $$\alpha = \min\left\{1, \frac{p(\mathbf{X}^*)\pi(\mathbf{X}|\mathbf{X}^*)}{p(\mathbf{X})\pi(\mathbf{X}^*|\mathbf{X})}\right\}. \quad (81)$$
    Accept or reject
    If $\alpha = 1$ we accept $\mathbf{X}^*$ and add it to our Markov chain such
    that $\mathbf{X}_{t+1} = \mathbf{X}^*$, otherwise we accept with probability $\alpha$. If
    rejected then we add the previous sample to the current
    position such that $\mathbf{X}_{t+1} = \mathbf{X}_t$.
end
```

T_B denotes the first burn-in samples, which we discard to avoid bias. When we have $\alpha < 1$, we compare this with a value $u \sim U(0, 1)$ to determine if we will accept or reject the sample.

3.7.2 Particle Marginal Metropolis-Hastings Sampler

We now move towards PMCMC methods. These types of methods rely on a non-trivial and non-standard combination of MCMC and SMC methods, taking advantage of the strength of both its components. What we will do further, is using SMC to design efficient high dimensional proposal distributions for MCMC algorithms. We then apply this to state space models for inference, i.e., parameter estimation. In this thesis we will restrict ourselves to the *particle marginal Metropolis-Hastings sampler*, which has been described in [13].

Say we have a parameter $\theta \in \Theta$, where Θ is our parameter space, which may be multidimensional. If we now consider the scenario where we wish to sample from $p(\theta, \mathbf{x}_{1:T}|\mathbf{y}_{1:T})$ which is proportional to $p_\theta(\mathbf{x}_{1:T}, \mathbf{y}_{1:T})p(\theta)$, where $p(\theta)$ is the prior for θ. The goal here is to jointly update θ and $\mathbf{x}_{1:T}$. Assuming we can sample from the conditional density and that we can do $p(\theta, \mathbf{x}_{1:T}|\mathbf{y}_{1:T}) = p(\theta|\mathbf{y}_{1:T})p_\theta(\mathbf{x}_{1:T}|\mathbf{y}_{1:T})$, we can suggest the following proposal density for an MH update:

$$\pi(\theta^*, \mathbf{x}^*_{1:T}|\theta, \mathbf{x}_{1:T}) = \pi(\theta^*|\theta)p_{\theta^*}(\mathbf{x}^*_{1:T}|\mathbf{y}_{1:T}). \quad (82)$$

Here the proposed $\mathbf{x}^*_{1:T}$ is perfectly 'adapted' to the proposed θ^*. What this really means, is that for every iteration of the PMCMC algorithm, we run the SMC algorithm one time with sampled parameters for that iteration to calculate the new likelihood. We get the acceptance probability to be

$$\alpha = \min\left\{1, \frac{p(\theta^*, \mathbf{x}^*_{1:T}|\mathbf{y}_{1:T})}{p(\theta, \mathbf{x}_{1:T}|\mathbf{y}_{1:T})} \frac{\pi(\theta, \mathbf{x}_{1:T}|\theta^*, \mathbf{x}^*_{1:T})}{\pi(\theta^*, \mathbf{x}^*_{1:T}|\theta, \mathbf{x}_{1:T})} = \frac{p_{\theta^*}(\mathbf{y}_{1:T})p(\theta^*)/\pi(\theta^*|\theta)}{p_\theta(\mathbf{y}_{1:T})p(\theta)/\pi(\theta|\theta^*)}\right\}. \tag{83}$$

Thus we can see how we are targeting the marginal density $p(\theta|\mathbf{y}_{1:T}) \propto p_\theta(\mathbf{y}_{1:T})p(\theta)$, and how we reduce the difficult problem of sampling from $p(\theta, \mathbf{x}_{1:T}|\mathbf{y}_{1:T})$ to sampling from $p(\theta|\mathbf{y}_{1:T})$. We describe the algorithm below.

```
Algorithm 4: Particle Marginal MH Algorithm
  Initialization, $j = 0$.
  - Set $\theta(0)$ arbitrarily.
  - Run SMC algorithm targeting $p_{\theta(0)}(\mathbf{x}_{1:T}|\mathbf{y}_{1:T})$, sample
    $\mathbf{X}_{1:T}(0) \sim \hat{p}_{\theta(0)}(\cdot|\mathbf{y}_{1:T})$ and let $\hat{p}_{\theta(0)}(\mathbf{y}_{1:T})$ denote the marginal
    likelihood estimate.

  Filtering and updating
  for $j = 1, ..., N$ do
      - Sample $\theta^* \sim \pi(\cdot|\theta(j-1)$.
      - Run SMC algorithm targeting $p_{\theta^*}(\mathbf{x}_{1:T}|\mathbf{y}_{1:T})$, sample
        $\mathbf{X}^*_{1:T} \sim \hat{p}_{\theta^*}(\cdot|\mathbf{y}_{1:T})$ and let $\hat{p}_{\theta^*}(\mathbf{y}_{1:T})$ denote the marginal
        likelihood estimate.
      - Compute acceptance probability
            $\alpha = \min\left\{1, \frac{\hat{p}_{\theta^*}(\mathbf{y}_{1:T})p(\theta^*)/\pi_j(\theta^*|\theta(j-1))}{\hat{p}_{\theta(j-1)}(\mathbf{y}_{1:T})p(\theta(j-1))/\pi_j(\theta(j-1)|\theta^*)}\right\}.$   (84)
        With probability $\alpha$ set $\theta(j) = \theta^*, \mathbf{X}_{1:T}(j) = \mathbf{X}^*_{1:T}$ and
        $\hat{p}_{\theta(j)}(\mathbf{y}_{1:T}) = \hat{p}_{\theta^*}(\mathbf{y}_{1:T})$; otherwise set
        $\theta(j) = \theta(j-1), \mathbf{X}_{1:T}(j) = \mathbf{X}_{1:T}(j-1)$ and
        $\hat{p}_{\theta(j)}(\mathbf{y}_{1:T}) = \hat{p}_{\theta(j-1)}(\mathbf{y}_{1:T})$.
  end
```

The notation $(j), (j-1)$ is here used for stronger clarification of the index j. The important part, is to see how we are proposing a new θ before the SMC algorithm runs and a new $\hat{p}(\mathbf{y}_{1:T})$ after every iteration j, and how the proposed parameter is used in the SMC algorithm. It's crucial to distinguish how the PMCMC algorithm runs for $j = 1, ..., K$ iterations, and every j'th iteration the SMC algorithm runs $t = 1, ..., T$ times while creating $i = 1, ..., N$ particles for every t. A possible drawback of the PMCMC method is therefore that it requires TxN iterations for creating a single sample. This gives a heavy computational load, as the MCMC method requires some time to enter its stationary distribution. A possible burn-in sample is 10^5 iterations, making it typical to run an MCMC algorithm for 10^6 iterations, meaning that the SIR filter would also run for 10^6 iterations. We also see here how we describe the algorithm for any parameter $\theta \in \Theta$. When applying the method later, we will show how it can be done using the whole parameter space Θ.

4 A Benchmark Experiment with the SIR Filter

We will now do a demonstration with our SIR algorithm from Chapter 3.5, using an *autoregressive* (AR) *model.* The AR model is a representation of a type of random process, which can be used to describe a time series data sequence, e.g., in nature or economics, in which each value is linearly dependent on its previous values. The AR(p) model is defined as follows

$$X_t = \sum_{i=1}^{p} \varphi_i X_{t-i} + \varepsilon_t, \tag{85}$$

where $\varphi_1, ..., \varphi_p$ are the parameters of the model up to lag p and ε_t being white noise.

4.1 The Model

We will now look at an AR(1) model, which is a linear state space form model, where the likelihood can be evaluated by the Kalman filter. The model in play is

$$\mathbf{y}_t = \mathbf{x}_t + \varepsilon_t, \quad \varepsilon_t \sim N(0, \sigma_\varepsilon^2) \tag{86}$$

$$\mathbf{x}_{t+1} = \mu + \varphi(\mathbf{x}_t - \mu) + \eta_t, \quad \eta_t \sim N(0, \sigma_\eta^2), \tag{87}$$

with $\Theta = (\sigma_\varepsilon^2, \sigma_\eta^2, \varphi, \mu)$ being our parameter space. For this example, we set $\sigma_\varepsilon^2 = 2, \sigma_\eta^2 = 0.02, \varphi = 0.975$ and $\mu = 0.5$, as these values are typical for the stochastic volatility (SV) model [14]. Here φ represents the persistence in variance, whilst σ_ε^2 is selected based on the curvature of the measurement density in the SV model. Thus, we generate our processes $\mathbf{x}_t$ and $\mathbf{y}_t$, filter out the noise ε_t by using the relationship between $\mathbf{y}_t$ and $\mathbf{x}_t$ and the SIR filter, which allows for comparison as we know the true values of $\mathbf{x}_t$. We can do likelihood evaluation of our weights by computing the log-likelihood

$$\log \hat{L}_N(\Theta) = \sum_{t=1}^{T} \log \left(\frac{1}{N} \sum_{i=1}^{N} w_t^{(i)} \right). \tag{88}$$

The log-likelihood is found by exploiting the relationship in the integral

$$p(\mathbf{y}_t|\Theta; F_{t-1}) = \int p(\mathbf{y}_t|\mathbf{x}_t; \Theta) p(\mathbf{x}_t|F_{t-1}; \Theta) d\mathbf{x}_t, \tag{89}$$

while we can obtain samples from the transition density $p(\mathbf{x}_t|\mathbf{x}_{t-1}; \Theta)$ as we have samples from $p(\mathbf{x}_{t-1}|F_{t-1}; \Theta)$ by our particle filter, hence we are able to estimate (89). We can calculate the likelihood by the Kalman filter, so we compare our results from this with our results from the SIR algorithm. On the next page we illustrate the error estimates for $\log p(\mathbf{y}_t|F_{t-1})$, where F_{t-1} is the information up to time $t-1$, meaning $\{\mathbf{y}_1, ..., \mathbf{y}_{t-1}\}$.

4.2 Results

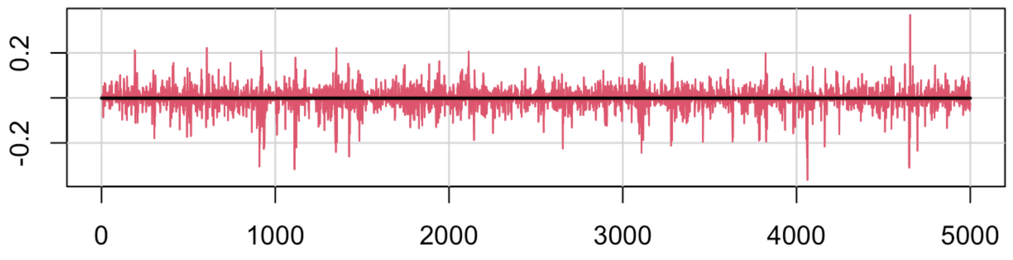

Figure 5: Error plot for $N = 300$ particles displaying the difference $\hat{l}_t - l_t$ with a horizontal line displaying the mean error.

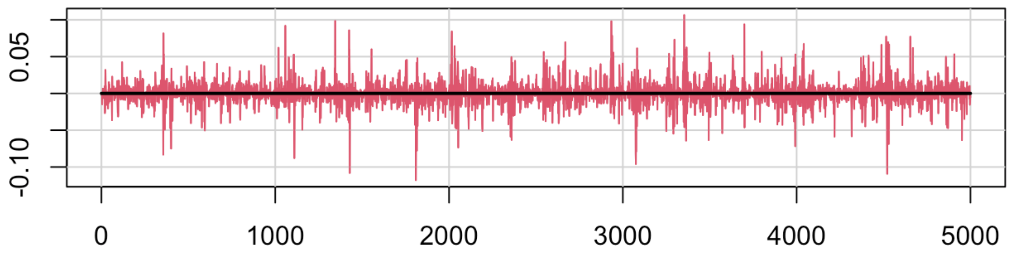

Figure 6: Error plot for $N = 3500$ particles displaying the difference $\hat{l}_t - l_t$ with a horizontal line displaying the mean error.

We have here generated a time series where $T = 5000$. The implementation of the SIR filter is fairly straightforward here, as we just need a few lines of code after initializing the filter. In figures 5 and 6 we look at the log-predictive density $l_t = \log p(\mathbf{y}_t|F_{t-1})$, and the corresponding estimate $\hat{l}_t$ from the SIR filter. We see when comparing with results from the Kalman filter, we get estimates with little error, and the error decreases as N increases. We also see that the mean is centered around zero and the variance homoscedastic, with each step uncorrelated. We then proceed by plotting the observations against the true values of our model, and our filtered values.

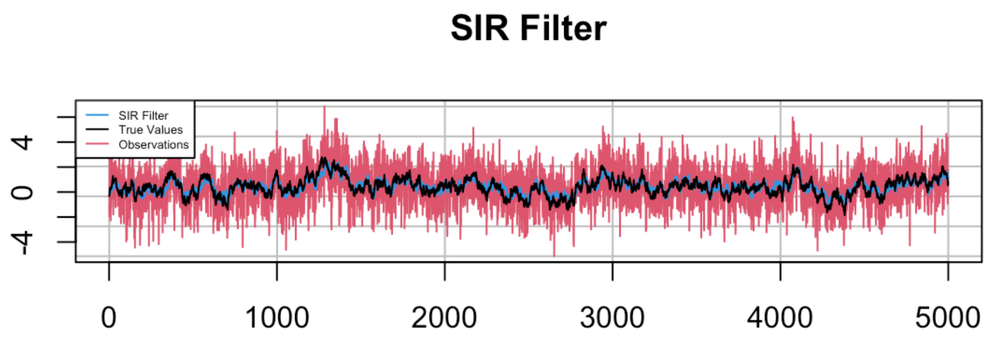

Figure 7: $N = 300$ particles.

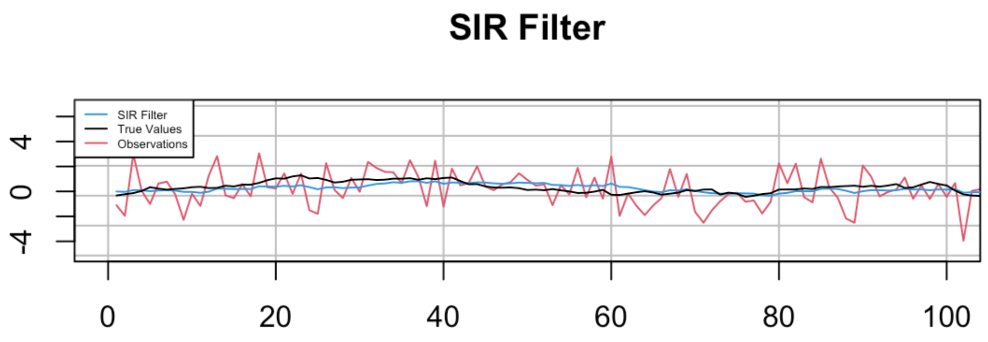

Figure 8: $N = 300$ particles, zoomed in for $t = 0, ..., 100$.

In figures 7 and 8 we plot the filtered values $\tilde{\mathbf{x}}_t$ against the true values $\mathbf{x}_t$ and the observations $\mathbf{y}_t$.

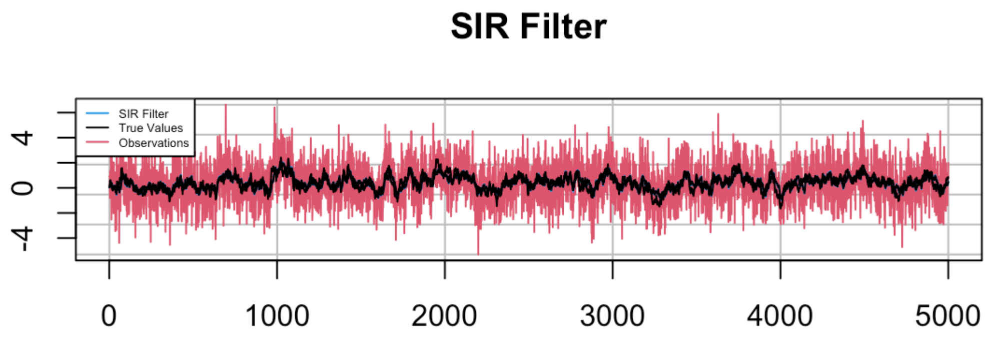

Figure 9: $N = 3500$ particles.

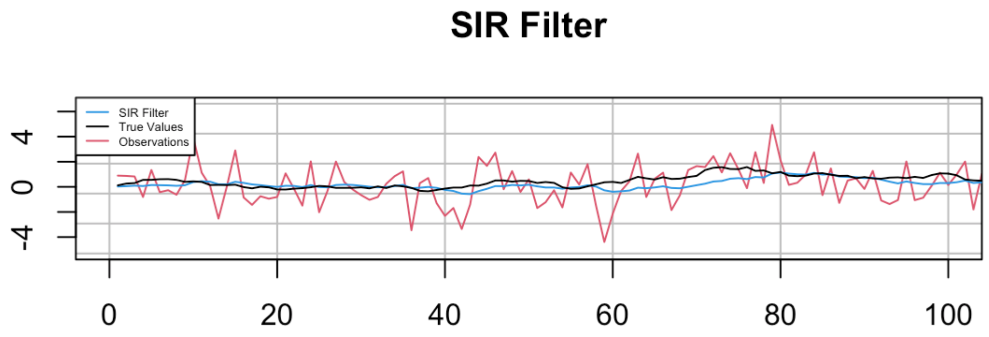

Figure 10: $N = 3500$ particles, zoomed in for $t = 0, ..., 100$.

We see as we increase to $N = 3500$ particles how our SIR filter performs better, and we can barely see any error when we look at $T = 5000$ time steps. We can then compare this with the results from the Kalman filter.

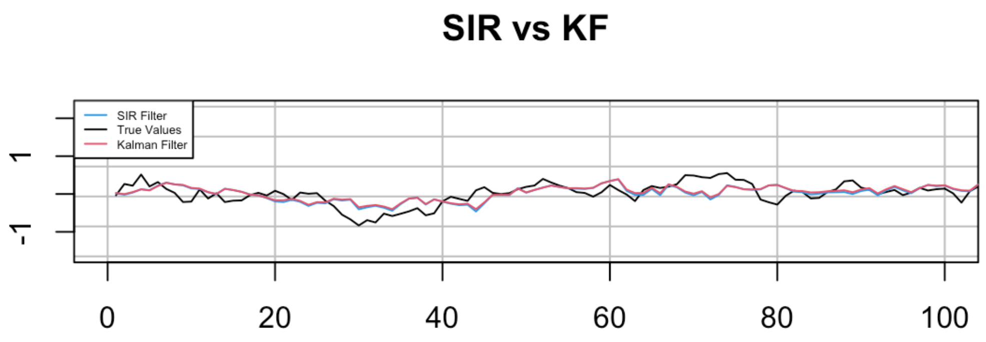

Figure 11: Values estimated by the Kalman filter, the SIR filter and the true values with $N = 3500$ particles, zoomed in for $t = 0, ..., 100$.

We see here how well the Kalman filter works for estimating a linear, Gaussian model. We now perform maximum likelihood estimation (MLE) for the parameter μ. We use (88) to find the MLE, by running the particle filter for a set of values for μ, to find the value with the highest likelihood. We see how we get good estimates close to the true value of $\mu = 0.5$ in figures 12 and 13.

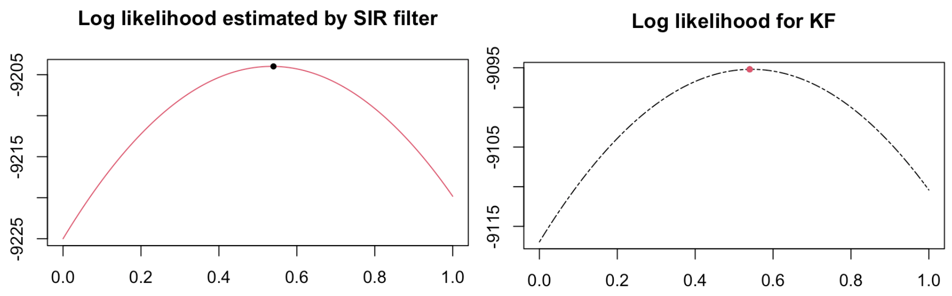

Figure 12: Kalman filter log-likelihood slice for μ together with its estimate. $N = 300$ particles used.

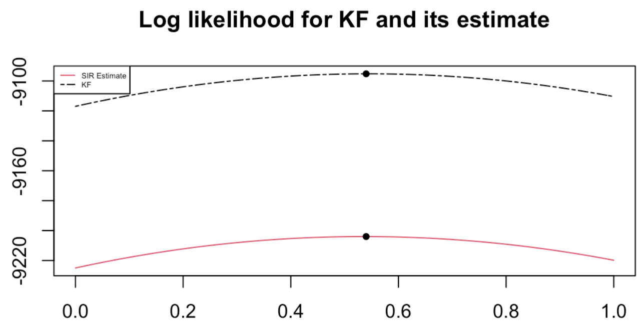

Figure 13: Kalman filter log-likelihood slice for μ together with its estimate. $N = 300$ particles used, zoomed out.

We see how we can verify how well the SIR filter works, when applying it to our own generated data. For achieving even better results, one can increase the number of particles, but at the expense of more computational complexity. The advantage we have in this experiment, is that we know the true values at each time step. In real-life, this is of course unrealistic. In the next chapter, we will conduct an experiment applying the SIR filter to observed returns from financial data. Our aim is to filter out the volatility, which cannot be directly observed.

5 Particle Filtering Using S&P 500 Returns

We now wish to produce some results using real-life data from the S&P 500 stock index. Our goal is still to estimate some latent states, using only observations, by running these through our particle filter. In our case, this will be the unobserved spot volatility, while our observed data are the return data. In the academic literature on stochastic volatility, this has been done many times for different purposes. Whether it is searching for volatility specifications to find which model can best describe the conditional returns distribution, or looking at intraday returns to construct daily realized volatility results that can help explain and predict the volatility distribution. Stochastic volatility models have also been used for fitting option prices across strike prices, maturities and time. As mentioned, our aim is to estimate the unobserved volatility by only looking at the observed returns. By taking the stochastic volatility and multiple other volatility factors into account, we wish to explain the filtered data in greater detail. Then we proceed by estimating our parameters using particle Markov chain Monte Carlo. The work presented in this chapter is largely based on Christoffersen et al. (2010)[15]. By using a different data set which includes the financial crisis of 2007-2008, we are trying to investigate whether the model we choose works well when taking extreme volatility spikes into account. It is crucial for a volatility model to be able to estimate volatility changes well during uncertain periods for managing risk. Thus, we focus in this thesis on establishing a well-specified volatility model. Also in contrast to [15], we are using the aforementioned PMCMC method to estimate our parameters, as opposed to the *maximum likelihood importance sampling* (MLIS) method.

5.1 The Model

Our model is based on the benchmark Heston (1993) affine square root (SQR) model[16], where we assume that the underlying spot price S and the instantaneous change in variance V have the following dynamics

$$dS = \mu S dt + \sqrt{V} S dz, \tag{90}$$

$$\text{SQR} : dV = \kappa(\theta - V)dt + \sigma\sqrt{V}dw, \tag{91}$$

where the parameters denote the following:

- κ is the speed of mean reversion, θ the unconditional variance and σ the variance of variance.
- μ is the instantaneous rate of return.
- dz and dw are Brownian motions with $corr(dz, dw = \rho)$.

If the parameters satisfy $2\kappa\theta > \sigma^2$, known as the *Feller condition*, the volatility process is strictly positive[17]. The SQR model accounts for time-varying volatility and a leverage effect, while implying that the instantaneous change in volatility should be Gaussian and homoscedastic. This is a fairly strong implication, and we can easily evaluate this when investigating our data. We will use the SQR model as a building block to introduce some alternative models. It is relatively straightforward to use more heavily parameterized models, that will outperform the SQR model. In general, a model with more parameters will always outperform a model with less. For convenience, we only introduce models with the same number of parameters as Heston's SQR model. For a more heavily parameterized model, see e.g., Dufays et al. (2022)[18]. We can generalize and extend the SQR model to

$$dV = \kappa V^a(\theta - V)dt + \sigma V^b dw, \tag{92}$$

where $a = \{0, 1\}$ and $b = \{1/2, 1, 3/2\}$.
This framework yields a total of six different models for the given values, which we denote:

a	b	Name
0	1/2	SQR
1	1/2	SQRN
0	1	ONE
1	1	ONEN
0	3/2	3/2
1	3/2	3/2N

We see that the ONE model becomes the continuous-time *Generalized Autoregressive Conditional Heteroscedasticity* (GARCH) model with $p = q = 1$, where the randomness of the diffusion term varies with the variance instead of the square root of the variance[19]. In this thesis, we will estimate one of these models, but due to time constraints, we will not estimate several for comparison. Instead, we will compare our results with those by [15].

5.2 Return Data

We use S&P 500 returns from January 1st 1995 to December 31st 2010, accumulating to 4027 trading days. Index returns are obtained from Yahoo Finance, which we transform to log returns.

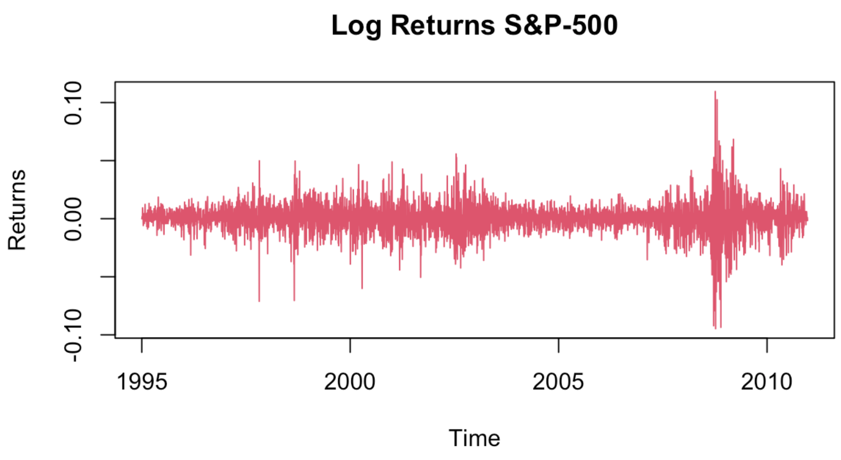

Figure 14: Log returns for the S&P 500 index.

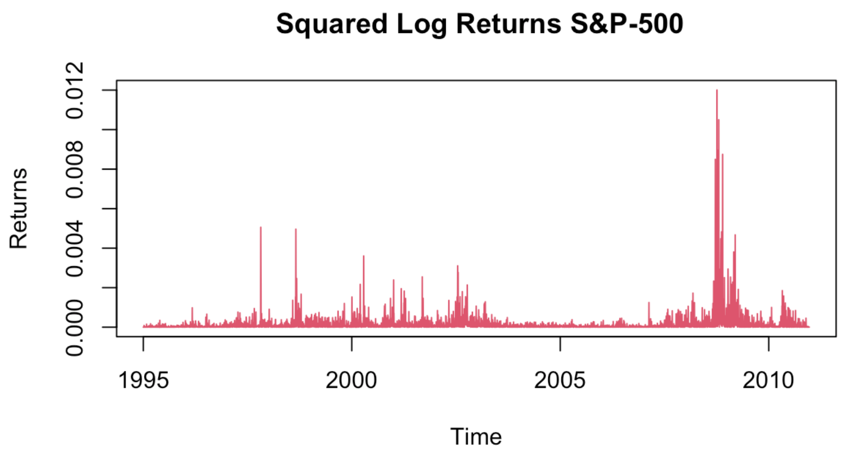

Figure 15: Squared log returns for the S&P 500 index.

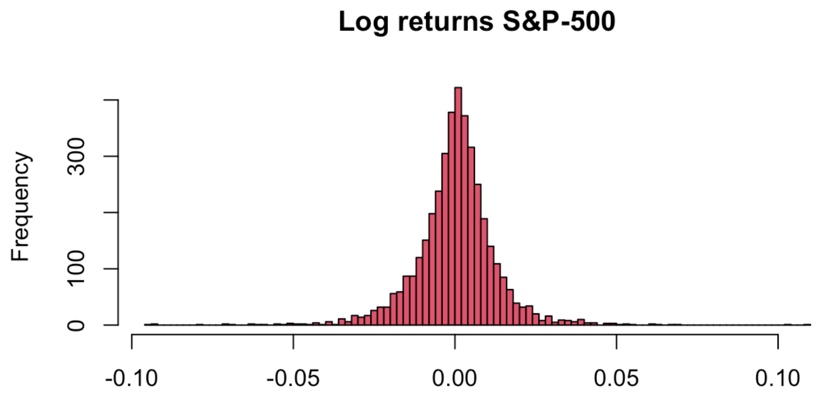

Figure 16: Distribution for log returns for the S&P 500 index.

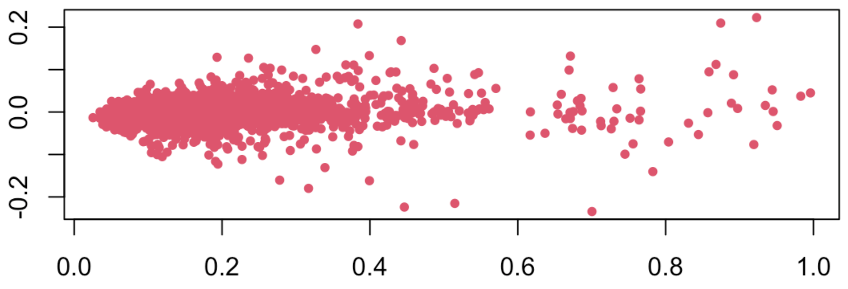

Figure 17: Realized volatility for a 20-day window, displaying the changes against the corresponding levels.

From our data we see that the returns appear to be normally distributed. We also see the biggest spikes happen around the time of the financial crisis, as well as some smaller ones around the dot-com bubble. We get the annualized value of the return drift parameter μ to be 0.063. We also see from Figure 17 that the volatility does not show any indication of being normally distributed.

5.3 Discretization and Estimation

We time-discretize the continuous-time model, and we do this by applying the well known *Itô's lemma*:
Assuming a process X has a stochastic differential by

$$dX(t) = \mu(t)dt + \sigma(t)dW(t), \tag{93}$$

we can define a process Z by $Z(t) = f(t, X(t))$ which has a stochastic differential given by

$$df = \frac{\partial f}{\partial t}dt + \frac{\partial f}{\partial x}dX + \frac{1}{2}\frac{\partial^2 f}{\partial x^2}(dX)^2, \tag{94}$$

where we have the following multiplication table

$$\begin{cases} (dt)^2 = 0, \\ dt \cdot dW = 0, \\ (dW)^2 = dt. \end{cases}$$

As we now proceed with log returns, we write the generic SV process as

$$\text{d } \ln(S) = \left(\mu - \frac{1}{2}V\right)dt + \sqrt{V}dz \tag{95}$$

$$dV = \kappa V^a(\theta - V)dt + \sigma V^b dw. \tag{96}$$

The equations (95) and (96) specifies here the relationship between the unobserved state of the volatility and the observed stock prices. We discretize these by the Euler scheme and applying Itô's[20]

$$\ln(S_{t+1}) = \ln(S_t) + \left(\mu - \frac{1}{2}V_t\right) + \sqrt{V_t}z_{t+1} \tag{97}$$

$$V_{t+1} = V_t + \kappa V_t^a(\theta - V_t) + \sigma V_t^b w_{t+1} \tag{98}$$

with $(z_{t+1}, w_{t+1}) \sim N(0,1)$. Going forward we will denote $\ln(S_{t+1}) - \ln(S_t)$ as R_{t+1}.

5.4 Volatility filtering

We continue by applying our SIR filter as previously, as our aim is to approximate V_{t+1} by a set of N particles, that are updated iteratively by the help of (97) and (98). We use particles $\left\{V_{t+1}^{(i)}\right\}_{i=1}^N$ from the empirical distribution of V_{t+1}, that are conditioned on particles $\left\{V_t^{(i)}\right\}_{i=1}^N$ from the empirical distribution of V_t.

5.4.1 Step 1: Sampling

We simulate the state forward with N particles $\left\{\tilde{V}_{t+1}^{(i)}\right\}_{i=1}^N$ by (98). For the first period we fix all our particles as $V_1^{(i)} = \theta$. We can then get the correlated shocks by

$$z_{t+1}^{(i)} = \left(\mathrm{R}_{t+1} - \left(\mu - \frac{1}{2}V_t^{(i)}\right)\right) \Big/ \sqrt{V_t^{(i)}} \tag{99}$$

$$w_{t+1}^{(i)} = \rho z_{t+1}^{(i)} + \sqrt{1-\rho^2}\varepsilon_{t+1}^{(i)}, \qquad \varepsilon \sim N(0,1) \tag{100}$$

thus yielding $corr(z_{t+1}^{(i)}, \varepsilon_{t+1}^{(i)}) = 0$ and $corr(z_{t+1}^{(i)}, w_{t+1}^{(i)}) = \rho$. μ is as mentioned the instantaneous rate of return, so we set this fixed to the mean of our observed returns. We now substitute (99) into (98) which gives

$$\tilde{V}_{t+1}^{(i)} = V_t^{(i)} + \kappa(V_t^{(i)})^a(\theta - V_t^{(i)}) + \sigma(V_t^{(i)})^b w_{t+1}^{(i)}. \tag{101}$$

We have now simulated N raw particles which provide a set of possible values of V_{t+1}.

5.4.2 Step 2: Computing and Normalizing the Weights

We now have a vector of N possible values of V_{t+1}, and from (97) we know given the other available information this is sufficient to generate $\ln(S_{t+2})$. Thus we use (97) to evaluate the likelihood that observation $\ln(S_{t+2})$ has been generated by V_{t+1}. We can now compute the weight given to each particle, i.e., the likelihood that the particle has generated $\ln(S_{t+2})$ by:

$$w_{t+1}^{(i)} = \frac{1}{\sqrt{2\pi\tilde{V}_{t+1}^{(i)}}}\exp\left\{-\frac{1}{2}\frac{\left[R_{t+2} - (\mu - \frac{1}{2}\tilde{V}_{t+1}^{(i)})\right]^2}{\tilde{V}_{t+1}^{(i)}}\right\} \tag{102}$$

for $i = 1, ..., N$, and we then normalize the weights as usual by

$$\tilde{w}_{t+1}^{(i)} = \frac{w_{t+1}^{(i)}}{\sum_{j=1}^{N} w_{t+1}^{(j)}}. \tag{103}$$

5.4.3 Step 3: Resampling

As we now have our set $\tilde{w}_{t+1}^{(i)}$, we can view this as a discrete probability distribution of $\tilde{V}_{t+1}$, hence we can resample from it. Let the ordered particles be defined as $\tilde{V}_{t+1}^{(i)}$, then the corresponding cumulated sum of the ordered weights $\tilde{w}_{t+1}^{(i)}$ yields the discrete, step-function CDF corresponding to the empirical distribution belonging to V_{t+1}. We can use a basic multinomial resampling scheme to sample from $\tilde{w}_{t+1}^{(i)}$. After this, the filtering for period $t+1$ is done, and we go back to step 1 for period $t+2$ and simulate based on our resampling at $t+1$.

5.5 Parameter Estimation

We now proceed by performing PMCMC to estimate our model parameters, which is based on the theory we presented in Chapter 3.7. Our parameter vector consists here of $\Theta = (\kappa, \theta, \sigma, \rho)$. We need to calculate the probability

$$\alpha = \min\left\{1, \frac{f(\mathbf{y}_{1:T}|\Theta_j^*)p(\Theta_j^*)/\pi_j(\Theta_j^*|\Theta_{j-1})}{f(\mathbf{y}_{1:T}|\Theta_{j-1})p(\Theta_{j-1})/\pi_j(\Theta_{j-1}|\Theta_j^*)}\right\} \tag{104}$$

$$= \min\left\{1, \frac{f(\mathbf{y}_{1:T}|\Theta_j^*)}{f(\mathbf{y}_{1:T}|\Theta_{j-1})}\right\}, \tag{105}$$

which is the probability for accepting a new parameter vector Θ_j^* from the proposal distribution $\pi_j(\Theta_j^*|\Theta_{j-1})$ at iteration j. $p(\Theta)$ is our prior for Θ, which we choose to be Gaussian, which we also choose for the proposal. The symmetric property of the Gaussian distribution allow for the simplification in (105), making the α computation much more straightforward. Here f is the total likelihood, with the latent states integrated out, meaning we are just using the product of the unnormalized weights, yielding

$$f(\mathbf{y}_{1:T}|\Theta) = \prod_{t=1}^{T}\left\{\frac{1}{N}\sum_{i=1}^{N} w_{t+1}^{(i)}\right\}. \tag{106}$$

As we also don't care about $\mathbf{X}_{1:T}$ here, it allows for some slightly simpler notation, but it should be clear that we are still doing the same as in Algorithm 4. In our case, we are running a slightly modified version of MCMC, as we are using stochastic differential equations to form our proposal distribution. This is known as the *pCN* method, and can be studied in greater detail in [21]. This method differs only slightly, as the proposal becomes of AR(1) type instead of a random walk. We compute log-likelihoods and take the sum, meaning we get the same expression as in (88), making the α computation out to be

$$\alpha = \min\left\{1, \exp\Big[l(\mathbf{y}_{1:T}|\Theta_j^*) - l(\mathbf{y}_{1:T}|\Theta_{j-1})\Big]\right\}, \tag{107}$$

where $l(\cdot)$ denotes the sum of the log-likelihood function, i.e., the total log-likelihood.

5.5.1 Step 1: Initialization

We start off with setting initial values for our parameters. We have a Gaussian prior with the following means: $\kappa_0 = 100, \theta_0 = 0.04, \sigma_0 = 2, \rho_0 = -0.5$, and then we define Θ as a matrix of size KxP, where K is number of iterations for our MH-algorithm, while P is number of parameters. Recall we still keep μ fixed as the observed mean of returns. We run the SMC filter one time with $\Theta_0 = (\kappa_0, \theta_0, \rho_0, \sigma_0)$, and compute $l(\mathbf{y}_{1:T}|\Theta_0)$.

5.5.2 Step 2: Sampling and Updating

For $j = 1, ..., K$ we run our SMC algorithm for $N = 200$ particles and $t = 1, ..., T$, where $T = 4027$. The choice of number of particles has been justified in [13] as a selection between computational complexity and a Markov chain that mixes well. By applying the pCN method, we are here sampling $\Theta_j^* \sim N(\sqrt{1-\beta^2}\Theta_{j-1}, \beta^2\mathbf{\Sigma})$, where $\mathbf{\Sigma}$ is the covariance matrix for the parameters. We adjust $\mathbf{\Sigma}$ with β to obtain an acceptance rate within the ideal interval $[23.4\%, 44\%]$ for exploring and exploiting the posterior distribution[22]. An acceptance rate too high might lead to poor exploration of the posterior, while if too low, it might not lead to enough samples from the important regions. When sampling Θ_j^* we also introduce certain restrictions for the proposed parameters. As θ and σ are variances, we make sure these do not get negative values, while ρ is kept in the interval $[-1, 1]$. After the $T'th$ iteration we calculate (106) given Θ_j^*, compute α and compare it with our random uniform value $u \sim U(0, 1)$, and if $u < \alpha$ we set $\Theta_j = \Theta_j^*$, else $\Theta_j = \Theta_{j-1}$. If we accept we also update our likelihood such that $l(\mathbf{y}_{1:T}|\Theta_j) = l(\mathbf{y}_{1:T}|\Theta_j^*)$, else $l(\mathbf{y}_{1:T}|\Theta_j) = l(\mathbf{y}_{1:T}|\Theta_{j-1})$.

5.5.3 Step 3: Parameter Inference

As the MH-algorithm has ran $K = 10^5$ times, we can now compute our estimated parameters. We discard the first 10^4 samples as burn-in, and having j rows we estimate

$$\widehat{\Theta} = (\hat{\kappa}, \hat{\theta}, \hat{\rho}, \hat{\sigma}) = \frac{1}{K}\sum_{j=1}^{K} \Theta_j. \tag{108}$$

5.6 Results

Proceeding with the algorithm described in 5.4, we set our initial parameters according to the values found by Christoffersen et al.[15], and convert to daily values. After running the particle filter we convert back to annualized values, and all parameters will be expressed as such. We choose the ONEN model for this experiment, meaning $a = b = 1$ in (92). The reason for choosing this model, is because it appears to be good for picking up on volatility spikes. As the SQR model, the ONE model and the 3/2 model are all famous volatility models, the thought is to try out a lesser applied model. As seen in [15], the SQR and SQRN models seem to be less sensitive, while the 3/2 and 3/2N models more sensitive. We first look at the spot volatility, which we have filtered on a daily basis. We are plotting the annualized daily filtered spot volatility path $\sqrt{\tilde{V}_t}$ in Figure 18.

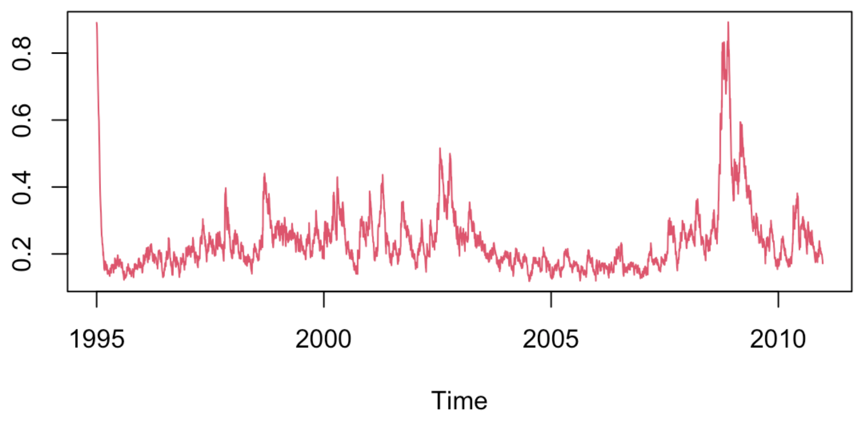

Figure 18: Daily filtered spot volatility $\sqrt{\tilde{V}_t}$ from 1995 through 2011 for the ONEN model with $N = 200$ particles.

We see from Figure 18 that the volatility during the financial crisis is clearly apparent. The spike here is where the highest value is reached, with about 88% volatility in annual terms. The filtered volatility also appears to be clearly heteroscedastic, which goes against the assumptions made by Heston for the SQR model. We also see that our filter has picked up on the volatility around the turn of the millennium, so our filtered spot volatility corresponds well with our data in Figure 14. We also see from Table 1 the first four moments for the filtered spot volatility and the likelihood. What is noticeable here is a high kurtosis, which is understandable as we have included the financial crisis in our data.

In Figure 20 we plot the distributional properties of the filtered spot volatilities (left column) and the natural logarithms of the filtered variances (right column). The top row shows the QQ plots, daily changes against the volatility (or log variance) in the middle row, and finally in the bottom row we display the daily absolute changes against the volatility (or log variance). We pick up several interesting observations here. We see on the QQ plots that data is non-Gaussian for both $\sqrt{\tilde{V}_t}$ and $\ln(\tilde{V}_t)$, but the latter is slightly more normalized apart from the highest values. From the bottom row, we see that the volatility is heteroscedastic for $\sqrt{\tilde{V}_t}$, but it appears more homoscedastic for $\ln(\tilde{V}_t)$. Based on the plots here, it seems like the ONEN model does not work as well as we would like, when modelling volatility in data including the financial crisis. In all plots, it appears as the data is normally distributed for $\ln(\tilde{V}_t)$, except for those values appearing around the time of the financial crisis.

Moving on, we look at the results from performing parameter estimation. We first plot histograms of the parameters in Figure 21, where we are saving every 10'th value after burning the first 10^4. The acceptance rate received here is about 40,8%, which is a little bit higher than desired, but still within

Filtered volatility 1995-2010

Mean	Std. dev.	Kurtosis	Skewness	Likelihood
0.20	0.12	12.21	2.48	12697.67

Table 1: First four moments of the filtered annualized volatility $\sqrt{\tilde{V}_t}$ and the likelihood with $N = 200$ particles.

the interval specified in 5.5.2. For reference, the estimated parameters for the ONEN model in [15] are $\kappa = 133.9347, \theta = 0.0560, \sigma = 2.4188$ and $\rho = -0.7559$. Clearly, only κ comes anywhere close to any of their reference values. What is worth noting, is that for all models where $a = 1$, meaning that an extra variance factor is included in the drift term, the value for κ is remarkably larger than for the other models in [15]. This might indicate that including this extra variance in the drift term is not optimal when estimating parameters in SV models. We then plot the path of the estimated parameters in Figure 22. The only parameter we can see some clear convergence for is θ. There might possibly be some convergence for κ and σ, while ρ is just linearly increasing. In Table 2 we are displaying the means and standard deviations for all estimated parameters in annualized units.

Parameter estimates 1995-2010

	$\hat{\kappa}$	$\hat{\theta}$	$\hat{\sigma}$	$\hat{\rho}$
Mean	103.479	0.132	0.233	-0.123
Std. dev.	14.756	0.005	0.092	0.148

Table 2: Parameter estimates for $\hat{\Theta}$ by PMCMC with $N = 200$ particles and $K = 10^5$ iterations.

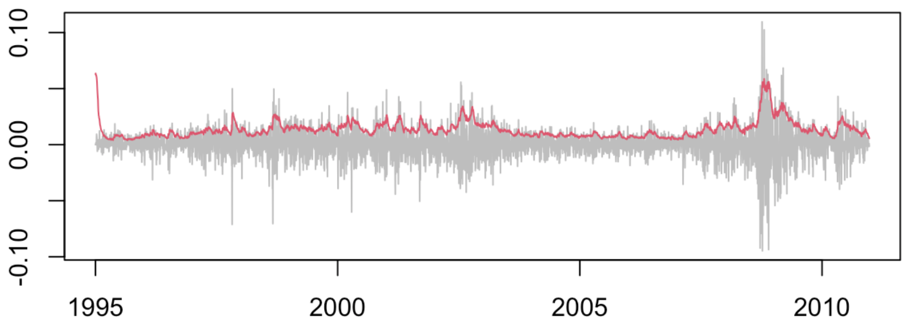

Figure 19: Log returns plotted against the filtered volatility $\sqrt{\tilde{V}_t}$ as in Figure 18. $\sqrt{\tilde{V}_t}$ is scaled down for comparison.

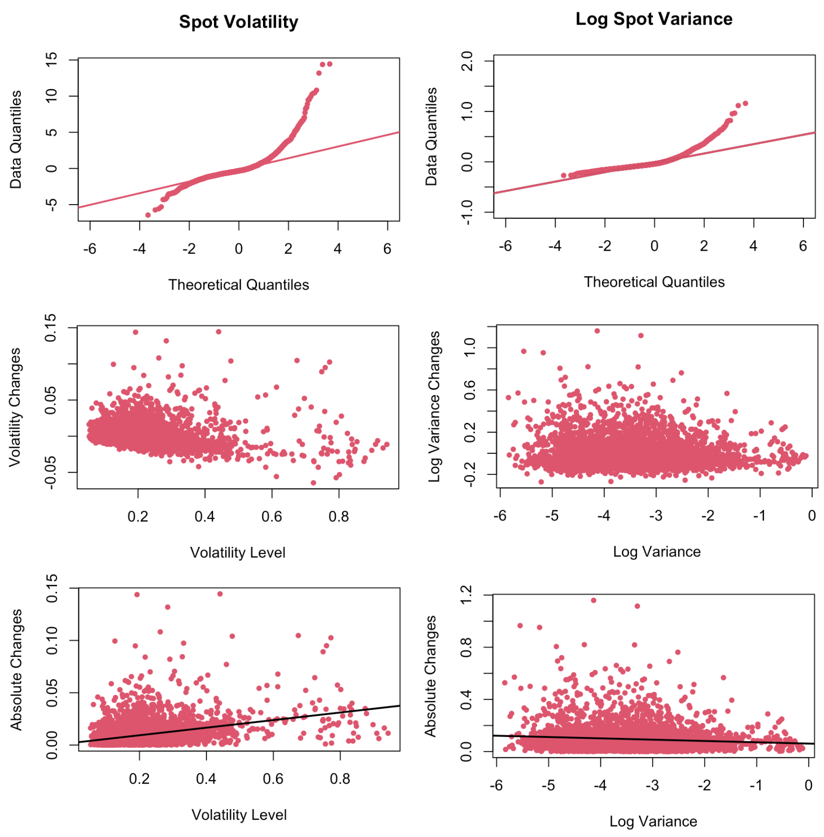

Figure 20: Using the filtered $\sqrt{\tilde{V}_t}$ (left column) and $\ln(\tilde{V}_t)$ (right column), the top row shows the quantiles of the daily changes. The middle row shows the daily changes against the daily levels of volatility. The bottom row shows the absolute daily changes against the daily levels, with a regression line.

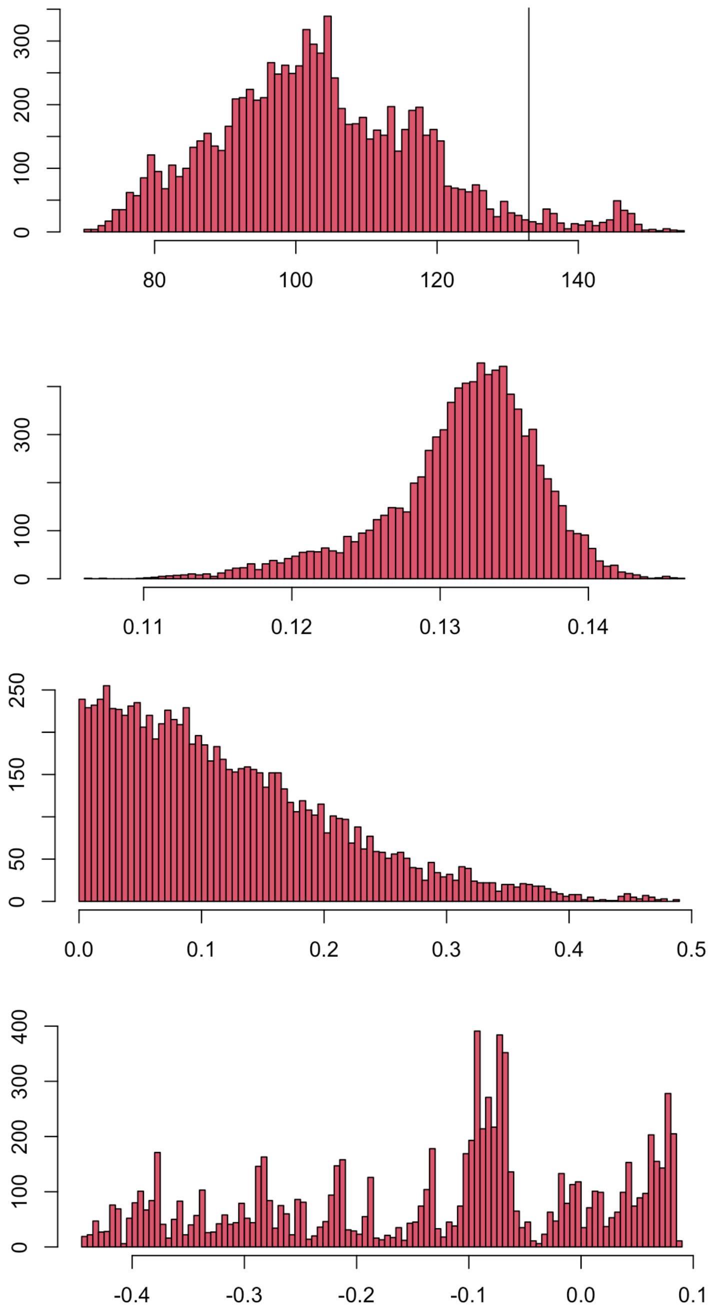

Figure 21: Histogram of estimated parameters by PMCMC with $N = 200$ particles and $K = 10^5$ iterations, for $\hat{\Theta} = (\hat{\kappa}, \hat{\theta}, \hat{\sigma}, \hat{\rho})$ respectively. All parameters are annualized, and reference value highlighted for κ.

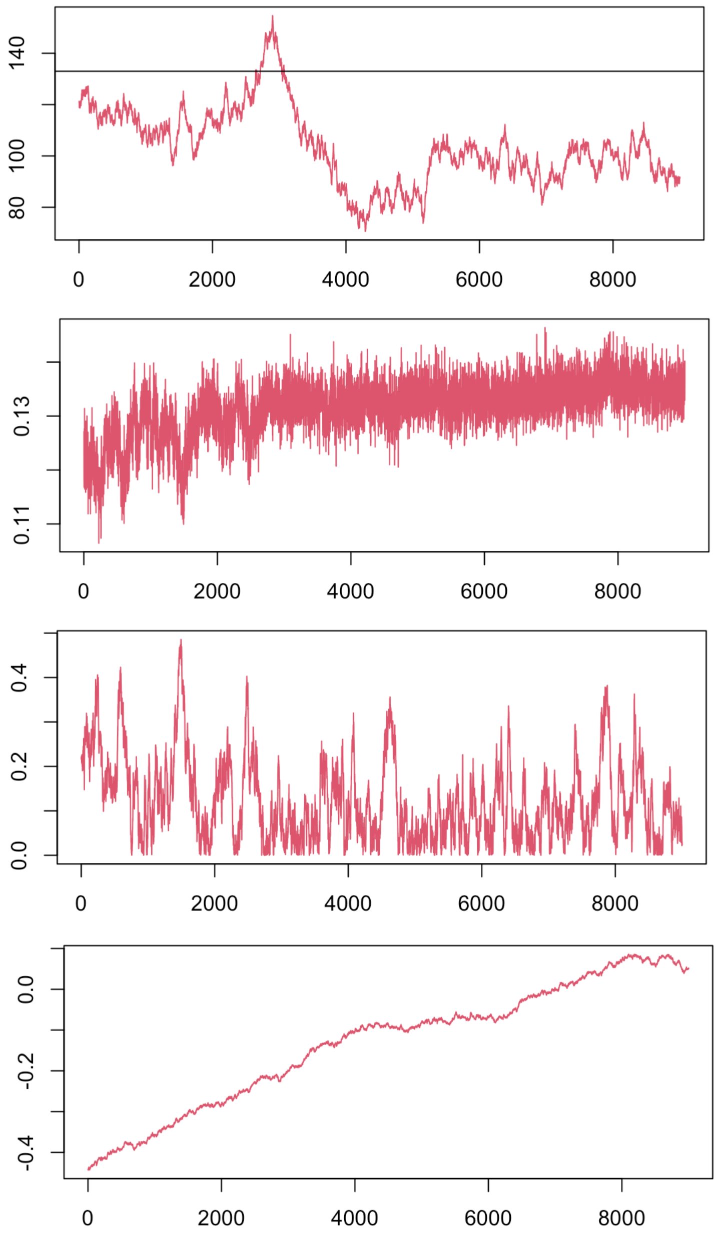

Figure 22: Path of estimated parameters by PMCMC with $N = 200$ particles and $K = 10^5$ iterations, for $\hat{\Theta} = (\hat{\kappa}, \hat{\theta}, \hat{\sigma}, \hat{\rho})$ respectively. All parameters are annualized, and reference value highlighted for κ.

6 Discussion and Future Work

As mentioned, there is strong evidence that our filtered values are not distributed Gaussian with homoscedastic variance when applying the ONEN model, which were the implications for the SQR model. As seen in Figure 20, the large jumps in volatility from the financial crisis appears to skew our filtered log values, as we only get close to normal log variance, and close to homoscedastic log variance changes. Our parameter estimates also clearly differ from our reference values found in [15]. The likely reason for this, is that we are including the financial crisis in our data set, while Christoffersen et al. (2010)[15] are only observing returns for the period 1996-2004. This should naturally increase the values for θ and σ. However, the results only showcase this for the former, while the latter decreases. Also, as we are using a Gaussian prior, this might explain some of the different outcomes. The inclusion of the financial crisis might as well make the PMCMC method dependent on more iterations to obtain convergence, e.g., 10^6 might be necessary. This would be desirable, but computational aspects makes this many iterations very time consuming. Thus, it was unfortunately not feasible for this thesis. When displaying the filtered volatility in Figure 18, we see how the SIR filter works well, as it displays similar volatility levels up to 2004 as found in [15], but also picks up the higher levels around 2007-2008.

In this thesis, we have implemented a basic layer for one stochastic volatility model, but as we have shown, we could have also implemented the standard SQR model, the ONE model and the 3/2 model for comparison, to see if any of these would have fared better. Implementing the standard SQR model has been done several times in the literature, but especially the ONE model would have been interesting to apply to our data set, as this gave the best results on the data set applied in [15]. As seen in [18], they apply a more heavily parameterized model with a data set more similar to ours. When applying it to the SQR model, they also conclude that several of their parameters significantly differ from much of the existing literature. Extending our model with more parameters would also have been a possibility with more time on hand. Adding jumps to the SQR model is also done in both [15] and [18]. This is achieved by adding a Poisson jump counter N to the model, with jump size J distributed normally. We can add jumps to both returns and variance, which would be particularly interesting as we have data including the financial crisis, as jumps in return and variance is a way of coping with volatility spikes. Another natural next step when including jumps, would be to evaluate option prices by the use of the filtered volatility, which has been done in both the mentioned articles. This is interesting, as we would attempt to price options by taking the stochastic dynamic of the volatility into account, as opposed to the well-known Black-Scholes model, where the volatility is fixed constant.

Finally, we could have went with a different discretization method. Another possible approach, would be to discretize as done by Chan and Joshi (2010)[23]. The idea then, is to discretize more efficiently by applying either a *GammaQE* scheme or a *Double Gamma* scheme for simulating the volatility.

7 Conclusion

We have in this thesis introduced some of the theoretical framework that makes up sequential Monte Carlo methods and then showed how it can be applied to solving real-life problems. We began by formulating the problem we are facing, before proposing both analytical and probabilistic solutions. We have described importance sampling, and the shortcomings of this method, leading to sequential importance sampling. Then we have described how the SIR filter combines this with bootstrapping to make a robust and efficient particle filter that can be applied to numerous problems for estimating unobservable states. Furthermore, we also show how particle filters can be combined with Markov chain Monte Carlo to design efficient models for parameter estimations in high dimensions.

We first showcase a benchmark experiment with the SIR filter, where the goal is to estimate unknown states generated by an AR(1) model. We saw how we got accurate estimates when comparing our results with those from the Kalman filter, and especially we saw how the increase in particles helped in creating even more precise estimates. Moreover, we performed MLE for one parameter with good results. This experiment displayed in a simple manner how well the SIR filter works when we have a model that connects our observed states to our hidden states, as well as the key differences between the Kalman filter and particle filters. Despite the Kalman filter not always being applicable, it is the right choice when the constraints are met. There is no need to over complicate if we can apply the Kalman filter to our model.

Then we conducted the main experiment for this thesis. We introduce Heston's stochastic volatility model, which has a direct link between the spot price and the underlying volatility for a financial asset. We proceed with a slightly modified version of this on observed return data from the S&P 500 index, to gain a different perspective on volatility modelling. By applying a model that considers an extra variance factor in the drift term, we compare this with more well used models as the SQR model and the ONE model. Our results are quite interesting. First, when we only run one iteration of the SIR filter, we see that we estimate the spot volatility quite well. The most volatile periods are clearly apparent. Second, when we move to parameter estimation, our results demand more interpretation. The speed of mean reversion parameter, κ, seems to be slightly lower than our reference value. This is reasonable, as we use a data set with more volatility, it is natural that it might take more time to return to the long term mean value. The unconditional variance θ gives a good estimate also, as it is naturally a bit higher than for a data set for the period 1996-2004. The two final parameters differ more from the reference values, and are harder to explain. We therefore make two conclusions; either the model has multiple possible solutions, or the inclusion of the extra variance factor yields poor results when accounting for extreme volatility. The results in Figure 20 support the latter. It's also hard to argue that the underlying spot price and the instantaneous change in variance would have such a weak relationship as described by ρ.

The aforementioned limitations and future research possibilities for this thesis are several, as mentioned in the previous chapter. We could extend our model with more parameters to account for more variables in the market. More computational power would allow us to run PMCMC for more iterations to obtain stronger convergence, and option pricing could also be implemented.

To make final conclusions, we confirm that the use of SMC methods belong in financial econometrics and more specifically volatility modelling. To explore more alternatives to the standard Heston model is key for building more robust volatility forecasting models as well as option pricing models, and remains left for future research. Even the most skilled market analyst have a hard time forecasting volatility, and we have seen how crucial this is in stabilizing both the domestic and global economy.

8 Bibliography

[1] N. J. Gordon, D. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," in *IEE Proceedings F (Radar and Signal Processing)*, 1993. [Online]. Available: https://api.semanticscholar.org/CorpusID:12644877

[2] A. Doucet, N. de Freitas, and N. Gordon, *An Introduction to Sequential Monte Carlo Methods*. New York, NY: Springer New York, 2001. [Online]. Available: https://doi.org/10.1007/978-1-4757-3437-9_1

[3] B. Ristic, S. Arulampalam, and N. J. Gordon, in *Beyond the Kalman Filter: Particle Filters for Tracking Applications*, 2004. [Online]. Available: https://api.semanticscholar.org/CorpusID:60500010

[4] V. E. Beneš, "Exact finite-dimensional filters for certain diffusions with nonlinear drift," *Stochastics*, vol. 5, no. 1-2, pp. 65–92, 1981. [Online]. Available: https://doi.org/10.1080/17442508108833174

[5] F. Daum, "Exact finite-dimensional nonlinear filters," *IEEE Transactions on Automatic Control*, vol. 31, no. 7, pp. 616–622, 1986.

[6] F. E. Daum, "Beyond Kalman filters: practical design of nonlinear filters," in *Signal and Data Processing of Small Targets 1995*, vol. 2561. SPIE, 1995, pp. 252–262.

[7] J. Geweke, "Bayesian Inference in Econometric Models Using Monte Carlo Integration," *Econometrica*, vol. 57, no. 6, pp. 1317–1339, 1989. [Online]. Available: http://www.jstor.org/stable/1913710

[8] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, 2000. [Online]. Available: https://doi.org/10.1023/A:1008935410038

[9] W. Gilks, S. Richardson, and D. Spiegelhalter, "Markov Chain Monte Carlo in practice," 1996.

[10] J. MacCormick and A. Blake, "A Probabilistic Exclusion Principle for Tracking Multiple Objects," *International Journal of Computer Vision*, vol. 39, no. 1, pp. 57–71, 2000. [Online]. Available: https://doi.org/10.1023/A:1008122218374

[11] M. K. Pitt and N. Shephard, "Filtering via Simulation: Auxiliary Particle Filters," *Journal of the American Statistical Association*, vol. 94, no. 446, pp. 590–599, 1999. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/01621459.1999.10474153

[12] W. R. Gilks and C. Berzuini, “Following a moving target—Monte Carlo inference for dynamic Bayesian models,” *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 63, no. 1, pp. 127–146, 2001. [Online]. Available: https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/1467-9868.00280

[13] C. Andrieu, A. Doucet, and R. Holenstein, “Particle Markov Chain Monte Carlo Methods,” *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 72, no. 3, pp. 269–342, 05 2010. [Online]. Available: https://doi.org/10.1111/j.1467-9868.2009.00736.x

[14] S. Malik and M. K. Pitt, “Particle filters for continuous likelihood evaluation and maximisation,” *Journal of Econometrics*, vol. 165, no. 2, pp. 190–209, 2011.

[15] P. Christoffersen, K. Jacobs, and K. Mimouni, “Volatility Dynamics for the S&P500: Evidence from Realized Volatility, Daily Returns, and Option Prices,” *The Review of Financial Studies*, vol. 23, no. 8, pp. 3141–3189, 04 2010. [Online]. Available: https://doi.org/10.1093/rfs/hhq032

[16] S. L. Heston, “A Closed-Form Solution for Options with Stochastic Volatility with Applications to Bond and Currency Options,” *The Review of Financial Studies*, vol. 6, no. 2, pp. 327–343, 1993. [Online]. Available: http://www.jstor.org/stable/2962057

[17] H. Albrecher, P. Mayer, W. Schoutens, and J. Tistaert, “The little Heston trap,” *Wilmott*, no. 1, pp. 83–92, 2007.

[18] A. Dufays, K. Jacobs, Y. Liu, and J. Rombouts, “Fast Filtering with Large Option Panels: Implications for Asset Pricing,” *Journal of Financial and Quantitative Analysis*, p. 1–32, 2023.

[19] C. Klüppelberg, A. Lindner, and R. Maller, “A continuous-time GARCH process driven by a Lévy process: stationarity and second-order behaviour,” *Journal of Applied Probability*, vol. 41, no. 3, p. 601–622, 2004.

[20] B. Eraker, “MCMC Analysis of Diffusion Models With Application to Finance,” *Journal of Business & Economic Statistics*, vol. 19, no. 2, pp. 177–191, 2001. [Online]. Available: https://doi.org/10.1198/073500101316970403

[21] S. L. Cotter, G. O. Roberts, A. M. Stuart, and D. White, “MCMC Methods for Functions: Modifying Old Algorithms to Make Them Faster,” *Statistical Science*, vol. 28, no. 3, pp. 424 – 446, 2013. [Online]. Available: https://doi.org/10.1214/13-STS421

[22] G. O. Roberts and J. S. Rosenthal, “Optimal scaling for various Metropolis-Hastings algorithms,” *Statistical Science*, vol. 16, no. 4, pp. 351 – 367, 2001. [Online]. Available: https://doi.org/10.1214/ss/1015346320

[23] J. H. Chan and M. S. Joshi, "Fast and accurate long stepping simulation of the Heston stochastic volatility model," *Available at SSRN 1617187*, 2010.