Parameter Estimation for State Space Models using Sequential Monte Carlo Algorithms

Christopher John Nemeth, B.Sc. (Hons.), M.Sc., M.Res.

Submitted for the degree of Doctor of Philosophy at

Lancaster University.

November 2014

STOR-i
excellence with impact

Dedicated to my wife, Lena

Abstract

State space models represent a flexible class of Bayesian time series models which can be applied to model latent state stochastic processes. Sequential Monte Carlo (SMC) algorithms, also known as particle filters, are perhaps the most widely used methodology for inference in such models, particularly when the model is nonlinear and cannot be evaluated analytically. The SMC methodology allows for the sequential analysis of state space models in online settings for fast inference, but can also be applied to study offline problems. This area of research has grown rapidly over the past 20 years and has lead to the development of important theoretical results.

This thesis builds upon the SMC framework to address problems of parameter estimation for state space models. Due to the nonlinearity of some models, maximising the likelihood function of a state space model cannot be done analytically. This thesis proposes a new methodology for performing parameter estimation based on a gradient ascent algorithm, where the gradient is approximated using a particle filter. This new approach is shown to estimate parameters both online and offline and with a computational cost that is linear in the number of particles. This is an improvement over previously proposed approaches which either display quadratically increasing

variance in the estimate of the gradient, or carry a computational cost which scales quadratically with the number of particles.

Combining the advantages of SMC and Markov chain Monte Carlo (MCMC) the recently proposed particle MCMC methodology can be applied to estimate parameters. This thesis proposes a new class of efficient proposal distributions which take account of the geometry of the target density. This is achieved by using particle approximations of the gradient of the target within the proposal mechanism.

Finally, a new algorithm is introduced for estimating piecewise time-varying parameters for target tracking problems.

Acknowledgements

I would like to thank my supervisors Professor Paul Fearnhead and Dr Lyudmila Mihaylova for their guidance and support over the past three years. I am honoured to have had the opportunity to work with Paul. A brilliant, modest and innovative statistician, from whom I have learned so much. Mila has been a constant source of encouragement during the PhD. She has always pushed me to better myself and my research, and for that I am grateful. I could not have asked for better supervisors.

Also, in terms of supervision, I would also like to thank Dr Chris Sherlock for teaching me so much about optimal scaling rules for MCMC and also Professor Simon Godsill for hosting my stay in the Signal Processing group at Cambridge earlier this year.

This final version of the thesis has been improved thanks to the helpful comments of my viva examiners, Dr Sumeetpal Singh and Dr Gareth Ridall.

My PhD has been supported by the EPSRC funded Statistics and Operational Research (STOR-i) doctoral training centre, and by MBDA UK. I am grateful to STOR-i for providing, not only a stimulating research environment, but also additional training opportunities that have allowed me to develop my research skills. In

particular, I would like to thank Professor Jon Tawn and Professor Idris Eckley for their guidance and support during my studies.

As part of STOR-i I have met many wonderful people who have made my time in Lancaster memorable and enjoyable. In particular, I would to like thank my close friends (soon-to-be Drs) Jamie Fairbrother, Shreena Patel, Tim Park and Mark Bell who have been there to suffer through my many frustrations and complaints. Thanks guys.

Finally, I would like to acknowledge the encouragement and support of my parents and grandparents, not only during my PhD, but in all of my academic pursuits. Most important of all, I am eternally grateful to my wife, Lena. Her love, patience, strength and unyielding support has made all of this possible. With the recent arrival of our son, Atticus, I will forever be astounded by the energy and passion that Lena has put into our small and growing family. I love you.

Declaration

I declare that the work in this thesis has been done by myself and has not been submitted elsewhere for the award of any other degree.

Christopher John Nemeth

Contents

List of Figures

List of Tables

List of Abbreviations

a.s.	almost surely
APE	adaptive parameter estimation
APF	auxiliary particle filter
CLT	central limit theorem
EM	expectation maximisation
GARCH	generalised autoregressive conditional heteroskedasticity
iid	independent and identically distributed
IMM	interacting multiple models
LW	Liu and West
MCMC	Markov chain Monte Carlo
ML	maximum likelihood
PL	particle learning
pMALA	particle Metropolis adjusted Langevin algorithm
PMCMC	particle Markov chain Monte Carlo
PMMH	particle marginal Metropolis Hastings
RMS	root mean squared

SIR sequential importance resampling

SIS sequential importance sampling

SMC sequential Monte Carlo

Chapter 1

Introduction

This thesis addresses the problem of performing inference for unobserved latent stochastic processes which evolve over time. While the latent process itself is not directly observable, it is assumed that partial and possibly noisy observations of it are available. Using these noisy partial observations the aim is to infer the latent process, or possible features of it. Such processes can be generally classed as time series models, where a stochastic process is used to describe a system of random variables as they evolve over time. The stochastic process itself is parameterised via a set of parameters θ, which are often unknown. Moreover, this thesis focuses on approaches for estimating the unknown parameters via the sequence of noisy partial observations.

The class of time series models covered in this thesis are generally known as *hidden Markov models*, and also referred to as *state space models*. This chapter provides a gentle introduction to state space modelling, where mathematical notation is kept to a minimum and will be covered in greater detail in subsequent chapters. This chapter also describes the contributions of this thesis and gives an outline of the chapters

herein.

1.1 Background to state space modelling

General time series models contain random variables $\{X_t\}_{t\geq 1}$, where t is a natural number, that describe a sequence of observed data points $\{x_t\}_{t\geq 1}$. In state space modelling, it is assumed that x_t is not directly observable, but can be observed indirectly and possibly with noise via a second process $\{Y_t\}_{t\geq 1}$. The observations y_t are assumed to be independent of one another conditional on the *hidden/latent* process X_t. Furthermore, the latent process X_t is Markov, (see Figure 1.1.1 for a graphical representation). The Markovian structure ensures that, conditional on X_{t-1}, X_t is independent of the history of the process.

Interest now lies in inferring the latent variables from the sequence of partial, noisy observations. This process is known as filtering.

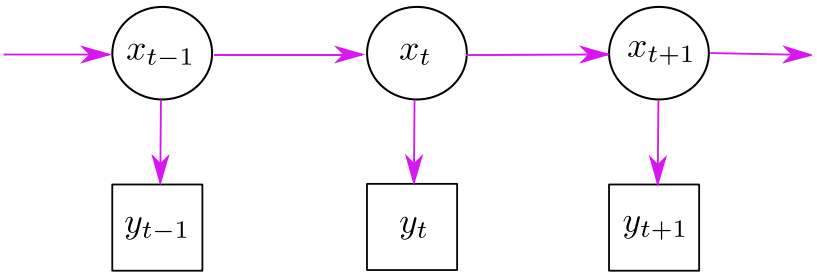


Figure 1.1.1: Graphical representation of a state space model

To better understand the state space framework, consider the following example. We have a first order autoregressive model where conditional on $X_{t-1} = x_{t-1}$, $X_t = \phi X_{t-1} + \epsilon_t$, where ϕ is the autoregressive parameter with the constraint $|\phi| \leq 1$ to ensure stationarity of the process, and $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$ is zero mean Gaussian noise

with variance σ^2. Notice that the state of the process at time t depends linearly on the previous state of the same process at time $t-1$. Extending this model to the state space framework it is assumed that X_t is not directly observable, but instead observed with noise. Our observation at time t is the realisation of a random variable, $Y_t = X_t + \nu_t$, where $\nu_t \sim \mathcal{N}(0, \tau^2)$ is Gaussian noise with variance τ^2. This particular example represents as a special class of state space models known as linear-Gaussian models.

A simulated realisation of this model with model parameters $\theta = (\phi, \sigma^2, \tau^2) = (0.99, 0.09, 1)$ is given in Figure 1.1.2. The aim is then to try and reconstruct the path of the latent process using only the sequence of observations $y_{1:T} = \{y_1, y_2, \ldots, y_T\}$. For linear-Gaussian state space models, it is possible to recover the latent process optimally using the Kalman filter (Kalman, 1960) (details are give in Chapter 3).

The importance of filters such as the Kalman filter is illustrated in Figure 1.1.2. Without directly observing the latent state the only information available with which to infer the latent state is generated by the observations y_t. Using the observations alone provides a poor representation of the latent state, but by applying a filter, and in this case the Kalman filter, it is possible to produce estimates of the latent process which are much closer to the truth than would be available from only the raw observations.

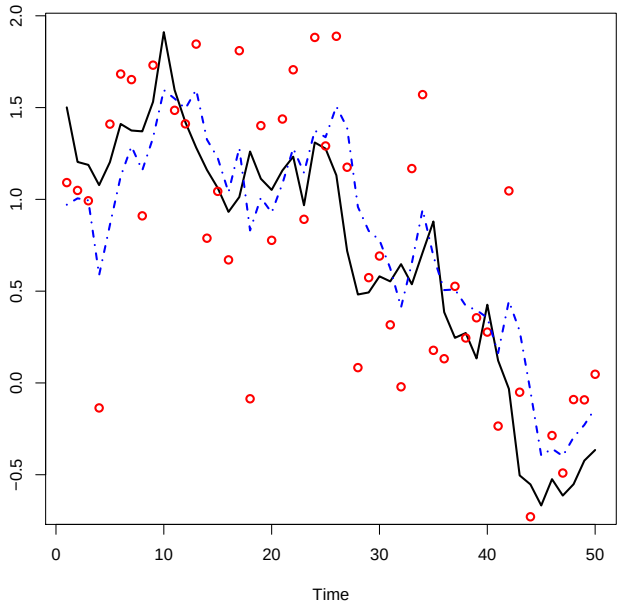


Figure 1.1.2: Example of a state space model. The true path of the latent state $\{X_t\}_{1\leq t\leq 50}$ (solid black line) is simulated from the autoregressive model, but is not directly observable. Partial observations of the state $\{y_t\}_{1\leq t\leq 50}$ (red dots) are noisy Gaussian perturbations of the true state. The Kalman filter provides an estimate of the latent process from the noisy observations (dashed blue line).

1.2 Challenges of state and parameter estimation

When working with state space models, one of the most common problems is to estimate the latent state x_t at time t given all of the observations recorded up to that point $y_{1:t}$. This problem is known as *filtering* and is perhaps the most addressed issue in the state space modelling literature (Jazwinski, 1970; Kitagawa, 1996; Cappé et al., 2005). Extensions to this problem include *smoothing*, here the aim is to estimate x_t given $y_{1:T}$, where $t < T$, and *prediction*, where we try to estimate the state k steps

ahead (i.e. x_{t+k} given $y_{1:t}$).

State space models can be applied to modelling a wide range of real world stochastic processes from missile tracking (Salmond and Gordon, 2001) to DNA segmentation (Fearnhead, 2007). However, many of the models used to represent real world processes contain nonlinear terms or are observed after being perturbed by non-Gaussian noise. In such situations, the Kalman filter is no longer capable of optimally reconstructing the latent process. Variations of the Kalman filter and alternative techniques, such sequential Monte Carlo algorithms (detailed in Chapter 3), can be applied.

As statisticians we are also interested in estimating the parameters θ of the state space model. The standard approach of maximising the likelihood function is difficult as this function is generally unavailable in closed form. The exception being when the state space model is discrete or linear-Gaussian. This is still an open research problem and addressed as the main theme of this thesis. In particular, this thesis focuses on parameter estimation based on sequential Monte Carlo algorithms.

1.3 Contributions and thesis outline

The focus of this thesis is parameter estimation for state space models with new approaches proposed from both likelihood and Bayesian perspectives. The research contained in this thesis provides computationally efficient, and accurate approaches, to parameter estimation for fixed and time-varying parameters. Furthermore, this thesis provides a methodology to perform parameter estimation *offline* using batches

of data, and *online*, where parameters are estimated recursively as new observations are received.

The main material in this thesis is presented in five chapters which contain a literature review of the area (Chapters 2 and 3), and new research that has been submitted for journal publication (Chapters 4, 5 and 6). A brief outline for each chapter is given as follows:

Chapter 2: Monte Carlo Methods

A review of popular Monte Carlo methods from the computational statistics literature is provided. In particular, methods including rejection sampling, importance sampling, Markov chain Monte Carlo and the Metropolis Hastings algorithm are presented as a background material. Some of these methods are later developed in subsequent chapters and applied to state space models.

Chapter 3: Bayesian Inference for State Space Models

This chapter presents a literature review of the important developments in Bayesian inference for state space models. After introducing the Bayesian framework, this chapter covers some of the most important methodology in the literature that has been applied to inference for state space models. Most notably, the sequential Monte Carlo approach is introduced and heavily utilised in future chapters. A section of this chapter is dedicated to reviewing previous approaches proposed for parameter estimation and a discussion of the various approaches is also provided.

Chapter 4: Particle Approximations of the Score and Observed Information Matrix for Parameter Estimation in State Space Models with Linear Computational Cost

This chapter is a journal contribution and has been submitted for publication with co-authors Professor Paul Fearnhead and Dr Lyudmila Mihaylova. This paper is available as an arXiv preprint, arXiv:1306.0735.

This chapter addresses the problem of performing maximum likelihood estimation on nonlinear state space models, where the likelihood function is unavailable in closed form. The solution proposed in this chapter utilises a gradient ascent algorithm to indirectly maximise the likelihood function. Gradient based methods for maximising the likelihood using particle approximations of the score and observed information matrix have previously been considered by Poyiadjis et al. (2011). However, these algorithms produce approximations which either display quadratically increasing variance in the estimate of the score function as the length of the data set increases, or carry a computational cost which scales quadratically in the number of particles. This chapter proposes a new algorithm for estimating the score and observed information matrix which displays only a linearly increasing variance with a computational cost that scales linearly in the number of particles. These approximations are then applied to a gradient ascent algorithm to estimate the model parameters. This approach is an improvement over competing methods in terms of

both accuracy in parameter estimation and computational savings.

Chapter 5: Particle Metropolis adjusted Langevin Algorithms for State Space Models

This chapter is a journal contribution and has been submitted for publication with co-author Professor Paul Fearnhead. This paper is available as an arXiv preprint, arXiv:1402.0694.

The recently developed particle MCMC methodology (Andrieu et al., 2010), and in particular the particle marginal Metropolis Hastings algorithm (PMMH), has become a widely popular approach for parameter estimation in state space models. Compared to the standard Metropolis Hastings (MH) algorithm the PMMH algorithm replaces the intractable likelihood with an unbiased estimator given by a particle filter. So far, the PMMH algorithm has been implemented using the random walk Metropolis (RWM) proposal, where recent results by Sherlock et al. (2013) have shown that the optimal acceptance rate for this proposal is 7% (compared to 23.4% for the MH algorithm). In this chapter a new proposal distribution which is referred to as particle MALA is introduced. This proposal can be viewed as a particle approximation of the Metropolis adjusted Langevin algorithm (MALA) (Roberts and Rosenthal, 1998) which uses the gradient of the posterior within the proposal. MALA has been shown in the MH literature to improve the mixing of the MCMC sampler and increase the acceptance rate (optimally 57.4%). Similarly, the particle

MALA proposal is shown to increase the acceptance rate of the particle MCMC sampler and reduce the autocorrelation of the resulting Markov chain compared to the RWM proposal.

Chapter 6: Sequential Monte Carlo Methods for State and Parameter Estimation in Abruptly Changing Environments

This chapter is a journal contribution and has been published with co-authors Professor Paul Fearnhead and Dr Lyudmila Mihaylova. This paper has appeared in the journal IEEE Transactions on Signal Processing, 62(5):1245-1255, 2014.

This chapter addresses the problem of tracking highly manoeuvrable targets where the aim is to estimate the position and velocity of a target based on noisy, partial observations. This work extends beyond standard target tracking problems where the model parameters which govern the target's motion are often assumed to be fixed, which is a reasonable assumption when tracking targets with predictable behaviour. However, by treating the parameters as piecewise time-varying, it is possible to account for a greater range of target behaviour and therefore reduce the possibility of losing track of a target. This work has been presented at the 9th IET Data Fusion and Target Tracking conference, London and the 15th international conference on Information Fusion, Singapore. This chapter is also part of an ongoing collaboration with an industrial partner, MBDA UK.

Chapter 2

Monte Carlo Methods

There are many statistical problems where it is necessary to evaluate an integral, such as when computing probabilities and expectations. Assume interest lies in integrating some measurable function $\psi : \mathcal{X} \to \mathbb{R}^d$ with respect to a probability density function p, often referred to as the target density, on some measurable space $\mathcal{X}$. If we let X be a random variable distributed according to $p(x)$ then the expected value of $\psi(x)$ is

$$\mathbb{E}[\psi(X)] = \int \psi(x)p(x)dx. \tag{2.0.1}$$

In an ideal setting, computing integrals such as (2.0.1) is done analytically. However, for a wide class of problems it is not possible to evaluate integrals analytically. Numerical integration methods such as those based on quadrature rules can be applied, but are infeasible for large dimensional spaces as the number of function evaluations required for a reasonable level of accuracy increases exponentially with dimension. This curse of dimensionality restricts the use of numerical integration methods to low dimensional problems.

Monte Carlo methods represent a class techniques where it is possible to approximate the integral (2.0.1) using samples simulated from the target density. This chapter reviews Monte Carlo techniques used to evaluate integrals, and provides a basis for future chapters where the evaluation of integrals is performed sequentially, known as sequential Monte Carlo. This is an expansive topic and only the parts which are relevant for the subsequent chapters of this thesis are presented. The interested reader is referred to Robert and Casella (2004) and Cappé et al. (2005) for further details.

2.1 Perfect Monte Carlo

We start by assuming that it is possible to draw independent identically distributed (iid) samples $\{x^{(i)}\}_{i=1}^{N}$ from $p(x)$. An empirical approximation $\hat{p}(x)$ to the target density $p(x)$ can then be given by perfect Monte Carlo sampling

$$\hat{p}(dx) = \frac{1}{N}\sum_{i=1}^{N}\delta_{x^{(i)}}(dx),$$

where $\delta_{x^{(i)}}(dx)$ denotes the Dirac delta function located at $x^{(i)}$. This empirical approximation of the target density can be used to evaluate the integral (2.0.1) to give the sample average

$$\hat{p}(\psi) = \int \psi(x)\hat{p}(dx) = \frac{1}{N}\sum_{i=1}^{N}\psi(x^{(i)}). \tag{2.1.1}$$

Monte Carlo approximations are popular methods due to their simplicity and good statistical properties. Firstly, it is possible to show that the Monte Carlo approximation $\hat{p}(\psi)$ is an unbiased estimator for $\mathbb{E}[\psi]$. Secondly, by the law of large numbers

the Monte Carlo approximation converges almost surely (a.s.),

$$\hat{p}(\psi) \overset{a.s.}{\to} \mathbb{E}[\psi]$$

as the number of iid samples $N \to \infty$.

Assuming that the variance of $\psi(x)$ is finite, then the variance of $\hat{p}(\psi)$ is

$$\text{var}\left[\hat{p}(\psi)\right] = \frac{1}{N^2}\sum_{i=1}^{N}\text{var}\left[\psi(x^{(i)})\right] = \frac{1}{N}\text{var}\left[\psi(X)\right],$$

which displays the desirable property that the accuracy of the Monte Carlo approximation increases with N. This property holds regardless of the dimension of $\mathcal{X}$, which highlights an important benefit of Monte Carlo methods over numerical integration alternatives. Furthermore, if the variance of $\psi(x)$ is finite then the following central limit theorem (CLT) holds,

$$\sqrt{N}(\hat{p}(\psi) - \mathbb{E}[\psi(x)]) \overset{\mathcal{D}}{\to} \mathcal{N}(0, \text{var}\left[\psi(X)\right]), \quad \text{as} \quad N \to \infty, \tag{2.1.2}$$

where $\overset{\mathcal{D}}{\to}$ represents convergence in distribution. The rate of convergence of (2.1.2) is $O(N^{-1/2})$ and independent of the dimension of $\mathcal{X}$.

The results given above for perfect Monte Carlo require that we can draw iid samples from the target distribution. For the remainder of this chapter will shall address the issue of applying Monte Carlo methods to problems where iid samples from the target distribution are unavailable. It is possible to circumvent this problem by using alternative sampling strategies. In this chapter we shall present rejection sampling, importance sampling and Markov chain Monte Carlo (MCMC) methods as possible alternatives.

2.1.1 Rejection sampling

There are many situations where it is not possible to sample from the target density $p(x)$. One such situation is when the target density is known only up to a constant of proportionality $\tilde{p}(x)$, i.e. $p(x) = \tilde{p}(x)/Z$, where Z is an unknown constant. This is a common problem in Bayesian statistics where for complex models the posterior distribution, which is proportional to the likelihood and prior distribution, is known only up to a constant of proportionality.

Rejection sampling was first presented by von Neumann (1951) as a Monte Carlo method to draw samples from the target density $p(x)$, via a proposal density $q(x)$ which can be easily sampled from. In order to ensure that the samples drawn from the proposal $q(x)$ are distributed according the target $p(x)$, we require the support of $q(x)$ to cover the support of $p(x)$ and for some constant M, $q(x)$ should be chosen such that $\forall x$, $\tilde{p}(x) \leq Mq(x)$.

Figure 2.1.1 provides a visual illustration of rejection sampling, where a sample x is drawn from the proposal density $q(x)$ and accepted as a sample from the target density $p(x)$ if $u \leq \tilde{p}(x)/Mq(x)$, where $u \sim \mathcal{U}(0, 1)$ is a uniform distribution between 0 and 1. This procedure is summarised in Algorithm 1.

The justification for this methods is as follows. Let A be a subset of the support of $p(x)$. Using Bayes' theorem we can calculate the distribution of the accepted samples,

$$\Pr(X \in A|\text{X is accepted}) = \frac{\Pr(X \in A, \text{X is accepted})}{\Pr(\text{X is accepted})}. \tag{2.1.3}$$

If X is distributed according to $q(\cdot)$, and the probability that X is accepted is

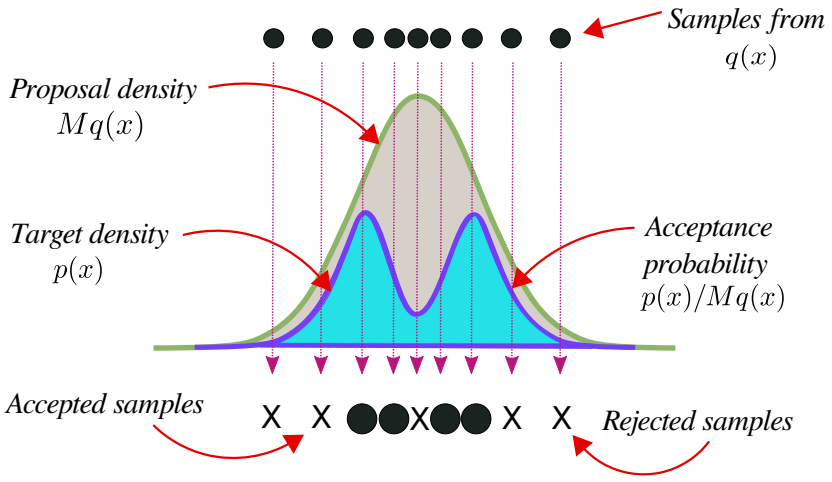


Figure 2.1.1: Rejection sampler. Samples are randomly drawn from the green shaded region (i.e. from the proposal distribution), whose support covers the support of the target. Samples which fall within the blue region are accepted as draws from the target. Whereas, samples which fall outside of the blue region are rejected.

$\tilde{p}(x)/Mq(x)$, then the joint probability given in the numerator is

$$\Pr(X \in A, \text{X is accepted}) = \int_A q(x)\frac{\tilde{p}(x)}{Mq(x)}dx = \int_A \frac{Z}{M}p(x)dx, \tag{2.1.4}$$

and the denominator term $\Pr(\text{X is accepted}) = Z/M$ is given by integrating (2.1.4).

Plugging both terms into (2.1.3) we have

$$\Pr(X \in A|\text{X is accepted}) = \int_A p(x)dx.$$

As A is arbitrary it can be concluded that x is distributed according to $p(x)$.

The efficiency of the rejection sampling algorithm is dependent on the choice of M and $q(x)$. It can be seen from the probability of acceptance that for large M we will accept fewer samples, which suggests that M should be small. However, from Figure 2.1.1 it can be seen that M must be sufficiently large so that $Mq(x)$ completely

Algorithm 1 Rejection Sampling

Step 1: Sampling

Sample $x \sim q(\cdot)$ and $u \sim \mathcal{U}(0, 1)$.

Step 2: Accept-Reject

If: $u \leq \tilde{p}(x)/Mq(x)$

then accept x as a sample from $p(x)$

else: reject x and return to *Step 1*

envelopes $p(x)$. The proposal $q(x)$ should be chosen such that it best mimics the characteristics of $p(x)$, e.g. captures the modes of the target density. In practice, and in particular for large $\mathcal{X}$, it is difficult to find an appropriate proposal density and therefore rejection sampling is often restricted to simple and low-dimensional problems.

2.1.2 Importance sampling

In rejection sampling only a subset of the samples drawn from the proposal are used to approximate the target density. This sampling scheme can therefore be computationally wasteful, particularly for large M as the probability of accepting a sample is $1/M$. Alternatively, importance sampling does not waste rejected samples, but instead weights all samples according to the similarity between the target and proposal distributions (Geweke, 1989) (see Figure 2.1.2).

Importance sampling is a Monte Carlo method for evaluating integrals (2.0.1) using samples drawn from a proposal density $q(x)$. The expectation of the function $\psi(x)$ over the target $p(x)$ can instead be evaluated over the proposal $q(x)$, where

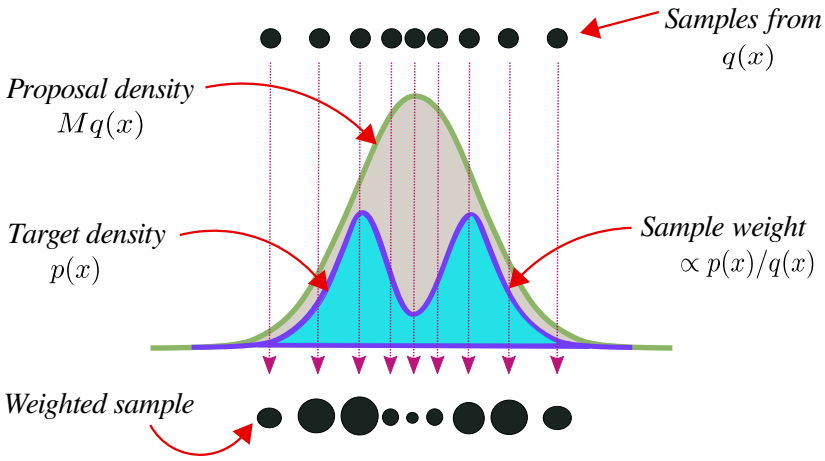


Figure 2.1.2: Importance sampler. Samples are drawn from the proposal distribution and rather than being accepted or rejected, as in the rejection sampler, all of the samples are accepted. The samples are weighted to give a measure of their fit to the target, where samples with larger weights contribute more when evaluating integrals of the form (2.0.1).

the proposal density is chosen such that its support covers the support of the target density, i.e. $p(x) > 0 \Rightarrow q(x) > 0$.

$$\begin{aligned} \mathbb{E}_p[\psi(x)] &= \int \psi(x)p(x)dx = \int \psi(x)\frac{p(x)}{q(x)}q(x)dx \\ &= \int \psi(x)w(x)q(x)dx = \mathbb{E}_q[\psi(x)w(x)], \end{aligned} \tag{2.1.5}$$

where $w(x) = p(x)/q(x)$ is the importance weight.

Sampling $\{x^{(i)}\}_{i=1}^N$ from $q(x)$ the integral (2.1.5) can be approximated with a Monte Carlo estimate

$$\hat{p}(\psi) = \int \psi(x)w(x)q(x)dx = \frac{1}{N}\sum_{i=1}^{N} w(x^{(i)})\psi(x^{(i)}),$$

which is an unbiased estimator that converges in the same sense as the perfect Monte

Carlo estimator. Using the samples from the proposal density it is possible to construct and empirical distribution as a weighted sample $\{w(x^{(i)}), x^{(i)}\}_{i=1}^{N}$ which approximates the target

$$\hat{p}(dx) = \frac{1}{N}\sum_{i=1}^{N} w(x^{(i)})\delta_{x^{(i)}}(dx).$$

In the previous section the problem of unknown normalising constants was considered, where the target $p(x) = \tilde{p}(x)/Z$ may be known only up to a constant of proportionality. In this setting the Monte Carlo estimate for (2.1.5) is

$$\mathbb{E}[\psi(x)] = \int \psi(x)\frac{\hat{p}(x)}{Zq(x)}q(x)dx = \frac{1}{ZN}\sum_{i=1}^{N}\tilde{w}^{(i)}\psi(x^{(i)}),$$

where $\tilde{w}^{(i)} = \tilde{p}(x^{(i)})/q(x^{(i)})$ are now unnormalised importance weights. The normalisation constant now appears in the Monte Carlo estimate, but using the samples $\{x^{(i)}\}_{i=1}^{N}$ drawn from the proposal we can approximate the normalising constant

$$Z = \int \tilde{p}(x)dx = \int \frac{\tilde{p}(x)}{q(x)}q(x)dx \approx \frac{1}{N}\sum_{i=1}^{N}\tilde{w}^{(i)}.$$

Using this approximation we now have a normalised importance sampling estimator

$$\hat{p}(\psi) = \frac{\frac{1}{N}\sum_{i=1}^{N}\tilde{w}^{(i)}\psi(x^{(i)})}{\frac{1}{N}\sum_{i=1}^{N}\tilde{w}^{(i)}} = \sum_{i=1}^{N} w^{(i)}\psi(x^{(i)}),$$

where $\{w^{(i)}\}_{i=1}^{N}$ are the normalised importance weights. By the law of large numbers this estimator converges to $\mathbb{E}_p[\psi(x)]$ (Robert and Casella, 2004). The Monte Carlo estimate $\hat{p}(\psi)$ is however biased for finite sample sizes with a decreasing bias as the number of samples increases. Casella and Robert (1998) have shown that even if Z were known, this estimator is often preferable as it can produce lower variance estimators than the standard importance sampler. Details of the importance sampler are summarised in Algorithm 2

Algorithm 2 Importance Sampling

Step 1: For $i = 1, \ldots, N$. Sample: $x^{(i)} \sim q(\cdot)$.

Step 2: For $i = 1, \ldots, N$.

Calculate the importance weights: $\tilde{w}^{(i)} = \tilde{p}(x^{(i)})/q(x^{(i)})$.

Normalise the importance weights: $w^{(i)} = \tilde{w}^{(i)} / \sum_{i=1}^{N} \tilde{w}^{(i)}$.

Step 3: Monte Carlo estimate: $\hat{p}(\psi) = \sum_{i=1}^{N} w^{(i)} \psi(x^{(i)})$.

As for the rejection sampler, the choice of proposal distribution plays an important role in the efficiency of the importance sampler. While any appropriately chosen proposal density will lead to a consistent estimator of $\hat{p}(\psi)$, the variance of the estimator

$$\text{var}_q[\hat{p}(\psi)] = \frac{1}{N}\text{var}_q[w(x)\psi(x)] = \frac{1}{N}\left[\mathbb{E}_q[w(x)^2\psi^2(x)] - \mathbb{E}_p[\psi(x)]^2\right]$$

is dependent on the choice of $q(x)$. It is clear to see that the variance can be reduced by minimising $\mathbb{E}_q[w(x)^2\psi^2(x)]$. Jensen's inequality gives,

$$\mathbb{E}_q[w(x)^2\psi^2(x)] \geq \mathbb{E}_q[w(x)|\psi(x)|]^2 = \left(\int |\psi(x)|p(x)dx\right)^2$$

as a lower bounded and choosing the proposal

$$q(x) = \frac{|\psi(x)|p(x)}{\int |\psi(y)|p(y)dy},$$

attains this lower bound. See Theorem 3.12 of Robert and Casella (2004) for a proof of this result. While the optimal proposal may not be available for many problems, it can, however, be used as a guide to designing near optimal proposals.

2.2 Markov chain Monte Carlo (MCMC)

So far we have seen how Monte Carlo methods can be used to approximate integrals of interest. This is done by drawing samples from the target distribution and using these samples to create a Monte Carlo estimator of the integral. However, it is often not possible to draw samples directly from the the target, and so to circumvent this problem we introduced a proposal distribution. It is often easier to sample from the proposal distribution and create a set of samples which approximate the target distribution. In this section we shall consider another approach for sampling from the target distribution based on a Markov chain which admits the target as its stationary distribution. Creating a Markov chain means that the samples are now no longer iid, but are in fact dependent. Convergence results in the literature (Robert and Casella, 2004; Gilks et al., 1999; Cappé et al., 2005) establish the necessary justification for this approach.

A Markov chain is a collection of dependent samples $\{x^{(n)}, n \geq 0\}$ where the probability distribution of $x^{(n)}$ given the previous samples is dependent only on $x^{(n-1)}$:

$$P(x^{(n)}|x^{(n-1)}, x^{(n-2)}, \ldots, x^{(0)}) = P(x^{(n)}|x^{(n-1)}) = K(x^{(n-1)}, x^{(n)}).$$

The conditional distribution $K(\cdot, \cdot)$ is known as the transition kernel. The target distribution $p(x)$ is said to be the invariant distribution if $x^{(n-1)} \sim p(\cdot) \implies x^{(n)} \sim p(\cdot)$. Formally, the kernel must satisfy

$$\int K(x, y)p(x)dx = p(y).$$

If the Markov chain $x^{(0)}, x^{(1)}, \ldots, x^{(n)}$ is irreducible and aperiodic with invariant dis-

tribution $p(x)$, then $x^{(n)} \to x \sim p(\cdot)$ in distribution as $n \to \infty$ and from (2.1.1)

$$\frac{1}{N}\sum_{i=1}^{N}\psi(x^{(i)}) \to \mathbb{E}_p[\psi(X)], \tag{2.2.1}$$

with probability one as $n \to \infty$ (Roberts and Rosenthal, 2004). Therefore samples generated from the transition kernel $K(\cdot,\cdot)$ can be used to create an empirical approximation of the expectation of $\psi(x)$ with respect to the target density $p(x)$. It is also possible to derive central limit theorem results for these estimators. Further theoretical results regarding Markov chain Monte Carlo algorithms can be found in Chapter 6 of Robert and Casella (2004).

2.2.1 Metropolis Hastings

The Metropolis Hastings algorithm, first presented by Metropolis et al. (1953) and later developed by Hastings (1970), is a method for constructing a Markov chain with the correct stationary distribution.

This algorithm works on the assumption that it is not possible to directly simulate samples from the target distribution $p(x)$, or that it may be known only up to a constant of proportionality. New samples x' are instead drawn from the target distribution via a proposal distribution $q(x'|x)$, where x is the current state of the Markov chain. The new sample is either accepted as the next state of the Markov chain or rejected with probability

$$\alpha(x,x') = \min\left\{1, \frac{p(x')q(x'|x)}{p(x)q(x|x')}\right\}.$$

The samples drawn from the proposal distribution $q(\cdot|\cdot)$ form a Markov chain which

admits $p(x)$ as its stationary distribution. See Algorithm 3 for a summary of the Metropolis Hastings algorithm.

Algorithm 3 Metropolis Hasting Algorithm

Step 1: At iteration n. Sample: $x' \sim q(\cdot|x^{(n-1)})$.

Step 2: Calculate acceptance probability: $\alpha(x, x') = \min\left\{1, \frac{p(x')q(x'|x^{(n-1)})}{p(x^{(n-1)})q(x^{(n-1)}|x')}\right\}$.

Step 3: Accept-reject sample: With probability α accept $x^{(n)} = x'$ otherwise $x^{(n)} = x^{(n-1)}$.

To prove that the Metropolis Hastings algorithm has the correct stationary distribution we use the idea of *detailed balance*.

Definition 2.2.1. *Let $\{x^{(n)}, n \geq 0\}$ be a Markov chain with an arbitrary transition kernel $K(x, y)$. The Markov chain is reversible if the transition kernel satisfies*

$$K(x, y)p(x) = K(y, x)p(y).$$

This condition, also known as detailed balance, shows that at stationarity, the probability of being at x and moving from x to x' is the same as the probability of being at x' and moving from x' to x.

If detailed balance is satisfied it is then straightforward to show that the Markov chain has $p(x)$ as its stationary distribution

$$\int K(x, y)p(x)dx = \int K(y, x)p(y)dx = p(y)\int K(y, x)dx = p(y),$$

as K is a normalised density which integrates to 1.

To check that this holds for the Metropolis Hasting algorithm consider the transition kernel

$$K(x, x') = \alpha(x, x')q(x'|x) + \left(1 - \int \alpha(x, y)q(y|x)dy\right)\delta_x(x'), \tag{2.2.2}$$

where $\delta_x(x')$ is the Dirac mass at x. From Algorithm 3, this kernel accounts for the possibility that a new sample x' is accepted to the Markov chain with probability $\alpha(x, x')$, or that the chain stays with x.

Proposition 2.2.2. *The Metropolis Hastings kernel* (2.2.2) *satisfies the detailed balance condition (Definition2.2.1) and therefore admits* $p(x)$ *as its stationary distribution.*

Proof. We consider the two components of the transition kernel separately. Firstly, it is trivial to show that

$$\left(1 - \int \alpha(x, y) q(y|x) dy\right) \delta_x(x') = \left(1 - \int \alpha(x', y) q(y|x') dy\right) \delta_{x'}(x)$$

satisfies detailed balance when x' is rejected. For the other term we have that

$$\begin{aligned} K(x, x')p(x) &= \alpha(x, x')q(x'|x)p(x) \\ &= \min\left\{1, \frac{q(x'|x)p(x')}{q(x|x')p(x)}\right\} q(x'|x)p(x) = \min\{q(x'|x)p(x), q(x'|x)p(x')\} \\ &= \min\left\{\frac{q(x|x')p(x)}{q(x'|x)p(x')}, 1\right\} q(x|x')p(x') = \alpha(x', x)q(x|x')p(x') = K(x', x)p(x') \end{aligned}$$

□

Finally, the validity of the Metropolis Hastings algorithm is completed by providing some weak conditions on the proposal distribution $q(\cdot|\cdot)$. These conditions ensure that the Markov chain converges to the stationary distribution, and furthermore the convergence of the Monte Carlo approximation (2.2.1) to $\mathbb{E}_p[\psi(X)]$. By the ergodic theorem (Theorem 6.63 of Robert and Casella (2004)), if the Markov chain is *aperiodic* and *Harris recurrent* (see Robert and Casella (2004) Chapter 6 for details), then the

Metropolis-Hastings algorithm will eventually generate samples from the stationary distribution $p(x)$, whereby

$$\lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} \psi(x^{(i)}) \overset{a.s.}{\to} \mathbb{E}_p[\psi].$$

The first of these sufficient conditions is aperiodicity. To satisfy this condition the Metropolis Hastings algorithm is to allow the event $x^{(n)} = x^{(n-1)}$ to occur with a non-zero probability and thus,

$$P[p(x^{(n-1)})q(y|x^{(n-1)}) \leq p(y)q(x^{(n-1)}|y)] < 1.$$

The property of irreducibility of the Metropolis Hastings chain is satisfied if

$$q(y|x) > 0 \qquad \forall (x, y) \in \mathcal{X} \times \mathcal{X}.$$

Furthermore, by Lemma 7.3 of Robert and Casella (2004) it can be shown that an irreducible Metropolis Hastings chain is also Harris recurrent. Therefore, any proposal distribution which satisfies these conditions will eventually produce samples from the stationary distribution $p(x)$ and by the ergodic theorem, the sample mean will converge to $\mathbb{E}_p[\psi(X)]$ almost surely.

It is important to mention that the asymptotic variance of the sample mean depends on the limiting autocovariance of the Markov chain. Choosing proposals which minimise the autocovariance will lead to more accurate estimators of $\mathbb{E}_p[\psi(X)]$ for finite N. A further discussion of the importance in choosing appropriate proposals is given in Chapter 5.

2.2.2 Gibbs sampling

The Gibbs sampler (Geman and Geman, 1984) is a specific case of the Metropolis Hastings sampler that can be applied to multivariate problems (i.e. $x^{(n)} = (x_1^{(n)}, \dots x_j^{(n)})$). At iteration n the ith component of the state is denoted $x_i^{(n)}$ with the remaining components denoted as $x_{-i}^{(n)}$. The ith component of the state can then be updated conditional on the remaining components $p(x_i^{(n)}|x_{-i}^{(n)})$.

It is straightforward to see that the Gibbs sampler is a special case of the Metropolis Hastings algorithm, where at iteration n the proposed sample x' is accepted. Consider the proposed move from $(x_i^{(n)}, x_{-i}^{(n)})$ to $(x_i', x_{-i}^{(n)})$ and denote the marginal distribution as

$$p(x_{-i}^{(n)}) = \int p(x_i^{(n)}, x_{-i}^{(n)}) dx_i^n.$$

From the Metropolis Hastings ratio the acceptance probability is

$$\frac{q(x_i^{(n)}|x_{-i}^{(n)})p(x_i', x_{-i}^{(n)})}{q(x_i'|x_{-i}^{(n)})p(x_i^{(n)}, x_{-i}^{(n)})} = \frac{q(x_i^{(n)}|x_{-i}^{(n)})p(x_i'|x_{-i}^{(n)})p(x_{-i}^{(n)})}{q(x_i'|x_{-i}^{(n)})p(x_i^{(n)}|x_{-i}^{(n)})p(x_{-i}^{(n)})} = 1,$$

implying that all proposed samples are accepted. To ensure that the Markov chain is reversible, at each iteration every component of $x^{(n)}$ is updated. Updates can proceed according to a deterministic ordering or can be chosen randomly (Liu et al., 1995). For the deterministic ordering approach, also known as systematic sweep, at each iteration the algorithm samples are drawn from x_1 to x_j, followed by a reverse sweep from x_j to x_1. This is to ensure that the Markov chain is reversible. Without this condition the chain would not satisfy detailed balance, but given that detailed balance is only a sufficient condition for $p(x)$ to be the stationary distribution, it does not directly imply that without this condition $p(x)$ cannot be the stationary distribution of the

chain.

The main drawback of this method is the requirement that we can sample from the conditional density $p(x_i^{(n)}|x_{-i}^{(n)})$. For many problems this density is unavailable, in which case the practitioner is likely to resort to the Metropolis Hastings sampler.

The Monte Carlo methods outlined in this section shall be used extensively in future chapters as approximation methods for intractable densities. The choice of appropriate Monte Carlo method is often specific to the problem in question, but all of the methods outlined above have been extensively applied in the statistical literature.

Chapter 3

Bayesian Inference for State Space Models

3.1 State space models

State space models, also known as *hidden Markov models*, represent a class of latent state models where the latent state $\{X_t\}_{t\geq 1}$ follows a Markov process taking values on some measurable space $\mathcal{X} \subseteq \mathbb{R}^{n_x}$. The Markov process is fully specified by its initial density $X_1 \sim \mu_\theta(\cdot)$ and transition density

$$X_t | X_{t-1} = x_{t-1} \sim f_\theta(\cdot | x_{t-1}), \tag{3.1.1}$$

where $\theta \in \Theta \subseteq \mathbb{R}^d$ represents a vector of model parameters. We assume that the latent process $\{X_t\}_{t\geq 1}$ is not directly observable, but inference about the latent states can be made via a set of partial observations $\{Y_t\}_{t\geq 1} \subseteq \mathbb{R}^{n_y}$, which we assume are independent, conditional on the latent process. The marginal probability density of

the observations conditional on the latent state is

$$Y_t|X_t = x_t \sim g_\theta(\cdot|x_t). \tag{3.1.2}$$

The structure of the hidden Markov model, and in particular the independence between the observations, can be best represented graphically as shown in Figure 1.1.1.

For the first part of this chapter the model parameters θ are assumed known. However, in practice this not the case and towards the end of this chapter approaches for estimating the parameters will be discussed.

3.2 Bayesian filtering

In the context of state space models, interest lies in estimating the latent process $X_{1:T} = \{X_1, X_2, \ldots, X_T\}$, given a sequence of observations $y_{1:T}$. This can be expressed as the posterior density of the latent process given the observations $p(x_{1:T}|y_{1:T}, \theta)$, which is proportional to $p(x_{1:T}, y_{1:T}, \theta)$,

$$p(x_{1:T}, y_{1:T}, \theta) = \mu_\theta(x_1) \prod_{t=2}^{T} f_\theta(x_t|x_{t-1}) \prod_{t=1}^{T} g_\theta(y_t|x_t). \tag{3.2.1}$$

Often we are only interested in the the marginal posterior $p(x_t|y_{1:t}, \theta)$, also known as the *filtered density*. Using Bayes theorem it is possible to recursively update the estimate of the filtered density

$$p(x_t|y_{1:t}, \theta) = \int \frac{g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1})}{p(y_t|y_{1:t-1}, \theta)} p(x_{t-1}|y_{1:t-1}, \theta) dx_{t-1}, \tag{3.2.2}$$

where the predictive likelihood is given by

$$p(y_t|y_{1:t-1}, \theta) = \int g_\theta(y_t|x_t) \int f_\theta(x_t|x_{t-1}) p(x_{t-1}|y_{1:t-1}, \theta) dx_t dx_{t-1}. \tag{3.2.3}$$

For the general state space model it is not possible to evaluate the filtered density analytically as the integrals in (3.2.2) and (3.2.3) are intractable. An exception is when the state space model is discrete (Rabiner, 1989) or linear-Gaussian (Durbin and Koopman, 2001), in which case the filtered density can be evaluated using the Kalman filter.

3.3 Kalman filter

The Kalman filter (Kalman, 1960) is an algorithm which estimates the filtered density $p(x_t|y_{1:t}, \theta)$ recursively as new observations are received. It can be shown that when the state space model is linear-Gaussian, the Kalman filter estimate of the filtered density is optimal in terms of minimising mean squared error. State space models which are linear-Gaussian are of the form

$$\begin{aligned} X_t|X_{t-1} = x_{t-1} &\sim \mathcal{N}(Fx_{t-1}, V_t) \\ Y_t|X_t = x_t &\sim \mathcal{N}(Gx_t, W_t), \end{aligned} \tag{3.3.1}$$

where F and G are the transition and observation matrices and V_t and W_t are the variances of the system and observation noise.

Utilising the linear-Gaussian structure of the state space model implies that the filtered densities are also Gaussian. Assuming that the filtered density at time $t-1$ is available as

$$p(x_{t-1}|y_{1:t-1}, \theta) = \mathcal{N}(\mu_{t-1}, \Sigma_{t-1}),$$

it is then possible to derive the filtered density at time t using the Kalman filter recursions. Given that the updates are linear and the densities Gaussian, the new

filtered density at time t will also be Gaussian and therefore can be fully specified by its mean and covariance. Essentially, the Kalman filter provides a way of calculating μ_t and Σ_t for $t = 1, \dots T$.

The problem of estimating the filtered density (3.2.2) can be decomposed into two procedures, *prediction*

$$p(x_t|y_{1:t-1}, \theta) = \int f_\theta(x_t|x_{t-1}) p(x_{t-1}|y_{1:t-1}, \theta) dx_{t-1} \tag{3.3.2}$$

and *update*

$$p(x_t|y_{1:t}, \theta) = \frac{g_\theta(y_t|x_t) p(x_t|y_{1:t-1}, \theta)}{p(y_t|y_{1:t-1}, \theta)}.$$

The predictive density (3.3.2) is the convolution of two Gaussian distributions, which in turn is a Gaussian distribution. The Gaussian distribution can be fully specified in terms of its mean and variance, where by using the decomposition of variance property, and Tower's property, we have

$$\begin{aligned}
\mathbb{E}[X_t|Y_{1:t-1}] &= \mathbb{E}[\mathbb{E}[X_t|X_{t-1}, Y_{1:t-1}]|Y_{1:t-1}] = F\mu_{t-1} \\
\text{var}\,[X_t|Y_{1:t-1}] &= \mathbb{E}[\text{var}\,[X_t|X_{t-1}, Y_{1:t-1}]|Y_{1:t-1}] + \text{var}\,[\mathbb{E}[X_t|X_{t-1}, Y_{1:t-1}]|Y_{1:t-1}] \\
&= V_t + F\Sigma_{t-1}F^T = C_t,
\end{aligned}$$

which gives $X_t|Y_{1:t-1} = y_{1:t-1} \sim \mathcal{N}(F\mu_{t-1}, C_t)$.

The predictive density is then updated to take account of the newest observation y_t, where the filtered density, given up to a constant of proportionality is

$$\begin{aligned}
p(x_t|y_{1:t}, \theta) &\propto g_\theta(y_t|x_t) p(x_t|y_{1:t-1}, \theta) \\
&\propto \exp\left(-\frac{1}{2}(y_t - Gx_t)^T W_t^{-1}(y_t - Gx_t)\right) \exp\left(-\frac{1}{2}(x_t - F\mu_{t-1})C_t^{-1}(x_t - F\mu_{t-1})\right) \\
&\propto \exp\left(-\frac{1}{2}(x_t^T(C_t^{-1} + G^T W_t^{-1} G)x_t - 2x_t^T(C_t^{-1}F\mu_{t-1} + G^T W_t^{-1} y_t))\right),
\end{aligned}$$

which is the form of a Gaussian distribution with mean and variance

$$\begin{aligned}
\mu_t &= \Sigma_t(C_t^{-1}F\mu_{t-1} + G^T W_t^{-1} y_t) \\
&= \Sigma_t((V_t + F\Sigma_{t-1}F^T)^{-1}F\mu_{t-1} + G^T W_t^{-1} y_t) \\
\Sigma_t^{-1} &= C_t^{-1} + G^T W_t^{-1} G \\
&= (V_t + F\Sigma_{t-1}F^T)^{-1} + G^T W_t^{-1} G,
\end{aligned}$$

resulting in the filtered density $X_t|Y_{1:t} = y_{1:t} \sim \mathcal{N}(\mu_t, \Sigma_t)$ at time t.

The Kalman filter has been extensively and successfully applied over the past 50 years in numerous fields including engineering, economics and seismology. For nonlinear state space models such as

$$\begin{aligned}
X_t|X_{t-1} = x_{t-1} &\sim \mathcal{N}(f(x_{t-1}), V_t), \\
Y_t|X_t = x_t &\sim \mathcal{N}(g(x_t), W_t),
\end{aligned}$$

where f and g are nonlinear functions, the optimality of the Kalman filter recursions no longer holds. The *extended Kalman filter* (Jazwinski, 1970; Bar-Shalom et al., 2001) is a popular technique applied to filtering problems involving nonlinear functions. It works by first assuming a Gaussian approximation to the filtered density at time $t-1$ which has mean $\hat{\mu}_{t-1}$ and covariance $\hat{\Sigma}_{t-1}$. Taking local linearisations of the nonlinear equations $f(\cdot)$ and $g(\cdot)$, we can approximate our model by,

$$\begin{aligned}
X_t|X_{t-1} = x_{t-1} &\sim \mathcal{N}(f(\hat{\mu}_{t-1}) + (x_{t-1} - \hat{\mu}_{t-1})\nabla f, V_t), \\
Y_t|X_t = x_t &\sim \mathcal{N}(\nabla g, W_t),
\end{aligned}$$

where,

$$\nabla f = \left.\frac{\partial f(x_{t-1})}{\partial x}\right|_{\hat{\mu}_{t-1}} \qquad \nabla g = \left.\frac{\partial g(x_t)}{\partial x}\right|_{f(\hat{\mu}_{t-1})}$$

are the derivatives of the nonlinear functions $f(\cdot)$ and $g(\cdot)$ taken with respect to a potentially multivariate state. Standard Kalman filter recursions, as given above, can then be applied to update the mean (from $\hat{\mu}_{t-1}$ to $\hat{\mu}_t$) and covariance (from $\hat{\Sigma}_{t-1}$ to $\hat{\Sigma}_t$) of the filtered density (see Chapter 10 of Bar-Shalom et al. (2001) for further details).

The extended Kalman filter has been known to struggle in situations where the state space model is highly nonlinear. Variations of the extended Kalman filter, such as the use of higher order Taylor series expansions, can be applied. However, such variations often carry an increased computational cost. A popular alternative to the extended Kalman filter is the unscented Kalman filter proposed by Julier et al. (2000) which relies on unscented transforms (Julier et al., 1995). The main difference between the extended and unscented Kalman filters is that the extended filter tries to approximate the nonlinear function, whereas the unscented filter aims to directly approximate the filtered density. This is done using a Gaussian approximation represented by a set of deterministically selected points called sigma points. Compared to the extended Kalman filter, which is based on a first order approximation, the unscented Kalman filter can be shown to be a third order approximation (Wan and van der Merwe, 2001) and can outperform the extended filter in many applications.

3.4 Sequential Monte Carlo (SMC)

As already discussed, for linear-Gaussian state space models the Kalman filter can be applied to optimally estimate the filtered density, $p(x_t|y_{1:t}, \theta)$. In the case of

nonlinear or non-Gaussian state space models, the integrals in (3.2.2) and (3.2.3) become intractable. Variations of the Kalman filter, such as the extended Kalman filter and unscented Kalman filter can be applied as approximation methods. However, these variations do not retain the optimality properties of the Kalman filter, and depending on the degree of nonlinearity and non-Gaussianity of the state space model, these approximations to the Kalman filter can perform poorly.

In Chapter 2 Monte Carlo methods were introduced as a class of techniques to overcome issues of intractability. Perhaps the most widely used Monte Carlo method, the MCMC algorithm, could be applied to estimate the latent state x_t conditional on the observations $y_{1:t}$. However, unlike the Kalman filter and its variants, MCMC methods are of limited use for online problems. The computational cost of applying MCMC increases linearly due to re-calculating the whole path when each new observation is received. A solution to this problem is to truncate the state space such that MCMC is applied to some fixed length subset of the data (Gilks and Berzuini, 2001). This approach introduces a bias into the state estimate and its efficiency will be dependent on the forgetting properties of the state space model (Cappé et al., 2005). In other words, may perform well if data from the distant past has little influence on the current state.

It is now widely accepted that the preferred approach to the recursive analysis of state space models is via sequential Monte Carlo methods, which, when applied to state space models, are more commonly known as *particle filters*. This class of approximation methods are based on importance sampling techniques which have the desirable property that they can be applied online with a fixed computational

cost. Particle filters provide an empirical approximation of the posterior density (3.2.2) using a collection of samples ("particles") and corresponding weights. From the discussion on importance sampling in Chapter 2, we can create a Monte Carlo approximation of the filtered density

$$\hat{p}(x_t|y_{1:t},\theta) = \sum_{i=1}^{N} w_t^{(i)} \delta_{x_t^{(i)}}(dx_t),$$

where $\delta(\cdot)$ is the Dirac delta function, $\{x_t^{(i)}\}_{i=1}^N$ is the set of N particles and $\{w_t^{(i)}\}_{i=1}^N$ are the associated weights.

The approximation to the filtered density $\hat{p}(x_t|y_{1:t},\theta)$ is updated recursively as new observations are received. Applying importance sampling sequentially, the particles are propagated from $x_t^{(i)}$ to $x_{t+1}^{(i)}$ and the weights are updated from $w_t^{(i)}$ to $w_{t+1}^{(i)}$, giving an approximation to the filtered density $\hat{p}(x_{t+1}|y_{1:t+1},\theta)$ at time $t+1$. The same procedure is then repeated for $t = 1,\ldots,T$.

3.4.1 Sequential importance sampling

We shall start by considering the sequential importance sampling (SIS) filter (see Liu and Chen (1998) and Arulampalam et al. (2002) for details). This simple filtering algorithm is the basis for most sequential Monte Carlo filters, and it shall be shown later, how this can be improved upon by choosing better proposal distributions for the importance sampler.

Due to the intractability of the normalising constant (3.2.3) the filtered density is often known only up to a constant of proportionality

$$p(x_t|y_{1:t},\theta) \propto \int g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1}) p(x_{t-1}|y_{1:t-1},\theta) dx_{t-1}. \tag{3.4.1}$$

If we assume that at time $t-1$ we have a particle approximation $\{w_{t-1}^{(i)}, x_{t-1}^{(i)}\}_{i=1}^{N}$ of the filtered density $p(x_{t-1}|y_{1:t-1}, \theta)$, then we can approximate (3.4.1) by

$$\hat{p}(x_t|y_{1:t}, \theta) \propto \sum_{i=1}^{N} w_{t-1}^{(i)} g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1}^{(i)}).$$

Using importance sampling, we can create a particle approximation for the filtered density at time t by sampling $x_t^{(i)}$ from a proposal distribution $q(x_t|x_{1:t-1}, y_t, \theta)$ and updating the weights

$$w_t^{(i)} \propto w_{t-1}^{(i)} \frac{g_\theta(y_t|x_t^{(i)}) f_\theta(x_t^{(i)}|x_{t-1}^{(i)})}{q(x_t^{(i)}|x_{1:t-1}^{(i)}, y_t, \theta)}. \tag{3.4.2}$$

As the filtered density is only known up to a constant of proportionality, the importance weights must be normalised (as discussed in Chapter 2), such that $\sum_{i=1}^{N} w_t^{(i)} = 1$.

The SIS filter also allows for the approximation of the joint density $p(x_{1:t}|y_{1:t}, \theta)$ using the particle filter

$$\hat{p}(x_{1:t}|y_{1:t}, \theta) = \sum_{i=1}^{N} w_t^{(i)} \delta_{x_{1:t}^{(i)}}(dx_{1:t}). \tag{3.4.3}$$

This is a straightforward extension of the SIS filter where we now store the entire path of the particles $x_{1:t}^{(i)} = \{x_{1:t-1}^{(i)}, x_t^{(i)}\}$ and update the weights in the same way (3.4.2). The accuracy of this approximation degrades as t increases, this is caused by the degeneracy of the importance weights, where after a few time steps only a small portion of the weights will contain most of the probability mass (Cappé et al., 2007). Thus only a few particles will contribute to the approximation of the joint density.

Particle degeneracy

Particle degeneracy is one of the major issues for the practical implementation of particle filters. The variance of the importance weights increases over time, and

as a result, fewer and fewer particles have non-negligible weights. Therefore, only a few particles will be used to approximate the filtered density, producing a poor approximation and also wasting computational effort in maintaining the remaining particles. The problem of degeneracy is apparent for all possible recursive proposal distributions as the variance of the importance weights increases over time (see Kong et al. (1994) for full discussion),

$$\begin{aligned}
\text{var}\left[w_t\right] &= \text{var}\left[w_{t-1}\frac{g_\theta(y_t|x_t)f_\theta(x_t|x_{t-1})}{q(x_t|x_{1:t-1},y_t,\theta)}\right] \\
&= \text{var}\left[\mathbb{E}\left[w_{t-1}\frac{g_\theta(y_t|x_t)f_\theta(x_t|x_{t-1})}{q(x_t|x_{1:t-1},y_t,\theta)}\middle| x_{1:t-1},y_t\right]\right] \\
&+ \mathbb{E}\left[\text{var}\left[w_{t-1}\frac{g_\theta(y_t|x_t)f_\theta(x_t|x_{t-1})}{q(x_t|x_{1:t-1},y_t,\theta)}\middle| x_{1:t-1},y_t\right]\right] \\
&\geq \text{var}\left[w_{t-1}\mathbb{E}\left[\frac{g_\theta(y_t|x_t)f_\theta(x_t|x_{t-1})}{q(x_t|x_{1:t-1},y_t,\theta)}\middle| x_{1:t-1},y_t\right]\right] = \text{var}\left[w_{t-1}\right].
\end{aligned}$$

The problem of increasing variance through time is an unfortunate feature of sequential importance sampling. In practice, a resampling scheme is introduced to stabilise the asymptotic variance.

Resampling the particles

Resampling the particles with replacement is one way to reduce the variance of the importance weights, where the probability of resampling a particle is proportional to its importance weight. The intuition behind this is that particles with negligible weights will be discarded while the useful particles will be replicated. After the particles have been resampled all of the weights are reset to $w_t^{(i)} = 1/N$, thus reducing the variance of the weights.

Resampling the particles has the advantage of reducing the degeneracy of the im-

portance weights, and improving the long term stability of the particle filter. This comes at the short term cost of increasing the Monte Carlo variance as the new particle set is now an approximation of the old set. Therefore, it is recommended that estimation is performed prior to resampling (Liu and Chen, 1998). By duplicating some of the particles we have now impoverished the particle set by reducing its diversity. This is particularly noticeable in the approximation to the joint density (3.4.3), as many paths of the particles x_k for $k << t$ will now be identical.

From a theoretical perspective, by resampling we lose the standard convergence results for importance sampling as the particle paths are now dependent. Central limit theorem results have, however, been given by Chopin (2004) for most of the popular resampling procedures. From a practical point of view the dependence between particles caused by the resampling step makes it difficult to parallelise the particle filter.

Resampling is a necessary evil when applying particle filters in order to reduce the degeneracy of the importance weights, but the issues given above can be mitigated by (i) only resampling when necessary and (ii) choosing an efficient resampling procedure. If resampling is used to reduce the degeneracy of the particle approximation, then it stands to reason that resampling should only be performed when the particle approximation has degenerated beyond some tolerance threshold. Kong et al. (1994) and Liu (1996) introduced the effective sample size heuristic N_{eff} as a measure of particle degeneracy. This metric considers the number of particles that would be required from the target density to achieve the same variance as N particles drawn from the proposal distribution. It is not possible to evaluate N_{eff} and so Kong et al.

(1994) introduced an estimate of the effective sample size

$$\hat{N}_{eff} = \frac{1}{1 + \sum_{i=1}^{N} (w_t^{(i)})^2},$$

and suggested that resampling should be performed when $\hat{N}_{eff}$ drops below some threshold, such as $N/2$.

In the simplest resampling scheme, multinomial resampling (Gordon et al., 1993), particles are resampled with probability proportional to their weights. If N_i is the number of times that particle i is resampled then the multinomial resampling scheme satisfies the important unbiasedness property (Cappe et al., 2005) where

$$\mathbb{E}[N_i] = N w_t^{(i)},$$

which ensures that the mean of the particle approximation is preserved through resampling. As already noted, resampling increases the Monte Carlo variance of the estimators, and so while it is necessary to resample, it is possible to choose a resampling strategy which minimises the variance introduced through sampling. These methods include residual resampling (Liu and Chen, 1998), stratified resampling (Carpenter et al., 1999) and systematic resampling (Kitagawa, 1996).

Sampling importance resampling (SIR)

One of the most simple and popular particle filters was proposed by Gordon et al. (1993) and is known as the sequential importance resampling (SIR) algorithm, or the bootstrap filter. This filter can be viewed as an extension to the SIS filter where we now include a resampling step, as detailed above, to reduce particle degeneracy. Resampling the particles with replacement can be applied at each iteration of the

algorithm, or only when the particles degenerate to such an extent that resampling is necessary. In the case of the SIR filter, the proposal distribution is chosen to be $q(x_t|x_{1:t-1}, y_t, \theta) = f_\theta(x_t|x_{t-1})$. Using the transition density as the proposal distribution is a popular choice as it is often easy to sample from, and also simplifies the importance weights

$$w_t^{(i)} \propto g_\theta(y_t|x_t^{(i)}).$$

Details of the complete SIR filter are summarised in Algorithm 4.

Algorithm 4 Sequential Importance Resampling Filter

Step 1: Iteration $t = 1$,

Sample particles $\{x_1^{(i)}\}$ from the prior $p(x_1|\theta)$ and $\forall i$ set weights $w_1^{(i)} = g_\theta(y_1|x_1^{(i)})$.

Step 2: Iteration $t = 2, \ldots, T$. Assume a set of particles $\{x_{t-1}^{(i)}\}_{i=1}^N$ and associated weights $\{w_{t-1}^{(i)}\}_{i=1}^N$ that approximate $p(x_{t-1}|y_{1:t-1}, \theta)$.

(a) Resample particles $\{x_{t-1}^{(i)}\}_{i=1}^N$ with probabilities $w_{t-1}^{(i)}$.

(b) Propagate particles $x_t^{(i)} \sim f_\theta(x_t^{(i)}|x_{t-1}^{(i)})$.

(c) Weight each particle $w_t^{(i)} \propto g_\theta(y_t|x_t^{(i)})$ and normalise the weights such that $\sum_{i=1}^N w_t^{(i)} = 1$.

Optimal proposal distributions

The use of resampling techniques helps to alleviate some particle degeneracy, but at a short term cost of increasing the Monte Carlo variance. Improved resampling techniques can reduce the introduced variance but only reduce the variance created by the resampling procedure. From the discussion on importance sampling in Chapter 2, the variance of the importance sampling estimators is dependent on the choice of

proposal distribution. It is therefore possible to improve the particle approximation of the target density by choosing a proposal which closely matches the target. This is equivalent to all of the importance weights being approximately equal, and thus the variance of the weights will be close to zero.

Proposition 3.4.1. *The optimal proposal density which minimises the variance of the importance weights is* $q(x_t|x_{1:t-1}, y_t, \theta) = p(x_t|x_{t-1}, y_t, \theta)$.

Proof. Recall that the importance weight at time t is $w_t^{(i)} = \frac{w_{t-1}^{(i)} g_\theta(y_t|x_t^{(i)}) f_\theta(x_t^{(i)}|x_{t-1}^{(i)})}{q(x_t^{(i)}|x_{1:t-1}^{(i)}, y_t, \theta)}$. Straightforward calculations of the variance gives

$$\begin{aligned}
\text{var}_q[w_t^{(i)}|w_{t-1}^{(i)}, x_{t-1}^{(i)}] &= \int (w_t^{(i)})^2 q(x_t|x_{1:t-1}, y_t, \theta) dx_t - \left(\int (w_t^{(i)}) q(x_t|x_{1:t-1}, y_t, \theta) dx_t \right)^2 \\
&= \int (w_{t-1}^{(i)})^2 \frac{(g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1}^{(i)}))^2}{q(x_t|x_{1:t-1}^{(i)}, y_t, \theta)} - \left(\int (w_t^{(i)}) g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1}^{(i)}) dx_t \right)^2 \\
&= (w_{t-1}^{(i)})^2 \left(\int \frac{(g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1}^{(i)}))^2}{p(x_t|x_{t-1}^{(i)}, y_t, \theta)} - \left(p(y_t|x_{t-1}^{(i)}, \theta) \right)^2 \right) \\
&= (w_{t-1}^{(i)})^2 \left(\int \frac{(g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1}^{(i)}))^2 p(y_t|x_{t-1}^{(i)}, \theta)}{g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1}^{(i)})} dx_t - \left(p(y_t|x_{t-1}^{(i)}, \theta) \right)^2 \right) \\
&= (w_{t-1}^{(i)})^2 \left(p(y_t|x_{t-1}^{(i)}, \theta) \int p(y_t, x_t|x_{t-1}^{(i)}, \theta) dx_t - (p(y_t|x_{t-1}^{(i)}, \theta))^2 \right) \\
&= (w_{t-1}^{(i)})^2 \left((p(y_t|x_{t-1}^{(i)}, \theta))^2 - (p(y_t|x_{t-1}^{(i)}, \theta))^2 \right) = 0
\end{aligned}$$

□

In order to apply the optimal proposal distribution it is necessary to sample from $p(x_t|x_{t-1}, y_t, \theta)$ and evaluate $p(y_t|x_{t-1}, \theta) = \int g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1}) dx_t$, neither of which may be available in closed form. The exception to this is the linear-Gaussian state space model (3.3.1) and the extension to nonlinear transition equations $f_\theta(\cdot|\cdot)$.

3.4.2 Auxiliary particle filter

Ideally one would sample particles from the optimal proposal distribution, but as discussed above, for most state space models the optimal proposal distribution is not available. Alternatively, it may be possible to approximate the optimal proposal distribution and instead sample particles from this approximation. If the approximation closely resembles the optimal proposal, then we can expect a better approximation to the target density than would be given by alternative proposals, such as the transition density $f\theta(\cdot|\cdot)$ as used in the bootstrap filter.

Pitt and Shephard (1999) introduced the *auxiliary particle filter* as a means to approximate the optimal proposal distribution. Consider re-writing the filtered density as,

$$\begin{aligned}\hat{p}(x_t|y_{1:t},\theta) &\propto \sum_{i=1}^{N} w_{t-1}^{(i)} g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1}^{(i)}) \\ &= \sum_{i=1}^{N} \frac{g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1}^{(i)})}{\int g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1}^{(i)}) dx_t} w_{t-1}^{(i)} \int g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1}^{(i)}) dx_t.\end{aligned}$$

We note that

$$p(x_t|x_{t-1}^{(i)}, y_t, \theta) = \frac{g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1}^{(i)})}{\int g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1}^{(i)}) dx_t},$$

and thus the filtered density can be written as

$$\hat{p}(x_t|y_{1:t},\theta) = \sum_{i=1}^{N} \zeta_t^{(i)} p(x_t|x_{t-1}^{(i)}, y_t, \theta), \tag{3.4.4}$$

where $\zeta_t^{(i)} \propto w_{t-1}^{(i)} \int g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1}^{(i)}) dx_t$.

The filtered density (3.4.4) can be viewed as a mixture of $p(x_t|x_{t-1}^{(i)}, y_t, \theta)$ densities each weighted by $\zeta_t^{(i)}$. The auxiliary particle filter introduces an auxiliary variable i

to represent the ith component of the mixture density. The joint distribution of x_t and index i is then,

$$\hat{p}(x_t, i|y_{1:t}, \theta) = \zeta_t^{(i)} p(x_t|x_{t-1}^{(i)}, y_t, \theta).$$

A proposal to sample from the joint distribution can then be constructed:

$$q(x_t, i|y_{1:t}, \theta) = \xi_t^{(i)} q(x_t|x_{t-1}^{(i)}, y_t, \theta),$$

where, $\{\xi_t^{(i)}\}_{i=1}^N$ is a set of probabilities which approximate $\zeta_t^{(i)}$, and as before $q(x_t|x_{t-1}^{(i)}, y_t, \theta)$ is a density that approximates $p(x_t|x_{t-1}^{(i)}, y_t, \theta)$.

It is then possible to sample from the joint target density by first sampling an index k_i from the discrete set $\{1, 2, \ldots, N\}$ with probability $\xi_t^{(k_i)}$ and then sampling x_t from the proposal $q(x_t|x_{t-1}^{(k_i)}, y_t, \theta)$, conditional on the index. Sampling the new particles in this manner means that the resampling strategy used to reduce particle degeneracy in the SIR filter is now intrinsic to the filter, and not simply an add on feature to improve the performance of the filter. In the original paper by Pitt and Shephard (1999) there was an extra resampling step similar to the SIR filter. Carpenter et al. (1999) showed that this extra resampling step was not necessary and can even reduce the performance of the filter, it is therefore not included here.

Applying the sampling procedure produces the pair $\{x_t^i, k_i\}$ which has importance weight

$$w_t^{(i)} \propto \frac{\hat{p}(x_t^{(i)}, k_i|y_{1:t}, \theta)}{q(x_t^{(i)}, k_i|y_{1:t}, \theta)} = \frac{\zeta_t^{(k_i)} p(x_t^{(i)}|x_{t-1}^{(k_i)}, y_t, \theta)}{\xi_t^{(k_i)} q(x_t^{(i)}|x_{t-1}^{(k_i)}, y_t, \theta)} = \frac{w_{t-1}^{(k_i)} g_\theta(y_t|x_t^{(i)}) f_\theta(x_t^{(i)}|x_{t-1}^{(k_i)})}{\xi_t^{(k_i)} q(x_t^{(i)}|x_{t-1}^{(k_i)}, y_t, \theta)}. \quad (3.4.5)$$

By dropping the indices k_i we now have a particle approximation $\{w_t^{(i)}, x_t^{(i)}\}_{i=1}^N$ to the filtered density $\hat{p}(x_t|y_{1:t}, \theta)$. Details of the auxiliary SIR filter are summarised in Algorithm 5.

Algorithm 5 Auxiliary Particle Filter

Step 1: iteration $t = 1$,

Sample particles $\{x_1^{(i)}\}$ from the prior $p(x_1|\theta)$ and $\forall i$ set weights $w_1^{(i)} = g_\theta(y_1|x_1^{(i)})$.

Step 2: iteration $t = 2, \ldots, T$. Assume a set of particles $\{x_{t-1}^{(i)}\}_{i=1}^N$ and associated weights $\{w_{t-1}^{(i)}\}_{i=1}^N$ that approximate $p(x_{t-1}|y_{1:t-1}, \theta)$ and user-defined set of proposal weights $\{\xi_t^{(i)}\}_{i=1}^N$ and family of proposal densities $q(\cdot|x_{t-1}, y_t, \theta)$.

(a) Sample indices $\{k_1, k_2, \ldots, k_N\}$ from $\{1, \ldots, N\}$ with probabilities $\xi_t^{(i)}$.

(b) Propagate particles $x_t^{(i)} \sim q(\cdot|x_{t-1}^{(k_i)}, y_t, \theta)$.

(c) Weight each particle $w_t^{(i)} \propto \frac{w_{t-1}^{(k_i)} g_\theta(y_t|x_t^{(i)}) f_\theta(x_t^{(i)}|x_{t-1}^{(k_i)})}{\xi_t^{(k_i)} q(x_t^{(i)}|x_{t-1}^{(k_i)}, y_t, \theta)}$ and normalise the weights such that $\sum_{i=1}^N w_t^{(i)} = 1$.

One of the nice features of the auxiliary filter is that it can be viewed as a generalisation of the SIR filter, where the SIR filter can be retrieved by setting $q(x_t|x_{t-1}^{(i)}, y_t, \theta) = f_\theta(x_t|x_{t-1}^{(i)})$ and $\xi_t^{(i)} = w_{t-1}^{(i)}$. For conditional linear-Gaussian state space models, the optimal proposal is $q(x_t|x_{t-1}^{(k_i)}, y_t, \theta) = p(x_t|x_{t-1}^{(i)}, y_t, \theta)$ with weights $\xi_t^{(i)} \propto w_{t-1}^{(i)} p(y_t|x_{t-1}^{(i)}, \theta)$, which will result in all $w_t^{(i)}$ being equal to $1/N$. If the observation likelihood $g_\theta(y_t|x_t)$ is log-concave then the proposal can be an approximation to the optimal density obtained by using a Taylor series expansion (see Pitt and Shephard (1999) for details) to give an approximately optimal filter with a fairly even set of weights.

For more general state space models, where it is difficult to approximate the optimal proposal, it is suggested that samples be drawn from the transition density $q(x_t|x_{t-1}^{(i)}, y_t, \theta) = f_\theta(x_t|x_{t-1}^{(i)})$ with weights $\xi_t^{(i)} \propto w_{t-1}^{(i)} p(y_t|\mu_t^{(i)}, \theta)$, where $\mu_t^{(i)}$ is the mean, mode or some possible value of the density $x_t|x_{t-1}^{(i)}$. This then simplifies the

weights from (3.4.4) to

$$w_t^{(i)} \propto \frac{w_{t-1}^{(k_i)} g_\theta(y_t|x_t^{(i)}) f_\theta(x_t^{(i)}|x_{t-1}^{(k_i)})}{\xi_t^{(k_i)} q(x_t^{(i)}|x_{t-1}^{(k_i)}, y_t, \theta)} = \frac{g_\theta(y_t|x_t^{(i)})}{p(y_t|\mu_t^{(i)}, \theta)},$$

which appears to be similar to the weights for the SIR filter. Compared to SIR filter, the weights of the auxiliary filter are generally less variable due to first sampling indices k_i with weights proportional to $\xi_t^{(i)}$, then sampling the state $x_t^{(i)}$. As the weights $\xi_t^{(i)}$ now take account of the predictive information from $p(y_t|\mu_t^{(i)}, \theta)$, the sampled particles $x_t^{(i)}$ will be much closer to the truth if the observations are highly informative (Johansen and Doucet, 2008).

3.5 Parameter estimation

So far we have considered the problem of inferring the latent state x_t conditional on the observations $y_{1:t}$, where the model parameters θ are treated as known. In practice this is often not the case and parameters are estimated as well as the latent state. Numerous approaches to parameter estimation for state space models have been proposed, which broadly speaking can be categorised as *online* or *offline* methods, where inference is approached from either a *Bayesian* or *maximum likelihood* perspective. This section will cover some of the literature in this area and shall be built upon in subsequent chapters of the thesis.

3.5.1 Maximum likelihood estimation

Maximum likelihood based parameter estimation for state space models aims to find the estimate $\hat{\theta}$ which maximises the marginal likelihood,

$$\hat{\theta} = \arg\max_{\theta \in \Theta} p(y_{1:T}|\theta).$$

The likelihood function can be decomposed as a product of predictive likelihoods,

$$p(y_{1:T}|\theta) = \prod_{t=1}^{T} p(y_t|y_{1:t-1}, \theta),$$

where the latent state is integrated out

$$p(y_t|y_{1:t-1}, \theta) = \int \left(g_\theta(y_t|x_t) \int f_\theta(x_t|x_{t-1}) p(x_{t-1}|y_{1:t-1}, \theta) dx_{t-1} \right) dx_t.$$

In practice, for reasons of numerical stability, we instead maximise the log-likelihood function $\log p(y_{1:T}|\theta)$, which is a straightforward replacement of the product of predictive likelihoods given above with the sum of predictive log-likelihoods.

Aside from discrete state space or linear-Gaussian state space models, as considered by the Kalman filter, the likelihood function is unavailable in closed form. Using a particle filter it is possible to create pointwise particle approximations of the likelihood function for a given θ. Assuming an auxiliary particle filter, an approximation of the predictive likelihood is given by the particle weights (3.4.5)

$$\hat{p}(y_t|y_{1:t-1}, \theta) = \sum_{i=1}^{N} w_t^{(i)}/N,$$

which are available for free from the particle filter and require no extra computation to obtain.

Theoretical results have shown that the particle approximation to the likelihood is an unbiased, consistent estimator (Proposition 9.4.1 Del Moral (2004) and Theorem 1 Pitt et al. (2012)) and a central limit theorem for this estimator exists

$$\sqrt{N}[\hat{p}(y_{1:T}|\theta) - p(y_{1:T}|\theta)] \sim \mathcal{N}(0, \lambda^2(\theta)),$$

see Proposition 9.4.1 Del Moral (2004) for the asymptotic variance $\lambda^2(\theta)$.

On the other hand, the log-likelihood estimator $\hat{l}(\theta)$ is obtained from a nonlinear transformation and is biased. A bias correction based on a second order Taylor series expansion was proposed by Andrieu et al. (2004). A central limit theorem for the log-likelihood estimator carries over by the second order delta method

$$\sqrt{N}[\log \hat{p}(y_{1:T}|\theta) - \log p(y_{1:T}|\theta)] \sim \mathcal{N}\left(\frac{-\gamma^2(\theta)}{2\sqrt{N}}, \gamma^2(\theta)\right),$$

where $\gamma^2(\theta) = \lambda^2(\theta)/p(y_{1:T}|\theta)^2$.

Maximising the likelihood is difficult as the particle filter approximation only allows us to evaluate the function pointwise for a given θ. For problems where Θ is discrete, or continuous but can easily be discretised, then maximising the likelihood can be viewed as a maximisation problem over a grid of points. This approach, however, does not scale well for large dimensional θ and can only be applied to a limited number of problems. Alternative approaches to this problem, which shall be considered in this thesis, avoid estimating the likelihood directly and instead maximise it indirectly.

Gradient ascent algorithm

Due to the intractability of the likelihood we cannot find the maximum likelihood estimate via direct maximisation of the function. An alternative approach is to instead calculate the *score function*, the gradient of the log-likelihood function, $\nabla \log p(y_{1:T}|\theta)$ with respect to θ. Using the score function it is then possible to maximise the log-likelihood function indirectly with the following iterative procedure. At iteration j we have some vector of parameters θ_j which we update at iteration $j+1$ as

$$\theta_{j+1} = \theta_j + \gamma_j \nabla \log p(y_{1:T}|\theta),$$

where $\{\gamma_j\}$ is a sequence of decreasing step-size parameters that are chosen such that they satisfy the conditions $\sum_j \gamma_j = \infty$ and $\sum_j \gamma_j^2 \leq \infty$, which ensures the convergence of the algorithm (Robbins and Monro, 1951). One such choice which satisfies these conditions is $\gamma_j = j^{-\alpha}$, where $0.5 \leq \alpha < 1$.

The gradient procedure given above is a batch method where the parameter is updated after observing $y_{1:T}$. Alternatively, the parameters can be updated recursively (LeGland and Mevel, 1997) and therefore online as new observations are received. At time t we have an estimate θ_t for the parameter based on observations $y_{1:t}$, $t \leq T$. The parameter is then updated as

$$\theta_{t+1} = \theta_t + \gamma_t \nabla \log p(y_t|y_{1:t-1}, \theta),$$

once the newest observation y_t is received.

Except for a few special cases it is not possible to calculate the score function in closed form. If we assume that regularity conditions allowing the change of order of

differentiation and integration hold, then as with the likelihood function, it is possible create particle approximations of the score function using *Fisher's identity* (Cappé et al., 2005)

$$\nabla \log p(y_{1:T}|\theta) = \int \nabla \log p(x_{1:T}, y_{1:T}|\theta) p(x_{1:T}|y_{1:T}, \theta) dx_{1:T},$$

which is the expectation of the derivative of the complete log-likelihood with respect to the posterior of the entire latent process.

An important part of this thesis is focused on creating efficient particle approximations of the score function. Chapter 4 provides a detailed review of the previous approaches to this problem from the literature and presents an improved method for estimating the score function. To avoid repetition, the interested reader is referred to Chapter 4 for further details on gradient based maximum likelihood parameter estimation.

Expectation-Maximisation algorithm

The expectation-maximisation algorithm, also known as the EM algorithm, was introduced by Dempster et al. (1977) and is a popular algorithm for general maximum likelihood estimation. The EM algorithm is a two step iterative procedure. At iteration $j+1$ of the algorithm we assume there is a parameter estimate θ_j from the previous iteration. The first step is the expectation (E) step, which in the context of state space modelling is taken over the posterior of the latent states,

$$Q(\theta_j, \theta) = \int \log p(x_{1:T}, y_{1:T}|\theta) p(x_{1:T}|y_{1:T}, \theta_j) dx_{1:T} = \mathbb{E}[\log p(X_{1:T}, y_{1:T}|\theta)|y_{1:T}, \theta_j],$$

where by introducing the notation $f_\theta(x_1|x_0) = \mu_\theta(x_1)$ the joint log-likelihood of the complete data now has the convenient additive form

$$\log p(x_{1:T}, y_{1:T}|\theta) = \sum_{t=2}^{T} \{\log f_\theta(x_t|x_{t-1}) + \log g_\theta(y_t|x_t)\}.$$

The second step is the maximisation (M) step which updates the parameters θ_j

$$\theta_{j+1} = \arg\max_{\theta\in\Theta} Q(\theta_j, \theta).$$

The EM algorithm applies the two steps iteratively until some stopping criterion is met, whereby there is no change in θ_j for subsequent iterations. At the end of the algorithm we have a sequence $\{p(y_{1:T}|\theta_j)\}_{j\geq 1}$ of non-decreasing likelihood values.

In the case where $p(x_{1:T}, y_{1:T}|\theta)$ is in the exponential family it depends on $(x_{1:T}, y_{1:T})$ via a set of finite dimensional sufficient statistics. We can then define functions $s_l : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, $l = 1, \ldots, m$, where m is the dimension of the sufficient statistics, such that the expectation and maximisation steps simplify to calculating the E-step as

$$S_l(x_{1:T}) = \sum_{t=1}^{T} s_l(x_{t-1}, x_t) \quad \text{and} \quad \mathcal{S}_{l,\theta_j} = \int S_l(x_{1:T}) p(x_{1:T}|y_{1:T}, \theta_j) dx_{1:T},$$

where $S_l(x_{1:T})$ are typically referred to as *sufficient statistics* of $(x_{1:T}, y_{1:T})$. In the case where both $f_\theta(x_t|x_{t-1})$ and $g_\theta(y_t|x_t)$ are in the exponential family there exists a set of sufficient statistics $s_l(x_{t-1}, x_t)$ such that

$$\log f_\theta(x_t|x_{t-1}) + \log g_\theta(y_t|x_t) = \sum_{t=2}^{T} \{h_l(\theta) s_l(x_{t-1}, x_t) + b_l(\theta)\},$$

where $h_l(\cdot)$ and $b_l(\cdot)$ are vector and real-valued functions respectively on Θ. The expectation step of the EM algorithm is then

$$Q(\theta_j, \theta) = \sum_{t=2}^{T} \left\{ h_l(\theta) \left[\int s_l(x_{t-1}, x_t) p(x_{1:T}|y_{1:T}, \theta) dx_{1:T} \right] + b_l(\theta) \right\},$$

which depends only on the sufficient statistics. Furthermore, for the maximisation step (M-step), there exists a function $\Lambda : \mathbb{R}^m \to \Theta$ which finds the maximising argument of $Q(\theta_j, \theta)$,

$$\theta_{j+1} = \arg\max_{\theta \in \Theta} Q(\theta_j, \theta) = \Lambda(\mathcal{S}_{1,\theta_j}, \ldots, \mathcal{S}_{m,\theta_j}).$$

For general state space models $Q(\theta_j, \theta)$ cannot be computed analytically, but can be replaced by a direct particle approximation (Andrieu et al., 2004). Alternative particle approximations based on particle smoothers (Olsson et al., 2008; Schon et al., 2011; Fearnhead et al., 2010) can be applied to reduce the Monte Carlo variance of the estimator. As $Q(\theta_j, \theta)$ is now computed numerically, the algorithm no longer guarantees that for each iteration the likelihood function is monotonically increasing. However, for a sufficiently large number of particles, this algorithm has been shown to perform well.

As with the gradient ascent algorithm, the EM algorithm can be applied online (Cappé and Moulines, 2009) to recursively update the parameters once new observations are received. In the online EM algorithm, running averages of the sufficient statistics $\mathcal{S}_{l,\theta}$ are calculated using the stochastic approximation method (see Del Moral et al. (2010), Doucet et al. (2009) and Yildirim et al. (2012) for details). Given that the parameters are updated at each iteration, the sufficient statistics $\mathcal{S}_{l,\theta}$ at time t are calculated using the most recent parameter estimates, $\theta = \theta_{t-1}$. Assume that at time $T-1$ we have a collection of parameter estimates $\{\theta_t\}_{t=1}^{T-1}$ which are estimated sequentially from observations $y_{1:T-1}$. Once the newest observation y_T is received the

sufficient statistics are updated. For each $l = 1, \ldots, m$ calculate,

$$\begin{aligned} \mathcal{S}_{l,\theta_{T-1}} &= \gamma_T \int s_l(x_{T-1}, x_T) p(x_{T-1}, x_T | y_{1:T}, \theta_{T-1}) dx_{T-1:T} \\ &+ (1-\gamma_T) \int \left(\sum_{t=1}^{T-1} s_l(x_{T-1}, x_T) \right) p(x_{1:T-1} | y_{1:T}, \theta_{T-1}) dx_{1:T-1}, \end{aligned}$$

where $\{\gamma_t\}_{t \geq 1}$ is a step size sequence satisfying the same conditions as stated for the gradient ascent algorithm. Note that conditioning on $\theta_{1:T-1}$ in $p(x_{1:T-1}|y_{1:T}, \theta_{T-1})$ indicates that the posterior density is calculated sequentially using parameter θ_{t-1} at time t.

The parameter is then updated at the M-step as

$$\theta_T = \Lambda(\mathcal{S}_{1,\theta_{T-1}}, \ldots, \mathcal{S}_{m,\theta_{T-1}}).$$

Online EM for state space models is implemented using particle approximations for $\mathcal{S}_t$, where for N particles we can proceed using either an $\mathcal{O}(N)$ (Cappé, 2009) or an $\mathcal{O}(N^2)$ (Del Moral et al., 2010) algorithm. The $\mathcal{O}(N)$ algorithm is based on an approximation of the entire path space $p(x_{1:t}|y_{1:t}, \theta)$. Computationally this approach is fast, however, results established by Del Moral et al. (2010), show that even under strong mixing assumptions the path space approach gives an asymptotic variance for $\mathcal{S}_l$ which does not tend to zero when $\gamma_t = t^{-\alpha}$, $0.5 \leq \alpha < 1$. Alternatively, the $\mathcal{O}(N^2)$ algorithm proposed by Del Moral et al. (2010) has an asymptotic variance which tends to zero in time T for all $0.5 \leq \alpha < 1$. However, the drawback of this algorithm is that the improvement in Monte Carlo accuracy carries a higher computational cost than the $\mathcal{O}(N)$. Therefore, for long data sets, implementing this algorithm online may require that only a small number of particles are used.

Iterated filter

An alternative maximum likelihood method, which is a strictly offline approach to parameter estimation, is the iterated filter (Ionides et al., 2011). This filter considers state space models with a latent process on both the states and parameters $\{X_t, \theta_t\}_{t=1}^T$. This approach introduces artificial dynamics to the parameters θ_t, creating a Markov chain on the parameters which follow a random walk with Gaussian noise. Assume that at the start of iteration j we have an estimate for θ, θ^j. An SMC filter is run on $\{X_t, \theta_t\}_{t=1}^T$ with observations $y_{1:T}$, where within the filter the parameters are perturbed by

$$\theta_t|\theta_{t-1} \sim \mathcal{N}(\theta_{t-1}, \sigma_j^2\Sigma), \qquad \theta_1 \sim \mathcal{N}(\theta^j, \tau_j^2\Sigma).$$

The noise parameters σ_j^2 and τ_j^2 produce artificial dynamics on θ_t, where Σ is an arbitrary positive-definite symmetric matrix, typically a diagonal matrix, used to scale the components of θ_t.

At each time step t, the mean and variance of θ_t are recorded with respect to the filtered and prediction densities (Ionides et al., 2006) respectively,

$$m_t = \mathbb{E}[\theta_t|y_{1:t}] \qquad V_t = \text{var}\,[\theta_t|y_{1:t-1}].$$

After running the SMC algorithm we then obtain a new estimate for the parameters θ^{j+1} using the update

$$\theta^{j+1} = \theta^j + \gamma_j \sum_{t=1}^{T} V_t^{-1}(m_t - m_{t-1}).$$

The parameters are updated in a similar fashion to the gradient ascent algorithm and in fact $\sum_{t=1}^T V_t^{-1}(m_t - m_{t-1})$ is an approximation of the gradient $\nabla \log p(y_{1:T}|\theta)$ evaluated at $\theta = \theta^{j-1}$.

The iterated filter can be viewed as a derivative free version of the gradient ascent algorithm with similarities to stochastic approximation methods (Spall, 2000). Convergence of the parameters to a local maximum is ensured if $\sigma_j^2, \tau_j^2 \to 0$ and the number of particles is increased to account for the reduced variance in the noise parameters. One of the main advantages of the iterated filter is that it is only necessary to sample from the transition density $f_\theta(\cdot|\cdot)$, this can be an advantage over the gradient ascent algorithm which requires that it is possible to calculate the derivative of this density. However, this filter can be difficult to tune for high dimensional θ and the requirement of an increasing number of particles for convergence can be an issue.

A second order derivative free extension to this filter has recently been introduced by Doucet et al. (2013). This extension approximates the observed information matrix and uses this estimate within the parameter update step.

3.5.2 Bayesian parameter estimation

In the Bayesian setting we now treat the model parameters θ as random variables and introduce a prior density on the parameters, $p(\theta)$. The posterior of the parameters and latent states $p(x_{1:T}, \theta|y_{1:T})$ can then be evaluated in an offline setting using batches of data $y_{1:T}$. MCMC is a popular approach for sampling from this posterior. However, applying MCMC to the posterior $p(x_{1:T}, \theta|y_{1:T}) = p(x_{1:T}|y_{1:T}, \theta)p(\theta)$ requires that it is possible to sample from $p(x_{1:T}|y_{1:T}, \theta)$ exactly and calculate $p(y_{1:T}|\theta)$ exactly, which, as discussed already, is not possible for general state space models. A solution to this problem is to use SMC approximations for these quantities and is referred to generally as *particle MCMC*.

Alternatively, in an online setting, the sequence of densities $\{p(x_{1:t}, \theta|y_{1:t})\}_{t=1}^{T}$ can be updated recursively as new observations y_t become available. The estimation of these posteriors is performed within an SMC filter, however, estimating this sequence of posteriors can be difficult for large t.

Particle MCMC

Particle MCMC (Andrieu et al., 2010) represents a class of MCMC algorithms where particle approximations are used to create efficient high dimensional proposal distributions. This class of algorithms covers particle versions of the Metropolis Hasting algorithm and the Gibbs sampler. In this section we shall consider the most popular of these methods, the particle marginal Metropolis Hasting (PMMH) algorithm which shall be important in Chapter 5 of this thesis.

Consider the posterior $p(x_{1:T}, \theta|y_{1:T})$ and assume for the moment that we can use the ideal MH sampler. A sensible proposal distribution for the posterior is then

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

where $x'_{1:T}$ is proposed from the posterior for the state given θ' and thus perfectly adapted to θ'. The proposal then requires only choosing a proposal on θ, $q(\theta'|\theta)$.

The Metropolis Hastings acceptance ratio for the resulting sampler is then

$$\min\left\{\frac{p(x'_{1:T}, \theta'|y_{1:T})q((x_{1:T}, \theta)|(x'_{1:T}, \theta'))}{p(x_{1:T}, \theta|y_{1:T})q((x'_{1:T}, \theta')|(x_{1:T}, \theta))}\right\} = \min\left\{\frac{p(y_{1:T}|\theta')p(\theta')q(\theta|\theta')}{p(y_{1:T}|\theta)p(\theta)q(\theta'|\theta)}\right\},$$

where by standard decomposition $p(x_{1:T}, \theta|y_{1:T}) = p(\theta|y_{1:T})p(x_{1:T}|y_{1:T}, \theta)$. The MH ratio illustrates why this is referred to as the marginal MH algorithm as we are targeting the lower dimensional posterior $p(\theta|y_{1:T}) \propto p(y_{1:T}|\theta)p(\theta)$. This is a very

useful property as it is much simpler to target the lower dimensional posterior as opposed to the full posterior. This idea has been considered more generally by Andrieu and Roberts (2009) and Beaumont (2003) and is known as *pseudo-marginal MCMC*.

As stated above, for general state space models we cannot sample from $p(x_{1:T}|y_{1:T}, \theta)$ or evaluate $p(y_{1:T}|\theta)$ exactly. The solution to this problem proposed by the PMMH algorithm is to replace $p(x_{1:T}|y_{1:T}, \theta)$ and $p(y_{1:T}|\theta)$ with particle approximations, where an SMC algorithm is used to sample the path $X_{1:T}$ and estimate the marginal likelihoods $\hat{p}(y_{1:T}|\theta)$ used in the acceptance ratio. The PMMH algorithm is summarised in Algorithm 6.

Algorithm 6 Particle marginal Metropolis Hasting algorithm

Step 1: iteration $i = 1$,

(a) Set θ^1 arbitrarily.

(b) Run the SMC algorithm (see Algorithm 5) targeting $p(x_{1:T}|y_{1:T}, \theta^1)$, sample $X_{1:T}^1 \sim \hat{p}(\cdot|y_{1:T}, \theta^1)$ and estimate marginal likelihood $\hat{p}(y_{1:T}|\theta^1)$.

Step 2: for iterations $j = 2, \ldots, M$.

(a) Sample $\theta' \sim q(\cdot|\theta^{j-1})$.

(b) Run the SMC algorithm (see Algorithm 5) targeting $p(x_{1:T}|y_{1:T}, \theta')$, sample $X_{1:T}' \sim \hat{p}(\cdot|y_{1:T}, \theta')$ and estimate marginal likelihood $\hat{p}(y_{1:T}|\theta')$.

(c) With probability

$$\min\left\{\frac{\hat{p}(y_{1:T}|\theta')p(\theta')q(\theta^{j-1}|\theta')}{\hat{p}(y_{1:T}|\theta^{j-1})p(\theta^{j-1})q(\theta'|\theta^{j-1})}\right\}$$

set $\theta^j = \theta', X_{1:T}^j = X_{1:T}'$ and $\hat{p}(y_{1:T}|\theta^j) = \hat{p}(y_{1:T}|\theta')$. Otherwise set $\theta^j = \theta^{j-1}, X_{1:T}^j = X_{1:T}^{j-1}$ and $\hat{p}(y_{1:T}|\theta^j) = \hat{p}(y_{1:T}|\theta^{j-1})$.

One of the remarkable features of the PMMH algorithm is that the sampler targets the invariant density $p(x_{1:T}, \theta|y_{1:T})$ regardless of the number of particles used in the SMC algorithm. However, the efficiency of the sampler will be affected by the number of particles, and increasing the number of particles will reduce the variance in the marginal likelihood estimate and improve the mixing of the sampler. The variance of the marginal likelihood is $O(T/N)$ and therefore, for a reasonable level of performance, $N = O(T)$ particles should be used in order to keep the variance constant as T increases (Doucet et al., 2012).

Compared to the standard MCMC approach for sampling from $p(x_{1:T}, \theta|y_{1:T})$, the PMMH sampler is highly efficient with minimal tuning. Rather than needing to tune complicated high dimensional proposal distributions, the PMMH sampler allows users to run an SMC algorithm, such as the simple bootstrap filter, to sample the latent states $X_{1:T}$. This reduces the problem of sampling from the full posterior to the simpler problem of tuning a lower dimensional proposal on θ. The one drawback of this algorithm is that it cannot be implemented online to update the posterior sequentially as new observations are obtained. This may reduce the applicability of this algorithm for large data sets where each run of the SMC algorithm can be costly.

Recently, the SMC^2 algorithm has been introduced by Chopin et al. (2012). This algorithm used nested SMC filters to explore the sequence of posteriors $\{p(x_{1:t}, \theta|y_{1:t})\}_{t\geq 1}$ as new observations are made available. The algorithm can be viewed as an extension to the iterated batch importance sampler (IBIS) (Chopin, 2002) and PMCMC. However, while this algorithm can be applied recursively, it is not an online algorithm as the associated cost of MCMC steps increase the computational cost with each

iteration.

MCMC within SMC

Online Bayesian parameter estimation aims to estimate the sequence of densities $\{p(x_{1:t}, \theta|y_{1:t})\}_{t=1}^{T}$ using a particle filter. The simplest approach to this problem is to estimate θ in the same manner as estimating $X_{1:t}$ and simply augment the state space to include the parameter vector, $Z_t = (X_t, \theta_t)$. Given a prior $p(\theta)$ on the parameters and transition density $f_\theta(x_t|x_{t-1})\delta_{\theta_{t-1}}(\theta_t)$ which ensures that $\theta_t = \theta_{t-1}$, the particle filter can be applied to estimate Z_t. The problem with this approach for parameter estimation is that N particles are sampled from the prior $\{\theta_1^{(i)}\}_{i=1}^{N}$ at initialisation, but as the transition density does not allow θ to evolve, the number of distinct particles for θ_t decays over time. This is because the filter uses resampling with replacement to reduce the degeneracy of the filter, which at each iteration reduces the number of unique particles and thus impoverishes the sample. Eventually all the particles will be equal, resulting in a single point mass representation of the parameters.

Gilks and Berzuini (2001) introduced the *resample-move* algorithm as a solution to the problem of sample impoverishment. This filter introduces an MCMC kernel K_t to the particle filter which admits $p(x_{1:t}, \theta|y_{1:t})$ as its invariant density, such that

$$p(x'_{1:t}, \theta'|y_{1:t}) = \int p(x_{1:t}, \theta|y_{1:t}) K_t(x'_{1:t}, \theta'|x_{1:t}, \theta) dx_{1:t} d\theta.$$

An important property of this kernel is that unlike most MCMC kernels it does not need to be ergodic. In practice the kernel is generally not ergodic as this would require the sampling of an increasing number of variables as t increases. Instead we tend to

consider kernels which sample $(X_{t-\Delta:t}, \theta_t)$ rather than $(X_{1:t}, \theta_t)$, where $\Delta \geq 1$ is some chosen lag.

For some state space models Fearnhead (2002) and Storvik (2002) showed that it is possible to reduce the memory requirements of the MCMC scheme by summarising the information about the parameters and states via low-dimensional *sufficient statistics* s_t. All of the information about the states and observations is contained within s_t so that $p(\theta|x_{1:t}, y_{1:t}) = p(\theta|s_t)$. Assuming that these sufficient statistics and be updated recursively, that is there exists a function $\mathcal{S}(\cdot)$ such that $s_t = \mathcal{S}(s_{t-1}, x_t, y_t)$, then rewriting the posterior as

$$p(x_{1:t}, \theta|y_{1:t}) \propto g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1}) p(\theta|s_{t-1}) p(x_{1:t-1}|y_{1:t-1}, \theta),$$

the parameters are now simulated from $p(\theta|s_{t-1})$ rather than $p(\theta|x_{1:t-1}, y_{1:t-1})$. Carvalho et al. (2010) have applied the sufficient statistics approach within an auxiliary particle filter framework, which they refer to as *particle learning*, and shown that using the auxiliary particle filter can yield further improvements.

The proposed MCMC methods can, to some extent, alleviate the path degeneracy problem. However, as shown by Andrieu et al. (2005), for large t it becomes increasingly difficult to adequately approximate the density of the whole path $p(x_{1:t}|y_{1:t}, \theta)$ with only a finite set of particles as s_t is a function of $x_{1:t}$. This problem also carries over to the sufficient statistics where errors will accumulate over time. This does not mean that this approach is entirely without merit as for shorter datasets, or models which are piecewise time-varying (as considered in Chapter 6), the accumulated error is not sufficiently great to hamper the parameter estimation process.

Artificial dynamics

The degeneracy problem exhibited by the parameter particles is caused by the fact that the parameters are fixed. Therefore, after resampling with replacement, some of the particles are duplicated thus reducing the number of distinct particles. An alternative approach to MCMC to alleviate the degeneracy problem is to introduce artificial dynamics to the parameters by introducing noise. The simplest approach is to add Gaussian noise and thus creating a random walk on the parameters,

$$\theta_t^{(i)} = \theta_{t-1}^{(i)} + \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, \sigma^2).$$

The problem with this approach is that the noise introduced to the parameters accumulates causing an overdispersion in the approximation of the posterior parameter density. Liu and West (2001) proposed a correction to this approach where the parameters are updated as

$$\theta_t^{(i)} = \lambda\theta_{t-1}^{(i)} + (1-\lambda)\overline{\theta}_{t-1} + \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, h^2 V_{t-1}),$$

where $\overline{\theta}_{t-1}$ and V_{t-1} are the empirical mean and variance of $\{\theta_{t-1}^{(i)}\}_{i=1}^N$.

This technique is based on the kernel density estimation method of West (1993), where λ and h^2 are the shrinkage and smoothing parameters which are chosen such that $\lambda^2 + h^2 = 1$. If the shrinkage term is not included then the variance of the approximation will accrue over time, $V_t = (1 + h^2)V_{t-1}$. The shrinkage parameter corrects for the overdispersion and preserves the mean and variance of the approximation. Overall, the parameter posterior is approximated by a mixture of Gaussians, which can be useful for dealing with multi-modal posteriors. It is also worth noting that this method is applied for online parameter estimation.

3.5.3 Discussion of parameter estimation methods

Parameter estimation for state space models is still an open research problem, as each of the methods proposed above have their drawbacks. From the proposed maximum likelihood approaches, the gradient ascent algorithm can be advantageous compared to the EM when it is possible to calculate the hessian Γ_t of $\log p(y_{1:T}|\theta)$ (see Poyiadjis et al. (2011) for details). In this setting the step size γ_t is replaced by $-\gamma_t\Gamma_t^{-1}$, where the rate of convergence for the parameters is now quadratic and thus faster than the linear convergence rate of the EM algorithm. Also, the gradient method can be applied when the maximisation step in the EM algorithm is not in closed form. On the other hand, for large dimensional θ, it can be difficult to scale the gradient components to attain the optimal convergence and when the maximisation step is in closed form, the EM algorithm can be more stable. One drawback of both approaches is that they can become trapped in local maxima.

Both the gradient and EM algorithm can be applied online which is a particular advantage over the particle MCMC algorithms when the data length is long. However, particle MCMC does allow for a Bayesian approach to parameter estimation where prior information about the parameters can be included. Also, with reasonably chosen proposal distributions it is less susceptible to local maxima than the gradient and EM algorithms.

It is possible to apply Bayesian parameter estimation online by either using MCMC or artificial dynamics to alleviate some of the particle degeneracy associated with estimating densities of increasing dimension. These methods do not completely bypass

the issue of degeneracy and can begin to suffer when applied to long time series data. However, for shorter time series with an informative prior, and a large number of particles, these methods can perform well.

Chapter 4

Particle Approximations of the Score and Observed Information Matrix for Parameter Estimation in State Space Models with Linear Computational Cost

Abstract

Poyiadjis et al. (2011) show how particle methods can be used to estimate both the score and the observed information matrix for state space models. These methods either suffer from a computational cost that is quadratic in the number of particles, or produce estimates whose variance increases quadratically with the amount of

data. This chapter introduces an alternative approach for estimating the score and information matrix, which has a computational cost that is linear in the number of particles. The method is derived using a combination of kernel density estimation to avoid the particle degeneracy that causes the quadratically increasing variance, and Rao-Blackwellisation. Crucially, we show the method is robust to the choice of bandwidth within the kernel density estimation, as it has good asymptotic properties regardless of this choice. Our estimates of the score and observed information matrix can be used within both online and batch procedures for estimating parameters for state space models. Empirical results show improved parameter estimates compared to existing methods at a significantly reduced computational cost.

4.1 Introduction

State space models have become a popular framework with which to model nonlinear time series problems in engineering, econometrics and statistics; see West and Harrison (1997), Cappé et al. (2005) and Durbin and Koopman (2001). State space models assume that there are two stochastic processes: X_t which evolves as a latent Markov process and Y_t, which are partial observations from the time series. The observations are conditionally independent given the latent Markov process X_t. We consider state space models where both the state dynamics and the observation process may depend on unknown parameters, θ.

In an online setting, such as with target tracking, we are interested in recursively

estimating the current state X_t of the latent process from the set of observations to date, $y_{1:t} = \{y_1, y_2, \ldots, y_t\}$. This is known as filtering. If the parameters are known this involves calculating or approximating the conditional density of the latent state given a sequence of observations, $p(x_t|y_{1:t}, \theta)$. In the case where the state and observation models are linear and Gaussian, the conditional filtered distribution can be estimated using a Kalman filter (Kalman, 1960). In the case of nonlinear, non-Gaussian state space models a closed form expression for the conditional filtered distribution is not available. This has led to the development of a class of approximation techniques known as sequential Monte Carlo (SMC) methods, or particle filters. These filters approximate the conditional density $p(x_t|y_{1:t}, \theta)$ with a weighted set of samples (Gordon et al., 1993; Kitagawa, 1996). The sample values are commonly referred to as particles.

A secondary problem is to also estimate the parameters, θ. Estimation of static parameters for state space models is still an open problem which has received a lot of interest over the last decade. Initial approaches to this problem suggested a fully Bayesian approach, where we introduce a prior for θ. Filtering then involves calculating $p(x_t, \theta|y_{1:t})$, which, in theory at least, can be approximating by augmenting the state to include the unknown parameters and using a particle filter. However, while this scheme can be employed online, it quickly leads to particle degeneracy in the approximation for the parameters as the θ component of the augmented state comprises only of particles selected at the initialisation stage (Doucet et al., 2009). One simple solution to this problem is to apply a kernel density approximation to θ, (Liu and West, 2001) where instead of sampling parameters from a finite set of particles,

parameters can now be sampled from a density. However, it is often not clear how to choose the bandwidth in the kernel density approximation, nor how this approximation impacts the accuracy of estimates of the parameters. Particle degeneracy can be partially alleviated by applying Markov chain Monte Carlo (MCMC) updates to θ (see Gilks and Berzuini, 2001; Fearnhead, 2002; Storvik, 2002; Carvalho et al., 2010), but such approaches still struggle when analysing long time series (Andrieu et al., 2005).

Alternatively, SMC methods can be used to perform maximum likelihood estimation of parameters. Whilst SMC techniques can provide pointwise estimates of the likelihood, for a continuous parameter θ it is difficult to determine the maximum likelihood estimate. The two most common solutions to the problem of maximum likelihood estimation in the literature are gradient based methods and the expectation-maximisation (EM) algorithm (Dempster et al., 1977). In this chapter we shall focus on the gradient based approach, however it is worth mentioning that EM algorithms have been applied to parameter estimation problems for state space models (see Fearnhead et al., 2010). Recently Cappé (2011) developed an online version of the EM algorithm for estimating parameters in state space models. The EM algorithm can be numerically more stable and computationally cheaper than gradient based approaches when θ is high dimensional, as it can be difficult to scale the gradients in high dimensions. However, faster rates of convergence can be achieved through gradient based approaches if it is possible to employ a Newton-Raphson gradient ascent scheme (Doucet et al., 2009).

This chapter proposes a gradient ascent approach to estimate the model param-

eters θ of the state space model. This requires the estimation of the score vector $\nabla \log p(y_{1:T}|\theta)$ which can be used to move the parameters in the direction of the gradient of the log-likelihood. Previous work by Poyiadjis et al. (2011), has provided two approaches for estimating the score vector and observed information matrix. The first has a computational complexity that is linear in the number of particles, but it has the drawback that the variance of the estimates increases quadratically through time. The second method manages to produce estimates whose variance increases linearly with time, but at the expense of a computational cost that is quadratic in the number of particles. The increased computational complexity of this algorithm limits its use for online applications.

We propose a new method for estimating the score vector and observed information matrix, which can then be used to find the maximum likelihood estimate of parameters using gradient ascent methods. This method is based on combining ideas from the linear-time algorithm of Poyiadjis et al. (2011) with the kernel density estimation ideas of Liu and West (2001). We are also able to use Rao-Blackwellisation ideas to reduce the Monte Carlo error of our estimates. The result is a linear-time algorithm which has substantially smaller Monte Carlo variance than the linear-time algorithm of Poyiadjis et al. (2011) – with empirical results showing the Monte Carlo variance of the estimate of the score vector often increasing only linearly with time. Furthermore, unlike standard uses of kernel density estimation, we are able to show our method is robust to the choice of bandwidth. For any fixed bandwidth our approach can lead to methods that consistently estimate the parameters as both the number of time-points and the number of particles go to infinity.

Our approach has similarities with the fixed-lag smoother of Dahlin et al. (2014), in terms of using shrinkage to help control the Monte Carlo error of estimates of the score vector. However, the derivation of our approach in terms of Rao-Blackwellisation of a kernel density estimate enables us to correct for this shrinkage when obtaining estimates of the observed information matrix. Empirical results show that these more accurate estimates can lead to an order of magnitude improvement in the rate of convergence when implementing a Newton-Raphson scheme to find the maximum likelihood estimates.

This chapter is structured as follows. Section 5.2 presents the sequential Monte Carlo framework and state space model notation. The SMC framework is extended to estimating the score vector and observed information matrix in Section 4.3 with the approach given by Poyiadjis et al. (2011). Section 4.4 presents the new approach for estimating the score vector and observed information matrix using a kernel density approximation with Rao-Blackwellisation.

We evaluate the new method empirically, showing that it gives more accurate estimates of the score than the methods of Poyiadjis et al. (2011) (Section 4.6). In Section 4.7 the proposed approach is applied to estimate the parameters of an autoregressive plus noise model and a stochastic volatility model. Here we show the new method can produce better parameter estimates than if either of the approaches of Poyiadjis et al. (2011), or the fixed-lag smoother of Kitagawa and Sato (2001), is used to estimate the score function and observed information matrix. In an online setting we show that the parameter estimates are more accurate than using particle learning (Carvalho et al., 2010) when analysing long time-series.

4.2 Inference for state space models

4.2.1 State space models

Consider the general state space model where $\{X_t; 1 \leq t \leq T\}$ represents a latent Markov process that takes values on some measurable space $\mathcal{X} \subseteq \mathbb{R}^{n_x}$. The process is fully characterised by its initial density $p(x_1|\theta) = \mu_\theta(x_1)$ and transition probability density

$$p(x_t|x_{1:t-1}, \theta) = p(x_t|x_{t-1}, \theta) = f_\theta(x_t|x_{t-1}), \tag{4.2.1}$$

where $\theta \in \Theta$ represents a vector of model parameters. For an arbitrary sequence $\{z_i\}$ the notation $z_{i:j}$ corresponds to $(z_i, z_{i+1}, \ldots, z_j)$ for $i \leq j$.

We assume that the process $\{X_t\}$ is not directly observable, but partial observations can be made via a second process $\{Y_t; 1 \leq t \leq T\} \subseteq \mathcal{Y}$. The observations $\{Y_t\}$ are conditionally independent given $\{X_t\}$ and are defined by the probability density

$$p(y_t|y_{1:t-1}, x_{1:t}, \theta) = p(y_t|x_t, \theta) = g_\theta(y_t|x_t). \tag{4.2.2}$$

In the standard Bayesian context the latent process $\{X_{1:T}\}$ is estimated conditional on a sequence of observations $y_{1:T}$, for $T \geq 1$. If the parameter vector θ is known then the conditional distribution $p(x_{1:T}|y_{1:T}, \theta) \propto p(x_{1:T}, y_{1:T}, \theta)$ can be evaluated where

$$p(x_{1:T}, y_{1:T}, \theta) = \mu_\theta(x_1) \prod_{t=2}^{T} f_\theta(x_t|x_{t-1}) \prod_{t=1}^{T} g_\theta(y_t|x_t). \tag{4.2.3}$$

If θ is unknown then it is possible to estimate θ within the Bayesian framework by assigning a prior distribution $p(\theta)$ to θ and then evaluate the joint posterior distribution

$$p(\theta, x_{1:T}|y_{1:T}) \propto p(x_{1:T}|y_{1:T}, \theta)p(\theta).$$

For nonlinear, non-Gaussian state space models it is not possible to evaluate the posterior density $p(\theta, x_{1:T}|y_{1:T})$ in closed form. A popular approach for approximating these densities is to use a sequential Monte Carlo algorithm.

4.2.2 Sequential Monte Carlo algorithm

SMC algorithms allow for the sequential approximation of the conditional density of the latent state given a sequence of observations, $y_{1:t}$, for a fixed θ, which in this section we assume are known model parameters. For simplicity we shall focus on methods aimed at approximating the conditional density for the current state, X_t, but the ideas can be extended to learning about the full path of the process, $X_{1:t}$. Approximations of the density $p(x_t|y_{1:t}, \theta)$ can be calculated recursively by first approximating $p(x_1|y_1, \theta)$, then $p(x_2|y_{1:2}, \theta)$ and so forth. Each conditional density can be approximated by a set of N weighted random samples, called particles, where

$$\hat{p}(x_t|y_{1:t}, \theta) = \sum_{i=1}^{N} w_t^{(i)} \delta_{X_t^{(i)}}(dx_t), \quad \forall i \; w_t^{(i)} \geq 0, \quad \sum_{i=1}^{N} w_t^{(i)} = 1$$

is an approximation for the conditional distribution and $\delta_{x_0}(dx)$ is a Dirac delta mass function located at x_0. The set of particles $\{X_t^{(i)}\}_{i=1}^N$ and their corresponding weights $\{w_t^{(i)}\}_{i=1}^N$ provide an empirical measure that approximates the probability density function $p(x_t|y_{1:t}, \theta)$, where the accuracy of the approximation increases as $N \to \infty$ (Crisan and Doucet, 2002).

To recursively calculate our particle approximations, we use the following filtering recursion,

$$p(x_t|y_{1:t}, \theta) \propto g_\theta(y_t|x_t) \int f_\theta(x_t|x_{t-1}) p(x_{t-1}|y_{1:t-1}, \theta) dx_{t-1}. \tag{4.2.4}$$

If we assume that at time $t-1$ we have a set of particles $\{X_{t-1}^{(i)}\}_{i=1}^N$ and weights $\{w_{t-1}^{(i)}\}_{i=1}^N$ which produce a discrete approximation to $p(x_{t-1}|y_{1:t-1},\theta)$, then we can create a Monte Carlo approximation for (4.2.4) as

$$p(x_t|y_{1:t},\theta) \approx c g_\theta(y_t|x_t) \sum_{i=1}^N w_{t-1}^{(i)} f_\theta(x_t|x_{t-1}^{(i)}), \tag{4.2.5}$$

where c is a normalising constant. Particle approximations as given above can be updated recursively by propagating and updating the particle set using importance sampling techniques. There is now an extensive literature on particle filtering algorithms, see for example, Doucet et al. (2000) and Cappé et al. (2007).

In this chapter the particle approximations of the latent process are created with the auxiliary particle filter of Pitt and Shephard (1999). This filter has a general form, and simpler filters can be derived as special cases (Fearnhead, 2007). The idea is to construct an approximation of

$$c w_{t-1}^{(i)} g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1}^{(i)})$$

using

$$\xi_t^{(i)} q(x_t|x_{t-1}^{(i)}, y_t, \theta)$$

as an importance sampling proposal to produce our weighted particle approximation to (4.2.5). We simulate from the proposal by first choosing a particle at time $t-1$, with particle $x_{t-1}^{(i)}$ being chosen with probability $\xi_t^{(i)}$. We then propagate this to time t by sampling our particle at time t, x_t, from $q(x_t|x_{t-1}^{(i)}, y_t, \theta)$. The weight assigned to our new particle $x_t^{(i)}$ is then

$$\frac{w_{t-1}^{(i)} g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1}^{(i)})}{\xi_t^{(i)} q(x_t|x_{t-1}^{(i)}, y_t, \theta)}.$$

This can be shown to be a valid importance sampling weight by viewing both the proposal and target as densities on the joint distribution of the state at time t and the particle at time $t-1$. Details are summarised in Algorithm 7.

Algorithm 7 Auxiliary Particle Filter

Step 1: iteration $t = 1$,

Sample particles $\{x_1^{(i)}\}$ from the prior $p(x_1|\theta)$ and $\forall i$ set weights $w_1^{(i)} = g_\theta(y_1|x_1^{(i)})$.

Step 2: iteration $t = 2, \ldots, T$. Assume a set of particles $\{x_{t-1}^{(i)}\}_{i=1}^N$ and associated weights $\{w_{t-1}^{(i)}\}_{i=1}^N$ that approximate $p(x_{t-1}|y_{1:t-1}, \theta)$ and user-defined set of proposal weights $\{\xi_t^{(i)}\}_{i=1}^N$ and family of proposal densities $q(\cdot|x_{t-1}, y_t, \theta)$.

(a) Sample indices $\{k_1, k_2, \ldots, k_N\}$ from $\{1, \ldots, N\}$ with probabilities $\xi_t^{(i)}$.

(b) Propagate particles $x_t^{(i)} \sim q(\cdot|x_{t-1}^{(k_i)}, y_t, \theta)$.

(c) Weight each particle $w_t^{(i)} \propto \frac{w_{t-1}^{(k_i)} g_\theta(y_t|x_t^{(i)}) f_\theta(x_t^{(i)}|x_{t-1}^{(k_i)})}{\xi_t^{(k_i)} q(x_t^{(i)}|x_{t-1}^{(k_i)}, y_t, \theta)}$ and normalise the weights such that $\sum_{i=1}^N w_t^{(i)} = 1$.

4.3 Parameter estimation for state space models

4.3.1 Maximum likelihood estimation

The maximum likelihood approach to parameter estimation is based on solving

$$\hat{\theta} = \arg\max_{\theta \in \Theta} \ \log p(y_{1:T}|\theta),$$

where,

$$\log p(y_{1:T}|\theta) = \sum_{t=1}^{T} \log p(y_t|y_{1:t-1}, \theta),$$

and

$$p(y_t|y_{1:t-1},\theta) = \int \left(g_\theta(y_t|x_t) \int f_\theta(x_t|x_{t-1}) p(x_{t-1}|y_{1:t-1},\theta) dx_{t-1} \right) dx_t.$$

Aside from a few simple cases, it is not possible to calculate the log-likelihood in closed form. Pointwise estimates of the log-likelihood can be obtained using SMC approximations for a fixed value θ. If the parameter space Θ is discrete and low dimensional, then it is relatively straightforward to find the θ which maximises $\log p(y_{1:T}|\theta)$. For problems where the parameter space is continuous, finding the maximum likelihood estimate (MLE) can be more difficult. One option is to evaluate the likelihood over a grid of θ values. This approach faces difficulties when the model dimension is large whereby optimising over a grid of values becomes computationally inefficient.

The gradient based method for parameter estimation, also known as the steepest ascent algorithm, maximises the log-likelihood function by evaluating the score vector (gradient of the log-likelihood) at the current parameter value and then updating the parameter by moving it in the direction of the gradient. For a given batch of data $y_{1:T}$ the unknown parameter θ can be estimated by choosing an initial estimate θ_0, and then recursively solving

$$\theta_k = \theta_{k-1} + \gamma_k \nabla \log p(y_{1:T}|\theta)|_{\theta=\theta_{k-1}} \tag{4.3.1}$$

until convergence. Here γ_k is a sequence of decreasing step sizes which satisfies the conditions $\sum_k \gamma_k = \infty$ and $\sum_k \gamma_k^2 < \infty$. One common choice is $\gamma_k = k^{-\alpha}$, where $0.5 < \alpha < 1$. These conditions on γ_k are necessary to ensure convergence to a value $\hat{\theta}$ for which $\nabla \log p(y_{1:T}|\hat{\theta}) = 0$. A key ingredient to good statistical properties of the resulting estimator of θ, such as consistency (Crowder, 1986), is that if the data is

generated from $p(y_{1:T}|\theta^*)$, then

$$E\{\nabla \log p(Y_{1:T}|\theta^*)\} = \int p(y_{1:T}|\theta^*)\nabla \log p(y_{1:T}|\theta^*)\mathrm{d}y_{1:T} = 0.$$

That is, the expected value of $\nabla \log p(y_{1:T}|\theta)$, with expectation taken with respect to the data, is 0 when θ is the true parameter value.

The rate of convergence of (4.3.1) can be improved if we are able to calculate the observed information matrix, which provides a measure of the curvature of the log-likelihood. When this is possible the Newton-Raphson method can be used and the step size parameter γ_k is replaced with $-\gamma_k\{\nabla^2 \log p(y_{1:T}|\theta)\}^{-1}$.

4.3.2 Estimation of the score vector and observed information matrix

Calculating the score vector and observed information matrix for state space models can be done analytically for linear-Gaussian models (Koopman and Shephard, 1992). In the general setting where the state space is nonlinear and non-Gaussian, it is impossible to derive the score vector and observed information exactly. In such cases sequential Monte Carlo methods can be used to produce particle approximations in their place (Poyiadjis et al., 2011).

If we assume that it is possible to obtain a particle approximation of the latent process $p(x_{1:T}|y_{1:T}, \theta)$, then this approximation can be used to estimate the score vector $\nabla \log p(y_{1:T}|\theta)$ using Fisher's identity (Cappé et al., 2005)

$$\nabla \log p(y_{1:T}|\theta) = \int \nabla \log p(x_{1:T}, y_{1:T}|\theta)p(x_{1:T}|y_{1:T}, \theta)dx_{1:T}. \tag{4.3.2}$$

A similar identity for the observed information matrix is given by Louis (1982)

$$-\nabla^2 \log p(y_{1:T}|\theta) = \nabla \log p(y_{1:T}|\theta) \nabla \log p(y_{1:T}|\theta)^\top - \frac{\nabla^2 p(y_{1:T}|\theta)}{p(y_{1:T}|\theta)}, \tag{4.3.3}$$

where,

$$\begin{aligned} \frac{\nabla^2 p(y_{1:T}|\theta)}{p(y_{1:T}|\theta)} &= \int \nabla \log p(x_{1:T}, y_{1:T}|\theta) \nabla \log p(x_{1:T}, y_{1:T}|\theta)^\top p(x_{1:T}|y_{1:T}, \theta) dx_{1:T} \\ &+ \int \nabla^2 \log p(x_{1:T}, y_{1:T}|\theta) p(x_{1:T}|y_{1:T}, \theta) dx_{1:T}. \end{aligned}$$

See Cappé et al. (2005) for further details of both identities.

Using Fisher's and Louis's identities it is possible to produce estimates for the score vector and observed information matrix from the first and second derivatives of the complete log-likelihood $\log p(x_{1:T}, y_{1:T}|\theta)$. This is straightforward to calculate if we assume that the conditional densities (4.2.1) and (4.2.2) are twice continuously differentiable, then from the joint density (4.2.3) we get

$$\nabla \log p(x_{1:T}, y_{1:T}|\theta) = \nabla \log \mu_\theta(x_1) + \sum_{t=1}^{T} \nabla \log g_\theta(y_t|x_t) + \sum_{t=2}^{T} \nabla \log f_\theta(x_t|x_{t-1}).$$

If we introduce the notation $f_\theta(x_1|x_0) = \mu_\theta(x_1)$, we can write this in the simpler form

$$\nabla \log p(x_{1:T}, y_{1:T}|\theta) = \sum_{t=1}^{T} \left\{ \nabla \log g_\theta(y_t|x_t) + \nabla \log f_\theta(x_t|x_{t-1}) \right\}.$$

Similarly we have

$$\nabla^2 \log p(x_{1:T}, y_{1:T}|\theta) = \sum_{t=1}^{T} \left\{ \nabla^2 \log g_\theta(y_t|x_t) + \nabla^2 \log f_\theta(x_t|x_{t-1}) \right\}.$$

Poyiadjis et al. (2011) give an SMC algorithm that provides estimates of the score vector and observed information matrix using identities (4.3.2) and (4.3.3). The procedure is summarised in Algorithm 8 where the vector of the score for the

complete and marginal log-likelihoods are denoted as $\alpha_t^{(i)} = \nabla \log p(x_{1:t}^{(i)}, y_{1:t}|\theta)$ and $S_t = \nabla \log p(y_{1:t}|\theta)$, respectively. The matrices for the observed information are given as $\beta_t^{(i)} = \nabla^2 \log p(x_{1:t}^{(i)}, y_{1:t}|\theta)$ and $I_t = -\nabla^2 \log p(y_{1:t}|\theta)$.

Algorithm 8 Particle approximation of the Score and Observed Information Matrix

Initialise: set $\alpha_0^{(i)} = 0$ and $\beta_0^{(i)} = 0$ for $i = 1 \ldots, N$.

Step 1: at iteration $t = 1, \ldots, T$.

Apply Algorithm 7 to obtain $\{x_t^{(i)}\}_{i=1}^N$, $\{k_i\}_{i=1}^N$ and $\{w_t^{(i)}\}_{i=1}^N$

Step 2: Update the estimates for α_t and β_t

$$\begin{aligned} \alpha_t^{(i)} &= \alpha_{t-1}^{(k_i)} + \nabla \log g_\theta(y_t|x_t^{(i)}) + \nabla \log f_\theta(x_t^{(i)}|x_{t-1}^{(k_i)}) \\ \beta_t^{(i)} &= \beta_{t-1}^{(k_i)} + \nabla^2 \log g_\theta(y_t|x_t^{(i)}) + \nabla^2 \log f_\theta(x_t^{(i)}|x_{t-1}^{(k_i)}) \end{aligned}$$

(b) Calculate the score vector and observed information matrix

$$S_t = \sum_{i=1}^N w_t^{(i)} \alpha_t^{(i)} \quad \text{and} \quad I_t = S_t S_t^\top - \sum_{i=1}^N w_t^{(i)} (\alpha_t^{(i)} {\alpha_t^{(i)}}^\top + \beta_t^{(i)})$$

4.3.3 Particle degeneracy

Estimation of the score vector and observed information given in Algorithm 8 does not require that we store the entire path of the latent process $\{X_{1:T}^{(i)}\}_{i=1}^N$. However, the values $\alpha_t^{(i)}$ and $\beta_t^{(i)}$ that are stored for each particle depend on the complete path-history of the associated particle. Particle approximations of this form are known to be poor due to inherent particle degeneracy over time (Andrieu et al., 2005). Poyiadjis et al. (2011) prove that the asymptotic variance of the estimates of the score vector and

observed information matrix provided by Algorithm 8 increases at least quadratically with time.

As a result Poyiadjis et al. (2011) introduce an alternative algorithm whose computational cost is quadratic in the number of particles, but which has better Monte Carlo properties. Del Moral et al. (2011) show that this alternative approach, under standard mixing assumptions, produces estimates of the score and observed information whose asymptotic variance only increases linearly with time. Details of this algorithm are omitted for brevity, for further details see Poyiadjis et al. (2011).

4.4 A new approach to estimating the score vector and observed information matrix

4.4.1 Kernel density methods to overcome particle degeneracy

Consider the score vector $\nabla \log p(y_{1:t}|\theta)$. For particle $x_t^{(i)}$, let $x_{1:t}^{(i)}$ denote the path associated with that particle (which exists, even if, as in Algorithm 8, it is not stored). At time t particle i stores a value $\alpha_t^{(i)} = \nabla \log p(x_{1:t}^{(i)}, y_{1:t}|\theta)$, which depends on the history of the particle, $x_{1:t}^{(i)}$. The quadratically increasing variance of the estimate of the score vector which is observed in Algorithm 8 can be attributed to the standard problem of particle degeneracy in particle filters when approximating the conditional distribution of the complete path of the latent state $p(x_{1:t}|y_{1:t})$ (Doucet and Johansen, 2011).

A similar issue of particle degeneracy occurs in particle filter methods that directly approximate the parameters of the model by augmenting the state vector to include the parameters. One approach to reduce this degeneracy is to use kernel density methods. Our approach is to use the same ideas, and in particular the approach of Liu and West (2001), but applied to the $\alpha_t^{(i)}$s and the $\beta_t^{(i)}s$.

For simplicity we will describe the approach as it is applied to the $\alpha_t^{(i)}$s. The idea of Liu and West (2001) is to combine shrinkage of the $\alpha_t^{(i)}$s towards their mean, together with adding noise. The latter is necessary for overcoming particle degeneracy, but the former is required to avoid the increasing variance of the $\alpha_t^{(i)}$s. In practice this approach is arranged so that the combined effect is to maintain the same mean and variance of the $\alpha_t^{(i)}$s.

Consider iteration t of a particle filter, which results in particles $x_t^{(i)}$ with associated weights $w_t^{(i)}$. As in Algorithm 8, assume that particle i is descended from particle k_i at time $t-1$. Currently we calculate

$$\alpha_t^{(i)} = \alpha_{t-1}^{(k_i)} + \nabla \log g_\theta(y_t|x_t^{(i)}) + \nabla \log f_\theta(x_t^{(i)}|x_{t-1}^{(k_i)}). \tag{4.4.1}$$

The idea of Liu and West (2001) is to replace $\alpha_{t-1}^{(k_i)}$ by a draw from a kernel. Note from Algorithm 8 that $S_{t-1} = \sum_{i=1}^N w_{t-1}^{(i)} \alpha_{t-1}^{(i)}$ is the mean of $\alpha_{t-1}^{(k_i)}$ as k_i is drawn from the discrete distribution with probabilities $\xi_t^{(i)}$. Additionally denote the variance by

$$\Sigma_{t-1}^\alpha = \sum_{i=1}^N w_{t-1}^{(i)} (\alpha_{t-1}^{(i)} - S_{t-1})^\top (\alpha_{t-1}^{(i)} - S_{t-1}).$$

Let $0 < \lambda < 1$ be the shrinkage parameter, which is a fixed constant, and the kernel density bandwidth parameter h^2 to be chosen such that $\lambda^2 + h^2 = 1$. The Liu

and West (2001) scheme is then complete by replacing $\alpha_{t-1}^{(k_i)}$ in (4.4.1) by

$$\lambda\alpha_{t-1}^{(k_i)} + (1-\lambda)S_{t-1} + \epsilon_t^{(i)}, \tag{4.4.2}$$

where $\epsilon_t^{(i)}$ is a realisation of a Gaussian distribution $\mathcal{N}(0, h^2\Sigma_{t-1}^{\alpha})$. Note that the choice of h^2 is used to ensure that the mean and variance of (4.4.2), when considering both k_i and ϵ_t as random, is the same as the mean and variance of $\alpha_{t-1}^{(k_i)}$. A similar approach can be applied for the β_ts.

4.4.2 Rao-Blackwellisation

The stored values of $\alpha_t^{(i)}$ and $\beta_t^{(i)}$ do not have any effect on the dynamics of the state. Furthermore we have a stochastic update for these terms which, when we use the kernel density approach, results in a linear-Gaussian update. This means that we can use the idea of Rao-Blackwellisation (Doucet et al., 2000) to reduce the variance in our estimates of the score vector and observed information matrix. In practice this means replacing values for $\alpha_t^{(i)}$ and $\beta_t^{(i)}$ by appropriate distributions which are sequentially updated. Therefore we do not need to add noise to the approximation at each time step as we do with the standard kernel density approach. Instead we can recursively update the distribution representing $\alpha_t^{(i)}$s and estimate the score vector S_t and observed information matrix I_t from the mean and variance of the distributions representing the $\alpha_t^{(i)}$s and $\beta_t^{(i)}$s.

For $t \geq 2$, assume that at time $t-1$ each $\alpha_{t-1}^{(j)}$ is represented by a Gaussian distribution,

$$\alpha_{t-1}^{(j)} \sim \mathcal{N}(m_{t-1}^{(j)}, h^2 V_{t-1}).$$

Then from (4.4.1) and (4.4.2) we have that

$$\alpha_t^{(i)} \sim \mathcal{N}(m_t^{(i)}, h^2 V_t), \tag{4.4.3}$$

where,

$$m_t^{(i)} = \lambda m_{t-1}^{(k_i)} + (1-\lambda) S_{t-1} + \nabla \log g_\theta(y_t | x_t^{(i)}) + \nabla \log f_\theta(x_t^{(i)} | x_{t-1}^{(k_i)}),$$

and

$$V_t = V_{t-1} + \Sigma_{t-1}^{\alpha} = V_{t-1} + \sum_{i=1}^{N} w_{t-1}^{(i)} (m_{t-1}^{(i)} - S_{t-1})^\top (m_{t-1}^{(i)} - S_{t-1}).$$

Similar ideas apply to the $\beta_t^{(i)}$s.

The estimated score vector at each iteration is a weighted average of the $\alpha_t^{(i)}$s, so we can use the means of these distributions to get the Rao-Blackwellised estimate of the score from Fisher's identity (4.3.2),

$$S_t = \sum_{i=1}^{N} w_t^{(i)} m_t^{(i)}.$$

If we only want to estimate the score vector, then this shows that we only need to calculate the expected value of the $\alpha_t^{(i)}$s.

Note that to calculate the estimate of the observed information matrix we only need the mean of the $\beta_t^{(i)}$s, together with the mean and variance of the $\alpha_t^{(i)}$s. This is because our estimate of the observed information as given in Algorithm 8 requires

$$\sum_{i=1}^{N} w_t^{(i)} \left\{ \alpha_t^{(i)} \alpha_t^{(i)^\top} + \beta_t^{(i)} \right\}.$$

The Rao-Blackwellised estimate of this quantity replaces this by the expectation with respect to the distributions of $\alpha_t^{(i)}$ and $\beta_t^{(i)}$. From (4.4.3) we have

$$E\{\alpha_t^{(i)} \alpha_t^{(i)^\top}\} = m_t^{(i)} m_t^{(i)^\top} + h^2 V_t,$$

and if we denote $E\{\beta_t^{(i)}\}$ by $n_t^{(i)}$, then we get the following estimate of the observed information

$$I_t = S_t S_t^\top - \sum_{i=1}^{N} w_t^{(i)} \left\{ m_t^{(i)} m_t^{(i)^\top} + h^2 V_t + n_t^{(i)} \right\}.$$

Note the inclusion of $h^2 V_t$ in this estimate. This term is important as it corrects for the fact that shrinking the values of α_t towards S_t at each iteration will reduce the variability in these values. Without this correction we would overestimate the observed information. Details of this approach are summarised in Algorithm 9.

Algorithm 9 Rao-Blackwellised Kernel Density Approximation of the Score Vector and Observed Information Matrix

Initialise: set $m_0^{(i)} = 0$ and $n_0^{(i)} = 0$ for $i = 1 \ldots, N$, $S_0 = 0$ and $B_0 = 0$.

Step 1: at iteration $t = 1, \ldots, T$.

Apply Algorithm 7 to obtain $\{x_t^{(i)}\}_{i=1}^N$, $\{k_i\}_{i=1}^N$ and $\{w_t^{(i)}\}_{i=1}^N$

Step 2:

(a) Update the mean of the approximations for α_t and β_t

$$\begin{aligned} m_t^{(i)} = &\quad \lambda m_{t-1}^{(k_i)} + (1-\lambda) S_{t-1} + \nabla \log g_\theta(y_t | x_t^{(i)}) + \nabla \log f_\theta(x_t^{(i)} | x_{t-1}^{(k_i)}) \\ n_t^{(i)} = &\quad \lambda n_{t-1}^{(k_i)} + (1-\lambda) B_{t-1} + \nabla^2 \log g_\theta(y_t | x_t^{(i)}) + \nabla^2 \log f_\theta(x_t^{(i)} | x_{t-1}^{(k_i)}) \end{aligned}$$

(b) Update the score vector and observed information matrix

$$S_t = \sum_{i=1}^{N} w_t^{(i)} m_t^{(i)} \quad \text{and} \quad I_t = S_t S_t^\top - \sum_{i=1}^{N} w_t^{(i)} (m_t^{(i)} m_t^{(i)^\top} + n_t^{(i)}) - h^2 V_t$$

where $V_t = V_{t-1} + \sum_{i=1}^N w_{t-1}^{(i)} (m_{t-1}^{(i)} - S_{t-1})^\top (m_{t-1}^{(i)} - S_{t-1})$ and $B_t = \sum_{i=1}^N w_t^{(i)} n_t^{(i)}$.

4.5 Theoretical justification

4.5.1 Monte Carlo accuracy

We have motivated the use of both the kernel density approximation and Rao-Blackwellisation as a means to avoid the impact of particle degeneracy on the $\mathcal{O}(N)$ algorithm for estimating the score vector and observed information matrix. However, what can we say about the resulting algorithm? Here we consider both the Monte Carlo accuracy of the resulting algorithm, and the effect of the approximation error within the kernel density approximation in terms of inferences for the parameters.

It is possible to implement Algorithm 9 so as to store the whole history of the state $x_{1:t}$, rather than just the current value, x_t. This just involves extra storage, with our particles being $x_{1:t}^{(i)} = (x_t^{(i)}, x_{1:t-1}^{(k_i)})$. Whilst unnecessary in practice, thinking about such an algorithm helps with understanding the algorithms properties.

One can fix θ, the parameter value used when running the particle filter algorithm, and the data $y_{1:t}$. For convenience we drop the dependence on θ from notation in the following. The $m_t^{(i)}$ values calculated by the algorithm are just functions of the history of the state and the past estimated score values. We can define a set of functions $\phi_s(x_{1:t})$, for $t \geq s > 0$, such that

$$\phi_s(x_{1:t}) = \nabla \log g_\theta(y_s|x_s) + \nabla \log f_\theta(x_s|x_{s-1}).$$

For $t \geq s > 0$, we can define a set of functions recursively. The function $m_s(x_{1:t})$ depends on $m_{s-1}(x_{1:t})$ and the estimated score functions at previous time-steps, $S_{0:s-1}$,

through

$$m_s(x_{1:t}) = \lambda m_{s-1}(x_{1:t}) + (1-\lambda)S_{s-1} + \phi_s(x_{1:t}), \tag{4.5.1}$$

with $m_0(x_{1:t}) = 0$. We then have that in Algorithm 9, $m_t^{(i)} = m_t(x_{1:t}^{(i)})$, is the value of this function evaluated for the state history associated with the ith particle at time t.

Note that it is possible to iteratively solve the recursion (4.5.1) to get

$$m_s(x_{1:t}) = \sum_{u=1}^{s} \lambda^{s-u}\phi_u(x_{1:t}) + (1-\lambda)\sum_{u=1}^{s} \lambda^{s-u} S_{u-1} \tag{4.5.2}$$

where $0 < \lambda < 1$ is the shrinkage parameter.

If we set $\lambda = 1$, then Algorithm 3 reverts to Algorithm 2. In this case (4.5.2) simplifies to a sum of additive functionals $\phi_u(x_{1:t})$. The poor Monte Carlo properties of Algorithm 2 stem from the fact that the Monte Carlo variance of SMC estimates of $\phi_u(x_{1:t})$ increases at least linearly with $s-u$. And hence the Monte Carlo variance of the SMC estimate of $\sum_{u=1}^{s} \phi_u(x_{1:t})$, increases at least quadratically with s.

In terms of Monte Carlo accuracy of Algorithm 3, the key is that in (4.5.2) we exponentially downweight the contribution of $\phi_u(x_{1:t})$ as $s-u$ increases. Under quite weak assumptions, such as the Monte Carlo variance of the estimate of $\phi_u(x_{1:t})$ being bounded by a polynomial in $s-u$, we will have that the Monte Carlo variance of estimates of $\sum_{u=1}^{s} \lambda^{s-u}\phi_u(x_{1:t})$ will now be bounded in s.

For $\lambda < 1$, we introduce the additional second term in (4.5.2). However estimating this term is less problematic: as the Monte Carlo variance of each S_{u-1} will depend only on u and will not increase as s increases. Empirically we observe that the resulting Monte Carlo variance of our estimates of the score only increases linearly with s for a wide-range of models.

4.5.2 Effect on parameter inference

Now consider the value of S_t in the limit as the number of particles goes to infinity, $N \to \infty$. We assume that standard conditions on the particle filter for the law of large numbers (Chopin, 2004) hold. Then we have that

$$S_t \to E_\theta\{m_t(X_{1:t})|y_{1:t}\} = \int m_t(x_{1:t})p(x_{1:t}|y_{1:t},\theta)\mathrm{d}t.$$

For $t = 1, \ldots, T$, where we fix the data $y_{1:T}$, define $\bar{S}_t = E_\theta\{m_t(X_{1:t})|y_{1:t}\}$ to be the large N limit of the estimate of the score at time t. The following lemma expresses $\bar{S}_t$ in terms of expectations of the $\phi_s(\cdot)$ functions.

Lemma 4.5.1. *Fix $y_{1:T}$. Then $\bar{S}_1 = E_\theta\{\phi_1(X_{1:t}|y_1)\}$ and for $2 \leq t \leq T$*

$$\bar{S}_t = \sum_{u=1}^{t} \lambda^{t-u} E_\theta\{\phi_u(X_{1:t}|y_{1:t})\} + (1-\lambda)\sum_{u=1}^{t-1}\sum_{s=u}^{t-1} \lambda^{s-u} E_\theta\{\phi_u(X_{1:t}|y_{1:s})\},$$

where the expectations are taken with respect to the conditional distribution of $X_{1:t}$ given $y_{1:u}$:

$$E_\theta\{\phi_s(X_{1:t}|y_{1:u})\} = \int \phi_s(x_{1:t})p(x_{1:t}|y_{1:u},\theta)\,dt.$$

The proof of this is given in the Appendix.

We now consider taking expectation of $\bar{S}_T$ with respect to the data. We write $\bar{S}_T(y_{1:T};\theta)$ to denote the dependence on the data $y_{1:T}$ and the choice of parameter θ when implementing the particle filter algorithm. A direct consequence of Lemma 1 is the following theorem.

Theorem 4.5.2. *Let θ^* be the true parameter value, and T a positive integer. Assume regularity conditions exist so that for all $t \leq T$,*

$$E\{\nabla \log p(X_{1:t}, Y_{1:t}|\theta^*)\} = 0, \tag{4.5.3}$$

where expectation is taken with respect to $p(X_{1:T}, Y_{1:T}|\theta^*)$. *Then*

$$E_{\theta^*}\{\bar{S}_T(Y_{1:T};\theta^*)\} = 0,$$

where expectation is taken with respect to $p(Y_{1:T}|\theta^*)$.

Proof. This follows from Lemma 1 by showing that the expectation of each term in $\bar{S}_T$ is 0. Consider the s,uth term for $u \leq s \leq T$. Then this is proportional to $E_{\theta^*}\{\phi_u(X_{1:T}|Y_{1:s})\}$. Taking expectation over $Y_{1:s}$ gives

$$\begin{aligned} & E_{\theta^*}\{E_{\theta^*}[\phi_u(X_{1:T}|Y_{1:s})]\} \\ &= \int \{\nabla \log g_{\theta^*}(y_u|x_u) + \nabla \log f_{\theta^*}(x_u|x_{u-1})\}\, p(x_{1:T}, y_{1:T}|\theta^*)\mathrm{d}x_{1:T}\mathrm{d}y_{1:T} \\ &= \int \{\nabla \log g_{\theta^*}(y_u|x_u) + \nabla \log f_{\theta^*}(x_u|x_{u-1})\}\, p(x_{1:u}, y_{1:u}|\theta^*)\mathrm{d}x_{1:u}\mathrm{d}y_{1:u} \end{aligned}$$

which is equal to 0 as (4.5.3) holds for $t = u$ and $t = u - 1$. □

Algorithm 9, run for parameter θ, gives a Monte Carlo estimate of $\bar{S}_T(y_{1:T};\theta)$. We can then use these estimates within the steepest gradient ascent algorithm (4.3.1) to get estimates for θ. Theorem 4.5.2 shows the robustness of this approach for estimating θ to the choice of λ.

The theorem shows that for any $0 < \lambda < 1$, the expectation of $\bar{S}_T(y_{1:T};\theta^*)$ at the true parameter θ^* is zero, and hence $\bar{S}_T(y_{1:T};\theta) = 0$ are a set of unbiased estimating equations for θ. Using our estimates of the score function within the steepest gradient ascent algorithm is thus using Monte Carlo estimates to approximately solve this set of unbiased estimating equations. The unbiasedness of the estimating equations means that, under standard regularity conditions (e.g. Crowder, 1986), solving the estimating equations will give consistent estimates for θ.

The accuracy of the final estimate of θ will depend both on the amount of Monte Carlo error, and also the accuracy of the estimator based on solving the underlying estimating equation. Note that the statistical efficiency of the estimator obtained by solving $\bar{S}_T(y_{1:T};\theta) = 0$ may be different, and lower, than that of solving $\nabla \log p(y_{1:T}|\theta) = 0$. However in practice we would expect this to be more than compensated by the reduction in Monte Carlo error we get. We investigate this empirically in the following sections.

4.6 Comparison of approaches

In this section we shall evaluate our algorithm, and compare existing approaches for estimating the score vector. We will also investigate how the performance of our method depends on the choice of shrinkage parameter, λ. In order to compare estimates of the score vector with the truth, we consider a first order autoregressive plus noise state space model where it is possible to analytically calculate the score vector using a Kalman filter (Kalman, 1960). Here we give only the score vector estimates however, similar results also hold for estimates of the observed information matrix.

Firstly, consider the first order one-dimensional autoregressive model plus noise AR(1) given by the state space form:

$$\begin{aligned} X_t|X_{t-1} =& x_{t-1} \sim \mathcal{N}(\phi x_{t-1}, \sigma^2), \qquad X_1 \sim \mathcal{N}\left(0, \frac{\sigma^2}{1-\phi^2}\right), \\ Y_t|X_t =& x_t \sim \mathcal{N}(x_t, \tau^2). \end{aligned} \tag{4.6.1}$$

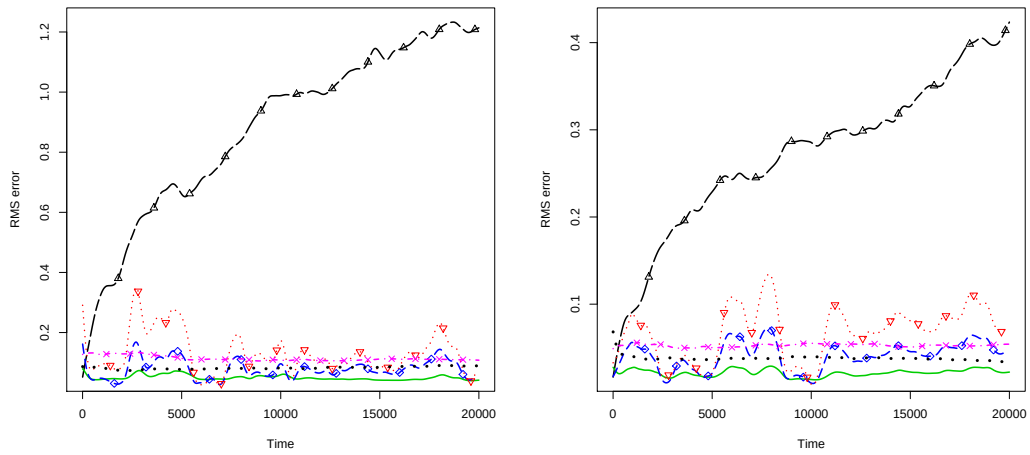
As this model is linear with Gaussian noise the optimal proposal distribution is available in closed form,

$$q(x_t|x_{t-1}^{(i)}, y_t) = \mathcal{N}\left(x_t \middle| \frac{\phi x_{t-1}^{(i)}\tau^2 + y_t\sigma^2}{\sigma^2+\tau^2}, \frac{\sigma^2\tau^2}{\sigma^2+\tau^2}\right),$$

with resample weights,

$$\xi_t^{(i)} \propto w_{t-1}^{(i)}\mathcal{N}(y_t|\phi x_{t-1}^{(i)}, \sigma^2+\tau^2).$$

A comparison of the approaches is performed on a data set simulated from the autoregressive model (4.6.1) with parameters $\theta^* = (\phi, \sigma, \tau)^\top = (0.8, 0.5, 1)^\top$. Figure 4.6.1 displays the accuracy of the score vector estimates, as measured by their root mean squared (RMS) error. We expect the Monte Carlo variance of the estimates to increase at least linearly with time t. Therefore we plot the RMS error scaled by the square root of t (i.e. RMS error$/\sqrt{t}$) to show that that if the variance increases linearly then the RMS error will tend to a constant. We compare our new method, for different values of λ against the $\mathcal{O}(N)$ (Alg. 8) and $\mathcal{O}(N^2)$ algorithms of Poyiadjis et al. (2011). Both our method (Alg. 9) and Algorithm 8 have similar computational costs, and were implemented with $N = 50,000$. The $\mathcal{O}(N^2)$ algorithm of Poyiadjis et al. (2011) is substantially slower, and we implemented this with $N = 500$ and $N = 1,000$. The comparisons were coded in the programming language C and run on a Dell Latitude laptop with a 1.6GHz processor. Using 1,000 observations, each iteration of the $\mathcal{O}(N)$ algorithm takes 1.11 minutes for $N = 50,000$. The $\mathcal{O}(N^2)$ takes 5.1 minutes for $N = 500$ and 21.54 minutes for $N = 1,000$. This corresponds to a CPU cost that is approximately 5 and 20 times greater than the $\mathcal{O}(N)$ methods, respectively.



(a) (b)

Figure 4.6.1: Comparison of RMS error of the score vector estimates, scaled by time, for (a) σ and (b) τ from the autoregressive model using our $\mathcal{O}(N)$ algorithm with $\lambda = 0.95$ (——), $\lambda = 0.85$ (– ◊ – – ◊ –), $\lambda = 0.7$ ($\cdot\nabla\cdot\cdot\nabla\cdot\cdot\nabla\cdot$) and the Poyiadjis et al. (2011) $\mathcal{O}(N)$ algorithm (- △ — - △ —), $\mathcal{O}(N^2)$ with $N = 500$ (— · × — · × —) and $\mathcal{O}(N^2)$ with $N = 1000$ ($\cdot\cdot\cdot\cdot\cdot$).

Firstly, we notice that the new method gives results that are reasonably robust to the choice of λ, with $\lambda = 0.95$ giving the most accurate results. For all three values considered $\lambda = 0.95$, 0.85 and 0.7, we have an RMS error that increases with the square-root of the number of observations, which is consistent with a Monte Carlo variance that is increasing linearly. This is substantially better than the $\mathcal{O}(N)$ algorithm of Poyiadjis et al. (2011), whose RMS error is increasing linearly with the number of observations. The results for our $\mathcal{O}(N)$ algorithm are comparable to those of the $\mathcal{O}(N^2)$ algorithm Poyiadjis et al. (2011). However our method with $\lambda = 0.95$ outperforms the $\mathcal{O}(N^2)$ algorithm, as well as having a significant reduction

in computational time.

4.7 Parameter estimation

4.7.1 Details for online parameter estimation

Our $\mathcal{O}(N)$ algorithm, as described in Section 4.4, produces estimates of the score vector and observed information matrix for a given state space model. These estimates can then be used within the steepest ascent algorithm (4.3.1) to give maximum likelihood estimates of the parameters θ.

The steepest ascent algorithm estimates parameters from batches of data $y_{1:T}$ which can be useful for offline parameter estimation or when dealing with small data sets. Alternatively, we could implement recursive parameter estimation, where estimates of the parameters θ_t are updated as new observations are made available. Ideally this would be achieved by using the gradient of the predictive log-likelihood,

$$\theta_t = \theta_{t-1} + \gamma_t \nabla \log p(y_t|y_{1:t-1}, \theta_t), \tag{4.7.1}$$

where,

$$\nabla \log p(y_t|y_{1:t-1}, \theta_t) = \nabla \log p(y_{1:t}|\theta_t) - \nabla \log p(y_{1:t-1}|\theta_t).$$

However, getting Monte Carlo estimates of $\nabla \log p(y_t|y_{1:t-1}, \theta_t)$ is difficult due to using different values of θ at each iteration of the sequential Monte Carlo algorithm. Thus, following LeGland and Mevel (1997) and Poyiadjis et al. (2011), we make a further approximation, and ignore the fact that θ changes with t. Thus we implement Algorithm 9, but updating θ_t at each iteration using the following approximation to

this gradient:

$$\nabla \log \hat{p}(y_t|y_{1:t-1}, \theta_t) = S_t - S_{t-1}.$$

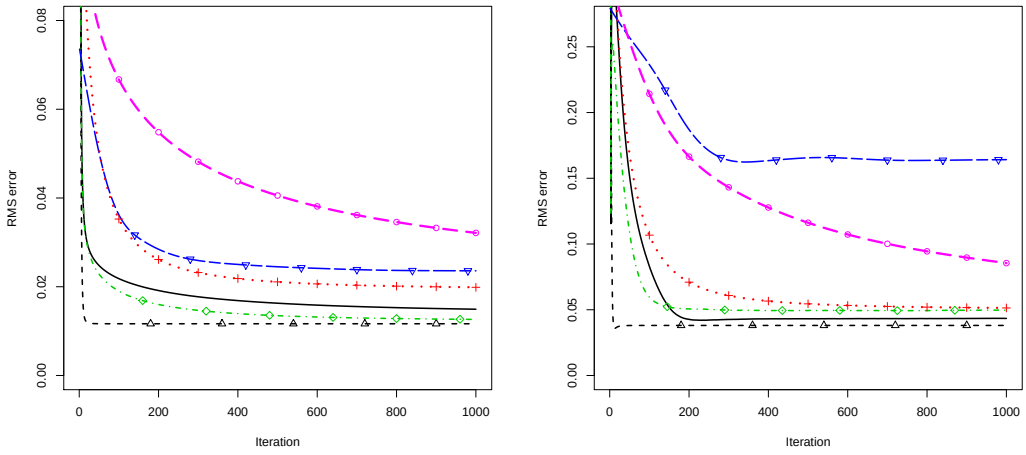
4.7.2 Autoregressive model

The efficiency of the various algorithms is compared in terms of the accuracy of the parameter estimates for the first order autoregressive plus noise model (4.6.1). Data is simulated from the model with parameters $\theta^* = (\phi, \sigma, \tau)^\top = (0.9, 0.7, 1)^\top$.

Firstly we consider batch estimation methods. We simulated data with 1,000 observations and estimated the score vector and observed information matrix using our $\mathcal{O}(N)$ algorithm and the $\mathcal{O}(N)$ and $\mathcal{O}(N^2)$ algorithms of Poyiadjis et al. (2011). The estimates of the score vector and observed information matrix were used within the Newton-Raphson gradient ascent algorithm (4.3.1) to estimate the parameters θ. The starting parameter values for the Newton-Raphson method are $\theta_0 = (\phi, \sigma, \tau)^T = (0.6, 1, 0.7)^T$. The AR(1) plus noise model is a linear-Gaussian model. It therefore allows for a direct comparison of the algorithms against the Kalman filter, where the score vector and observed information matrix can be calculated analytically.

We also provide a comparison to the fixed-lag smoother presented by Kitagawa and Sato (2001). This method works on the assumption that the state space model has good forgetting properties (Cappé et al., 2005). If this is the case then we can approximate $p(x_{1:t}|y_{1:T}, \theta)$ with $p(x_{1:t}|y_{1:\min\{t+L,T\}}, \theta)$, where L is some prespecified lag. This approximation works well if the observations received at times $k > t + L$ do not provide any additional information about $X_{1:t}$. The fixed-lag smoother can be seen as an alternative approach for reducing the particle degeneracy in the score

estimate and an alternative to our $\mathcal{O}(N)$ algorithm in terms of computational time. This method also introduces an increased storage cost compared to the other methods considered, where it is now necessary to store the L previous ancestor particles of $X_t^{(i)}$. Estimates given by the fixed-lag smoother are known to introduce a bias at a cost of reducing the variance of the estimates. Theoretical results given by Olsson et al. (2008), which trade-off the bias and variance, show that asymptotically as the number of observations, T, increases we should set the lag to be proportional to the log of T. Following Dahlin et al. (2014), who implemented the fixed lag smoother for an AR(1) model, we chose $L = \log(T)$.



(a) (b)

Figure 4.7.1: Root mean squared error of parameter estimates (a) ϕ and (b) σ averaged over 20 data sets from our $\mathcal{O}(N)$ algorithm with $N = 50,000$ (——), Poyiadjis et al. (2011) $\mathcal{O}(N)$ with $N = 50,000$ (- ▽ — - ▽ —), Poyiadjis et al. (2011) $\mathcal{O}(N^2)$ (— · ◇ — · — ◇) with $N = 1000$, Fixed-lag smoother (– ○ – – ○ –) with $N = 50,000$, Fixed-lag smoother (· · + · · + ·) with score only and the Kalman filter estimate (- △ - - △ -).

Figure 4.7.1 gives the RMS error of the parameters estimated using the Newton-Raphson gradient ascent algorithm (4.3.1) averaged over 20 data sets simulated from the model. Approximations of the score vector and observed information matrix are obtained from our $\mathcal{O}(N)$, the $\mathcal{O}(N)$ and $\mathcal{O}(N^2)$ algorithms of Poyiadjis et al. (2011) and the fixed-lag smoother. The $\mathcal{O}(N)$ algorithm of Poyiadjis et al. (2011) is implemented with 50,000 particles and the $\mathcal{O}(N^2)$ algorithm is implemented with 1,000 particles. Our $\mathcal{O}(N)$ algorithm is implemented with 50,000 and shrinkage parameter $\lambda = 0.95$. In this setting our $\mathcal{O}(N)$ algorithm corresponds to a 20 fold computational time saving, respectively, over the $\mathcal{O}(N^2)$ algorithm. The fixed-lag smoother is implemented with 50,000 particles and lag $L = 7$, where the parameters are estimated in two ways, using the observed information matrix in the Newton-Raphson gradient ascent algorithm and also estimated using the standard gradient ascent algorithm without the observed information matrix.

The RMS error of the Poyiadjis et al. (2011) $\mathcal{O}(N^2)$ algorithm given in Figure 4.7.1 is comparable to the RMS error given by our $\mathcal{O}(N)$ algorithm, however, it is important to remember that this is achieved with a significant computational saving (20 times faster). Compared to the Poyiadjis et al. (2011) $\mathcal{O}(N)$ algorithm, our $\mathcal{O}(N)$ algorithm and the fixed-lag smoother produce lower RMS error when the fixed lag smoother uses only the score estimate in the gradient ascent algorithm.

On the other hand, estimating the parameters using the fixed-lag smoother and the Newton-Raphson gradient ascent algorithm leads to larger RMS error than the fixed lag-smoother used to estimate only the score. We believe the poor performance of the fixed-lag approach is due to the difficulty it has in estimating the observed

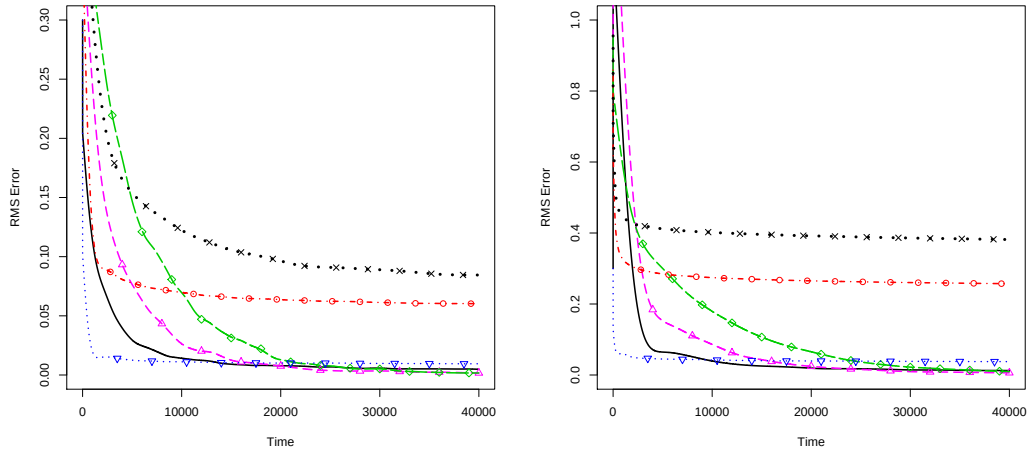
information matrix. The fixed-lag approach reduces the variability in the estimates of $\nabla \log p(x_{1:t}, y_{1:t}|\theta)$ associated with each particle, which means that it under-estimates the first term in Louis's identity (4.3.3). Whilst our approach also reduces the variability in the estimates of $\nabla \log p(x_{1:t}, y_{1:t}|\theta)$ associated with each particle, we are able to correct for this within the Rao-Blackwellisation scheme (see Section 4.4.2 for details). For this example, we find the fixed-lag smoother gives a poor estimate of the terms in the observed information matrix that are related to τ, and that these errors become more pronounced as τ gets smaller.

The recursive gradient ascent scheme (4.7.1) allows us to apply our method for online parameter estimation. Here we compare our method with a fully-Bayesian online method, implemented using the particle learning algorithm (Carvalho et al., 2010), which uses MCMC moves to update the parameters within a sequential Monte Carlo algorithm. This is an interesting comparison as the particle learning algorithm performs sequential Bayesian parameter estimation. A prior distribution is selected for each of the parameters which is updated at each time point via a set of low-dimensional sufficient statistics.

If we assume standard conjugate priors which can be updated recursively, then the state parameters (ϕ, σ^2) at time t will be drawn from a normal-inverse gamma distribution and the observation variance parameter τ^2 will be drawn from an inverse gamma distribution. Specifically, $\phi|x_{1:t}, y_{1:t}, \sigma^2 \sim \mathcal{N}(p_t, \sigma^2 q_t)$, $\sigma^2|x_{1:t}, y_{1:t} \sim \mathcal{IG}(a_t/2, b_t/2)$ and $\tau^2|x_{1:t}, y_{1:t} \sim \mathcal{IG}(c_t/2, d_t/2)$, where $(p_t, q_t, a_t, b_t, c_t, d_t)$ are sufficient statistics which can be updated recursively by standard results from linear model theory (see the Appendix for details of the updates). The priors are initialised with the following

hyperparameters, $a_1 = 5, b_1 = 3.5, c_1 = 5, d_1 = 5, p_1 = 0.6, q_1 = 1$ which are chosen so that the prior distributions are centred around the initial parameter values of the gradient scheme.

We generated $40,000$ observations from the AR(1) plus noise model and considered three different sets of true parameter values, chosen to represent different degrees of dependence within the underlying state process: $\phi = 0.9$, 0.99 and 0.999. We set $\sigma^2 = 1 - \phi^2$ so that the marginal variance of the state is 1 and fixed $\tau = 1$. We maintain the same initial parameters θ_0 for the gradient scheme as was used for the batch analysis.



(a) (b)

Figure 4.7.2: Root mean squared error of parameter estimates (a) ϕ and (b) σ averaged over 100 data sets from our algorithm with $\phi = 0.9$ (——), $\phi = 0.99$ (– △ – – △ –), $\phi = 0.999$ (- ◊ — - ◊ —) and the particle learning algorithm with $\phi = 0.9$ ($\cdot\triangledown\cdot\cdot\triangledown\cdot\cdot\cdot$), $\phi = 0.99$ (— ○ · — ○ · —), $\phi = 0.999$ (•• × •• × ••).

Figure 4.7.2 shows the RMS error of our $\mathcal{O}(N)$ algorithm applied to estimate the

parameters θ_t, against the particle learning filter over 100 data sets. The results show that the particle learning filter produces a lower RMS error than our algorithm for the first few thousand observations. However the particle learning filter degenerates for very long time-series, particularly in the case of strong dependence ($\phi = 0.99$ and 0.999). This is due to degeneracy in the sufficient statistics that occurs as a result of their dependence on the complete latent process, and the fact that the Monte Carlo approximation to $p(x_{1:T}|y_{1:T})$ degrades as T increases (Andrieu et al., 2005). This degeneracy is particularly pronounced for large ϕ, as this corresponds to cases where the underlying MCMC moves used to update the parameters mix poorly.

Over longer data sets applying the gradient ascent method with our $\mathcal{O}(N)$ algorithm outperforms the particle learning filter. Our method appears to take longer to converge as ϕ approaches 1. This is because the true parameter values are moving further away from the fixed starting value used to initiate the gradient scheme, however compared to the particle learning filter our method appears to be robust to the choice of ϕ. For this reason maximum likelihood methods are to be preferred over particle learning when estimating the parameters from a long time series.

4.7.3 Stochastic volatility model

Stochastic volatility models have been applied extensively to analyse financial time series data (Hull and White, 1987). The model is a discrete time version of the Black-Scholes option pricing equation that accounts for changes in variance over time:

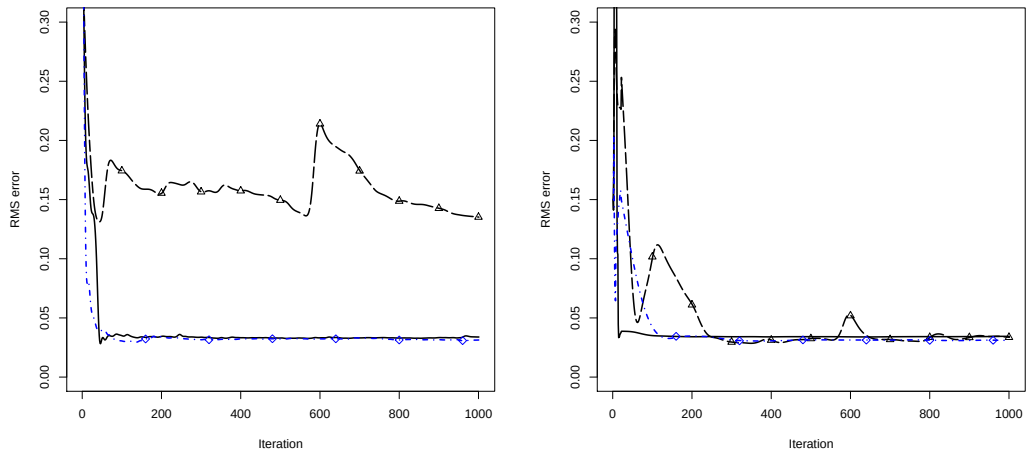
$$X_t|X_{t-1} = x_{t-1} \sim \mathcal{N}(\phi x_{t-1}, \sigma^2), \qquad Y_t|X_t = x_t \sim \mathcal{N}(0, \beta^2 \exp(x_t)).$$

In this model the observations y_t represent the logarithm of the daily difference in the exchange rate and x_t is the unobserved volatility. It is assumed that the volatility process is stationary (i.e. $0 < \phi < 1$), where ϕ is the persistence in volatility and β is the instantaneous volatility.

Previous approaches to estimating the unobserved state have involved MCMC methods (Kim et al., 1998; Jacquier et al., 1994). In recent years reliable approximations have been achieved through sequential Monte Carlo methods (Pitt and Shephard, 1999). Again our method for estimating the score vector and observed information matrix can be incorporated into the Newton-Raphson steepest ascent algorithm (4.3.1) to find the maximum likelihood estimates of the model parameters from batch data and recursive data.

A comparison of the parameter estimation algorithms is performed on data simulated from the stochastic volatility model with parameters $\theta^* = (\phi, \sigma, \beta)^\top = (0.9, 0.25, 0.65)^\top$. Figure 4.7.3 displays the RMS error over 20 data sets for parameters estimated from batch data (1,000 observations). The results are consistent with those of the AR(1) plus noise model where our $\mathcal{O}(N)$ algorithm produces similar results to the $\mathcal{O}(N^2)$ algorithm of Poyiadjis et al. (2011), but at a significantly reduced computational cost, and improved parameter estimates over their $\mathcal{O}(N)$ algorithm. The Poyiadjis et al. (2011) $\mathcal{O}(N)$ algorithm displays erratic behaviour due to the increasing variance of the estimates. Poor estimates of the score vector or observed information matrix can cause large parameter jumps when applied within the gradient ascent scheme.

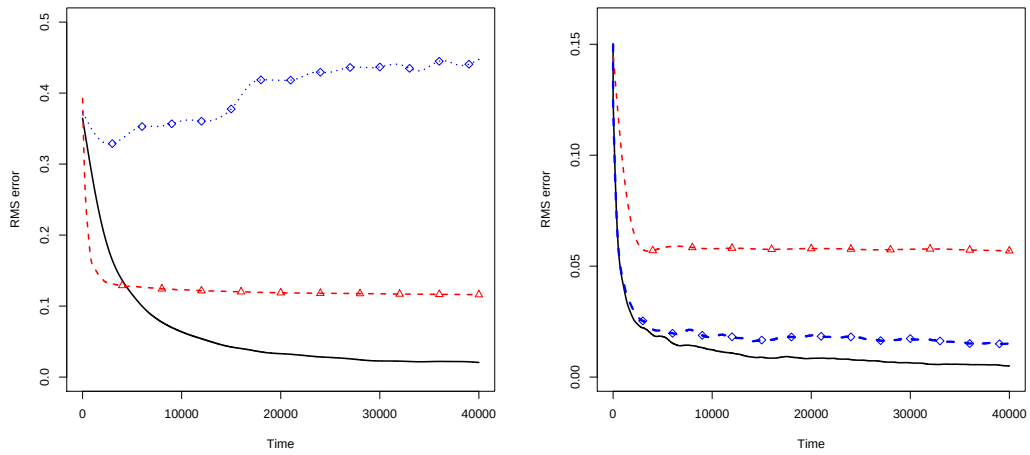
Figure 4.7.4 gives the RMS error of the parameters estimated online using our algorithm with the recursive gradient ascent scheme (4.7.1). The parameters are esti-



(a) (b)

Figure 4.7.3: Root mean squared error of parameter estimates (a) ϕ and (b) β averaged over 20 data sets from our $\mathcal{O}(N)$ algorithm with $N = 50,000$ (——), Poyiadjis et al. (2011) $\mathcal{O}(N)$ with $N = 50,000$ (- △ — - △ —), Poyiadjis et al. (2011) $\mathcal{O}(N^2)$ (— · ◊ — · ◊ —) with $N = 1000$. Smoothing spline applied for ease of visualisation.

mated from 40,000 observations generated by the stochastic volatility model and the RMS error is averaged over 100 data sets simulated from this model. Compared to the particle learning filter and the Poyiadjis et al. (2011) $\mathcal{O}(N)$ algorithm, our algorithm produces lower RMS error which is consistent with the autoregressive model example. The results illustrate the problem of estimating parameters for long data sets, where for the Poyiadjis et al. (2011) $\mathcal{O}(N)$ algorithm the parameter estimates suffer due to the quadratically increasing variance of the score estimate. The particle learning algorithm begins to suffer from particle degeneracy begins after a few thousand observations whereas our algorithm is robust to the length of the observations.



(a) (b)

Figure 4.7.4: Root mean squared error of parameter estimates (a) ϕ and (b) β averaged over 100 data sets from our $\mathcal{O}(N)$ algorithm (——), Poyiadjis et al. (2011) $\mathcal{O}(N)$ ($\cdot\cdot\Diamond\cdot\cdot\Diamond\cdot\cdot$) and the particle learning algorithm (– $\triangle$ – – $\triangle$ –).

4.8 Discussion

In this chapter a novel approach for estimating the score vector and observed information matrix for nonlinear, non-Gaussian state space models is presented. Using sequential Monte Carlo methods to estimate the latent process of the state space model we are able to produce particle approximations of the score vector and observed information matrix. Previous approaches have achieved estimates with quadratically increasing variance at linear computational cost in the number of particles or achieved linearly increasing variance at a quadratic computational cost.

The algorithm we have developed combines techniques from kernel density estimation and Rao-Blackwellisation to yield estimates of both the score vector and the

observed information matrix which display only linearly increasing variance, which is achieved at a linear computational cost. The use of kernel density estimation introduces error into the approximation of the score vector. Importantly, we have shown that this approximate score vector, at the true parameter value, has expectation zero when taken with respect to the data. Thus, the resulting gradient ascent scheme uses Monte Carlo methods to approximately find the solution to a set of unbiased estimating equations.

The estimates of the score vector and observed information matrix given by our $\mathcal{O}(N)$ algorithm can be applied to the gradient ascent and Newton-Raphson algorithms to obtain maximum likelihood estimates of the parameters of the state space model. This can be achieved in either an offline or online setting where the parameters are estimated from a batch of observations or recursively from observations received sequentially. This is a significant improvement over the $\mathcal{O}(N^2)$ algorithm of Poyiadjis et al. (2011). We have shown that our algorithm compares favourably with other gradient based methods for offline parameter estimation.

For a significant reduction in computational time we can achieve improved parameter estimation over competing methods in terms of minimising root mean squared error. We also compared our algorithm to the particle learning filter for online estimation. The particle learning filter performs well initially but degenerates over long series, whereas our algorithm displays lower root mean squared error over longer series of observations. Our method also appears to be robust to the choice of model parameters compared to the particle learning filter which struggles to estimate the parameters when the states are highly dependent.

Faster parameter convergence using our $\mathcal{O}(N)$ algorithm can be achieved by initialising the gradient ascent procedure with parameters closer to the true parameter values. Bayesian methods such as particle learning could therefore be useful in identifying the region of the parameter space where the true values are found, and then selecting initial parameters for the gradient procedure from this region.

It may be possible to use our method for estimating the score vector and observed information matrix to design efficient MCMC proposal distributions which take into account the local geometry of the target distribution. For example, estimates of the score vector could be used in combination with Langevin dynamics (see Neal, 2010) to create a gradient scheme which incorporates parameter uncertainty in a Bayesian manner. For example, the particle Metropolis Hastings algorithm (Andrieu et al., 2010) uses a particle filter to gain an unbiased estimate of the likelihood, which is then evaluated within the Metropolis Hastings acceptance ratio. In the same particle filter it would be possible to use our algorithm to estimate the score vector with little increase in computational cost and use this estimate within the proposal mechanism. Further work in this area is ongoing.

APPENDIX

Proof of Lemma 1

Define

$$C_{u,s}^{k} = \begin{cases} \lambda^{s-u} & \text{if } s = k, \text{ and} \\ (1-\lambda)\lambda^{s-u} & \text{otherwise.} \end{cases}$$

Then the Lemma states that for $k = 1, \ldots, T$

$$\bar{S}_k = \sum_{u=1}^{k} \sum_{s=u}^{k} C_{u,s}^{k} E_\theta\{\phi_u(X_{1:T}|y_{1:s})\}. \tag{4.8.1}$$

We prove this by induction. Firstly for $k = 1$, we have $m_1(x_{1:T}) = \phi_1(x_{1:T})$ and hence

$$\bar{S}_1 = E_\theta(\phi_1(X_1|y_1),$$

which is of the form (4.8.1) as $C_{1,1}^{1} = 1$. Now assume (4.8.1) holds for $k = 1, \ldots, t-1$, for some $t \leq T$. By taking expectations of (4.5.2) with respect to $p(x_{1:T}|y_{1:t}, \theta)$. Let $\tilde{C}_{u,s}^{t}$ be the resulting coefficient of $E_\theta(\phi_u(X_{1:T}|y_{1:s}))$, we get that

$$\tilde{C}_{u,t}^{t} = \lambda^{t-u},$$

and for $s < t$

$$\tilde{C}_{u,s}^{t} = \sum_{k=s}^{t-1} (1-\lambda)\lambda^{t-k-1} C_{u,s}^{k}.$$

By definition of $C_{u,s}^{s}$, this is equal to λ^{s-u} if $s = t-1$. Whilst for $s < t-1$ we get

$$\begin{aligned} \tilde{C}_{u,s}^{t} &= (1-\lambda)\lambda^{t-s-1}\lambda^{s-u} + \sum_{k=s+1}^{t-1} (1-\lambda)\lambda^{t-k-1}(1-\lambda)\lambda^{s-u} \\ &= (1-\lambda)\lambda^{s-u}\left\{\lambda^{t-s-1} + \sum_{k=s+1}^{t-1} (1-\lambda)\lambda^{t-k-1}\right\}, \end{aligned}$$

which is also equal to $(1-\lambda)\lambda^{s-u}$. Together these show the coefficients of $E_\theta(\phi_u(X_{1:T}|y_{1:s})$ in $\bar{S}_t$ are of the form specified by Lemma 1, as required.

Particle learning updates

Parameter updates for the particle learning filter are as follows:

The state model parameters (ϕ, σ^2) are sampled from an normal-inverse gamma distribution,

$$\begin{aligned} \phi|x_{1:t}, y_{1:t}, \sigma^2 &\sim \mathcal{N}(p_t, \sigma^2 q_t) \qquad \text{and} \\ \sigma^2|x_{1:t}, y_{1:t} &\sim \mathcal{IG}(a_t/2, b_t/2) \end{aligned}$$

where, $q_t^{-1} = q_{t-1}^{-1} + x_t^2$, $q_t^{-1} p_t = q_{t-1}^{-1} p_{t-1} + x_{t-1} x_t$, $a_t = a_{t-1} + 1$, $b_t = b_{t-1} + (x_t - p_t x_{t-1}) x_t + (p_{t-1} - p_t) q_{t-1}^{-1} p_{t-1}$.

The variance of the observation model τ^2 is sampled from an inverse gamma distribution,

$$\tau^2|x_{1:t}, y_{1:t} \sim \mathcal{IG}(c_t/2, d_t/2)$$

where, $c_t = c_{t-1} + 1$ and $d_t = d_t + (y_t - x_t)^2$.

Chapter 5

Particle Metropolis Adjusted Langevin Algorithms for State Space Models

Abstract

Particle MCMC is a class of algorithms that can be used to analyse state space models. They use MCMC moves to update the parameters of the models, and particle filters to propose values for the path of the state space model. Currently the default is to use random walk Metropolis to update the parameter values. We show that it is possible to use information from the output of the particle filter to obtain better proposal distributions for the parameters. In particular it is possible to obtain estimates of the gradient of the log posterior from each run of the particle filter, and use these estimates within a Langevin-type proposal. We propose using the recent

computationally efficient approach of Nemeth et al. (2013) for obtaining such estimates. We show empirically that for a variety of state space models this proposal is more efficient than the standard random walk Metropolis proposal in terms of: reducing autocorrelation of the posterior samples, reducing the burn-in time of the MCMC sampler and increasing the squared jump distance between posterior samples.

5.1 Introduction

Markov chain Monte Carlo (MCMC) algorithms are a popular and well studied methodology that can be used to draw samples from posterior distributions. MCMC has allowed Bayesian statistics to evolve beyond simple tractable models to more complex and realistic models where the posterior may only be known up to a constant of proportionality. Over the past few years MCMC methodology has been extended further to tackle problems where the model likelihood is intractable. For such models it is often possible to replace the intractable likelihood with an estimate (Beaumont, 2003), which can be obtained from Monte Carlo simulations. Andrieu and Roberts (2009) showed that within the MCMC sampler, if the likelihood is replaced with an unbiased estimate, then the sampler still targets the correct stationary distribution. Andrieu et al. (2010) extended this work further to create a class of MCMC algorithms for state space models based on sequential Monte Carlo methods (also known as particle filters). This class of algorithms is referred to as particle MCMC. In this chapter we shall focus on one particular algorithm, the particle marginal Metropo-

lis Hastings algorithm which replaces the likelihood term in the Metropolis Hastings (MH) sampler with an unbiased particle filter estimator.

In the standard Metropolis Hastings algorithm a popular proposal is the random walk Metropolis (RWM). This proposal selects new parameter values by perturbing the previous values with random Gaussian noise. The efficiency of the MH algorithm is dependent on the magnitude of the noise added. Theoretical results have established that tuning this proposal such that approximately 23.4% of the proposed samples are accepted is optimal as the number of parameters tend to infinity (Roberts et al., 1997). An extension to the RWM proposal is the Metropolis adjusted Langevin algorithm (MALA) which incorporates an estimate of the gradient of the posterior within the proposal distribution. This proposal has the advantage of steering the proposed parameters towards the mode of the posterior. Thus proposed values are more likely to be accepted, and it has an optimal acceptance rate of 57.4% (Roberts and Rosenthal, 1998).

As with the standard Metropolis Hastings algorithm the efficiency of the particle marginal Metropolis Hastings algorithm is also affected by the choice of proposal distribution. For state space models, it is generally not possible to use the MALA proposal because, as with the likelihood, the gradient of the log posterior is intractable. In this chapter we present an algorithm for creating approximations of the gradient of the log posterior using output from the particle filter, based on the algorithm given by Nemeth et al. (2013). The particle approximation of the gradient is then used within the MALA framework to create a new proposal which we refer to as particle MALA (pMALA).

In order for the pMALA algorithm to be practicable it is important that the extra computational cost of estimating the gradient of the log posterior is small, while the estimate itself is accurate. As such, the use of the algorithm from Nemeth et al. (2013) is central to our algorithm. This algorithm has a cost that is linear in the number of particles, and requires only a small overhead on top of running a standard particle filter. It has been shown to have a much smaller Monte Carlo error for estimating the gradient than other algorithms whose computational cost is linear in the number of particles. A similar pMALA algorithm has been independently proposed by Dahlin et al. (2013), but their algorithm for estimating the gradient has a computational cost that is quadratic in the number of particles, and hence leads to a much slower pMALA algorithm.

The outline of the chapter is as follows. We first give an introduction to state space models, and to MCMC and sequential Monte Carlo (particle filter) algorithms for analysing these models. In Section 5.3 we introduce particle MCMC, and show that information from running the particle filter can be used to guide the choice of proposal distribution for the parameters. We then introduce our pMALA algorithm. Section 5.4 presents empirical results comparing pMALA and standard particle MCMC algorithms across a range of examples. The chapter ends with a discussion.

5.2 Inference for state space models

5.2.1 State space models

Consider the general state space model where there is a latent Markov process $\{X_t; 1 \leq t \leq T\}$ that takes values on some measurable space $\mathcal{X} \subseteq \mathbb{R}^{n_x}$. The process is fully characterised by its initial density $p(x_1|\theta) = \mu_\theta(x_1)$ and transition probability density

$$p(x_t|x_{1:t-1}, \theta) = p(x_t|x_{t-1}, \theta) = f_\theta(x_t|x_{t-1}),$$

where $\theta \in \Theta$ represents a vector of model parameters. For an arbitrary sequence $\{z_i\}$ the notation $z_{i:j}$ corresponds to $(z_i, z_{i+1}, \ldots, z_j)$ for $i \leq j$.

We assume that the process $\{X_t\}$ is not directly observable, but partial observations are received via a second process $\{Y_t; 1 \leq t \leq T\} \subseteq \mathcal{Y}^{n_y}$. The observations $\{Y_t\}$ are conditionally independent given $\{X_t\}$ and are defined by the probability density

$$p(y_t|y_{1:t-1}, x_{1:t}, \theta) = p(y_t|x_t, \theta) = g_\theta(y_t|x_t).$$

The marginal likelihood of observations for a given θ can be decomposed as

$$p(y_{1:T}|\theta) = p(y_1|\theta) \prod_{t=2}^{T} p(y_t|y_{1:t-1}, \theta), \tag{5.2.1}$$

where,

$$p(y_t|y_{1:t-1}, \theta) \;\; = \;\; \int g_\theta(y_t|x_t) \int f_\theta(x_t|x_{t-1}) p(x_{t-1}|y_{1:t-1}, \theta) dx_{t-1} dx_t$$

is the predictive likelihood.

Aside from a few special cases, it is generally not possible to evaluate the likelihood analytically, but it is often possible to approximate the likelihood using importance

sampling (Pitt, 2002). Model parameters θ can be estimated by maximising the likelihood using expectation maximisation (EM) or gradient based maximum likelihood methods (Nemeth et al., 2013; Poyiadjis et al., 2011; Dempster et al., 1977). Alternatively, within the Bayesian framework, MCMC techniques (Andrieu et al., 2010; Fearnhead, 2011) can be applied to estimate the posterior density $p(\theta|y_{1:T})$ of the parameters conditional on the observed data. Within this chapter we shall consider only the latter case of applying MCMC to state space models.

5.2.2 MCMC for state space models

We start by considering the generic MCMC algorithm used to perform Bayesian inference on the parameters θ. Firstly, we introduce a prior distribution for the parameters, $p(\theta)$. Our goal is then to estimate the posterior density $p(\theta|y_{1:T}) \propto p(y_{1:T}|\theta)p(\theta)$, which is known only up to a constant of proportionality. In this setting we are considering the ideal case, where we assume that the likelihood (5.2.1) is known and tractable.

Samples from the posterior $(\theta_1, \theta_2, \ldots, \theta_j, \ldots, \theta_J)$ are generated using the Metropolis Hastings algorithm where proposed values θ' are sampled from a proposal distribution $q(\cdot|\theta_{j-1})$ and accepted (i.e. $\theta_j = \theta'$) with probability

$$\alpha(\theta'|\theta_{j-1}) = \min\left\{1, \frac{p(y_{1:T}|\theta')p(\theta')q(\theta_{j-1}|\theta')}{p(y_{1:T}|\theta_{j-1})p(\theta_{j-1})q(\theta'|\theta_{j-1})}\right\}. \tag{5.2.2}$$

The samples $\{\theta_j\}_{j=1}^J$ generated by the MH algorithm form a Markov chain of correlated samples. The choice of proposal distribution $q(\theta'|\theta)$ is important as it affects the autocorrelation of the samples from the algorithm. A standard choice

of proposal is the Gaussian random walk proposal. This proposal generates new parameter values by perturbing the current parameters with Gaussian noise. If we denote σ_ϵ to be a user chosen step size and $\mathbf{I}$ the identity matrix then:

$$\theta' = \theta_{j-1} + \sigma_\epsilon z \quad \text{where} \quad z \sim \mathcal{N}(0, \mathbf{I}).$$

The efficiency of the Gaussian random walk proposal is determined by the scaling of the step size parameter. Theoretical results show that as the number of parameters $d \to \infty$ the optimal acceptance rate for the MH ratio (5.2.2) is 0.234 (Roberts et al., 1997). A great deal of research has been dedicated to the optimal scaling of the Gaussian random walk proposal and the interested reader is referred to Roberts and Rosenthal (2001) for a review.

Alternatively, efficient proposal distributions can be designed using the geometry of the posterior density (Roberts and Tweedie, 1996; Girolami and Calderhead, 2011) to improve the mixing of the MCMC sampler. One such approach is the Metropolis adjusted Langevin algorithm (Roberts and Rosenthal, 1998) which uses the gradient of the log posterior $\nabla \log p(\theta|y_{1:T})$ within the proposal

$$\theta' = \theta_{j-1} + \sigma_\epsilon z + \frac{\sigma_\epsilon^2}{2} \nabla \log p(\theta_{j-1}|y_{1:T})$$

where $z \sim \mathcal{N}(0, \mathbf{I})$ and σ_ϵ is the step size. The gradient of the log posterior can be given in terms of the score vector (gradient of the log-likelihood) $\nabla \log p(y_{1:T}|\theta)$ and the gradient of the log prior density $\nabla \log p(\theta|y_{1:T}) = \nabla \log p(y_{1:T}|\theta) + \nabla \log p(\theta)$.

Samples proposed using MALA tend to be less correlated compared to samples generated from the Gaussian random walk proposal. Intuitively, this is because using the gradient of the log posterior steers the proposed samples towards the mode of the

posterior allowing for more ambitious jumps in the parameter space which are likely to be accepted. Roberts and Rosenthal (1998) showed that applying MALA within the MCMC sampler gives an optimal acceptance rate of 0.574, much higher than the standard Gaussian random walk proposal. Also they show that the mixing of MALA scales much better with dimension than random walk Metropolis.

Finally, it is worth noting that adaptive MCMC methods (see Andrieu and Thoms (2008) for a review) can be applied to tune the proposal distribution. These methods use the empirical covariance matrix of the Markov chain as the proposal variance. The proposal can then be tuned (i.e. increasing or decreasing the proposal variance) so that a given percentage of samples are accepted. The main difference between this approach to tuning the proposal and the geometric approach, such as MALA, is that the adaptive MCMC algorithm produces a global proposal, whereas the geometric approach produces local proposals.

The outline of MCMC given above is appropriate for the idealised scenario where the likelihood $p(y_{1:T}|\theta)$ is tractable. However, for most state space models this is not the case. Andrieu and Roberts (2009) showed that by replacing the likelihood with a Monte Carlo estimate $\hat{p}(y_{1:T}|\theta)$, which is non-negative and unbiased (Del Moral, 2004), the MCMC sampler will still target the correct posterior distribution. One way of obtaining unbiased estimates of the likelihood for state space models is to use a particle filter.

5.2.3 Sequential Monte Carlo

Sequential Monte Carlo algorithms represent a class of simulation methods for the sequential approximation of posterior probability distributions. In the context of state space modelling, we are interested in approximating the posterior $p(x_t|y_{1:t}, \theta)$ of the filtered latent state x_t, given a sequence of observations $y_{1:t}$. In this section we shall assume that the model parameters θ are fixed. Approximations of $p(x_t|y_{1:t}, \theta)$ can be calculated recursively by first approximating $p(x_1|y_1, \theta)$, then $p(x_2|y_{1:2}, \theta)$ and so forth for $t = 1, \ldots, T$. At time t the posterior of the filtered state is

$$p(x_t|y_{1:t}, \theta) \propto g_\theta(y_t|x_t) \int f_\theta(x_t|x_{t-1}) p(x_{t-1}|y_{1:t-1}, \theta) dx_{t-1} \tag{5.2.3}$$

where $p(x_{t-1}|y_{1:t-1}, \theta)$ is the posterior at time $t-1$.

The posterior at time t can be approximated if we assume that at time $t-1$ we have a set of particles $\{x_{t-1}^{(i)}\}_{i=1}^N$ and corresponding weights $\{w_{t-1}^{(i)}\}_{i=1}^N$ which produce a discrete approximation of $p(x_{t-1}|y_{1:t-1}, \theta)$. The Monte Carlo approximation for (5.2.3) at time t is then

$$\hat{p}(x_t|y_{1:t}, \theta) \approx c g_\theta(y_t|x_t) \sum_{i=1}^N w_{t-1}^{(i)} f_\theta(x_t|x_{t-1}^{(i)}), \tag{5.2.4}$$

where c is a normalising constant. The particle approximation $\hat{p}(x_t|y_{1:t}, \theta)$ tends to the true density $p(x_t|y_{1:t}, \theta)$ as the number of particles $N \to \infty$ (Crisan and Doucet, 2002). The filtered density, as given above, can be updated recursively by propagating and updating the particle set using importance sampling techniques. The resulting algorithms are called particle filters, see Doucet et al. (2000), Doucet and Johansen (2011) and Cappé et al. (2007) for a review.

In this chapter the particle approximations of the latent process are created with the auxiliary particle filter of Pitt and Shephard (1999). This filter can be viewed as a general filter from which simpler filters are given as special cases. We shall consider the version of this filter as presented in Fearnhead et al. (2010). The aim is to view the target (5.2.4) as defining a joint distribution on the particle at time $t-1$ and the value of a new particle at time t. The probability of sampling particle $x_{t-1}^{(i)}$ and using a conditional density for then sampling x_t is

$$cw_{t-1}^{(i)} g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1}^{(i)}).$$

We approximate this with $\xi_t^{(i)} q(x_t|x_{t-1}^{(i)}, y_t, \theta)$, where $q(x_t|x_{t-1}^{(i)}, y_t, \theta)$ is a density function that can be sampled from and $\{\xi_t^{(i)}\}_{i=1}^N$ are a set of probabilities. This defines a proposal which we can simulate from by first choosing particle $x_{t-1}^{(i)}$ with probability $\xi_t^{(i)}$, and then, conditional on this, a new particle value, x_t, is sampled from $q(x_t|x_{t-1}^{(i)}, y_t, \theta)$. The weight assigned to our new particle is then

$$\tilde{w_t} = \frac{w_{t-1}^{(i)} g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1}^{(i)})}{\xi_t^{(i)} q(x_t|x_{t-1}^{(i)}, y_t, \theta)}.$$

Details are summarised in Algorithm 10.

Simpler filters, such as the bootstrap filter (Gordon et al., 1993), can be derived from the general filter by setting $q(x_t|x_{t-1}^{(i)}, y_t, \theta) = f_\theta(x_t|x_{t-1}^{(i)})$ and $\xi_t^{(i)} = w_{t-1}^{(i)}$. The bootstrap filter is a popular choice due to its simplicity, however, this filter can be inefficient as it does not take account of the newest observations in the proposal, and therefore can lead to the propagation of particles that are likely to be given small weights.

The optimal proposal density, in terms of minimising the variance of the weights (Doucet et al., 2000), is available when $q(x_t|x_{t-1}^{(i)}, y_t, \theta) = p(x_t|x_{t-1}^{(i)}, y_t, \theta)$ and $\xi_t^{(i)} \propto w_{t-1}^{(i)} p(y_t|x_{t-1}^{(i)})$. This filter is said to be *fully adapted* as all the weights $w_t^{(i)}$ will equal $1/N$. Generally it is not possible to sample from the optimal proposal, but alternative proposals can be used which approximate the fully adapted filter.

Algorithm 10 Auxiliary Particle Filter

Step 1: Iteration $t = 1$.

(a) For $i = 1, \ldots, N$, sample particles $\{x_1^{(i)}\}_{i=1}^N$ from $p(x_1|\theta)$ and set $\tilde{w}_1^{(i)} = p(y_1|x_1^{(i)})$.

(b) Calculate $C_1 = \sum_{i=1}^N \tilde{w}_1^{(i)}/N$; set $\hat{p}(y_1) = C_1/N$; and calculate normalised weights $w_1^{(i)} = \tilde{w}_1^{(i)}/C_1$ for $i = 1, \ldots, N$.

Step 2: Iteration $t = 2, \ldots, T$. Assume a user-defined set of proposal weights $\{\xi_t^{(i)}\}_{i=1}^N$ and family of proposal distributions $q(x_t|x_{t-1}^{(i)}, y_t, \theta)$.

(a) Sample indices $\{k_1, k_2, \ldots, k_N\}$ from $\{1, \ldots, N\}$ with probabilities $\xi_t^{(i)}$.

(b) Propagate particles $x_t^{(i)} \sim q(\cdot|x_{t-1}^{(k_i)}, y_t, \theta)$.

(c) Weight particles $\tilde{w}_t^{(i)} = \frac{w_{t-1}^{(k_i)} g_\theta(y_t|x_t^{(i)}) f_\theta(x_t^{(i)}|x_{t-1}^{(k_i)})}{\xi_t^{(k_i)} q(x_t^{(i)}|x_{t-1}^{(k_i)}, y_t, \theta)}$ and calculate $C_t = \sum_{i=1}^N \tilde{w}_t^{(i)}$.

(d) Obtain an estimate of the predictive likelihood, $\hat{p}(y_t|y_{1:t-1}, \theta) = C_t/N$, and calculate normalised weights $w_t^{(i)} = \tilde{w}_t^{(i)}/C_t$ for $i = 1, \ldots, N$.

One of the benefits of using the particle filter is that an estimate for the likelihood $p(y_{1:T}|\theta)$ is given for free from the particle filter output. We can estimate $p(y_t|y_{1:t-1}, \theta)$ by

$$\hat{p}(y_t|y_{1:t-1}, \theta) = \left[\sum_{i=1}^N \frac{\tilde{w}_t^{(i)}}{N}\right], \tag{5.2.5}$$

where, $\tilde{w}_t^{(i)}$ are unnormalised weights. An unbiased estimate of the likelihood is then

$$\hat{p}(y_{1:T}|\theta) = \hat{p}(y_1|\theta) \prod_{t=2}^{T} \hat{p}(y_t|y_{1:t-1}, \theta).$$

See Algorithm 10, and Pitt et al. (2012) and Del Moral (2004) for further details and a proof of the unbiasedness property of the marginal likelihood estimate.

5.3 Particle MCMC

5.3.1 Particle marginal Metropolis Hastings

The auxiliary particle filter given in Algorithm 10 provides a positive, unbiased estimate of the likelihood based on the importance weights (5.2.5). Andrieu and Roberts (2009) and Andrieu et al. (2010) have shown how we can use such estimates in place of the likelihood function within MCMC. The idea is to run Algorithm 10 at each iteration of an MCMC algorithm to get an estimate of the likelihood for the current parameter value. We then use this estimate instead of the true likelihood value within the accept-reject probability. If interest lies just in the posterior for the parameter, this results in the particle marginal Metropolis Hastings (PMMH) algorithm (see Algorithm 11). We will focus on this algorithm in the following (see Andrieu et al., 2010, for alternative particle MCMC algorithms).

A key result is that PMMH has $p(\theta|y_{1:T})$ as its stationary distribution (Andrieu and Roberts, 2009; Andrieu et al., 2010). Let $\mathcal{U}$ denote the random variables used in the particle filter to generate the estimate of the likelihood, and $p(\mathcal{U}|\theta)$ their conditional

Algorithm 11 Particle Marginal Metropolis Hastings (PMMH) Algorithm

Step 1: Iteration $j = 1$.

(a) Set θ_1 arbitrarily.

(b) Run Algorithm 10 and compute the marginal likelihood $\hat{p}(y_{1:T}|\theta_1)$ from the importance weights (5.2.5).

Step 2: Iteration $j = 2, \ldots, M$.

(a) Sample $\theta' \sim q(\cdot|\theta_{j-1})$.

(b) Run Algorithm 10 and compute the marginal likelihood $\hat{p}(y_{1:T}|\theta')$ from the importance weights (5.2.5).

(c) Set $\theta_j = \theta'$ and $\hat{p}(y_{1:T}|\theta_j) = \hat{p}(y_{1:T}|\theta')$ with probability $1 \wedge \frac{\hat{p}(y_{1:T}|\theta')p(\theta')q(\theta_{j-1}|\theta')}{\hat{p}(y_{1:T}|\theta_{j-1})p(\theta_{j-1})q(\theta'|\theta_{j-1})}$.

else set $\theta_j = \theta_{j-1}$ and $\hat{p}(y_{1:T}|\theta_j) = \hat{p}(y_{1:T}|\theta_{j-1})$.

density given θ. We can define a target distribution on $(\theta, \mathcal{U})$ which is

$$\hat{p}(\theta, \mathcal{U}|y_{1:T}) \propto \hat{p}(y_{1:T}|\theta, \mathcal{U})p(\mathcal{U}|\theta)p(\theta). \tag{5.3.1}$$

It is straightforward to show that PMMH is a standard MCMC algorithm with this target distribution, and with a proposal distribution $q(\theta'|\theta)p(\mathcal{U}|\theta')$. Furthermore, the marginal target distribution for θ is just

$$\int \hat{p}(\theta, \mathcal{U}|y_{1:T})d\mathcal{U} \propto \int \hat{p}(y_{1:T}|\theta, \mathcal{U})p(\mathcal{U}|\theta)p(\theta)d\mathcal{U} = p(y_{1:T}|\theta)p(\theta),$$

the posterior, $p(\theta|y_{1:T})$. The last equality follows from the fact that $\hat{p}(y_{1:T}|\theta, \mathcal{U})$ is unbiased estimator of the likelihood. Note that implementation of PMMH does not require storing all details of the particle filter, $\mathcal{U}$, just the resulting estimate of the likelihood $\hat{p}(y_{1:T}|\theta, \mathcal{U})$.

Whilst PMMH admits $p(\theta|y_{1:T})$ as the invariant density regardless of the variance

of the likelihood estimator $\hat{p}(y_{1:T}|\theta,\mathcal{U})$, the variance does affect the mixing properties of the algorithm; see Pitt et al. (2012), Doucet et al. (2012) and Sherlock et al. (2013) for details. The choice of proposal distribution for the parameter, $q(\cdot|\theta)$, will also have an important impact on the mixing properties of the algorithm. We now show that information from the particle filter can be used to guide this choice of proposal.

5.3.2 Efficient use of the particle filter output

Consider using some information from the particle filter, which we will denote $\mathcal{I}(\mathcal{U})$, within the proposal distribution. So if the current state of the Markov chain is $(\theta,\mathcal{U})$, our proposal will be $\hat{q}(\theta'|\theta,\mathcal{I}(\mathcal{U}))$. The acceptance probability of a new state $(\theta',\mathcal{U}')$ will then be

$$\alpha(\theta',\mathcal{U}'|\theta,\mathcal{U}) = \min\left\{1, \frac{\hat{p}(y_{1:T}|\theta',\mathcal{U}')p(\theta')\hat{q}(\theta|\theta',\mathcal{I}(\mathcal{U}'))}{\hat{p}(y_{1:T}|\theta,\mathcal{U})p(\theta)\hat{q}(\theta'|\theta,\mathcal{I}(\mathcal{U}))}\right\}. \tag{5.3.2}$$

It is straightforward to show that such an algorithm admits $p(\theta|y_{1:T})$ as the invariant density :

Proposition 5.3.1. *Implementing PMMH with proposal distribution $\hat{q}(\theta'|\theta,\mathcal{I}(\mathcal{U}))$, and acceptance probability given by (5.3.2), gives an MCMC algorithm which admits $p(\theta|y_{1:T})$ as the invariant density.*

Proof. As before the PMMH is a standard MCMC algorithm with $\hat{p}(\theta,\mathcal{U}|y_{1:T})$ as its invariant distribution, but now the proposal distribution is $\hat{q}(\theta'|\theta,\mathcal{I}(\mathcal{U}))p(\mathcal{U}'|\theta')$. The acceptance probability for such an MCMC algorithm is

$$\alpha(\theta',\mathcal{U}'|\theta,\mathcal{U}) = \min\left\{1, \frac{\hat{p}(y_{1:T}|\theta',\mathcal{U}')p(\mathcal{U}'|\theta')p(\theta')\hat{q}(\theta|\theta',\mathcal{I}(\mathcal{U}'))p(\mathcal{U}|\theta)}{\hat{p}(y_{1:T}|\theta,\mathcal{U})p(\mathcal{U}|\theta)p(\theta)\hat{q}(\theta'|\theta,\mathcal{I}(\mathcal{U}))p(\mathcal{U}'|\theta')}\right\}$$

which simplifies to (5.3.2) as required. □

Again, when implementing this version of PMMH we do not need to store all details of the particle filter. All we need is to store our estimate of the likelihood $\hat{p}(y_{1:T}|\theta, \mathcal{U})$ and the information $\mathcal{I}(\mathcal{U})$.

Our choice of information will be an estimate of the score, $\mathcal{I}(\mathcal{U}) = \nabla \log \hat{p}(y_{1:T}|\theta)$, where we give details of how to obtain such an estimate in the next section. We then use this estimate in place of the true score within a MALA proposal:

$$\hat{q}(\theta'|\theta, \mathcal{I}(\mathcal{U})) = \mathcal{N}\left(\theta_{j-1} + \frac{\sigma_\epsilon^2}{2}\left[\nabla \log \hat{p}(\theta|y_{1:T})\right], \sigma_\epsilon^2\right), \tag{5.3.3}$$

where $\nabla \log \hat{p}(\theta|y_{1:T}) = \nabla \log \hat{p}(y_{1:T}|\theta) + \nabla \log p(\theta)$, and σ_ϵ is the step-size parameter. The new proposal (5.3.3) can be used in place of $q(\theta'|\theta)$ in Algorithm 11 to give the particle MALA algorithm.

5.3.3 Particle approximations of the score vector

We can create a particle approximation of the score vector based on Fisher's identity (Cappé et al., 2005)

$$\begin{aligned} \nabla \log p(y_{1:T}|\theta) &= \int \nabla \log p(x_{1:T}, y_{1:T}|\theta) \times p(x_{1:T}|y_{1:T}, \theta) dx_{1:T} \\ &= \mathbb{E}[\nabla \log p(x_{1:T}, y_{1:T}|\theta)|y_{1:T}, \theta] \end{aligned}$$

which is the expectation of

$$\nabla \log p(x_{1:T}, y_{1:T}|\theta) = \nabla \log p(x_{1:T-1}, y_{1:T-1}|\theta) + \nabla \log g_\theta(y_T|x_T) + \nabla \log f_\theta(x_T|x_{T-1})$$

over the path $x_{1:T}$.

The particle approximation to the score vector is obtained by replacing $p(x_{1:T}|y_{1:T}, \theta)$ with a particle approximation $\hat{p}(x_{1:T}|y_{1:T}, \theta)$. Here we outline this idea, but see Poyiadjis et al. (2011) for more details.

For each particle at a time $t-1$, there is an associated path, defined by tracing the ancestry of each particle back in time. With slight abuse of notation denote this path by $x_{1:t-1}^{(i)}$. We can thus associate with particle i at time $t-1$ a value $\alpha_{t-1}^{(i)} = \nabla \log p(x_{1:t-1}^{(i)}, y_{1:t-1}|\theta)$. These values can be updated recursively. Remember that in step 2(b) of Algorithm 10 we sample k_i, which is the index of the particle at time $t-1$ that is propagated to produce the ith particle at time t. Thus we have

$$\alpha_t^{(i)} = \alpha_{t-1}^{(k_i)} + \nabla \log g_\theta(y_t|x_t^{(i)}) + \nabla \log f_\theta(x_t|x_{t-1}^{(k_i)}). \tag{5.3.4}$$

The problem with this approach is that the variance of the score estimate $\nabla \log p(y_{1:t}|\theta)$ increases quadratically with t (Poyiadjis et al., 2011) due to degeneracy in the approximation of α_t. Poyiadjis et al. (2011) suggest an alternative particle filter algorithm, which avoids a quadratically increasing variance but at the expense of a computational cost that is quadratic in the number of particles. Instead we will use the algorithm of Nemeth et al. (2013), which uses kernel density estimation and Rao-Blackwellisation to substantially reduce the Monte Carlo variance, but still maintains an algorithm whose computational cost is linear in the number of particles.

An outline of their approach is as follows. We first use kernel density estimation to replace each discrete $\alpha_{t-1}^{(i)}$ value by a Gaussian distribution:

$$\alpha_{t-1}^{(i)} \sim \mathcal{N}(m_{t-1}^{(i)}, V_{t-1}). \tag{5.3.5}$$

The mean of this distribution is obtained by shrinking $\alpha_{t-1}^{(i)}$ towards the mean of α_{t-1},

$$m_{t-1}^{(i)} = \lambda \alpha_{t-1}^{(i)} + (1-\lambda) \sum_{i=1}^{N} w_{t-1}^{(i)} \alpha_{t-1}^{(i)}.$$

Here $0 < \lambda < 1$ is a user-defined shrinkage parameter. The idea of this shrinkage is that it corrects for the increase in variability introduced through the kernel density estimation of West (1993). For a definition of V_{t-1} see Nemeth et al. (2013), however its actual value does not affect the following details.

The resulting model for the α_ts, including their updates (5.3.4), is linear Gaussian. Hence we can use Rao-Blackwellisation to avoid sampling the $\alpha_t^{(i)}$s, and instead calculate the parameters of the kernel (5.3.5) directly. This gives the following recursion for the means,

$$m_t^{(i)} = \lambda m_{t-1}^{(k_i)} + (1-\lambda) \sum_{i=1}^{N} w_{t-1}^{(i)} m_{t-1}^{(i)} + \nabla \log g_\theta(y_t | x_t^{(i)}) + \nabla \log f_\theta(x_t^{(i)} | x_{t-1}^{(k_i)}).$$

The final score estimate depends only on these means, and is

$$\nabla \log \hat{p}(y_{1:t}|\theta) = \sum_{i=1}^{N} w_t^{(i)} m_t^{(i)}.$$

See Algorithm 12 for a summary.

When $\lambda = 1$ the recursion simplifies to the method given by Poyiadjis et al. (2011), where the variance of the score estimate will increase quadratically with t. The use of a shrinkage parameter $\lambda < 1$ alleviates the degeneracy problems that affect the estimation of the score and significantly reduces the estimate's variance. As a rule of thumb, setting $\lambda = 0.95$ produces reliable estimates where the variance of the score estimate increases only linearly with t (see Nemeth et al. (2013) for further details). We shall use this tuning for all examples given in the Section 5.4.

Algorithm 12 Rao-Blackwellised Kernel Density Estimate of the Score Vector

Add the following steps to Algorithm 10.

Step 1: (c) Set $\nabla \log \hat{p}(y_1|\theta) = \nabla \log g_\theta(y_1|x_1^{(i)}) + \nabla \log \mu_\theta(x_1^{(i)})$.

Step 2: (e) For $i = 1, \ldots, N$, calculate

$$m_t^{(i)} = \lambda m_{t-1}^{(k_i)} + (1-\lambda) \sum_{i=1}^{N} w_{t-1}^{(i)} m_{t-1}^{(i)} + \nabla \log g_\theta(y_t|x_t^{(i)}) + \nabla \log f_\theta(x_t^{(i)}|x_{t-1}^{(k_i)}).$$

(f) Update and store the score vector

$$\nabla \log \hat{p}(y_{1:t}|\theta) = \sum_{i=1}^{N} w_t^{(i)} m_t^{(i)}.$$

5.4 Simulation Studies

In this section we compare the particle marginal Metropolis Hastings algorithm, using the random walk Metropolis proposal (5.3.5), which we shall refer to as PMMH, against the particle MALA proposal (5.3.3). The two proposals shall be compared in terms of the their inefficiency, which is measured by the integrated autocorrelation time of the Markov chain, Ineff $= 1 + 2\sum_{m=1}^{\infty} \rho_l$, where ρ_l is the autocorrelation of the Markov chain at lag l. The infinite sum in the integrated autocorrelation time is truncated to L^*, which is lag after which the autocorrelations are approximately zero. As a rule of thumb the maximum number of lags $L^* = \min\{1000, L\}$, where L is the lowest index for l such that $|\rho_l| < 2/\sqrt{M}$ and M is the sample size used to compute ρ_l. Lower values for the inefficiency indicate less correlation between samples.

The MCMC algorithms can also be compared using the squared jump distance

$$\text{SJD} = \frac{1}{M-1} \sum_{m=1}^{M} |\theta_{m+1} - \theta_m|^2.$$

This metric measures the average distance between successive posterior samples, where larger jumps correspond to better mixing of the MCMC sampler and improved exploration of the posterior.

All results are given as the average of 10 independent Monte Carlo simulations, where for each simulation the PMMH algorithm (Alg. 11) is run for $100,000$ iterations. Only the last $50,000$ iterations are taken as samples from the posterior with the first $50,000$ iterations treated as burn-in.

5.4.1 Linear Gaussian Model

We start by considering the linear Gaussian state space model, where is it is possible to estimate the marginal likelihood $p(y_{1:T}|\theta)$ and score vector exactly with the Kalman filter (Durbin and Koopman, 2001). This model provides a benchmark for comparing the efficiency of PMMH and pMALA. We also implement the MH and MALA algorithms using the exact estimates of the likelihood and score vector given by the Kalman filter. Finally, a comparison is also given for both the $\mathcal{O}(N)$ and $\mathcal{O}(N^2)$ algorithms of Poyiadjis et al. (2011) to estimate the score vector. Proposals created using the $\mathcal{O}(N^2)$ algorithm have been implemented by Dahlin et al. (2013).

Consider the following linear Gaussian model

$$y_t = \alpha + \beta x_t + \tau\epsilon_t, \quad x_t = \mu + \phi x_{t-1} + \sigma\eta_t, \quad x_0 \sim \mathcal{N}(\mu/(1-\phi), \sigma^2/(1-\phi^2)),$$

where ϵ_t and η_t are standard independent Gaussian random variables and $\theta = (\alpha, \beta, \tau, \mu, \phi, \sigma)$ are model parameters.

For this model it is possible to use the fully adapted particle filter using the optimal

proposal for the latent states (see the Appendix for details). Compared to the simpler bootstrap filter this will reduce the variance of the weights, which will therefore reduce the variance of the likelihood estimate.

We use simulated data from the model where 500 observations are generated with model parameters $\alpha = 0.2$, $\beta = 1$, $\tau = 1$, $\mu = 0.1$, $\phi = 0.9$, $\sigma = 0.15$. At each iteration of the PMMH/pMALA algorithm an estimate of the likelihood and score vector was calculated from the particle filter (Alg. 10 and 12) using 500 and 2000 particles. For the Poyiadjis $\mathcal{O}(N^2)$ algorithm, the particle filter is run with $\sqrt{N}$ particles to match the computational cost of the PMMH and pMALA algorithms.

The MCMC sampler was run with the following prior distributions

$$\begin{pmatrix} \alpha \\ \beta \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} 0.3 \\ 1.2 \end{pmatrix}, \tau^2 \begin{pmatrix} 0.25 & 0 \\ 0 & 0.5 \end{pmatrix} \right), \tau^2 \sim \mathcal{IG}(1, 7/20),$$

$\mu \sim \mathcal{N}(0.15, 0.5)$, $(\phi + 1)/2 \sim Beta(20, 5)$ and $\sigma^2 \sim \mathcal{IG}(2, 1/40)$, where $\mathcal{IG}$ is an inverse gamma distribution.

The parameters (ϕ, σ, τ) are constrained such that $|\phi| < 1$, $\sigma > 0$ and $\tau > 0$. These parameters are transformed for the MCMC sampler as $\tanh(\phi)$, $\log(\sigma)$ and $\log(\tau)$, noting that this transformation now introduces a Jacobian term into the MH acceptance ratio (5.2.2).

As discussed in Section 5.2.2 the optimal acceptance rate for the standard random walk Metropolis algorithm is 0.234 as the number of parameters $d \to \infty$. Recent results given by Sherlock et al. (2013) show that for MH algorithms where the likelihood is replaced with an unbiased estimate, the optimal acceptance rate is approximately 0.07. For the PMMH algorithm with RWM proposal we shall use the scaling suggested

by the authors $\sigma_\epsilon^2 = (2.562)^2\Sigma/d$, where Σ is an estimate of the posterior covariance of the parameters θ given from a pilot run (we use 2.38 in place of 2.562 for the Kalman RWM). The particle MALA and Kalman MALA algorithms were scaled as $\sigma_\epsilon^2 = \Sigma/d^{1/3}$ to match the mixing rate of MALA algorithms (Roberts and Rosenthal, 1998).

Algorithm	Particles	Acc. rate	Inefficiency	
			Min	Max
Kal. RWM		0.13	52.47	78.31
Kal. MALA		0.25	27.33	51.55
PMMH	2000	0.14	59.02	107.87
	500	0.13	52.28	116.83
pMALA	2000	0.25	26.87	47.58
	500	0.24	29.65	58.91
Poy. $\mathcal{O}(N)$	2000	0.25	31.62	58.41
	500	0.12	30.26	61.71
Poy. $\mathcal{O}(N^2)$	$\sqrt{2000}$	0.16	50.05	111.67
	$\sqrt{500}$	0.12	128.20	170.97

Table 5.4.1: Linear Gaussian example. Comparison of the efficiency of PMMH, pMALA, Poyiadjis MALA and the exact estimates of the likelihood and score vector from the Kalman filter. Particle approximations are based on 500 and 2000 particles.

Table 5.4.1 summarises the results of the MCMC simulations where for ease of

presentation we have presented the minimum and maximum inefficiencies for each algorithm over all parameters. The inefficiency of all particle filter based samplers is increased, and the acceptance rate decreased, when the number of particles is reduced. The sampler still targets the correct stationary distribution, but less efficiently as a smaller number of particles leads to an increase in the variance of the estimate of the likelihood. This increased inefficiency would be more noticeable for models where it is not possible to use the fully adapted importance proposal distribution. Increasing the number of the particles reduces the variance of the likelihood estimate and increases the acceptance rate of the MCMC sampler (Pitt et al., 2012).

The pMALA algorithm has an increased acceptance rate compared to PMMH and also displays reduced inefficiency. The estimate of the gradient of the log posterior which is used in the proposal allows for greater jumps in the posterior, which leads to reduced autocorrelation between posterior samples. The Poyiadjis $\mathcal{O}(N^2)$ implementation is less efficient than the pMALA algorithm when compared with equal computational effort. The increase in inefficiency of the $\mathcal{O}(N^2)$ algorithm is caused by an increase in the variance of both the likelihood and score estimate. This is caused by the reduced number of particles used to run the particle filter at equal computational cost to pMALA.

Comparing algorithms with equal computational cost, pMALA is more efficient than the Poyiadjis $\mathcal{O}(N)$ implementation of MALA. This is due to the increased variance in the score vector estimate given by the Poyiadjis $\mathcal{O}(N)$ algorithm. Table 5.4.2 gives the inefficiencies for pMALA and the Poyiadjis $\mathcal{O}(N)$ algorithm for datasets with a larger number of observations. As T increases the variance in the estimates of

the likelihood and score vector increase, but can be reduced by increasing the number of particles. For the Poyiadjis $\mathcal{O}(N)$ algorithm, the variance of the score estimate is quadratically increasing in T, which leads to a greater increase in the inefficiency compared to pMALA.

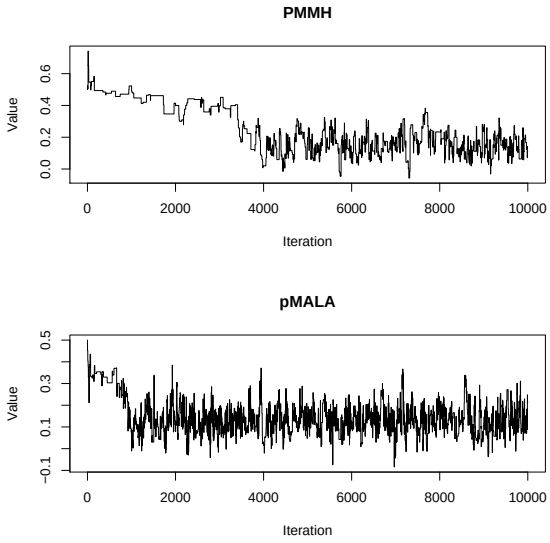


Figure 5.4.1: Linear Gaussian example. Trace plots of PMMH and pMALA for μ parameter.

The pMALA algorithm also has the advantage of reducing the burn-in time of the MCMC sampler. Consider the trace plot of the first $10,000$ samples of the parameter μ shown in Figure 5.4.1. The MCMC sampler using PMMH reaches stationarity after approximately $4,000$ iterations, where the chain is initially sticky with few new parameters accepted. Whereas pMALA reaches stationarity with less than $1,000$ iterations, given the same starting values for both samplers. Using information about the posterior (i.e. the gradient of the log posterior) pMALA can quickly reach stationarity and therefore significantly reduce the burn-in time of the MCMC sampler.

5.4.2 GARCH with noisy observations

This example considers the GARCH(1,1) model (Bollerslev et al., 1994) which has been extensively applied to financial returns data. We assume that the observations are observed with Gaussian noise

$$y_t = x_t + \tau\epsilon_t, \quad x_t = \sigma_t^2\eta_t, \quad \sigma_t^2 = \alpha + \beta x_{t-1}^2 + \gamma\sigma_{t-1}^2, \quad x_0 \sim \mathcal{N}(0, \alpha/(1-\beta-\gamma))$$

where ϵ_t and η_t are standard independent Gaussian random variables and $\theta = (\alpha, \beta, \gamma, \tau)$ are the model parameters.

A dataset with 500 observations is sampled from the model using the parameters $\alpha = 0.1$, $\beta = 0.8$, $\gamma = 0.05$ and $\tau = 0.3$. The model parameters are estimated using the PMMH algorithm and compared against the pMALA implementation. Estimates of the likelihood and score vector, in the case of pMALA, are obtained from the particle filter using 1000 particles.

The parameters of this model must satisfy the following constraints: $\alpha > 0$, $\beta > 0$, $\gamma > 0$, $\tau > 0$ and $\beta + \gamma < 1$. These constraints can be satisfied by re-parameterising the model so that $\phi = \alpha + \beta$, $\mu = \alpha/(1-\phi)$ and $\lambda = \beta/\phi$. The MCMC scheme is then completed by setting the prior distributions for the parameters: $(\phi+1)/2 \sim Beta(10, 3/2)$, $\mu \sim U(0, 2)$, $(\lambda+1)/2 \sim Beta(20, 3/2)$ and $\tau^2 \sim \mathcal{IG}(2, 1/2)$. Finally, the parameters are transformed to the unconstrained scale $\text{logit}(\phi)$, $\log(\mu)$, $\text{logit}(\lambda)$ and $\log(\tau)$ where the appropriate Jacobian is included in the MH ratio.

The proposal of the PMMH algorithm is scaled as $\sigma_\epsilon^2 = (2.562)^2(0.23, 1.43, 0.58, 0.011)/4$, where the vector $(0.23, 1.43, 0.58, 0.011)$ is the diagonal of the covariance matrix for θ obtained from a pilot run. For the pMALA algorithm the proposal is scaled as

$\sigma_\epsilon^2 = (0.23, 1.43, 0.58, 0.011)/4^{1/3}$.

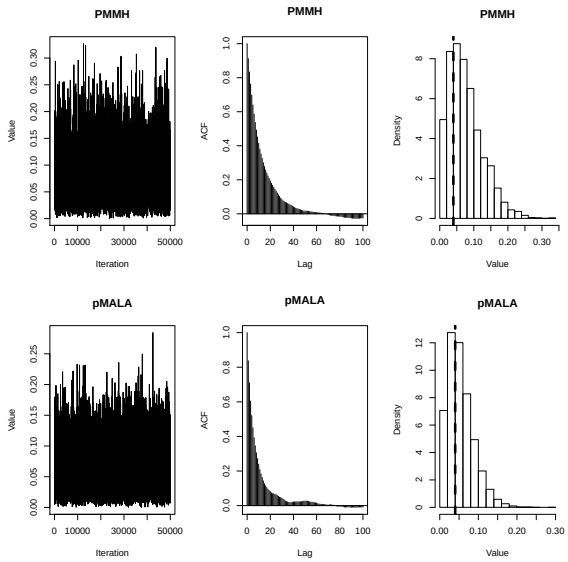


Figure 5.4.2: GARCH example. Trace plots, autocorrelation plots and posterior density (red line indicates true parameter) plots of the parameter γ from the MCMC sampler using PMMH and pMALA.

The trace plots given in Figure 5.4.2 show that the MCMC sampler mixes well for both PMMH and pMALA. The posterior provides a good approximation of the parameter γ with the mode of the density matching the true parameter value. The autocorrelation plots show the reduced lagged correlation and improved mixing of the MCMC sampler using the pMALA algorithm compared to PMMH.

The MCMC simulation results summarised in Table 5.4.3 show the significant improvement of pMALA over PMMH in terms of efficiency (except one parameter) and squared jump distance. Both metrics indicate that pMALA improves the mixing of the MCMC sampler by proposing new parameter values which are in the direction of the mode of the posterior. This increases the acceptance rate of the MCMC scheme

as the sampler is less likely to become stuck in the tails of the density where new samples are unlikely to be accepted.

5.4.3 Stochastic volatility with leverage

The univariate stochastic volatility model is a state space model with non-Gaussian observations, where the latent volatility follows an autoregressive process (see Shephard (2005) for a book length review). Variations of the stochastic volatility model have been extensively applied to model stock market returns. In this example we shall consider the stochastic volatility model with leverage given by Omori et al. (2007),

$$y_t = \exp(x_t/2)\epsilon_t, \quad x_t = \mu + \phi(x_{t-1} - \mu) + \eta_t, \quad x_0 \sim \mathcal{N}(0, \sigma^2/(1-\phi^2))$$

where,

$$\begin{pmatrix} \epsilon_t \\ \eta_t \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & \rho\sigma \\ \rho\sigma & \sigma^2 \end{pmatrix} \right).$$

The observations y_t are the returns, x_t is the latent log-volatility, μ is the drift, σ^2 is the volatility of the log-volatility and ϕ is the persistence parameter. This model also allows the errors in the observation and state transition equations to be correlated through the parameter ρ. In the context of stock market data a negative value of ρ corresponds to an increase in volatility which follows from a drop in returns (Yu, 2005).

We apply the PMMH and pMALA algorithms to estimate the parameters $\theta = (\mu, \phi, \sigma, \rho)$, where the likelihood and score vector are obtained from a particle filter using 1000 particles. We use daily returns data from the S&P 500 index taken from

January 1980 to December 1987 (2022 observations). This dataset has previously been studied by Yu (2005) using MCMC methods and by Jungbacker and Koopman (2007) using a Monte Carlo likelihood method.

The prior distributions for the parameters are: $\mu \sim \mathcal{N}(0, 1)$, $(\phi+1)/2 \sim Beta(20, 1.5)$, $\sigma^2 \sim \mathcal{IG}(2.5, 0.025)$ and $\rho \sim U(-1, 1)$. The constrained parameters ϕ, ρ and $\sigma > 0$ are transformed to unconstrained parameters $\text{logit}(\phi)$, $\text{atanh}(\rho)$ and $\log(\sigma)$, where the Jacobian of the transformation is included in the MH acceptance ratio.

The RWM proposal of the PMMH algorithm is scaled as $\sigma_\epsilon^2 = (2.562)^2(0.10, 0.29, 0.035, 0.02)/4$, where the vector $(0.10, 0.29, 0.035, 0.02)$ is the diagonal of the covariance matrix for θ obtained from a pilot run. For the pMALA proposal the step-size is scaled as $\sigma_\epsilon^2 = (0.10, 0.29, 0.035, 0.02)/4^{1/3}$.

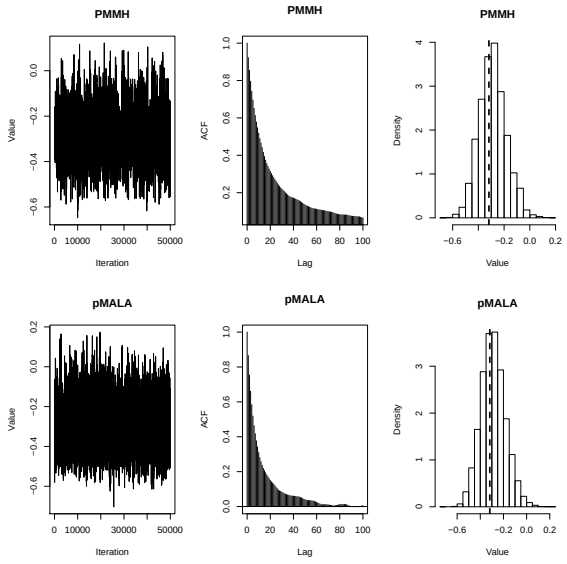


Figure 5.4.3: Stochastic Volatility example. Trace plots, autocorrelation plots and posterior density plots of the parameter ρ from the MCMC sampler using PMMH and pMALA. Red line indicates the posterior mean given by Yu (2005).

As with the previous examples Figure 5.4.3 displays the mixing of the MCMC samplers as well as the autocorrelation and posterior density plots. Both proposals show good mixing and sensible posteriors which contain the parameter estimates given by Yu (2005) and Jungbacker and Koopman (2007). The lagged correlation of the Markov chain given by the autocorrelation plots shows that, compared to PMMH, pMALA explores the posterior density more efficiently.

Table 5.4.4 gives the efficiency of the two proposals in terms of their inefficiency and squared jump distance. For all parameters of the stochastic volatility model pMALA creates a more efficient MCMC sampler than PMMH. The increased acceptance rate indicates that pMALA allows the sampler to propose samples which are more likely to be accepted and therefore better explore the posterior. The result is an increased squared jump distance between samples and reduced correlation between samples.

5.5 Discussion

The particle MALA proposal presented in this chapter shows a significant improvement over the standard random walk Metropolis proposal when applied to the particle marginal Metropolis Hastings algorithm. One of the main advantages of this algorithm is its fast computational time, where the order of computation is equivalent to the computational effort required to estimate the likelihood. This means that more particles can be used to estimate the score vector and likelihood, resulting in estimates with lower variance and improved mixing of the MCMC sampler.

This proposal can also be used with more complex models where the derivative of

the log posterior is not available for all parameters, but can be computed for a subset of the parameters. This will improve the overall efficiency of the sampler as pMALA will sample, from this subset, parameters more likely to be accepted. The remaining parameters can be sampled with the random walk Metropolis proposal.

A second order MALA proposal (Dahlin et al., 2014) which takes account of the curvature of the posterior could be found using an estimate of the observed information matrix. This idea was discussed by Doucet, Jacob and Johansen in the discussion section of Girolami and Calderhead (2011). In principle, this would improve the mixing of the MCMC sampler by taking account of the local parameter covariance structure. However, estimates of the observed information matrix are not guaranteed to be positive definite which is an issue that would need to be addressed.

An important extension to this work would be to develop theoretical results establishing the optimal acceptance rate for pMALA. Recent results (Sherlock et al., 2013) have established optimal acceptance rates for the random walk Metropolis proposal which are helpful when tuning these proposals. Similar results for pMALA would make it easier to implement and reduce the time spent tuning the algorithm.

APPENDIX

Importance proposals

The importance proposals $\sum_{i=1}^{N} \xi_t^{(i)} q(x_t|x_{t-1}^{(i)}, y_t, \theta)$ for each example in Section 5.4 are given below.

Linear Gaussian model. For this model the fully adapted filter is available where x_t is sampled from the optimal proposal $q(x_t|x_{t-1}, y_t, \theta) = p(x_t|x_{t-1}, y_t, \theta)$ (see

Doucet et al. (2000) for details) and the normalised posterior weights $w_t^{(i)} = 1/N$ are all equal. Explicitly the importance proposal is

$$\begin{aligned} \xi_t^{(i)} &\propto w_{t-1}^{(i)} \mathcal{N}(\alpha + \beta(\mu + \phi x_{t-1}^{(i)}), \beta^2\sigma^2 + \tau^2) \\ q^{\text{opt}}(x_t | x_{t-1}^{(i)}, y_t, \theta) &= \mathcal{N}(\Omega_t(\beta(y_t - \alpha)\tau^{-2} + (\mu + \phi x_{t-1}^{(i)})\sigma^{-2}), \Omega_t) \end{aligned}$$

where $\Omega_t = (\sigma^{-2} + \beta^2\tau^{-2})^{-1}$.

GARCH model. We again use the fully adapted filter for the GARCH model with noise where the importance proposal is

$$\begin{aligned} \xi_t^{(i)} &\propto w_{t-1}^{(i)} \mathcal{N}(0, \sigma_t^{2^{(i)}} + \tau^2) \\ q^{\text{opt}}(x_t | x_{t-1}^{(i)}, y_t, \theta) &= \mathcal{N}(\Omega_t y_t \tau^{-2}, \Omega_t) \end{aligned}$$

and $\Omega_t = (\sigma_t^{-2^{(i)}} + \tau^{-2})^{-1}$.

Stochastic volatility model with leverage. We use the importance proposal described by Omori et al. (2007),

$$\xi_t^{(i)} \propto w_{t-1}^{(i)} \mathcal{N}(0, \exp(\mu_t^{(i)}))$$

$$q(x_t | x_{t-1}^{(i)}, y_t, \theta) = \mathcal{N}(\mu_t^{(i)}, (1 - \rho^2)\sigma^2)$$

where $\mu_t^{(i)} = \mu + \phi(x_{t-1}^{(i)} - \mu) + \rho\sigma \exp(-x_{t-1}^{(i)}/2) y_t$.

Algorithm	Particles	Observations	Inefficiency	
			Min	Max
pMALA	2000	5000	203.99	903.78
	500	5000	383.95	1066.79
	2000	2000	64.70	371.09
	500	2000	79.13	384.46
	2000	1000	39.26	50.20
	500	1000	46.58	58.95
Poy. $\mathcal{O}(N)$	2000	5000	236.44	962.09
	500	5000	404.95	1197.02
	2000	2000	69.86	384.98
	500	2000	92.83	397.54
	2000	1000	41.37	57.88
	500	1000	47.12	64.83

Table 5.4.2: Linear Gaussian example. Comparison of the efficiency of pMALA and Poyiadjis $\mathcal{O}(N)$ MALA. Particle approximations are based on 500 and 2000 particles over datasets of length $T = 1000, 2000, 5000$.

		PMMH	pMALA
Acc. rate		0.15	0.33
Ineff	logit(ϕ)	35.01	**26.72**
	log(μ)	**38.42**	38.61
	logit(γ)	30.50	**21.24**
	log(τ)	32.80	**25.28**
SJD (x10^{-2})	logit(ϕ)	3.38	**3.79**
	log(μ)	1.13	**1.44**
	logit(γ)	10.06	**12.47**
	log(τ)	0.17	**0.18**

Table 5.4.3: GARCH example. Comparison of the inefficiency and squared jump distance of PMMH and pMALA. Bold font indicates the best algorithm in terms of inefficiency and squared jump distance for each parameter.

		PMMH	pMALA
Acc. rate		0.09	0.19
Ineff	μ	46.22	**43.47**
	logit(ϕ)	66.26	**51.29**
	log(σ)	67.88	**36.42**
	atanh(ρ)	73.33	**23.99**
SJD (x10^{-3})	μ	1.27	**1.32**
	logit(ϕ)	12.91	**13.24**
	log(σ)	2.91	**3.29**
	atanh(ρ)	31.98	**32.87**

Table 5.4.4: Stochastic volatility example. Comparison of the inefficiency and squared jump distance of PMMH and pMALA. Bold font indicates the best algorithm in terms of inefficiency and squared jump distance for each parameter.

Chapter 6

Sequential Monte Carlo Methods for State and Parameter Estimation in Abruptly Changing Environments

Abstract

This chapter develops a novel sequential Monte Carlo (SMC) approach for joint state and parameter estimation that can deal efficiently with abruptly changing parameters which is a common case when tracking manoeuvring targets. The approach combines Bayesian methods for dealing with changepoints with methods for estimating static parameters within the SMC framework. The result is an approach which adaptively estimates the model parameters in accordance with changes to the target's trajectory.

The developed approach is compared against the Interacting Multiple Model (IMM) filter for tracking a manoeuvring target over a complex manoeuvring scenario with nonlinear observations. In the IMM filter a large combination of models is required to account for unknown parameters. In contrast, the proposed approach circumvents the combinatorial complexity of applying multiple models in the IMM filter through Bayesian parameter estimation techniques. The developed approach is validated over complex manoeuvring scenarios where both the system parameters and measurement noise parameters are unknown. Accurate estimation results are presented.

6.1 Introduction

State and parameter estimation for nonlinear systems is a challenging problem which arises in many practical areas, such as target tracking, control and communication systems, biological systems and many others. The main methods for state and parameter estimation or for parameter estimation only can be classified into two broad groups (Kantas, 2009; Doucet et al., 2009): Bayesian and Maximum Likelihood (ML) methods. Such methods may also be categorized as *online* or *offline* depending on whether the data are processed sequentially as new observations become available, or processed in batches of observations. In ML estimation the optimal solution reduces to finding the estimate which maximises the marginal likelihood of the observed data. The Bayesian approach, however, considers the parameters as random variables which are updated recursively using prior knowledge of the parameters (if available) and the

measurement likelihood function. The approach proposed in this chapter is an on-line Bayesian approach which uses sequential Monte Carlo (SMC) techniques.

Early attempts to solve the problem of estimating the parameters online involved selecting a prior distribution for the parameters and augmenting the state vector to include the unknown parameters. The parameters can then be estimated using the same filtering technique that is applied to the state. However, through successive time steps this approach quickly leads to particle degeneracy of the parameter space. The fixed nature of the parameters means that the particles which are sampled from the initial prior distribution do not vary with time, thus the same set of particles will be resampled with replacement from one time step to the next, reducing the number of unique particles, eventually resulting in multiple copies of the same particle. This creates a point mass approximation of the marginal posterior parameter distribution.

One solution to this problem is to perturb particles by adding artificial noise (Gordon et al., 1993). However, naively adding noise at each iteration can lead to overly diffuse distributions for the parameters, relative to the true posterior distribution (Liu and West, 2001). An improved and related approach is the Liu and West (2001) filter. This filter uses *kernel density estimation* to estimate the posterior distribution of the parameters, and in particular the idea of shrinkage to avoid producing overly-diffuse approximations. An alternative approach to combat particle degeneracy is to use MCMC moves to sample new parameter values at each iteration. For some models this can be implemented efficiently, in an on-line setting, through the use of sufficient statistics (Fearnhead, 2002; Storvik, 2002). This class of methods has been recently termed *particle learning*, (Carvalho et al., 2010).

Whilst these methods can work well with static parameters, the case with dynamically changing parameters remains still unresolved. Therefore, we are considering applications with time-varying parameters, and especially the cases where the parameter values can change abruptly at a small set of time-points (Whiteley et al., 2011). A motivating application is in target tracking, where a manoeuvring target typically has "periods/segments" of high and low manoeuvrability. The parameters, such as the turn-rate of a model for the target's dynamics will be constant within a segment but different between segments. This can be modelled through a time-varying parameter, but under the constraint that the parameter values are piecewise constant functions of time. We shall refer to this scenario as models with *time-varying parameters* in the sequel.

Previous approaches to this problem include the jump Markov linear (JML) filter (Doucet et al., 2001), where the parameters evolve according to a finite state Markov chain and the Interacting Multiple Model filter of Blom and Bar-Shalom (1988).

In the IMM filter, numerous models are used (e.g. models for constant velocity and coordinated turn), each of which permit different fixed parameters, allowing the filter to switch between models depending on the motion of the target. The IMM filter has proven to be very successful for tracking highly manoeuvrable targets. However, the reliability of the IMM filter is dependent on the number and choice of models.

The IMM filter applies several proposed models (e.g. models for constant velocity and coordinated turn), each of which permit different fixed parameters, allowing the filter to account for various possible target behaviours. The IMM filter then merges the estimates of the various models based on their respective likelihood values to produce

a single estimate of the target's state. This filter has proven to be very successful for tracking highly manoeuvrable targets. However, the reliability of both the JML filter and the IMM filter are dependent on the *a priori* tuning of the filters as neither of these filters aim to estimate the unknown parameters online. They also suffer a curse-of-dimensionality, if we wish to account for multiple unknown parameters, then the number of models required increases exponentially with the number of parameters.

The proposed approach accounts for time-varying parameters using changepoints, and then combining SMC approaches for changepoint models (Fearnhead and Liu, 2007; Yildirim et al., 2012) with the standard SMC approaches for estimating static parameters (Carvalho et al., 2010; Liu and West, 2001). We call the resulting approach *adaptive parameter estimation*. It allows learning of parameters within segments between changepoints, and also allows the parameter estimates to adapt and learn new values once a changepoint has occurred. Preliminary results were reported in Nemeth et al. (2012a) and Nemeth et al. (2012b). This chapter refines further the adaptive parameter estimation filter and presents a comparison with the IMM algorithm for complex manoeuvring target scenarios.

The rest of the chapter is organised as follows. Section 6.2 presents the Bayesian formulation of the joint state and parameter estimation problem. Section 6.3 describes Bayesian approaches for joint state and parameter estimation. Section 6.4 presents the novel adaptive estimation algorithm. Section 6.5 evaluates the performance of the developed approach over two challenging scenarios with a manoeuvring target. Finally, Section 6.6 generalises the results and discusses future work.

6.2 Bayesian filtering

A state space model can be defined by two stochastic processes X_t and Y_t. The process X_t is referred to as a *hidden* or latent Markov process representing the state of interest at discrete time t, which takes values on the measurable space $\mathcal{X} \subseteq \mathbb{R}^{n_x}$. The stochastic process Y_t represents the observation process which takes values on the observation space $\mathcal{Y} \subseteq \mathbb{R}^{n_y}$, where observations are assumed to be dependent only on the current state X_t and independent of previous states $X_{1:t-1}$, where $X_{1:t-1} = \{X_1, X_2, \ldots, X_{t-1}\}$. We also assume that these stochastic processes are conditional upon the parameter vector θ, and that there exists a prior distribution, $p(\theta)$, for the parameter vector. The general state space model is characterised by the densities:

$$X_t|\{x_{0:t-1}, y_{1:t-1}\} \sim p(x_t|x_{t-1}, \theta), \tag{6.2.1}$$

$$Y_t|\{x_{0:t}, y_{1:t-1}\} \sim p(y_t|x_t, \theta), \tag{6.2.2}$$

where the state model is conditional only on the previous state and the observations y_t are independent of previous observations conditional only on the state x_t at time t. Here $y_{1:t-1}$ denotes the measurements from time 1 to time $t-1$.

In filtering, the aim is to estimate the hidden state at time point t given a sequence of observations. This process requires the evaluation of the posterior probability density function $p(x_t, \theta|y_{1:t})$ of the hidden state vector and parameter vector conditional on the observations. Using Bayesian estimation techniques it is possible to evaluate the posterior density recursively by first predicting the next state

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

and then updating this prediction to account for the most recent observation y_t,

$$p(x_t, \theta|y_{1:t}) = \frac{p(y_t|x_t, \theta)p(x_t, \theta|y_{1:t-1})}{p(y_t|y_{1:t-1}, \theta)}, \tag{6.2.3}$$

where

$$p(y_t|y_{1:t-1}, \theta) = \int p(y_t|x_t, \theta)p(x_t, \theta|y_{1:t-1})dx_t. \tag{6.2.4}$$

is the normalising constant. See Arulampalam et al. (2002) for a full details of this derivation.

Determining an analytic solution for the posterior distribution (6.2.3) is generally not possible due to the normalising constant (6.2.4) being intractable. One exception is when the state space is finite or linear-Gaussian in which case an analytic solution can be found using a Kalman (1960) filter. Generally, it is necessary to create an approximation of the posterior distribution, one such approach is through sequential Monte Carlo methods, also known as particle filters.

Particle filters present a method for approximating a distribution using a discrete set of N samples/particles with corresponding weights $\{x_t^{(i)}, \theta^{(i)}, w_t^{(i)}\}_{i=1}^N$ which create a random measure characterising the posterior distribution $p(x_t, \theta|y_{1:t})$. The empirical distribution given by the particles and weights can then be used to approximate (6.2.3) as

$$p(x_t, \theta|y_{1:t}) \approx \sum_{i=1}^{N} w_t^{(i)} \delta((x_t, \theta) - (x_t^{(i)}, \theta^{(i)})), \tag{6.2.5}$$

where $\delta(\cdot)$ is the Dirac delta function and each pair of particles $x_t^{(i)}$ and $\theta^{(i)}$ is given a weight $w_t^{(i)}$.

Using the empirical posterior distribution (6.2.5) as an approximation to the true posterior distribution $p(x_t, \theta|y_{1:t})$ it is possible to recursively update the posterior

probability density by *propagating* and *updating* the set of particles. The particles are propagated according to the dynamics of the system to create a predictive distribution of the hidden state at the next time step. These particles are then updated by weighting each particle based on the newest observations using principles from importance sampling (Arulampalam et al., 2002). Particle filtered approximations display inherent particle degeneracy throughout time due to an increase in the variance of the importance weights (Kong et al., 1994). A popular solution to this problem is to discard particles with low (normalised) weights and duplicate particles with high (normalised) weights by using a resampling technique (Gordon et al., 1993). Resampling the particles introduces Monte Carlo variation which produces poorer state estimation in the short term, but preserving particles with higher importance weights will provide greater stability for the filter and produces better future estimates. There are several approaches to resampling particles, the simplest being simple multinomial resampling. However, improved resampling strategies such as stratified resampling (Carpenter et al., 1999) can minimise the introduced Monte Carlo variation (see Douc and Cappé (2005) for a review of resampling strategies). In this chapter we will use the systematic resampling technique (Kitagawa, 1996) which minimises Monte Carlo variation and runs in $\mathcal{O}(N)$ time.

The next section describes important Bayesian approaches for state and parameter estimation: the auxiliary particle filter (APF) (Pitt and Shephard, 1999) and particle learning techniques (Carvalho et al., 2010; Storvik, 2002; Liu and West, 2001) which we use as a starting point to develop a novel adaptive Bayesian approach for state and parameter estimation.

6.3 Bayesian state and parameter estimation

6.3.1 Auxiliary particle filter

The original particle filter proposed by Gordon et al. (1993) suggests that the state particles $\{x_t^{(i)}\}_{i=1}^N$ should be sampled from the transition density $p(x_t|x_{t-1},\theta)$ and then weighted against the newest observation, which we shall refer to as *propagate - resample*. However, following this approach can lead to poor state estimates as the particles which are sampled from the transition density do not take account of the newest observations y_t. Ideally the state particles $x_t^{(i)}$ would be sampled from the optimal importance distribution $p(x_t|x_{t-1},y_t,\theta)$, which can be proven to be optimal (Doucet et al., 2000) in the sense that when applied it will minimise the variance of the importance weights. Sampling from the optimal importance distribution is generally not possible due to reasons of intractability. The auxiliary filter as proposed by Pitt and Shephard (1999), offers an intuitive solution to this problem by resampling particles based on their predictive likelihood $p(y_t|x_{t-1},\theta)$, thus accounting for the newest observations y_t before the particles are propagated. This method can be viewed as a *resample - propagate* filter.

This filter can be considered as a general filter from which simpler particle filters are derived as special cases. Consider a modified posterior density $p(x_t,\theta,k|y_{1:t})$ of both state x_t, parameter θ and auxiliary variables k, where k is the index of the particle at $t-1$. Applying Bayes theorem it can be shown that up to proportionality the target distribution is given by

$$p(x_t,\theta,k|y_{1:t}) \propto p(y_t|x_t,\theta^{(k)})p(x_t|x_{t-1}^{(k)},\theta^{(k)})w_{t-1}^{(k)}, \tag{6.3.1}$$

however, $p(y_t|x_t, \theta^{(k)})$ is unavailable so instead we can sample from the proposal distribution

$$q(x_t, \theta, k|y_{1:t}) \propto p(y_t|g(x_{t-1}^{(k)}), \theta^{(k)})p(x_t|x_{t-1}^{(k)}, \theta^{(k)})w_{t-1}^{(k)}$$

where $g(x_{t-1}^{(k)})$ characterises x_t given $x_{t-1}^{(k)}$, usually we choose $g(x_{t-1}^{(k)}) = \mathbb{E}[X_t|x_{t-1}^{(k)}, \theta^{(k)}]$. Estimates of the posterior density $p(x_t, \theta|y_{1:t})$ are given from the marginalised form of the density $p(x_t, \theta, k|y_{1:t})$ by omitting the auxiliary variable. Finally the importance sampling weights which are given by the ratio of the target and proposal distributions, simplify to

$$w_t \propto \frac{p(y_t|x_t, \theta^{(k)})}{p(y_t|g(x_{t-1}^{(k)}, \theta^{(k)}))}.$$

6.3.2 Particle learning

Gilks and Berzuini (2001) proposed a Bayesian approach to parameter estimation based on Markov chain Monte Carlo (MCMC) steps, where the entire history of the states and the observations is used to update the vector of unknown parameters $p(\theta|x_{0:t}, y_{1:t})$. The complexity of this approach grows in time and it suffers from the curse of dimensionality (Bengtsson et al., 2008).

Sampling parameters from the posterior parameter distribution $p(\theta|x_{0:t}, y_{1:t})$ becomes computationally more difficult as the time t increases. For some models a solution to this problem is to summarise the history of the states $x_{0:t}$ and observations $y_{1:t}$ via a set of low-dimensional sufficient statistic s_t (Fearnhead, 2002; Storvik, 2002). We define s_t to be *sufficient statistic* if all the information from the states and observations can be determined through it, (i.e. $p(\theta|x_{0:t}, y_{1:t}) = p(\theta|s_t)$). The

sufficient statistic should be chosen such that it can be updated recursively as new states and observations become available $s_t = \mathcal{S}_t(s_{t-1}, x_t, y_t)$. It is possible to determine whether a function s_t is sufficient by the factorisation theorem (Lindgren, 1993), which states that a function s_t is sufficient if there exist functions $k_1(\cdot)$ and $k_2(\cdot)$ such that

$$p(\theta_t, x_{0:t}, y_{1:t}) = k_1(\theta_t, \mathcal{S}_t(s_{t-1}, x_t, y_t))k_2(x_{0:t}, y_{1:t}). \tag{6.3.2}$$

By the factorisation theorem, s_t is a valid sufficient statistic if the parameters θ_t, are depend on x_t and y_t only via $\mathcal{S}_t$. Therefore, any sufficient statistic that follows the factorisation theorem can be used in particle learning.

The particle learning filter of Carvalho et al. (2010) can be viewed as an extension to the works of Fearnhead (2002) and Storvik (2002) where sufficient statistics are used to recursively update the posterior parameter distribution. Particle learning differs from previous sufficient statistic approaches in that it is based on the auxiliary particle filter which works within the *resample - propagate* framework. This approach produces better proposal distributions which more closely approximate the optimal proposal distribution, thus producing better state and parameter estimates. Particle learning also creates sufficient statistics for the states when possible. This reduces the variance of the sample weights and is often referred to as Rao-Blackwellisation.

The particle learning filter is summarised in Algorithm 13. The first step is a resampling step based on an approximation to the predictive density, where the first stage weight uses the state and parameter particles from the previous iteration.

Algorithm 13 Particle Learning Filter

Sample particles $\{x_{t-1}^{(i)}, \theta^{(i)}\}_{i=1}^{N}$ with weights $w_t^{(i)} \propto p(y_t|\mu_t^{(i)}, \theta^{(i)})$,

where $\mu_t^{(i)} = \mathbb{E}[x_t|x_{t-1}^{(i)}, \theta^{(i)}]$.

For $i = 1, \ldots, N$,

(a) Propagate state particles $x_t^{(i)} \sim p(\cdot|x_{t-1}^{(i)}, \theta^{(i)})$.

(b) Update sufficient statistics with the newest state estimate and observation

$s_t^{(i)} = \mathcal{S}_t(s_{t-1}^{(i)}, x_t^{(i)}, y_t)$.

(c) Sample new parameter values $\theta^{(i)} \sim p(\cdot|s_t^{(i)})$.

6.3.3 Liu and West filter

The implementation of the particle learning filter is dependent on producing a closed form conjugate prior for the parameters in order to define a sufficient statistic structure. For many complex models finding a closed form conjugate prior is not possible, therefore, it is necessary to approximate the posterior marginal parameter distribution in an alternative way. Liu and West (2001) propose an approach for approximating the posterior marginal parameter distribution through a kernel density approximation, where the marginal posterior parameter distribution is approximated as a mixture of multivariate Gaussian distributions.

Using Bayes theorem it is possible to determine the joint posterior distribution for the state and parameter $p(x_t, \theta_t|y_{1:t})$ as

$$\begin{aligned} p(x_t, \theta|y_{1:t}) &\propto p(y_t|x_t, \theta)p(x_t, \theta|y_{1:t-1}) \\ &\propto p(y_t|x_t, \theta)p(x_t|y_{1:t-1}, \theta)p(\theta|y_{1:t-1}), \end{aligned}$$

where the parameters are explicitly dependent on the observations.

The Liu and West filter can be interpreted as a modification of the artificial noise approach of Gordon et al. (1993) without the loss of information. The marginal posterior of the parameter distribution is represented as a mixture

$$p(\theta|y_{1:t-1}) \approx \sum_{i=1}^{N} w_{t-1}^{(i)} \mathcal{N}(\theta|m_{t-1}^{(i)}, h^2 V_{t-1}),$$

where $\mathcal{N}(\theta|m_{t-1}^{(i)}, h^2 V_{t-1})$ is a multivariate normal density with mean and variance,

$$m_{t-1}^{(i)} = a\theta^{(i)} + (1-a)\overline{\theta}, \tag{6.3.3}$$

$$V_{t-1} = \sum_{i=1}^{N} w_{t-1}^{(i)} (\theta^{(i)} - \overline{\theta})(\theta^{(i)} - \overline{\theta})^{\top}, \tag{6.3.4}$$

where $\overline{\theta} = \sum_{i=1}^{N} w_{t-1}^{(i)} \theta^{(i)}$ and V_{t-1} are the Monte Carlo posterior mean and variance of θ, respectively. The kernel smoothing parameter is denoted h^2 with shrinkage parameter $a = \sqrt{1-h^2}$ (discussed below) and $^{\top}$ as the transpose operation.

Standard kernel smoothing approximations suggest that kernel components should be centred around the parameter estimates, $m_{t-1}^{(i)} = \theta^{(i)}$. However, this approach can lead to overly-dispersed posterior distributions as the variance of the overall mixture is $(1+h^2)V_{t-1}$ and therefore larger than the true variance V_{t-1}. The overly dispersed approximation for the posterior $p(\theta|y_{1:t-1})$ at time $t-1$ will lead to an overly-dispersed posterior $p(\theta|y_{1:t})$ at time t, which will grow with time. West (1993) proposed a shrinkage step to correct for the over-dispersion by taking the kernel locations as in (6.3.3), where the shrinkage parameter a corrects for the over-dispersion by pushing particles $\theta^{(i)}$ back towards their overall mean. This results in a multivariate mixture distribution which retains $\overline{\theta}$ as the overall mean with correct variance V_{t-1}.

The Liu and West (2001) filter is summarised in Algorithm 14.

Algorithm 14 Liu and West Filter

Sample particles $\{x_{t-1}^{(i)}, \theta^{(i)}\}_{i=1}^{N}$ with weights $w_t \propto w_{t-1}^{(i)} p(y_t|\mu_t^{(i)}, m_{t-1}^{(i)})$,

where $\mu_t = \mathbb{E}[x_t|x_{t-1}^{(i)}, \theta^{(i)}]$ and $m_{t-1}^{(i)}$ is given in (6.3.3).

For $i = 1, \ldots, N$,

(a) Parameters are sampled from the kernel $\boldsymbol{\theta}^{(i)} \sim \mathcal{N}(\cdot|\boldsymbol{m}_{t-1}^{(i)}, h^2 \boldsymbol{V}_{t-1})$,

where $\boldsymbol{m}_{t-1}^{(i)}$ and $\boldsymbol{V}_{t-1}$ are given in (6.3.3) and (6.3.4).

(b) Propagate state particles $\boldsymbol{x}_t^{(i)} \sim p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}^{(i)}, \boldsymbol{\theta}^{(i)})$

(c) Assign weights $w_t^{(i)} \propto \frac{p(\boldsymbol{y}_t|\boldsymbol{x}_t^{(i)}, \boldsymbol{\theta}^{(i)})}{p(\boldsymbol{y}_t|\boldsymbol{\mu}_t^{(i)}, \boldsymbol{m}_{t-1}^{(i)})}$

6.4 Adaptive parameter estimation

Particle filters designed for parameter estimation, such as the Liu and West (2001) filter or particle learning filter (Carvalho et al., 2010) treat the estimated parameters as strictly fixed. In most cases this means that the marginal posterior distribution of the parameters will become increasingly concentrated around a single value as more observations are observed. As a result, if the parameters are time-varying then these filters often collapse, as they are unable to adapt to any abrupt change in the parameter.

For tracking applications it is more realistic to consider time-varying parameters where the parameters change abruptly at a set of unknown time-points. For example, in Section 6.5 we shall consider the case of tracking a manoeuvring target where the parameter vector which determines the target's trajectory changes depending on the target's manoeuvres. This problem can be solved by bringing together changepoint models with parameter estimation methods. In order to emphasise that the param-

eters are no longer static but are piecewise time-varying we change the parameter notation from θ to θ_t which now accounts for the time index t.

6.4.1 Changepoint approach

In some applications there are models whereby some of the parameters are fixed while others are time-varying. To account for such models we shall partition the parameter vector θ_t into fixed and time-varying parameters (see Section 6.4.5 for an example). This approach is advantageous for target tracking problems, where initially there may be several unknown parameters causing high variability in the state estimates. Over time this variability will decrease as the filter refines the estimate of the fixed parameters while still allowing the time-varying parameters to change according to the target's manoeuvres. This approach is preferable compared to model switching schemes such as the IMM filter which handles fixed and time-varying parameters in the same manner and therefore does not benefit from fixing some subset of the parameters over time.

The fixed parameters can be estimated using the techniques outlined in Section 6.3. As for the time-varying parameters, we focus on the case where the parameters are piecewise constant through time. Thus there will be a set of unknown points in time, known as changepoints, where the parameters can change. We use segments to denote the time-periods between changepoints, with parameters are assumed to be constant within each segment.

Rather than estimate these changepoints, and then perform inference conditional on a set of inferred changepoints, we introduce a probabilistic model for the location

of changepoints and perform inference by averaging over the resulting uncertainty in changepoint locations. For simplicity our prior model for changepoints is that there is a probability, β, of a changepoint at each time-point; and that changepoints occur independently. We assume that β is known. For a given β value, the expected segment length is $1/\beta$. Thus prior knowledge about the length of segments can be used to choose a reasonable value of β to use for a given application. In practice the data often gives strong indication about the location of changepoints, and thus we expect the results to be robust to reasonable choices of β. This is shown empirically in a simulated example (Section 6.5.2), where we observed that similar results are obtained for values of β varying by about an order of magnitude.

If there is a changepoint at time t, then new parameter values will be drawn from some distribution $p_{\theta_{t-1}}(\cdot)$ which depends on the current parameter values, θ_{t-1}. For ease of notation we consider distributions where we can partition the parameter, $\theta = (\theta', \theta'')$, into components that are fixed and those which change to a value independent of the current parameter value; though more general choices of distribution are possible. Thus we assume

$$p_{\theta_{t-1}}(\theta_t) = \delta(\theta'_t - \theta'_{t-1})p(\theta''_t), \tag{6.4.1}$$

where $\delta(\cdot)$ is the Dirac-delta function, and $p(\cdot)$ is some known density function. It is natural to assume that $p(\cdot)$ corresponds to the prior distribution for θ''_1. Thus the parameter dynamics can be described as

$$\theta_t = \begin{cases} \theta_{t-1} & \text{with probability } 1-\beta, \\ \gamma_t & \text{with probability } \beta, \end{cases} \tag{6.4.2}$$

where $\gamma_t \sim p_{\theta_{t-1}}(\cdot)$ represents the new parameter values.

6.4.2 SMC inference for time-varying parameters

It is straightforward to implement SMC inference under our changepoint model for parameters, whereby we simulate parameter values from (6.4.2) as part of the state update at each iteration. However this naive implementation can be improved upon using the ideas behind the APF filter to update the prior probability of a changepoint β with the newest observations y_t. Consider the posterior distribution $p(x_t, \theta_t, k|y_{1:t})$ which from (6.4.2) now takes account of the potentially new parameter vector γ_t. Only one of the parameter vectors θ_{t-1} or γ_t is chosen with probabilities $1-\beta$ and β, respectively. Therefore our posterior can be written as

$$\begin{aligned} p(x_t, \theta_t, k|y_{1:t}) &\propto (1-\beta)p(y_t|x_t, \theta_{t-1}^{(k)})p(x_t|x_{t-1}^{(k)}, \theta_{t-1}^{(k)})\delta(\theta_t - \theta_{t-1}^{(k)})w_{t-1}^{(k)} \\ &+ \beta p(y_t|x_t, \theta_t^{(k)})p(x_t|x_{t-1}^{(k)}, \theta_t^{(k)})p_{\theta_{t-1}^{(k)}}(\theta_t^{(k)})w_{t-1}^{(k)}. \end{aligned}$$

Using the auxiliary particle filter outlined in Section 6.3.1 it is possible to sample from this posterior distribution with an appropriate proposal distribution using the *resample-propagate* approach.

At time $t-1$ the posterior is represented by a set of equally-weighted particles $\{x_{t-1}^{(i)}, \theta_{t-1}^{(i)}\}_{i=1}^N$. Each particle is given a weight proportional to its predictive likelihood, corresponding to either a changepoint or no changepoint. For N particles this leads to $2N$ weights where for $i = 1, \ldots, N$

$$w_{t,1}^{(i)} \propto p(y_t|\mu_t^{(i)}, \theta_{t-1}^{(i)}), \text{ where } \mu_t^{(i)} = \mathbb{E}[x_t|x_{t-1}^{(i)}, \theta_{t-1}^{(i)}]$$

and

$$w_{t,2}^{(i)} \propto p(y_t|\mu_t^{(i)}, \gamma_t^{(i)}) \text{ with } \gamma_t^{(i)} \sim p_{\theta_{t-1}^{(i)}}(\cdot).$$

The first of these weights is an estimate of the probability of y_t given the value of the ith particle at time $t-1$ when there is no changepoint. The second weight corresponds to there being a changepoint with new parameters $\gamma_t^{(i)}$. It is possible that this problem could be reformulated using the auxiliary filter so that only one particle is propagated, thus reducing the number of particles.

Next, resampling is performed, where N particles are sampled from $2N$ particles with probabilities proportional to the union of $\{(1-\beta)w_{t,1}^{(i)}\}_{i=1}^N$ and $\{\beta w_{t,2}^{(i)}\}_{i=1}^N$. If the ith index is sampled from the first set of the union, then the particle corresponding to the state and current parameters for index i are propagated. If the ith index is sampled from the second set, the state of the corresponding particle is propagated together with the new parameter value, $\gamma_t^{(i)}$. Finally, the appropriate weights for the particles are calculated as in the auxiliary particle filter. A similar idea was considered by Whiteley et al. (2010).

Within this approach it is possible to use either the particle learning filter (Algorithm 13), the Liu and West filter (Algorithm 14) or both to update the parameter values in the segments between changepoints. The parameter vector can be partitioned as follows $\theta_t = (\xi_t^\top, \zeta_t^\top)^\top$ where ξ_t are parameters to be updated using the particle learning filter and ζ_t are parameters updated using the Liu and West filter. This is a slight abuse of notation as θ_t is further partitioned into fixed and time-varying parameters. It is possible to resolve this problem by partitioning ξ_t and ζ_t

into fixed and time-varying parameters. In the sequel, we shall use the standard particle learning (Alg. 13) and Liu and West (Alg. 14) algorithms to estimate the fixed parameters. However, as these parameters are treated as fixed, alternative strategies for estimating these fixed parameters could be used. For example, the gradient ascent algorithm proposed in Chapter 4.

6.4.3 Applying the Liu and West filter to time-varying parameters

At time $t-1$ parameters ζ_{t-1} with no sufficient statistic structure can be updated with the Liu and West filter by first estimating the kernel locations $m_{t-1}^{(i)} = a\zeta_{t-1}^{(i)} + (1-a)\overline{\zeta}_{t-1}$, where a is the shrinkage parameter. The ith kernel location is propagated and the parameters are updated as $\zeta_t^{(i)} \sim \mathcal{N}(\cdot|m_{t-1}^{(i)}, h^2 V_{t-1})$ if the index $i \in \{1,\ldots,N\}$, where V_{t-1} is given in (6.3.4). Alternatively, if $i \in \{N+1,\ldots,2N\}$ then $\zeta_t^{(i)}$ is drawn from the appropriate part of distribution (6.4.1).

6.4.4 Applying particle learning to time-varying parameters

The particle learning filter can be viewed as a special case of the Bayesian parameter estimation approach where the parameters ξ_t have a conjugate prior distribution which can be recursively updated via the sufficient statistics s_t. The sufficient statistics are updated differently depending on whether the parameters are fixed or time-varying. For the case of the fixed parameters the sufficient statistics are updated as described in Section 6.3.2, where $s_t^{(i)} = \mathcal{S}(s_{t-1}^{(i)}, x_t^{(i)}, y_t)$ for $i \in \{1,\ldots,2N\}$.

If we assume that ξ_t is a time-varying parameter then the parameters are updated at time t by sampling $\xi_t^{(i)} \sim p(\cdot|s_t^{(i)})$, where $s_t^{(i)} = \mathcal{S}(s_{t-1}^{(i)}, x_t^{(i)}, y_t)$ if no changepoint is detected (i.e. the resampling index $i \in \{1, \ldots, N\}$). Alternatively, if there is a changepoint and $i \in \{N+1, \ldots, 2N\}$ then the sufficient statistics are reset to their initial prior values, $s_{t-1} = s_0$ (see Section 6.4.5 for an example). If some parameters are fixed and others time-varying then the sufficient statistics for each parameter are updated accordingly.

Applying the Liu and West filter and the particle learning filter to the estimation of time-varying parameters produces an efficient filter for both state and parameter estimation which we refer to as the adaptive parameter estimation (APE) filter. Algorithm 15 presents an instance of the filter where the parameters ζ_t are assumed to be time-varying and the parameters ξ_t are assumed to be fixed. This setting conforms with the scenario given in the performance validation section.

6.4.5 Target tracking motion and observation models

We present a motivating example from the target tracking literature to highlight the importance of estimating time-varying parameters. The model considered is used to track a target which moves within the $x-y$ plane, where the target's state is a vector of position and velocity $\underline{x}_t = (x_t, \dot{x}_t, y_t, \dot{y}_t)^\top$.

The motion of the target is modelled using a coordinated-turn model (Rong Li and Bar-Shalom, 1993) of the form

$$\underline{x}_t = F^\top \underline{x}_{t-1} + \Gamma \nu_t$$

Algorithm 15 Adaptive Parameter Estimation Filter

Step 1: For $i = 1, \ldots, N$

(a) Update current parameter values $\xi_{t-1}^{(i)} \sim p(\cdot|s_{t-1}^{(i)})$, $m_{t-1}^{(i)} = a\zeta_{t-1}^{(i)} + (1-a)\overline{\zeta}_{t-1}$ and set the parameter vector $\theta_t^{(i)} = [{\xi_{t-1}^{(i)}}^\top, {m_{t-1}^{(i)}}^\top]^\top$.

(b) Sample new parameter particles $\gamma_t^{(i)} \sim p_{\theta_{t-1}^{(i)}}(\cdot)$.

(c) Calculate pre-weights $w_{t,1}^{(i)} \propto p(y_t|\mu_t^{(i)}, \theta_t^{(i)})$ and $w_{t,2}^{(i)} \propto p(y_t|\mu_t^{(i)}, \gamma_t^{(i)})$,

where $\mu_t^{(i)} = \mathbb{E}[x_t|x_{t-1}^{(i)}, \theta_{t-1}^{(i)}]$.

Step 2: For $i = 1, \ldots, N$.

(a) Sample indices k^i from $\{1, \ldots, 2N\}$ with probabilities $\{(1-\beta)w_{t,1}^{(i)}\}_{i=1}^N$ and $\{\beta w_{t,2}^{(i)}\}_{i=N+1}^{2N}$.

No Changepoint:

For $k^i \in \{1, \ldots, N\}$.

(b) Update parameters $\zeta_t^{(i)} \sim \mathcal{N}(\cdot|m_{t-1}^{(k^i)}, h^2 V_{t-1})$, where V_{t-1} is given by (6.3.4).

(c) Set parameters $\theta_t^{(i)} = [{\xi_{t-1}^{(k^i)}}^\top, {\zeta_t^{(i)}}^\top]^\top$ and sufficient statistics $s_{t-1}^{(i)} = s_{t-1}^{(k^i)}$.

(d) Propagate states $x_t^{(i)} \sim p(\cdot|x_{t-1}^{(k^i)}, \theta_t^{(i)})$ and assign weights $w_t^{(i)} \propto \frac{p(y_t|x_t^{(i)}, \theta_t^{(i)})}{w_{t,1}^{(k^i)}}$.

Changepoint:

For $k^i \in \{N+1, \ldots, 2N\}$.

(b) Propagate states $x_t^{(i)} \sim p(x_t|x_{t-1}^{(k^i)}, \gamma_t^{(k^i)})$.

(c) Set parameters $\theta_t^{(i)} = \gamma_t^{(k^i)}$ and assign weights $w_t^{(i)} \propto \frac{p(y_t|x_t^{(i)}, \theta_t^{(i)})}{w_{t,2}^{(k^i)}}$.

Resample particles $\{x_t^{(i)}, s_{t-1}^{(i)}, \zeta_t^{(i)}\}_{i=1}^N$ with replacement with probabilities $\{w_t^{(i)}\}_{i=1}^N$ to obtain the particle set $\{x_t^{(i)}, s_{t-1}^{(i)}, \zeta_t^{(i)}\}_{i=1}^N$ with weights $1/N$.

Update sufficient statistics $s_t^{(i)} = \mathcal{S}(s_{t-1}^{(i)}, x_t^{(i)}, y_t)$.

where,

$$F = \begin{bmatrix} 1 & \frac{\sin \omega_t \Delta T}{\omega_t} & 0 & -\frac{1-\cos \omega_t \Delta T}{\omega_t} \\ 0 & \cos \omega_t \Delta T & 0 & -\sin \omega_t \Delta T \\ 0 & \frac{1-\cos \omega_t \Delta T}{\omega_t} & 1 & \frac{\sin \omega_t \Delta T}{\omega_t} \\ 0 & \sin \omega_t \Delta T & 0 & \cos \omega_t \Delta T \end{bmatrix},$$

$$\Gamma = \begin{bmatrix} \frac{\Delta T}{2} & 0 \\ \Delta T & 0 \\ 0 & \frac{\Delta T}{2} \\ 0 & \Delta T \end{bmatrix}$$

and system noise ν_t is modelled as a zero mean Gaussian white noise process $\mathcal{N}(0, \eta^2 I_2)$.

This model simplifies to the constant velocity model when $\omega_t = 0$. The model is flexible and able to account for the motion of highly manoeuvrable targets, where the target may change direction abruptly and switch between periods of high and low manoeuvrability (see Figure 6.5.2 for a simulated trajectory).

Noisy nonlinear observations of the target in the form of a range and bearing measurement are taken by a fixed observer positioned at (s_x, s_y)

$$\underline{y}_t = \begin{bmatrix} \sqrt{(x_t - s_x)^2 + (y_t - s_y)^2} \\ \arctan((y_t - s_y)/(x_t - s_x)) \end{bmatrix} + \epsilon_t,$$

where the observation noise ϵ_t is a zero mean Gaussian white noise process with known covariance matrix R.

It is possible to use this model to track a manoeuvring target if we treat the turn rate parameter ω_t as a time-varying parameter and the remaining parameters

η^2 and R as fixed. This is an ideal scenario for the adaptive parameter estimation filter as it can easily handle both fixed and time varying parameters. In Section 6.5 a comparison of this filter with the IMM filter illustrates the benefit of treating fixed and time-varying parameters separately.

The APE filter can be applied to the target tracking model in the following way. The turn rate parameter ω_t appears non-linearly in the model and does not admit a sufficient statistic structure. We therefore estimate this parameter using the kernel density approach for time-varying parameters as outlined in Section 6.4.3. The noise variance parameters η^2 and R can be estimated via the set of sufficient statistics $s_t = (a_t, b_t, c_t, d_t, e_t, f_t)$ which is a vector of the parameters for the conjugate priors. The conjugate prior for η^2 is an inverse-gamma distribution $IG(a_t/2, b_t/2)$, where the sufficient statistics a_t and b_t are updated as $a_t = a_{t-1} + \dim(\underline{x}_t)$ and $b_t = b_{t-1} + (\underline{x}_t - F^\top \underline{x}_{t-1})^\top (\mathrm{diag}(\Gamma\Gamma^\top))^{-1}(\underline{x}_t - F^\top \underline{x}_{t-1})$. The conjugate prior for R is an inverse Wishart distribution. However, if we assume that the range and bearing measurements are uncorrelated then we can model their variances separately, where the range variance follows an inverse-gamma distribution $IG(c_t/2, d_t/2)$, with the sufficient statistics c_t and d_t which are updated as follows, $c_t = c_{t-1} + 1$ and $d_t = d_{t-1} + (\underline{y}_t[1] - \sqrt{(\underline{x}_t[1] - s_x)^2 + (\underline{x}_t[3] - s_y)^2})^2$ and the sufficient statistics e_t and f_t for the variance of the bearing measurements are updated similarly.

The example given in Section 6.5 treats the variances as fixed and therefore we do not need to reset the sufficient statistics for these parameters when a changepoint is detected. It is possible to allow one of the variances to change between segments by resetting a subset of the sufficient statistics. For example, it may be reasonable

to assume that the variance of observations R is fixed, but that ν^2 changes when the target performs a manoeuvre. The change in ν^2 can be accounted for by setting $a_t = a_0$ and $b_t = b_0$, thus the new variance parameter will be sampled from the initial prior distribution. The sufficient statistics will again be updated accordingly to estimate ν^2 as given above.

To summarise, the adaptive parameter estimation filter can be used to estimate fixed and time-varying parameters for models with both conjugate and non-conjugate parameter distributions. In the next section we will show how this approach works well when there are multiple unknown parameters with vague prior knowledge of their true values.

6.5 Performance validation

This section presents a comparison of the adaptive parameter estimation filter developed in Section 6.4 against the IMM filter. The filters' performance is validated on a simulated dataset taken from the coordinated turn model given in Section 6.4.5. The aim of the comparisons is to illustrate the improvement of the APE filter over the IMM filter as the number of unknown parameters increases. The accuracy of the algorithms is characterised by the relative root mean squared (RMS) representing the ratio, i.e. the IMM RMS error/APE RMS error. Results showing the filters' accuracy and computational time are given.

6.5.1 Testing scenario

A challenging testing scenario is considered in which the moving object performs complex manoeuvres consisting of abrupt turns followed by a straight line motion. The turn rate parameter $\omega_t \in (-20^\circ/s, 20^\circ/s)$ is unknown and is estimated in conjunction with the target's state vector. For variance parameters ν^2 and R the initial parameters for the conjugate priors are $\boldsymbol{s}_0 = (9, 15, 4, 5000, 4, 0.0025)$. A target track is simulated from the coordinated turn model over 400 time steps, with sampling period $\Delta T = 1s$. The turn rate parameter ω_t takes values $\{0, 3, 0, 5.6, 0, 8.6, 0, -7.25, 0, 7.25\}^\circ/s$ with changes occurring at times $\{60, 120, 150, 214, 240, 272, 300, 338, 360\}$, respectively. This set-up creates a highly dynamic target trajectory, where the target switches between periods of high and low manoeuvrability, as shown in Figure 6.5.2. The testing scenario is completed by specifying the system noise variance $\eta^2 = 2\text{m/s}^2$ and observation noise covariance matrix $R = \text{diag}(50^2\text{m}, 1^\circ)$. The trajectory is simulated with the initial state of the target $\underline{x}_1 = (30\text{km}, 300\text{m/s}, 30\text{km}, 0\text{m/s})^\top$ and observations taken from a fixed observer positioned at (55 km, 55km).

6.5.2 Choosing β

The accuracy of the APE filter is dependent on the choice of the *a priori* changepoint probability β. If β is large (close to 1) then the filter may struggle to estimate the parameters as it will introduce excess parameters from the diffuse prior $p_{\theta_1}(\gamma_t)$ when no changepoint has occurred. On the other hand, if β is too small then the filter will simplify to the standard Bayesian parameter estimation filter for static parameters,

and will struggle to handle time-varying parameters. Figure 6.5.1 gives the RMS error for the parameter ω_t using the APE filter with various choice for β using the simulated trajectory described in Section 6.5.3. The vertical lines correspond to the changepoints where the target performs a manoeuvre. In this scenario there are 9 changepoints over 400 time steps, therefore using the inverse of the average segment length we would expect $\beta \approx 0.025$ to give the lowest RMS error. The results show that setting $0.01 < \beta < 0.05$ will give the lowest RMS error, consistent with results from other simulated trajectories. The filter does not require that the changepoint probability β parameter is known exactly. In fact the filter appears to be robust to a range of β values. For example, when $\beta = 0.001$ the filter displays higher RMS error after a changepoint, this is to be expected as setting β close to 0 assumes there is no changepoint. However, even for such low values the filter is still able to track the target. This is in contrast to the Liu and West and particle learning filters which often collapse when used to estimate abruptly changing parameters (see Figure 6.5.2).

Estimating β within the filter using the particle learning or Liu West algorithm could be applied. However, this may cause the filter to struggle to estimate the changes effectively. Alternatively, β could be calibrated by running a few pilot filters with different β values and then use the value which maximises the model likelihood.

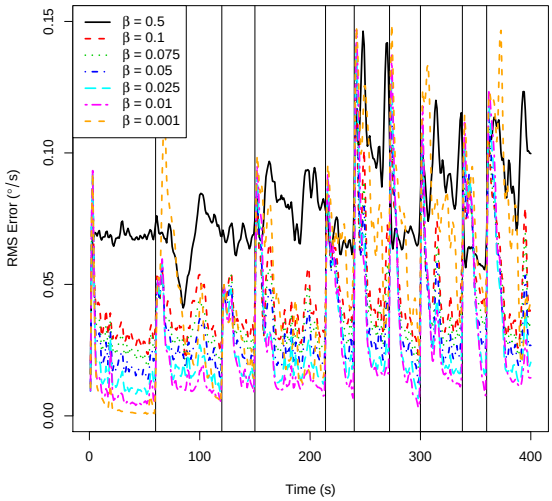


Figure 6.5.1: Root mean squared of the turn rate parameter from model (6.4.5) for various β values.

6.5.3 Estimation of the state vector, jointly with the turn rate.

The APE filter is compared with the IMM filter to estimate the state vector of a manoeuvring target. The main difference between these two approaches is in the way that each filter handles the unknown turn rate ω_t. The IMM attempts to account for the unknown, time-varying turn rate by selecting one model from a bank of potential models. The adaptive parameter estimation filter, on the other hand, estimates ω_t and is therefore not constrained by a finite set of potential models.

The APE filter is implemented with 5,000 particles and as there does not exist a conjugate prior for the turn rate parameter ω_t, the Liu and West procedure shall be used within Algorithm 15 to estimate this parameter. The smoothing parameter of the kernel density estimate is set to $h^2 = 0.01$ as recommended by Liu and West (2001),

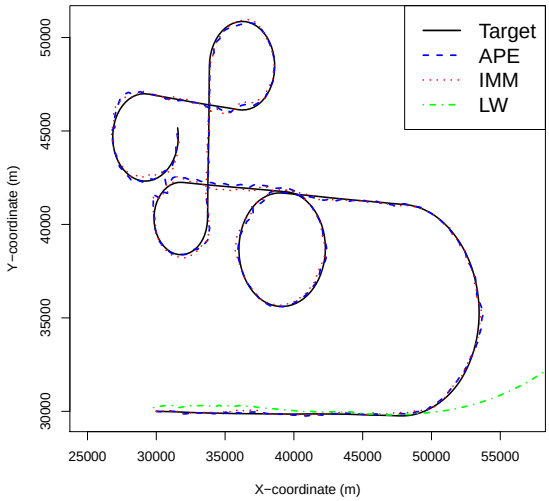


Figure 6.5.2: This Figure shows the simulated target trajectory and the estimated trajectories obtained by the APF, IMM and LW algorithms.

where the probability of a changepoint at any point in time is $\beta = 0.05$. The initial prior distribution for the turn rate parameter ω_t follows a non-informative uniform distribution over the range $[-20°, 20°]$. In this scenario the IMM filter is implemented using 20 and 60 coordinated turn models. The models differ in the choice of the parameters ω_t and η^2, where 20 or 60 equally spaced values of ω_t are sampled over the range $[-20°, 20°]$ and $\eta^2 = 2\text{m/s}^2$ when $\omega_t = 0$ and 2.5m/s^2 when the turn rate is non-zero to allow for greater ease of turn.

The transition probabilities between models of the IMM filter are balanced equally between all alternative models and sum to 0.05 with a 0.95 probability of no model transition. This parameter acts in a similar way to the β parameter of the APE filter and must also be tuned. For this example we have set the model transition probability to be equal to β to create a fair comparison. As the observation model is nonlinear

the IMM filter is implemented with an unscented Kalman filter (Julier and Uhlmann, 2004). In this setting the computational time required to run the IMM filters, relative to the adaptive parameter estimation filter, is 0.5 and 2.5 times greater for 20 and 60 models respectively. The filters are compared over 100 independent Monte Carlo runs.

Simulation results show that both the IMM and the adaptive parameter estimation filter are able to track the target well. However, if the standard Liu and West (LW) filter (Algorithm 14) with no adaptation is applied to this scenario then after the first manoeuvre, when the turn rate changes, the parameter estimated by the LW filter no longer matches the target's dynamics, which after a few time steps causes the filter to collapse (Fig. 6.5.2). The efficacy of the adaptive parameter estimation filter is dependent upon the accuracy of the parameter estimates. Figure 6.5.3 shows the estimates of the unknown turn rate given by the APE filter. The filter appears to estimate the turn rate well under difficult conditions. During long periods between manoeuvres the filter is able to produce reliable estimates of the turn rate parameter and update this estimate to account for changes in the target's dynamics.

Figure 6.5.4 gives the RMS error of several filters relative to the APE filter. It also displays a comparison to the auxiliary particle filter where θ is known. This comparison illustrates the importance and potential gains that are achievable by correctly estimating the unknown model parameters. Improvements in the accuracy of the IMM filter may be attained by tuning the filter to better match the dynamics of the target. However, with minimal tuning, the adaptive parameter estimation filter is able to track the target at least as well as the IMM and requires no prior knowledge

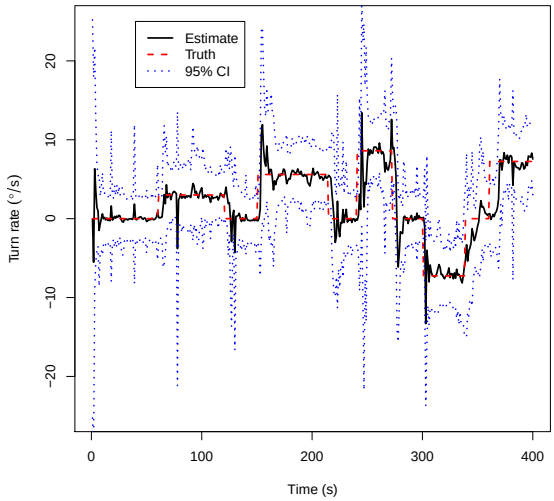


Figure 6.5.3: Estimated turn rate parameter (black solid line) with the APE filter versus the true parameter value (red dashed line)

of the target's dynamics.

The average RMS error over the trajectory for the following filters: "APE", "IMM 20 models", "IMM 60 models" and "APF filters" is, 81.41m, 110.23m, 92.97m and 61.86m, respectively. Compared to both IMM filters, the APE filter produces lower RMS error of the target's position. The benefit of the APE filter is most notable during longer segments between changepoints. This is to be expected as longer segments allow the APE filter to refine its estimate of the turn rate parameter. Increasing the number of models for the IMM filter can reduce the RMS error, but at an increase in computational complexity. In the next section we shall see that, computational complexity aside, increasing the number of models does not guarantee a reduction in RMS error.

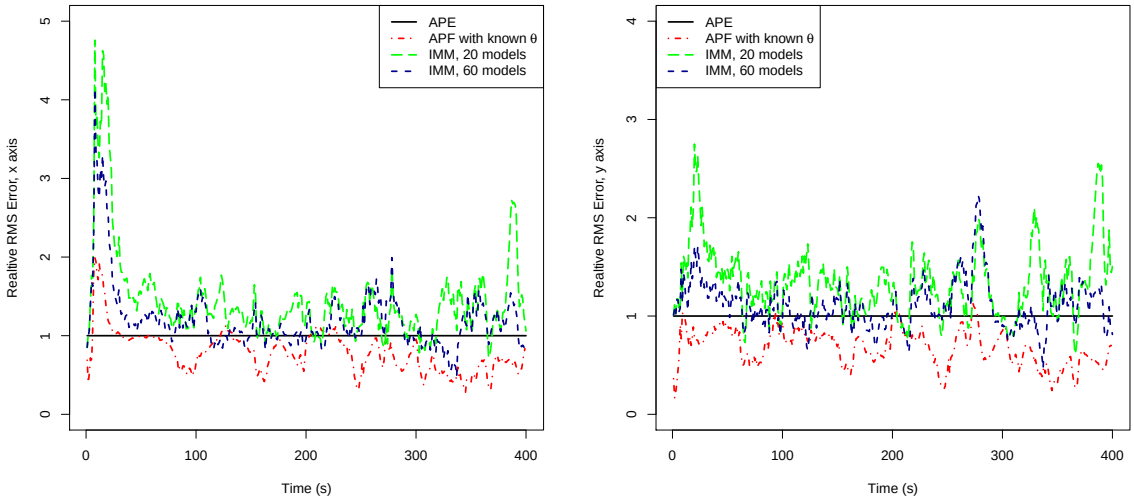


Figure 6.5.4: Relative RMS error (IMM RMS error/APE RMS error) of target position for the x and y axes, respectively (left x axis, right y axis.)

6.5.4 Estimation of the state vector, jointly with the turn rate, system and observation covariance parameters.

In scenarios where there are multiple unknown parameters it becomes increasingly difficult to design an effective IMM filter as increasing the number of unknown parameters requires an increase in the number of potential model combinations. Figure 6.5.5 displays the RMS error of the APE and IMM filters when the turn rate ω_t, system noise η^2 and observation covariance R parameters are unknown. This is an interesting problem as the turn rate parameter is treated as piecewise time-varying and the variances of the noise parameters are assumed to be fixed. In this setting the APE filter uses the Liu and West filter to estimate the turn rate parameter as in the last example and uses the particle learning filter to estimate the noise variances via

their sufficient statistics (see Section 6.4.5 for details).

Implementing the IMM filter becomes more complicated as the number of unknown parameters increases and the number of model combinations will also increase. If we assume that only ω_t and η^2 are unknown then one potential implementation of the IMM filter with 20 models would be $\omega_t \in [-20^\circ/s, -10^\circ/s, 0^\circ/s, 10^\circ/s, 20^\circ/s]$ and $\eta^2 \in [1.5\text{m/s}^2, 2\text{m/s}^2, 2.5\text{m/s}^2, 3\text{m/s}^2]$. Or using 60 models it would be possible to have 10 models for ω_t evenly sampled from the interval $[-20^\circ/s, 20^\circ/s]$ and 6 models for η^2. This quickly leads to a combinatorial problem where it becomes difficult to match the various model combinations required to cover the unknown parameters. Increasing the number of models from 20 to 60 incurs a 3 fold increase in computational time but only offers marginal increase in the number of model combinations.

Figure 6.5.5 gives the RMS error for the APE and IMM filters when 2 parameters are unknown $\{\omega_t, \eta^2\}$ and when 3 parameters are unknown $\{\omega_t, \eta^2, R\}$. The RMS error is plotted relative to the APE filter for 2 unknown parameters. For the case of 3 unknown parameters the IMM is implemented with 45 models combined from: 5 models for ω_t evenly sampled from the interval $[-20^\circ/s, 20^\circ/s]$, 3 models for $\eta^2 \in [2\text{m/s}^2, 2.5\text{m/s}^2, 3\text{m/s}^2]$ and 3 models for $R \in [\text{diag}(50^2\text{m}, 1^\circ), \text{diag}(25^2\text{m}, 2^\circ), \text{diag}(100^2\text{m}, 1^\circ)]$. The average RMS error for the following filters: "APE 2 unknowns", "APE 3 unknowns", "IMM models 20 2 unknowns", "IMM 60 2 unknowns" and the "IMM 45 models 3 unknowns" over the trajectory is, 82.79m, 101.63m, 138.93m, 127.33m and 155.65m, respectively. For the case of 2 unknown parameters, Figure 6.5.5 illustrates that increasing the number of models in the IMM filter (from 20 to 60) does not greatly improve state estimation given the significant increase in computational time.

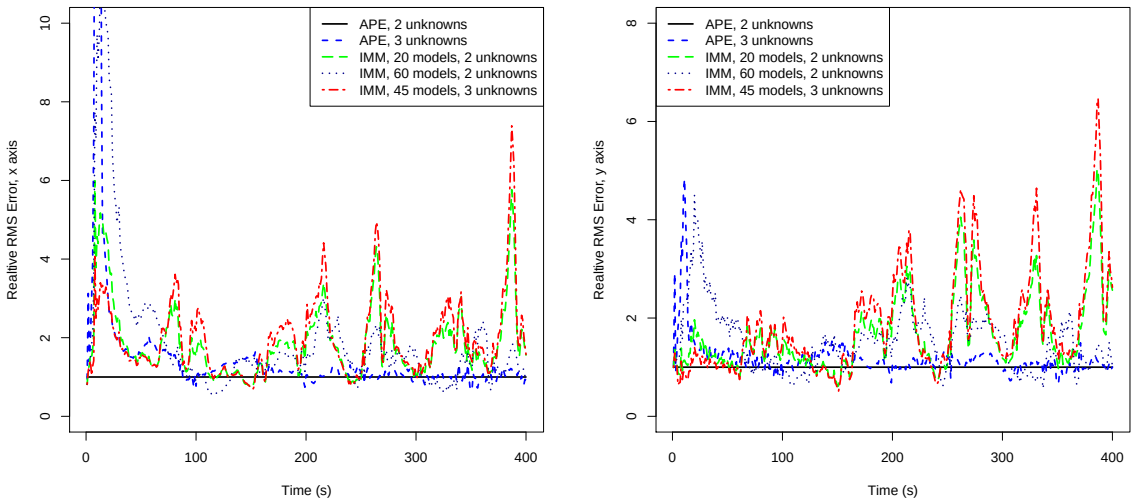


Figure 6.5.5: Relative RMS error of target position with multiple unknown parameters (left x axis, right y axis.)

In this scenario as the target is initially moving with almost constant velocity, the extra turn rate combinations are redundant, but once the target begins to manoeuvre the benefit of extra models is observed.

It is important to note that for the IMM filters the true noise variances are included as potential models, whereas for the APE filter, all of the parameters are truly unknown as the initial parameter values are sampled from their prior distributions. This explains why initially the RMS error of the APE filter with 3 unknown parameters is high in Figure 6.5.5. Interestingly, the RMS error of the APE filter for 3 unknown parameters approaches the levels observed for the case of 2 unknown parameters as the parameter estimates converge to their true values. This is not the case for the IMM filter as increasing the number of unknown parameters corresponds to a consistent increase in RMS error throughout time.

6.6 Conclusions

This chapter considers the difficult problem of joint state and parameter estimation of nonlinear and highly dynamic systems. The chapter presents a sequential Monte Carlo filter that is capable of estimating parameters with conjugate and non-conjugate structures, but most importantly, parameters which may be time-varying as in the case of tracking manoeuvring targets. The main advantage of the adaptive parameter estimation approach is its ability to provide quick estimation of the abruptly changing parameters from non-informative prior knowledge, and to do this for multiple unknown parameters. Its scalability to the case of estimating multiple unknown parameters is an advantage over filters such as the IMM which are based on a multiple model implementation. There has been little work in the literature to address the issue of calibrating target tracking models, this is an important and interesting problem which should be studied further.

One of the drawbacks of the particle learning approach is the requirement that the parameters follow a conjugate structure for the sufficient statistics. This limits the class of models to which particle learning can be applied. Recent work on the extended parameter filter (Erol et al., 2013) aims to overcome this problem by considering a Taylor series approximation to the parameters. A possible extension for future work would be to apply the extended parameter filter within the adaptive parameter estimation framework.

Chapter 7

Conclusions

7.1 Final remarks and contributions

This thesis addresses the challenging problem of parameter estimation for nonlinear state space models. Sequential Monte Carlo algorithms have been proposed as a method to tackle the problem of parameter estimation. These algorithms have a strong theoretical foundation and provide an efficient and highly accurate approach for estimating quantities of interest, such as the marginal likelihood.

In Chapter 4 this thesis proposes a computationally efficient method for estimating the score vector of state space models using sequential Monte Carlo algorithms. These particle approximations of the score have then been applied to maximum likelihood parameter estimation via the gradient ascent algorithm. It has been shown that compared to competing algorithms, which are either computationally costly, or display a quadratically increasing variance in the estimate of the score, our algorithm is linear in the number of particles in terms of cost and displays only linearly increasing

variance. It has also been shown that this gradient based approach to parameter estimation can be applied recursively to update the parameters as new observations arrive. Given the computational efficiency of the proposed algorithm, this type of parameter estimation approach could work well within a "big data" environment where new observations are arriving rapidly, such as in high frequency trading.

Chapter 5 considers Bayesian parameter estimation using a new proposal for the particle marginal Metropolis Hastings sampler, which we call particle MALA. This proposal is a particle approximation of the idealised Metropolis adjusted Langevin algorithm. It is shown in this chapter that using the computationally efficient particle approximation of the score, it is possible to create a proposal distribution for the parameters which takes account of the local geometry of the target density. This proposal is shown to increase the acceptance rate of the PMMH sampler compared to the standard random walk proposal and also produces a Markov chain with reduced autocorrelation.

In the case of tracking manoeuvrable targets, Chapter 6 proposes a new algorithm to estimate the changing model parameters. Previous approaches have used a model switching approach where the tracking algorithm switches between potential models to choose a model which best captures the targets behaviour. It is shown in this chapter that our algorithm, which rather than switching between competing models aims to learn the model parameters, produces improved target estimates compared to the switching model approach. This is tested on nonlinear tracking models where sequential Monte Carlo algorithms are used to infer the motion of the target based on nonlinear observations.

7.2 Future work and possible extensions

The area of parameter estimation for state space models is still an open area of research from both the maximum likelihood and Bayesian perspectives. In terms of maximum likelihood parameter estimation, the gradient based approach proposed in Chapter 4 uses particle approximations of the score vector. While the reduction in variance of the proposed method is shown empirically, it would be beneficial to prove that this method produces score estimates with only linearly increasing variance. Given the connections of this method to the fixed-lag smoother, it may be possible to derive bounds on the Lp error and bias similar to those given by Olsson et al. (2008) for the fixed-lag smoother. In which case it is expected that the Lp error and bias of this method are upper bounded by terms proportional to $T/\sqrt{N}$ and T/N respectively, where T is the length of the observation set and N is the number of particles. Therefore, while this method does introduce some bias, tuning the number of particles such that they are proportional to the length of the dataset will produce estimates of the score with only a small amount of bias, but with a significant reduction in variance.

Efficient proposal distributions for MCMC algorithms can lead to significant improvements over simpler proposals. It has been shown that using Hamiltonian Monte Carlo (Neal, 2010) and Riemann manifold (Girolami and Calderhead, 2011) methods, which account for the local geometry of the target within the proposal, can increase the Metropolis Hastings acceptance rate and reduce the autocorrelation of the Markov chain. In the particle MCMC context, proposals such as particle MALA can offer similar improvements. However, the particle MALA proposal is based on

first order Langevin dynamics, similar to the standard MALA algorithm, and does not account for the curvature of the target density. It may be possible to improve the particle MALA proposal by using a particle approximation of the observed information matrix, as discussed in Chapter 4, within the proposal. However, unlike the expected information matrix, the observed information matrix is not guaranteed to be positive definite and therefore using it within the proposal, in a similar fashion to the manifold MALA proposal (Girolami and Calderhead, 2011), would require care. It may be possible to work around this issue by employing matrix regularisation or mapping the observed information matrix onto some positive definite matrix. However, it is not clear whether such a proposal would preserve detailed balance, or produce significant improvements over the simpler particle MALA proposal.

Bibliography

Andrieu, C., Doucet, A., and Holenstein, R. (2010). Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342.

Andrieu, C., Doucet, A., Singh, S. S., and Tadic, V. B. (2004). Particle Methods for Change Detection, System Identification, and Control. *Proceedings of the IEEE*, 92(3):423–438.

Andrieu, C., Doucet, A., and Tadic, V. B. (2005). On-line parameter estimation in general state-space models. In *IEEE Conference on Decision and Control*, pages 332–337.

Andrieu, C. and Roberts, G. O. (2009). The pseudo-marginal approach for efficient Monte Carlo computations. *The Annals of Statistics*, 37(2):697–725.

Andrieu, C. and Thoms, J. (2008). A tutorial on adaptive MCMC. *Statistics and Computing*, 18(4):343–373.

Arulampalam, M. S., Maskell, S., Gordon, N., and Clapp, T. (2002). A tutorial on

particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188.

Bar-Shalom, Y., Li, X. R., and Kirubarajan, T. (2001). *Estimation with Applications to Tracking and Navigation*. John Wiley & Sons.

Beaumont, M. a. (2003). Estimation of population growth or decline in genetically monitored populations. *Genetics*, 164(3):1139–60.

Bengtsson, T., Bickel, P., and Li, B. (2008). Curse-of-dimensionality revisited : Collapse of the particle filter in very large scale systems. *Probability and Statistics: Essays in Honor of David A. Freedman*, 2:316–334.

Blom, H. and Bar-Shalom, Y. (1988). The interacting multiple model algorithm for systems with markovian switching coefficients. *IEEE Transactions on Automatic Control*, 33(8):780 –783.

Bollerslev, T., Engle, R., and Nelson, D. (1994). ARCH models. In Engle, R. and McFadden, D., editors, *Handbook of Econometrics*, volume IV, chapter 49, pages 2959–3028. Elsevier, 4 edition.

Cappé, O. (2009). Online sequential Monte Carlo EM algorithm. In *IEEE Workshop on Statistical Signal Processing*, Cardiff, UK.

Cappé, O. (2011). Online EM algorithm for hidden Markov models. *Journal of Computational and Graphical Statistics*, 20(3):728–749.

Cappe, O., Douc, R., and Moulines, E. (2005). Comparison of resampling schemes for

particle filtering. In *Proc. of the 4th International Symposium on Image and Signal Processing and Analysis (ISPA)*, Croatia.

Cappé, O., Godsill, S., and Moulines, E. (2007). An Overview of Existing Methods and Recent Advances in Sequential Monte Carlo. *Proceedings of the IEEE*, 95(5):899–924.

Cappé, O. and Moulines, E. (2009). On-line expectation-maximization algorithm for latent data models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(3):593–613.

Cappé, O., Moulines, E., and Ryden, T. (2005). *Inference in Hidden Markov Models*. Springer Series in Statistics. Springer.

Carpenter, J., Clifford, P., and Fearnhead, P. (1999). Improved particle filter for nonlinear problems. *IEE Proceedings - Radar, Sonar and Navigation*, 146(1):2.

Carvalho, C. M., Johannes, M. S., Lopes, H., and Polson, N. G. (2010). Particle Learning and Smoothing. *Statistical Science*, 25(1):88–106.

Casella, G. and Robert, C. (1998). Post-processing accept-reject samples: recycling and rescaling. *Journal of Computational and Graphical Statistics*, 7(2):139–157.

Chopin, N. (2002). A sequential particle filter method for static models. *Biometrika*, 89(3):539–551.

Chopin, N. (2004). Central limit theorem for sequential Monte Carlo methods and its application to Bayesian inference. *The Annals of Statistics*, 32(6):2385–2411.

Chopin, N., Jacob, P., and Papaspiliopoulos, O. (2012). SMCˆ2 : an efficient algorithm for sequential analysis of state space models. *Journal of the Royal Statistical Society, Series B*, 73(3):1–30.

Crisan, D. and Doucet, A. (2002). A survey of convergence results on particle filtering methods for practitioners. *IEEE Transactions on Signal Processing*, 50(3):736–746.

Crowder, M. (1986). On consistency and inconsistency of estimating equations. *Econometric Theory*, 2(3):305–330.

Dahlin, J., Lindsten, F., and Schon, T. (2013). Particle Metropolis Hastings Using Langevin Dynamics. In *Proceedings of the 38th International Conference on Acoutics and Speech and Signal Processing*.

Dahlin, J., Lindsten, F., and Schön, T. (2014). Second-order Particle MCMC for Bayesian Parameter Inference. In *Proceedings of the 19th World Congress of the International Federation of Automatic Control (IFAC)*.

Del Moral, P. (2004). *Feynman-Kac Formulae: Genealogical and Interacting Particle Systems with Applications*. Probability and its Applications. Springer.

Del Moral, P., Doucet, A., and Singh, S. S. (2010). Forward smoothing using sequential Monte Carlo. *arXiv preprint arXiv:1012.5390v1*.

Del Moral, P., Doucet, A., and Singh, S. S. (2011). Uniform stability of a particle approximation of the optimal filter derivative. *arXiv preprint arXiv:1106.2525v1*.

Dempster, A., Laird, N., and Rubin, D. (1977). Maximum likelihood from incomplete

data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38.

Douc, R. and Cappé, O. (2005). Comparison of resampling schemes for particle filtering. In *ISPA 2005. Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis, 2005.*, pages 64–69.

Doucet, A., Godsill, S., and Andrieu, C. (2000). On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and computing*, 10(3):197–208.

Doucet, A., Gordon, N., and Krishnamurthy, V. (2001). Particle filters for state estimation of jump Markov linear systems. *IEEE Transactions on Signal Processing*, 49(3).

Doucet, A., Jacob, P. E., and Rubenthaler, S. (2013). Derivative free estimation of the score vector and observed information matrix with application to state-space models. *arXiv preprint arXiv:1304.5768v1*.

Doucet, A. and Johansen, A. M. (2011). A tutorial on particle filtering and smoothing: Fifteen years later. In *The Oxford Handbook of Nonlinear Filtering*, pages 656–704. OUP.

Doucet, A., Kantas, N., Singh, S. S., and Maciejowski, J. (2009). An Overview of Sequential Monte Carlo Methods for Parameter Estimation in General State-Space Models. In *15th IFAC Symposium on System Identification*, Saint Malo, France.

Doucet, A., Pitt, M. K., and Kohn, R. (2012). Efficient implementation of Markov

chain Monte Carlo when using an unbiased likelihood estimator. *arXiv preprint arXiv:1210.1871v2*.

Durbin, J. and Koopman, S. (2001). *Time Series Analysis by State Space Methods*. Oxford Statistical Science Series. Oxford University Press.

Erol, Y. B., Li, L., Ramsundar, B., and Russell, S. (2013). The Extended Parameter Filter. In *30th International Conference on Machine Learning*.

Fearnhead, P. (2002). Markov chain Monte Carlo, Sufficient Statistics, and Particle Filters. *Journal of Computational and Graphical Statistics*, 11(4):848–862.

Fearnhead, P. (2007). Computational methods for complex stochastic systems: a review of some alternatives to MCMC. *Statistics and Computing*, 18(2):151–171.

Fearnhead, P. (2011). MCMC for state space models. In *Handbook of Markov Chain Monte Carlo*, pages 513–529. Chapman and Hall.

Fearnhead, P. and Liu, Z. (2007). On-line Inference for Multiple Change Points Problems. *Journal of the Royal Statistical Society, Series B*, 69:589–605.

Fearnhead, P., Wyncoll, D. P., and Tawn, J. (2010). A sequential smoothing algorithm with linear computational cost. *Biometrika*, 97(2):447–464.

Geman, S. and Geman, D. (1984). Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741.

Geweke, J. (1989). Bayesian inference in econometric models using Monte Carlo integration. *Econometrica: Journal of the Econometric Society*, 57(6):1317–1339.

Gilks, W. and Berzuini, C. (2001). Following a Moving Target-Monte Carlo Inference for Dynamic Bayesian Models. *Journal of the Royal Statistical Society, Series B*, 63(1):127–146.

Gilks, W., Richardson, S., and Spiegelhalter, Eds., D. (1999). *Markov Chain Monte Carlo in Practice.* Chapman and Hall.

Girolami, M. and Calderhead, B. (2011). Riemann manifold langevin and hamiltonian monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(2):123–214.

Gordon, N., Salmond, D., and Smith, A. (1993). Novel approach to nonlinear and linear Bayesian state estimation. *IEE Proceedings*, 140(2):107–113.

Hastings, W. (1970). Monte carlo sampling methods using Markov chains and their applications. *Biometrika*, 57:97–109.

Hull, J. and White, A. (1987). The pricing of options on assets with stochastic volatilities. *The Journal of Finance*, 42(2):281–300.

Ionides, E. L., Bhadra, A., Atchadé, Y., and King, A. (2011). Iterated filtering. *The Annals of Statistics*, 39(3):1776–1802.

Ionides, E. L., Bretó, C., and King, a. a. (2006). Inference for nonlinear dynamical

systems. *Proceedings of the National Academy of Sciences of the United States of America*, 103(49):18438–43.

Jacquier, E., Polson, N. G., and Rossi, P. (1994). Bayesian analysis of stochastic volatility models. *Journal of Business & Economic Statistics*, 12(4):69–87.

Jazwinski, A. H. (1970). *Stochastic Processes and Filtering Theory*. Academic Press, New York.

Johansen, A. M. and Doucet, A. (2008). A Note on Auxiliary Particle Filters. *Statistics & Probability Letters*, 78(12):1498–1504.

Julier, S. and Uhlmann, J. (2004). Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422.

Julier, S., Uhlmann, J., and Durrant-White, H. (1995). A new approach for filtering nonlinear systems. In *Proc. of the Amer. Control Conf.*, pages 1628–1632, Washington, DC.

Julier, S., Uhlmann, J., and Durrant-Whyte, H. (2000). A new method for the nonlinear transformation of means and covariances in filters and estimators. *IEEE Transactions on Automatic Control*, 45(3):477–482.

Jungbacker, B. and Koopman, S. J. (2007). Monte Carlo Estimation for Nonlinear Non-Gaussian State Space Models. *Biometrika*, 94(4):827–839.

Kalman, R. E. (1960). A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME, Journal of Basic Engineering*, 82(Series D):35–45.

Kantas, N. (2009). *Sequential Decision Making in General State Space models*. PhD thesis, Cambridge.

Kim, S., Shephard, N., and Chib, S. (1998). Stochastic volatility: likelihood inference and comparison with ARCH models. *The Review of Economic Studies*, 65(3):361–393.

Kitagawa, G. (1996). Monte Carlo Filter and Smoother for Non-Gaussian Nonlinear State Space Models. *Journal of Computational and Graphical Statistics*, 5(1):1–25.

Kitagawa, G. and Sato, S. (2001). Monte carlo smoothing and self-organising state-space model. In *Sequential Monte Carlo Methods in Practice*, pages 178–195. Springer-Verlag, New York.

Kong, A., Liu, J. S., and Wong, W. H. (1994). Sequential Imputations and Bayesian Missing Data Problems. *Journal of the American Statistical Association*, 89(425):278–288.

Koopman, S. and Shephard, N. (1992). Exact score for time series models in state space form. *Biometrika*, 79(4):823–826.

LeGland, F. and Mevel, L. (1997). Recursive estimation in hidden Markov models. In *36th IEEE Conference on Decision and Control*, pages 3468–3473, San Diego, CA.

Lindgren, B. (1993). *Statistical Theory, Fourth Edition*. Chapman & Hall Texts in Statistical Science.

Liu, J. and West, M. (2001). Combined parameter and state estimation in simulation-based filtering. In *Sequential Monte Carlo Methods in Practice*, pages 197–223. Springer-Verlag, New York.

Liu, J., Wong, W., and Kong, A. (1995). Covariance structure and convergence rate of the Gibbs sampler with various scans. *Journal of the Royal Statistical Society. Series B*, 57(1):157–169.

Liu, J. S. (1996). Metropolized independent sampling with comparisons to rejection sampling and importance sampling. *Statistics and Computing*, 6(2):113–119.

Liu, J. S. and Chen, R. (1998). Sequential Monte Carlo Methods for Dynamic Systems. *Journal of the American Statistical Association*, 93(443):1032–1044.

Louis, T. (1982). Finding the observed information matrix when using the EM algorithm. *Journal of the Royal Statistical Society. Series B*, 44(2):226–233.

Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller, E. (1953). Equations of state calculations by fast computing machine. *J. Chem. Phys.*, 21(8):1087–1091.

Neal, R. M. (2010). MCMC Using Hamiltonian Dynamics. In *Handbook of Markov Chain Monte Carlo*, pages 113–162. Chapman & Hall.

Nemeth, C., Fearnhead, P., and Mihaylova, L. (2013). Particle approximations of the score and observed information matrix for parameter estimation in state space models with linear computational cost. *arXiv preprint arXiv:1306.0735*.

Nemeth, C., Fearnhead, P., Mihaylova, L., and Vorley, D. (2012a). Bearings-only tracking with joint parameter learning and state estimation. In *Proc. of 15th International Conference on Information Fusion*, Singapore. IEEE.

Nemeth, C., Fearnhead, P., Mihaylova, L., and Vorley, D. (2012b). Particle learning methods for state and parameter estimation. In *Proc. of 9th IET Data Fusion and Target Tracking Conference*, London. IET.

Olsson, J., Cappé, O., Douc, R., and Moulines, E. (2008). Sequential Monte Carlo smoothing with application to parameter estimation in nonlinear state space models. *Bernoulli*, 14(1):155–179.

Omori, Y., Chib, S., Shephard, N., and Nakajima, J. (2007). Stochastic volatility with leverage: Fast and efficient likelihood inference. *Journal of Econometrics*, 140(2):425–449.

Pitt, M., Silva, R., Giordani, P., and Kohn, R. (2012). On some properties of Markov chain Monte Carlo simulation methods based on the particle filter. *Journal of Econometrics*, 171(2):134–151.

Pitt, M. K. (2002). Smooth Particle Filters for Likelihood Evaluation and Maximisation. Technical report, Warwick University.

Pitt, M. K. and Shephard, N. (1999). Filtering via Simulation: Auxiliary Particle Filters. *Journal of the American Statistical Association*, 94(446):590–599.

Poyiadjis, G., Doucet, A., and Singh, S. S. (2011). Particle approximations of the

score and observed information matrix in state space models with application to parameter estimation. *Biometrika*, 98(1):65–80.

Rabiner, L. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3).

Robert, C. and Casella, G. (2004). *Monte Carlo Statistical Methods*. Springer.

Roberts, G. and Rosenthal, J. S. (2004). General state space Markov chains and MCMC algorithms. *Probability Surveys*, 1:20–71.

Roberts, G. and Tweedie, R. (1996). Exponential convergence of Langevin distributions and their discrete approximations. *Bernoulli*, 2(4):341–363.

Roberts, G. O., Gelman, A., and Gilks, W. (1997). Weak Convergence and Optimal Scaling of the Random Walk Metropolis Algorithms. *The Annals of Applied Probability*, 7(1):110–120.

Roberts, G. O. and Rosenthal, J. S. (1998). Optimal scaling of discrete approximations to Langevin diffusions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 60(1):255–268.

Roberts, G. O. and Rosenthal, J. S. (2001). Optimal scaling for various Metropolis-Hastings algorithms. *Statistical Science*, 16(4):351–367.

Rong Li, X. and Bar-Shalom, Y. (1993). Design of an Interacting Multiple Model Algorithm for Air Traffic Control Tracking. *IEEE Transactions on Control Systems Technology*, 1(3):186–194.

Salmond, D. and Gordon, N. (2001). Particles and Mixtures for Tracking and Guidance. In *Sequential Monte Carlo Methods in Practice*, pages 517–532. Springer-Verlag, New York.

Schon, T., Wills, A., and Ninness, B. (2011). System identification of nonlinear state-space models. *Automatica*, 37(1):39–49.

Shephard, N. (2005). *Stochastic volatility: selected readings*. Advanced texts in econometrics. Oxford University Press.

Sherlock, C., Thiery, A., Roberts, G. O., and Rosenthal, J. S. (2013). On the efficiency of pseudo-marginal random walk Metropolis algorithms. *arXiv preprint arXiv:1309.7209*.

Spall, J. (2000). Adaptive stochastic approximation by the simultaneous perturbation method. *IEEE Transactions on Automatic Control*, 45(10):1839–1853.

Storvik, G. (2002). Particle filters for state-space models with the presence of unknown static parameters. *IEEE Transactions on Signal Processing*, 50(2):281–289.

von Neumann, J. (1951). Various techniques used in connection with random digits. *Journal of Research of the National Bureau of Standards*, 12:36–38.

Wan, E. and van der Merwe, R. (2001). The Unscented Kalman filter. In *Kalman Filtering and Neural Networks*, pages 221–280. Wiley Publishing.

West, M. (1993). Approximating posterior distributions by mixture. *Journal of the Royal Statistical Society. Series B*, 55(2):409–422.

West, M. and Harrison, J. (1997). *Bayesian Forecasting and Dynamic Models*. Springer Series in Statistics. Springer.

Whiteley, N., Johansen, A. M., and Godsill, S. (2011). Monte Carlo Filtering of Piecewise Deterministic Processes. *Journal of Computational and Graphical Statistics*, 20(1):119–139.

Whiteley, N., Singh, S. S., and Godsill, S. (2010). Auxiliary Particle Implementation of Probability Hypothesis Density Filter. *IEEE Transactions on Aerospace and Electronic Systems*, 46(3):1437–1454.

Yildirim, S., Singh, S. S., and Doucet, A. (2012). An Online Expectation-Maximization Algorithm for Changepoint Models. *Journal of Computational and Graphical Statistics*, 22(4):902–926.

Yu, J. (2005). On leverage in a stochastic volatility model. *Journal of Econometrics*, 127(2):165–178.