UiO : Department of Mathematics
University of Oslo

Sequential Monte Carlo and twisted state space models

Twisting models to reduce variance

Peder Nørving Viken

Master's Thesis, Spring 2021

Abstract

Modeling time series and systems that exhibit an evolution in time is of interest in several fields, and one class of models that can be used for modeling such systems is state space models. One of the most common tools for inference in non-linear and non-Gaussian state space models is sequential Monte Carlo, also known as particle filters, which uses importance sampling and the sequential structure of the model.

When considering state space models, it is also possible to consider twisted state space models, which are defined by a sequence of functions transforming the transitions and emission of the state space model. Several quantities of interest are identical for the original model and the twisted model, thus we can use particle filters to estimate these quantities in the twisted model, and obtain an estimate of the quantities in the original model. The reason the twisted models are of interest is that we can obtain better estimates of these quantities with the twisted models than with the original model, and when considering likelihood estimation there is an optimal sequence that can be used to obtain zero variance estimates of the marginal likelihood. However, this sequence is not obtainable in practice, thus we result in using an approximation.

We propose a simple method for creating an approximation of this optimal sequence and see how this method works when considering simulated data. We also demonstrate how the twisted models can be used for both parameter estimation and smoothing and is a viable alternative to using for instance the traditional bootstrap filter on the original model.

Forord

Jeg ønsker først å takke min veileder Geir Storvik for all hjelp og støtte jeg har fått gjennom arbeidet med denne oppgaven.

Jeg ønsker også å takke Mamma, Pappa, Theo og hele resten av familien for all støtte de siste 23 årene.

En takk rettes også til alle som sitter på lesesal 801 for alle lunsjpauser og gode diskusjoner. En spesiell takk gis til Lars og Ingrid for alle samtalene vi har hatt om inferensteori og alt annet mellom himmel og jord.

Takk også til alle som ikke står nevnt over, men har støttet, motivert og hjulpet meg, jeg er evig takknemlig.

Contents

Introduction

Modeling time series and systems that exhibit an evolution in time is of interest in several fields, and one class of models that can be used for modeling such systems is state space models. The simplest such model is the linear Gaussian state space model where the Kalman filter introduced by Kalman (1960) and related methods can provide analytical solutions to problems such as filtering and smoothing. However, this is not possible for most models and we use numerical methods instead. Most common here are Monte Carlo methods, and in particular sequential Monte Carlo. Within the framework of sequential Monte Carlo, one topic of interest is particle filters that can provide numerical approximations to amongst other the filtering and smoothing problem as well as estimates of the marginal likelihood of the model.

In Chapter 1 we describe the state space model which is the overarching model we consider for the entirety of this thesis. Discussions on time series modeling and state space models can be found in e.g. Tsay and Chen (2018) and Harvey (1989). Here we discuss amongst others why these models are of interest and the basic goals of inference. The primary goals of when considering state space models are filtering, smoothing, and parameter estimation, where the latter two will be our main focus in Chapter 3 and Chapter 4.

In Chapter 2 we first discuss Monte Carlo methods in general and how these can be used to solve integrals numerically. Monte Carlo methods are a very common tool in statistics and an introduction to the topic can be found in e.g. Givens and Hoeting (2013) and Rizzo (2007). When considering state space models we encounter integrals on a specific form which we can approximate using sequential Monte Carlo, and we discuss how this can be done with examples. This chapter serves as an introduction to sequential Monte Carlo and provides some of the tools required for inference in state space models. We also introduce the bootstrap particle filter (Gordon, Salmond and Smith, 1993) which can be used for inference in state space models. There are several works such as Creal (2012) and Doucet and Johansen (2009) which give an introduction to sequential Monte Carlo. There are also different variations of the particle filters which differ in how they do importance sampling. Variations here include lookahead schemes (Lin, Chen and J. S. Liu, 2013), and the auxiliary particle filter, see e.g. Pitt and Shephard (1999).

In Chapter 3 we discuss twisted state space models and in large part follow the work of Guarniero, Johansen and Lee (2017). The twisted models are defined based on a sequence of functions $(\psi_1(x_1), \ldots, \psi_T(x_T)) \equiv \boldsymbol{\psi}$ that transform a state space model, and we show that the marginal likelihood of the original model is identical to the marginal likelihood of any twisted model. Thus we are able to estimate the marginal likelihood of the original model using a twisted model. For the purpose of estimating the marginal likelihood, there is a specific sequence $\boldsymbol{\psi}^{\star}$ which is optimal in the sense that it can be used to obtain a zero variance estimate of the marginal likelihood. This sequence is also useful for smoothing as it can be used to obtain samples from the exact smoothing density. However, the exact optimal sequence is not attainable in practice, and we discuss how we can obtain an approximation of the functions in this sequence. When basing the twisted model on a good approximation of the optimal sequence we show through simulation experiments that the estimate of the marginal likelihood obtained has lower variance than estimates from the original model. This makes the twisted models useful as they can be used to obtain an estimate of the marginal likelihood with low variance, provided we can obtain a sufficiently good approximation of the optimal sequence. Methods for obtaining approximations of this optimal sequence can be found in Guarniero, Johansen and Lee (2017) and Heng et al. (2020). These methods for obtaining an approximation are quite similar and both take an iterative approach for determining the approximation. Based on these approaches we propose a new different setup for determining an approximation of the optimal sequence which is primarily suited for one-dimensional problems. In particular, we start with creating an approximation of the optimal functions at time T, $\psi_T^{\star}(x_T)$, and create a recursive approximation for the rest of the sequence. Each of the approximations is restricted to a class Ψ such that we obtain a twisted model we are able to work with. This is similar to the iterative procedure, but our proposed setup removes the iterative steps and aims to obtain a good approximation in a single step, i.e. passing through the recursion only once. This makes our proposed setup easy to implement in practice, and the results from using this method on a simple state space model show that it can be used to obtain low variance estimates of the marginal likelihood.

In Chapter 4 we look at several examples with the simple method we proposed for approximating the optimal sequence, and how it performs on a simple state space model. We also look at factors that influence the approximation of the optimal sequence and the estimates of the marginal likelihood obtained. When restricting the approximations to a class Ψ we observe that it is key that this class contains good approximations of a large number of functions in order to get the best possible approximation of each of the optimal functions. We also observe that even using a crude approximation of the optimal sequence can lead to variance reduction in the estimate of the marginal likelihood, compared to an untwisted model. We exploit this for parameter estimation with particle marginal Metropolis-Hastings (Andrieu, Doucet and Holenstein, 2010) where we have to compute the marginal likelihood for every proposed set of parameters. We compute this marginal likelihood using twisted models, but rather than creating an approximation of the optimal sequence for each proposed set of parameters, we show that it can be sufficient to use the same approximation for all sets of parameters and still get variance reduction in the estimates of the marginal likelihood. In Section 4.6 we look at how the twisted model can be used for smoothing. We demonstrate how using an approximation of the sequence $\boldsymbol{\psi}^{\star}$, which was optimal for estimating the marginal likelihood, can be used to define a twisted model which samples trajectories from the smoothing density. This use of the twisted models is inherently different from using the twisted models to estimate the marginal likelihood. Whereas when using the twisted models for estimating the marginal likelihood we aimed to obtain a low variance estimate of a single quantity, when considering smoothing we aim to sample from the smoothing density. This provides a estimate of the smoothing density itself which can be used to obtain estimates of several quantities of interest.

In Chapter 5 we discuss and summarize our results related to twisted models. We also discuss other topics of interest related to twisted models and approximating the sequence $\boldsymbol{\psi}^\star$ which we did not cover in the previous sections, and could be of interest for future work. Finally, we discuss some of the differences between using twisted models and different particle filters for inference in state space models. We also discuss how the work we have done is different from some of the other work that has been done in relation to twisted models.

All the numerical results and figures present in the examples in this thesis are created using python (Van Rossum and Drake, 2009) and open source libraries.

CHAPTER 1

Concerning state space models

1.1 Use of state space models

Modeling systems that exhibit an evolution in time is common in several fields such as finance, engineering, and physics to name a few. Here we consider a setting in which we cannot observe the system of interest directly, and we have to rely on indirect observations of the system.

A general framework for modeling such systems is state space models, where we introduce a sequence of latent variables that represent the state of the underlying system at discrete points in time. Each of the latent variables in the sequence will depend on some or all the latent variables preceding it, and estimating these variables can be used to provide information about the underlying system. The state space model framework has proven to be successful and has applications in several fields. One instance of this is stochastic volatility models which are discussed by e.g. Kim, Shepherd and Chib (1998), and another example in which state space models have been used is target tracking as seen in J. S. Liu, Chen and Logvinenko (2001). A different application lies within the field of epidemiology and considers how an epidemic process spreads in a population. This is discussed by among others Dukic, Lopes and Polson (2012) and Kucharski et al. (2020).

When we cannot observe the system directly, the state space model makes use of indirect observations of the system to perform inference on the latent variables. This is done by introducing a separate sequence of random variables representing the observations at each point in time. These observations are then assumed to be dependent on some or all the latent variables representing the underlying system preceding it. This gives the interpretation as indirect observations of the system and allows for estimation of the state through the latent variables conditional on the observations.

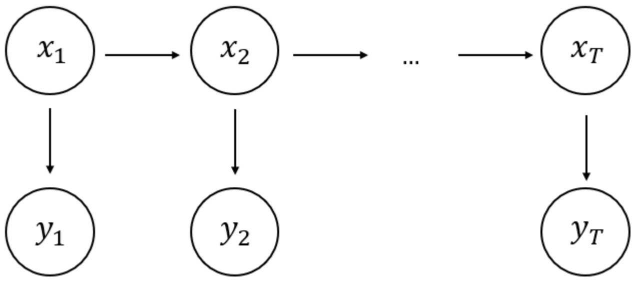

Figure 1.1: Structure of a simple state space model. We have transitions between the states x_t, and for each state we have a corresponding observation y_t.

State space models deal with the task of modeling the state of the underlying system and the observations separately. This breaks the task of modeling the system into two parts, modeling the state transitions, and modeling the observations. A major benefit of this is when the state space represents a process from e.g. physics, modeling the state transitions essentially become a physics problem. This can often be easier to deal with than modeling the states based on the previous observations directly. This also allows us to efficiently deal with changes in the system. If we expect that there is a change in the process that drives the underlying system at a specific point in time, this can easily be accounted for by changing how we model the transitions beyond this point without changing the model for the previous times.

Another benefit of state space models lies in the interpretability of the models. Looking at Figure 1.1, given an initial estimate of a latent variable, it is quite easy to predict the concurrent observation and the next state. Additionally, in the state space formulation, every component has a clear purpose with a direct interpretation and is much less of a black box than for instance neural networks. Other methods for modeling time series such as autoregressive models, which are discussed in introductory books on time series such as Shumway and Stoffer (2017), deals with the time series in a different manner by modeling each observation directly from the previous ones. Interpreting such models is somewhat harder than interpreting the state space model since we cannot use the state and observation structure in the same manner.

As a simple example to illustrate the benefit of the model structure in state space models, consider a setting where the latent variables represent some underlying signal and we have noisy observations of this signal. Whereas modeling the next observations based on the previous ones provides little insights onto the underlying signal itself, it may also be difficult to model since the relation between the observations is not always clear. Instead by using the state space model we can split this into two parts, one which handles the underlying signal itself, and one which deals with the noisy observations. If the process which drives the signal is known, the task of modeling this is relatively straightforward, and simultaneously modeling noisy observations is also a relatively simple task. Hence the state space model allows us to model the system using two separate processes. Additionally, with the state space model, we get estimates of properties related to the underlying signal naturally from the latent variables, something which can be difficult in an autoregressive model.

1.2 Structure of state space models

A state space model consists of a sequence of observations $(y_1, \ldots, y_T) \equiv y_{1:T}$ and a sequence of latent variables $(x_1, \ldots, x_T) \equiv x_{1:T}$, where each x_i and y_i can be either scalar values or multidimensional vectors. For a state space model, the relations between the latent variables themselves can be described through a transition model, and the relations between an observation and the latent variables can be described through the emission model. Together these form the base of the state space model, which can be seen in Figure 1.1

1. The state/transition model $x_t \sim p_x(\cdot|x_{1:t-1})$.
   This model describes the time evolution of the system we are modeling and how the current state depends on the previous states of the system. In state space models this time evolution can be described by the transition density, for a continuous state variable, giving the probability of the next latent variable conditional on the current and past latent variables. For the first state x_1 there are no previous states, and we simply have the probability density for the state x_1.

2. The emission model $y_t \sim p_y(\cdot|x_{1:t})$.
   This is also known as the observation model and describes how the observations depend on the state of the system i.e. the latent variables. In state space models the relationship between the observations and latent variables representing the state can be described by the probability density (or mass in the discrete case) of the observation y_t conditional on the latent state variables. Importantly each new observation does not depend on any of the previous observations. This is a result of the model structure where the states will contain all the information of the observations, hence there is no need to condition on the observations.

Throughout this thesis, we will refer to these as transition and emission densities which correspond to continuous variables. Note that these can also be probability mass functions for discrete variables, however, we will keep calling them densities for simplicity. The model described here is a very general state space model, where the state x_t can depend on part of or the complete history of the system up until that point. However, the model complexity and the computational cost related to the model increase with the number of the preceding variables a latent variable depend on due to the increasing number of dependencies. For this reason, ideally, each latent variable will only depend on a few of the latent variables preceding it. This gives rise to the simplest setting where we assume that each latent variable only depends on the one directly preceding it and not the entire history. Then the evolution of the underlying system forms a first-order Markov process. This assumption, plus assuming that each observation only depends on the concurrent state of the system, greatly reduce the model complexity. This will not be applicable for all systems of interest due to the intrinsic structure of the systems themselves, but when applicable it gives the following state- and emission models

$$x_1 \sim p_x(\cdot) = \mu(\cdot) \tag{1.1}$$
$$x_t \sim p_x(\cdot|x_{1:t-1}) = f(x_{t-1}, \cdot) \text{ for } t \in 2, \dots, T \tag{1.2}$$
$$y_t \sim p_y(\cdot|x_{1:t}) = g(x_t, \cdot) \tag{1.3}$$

This model structure, when applicable, has seen success with amongst other hidden Markov models (HMM) as a special case of state space models. This model structure is also by far the most common for state space models since it is easy to work with and gives rise to relatively simple models. However, these Markov models are not always able to capture all the details in the transitions, so modeling these transitions as Markov processes is not always ideal. A setting in which this is the case is when the trend for the last few states affects the current state, and D'Amico et al. (2019) lists this as one of the main limitations of Markov models. As an analogy to a physical system, where the state x_t is the position of a moving object, the momentum of an object will try to keep the object moving in the same direction. However, the momentum of an object cannot be computed from a single observation of its position, since it depends on the velocity which is the change in position. Certain systems can exhibit similar effects which can be interpreted as the momentum of a system, and the effect of momentum will be difficult to capture in a first-order Markov model that just depends on the previous state.

One relatively simple method of dealing with this is by expanding the state vector by introducing the state $z_t = (x_{t-k}, \dots, x_t)$ for some appropriate k. The new state z_t can capture the trend of the states x_t, and the transitions between each state z_t can still be Markovian.

A discussion on various aspects of trends for general time series models can be found in Harvey (1989, Chapter 6), which includes a discussion on the deceivingly simple question of what a trend is. The same chapter also discusses seasonal factors and how there can be seasonal fluctuations for longer time series which may influence the analysis.

An additional factor that is not accounted for with this Markov structure is lag in the observations. In certain settings, there may be some lag between when the state changes and when this manifests itself in the observations. One instance of this is when modeling epidemics in which there can be a delay between when an individual is infected and showing symptoms (Kucharski et al., 2020). When this is the case it is not sufficient to relate the observation solely to the current state. Instead, we may require some dependence between the observations and the previous states.

Linear and nonlinear state space models

The simplest model structure occurs when the underlying system is modeled as a Markov chain and the observations only depend on the concurrent state of the system. We call this a Markovian state space model, and within these models, we often differentiate between linear and nonlinear state space models based on the structure of the state and emission model. First, we will look at linear state space models which get their names from having state- and emission models that are linear in the state variable x_t. A major benefit of using these linear models is that they become simple to work with since we only have to deal with linear functions. One way to express these models for a potentially multidimensional state x_t and observation y_t is as follows

$$x_t = \mathbf{H}_t x_{t-1} + b_t + \mathbf{W}_t w_t \tag{1.4}$$

$$y_t = \mathbf{G}_t x_t + c_t + \mathbf{V}_t v_t \tag{1.5}$$

Here $\mathbf{H}_t$ and $\mathbf{G}_t$ are assumed to be known matrices forming linearity in the state transitions and emissions. Further b_t and c_t serve as known constants in the model. Finally, v_t and w_t represent measurement noise, which allows us to use this particular model to e.g. analyze an underlying signal x_t which is affect by some measurement noise such that we observe y_t. Commonly used is a multivariate normal distribution centered at the origin with identity covariance matrix. Thus $\mathbf{V}_t$ and $\mathbf{W}_t$ are related to the covariance for the emission and state models respectively. The combined equations (1.4) and (1.5) form a linear state space model. Other choices are also possible as long as the models are linear in the state variable. Such linear state space models are common since they are fairly easy to work with, and when the noise term is Gaussian we can derive analytical results such as the Kalman filter. Analytical results can also be found when we have discrete latent variables by considering the state space as a discrete Markov chain. Using for instance the EM algorithm, parameter estimation can be performed in this setting, and a general discussion on the EM algorithm can be found in Givens and Hoeting (2013). Furthermore, when considering a discrete state space, the calculations will be much simpler than in the continuous case which permits for summing over the states to obtain estimates in this setting.

When the state- and emission models cannot be expressed or reasonably be approximated as linear functions such as (1.4) and (1.5), we have to consider a wider class of models. The nonlinear state space models allow a wider range of state- and emission structures when compared to the linear models since the restrictions on linearity in the state- and emission models are removed. This increases the flexibility of the model at the cost of increased model complexity, and it has the downside that analytical methods such as the Kalman filter are no longer available.

A benefit, and one of the major reasons for extending to nonlinear state space models, is that the nonlinear models enable modeling of a wide array of systems. Some systems cannot accurately be modeled as a linear model, due to some intrinsic non-linearity in the transition or emission structures. One class of models that exhibit this is stochastic volatility models which are discussed in among others Kim, Shepherd and Chib (1998), and in these models the emissions are nonlinear in the state variable.

1.3 Inference in state space models

Latent variables and state estimation

When considering state space models, we often want to make statements about the underlying system we are modeling. Since we have no direct observations of the latent variables representing the system the goal is to make inference on these latent variables, $x_1, \ldots, x_T$, conditional on the observations $y_1, \ldots, y_T$. This is done through a posterior density of the latent variables conditional on observations, such that we for instance can obtain the probability density of the state x_t conditional on a set of observations.

A common target for inference in state space models is estimating the current state conditional on the observations up until the current time, and a natural estimate here is the expected value $\mathrm{E}[x_t|y_{1:t}]$. This is also called filtering and can be done based on the posterior density $p(x_t|y_1, \dots y_t)$ which is also called the filtering density (Creal, 2012). Computing the filtering density directly by some brute force strategy would be infeasible in most cases, which leads to alternate ways to compute the density. By exploiting the model structure, assuming the state transitions are known, it is possible to compute the filtering density recursively by first computing $p(x_{t-1}|y_1, \dots, y_{t-1})$ and updating this using Bayes theorem to get $p(x_t|y_1, \dots y_t)$. An instance of this is the Kalman filter, which was introduced in Kalman (1960) and discussed in among other Tsay and Chen (2018), and can be used to obtain analytical solutions. However, the scope of models that it can be applied to is limited to linear Gaussian models which limit the applicability of this method. The filtering problem is quite common since it can be performed online, which is often desirable as it easily allows for the inclusion of new observations. Furthermore, there is not only estimating the state x_t that is desired. Of interest is sometimes estimates of other functions of x_t, which includes the variance and percentiles of the filtering density.

In practice, the filtering density is obtained recursively. In particular, as we will see in Section 2.2, given the filtering density at time $t-1$ this can be updated to give the filtering density at time t which leads to a recursive definition. One of the main reasons we are able to obtain analytical solutions with the Kalman filter is that all the calculations are relatively simple since all the densities involved are Gaussian and can be parameterized by an expected value and a variance, which are sufficient for describing the whole density. Hence in the Kalman filter, we only need to determine the updates for the parameters of the expected value and variance from the inclusion of a new observation. For more general filtering densities we require different methods for inference. For these models, we may not have a suitable parameterization and we have to update an estimate of the filtering density $p(x_t|y_{1:t})$ itself rather than a set of parameters.

Another common target for inference is smoothing where we consider the posterior density $p(x_t|y_1, \dots y_T)$, with $T > t$ to distinguish from the filtering problem. Compared to the filtering discussed above, the smoothing density takes into account future observations giving the smoothing estimates $\mathrm{E}[x_t|y_{1:T}]$. For this reason, smoothing estimates for the state x_t can only be performed offline, since it requires future observations. Computing the expected values from the smoothing density $\mathrm{E}[x_t|y_{1:T}]$ or estimating the smoothing density itself, is a more involved process than filtering. Much of the reason for this lies in how smoothing depends on future observations, and the recursive approach that is taken when estimating the filtering density must be extended to account for the future observations. For linear Gaussian models, the same approach as the Kalman filter can be taken to give analytical results and is described in among other Tsay and Chen (2018). It is also possible to provide approximations of the smoothing density, one example of this is fixed-lag smoothing where instead of looking at all the future observations we only look at a few, and this will be discussed further in Section 2.4.

It is worth noting that estimation of the latent variables representing the state is not limited to filtering and smoothing. Both are frequently encountered but depending on the problem we might be interested in making statements related to some subset of the latent variables. For instance we could be interested in estimating the full trajectory in a smoothing setting through the posterior density $p(x_1, \ldots, x_T|y_1, \ldots y_T)$ or a specific part of the trajectory $p(x_t, \ldots, x_{t'}|y_1, \ldots y_T)$.

Until now we have only mentioned estimating the states up to the point where we have observations, and we have not focused on states for which we do not have observations. Estimating the states beyond where we have observations, with the posterior $p(x_t, |y_1, \ldots y_T)$ where $t > T$, falls in the category of prediction. A major difference between prediction and the previously mentioned filtering and smoothing is that for the latter two we always had observations. The lack of observations for prediction gives a larger variance in the estimates compared to filtering and smoothing since the observations served as a correction of the state estimate. To illustrate this we consider a setting in which the filter density $p(x_{t-1}|y_{1:t-1})$ is known. The process of computing the filtering density $p(x_t|y_{1:t})$ consists of first predicting the x_t based on the state transitions and correcting this estimate using the emission density which in some sense confines the estimate and lower its variance. For pure prediction the observations are not available, so we do not gain the variance reduction from the observation, and the pure prediction becomes similar to a random walk or Brownian motion based on the transition density in the sense that we simply predict the new state without any means of correcting the predictions.

Again, for smoothing and prediction, the same problem encountered for filtering occurs. Computing the full posterior density analytically is infeasible in most cases, which leads to alternative approaches for inference. Whereas the Kalman filter and related approaches work only in certain settings, or at best via Taylor approximations for non-linear models, Monte Carlo methods have proven to be an effective tool for inference and are applicable for a wide selection of models. In the setting of state space models, sequential Monte Carlo (SMC) is widely used and has proven to be an effective tool for inference, and will be discussed in greater detail later in Chapter 2.

Parameter and likelihood estimation

We have not made any statements about model parameters, and essentially assumed that the parameters of the model are known. Including the parameters in the state- and emission models yield

$$\begin{aligned} x_t &\sim p_x(\cdot|x_{1:t-1}) = p_x(\cdot|x_{1:t-1}, \theta) \\ y_t &\sim p_y(\cdot|x_{1:t}) = p_y(\cdot|x_{1:t}, \theta) \end{aligned}$$

Here θ is a multidimensional vector of all the parameters of the model, which was omitted and presumed known in the previous section. In a setting where some or all of the parameters are unknown, which is often the case in practice, estimating the unknown parameter is required in order to obtain filtering and smoothing estimates of the states. In addition, the model parameters themselves can also be of interest since they say something about the transition probabilities in the model and the underlying system itself. This can for instance be how the transitions in the model behave, and understanding this process is key for creating good estimates of the states.

Key to several methods of parameter estimation is accurate estimates of the marginal likelihood $p(y_{1:t}|\theta)$ which gives the likelihood of the known observations. The marginal likelihood is of interest since it allows us to compare sets of parameters and can be expressed as

$$p(y_{1:t}|\theta) = \int p(x_{1:t}, y_{1:t}|\theta) dx_{1:t} \tag{1.6}$$

Where $p(x_{1:t}, y_{1:t}|\theta)$ is the joint probability of $x_{1:t}$ and $y_{1:t}$ conditional on the model parameters θ. In most practical situations the marginal likelihood, equation (1.6), will be tedious and impractical to compute analytically. This is in part due to the complexity of the integral and the model structure for large t when more terms are added with each new observation. For the linear Gaussian model analytical results are obtainable since equations are always on a form we can deal with, which leads to analytical expressions for the marginal likelihood (Tsay and Chen, 2018). In general, this will not be possible as the expressions will become more complicated, hence similar analytical results are not available except for special cases.

As mentioned, the likelihood estimates are often used in parameter estimation methods. These methods include Metropolis-Hastings setups where the likelihood is required to determine the acceptance ratio for new proposals. An alternative is direct maximum likelihood parameter estimation which requires evaluating the likelihood function. We will go into further details on this in Section 2.5, and an overview for parameter estimation based on sequential Monte Carlo can be found in Kantas et al. (2015).

Brief summary of inference in state space models

In this chapter, we introduced two main objects of interest in regard to inference in state space models. These were state estimation and parameter estimation. We started with looking at state estimates, and we introduced the filtering and smoothing estimates which allow us to estimate the latent variables in the model. This allows us to gain insight into the system which produced the observation, in a different manner to what is possible with e.g. autoregressive models. We also introduced parameter and likelihood estimation in the context of state space models. These are key as they provide insight to the model itself, beyond the general structure. However, we have yet to introduce methods for how we can obtain estimates of these quantities, other than the Kalman filter which as mentioned is limited to linear Gaussian models. Both for non-linear and non-Gaussian models, obtaining estimates of the filtering and smoothing density is difficult without numerical tools as the quantities are complex. We are able to deal with the linear Gaussian setting in large part due to the simple model structure, which allows us to recursively move through the model to produce the estimates. When we aim to estimate the state-, parameter- and likelihood more generally we will retain the recursive approach taken in the Kalman filter. The major difference will be that the analytical results will be replaced with numerical approximations and rather than updating the parameters giving the expected value and variance, we instead update a non-parametric estimate of the densities themselves.

For state space models the most common numerical approximations are based on Monte Carlo methods, and in general Monte Carlo methods have proven to be successful for estimating quantities related to different probability densities and are a common tool in statistics. The methods of interest for state space models are sequential Monte Carlo, which are also known as particle methods. These particle methods will be discussed more Chapter 2, but first, we will spend some time on Monte Carlo methods in general.

CHAPTER 2

Sequential Monte Carlo

2.1 Numerical integration and Monte Carlo methods in general

Basics of Monte Carlo

Monte Carlo methods are a common tool in statistics and Sequential Monte Carlo (SMC) is widely used for inference in state space models. In many cases where we cannot compute the expected value or other quantities related to a probability distribution analytically. In these cases, Monte Carlo methods are a useful tool that can be used to create an estimate of these quantities based on a sample from the distribution of interest. As long as we can provide enough samples, either from some direct simulation technique or more involved methods such as Markov Chain Monte Carlo, these samples can be used to create a Monte Carlo estimate of the desired quantity. Instead of jumping in at the deep end with SMC, we start by giving a more general introduction to Monte Carlo methods, some of the concepts required for SMC, and how to use stochastic simulation to our benefit.

Monte Carlo estimates are based on the fact that the sample mean is a consistent estimator for the expected value of a distribution. This follows from the weak law of large numbers, under reasonable conditions on the mean variance of the distribution. It can be shown, see e.g. Casella and Berger (2002, theorem 5.5.2), when we have an independent sample $X_1, \dots, X_n$ with $\mathrm{E}X_i = \mu$, $\mathrm{Var}\ X_i = \sigma^2 < \infty$, and $\bar{X}_n = 1/n \sum_{i=1}^n X_i$, then

$$\Pr(|\bar{X}_n - \mu| \geq \epsilon) = \Pr((\bar{X}_n - \mu)^2 \geq \epsilon^2) \leq \frac{\mathrm{E}(\bar{X}_n - \mu)^2}{\epsilon^2} = \frac{\sigma^2}{n\epsilon^2}. \tag{2.1}$$

Clearly the right hand side of (2.1) goes to zero as n goes to infinity for every $\epsilon > 0$ as it is proportional to n^{-1}. This shows that the sample mean $\bar{X}_n$ converges in probability to the expected value of the distribution. Thus, the sample mean will be a consistent estimator for the expected value, and this is the basis of Monte Carlo methods in statistics. In practice, this means that if we can obtain a large sample from a distribution, we can use the sample mean as an estimate for the expected value which will converge to the true expected value as the number of samples goes to infinity. Estimates of other quantities can also be obtained since many quantities of interest can be expressed through expected values. This includes for instance the variance which can be expressed using the second moment as $\mathrm{Var}\ (X) = \mathrm{E}[X^2] - \mathrm{E}[X]^2$.

Monte Carlo methods are often introduced as a method for numerical integration since the expected value of a continuous random variable can be expressed as an integral over its density. Recall that the expected value of a continuous random variable with density $f(x)$ can be written as

$$\mu = \mathrm{E}[X] = \int_a^b x f(x)\,\mathrm{d}x$$

with appropriate limits of integration a and b. When computing the Monte Carlo estimates in practice what is done is approximating the above integral as the sample mean of a sample from the density $f(x)$. Since the sample mean is a consistent estimator this will converge to the true expected value as the number of samples increase. Other numerical methods for approximating the same integral such as quadratures can also be applied. However, in higher dimensions using quadratures to handle multiple integrals is not always feasible, making Monte Carlo methods preferable. This is because, in high dimensional settings, the convergence rate is generally better for the Monte Carlo estimates than for quadratures. This is briefly mentioned in Givens and Hoeting (2013, Chapter 5.4.3), making Monte Carlo integration preferable in higher dimensions.

Monte Carlo estimates can also be used to estimate the expected value of functions of random variables. Given a sample $X_1, \ldots, X_n$ from the density $f(x)$ we have, for some function h under reasonable conditions, the following Monte Carlo estimate

$$\hat{\mu}_{MC} = \frac{1}{n}\sum_{i=1}^{n} h(x_i) \to \mu_h = \mathrm{E}[h(X)] = \int_a^b h(x) f(x)\,\mathrm{d}x \tag{2.2}$$

again with appropriate limits of integration a and b. This motivates Monte Carlo methods as a simple way to estimate the expected value of a function $h(x)$ and estimate the integral (2.2). This is also a natural method for constructing approximations of integrals that are not inherently related to a probability density since we can multiply and divide the integrand by an appropriate density which we sample from.

When the probability density we sample from represents a physical process the individual Monte Carlo samples can be seen as a realization of that process. This allows for the use of Monte Carlo methods to analyze the behavior of complex systems by tracing the path a sample takes. This idea is not only applicable for state space models but can also be applied to other systems and has been used for the Ising model in statistical physics see e.g. Walter and Barkema (2015).

Importance Sampling

The Monte Carlo estimate is based on a sample from the probability distribution of interest but in many cases, a sample cannot be obtained directly from the distribution of interest. In such cases where we are unable to sample directly from a probability distribution, a common tool is importance sampling to obtain a sample, which additionally can be used to obtain variance reduction in the estimates. The core idea of importance sampling is to weigh a sample from a different distribution to mimic a sample of the distribution we are interested in.

Consider a target density $f(x)$ we want to sample from. For simplicity we assume that x is one-dimensional, but the same idea works in higher dimensions. Importance sampling works by creating an appropriate importance function $q(x)$ and relies on the following

$$\mu = \int_a^b h(x)f(x)\,\mathrm{d}x = \int_a^b h(x)\frac{f(x)}{q(x)}q(x)\,\mathrm{d}x = \int_a^b h^\star(x)q(x)\,\mathrm{d}x. \tag{2.3}$$

Equation (2.3) shows how we can rewrite the integral such that we can use a sample from $q(x)$ to create the Monte Carlo estimate of the new function $h^\star(x) = h(x)\frac{f(x)}{q(x)}$, that will equal the integral we are interested in. The ratio $\frac{f(x)}{q(x)} = w^\star(x)$ is called the importance weight and is used to weigh the sample from $q(x)$ to match that of the target $f(x)$. In practice for a sample $x_1, \dots, x_N$ from the importance function $q(x)$, the Monte Carlo estimate of μ becomes

$$\hat{\mu}_{IS} = \sum_{i=1}^{N} h(x_i)w(x_i). \tag{2.4}$$

Where $w(x_i) = w^\star(x_i)/\sum_{i=1}^N w^\star(x_i)$ are the normalized importance weights. This approach allows us to create a sample from a density we cannot sample from directly. It is worth noting that we only need to be able to compute the target density up to a proportionality constant. This is because it only appears in the importance weights which we rescale. Only requiring the density up to proportionality can be beneficial for amongst other Bayesian posterior densities where the normalizing constant can be quite difficult to compute.

The normalized weights and the sample $x_1 \dots, x_N$ can also be used to create an unweighted sample from $f(x)$ when that is of interest. This can be done by resampling the particles $x_1, \dots, x_N$ with weights equal to the normalized importance weights $w(x_i)$, but this is known to increase the variance of the sample.

When performing importance sampling we need to choose an appropriate importance function $q(x)$ that we are able to sample from. Introductory books on stochastic simulation such as Givens and Hoeting (2013) and Rizzo (2007) both discuss factors for choosing an importance function that come in addition to being able to efficiently generate a sample. The most important of which is that the target $f(x)$ and importance function $q(x)$ must have the same support, i.e. be non-zero in the same regions. If this is not the case and the target is non-zero for values of x where the importance function is zero, we will not obtain any samples from this region even though the target density is non-zero. This will lead to errors in the Monte Carlo estimates as we are not obtaining samples from the entire density. In practice, we want the importance function which we sample from to have heavier tails than the target. This is in part to make sure that we obtain enough samples from the tail of the target density to sufficiently explore the tail of the distribution. The other way around, with $f(x)$ being zero while $q(x)$ is not, will not be as critical. In such setting samples from $q(x)$ will have zero weight in regions where $f(x) = 0$. As a result, these samples do not contribute to the sum (2.4) and will go to waste.

Importance sampling can also be used as a variance reduction technique if an appropriate importance function $q(x)$ is used. The variance of the importance sampling estimator (2.3) is given by

$$\text{Var}\,(\hat{\mu}) = \text{E}[\hat{\mu}^2] - (\text{E}[\hat{\mu}])^2 = \int h^\star(x)^2 q(x)dx - \mu^2 =$$
$$\int \left(\frac{h(x)f(x)}{q(x)}\right)^2 q(x)dx - \mu^2 = \int \frac{h(x)^2 f(x)}{q(x)} f(x)dx - \mu^2.$$

By choosing the importance function $q(x) = h(x)f(x)/\mu$ the variance of the estimator would become zero, though this choice of $q(x)$ implies that we know the value μ that we are estimating, and as a result this choice of $q(x)$ will in practice not be possible. As an alternative Rizzo (2007) suggest using $q(x) \approx |h(x)|f(x)$ for variance reduction, with the absolute value being introduced such that $q(x) \geq 0$.

2.2 Sampling from state space models and Sequential Monte Carlo

We now want to perform inference in state space models and require methods for estimating the filtering density and other quantities of interest. Where the Kalman filter is only applicable for inference in linear Gaussian state space models, Monte Carlo methods are applicable for inference when considering more general models and have shown great success. Using a Monte Carlo setup allows us to consider both non-linear and non-Gaussian models and still obtain consistent estimates of quantities of interest. That is, the estimates will converge to the true expected values as the number of samples increase. This is because we can express several of the quantities of interest for inference in state space models as expected values which we can estimate using a Monte Carlo setup. This includes the expected values of the filtering and smoothing densities, in addition to other quantities related to these densities. Since both the filtering and smoothing densities are known up to proportionality, we can use importance sampling to obtain a weighted sample from these densities and create Monte Carlo estimates of various quantities. For the purpose of illustrating Monte Carlo in the setting of state space models, we assume that we have a very simple model structure where the underlying system can be modeled as a Markov process and the observation at time t only depends on the state of the system at time t. We will consider models on the form

$$x_1 \sim \mu(\cdot) \tag{2.5}$$
$$x_t \sim f(x_{t-1}, \cdot) \text{ for } t \in 2, \ldots, T \tag{2.6}$$
$$y_t \sim g(x_t, \cdot) \tag{2.7}$$

For this model the filtering and smoothing densities can be written as follows

$$p(x_1, \ldots, x_t | y_1, \ldots, y_t) \propto \mu(x_1)g(x_1, y_1) \prod_{i=2}^{t} f(x_{i-1}, x_i)g(x_i, y_i)$$
$$p(x_1, \ldots, x_T | y_1, \ldots, y_T) \propto \mu(x_1)g(x_1, y_1) \prod_{i=2}^{T} f(x_{i-1}, x_i)g(x_i, y_i).$$

It is these densities we primarily want to obtain a sample from. By using importance sampling we can obtain a weighted sample from the desired densities, without calculating the normalizing constants. Creating a proposal density to directly sample from the full filtering or smoothing density, that is $x_{1:t}$ or $x_{1:T}$ conditional on an appropriate set of observations, all at once will be difficult. That is why rather than obtaining a sample $x_1, \ldots, x_T$ directly we can instead exploit the sequential structure of the model in a similar manner to what is done in the Kalman filter. We can first make a sample of the state x_1 conditional on y_1 and update this estimate as we move through the model. This allows for online estimation of the filtering estimates similar to the Kalman filter. Obtaining a sample from the smoothing density requires some more effort and will be discussed further in Section 2.4. A major difference between the Monte Carlo approach and the Kalman filter approach is that using Monte Carlo gives a sample from the density. In contrast for the Kalman filter, we only computed the parameters which fully described the density. Obtaining a sample from these densities can be done using the framework of sequential Monte Carlo, also known as particle filters. Sequential Monte Carlo can also be used to solve more general integrals on the form

$$I = \int h(x_{1:t})\mu(x_1)g(x_1, y_1) \prod_{i=2}^{t} f(x_{i-1}, x_i)g(x_i, y_i) \, \mathrm{d}x_{1:t} \tag{2.8}$$

for appropriate functions μ, f, g, and h. This general form includes amongst other expected values related to both the filtering and smoothing densities as seen above. The marginal likelihood of the model can also be written on this form by setting $h = 1$, a constant function. There are several works, such as Creal (2012) and Doucet and Johansen (2009), that give an introduction to the field of sequential Monte Carlo and how it can be applied to state space models. For the next sections in this chapter we will look at how we can exploit the model structure to sample from the state space model, and how such samples can be used for inference.

Sequential importance sampling

Inference in state space models is based a sample from the density $p(x_{1:t}|y_{1:t}; \theta)$ where $x_{1:t}$ is a sequence of latent state variables representing the state, $y_{1:t}$ is a sequence of observations and θ are the (known) model parameters. That is, we want a sample from the density of the latent variables $x_{1:t}$ conditional on the observations $y_{1:t}$. As mentioned previously, state space models are often complex and rarely allow for direct sampling of the desired density. To get around this, we can employ the ideas from importance sampling to get a weighted sample from the desired density.

The general idea of sequential importance sampling is to divide the density we want to sample from into smaller parts that are easier to handle and use importance sampling for each part. We denote the density we want a sample from at time t as $p_{1:t}(x_{1:t}|y_{1:t})$ which is the joint density of all the latent variables conditional on all the observations up until time t. As mentioned, creating an efficient importance function to sample from for this density will be quite difficult. One reason for this is, in order to efficiently sample when t becomes large, we should include the relation between all the latent state variables since these are modeled like a Markov process, and capturing this structure can be difficult. For sequential importance sampling, it is this Markov structure we aim to exploit to create an efficient sampling setup and we factor the density to get a recursive expression as follows

$$p_{1:t}(x_{1:t}|y_{1:t}) \propto p_{1:t-1}(x_{1:t-1}|y_{1:t-1})f(x_{t-1}, x_t)g(x_t, y_t) \tag{2.9}$$

What is shown in recursion (2.9) is that we do not need to deal with all the $x's$ at once. Expanding this recursion, starting with $p_1(x_1|y_1)$ we can update this density sequentially to get our desired final density at time t. The same idea carries over to the sampling. If we first have a sample from $p_1(x_1|y_1)$ we can update this sample until we reach $p_{1:t}(x_{1:t}|y_{1:t})$. To sample at each time t we aim to use importance sampling, so we also want an importance function that can be factored in a similar manner. Following some of the notation in Creal (2012) the sampling density can be expressed as

$$q_{1:t}(x_{1:t}|y_{1:t}) = q_{1:t-1}(x_{1:t-1}|y_{1:t-1})q_t(x_t|x_{1:t-1}, y_{1:t}). \tag{2.10}$$

The first factor represents the history and path taken by a particle up to and including time $t-1$. In practice when sampling this is known based on the previous samples and Creal (2012) describes this factor as a Dirac measure for each particle. The second factor represent the possible paths forward at time t and the next set of samples is drawn from $q_t(x_t|x_{1:t-1}, y_{1:t})$. With this general expression for the sampling density, the importance weights can be written as the ratio between the density we aim to sample from (2.9) and the importance function (2.10)

$$\begin{aligned}
w_t &\propto \frac{p_{1:t}(x_{1:t}|y_{1:t})}{q_{1:t}(x_{1:t}|y_{1:t})} \propto \frac{p_{1:t-1}(x_{1:t-1}|y_{1:t-1})f(x_{t-1}, x_t)g(x_t, y_t)}{q_{1:t-1}(x_{1:t-1}|y_{1:t-1})q_t(x_t|x_{1:t-1}, y_{1:t})} \\
&\propto w_{t-1}\frac{f(x_{t-1}, x_t)g(x_t, y_t)}{q_t(x_t|x_{1:t-1}, y_{1:t})} = w_{t-1}\tilde{w}_t \\
\tilde{w}_t &= \frac{f(x_{t-1}, x_t)g(x_t, y_t)}{q_t(x_t|x_{1:t-1}, y_{1:t})}
\end{aligned}$$

At each time t, the weights will be scaled from the step at $t-1$ so we only need to compute the scaling factor $\tilde{w}_t$. This allows us to first sample from x_1 and use this to create the sample of x_2 and so on. By weighting the sample based on the observation y_t, we obtain the pairs $\{x_t^i, w_t^i\}_{i=1}^N$ representing a weighted sample of the current state which can be used to make inference on the state x_t. In particular, this sample at time t is a weighted sample of trajectories which corresponds to the filtering density at time t, namely $p(x_1, \ldots, x_t|y_1, \ldots, y_t)$. In practice, we often focus on the part of the trajectory which gives the last state x_t. Alongside the weight, this will be a weighted sample from $p(x_t|y_{1:t})$ which is what we desired to obtain, and it is this sample that can be updated to give the filter estimate when a new observation is included. While this setup gives the full trajectory conditional on all the observations up to time t, this setup is rarely used to provide smoothing estimates of the state. This is due to issues with degeneracy, and methods for smoothing and determining the full trajectory will be discussed in Section 2.4. This can all be combined into algorithm 1, which is a general algorithm for a particle filter.

Algorithm 1: General Sequential Importance Sampling for state space models

```
1 Set $t = 1$, and sample $x_1^i \sim q_1(\cdot|y_1)$ for $i \in (1, \dots, N)$
2 Compute weights $w_1^i \leftarrow \frac{\mu(x_1^i)g(x_1^i,y_i)}{q_1(x_1^i|,y_1)}$ and increment $t$
3 Sample $x_t^i \sim q_t(\cdot|x_{1:t-1}^i, y_{1:t})$ for $i \in (1, \dots, N)$
4 Compute weights $w_t^i \leftarrow w_{t-1}^i \frac{f(x_{t-1}^i,x_t^i)g(x_t^i,y_t)}{q_t(x_t^i|x_{1:t-1}^i,y_{1:t})}$ and increment $t$ and
  return to step 3
```

This algorithm relies on the proposal density $q_t(x_t|x_{1:t-1}, y_{1:t})$ which in practice often will be reduced to $q_t(x_t|x_{t-1}, y_t)$ for simplicity (Creal, 2012). This proposal draws a the state at time t both based on the previous state x_{t-1} and the current observation y_t. A proposal density on this form admits the probability density of the state x_t conditional on y_t and x_{t-1}, equation (2.11), as a proposal. This choice of proposal will minimize the variance of the importance weights, and a proof of this can be found in Doucet, S. Godsill and Andrieu (2000)

$$q_t(x_t|x_{t-1}, y_t) = p(x_t|x_{t-1}, y_t). \tag{2.11}$$

This proposal is optimal in the sense that the variance of the weights is minimized, and zero if exactly (2.11) is used as a proposal. This is desired since with zero variance all the weights will be equal, and we sample from the exact density.

While (2.11) is optimal in the sense of variance reduction on the weights, in practice it suffers from drawbacks as illustrated by Doucet, S. Godsill and Andrieu (2000). One of these drawbacks is that sampling from the density $p(x_t|x_{t-1}, y_t)$ directly will often be difficult, especially when comparing to a proposal such as $p(x_t|x_{t-1})$. The second drawback is lies in the weight updates. For the optimal proposal the weights updates in algorithm 1 become $w_t^i \leftarrow w_{t-1}^i p(y_t|x_{t-1}^i)$ where the updates $p(y_t|x_{t-1}^i)$ has no general analytical expression but can be computed for specific models.

Part of the difficulty with the optimal proposal lies in the conditioning on y_t, and dropping this term gives the proposal $p(x_t|x_{t-1})$. This is the basis for the simplest algorithm for sequential importance sampling, namely the bootstrap particle filter from Gordon, Salmond and Smith (1993). In the bootstrap filter the proposal density is simply the state transition density, i.e. $q_t(x_t|x_{1:t-1}, y_{1:t}) = f(x_{t-1}, x_t)$ for the model we consider here. Equation (2.12) shows that with this proposal the weight updates is the probability density of observing y_t given the current state x_t. A practical interpretation of this is that when given a set of particles and weights at time t, the new latent variables at time $t + 1$ will be estimated via the transition density and weighted based on how likely they are to give the observed value y_{t+1}.

$$w_t \propto w_{t-1} \frac{f(x_{t-1}, x_t)g(x_t, y_t)}{f(x_{t-1}, x_t)} = w_{t-1} g(x_t, y_t) \tag{2.12}$$

In practice, the bootstrap proposal density is far easier to implement than the optimal proposal (2.11). In large part, this is because sampling from the transition density of the model $f(x_{t-1}, x_t)$ is easier than sampling from the optimal proposal and the weight update shown in (2.12) is very simple and will always be available. This makes the bootstrap particle filter quite versatile as we only need to sample from the transition density $f(x_{t-1}, x_t)$ which admits a large number of models to be implemented efficiently. Furthermore, as the bootstrap particle filter provides Monte Carlo estimates, general convergence results for convergence of Monte Carlo estimates can be applied and estimates based on the bootstrap particle filter are consistent estimates. Additionally, for sequential Monte Carlo, there are central limit theorems available such as Chopin et al. (2004, theorem 1), which allows us to get a better understanding of the effect from using different proposal densities in the particle filter.

Other approaches also exist and have proven to be successful. Notable amongst these are the auxiliary particle filter introduced by Pitt and Shephard (1999) and a discussion of this method can also be found in among other Johansen and Doucet (2008). Another approach is trying to use the future observations in a lookahead setup (Lin, Chen and J. S. Liu, 2013).

Degeneracy and resampling

A problem with this setup is that when we start with a finite number of particles and have data spanning a large period of time, eventually we will end up with most of the weights being very small and a few weights dominating. This is known as weight degeneracy and will lead to poor estimates unless it is dealt with. To illustrate this problem, consider a setting where at time t we have sampled particles $\{\mathbf{x}_t^i\}_{i=1}^N$ with weights $\{w_t^i\}_{i=1}^N$. If one of these particles turns out to be a bad estimate of the current state and is given a low weight compared to the rest of the particles, the effect of this will propagate to the weight of the next particle. This is because the weights at time $t+1$ are scaled based on the weights at time t. Thus a relatively low weight at time t can lead to a low weight at time $t+1$, and unless dealt with this process will continue until a few particles are dominating. A theoretical result showing this can be found in Doucet, S. Godsill and Andrieu (2000) which states that the unconditional variance of the weights are always increasing such that Var $(w_t) \leq$ Var (w_{t+1}). Thus, when updating the weights, they will be distributed such that their variance is increasing which leads to some particles with higher weight dominating.

The prevailing method for dealing with degeneracy is resampling where the core idea is to replace the poor particles with a low weight with particles that has a higher weight, hence rejuvenating the sample. There are several resampling methods, and a very simple method is to resample the particles based on their weights when a specific criterion is met to get an equally weighted sample. J. S. Liu and Chen (1995) suggests the effective number of samples or effective sample size, given by equation (2.13), as a criterion for when to resample. In practice, it is common to perform resampling only when N_{eff} is lower than some threshold αN where N is the number of particles and α is a predefined constant between 0 and 1 set by the user. A discussion on methods of resampling can be found in among other Doucet and Johansen (2009).

$$N_{\text{eff}} = \frac{\left(\sum_{i=1}^N w_t^i\right)^2}{\sum_{i=1}^N (w_t^i)^2} \tag{2.13}$$

Although resampling handles the weight degeneracy by rejuvenating the sample it will lead to other problems. Doucet and Johansen (2009) and Doucet, S.

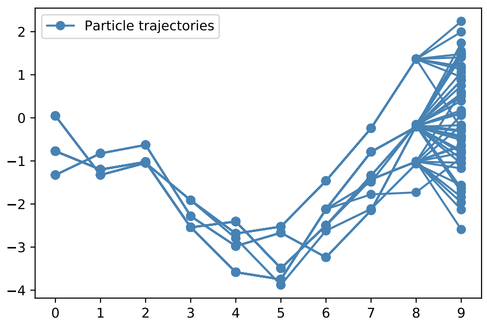

Figure 2.1: Tracing the paths 50 particles from a simple linear Gaussian state space model. The trajectories are created base on algorithm 2 with resampling when $N_{\text{eff}} \leq 0.5N$. Since the whole path is resampled, there are fewer unique particles at the earlier times as only the particles with high weight is kept when resampling.

Godsill and Andrieu (2000) bring this up and it is worth mentioning some of them here. When starting the SMC algorithm all the particles will be independent, but as we resample this will no longer be the case as the resampling process introduces dependencies between the particles. Additionally, rejuvenating the sample will also decrease the diversity of the sample. This comes as a result of when we resample to get the next set of particles, these resampled particles will mostly be based on the particles with higher weights. This reduces the diversity of the sample by discarding the particles in the tail of the density and focusing on those near the mode. Furthermore, resampling algorithms, such as the one provided in algorithm 2 and the *SIS/Resampling Monte Carlo filter* from Doucet, S. Godsill and Andrieu (2000), resample the entire trajectory up to time t when resampling at this time. As a result, these resampling schemes will provide a good Monte Carlo estimate of $\mathrm{E}[x_t|y_{1:t}]$ and the filtering density at time t, while at earlier times we observe many identical particles which will lead to poor Monte Carlo estimates. This is illustrated in Figure 2.1 which shows the trajectories of 50 particles after resampling. This sample will give a good approximation of the filtering density at time $t = 9$, simultaneously we notice that there are fewer unique particles at the earlier times prior to resampling and at those time it will be a poor Monte Carlo estimate.

Including a resampling step in the SMC algorithm is fairly easy. Algorithm 2 gives the bootstrap particle filter with a resampling according to the scheme outlined above. With this setup, particle filtering can still be performed online, and it is very common to include a resampling step to rejuvenate the sample in practice.

Algorithm 2: Bootstrap particle filter with resampling.

```
1 Set $t = 1$, and sample $x_1^i \sim \mu(\cdot)$ for $i \in (1, \dots, N)$
2 Compute weights $w_1^i \leftarrow g(x_1^i, y_1)$ and normalize $w_1^i$
3 If $N_{\text{eff}} < \alpha N$ then resample $\{x_1^i\}_{i=1}^N$ with probability $w_1^i$ and set
    $w_1^i = 1/N$
4 Increment $t$
5 Sample $x_t^i \sim f(x_{t-1}, \cdot)$ for $i \in (1, \dots, N)$
6 Compute weights $w_t^i \leftarrow w_{t-1}^i g(x_t^i, y_t)$ and normalize $w_t^i$
7 If $N_{\text{eff}} < \alpha N$ then resample $\{x_{1:t}^i\}_{i=1}^N$ with probability $w_t^i$ and set
    $w_t^i = 1/N$
8 Increment $t$ and return to step 5
```

2.3 Particle filters for inference in state space models

Having provided the tools for creating a sample from a probability density on the form (2.9), we now want to use this sample for inference in state space models. For the purpose of illustrating the methods, we limit our focus to models on the form used in the previous section, namely the state and emission densities (2.5)-(2.7).

Estimating the state using a particle filter

The filtering problem is common when working with state space models and is the easiest to solve. We can use the particle filters described in the previous section directly to obtain an estimate of the filtering density. Of interest the filtering density $p(x_t|y_1, \dots, y_t)$, and its expected value can be used as an estimate of the state. We will refer to the expected value of the filtering density as the filter estimate, and it is important to make the distinction between the true state x_t^{true} and the filter estimate $\hat{x}_t$. The filter estimate from the particle filters is a Monte Carlo estimate of the expected value of the filtering density, that is $\hat{x}_t \approx \mathrm{E}[x_t|y_1, \dots, y_t]$. On the other hand, we have the true state x_t^{true} that is described by the transitions in the state space model, and this is not known in practice. However, for simulated data which we will consider here, it will be known and is used to obtain the simulated observations. This is an important distinction as the filter estimate $\hat{x}_t$ does not converge to the true state as more particles are introduced. Instead, the filter estimate converges to the expected value of the state conditional on the observations. This may be a good approximation of the true state, and will in part depend on how informative the observations are, the two are distinct quantities. We are also interested in estimating the filtering density itself since this gives more information on the underlying state than the expected value alone.

Using the particle filter described in algorithms 1 and 2 we get at each time t pairs of samples $\{x_t^i\}_{i=1}^N$ and weights $\{w_t^i\}_{i=1}^N$ which together at time t become a weighted sample that can be used to approximate the filtering density $p(x_t|y_{1:t})$. This weighted sample can be used to solve the filtering problem by creating a Monte Carlo estimate of the expected value of the filtering density. Based on the weighted sample, the filter estimate of the state conditional on the observations is given by

$$\mathrm{E}[x_t|y_{1:t}] \approx \hat{x}_t = \sum_{i=1}^{N} w_t^i x_t^i. \tag{2.14}$$

Where w_t^i are the normalized weights of algorithm 2. Filtering has the benefit that it can be computed in an online setting by including a step for computing the filtering estimates in algorithm 2 at each time t. While the Kalman filter gives the analytical expectation of the filtering density, the filtering expected value given by (2.14) is a Monte Carlo estimate and will include some Monte Carlo variance based on the number of particles used. The same weighted sample can also be used to create Monte Carlo estimates of other properties of the filtering density. In general we have that

$$\mathrm{E}[h(x_t)|y_{1:t}] \approx \widehat{h(x_t)} = \sum_{i=1}^{N} w_t^i h(x_t^i)$$

Where again w_t^i are the normalized weights allowing us to estimate other properties of the filtering density such as the variance and the quantiles. Using the weighted sample we can also create a discrete approximation of the filtering density $p(x_t|y_1, \dots, y_t)$. Using the normalized weights, we get the likelihood of observing a particular value of x_t which leads to the following approximation of the density

$$\hat{p}(x_t|y_1, \dots, y_t) = \sum_{i=1}^{N} w_t^i \delta_{x_t^i}(x_t)$$

In this discrete approximation the value of the filtering density zero for $x_t \notin \{x_t^1, \dots, x_t^N\}$ and at x_t^i the density attains the value of w_t^i for $i \in 1, \dots, N$.

Examples of filtering by the use of particle filter

Now having introduced the methodology of filtering with particle filters in state space models, now we want to show some practical examples. In the following examples, we look at simulated data and aim to produce filtering estimates of the states.

Example 1 - Gaussian state process and Poisson emissions

Here we consider a simple one-dimensional example where the state transitions are linear in the state variable and are Gaussian, while the emission model follows a Poisson distribution.

$$\begin{aligned} x_t &\sim \mathcal{N}(\cdot|a + bx_{t-1}, \sigma^2) \\ y_t &\sim \mathrm{Poisson}(\cdot, \lambda = e^{x_t}) \end{aligned}$$

Here $a, b \in \mathbb{R}$, $\sigma \in \mathbb{R}_+$, and λ is the rate parameter of the Poisson distribution. The initial value of the state x_0 is set $x_0 \equiv 0$, such that $x_1 \sim \mathcal{N}(\cdot|a, \sigma^2)$. In this example, we are primarily interested in the filtering problem, that is estimating the states given the observations up to that point. While the state transitions are linear and Gaussian, the emissions are Poisson distributed, thus the Kalman

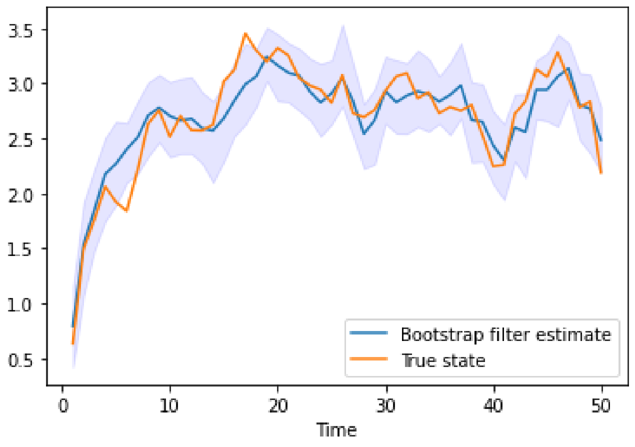

Figure 2.2: Filter estimates of $\mathrm{E}[x_t|y_{1:t}]$ from the bootstrap particle filter with $N = 1000$ particles and resampling when the N_{eff} is less than $0.5N$. The shaded region is between the 0.025 and 0.975 quantiles of the samples which provides the filter estimate.

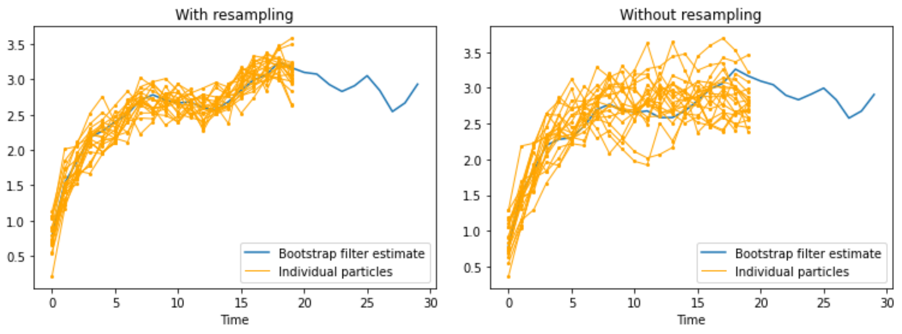

Figure 2.3: Visualizing the trajectory for some of the particles.

filter cannot be applied and we will use the bootstrap particle filter. Here we also have the benefit of knowing the exact transition- and emission densities, so no parameter estimation is required.

For this example, we set $a = 0.85$, $b = 0.7$ and $\sigma = 0.2$, in addition t runs from 1 to $T = 50$. The data is generated by a simulation of the dynamic process and at each step t an observation y_t was simulated yielding a sequence of observations $y_{1:50}$ and true states $x_{1:50}$.

Figure 2.2 shows the filtered estimates for each of the states estimated by use of the bootstrap filter. The bootstrap estimates are based on 250 simulated particles and resampling is based on the weights of each sample. Resampling is performed if the effective number of samples is smaller than half the original number of particles.

We observe that the bootstrap filter estimate of $\mathrm{E}[x_t|y_{1:t}]$ is not identical to the true value of the latent variable x_t which is expected. The filter estimate is able to capture several of the features in the sequence of latent variables,

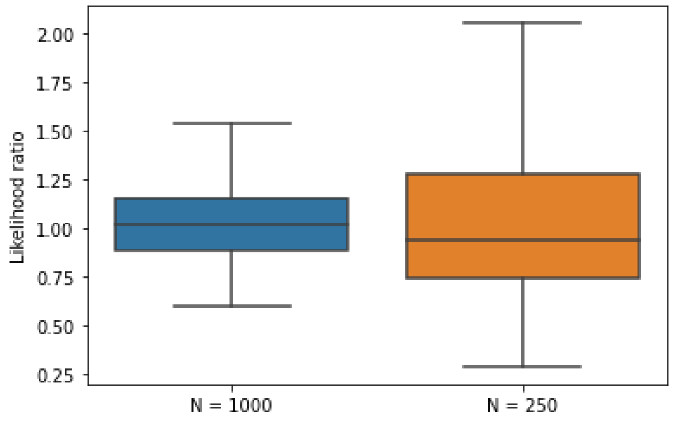

Figure 2.4: Ratio of likelihood between estimates computed with 1000 and 250 particles and the likelihood estimate based on a bootstrap filter with 5000 particles.

motivating the use of the filter estimate as an estimate for the state variables. We also observe that the particles cover the true state at almost every time t. The shaded region contains 95% of the particles, and the true state mostly lies within this region with a few exceptions. The boundaries of the shaded region are created at each time by, independently from the bootstrap filter itself, sampling the filtered particles according to their weights and compute the 0.025 and the 0.975 quantiles of this sample.

Figure 2.3 shows the trajectory of 20 particles from the bootstrap filter with 1000 particles. Here we show two settings, one where no resampling is performed and one that resamples when $N_{\text{eff}} < 0.5N$, to illustrate the effect of resampling. From Figure 2.3 we see that the particles that are never resampled vary more around the expected value than the ones that are resampled. This is because we have resampled the entire trajectory of the particle, and the poor trajectories get discarded in the resampling while the remaining trajectories lie close to the mode. The downside is that as we resample we end up with fewer unique particles at the previous times which we saw earlier in Figure 2.1.

While working with this simple model it will also be beneficial to look at the effect from the number of particles. Having shown the filter estimates, we now look at likelihood estimates and how they vary based on the number of particles. For this, we first compute the likelihood Z_{5000} based on a bootstrap filter with 5000 particles to serve as a comparison, and we compare the effect of 1000 and 250 particles in the bootstrap filter when it comes to the variance of likelihood estimates. The comparison is performed by computing the likelihood 100 times with 1000 and 250 particles and we record Z_{1000}, Z_{250} yielding 100 realizations for each. We then look at the ratio Z_N/Z_{5000} for comparing the likelihood estimates. Figure 2.4 shows boxplots for the two cases we consider. Over 100 repetitions we see that increasing the number of particles reduces the variability in the likelihood estimates. This is expected since increasing the number of particles should reduce the variance in the estimates.

Example 2 - Linear Gaussian model in 2 dimensions

Here we consider a somewhat similar setting to the previous example, but this time for a higher dimensional state space. Additionally, we use a Gaussian emission density so we end up with a linear Gaussian state space model. The model for this example is on the form

$$\begin{aligned} x_t &= Hx_{t-1} + \varepsilon \\ y_t &= Gx_t + \eta \end{aligned}$$

Here ε and η both are multivariate normal distributions, centered at zero with covariance matrix equal to the identity matrix. Additionally we set G equal to the identity matrix, and define the matrix H as a diagonal matrix where the non-zero entries both equals 0.42, that is $h_{1,1} = h_{2,2} = 0.42$. We can consider this example in any number of dimensions, but for simplicity we consider $n = 2$. Again, we are interested in the filtering problem, but now we also have the Kalman filter to serve as an analytical comparison. The data is simulated from the model described above with $x_0 = \vec{0}$, i.e. the zero vector in $\mathbb{R}^2$. We let the time index t run from 1 to 50, which yields a sequence of latent variables $x_{1:50}$ and a sequence of observations $y_{1:50}$ where each $x_t, y_t \in \mathbb{R}^2$ for $t \in 1, \ldots, 50$. We then use algorithm 2 with the inclusion of a step computing the filter estimates.

Figure 2.5 shows a comparison between the bootstrap filter estimates and the values computed from the Kalman filter, and we see that the bootstrap filter estimate has almost converged to the analytical values from the Kalman filter when using 1000 particles. Here we see that the estimates of $\mathrm{E}[x_t|y_{1:t}]$ computed with the bootstrap filter are consistent with those computed from the Kalman filter. This is expected since the filter estimates from the bootstrap filter converge to the analytical values obtained with the Kalman filter. Again, the shaded regions are the 0.025 and 0.975 quantiles for a resampled set of particles. These are found by sampling the particles according to their weight at each time and computing the desired quantiles of this sample, hence this region gives a good indication of the location of the samples at each time t. From this, we see that at each time t the particles vary around the expected value as we expect to see, and we obtain a good estimate of the filtering expected value when looking at all the particles together.

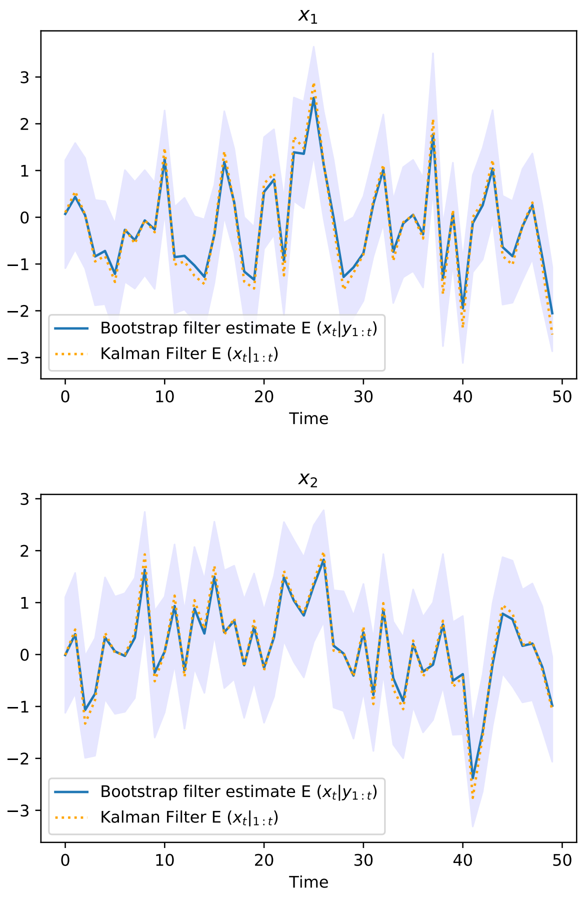

Figure 2.5: Bootstrap filter estimates of $\mathrm{E}[x_t|y_{1:t}]$ and Kalman filter computation of $\mathrm{E}[x_t|y_{1:t}]$ for the two dimensional state x. The top figure shows the first dimension x_1 and the bottom shows the second dimension x_2. Bootstrap estimate is based on 1000 particles with resampling when the effective number of samples is less than 500.

2.4 Smoothing in state space models

So far we have primarily discussed the filtering problem and how to use particle filters to provide estimates of $\mathrm{E}[x_t|y_1, \ldots, y_t]$ and other quantities related to the filtering density. An alternative to the filtering estimates of the states x_t are the smoothed estimates, namely $\mathrm{E}[x_t|y_1, \ldots, y_T]$. Whilst creating filtering is straightforward using particle filters, the process of creating smoothed state estimates is as previously mentioned a bit more involved. In particular, the process of obtaining the filtering estimates could be performed online, and this is not the case for smoothing which requires conditioning on the complete data. In practice, smoothing can be done by first running a particle filter over all the data, and then performing a backward pass over the data to get the smoothed estimates.

The same approach taken for the Kalman filter can be used to provide exact analytical expressions for the expected value and variance of the smoothing density, but its use is limited as it is only applicable to the linear Gaussian models. This Kalman approach provides analytical expression rather than Monte Carlo approximations which makes this approach desirable when it can be applied. A derivation of the Kalman smoother can be found in Tsay and Chen (2018). For problems that cannot be solved by the Kalman smoother, such as problems related to non-linear and non-Gaussian models, we can again use sequential Monte Carlo. We now aim to provide samples from the full smoothing density $p(x_t|y_1, \ldots, y_T)$, and as previously mentioned this process is more involved as it requires the conditioning on the full data. Thus, the simple sequential approach taken for the standard particle filters is not suited for providing samples from the smoothing density.

Even though the standard sequential Monte Carlo algorithm, namely algorithm 1, is not suited for sampling from the smoothing density, it can be used to obtain samples of the full trajectory $x_{1:T}$ conditional on $y_{1:T}$ directly. When considering the full path of the particles and the weights at time T these will give a weighted sample from the smoothing density $p(x_1, \ldots, x_T|y_1, \ldots, y_T)$. When we applied this to the filtering problem, at each time t we focused solely on the state x_t and we ignored the trajectory. The reason for this is that in this specific setting without resampling we end up having particles with very low weight and a few particles dominating. To counter this, we introduced resampling as a method of rejuvenating the sample. This solves the problem of weight degeneracy at time t, but it introduces a new problem. We resample the entire path and end up with few unique particles at the early times as shown in Figure 2.1. As a result, this setup will not provide good estimates of the smoothing density at the early time points due to there being few unique particles. Now we will have a brief look at two other methods which can be used for obtaining particle estimates of the smoothed density.

Fixed lag approximations

One common method for obtaining an approximate solution to the smoothing problem is fixed lag smoothing which is discussed by amongst other Kantas et al. (2015) and Briers, Doucet and Maskell (2010). The idea behind this approach is assuming that the process we are considering has good forgetting properties, that is for the state at time t the observations after time $t + \Delta$ will have little effect on the smoothing density of the state x_t. Thus, we get the following approximation of the smoothing density

$$p(x_{1:t}|y_{1:T}) \approx p(x_{1:t}|y_{1:\min(t+\Delta,T)}). \tag{2.15}$$

That is, for some appropriate $\Delta \in \mathbb{N}$ we can approximate the full smoothing density by only looking at Δ steps ahead. In practice implementing a fixed lag setup can be done with particle filters and will follow much of the same setup as we described when discussing the filtering problem with particle filters. From the filtering problem we already have the machinery to provide estimates of the filtering density, hence we only need to extend this to include conditioning on observation forward in time. To provide the conditioning on the future observations, a simple method will be to run the particle filter up to time $t + \Delta$.

When no resampling is done after time t, the weights at time $t + \Delta$ alongside the particles at time t can be used to provide fixed lag smoothing estimates at time t.

To use this method in practice we need to determine a suitable Δ to provide the fixed lag estimates. The value of Δ must be sufficiently large so observations at times after $t + \Delta$ do not bring any additional information satisfying (2.15). Simultaneously Δ must be small enough to not introduce issues with degeneracy when looking ahead at time t providing the smoothing estimates. As no resampling is done between times t and $t + \Delta$, the sample will not be rejuvenated in this interval which may cause issues with degeneracy at time t with some particles dominating.

Smoothing using a backward pass

Another simple method, which takes a different approach than the fixed lag setup, is smoothing using a backward pass which is discussed in S. J. Godsill, Doucet and West (2004) and Kantas et al. (2015) as Forward-Backward Smoothing. This approach relies on having filter estimates available for all the states and uses a backward pass over these to provide the samples from the joint smoothing density $p(x_{1:T}|y_{1:T})$.

As the state space model describes how to move forward in time, to provide the smoothing estimate using a backward pass we need to look at how we can pass through the model backwards. This can be done exploiting the Markov structure of the model, and using Bayes theorem we get the following

$$\begin{aligned} p(x_t|x_{t+1:T}, y_{1:T}) &= p(x_t|x_{t+1}, y_{1:t}) \\ &= \frac{f(x_t, x_{t+1})p(x_t|y_{1:t})}{p(x_{t+1}|y_{1:t})} \propto f(x_t, x_{t+1})p(x_t|y_{1:t}) \end{aligned} \tag{2.16}$$

As a result, we can estimate the density $p(x_t|x_{t+1:T}, y_{1:T})$ up to proportionality, with $f(x_{t+1}, x_t)$ which simply is the transition density of the model and $p(x_t|y_{1:t})$ which is the filtering density, which we have from the filtering problem. Similar to how we move forward through the model when solving the filtering problem, a similar approach can be taken moving backwards through the model with the backward recursion (2.16). A simple way of doing this is given in S. J. Godsill, Doucet and West (2004), which reuses the particles produced by the filtering algorithm. This produces trajectories from the smoothing density $p(x_{1:T}|y_{1:T})$, which can be used to create Monte Carlo estimate of amongst other $\mathrm{E}[x_t|y_{1:T}]$ and $\mathrm{E}[x_{1:T}|y_{1:T}]$.

A different method is the two way filter where forward and backward filtering is combined to create the smoothed estimate. This approach is discussed amongst other by Kitagawa (1994) and Kitagawa (1996), and an alternative is proposed by the more recent Briers, Doucet and Maskell (2010) which does this in a slightly different manner. The two filter approach taken by Kitagawa (1996) relies to the backward information filter which can be defined as follows

$$p(y_t, \dots, y_T|x_t) = g(x_t, y_t)\mathrm{E}\left[\prod_{p=t+1}^{T} g(X_p, y_p)|X_t = x_t\right] \tag{2.17}$$

This can then be used to formulate an expression for the full smoothing density. When combining the backward information filter with a prediction based on the regular forward filter we can obtain the full smoothing density up to proportionality as

$$p(x_t|y_1, \dots, y_T) \propto p(x_t|y_1, \dots, y_{t-1})p(y_t, \dots, y_T|x_t). \tag{2.18}$$

This method is a bit more involved than the previous methods mentioned since we now have the inclusion of the backward information filter. We now have to account for the backward information filter which in itself is not trivial and combine these with the filtering estimates. A setup in which this can be done will be discussed further in Section 4.6.

Example 3 - A simple smoothing example

Now we want to have a look at how smoothing can be done in practice, and we will compare a fixed lag setup with smoothing using a backward pass as described in S. J. Godsill, Doucet and West (2004). We will here use a linear Gaussian model so we can compare our estimates with the analytical values obtained from a Kalman smoother. For the Kalman smoother we have implemented algorithm 6.2 from Tsay and Chen (2018). Here we will consider the following model

$$\begin{aligned} x_t &= 0.7x_{t-1} + 0.25 + \varepsilon \\ y_t &= x_t + \eta \end{aligned}$$

where both ε and η are standard normal random variables, i.e. mean zero and variance 1, and from this model we simulate a sequence of 25 observations and latent variables with $x_0 = 0$.

Before looking at the particle approximation of the smoothing density, it is worth comparing the state estimates from the Kalman filter and the Kalman smoother, both of which yield analytical state estimates. Figure Figure 2.6 shows the state estimates from both methods, and these are quite similar for this model. Both can reasonably be used as estimates of the state but the smoothing values require more computational effort to obtain. The difference in the two estimates lies in the additional conditioning on the future states in the smoothing estimates.

Now turning our attention back to the particle approximation, we already know that we can use particle estimates to obtain a good approximation of the filtering expectation, and now we will look at this in the smoothing setting. For the smoothing with a backward pass from we first run a regular bootstrap particle filter with $N = 500$ particles where we resample when $N_{\text{eff}} < 0.5N$. We then use these particles for the backward pass to create 500 trajectories from the smoothing density. For the fixed-lag setup we use $N = 1000$ particles and look $\Delta = 5$ steps ahead. Additionally, we resample when $N_{\text{eff}} < 0.5N$, and any resampling is done before looking ahead. This is because if there is a need to resample, this need will only increase as we look ahead, so removing this need is desirable.

For comparing the results of the two particle methods we look at their expected value. Figure 2.7 shows the estimates of the smoothing expected value compared to the analytical value obtained from the Kalman smoother. We see here that neither method provides a perfect estimate of the expected value of the smoothing everywhere, but both seem to give reasonable estimates of

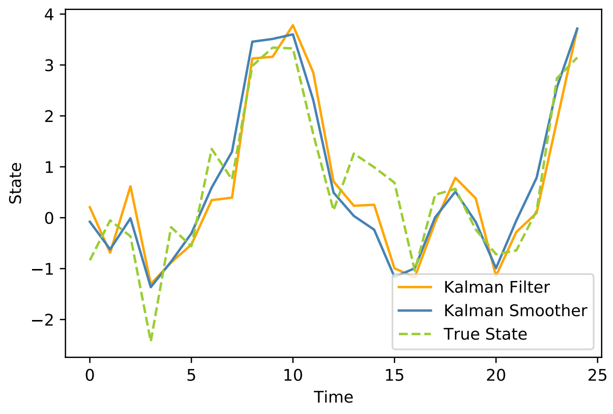

Figure 2.6: Comparison of the state estimate from the Kalman filter and Kalman smoother.

the analytical expected value of the smoothing density. Further, we can look at the quantiles from the particles produced with the two methods and these are shown in figure 2.8. Here we see that the expected value of the smoothing density, obtained via the Kalman smoother, is located in a region covered by the particles at each time t. This indicates that in a setting where we do not know the true expected value, using these particle approximations, we can provide good estimates of the expected value.

Differences between these particle approximations and the analytical value from the Kalman smoother will in part come from these being Monte Carlo approximation and increasing the number of particles will provide better estimates. Additionally, for the fixed-lag approximation, we do not use the full sequence of observations by construction. This may also cause part of the difference between the approximation and the analytical value and using a different value for Δ will provide a different approximation. For the approximation based on the backward pass, this will be heavily dependent on the filtering estimates obtained, as the particles from here come directly from the particles used to approximate the filtering density.

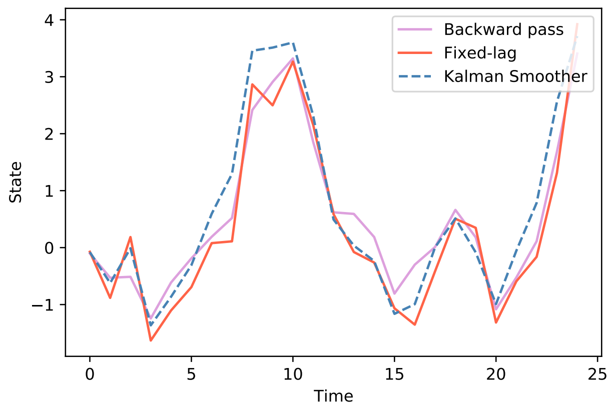

Figure 2.7: Comparison of particle estimates of the expected value of the smoothing density

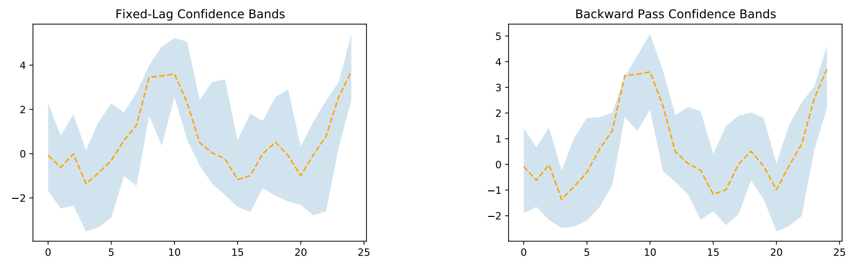

Figure 2.8: Confidence bands for the particle approximations. The boundaries for the shaded region is the 0.025 and 0.975 quantiles for the particles at each time. The orange line in the center is the expected value of the smoothing density as obtained from the Kalman smoother.

2.5 Parameter estimation in state space models

Having so far discussed estimation of the latent variables representing the states, we now move to parameter estimation which is another topic of interest when working with state space models. We again consider the model described by (2.5)-(2.7), where we now include a vector of parameters θ which represents the unknown model parameters. The goal of parameter estimation is, as the name implies, to estimate the unknown parameters θ. This is essential as the values of the parameters provide insights to the model, while also being necessary for obtaining amongst other good estimates of the filtering density. We can separate parameter estimation methods into two main categories, maximum likelihood approaches, and Bayesian approaches.

Closely related to parameter estimation is likelihood estimation and estimating the marginal likelihood (1.6) is key since this allows us to compare different sets of parameters based on the data. The marginal likelihood can be computed using standard particle filters and can be obtained as a byproduct when running the standard bootstrap particle filter without much effort. We will go into further detail on how the marginal likelihood can be estimated in Chapter 3 where we discuss how to provide low variance estimates of the marginal likelihood, but for this section, we will focus on what the marginal likelihood can be used for.

Starting with maximum likelihood approaches, Kantas et al. (2015) lists several methods for parameter estimation in state space models using maximum likelihood approaches. One possible approach is gradient ascent to determine the parameters maximizing the likelihood function, or potentially log-likelihood. This is natural as the general idea for maximum likelihood estimation is to find the parameters the maximize that marginal likelihood function. The likelihood can be computed analytically in linear Gaussian models, while for more general models we can use particle filters for estimating the likelihood. As mentioned, we will go into more details for computing the likelihood in chapter Chapter 3, but a good estimate of the marginal likelihood which easily can be implemented alongside the bootstrap particle filter is

$$p(y_{1:t}|\theta) \approx \prod_{t=1}^{T} \left[\frac{1}{N} \sum_{i=1}^{N} g(x_t^i, y_t) \right]. \tag{2.19}$$

Where x_t^i are particles from the bootstrap filter at time t, and we will use this method for calculating the marginal likelihood in the next example.

For Bayesian approaches to parameter estimation, our interest lies in the posterior $p(\theta|y_{1:T})$ which can be used for inference for the unknown parameters θ. In practice, the joint posterior of the latent variables and unknown parameters, $p(x_{1:t}, \theta|y_{1:t})$, is also of interest since it can also be used for inference on the parameters and simultaneously the latent variables. Perhaps the simplest method for parameter estimation, albeit a fairly naive one, is simply extending the latent variables with the unknown parameters θ in the particle filter. This is by Kantas et al. (2015) called augmenting the state with the parameter, and is an online method of parameter estimation which looks at the combined state $Z_t = (X_t, \theta_t)$ where $\theta_t = \theta_{t-1}$. Viewing this as particles, each particle representing a state is paired with a particle representing the parameters of the model drawn from an appropriate prior distribution. The particles representing the parameters will remain unchanged when moving from t to $t+1$ while the particles representing the state will evolve based on the specified model. This approach has the benefit of being very easy to implement and can easily be implemented alongside the bootstrap particle filter by including a step for dealing with the parameters. This method will degenerate as resampling is done in the particle filter since we start with a finite number of particles representing the parameters, and the process of resampling will reduce the number of unique particles. While the state is rejuvenated by resampling, the same will not be the case for the parameters leading to degeneracy among the particles. To further illustrate this, consider starting with a large sample $\{\theta^i\}_{i=1}^N$ from the prior of the parameters $p(\theta)$. For the first few steps, this will be able to explore the parameter space of the unknown parameters. Each time the particle filters resamples the number of unique samples in $\{\theta^i\}_{i=1}^N$ will be reduced as the pairs of particles representing the state and parameters with low weight will be discarded until we potentially reach the extreme with a single unique particle. For this reason, this method is flawed since the parameter space is not adequately explored at later times with few particles, but it can still be used in some simple cases. A proposed solution to this can be found in J. Liu and West (2001) that propose to rejuvenate the sample by adding random noise to the parameter such that $\theta_{t+1} = \theta_t + \zeta_t$ where ζ_t is sampled from a known mean zero normal distribution. The immediate downside of this is that the model parameters are no longer fixed and additional variability is introduced in the parameter estimates. This aids in the process of exploring the parameter space at later times by diversifying the sample and again will be relatively easy to implement in practice as well as having the benefit of being an online method.

More sophisticated methods for parameter estimation can be done in an offline setting, and one such approach use Markov Chain Monte Carlo to sample from the joint distribution of the parameters and latent state variables conditional on the observations, that is the density $p(\theta, x_{1:T}|y_{1:T})$. One common algorithm for MCMC is the Metropolis-Hastings algorithm which also can be used when dealing with state space models. One such setup is the marginal Metropolis-Hastings sampler which is discussed in Andrieu, Doucet and Holenstein (2010). A general introduction to the Metropolis-Hastings algorithm can be found in Givens and Hoeting (2013, chapter 7), and one of the main parts of this algorithm is computing the Metropolis-Hastings ratio which determines the acceptance probability of the sampler and depends on the proposal density for the MCMC algorithm. Andrieu, Doucet and Holenstein (2010) propose the following proposal density for to determine the jumps in the marginal Metropolis-Hastings sampler to sample from the joint density $p(\theta, x_{1:T}|y_{1:T})$

$$q([\theta', x'_{1:T}|\theta, x_{1:T}]) = q(\theta'|\theta)p(x'_{1:T}|y_{1:T}, \theta')$$

With this choice of proposal we do not need to sample from the joint density of θ and $x_{1:T}$ which may be difficult, and when we solely want to focus on parameter estimation we do not need to make any additional effort for handling the states. This can be seen through the expression for the acceptance probability, which can be written on the following form by rewriting the joint density of θ and $x_{1:T}$

$$\begin{aligned} R &= \min\left(1, \frac{p(\theta', x'_{1:T}|y_{1:T})q([\theta, x_{1:T}|\theta', x'_{1:T}])}{p(\theta, x_{1:T}|y_{1:T})q([\theta', x'_{1:T}|\theta, x_{1:T}])}\right) \\ &= \min\left(1, \frac{p_{\theta'}(y_{1:T})p(\theta')q(\theta|\theta')}{p_\theta(y_{1:T})p(\theta)q(\theta'|\theta)}\right) \end{aligned} \tag{2.20}$$

Where θ is the current parameter values and θ' is the new proposed parameter values. Further $p(\theta)$ a prior for the parameters, $q(\theta'|\theta)$ is the proposal for the new parameters and $p_\theta(y_{1:T})$ is the likelihood for a given set of parameters as given by (1.6). Here we have used that $p(\theta, x_{1:T}|y_{1:T}) = p(x_{1:T}|y_{1:T}, \theta)p(\theta|y_{1:T})$, and the factor containing the states $x_{1:T}$ will cancel out in (2.20). The main difficulty here lies in computing the likelihood which in general cannot be done analytically. An alternative here is using an unbiased estimate based on a particle filter as a substitution, and as a result we require low variance likelihood estimates which will be discussed further in Chapter 3. This can the be used for to create a sample of from the posterior distribution of the parameters conditional on the observations using the Metropolis-Hastings algorithm, and a setup for this can be found in Andrieu, Doucet and Holenstein (2010).

Comparing this Metropolis-Hastings setup with the online methods for parameter estimation, the Metropolis-Hastings setup has a greater computational cost. This is in large part because for each sample from $p(\theta|y_{1:T})$ we have to compute the full likelihood $p_\theta(y_{1:T})$ in order to determine the Metropolis-Hastings ratio (2.20). Computing the marginal likelihood can quickly become computationally expensive, hence providing a large sample from $p(\theta|y_{1:T})$ can be cumbersome using a Metropolis-Hastings setup. Compared to the online approaches previously discussed, an offline Metropolis-Hastings setup for parameter estimation does not encounter the same problems with degeneracy in the sample. Traditionally Metropolis-Hastings setups will be useful in settings with a large number of unknown parameters. This will also be the case here as the algorithm can be constructed to explore the parameter space in an efficient manner.

Example 4 - Estimating parameters

Now we will illustrate parameter estimation in a simple example. Here we will consider a simple linear model were augmenting the state with the parameters is sufficient for providing adequate parameter estimates. Additionally, we will consider a simple particle marginal Metropolis-Hastings setup. Although the latter might be superfluous in this simple setting, it is still worth having a look at. For this example, we will use simulated data based on the following model

$$\begin{aligned} f(x_{t-1}, \cdot) &\sim \mathcal{N}(\cdot; a + bx_{t-1}, 1) \\ g(x_t, \cdot) &\sim \mathcal{N}(\cdot; x_t, 0.5^2) \end{aligned}$$

We set $x_0 = 0$ as well as $a = 0$ and $b = 0.7$, and simulate a sequence of 25 latent variables and observations, $x_{1:25}$ and $y_{1:25}$ respectively. When augmenting the state with the parameters our goal is to create an estimate of the scaling factor b based on the observations, and we assume that $a = 0$. For the particle marginal Metropolis-Hastings we want to estimate both parameters simultaneously.

For creating the parameter estimates by augmenting the state, we use a bootstrap particle filter with $N = 10000$ particles and resample when $N_{\text{eff}} < 0.5N$. As a prior for the parameter, we use a uniform prior on the interval $(0, 1)$ which we know includes the true value, but this should also provide some samples that are quite far away from the true value.

Figure 2.9 shows a histogram of the samples of the parameters at the final time. As we can see, the majority of the samples we are left with are close to the true value, and this provides a parameter estimate $\hat{b} = 0.68$. Furthermore we can look at the spread of the sample in particular we have the 0.025 and 0.975 quantiles of the samples at $q_{0.025} = 0.41$ and $q_{0.975} = 0.96$. As we can see from both the quantiles and the histogram, there is some spread in the samples, but the majority lies close to the true value. It is also worth remembering that this sample started as a uniformly distributed sample and by running the particle filter, we have been able to obtain information about the location of the true state. Additionally, this can be used as a first crude parameter estimate which can be improved upon since it is fairly fast and easy to implement alongside the bootstrap particle filter.

The primary issue with this approach is that the diversity in the sample of the parameters is decreasing and we do not make any effort to rejuvenate

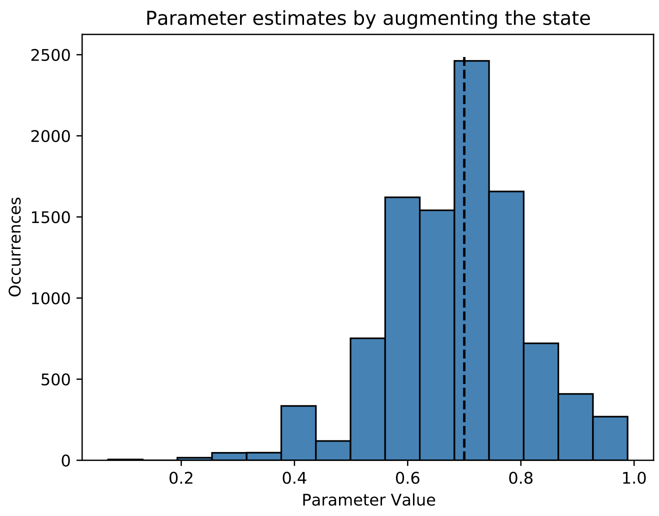

Figure 2.9: Histogram showing the particles representing the parameters at the final time T with the true value shown as a dashed line.

the sample. We started with 10000 particles from the prior of the parameters, and at the final time, we are left with 67 unique particles. This effect would be even more noticeable for longer sequences and if we had more parameters to estimate. This forces us to use a large number of particles in the bootstrap filter so that we are left with enough unique particles to obtain a Monte Carlo estimate of the parameter.

For the particle marginal Metropolis-Hastings setup, we take this a bit further since for estimating a single parameter this method is redundant. Here we want estimates for both a and b for the transitions in the model and with a Metropolis-Hastings setup, we can easily sample from the joint density. Here we follow the particle marginal Metropolis-Hastings setup, and alternate between proposing a sample for b and a, and for both we use a random walk centered at the previous sample with steps that are normally distributed with variance $\sigma = 0.125^2$ for both. As a prior for the parameters we use a uniform prior on the interval $(-0.5, 0.5)$ for a and $(0, 1.5)$ for b. For proposing a new sample, we alternate between proposing samples and determining if they should be kept for a and b, and we do this 1000 for each parameter and use the first 100 samples as a burn-in. When computing the likelihood, we use the bootstrap particle filter with $N = 200$ particles and resample when $N_{\text{eff}} < 0.5N$. The settings of the Metropolis-Hastings algorithm can be tweaked to provide better estimates, but for the purposes here this is sufficient as the goal is to show that such a Metropolis-Hastings setup can be used for parameter estimation, rather than creating the best possible estimates.

Figure 2.10 shows a histogram of the samples obtained, and for both parameters, the true value lies slightly below the average of the samples. We get the estimates $\hat{a} = 0.06$ and $\hat{b} = 0.73$, and both slightly overestimate the true value. Figure 2.10 visualizes the samples obtained. For both the parameters, we see that there are several parameter values that frequently occur, but all

Parameter estimates		
Method	$\hat{a}$	$\hat{b}$
Agumenting state	-	0.68
PMMH Particle filter	0.06	0.73
PMMH Kalman	0.06	0.74

Table 2.1: Parameter estimates for the transitions in the state space model based on different approaches.

are located close to the true value. This may be due to the model itself and potentially the observations not being informative enough such that several parameter values get similar marginal likelihood values. Another factor is that we are only using an approximation of the marginal likelihood and this may influence the estimates. The effect here can easily be investigated by using a Kalman approach for computing the marginal likelihood.

As a comparison to this we repeat the exact same setup but this time rather than computing the likelihood with a particle filter we now use the Kalman approach for computing the likelihoods required. The estimates obtain here can be found in table 2.1 alongside the estimates from the other methods.

Figure 2.11 shows histograms of the samples obtained, and a scatter plot of the samples can be found in the same figure. Visually the scatter plots resulting from the two approaches are quite different and using a Kalman approach for computing the marginal likelihood gives samples that are close to being radially symmetric around the true parameter values, at least when comparing with the particle filter approach. This is also shown in the histograms which are more symmetric for the Kalman approach. This may be a result of using the exact likelihood, and when using an approximation of the marginal likelihood there is a risk of overestimating such that a poor sample appears to be better than it is.

When comparing with the first setup with a being known and we augmented the state, we observe that the histogram of the samples obtained with this approach is narrower around the true value. This may be in part due to a being known which is not the case for the Metropolis-Hastings histograms in which we consider pairs (a, b). There are several pairs of (a, b) which will have similar likelihoods, and the Metropolis-Hasting algorithm can jump between these rather than settling on one where a is fixed a b is varying.

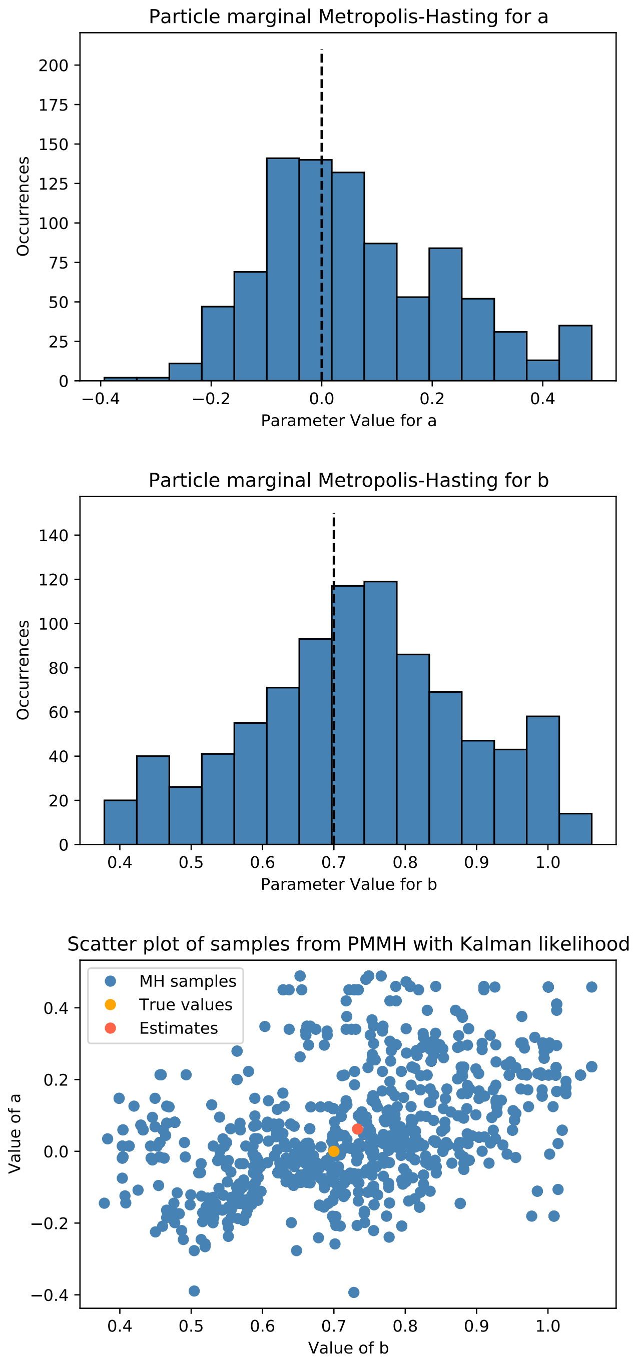

Figure 2.10: Visualizing the distribution of the samples obtained based on the Monte Carlo estimates of the marginal likelihood.

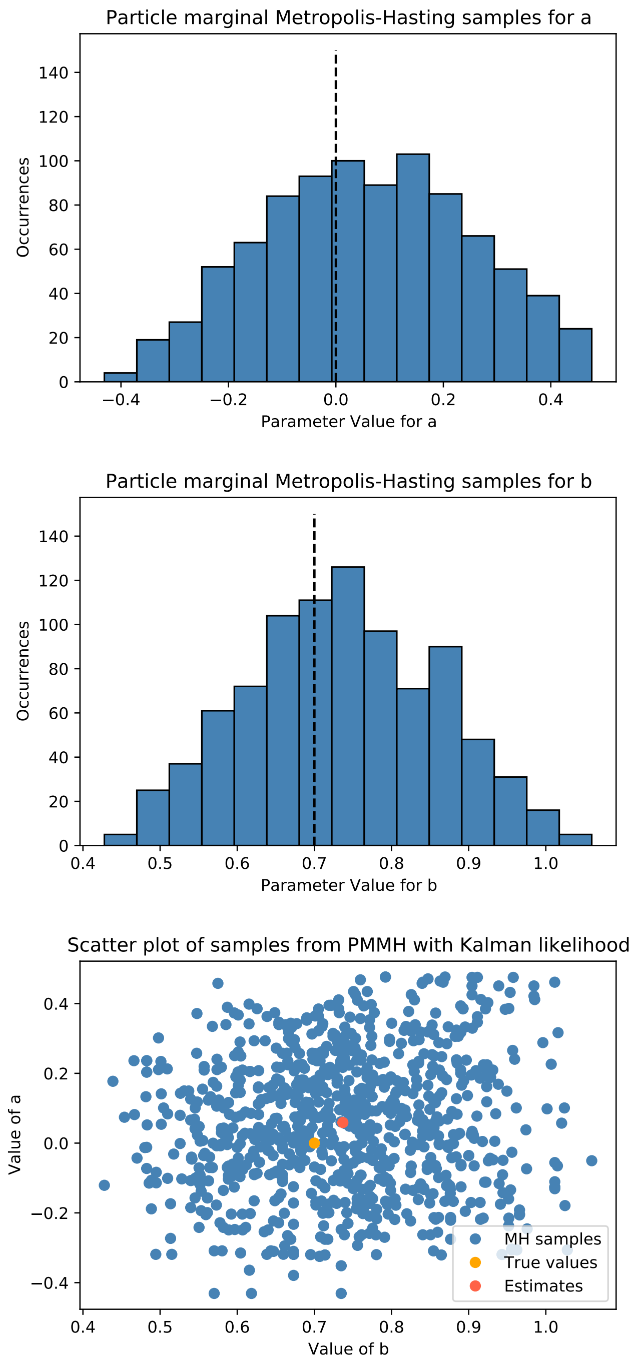

Figure 2.11: Visualizing the distribution of the samples obtained based on the Kalman marginal likelihood.

CHAPTER 3

Twisted Models and their use

3.1 Introducing twisted models and their use

The bootstrap particle filter has several nice properties as discussed previously in Chapter 2. This includes being easy to implement, being versatile in the sense that we are able to obtain estimates of a large number of quantities, and the fact that the estimates obtained are consistent. The latter of these is due to the Monte Carlo setup which is the basis for the particle filter. This allows particle filters to provide consistent estimates of quantities such as the marginal likelihood related to a series of observations, and state estimates through filtering and smoothing. While analytical methods such as the Kalman filter and related approaches only are available in special cases, Monte Carlo methods and particle filters are can be used more generally, and obtaining low variance estimates from particle filters is desirable.

Here we will primarily focus on improving the estimates of the marginal likelihood beyond those obtained from the bootstrap particle filter. In this context improving the estimates means obtaining lower variance estimates when using fewer particles than for estimates obtained from using the bootstrap filter directly on the state space model. To do this we will take the same approach as Guarniero, Johansen and Lee (2017) and Heng et al. (2020) and introduce twisted state space models which will be used to provide these low variance estimates. This is also a similar strategy to the one taken by Whiteley and Lee (2014), which was the first instance of twisting the model and the particle filter we found. Ala-Luhtala et al. (2016) builds further on this work and uses the same strategy. Such low variance likelihood estimates are desired as they are amongst other a key part of parameter estimation methods such as particle marginal Metropolis-Hastings (Andrieu, Doucet and Holenstein, 2010) which we discussed in Section 2.5. Additionally, this setup can be used to obtain good smoothing estimates of the state, which we will have a closer look at in Section 4.6.

Twisted state space models

We will start by introducing the twisted models. First, we consider a state space model where the density for the first state x_1 is given by $\mu(x_1)$ and the remaining transition densities are given by $f(x_{t-1}, x_t)$ for $t \in 2, \ldots, T$ and we have emission densities or potentials $g(x_t, y_t)$ for $t \in 1, \ldots, T$ for potentially multidimensional sequence latent variables and observations $x_{1:T}$ and $y_{1:T}$

respectively. This is the same model structure we considered in Chapter 2, and we will refer to this model as the original or untwisted model. To define a twisted model we introduce a sequence of bounded, strictly positive, continuous, and real-valued functions $\boldsymbol{\psi} := (\psi_1, \dots, \psi_T)$ and using these we define the following:

$$\begin{aligned}
\widetilde{\psi}_0 &= \int \mu(x')\psi_1(x')\,\mathrm{d}x' \\
\widetilde{\psi}_t(x_t) &= \int f(x_t, x')\psi_{t+1}(x')\,\mathrm{d}x' \text{ for } t \in 1, \dots, T-1 \\
\widetilde{\psi}_T(x_T) &= 1.
\end{aligned}$$

Using the above definitions we define the twisted model as

$$\mu^{\psi}(x_1) = \frac{\mu(x_1)\psi_1(x_1)}{\widetilde{\psi}_0} \tag{3.1}$$

$$f_t^{\psi}(x_{t-1}, x_t) = \frac{f(x_{t-1}, x_t)\psi_t(x_t)}{\widetilde{\psi}_{t-1}(x_{t-1})} \tag{3.2}$$

$$g_1^{\psi}(x_1, y_1) = g(x_1, y_1)\frac{\widetilde{\psi}_1(x_1)}{\psi_1(x_1)}\widetilde{\psi}_0 \tag{3.3}$$

$$g_t^{\psi}(x_t, y_t) = g(x_t, y_t)\frac{\widetilde{\psi}_t(x_t)}{\psi_t(x_t)} \tag{3.4}$$

The twisted state space model introduced here can be seen as a different model than the original model described by μ, f, and g, and we can consider this as a model which describes an entirely different system than the original model.

With this new twisted model, we can still use the bootstrap particle filter to obtain amongst other estimates of the states and an estimate of the marginal likelihood in the same manner as we would for any other model. These estimates will be related to the twisted model, however, due to the construction of the twisted model, some quantities of interest will be the same for the original and twisted model. In particular, when twisting a model according to (3.1)-(3.4) and the restrictions placed on $\boldsymbol{\psi}$, the original model and the twisted model will have the same marginal likelihood, which is the motivation for introducing the twisted models. To show this consider the marginal likelihood associated with the observations $y_{1:T}$ which is defined as

$$L \equiv \mathrm{E}\left[\prod_{t=1}^{T} g(X_t, y_t)\right] = \int_X \mu(x_1)g(x_1, y_1)\prod_{t=2}^{T} f(x_{t-1}, x_t)g(x_t, y_t)\,\mathrm{d}x_{1:T}. \tag{3.5}$$

Here the expected value is taken with respect to the random variables $X_{1:T}$. This expression for the marginal likelihood is equivalent to the more general expression presented in (1.6). Representing the marginal likelihood as an integral is convenient, as it is integrals on this form we can solve numerically using sequential Monte Carlo. We define the integral corresponding to the marginal likelihood of the original model explicitly as Z

$$Z \equiv \int_X \mu(x_1)g(x_1, y_1)\prod_{t=2}^{T} f(x_{t-1}, x_t)g(x_t, y_t)\,\mathrm{d}x_{1:T}. \tag{3.6}$$

Now turning our attention to the twisted model, we can also write corresponding integral to (3.6) for the twisted model

$$
\begin{aligned}
Z_\psi &= \int_X \mu^\psi(x_1) g_1^\psi(x_1, y_1) \prod_{t=2}^{T} f_t^\psi(x_{t-1}, x_t) g_t^\psi(x_t, y_t) \, \mathrm{d}x_{1:T} \\
&= \int_X \frac{\mu(x_1)\psi_1(x_1)}{\widetilde{\psi}_0} \frac{g(x_1, y_1)\widetilde{\psi}_1(x_1)}{\psi_1(x_1)} \widetilde{\psi}_0 \prod_{t=2}^{T} \frac{f(x_{t-1}, x_t)\psi_t(x_t)}{\widetilde{\psi}_{t-1}(x_{t-1})} \frac{g(x_t, y_t)\widetilde{\psi}_t(x_t)}{\psi_t(x_1)} \, \mathrm{d}x_{1:T} \\
&= \int_X \mu(x_1) g_1(x_1, y_1) \prod_{t=2}^{T} f(x_{t-1}, x_t) g(x_t, y_t) \, \mathrm{d}x_{1:T} \equiv Z.
\end{aligned}
$$

As shown above the quantity Z, which is equivalent to the marginal likelihood, is invariant under the transformation from the original model to the twisted model defined by (3.1)-(3.4). Notably, this quantity will be the same for any twisted model as long as it is defined based on a valid sequence $\boldsymbol{\psi}$. Hence the marginal likelihood of the twisted model is equal to that of the original model that we started with. The same is true for the joint density of $x_{1:T}$ and $y_{1:T}$ and by appending a function φ to the integral we can obtain the expected values of functions with respect to the smoothing density. We will have a closer look at smoothing in Section 4.6.

$$
\begin{aligned}
\mathrm{E}[\varphi(x_{1:T})|y_{1:T}] &= \int_X \varphi(x_{1:T}) p(x_{1:T}|y_{1:T}) \, \mathrm{d}x_{1:T} \\
&= \frac{1}{Z} \int_X \varphi(x_{1:T}) \mu(x_1) g_1(x_1, y_1) \prod_{t=2}^{T} f(x_{t-1}, x_t) g(x_t, y_t) \, \mathrm{d}x_{1:T} \\
&= \frac{1}{Z} \int_X \varphi(x_{1:T}) \mu^\psi(x_1) g_1^\psi(x_1, y_1) \prod_{t=2}^{T} f_t^\psi(x_{t-1}, x_t) g_t^\psi(x_t, y_t) \, \mathrm{d}x_{1:T}.
\end{aligned}
$$

Since the marginal likelihood of any twisted model is equal to that of the original model that we started with, we can estimate the marginal likelihood of the original model by estimating the marginal likelihood of any twisted model. That is to estimate the integral Z we can instead estimate the integral Z_ψ since these quantities are identical, and this will be the case for any valid sequence $\boldsymbol{\psi}$. Note that we will only have this equality at the final time T. The joint density of $x_{1:t}$ and $y_{1:t}$ for $t < T$ will not be identical for the original and twisted model. Estimating the marginal likelihood of any model is straightforward using the bootstrap filter. For the twisted model this corresponds to using the functions μ^ψ, f^ψ and g^ψ in the bootstrap particle filter, rather than μ, f and g. For a twisted model with emission density $g_t^\psi(x_t, y_t)$ at time t, running a bootstrap particle filter which at time t produces particles $\{x_t^i\}_{i=1}^N$, a Monte Carlo estimate of the marginal likelihood is given by

$$
Z_\psi^N = \prod_{t=1}^{T} \left[\frac{1}{N} \sum_{i=1}^{N} g_t^\psi(x_t^i, y_t) \right] = \widetilde{\psi}_0 \prod_{t=1}^{T} \left[\frac{1}{N} \sum_{i=1}^{N} g(x_t^i, y_t) \frac{\widetilde{\psi}_t(x_t^i)}{\psi_t(x_t^i)} \right]. \tag{3.7}
$$

The estimate obtained for the marginal likelihood, Z_ψ^N, will still be consistent and converge to the true marginal likelihood as $N \to \infty$, as it would for the

Algorithm 3: Bootstrap particle filter applied to a twisted model with adaptive resampling

```
1 Set $t = 1$, $\log(Z_\psi^N) = 0$
    and sample $x_1^i \sim \mu_1^\psi(\cdot)$ for $i \in (1, \dots, N)$
2 Compute weights $w_1^i \leftarrow g_1^\psi(x_1^i)$ and increment $t$
3 If $N_{\text{eff}}(w_{t-1}) < \alpha N$
    Set $\log(Z_\psi^N) \mathrel{+}= \log\left(\frac{1}{N}\sum_{i=1}^N w_{t-1}^i\right)$
    Sample $x_t^i \sim \frac{\sum_{j=1}^N w_{t-1}^j f_t^\psi(x_{t-1}^j, \cdot)}{\sum_{j=1}^N w_{t-1}^j}$ and set $w_t^i = g_t^\psi(x_t^i)$
4 Otherwise sample $x_t^i \sim f_t^\psi(x_{t-1}^i, \cdot)$ and set $w_t^i = w_{t-1}^i g_t^\psi(x_t^i)$
5 If $t = T$
    Set $\log(Z_\psi^N) \mathrel{+}= \log\left(\frac{1}{N}\sum_{i=1}^N w_T^i\right)$
6 Otherwise, increment $t$ and return to step 3
```

original model. This is because we are still using the bootstrap particle filter on the twisted model. Algorithm 3 describes how to run a bootstrap filter on the twisted model and simultaneously compute the marginal likelihood as described by Guarniero, Johansen and Lee (2017).

Being able to estimate the marginal likelihood of the original model with a twisted model rather than using the original model is the basis for the work of Guarniero, Johansen and Lee (2017) and Heng et al. (2020). Of interest is the effect of using the twisted model on the convergence of the marginal likelihood estimates. Since no matter how we twist the model, within the boundaries of (3.1)-(3.4) and the previously mentioned restrictions on $\boldsymbol{\psi}$, all twisted models will have the same marginal likelihood as the original model. Hence it is natural to consider if there is any twisted model that is preferable for estimating the marginal likelihood in terms of convergence and variance of the estimate. Of interest is if there is possible to choose a twisted model such that we obtain lower variance estimates of the marginal likelihood with fewer particles, which is computationally faster than running the bootstrap particle filter on the original model.

The optimal sequence

Having introduced the twisted state space models and shown that these will have the same marginal likelihood as the original untwisted model, we now want to see how these twisted models can be used to improve the estimates of the marginal likelihood. We want to determine a sequence of functions $\boldsymbol{\psi}$ which lowers the variance of the marginal likelihood estimates obtained by running the bootstrap particle filter on the twisted model while using fewer particles compared to estimates obtained from running the bootstrap filter on the original model. Since there are few restrictions on the functions twisting the model we are free to choose how to twist the model, however, not every sequence $\boldsymbol{\psi}$ will lower the variance of the estimates. For the purpose of lowering the variance Guarniero, Johansen and Lee (2017) propose the following sequence of twisting functions $\boldsymbol{\psi}^\star$

$$\begin{aligned}
&\psi_t^\star(x_t) = g(x_t, y_t)\mathrm{E}\left[\prod_{p=t+1}^{T} g(X_p, y_p)|\{X_t = x_t\}\right] \text{ with } t \in (1, \dots, T-1)\\
&\psi_T^\star(x_T) = g(x_T, y_T)
\end{aligned} \tag{3.8}$$

Here the expected value is with respect to the distribution of the latent variables X, in the same manner as (3.5). With this choice of twisting functions, it can be shown that the marginal likelihood estimate from the bootstrap filter, based on the twisted model defined by $\boldsymbol{\psi}^\star$, equals the true marginal likelihood for any number of particles. In contrast when considering the original model this estimate only converges to the true marginal likelihood. This makes $\boldsymbol{\psi}^\star$ optimal for the purpose of estimating the marginal likelihood since if this sequence can be obtained we can determine the exact marginal likelihood with zero variance, a proof of this can be found in Guarniero, Johansen and Lee (2017, proposition 2). It will still be beneficial to have a closer look at why this choice $\boldsymbol{\psi} = \boldsymbol{\psi}^\star$ will give an improvement, and simultaneously illustrate the main ideas of the proof. The rationale for why this particular sequence gives such an improvement lies in how the sequence $\boldsymbol{\psi}^\star$ that twist the model is constructed. We start by considering the definition of the optimal sequence of functions $\boldsymbol{\psi}^\star$ given by (3.8). Of particular interest is the expected value present for times $t < T$ in (3.8). At time t, this expected value can be seen as the likelihood for the future observation, i.e. $y_{t+1:T}$, conditional on the current state x_t. Furthermore, at time t, the inclusion of $g(x_t, y_t)$ in the optimal functions $\psi_t^\star(x_t)$ allows us to interpret these optimal functions as the likelihood of $y_{t:T}$ conditional on the state x_t. At time $t = 1$ this will be the likelihood of all the observations, and we get that quantity $\widetilde{\psi}_0^\star$ becomes

$$\begin{aligned}
\widetilde{\psi}_0^\star &= \int \mu(x_1)\psi_1(x_1)\,\mathrm{d}x_1\\
&= \int \mu(x_1)g(x_1, y_1)\mathrm{E}\left[\prod_{p=2}^{T} g(X_p, y_p)|\{X_1 = x_1\}\right]\mathrm{d}x_1 = L.
\end{aligned} \tag{3.9}$$

That is, when twisting the model with the sequence $\boldsymbol{\psi}^\star$ the quantity $\widetilde{\psi}_0^\star$ is equal to the marginal likelihood of all the observations. However, computing $\widetilde{\psi}_0^\star$ directly is not practical, in a similar manner to how computing (3.6) directly is not a practical way to determine the marginal likelihood.

With this result in the back of our minds, we now move on to why this provides low variance estimates in the bootstrap filter. Recall that a Monte Carlo estimate of the marginal likelihood of a twisted model is given by (3.7). When considering the optimal sequence $\boldsymbol{\psi}^\star$, this expression can be simplified by another property of the optimal sequence that we are yet to discuss. Namely that we can define each function in the sequence $\boldsymbol{\psi}^\star$ recursively as shown in proposition 3.1.1, and the same result and proof is given by proposition 4 in Guarniero, Johansen and Lee (2017). With this recursive definition, each factor of the product in (3.7) will be equal to 1 for the optimal sequence $\boldsymbol{\psi}^\star$. This is because we have we then have $\psi_t^\star(x_t) = g(x_t, y_t)\widetilde{\psi}_t^\star(x_t)$. As a result we have that $Z_{\psi^\star}^N = \widetilde{\psi}_0^\star = L$. As mentioned, this is a major difference from using the bootstrap particle filter on the original model to estimate the likelihood. We know that Z_{ψ}^{N} will converge to the likelihood when more samples are used such that the Monte Carlo estimate improves. For the optimal sequence $\boldsymbol{\psi}^{\star}$ we have equality between $Z_{\psi^{\star}}^{N}$ and L, and the particles do not affect the estimate obtained. Thus we will always obtain the same estimate of the marginal likelihood. As a result, running the bootstrap filter on the optimal twisted model and estimating the likelihood will yield the analytical marginal likelihood of the twisted model. Recalling that the marginal likelihood is invariant when twisting the model, rather than estimating the marginal likelihood on the original model we can use a twisted model based on the optimal sequence to obtain the exact marginal likelihood.

We previously mentioned that the twisted models have applications when it comes to smoothing. With the sequence $\boldsymbol{\psi}^{\star}$, the trajectories of a particles at time T from the bootstrap particle filter is a sample from the smoothing density $p(x_{1:T}|y_{1:T})$. Unlike the previous smoothing setups we have considered, in this sample, all the trajectories will have the same weight, hence we are sampling from the exact smoothing density. We will have a closer look at this in Section 4.6.

Proposition 3.1.1. *Following Guarniero, Johansen and Lee, 2017, proposition 4, $\psi^{\star}$ can be defined recursively as*

$$\begin{gathered}\psi_t^{\star}(x_t) = g(x_t, y_t)\int f(x_t, x')\psi_{t+1}^{\star}(x')\,\mathrm{d}x' = g(x_t, y_t)\widetilde{\psi}_t^{\star}(x_t),\ \textit{for}\ t \in 1, \dots, T\text{-}1 \\ \psi_T^{\star}(x_T) = g(x_T, y_T)\end{gathered} \tag{3.10}$$

Proof. The proof given here is identical that of Guarniero, Johansen and Lee, 2017, proposition 4, and follow from straight forward calculations. For $t = T$ the statements holds trivially by the definition of the optimal sequence, so we are interested in $t < T$. We then have from the definition of $\psi_t^{\star}(x_t)$

$$\begin{aligned}g(x_t, y_t)\widetilde{\psi}_t^{\star}(x_t) &= g(x_t, y_t)\int f(x_t, x')\psi_{t+1}^{\star}(x')\,\mathrm{d}x' \\ &= g(x_t, y_t)\int f(x_t, x')g(x_{t+1}, y_{t+1})\mathrm{E}\left[\prod_{p=t+2}^{T} g(X_p, y_p)\big|X_{t+1} = x'\right]\mathrm{d}x' \\ &= g(x_t, y_t)\mathrm{E}\left[\prod_{p=t+1}^{T} g(X_p, y_p)\big|X_t = x_t\right] = \psi_t^{\star}(x_t)\end{aligned}$$

■

Converging to the optimal sequence

So far we have primarily discussed the optimal case, but the optimal $\boldsymbol{\psi}^{\star}$ is generally not attainable, even with the recursive definition, so we have to settle for an approximation. Hence it is beneficial to have a look at what happens when we are close to the optimal sequence but not there yet, which will be the case for a good approximation of the optimal sequence. Proposition 3 of Guarniero, Johansen and Lee (2017) provides a central limit theorem for the likelihood estimates obtained from a twisted model and shows that the marginal likelihood estimate of the twisted model converges to the true marginal likelihood. Further, this central limit theorem shows that the variance of the ratio Z_{ψ}^{N}/Z tends to zero as $\boldsymbol{\psi} \to \boldsymbol{\psi}^{\star}$, where Z is the true marginal likelihood and Z_{ψ}^{N} is the marginal likelihood estimate from a twisted model based on the sequence $\boldsymbol{\psi}$. Here we have $\boldsymbol{\psi} \to \boldsymbol{\psi}^{\star}$ in an appropriate manner such that the expression for the variance, which depends on $\boldsymbol{\psi}$ and $\boldsymbol{\psi}^{\star}$, goes to zero, and is zero $\boldsymbol{\psi} = \boldsymbol{\psi}^{\star}$. Hence even if we are unable to use the exact optimal sequence $\boldsymbol{\psi}^{\star}$, we can obtain low variance estimates by instead basing the twisted model on a suitable approximation $\hat{\boldsymbol{\psi}}$, since this approximation presumably will be close to the optimal sequence. This motivates using an approximation of the optimal sequence in practice as this will be easier to compute and implement than the exact optimal sequence itself.

When using an approximation rather than the optimal sequence itself we have to use the particle filter to obtain the marginal likelihood estimate. The particles themselves will now have an effect on the estimates produced and depending on the quality of the approximation we will require several particles to obtain a satisfactory estimate of the marginal likelihood.

To quantify the effect of introducing the twisted model and the quality of the approximation it is natural to look at the variance of the estimates of the marginal likelihood directly since this is what we aim to improve by introducing the twisted model. Additionally, we can use the effective number of samples in the particle filter to evaluate the quality of the approximation. When the model is twisted by the optimal sequence $\boldsymbol{\psi}^{\star}$ the algorithm will never resample the particles as all the weights will be equal and the effective number of samples will be equal to the number of particles used. With a good approximation $\hat{\boldsymbol{\psi}}$ we then expect that the effective number of samples in the twisted model will be close to the number of particles used. For a good approximation, we expect that the bootstrap particle filter rarely resamples and keeps a high effective number of samples. This also illustrates an additional benefit with the twisted model, namely that when we have a high effective number of samples we avoid issues with degeneracy as we observed for the original model.

3.2 Approximating the optimal sequence

Creating an approximation

In practice, the optimal sequence cannot be obtained exactly, and we have to create an approximation. Since any valid twisted model will have the same marginal likelihood as the original model, any sequence which gives rise to a valid twisted model can be used to estimate the marginal likelihood. Two methods for approximating the optimal sequence of functions $\boldsymbol{\psi}^{\star}$ and estimating the marginal likelihood can be found Guarniero, Johansen and Lee (2017) and Heng et al. (2020). These methods are the iterated auxiliary particle filter and controlled sequential Monte Carlo, and both take a similar iterative approach for approximating the optimal sequence within a suitable class Ψ. For the iterated auxiliary particle filter, the core idea is to approximate the optimal sequence $\boldsymbol{\psi}^{\star}$ based on an iterative scheme. This method starts by creating a single approximation of $\boldsymbol{\psi}^{\star}$ and uses this to define a twisted model. The next step is to run the bootstrap particle filter on the twisted model and use the particles obtained to define a new approximation of $\boldsymbol{\psi}^\star$. The iterated auxiliary particle filter repeats this procedure to provide successive approximations of the optimal sequence and is repeated until some criterion is met, thus creating an iterative approximation of the optimal sequence $\boldsymbol{\psi}^\star$.

In order to obtain an approximation of the optimal sequence, we need to obtain approximations of each of the functions $\hat{\psi}_t(x_t) \approx \psi_t^\star(x_t)$. A simple way to do this is to use the recursive definition given in proposition 3.1.1, rather than (3.8) directly. This allows us to approximate each $\psi_t^\star(x_t)$ recursively using $\widetilde{\psi}_t^\star(x_t)$. When using the recursive definition, we start with $\psi_T^\star(x_T)$ as this is known to be equal to $g(x_T, y_T)$ per definition, and from this create approximations of the rest of the sequence. For the recursive definition we require $\widetilde{\psi}_t^\star(x_t)$ which is defined based on $\psi_{t+1}^\star(x_{t+1})$ which we do not know. However, since we are defining the sequence recursively starting at time T, we already have an approximation $\hat{\psi}_{t+1}(x_{t+1}) \approx \psi_{t+1}^\star(x_{t+1})$. This gives the following approximation of $\widetilde{\psi}_t^\star(x_t)$

$$\widetilde{\psi}_t^\star(x_t) \approx \overline{\psi}_t(x_t) = \int f(x_t, x')\hat{\psi}_{t+1}(x')\,\mathrm{d}x'. \tag{3.11}$$

Whenever $\hat{\psi}_{t+1}(x_{t+1})$ is a good approximation of $\psi_{t+1}^\star(x_{t+1})$ it is reasonable to assume that $\overline{\psi}_t(x_t)$ is a good approximation of $\widetilde{\psi}_t^\star(x_t)$. Note that for $t = T$ we can set $\overline{\psi}_T(x_T) = \widetilde{\psi}_T^\star(x_T) = 1$. Then based on (3.12), we can create approximations of the optimal functions recursively $\hat{\psi}_t(x_t) \approx \psi_t^\star(x_t)$. However, the approximation of (3.12) is not on a form we are able to work with, which will be the next topic of discussion.

$$\psi_t^\star(x_t) = g(x_t, y_t)\widetilde{\psi}_t^\star(x_t) \approx g(x_t, y_t)\overline{\psi}_t(x_t) \tag{3.12}$$

Approximating the functions

Given (3.11) and (3.12) we need to determine approximations of the optimal functions that we are able to work with. That is, we want the right-hand side of (3.12) on a form we are able to work with, and this will be the approximation of the optimal function. A simple method for creating the approximations of the optimal functions, which is used in the iterative steps of Guarniero, Johansen and Lee (2017), consists of restricting the functions to a suitable class Ψ and determining the best approximation within this class. Our criterion for determining the best approximation will be a distance measure or cost function, and the approximation will be the function in the class Ψ which is closest to the target, i.e. closest to the right-hand side of (3.12). We will have a closer look at how we can restrict the functions to a class Ψ later, and for now, we want to demonstrate how we can create the approximations given a class Ψ.

To create the approximations of the functions, the core idea is to look at the distance between a function $\psi_t(x_t) \in \Psi$ and the target. We want to compute both on a grid of points such that we can determine the distance between them, and the approximation is the function in the class Ψ which minimizes this distance. Thus we end up making two approximations, and the first one is (3.12). However, the right-hand side of (3.12) is not necessarily on a form we are able to work with. Hence, we want to determine an approximation within the class Ψ such that we get an approximation we are able to work with, which is the second approximation. Given a class Ψ this leaves two major components, the grid of points where we compute the functions, and the distance measure used to determine the approximation.

For the grid of points where we compute the functions, there are several options. Guarniero, Johansen and Lee (2017) run the bootstrap filter and use the particles produced as the grid points. With this choice, the approximation will become better in the region where the particles are located, but it requires that we run the particle filter once to obtain the particles. A simpler method, which we will use for the examples in Section 3.3, is to subjectively choose the points where we compute the functions. This method is not ideal as it requires a subjective choice, but for the purpose of illustration, it will be sufficient in our examples later. In Section 3.4 we will have a closer look at how we can create a simple grid where we can compute the functions, but first, we want to show how we can obtain an approximation of the optimal sequence and that there is a benefit in introducing the twisted models.

For the final component we want a distance measure to determine the best approximation within the chosen class Ψ. One possible choice of distance measure, that is computationally inexpensive, proposed by Guarniero, Johansen and Lee (2017) is the following

$$\mathcal{A} = \sum_{i=1}^{N} \left[\psi_t(x^i) - \lambda g(x^i, y_t)\widetilde{\psi}_t(x^i)\right]^2 \tag{3.13}$$

Here $\psi_t(x_t)$ is a function in the chosen class Ψ and we minimize (3.13) to determine the optimal parameters of $\psi_t(x_t)$ which provides the approximation $\hat{\psi}_t(x_t)$ within this class. In practice with the recursive definition, $\widetilde{\psi}_t(x^i)$ will be replaced with $\overline{\psi}_t(x^i)$ as defined in (3.11). It is worth noting that including the value of λ from (3.13) in the approximation $\hat{\psi}_t(x_t)$ will not be necessary. This is because we are free to scale the twisting functions since the likelihood estimate is invariant to the scaling of these functions as shown in proposition 3.2.1 below. This also means that each of the optimal functions $\psi_t^\star(x_t)$ is unique up to scaling when we consider the full joint density up to and including time T. Even though we do not include λ in the approximation, it is still desirable to include it in the minimization step. This is so we can deal with a potential situation where the class we want to select an approximation from is proportional to the target, that is $\psi_t(x_t) \approx c_t g(x_t, y_t)\overline{\psi}_t(x_t)$ for some real constant c_t and $\psi_t(x_t) \in \Psi$. Note that we could also extend the class Ψ to include this scaling factor, but for the purpose here it will be unnecessary since it has no effect on the estimates of the marginal likelihood produced. Nor will it have any effect on state estimates which rely on the full posterior density since also here the particles produced will be identical, and for the posterior density itself the scaling factors will cancel out.

A slightly different approach is taken by Heng et al. (2020), where instead of approximating $\psi_t^\star(x_t)$ directly, they approximate $V_t^\star(x_t) = -\log(\psi_t^\star(x_t))$. As working on a logarithmic scale often provides greater numerical stability, using a distance measure based on the logarithm of the optimal sequence may be preferable, and one possible measure incorporating this is the following

$$\mathcal{A}_{\log} = \sum_{i=1}^{N} \left[V_t(x^i) + \log[\lambda g(x^i, y_t)\widetilde{\psi}_t(x^i)]\right]^2 . \tag{3.14}$$

By minimizing (3.14) in terms of the parameters of V_t we can obtain the best approximation $\hat{V}_t$. Based on this we use that $\hat{\psi}_t(x_t) = \exp(-\hat{V}_t(x_t))$ to obtain the approximation of $\psi_t^\star(x_t)$. This can be a viable alternative to (3.13), and the difference lies in which regions we prioritize minimizing the function. This can be useful if there are some regions that are key for obtaining good approximations of the functions. Changing the distance measure also allows us to decide how to penalize errors in the approximation, and different distance measures will penalize errors differently.

Note here that (3.13) and (3.14) could be accompanied by a penalizing term on λ and the parameters of $\psi_t(x)$. This is to avoid unreasonable solutions where λ is close to zero and $\psi_t(x)$ being diffuse and small in the region where we evaluate the function. In our examples, we did not include such a term which could make the approximations more robust. Instead, we manually verified the obtained parameter values to make sure they were reasonable.

The approach for creating the approximation using the recursive definition can be summarized by algorithm 4. This outlines the general idea that we will follow to obtain a simple approximation to the optimal sequence, but variations within this are possible.

Algorithm 4: Outline of the procedure for approximating the optimal sequence

```
1 For time $t = T$ we have $\psi_T^\star(x_T) = g(x_T, y_T)$. To obtain the
    approximation we compute the distance between a function $\psi_T(x_T)$
    from the class $\Psi$ and the target $g(x_T, y_T)$ according to some distance
    measure on an appropriate grid. The approximation $\hat{\psi}_T(x_T)$ is the
    function which minimizes this distance.
2 Using the recursive definition and (3.12) we have
    $\psi_t^\star(x_t) \approx g(x_t, y_t)\overline{\psi}_t(x_t)$ which can be computed pointwise. To obtain
    the approximation we compute the distance between a function
    $\psi_{T-1}(x_{T-1})$ from the class $\Psi$ and the target
    $g(x_{T-1}, y_{T-1})\overline{\psi}_{T-1}(x_{T-1})$ according to some distance measure on an
    appropriate grid. The approximation $\hat{\psi}_{T-1}(x_{T-1})$ is the function
    which minimizes this distance.
3 The procedure in step 2 can then be repeated for each $t$ to create an
    approximation of the optimal sequence in $\hat{\psi}_1(x_1), \ldots, \hat{\psi}_T(x_T)$
```

Proposition 3.2.1. *Given a sequence of functions $\psi_1(x_1), \ldots, \psi_T(x_T)$ which defines a twisted model. We can scale the functions $\psi_t(x_t)$ by real constants $c_1, \ldots, c_{T+1}$ with $c_{T+1} = 1$, such that $\psi_t(x_t) \to c_t \cdot \psi_t(x_t)$, $\widetilde{\psi}_t(x_t) \to c_{t+1}\widetilde{\psi}_t(x_t)$. By scaling the functions in this manner, the particles produced will not be changed and likelihood estimates Z_ψ^N, as given by equation (3.7), are invariant to this scaling.*

Proof. When multiplying with the scaling factor the transition densities of the twisted model (3.1) and (3.2) remain unchanged since the scaling factor will simply cancel out. The same is true for the effective sample size, given by (2.13), which determines the resampling, so this will also remain unchanged. As a result, we only need to look at the likelihood estimate itself, since the scaling has no effect on the particles produced. The likelihood estimate for a twisted model, with the inclusion of a scaling factor, can be written as

$$Z_{\psi}^{N} = c_1 \widetilde{\psi}_0 \prod_{t=1}^{T} \left[\frac{1}{N} \sum_{i=1}^{N} g(x_t^i, y_t) \frac{c_{t+1} \widetilde{\psi}_t(x_t^i)}{c_t \psi_t(x_t^i)} \right]$$

Recalling that $\widetilde{\psi}_T(x_T) \equiv 1$ and we defined $c_{T+1} = 1$ so that $c_{T+1}\widetilde{\psi}_T(x_T) = 1$. We then get that all the remaining c_t's will cancel out as each c_t will occur once in the numerator and once in the denominator of the product. As a result given a sequence of functions $\boldsymbol{\psi}$ that defines a twisted model, the likelihood estimate based on this model is invariant when scaling $\psi_t(x_t)$ by factors $c_1, \ldots, c_T$. ■

Restricting the functions

As mentioned we wanted to restrict the approximations to a class Ψ. Imposing such a restriction allows us to focus on a particular class from which we can obtain the approximation. An additional reason for this restriction is that we want to run the bootstrap filter on the twisted model, which is defined by (3.1)-(3.4) based a sequence $\hat{\boldsymbol{\psi}}$. The main restriction here lies in being able to efficiently sample from (3.1) and (3.2) to produce the particles in algorithm 3. Simultaneously we also need to be able to compute (3.3) and (3.4) pointwise for computing the weights in the algorithm and provide the likelihood estimates. This restricts the approximation and our goal is now to obtain a good approximation of the optimal sequence which adheres to these criteria.

When creating the approximations, we want that the resulting twisted model is one that we are guaranteed to be able to work with when it comes to sampling. One possible method of dealing with this, which we will take here, is to choose each ψ_t from a class of functions Ψ that is conjugate to $f(x_{t-1}, x_t)$ such that $f(x_{t-1}, x_t)$ and $f_t^{\psi}(x_{t-1}, x_t)$ are on the same form, and the same being the case for μ and μ^{ψ}. This is beneficial as we end up with a known form that we are able to deal with when sampling from the twisted model, assuming we are able to sample from the original model. This is also convenient since having everything on the same parametric form makes the calculations identical at each time t up to the parameter values. In practice, this makes implementing the particle filters for the twisted models much easier. Note that this is a requirement mostly for convenience as it is possible to use a non-conjugate setup but repeat all the calculations at each time t and make sure that sampling from the twisted model is possible.

Here we only consider models where $f(x_{t-1}, x_t)$ and $\mu(x_1)$ are on the same form as in the linear Gaussian state space model, but we will consider more emission densities than the linear Gaussian model. Based on this choice of transition densities we force each $\psi_t(x_t)$ to be conjugate to Gaussian densities, and we use the same class Ψ as Guarniero, Johansen and Lee (2017) since this is suited to the class of models we will consider. We consider the following class Ψ

$$\psi_t(x_t) = C^t + \sum_{k=1}^{M} c_k^t \mathcal{N}(x_t; a_k^t, b_k^t). \tag{3.15}$$

This is one possible choice for the class Ψ which is suited for the problem at hand. Here $M \in \mathbb{N}$, $C \in \mathbb{R}_+ \cup \{0\}$ and sequences $\{a_k^t\}_{k=1}^M$, $\{b_k^t\}_{k=1}^M$, $\{c_k^t\}_{k=1}^M$ of mean vectors, covariance matrices and positive real constants respectively. With this choice of $\psi_t(x_t)$, and the densities f and μ in the original model being Gaussian, the densities which govern the latent variables in the twisted model will also be Gaussian mixtures. This is because mixtures of the normal distribution have the normal distribution as a conjugate distribution, which is why we chose the functions on this form. Note that for this choice of $\psi_t(x_t)$ if C^t is non-zero it has the benefit of making potentials g_t^{ψ} become more robust numerically. It ensures that we never divide by zero, or some very small value when computing the weights since in the bootstrap filter computing the weights include dividing by $\psi_t(x_t)$. Since the Gaussian densities themselves are strictly positive, by setting $C^t > 0$ we ensure that we never divide by a value smaller than C^t when computing the weights in the twisted model. It is also possible having $C^t = 0$ which makes the computations somewhat simpler. By construction of the twisted model, particles are unlikely to occur in regions where $\psi_t(x_t)$ is small since the density the particles are sampled from is proportional to the functions twisting the model. For the models, we have considered this was not an issue, so we used $C^t = 0$. Additionally, forcing $C^t > 0$ can lead to issues with this approximation. Assuming $\psi_t^\star(x_t) \to 0$ as $x_t \to \pm\infty$, which is the case for Gaussian functions, setting $C^t > 0$ makes it so the approximations will not have the same values in this limit. A key difference here is if we consider the constant C^t as a parameter of the class and obtain the best approximation with the restriction that C^t is greater than some specific value, or if we include C^t at after creating the approximation. Doing the latter makes the approximation of the optimal sequence worse and can potentially influence the results.

The restrictions we use here are specific to the models we are considering. When applying the twisted models to other non-Gaussian models it is possible to change the restrictions of f, μ, and ψ_t to suit the problem at hand. However, to restrict the approximations of each $\psi_t^\star(x_t)$ to functions to some class Ψ such as (3.15) is not ideal. While such restrictions have practical benefits when it comes to sampling in the bootstrap filter, imposing such restriction on the approximation when the exact $\psi_t^\star(x_t) \notin \Psi$ makes the optimal sequence become unobtainable. An option in practice is then to restrict the functions $\psi_t(x_t)$ to a class Ψ that allows efficient sampling in the bootstrap particle filter, and simultaneously permits a good approximation of the functions $\psi_t^\star(x_t)$. The former of these is quite easy to ensure for instance by using conjugate functions, but the latter can be a bit more involved. One relatively straightforward method to deal with this is to use some very general class of functions Ψ such as the Gaussian mixture of (3.15) as this can create a good approximation of a large number of functions. But, for a complex class Ψ with a large number of parameters, it can be difficult to determine the best parameters. A possibly simpler method, that we will take later, is using knowledge of the model and the functions $\psi_t^\star(x_t)$ to choose a suitable class Ψ. If for instance, we know from either experience or other sources such as experiments the general form of $\psi_t^\star(x_t)$ we can from this choose an appropriate class Ψ to which we restrict the functions twisting the model.

3.3 Two simple examples

Example 1- Likelihood estimates of a twisted model. Ideal setting

Our interest here is in investigating the effect of twisting the model when it comes to the likelihood estimates, and to determine if there is a benefit to introducing the twisted models. The model used here is a one-dimensional linear Gaussian state space model so the state- and emission models are on the form of (1.4) and (1.5). This model can be described in terms of the following transition and emission densities

$$\begin{aligned}
&\mu(x_1) = \mathcal{N}(x_1; 0.85, 1)\\
&f(x_{t-1}, x_t) = \mathcal{N}(x_t; 0.85 + 0.7x_{t-1}, 1), \text{ for } t \in 2, \dots, T\\
&g(x_t, y_t) = \mathcal{N}(y_t; 2x_t, 1))
\end{aligned}$$

For the data, we simulate a sequence of 50 latent variables $x_{1:50}$ and corresponding observations $y_{1:50}$ based on this model. To approximate the optimal sequence of functions $\boldsymbol{\psi}^\star$ we take the approach of minimizing a distance measure, and in this case, we use least squares distance as given by (3.13), and follow the procedure outlined in algorithm 4. Furthermore, in this example the optimal sequence $\boldsymbol{\psi}^\star$ can easily be approximated by a single Gaussian density as seen in Figure 3.1, hence the class Ψ is here Gaussian densities, which can be parameterized by their mean and variance. To determine the best approximation, we need to determine the optimal values for these parameters. For the grid in this example, we manually choose a fairly wide grid for the optimization. We want the grid used here to contain the mode of $\psi_t^\star(x_t)$ since this corresponds to the expected value in the parametrization of the approximation. The grid used here consists of 250 points evenly spread between -5 and 5 for all the functions. This grid a chosen based on the observations y_t and after inspecting Figure 3.1. The observations and the emission density g can be used to determine a region where $g(x_t, y_t)$ is relatively large as a function of x_t. Based on this we chose the grid and created an approximation. We plotted some of the targets in Figure 3.1 to verify that the functions we were approximating were located in the grid.

This subjective choice is not ideal as it would be preferable to use some more general method of choosing the grid points. Here we ended up including much of the tails of the Gaussian density which is unnecessary as we would like to create the approximation near the mode. This can be avoided with a more systematic setup.

We note that such optimization problems can be sensitive to the starting location. To counter this, we exploit that the approximation can be parameterized by an expected value and a variance, and providing estimates of these can serve as good starting points for the optimization. Since we can compute our target $g(x_t, y_t)\overline{\psi}_t(x_t)$ pointwise we can compute this function for each point on the grid and create a sample of the grid values with probability equal to the value of the function at that point. This allows us to make Monte Carlo estimates of the mean and variance based on this sample to serve as starting points for the optimization. We will also exploit this to obtain starting points when considering this Gaussian approximation in later sections.

For comparing the likelihood estimates we rely on running both the regular bootstrap particle filter on the original and twisted model multiple times and

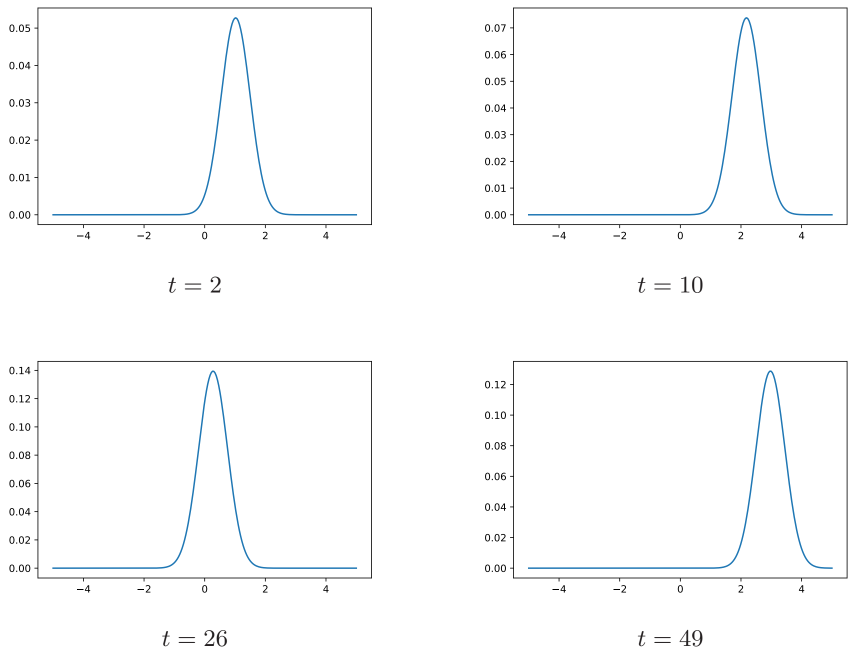

Figure 3.1: Graphs showing some of the functions we want to approximate as Gaussian on the chosen grid. The graphs show the function $g(x_t, y_t)\overline{\psi}_t(x_t)$ where $\overline{\psi}_t(x_t)$ is based on the approximation of the optimal sequence at time $t+1$ (3.11). Clearly, we can achieve good approximations using a single Gaussian function on the given interval for the cases shown here. From this we could potentially shorten the interval by placing the lower bound at $x_t = -2$

looking at the variance in the likelihood estimates. For running the particle filters we rely on algorithm 3. For the bootstrap filter likelihood estimates we run the particle filter 100 times each time with $N = 500$ particles. We also run the twisted particle filter 100 times this time with $N = 125$ particles. In both cases, we resample when the effective number of samples, (2.13), is lower than $0.5N$.

Since this is a linear Gaussian model, we can compute the true likelihood Z using a Kalman approach. This can then be used as a comparison to our estimates Z^N obtained from the particle filters. Figure 3.2 shows box plots of the ratio of likelihoods $\frac{Z^N}{Z}$. Since both particle approximations are consistent, we expect both to be close to one, and by scaling the likelihood in such a manner it easy to see how far off the estimates are. Figure 3.2 clearly shows that the spread of the likelihood estimates based on the twisted model is lower than that of the untwisted model. Table 3.1 includes the standard deviation from the samples of 100 particles where we also see that the standard deviation σ for the 100 repetitions is much lower for the twisted model.

For this example, we see that twisting the model appropriately results in improved likelihood estimates, in terms of lower variance, than the bootstrap filter applied to the original model. Additionally, the twisted model obtains better results with far fewer particles than the original model which makes it

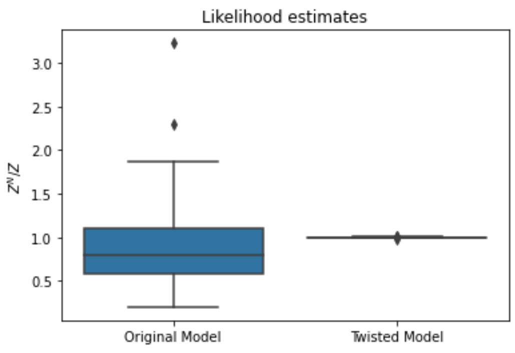

Figure 3.2: Boxplots of the ratio Z^N/Z, that is the ratio between the estimated likelihood and the true likelihood, for the 100 repetitions.

Results for likelihood ratio Z^N/Z		
Method (Number of particles)	Standard deviation of the estimates	Average resampling count
Original model (500)	0.475	32.89 ± 1.12
Twisted model (125)	0.006	0 ± 0

Table 3.1: Simple summary of the results for the linear Gaussian model. The leftmost column shows which model is used alongside the number of particles. The middle column shows the standard deviation of the 100 estimates obtained for each model. The rightmost column shows the average (± standard deviation) number of times resampling occurred in the particle filter for each of the models.

faster to run as we need to simulate fewer particles. While this is somewhat redundant for this model since we have the Kalman approach available, it is still interesting to see the benefit of introducing the twisted model. These results motivate the use of the twisted models for likelihood estimation. It is also worth noting that the setting for this example is idealized as we are able to provide a very good approximation, without too much effort, to the optimal sequence $\boldsymbol{\psi}^\star$.

Example 2 - Poisson observations

Now we want to consider a different model than the linear Gaussian model from the previous example. We will consider a setting where the approximations obtained are not as good as those obtained in the linear Gaussian case and see if introducing the twisted models still give an improvement. For this example, we keep the linear Gaussian transitions from the previous example, but instead of a linear Gaussian emission model we now consider observations from the Poisson distribution. This can be described in terms of the following emission and transition densities

$$\begin{aligned}
\mu(x_1) &= \mathcal{N}(x_1; 0.85, 1) \\
f(x_{t-1}, x_t) &= \mathcal{N}(x_t; 0.85 + 0.7x_{t-1}, 1), \text{ for } t \in 2, \ldots, T \\
g(x_t, y_t) &= \frac{\exp(x_t)^{y_t}}{y_t!} \exp(-\exp(x_t))
\end{aligned}$$

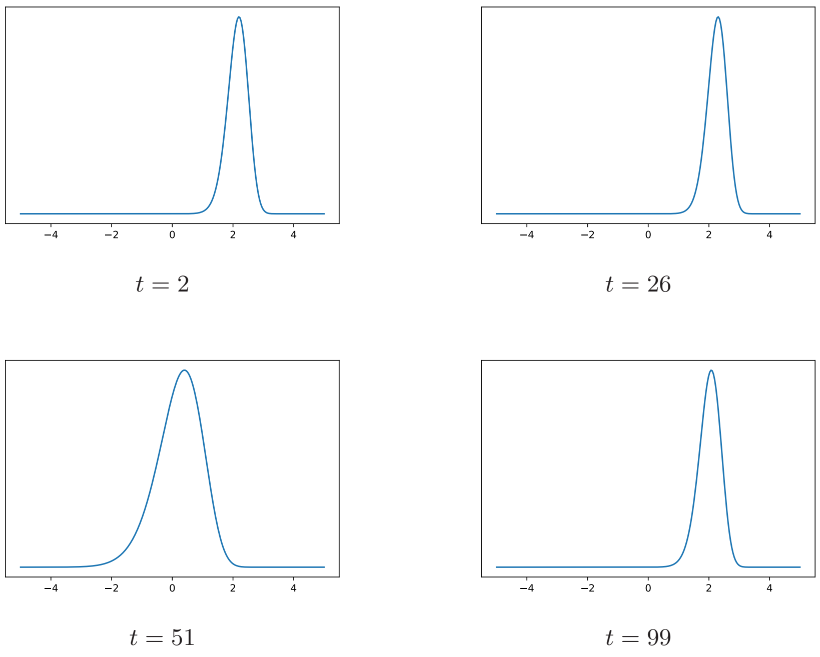

Figure 3.3: The graphs show the function $g(x_t, y_t)\overline{\psi}_t(x_t)$, which we want to approximate, where $\overline{\psi}_t(x_t)$ is based on the approximation of the optimal sequence at time $t + 1$, (3.11). From these figures it is reasonable to use a single Gaussian density for the approximation.

This is a more complicated model since this emission model is non-linear in the state variable, and we can no longer use the Kalman filter to obtain analytical results. For the data in this model we simulate sequences of 100 latent variables and observations according to the specified model.

To approximate the optimal sequence of functions $\boldsymbol{\psi}^\star$ we take the same approach as in the previous example. Again we assume that each $\psi_t^\star(x_t)$ can be approximated as a single Gaussian density. The grid used for the optimization step in this example consists of 1000 evenly spaced points on the interval $[-10, 10]$. We created this grid in the same manner as in the previous example. Since this is a simple example we also had the freedom to try out different configurations to see what worked well when it comes to creating a good approximation and convergence in the minimization. From this grid we created an approximation and in Figure 3.3 we plotted again $g(x_t, y_t)\overline{\psi}_t(x_t)$ for some values of t. This allows us to verify that the mode of the functions lies within the grid and that the Gaussian approximation is reasonable for the functions we are approximating. We do not expect to obtain a perfect approximations as the target functions, $g(x_t, y_t)\overline{\psi}_t(x_t)$, are not Gaussian.

Again, we are interested in comparing the likelihood estimate and see if there is an improvement in the likelihood estimate from introducing the twisted model. Similar to the previous example, we run the bootstrap particle filter 100 times with the regular model and 100 times with the twisted model and compare the results. For the bootstrap filter on the original model, we use $N = 500$

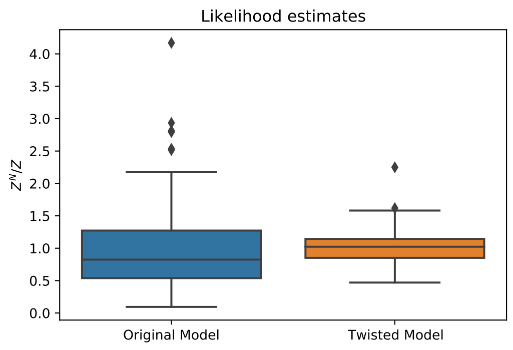

Figure 3.4: Box plot comparing the likelihood ratio Z^N/Z for twisted model and original model with Poisson observations. For each approach, the likelihood was computed 100 times, and we see that the twisted model provides lower variance likelihood estimates with fewer particles. Here the likelihoods are compared to that estimated from a bootstrap filter with 10000 particles.

particles and for the twisted model, we use $N = 125$ particles. In both cases, we resample when the effective number of samples, given by equation (2.13), is lower than $0.5N$. Again we want to look at a likelihood ratio Z^N/Z, and since we cannot find the true likelihood Z using the Kalman approach we now run a bootstrap particle filter with 10000 particles to serve as our comparison. Again, we do this rescaling since comparing the log-likelihoods is not always the most intuitive in terms of scale and relative difference and comparing the likelihoods directly is not viable since these will be extremely small.

Figure 3.4 shows a box plot comparing the likelihood estimates from the two different particle approximations. Again, we see an improvement from introducing the twisted model, but this is not as significant as for the linear Gaussian model. This is also shown in table 3.2, where we notice that in this case we still resample when using the twisted model. This is an indication that our approximation still differs from the optimal sequence $\boldsymbol{\psi}^\star$. This is what we also observe from figure 3.5 where we observe that the approximation with the single Gaussian density does not provide a perfect approximation to the optimal sequence of functions. We will have a closer look at using a more flexible approximation in Chapter 4 which will be able to provide a better approximation of the optimal sequence. Using the twisted model we still obtain likelihood estimates with lower variance when comparing to the original model. This is also done with the twisted model using far fewer particles reducing the computational cost, provided we already have obtained the approximation $\hat{\boldsymbol{\psi}}$. This further motivates the use of the twisted model as the likelihood estimate obtained has lower variance than the original model.

Results for estimating the likelihood ratio Z^N/Z		
Model (Number of particles)	Standard deviation of the estimates	Average resampling count
Original model (500)	0.673	88.63 ± 0.59
Twisted model with subjectively chosen grid (125)	0.265	4.33 ± 0.60

Table 3.2: Simple summary of the results for this example with Poisson observations. The leftmost column shows which model is used alongside the number of particles. The middle column shows the standard deviation of the 100 estimates obtained for each model. The rightmost column shows the average ($\pm$ standard deviation) number of times resampling occurred in the particle filter for each of the models.

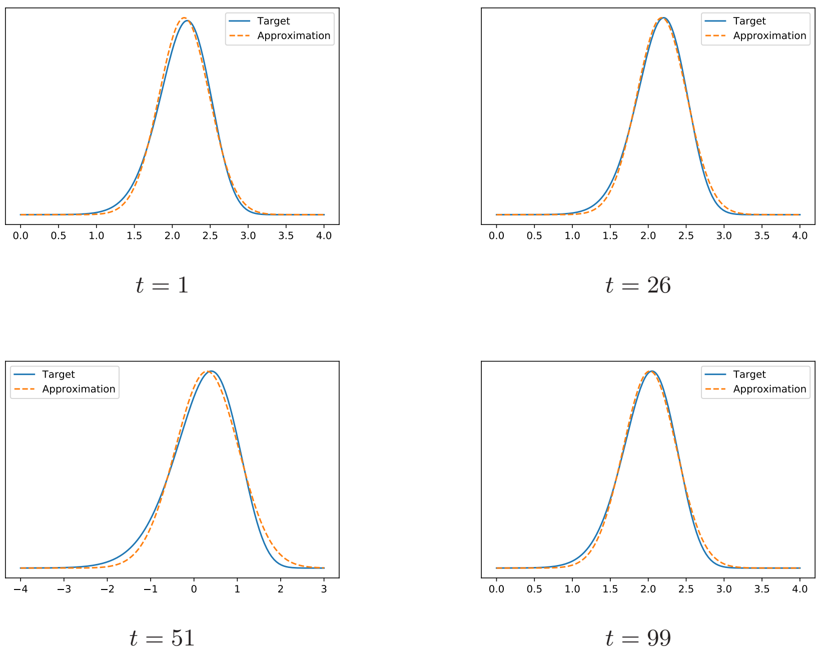

Figure 3.5: The graphs show the approximation and the function $g(x_t, y_t)\overline{\psi}_t(x_t)$ for different values of t. This is an extension of Figure 3.3

3.4 Using a fixed grid for the approximation of the optimal functions

A key point of introducing the twisted models is that there is some optimal sequence, but since this sequence cannot be obtained and we want an approximation. Methods such as the iterated auxiliary particle filter by Guarniero, Johansen and Lee (2017) and controlled sequential Monte Carlo by Heng et al. (2020) use iterative approaches, based on particles from a previous iteration for determining improving the approximation of $\boldsymbol{\psi}^{\star}$. In our examples so far we took a different approach and discarded the iterative setup. In this section, we will propose a setup that can be used to obtain an approximation of the optimal sequence which does not rely on any iterative steps.

Recall that the purpose of this optimization step is to determine the best approximation of the optimal sequence of functions given by (3.8) using functions on the form (3.15) or some other appropriate class of functions Ψ. In Section 3.2 we describe how the recursive definition can be used to obtain an approximation by solving a minimization problem to determine the optimal parameters within our class. For the moment we will focus on creating the best approximation within a given class of functions, and for the minimization step in our examples, we used the recursive definition alongside a simple distance measure. In doing this we compute an approximation of the target, $g(x_t, y_t)\overline{\psi}(x_t)$, on a fixed grid of points rather than relying on particles from a previous iteration. For the simple models we have looked at so far, we observe that even a simple approximation of the optimal sequence will give lower variance likelihood estimates with the twisted model. This is without introducing the iterative setups, and it bears consideration if the iterative setup is required for practical purposes where we primarily seek to obtain low variance likelihood estimates. For the full iterative setups to be beneficial in practice we would require that the iterative steps further improves the approximation of the optimal sequence. This is to justify the additional computational effort required from running the particle filter several times in the iterative process rather than using a single approximation as we did in the previous examples. Of interest is then if we can obtain a sufficient approximation of the optimal sequence in a single step, rather than using an iterative approach. We expect this to be useful in settings where it may be sufficient to provide lower variance estimates by introducing the twisted model, and we do not require the zero variance likelihood estimates obtained by using the exact optimal sequence. When this is the case it may be sufficient with a, possibly crude, approximation as seen in the previous examples, making the full iterative setup redundant. Of interest is if an approximation created by computing the target function on some appropriate grid without the iterative approach as we did in the previous examples can be sufficient to provide an adequate approximation of the optimal sequence, and how this best can be done in practice.

When discarding the iterative process, i.e. only creating a single approximation of the sequence as in the examples in Section 3.3, it is worth considering how we create the approximation. This is because we only create a single approximation of the sequence and we want to obtain the best possible approximation. There are several factors here, but for now, we will focus primarily on the grid where we compute the approximation since this is what changes compared to the iterative approaches. In the examples we considered, we only dealt with one-dimensional functions so we were able to visualize the functions and subjectively choose a suitable grid of points. In comparison, Guarniero, Johansen and Lee (2017) uses the particles from a bootstrap particle filter on the original model as the grid for the first iteration of the iterated auxiliary particle filter. A third option, which is an idea we have not seen anywhere else, is to construct a fixed grid rather than using the particles directly. The point here being that we want a good approximation, and to do this we can consider alternatives to using the particles directly. One such alternative is to specify a grid where we compute and minimize the target based on the model and observations directly which would allow us to approximate the optimal sequence without running the particle filter once.

Model specific grid

Now we will discuss how we can construct a grid for the optimization step in algorithm 4 which does not rely on the particles from some previous iteration which is in contrast to the iterative approaches proposed in Guarniero, Johansen and Lee (2017) and Heng et al. (2020). Ideally, we would like to construct a grid that can be used for the approximation of the optimal sequence directly which depends on the model and observations. Thus avoiding running a particle filter to provide the approximation and allowing us to obtain an approximation in a single step which will be faster than an iterative approach. The primary issue of constructing such a model specific grid is where to locate the grid to provide an adequate approximation of $\psi_t^\star(x_t)$. In d dimensions a grid consisting of N points in each dimension would consist of N^d points in total which quickly will become computationally expensive for large values of d and we will focus on low dimensional setting and specifically $d = 1$.

In the case when approximating the target with a single Gaussian density, and we know that the target is unimodal such that we can obtain a good approximation, we want the grid to be located near the mode of the target. This is because it is in this region we are likely to evaluate the function and this region contains the only defining feature of the Gaussian density, namely the mode. Simultaneously placing the grid in regions far from the mode where $\psi_t^\star(x_t) \approx 0$ is not ideal as several choices for the parameters of the approximation may be similar in this region but might become widely different in other regions near the mode as all Gaussian densities go to zero far from the mode.

The target functions we want to approximate are $g(x_t, y_t)\widetilde{\psi}_t^\star(x_t)$, where $\widetilde{\psi}_t^\star(x_t)$ is replaced with the approximation $\overline{\psi}_t(x_t)$ as defined in (3.11) in practice, and it is worth looking at where we need to evaluate these functions in the particle filter. When using algorithm 3 we compute each $\psi_t(x_t)$ at the values of the particles at time t when computing the weights, and in the definition of twisted model, the particles are sampled from the densities defined by (3.1) and (3.2). Hence a majority of the particles will be located in regions with a high probability in these densities. This motivates using particles for the later iterative steps in the iterated auxiliary particle filter for creating the approximation as we then create the approximation where the particles are located which also is where we evaluate the function. Further, this motivates placing the grid near the mode of our target function $\psi_t(x_t)$ as we did in the above examples as the particles will end up near this function. This is because we are likely to evaluate the function in the regions where (3.1) and (3.2) are large, which will coincide with the regions where $\psi_t^\star(x_t)$ is large in most situations.

Simple setup

Here we will focus on locating a region of $\psi_t^\star(x_t)$ that does not include the tails, and how we can provide a suitable grid of points in this region based on the model. We will use $g(x_t, y_t)$ and the observations $y_{1:T}$ to determine a region where we can approximate the functions. Assuming x_t is one-dimensional and $g(x_t, y_t)$ can be reasonably approximated as a single Gaussian density, or another unimodal function, for a fixed y_t, then a simple method obtaining a suitable grid in a region where $\psi_t^\star(x_t)$ is large is as follows. For $t = T$ we can start near the mode of $g(x_T, y_T)$, with y_T fixed, and extending the grid outward in both directions until a suitable stopping point is reached, and then the grid will be evenly spaced points in this interval. One such stopping point is when $g(x_T, y_T) \approx 0$ or less than some predefined threshold and all the features of the target function will be located inside the grid since we assumed the target function was unimodal and we started near the mode. As it will be cumbersome to determine stopping points in this manner in higher-dimensional settings, this will primarily be suitable in lower-dimensional settings. This grid can then be used to obtain the approximation $\hat{\psi}_T(x_T)$, by minimizing distance measures such as (3.13) and (3.14) where the functions are computed on the grid as outlined in algorithm 4. Using the recursive definition from proposition 3.1.1 and (3.12) we can take a similar approach to determine a suitable grid for computing and approximating $\psi_{T-1}^\star(x_{T-1})$. Using (3.11) and starting near the mode of $g(x_{T-1}, y_{T-1})$ we can extend the grid until $g(x_{T-1}, y_{T-1})\overline{\psi}_{T-1}(x_{T-1}) \approx 0$ or some threshold is reached. Computing the target function on this grid can then be used to obtain the approximation $\hat{\psi}_{T-1}(x_{T-1})$, and this process can be repeated until $t = 1$ to obtain an approximation of the optimal sequence $\boldsymbol{\psi}^\star$. Note that there is no need starting at the mode. We do this simply because we want to avoid the tail of $\psi_t^\star(x_t)$ and it is unlikely that the mode of $g(x_t, y_t)$ is close to the tail of $\psi_t^\star(x_t)$.

A summary for creating the grid for approximating a function $\varphi(x)$ can be found in algorithm 5. For approximating the sequence $\boldsymbol{\psi}^\star$ with this algorithm we start at time T and set $\varphi(x_T) = g(x_T, y_T)$ to create the approximation of $\psi_T^\star(x_T)$, and $\varphi(x_t) = g(x_t, y_t)\overline{\psi}_t(x_t)$ for the remaining times. This can then be used in conjunction with algorithm 4 to obtain the approximations of the optimal sequence

Algorithm 5: Determine suitable grid in 1 dimension

```
1 Determine a suitable starting point $x$ where $\varphi(x) > \varepsilon$. Here $\varepsilon$ is some
    predefined threshold
2 Set $x_{max} = x + \kappa \cdot \delta$. Here $\delta$ is some predefined step size, and $\kappa \in \mathbb{N}$ is
    chosen such that $\varphi(x_{max}) < \varepsilon$ while $\varphi(x_{max} - \delta) \geq \varepsilon$.
3 Set $x_{min} = x - \kappa \cdot \delta$. Here $\delta$ is some predefined step size, and $\kappa \in \mathbb{N}$ is
    chosen such that $\varphi(x_{min}) < \varepsilon$ while $\varphi(x_{min} + \delta) \geq \varepsilon$.
4 The grid is then $N$ evenly space points between $x_{min}$ and $x_{max}$.
```

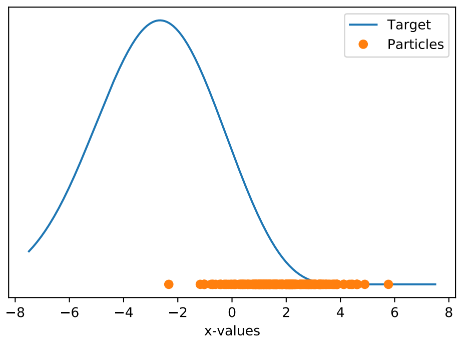

Figure 3.6: The location of the particles relative to the optimal function we want to approximate. Here all the particles are located near one of the tails. The target here is $g(x_t, y_t)\overline{\psi}_t(x_t)$ and the particles are from a bootstrap filter applied on the original model, which here is the Poisson model from Chapter 4.

The benefit here is that we avoid using the particle filter for approximating the optimal sequence $\boldsymbol{\psi}^\star$, and as we compute the target function and create the approximation based on a model specific grid there will be no need for an iterative setup. Furthermore, we create the approximation in the region we are evaluating the function which is also where we will find all the features of the target function, that is near its mode, for a unimodal function. If the function is multimodal, this setup will still work as long as $\varphi(x) > \varepsilon$ in the region between the tails. Additionally each time we run algorithm 4 we obtain the same approximation since the grid will be the same each time, this is in contrast to using the particles as these will vary each time we run the bootstrap particle filter. It is worth noting that this approach is limited to low dimensional settings and relatively simple models as determining the grid of points used will be cumbersome for complex models and in high dimensional settings. Further determining a starting point can be tedious when we have a long sequence of observations, and one possible solution to this is using a simple particle filter as we will discuss next. Also using this approach makes it so that the grid is not located in the tail of $\psi_t^\star(x_t)$ which can happen when the particles from the bootstrap particle filter are used directly. This is illustrated in Figure 3.6, and when this is the case it can become difficult to obtain good approximations. None of these particles provide much information about the mode of the optimal function which makes the task of creating a good parametric approximation that is centered around the mode difficult.

Extending the method

At times it can be difficult to determine a suitable starting point for algorithm 5 solely based on the emission density and the observations. We previously suggested starting near the mode of $g(x_t, y_t)$, however, it is not always clear where this is located, so now we will consider a setup where we run the particle filter once on the original model and use its output to provide the grid. The method proposed above for determining the grid can be altered slightly, where instead of extending the grid from or near the mode of $g(x_t, y_t)$ which presumably is unknown, we can extend the grid from e.g. the filter estimates based on a simple bootstrap filter. Alternatively, for each time t the grid can be extended from the particle with the highest weight in the bootstrap filter, as it is likely that this will be located close to the mode of $g(x_t, y_t)$ and in a region near the mode of $\psi_t^\star(x_t)$ a time t. The point here is to use the filter estimates of the states to provide a general region where $\psi_t^\star(x_t)$ is non zero, hence the quality of the estimates from the bootstrap filter is not of interest here, so it can be used with few particles. This slight adjustment removes the need to manually select a starting point for each t which in practice can become tedious.

While so far we have solely focused on approximating the optimal sequence $\boldsymbol{\psi}^\star$ in a single step, other works such as the previously mentioned Heng et al. (2020) and Guarniero, Johansen and Lee (2017) take an iterative approach. The latter of these use the particles from the bootstrap filter directly when creating the first approximation of the optimal sequence. We have proposed a slightly different method than using the particles, where rather than minimizing a distance measure with the particles as support, we create a suitable grid based on the model and the data to perform the minimization and create the approximation as in algorithm 5. Our motivation for taking a different approach is that we saw in the examples in Section 3.3 that we could obtain good approximation with simpler methods and these approximations were sufficient for providing estimates of the marginal likelihood with low variance.

The iterative approach can still be used in conjunction with our setup, and our proposed setup for creating the approximation can be extended into for instance the iterated auxiliary particle filter from Guarniero, Johansen and Lee (2017). Starting with our proposed setup for approximating the optimal sequence $\boldsymbol{\psi}^\star$, we can extend this with the iterative approach from Guarniero, Johansen and Lee (2017) and use the particles from the twisted model to improve the approximation of the optimal sequence. This allows us to obtain any benefit from the iterative approach while simultaneously making the first step more robust.

CHAPTER 4

Experimental results

4.1 Topics for numerical experiments

The previous examples, in Section 3.3, showed that introducing the twisted model can potentially reduce the variance of the marginal likelihood estimate obtain from the bootstrap filter. Now we are interested in a more detailed analysis. There are several possible approaches for creating these approximations, and we proposed a new and very simple setup which in practice is easier to use to create the approximation than the iterative methods. In this chapter, we are interested in exploring how well this simple setup performs and if it can be used to create a good approximation of the sequence $\boldsymbol{\psi}^\star$ with low variance estimates of the marginal likelihood on simulated data. Of interest is if our simple method can be used instead of a more complex iterative setup to create approximations of the sequence $\boldsymbol{\psi}^\star$ for simple models. In this setting, we refer to a good approximation as an approximation that reduces the variance in the estimates of the marginal likelihood.

In addition to variance reduction, we are interested in investigating other properties with these twisted models, including the importance of flexibility in the approximation, and other benefits of the twisted models in practice. In this context, we consider flexibility in the sense that we can obtain good approximations of a larger number of functions, and this is done by considering a larger class of functions for the approximation. Note that this class still needs to be somewhat limited to avoid issues with overfitting. We expect that when selecting an approximation from a class Ψ that there is a limit to how good this approximation can become within the class. Improving the flexibility will be key in order to obtain better approximations of the optimal sequence for several models. In the process, we also aim to investigate other key factors and gain experience working with these twisted models to determine what works and which pitfalls to avoid.

In Section 4.5 we will have a look at how the marginal likelihood estimates from a twisted model can be used in an MCMC setup akin to what was done in Section 2.5. In Section 4.6 we will focus on how an approximation of the sequence $\boldsymbol{\psi}^\star$ can be used to obtain estimates of the states. In particular, we will see how we can obtain high-quality smoothing estimates by sampling directly from the smoothing posterior in a different manner to the approaches discussed in Section 2.4.

For all the particle filters in this chapter we use algorithm 3 and resample when $N_{\text{eff}} < 0.5N$ unless something else is stated. For comparing the different models our main criterion will be the variance of the marginal likelihood estimates. In addition to the variance of the estimates, we will also look at how often the twisted model resamples and the effective number of samples. This can give an indication of whether we have a good approximation of the optimal sequence.

4.2 Experiment setups

We will compare several different methods for creating approximations of the sequence $\boldsymbol{\psi}^{\star}$, and here we will go through how we obtain each of the approximations we will use in Section 4.3. From Section 3.3 we already have one simple method for creating the approximations, and we will also consider the bootstrap particle filter with the original model. Here we will consider a few more approximations of the optimal sequence. In common for each of the approximations is that we use algorithm 4, but within this there is room for variations. The main variations we consider are

- The grid used in the minimization step to obtain the approximation. The approximations $\hat{\boldsymbol{\psi}}_{\text{org}}$, $\hat{\boldsymbol{\psi}}_{\text{log}}$, and $\hat{\boldsymbol{\psi}}_{\text{mixture}}$ use a fixed grid following algorithm 5 from Section 3.4. The remaining approximations $\hat{\boldsymbol{\psi}}_{\text{particles}}$, $\hat{\boldsymbol{\psi}}_{\text{it1}}$, and $\hat{\boldsymbol{\psi}}_{\text{it2}}$ uses the particles from a bootstrap filter.
- The distance measure used to determine the best approximation of each function, and we consider two alternatives. The approximations $\hat{\boldsymbol{\psi}}_{\text{org}}$, $\hat{\boldsymbol{\psi}}_{\text{particles}}$, $\hat{\boldsymbol{\psi}}_{\text{it1}}$, $\hat{\boldsymbol{\psi}}_{\text{it2}}$, and $\hat{\boldsymbol{\psi}}_{\text{mixture}}$ use the square distance from (3.13). The approximation $\hat{\boldsymbol{\psi}}_{\text{log}}$ uses the square of the difference on log scale given by (3.14).
- The class to which we restrict the functions. We consider two different classes for the approximations, in particular we use either a single Gaussian density or a Gaussian mixture with two components. The approximations $\hat{\boldsymbol{\psi}}_{\text{org}}$, $\hat{\boldsymbol{\psi}}_{\text{particles}}$, $\hat{\boldsymbol{\psi}}_{\text{it1}}$, $\hat{\boldsymbol{\psi}}_{\text{it2}}$, and $\hat{\boldsymbol{\psi}}_{\text{log}}$ uses the single Gaussian density. The approximation $\hat{\boldsymbol{\psi}}_{\text{mixture}}$ use the Gaussian mixture.
- The iterative steps. We consider if there is a benefit in introducing the iterative steps. The approximations $\hat{\boldsymbol{\psi}}_{\text{it1}}$ and $\hat{\boldsymbol{\psi}}_{\text{it2}}$ uses the particles from a twisted model in an iterative fashion in the same manner as Guarniero, Johansen and Lee (2017). Thus, obtaining the approximations requires that we create an approximation of the optimal sequence more than once. The approximations $\hat{\boldsymbol{\psi}}_{\text{org}}$, $\hat{\boldsymbol{\psi}}_{\text{particles}}$, $\hat{\boldsymbol{\psi}}_{\text{log}}$, and $\hat{\boldsymbol{\psi}}_{\text{mixture}}$ does not contain this iterative step.

These are the main variations between how we create the approximations we will consider in Section 4.3. For the approximations of the optimal sequence in Section 4.4, Section 4.5, and Section 4.6 we will not consider all these variations and primarily focus on the approach taken for $\hat{\boldsymbol{\psi}}_{\text{org}}$ and $\hat{\boldsymbol{\psi}}_{\text{mixture}}$. One reason for this is that in our preliminary experiments we found that using the particles directly from the bootstrap filter on the original model was not always ideal for creating the approximations. We will go into more detail on some of the issues we encountered when describing how we obtained the approximation $\hat{\boldsymbol{\psi}}_{\text{particles}}$, but now we will have a closer look at how we create the specific approximations.

First we will look at the approximation $\hat{\boldsymbol{\psi}}_{\text{org}}$, which comes from simple setup we proposed in Section 3.4. For this approximations we use a single Gaussian density as the approximation which is on the form

$$\psi_t(x_t) = \mathcal{N}(x_t; \mu, \sigma^2). \tag{4.1}$$

To obtain the approximation of the optimal sequence we will follow algorithm 4. For the distance measure we use (3.13), and for the grid where we create the approximations we will use algorithm 5 where we set $\varepsilon = 10^{-3}$ and $\delta = 0.25$ to create a grid consisting of 250 points. Additionally we set $\varphi(x_t) = g(x_t, y_t)\overline{\psi}_t(x_t)$ which is the right-hand side from (3.12). Here $\overline{\psi}_t(x_t)$ is the approximation of $\widetilde{\psi}_t(x_t)$ from (3.11). For a starting point we run the bootstrap filter once with 100 particles, and use the particle with the highest weight at time t as the starting point for $\hat{\psi}_t(x_t)$. For obtaining starting points for the minimization, we exploit that we can compute $g(x_t, y_t)\overline{\psi}_t(x_t)$ which the right-hand side of (3.12) pointwise on the grid. Using this we can create simple Monte Carlo estimates of the expected value and variance of $\psi^\star_t(x_t) \approx g(x_t, y_t)\overline{\psi}_t(x_t)$ which we can use as starting points for the minimization. This setup provides the approximation $\hat{\boldsymbol{\psi}}_{\text{org}}$, and we will take the same approach for providing most of the other approximations with the same settings in the algorithms unless otherwise is stated.

We want to compare this with using the particles directly as the grid points, and this is $\hat{\boldsymbol{\psi}}_{\text{particles}}$. Here we still want approximations of the optimal functions on the form (4.1). To obtain the approximation of the optimal sequence we will follow algorithm 4. For the distance measure, we use (3.13), and for the grid, we run the bootstrap particle filter with 500 particles and use the particles as the gridpoints. However, for some times t with this model, we encountered the issue present in Figure 3.6, where all the particles are located in the tails of the function we aim to approximate. This makes it difficult to obtain sensible starting points for the approximation which is parametrized by an expected value and a variance. Thus we use the same starting points as for $\hat{\boldsymbol{\psi}}_{\text{org}}$. These are simple to obtain, and since we are still creating an approximation of the same function, it is reasonable to assume that these will be suitable.

Even with this inclusion, we are unable to obtain good approximations for every time t with the particles directly using the minimization setup we employ. Although steps can be taken to make the minimization more robust which could allow for the particles to be used directly, we instead use the same approach as for $\hat{\boldsymbol{\psi}}_{\text{org}}$ at these times. That is, we use a fixed grid to provide the approximation, and we do not use the particles at these times. This occurred at 3 separate times and brings a major issue with solely using the particles to light. Specifically, the particles are not always suited for creating the approximations. While it can be possible to make minimization more robust, e.g. by adding a penalization term for the parameters, it gives merit to using a simpler grid where the same issues are not encountered. With this setup, we get the sequence $\hat{\boldsymbol{\psi}}_{\text{particles}}$ which also is an approximation to the optimal sequence.

We also want to discuss how changing the distance measure influences the approximations. In particular, we want to use (3.14) as the distance measure, which works on a log scale. To do this we create the approximation $\hat{\boldsymbol{\psi}}_{\log}$ by repeating the process taken to obtain $\hat{\boldsymbol{\psi}}_{\text{org}}$, but changing the distance measure to (3.14).

Guarniero, Johansen and Lee (2017) propose an iterative method creating the approximation of the optimal sequence. This is done by using the particles from the bootstrap filter run on a twisted model as the grid points when creating the approximations. Of interest is seeing if this can improve the approximation within a class Ψ, which in this case is functions on the form (4.1). To do this we start with running the bootstrap particle filter on the twisted model based on $\hat{\boldsymbol{\psi}}_{\text{org}}$ with 500 particles which give the gridpoints. To obtain the approximation we process for obtaining $\hat{\boldsymbol{\psi}}_{\text{org}}$ but changing the grid points to the particles from the twisted model. This gives the approximation $\hat{\boldsymbol{\psi}}_{\text{it1}}$. Of interest is also using multiple iterative steps, hence we run the bootstrap particle filter on the twisted model based on $\hat{\boldsymbol{\psi}}_{\text{it1}}$ with 500 particles to provide a new set of gridpoints. Using the same process taken for obtaining $\hat{\boldsymbol{\psi}}_{\text{org}}$, we obtain the approximation $\hat{\boldsymbol{\psi}}_{\text{it2}}$ by replacing the grid for the minimization.

All the approximations we have considered so far have been on the form (4.1). We now want to expand this class and obtain the approximation $\hat{\boldsymbol{\psi}}_{\text{mixture}}$. In particular, we want approximations on the following form

$$\psi_t(x_t) = c_1\mathcal{N}(x_t, \mu_1, \sigma_1^2) + (1 - c_1)\mathcal{N}(x_t, \mu_2, \sigma_2^2). \tag{4.2}$$

Expanding the class to functions on the form (4.2) will allow us to obtain good approximations of a larger number of functions. However, the process of obtaining the approximations is also more involved, especially if we want to use the simple setup we proposed for creating the approximation. Under the assumption that each $g(x_t, y_t)\overline{\psi}_t(x_t) > \varepsilon$ between the tails or is unimodal we can still use our simple setup to provide the approximation. This is because we want to determine a region where $g(x_t, y_t)\overline{\psi}_t(x_t) > \varepsilon$. To verify that this assumption holds, we take a straightforward approach where we create the approximations and look at the resulting function $g(x_t, y_t)\overline{\psi}_t(x_t)$. For this example, we were able to verify that the assumption holds and we used the same setup as for $\hat{\boldsymbol{\psi}}_{\text{org}}$ to provide the approximation up to the starting points for the minimization. Since Gaussian mixtures on this form are not unique up to a set of parameters, i.e. multiple different combinations of parameters give the same function, there are multiple minima in this minimization problem. What we found to work well as starting points here was giving one mode the parameters obtained when using a single Gaussian density as the approximation and treating the other as a correction term. This gave the approximation $\hat{\boldsymbol{\psi}}_{\text{mixture}}$.

With each of the methods for obtaining an approximation of $\boldsymbol{\psi}^{\star}$, we expect to see little variation in the approximation if the process is repeated since the approximations are of the same function. We also experienced this when working with these models, and when creating several approximations of the optimal sequence with the same method, the resulting approximations were quite similar. This was also the case when using the particles for the grid. For this reason, we only create one approximation $\hat{\boldsymbol{\psi}}$ with each method, and to compare the approximations run the twisted model several times based on each of the approximations.

For comparing the different methods, we will primarily consider two points. First, we consider the standard deviation of the marginal likelihood estimates. We do this by creating several estimates of the marginal likelihood with the

Approximations of $\boldsymbol{\psi}^\star$		
Approximation	Class	Grid and Distance measure
$\hat{\boldsymbol{\psi}}_{\text{org}}$	Gaussian Density	Algorithm 5 with $\varepsilon = 10^{-3}$ $\delta = 0.25$ $\varphi(x_t) = g(x_t, y_t)\overline{\psi}_t(x_t)$ $\sum_{i=1}^{N} \left[\psi_t(x^i) - \lambda g(x^i, y_t)\widetilde{\psi}_t(x^i)\right]^2$
$\hat{\boldsymbol{\psi}}_{\text{particles}}$	Gaussian Density	Particles from bootstrap filter (see additional notes above) $\sum_{i=1}^{N} \left[\psi_t(x^i) - \lambda g(x^i, y_t)\widetilde{\psi}_t(x^i)\right]^2$
$\hat{\boldsymbol{\psi}}_{\text{log}}$	Gaussian Density	Algorithm 5 with $\varepsilon = 10^{-3}$ $\delta = 0.25$ $\varphi(x_t) = g(x_t, y_t)\overline{\psi}_t(x_t)$ $\sum_{i=1}^{N} \left[V_t(x^i) + \log[\lambda g(x^i, y_t)\widetilde{\psi}_t(x^i)]\right]^2$
$\hat{\boldsymbol{\psi}}_{\text{it1}}$	Gaussian Density	Particles from bootstrap filter with twisted model based on $\hat{\boldsymbol{\psi}}_{\text{org}}$ $\sum_{i=1}^{N} \left[\psi_t(x^i) - \lambda g(x^i, y_t)\widetilde{\psi}_t(x^i)\right]^2$
$\hat{\boldsymbol{\psi}}_{\text{it2}}$	Gaussian Density	Particles from bootstrap filter with twisted model based on $\hat{\boldsymbol{\psi}}_{\text{it1}}$ $\sum_{i=1}^{N} \left[\psi_t(x^i) - \lambda g(x^i, y_t)\widetilde{\psi}_t(x^i)\right]^2$
$\hat{\boldsymbol{\psi}}_{\text{mixture}}$	Gaussian Mixture	Algorithm 5 with $\varepsilon = 10^{-3}$ $\delta = 0.25$ $\varphi(x_t) = g(x_t, y_t)\overline{\psi}_t(x_t)$ $\sum_{i=1}^{N} \left[\psi_t(x^i) - \lambda g(x^i, y_t)\widetilde{\psi}_t(x^i)\right]^2$

Table 4.1: List of the different approximation of $\boldsymbol{\psi}^\star$, see previous paragraphs for further details.

twisted models obtained from the different methods and computing the standard deviation of the estimates obtained. This allows us to determine if we have been successful when it comes to variance reduction, which was the motivation for introducing the twisted models. Additionally, we have the resampling count of the different methods. For the exact optimal sequence, the bootstrap filter should never resample with the twisted model, and with a good approximation, we expect to see little resampling in the twisted model. In relation to the resampling, we can also look at the effective number of samples, and for a good approximation, we expect this to be close to the number of particles used in the filter and remain constant.

4.3 Model and results

Here we will consider a state space model where the state transitions are linear and Gaussian while the emissions are Poisson distributed. This is the same model structure as in one of the previous examples in Section 3.3 where we saw an improvement by introducing the twisted model. This model can be described as follows

$$\mu(x_1) = \mathcal{N}(x_1; 0.85, 1)$$

$$f(x_{t-1}, x_t) = \mathcal{N}(x_t; 0.85 + 0.7x_{t-1}, 1), \text{ for } t \in 2, \dots, T$$
$$g(x_t, y_t) = \frac{\exp(x_t)^{y_t}}{y_t!} \exp(-\exp(x_t)).$$

We will also use the same sequence of simulated observations as in Section 3.3. This allows us to directly compare the results here to those obtained earlier. Again we will look at the ratio Z^N/Z, where Z is the likelihood estimate based on the same particle filter with 10000 particles as used in the previous example in Section 3.3. The twisted models we consider will be based on the approximations described in Section 4.2.

Experiment 1 - Systematically laying out the grid

We now want to see how our approach with systematically laying out the grid with the method from Section 3.4 performs, and compare the results here with those from the example in Section 3.3. Here we are interested in the twisted models given by the sequences $\hat{\boldsymbol{\psi}}_{\text{org}}$ and $\hat{\boldsymbol{\psi}}_{\text{particles}}$. For the comparison, we have used $N = 500$ particles and resample when $N_{\text{eff}} < 0.5N$ in the bootstrap filter for the original model. For all the twisted models we have used $N = 125$ particles and again resampling when $N_{\text{eff}} < 0.5N$. To provide a comparison we run all the particle filters 100 times for each model and again look at the ratio Z^N/Z.

Results

Figure 4.1 shows boxplots comparing the 100 likelihood estimates for the different approaches, and Table 4.2 provides a summary of the results. Again we see that the twisted model improves the likelihood estimates when we provide an adequate approximation of the optimal sequence even with far fewer particles. For this example, all three approaches for creating the approximation provide similar results. Additionally, looking at parameters in the approximations directly (not shown here) we see that these are very similar. The difference between the parameters giving the expected value of the functions in $\hat{\boldsymbol{\psi}}_{\text{org}}$ and $\hat{\boldsymbol{\psi}}_{\text{particles}}$ are on an order of magnitude 10^{-2} to 10^{-5} and the parameters themselves are on an order of magnitude around 10^0 to 10^1. This is a clear indication that all the approximations are quite similar. The same is true when comparing $\hat{\boldsymbol{\psi}}_{\text{org}}$ and the sequence from the subjectively chosen grid where the differences in the parameters are on an order of magnitude 10^{-2} to 10^{-8}. Since all the twisted models obtained are similar, we also expect that the results produced are similar which is what we observe.

With a similar standard deviation of all 100 estimates of the three twisted models, it is difficult to assert that one method is better than the others. All the twisted models provide lower variance estimates of the marginal likelihood than the original model and can in practice be good alternatives to using the original model directly.

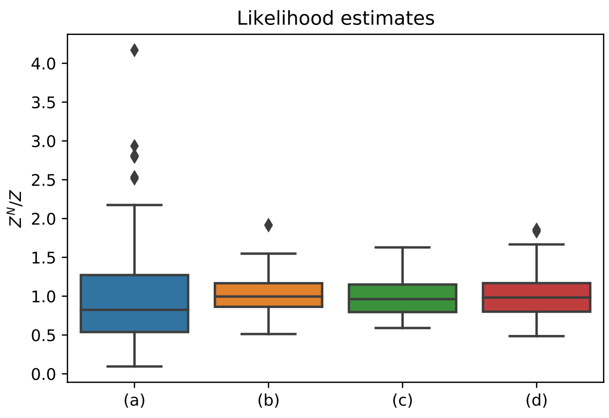

Figure 4.1: Boxplots of the 100 estimates of the likelihood ratio Z^N/Z based the different models, with approximations of the optimal sequence based on (3.13). Here (a) is bootstrap particle filter applied while the others are based on twisted models where the optimal sequence has been approximated using different grids. The different grids are (b) a subjectively chosen grid (same as in Section 3.3), (c) $\hat{\psi}_{\text{particles}}$, (d) is based on $\hat{\psi}_{\text{org}}$

Results for estimating the likelihood ratio Z^N/Z		
Model (Number of particles)	Standard deviation of the estimates	Average resampling count
Original model (500)	0.673	88.63 ± 0.59
Twisted model with subjectively chosen grid (125)	0.265	4.33 ± 0.60
Twisted model with $\hat{\psi}_{\text{particles}}$ (125)	0.242	3.19 ± 0.75
Twisted model with $\hat{\psi}_{\text{org}}$ (125)	0.269	4.54 ± 0.57

Table 4.2: Summary of the likelihood estimates when using the distance measure (3.13) with different grids for providing the approximations of the optimal sequence. The leftmost column shows which model is used alongside the number of particles. The middle column shows the standard deviation of the 100 estimates of the ratio Z^N/Z obtained for each model. The rightmost column shows the average ($\pm$ standard deviation) number of times resampling occurred in the particle filter for each of the models.

Experiment 2 - Changing distance measure

We can also consider using a different distance measure for determining the best approximation within the class Ψ. Of interest here is the sequences $\hat{\psi}_{\mathrm{org}}$ and $\hat{\psi}_{\mathrm{log}}$

We can compare the results based on $\hat{\psi}_{\mathrm{org}}$ with those based on $\hat{\psi}_{\mathrm{log}}$. Again, we will run the particle filter on the twisted model 100 times to provide estimates we can use for comparison and use the same rules for resampling in the model. That is for each particle filter with the twisted model we use $N = 125$ particles and resample when $N_{\mathrm{eff}} < 0.5N$.

Results

Figure 4.2 shows boxplots comparing the likelihood estimates based on $\hat{\psi}_{\mathrm{org}}$ and $\hat{\psi}_{\mathrm{log}}$, and a summary of the results can be found in Table 4.3. For this model it appears that using $\hat{\psi}_{\mathrm{org}}$ provides slightly lower variance likelihood estimates than using $\hat{\psi}_{\mathrm{log}}$. In addition, using $\hat{\psi}_{\mathrm{log}}$ gives a slightly higher resample rate, but these differences are minor and again the twisted models outperform the original model even with far fewer particles. Again the difference in the results can also come from only creating 100 estimates of the marginal likelihood, and from the boxplot, we observe that there are a couple of outliers with $\hat{\psi}_{\mathrm{log}}$ which increases the variance.

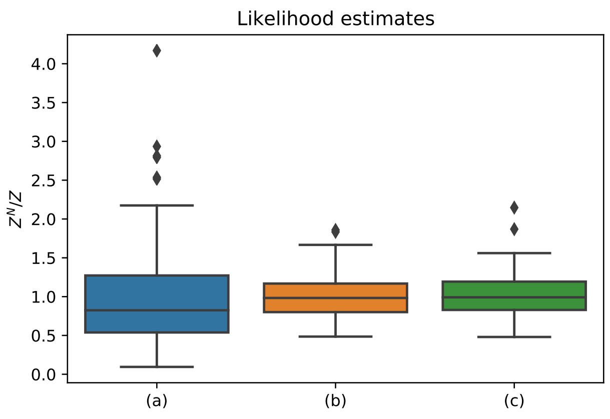

Figure 4.2: Boxplots of the 100 estimates of the likelihood ratio Z^N/Z based the different models, with approximations of the optimal sequence based on the distance measures (3.13) and (3.14). Here (a) is the bootstrap filter applied on the original model, (b) twisted model based on $\hat{\psi}_{\mathrm{org}}$, and (c) twisted model based on $\hat{\psi}_{\mathrm{log}}$

.

Results for estimating the likelihood ratio Z^N/Z		
Model (Number of particles)	Standard deviation of the estimates	Average resampling count
Original model (500)	0.673	88.63 ± 0.59
Twisted model with $\hat{\boldsymbol{\psi}}_{\mathrm{org}}$ (125)	0.269	4.54 ± 0.57
Twisted model with $\hat{\boldsymbol{\psi}}_{\mathrm{log}}$ (125)	0.272	5.80 ± 0.66

Table 4.3: Summary of the likelihood estimates when using the approximations $\hat{\boldsymbol{\psi}}_{\mathrm{org}}$ and $\hat{\boldsymbol{\psi}}_{\mathrm{log}}$. The leftmost column shows which model is used alongside the number of particles. The middle column shows the standard deviation of the 100 estimates of the ratio Z^N/Z obtained for each model. The rightmost column shows the average ($\pm$ standard deviation) number of times resampling occurred in the particle filter for each of the models.

Experiment 3 - Iterative setup

Now we will have a look at the effect of the iterative procedure, and if using the particles from a twisted model can provide an even better approximation of the optimal sequence. When estimating the ratio Z^N/Z all the twisted models were quite similar, so we expect that expanding to an iterative setup starting from any twisted models we have considered so far will provide similar results. Here we are interested in the sequences $\hat{\boldsymbol{\psi}}_{\mathrm{it1}}$, $\hat{\boldsymbol{\psi}}_{\mathrm{it2}}$, and $\hat{\boldsymbol{\psi}}_{\mathrm{org}}$

For comparing the twisted models based on $\hat{\boldsymbol{\psi}}_{\mathrm{it1}}$ and $\hat{\boldsymbol{\psi}}_{\mathrm{it2}}$ we take the same approach as we did for the previous models by creating 100 estimates of the marginal likelihood using algorithm 3. Again we use $N = 125$ particles in the bootstrap filter with the twisted models and resample when $N_{\mathrm{eff}} < 0.5N$.

Results

We already know that the twisted model performs better than the original model, and here we are primarily interested in the effect of including an iterative step. Figure 4.3 shows boxplots of the 100 marginal likelihood ratio estimates from the different models, and from this figure, we get an indication that using the iterative approach may provide a slight improvement. This is shown further in Table 4.4 where both the standard deviation of the marginal likelihood the average resampling count is lower when we use the iterative approach. The improvement seen by introducing the iterative steps is smaller than what we observed when going from the original model to the first twisted models. Since creating the iterative approximation is more expensive than just the single approximation such as $\hat{\boldsymbol{\psi}}_{\mathrm{org}}$, it is worth considering the benefit of introducing the iterative procedure.

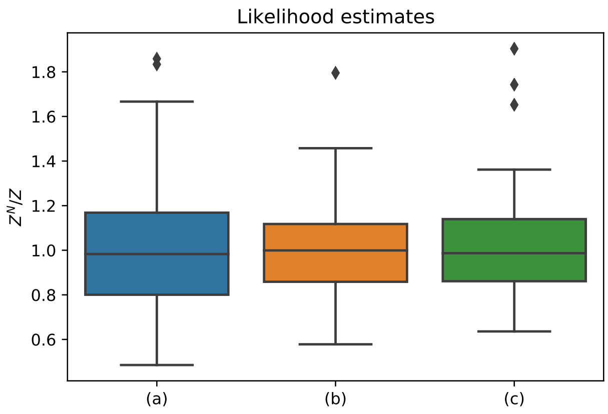

Figure 4.3: Boxplots of the 100 estimates of the marginal likelihood ratio Z^N/Z based on different twisted models. Here (a) is $\hat{\psi}_{\text{org}}$, (b) is $\hat{\psi}_{\text{it1}}$ and (c) is $\hat{\psi}_{\text{it2}}$

Results for estimating the likelihood ratio Z^N/Z		
Model (Number of particles)	Standard deviation of the estimates	Average resampling count
Twisted model with $\hat{\psi}_{\text{org}}$ (125)	0.269	4.54 ± 0.57
Twisted model with $\hat{\psi}_{\text{it1}}$ (125)	0.194	2.81 ± 0.56
Twisted model with $\hat{\psi}_{\text{it2}}$ (125)	0.222	2.97 ± 0.60

Table 4.4: Summary of the likelihood estimates when using the approximations $\hat{\psi}_{\text{org}}$ and the iterative approximations. The leftmost column shows which model is used alongside the number of particles. The middle column shows the standard deviation of the 100 estimates of the ratio Z^N/Z obtained for each model. The rightmost column shows the average ($\pm$ standard deviation) number of times resampling occurred in the particle filter for each of the models.

Experiment 4 - Flexibility of the approximation

Having so far focused solely on an approximation based on a single Gaussian density, we now want to employ a more flexible approximation that should admit better approximations of the functions in the optimal sequence. Here we are interested in the sequences $\hat{\psi}_{\text{org}}$, $\hat{\psi}_{\text{it1}}$, and $\hat{\psi}_{\text{mixture}}$, where the last one is based on an extended class using two components in the Gaussian mixture.

For comparing the methods we still use $N = 125$ particles in the bootstrap filter for the twisted models, and resample when $N_{\text{eff}} < 0.5N$.

Results

Figure 4.4 shows boxplots of the estimates from the different approaches. Here we see that using a more flexible approximation gives the lowest variance

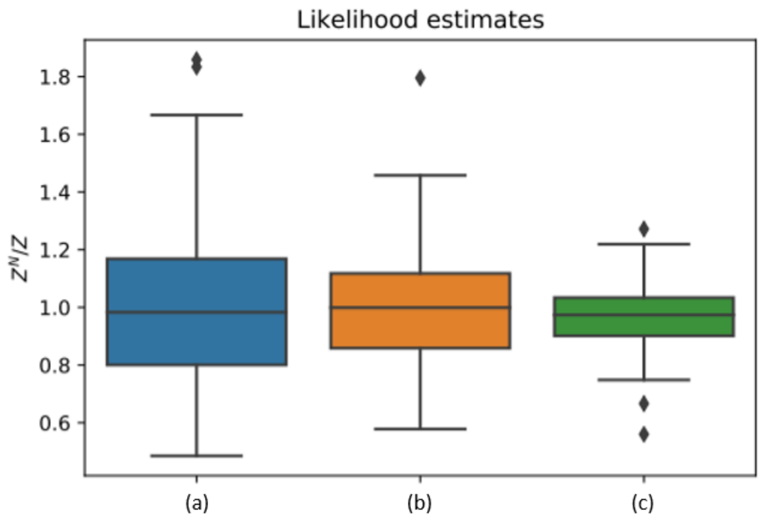

Figure 4.4: Boxplots of the 100 estimates of the likelihood ratio Z^N/Z based on different approximation of the optimal sequence. Here (a) is based on $\hat{\psi}_{\text{org}}$, (b) is based on $\hat{\psi}_{\text{it1}}$ and (c) is based on $\hat{\psi}_{\text{mixture}}$

estimates of the marginal likelihood of all the methods we considered. This is also made clear by table 4.5 which shows some of the numerical results. We observe a significant improvement in the variance estimates obtained by expanding the class Ψ, giving a better approximation in a single step than approximation obtained with the iterative setup. It is not surprising that expanding the class permits a better approximation. The expanded class (4.2) contains good approximations of a larger class of functions than the single Gaussian density. However, since the particle filter still resamples for this twisted model there is a potential for further improvement, either within the current class or by expanding the class even further. This example makes it clear that there is a benefit in expanding the class to improve the approximations.

Results for estimating the likelihood ratio Z^N/Z		
Model (Number of particles)	Standard deviation of the estimates	Average resampling count
Twisted model with $\hat{\psi}_{\text{org}}$ (125)	0.269	4.54 ± 0.57
Twisted model with $\hat{\psi}_{\text{it1}}$ (125)	0.194	2.81 ± 0.56
Twisted model with $\hat{\psi}_{\text{mixture}}$ (125)	0.134	1.04 ± 0.24

Table 4.5: Summary of the likelihood estimates when using the approximations $\hat{\psi}_{\text{org}}$, the iterative approximation and the more flexible approximation. The leftmost column shows which model is used alongside the number of particles. The middle column shows the standard deviation of the 100 estimates of the ratio Z^N/Z obtained for each model. The rightmost column shows the average ($\pm$ standard deviation) number of times resampling occurred in the particle filter for each of the models.

Summary and discussion of these results

Now we have made a significant effort to show that these twisted models can provide lower variance estimates of the marginal likelihood than using the original model directly, and there are several points that merit further discussion both regarding the approach taken and the results obtained.

For the Poisson model, that we have considered here, it is clear that there is a benefit in introducing the twisted models. We considered several methods for approximating the optimal sequence, and each of these methods gave an approximation $\hat{\psi}$ which we used to define a twisted model. For each of the models that we considered we created 100 estimates of the marginal likelihood. That is for each sequence $\hat{\psi}$, we estimated the marginal likelihood of the corresponding twisted model 100 times to investigate the variance with this particular sequence which was of interest. All the methods we considered gave twisted models with lower variance estimates of the marginal likelihood than the original model, with far fewer particles used in the twisted models. This makes it clear that there is a benefit of introducing the twisted models since we can lower the variance of the likelihood estimates which is what we set out to do when introducing the twisted models. The twisted models are also faster in terms of computational time for obtaining low variance estimates since we can use fewer particles, provided we can obtain a sufficiently good approximation of $\psi^\star$ in a reasonable time.

For comparing the different setups we based our conclusions on a single approximation of $\psi^\star$ for each method. This is because when working with these examples we noticed that creating several approximations would be somewhat superfluous since the parameters in each approximation would be almost identical. Thus if we were to base the estimates on different approximations from the same method, there would be little change in the results since the twisted models would remain virtually unchanged. We also observed this when using the particles as the gridpoints for the approximations. To compare the different models and approximations $\hat{\psi}$ we primarily used the variance of the marginal likelihood estimates. This is because we introduced the twisted model to obtain estimates of the marginal likelihood with lower variance than those from the original model, hence it is a natural measure to see if we have been successful.

One method for comparing the different approaches that we are yet to discuss in detail is the effective number of samples, N_{eff}, as defined in (2.13). While this is directly related to the resampling rate which we observed was smaller for all the twisted models we considered, we can also get some information directly from the effective number of samples. When we have a twisted model based on a good approximation of the optimal sequence, then the effective number of samples in the bootstrap filter should stay approximately constant and be close to the actual number of particles. This is because for a good approximation all the importance weights will be similar by the construction of the twisted model. In particular, we create the approximations such that $\hat{\psi}_t(x_t) \approx g(x_t, y_t)\overline{\psi}_t(x_t)$ which makes the weights of the twisted model given by (3.3) and (3.4) similar for a good approximation. Figure 4.5 shows the effective number of samples in the original model and three of the twisted models we considered. For the twisted models, N_{eff} being constant or slowly decreasing is an indication that we have a good approximation of the optimal sequence at this time. Observing sudden drops is an indication that we have a poor approximation which can be improved and jumps upwards, are a result of resampling. Using the N_{eff} in such a manner allows us to determine where we should focus if we want to improve the approximation further. Looking at $t = 63$, it appears that the approximations are poor, and there is potential for improvement. The reason for this may be that at $t = 63$ we have $y_{63} = 0$ which makes $\psi^{\star}_{63}(x_{63})$ difficult to approximate within our chosen class due the shape of $g(x_{63}, y_{63})$ which the optimal function at this time is proportional to. Hence, in the case of the Poisson model considered here, it appears that using an approximation on the form (4.2) is not sufficient, and there is potential for further improvements. Using the effective number of samples in such a manner allows us to diagnose which function approximations are good and which requires further attention in order to obtain a good approximation of the entire optimal sequence $\boldsymbol{\psi}^{\star}$.

As an additional note when looking at the effective number of samples for the good approximation, especially $\hat{\boldsymbol{\psi}}_{\text{mixture}}$, we note that the effective number of samples mostly remains constant. The times where we observe drops it is likely due to a poor approximation, and we see an increase when we resample. If the goal is to keep the effective number of samples high at all times it can be beneficial to be relatively liberal in how often resampling is done. This is because with a moderate resampling rate it is possible that we never resample but remain with a relatively low effective number of samples for a longer period of time since it will be slowly decreasing, which is seen for $\hat{\boldsymbol{\psi}}_{\text{mixture}}$. Hence a possible situation is where the effective number of samples is around $0.6N$ for a long period of time which is not low enough to resample but could be increased by resampling.

One limiting factor mentioned in Section 3.2 is the restriction of the class of functions for the approximation. When choosing an approximation from a class Ψ we are limited to functions within this class, and if the optimal $\psi^{\star}_t(x_t)$ is not in this class we cannot obtain optimal sequence exactly and there is a limit to how good of an approximation we can obtain. Hence we want a flexible class to make sure that functions from the class Ψ will provide a good approximation

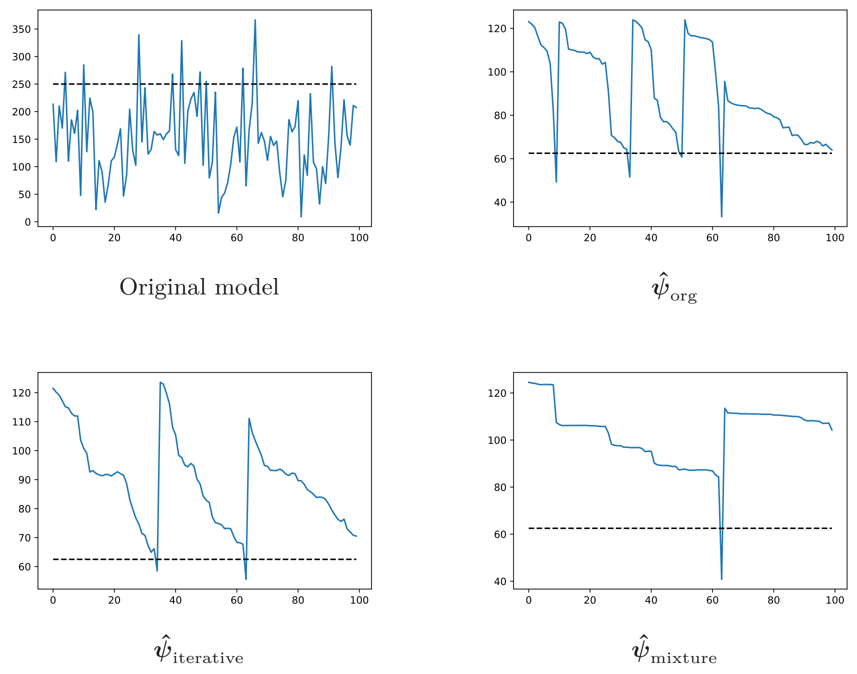

Figure 4.5: Comparison of the effective number of samples in the different models when the bootstrap filter is applied. For the original, untwisted model we have used 500 particles in the bootstrap filter, and we resample when $N_{\text{eff}} < 250$. For the twisted models we have used 125 particles and resample when $N_{\text{eff}} < 62.5$

to each of the functions in the sequence $\boldsymbol{\psi}^{\star}$.

For the first approximations, we used a relatively simple parametric approximation of the optimal sequence, namely a single Gaussian density, and yet we were still able to improve the marginal likelihood estimates by introducing the twisted model. This simple parametric approximation did not provide the exact optimal sequence in part due to lack of flexibility and using a more complex parametric approximation provided an even better approximation of the optimal sequence. Introducing the iterative step on the simple approximation $\hat{\boldsymbol{\psi}}_{\text{org}}$ gave a better approximation at the points where N_{eff} dropped, as seen in Figure 4.5. This was still done within the same simple class Ψ and indicates that there is some merit in using the iterative setup to further improve the approximation within the class. It is worth noting that the variance reduction from the iterative step is relatively small especially when comparing to the reduction from introducing the twisted models in the first place. Since we also are able to obtain variance reduction from increasing the number of particles, introducing the iterative step should give lower variance than reducing the number of particles to be viable. The variance of Monte Carlo estimates is proportional to $\frac{1}{N-1}$, where N is the number of samples(Givens and Hoeting, 2013). Thus, if including a new iteration takes as long as doubling the number of particles, the variance of the estimates should be reduced by more than half for the additional iteration to be viable. This also brings up the question of why the iterative setup gives a better approximation since we are creating approximations of the same function based on similar criteria. The key here is that the criteria are similar but not identical as the functions are evaluated at different points when creating the approximations. Of interest would then be to mimic the effect of using the particles from a twisted model for creating the approximation within the simple setup we proposed. This can be done by changing how the grid is laid out or by changing the distance measure used, and then we would be able to obtain the same approximations but this time in a single step.

While this simple parametric structure that was chosen provided an adequate approximation of the optimal sequence which reduced the variance of the marginal likelihood estimates in this Poisson model, the target functions we are approximating are themselves not Gaussian densities, hence there is still potential for improvement. This is seen in Figure 3.5 where some of the functions we are approximating are slightly skewed and not Gaussian, hence a Gaussian approximation will not be perfect. As a result of this, we also considered a more flexible class that is able to provide good approximations of more functions.

When using the more flexible approximation on the form (4.2), we observe that the variance of the marginal likelihood estimates based on this twisted model is smaller than those obtained from all the twisted models using a single Gaussian density since the approximation of the optimal sequence is better. This is shown by the boxplots in Figure 4.4, and in our example using a flexible approximation can give a larger improvement than using an iterative procedure. This is expected since there is a limit to the quality of the approximation that can be obtained within a class Ψ and expanding that class can provide a better approximation. It is possible that we were close to the limit of how good of an approximation we can obtain with the single Gaussian density with a single iterative step and making the approximation more flexible immediately allowed us to obtain a better approximation. It still bears consideration if using a more complex parametric approximation is necessary for practical use since using a more flexible approximation makes determining the best approximation within the class Ψ more complicated and has a greater computational cost.

In terms of the computational cost, using a twisted model is often preferable to the original model. The particle filter itself is faster since one can use fewer particles, but we also have to account for the time required to create the approximations of the optimal sequence which increases the computational cost. Our focus here has not been on the computational cost, but from our experience working with the twisted models, we observe that at a similar computational time the twisted models provide lower variance estimates than the original model with the bootstrap filter when accounting for the time taken to obtain the approximation.

To determine the best approximation we also considered, albeit briefly, using a different distance measure for the optimization. There are several possibilities for the distance measure including the square error of (3.13) and a logarithmic version in (3.14) both of which we considered above, and another natural candidate is the Kullback-Leibler divergence. While we saw little effect here, the motivation is that different distance measures will focus on creating the best approximation in different regions. This is similar to how using the particles directly for the approximation allows us to focus on a particular region where the parameters are located, changing the distance measure could be done to obtain a similar effect. If we want a good approximation in a particular region, we can use a distance measure that heavily penalizes errors in this region.

For practical use, all the twisted models we have considered provided lower variance estimates of the marginal likelihood with the bootstrap filter than the original model, and using a crude approximation in the twisted model can be better than not using a twisted model at all.

All the setups we considered gave approximations of the optimal sequence. For practical use, all the twisted models would be preferable over the original since they all provide low variance estimates of the marginal likelihood using far fewer particles. For the end result, it matters little how the approximation is obtained, other than that the approximation must be one we are able to work with and is obtained in a reasonable amount of time. The iterative approaches can be used to obtain good approximations of the optimal sequence within a specific class Ψ and performed better than our simple setup for this model when it comes to creating the best approximation. However, if we can obtain a better approximation in a different manner it becomes superfluous. In practice, it appears that flexibility in the approximation is desirable, as it is the most flexible approximation that gives the best results in our experiments with this model. When expanding the class is viable and gives a better approximation than the iterative approach, the iterative approach becomes redundant. There is an argument to be made that the iterative approach can be used on the expanded class to improve the approximation further, however, there is again the question of how much that can be gained by the iterative approach. For simple models as in this example, if given the choice between using an iterative approach in a simple class or expanding this class, it will often be better to expand the class since this is likely to produce the best approximation.

4.4 Other factors related to twisted models

Reducing the number of particles for the twisted model

In the previous section, our focus was on creating the best approximation within a class Ψ, but it is also natural to consider if using a good approximation allows for greater freedom in the particle filter itself. When using the bootstrap particle filter on the original model, increasing the number of particles will reduce the variance of the estimate produced. The same is true for the twisted model and we will have a look at the effect from the number of particles in the twisted model. As we have seen when using the twisted model, we can get lower variance estimates of the marginal likelihood with far fewer particles in the bootstrap filter compared original model. Hence, it is natural to consider if there is some lower limit to the number of particles we can use. If we are able to obtain the exact optimal sequence the answer is that we only need a single particle, but when we only have an approximation of the optimal sequence it is not so clear. To have a closer look at this we will consider again the Poisson model from the previous section, but this time with a shorter sequence of observations. For this model, we know that we can obtain low variance estimates provided we have enough particles, so it will be of interest to see how few particles we can use here.

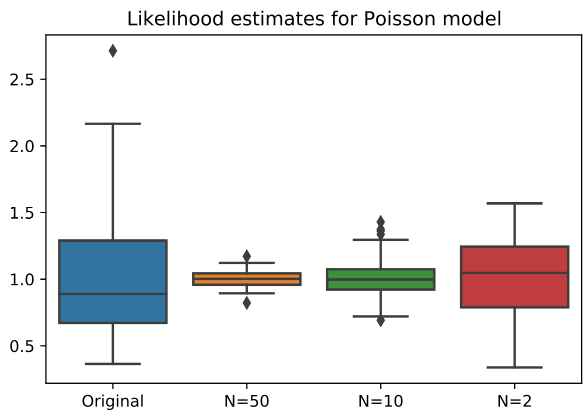

Figure 4.6: Boxplot of the 100 estimates in the Poisson model of the likelihood ratio Z^N/Z with the different numbers of particles in the bootstrap filter. We also include the original model as a comparison where the estimates are based on 500 particles in the bootstrap filter.

Poisson model

We consider the same model as in Section 4.3 and the approximation of the optimal sequence is made with the same approach as $\hat{\psi}_{\text{mixture}}$, still using a Gaussian mixture density as the approximation. This time we will simulate a sequence with 25 observations.

To see the effect of the number of particles used we again run repeated simulations and estimate the marginal likelihood with a different number of particles in the bootstrap filter for the twisted model. Here we consider $N = 50$, $N = 10$ and $N = 2$ particles in the bootstrap filter and resample when $N_{\text{eff}} < 0.5N$. With each of the particle configurations, we run the bootstrap filter 100 times, to get a large number of estimates of the marginal likelihood. Again, we consider the likelihood ratio Z^N/Z, where Z is the average of the 100 likelihood estimates obtained from the bootstrap particle filter on the twisted model with $N = 50$ particles. We use this as we know this value converges to the true likelihood, and this will be much faster than running a new particle filter with a lot of particles.

Figure 4.6 shows boxplots with comparisons of using different particles. Clearly using fewer particles increases the variance of the estimate obtained with the twisted model as expected. Comparing this with estimates based on the original model we see that using few particles is still better than the original model in this setting where we can obtain a good approximation.

For practical use, this is very beneficial as using few particles greatly reduces the computational cost. For this example, we were able to obtain a good approximation of the optimal sequence in terms of variance reduction in the estimates of the marginal likelihood. Considering for instance the original model, which is a twisted model based on a sequence of constant functions, we would not be able to use few particles since a sequence of constant functions is not a good approximation of the optimal sequence. Hence in settings where we do not have a good approximation, we also need a larger number of particles to get low variance estimates of the marginal likelihood. This is illustrated in the examples in Section 3.3. For both models, we used the same number of particles in the bootstrap filters for the twisted models. For the examples in Section 3.3, we were able to obtain a better approximation of the optimal sequence for the linear Gaussian model compared to the Poisson model. This resulted in a lower variance in the estimate of the marginal likelihood of the linear Gaussian model than for the Poisson model. This clearly illustrates that the quality of the approximation is key also for considering how many particles are required in the particle filter.

We note here that we had relatively few observations, and this is part of the reason we could get away with using few particles. For longer sequences of observations the bootstrap particle filter applied on the original model, the quality of the Monte Carlo estimates will deteriorate over time when few particles are used (Petris, Petrone and Campagnoli, 2009). Unless we are able to obtain the exact $\boldsymbol{\psi}^{\star}$ we will require more particles for the twisted model, especially for longer sequences where we resample, but this increase would still be less than if we were to use the original model.

Misspecified models and sequences

If we want to use the twisted models for parameter estimation within the PMMH setting, which we discussed in Section 2.5, the task which took the longest time and is arguably the limiting factor is estimating the marginal likelihood for a set of parameters. We have discussed how even crude approximations to the optimal sequence will provide lower variance estimates to the marginal likelihood than the bootstrap particle filter applied on the original model, and now we will have a closer look at how we can reduce the computational effort in a PMMH setup.

The motivation here is that when using the twisted models in any Metropolis-Hastings setup we would need to approximate the optimal sequence for each new sample of the parameters. This will be tedious, and with the sampled parameters presumably being quite similar it is a reasonable hypothesis that the optimal sequences obtained for each sample also will be quite similar. Hence it would save a significant amount of time if rather than creating a new approximation $\hat{\boldsymbol{\psi}}$ for each sample, we could instead create a single approximation and use this for all the samples. We will have a closer look at PMMH and twisted models in Section 4.5, but first, we will consider this potential simplification and determine if it is viable.

The basis for this simplification is that within a region of the parameter space the optimal sequence is almost identical for all the parameters. This may not always be true, but it merits consideration as it may potentially save a large amount time and of computational effort. In relation to this, a factor to consider here are the observations, and whether they are informative in the sense that they provide information about the underlying state. With an informative observation, the optimal sequences should be similar since the observations will provide more information to the optimal sequence than the other parameters in the state transition of the model.

To test this we will again look at a linear Gaussian model as it is easy to work with, and we will consider two different configurations that vary in how informative the observations are. We do this by changing the variance of the emission density, and here we will use the following model

$$\begin{aligned}
&\mu(x_1) = \mathcal{N}(x_1; a, 1) \\
&f(x_{t-1}, x_t) = \mathcal{N}(x_t; a + bx_{t-1}, 1), \text{ for } t \in 2, \dots, T \\
&g(x_t, y_t) = \mathcal{N}(y_t; x_t, \sigma^2)).
\end{aligned}$$

We set $a = 0.25$ and $b = 0.7$. We will also consider both $\sigma^2 = 0.1$ and $\sigma^2 = 1$ which determines how informative the observations are, and we will then simulate a sequence of 25 observations and create approximations of the optimal sequences based on these models. In each of the settings, we will vary the parameters in the linear transitions of f and μ and compute the marginal likelihood using a twisted model with an approximation of the optimal sequence created based on the true parameters. That is we will estimate the marginal likelihood for different combinations of the parameters a and b using a twisted model based on an approximation of the optimal sequence for the parameters $a = 0.25$ and $b = 0.7$. Since these all are linear Gaussian models, we can compare our estimates with the Kalman approach, and to create the approximation of the optimal sequence we will take the same approach as in the previous examples.

Starting with $\sigma^2 = 1$ we have an approximation of the optimal sequence in $\hat{\boldsymbol{\psi}}_{\mathbf{1}}$, which we obtain in the same manner as $\hat{\boldsymbol{\psi}}_{\text{org}}$. We already know from our previous work that this will provide low variance estimates of the marginal likelihood. Now we will change the model slightly and set $\mu(x_1) = \mathcal{N}(x_1; a, 1)$ and $f(x_{t-1}, x_t) = \mathcal{N}(x_t; a+bx_{t-1}, 1)$, and now we want to estimate the likelihood of the model with different values for the parameters a and b but still use the observations that we already have obtained. We estimate the marginal likelihood using the twisted model defined with the sequence $\hat{\boldsymbol{\psi}}_{\mathbf{1}}$. Although this is not the optimal sequence for this particular model our hypothesis is that this sequence will be similar to the optimal sequence for different sets of parameters and will provide low variance estimates of the marginal likelihood. We do this for several combinations of parameters the results can be seen in table 4.6. Since this is from a linear Gaussian model we can use the Kalman approach to assess the estimates obtained, so again we look at the ratio Z^N/Z where Z^N are the estimates from the twisted models each here using $N = 50$ particles. For a set of parameters, we create 100 estimates and report the results.

We repeat the process for $\sigma^2 = 0.1^2$, and create a sequence $\hat{\boldsymbol{\psi}}_{0.1}$ to see if there is an effect in how informative the observations are. The entire setup is the same for both methods, but we change the variance of the emission density.

Although the variance likelihood estimates are larger when we are not using an approximation of the optimal sequence for a set of parameters directly, we still obtain low variance estimates for parameter values close to those used in the sequence. This indicates that the optimal sequence does not change much when altering the parameters in the transitions, and we can obtain good estimates of the marginal likelihood, especially when considering the number of particles used. This also demonstrates that a crude approximation can be used to obtain low variance estimates of the marginal likelihood.

The results here also indicate how informative the observations are is having an effect when altering the parameters as we have done here. For $\sigma^2 = 0.1^2$, most informative of the two settings considered, we observe less change in

Estimates of Z^N/Z with $\sigma^2 = 1$			
Parameter values	Average value	Standard deviation	Average log likelihood
$a = 0.25$ and $b = 0.7$	0.999	$5.50 \cdot 10^{-8}$	-54.919
$a = 0.2$ and $b = 0.6$	1.003	0.067	-55.790
$a = 0.1$ and $b = 0.8$	0.997	0.072	-54.967
$a = 0$ and $b = 0.4$	0.994	0.204	-59.727
Estimates of Z^N/Z with $\sigma^2 = 0.1^2$			
Parameter values	Average value	Standard deviation	Average log likelihood
$a = 0.25$ and $b = 0.7$	0.999	$1.58 \cdot 10^{-9}$	-40.900
$a = 0.2$ and $b = 0.6$	1.001	0.011	-41.283
$a = 0.1$ and $b = 0.8$	0.999	0.012	-41.485
$a = 0$ and $b = 0.4$	0.997	0.028	-45.364

Table 4.6: Transitions are on the form $\mu(x_1) = \mathcal{N}(x_1; a, 1)$ and $f(x_{t-1}, x_t) = \mathcal{N}(x_t; a + bx_{t-1}, 1)$. The observations are simulated from a model with $a = 0.25$ and $b = 0.7$. For each set of parameters we estimate the marginal likelihood 100 times.

the variance as we move away from the parameter which gave the optimal sequence compared to the less informative $\sigma^2 = 1^2$. This is as expected since $\psi_t^\star(x_t) \propto g(x_t, y_t)$ and by making the observations informative we also make $g(x_t, y_t)$ narrower.

When considering a PMMH setup we need to compute the likelihood for each new sample. Since the optimal sequence presumably is similar for several parameters, we can use the same sequence several times rather than approximating a new sequence for every sample which will save time Metropolis-Hastings setup. We also note that we do get some variance in the estimates when moving away from the parameters which gave the sequence, and one way of dealing with this would be to divide the parameter space into several regions and for each use a sequence $\hat{\psi}$ which approximates the optimal sequence in that region of the parameter space. Importantly the results here show that we do not need to create an approximation of the optimal sequence for each new set of parameters in a Metropolis-Hasting setup which will greatly reduce the computations required.

In this example, we only considered how variations of the parameters in the transition model affected the approximation of the optimal sequence. However, there may also be changes in the emission model $g(x_t, y_t)$, in particular for parameter estimation settings. While we have not looked at this setting here, it is sometimes possible to reparametrize the model such that the parameters of interest are located in the transition model. Reparametrizing the model in such a manner can allow us to consider how variations of the parameters in the emission model affects the approximation of the optimal sequence through the transition model.

4.5 Particle marginal Metropolis-Hastings and twisted models

Using twisted models

Having shown that the twisted models can be used to obtain low variance estimates of the marginal likelihood, we want to illustrate the benefit of the twisted model for an application. In Section 2.5 we looked at how the marginal likelihood was key for parameter estimations methods such as particle marginal Metropolis-Hastings (Andrieu, Doucet and Holenstein, 2010). When considering this method, low variance estimates of the marginal likelihood which we can obtain using the twisted models, are important for the Metropolis-Hastings ratio (2.20) and the accuracy of the algorithm. Using the twisted model to provide the estimates of the marginal likelihood is here an alternative particle approximation to the bootstrap particle filter applied on the original model, and for a good approximation of $\boldsymbol{\psi}^\star$, using the twisted model can be preferable as we can get lower variance with fewer particles.

Particle marginal Metropolis-Hastings has a relatively high computational cost since to provide a single sample from $p(\theta|y_{1:T})$ we need to compute the likelihood based on the parameters θ. When using the twisted models, we still have to create an approximation for each new set of parameters we sample which increases the computational cost. We can get around this in part by exploiting similarities in the optimal sequences $\boldsymbol{\psi}^\star$ that belong to different sets of parameters. All the proposed samples in a Metropolis-Hasting setup will be relatively similar and in Section 4.4 we saw that small changes in the parameters had a relatively small effect on the corresponding sequence $\boldsymbol{\psi}^\star$ and the variance of the likelihood estimates. This experiment was motivated by its potential use for a Metropolis-Hasting setup where the idea is that we can use the same sequence $\boldsymbol{\psi}$ for several parameters. Instead of creating an approximation for each proposed sample, we can create some approximations of the optimal sequence in different regions of the parameter space before running the Metropolis-Hasting setup and for each proposed sample choose an appropriate approximation depending on where in the parameter space the proposed sample is located.

Example of parameter estimation

This can best be illustrated with an example and continue with the example discussed in Section 2.5. We will use the same simulated set of observations such that the methods are directly comparable. Recall that the model used is

$$\begin{aligned}
\mu(x_1) &= \mathcal{N}(x_1; a, 1) \\
f(x_{t-1}, x_t) &= \mathcal{N}(x_t; a + bx_{t-1}, 1), \text{ for } t \in 2, \dots, T \\
g(x_t, y_t) &= \mathcal{N}(y_t; x_t, 0.5^2))
\end{aligned}$$

where we set $a = 0$ and $b = 0.7$. Since this is a linear Gaussian model we can obtain close to the exact sequence $\boldsymbol{\psi}^\star$ with a simple approximation, but the result would be quite similar to using a Kalman approach directly. The primary difference is that using the twisted model will be quite a bit slower than a Kalman approach which arguably makes it superfluous to look at what happens when we compute an approximation for each new set of parameters in this setting. Instead, we will take the approach outlined above and create an approximation of the optimal sequence in the true parameter values $a = 0$ and $b = 0.7$, and use a twisted model based on this approximation when computing every likelihood. I.e. for each proposed set of parameters, we will compute the likelihood with a twisted model based on the same sequence $\boldsymbol{\psi}$. This sequence, $\boldsymbol{\psi}$, used in this twisted model will be an approximation of the optimal sequence for the true parameter values and will be used for every set of parameters we propose. To obtain this approximation we follow the same procedure as for $\hat{\boldsymbol{\psi}}_{\text{org}}$, and use a single Gaussian density as the approximation for each of the functions. Note that we also could divide the parameter space into several regions and for each region create an approximation of the optimal sequence.

For the remainder of the Metropolis-Hastings algorithm, we will use the same setup as in Section 2.5. That is, we use a Gaussian random walk proposal centered at the previous sample with variance $\sigma^2 = 0.125^2$. Additionally, we use uniform priors for both parameters that are located in the intervals $(-0.5, 0.5)$ and $(0, 1.5)$ for a and b respectively. Finally, we create 1000 pairs of (a, b) by alternating between sampling from a and b with the first 100 samples as a burn-in, similar to what we did previously. For estimating the marginal likelihood, we use the twisted model defined above but this time with $N = 50$ particles.

A visualization of the distribution can be seen in Figure 4.7, and from the histogram, we see that the samples of both parameters are centered around the true value. Using the mean of the samples as the parameter estimates we get $\hat{a} = 0.03$ and $\hat{b} = 0.71$ which is reasonable given the true parameters. Additionally, the samples obtained here are quite similar to what we obtained when using the Kalman approach to compute the marginal likelihood, as seen when comparing with Figure 2.11. This is expected since we have a good approximation of $\boldsymbol{\psi}^\star$ and the likelihood estimates obtained will be close to the analytical values obtained from the Kalman approach.

This example shows that using the twisted models is a viable option for this Metropolis-Hastings setup for state space models. Importantly using the twisted model was faster than using the original model since we were able to use fewer particles in the bootstrap filter, and the variance of the likelihood estimates will also have been lower since we used the twisted model. This makes the twisted models preferable especially when looking at the distribution of the samples we obtain which is similar to the one obtained using the Kalman approach in Section 2.5. We also see that for the example considered here there was no need to create an approximation for every set of parameters we sampled. This is worth considering for other models as well since this greatly reduces the computational cost of the algorithm. The effect of this will vary based on the model, especially when it comes to how informative each observation is. The effect may also depend on the number of parameters and the size of the parameter space we are working with. When each of the optimal functions $\psi_t^\star(x_t)$ depends on a larger number of parameters, the effect on the optimal function of altering all the parameters is likely to be greater than altering two. This is also a model specific, aspect and the number of parameters we aim to sample will not affect all models equally. Thus in this setting, we obtained good results with a single approximation located at the center of the distribution, and other settings may require several approximations located in different regions

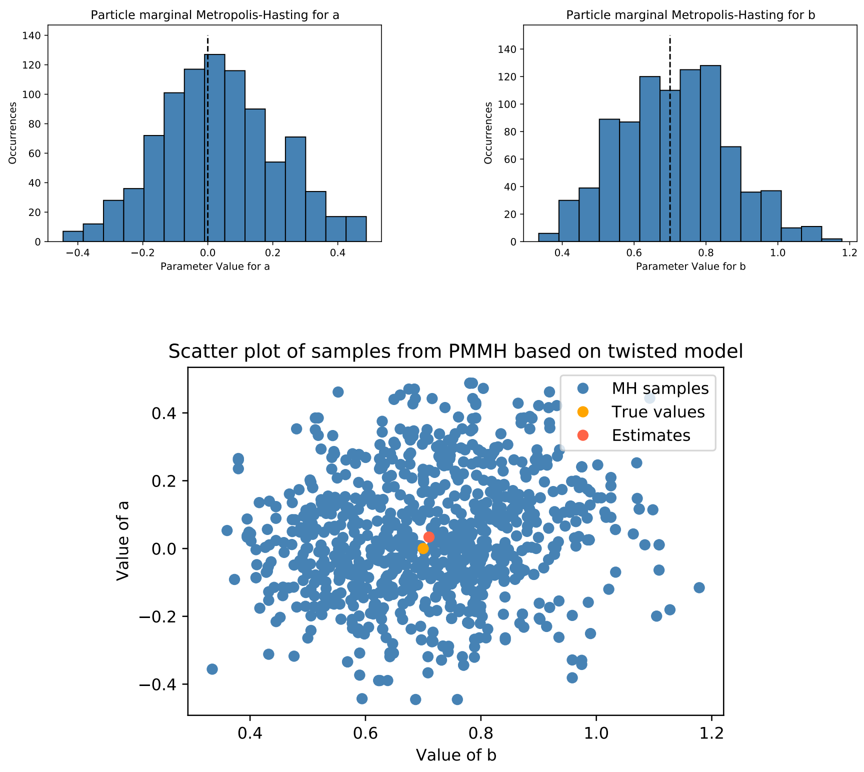

Figure 4.7: Visualizing the distribution of the samples obtained from PMMH with a twisted model for estimating the marginal likelihood.

of the parameter space.

4.6 Smoothing with twisted models

Every topic we have discussed so far regarding the twisted models has been related to estimating the marginal likelihood, but the work we have done can also be used to obtain estimates of the states. Filtering estimates of the states are not easily obtainable with the twisted models, but we can use the twisted models to obtain smoothing estimates of the states which will be the topic of discussion for this section.

When we considered parameter estimation we used the same Metropolis-Hastings setup in both Section 2.5 and Section 4.5. The difference between the two lied in how we estimated the marginal likelihood while the Metropolis-Hastings setup remained unchanged. In contrast, the smoothing approaches here are inherently different from the methods discussed in Section 2.4 while the end goal remains the same. First, we will briefly look at smoothing by transforming the emission model and see how the approach we have taken for estimating the marginal likelihood can be used for obtaining smoothing estimates. Second, we will look at how the sequence $\psi^\star$ can be used to provide a sample from the smoothing density directly from the particle filter without requiring any backward pass or fixed lag approximation.

Smoothing by modification of potentials

Smoothing can be seen as an integration problem where we are interested in obtaining Monte Carlo estimates of different expected values as these can be expressed as integrals where the integrand is proportional to the smoothing density itself. Recall from Chapter 3 that the smoothing density is the same for the original and any twisted model, and given an appropriate function φ, we have

$$
\begin{aligned}
\mathrm{E}[\varphi(x_{1:T})|y_{1:T}] &= \int_X \varphi(x_{1:T}) p(x_{1:T}|y_{1:T}) \,\mathrm{d}x_{1:T} \\
&= \frac{1}{Z} \int_X \varphi(x_{1:T}) \mu(x_1) g_1(x_1, y_1) \prod_{t=2}^{T} f(x_{t-1}, x_t) g(x_t, y_t) \,\mathrm{d}x_{1:T} \\
&= \frac{1}{Z} \int_X \varphi(x_{1:T}) \mu^{\psi}(x_1) g_1^{\psi}(x_1, y_1) \prod_{t=2}^{T} f_t^{\psi}(x_{t-1}, x_t) g_t^{\psi}(x_t, y_t) \,\mathrm{d}x_{1:T}.
\end{aligned}
$$

That is, the expected values related to the smoothing densities will be the same for both models, and $\varphi(x_{1:T}) = 1$ is the special case of the marginal likelihood. One possible method for obtaining the smoothing estimate is to simply solve the above integral with a particle filter. This is briefly mentioned in Guarniero, Johansen and Lee (2017), and while the sequence $\boldsymbol{\psi}^{\star}$ is not optimal for this particular integral, a similar approach with the twisted models can be taken to provide an estimate of the integral. We will also propose a different setup. Note that with the simple factorization $\varphi(x_{1:T}) = \prod_{i=1}^{T} \varphi_i(x_i)$ we can introduce the potentials $\gamma_t(x_t) = g(x_t, y_t)\varphi_t(x_t)$ and the integral can be written as follows

$$
\Gamma = \int_X \mu(x_1)\gamma_1(x_1) \prod_{t=2}^{T} f(x_{t-1}, x_t)\gamma_t(x_t) \,\mathrm{d}x_{1:T}. \tag{4.3}
$$

Integrals on the form (4.3) are what we exclusively dealt with in Chapter 3, and we can easily obtain approximations of integrals on this form using particle filters. By creating a twisted model based on the modified potentials γ_t for $t \in 1, \dots, T$ we can use the same idea as we used for approximating the marginal likelihood Z when approximating the integral Γ. We can again determine an optimal sequence $\boldsymbol{\psi}'$ in the same manner as (3.8), but rather than this sequence being optimal when estimating Z it will be optimal when estimating Γ.

This results in a different method for approximating smoothing expectations, but the choice of $\varphi_t(x_t)$ is somewhat limited since $\gamma_t(x_t)$ must be a valid potential. For the potential to be valid we require that each γ_t is bounded, non-negative and continuous. This somewhat limits the usefulness of this approach since e.g. $\varphi(x_i) = x_i$ is not available due to the resulting potential being negative in certain regions.

For functions φ which gives rise to valid potentials, this method can be useful as we now can obtain low or in theory zero variance estimates of the different expected values. This follows from the same reasoning which led to the low variance estimates of the marginal likelihood in Chapter 3. This is in stark contrast again to the methods of smoothing considered previously in two ways.

First, the previously discussed smoothing estimates were only Monte Carlo estimates while using the twisted models here we have equality with the optimal sequence $\boldsymbol{\psi}'$. Secondly, this approach only allows us to provide estimates of expected values, though with low variance. In contrast the other methods we considered previously provided samples from the smoothing density. This can become quite noticeable if we want to estimate several quantities related to the smoothing density, as a sample can easily be used to estimate several different quantities. This somewhat limits the applicability of this method, and together with the fact that not all functions φ are usable, the scope for this smoothing method is limited to special cases.

Smoothing with $\psi^\star$

What we often want when considering the smoothing problem is a sample from the smoothing density itself. While this can be done directly with the bootstrap filter on the original model, this is rarely done in practice due to issues with degeneracy and few unique particles from resampling which we discussed in Section 2.2 and Section 2.4. To get around this issue we can make use of the twisted model which we have shown earlier in this chapter resamples less frequently when constructed appropriately. This can be used to sample particle trajectories $x_1, \dots, x_T$ from the smoothing density which does not suffer from limitations related to degeneracy.

To see how this works, it will be beneficial to first look at what happens when we use the optimal sequence $\boldsymbol{\psi}^\star$ from which we obtain the zero variance estimates of the marginal likelihood and all the weights are equal. Recall that the sequence $\boldsymbol{\psi}^\star$ as defined in (3.8) is equal to the backward information filter which motivates why this approach works. At time $t = 1$, the twisted model samples from a density proportional to $p(x_1|y_{1:T})$, and since all the importance weights are identical we are able to obtain a sample from the exact smoothing density at this time. For the remaining times $t = 2, \dots, T$ it can be shown by simple manipulations that the twisted model samples from a density proportional to $p(x_t|x_{t-1})p(y_{t:T}|x_t)$ and $p(x_t|x_{t-1}, y_{t:T})$. As a result, we can obtain the full trajectory by combining the samples at each time t, and this will be a sample from the exact smoothing density since all the importance weights will be equal.

In practice we only obtain an approximation $\hat{\boldsymbol{\psi}} \approx \boldsymbol{\psi}^\star$. When considering an arbitrary approximation that results in a valid twisted model, the main result will remain unchanged and we can still obtain a sample from the smoothing density. This is due to the construction of the twisted model, and it is the same reason as to why the marginal likelihood and joint density is the same in any twisted model and the original model which we discussed in Chapter 3. As the weights are no longer equal, when using an approximation, the sample obtained is a weighted sample representing trajectories from the smoothing density. This weighted sample can then be used to obtain Monte Carlo estimates of quantities related to the smoothing density. The quality of these estimates will depend on the number of particles used in the bootstrap filter with the twisted model and the effective number of samples N_{eff} as given by (2.13). When we have a good approximation of the optimal sequence the effective number of samples will be close to the number of particles used. Thus to quantify the quality of the approximation for the purposes of smoothing it is natural to consider the effective number of samples.

The use of the twisted models for likelihood estimation and smoothing based on $\psi^\star$ are inherently different. When considering likelihood estimation, we aim to obtain low variance estimates of the marginal likelihood which is possible with the optimal sequence. When using $\psi^\star$ for smoothing we do not obtain the same low variance estimates, and instead, we use the optimal sequence to obtain samples from the smoothing density which does not suffer from degeneracy. These samples can be used to provide consistent estimates of quantities of interest, and the twisted models give a different method for sampling from the smoothing density.

Example - Smoothing with twisted models

Here we will illustrate how an approximation of the optimal sequence $\psi^\star$ can be used to sample directly from the smoothing density. For this example, we will consider a linear Gaussian model such that we both have the analytical smoothing density which can be obtained with the Kalman approach as we did in Section 2.4 and we are able to obtain a good approximation of the optimal sequence with the setup we considered in Chapter 3. We will consider the following model

$$\begin{aligned}
&\mu(x_1) = \mathcal{N}(x_1; 0.25, 1)\\
&f(x_{t-1}, x_t) = \mathcal{N}(x_t; 0.25 + 0.7x_{t-1}, 1), \text{ for } t \in 2, \dots, T\\
&g(x_t, y_t) = \mathcal{N}(y_t; x_t, 1).
\end{aligned}$$

Here we simulate a sequence of 25 latent variables and observations, and we will look at three different approaches for smoothing. First, we have the Kalman approach which we know will provide analytical results which here serves as a comparison. Additionally, we will use a twisted model with a good approximation of $\psi^\star$ such that we can run the bootstrap particle filter without resampling to sample from the smoothing density. To obtain the sequence which defines this twisted model we will take the same approach as we did in the first examples in this chapter and use a single Gaussian density as the approximation. I.e. we follow the same procedure as for $\hat{\psi}_{\text{org}}$. This will provide sample trajectories that we can use to obtain Monte Carlo estimates that will not have issues with degeneracy. Finally, we want to use a twisted model with a poor approximation of the optimal sequence. To obtain a poor approximation we start with the sequence of functions which are the good approximations, where each is parameterized by an expected value and a variance, and we simply multiply the variance of all the functions in the sequence by 0.5 thus obtaining a poor approximation. For both the particle approximations we will use an implementation of the bootstrap particle filter with multinomial resampling which allows us to easily resample the entire trajectory, and we resample when $N_{\text{eff}} < 0.5N$ where N is the number of particles used.

For comparing the methods, we run the Kalman smoother once to obtain the analytical values for the expected value and variance of the smoothing density which can be used as a comparison. For the particle approximations, we use 500 particles for the good approximation and 1000 particles for the poor approximation. For comparing the methods we primarily look at the expected

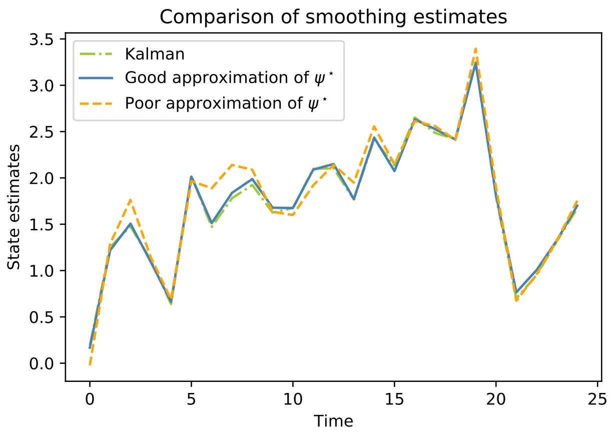

Figure 4.8: The expected value of the smoothing density computed analytically with the Kalman smoother and compared with particle estimates.

value of the smoothing density since this is a common estimate of the state, and was our focus earlier in Section 2.4. Figure 4.8 shows Monte Carlo estimates of the expected value of the smoothing density from the particle approximations alongside the analytical value from the Kalman approach. We see here that using a good approximation of the optimal sequence requires fewer particles to obtain a good estimate of the expected value of the smoothing density. This is expected since when the twisted model is based on the poor approximation it resamples and we are left with fewer unique particles especially at the early times where we see the largest discrepancy in the estimates from the twisted models. At the later times, both particle approximations are consistent with the analytical values. This is also expected since these will not have been resampled as frequently as those in the early times. We repeated the process of running the particle filter on the twisted model based on the poor approximation with $N = 1000$ particles 100 times. For each, we record the number of unique values for the state x_1 which occurs in the 1000 trajectories at time T after resampling. Over the 100 repetitions, the average is 31.5 unique values of the state with a standard deviation of 5.3. This gives an indication of the diversity in the sample at the early times, and clearly shows that degeneracy is a problem with this twisted model, but is still less so than for the original model. This also illustrates the issue with using a poor approximation when sampling from the smoothing density, and why a good approximation is required for smoothing in this setting. One possible method for dealing with this is to increase the number of particles used such that there is a sufficient number of particles at the early times even when accounting for degeneracy to obtain a good Monte Carlo estimate. This comes with a greater computational cost and it may be simpler to focus on obtaining a better approximation.

Having few unique particles will also affect other estimates, and for quantities such as the second moment, we expect to see a larger difference. This is caused by resampling discarding values from the tails of the density which can lead to inaccurate estimates of the variance. In this setting, we can compute the variance of the smoothing density of x_t conditional on all the observations analytically,

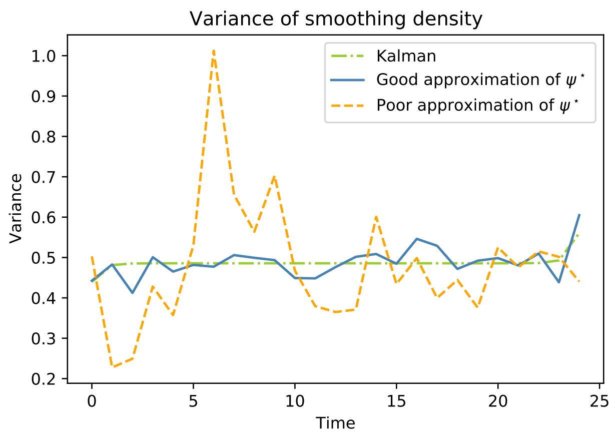

Figure 4.9: Variance of the samples $\mathrm{E}[X_t^2] - \mathrm{E}[X_t]^2$, compared to the analytical variance of the smoothing density computed with the Kalman smoother.

and these analytical results can be compared to the particle estimates of the variance.

Figure 4.9 shows a comparison of estimates of the variance of the smoothing density based on the two particle approximations and the Kalman smoother, and the particles here are the same particles used in Figure 4.8. Here we see that there is a much larger difference between the particle approximation and the analytical values when compared to the expected value. When using a good approximation of $\psi^\star$ to define the twisted model we are able to obtain variance estimates which are close to the analytical values obtained from the Kalman smoother. In comparison, the variance estimates from the twisted model based on a poor approximation of the optimal sequence are much less stable. Again, this may in part be caused by the fact that we resample in this model which explains why the approximation is better at the end of the sequence. When we resample, while removing the trajectories with low weight, we also are left with fewer unique trajectories at the early times which will influence the Monte Carlo estimates.

Having spent an undue amount of time when it comes to the results based on the poor approximation, we now move our focus to the good approximation. We already knew from Section 3.3 that we could obtain a good approximation here, and as we mentioned previously with a good approximation we are close to sampling from the exact smoothing density. When comparing the Monte Carlo estimates of the expected value and variance based on the good approximation of the optimal sequence with the analytical results from the Kalman smoother, these estimates are consistent with the analytical results. This is expected since we know we have a good approximation of the optimal sequence and we are close to sampling from the exact smoothing density at each time. In this case, the approximation of the optimal sequence is accurate enough such that all the trajectories have the same weight at time T in the bootstrap filter, and thus we are essentially sampling from the exact smoothing density. This comes as a result of the optimal functions being simple to approximate in this setting, and for other more complex models getting the same quality in the smoothing estimates would be difficult.

This example illustrates that, with a good approximation, using the twisted models in this manner is a viable approach for obtaining a sample from the smoothing density. Looking at the estimates of the expected value, the estimates here are much closer to the analytical value than those we considered in Section 2.4. However, this example is idealized in the sense that we are able to obtain a good approximation of the optimal sequence as well as we have analytical results available to verify our results. Unless we have a good approximation there will still be issues with degeneracy in the samples from the smoothing density, especially when estimating the variance, as seen with the poor approximation. This is a flaw with using the twisted models for smoothing since depending on the model there is a limit to how good of an approximation we can obtain within a class Ψ. If we were to consider the Poisson model from Section 4.3 we would need to improve the approximation further in order to not resample. That being said, when comparing to the smoothing setups we considered previously, this approach is preferable in this setting where we can obtain a good approximation of the optimal sequence.

CHAPTER 5

Discussion and final remarks

5.1 More on the use of twisted models

Further notes on results

The motivation for introducing the twisted models is first and foremost for obtaining estimates of the marginal likelihood with low variance without having to increase the number of particles in the bootstrap particle filter. In Chapter 3 we showed how defining a twisted model based on the optimal sequence $\boldsymbol{\psi}^{\star}$ would provide zero variance estimates of the marginal likelihood. While this exact sequence will be unobtainable in practice, we have demonstrated that with a sufficiently good approximation of the optimal sequence we can still obtain an estimate with low variance by running the bootstrap particle filter on the twisted model.

When considering the twisted models the variance of the marginal likelihood estimate is related to the sequence used to define the twisted model and the number of particles in the particles filter. To reduce the variance of the marginal likelihood estimate when using the twisted models, we can either increase the number of particles in the bootstrap filter or obtain a better approximation of the optimal sequence. While in theory both the number of particles and the quality of the approximation can be increased or improved indefinitely to reduce the variance of the marginal likelihood estimate, there will eventually be a point where this is no longer feasible due to practical limitations related to the computational cost or potential issues with overfitting.

We can divide the strategy for obtaining a good approximation into two parts. Either we can take the same approach as Guarniero, Johansen and Lee (2017) and find the best approximation within a relatively simple class Ψ. Alternatively, we can expand the class Ψ which allows for greater flexibility in the approximation and admits good approximations of a larger number of functions. We expanded the class by going from a single Gaussian density to a Gaussian mixture as the class used for the approximation, which led to increased complexity when it comes to creating the approximations as we have to determine the additional parameters for the mixture. In general, using a flexible approximation makes the task of obtaining a good approximation more complex since there are more options to consider, which is why making an overly flexible approximation is impractical. Furthermore, with the approach we have taken for obtaining an approximation, there is also a risk of overfitting such that approximation only is good for the gridpoints used to create the approximation.

For obtaining the best approximation of the sequence $\boldsymbol{\psi}^\star$ with functions within a class Ψ, Guarniero, Johansen and Lee (2017) introduced an iterative approach called the iterated auxiliary particle filter. For the majority of our examples here we discarded the iterative approach in favor of creating the approximation in a single step. This worked well in the cases we considered and is simple to implement, showing that the iterative setup is not always necessary. In Section 4.3 we started by using a simple class Ψ. However, since the functions $\psi_t^\star(x_t)$ were not contained in this simple class for the model we considered there was potential to reduce the variance further. In general, for most practical classes, the optimal functions will never be included in the class Ψ, thus we want a class that admits a good approximation of the optimal sequence. To further improve the approximation, we considered a larger and more flexible class, which as expected provided an even lower variance estimate of the marginal likelihood. This indicates that selecting a suitable class Ψ is of importance since the quality of the marginal likelihood estimate is connected to the quality of the approximation we can obtain within the class Ψ.

A related question to consider is how good of an approximation is necessary for practical use since we observe that even with a crude approximation of the twisted model, we can provide lower variance estimates of the marginal likelihood than with the original model directly. The answer to this question is problem specific and depends on how much variance is acceptable in the estimates. For applications such as Metropolis-Hastings parameter estimation, which we discussed in Section 4.5, we desire low variance estimates of the marginal likelihood that can be computed quickly.

We have primarily discussed how to estimate the marginal likelihood, but we also spent some time on state estimation. We looked at obtaining smoothing estimates in Section 4.6 since smoothing can be done with the twisted models as well. We considered two setups for smoothing, and the one we spent the least time on was considering smoothing as an integration problem. For this approach, we considered how it is possible to express quantities of interest related to the smoothing density as integrals on the same form as the marginal likelihood. Hence, we can obtain low variance estimates of these quantities by using the same setup with the twisted models as we did for the marginal likelihood. The second method we considered allowed us to use the optimal sequence $\boldsymbol{\psi}^\star$ to obtain a sample from the smoothing density. This is inherently different from the previous approach as we are not aiming for the low variance estimates we obtain with the twisted models, and instead, we sample from the smoothing density directly. This is beneficial as it gives a non-parametric estimate of the smoothing density similar to the smoothing setups considered in Section 2.4 which can be used to obtain Monte Carlo estimates of quantities of interest related to the smoothing density. While we are not able to obtain the same effect when it comes to variance reduction as we did when introducing the twisted model for estimating the marginal likelihood, the estimates obtained will still be consistent and the variance can be reduced by increasing the number of samples. For smoothing, we also demonstrated in Section 4.6 that using a good approximation of the optimal sequence is of importance. When using the exact sequence $\boldsymbol{\psi}^\star$ the twisted model allows us to sample from the exact smoothing density of the original model. That is, the sample obtained is not weighted, which is the case when using an approximation of the optimal sequence. Using a good approximation of the optimal sequence limits the negative effects of weight degeneracy on this sample.

Working with twisted models

To be able to obtain any estimates from the twisted model we need to be able to efficiently sample from (3.1) and (3.2), that is and $\mu^{\psi}(x_1)$ and $f_t^{\psi}(x_{t-1}, x_t)$, to obtain the particles when using the bootstrap particle filter. This a restriction that is present for all twisted models, and the requirement that we must be able to efficiently sample from the process which drives the latent variables limits the applicability of the twisted model itself. In particular, we cannot use any sequence of functions to define the twisted model. We deal with this by restricting the functions $\psi_t(x_t)$ used to define the twisted model to a specific class Ψ such that we can guarantee that sampling is possible. This also simplifies calculations in that we can obtain general expressions for the weight updates, and the integrals $\widetilde{\psi}_t(x_t)$ will all be on the same form making them much easier to compute. In practice, we chose the class Ψ such that the resulting approximations were conjugate to the transitions in the model, thus as long as we can sample from the original model we are also able to sample from the twisted model.

Restricting the class in this manner puts some limitations on the twisted models since every twisted model cannot be used in practice. The effect of this limitation can be largely diminished by making the class Ψ flexible such that it includes a large number of functions directly, but also making sure that within this class there will be good approximations to a large number of functions that are not included in the class. Using a flexible class Ψ will also be more practical than not restricting the functions and create a, potentially different, sampling routine for each of the transitions.

Another limitation lies in running the bootstrap particle filter on the twisted model. When running the bootstrap particle filter on the twisted model following algorithm 3 we compute the weights with (3.3) and (3.4). In theory for appropriate functions such that $\psi_t(x_t) > 0$ this will not be a problem. However, in practice for very small values of $\psi_t(x_t)$, there may be numerical issues when we divide by number close to zero. When twisting the model in our simple examples we did not encounter this problem ourselves, however, they are still present and may cause issues for other models. One way this can be dealt with is by introducing a constant term in the functions twisting the models such as C^t from (3.15). This essentially guarantees that we never divide by a number smaller than C^t, but this comes at the cost of making the transitions in the twisted model further from the optimal as these now include the product of this constant and the transition of the original model.

Other limitations come with approximating the optimal sequence $\boldsymbol{\psi}^{\star}$ in practice. A topic that merits discussion is how the setup we have used is influenced by the number of observations. First and foremost with a large number of observations we will have to perform a large number of function approximations since for every observation y_t there is a function $\psi_t(x_t)$ in the sequence which defines the twisted model we have to include. For likelihood estimation we have to create approximations for each $\psi_t^{\star}(x_t)$ which can be computationally expensive for large T. While the additional computational effort resulting from the additional approximations, it can be made up by the twisted models requiring fewer particles, however, we still have to create an approximation to each of the functions.

An additional note with the twisted model when considering the optimal sequence $\boldsymbol{\psi}^\star$, is that it becomes an offline method. If we want to define the optimal sequence with the inclusion of a new observation, we need to define the whole sequence all over again since it is defined recursively starting at time T and all the functions in the sequence will depend on this new observation. We did not see an effect of this when considering parameter estimation in Section 4.5 since the Metropolis-Hastings setup we used also is an offline method. Neither did we observe an effect of this for smoothing in Section 4.6 as smoothing is also done offline. However, this makes the twisted models unsuited for online estimation in settings where new observations are obtained frequently or at regular intervals, at least when using approximations of the optimal sequence $\boldsymbol{\psi}^\star$. Of interest in relation to this is if the twisted models can be used in a manner more suited for online estimation. Transforming this setup with the twisted models into a fully online approach is difficult. This is because the weights at time t depends on $\widetilde{\psi}_t(x_t)$ which is defined based on the the function which defines the twisted model at time $t+1$ in (3.3) and (3.4). One possibility here is to take the same approach as the fixed lag smoothing approximation and assume that each $\psi_t^\star(x_t)$ only depends on a few observations and not all such that when obtaining a new observation we do not need to redefine the entire sequence. Hence when a new observation is obtained, only a few of the functions in the sequence which define the twisted model must be redefined and is somewhat similar to lookahead strategies such as Lin, Chen and J. S. Liu (2013).

What we have done

We started this thesis by considering state space models. In Chapter 1 we gave three main targets for inference in state space models, and these are filtering, smoothing and parameter estimation. One of the main tools for inference in this class of models is sequential Monte Carlo in the form of particle filters which we discussed in Chapter 2. The benefit of using a Monte Carlo approach is that the estimates are consistent, and the variance of the estimates will diminish as more Monte Carlo samples are obtained. This allows us to obtain estimates of a variety of quantities of interest in relation to state space models which are used to model different systems.

In Chapter 3 we introduced twisted models to obtain estimates of the marginal likelihood with low variance, which is useful for tasks such as parameter estimation. Taking a step back, in general, there are two avenues that can be followed to provide low variance estimates of the marginal likelihood or other quantities, when considering sequential Monte Carlo and not including twisted models. These can either be followed separately or in conjunction, and the simplest is to increase the number of samples until the desired accuracy is obtained. Taking this approach for improving estimates is guaranteed to lower the variance of the estimates since these are Monte Carlo estimates and increasing the number of samples will lower the variance. The reason we sometimes avoid going to the end of this avenue is that creating more samples is computationally expensive and creating enough samples to sufficiently lower the variance may take too long. This is a prominent issue in higher dimensions where it often is necessary with a large number of samples to sufficiently explore the space we sample from. The alternative avenue is to change how we sample from the traditional bootstrap filter, and this naturally leads to the twisted models. By constructing a setup that makes better use of each sample allows us to obtain variance reduction in the Monte Carlo estimates this time without changing the number of samples.

When first introducing the particle filters in Chapter 2, we showed how the particle filters and sequential Monte Carlo could make use of the traditional importance sampling in order to sample from densities of interest when working with state space models. Methods such as the auxiliary particle filter (Pitt and Shephard, 1999) (Johansen and Doucet, 2008) and others change how we sample in the particle filter itself. Where the different variations of the particle filter change the importance density we sample from, the same effect can be achieved by choosing an appropriate sequence $\boldsymbol{\psi}$ for the twisted model. By changing the functions $\psi_t(x_t)$ we can sample from the same densities as from changing the proposal density in the particle filter, but the difference lies in how the samples are weighted. Using the general particle filter, described in algorithm 1, the samples are weight such that the filtering density is obtained. This is not the case for the twisted model where the weighting of the samples depends on the sequence $\boldsymbol{\psi}$. It is only at the final time T all twisted models estimate the same density as we saw in Chapter 3.

For using the twisted models to estimate the marginal likelihood, Guarniero, Johansen and Lee (2017) gave an optimal sequence $\boldsymbol{\psi}^\star$ which yields zero variance estimates of the marginal likelihood. As this exact sequence is unobtainable, in practice we create an approximation that is restricted to a suitable class Ψ. Within this class, there is an approximation which by some criteria is the best, and it is our task is to determine this approximation which we spent time on in Chapter 3 and Chapter 4. This is a function approximation problem that there are several ways to solve. We took a different approach than Guarniero, Johansen and Lee (2017) and Heng et al. (2020) which was simpler to implement than these iterative approaches. Our examples in Chapter 4 demonstrated that our approach can be used to obtain an approximation of the optimal sequence which gives low variance estimates of the marginal likelihood. For practical use, while zero variance estimates are desirable and can in theory be obtained with the twisted models, it will often be sufficient with low variance estimates. For the models we considered here when using our approach, we were able to obtain variance estimates that were significantly lower than those obtained from the untwisted model. This makes our approach a viable option in practice when the purpose of introducing the twisted models is to obtain an estimate of the marginal likelihood with lower variance than an estimate from the original model. While the iterative approaches can be used to provide lower variance estimates than the non-iterative approach we took, in the settings we considered in our examples the iterative setups can be redundant since the variance can be reduced further by increasing the number of particles.

The iterative setups provide approximations of $\boldsymbol{\psi}^\star$ from a class Ψ but there is no reason that we will not be able to obtain the same approximation in a single step. This was the motivation of our simple setup. We wanted to obtain the same approximation without having to introduce the iterative setups. We were able to obtain estimates of the marginal likelihood with low variance compared to the original model, proving that the iterative approaches can be discarded when the purpose is to simply give a lower variance estimate than the original model. However, the variance of the estimates resulting from the iterative approach was slightly lower compared to our method. Thus, within the class Ψ, our method does not necessarily find the best approximation in terms of variance reduction. However, there is still no reason the same approximation cannot be obtained in a single step. By changing either how the grid is laid out or the distance measure used to create the approximations, it could be possible to mimic the effect of the iterative approaches and obtain an even better approximation in a single step within the same class Ψ. Our proposed method for creating the approximations is relatively simple, and one possibility for improvement is to make better use of the observations we have available.

The improvement we observed by introducing the iterative setups was, however, mitigated by expanding the class Ψ because we obtained a better approximation in a single step. For the Poisson model that we considered, expanding the class provided a better approximation in a single step compared to introducing a single iterative step. Thus, while the iterative approaches can be used to further improve the approximation within a class Ψ, it can be better to expand the class and making it more flexible to accommodate a larger number of approximations. In practice, this can be a good alternative to increasing the number of particles, the iterative setups, or any other method for creating an approximation from a simple class. It can also be the case that there is some relation between the chosen class and the iterative procedure, which we did not look for here, such that an expanded class has more benefit from the iterative setups.

The end goal when introducing the twisted models is to obtain an estimate of the marginal likelihood with low variance or sample from the smoothing density. While our focus has mainly been on the former of these, both require a suitable sequence $\boldsymbol{\psi}$ to be beneficial compared to using the original model. We considered sequences that were sufficiently good approximations of the optimal sequence $\boldsymbol{\psi}^{\star}$, but what constitutes a sufficiently good approximation is problem specific. Given a model, that is a sufficiently good approximation for one problem, may not be sufficiently good for another. We have looked at how to create approximations of this sequence and factors which influence the approximations. There are several possible methods for creating an approximation of the optimal sequence and obtaining estimates of the marginal likelihood, such as the iterative approaches of Guarniero, Johansen and Lee (2017) and Heng et al. (2020), which are applicable in very general settings. We also proposed a simpler method that worked well in the settings we considered, however, its use is limited to simpler settings like the ones we considered here. Since this is a function approximation problem there are potentially several ways we can obtain the approximations which are model specific. As long as we obtain an approximation of the optimal sequence that we are able to work with there are no restrictions on how the approximations are obtained. Thus it matters little how we obtain the approximation, as long as it is sufficiently good and obtained in a reasonable time. This is a benefit of the method we proposed, as it is able to obtain good approximations for the models we considered. Since the method itself also is relatively simple we were also able to obtain these estimates relatively quickly.

5.2 Further work and other topics of interest

In relation to the twisted models, there are several topics of interests that we did not have time to discuss in the detail it deserves in Chapter 3 and Chapter 4, but we still want to mention these as they could be of interest for future work. A major topic is how twisted models work in higher-dimensional settings. Guarniero, Johansen and Lee (2017) considers this setting with the iterated auxiliary particle filter, and they are able to obtain low variance estimates for the marginal likelihood when considering linear Gaussian models. This demonstrates that the twisted models also can be used in high-dimensional settings. However, obtaining a good approximation of the optimal sequence is more difficult than in the one-dimensional settings we considered. This has to do with the functions in the optimal sequence also being high-dimensional making the task of obtaining a good approximation a more complex problem.

When considering high-dimensional models, the functions $\psi_t(x_t)$ can also be high-dimensional and it is natural to consider the class to which we restrict these functions. Using a high-dimensional Gaussian mixture density as the approximation will still work as long as it results in a model we are able to work with. For the one-dimensional model in Section 4.3, we observed that increasing the flexibility of the approximation was key for reducing the variance in the estimates of the marginal likelihood, and of interest for future work would be to see if the same variance reduction occurs in higher-dimensional settings.

Another setting we have not discussed is when $g(x_t, y_t)$ and $\psi_t^\star(x_t)$ is multimodal as a function of x_t. This can occur when the state x_t contains information from several different sources, and of interest is if twisted models are applicable in this setting. While the Poisson model we discussed is non-linear in the emissions, we have not discussed models where the transitions are non-linear in the state variable. For both these models, state estimation and smoothing would be a topic of interest, and being able to sample from close to the exact smoothing density would be very useful for these models. The major difference between these models and the one we have considered here is complexity. In these settings the functions $\psi_t^\star(x_t)$ will become much more complex than those we have considered so far, and obtaining good approximations can be difficult. Of interest is if also here we are able to use crude approximations of the optimal functions and still see a benefit from introducing the twisted models.

In addition to the models mentioned above, there are factors related to the approximations of the functions $\psi_t^\star(x_t)$ which also merit discussion. Notable here is how we obtain the approximations of the optimal sequence. One possibility (Guarniero, Johansen and Lee, 2017) is using a non-parametric approximation, rather than the simple parametric approximation we used here. There are no restrictions on how to create approximations of the optimal sequence other than that we need to end up with a twisted model which we are able to work with. For our examples, we used a simple method that was easy to implement and worked well on the examples we considered, but of interest is if there are other efficient ways to obtain good approximations which are suitable for different problems.

To make the most out of the twisted models, we naturally want a good approximation of each $\psi_t^\star(x_t)$ in the regions where we evaluate the functions. In Section 3.4 we discussed this and when $\psi_t^\star(x_t)$ is unimodal we should evaluate the function near the mode. For the models we have considered here we were able to obtain a good approximation of the optimal functions for a large part of the domain where we evaluated the functions. For more complex models there can be regions of the domain where we are not able to obtain good approximations which we observed a tendency of with the Poisson model in Section 4.3. Hence it would be desirable to gain insight into how a poor approximation affects the twisted model and what makes an approximation poor. We have already seen some effect of having a poor approximation. In Section 4.6 we demonstrated how a poor approximation makes it so we are not able to sample from the exact smoothing density and in the same chapter we observed how it affects the variance of the marginal likelihood estimates obtained and the effective number of samples. These can all to some extent be quantified and we use these to gauge the quality of the approximation obtained, but they do not say why a particular approximation is poor. Being able to say which factors make a particular approximation poor and being able to incorporate these factors when creating the approximation could allow for simpler approximations in complex settings. This can in part be done with the central limit theorem for the marginal likelihood given in Guarniero, Johansen and Lee (2017) which gives an expression for the variance of the marginal likelihood estimate. Incorporating knowledge of this variance into the approximation can make the approximation better, and an instance of this is using the particles directly. However, as we observed, this also leads to other difficulties with creating the approximations. Rather than using the particles, the same effect could potentially also be achieved by changing how the particles are laid out or changing the distance measure. For instance, we could attempt to include more information from the observations themselves in order to obtain a better approximation.

In Section 4.3 we commented on how the effective number of samples could be used to diagnose at which times we have a poor approximation. However, this quantity does not say anything about why a particular approximation is poor, but it still has potential uses for improving the approximations further. One potential use would be to include this information when creating the approximations. When considering the iterative approach of Guarniero, Johansen and Lee (2017), it could be used in conjunction with insight on why a particular approximation is poor to further improve the approximation of the optimal sequence each iteration. If we know that a particular approximation is poor we can focus on improving that particular approximation and not the rest of the sequence, and in order to do this, it would be beneficial to have insight into what makes an approximation poor.

APPENDIX A

Calculations

A.1 Calculations related to the twisted model

Here we have considered a setting in which the transitions of the model are linear and Gaussian and the sequence $\boldsymbol{\psi}$ are either a single Gaussian density or a Gaussian mixture. When using the bootstrap particle filter in conjunction with the twisted model we need to sample from the densities (3.1) and (3.2), and when the transitions in the original model is Gaussian and the approximations $\boldsymbol{\psi}$ are Gaussian mixtures the densities we end up sampling from will also be a Gaussian mixture. The calculations giving the densities of the the twisted model are straight forward in this setting since we only have to deal with the product of Gaussian densities. For a product of Gaussian densities $\mathcal{N}(x;\mu_1,\sigma_1^2)$ $\mathcal{N}(x;\mu_2,\sigma_2^2)$ we have the following for a one dimensional x

$$
\begin{aligned}
&\mathcal{N}(x;\mu_1,\sigma_1^2)\cdot\mathcal{N}(x;\mu_2,\sigma_2^2) = \frac{1}{\sqrt{2\pi\sigma_1^2}}e^{-(x-\mu_1)^2/(2\sigma_1^2)}\frac{1}{\sqrt{2\pi\sigma_2^2}}e^{-(x-\mu_2)^2/(2\sigma_2^2)}\\
&\propto \exp\left\{-\frac{1}{2}\left[\frac{x^2}{\sigma_1^2}+\frac{x^2}{\sigma_2^2}-\frac{2x\mu_1}{\sigma_1^2}-\frac{2x\mu_2}{\sigma_2^2}+\frac{\mu_1^2}{\sigma_1^2}+\frac{\mu_2^2}{\sigma_2^2}\right]\right\}\\
&\propto \exp\left\{-\frac{1}{2}\left(\frac{1}{\sigma_1^2}+\frac{1}{\sigma_2^2}\right)\left[x^2-\left(\frac{2x\mu_1}{\sigma_1^2}+\frac{2x\mu_2}{\sigma_2^2}\right)\left(\frac{1}{\sigma_1^2}+\frac{1}{\sigma_2^2}\right)^{-1}\right]\right\}\\
&\propto \exp\left\{-\frac{1}{2}\left(\frac{1}{\sigma_1^2}+\frac{1}{\sigma_2^2}\right)\left[x-\left(\frac{1}{\sigma_1^2}\mu_1+\frac{1}{\sigma_2^2}\mu_2\right)\left(\frac{1}{\sigma_1^2}+\frac{1}{\sigma_2^2}\right)^{-1}\right]^2\right\}\\
&\propto \mathcal{N}(x;\mu,\tau^2) \text{ where } \mu=\left(\frac{1}{\sigma_1^2}\mu_1+\frac{1}{\sigma_2^2}\mu_2\right)\left(\frac{1}{\sigma_1^2}+\frac{1}{\sigma_2^2}\right)^{-1},\ \frac{1}{\tau^2}=\left(\frac{1}{\sigma_1^2}+\frac{1}{\sigma_2^2}\right)
\end{aligned}
$$

Since we only need to sample from this product we can do this calculations up to proportionality and at the final step we recognize the result as a normal density.

For computing the integrals $\widetilde{\psi}_t$ for the models we considered we need to compute integrals related to the product of two gaussian densities. We then get

$$
\begin{aligned}
\mathcal{I} &= \int_{-\infty}^{\infty}\frac{1}{\sqrt{2\pi\sigma_1^2}}e^{-(x-\mu_1)^2/(2\sigma_1^2)}\frac{1}{\sqrt{2\pi\sigma_2^2}}e^{-(x-\mu_2)^2/(2\sigma_2^2)}\,\mathrm{d}x\\
&= \frac{1}{2\pi\sigma_1\sigma_2}\int_{-\infty}^{\infty}\exp\left\{-\frac{1}{2}\left[\frac{x^2}{\sigma_1^2}+\frac{x^2}{\sigma_2^2}-\frac{2x\mu_1}{\sigma_1^2}-\frac{2x\mu_2}{\sigma_2^2}+\frac{\mu_1^2}{\sigma_1^2}+\frac{\mu_2^2}{\sigma_2^2}\right]\right\}\mathrm{d}x
\end{aligned}
$$

There are several ways to solve this integral, perhaps simplest is to consider the term in the exponent as $ax^2 + bx + c$. In particular we then have

$$\mathcal{J} = \int_{-\infty}^{\infty} e^{-(ax^2+bx+c)}\,\mathrm{d}x = \sqrt{\frac{\pi}{a}}e^{-b^2/4a-c}$$

which follows directly from completing the square in the exponent. For the integral we are looking at here, expressed on a logarithmic scale for simplicity after combining all the terms we obtain

$$\log(\mathcal{I}) = \frac{1}{2}\log\left(2\pi(\sigma_1^2+\sigma_2^2)\right) - \frac{1}{2}\left(\frac{\mu_1^2}{\sigma_1^2}+\frac{\mu_2^2}{\sigma_2^2}\right) + \frac{1}{2}\left(\frac{1}{\sigma_1^2}+\frac{1}{\sigma_2^2}\right)^{-1}\left(\frac{\mu_1}{\sigma_1^2}+\frac{\mu_2}{\sigma_2^2}\right)^2$$

These calculations where for one dimension products of Gaussians, for the higher dimensional case see e.g. Petersen and Pedersen (2012). When dealing with Gaussian mixtures, the products which occur will still be on the same form so all the calculations are still valid but will have to be repeated for each product that occurs.

Bibliography

Ala-Luhtala, J. et al. (2016). ‘An introduction to twisted particle filters and parameter estimation in non-linear state-space models’. In: *IEEE Transactions on Signal Processing* vol. 64, no. 18, pp. 4875–4890.

Andrieu, C., Doucet, A. and Holenstein, R. (2010). ‘Particle markov chain monte carlo methods’. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* vol. 72, no. 3, pp. 269–342.

Briers, M., Doucet, A. and Maskell, S. (2010). ‘Smoothing algorithms for state–space models’. In: *Annals of the Institute of Statistical Mathematics* vol. 62, no. 1, pp. 61–89.

Casella, G. and Berger, R. L. (2002). *Statistical inference*. eng. 2nd ed. Duxbury advanced series. Pacific Grove, Calif: Duxbury.

Chopin, N. et al. (2004). ‘Central limit theorem for sequential Monte Carlo methods and its application to Bayesian inference’. In: *Annals of statistics* vol. 32, no. 6, pp. 2385–2411.

Creal, D. (2012). ‘A Survey of Sequential Monte Carlo Methods for Economics and Finance’. eng. In: *Econometric Reviews* vol. 31, no. 3, pp. 245–296.

D’Amico, G. et al. (2019). ‘A Review of Non-Markovian Models for the Dynamics of Credit Ratings’. In: *Reports on Economics and Finance* vol. 5, no. 1, pp. 15–33.

Doucet, A., Godsill, S. and Andrieu, C. (July 2000). ‘On sequential Monte Carlo sampling methods for Bayesian filtering’. In: *Statistics and Computing* vol. 10, no. 3, pp. 197–208.

Doucet, A. and Johansen, A. M. (2009). ‘A tutorial on particle filtering and smoothing: Fifteen years later’. In: *Handbook of nonlinear filtering*.

Dukic, V., Lopes, H. F. and Polson, N. G. (2012). ‘Tracking Epidemics With Google Flu Trends Data and a State-Space SEIR Model’. In: *Journal of the American Statistical Association* vol. 107, no. 500, pp. 1410–1426.

Givens, G. H. and Hoeting, J. A. (2013). *Computational statistics*. eng. 2nd ed. Wiley series in computational statistics. Hoboken, N.J: Wiley.

Godsill, S. J., Doucet, A. and West, M. (2004). ‘Monte Carlo Smoothing for Nonlinear Time Series’. In: *Journal of the American Statistical Association* vol. 99, no. 465, pp. 156–168.

Gordon, N. J., Salmond, D. J. and Smith, A. F. (1993). ‘Novel approach to nonlinear/non-Gaussian Bayesian state estimation’. In: *IEE proceedings F (radar and signal processing)*. Vol. 140. 2. IET, pp. 107–113.

Guarniero, P., Johansen, A. M. and Lee, A. (2017). ‘The Iterated Auxiliary Particle Filter’. In: *Journal of the American Statistical Association* vol. 112, no. 520, pp. 1636–1647.

Harvey, A. C. (1989). *Forecasting, structural time series models and the Kalman filter*. eng. Cambridge: Cambridge University Press.

Heng, J. et al. (2020). ‘Controlled sequential Monte Carlo’. In: *The Annals of Statistics* vol. 48, no. 5, pp. 2904–2929.

Johansen, A. M. and Doucet, A. (2008). ‘A note on auxiliary particle filters’. In: *Statistics & Probability Letters* vol. 78, no. 12, pp. 1498–1504.

Kalman, R. E. (Mar. 1960). ‘A New Approach to Linear Filtering and Prediction Problems’. In: *Journal of Basic Engineering* vol. 82, no. 1, pp. 35–45.

Kantas, N. et al. (Aug. 2015). ‘On Particle Methods for Parameter Estimation in State-Space Models’. In: *Statist. Sci.* vol. 30, no. 3, pp. 328–351.

Kim, S., Shepherd, N. and Chib, S. (1998). ‘Stochastic Volatility: Likelihood Inference and Comparison with ARCH Models’. In: *The review of economic studies.* vol. 65, no. 3, pp. 361–393.

Kitagawa, G. (1994). ‘The two-filter formula for smoothing and an implementation of the Gaussian-sum smoother’. In: *Annals of the Institute of Statistical Mathematics* vol. 46, no. 4, pp. 605–623.

Kitagawa, G. (1996). ‘Monte Carlo Filter and Smoother for Non-Gaussian Nonlinear State Space Models’. In: *Journal of Computational and Graphical Statistics* vol. 5, no. 1, pp. 1–25.

Kucharski, A. J. et al. (2020). ‘Early dynamics of transmission and control of COVID-19: a mathematical modelling study’. In: *The Lancet Infectious Diseases* vol. 20, no. 5, pp. 553–558.

Lin, M., Chen, R. and Liu, J. S. (2013). ‘Lookahead Strategies for Sequential Monte Carlo’. In: *Statistical Science* vol. 28, no. 1, pp. 69–94.

Liu, J. and West, M. (2001). ‘Combined Parameter and State Estimation in Simulation-Based Filtering’. In: *Sequential Monte Carlo Methods in Practice.* Ed. by Doucet, A., Freitas, N. de and Gordon, N. New York, NY: Springer New York, pp. 197–223.

Liu, J. S. and Chen, R. (1995). ‘Blind deconvolution via sequential imputations’. In: *Journal of the american statistical association* vol. 90, no. 430, pp. 567–576.

Liu, J. S., Chen, R. and Logvinenko, T. (2001). ‘A Theoretical Framework for Sequential Importance Sampling with Resampling’. In: *Sequential Monte Carlo Methods in Practice.* Ed. by Doucet, A., Freitas, N. de and Gordon, N. New York, NY: Springer New York, pp. 225–246.

Petersen, K. B. and Pedersen, M. S. (Nov. 2012). *The Matrix Cookbook.* Version 20121115.

Petris, G., Petrone, S. and Campagnoli, P. (2009). *Dynamic Linear Models with R.* eng. 1st ed. 2009. New York, NY: Springer New York : Imprint: Springer.

Pitt, M. K. and Shephard, N. (1999). ‘Filtering via simulation: Auxiliary particle filters’. In: *Journal of the American statistical association* vol. 94, no. 446, pp. 590–599.

Rizzo, M. L. (2007). *Statistical Computing with R.* eng. 1st ed. London: CRC Press LLC.

Shumway, R. H. and Stoffer, D. S. (2017). *Time Series Analysis and Its Applications: With R Examples.* eng. Springer Texts in Statistics. Springer International Publishing AG.

Tsay, R. S. and Chen, R. (2018). ‘State Space Models’. In: *Nonlinear Time Series Analysis*. John Wiley & Sons, Ltd. Chap. 6, pp. 265–335.

Van Rossum, G. and Drake, F. L. (2009). *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace.

Walter, J.-C. and Barkema, G. (2015). ‘An introduction to Monte Carlo methods’. eng. In: *Physica A* vol. 418, pp. 78–87.

Whiteley, N. and Lee, A. (2014). ‘Twisted particle filters’. In: *The Annals of Statistics* vol. 42, no. 1, pp. 115–141.