Linköping studies in science and technology. Thesis.
No. 1652
Licentiate's Thesis

Sequential Monte Carlo for inference in nonlinear state space models

Johan Dahlin

Division of Automatic Control
Department of Electrical Engineering
Linköping University, SE-581 83 Linköping, Sweden
<http://www.control.isy.liu.se>
johan.dahlin@isy.liu.se

Linköping 2014

This is a Swedish Licentiate's Thesis.

Swedish postgraduate education leads to a Doctor's degree and/or a Licentiate's degree.

A Doctor's degree comprises 240 ECTS credits (4 years of full-time studies).

A Licentiate's degree comprises 120 ECTS credits,
of which at least 60 ECTS credits constitute the Licentiate's thesis.

Linköping studies in science and technology. Thesis.

No. 1652

Sequential Monte Carlo for inference in nonlinear state space models

Johan Dahlin

johan.dahlin@isy.liu.se

www.control.isy.liu.se

Department of Electrical Engineering

Linköping University

SE-581 83 Linköping

Sweden

ISBN 978-91-7519-369-4

ISSN 0280-7971

LIU-TEK-LIC-2014:85

Copyright © 2014 Johan Dahlin

Printed by LiU-Tryck, Linköping, Sweden 2014

Denna avhandling tillägnas min familj!

Abstract

Nonlinear *state space models* (SSMs) are a useful class of models to describe many different kinds of systems. Some examples of its applications are to model; the volatility in financial markets, the number of infected persons during an influenza epidemic and the annual number of major earthquakes around the world. In this thesis, we are concerned with state inference, parameter inference and input design for nonlinear SSMs based on *sequential Monte Carlo* (SMC) methods.

The *state inference problem* consists of estimating some *latent variable* that is not directly observable in the output from the system. The *parameter inference problem* is concerned with fitting a pre-specified model structure to the observed output from the system. In *input design*, we are interested in constructing an input to the system, which maximises the information that is available about the parameters in the system output. All of these problems are analytically intractable for nonlinear SSMs. Instead, we make use of SMC to approximate the solution to the state inference problem and to solve the input design problem. Furthermore, we make use of *Markov chain Monte Carlo* (MCMC) and *Bayesian optimisation* (BO) to solve the parameter inference problem.

In this thesis, we propose new methods for parameter inference in SSMs using both Bayesian and maximum likelihood inference. More specifically, we propose a new proposal for the particle Metropolis-Hastings algorithm, which includes gradient and Hessian information about the target distribution. We demonstrate that the use of this proposal can reduce the length of the burn-in phase and improve the mixing of the Markov chain.

Furthermore, we develop a novel parameter inference method based on the combination of BO and SMC. We demonstrate that this method requires a relatively small amount of samples from the analytically intractable likelihood, which are computationally costly to obtain. Therefore, it could be a good alternative to other optimisation based parameter inference methods. The proposed BO and SMC combination is also extended for parameter inference in nonlinear SSMs with intractable likelihoods using approximate Bayesian computations. This method is used for parameter inference in a stochastic volatility model with α -stable returns using real-world financial data.

Finally, we develop a novel method for input design in nonlinear SSMs which makes use of SMC methods to estimate the expected information matrix. This information is used in combination with graph theory and convex optimisation to estimate optimal inputs with amplitude constraints. We also consider parameter estimation in ARX models with Student- t innovations and unknown model orders. Two different algorithms are used for this inference: reversible Jump Markov chain Monte Carlo and Gibbs sampling with sparseness priors. These methods are used to model real-world EEG data with promising results.

Populärvetenskaplig sammanfattning

Den värld som vi lever i är fylld av olika typer av system som kan beskrivas med matematiska modeller. Med hjälp av dessa modeller kan vi skapa oss en bättre förståelse av hur dessa system påverkas av sin omgivning, samt förutsäga hur de kommer att utvecklas över tid. Exempelvis kan man konstruera modeller av vädret baserad på kunskap om fysik samt tidigare års väder. Dessa modeller kan sedan användas för att exempelvis förutsäga om det kommer att regna imorgon. En annan typ av modeller kan användas för att prissätta olika typer av finansiella kontrakt, baserat på tidigare utfall och ekonomisk teori. Ett tredje exempel är modeller för att förutsäga antalet framtida jordbävningar i världen, givet historisk data och några modellantaganden.

Det som är gemensamt för dessa exempel är att de alla beskriver icke-linjära dynamiska system, alltså system som utvecklas över tid. I denna avhandling är vi intresserade av att bygga *icke-linjära tillståndsmodeller* av dynamiska system med hjälp av *datadrivna statistiska inferensmetoder*. Med hjälp av dessa modeller och metoder är det möjligt att kombinera teoretiska kunskaper som beskrivs av en modellstruktur med observationer från systemet. Den senare informationen kan användas för att bestämma värdet på några okända parametrar i modellen, detta kallas även för *parameterskattning*. Ett annat vanligt förekommande problem är *tillståndsskattning*, där vi vill bestämma värdet på någon dynamisk storhet i systemet som inte kan observeras direkt. Det huvudsakliga problemet med detta angreppssätt är att inget av dessa skattningsproblem kan lösas exakt med hjälp av analytiska metoder.

Istället nyttjar vi approximativa metoder som baseras på *statistiska simuleringer* för att lösa problemen. Så kallade *sekventiella Monte Carlo*-metoder används för att approximera lösningen till tillståndsskattningsproblemet. Detta görs med hjälp av en dator som simulerar en stor mängd hypoteser (även kallade partiklar) om hur systemet fungerar. De hypoteser som stämmer väl överens med det verkliga observerade beteendet sparas och förfinas i nästkommande steg. De övriga hypoteserna tas bort från simuleringen för att fokusera beräkningskraften på de hypoteser som har relevans enligt den observerade informationen. Parameterskattningsproblemet kan lösas approximativt med liknande metoder som även de bygger på simulering.

I denna avhandling arbetar vi främst med att försöka förbättra parameterskattningsmetoder som bygger på partikel Markovkedje-Monte Carlo (MCMC) och Bayesiansk optimering. De förbättringar som vi föreslår leder till att skattningarna kan beräknas snabbare än tidigare genom att bättre ta tillvara på den information som finns i det observerade datamaterialet. Dessa metoder används för att exempelvis prissätta finansiella optioner. Vi föreslår även en ny algoritm för att skapa insignaler till system så att observationerna som erhålls från systemet, innehåller så mycket information som möjligt om de okända parametrarna. Slutligen demonstrerar vi hur man kan använda MCMC-metoder för parameterskattning i modeller som kan användas för att beskriva EEG-signaler.

Acknowledgments

I could never have written my thesis without the help, love and support from all the people around me, now and in the past. I will now spend a few lines to express my gratitude to some of the special people that have helped me along the way.

First and foremost, I would like to acknowledge the help and support from my supervisors. My main supervisor Prof. Thomas Schön has always provided me with encouragement, ideas, new people to meet, new opportunities to grab and challenges to help me grow. Also, my co-supervisor Dr. Fredrik Lindsten has always been there for me with answers to my questions, explanations to my problems and ideas/suggestions for my work. I am most impressed by your enthusiasm and dedication in the roles as supervisors. I think that you both have surpassed what can be expected from a supervisor and for this I am extremely grateful. I could never have done this without you! Thank you for your support and all our time running together in Stanley park, Maastricht, Warwick and Söderköping!

Furthermore, to be able to write a good thesis you require a good working environment. Prof. Svante Gunnarsson and Ninna Stensgård are two very important persons in this effort. Thank you for all your kindness, support and helpfulness in all matters; small and large. I would also like to acknowledge Dr. Henrik Tidefelt and Dr. Gustaf Hendeby for constructing and maintaining the L^AT_EX-template in which this thesis is written. My brother Fredrik Dahlin, Lic. Joel Kronander, Jonas Linder, Dr. Fredrik Lindsten, Prof. Thomas Schön, Andreas Svensson, Patriocio Valenzuela and Johan Wågberg have all helped with proof-reading and with suggestions to improve the thesis. All remaining errors are entirely my own.

I am most grateful for the financial support from the project *Probabilistic modelling of dynamical systems* (Contract number: 621-2013-5524) funded by the Swedish Research Council.

Another aspect of the atmosphere at work is all my wonderful colleagues. Especially, my room mate Lic. Michael Roth, who helps with his experience and advice about balancing life as a PhD student. Thank you for sharing the room with me and watering our plants while I am away! I would also like to thank Jonas Linder for all our adventures and our nice friendship together both at and outside of work. Manon Kok and I started in the group at the same time and have helped eachother over the years. Thank you for your positive attitude and for always being open for discussions. Also, I would like to acknowledge the wonderful BBQs and other funny things that Lic. Sina Khoshfetrat Pakazad arranges and lets me participate in!

Furthermore, I would like to thank my remaining friends and colleagues at the group. Especially, (without any specific ordering) Dr. Christian Lyzell, Lic. Ylva Jung, Isak Nielsen, Hanna Nyquist, Dr. Daniel Petersson, Clas Veibäck and Dr. Emre Özkan for all the funny things that we have done together. This includes everything from beer tastings, wonderful food in France and fitting as many of RTs PhD students into a jacuzzi as possible to hitting some shallows on open sea with

a canoe, cross country skiing in Chamonix and eating food sitting on the floor in a Japanese restaurant with screaming people everywhere. You have given me the most wonderful memories and times!

I have also had the benefit of working with a lot of talented and enthusiastic researchers during my time in academia. I would first like to thank all my fellow students at Teknisk Fysik, Umeå University for a wonderful time as an undergraduate student. Also, Prof. Anders Fällström, Dr. Konrad Abramowicz, Dr. Ulf Holmberg and Dr. Sang Hoon Lee have inspired, supported and encouraged me to pursue a PhD degree, something I have not (a.s.) regretted!

Also, my wonderful (former) colleagues at the Swedish Defence Research Agency (FOI) have supported and encouraged me to continue climbing the educational ladder. Thank you, Dr. Pontus Svensson, Dr. Fredrik Johansson, Dr. Tove Gustavi and Christian Mårtensson. Finally, I would like to thank all my co-authors during my time at Linköping University for some wonderful and fruitful collaborations: Daniel Hultqvist, Lic. Daniel Jönsson, Lic. Joel Kronander, Dr. Fredrik Lindsten, Cristian Rojas, Dr. Jakob Roll, Prof. Thomas Schön, Fredrik Svensson, Dr. Jonas Unger, Patricio Valenzuela, Prof. Mattias Villani and Dr. Adrian Wills.

Furthermore, Lic. Joel Kronander and Dr. Jonas Unger have helped me with the images from the computer graphics application in the introduction. Prof. Mattias Villani together with Stefan Laséen and Vesna Crobo at Riksbanken helped with the economics application and made the forecasts from RAMSES II.

Finally, I am most grateful to my loving family and close relatives for their support all the way from childhood until now and beyond. I love you all every much! Also, my friends are always a great source for support, inspiration and encouragement both at work and in life! My life would be empty and meaningless without you all! I hope that we all can spend some more time now when my thesis is done and when new challenges awaits! Because, what would life be without challenges, meeting new people and spending time with your loved ones. Empty.

Linköping, May 2014
Johan Dahlin

Contents

Notation	xv
----------	----

I Background

1 Introduction	3
1.1 Examples of applications	4
1.1.1 Predicting GDP growth	5
1.1.2 Rendering photorealistic images	7
1.2 Thesis outline and contributions	9
1.3 Publications	14
2 Nonlinear state space models and statistical inference	17
2.1 State space models and inference problems	17
2.2 Some motivating examples	19
2.2.1 Linear Gaussian model	19
2.2.2 Volatility models in econometrics and finance	19
2.2.3 Earthquake count model in geology	24
2.2.4 Daily rainfall models in meteorology	26
2.3 Maximum likelihood parameter inference	28
2.4 Bayesian parameter inference	32
3 State inference using particle methods	37
3.1 Filtering and smoothing recursions	37
3.2 Monte Carlo and importance sampling	39
3.3 Particle filtering	41
3.3.1 The auxiliary particle filter	43
3.3.2 State inference using the auxiliary particle filter	46
3.3.3 Statistical properties of the auxiliary particle filter	49
3.3.4 Estimation of the likelihood and log-likelihood	51
3.4 Particle smoothing	53
3.4.1 State inference using the particle fixed-lag smoother	55
3.4.2 Estimation of additive state functionals	59

3.4.3	Statistical properties of the particle fixed-lag smoother . . .	63
3.5	SMC for Image Based Lighting	64
4	Parameter inference using sampling methods	69
4.1	Overview of computational methods for parameter inference	69
4.1.1	Maximum likelihood parameter inference	70
4.1.2	Bayesian parameter inference	70
4.2	Metropolis-Hastings	71
4.2.1	Statistical properties of the MH algorithm	76
4.2.2	Proposals using Langevin and Hamiltonian dynamics	78
4.3	Particle Metropolis-Hastings	83
4.4	Bayesian optimisation	85
4.4.1	Gaussian processes as surrogate functions	88
4.4.2	Acquisition rules	93
4.4.3	Gaussian process optimisation	96
5	Concluding remarks and future work	103
5.1	Summary of the contributions	103
5.2	Outlook and future work	104
5.2.1	Particle Metropolis-Hastings	104
5.2.2	Gaussian process optimisation using the particle filter . . .	105
5.2.3	Input design in SSMs	106
5.3	Source code and data	106
	Bibliography	107

II Publications

A	PMH using gradient and Hessian information	121
1	Introduction	124
2	Particle Metropolis-Hastings	126
2.1	MH sampling with unbiased likelihoods	126
2.2	Constructing the first and second order proposals	127
2.3	Properties of the first and second order proposals	128
3	Estimation of likelihoods, gradients, and Hessians	129
3.1	Auxiliary particle filter	129
3.2	Estimation of the likelihood	130
3.3	Estimation of the gradient	131
3.4	Estimation of the Hessian	132
3.5	Accuracy of the estimated gradients and Hessians	134
3.6	Resulting SMC algorithm	135
4	Numerical illustrations	135
4.1	Estimation of the log-likelihood and the gradient	135
4.2	Burn-in and scale-invariance	138
4.3	The mixing of the Markov chains at stationarity	140
5	Discussion and future work	143

Bibliography	145
B Particle filter-based GPO for parameter inference	149
1 Introduction	152
2 Maximum likelihood estimation with a surrogate cost function . .	153
3 Estimating the log-likelihood	154
3.1 The particle filter	154
3.2 Estimation of the likelihood	155
3.3 Estimation of the log-likelihood	156
4 Modelling the surrogate function	156
4.1 Gaussian process model	157
4.2 Updating the model and the hyperparameters	158
4.3 Example of log-likelihood modelling	158
5 Acquisition rules	158
5.1 Expected improvement	160
6 Numerical illustrations	160
6.1 Implementation details	161
6.2 Linear Gaussian state space model	161
6.3 Nonlinear stochastic volatility model	164
7 Conclusions	164
Bibliography	165
C Approximate inference in SSMs with intractable likelihoods using GPO	167
1 Introduction	170
2 An intuitive overview	171
3 Estimating the posterior distribution	172
3.1 State inference	172
3.2 Estimation of the log-likelihood	174
4 Gaussian process optimisation	174
4.1 Constructing the surrogate function	174
4.2 The acquisition rule	176
5 Putting the algorithm together	176
6 Numerical illustrations	177
6.1 Inference in α -stable data	177
6.2 Linear Gaussian model	181
6.3 Stochastic volatility model with α -stable returns	181
7 Conclusions and outlook	184
Bibliography	186
A α -stable distributions	188
A.1 Definitions	188
A.2 Simulating random variables	189
A.3 Parameter estimation	191
D A graph/particle-based method for experiment design	193
1 Introduction	196
2 Problem formulation	197

3	New input design method	198
3.1	Graph theoretical input design	199
3.2	Estimation of the score function	201
3.3	Monte Carlo-based optimisation	203
3.4	Summary of the method	204
4	Numerical examples	205
4.1	Linear Gaussian state space model	205
4.2	Nonlinear growth model	206
5	Conclusion	207
	Bibliography	208
E	Hierarchical Bayesian approaches for robust inference in ARX models	211
1	Introduction	214
2	Hierarchical Bayesian ARX Models	215
2.1	Student's t distributed innovations	215
2.2	Parametric model order	216
2.3	Automatic relevance determination	216
3	Markov chain Monte Carlo	217
4	Posteriors and proposal distributions	218
4.1	Model order	218
4.2	ARX coefficients	219
4.3	ARX coefficients variance	220
4.4	Latent variance variables	220
4.5	Innovation scale parameter	221
4.6	Innovation DOF	221
5	Numerical illustrations	222
5.1	Average model performance	222
5.2	Robustness to outliers and missing data	224
5.3	Real EEG data	226
6	Conclusions and Future work	228
	Bibliography	229

Notation

PROBABILITY

Notation	Meaning
$\xrightarrow{a.s.}$	Almost sure convergence.
$\xrightarrow{d}$	Convergence in distribution.
$\xrightarrow{p}$	Convergence in probability.
$\delta_z(dx)$	Dirac point mass located at $x = z$.
$\mathbb{P}, \mathbb{E}, \mathbb{V}$	Probability, expectation and covariance operators.
$\sim$	Sampled from or distributed according to.

STATISTICAL DISTRIBUTIONS

Notation	Meaning
$\mathcal{A}(\alpha, \beta, \gamma, \eta)$	α -stable distribution with stability α , skewness β , scale γ and location η .
$\mathcal{B}(p)$	Bernoulli distribution with success probability p .
$\mathcal{N}(\mu, \sigma^2)$	Gaussian (normal) dist. with mean μ and variance σ^2
$\mathcal{G}(\alpha, \beta)$	Gamma distribution with rate α and shape β .
$\mathcal{IG}(\alpha, \beta)$	Inverse Gamma distribution with rate α and shape β .
$\mathcal{P}(\lambda)$	Poisson distribution with mean λ .
$\mathcal{U}(a, b)$	Uniform distribution on the interval $[a, b]$.

OPERATORS AND OTHER SYMBOLS

Notation	Meaning
I_d	$d \times d$ identity matrix.
$\triangleq$	Definition.
$\text{diag}(v)$	Diagonal matrix with the vector v on the diagonal.
$\nabla f(x)$	Gradient of $f(x)$.
$\nabla^2 f(x)$	Hessian of $f(x)$.
$\mathbb{I}$	Indicator function.
$\det(A)$, $ A $	Matrix determinant of A .
A^{-1}	Matrix inverse of A .
$\text{tr}(A)$	Matrix trace of A .
$A^\top$	Matrix transpose of A .
$v^2 = vv^\top$	Outer product of the vector v .
$a_{n:m}$	Sequence $\{a_n, a_{n+1}, \dots, a_{m-1}, a_m\}$, for $m > n$.
$\text{sign}(x)$	Sign of x .
$\text{supp}(f)$	Support of the function f , $\{x : f(x) > 0\}$.

STATISTICAL QUANTITIES

Notation	Meaning
$\mathcal{I}(\theta)$	Expected information matrix evaluated at θ .
$\mathcal{L}(\theta)$	Likelihood function evaluated at θ .
$\ell(\theta)$	Log-likelihood function evaluated at θ .
$\hat{\theta}_{\text{ML}}$	Maximum likelihood parameter estimate.
$\mathcal{J}(\theta)$	Observed information matrix evaluated at θ .
$\hat{\theta}$	Parameter estimate.
$p(\theta y_{1:T})$	Parameter posterior distribution.
$p(\theta)$	Parameter prior distribution.
θ	Parameter vector, $\theta \in \Theta \subseteq \mathbb{R}^d$.
$S(\theta)$	Score function evaluated at θ .

ALGORITHMIC QUANTITIES

Notation	Meaning
$a_t^{(i)}$	Ancestor of particle i at time t .
Z	Normalisation constant.
$x_t^{(i)}$	Particle i at time t .
$R_\theta(x_t x_{0:t-1}, y_t)$	Particle proposal kernel.
$W_\theta(x_t, x_{t-1})$	Particle weighting function.
$q(\theta)$	Proposal distribution.
$\pi(\theta)$	Target distribution.
$\gamma(\theta)$	Unnormalised target distribution.
$w_t^{(i)}, \tilde{w}_t^{(i)}$	Un- and normalised weight of particle i at time t .

ABBREVIATIONS

Abbreviation	Meaning
a.s.	Almost surely (with probability 1).
ABC	Approximate Bayesian computations.
ACF	Autocorrelation function.
AIS	Adaptive importance sampling.
APF	Auxiliary particle filter.
AR(p)	Autoregressive process of order p .
ARD	Automatic relevance determination.
ARCH(p)	AR conditional heteroskedasticity process of order p .
ARX(p)	Autoregressive exogenous process of order p .
BIS	Bidirectional importance sampling.
BO	Bayesian optimisation.
bPF	Bootstrap particle filter.
BRDF	Bidirectional reflectance distribution function.
CDF	Cumulative distribution function.
CLT	Central limit theorem.
CPI	Consumer price index.
DSGE	Dynamic stochastic general equilibrium.
EB	Empirical Bayes.
EEG	Electroencephalography.
EI	Expected improvement.
EM	Environment map.
ESS	Effective sample size.
faPF	Fully-adapted particle filter.
FFBSm	Forward-filtering backward-smoothing.
FFBSi	Forward-filtering backward-simulation.
FL	Fixed-lag (particle smoother).
GARCH(p, q)	Generalised ARCH process of order (p, q) .
GPO	Gaussian process optimisation.
GPU	Graphical processing unit.
HMM	Hidden Markov model.
IACT	Integrated autocorrelation time.
IBL	Image-based lightning.
IID	Independent and identically distributed.
IS	Importance sampling.
KDE	Kernel density estimate/estimator.
LGSS	Linear Gaussian state space.
LTE	Light transport equation.
MCMC	Markov chain Monte Carlo.
MH	Metropolis-Hastings.
MIS	Multiple importance sampling.
ML	Maximum likelihood.
MLE	Maximum likelihood estimator.
MLT	Metropolis light transport.
MSE	Mean square error.

ABBREVIATIONS (CONT.)

Abbreviation	Meaning
PD	Positive definite.
PDF	Probability density function.
PMF	Probability mass function.
PF	Particle filter.
PG	Particle Gibbs.
PI	Probability of improvement.
PMCMC	Particle Markov chain Monte Carlo.
PMH	Particle Metropolis-Hastings.
PMH0	Marginal particle Metropolis-Hastings.
PMH1	PMH using first order information
PMH2	PMH using first and second order information
PS	Particle smoother.
RJ-MCMC	Reversible jump Markov chain Monte Carlo.
RTS	Rauch-Tung-Stribel.
RW	Random walk.
SIS	Sequential importance sampling.
SIR	Sequential importance sampling and resampling.
SLLN	Strong law of large numbers.
SMC	Sequential Monte Carlo.
SPSA	Simultaneous perturbation stochastic approximation.
SSM	State space model.
UCB	Upper confidence bound.

Part I

Background

1

Introduction

Science is the art of collecting and organising knowledge about the universe by tested explanations and validated predictions. Therefore, modelling the world using *observations* and *statistical inference* is an integral part of the scientific method. The resulting statistical models can be used to *describe* certain observed phenomena or to *predict* new phenomena and future behaviours. An example of the former is to discover new physical models by generalising from observed data using *induction*. Examples of prediction applications are to validate scientific theories or to forecast the future GDP of Sweden, the probability of rainfall tomorrow and the number of earthquakes during the coming year. We discuss some of the details of these problems in the following chapters.

This thesis is concerned with *building dynamical models from recorded observations*, i.e. models of systems that evolves over time. The observations are combined with past experiences and established scientific theory to build models using statistical tools. Here, we limit ourselves to discussing *nonlinear state space models* (SSMs), where most of the structure is known beforehand except a few parameters. A fairly general class of SSMs can be expressed as

$$\begin{aligned}x_{t+1}|x_t &\sim f_{\theta}(x_{t+1}|x_t), \\ y_t|x_t &\sim g_{\theta}(y_t|x_t),\end{aligned}$$

where x_t and y_t denotes an unobserved (latent) state and an observation from the system at time t . Here, $f_{\theta}(x_{t+1}|x_t)$ and $g_{\theta}(y_t|x_t)$ denotes two Markov kernels parametrised by an unknown static real-valued parameter vector θ . Applications of this class of SSMs can be found in almost all of the natural sciences and most of the social sciences. Some specific examples are biology (Wilkinson, 2011), control (Ljung, 1999), epidemiology (Keeling and Rohani, 2008) and finance (Tsay, 2005; Hull, 2009).

The procedure to determine the parameter vector θ from the observations $y_{1:T}$ is referred to as *parameter inference* and this problem is analytically intractable for nonlinear SSMs. Another related problem is the *state inference* problem, where we would like to determine the value of x_t given the information in the observations $y_{1:T}$ or $y_{1:t}$. This problem is also analytically intractable for most SSMs.

Instead, we make use of *statistical simulation* methods to estimate the parameters and states. As the name suggests, these methods are based on simulating many (often thousands or millions) of hypotheses referred to as *particles*. The particles that match the recorded observations are retained and the others are discarded. This procedure can be repeated in a sequential manner, where the solution of the problem is obtained as the solution to many subproblems.

This idea is the basis for the *sequential Monte Carlo* (SMC) methods that are an integral part of the methods that we consider for state inference in SSMs. For example, the marginal filtering distribution $p_\theta(x_t|y_{1:t})$ can be approximated by an *empirical distribution*,

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

where $x_t^{(i)}$ and $\tilde{w}_t^{(i)}$ denotes the particle i and its corresponding (normalised) weight obtained from the SMC algorithm. Here, $\delta_z(dx)$ denotes a Dirac point mass located at $x = z$. The empirical distribution summarises all the information that is contained within the data about the value of the latent state at some time t . This information can then be used by other methods for solving the parameter inference problem in SSMs.

The number of particles N in the SMC method controls both the accuracy of the empirical distributions and the computational cost. That is, a high accuracy requires many particles which incurs a high computational cost and this results in a trade-off between accuracy and speed. Also, we make use of the SMC algorithm within some iterative parameter inference methods to solve the state inference problem at each iteration. Therefore, we would like to limit the number of iterations required by the parameter inference methods to obtain an accurate parameter estimate with a reasonable computational cost.

These ideas are the two main themes of this thesis. The first theme is *to propose some developments to improve the efficiency of existing parameter inference methods based on SMC algorithms*. The second theme is *to extend some of the current methods for linear SSMs to nonlinear SSMs by making use of SMC algorithms*. We return to the contributions of this thesis in Section 1.2.

1.1 Examples of applications

In this section, we give two examples of problems in which computational statistical methods based on the SMC algorithm are useful. In the first example, we use a

model to forecast the future development of the Swedish economy given past experience and economical theory. In the second example, we use a model constructed from the physics of light transport to render photorealistic images.

1.1.1 Predicting GDP growth

The economy of a country is a complex system with an emergent behaviour depending on the actions of many interacting heterogeneous agents. In an economy, these agents correspond to consumers, companies, banks, politicians, governmental agencies, other countries, etc. As such, these agents may or may not act rationally to their situation and could therefore be difficult to model on an individual level.

As a result, economical models mostly deal with the aggregated behaviour of many homogeneous agents, i.e. rational utility maximising agents with a common valuation of goods and services. These models can be used to produce forecasts, to gain understanding about the current situation in the economy and simulate the result of different policy decisions. An example of this could be to study the impact of changing the repo rate on the unemployment level and the GDP growth of the economy.

For this purpose, many central banks are today using *dynamic stochastic general equilibrium* (DSGE) models (An and Schorfheide, 2007; Del Negro and Schorfheide, 2004) for modelling the economy of a country. The outputs from these models are various macroeconomic quantities, such as GDP growth, unemployment rate, inflation, etc. The general structure is given by economic theory, but there are some unknown parameters that needs to be inferred from data.

Riksbanken (the Swedish central bank) has developed a DSGE model called *the Riksbank Aggregate Macromodel for Studies of the Economy of Sweden II* (RAMSES II) (Adolfson et al., 2013, 2007a) to model the Swedish economy. Essentially, RAMSES II is a nonlinear SSM with 12 outputs, about 40 latent states and about 65 unknown parameters.

For computational convenience, only the log-linearised version of the full model is considered in most of the analysis. Consequently, Kalman filtering methods can be used to solve the state inference problem. The parameter inference problem is solved using a *Metropolis-Hastings* (MH) algorithm, where the proposal is a multivariate Gaussian distribution with the covariance matrix given by the inverse of the observed information matrix at the posterior mode. The information matrix is estimated using Quasi-Newton optimisation algorithms such as the BFGS algorithm (Nocedal and Wright, 2006). For more details, see Adolfson et al. (2007b).

In Chapters 3 and 4, we discuss alternative methods that could solve the state and parameter inference problem in the original nonlinear version of RAMSES II. For related treatments using SMC and MCMC in combination with DSGE models, see Flury and Shephard (2011), Fernández-Villaverde and Rubio-Ramírez (2007) and Amisano and Tristani (2010).

In Figure 1.1, we give an example of how the RAMSES II model can be used for forecasting the changes in GDP, the consumer price index with fixed interest rates

Figure 1.1: The quarterly GDP growth (green), the repo rate (red), the annual change in the CPIF (blue) and unemployment gap (orange). The historical data is presented for each variable up until the dotted vertical lines. The predicted means are presented with the 50%, 70%, 90% and 95% credibility intervals (from lighter to darker gray). The published forecasts from Riksbanken are presented as crosses. The data and predictions are obtained by the courtesy of Riksbanken.

(CPIF) inflation, the repo rate and the unemployment gap¹ in Sweden. We first make use of some historical data (up until the dotted vertical line) to estimate the parameters and the latent states. The resulting model is then used to forecast the predictive posterior mean (solid lines) with some credibility intervals (gray areas). These predictive means are used by Riksbanken together with experts and other models to construct forecasts of the economy. This forecast is presented by crosses and differs from the output of the RAMSES II model.

1.1.2 Rendering photorealistic images

To simulate light transport, we make use of a geometrical optics model, developed during centuries of research in the field of physics (Hecht, 2013). By the use of this models, we can simulate how light behaves in different environments and make use of this to *render* images. A popular related application is to add objects into an image or video sequence that were not present when the scene was captured. In this section, we shortly discuss how to do this using the so-called *image-based lighting* (IBL) method (Debevec, 1998; Pharr and Humphreys, 2010).

To make use of the IBL method, we require an panoramic image of the real scene captured using a *high dynamic range* (HDR) camera. This type of camera can record much larger variations in brightness than a standard camera, which are needed to capture all the different light sources within the scene. The resulting image is referred to as an *environment map* (EM). In IBL, this panoramic image serves as the source of illumination when rendering images, allowing the real objects to cast shadows and interact with the virtual objects. Secondly, we need geometrical models of the objects that we would like to add into the scene. Finally, we require a mathematical description of the optical properties of the materials in these objects to be able to simulate how the light scatters over their surfaces.

The IBL method combines all of this information using the *light transport equation* (LTE), which is a physical model of how light rays propagates through space and reflects off surfaces. The LTE model cannot be solved analytically, but it can be approximated using methods related to SMC algorithms. To see how this can be done, consider a cartoon of the setup presented in Figure 1.2. In the first step, a set of light rays originating from a pixel in the image plane is generated. We then track how these rays bounces around in the scene until they finally hit the EM. The colours and brightnesses of the EM in these locations are recorded and used to compute the resulting colour and brightness of the pixel in the image plane. This approach is repeated for all the pixels in the image plane.

However in the real world, there are infinitely many rays that bounces around in the scene before they hit the pixels in the image plan. As a result, it is computationally infeasible to simulate all the light rays and all the bounces in the scene. Instead, there are methods to select only the light rays which contributes the most to

¹The unemployment gap is the amount (in percent) that the GDP must increase to be able to achieve full employment of the work force. The decrease of the unemployment gap is an important aim of financial policy making and can be achieved (in Keynesian economic theory) by increasing public spending or lowering taxes.

Figure 1.2: The basic setup of ray tracing underlying photorealistic image synthesis. The colour and brightness of a pixel in the image plane is determined by the EM, the geometry of the scene and the optical properties of the objects.

Figure 1.3: The scene before (left) and after (right) the rendering using a version of the IBL method. The image is taken from Unger et al. (2013) and is used with courtesy of the authors.

the brightness and colour each pixel in the image plane. This can be done in a similar manner to the methods discussed later in Section 4.2 resulting in the *Metropolis light transport* (MLT) algorithm (Veach and Guibas, 1997). The basis for these methods is to solve the LTE problem by simulating different hypotheses and improving them in analogue with SMC methods. That is, light rays that hit bright areas of the EM are kept and modified, whereas rays that does hit the EM in dim regions or bounces around too long are discarded.

Note that, it can take several days to render a single image using the IBL algorithm, even when only allowing for a few bounces and light rays per pixel in the image plane. This problem grows even further when we would like to render a sequence of images. A possible solution could be to start from the solution from the previous frame and adapt it to the new frame. If the EMs are similar, this could lead to a decrease in the total computational cost. We return to this idea in Section 3.5.

In Figure 1.3, we present an example from Unger et al. (2013) of a scene before (left) and after (right) it is rendered in a computer by the use of the IBL method and the methods discussed in Section 4.2. Note that, in the final result we have added several photorealistic objects into the scene such as the sofa, the table and have also changed the floor in the room. These methods are used in many entertainment applications to create special effects and to modify scenes in post production. Furthermore, they are useful in rendering images of scenes that are difficult or costly to build in the real-world. Some well-known companies (such as IKEA and Volvo) make use of these methods for digital design and advertisements as a cost effective alternative to traditional photography.

1.2 Thesis outline and contributions

This thesis is divided into two parts. In Part I, we give some examples of models and applications together with an introduction to the different computational inference methods that are used. In Part II, we present edited versions of some published peer-reviewed papers and unpublished technical reports.

Part I - Background

In this part, we begin by introducing the SSM and provide some additional examples of its real-world applications in Chapter 2. Furthermore, we introduce two different statistical paradigms for parameter inference problems in SSMs: the maximum likelihood (ML) based approach and the Bayesian approach. Finally, we discuss why computational methods are required for estimating the solution to these problems.

In Chapter 3, we review the state inference problem in SSMs and discuss the use of SMC methods for approximating the solution to these problems. We also discuss the use of SMC algorithms for other classes of models, which includes the computer graphics example discussed in Section 1.1.2.

Chapter 4 is devoted to discussing the parameter inference problem for nonlinear

SSMs. We begin by giving an overview of different parameter inference methods and then discuss Markov chain Monte Carlo (MCMC) and Bayesian optimisation (BO) in more detail. The former can be used for Bayesian parameter inference and the latter can be used for ML or maximum a posteriori (MAP) based parameter inference.

We conclude Part I by Chapter 5 which contains a summary of the contributions of the thesis together with some general conclusions and possible avenues for future work.

Part II - Publications

The main part of this thesis is the compilation of five papers published in peer-reviewed conference proceedings or as technical reports. These papers contain the main contributions of this thesis:

- In Paper A, we develop a novel particle MCMC algorithm that combines the Particle Metropolis-Hastings (PMH) with the Langevin dynamics. The resulting algorithm explores the posterior distribution more efficient than the marginal PMH algorithm, is invariant to affine transformations of the parameter vector and reduces the length of the burn-in. As a consequence, the proposed algorithm requires less iterations, which makes it more computationally efficient than the marginal PMH algorithm.
- In Paper B, we develop a novel algorithm for ML parameter inference by combining ideas from BO with SMC for log-likelihood estimation. The resulting algorithm is computationally efficient as it requires less samples from the log-likelihood compared with other popular methods.
- In Paper C, we extend the combination of BO and SMC to parameter inference in nonlinear SSMs with intractable likelihoods. Computationally costly approximate Bayesian computations (ABC) are used to approximate the likelihood. We illustrate the proposed algorithm for parameter inference in stochastic volatility model with α -stable returns using real-world data.
- In Paper D, we develop a novel algorithm for input design in nonlinear SSMs, which can handle amplitude constraints on the input. The proposed method makes use of SMC for estimating the expected information matrix. The algorithm performs well compared with some other methods in the literature and decreases the variance of the parameter estimates with almost an order of magnitude.
- In Paper E, we propose two algorithms for parameter inference in ARX models with Student- t innovations which includes automatic model order selection. These methods makes use of *reversible jump MCMC* (RJMCMC) and the Gibbs sampler together with sparseness priors to estimate the model order and the parameter vector. We illustrate the use of the proposed algorithm to model real-world EEG data with promising results.

Here, we present an abstract of each paper together with an account of the contribution of the author of this thesis.

Paper A

Paper A of this thesis is an edited version of,

J. Dahlin, F. Lindsten, and T. B. Schön. Particle Metropolis-Hastings using gradient and Hessian information. *Pre-print*, 2014b. arXiv:1311.0686v2.

which is a combination and development of the two earlier publications

J. Dahlin, F. Lindsten, and T. B. Schön. Second-order particle MCMC for Bayesian parameter inference. In *Proceedings of the 19th IFAC World Congress*, Cape Town, South Africa, August 2014a. (accepted for publication).

J. Dahlin, F. Lindsten, and T. B. Schön. Particle Metropolis Hastings using Langevin dynamics. In *Proceedings of the 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vancouver, Canada, May 2013a.

Abstract: PMH allows for Bayesian parameter inference in nonlinear state space models by combining MCMC and particle filtering. The latter is used to estimate the intractable likelihood. In its original formulation, PMH makes use of a marginal MCMC proposal for the parameters, typically a Gaussian random walk. However, this can lead to a poor exploration of the parameter space and an inefficient use of the generated particles.

We propose two alternative versions of PMH that incorporate gradient and Hessian information about the posterior into the proposal. This information is more or less obtained as a byproduct of the likelihood estimation. Indeed, we show how to estimate the required information using a fixed-lag particle smoother, with a computational cost growing linearly in the number of particles. We conclude that the proposed methods can: (i) decrease the length of the burn-in phase, (ii) increase the mixing of the Markov chain at the stationary phase, and (iii) make the proposal distribution scale invariant which simplifies tuning.

Contributions and background: The author of this thesis contributed with the majority of the work including the design, the implementation, the numerical illustrations and the written presentation.

Paper B

Paper B of this thesis is an edited version of,

J. Dahlin and F. Lindsten. Particle filter-based Gaussian process optimisation for parameter inference. In *Proceedings of the 19th IFAC World Congress*, Cape Town, South Africa, August 2014. (accepted for publication).

Abstract: We propose a novel method for maximum-likelihood-based parameter inference in nonlinear and/or non-Gaussian state space models. The method is an iterative procedure with three steps. At each iteration a particle filter is used to estimate the value of the log-likelihood function at the current parameter iterate. Using these log-likelihood estimates, a surrogate objective function is created by utilizing a Gaussian process model. Finally, we use a heuristic procedure to obtain a revised parameter iterate, providing an automatic trade-off between exploration and exploitation of the surrogate model. The method is profiled on two state space models with good performance both considering accuracy and computational cost.

Contributions and background: The author of this thesis contributed with the majority of the work including the design, the implementation, the numerical illustrations and the written presentation.

Paper C

Paper C of this thesis is an edited version of,

J. Dahlin, T. B. Schön, and M. Villani. Approximate inference in state space models with intractable likelihoods using Gaussian process optimisation. Technical Report LiTH-ISY-R-3075, Department of Electrical Engineering, Linköping University, Linköping, Sweden, April 2014c.

Abstract: We propose a novel method for MAP parameter inference in nonlinear state space models with intractable likelihoods. The method is based on a combination of BO, SMC and SBC. SMC and ABC are used to approximate the intractable likelihood by using the similarity between simulated realisations from the model and the data obtained from the system. The BO algorithm is used for the MAP parameter estimation given noisy estimates of the log-likelihood.

The proposed parameter inference method is evaluated in three problems using both synthetic and real-world data. The results are promising, indicating that the proposed algorithm converges fast and with reasonable accuracy compared with existing methods.

Contributions and background: The author of this thesis contributed with the majority of the work including the design, the implementation, the numerical illustrations and the written presentation. This contribution resulted from the participation in the course *Bayesian learning* given by Prof. Mattias Villani at Linköping University during the autumn of 2013.

Paper D

Paper D of this thesis is an edited version of,

P. E. Valenzuela, J. Dahlin, C. R. Rojas, and T. B. Schön. A graph/particle-based method for experiment design in nonlinear systems. In *Proceedings of the 19th IFAC World Congress*, Cape Town, South Africa, August 2014. (accepted for publication).

Abstract: We propose an extended method for experiment design in nonlinear state space models. The proposed input design technique optimizes a scalar cost function of the information matrix, by computing the optimal stationary probability mass function (PMF) from which an input sequence is sampled. The feasible set of the stationary PMF is a polytope, allowing it to be expressed as a convex combination of its extreme points. The extreme points in the feasible set of PMFs can be computed using graph theory.

Therefore, the final information matrix can be approximated as a convex combination of the information matrices associated with each extreme point. For nonlinear SSMs, the information matrices for each extreme point can be computed by using particle methods. Numerical examples show that the proposed technique can be successfully employed for experiment design in nonlinear SSMs.

Contributions and background: This is an extension of the work presented in Valenzuela et al. (2013) and a result of the cooperation with the Department of Automatic Control at the Royal Institute of Technology (KTH). The author of this thesis designed and implemented the algorithm for estimating the expected information matrix and the Monte Carlo method for estimating the optimal input. The corresponding sections in the paper were also written by the author of this thesis.

Paper E

Paper E of this thesis is an edited version of,

J. Dahlin, F. Lindsten, T. B. Schön, and A. Wills. Hierarchical Bayesian ARX models for robust inference. In *Proceedings of the 16th IFAC Symposium on System Identification (SYSID)*, Brussels, Belgium, July 2012b.

Abstract: Gaussian innovations are the typical choice in most ARX models but using other distributions such as the Student- t could be useful. We demonstrate that this choice of distribution for the innovations provides an increased robustness to data anomalies, such as outliers and missing observations. We consider these models in a Bayesian setting and perform inference using numerical procedures based on MCMC methods. These models include automatic order determination by two alternative methods, based on a parametric model order and a sparseness prior, respectively. The methods and the advantage of our choice of innovations are illustrated in three numerical studies using both simulated data and real EEG data.

Contributions and background: The author of this thesis contributed to parts of the the implementation, generated most of the numerical illustrations and wrote the sections covering the numerical illustrations and the conclusions in the paper. The EEG data was kindly provided by Eline Borch Petersen and Thomas Lunner at Eriksholm Research Centre, Oticon A/S, Denmark.

1.3 Publications

Published work of relevance to this thesis are listed below in reverse chronological order. Items marked with $\star$ are included in Part II of this thesis.

- $\star$ J. Dahlin, T. B. Schön, and M. Villani. Approximate inference in state space models with intractable likelihoods using Gaussian process optimisation. Technical Report LiTH-ISY-R-3075, Department of Electrical Engineering, Linköping University, Linköping, Sweden, April 2014c.
- $\star$ J. Dahlin, F. Lindsten, and T. B. Schön. Particle Metropolis-Hastings using gradient and Hessian information. *Pre-print*, 2014b. arXiv:1311.0686v2.
- $\star$ J. Dahlin and F. Lindsten. Particle filter-based Gaussian process optimisation for parameter inference. In *Proceedings of the 19th IFAC World Congress*, Cape Town, South Africa, August 2014. (accepted for publication).
- J. Dahlin, F. Lindsten, and T. B. Schön. Second-order particle MCMC for Bayesian parameter inference. In *Proceedings of the 19th IFAC World Congress*, Cape Town, South Africa, August 2014a. (accepted for publication).
- $\star$ P. E. Valenzuela, J. Dahlin, C. R. Rojas, and T. B. Schön. A graph/particle-based method for experiment design in nonlinear systems. In *Proceedings of the 19th IFAC World Congress*, Cape Town, South Africa, August 2014. (accepted for publication)
- J. Dahlin, F. Lindsten, and T. B. Schön. Particle Metropolis Hastings using Langevin dynamics. In *Proceedings of the 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vancouver, Canada, May 2013a.
- $\star$ J. Dahlin, F. Lindsten, T. B. Schön, and A. Wills. Hierarchical Bayesian ARX models for robust inference. In *Proceedings of the 16th IFAC Symposium on System Identification (SYSID)*, Brussels, Belgium, July 2012b.

Other published works related to but not included in the thesis are:

- J. Kronander, J. Dahlin, D. Jönsson, M. Kok, T. B. Schön, and J. Unger. Real-time Video Based Lighting Using GPU Raytracing. In *Proceedings of the 2014 European Signal Processing Conference (EUSIPCO)*, Lisbon, Portugal, September 2014a. (submitted, pending review).
- J. Kronander, T. B. Schön, and J. Dahlin. Backward sequential Monte Carlo for marginal smoothing. In *Proceedings of the 2014 IEEE Statistical Signal Processing Workshop (SSP)*, Gold Coast, Australia, July 2014b. (accepted for publication).

D. Hultqvist, J. Roll, F. Svensson, J. Dahlin, and T. B. Schön. Detection and positioning of overtaking vehicles using 1D optical flow. In *Proceedings of the IEEE Intelligent Vehicles (IV) Symposium*, Dearborn, MI, USA, June 2014. (accepted for publication).

J. Dahlin and P. Svenson. Ensemble approaches for improving community detection methods. *Pre-print*, 2013. arXiv:1309.0242v1.

J. Dahlin, F. Lindsten, and T. B. Schön. Inference in Gaussian models with missing data using Equalisation Maximisation. *Pre-print*, 2013b. arXiv:1308.4601v1.

J. Dahlin, F. Johansson, L. Kaati, C. Mårtensson, and P. Svenson. A Method for Community Detection in Uncertain Networks. In *Proceedings of International Symposium on Foundation of Open Source Intelligence and Security Informatics 2012*, Istanbul, Turkey, August 2012a.

J. Dahlin and P. Svenson. A Method for Community Detection in Uncertain Networks. In *Proceedings of 2011 European Intelligence and Security Informatics Conference*, Athens, Greece, August 2011.

2

Nonlinear state space models and statistical inference

In this chapter, we introduce the SSM and give some motivating examples of different applications in which the model is used. We also review ML inference and Bayesian inference in connection with SSMs. Interested readers are referred to Douc et al. (2014), Cappé et al. (2005), Ljung (1999), Shumway and Stoffer (2010) and Brockwell and Davis (2002), for more detailed accounts of the topics covered here.

2.1 State space models and inference problems

An SSM or *hidden Markov model* (HMM) consists of a pair of discrete-time stochastic processes¹ $x_{0:T} \triangleq \{x_t\}_{t=0}^T$ and $y_{1:T} \triangleq \{y_t\}_{t=1}^T$. Here, $x_t \in \mathbf{X} \in \mathbb{R}^n$ denotes the *latent state* and $y_t \in \mathbf{Y} \in \mathbb{R}^m$ denotes the *observation* obtained from the system at time t .

The latent state is modelled as a Markov chain with initial state $x_0 \sim \mu(x_0)$ and the transition kernel $f_\theta(x_{t+1}|x_t, u_t)$. Furthermore, we assume that the observations are mutually independent given the latent states and have the conditional observation density $g_\theta(y_t|x_t, u_t)$. In both kernels, $\theta \in \Theta \subseteq \mathbb{R}^d$ denotes the static parameter vector of the Markov kernel and u_t denotes a known input to the system. With these definitions, we can write the SSM on the compact form

$$x_0 \sim \mu(x_0), \tag{2.1a}$$

$$x_{t+1}|x_t \sim f_\theta(x_{t+1}|x_t, u_t), \tag{2.1b}$$

$$y_t|x_t \sim g_\theta(y_t|x_t, u_t), \tag{2.1c}$$

¹In this thesis, we do not make any difference in notation between a random variable and its realisation. This is done to ease the notation.

Figure 2.1: Graphical model of an SSM with latent process (red) and observed process (blue).

which we make use of in this thesis. This is a fairly general class of models and can be used to model both nonlinear and non-Gaussian systems.

Another popular description of stochastic models like the class of SSMs is *graphical model* (Murphy, 2012; Bishop, 2006). The corresponding graphical representation of an SSM is depicted in Figure 2.1, where the latent state process is presented in red and the observed process in blue. From this graphical model, we see that the state x_t is only dependent on the last state x_{t-1} due to the *Markov property* inherent in the model. That is, all the information about the past is summarised in the state at time $t-1$. Also, we see that the observations are mutually independent given the states, as there are no arrows directly between two observations.

In this thesis, we are interested in two different inference problems connected to SSMs: (i) the *state inference* problem and (ii) the *parameter inference* problem. The first problem is to infer the density of the latent state process given the observations and the model. If we are interested in the current state given all the observations up until now, we would like to estimate the *marginal filtering density* $p_\theta(x_t|y_{1:t})$. We return to this and other related problems in Chapter 3. There, we define the problem in mathematical terms and present numerical methods designed to approximate the filtering and smoothing densities.

The second problem is to infer the values of the parameter θ given the set of observations $y_{1:T}$ and the model structure encoded by the Markov transition kernels $f_\theta(x_{t+1}|x_t)$ and $g_\theta(y_t|x_t)$. It turns out that we have to solve the state inference problem as a part of the parameter inference problem. We later return to the mathematical formulation of this problem in the ML setting in Section 2.3 and in the Bayesian setting in Section 2.4. These problems are analytically intractable and cannot be computed in closed-form. Therefore, we present computational methods based on sampling methods for parameter inference in Chapter 4.

2.2 Some motivating examples

SSMs have been successfully applied in various areas for modelling dynamical systems. In this section, we give three examples from different research fields and connect them with the inference problems discussed in the previous section. The first model is taken from finance, where we would like to model the real-valued latent volatility given some stock or exchange rate data. In the second model, we would like to make predictions of the number of annual major earthquakes. The third model is taken from meteorology, where we would like to predict the probability of rain fall during the coming days. However, we start with the well-known linear Gaussian state space (LGSS) model, which we make use of as a benchmark throughout the thesis.

2.2.1 Linear Gaussian model

Consider the scalar LGSS model²,

$$x_{t+1}|x_t \sim \mathcal{N}(x_{t+1}; \phi x_t + \gamma u_t, \sigma_v^2), \quad (2.2a)$$

$$y_t|x_t \sim \mathcal{N}(y_t; x_t, \sigma_e^2), \quad (2.2b)$$

where the parameter vector is $\theta = \{\phi, \gamma, \sigma_v, \sigma_e\}$. Here, ϕ describes the persistence of the state and $\{\sigma_v, \sigma_e\}$ controls the noise levels. In this model, we have added an optional input $u_{1:T}$ to the system, which is scaled by the parameter γ . Here, we require that $\phi \in (-1, 1) \subset \mathbb{R}$ to obtain a stable system and that $\{\sigma_v, \sigma_e\} \in \mathbb{R}_+^2$ as they correspond to standard deviations.

The state inference problem can be solved exactly for this model using Kalman filters and smoothers (Kailath et al., 2000). This is a result of that the model is linear and only includes Gaussian kernels. Due to this property, we make use of the model as a benchmark problem for some of the algorithms reviewed in this thesis. Comparing the methods that we develop for the nonlinear SSMs with the LGSS model can reveal important properties of the algorithms and help with insights about how to calibrate them. The Kalman methods can also be used to estimate the log-likelihood, the score function and the information matrix, which are important quantities in the ML parameter inference problem discussed in Section 2.3.

2.2.2 Volatility models in econometrics and finance

Nonlinear SSMs are often encountered in econometric and financial problems, where we would like to e.g. model the variations in the log-returns of a stock or an index. The log-returns are calculated by $y_t = \log(s_t/s_{t-1})$, where s_t denotes the price of some financial asset at time t . The variations in the log-returns can be seen as the instantaneous standard deviation and is referred to as the *volatility*.

²This model is also known as ARX(1) in noise, where ARX(1) denotes an exogenous autoregressive process of order 1.

The volatility plays an important role in the famous Black-Scholes pricing model (Black and Scholes, 1973). In this model, the log-returns are assumed to follow a Brownian motion with independent and identically distributed IID increments distributed according to some Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$, where σ denotes the volatility. Therefore, the volatility is an important component when calculating the price of options and other financial instrument based on the Black-Scholes model. For a discussion of how the volatility is used for pricing options, see Hull (2009), Björk (2004) and Glasserman (2004). For a more extensive treatment of financial time series and alternative inference methods for volatility models, see Tsay (2005).

In Figure 2.2, we present the closing prices and daily log-returns for the the NASDAQ OMX Stockholm 30 Index during a 14 year period. Note, the large drops in the closing prices (middle) in the period around the years 2001, 2008 and 2011 in connection with the most recent financial crises (shocks). At these drops, we see that the log-returns are quite volatility, as they vary much between consecutive days. During other periods, the volatility is quite low and the log-returns does not vary much between consecutive days. These variations in the volatility are referred to as *volatility clustering* in finance.

Also, from the QQ-plots we see that the log-returns are heavy-tailed and clearly non-Gaussian with large deviations from theoretical quantiles in the tail behaviour. All these features (and some other) are known as *stylized facts* (Cont, 2001). In this section, we present three different volatility models that tries to capture different aspects of the observed properties of financial data. A complication is that the resulting inference problems become more challenging as when try to capture more and more of the stylized facts. This results in a trade-off between accuracy of the model and the computational complexity of the resulting inference problems.

A common theme for all the models considered here, is that they are all based on a (slowly) varying random walk-type model of the volatility. This model can be motivated by the volatility clustering behaviour, i.e. the underlying volatility varies slowly and gives rise to volatility clustering. Furthermore, the log-returns are modelled as a non-stationary white noise process where the variance is determined by the latent volatility. This corresponds quite well with the log-returns presented in the upper part of Figure 2.2. For a more through discussion about different volatility models, see Mitra (2011) and Kim et al. (1998).

The first model is the generalised autoregressive conditional heteroskedasticity (GARCH) model (Bollerslev, 1986), which is a generalisation of the ARCH model (Engle, 1982). Here, we consider the GARCH(1,1) in noise model given by

$$h_{t+1}|x_t, h_t = \alpha + \beta x_t^2 + \gamma h_t, \quad (2.3a)$$

$$x_{t+1}|x_t, h_t = \mathcal{N}(x_{t+1}; 0, h_{t+1}), \quad (2.3b)$$

$$y_t|x_t = \mathcal{N}(y_t; x_t, \tau^2), \quad (2.3c)$$

where the parameter vector is $\theta = \{\alpha, \beta, \gamma, \tau\}$ with the constraints $\{\alpha, \beta, \gamma, \tau\} \in$

Figure 2.2: Daily log-returns (upper) for the NASDAQ OMX Stockholm 30 Index from 2000-01-04 to 2014-03-14. The daily closing prices (middle), QQ-plot of the log-returns (lower left) and histogram of the log-returns and the kernel density estimate (KDE) (lower right) are also presented.

$\mathbb{R}_+^4$ and $\beta + \gamma \in (0, 1) \subset \mathbb{R}$ for stability. In this model, the current log-return is taken into account when computing the volatility at the next time step. This construction tries to capture the volatility clustering.

The second model is the Hull-White stochastic volatility (HWSV) model³ (Hull and White, 1987) given by

$$x_{t+1}|x_t \sim \mathcal{N}(x_{t+1}; \mu + \phi(x_t - \mu), \sigma^2), \quad (2.4a)$$

$$y_t|x_t \sim \mathcal{N}(y_t; 0, \beta^2 \exp(x_t)), \quad (2.4b)$$

where the parameter vector is $\theta = \{\mu, \phi, \sigma, \beta\}$ with the constraints $\{\mu, \sigma, \beta\} \in \mathbb{R}_+^3$ and $\phi \in (-1, 1) \subset \mathbb{R}$ for stability. There are many variants of the HWSV model that includes correlations between the noise sources (called leverage models) and outliers in the form of jump processes. See Chib et al. (2002) and Jacquier et al. (2004) for more information.

One problem with HWSV is that the Gaussian observation noise in some cases cannot fully capture the heavy-tail behaviour found in real-world data. Instead, e_t is often assumed to be simulated from a Student- t distribution, which has heavier tails than the Gaussian distribution. Another modification is to assume that e_t is generated from an α -stable distribution, which can model both large outliers in the data and non-symmetric log-returns. This results in the third model, the stochastic volatility model with symmetric α -stable returns (SV α) Casarin (2004) given by

$$x_{t+1}|x_t \sim \mathcal{N}(x_{t+1}; \mu + \phi x_t, \sigma^2), \quad (2.5a)$$

$$y_t|x_t \sim \mathcal{A}(y_t; \alpha, 0, \exp(x_t/2), 0), \quad (2.5b)$$

where the parameter vector is $\theta = \{\mu, \phi, \sigma, \alpha\}$ and the constraints $\{\mu, \sigma\} \in \mathbb{R}_+^2$, $\alpha \in (0, 2] \setminus \{1\} \subset \mathbb{R}$ and $\phi \in (-1, 1) \subset \mathbb{R}$ for stability. Here, $\mathcal{A}(\alpha, 0, 1, 0)$ denotes a symmetric α -stable distribution⁴ with stability parameter α . For this distribution, we cannot evaluate $g_\theta(y_t|x_t)$ and this results in problems when inferring the states and parameter vector of the model. In Paper C, we apply ABCs for solving this problem.

As previously discussed, the inference problem in volatility models is mainly state inference, which requires a model and hence results in the need for parameter inference. For example, the state estimate can be used as the volatility estimate for option pricing. Also, the parameter vector of the model can be used to analyse if the log-return are symmetric and heavy-tailed or the persistence of the underlying volatility process.

³A similar model is used for modelling the glacial varve thickness (the thickness of the clay collected within the glacial) in Shumway and Stoffer (2010). This model is obtained by replacing the noise in (2.4b) with gamma distributed noise, i.e. $e_t \sim \mathcal{G}(\alpha^{-1}, \alpha)$ for some parameter $\alpha \in \mathbb{R}_+$.

⁴See Appendix A of Paper C for a brief summary about α -stable distributions and their properties. For a more detailed presentation, see Nolan (2003).

Figure 2.3: The major earthquakes in the world between 2000 and 2013. The size of the circle is proportional to the relative magnitude of the earthquake.

2.2.3 Earthquake count model in geology

Another application of SSMs is to model the number of major (with magnitude 7 or higher on the Richter scale) earthquakes each year. As a motivation for the model, we consider some real-world data from the *Earthquake Data Base System of the U.S. Geological Survey*⁵. The data describes the number of major earthquakes around the world between the years 1900 and 2013. In Figure 2.3, we present the locations of the major earthquakes during the period 2000 to 2013. In Figure 2.4, we present the annual number of major earthquakes with some exploratory plots. From the data, we see that the number of earthquakes is clearly correlated, similar to the clustering behaviour found in the volatility models from the previous application. That is, a year with many earthquakes is likely to be followed by another year with high earthquake intensity. The reason for this follows quite intuitive as pointed out in Langrock (2011) since earthquakes are due to stresses in the tectonic plates of the Earth. Therefore, the underlying process which model these stresses should be slowly varying over the time span of years.

Due to the autocorrelation in the number of earthquakes, it is reasonable to assume that the intensity is determined by some underlying latent variable. Therefore, we can model the number of major earthquakes as an SSM. To this end, we use the model⁶ from Zeger (1988) and Chan and Ledolter (1995), which assumes that the number of earthquakes y_t is a Poisson distributed variable (corresponding a positive integer or count data). Also, we assume that the mean of the Poisson process λ_t follows an AR(1) process,

$$\log(\lambda_t) - \mu = \phi \left(\log(\lambda_{t-1}) - \mu \right) + \sigma_v v_t,$$

where v_t denotes a standard Gaussian random variable. By introducing $x_t = \log(\lambda_t) - \mu$ and $\beta = \exp(\mu)$, we obtain the SSM

$$x_{t+1}|x_t \sim \mathcal{N}(x_{t+1}; \phi x_t, \sigma_v^2), \quad (2.6a)$$

$$y_t|x_t \sim \mathcal{P}(y_t; \beta \exp(x_t)), \quad (2.6b)$$

where the parameter vector is $\theta = \{\phi, \sigma_v, \beta\}$ with the constraints $\phi \in (-1, 1) \subset \mathbb{R}$ and $\{\sigma_v, \beta\} \in \mathbb{R}_+^2$. Here, $\mathcal{P}(\lambda)$ denotes a Poisson distributed variable with mean λ . That is, the probability of $k \in \mathbb{N}$ earthquakes during year t is given by the probability mass function (PMF),

$$\mathbb{P}[N_t = k] = \frac{\exp(-\lambda) \lambda^k}{k!}.$$

The inference problem in this type of model could be to determine the underlying intensity of the process (the state). This information could be useful in making predictions of the future number of major earthquakes.

⁵This data can be accessed from <http://earthquake.usgs.gov/earthquakes/eqarchives/>.

⁶This type of Poisson count model can also be used to model the number of yearly polio infections (Zeger, 1988) and the number of transactions per minute of a stock on the market (Fokianos et al., 2009).

Figure 2.4: Upper: number of annual major earthquakes (with magnitude 7 or higher on the Richter scale) during the period between the years 1900 and 2013. Middle: the corresponding histogram with the KDE (blue). Lower: the estimated autocorrelation function (ACF).

2.2.4 Daily rainfall models in meteorology

A common problem in meteorology and weather forecasting is to estimate the probability and amount of rainfall. In practice, this problem is often split into two subproblems, each with its own model. The first model determines the probability of rainfall and the second model determines the amount of rainfall. For more information, see Srikanthan and McMahon (1999) and Woolhiser (1992). Here, we consider the problem to construct a model to determine the probability of rainfall given historic data of rainfall in the region of interest. It is also possible to use user data from the Internet to predict the probability of rainfall in some region. An example of this is to make use of some collected data from Twitter, see Naesseth (2012) for more information about this approach.

We first consider some real-world data from the Swedish weather service (SMHI) collected daily at Malmslätt near Linköping during the period between the years 1952 and 2002. The data is presented in Figure 2.5 as the daily probability of rainfall (upper) and the average daily amount of rainfall (middle) calculated per week. We also present the ACF of rainfall, which indicates that there is a correlation between rainy and non-rainy days. Furthermore, the probability of rainfall seems to follow a cyclic behaviour with a high probability during the latter part of the year. The amount of rainfall also varies with a peak around week 30.

To construct a model of the probability of rainfall, we follow the insights from the previous analysis and review the model proposed by Langrock and Zucchini (2011). To account for the autocorrelation in the rainfall, we make use of a latent process to describe the persistence of the weather. This construction can be used to create a rough model to account for the structure of low pressure weather systems which passes over a period of days. Hence, this can be seen as a short term model of the probability of rainfall. From practical knowledge, we also know that the probability of rainfall is connection with the season of the year. This cyclic behaviour was also seen in the weather data from Malmslätt. Therefore, we assume that there also exists a cyclical part of the latent process and that this together with the short term persistence determines the probability of rainfall.

Furthermore, we assume that the output from the model is a binary variable $y_t \sim \mathcal{B}(p_t)$ from a Bernoulli distribution with success probability p_t . That is, the variable assumes the value 1 with probability p_t (rain falls during day t) and the value 0 with probability $1 - p_t$ (no rain falls during day t). The final model on SSM form is given by

$$h_t = \sum_{k=1}^2 \alpha_k \cos\left(\frac{2k\pi t}{365}\right) + \sum_{k=1}^2 \beta_k \sin\left(\frac{2k\pi t}{365}\right), \quad (2.7a)$$

$$x_{t+1}|x_t \sim \mathcal{N}\left(x_{t+1}; \phi x_t, \sigma_v^2\right), \quad (2.7b)$$

$$y_t \sim \mathcal{B}\left(\frac{\exp(\mu + x_t + h_t)}{1 + \exp(\mu + x_t + h_t)}\right), \quad (2.7c)$$

where the parameter vector is $\theta = \{\phi, \sigma_v, \mu, \alpha_1, \alpha_2, \beta_1, \beta_2\}$ with the constraints

Figure 2.5: The daily probability of rainfall (upper), the daily amount of rainfall (middle) and the ACF of rainfall (lower). The values are calculated as weekly averages of daily data from Malmöslätt during the period between the years 1952 and 2002. The data is provided by the Swedish weather service (SMHI) and is used under the creative commons license.

$\phi \in (-1, 1) \subset \mathbb{R}$ and $\sigma_v \in \mathbb{R}_+$. The inference problem in this model could be to determine the probability of rainfall (estimate p_t) given the data. Also, it could be interesting to determining the strength of the persistence in the system determined by ϕ or the strength of the seasonal components determined by $\{\alpha_1, \alpha_2, \beta_1, \beta_2\}$.

2.3 Maximum likelihood parameter inference

In this section, we present the fundamentals of ML based parameter inference in SSMs. This presentation is mainly included to set the notation and to highlight some features of the method that are needed in the sequel. An accessible general introduction to ML inference is given by Casella and Berger (2001). For more extensive treatments, see Rao (1965) and Lehmann and Casella (1998).

Inference in the ML paradigm is focused on optimising the likelihood function $\mathcal{L}(\theta) = p_\theta(y_{1:T})$, which also appears in Bayesian inference. The likelihood encodes the information contained with the observations $y_{1:T}$ into a quantity that can be used for inference.

2.1 Definition (Likelihood function for an SSM). The *likelihood (function)* of an SSM can be expressed as the decomposition

$$\mathcal{L}(\theta) = p_\theta(y_{1:T}) = p_\theta(y_1) \prod_{t=2}^T p_\theta(y_t | y_{1:t-1}), \quad (2.8)$$

where $p_\theta(y_t | y_{1:t-1})$ denotes the one-step-ahead predictor.

It is common to replace the likelihood function with the *log-likelihood (function)* in many inference problems. This is done to simplify analytical calculations and improve the numerical stability of many algorithms. The log-likelihood is given by

$$\ell(\theta) = \log p_\theta(y_{1:T}) = \log p_\theta(y_1) + \sum_{t=2}^T \log p_\theta(y_t | y_{1:t-1}). \quad (2.9)$$

In general, there are two different interpretations of the likelihood in statistics. The first is that the likelihood is a *function of the data* for a fixed parameter θ . A common name for this distribution is the *sampling distribution* and it plays an important role when calculating the distribution of some sampled data. The second interpretation (which we adopt in this thesis) is that the likelihood is a function of the parameter. That is the data is fixed and that the likelihood therefore summaries the information in the data.

The ML parameter inference problem is formulated as a maximisation problem of the likelihood or equivalently the log-likelihood. This follows from that the logarithm is a monotone function and hence any maximiser of the likelihood is also a maximiser of the log-likelihood.

2.2 Definition (Maximum likelihood parameter inference problem). The parameter inference problem in the ML setting is given by

$$\hat{\theta}_{\text{ML}} = \operatorname{argmax}_{\theta \in \Theta} \mathcal{L}(\theta) = \operatorname{argmax}_{\theta \in \Theta} \ell(\theta), \quad (2.10)$$

where $\hat{\theta}_{\text{ML}}$ denotes the ML parameter estimate. _____

The interpretation of (2.10) is that we should select the parameter that together with the model is the most likely to have been generated the observations. This definition makes good intuitive sense, but as we shall see in the following section it is not the only method for estimating the parameter given the data.

We continue by discussing some useful quantities connected with the log-likelihood and will then return to discuss the properties of the estimator in (2.10). The gradient of the log-likelihood is referred to as the *score function* and the negative Hessian of the log-likelihood is referred to as the *observed information matrix*. These quantities are useful in many optimisation algorithms as they are the first order and second order information about the objective function (the log-likelihood) in (2.10), respectively. Also, this information can be used to build efficient proposals in some sampling algorithms, see Paper A.

2.3 Definition (Score function). The *score function* is defined as the gradient of the log-likelihood,

$$\mathcal{S}(\theta') = \nabla \ell(\theta) \big|_{\theta=\theta'}, \quad (2.11)$$

where the gradient is taken with respect to the parameter vector. _____

The score function has a natural interpretation as the slope of the log-likelihood. Hence, the score function is zero when evaluated at the true parameter vector, $\mathcal{S}(\theta^*) = 0$. However, note that this is not necessarily true when we work with finite data samples as is discussed in Example 2.6.

2.4 Definition (Observed information matrix). The *observed information matrix* is defined as the negative Hessian of the log-likelihood,

$$\mathcal{J}(\theta') = -\nabla^2 \ell(\theta) \big|_{\theta=\theta'} \quad (2.12)$$

where the Hessian is taken with respect to the parameter vector. _____

The statistical interpretation of the observed information matrix is as a measure of the amount of information in the data regarding the parameter θ . That is, if the data is informative the resulting information matrix is large (according to some measure). Also, the information matrix can geometrically be seen as the negative curvature of the log-likelihood. As such, we expect it to be positive definite (PD) at the ML parameter estimate (c.f. the second-derivative test in basic calculus). Finally, we note that there exists a limiting behaviour for the observed information matrix, which approaches the so called expected information matrix as the number of data points tends to infinity.

2.5 Definition (Expected information matrix). The *expected information matrix* (or the Fisher information matrix) is defined by the expected value of the observed information matrix (2.12),

$$\mathcal{I}(\theta') = -\mathbb{E}_{y_{1:T}} \left[\nabla^2 \ell(\theta) \big|_{\theta=\theta'} \right] = \mathbb{E}_{y_{1:T}} \left[\left(\nabla \ell(\theta) \big|_{\theta=\theta'} \right)^2 \right], \quad (2.13)$$

which is evaluated with respect to the data record. _____

Note, that the expected information matrix is independent of the data realisation, whereas the observed information is dependent on the realisation. The expected information matrix is PD for all values of θ as it can be seen as the variance of the score function. We make use of this property in Section 4.2.2 and in Paper A to construct a random walk on a Riemann manifold using the information matrices. We conclude the discussion on the score function and the information matrix by Example 2.6, where we investigate the defined quantities for an LGSS model as a function of the parameter ϕ .

2.6 Example: Score and information matrix in the LGSS model

Consider the LGSS model (2.2) with the parameter vector $\theta^* = \{0.5, 0, 1, 0.1\}$ from which we generate a realisation of length $T = 250$ using the initial value $x_0 = 0$. We fix $\{\sigma_v, \sigma_e\}$ at their *true* values and create a grid over ϕ . For each grid point, we calculate the log-likelihood, the score function and the expected information matrix. The results are presented in Figure 2.6.

We note that the log-likelihood has a distinct maximum near the *true* parameter (presented as dotted vertical lines) and that the score function is zero close to this point. The small difference in the score function is due to that we use a finite amount of data. Finally, the zero of the score function results in a maximum of the log-likelihood function as the expected information (negative Hessian) is positive. _____

With the definition of the expected information matrix in place, we now return to discussing the properties of the ML parameter estimate. The ML estimator obtained from (2.10) has a number of strong asymptotic properties, i.e. when the number of observations tends to infinity. It is possible to show that this estimator is *consistent*, *asymptotically normal* and *efficient* under some regularity conditions. These conditions include that the parameter space is compact and that the likelihood, score function and information matrix exist and are well-behaved.

The (ML) estimator is said to be *consistent* as it fulfils that

$$\hat{\theta}_{\text{ML}} \xrightarrow{a.s.} \theta^*, \quad T \rightarrow \infty.$$

That is, the estimate almost surely converges to the true value of the parameter in the limit of infinite data. Furthermore, as the estimator is asymptotically normal, we have that the error in the estimate satisfies a CLT given by

$$\sqrt{T} \left(\hat{\theta}_{\text{ML}} - \theta^* \right) \xrightarrow{d} \mathcal{N} \left(0, \mathcal{I}^{-1}(\theta^*) \right),$$

Figure 2.6: The estimates of the log-likelihood function (upper) the score function (lower left) and the expected information matrix (lower right) of the LGSS model in Example 2.6. The dotted lines indicate the true parameter and the zero-level.

which follows from a Taylor expansion around the point $\theta = \theta^*$ of the log-likelihood. Note that, the expected information matrix enters into the expression and limits the accuracy of the estimate.

Therefore, a natural question is if we can somehow change the size of this matrix to decrease the lower-bound on the variance of the estimate. This is a key problem in the field of input design, where an input is added to the SSM to maximise some scalar function of the expected information matrix. This problem is discussed for nonlinear SSMs in Example 4.10 and Paper D.

Lastly, we say that an estimator is *efficient* if it attains the *Cramér-Rao lower bound*, which means that no other consistent estimator has a lower mean square error (MSE). That is, the ML estimator is the best unbiased estimator in the MSE sense and there are no better unbiased estimators. This last property is appealing and one might be tempted to say that this estimator is the *best* choice for parameter inference. However, this result is only valid when we have an infinite number of samples. Therefore, other estimators (e.g. Bayes estimators discussed in the next section) could have better properties in the finite sample regime. Also, there can exist biased estimators with lower MSE than the ML estimator.

We have now introduced the ML parameter inference problem and some of the properties of the resulting estimate. In practice, we usually encounter a number of problems when we try to solve the optimisation problem in (2.10). These include that for nonlinear SSMs: (i) the log-likelihood, the score and the information matrix are all intractable, (ii) the optimisation problem cannot be solved in closed-form and (iii) there could exist many local maxima of the log-likelihood making numerical optimisation problematic.

To solve these problems, a number of different numerical approaches have been developed. We return to the use of these approaches for estimating the log-likelihood, the score and the information matrix in Section 3. We also discuss other numerical methods for solving the ML parameter inference problem in Chapter 4.

2.4 Bayesian parameter inference

Previously in the ML paradigm, we implicitly assumed that the true parameter is a specific value. In this section, we instead assume that the true parameter is distributed according to some probability distribution. From this assumption, we can construct a different statistical paradigm called Bayesian inference, which can be used for state and parameter inference in nonlinear SSMs.

As in the previous section, we only briefly discuss Bayesian inference for SSMs to set the notation and highlight some important features that we make use of in this thesis. For general treatments of Bayesian analysis, see Robert (2007) and Berger (1985). Furthermore, two accessible introductions are Gelman et al. (2013) and Casella and Berger (2001).

In the Bayesian parameter inference problem, we combine the information in the

data described by the likelihood $p_\theta(y_{1:T})$ with prior information encoded as a probability distribution denoted $p(\theta)$. This combination of the prior information and the information in the data is made using Bayes' theorem. This procedure is referred to as the *prior-posterior update* in Bayesian inference. The result is an updated probability distribution called the *posterior distribution*, which we denote $p(\theta|y_{1:T})$ in the parameter inference problem. Similar distributions can be defined for the state inference problem and we return to this in Section 3.1.

2.7 Definition (Bayesian parameter inference problem). Given the parameter prior $p(\theta)$ and the likelihood $p_\theta(y_{1:T})$, we obtain the *parameter posterior* by Bayes' theorem as

$$p(\theta|y_{1:T}) = \frac{p_\theta(y_{1:T})p(\theta)}{p(y_{1:T})} \propto p_\theta(y_{1:T})p(\theta), \quad (2.14)$$

where $p(y_{1:T})$ denotes the *marginal likelihood* (or the *evidence*). The name is due to that it can be computed by marginalisation

$$p(y_{1:T}) = \int p_\theta(y_{1:T})p(\theta) d\theta. \quad (2.15)$$

One of the main questions in Bayesian inference is the choice of the prior and how we may encode our prior beliefs about the data into it. A common view is that this choice is *subjective* as it is often done using intuition and previous knowledge, which depends on subjective experiences. In this thesis, we make use of simple priors to encode stability properties of the nonlinear system or to keep the standard deviation positive at all times. For this, we make use of uniform priors over different subsets of the parameter space. An example of this is that we assume a uniform prior over $\phi \in (-1, 1) \subset \mathbb{R}$ for the LGSS model (2.2), which can be expressed as $p(\phi) = \mathcal{U}(\phi; -1, 1)$. Note, that the use of improper priors can be seen as a link between ML and Bayesian inference, but we shall not discuss this further. Instead, interested readers are referred to Robert (2007) for more information.

Another important class of priors that we make use of in Paper E is *conjugate priors*, which enables us to compute closed-form expressions for the posterior. A conjugate prior has the same functional form as the posterior and depends on the form of the likelihood. For example, the conjugate prior for the mean (given that the variance is known) of the Gaussian distribution is again a Gaussian distribution. This results from the fact that the product of two Gaussian distributions (the likelihood and the prior) is again a Gaussian distribution and hence it is a conjugate prior. We give another example of a conjugate prior in Example 2.8. Conjugate priors exist mainly for some combinations of priors and likelihoods in the exponential family of distributions, see Robert (2007) for a discussion.

Figure 2.7: The posterior (upper) of the parameter σ^2 in the Gaussian distribution resulting from the combination of an inverse-Gamma prior (lower left) and the data log-likelihood (lower right).

2.8 Example: Prior-posterior update in a conjugate model

Consider the problem of inferring the value of the variance σ^2 in a Gaussian distribution given that we know the value of the mean μ . Also, we assume that we have obtained a set of IID data $y_{1:T}$ generated by the model. The conjugate prior to the variance in a Gaussian distribution is the inverse-Gamma distribution with rate α_0 and shape β_0 denoted $\mathcal{IG}(\alpha_0, \beta_0)$. By direct calculation, we obtain that the posterior has the parameters $\alpha_T = \alpha_0 + T/2$ and $\beta_T = \beta_0 + \frac{1}{2} \sum_{i=1}^T (y_i - \mu)^2$.

Here, we simulate the data realisation using the parameter $\{\mu, \sigma^2\} = \{1.5, 4\}$ of length $T = 200$ and use the prior coefficients $\{\alpha_0, \beta_0\} = \{1, 4\}$. The corresponding posterior, prior and log-likelihood are presented in Figure 2.7. We note that the prior assigns a small probability of the variance equal to 4. However, as the data record is relatively large, the resulting posterior is centred close to the true value of the parameter.

The Bayesian parameter inference problem is completely described by (2.14) and everything that we need to know is encoded in the posterior. This follows from the *likelihood principle*, which states that all information obtained from the data is contained within the likelihood function, see Robert (2007). However, we are sometimes interested in computing *point estimates* of the parameter vector. To compute point estimates, we can make use of *statistical decision theory* to make a decision about what information from the posterior to use. Consider a *loss function* $L : \Theta \times \Theta \rightarrow \mathbb{R}_+$, which takes the parameter and its estimate as input and returns a real-valued positive loss. The *expected posterior loss* (or posterior risk) $\rho(p(\theta), \delta|y_{1:T})$ is given by

$$\rho(p(\theta), \delta|y_{1:T}) = \int L(\theta, \delta(y_{1:T})) p(\theta|y_{1:T}) d\theta, \quad (2.16)$$

where $\delta(y_{1:T})$ denotes the decision of the parameter estimate given the data. The *Bayes estimator* is defined as the minimising argument of the expected posterior loss,

$$\delta^*(y_{1:T}) = \operatorname{argmin}_{\delta(y_{1:T}) \in \Theta} \rho(p(\theta), \delta|y_{1:T}).$$

Here, we restrict ourselves to discussing the resulting Bayes estimators for three different loss functions in Table 2.1, but there are many other possibly suitable choices for our application. From this, we see that the estimate that minimises the quadratic loss function is the expected value of the posterior,

$$\mathbb{E}[\theta] = \int \theta p(\theta|y_{1:T}) d\theta. \quad (2.17)$$

That is, the Bayes estimator is given by the posterior mean for this choice of loss function. This is an example of a common expectation operation in Bayesian inference, as the expected value of the parameter is computed with respect to the parameter posterior distribution. Similar expressions can be written for many other Bayes estimators.

	Loss function	Bayes estimator
Linear	$L(\theta, \delta) = \theta - \delta $	Posterior median
Quadratic	$L(\theta, \delta) = (\theta - \delta)^2$	Posterior mean
0-1	$L(\theta, \delta) = \mathbb{I}(\theta \neq \delta)$	Posterior mode

Table 2.1: Different loss functions and the resulting Bayes estimator.

The statistical properties of the Bayes estimator depends on the choice of loss function when we work with finite data. However, it follows (under some conditions) from the *Bernstein–von Mises theorem* that the influence of the prior diminishes when the amount of data grows. Consequently, the MAP estimator (the posterior mode) converges to the ML estimator in some cases when the amount of data increases. Therefore it is possible to see ML parameter inference as a special case of Bayesian parameter inference in some sense. Finally, we note that Bayes estimators asymptotically have the same properties as the ML estimator, i.e. they are *consistent*, *asymptotically normal* and *efficient*. Also, as the number of data tends to infinity, the posterior tends to a Gaussian distribution which is known as the *Bayesian CLT*. For more information about the statistical properties of the Bayes estimators, see Robert (2007), Lehmann and Casella (1998) and Berger (1985).

We have now introduced the Bayesian paradigm for parameter and state inference in nonlinear SSMs. We have seen that the key quantity in Bayesian inference is the posterior distribution, which depends on the likelihood. Therefore, we in practice encounter the same problems with analytically intractable likelihoods as for the ML paradigm. Another complication is that the posterior might not be described by any known distribution. As a consequence, we cannot compute many of the integrals depending on the posterior distribution encountered in Bayesian analysis in closed-form. Examples of such integrals are the normalisation in (2.15), the marginalisation in (2.16) and the expectation in (2.17).

To counter these problem, we can make use of the same numerical methods from Chapter 3 as for the ML paradigm to estimate the likelihood function. To solve the Bayesian parameter inference problem, we can make use of sampling methods to obtain approximations of the posterior distributions of interest. These methods use computer simulations and have therefore only been available for practical use during the last decades. However, during this period these methods have been well-studied and are today established tools for Bayesian inference. Another approach is to make analytical approximations of the posterior distribution using known distributions or iterative procedures. We discuss some of these methods in Chapter 4

3

State inference using particle methods

In this chapter, we discuss the state inference problem in detail and presents some algorithms for approximate state inference. The foundation of state inference in SSMs is given by the *Bayesian filtering and smoothing recursions* that are discussed in Section 3.1. Unfortunately, these recursions are analytically intractable for most SSMs. Therefore, the remaining part of this chapter is devoted to discussing numerical methods based on SMC for approximate state inference. We also discuss how to make use of SMC methods to estimate the likelihood, the score function and the information matrix in SSMs. This chapter is concluded by discussing the direct illumination problem in computer graphics and how it can be solved using SMC methods.

3.1 Filtering and smoothing recursions

There are mainly two types of state inference problems in an SSM: *filtering* and *smoothing*. Both the filtering and smoothing problem can be *marginal* (inference on a single state), *k-interval* (inference of k states) or *joint* (inference on all states). In marginal filtering, only observations $y_{1:t}$ collected until the current time step t are used to infer the current value of the state x_t . In marginal smoothing, all the collected observations including (possibly) future observations $y_{1:T}$ are used to infer the value of the current state x_t with $t \leq T$. In Table 3.1, we summarise some common filtering and smoothing problems in SSMs.

The solutions to the filtering and smoothing problems are given by the Bayesian filtering and smoothing recursions (Jazwinski, 1970; Anderson and Moore, 2005). These (as the names suggest) are based on Bayesian inference similar to the parameter inference problems discussed in Section 2.4. These recursions can be used

Name	Density
(Marginal) filtering	$p_\theta(x_t y_{1:t})$
(Marginal) smoothing ($t \leq T$)	$p_\theta(x_t y_{1:T})$
Joint smoothing	$p_\theta(x_{0:T} y_{1:T})$
Fixed-interval smoothing ($s < t \leq T$)	$p_\theta(x_{s:t} y_{1:T})$
Fixed-lag smoothing (for lag Δ)	$p_\theta(x_{t-\Delta-1:t} y_{1:t})$

Table 3.1: Common filtering and smoothing densities in SSMs.

to iteratively solve the filtering problem for each time t using two steps given by

$$p_\theta(x_t|y_{1:t}) = \frac{p_\theta(x_t|y_{1:t-1})g_\theta(y_t|x_t)}{p_\theta(y_t|y_{1:t-1})}, \quad (3.1a)$$

$$p_\theta(x_t|y_{1:t-1}) = \int f_\theta(x_t|x_{t-1})p_\theta(x_{t-1}|y_{1:t-1})dx_{t-1}. \quad (3.1b)$$

In the first step, the state estimate at time t is updated with the new observation y_t (also known as the *measurement update*) using Bayes' theorem. In the second step, a marginalisation is used to predict the next state x_t using the information in the current state and the collected observations (also known as the *time update*). In a similar manner, the smoothing problem can be solved by iterating the marginalisation

$$p(x_t|y_{1:T}) = p(x_t|y_{1:t}) \int \frac{f(x_{t+1}|x_t)p(x_{t+1}|y_{1:T})}{p(x_{t+1}|y_{1:t})} dx_{t+1}, \quad (3.2a)$$

$$p(x_{t+1}|y_{1:t}) = \int f(x_{t+1}|x_t)p(x_t|y_{1:t}) dx_t, \quad (3.2b)$$

backwards in time. Here the smoothing recursion, makes use of the filtering distribution computed in a forward pass. The resulting smoother from this procedure is therefore known as the *forward filtering backward smoothing* (FFBSm) algorithm. We return to a numerical method based on the FFBSm algorithm for the use in SSMs later in this chapter.

The filtering (3.1) and smoothing (3.2) recursions can only be solved analytically for two different classes of SSMs: linear Gaussian SSMs and SSMs with finite state processes. For the former, the recursions can be solved by the *Kalman filter* (KF) and e.g. the *Rauch–Tung–Striebel* (RTS) smoother (Rauch et al., 1965), respectively. Using the Kalman methods, we can exactly compute the likelihood and the states in an LGSS model. We do not discuss the details of the Kalman methods here and refer readers to Anderson and Moore (2005) and Kailath et al. (2000) for extensive treatments of Kalman filtering and different Kalman smoothers.

The general intractability of the Bayesian filtering and smoothing recursions results from that there are no closed-form expressions for the densities in the recursions. This is similar to the problems that we encountered in the Bayesian parameter inference problem. Instead, we consider numerical approximations that relies on

Monte Carlo (MC) methods to estimate the filtering and smoothing distributions. This results in SMC and related methods, which we return to in Section 3.3.

3.2 Monte Carlo and importance sampling

MC methods are a collection of statistical simulation methods based on sampling and the *strong law of large numbers* (SLLN). For a general introduction to MC methods, see Robert and Casella (2004) for an extensive treatment or Ross (2012) for an accessible introduction. In this thesis, we mainly use MC methods for estimating the expected value (an integral) of an arbitrary well-behaved *test function* $\varphi(x)$,

$$\hat{\varphi} = \mathbb{E}_{\pi}[\varphi(x)] = \int \varphi(x)\pi(x) dx, \quad (3.3)$$

where $\pi(x)$ denotes a (normalised) *target distribution* from which we can simulate IID *particles* (or samples). As a consequence of the SLLN¹, we can estimate the expected value by the sample average

$$\hat{\varphi}_{\text{MC}} = \frac{1}{N} \sum_{i=1}^N \varphi(x^{(i)}), \quad x^{(i)} \sim \pi(x),$$

taken over independent realisations $x^{(i)}$ simulated from $\pi(x)$. This estimator is *strongly consistent*, i.e.

$$\hat{\varphi}_{\text{MC}} \xrightarrow{a.s.} \hat{\varphi}, \quad N \rightarrow \infty.$$

Also, we can construct a CLT for the error of the MC estimator, given by

$$\sqrt{N}(\hat{\varphi} - \hat{\varphi}_{\text{MC}}) \xrightarrow{d} \mathcal{N}(0, \sigma_{\text{MC}}^2), \quad \sigma_{\text{MC}}^2 = \mathbb{V}[\varphi(x)] < \infty,$$

where we assume that the function $\varphi(x)$ has a finite second moment. Here, we see that the MC estimator is asymptotically unbiased with Gaussian errors. Also, the variance of the error decreases as $1/N$ independent of the dimension of the problem, which is one of the main advantages of the MC methods.

Consider, the problem of estimating (3.3) when the normalised target $\pi(x) = \gamma(x)Z^{-1}$ is unknown together with the normalisation factor Z . Instead, we can only evaluate the unnormalised target $\gamma(x)$ point-wise. In this case, *importance sampling* (IS) (Marshall, 1956) can be used to sample from another distribution called the *proposal distribution* (or importance distribution) $q(x)$ and adapt the particles using a weighting scheme. For this, it is required that $q(x) \neq 0$ for almost all $x \in \text{supp}(\pi)$ and that the support of $\varphi(x)q(x)$ contains the support of $\pi(x)$, i.e. $\text{supp}(\pi) \subset \text{supp}(\varphi q)$. The IS algorithm follows by rewriting the expected value in

¹The SLLN states that the sample average $\bar{x}$ computed using N IID samples from $\pi(x)$, converges almost surely to $\mu = \mathbb{E}_{\pi}[x]$ when N tends to ∞ , i.e. $\bar{x} \xrightarrow{a.s.} \mu$.

Algorithm 1 Importance sampling (IS)

INPUTS: $\gamma(x)$ (unnormalised target), $q(x)$ (proposal) and $N > 0$ (no. particles).
 OUTPUT: $\hat{p}(dx)$ (empirical distribution).

- 1: **for** $i = 1$ to N **do**
 - 2: Sample a particle by $x^{(i)} \sim q(x)$.
 - 3: Compute the weight by $w^{(i)} = \gamma(x^{(i)})/q(x^{(i)})$.
 - 4: **end for**
 - 5: Normalise the particle weights by (3.5).
 - 6: Estimate $\hat{p}(dx)$ using (3.7).
-

(3.3) to obtain

$$\mathbb{E}_\pi[\varphi(x)] = \int \varphi(x) \pi(x) dx = \int \varphi(x) \underbrace{\frac{\gamma(x)}{q(x)}}_{\triangleq w(x)} q(x) dx = \mathbb{E}_q[w(x)\varphi(x)],$$

where $w(x)$ denotes the (unnormalised) *importance weights*. The IS estimator follows in analogue with the vanilla MC estimator,

$$\hat{\varphi}_{\text{IS}} = \sum_{i=1}^N \tilde{w}^{(i)} \varphi(x^{(i)}), \quad x^{(i)} \sim q(x), \quad (3.4)$$

where the *normalised weights* are given by

$$\tilde{w}(x^{(i)}) \triangleq \tilde{w}^{(i)} = \frac{w^{(i)}}{\sum_{k=1}^N w^{(k)}}, \quad i = 1, \dots, N. \quad (3.5)$$

We can also construct an unbiased estimator for the normalisation constant by

$$\hat{Z}_{\text{IS}} = \frac{1}{N} \sum_{i=1}^N w^{(i)}, \quad (3.6)$$

which we later shall make use of to estimate the (marginal) likelihood in SSMs. The IS algorithm can be interpreted as generating an *empirical distribution*, which can be seen by rewriting the estimator in (3.4) as a Dirac mixture given by

$$\hat{p}(dx) = \sum_{i=1}^N \tilde{w}^{(i)} \delta_{x^{(i)}}(dx), \quad (3.7)$$

which we can insert into (3.3) to recover (3.4) as presented in Algorithm 1. Here, $\delta_z(dx)$ denotes a Dirac point mass located at $x = z$. In the following, we make use of this property to compute expectations in state inference problems using sequential variants of the IS algorithm.

We conclude the discussion of the IS algorithm by Example 3.1 in which we estimate the parameters of the HWSV model using some real-world data. Note that, the IS algorithm can be developed further to include multiple and adaptive

proposals. In *multiple importance sampling* (MIS), the algorithm can make use of several proposal algorithms that are then combined to form the estimate. In *adaptive importance sampling* (AIS), we update a mixture of proposals or the proposal after each iteration to fit the target better. These methods could be interesting in developing new approximate state inference algorithms similar to the particle methods discussed in the next section. For more information, see Kronander and Schön (2014), Cornuet et al. (2011) and Veach and Guibas (1995).

3.1 Example: IS for Bayesian parameter inference in the HWSV model

Consider the Bayesian parameter inference problem in the Hull-White SV model (2.4) and the data presented in Section 2.2.2. In this model, we would like to estimate the parameters given the data by the use of the IS algorithm. The unnormalised target distribution is given by

$$\gamma(\theta) = p_\theta(y_{1:T})p(\theta),$$

where we assume that we can evaluate the likelihood point-wise. Note that, the solution to this problem is analytically intractable but can be approximated using methods discussed later in Section 3.3.4. Furthermore, we assume the uniform priors over $\phi \in (-1, 1) \subset \mathbb{R}$ and $\{\sigma_v, \beta\} \in \mathbb{R}_+^2$. The parameter proposals are given by

$$\begin{aligned} q(\phi) &= \mathcal{U}(\phi; -1, 1), \\ q(\sigma_v) &= \mathcal{G}(\sigma_v; 2, 0.1), \\ q(\beta) &= \mathcal{G}(\beta; 7, 0.1). \end{aligned}$$

These proposals are based on prior information regarding the typical values for parameters in the model. Hence, they can be seen as a kind of prior distributions in the Bayesian inference. For the IS algorithm, we use $N = 200$ samples and the procedure outlined in Algorithm 1.

In Figure 3.1, we present the resulting posterior distributions with the proposal distributions. The parameter estimates obtained by the posterior mode are $\theta_{\text{IS}} = \{0.996, 0.129, 0.837\}$, which indicates a slowly varying latent volatility process. In Chapter 4, we discuss other methods for parameter inference that are more efficient when the number of parameters increases. However, for small problems the IS algorithm could be a useful alternative in cases when the prior information is good. Otherwise, the number of samples required from the parameter posterior increases exponentially with the number of elements in the parameter vector.

3.3 Particle filtering

The IS algorithm discussed in the previous section can be combined with the filtering recursions in Section 3.1 to sequentially estimate the joint smoothing distribution $\pi(x_{0:t}) = p_\theta(x_{0:t}|y_{1:t})$. To this end, we can apply Algorithm 1 to sequentially approximate the target using an empirical distribution. The resulting algorithm is known as the *sequential importance sampling* (SIS) algorithm. The

Figure 3.1: The estimates posterior distributions for ϕ (green), σ_v (red) and β (orange) compute as KDEs with the proposal distributions presented in blue. The grey dots indicates the samples obtained from the proposal and the dotted lines indicate the posterior means.

main problem with this algorithm is that the variance of the estimate increases rapidly as t increases. For a long time, this was a major obstacle in approximate state inference. This results from that the particle weights deteriorate over time and in the limit only a single particle has a non-zero weight after a few iterations. Hence, the effective number of particles is one, which is the reason for the high variance in the estimates.

However, by including a resampling step into the SIS algorithm, we can focus the attention of the algorithm on the areas of interest. The resampling step essentially duplicates particles with large weights and discard particles with small weights, while keeping the total number of particles fixed. Hence, we do not end up with only one effective particle in the estimator. This development of the algorithm leads to the SIS with resampling (SIR) algorithm. When this method is applied on SSMs, we obtain the basic particle filtering algorithm introduced by Gordon et al. (1993). In this thesis, we refer to this algorithm as the *bootstrap particle filter* (bPF).

In subsequent developments, the particle filter is generalised to include more advanced proposals and resampling schemes. In this section, we present a refined version of the bPF called the *auxiliary particle filter* (APF) (Pitt and Shephard, 1999), which can use more general particle proposals and weighting functions than in the original formulation. This can result in a large decrease in the variance of the estimate and also a decrease in the number of particles required to achieve a certain accuracy. The APF also allows for the use of different resampling schemes than the multinomial resampling that is used in the original formulation of the bPF.

To keep the presentation brief, we do not derive the APF and refer interested readers to Doucet and Johansen (2011) for a derivation of the APF starting from the IS algorithm. Furthermore, we note that the APF is a member of the more general family of SMC algorithms, which are discussed in Cappé et al. (2007) and Del Moral et al. (2006).

3.3.1 The auxiliary particle filter

The APF algorithm operates by constructing a particle system $\{w_t^{(i)}, x_t^{(i)}\}_{i=1}^N$ sequentially over time. By the use of this particle system, we can construct an *empirical marginal filtering distribution* in analogue with (3.7) as

$$\hat{p}_\theta(dx_t|y_{1:t}) \triangleq \sum_{i=1}^N \tilde{w}_t^{(i)} \delta_{x_t^{(i)}}(dx_t), \quad (3.8a)$$

$$\tilde{w}_t^{(i)} \triangleq \frac{w_t^{(i)}}{\sum_{k=1}^N w_t^{(k)}}, \quad (3.8b)$$

which can be seen as a discrete approximation of the filtering distribution using a collection of weighted Dirac point masses. Here $x_t^{(i)}$ and $w_t^{(i)}$ denote particle i at time t and its corresponding (unnormalised) importance weight. Also, the

empirical joint smoothing distribution follows similarly as

$$\hat{p}_\theta(dx_{0:t}|y_{1:t}) \triangleq \sum_{i=1}^N \tilde{w}_t^{(i)} \delta_{x_{0:t}^{(i)}}(dx_{0:t}). \quad (3.9)$$

Three steps are carried out during each iteration of the APF to update the particle system from time $t - 1$ to t :

- (i) The particles are *resampled* according to their auxiliary weights. The means that particles with small weights are discarded and particles with large weights are multiplied. This mitigates the weight depletion problem experienced by the SIS algorithm. This is often the computational bottleneck of the APF algorithm as all the other steps can easily be implemented in parallel.
- (ii) The particles are *propagated* from time $t - 1$ to t . This can be seen as a simulation step, where new particles are generated by sampling from a Markov kernel.
- (iii) The (unnormalised) particle weight is calculated for each particle. These weights compare the particle with the recorded output from the system. A high weight is given to a particle that is likely to generate y_t .

After these three steps, we can construct empirical distributions using (3.8) and (3.9). It is also possible to make use of the particle system for estimating other quantities. We return to this in the following. Note that, by comparison with the filtering recursions, Step (ii) corresponds to the time update in (3.1b) and Step (iii) corresponds to the measurement update in (3.1a). We now proceed to discuss each step of the APF in more detail.

Step (i) can be seen as a sampling of *ancestor indices* denoted $a_t^{(i)}$, i.e. the index of the particle at time $t - 1$ from which the particle i at time t originates from. This can be expressed as simulating from a multinomial distribution with probabilities

$$\mathbb{P}(a_t^{(i)} = j) = \tilde{\nu}_t^{(j)}, \quad j = 1, \dots, N, \quad i = 1, \dots, N, \quad (3.10)$$

where $\tilde{\nu}_t^{(j)}$ denotes a *normalised auxiliary weight* given by

$$\tilde{\nu}_{t-1}^{(j)} = \nu_{t-1}^{(j)} \left[\sum_{k=1}^N \nu_{t-1}^{(k)} \right]^{-1},$$

for some auxiliary weight function $\nu_{t-1}^{(k)}$ determined by the user. After the resampling step, we obtain the unweighted particle system $\{\tilde{x}_{t-1}^{(i)}, 1/N\}_{i=1}^N$.

In this thesis, we mainly consider two different types of APFs. The bPF is obtained by selecting the auxiliary weight as the particle weights, i.e. $\nu_t = w_t$. The *fully-adapted particle filter* (faPF) (Pitt and Shephard, 1999), which takes into account the (future) observation in the resampling step. This information is included by

using the auxiliary weight given by

$$\nu_{t-1} = p_\theta(y_t|x_{t-1}) = \int g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1}) dx_t, \quad (3.11)$$

when it can be computed in closed-form. This is only possible for some systems, e.g. the LGSS model (2.2) and the GARCH(1,1) model (2.3).

In (3.10), we make use of multinomial resampling but there are other resampling algorithms that can be useful. *Systematic resampling* and *stratified resampling* are good alternatives to the multinomial resampling. These resampling methods often decrease the variance in the estimates of the filtering distribution. See Douc and Cappé (2005) and Hol et al. (2006) for comparisons of different resampling strategies.

In Step (ii), each particle is propagated using a propagation kernel as,

$$x_t^{(i)} \sim R_\theta\left(x_t|x_{0:t-1}^{a_t^{(i)}}, y_t\right), \quad i = 1, \dots, N. \quad (3.12)$$

The bPF is recovered by selecting the state dynamics as the propagation kernel,

$$R_\theta(x_t|x_{0:t-1}, y_t) = f_\theta(x_t|x_{t-1}).$$

The faPF makes use of the current observation when proposing new particles. This results in the propagation kernel

$$R_\theta(x_t|x_{0:t-1}, y_t) = p_\theta(x_t|y_t, x_{t-1}) = \frac{p_\theta(y_t|x_t)p(x_t|x_{t-1})}{p(y_t|x_{t-1})}, \quad (3.13)$$

when it can be computed in closed-form. Each new particle is appended to the particle trajectory by $x_{0:t}^{(i)} = \{x_{0:t-1}^{a_t^{(i)}}, x_t^{(i)}\}$, for $i = 1, \dots, N$.

Finally, in Step (iii) each particle is assigned an importance weight by a weighting function $W_\theta(x_t, x_{t-1})$, similar to the IS algorithm in (3.4). The resulting weight is given by

$$w_t^{(i)} = W_\theta\left(x_t^{(i)}, x_{0:t-1}^{a_t^{(i)}}\right), \quad i = 1, \dots, N \quad (3.14a)$$

$$W_\theta(x_t, x_{0:t-1}) \triangleq \frac{w_{t-1}}{\nu_{t-1}} \frac{g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1})}{R_\theta(x_t|x_{0:t-1}, y_t)}. \quad (3.14b)$$

For the bPF, the choice of resampling step and propagation kernel leads to that the weights are given by

$$\nu_t = w_t = g_\theta(y_t|x_t),$$

i.e. the probability of observing y_t given x_t . For the faPF, we have that the particle weights are given by $w_t \equiv 1$.

The general form of the APF is presented in Algorithm 2 for approximate state inference in nonlinear SSMs. The complexity of the algorithm is linear in the number of time steps and particles, i.e. $\mathcal{O}(NT)$. The user choices are mainly the number of particles N , the auxiliary weights ν , the propagation kernel $R_\theta(x_t|x_{0:t-1}, y_t)$ and

Algorithm 2 Auxiliary particle filter (APF)

INPUTS: $y_{1:T}$ (observations), $R_\theta(x_t|x_{0:t-1}, y_t)$ (propagation kernel), ν_t (auxiliary weights) and $N > 0$ (no. particles).

OUTPUTS: $\hat{p}_\theta(x_t|y_{1:t})$ for $t = 1$ to T (the empirical marginal filtering distributions) and $\hat{p}_\theta(x_{0:t}|y_{1:t})$ for $t = 1$ to T (the empirical joint smoothing distributions).

- 1: Initialise each particle $x_0^{(i)}$.
 - 2: **for** $t = 1$ to T **do**
 - 3: Sample new the ancestor indices to obtain $\{a_t^{(i)}\}_{i=1}^N$ using (3.10).
 - 4: Propagate the particles to obtain $\{x_t^{(i)}\}_{i=1}^N$ using (3.12).
 - 5: Calculate new importance weights to obtain $\{w_t^{(i)}\}_{i=1}^N$ using (3.14).
 - 6: Estimate the marginal filtering distribution $\hat{p}_\theta(x_t|y_{1:t})$ using (3.8).
 - 7: Estimate the joint smoothing distribution $\hat{p}_\theta(x_{0:t}|y_{1:t})$ using (3.9).
 - 8: **end for**
-

the resampling method. We now proceed with discussing the use of the APF for state inference and log-likelihood estimation.

3.3.2 State inference using the auxiliary particle filter

Algorithm 2 can be used to estimate the marginal filtering and joint smoothing distributions in an SSM given some observations. From these empirical distributions, we can compute an estimate of the filtered marginal state $x_{t|t}$,

$$\hat{x}_{t|t} = \int x_t \hat{p}_\theta(x_t|y_{1:t}) dx_t = \sum_{i=1}^N \tilde{w}_t^{(i)} x_t^{(i)}, \quad (3.15)$$

where we have inserted (3.8). Note that, the state trajectory $x_{0:t|t}$ can be estimated using an analogue expression given by

$$\hat{x}_{0:t|t} = \int x_{0:t} \hat{p}_\theta(x_{0:t}|y_{1:t}) dx_{0:t} = \sum_{i=1}^N \tilde{w}_t^{(i)} x_{0:t}^{(i)}, \quad (3.16)$$

by the use of the empirical smoothing distribution (3.9). In Example 3.2, we illustrate the use of the bPF for marginal state inference in the earthquake model (2.6) and data discussed in Section 2.2.3,

3.2 Example: State inference in the earthquake count model

Consider the earthquake count model (2.6) using the data presented in Figure 2.4. To infer the state in the model, we make use of the bPF with $N = 1\,000$ particles and the parameter vector $\hat{\theta} = \{0.88, 0.15, 17.65\}$ estimated later in Example 4.9.

In the upper part of Figure 3.2, we present the filtered number of major earthquakes obtained from the bPF (red line) together with the data (black dots). We also present the predicted number of major earthquakes (dark green) for 2014 to 2022 together with a 95% CI for the predictions. The uncertainty grows fast for the

predictions (gray area), which indicates that a more advanced model is probably required to make good predictions.

In the lower part of Figure 3.2, we present the estimated state obtained from the bPF, which indicates that the world currently is in a calmer period with a smaller earthquake intensity compared with e.g. the 1950s. This conclusion is also supported in the actual number of earthquakes recorded during these two periods. We also present the predicted latent state (dark green) for 2014 to 2022 together with a 95% CI for the predictions (gray area). Again, we see that there is a large uncertainty in the future latent state.

From Algorithm 2, we know that the APF algorithm can be used to approximate the joint smoothing distribution. In the following, we require estimates of this distribution for the use in some of the proposed methods for approximate parameter inference. However, the accuracy of these estimates are poor due to problems with *path degeneracy*. To explain the nature of this problem, consider the resampling step in the APF algorithm. During each resampling, we discard some particle with a non-zero probability due to that they have small auxiliary weights. This means that the number of unique particles is smaller than N with a non-zero probability after each resampling step.

As t increases, we repetitively resample the particle system and this leads to that the number of unique particles tends to one before some time $s < t$. That is, the particle trajectory collapses into a single trajectory for all the particles before time s . Consequently, the variance of the estimates of the joint smoothing distribution is large due to the same problem as in the SIS algorithm. To mitigate this effect, we could make use of the faPF or increase the number of particles. However, this could be problematic as the faPF is not available for many interesting models. Also the computational cost of the APF algorithm increases linearly with N . We illustrate this effect and compare these two alternatives in Example 3.3.

There exists additional alternatives to mitigate the path degeneracy to obtain accurate approximation of the joint smoothing distribution. A popular alternative in practice is to resample only when the variance in the particle weights is larger than some threshold (Doucet and Johansen, 2011). This leads to a decrease in the path degeneracy problem but this is often not enough to completely solve the problem. Instead, a particle smoother is often used for this problem, which gives better accuracy in the estimates but often increases the computational cost. In Section 3.4, we return to the use of particle smoothing for estimating the smoothing distributions. We now continue with discussing the statistical properties of the APF and how to use the APF to estimate the likelihood and log-likelihood for an SSM.

Figure 3.2: Upper: the filtered number of major earthquakes (red line) and the data (black dots) obtained from the bPF using the data and the model from Example 3.2. The predicted number of earthquakes (green line) is also presented together with a 95% CI (gray area). Lower: the estimated latent earthquake intensity obtained from the bPF. The predicted latent state (green line) is also presented together with a 95% CI (gray area).

3.3 Example: Path degeneracy in the GARCH(1,1) model

Consider the GARCH(1,1) model in (2.3) from which we generate $T = 20$ observations using the parameter vector $\theta^* = \{0.10, 0.80, 0.05, 0.30\}$. Here, we make use of the bPF and faPF with systematic resampling at every iteration. The aim is to estimate the state trajectory $x_{0:t|t}$ using the APF and (3.16).

For this model, the weight function (3.11) and the proposal (3.13) required for the faPF can be computed in closed form by properties of the joint Gaussian distributions, see Bishop (2006). The required quantities are given by

$$\begin{aligned}
 p(y_t|x_{t-1}, h_t) &= \int \mathcal{N}(y_t; x_t, \tau^2) \mathcal{N}(x_t; 0, h_t) dx_t \\
 &= \mathcal{N}(y_t; 0, \tau^2 + h_t), \\
 p(x_t|y_t, x_{t-1}) &\propto p(y_t|x_t)p(x_t|x_{t-1}) \\
 &= \mathcal{N}(y_t; x_t, \tau^2) \mathcal{N}(x_t; 0, h_t) \\
 &= \mathcal{N}\left(x_t; [\tau^{-2} + h_t^{-1}]^{-1} \tau^{-2} y_t, \tau^{-2} + h_t^{-1}\right),
 \end{aligned}$$

from the marginalisation property and the conditioning property.

In Figure 3.3, we present the state trajectories obtained by tracing the ancestor lineage backwards from time T to 0. We see that for the bPF with $N = 10$ particles, the ancestral lines collapse to a single unique particle before time $t = 10$ due to the path degeneracy problem. Increasing the number of particles to $N = 20$ results in that the path degenerate before $t = 8$ instead. However, the faPF does not have the same problem with degeneracy as many ancestors survive the repeated resamplings and contributes with information for estimating the joint smoothing distribution.

3.3.3 Statistical properties of the auxiliary particle filter

In this section, we review some results regarding two aspects of the statistical properties of the APF. Note that, the analysis of the APF is rather complicated compared with other estimators that makes use of independent samples from the target distribution. Instead, the particles obtained from the APF are not independent due to the interaction during the resampling step. However, there are many strong results regarding the statistical properties of the bPF in the literature. Extensive technical accounts are found in Del Moral (2013) and Del Moral (2004). Some of the statistical properties are also discussed in a more application oriented setting in Crisan and Doucet (2002), Douc et al. (2014) and Doucet and Johansen (2011).

Assume that we would like to compute the expected value of some well-behaved test function $\varphi(x)$ using the particle system generated by the APF. This is in analogue with the IS estimator in (3.3). From the empirical filtering distribution

Figure 3.3: The ancestral paths obtained from the bPF with 10 particles (upper), the bPF with 20 particles (middle) and the faPF with 10 particles (lower) with $T = 20$ in the GARCH(1,1) model in Example 3.3. The discarded particles are presented as gray dots.

(3.8), we have that the expected value can be approximated by

$$\hat{\varphi}_{\text{APF}} = \sum_{i=1}^N \tilde{w}_t^{(i)} \varphi(x_t^{(i)}). \quad (3.17)$$

Under some assumptions, it is possible to prove that this estimator is consistent and therefore converges in probability to the true expectation as the number of particles tend to infinity,

$$\hat{\varphi}_{\text{APF}} \xrightarrow{p} \mathbb{E}[\varphi(x)], \quad N \rightarrow \infty.$$

Hence, this estimator is asymptotically unbiased and equivalent with some unknown *optimal filter*. However, for finite N the estimator is often biased and we return to this problem in Section 3.4.3.

We would also like to say something about the MSE of the estimator in (3.17). For this we assume that the function $\varphi(x)$ is bounded² for all x and some additional assumptions that are discussed by Crisan and Doucet (2002). It then follows that the MSE of the estimator obtained by the bPF can be upper bounded as

$$\mathbb{E} \left[\left(\hat{\varphi}_{\text{APF}} - \mathbb{E}[\varphi(x)] \right)^2 \right] \leq C_T \frac{\|\varphi\|^2}{N}, \quad (3.18)$$

where $\|\cdot\|$ denotes the supremum norm. Here, C_T denotes a function that possibly depends on T but is independent of N .

There exists numerous other results regarding the MSE of the estimator in (3.17). For example, it is possible to relax the assumption that $\varphi(x)$ should be bounded and that we only use the bPF with multinomial resampling. The resulting upper bounds have a similar structure to (3.18) but with different functions C . For more information, see Del Moral (2013) and Douc et al. (2014).

From this upper bound on the MSE, we would like to give some general recommendations regarding how to select N given T . However, this is difficult as the accuracy of the estimates is connected with the mixing property of the SSM (see Example 3.5). However, in practice it is recommended to use at least $N \propto T$ particles in the APF but sometimes even more particles are required to obtain reasonable estimates. Hence, we recommend that the user estimates the MSE for each model using e.g. a pilot run with some Monte Carlo simulations or by comparing with solution obtain from a particle smoother.

3.3.4 Estimation of the likelihood and log-likelihood

As previously discussed, the likelihood $\mathcal{L}(\theta)$ and log-likelihood $\ell(\theta)$ play important roles in both ML and Bayesian parameter inference. We also discussed that they are analytically intractable for nonlinear SSMs. However, we can obtain an unbiased estimate of the likelihood for any number of particles using the weights

²This is a rather restrictive assumption as it does not satisfied by the function $\varphi(x) = x$, which is used to compute the estimate of the filtered state $\hat{x}_{t|t}$.

generated by the APF. To understand how this can be done, we consider the decomposition in Definition 2.1 and the fact that each predictive likelihood can be written as

$$\begin{aligned} p(y_t|y_{1:t-1}) &= \int p_\theta(y_t, x_t|y_{1:t-1}) dx_t \\ &= \int g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1}) p_\theta(x_{t-1}|y_{1:t-1}) dx_{t-1:t}. \end{aligned}$$

If we consider the bPF algorithm, we can rewrite this as

$$\begin{aligned} p_{\text{bPF}}(y_t|y_{1:t-1}) &= \int \frac{g_\theta(y_t|x_t) f_\theta(x_t|x_{t-1})}{R_\theta(x_t|x_{t-1}, y_t)} R_\theta(x_t|x_{t-1}, y_t) p_\theta(x_{t-1}|y_{1:t-1}) dx_{t-1:t} \\ &= \int W_\theta(x_t, x_{t-1}) R_\theta(x_t|x_{t-1}, y_t) p_\theta(x_{t-1}|y_{1:t-1}) dx_{t-1:t} \end{aligned}$$

by the expression for the weight function (3.14) as $w_{t-1} = \nu_{t-1}$ for the bPF.

To approximate the predictive likelihood, we make use of that $\{\tilde{x}_t^{(i)}, x_{t-1}^{(i)}\}_{i=1}^N$ is approximately distributed according to $R_\theta(x_t|x_{t-1}, y_t) p_\theta(x_{t-1}|y_{1:t-1})$. From this, it follows that

$$\begin{aligned} p_{\text{bPF}}(y_t|y_{1:t-1}) &\approx \frac{1}{N} \sum_{i=1}^N \int W_\theta(x_t, x_{t-1}) \delta_{\tilde{x}_t^{(i)}, x_{t-1}^{(i)}}(dx_{t-1:t}) \\ &= \frac{1}{N} \sum_{i=1}^N W_\theta(\tilde{x}_t^{(i)}, x_{t-1}^{(i)}) \\ &= \frac{1}{N} \sum_{i=1}^N w_t^{(i)}. \end{aligned}$$

It is also possible to show that the faPF leads to a similar estimator by replacing w_t with ν_t , see Pitt (2002) for the derivation. The resulting estimator for the likelihood using the APF (including both the bPF and faPF as special cases) is given by

$$\hat{\mathcal{L}}(\theta) = \hat{p}_\theta(y_{1:T}) = \frac{1}{N^{T+1}} \left\{ \sum_{i=1}^N w_T^{(i)} \right\} \left\{ \prod_{t=0}^{T-1} \sum_{i=1}^N \nu_t^{(i)} \right\}, \quad (3.19)$$

where the first summation is unity for the faPF and where $\nu_t = w_t$ for the bPF. To implement this estimator, we run Algorithm 2 and then calculate the likelihood estimate using (3.19) by inserting the particle weights generated by the APF.

The statistical properties of the likelihood estimator are studied by Del Moral (2004). It turns out that the estimator is consistent and unbiased for any $N \geq 1$. Furthermore, the error of the estimate satisfies a CLT,

$$\sqrt{N} [\mathcal{L}(\theta) - \hat{\mathcal{L}}(\theta)] \xrightarrow{d} \mathcal{N}(0, \psi^2(\theta)), \quad (3.20)$$

for some asymptotic variance $\psi^2(\theta)$, see Proposition 9.4.1 in Del Moral (2004).

It is also straightforward to obtain an estimator for the log-likelihood from (3.19),

$$\widehat{\ell}(\theta) = \log \widehat{p_\theta(y_{1:T})} = \log \left\{ \sum_{i=1}^N w_T^{(i)} \right\} + \sum_{t=0}^{T-1} \log \left\{ \sum_{i=1}^N \nu_t^{(i)} \right\} - T \log(N+1). \quad (3.21)$$

However, this estimator is biased for a finite number of particles, but it is still consistent and asymptotically normal. This result follows from applying the *second-order delta method* (Casella and Berger, 2001) on (3.20). The resulting CLT for the log-likelihood estimator is given by

$$\sqrt{N} \left[\ell(\theta) - \widehat{\ell}(\theta) + \frac{\gamma^2(\theta)}{2N} \right] \xrightarrow{d} \mathcal{N}(0, \gamma^2(\theta)), \quad (3.22)$$

where we introduce $\gamma(\theta) = \psi(\theta)/\mathcal{L}(\theta)$. As a result, we have an expression for the bias of the estimator given by $-\gamma^2(\theta)/2N$ for a finite number of particles. Consequently, it is possible to compensate for the bias as the variance of the estimator $\gamma^2(\theta)$ can be estimated using Monte Carlo simulations by repeated application of the APF on the same data. This could be an interesting improvement for the proposed methods in Papers B and C, where we make use of the log-likelihood estimator.

3.4 Example: Bias and variance of the log-likelihood estimate

Consider the setup in Example 2.6 and the problem of estimating the log-likelihood at $\theta = \theta^*$ using the faPF with $N = 10$ particles. We repeat the estimation over 1 000 Monte Carlo simulations using the same data. The error of the log-likelihood estimate is calculated using the true valued obtain from the Kalman filter.

In Figure 3.4, we present the histogram of the error together with a Gaussian approximation (upper), the box plot of the errors (lower left) and the QQ-plot of the errors (lower right). The Gaussian approximation fits the data quite well and the resulting average error is -0.03 with variance $\widehat{\gamma}^2(\theta^*) = 0.05$. The predicted bias is calculated using (3.22) by $-\widehat{\gamma}^2(\theta^*)/2\sqrt{N} = -0.01$. The QQ-plot validates the Gaussian assumption as we do not see any deviating tail behaviour.

3.4 Particle smoothing

Particle smoothers approximate the solution to the smoothing problem in an SSM similar to how the APF approximates the corresponding filtering problem. However, there exists a number of different approaches to carry out the smoothing given the particle system from the APF. The simplest smoother is to make use of the APF to approximate the joint smoothing distribution as discussed in the previous. The main problem with this approach is that the path degeneracy problem limits the accuracy of the estimate. Another similar approach is to make use of the fixed-lag (FL) particle smoother (Kitagawa and Sato, 2001), which is based on using the APF to estimate the fixed-lag smoothing distribution (recall Table 3.1). In the following, we make use of this smoother as it has a low computational cost and a reasonable accuracy compared with other particle smoothers.

Figure 3.4: The histogram with a Gaussian approximation (blue line) of the error in the log-likelihood estimates (upper) in the LGSS model using a faPF. The boxplot (lower left) and QQ-plot (lower right) supports the Gaussian assumption of the error.

More advanced smoothers are often based on approximations of the Bayesian smoothing recursion (3.2). There are two main families of smoothers that result from this approach: the FFBSm and the forward filtering backward simulator (FFBSi). The original marginal FFBSm and FFBSi algorithms are discussed in Doucet et al. (2000) and Godsill et al. (2004), respectively. Another type of smoother makes use of two APFs (one running forward in time and the other backwards) and combines the output of these by using a *two-filter formula* (Briers et al., 2010; Fearnhead et al., 2010). Furthermore, other types of smoothers have been proposed in Bunch and Godsill (2013) and Dubarry and Douc (2011) using MCMC methods (discussed in the next chapter). The interesting improvement in these two smoothers are that they generate new particles in the backward sweep, which is not done in the FFBSm and FFBSi. See Lindsten and Schön (2013) for a recent survey of different particle smoothing methods.

In this section, we focus on the use of the FL smoother for estimating some quantities that are required for the proposed methods in Papers A and D. We introduce the underlying assumptions of the smoother and discuss how to implement it. We also show how it can be used to estimate the score function of an SSM and parts of the information matrix. We conclude by discussing the properties of the estimates obtained from the FL smoother.

3.4.1 State inference using the particle fixed-lag smoother

The FL smoother relies on the forgetting properties of an SSM, i.e. that the Markov chain quickly forgets about its earlier states. This property is illustrated in Example 3.5 for an LGSS model.

3.5 Example: Mixing property in the LGSS model

Consider the LGSS model using the same setup as in Example 2.6, where 20 different state processes are simulated during 8 time steps. Here, each state process has a randomly selected initial states distributed as $x_0 \sim \mathcal{N}(0, 20^2)$.

In Figure 3.5, we present the evolution of the state processes in three different LGSS models. We note that the value of ϕ determines the rate at which the processes converge to a stationary phase. A larger value of ϕ gives the process a longer memory and this results in that it requires longer time to forget its initial condition. That is, the state process mixes slowly and therefore future observations contain useful information about the current state.

The observation that some SSMs mixes quickly is the basis for the assumption that

$$p_\theta(x_{0:t}|y_{1:T}) \approx p_\theta(x_{0:t}|y_{1:\kappa_t}), \quad (3.23)$$

where $\kappa_t = \min\{T, t + \Delta\}$ and Δ denotes some lag determined by the user. This means that future observations contain a decreasing amount of information about the current state. The rate of this decrease is determined by the mixing of the model as discussed in Example 3.5. If the model mixes quickly, future observations have a limited amount of information about the current state as the state process

Figure 3.5: The evolution of 20 different state processes in the LGSS model from Example 2.6 using different initial values. Three different values of ϕ are used in the LGSS model: $\phi = 0.2$ (upper), $\phi = 0.5$ (middle) and $\phi = 0.8$ (lower).

Algorithm 3 Two-step fixed-lag (FL) particle smoother

INPUTS: $y_{1:T}$ (observations), $R_\theta(x_t|x_{0:t-1}, y_t)$ (propagation kernel), ν_t (auxiliary weights), $N > 0$ (no. particles) and Δ (lag).

OUTPUTS: $\hat{p}_\theta(x_{t-1:t}|y_{1:T})$ for $t = 1$ to T (empirical two-step smoothing dist.).

```

1: Initialise each particle  $x_0^{(i)}$ .
2: for  $t = 1$  to  $T$  do
3:   Sample new the ancestor indices to obtain  $\{a_t^{(i)}\}_{i=1}^N$  using (3.10).
4:   Propagate the particles to obtain  $\{x_t^{(i)}\}_{i=1}^N$  using (3.12).
5:   Calculate new importance weights to obtain  $\{w_t^{(i)}\}_{i=1}^N$  using (3.14).
6:   if  $t > \Delta + 1$  then
7:     Compute  $\kappa_t = \min\{t + \Delta, T\}$  and recover  $\{x_{\kappa_t, t-1:t}^{(i)}\}_{i=1}^N$ .
8:     Compute (3.24) to obtain  $\hat{p}_\theta(x_{t-1:t}|y_{1:T})$ .
9:   end if
10: end for

```

quickly forgets its past. Hence, the lag Δ can be selected to be rather small and still make (3.23) a valid approximation. This is the main intuition behind the FL smoother and the procedure follows directly from the APF in Algorithm 2.

In the following, we require estimates of the two-step smoothing distribution $\hat{p}_\theta(dx_{t-1:t}|y_{1:\kappa_t})$ to estimate the score function and the information matrix for an SSM. By using the FL smoother assumption, we can compute the required estimate by a marginalisation over the joint smoothing distribution,

$$p_\theta(x_{t-1:t}|y_{1:\kappa_t}) = \int p_\theta(x_{0:\kappa_t}|y_{1:\kappa_t}) dx_{0:t-2} dx_{t+1:\kappa_t}.$$

By inserting the empirical joint smoothing distribution $p_\theta(x_{1:\kappa_t}|y_{1:\kappa_t})$ from (3.9), an estimate of the two-step smoothing distribution is obtained as

$$\hat{p}_\theta(dx_{t-1:t}|y_{1:\kappa_t}) = \sum_{i=1}^N \tilde{w}_{\kappa_t}^{(i)} \delta_{x_{\kappa_t, t-1:t}^{(i)}}(dx_{t-1:t}), \quad (3.24)$$

where $x_{\kappa_t, t}^{(i)}$ denotes the ancestor particle from which the particle $x_{\kappa_t}^{(i)}$ originated from at time t . This ancestor is obtained by tracing the lineage backwards similar to Figure 3.3, where the same is done for the particles starting at time T .

We summarise the procedure for estimating the two-step smoothing distribution using the FL-smoother in Algorithm 3. The marginal smoothing distribution can be computed using an analogue expression and this results in a similar procedure. Finally, we present an application of these methods for volatility estimation in Example 3.6.

Figure 3.6: Upper: the log-returns (gray dots) from the Nasdaq OMX Stockholm 30 Index presented in Figure 2.2. The estimated 95% CIs are also presented for the bPF (red) and the FL smoother (blue). Middle: the estimated latent volatility obtained from the bPF. Lower: the estimated latent volatility obtained from the FL smoother.

3.6 Example: State inference in the Hull-White SV model

Consider the Hull-White SV model (2.4) with data from Section 2.2.2 and the parameter vector estimate from Example 3.1. To infer the latent volatility (the state), we apply the bPF using Algorithm 2 and the FL smoother according to Algorithm 3 using $N = 5000$ particles and $\Delta = 12$.

In the upper part of Figure 3.6, we present the filtered log-returns with a 95% CI for the bPF (red) and the FL smoother (blue). Here, we present the upper CI for the bPF and the lower CI for the FL smoother so that the two lines do not overlap. Note that the mean of the log-returns is zero and therefore the CIs are symmetric around zero. We conclude that the CIs seem reasonable as they cover most of the log-returns except at the financial crises. Also, the CI computed from the bPF is a bit rougher than the CI computed using the FL smoother.

In the middle and lower parts of Figure 3.6, we present the estimated volatilities using both methods. We see that the estimates correspond reasonable well with variation of the log-returns. Note that, the periods with larger volatility are connected with the financial crises.

3.4.2 Estimation of additive state functionals

In this section, we consider the use of particle smoothing to estimate the expected value of an additive functional given the observations. This is in analogue with the Monte Carlo estimate of the expectation of a function in (3.3). An *additive functional* satisfies the expression

$$S_\theta(x_{0:T}) = \sum_{t=1}^T s_{\theta,t}(x_{t-1:t}), \quad (3.25)$$

which means that we can decompose a function that depends on the entire particle trajectory into several functionals. Here, $s_{\theta,t}(x_{t-1:t})$ denotes some general functional that depends on only two states of the trajectory. This type of additive functionals occurs frequently in nonlinear SSMs when we would like to compute functions that depend on the kernels $f_\theta(x_{t+1}|x_t)$ and $g_\theta(y_t|x_t)$. In the following, we give some concrete examples of these functionals connected with parameter inference in SSMs. In these problems, we would like to compute the expected value of the additive functional given the observations,

$$\begin{aligned} \mathbb{E}[S_\theta(x_{0:T})|y_{1:T}] &= \int S_\theta(x_{0:T}) p_\theta(x_{0:T}|y_{1:T}) dx_{0:T} \\ &= \sum_{t=1}^T \int s_{\theta,t}(x_{t-1:t}) p_\theta(x_{t-1:t}|y_{1:T}) dx_{t-1:t}. \end{aligned} \quad (3.26)$$

This can be done by inserting the two-step smoothing distribution estimated by any particle filter or smoothing algorithm. Examples of some different approaches for this are found in Poyiadjis et al. (2011) using the APF and in Del Moral et al. (2010) using a forward smoother based on the FFBSm. In this section and in

Paper A, we discuss the use of the FL smoother for this application. The resulting estimate is obtained by inserting (3.24) into (3.26),

$$\widehat{S}_\theta(x_{0:T}) = \sum_{t=1}^T \sum_{i=1}^N \widetilde{w}_{\kappa_t}^{(i)} s_{\theta,t}(x_{\kappa_t,t-1:t}^{(i)}), \quad (3.27)$$

which can be computed by some minor modifications of Algorithm 3.

We encounter this type of additive functionals are encountered in two common problems concerning SSMs: when estimating the score function and parts of the information matrix. In this thesis, we make use of the material in this section for estimating the latter two quantities in Papers A and D. To derive the expressions for the additive functions related to these problems, we consider the logarithm of the joint distribution of states and observations in an SSM,

$$\log p_\theta(x_{0:T}, y_{1:T}) = \log \mu(x_0) + \sum_{t=1}^T \left[\log f_\theta(x_t|x_{t-1}) + \log g_\theta(y_t|x_t) \right]. \quad (3.28)$$

By using this quantity, the score function can be estimated using *Fisher's identity* (Fisher, 1925; Cappé et al., 2005),

$$\mathcal{S}(\theta') = \nabla \ell(\theta)|_{\theta=\theta'} = \int \left[\nabla \log p_\theta(x_{0:T}, y_{1:T})|_{\theta=\theta'} \right] p_\theta(x_{0:T}|y_{1:T}) dx_{0:T},$$

which results in the functional

$$\xi_{\theta,t}(x_{t-1:t}) = \nabla \log f_\theta(x_t|x_{t-1})|_{\theta=\theta'} + \nabla \log g_\theta(y_t|x_t)|_{\theta=\theta'}, \quad (3.29)$$

corresponding to the gradient of (3.28) evaluated at $\theta = \theta'$. An estimator of the score function is obtained by inserting the additive function (3.29) into the empirical distribution obtained by the FL smoother (3.27) as

$$\widehat{\mathcal{S}}(\theta') = \sum_{t=1}^T \sum_{i=1}^N \widetilde{w}_{\kappa_t}^{(i)} \xi_{\theta,t}(x_{\kappa_t,t-1:t}^{(i)}).$$

The observed information matrix can be estimated using *Louis' identity* (Louis, 1982; Cappé et al., 2005). However, only some parts of the identity can be directly estimated using the FL smoother. The remaining parts must be estimated using the APF or a more advanced smoother like the FFBSm. To see this, we rewrite the observed information matrix using Louis' identity,

$$\mathcal{J}(\theta') = -\nabla^2 \ell(\theta)|_{\theta=\theta'} = \left[\nabla \ell(\theta)|_{\theta=\theta'} \right]^2 - \left[\nabla^2 \mathcal{L}(\theta)|_{\theta=\theta'} \right] \left[\mathcal{L}(\theta) \right]^{-1}.$$

Here, the second term of the identity can be written as

$$\begin{aligned} \left[\nabla^2 \mathcal{L}(\theta)|_{\theta=\theta'} \right] \left[\mathcal{L}(\theta) \right]^{-1} &= \int \left[\nabla \log p_\theta(x_{0:T}, y_{1:T})|_{\theta=\theta'} \right]^2 p_\theta(x_{0:T}|y_{1:T}) dx_{0:T} \\ &\quad + \int \left[\nabla^2 \log p_\theta(x_{0:T}, y_{1:T})|_{\theta=\theta'} \right] p_\theta(x_{0:T}|y_{1:T}) dx_{0:T}. \end{aligned} \quad (3.30)$$

The first term in (3.30) cannot be directly estimated by the FL smoother, as it does not provide approximations of the densities needed. Instead, it can be estimated using the APF directly as proposed by Poyiadjis et al. (2011) or by the use of a combination of the FL smoother and the APF. The latter alternative is discussed in Paper A. However, the second term in (3.30) can be estimated using the FL smoother directly as it can be written as an additive functional given by

$$\zeta_{\theta,k}(x_{t-1:t}) = \nabla^2 \log f_{\theta}(x_t|x_{t-1})|_{\theta=\theta'} + \nabla^2 \log g_{\theta}(y_t|x_t)|_{\theta=\theta'}. \quad (3.31)$$

corresponding to the Hessian of (3.28) evaluated at $\theta = \theta'$. An estimator for this part of Louis' identity is obtained by inserting the additive function (3.31) into the empirical distribution obtained by the FL smoother (3.27). The expected information matrix can be estimated as the sample covariance matrix of a large number of score functions estimated using different data sets. We discuss this method further in Paper D.

We have now seen two examples of additive functionals in connection with parameter inference in SSMs and how to estimate them using the FL smoother. The same setup can also be used in the expectation maximisation algorithm (Dempster et al., 1977; McLachlan and Krishnan, 2008) as discussed in Del Moral et al. (2010) using a forward smoother. We return to this problem in Section 4.1, where we discuss different methods for parameter inference. We end this section with Example 3.7, where we estimate the log-likelihood, score and observed information matrix for a nonlinear SSM using the FL smoother.

— 3.7 Example: Score and information matrix in the Hull-White SV model —

Consider the problem of estimating the log-likelihood, the score function and the natural gradient (the score function divided by the observed information matrix) for $\theta = \phi$ in the HWSV model (2.4). We again make use of the data from Section 2.2.2 and the parameter vector estimate from Example 3.1. For this model, the additive functional connected with the score function (3.29) is

$$\begin{aligned} \xi_{\theta,t}(x_{t-1:t}) &= \nabla \left[-\frac{1}{2} \log(2\pi\sigma_v^2) - \frac{1}{2\sigma_v^2} (x_{t+1} - \phi x_t)^2 \right] \Big|_{\phi=\phi'} \\ &\quad + \nabla \left[-\frac{1}{2} \log(2\pi\beta^2 \exp(x_t)) - \frac{y_t^2 \exp(-x_t)}{2\beta^2} \right] \Big|_{\phi=\phi'} \\ &= x_t(x_{t+1} - \phi' x_t), \end{aligned}$$

and similar for the observed information matrix (3.31),

$$\zeta_{\theta,t}(x_{t-1:t}) = \nabla \left[x_t(x_{t+1} - \phi x_t) \right] \Big|_{\phi=\phi'} = -x_t^2.$$

Here, we make use of the FL smoother in Algorithm 3 for estimating (3.27) with lag $\Delta = 12$ and $N = 5000$ particles. We vary the parameter on a grid within the interval $\phi \in [0.70, 0.99]$ and estimate the required quantities at each grid point. Here, we fix $\{\sigma_v, \beta\}$ to their estimated value. The observed information matrix is estimated using the combination of the APF and the FL smoother introduced in Paper A.

Figure 3.7: The estimates of the log-likelihood function (upper) the score function (lower left) and the natural gradient (lower right) of the HWSV model in Example 3.7. The dotted lines indicate the true parameters and the zero level.

The resulting estimates are presented in Figure 3.7. We see that the log-likelihood and score estimates seem reasonable and have a rather low variance. The natural gradient is more difficult to estimate due to that the observed information matrix estimates are noisy. We see that the sign of the estimate is correct but the gradient is noisy and rather small.

3.4.3 Statistical properties of the particle fixed-lag smoother

We conclude the discussion of particle smoothing by reviewing some results regarding the statistical properties of the estimates obtained by the FL smoother for the additive functionals. In Olsson et al. (2008), the authors analyse the bias and variance of the estimates from the APF and the FL smoother, respectively. The variances of the estimates are concluded (under some regularity conditions) to be upper bounded by quantities proportional to $T^2/\sqrt{N}$ and $(T \log T)/\sqrt{N}$ for the APF and the FL smoother, respectively.

The bias is also analysed in Olsson et al. (2008) and the authors conclude (under some regularity conditions) that the bias is upper bounded by quantities proportional to T^2/N and $\lambda + (T \log T)/N$, respectively. Here, λ denotes a quantity which is independent of the number of particles. Hence, the FL smoother gives a biased estimate for all N whereas the bias of the estimate obtained from the APF decreases as $1/N$.

In the analysis, it is assumed that the lag is selected according to $\Delta \propto c \log T$, where $c > -1/\log \rho$ and $\rho \in [0, 1]$ denotes a measure of the mixing of the model (Olsson et al., 2008). If Δ is too small, the underlying approximation of the FL smoother (3.23) and the accuracy of the estimate are rather poor. This is the result of that more information about the current state is available in future observations that are not taken into account by the smoother. If instead Δ is too large, we get problems with path degeneracy and in the limit when $\Delta = T$, the APF estimate is recovered. The choice of Δ is therefore crucial for obtaining good estimates from the FL smoother and its optimal value depends on both the number of the observations and the mixing properties of the model. Hence, it must be tailored for each application individually as is further discussed in Paper A.

Finally, we mention that even better accuracy can be obtained by using other more advanced smoothing algorithms. The main drawback with these algorithms is that their computational cost is larger than for the APF and the FL smoother, which have a computational cost that is proportional to $\mathcal{O}(NT)$. For example, the FFBSm algorithm (Doucet et al., 2000) has a computational cost that is proportional to $\mathcal{O}(N^2T)$. The benefit of using this smoother is that the variance of the estimate grows proportional to $\mathcal{O}(T)$, instead of as $\mathcal{O}(T^2)$ and $\mathcal{O}(T \log T)$ for the APF and the FL smoother, respectively. These properties are discussed in more detail in Del Moral et al. (2010) and Poyiadjis et al. (2011). There are also some new promising particle smoothers with a computational complexity proportional to $\mathcal{O}(TN \log N)$. For more information, see Klaas et al. (2006), Taghavi et al. (2013) and Gray (2003).

3.5 SMC for Image Based Lighting

In Sections 3.3 and 3.4, we have discussed the application of SMC to the filtering and smoothing problems in SSMs. However, the SMC algorithm can be applied to other problems that are sequential in nature and where the target distribution $\pi(x_{0:k})$ grows over some index $k = 1, \dots, K$. It is also possible to make use of SMC methods for static problems by defining an artificial target that grows over k even if the original target does not. This approach is discussed by Del Moral et al. (2006) and opens up for using SMC for a wide range of problems.

Figure 3.8: The setup used in the IBL approach, where the LTE gives the outgoing radiance at the angle ω_r that hits the image plane. This radiance is computed by taking the EM, the BRDF and the visibility into account. Note, that one of the light sources are occluded in this scene and does not contribute to the outgoing radiance.

We exemplify the usefulness of this general class of SMC algorithm by returning to the problem of rendering a sequence of photorealistic images using IBL (Debevec, 1998; Pharr and Humphreys, 2010) as discussed in Section 1.1.2. In Figure 3.8, we present a cartoon of the setup of the LTE. Here, we are interested in calculating the outgoing radiance $L_{r,k}(\omega_r)$ from a point at the outgoing angle ω_r at frame k , which is given by the LTE as

$$L_{r,k}(\omega_r) = \int f_r(\omega_i \rightarrow \omega_r) L_k(\omega_i) V(\omega_i)(\omega_i \cdot \mathbf{n}) d\omega_i, \quad (3.32)$$

where $\mathbf{n}$ denotes the surface normal and ω_i denotes incoming angles of the light rays that hits the object at n . Here, f_r , L_k and V denote the *bidirectional reflectance distribution function* (BRDF), the EM in frame k and the binary visibility function, respectively. Furthermore, we assume that these functions can be evaluated point-wise. In this problem, the BRDF describes the optical properties of the object and can be seen as a distribution over incoming angles.

To see how SMC can be used to solve (3.32), we define this as a sequence of normalisation problems, where k th unnormalised target distribution is given by

$$\gamma_k(\omega_i) = f_r(\omega_i \rightarrow \omega_r) L_k(\omega_i)(\omega_i \cdot \mathbf{n}),$$

where we neglect the visibility term for computational convenience. Calculating the outgoing radiance (without regarding visibility) $L_{r,k}(\omega_r)$ can therefore be seen as normalisation,

$$L_{r,k}(\omega_r) = \int \gamma_k(\omega_i) d\omega_i,$$

for a sequence of EMs indexed by k . From this setup, we see that the desired particles corresponds to incoming angles ω_i of the light that hits the object and contributes to the outgoing radiance. As previously discussed, the challenge is to select only a few important angles to take into account when solving the LTE. Otherwise, the problem becomes computational infeasible.

In the filtering problem, this normalisation factor is the marginal likelihood of the problem, which can be computed using the particle weights. In a similar manner, we can compute $L_r(\omega_r)$ and the only difference between the APF in Algorithm 2 is the choice of propagation kernel. Here, we do not have an SSM to describe the dynamical behaviour of the EM between frames. Instead, we make use of an MCMC kernel to adapt the particles between EMs. Details of this approach are found in Kronander et al. (2014a) and Ghosh et al. (2006).

Another approach to solve (3.32) is to use the IS algorithm directly on $L_{r,k}(\omega_r)$ for each frame individually. This can be done with an algorithm similar to SIR, where we first sample from the EM and compute the particle weights using the BRDF. The resulting particles are obtained after a resampling step. This method is referred to as the *bidirectional importance sampling* (BIS) (Burke et al., 2005) algorithm. This approach is computationally cheap, but cannot make use of information from the previous frame when estimating $L_{r,k}(\omega_r)$ in the current frame k . As previously discussed, this problem can be solved with the SMC setup that we consider here. We conclude this section by Example 3.8 in which we compare these two approaches.

3.8 Example: SMC for direct illumination

Consider a simple setup, where we would like to render a sphere given a sequence of HDR images. This is similar to the problem considered in Section 1.1.2 but simpler since we do not consider multiple bounces. Hence, this problem is referred to as *direct illumination*. Here, we compare four different approaches to render the sphere in the image. The first and second methods are to make use of the IS algorithm to sample directly from the EM and the BRDF, respectively. Hence, we disregard the information in the BRDF and the EM, respectively. We also make use of the BIS algorithm and the SMC algorithm previously outlined.

In Figure 3.9, we compare the four different approaches at frame $k = 10$ in a sequence of EMs. The first three methods solves the problem for each frame individually and the SMC approach makes use of the previous information. We

see that using the IS algorithm directly to sample from the EM does not perform well. The reason is that we consider a mirror sphere, where the BRDF is rather narrow, which means that only light from a few incoming angles contribute to the outgoing radiance. The remaining three methods gives comparable results. Therefore, we conclude that there could be advantages by using the SMC algorithm for this application. See Kronander et al. (2014a) for a more detailed discussion and comparison of these approaches.

Figure 3.9: Mirror sphere, frame 10 with different sampling techniques: IS the EM (upper left), IS the BRDF (upper right), BIS (lower left), and the SMC algorithm (lower right).

4

Parameter inference using sampling methods

In this chapter, we consider some sampling based methods for ML and Bayesian parameter inference in SSMs. We start by giving a small overview of some of the current algorithms in the field. Later, we introduce and discuss the use of MCMC (Robert and Casella, 2004) and particle MCMC (PMCMC) (Andrieu et al., 2010) approaches to Bayesian parameter inference. Here, we also discuss the use of Langevin and Hamiltonian dynamics to improve the efficiency of the method. This material is later used in Paper A to construct a particle version of these MCMC algorithms.

The second part of this chapter is devoted to discussing the BO algorithm (Jones, 2001; Boyle, 2007; Lizotte, 2008; Osborne, 2010) and how it can be used together with GPs (Rasmussen and Williams, 2006) for ML parameter inference and input design. This approach is used in Papers B and C for ML based parameter inference and MAP based parameter inference, respectively.

4.1 Overview of computational methods for parameter inference

There are many different methods developed for parameter inference in SSMs using both ML and Bayesian methods. Here, we review some of these methods to give an overview of the area and discuss some of the more popular methods. For more exhaustive accounts of different methods, see Douc et al. (2014), Kantas et al. (2009), Cappé et al. (2007) and Cappé et al. (2005).

4.1.1 Maximum likelihood parameter inference

Most parameter inference problems in the ML framework cannot be solved by analytical calculations. Instead popular approaches are based on optimisation and other iterative algorithms to solve the inference problem in Definition 2.2. From Chapter 3, we know that the APF can be used to calculate the log-likelihood and that the FL smoother can be used to estimate the score function (the gradient of the log-likelihood). Hence, we can make use of a *gradient-based optimisation algorithm* to maximise the log-likelihood and thereby estimate the parameters of the model. This method operates by an iterative application of

$$\hat{\theta}_{k+1} = \hat{\theta}_k + \epsilon_k \hat{\mathcal{S}}(\theta_k), \quad (4.1)$$

where $\hat{\theta}_k$ and ϵ_k denote the current estimate of the parameter vector and the step length, respectively. This approach is used by Poyiadjis et al. (2011) and Yildirim et al. (2013) for parameter inference in SV models with Gaussian returns and α -distributed returns, respectively. Furthermore, we can estimate the Hessian information by a particle smoother and make use of it in (4.1). This results in a *Newton optimisation algorithm* which operates by an iterative application of

$$\hat{\theta}_{k+1} = \hat{\theta}_k + \epsilon_k \hat{\mathcal{J}}^{-1}(\theta_k) \hat{\mathcal{S}}(\theta_k). \quad (4.2)$$

Another approach is to estimate the Hessian on-the-fly in a Quasi-Newton algorithm using finite difference approximation. A popular member of this class of algorithms is the *simultaneous perturbation stochastic approximation* (SPSA) algorithm discussed in Spall (1987). The SPSA algorithm is used for inference in the α SV model in Ehrlich et al. (2012).

The *expectation maximisation* algorithm (Dempster et al., 1977; McLachlan and Krishnan, 2008) is another popular alternative for ML inference. In this method, the latent states are seen as missing information and are estimated together with the parameter vector of the model. The quantity required by the expectation maximisation algorithm is an additive functional that can be estimated using a particle smoother. Many different versions of the expectation maximisation algorithm are available in the literature using different particle smoothers. Some examples are Del Moral et al. (2010) using a forward smoother, Olsson et al. (2008) using the FL smoother, Lindsten (2013) using the conditional particle filter with ancestor sampling and Schön et al. (2011) using the FFBSm.

Finally, we mention another approach based on *Bayesian optimisation* (Jones, 2001; Boyle, 2007; Lizotte, 2008), introduced in Papers B and C for ML parameter inference in SSMs where the likelihood can be intractable. We return to this approach in Section 4.4, where we introduce the method in detail.

4.1.2 Bayesian parameter inference

As for the ML parameter inference approach, most interesting Bayesian parameter inference problems are analytically intractable. Some problems can be solved in closed-form using conjugate priors as discussed in Section 2.4. However, in general we require approximate methods to solve expectation, marginalisation and

normalisation problems in Bayesian inference. Previously, we discussed the sampling based approach in which we make use of statistical simulations and computers to approximate the posterior. Examples of some sampling based approaches are MCMC methods or SMC methods. Some recent methods that makes us of the latter method are *SMC-squared* (SMC²) (Chopin et al., 2013) and *particle learning* (Carvalho et al., 2010). An alternative class of methods are based on approximate analytical computations, which often makes use of iterated approximations to approximate the posterior. Examples of analytical approximations are the *Laplace approximation*, the *integrated nested Laplace approximation* (INLA) (Rue et al., 2009), *variational Bayes* (VB) (Bishop, 2006) and *expectation propagation* (EP) (Minka, 2001).

In this thesis, we are primarily interested in sampling based inference using MCMC (Robert and Casella, 2004) for Bayesian parameter inference. MCMC is a family of algorithms that all are based on simulating a Markov chain with the target as its stationary distribution. After some *burn-in*, the chain reaches stationarity and the samples obtained from the chain can be treated as independent samples from the parameter posterior by the *ergodic theorem* discussed in Section 4.2.1. Here, we consider two instances of MCMC algorithms, the *Metropolis-Hastings* (MH) algorithm (Metropolis et al., 1953; Hastings, 1970) and the *Gibbs sampler* (Geman and Geman, 1984). The Gibbs sampler can be seen as a special case of the more general MH algorithm. As such it can only be used in models with a certain conditional structure of the joint parameter posterior. We return to the Gibbs sampler in Paper E for parameter inference in a non-Gaussian ARX model.

The MH algorithm is a general method for sampling from intractable distributions and is applied in many different fields. Some applications of the MH algorithm are found in biology (Wilkinson, 2011), system identification (Ninness and Henriksen, 2010), finance (Johannes and Polson, 2009) and statistics (Robert and Casella, 2004). A major problem with this algorithm is that it requires evaluations of the likelihood of the SSMs. As discussed in Section 3.3.4, we can only obtain an unbiased estimator of this quantity, but it turns out that this is enough. As a result of the unbiasedness property, the resulting Markov chain keeps the target as its stationary distribution. This type of methods are known as *pseudo-marginal* algorithms (Andrieu and Roberts, 2009; Andrieu et al., 2010) and this family includes the *particle MH* (PMH) and *particle Gibbs* (PG) algorithms. This opens up for Bayesian parameter inference in SSMs. We return to discussing the PMH algorithm in Section 4.3 and propose refinements to it in Paper A.

4.2 Metropolis-Hastings

In this section, we discuss the use of the MH algorithm for sampling from the parameter posterior $p(\theta|y_{1:T})$. Here, we give a short introduction to the MH algorithm and also briefly discuss why it works and discuss some interesting extensions of the algorithm based on random walks on Riemann manifolds. Interested readers are referred to Robert and Casella (2004), Gelman et al. (2013) and Liu (2008) for

more detailed accounts of the algorithm and its general application.

The MH algorithm samples the *target distribution* $\pi(\theta) = p(\theta|y_{1:T})$ by simulating a carefully constructed Markov chain on the parameter space Θ . The Markov chain is constructed in such a way that it admits the target as its unique stationary distribution. The algorithm consists of two steps: (i) a new parameter θ'' is sampled from a *proposal distribution* $q(\theta''|\theta')$ given the current state θ' and (ii) the current parameter is changed to θ'' with *acceptance probability* $\alpha(\theta'', \theta')$, otherwise the chain remains at the current parameter. The acceptance probability is given by

$$\alpha(\theta'', \theta') = 1 \wedge \frac{\pi(\theta'') q(\theta'|\theta'')}{\pi(\theta') q(\theta''|\theta')} = 1 \wedge \frac{p(\theta'') p_{\theta''}(y_{1:T}) q(\theta'|\theta'')}{p(\theta') p_{\theta'}(y_{1:T}) q(\theta''|\theta')}, \quad (4.3)$$

where $a \wedge b \triangleq \min\{a, b\}$ and $p_{\theta}(y_{1:T})$ denotes the likelihood. The resulting procedure is outlined in Algorithm 4 for parameter inference in models where we can compute the likelihood exactly. Here, we have used the form of the parameter posterior from (2.14), where the marginal likelihoods cancel in the acceptance probability. Therefore, the algorithm only requires that we can point-wise evaluate the unnormalised target distribution.

Note that the IS algorithm could be used to estimate the parameter posterior using e.g. a multivariate Gaussian distribution as the proposal as in Example 3.1. However, if θ is high dimensional this approach would require many samples to accurately estimate the posterior. In the MH algorithm, we could instead construct a Markov chain that explores the posterior distribution by local moves thus exploiting the previously accepted parameter. Hence, it focuses its attention to areas of the parameter space in which the posterior assigns a relatively large probability mass. This makes sampling of high dimensional target more efficient, as less iterations of the algorithm are required to obtain an accurate estimate.

To see this, assume that we have a *symmetric* proposal $q(\theta''|\theta') = q(\theta'|\theta'')$, so that the ratio between the proposals cancel in (4.3). The remaining part is the ratio between the target evaluated at θ'' and θ' . If the proposed parameter θ'' results in that the target assumes a larger value than in θ' , it is always accepted. Also, we accept a proposed parameter with some probability if this results in a small decrease of the target compared with the previous iteration. This results in that the MH sampler both allows for the algorithm to climb the posterior to its mode and explore the area surrounding the mode. Hence, the MH algorithm can possibly escape local extrema which is a problem for many local optimisation algorithms used in numerical ML parameter inference.

The performance of the MH algorithm is dependent on the choice of the proposal distribution. A poor proposal leads to a poor exploration of the posterior distribution, which results in that many iterations of the algorithm are required to obtain a good approximation of the posterior. Here, we discuss some common choices of proposals which are needed for the discussion in Section 4.2.2 and in Paper A. The perhaps simplest example is the *independent* proposal in which $q(\theta''|\theta') = q(\theta'')$, i.e. a parameter is proposed without taking the previously accepted parameter into

Algorithm 4 Metropolis-Hastings (MH) for Bayesian inference in SSMs

INPUTS: $M > 0$ (no. MCMC steps), $q(\cdot)$ (proposal) and θ_0 (initial parameter).
 OUTPUT: $\theta = \{\theta_1, \dots, \theta_M\}$ (samples from the parameter posterior).

```

1: Initialise using  $\theta_0$ .
2: for  $k = 1$  to  $M$  do
3:   Sample  $\theta'$  from the proposal  $\theta' \sim q(\theta'|\theta_{k-1})$ .
4:   Calculate the likelihood  $p_{\theta'}(y_{1:T})$ .
5:   Sample  $\omega_k$  from  $\mathcal{U}(0, 1)$ .
6:   if  $\omega_k < \alpha(\theta', \theta_{k-1})$  given by (4.3) then
7:      $\theta_k \leftarrow \theta'$ . {Accept the parameter}
8:   else
9:      $\theta_k \leftarrow \theta_{k-1}$ . {Reject the parameter}
10:  end if
11: end for

```

account. This proposal cannot exploit the previous accepted parameters and this could be a difficulty in inference problems when the posterior is quite complicated. However, if the proposal is similar to the posterior, this leads to a good mixing of the Markov chain as the proposed parameters are uncorrelated. This insight have been used in Giordani and Kohn (2010) to construct a proposal distribution based on a mixture of Gaussian distributions, which results in an efficient MH sampler in some applications.

Another popular choice is the (symmetric) Gaussian random walk (RW) with some step length ϵ , which results from selecting $q(\theta''|\theta') = \mathcal{N}(\theta''; \theta', \epsilon^2)$. The choice of the step length in the RW proposal determines the mixing of the Markov chain and thereby the efficiency of the exploration of the parameter posterior. If the step length is too small, the exploration is poor since that it takes a long time to explore the target. If it is too large, we seldom accept the proposed parameter as the difference in the values that the posterior assume is too large, resulting in a small acceptance probability. This is often referred to as the *mixing of the Markov chain* (c.f. Example 3.5). That is we would like to balance the mixing of the Markov chain to get a reasonable exploration of the posterior and acceptance rate. This problem is illustrated in Example 4.1, where we make use of Algorithm 4 to infer the parameters in an LGSS model for which we can compute the likelihood exactly. We return to the choice of proposal and the mixing of the Markov chain in following sections.

4.1 Example: Parameter inference in the LGSS model

Consider the parameter inference problem in the LGSS model (2.2) with the unknown parameter $\theta = \phi$. We use the parameter $\theta^* = 0.5$ together with $\{\sigma_v, \sigma_e\} = \{1.0, 0.1\}$ to generate a realisation from the model with $T = 250$ and known initial state $x_0 = 0$. Here, we use the MH algorithm defined in Algorithm 4 together with the Kalman filter for estimating the likelihood. A Gaussian random walk proposal is used with zero mean and variance ϵ . For simplicity, we use a uniform prior over

Figure 4.1: The traces (left) of the first 500 iterations of ϕ in the LGSS model (2.2) using the RW proposal with three different step lengths $\epsilon = 0.01$ (upper), $\epsilon = 0.10$ (middle) and $\epsilon = 1.00$ (lower). The estimated autocorrelations function (right) using 9000 iterations corresponding to each step length in the RW proposal. The dotted lines show the true parameters from which the data was generated.

Figure 4.2: The trace of the first 500 iterations of ϕ (upper left) and σ_v (lower left) in the LGSS model in Example 4.1. The estimated parameter posteriors (right) obtained from 9 000 iterations of the MH algorithm. The dotted lines show the true parameters from which the data was generated.

$\phi \in (-1, 1) \subset \mathbb{R}$ to ensure that the system is stable at all times.

In Figure 4.1, we present the trace plot of the Markov chain together with the estimated autocorrelation function from $M = 10\,000$ iterations (discarding the first 1 000 iterations as burn-in) for the RW proposal with three different step lengths. We see that the mixing is rather poor for the smallest step size $\epsilon = 0.01$, as this results in a poor exploration of the posterior and therefore a large correlation in the Markov chain. This is also the case if the step length is too large as illustrated by the proposal with step length $\epsilon = 1.00$. Here, the large step length results in a small acceptance probability and therefore a larger correlation in the Markov chain.

We now add the parameter σ_v to our inference problem and would therefore like to infer the parameters $\{\phi, \sigma_v\}$ using the same setup as before. Here, we also add an uniform prior over $\sigma_v \in \mathbb{R}_+$ as it corresponds to a standard deviation. We use the step length $\epsilon = 0.1$ with the RW proposal and simulate the Markov chain for $M = 10\,000$ iterations (discarding the first 1 000 iterations as burn-in). In Figure 4.2, we present the trace of each parameter and the resulting posterior density estimate. We see that the Markov chain mixes well in both parameters and that the posterior estimates rather close to the *true* parameters.

4.2.1 Statistical properties of the MH algorithm

In this section, we discuss some of the statistical results that the MH algorithm relies upon and briefly mention their underlying assumptions. For more information about the properties of the MH algorithm and MCMC algorithms in general, see e.g. Tierney (1994), Robert and Casella (2004) and Meyn and Tweedie (2009). The core result that is used in the MH algorithm is the *ergodic theorem* (Tierney, 1994; Robert and Casella, 2004). Given a well-behaved test function φ , we can construct a Monte Carlo estimator,

$$\hat{\varphi}_{\text{MH}} = \frac{1}{N} \sum_{i=1}^N \varphi(\theta_i),$$

where $\{\theta_i\}_{i=1}^M$ denotes the samples obtain from the MH algorithm. If the Markov chain is ergodic, then by the ergodic theorem this estimator is *strongly consistent*, i.e.

$$\hat{\varphi}_{\text{MH}} \xrightarrow{a.s.} \mathbb{E}[\varphi(\theta)], \quad N \rightarrow \infty.$$

Note that this property does not follow directly from the SLLN as the samples obtained for the posterior are not IID, as the proposed parameters are correlated. Hence, the ergodic theorem can be seen as the SLLN for correlated samples. Also, a CLT for the error in the estimator (under some conditions) is given by

$$\sqrt{N} \left[\hat{\varphi}_{\text{MH}} - \mathbb{E}[\varphi(\theta)] \right] \xrightarrow{d} \mathcal{N}(0, \sigma_\varphi^2),$$

where σ_φ^2 denotes the variance of the estimator, which is proportional to the mixing properties of the Markov chain. In fact, this variance is proportional to the *integrated autocorrelation time* (IACT), which is given by

$$\text{IACT}(\theta_{1:M}) = 1 + 2 \sum_{k=1}^{\infty} \rho_k(\theta_{1:M}).$$

The IACT cannot be computed analytically in practice as we do not know the true autocorrelation function for many models. Therefore, we often approximate it by

$$\widehat{\text{IACT}}(\theta_{1:M}) = 1 + 2 \sum_{k=1}^K \widehat{\rho}_k(\theta_{1:M}),$$

where $\widehat{\rho}_k(\cdot)$ denotes the empirical autocorrelation function at lag k and K denotes some maximum lag for which to compute the IACT. This value can be selected as a fixed value or by the first index at which the ACF becomes statistically insignificant, i.e. the first index K such that $|\widehat{\rho}_K(\theta_{1:M})| < 2/\sqrt{M}$. Another related measure is the *effective sample size* (ESS) defined as

$$\text{ESS}(\theta_{1:M}) = \frac{M}{\widehat{\text{IACT}}(\theta_{1:M})} = \frac{M}{1 + 2 \sum_{k=1}^{\infty} \rho_k(\theta_{1:M})}, \quad (4.4)$$

which can be approximated in the same manner as the IACT. The IACT and ESS have the interpretations as the number of iterations between each independent sample and the number of independent samples obtained from the posterior, respectively. Hence, we would like to minimise the IACT and maximise the ESS to get an optimal mixing in the Markov chain and many uncorrelated samples from the posterior. We illustrate the ESS values for parameter inference in an LGSS model in Example 4.2.

4.2 Example: ESS values for inference in the LGSS model

We return to Example 4.1 and calculate the ESS values for the three different Markov chains considered in Figure 4.1. The ESS values (4.4) are $\{22, 1292, 353\}$ when calculated using the adaptive method for the RW proposal with each of the three step lengths, respectively. These number validates the previous discussion that the proposal with $\epsilon = 0.10$ is preferable for this problem, as this maximises the number of uncorrelated samples from the posterior.

The consistency results and the CLT relies on the ergodic theorem, which assumes that the Markov chain is *irreducible* and *aperiodic*. Irreducible means that we should be able to get from any state to any state in the state space. Aperiodicity means that the Markov chain does not return to a state with regular intervals, i.e. no loops that the chain can get stuck in. These requirements need to be check for each MH algorithm to validate its assumptions. However, often in practice these assumptions hold for at least unimodal parameter posteriors.

Another property that is important for a Markov chain used in MCMC is *reversible*.

This property holds if the chain satisfies the *detailed balance* equation,

$$\pi(\theta'')K(\theta'', \theta') = \pi(\theta')K(\theta', \theta''),$$

where $K(\theta'', \theta')$ denote the Markov transition kernel. This results in that we can write

$$\int \pi(\theta'')K(\theta'', \theta') d\theta'' = \int \pi(\theta')K(\theta', \theta'') d\theta'' = \pi(\theta') \int K(\theta', \theta'') d\theta'' = \pi(\theta'),$$

which shows that π is an invariant distribution of K , i.e. admits $\pi(\theta)$ as its stationary distribution. To show that the MH algorithm, satisfies the detailed balance equation we follow Liu (2008) and write the transition kernel of the Markov chain as,

$$K(\theta'', \theta') = q(\theta'', \theta') \min \left\{ 1, \frac{\pi(\theta')q(\theta', \theta'')}{\pi(\theta'')q(\theta'', \theta')} \right\}, \quad \theta'' \neq \theta'$$

which gives

$$\pi(\theta'')K(\theta'', \theta') = \min\{\pi(\theta'')q(\theta'', \theta'), \pi(\theta')q(\theta', \theta'')\},$$

which is a symmetric function in θ'' and θ' . This results in that the detailed balance is fulfilled and the resulting Markov chain obtained from the MH algorithm is reversible.

4.2.2 Proposals using Langevin and Hamiltonian dynamics

We have previously discussed two different proposals for the MH algorithm and that these could perform better than the IS algorithm when the target is high dimensional. It turns out that the RW proposal in the MH algorithm still scales rather poorly as the dimension of the parameter space d increases, see Roberts et al. (1997). That is, the mixing of the Markov chain decreases and more iterations are required to maintain the number of independent samples from the target. This is a result of that the random walk does not explore the parameter space well when the dimension increases.

A modification to the random walk proposal that could increase the mixing in the Markov chain is to add a drift term that is proportional to the gradient of the log-target. This leads to a proposal which is the noisy equivalent of (4.1). In statistics, the resulting algorithm is known as the *Metropolis adjusted Langevin algorithm* (MALA) (Roberts and Rosenthal, 1998; Neal, 2010), where the proposal is said to follow a Langevin dynamics. This means that the proposal can be seen as the discrete version of a continuous time *Langevin diffusion* process (Øksendal, 2010; Kloeden and Platen, 1992). The Langevin diffusion with stationary distribution $p(\theta|y_{1:T})$ is given by the stochastic differential equation,

$$d\theta(\tau) = \frac{1}{2} \left[\nabla \log p(\theta|y_{1:T}) \Big|_{\theta=\theta(\tau)} \right] d\tau + dB(\tau),$$

where $B(\tau)$ denotes a Brownian motion. To obtain samples from the parameter posterior, we can simulate the Langevin diffusion to stationarity, which is useful in the proposal as it guides the process towards the mode of the posterior. The

discrete time proposal used in MALA follows from a first order Euler discretisation of (4.5) as

$$q(\theta''|\theta') = \mathcal{N}\left(\theta''; \theta' + \frac{\epsilon^2}{2}\mathcal{S}(\theta'), \epsilon^2 I_d\right),$$

where ϵ denotes a step length of the discretisation and the corresponding proposal. The proposal can also be derived using a Laplace approximation of the log-posterior as discussed in Paper A. Another version of this algorithm is the manifold MALA (mMALA) (Neal, 2010; Girolami and Calderhead, 2011) which also includes the Hessian information of the log-posterior in analogue with Newton algorithms (c.f. (4.2)). This proposal can be derived similar to the MALA proposal and has the form

$$q(\theta''|\theta') = \mathcal{N}\left(\theta''; \theta' + \frac{\epsilon^2}{2}\mathcal{J}^{-1}(\theta')\mathcal{S}(\theta'), \epsilon^2 \mathcal{J}^{-1}(\theta')\right),$$

where we include the observed information matrix $\mathcal{J}(\theta)$ into the proposal. We make use of the MALA and mMALA proposals in Paper A to construct particle versions of the algorithms for parameter inference in SSMs. Alternative algorithms that solves the same problem are proposed by Girolami and Calderhead (2011), where the MALA and mMALA are used for parameter inference in the SV model.

In this thesis, we adopt an optimisation mind set and refer to the MALA and mMALA algorithm as first order and second order proposals, respectively. The names of the proposals refer to the use of first order information (the gradient) and the second order information (the Hessian) in the proposals. In the optimisation literature, this corresponds to the first order gradient-based search method and second order Newton method, respectively. The corresponding MH algorithms are therefore called MH1 and MH2, respectively.

Another perspective of the second order proposal is that the added gradient and Hessian information about the log-target is used to construct a random walk on a Riemann manifold (Livingstone and Girolami, 2014; Girolami and Calderhead, 2011). For this end, we require that the negative Hessian is a PD matrix which is not always the case when we use the observed information matrix as the second order information. In Girolami and Calderhead (2011), it is proposed that the expected information matrix should be used instead, but this is computational costly to estimate for an SSM. Therefore, we make use of methods from optimisation (Nocedal and Wright, 2006) to make the observed information matrix PD if necessary. This is done by *regularisation*, i.e. adding an appropriate diagonal matrix to make the negative eigenvalues positive, see Paper A for more information.

Another related method is called manifold Hamiltonian MCMC (mHMC) and it can improve the mixing of the Markov chain further. This method originates from physics and is one instance of *hybrid MC* (Duane et al., 1987) algorithms, which are used in statistical physics to simulate from high dimensional targets. Here, we shortly discuss their use in a statistical setting for parameter inference. Interested readers are referred to Liu (2008), Neal (2010) and Girolami and Calderhead (2011)

for more information. The mHMC algorithm is based on simulating a physical system with the *Hamiltonian* (the total energy),

$$H(\theta, p) = U(\theta) + K(p), \quad (4.5)$$

where $p \sim \mathcal{N}(0, \mathcal{J}^{-1}(\theta))$ denotes a random fictitious momentum for each parameter. Here, the *potential energy function* and the *kinetic energy function* are given by $U(\theta) = -\log \pi(\theta)$ and $K(p) = p^\top \mathcal{J}^{-1}(\theta)p/2$, respectively. The proposal simulates this Hamiltonian system for a number of steps L using the so-called *leap-frog algorithm* (Neal, 2010) with some step size ϵ . The result from this procedure is the proposed parameter and the resulting momentum $\{\theta'', p''\}$. This pair is accepted/rejected using a similar procedure as in the MH algorithm. The acceptance probability compares the Hamiltonian of the system at the last accepted parameter and the proposed parameter by

$$\alpha(\theta'', \theta') = 1 \wedge \exp(H(\theta'', p'') - H(\theta', p')). \quad (4.6)$$

As the accept/reject decision is delayed for L steps, this proposal allows for the chain to move a larger distance between the iterations of the algorithm. This increases the mixing of the Markov chain and it also allows for the Markov chain to visit isolated modes in the posterior. This leads to a better exploration of the posterior as well as more effective samples. The mHMC algorithm is used in many applications and is currently a popular algorithm in statistics and machine learning, see e.g. Chen et al. (2014), Beam et al. (2014) and Betancourt and Girolami (2013). In Girolami and Calderhead (2011), the authors make use of the mHMC algorithm for parameter inference in a SV model with impressive results. To adapt the mHMC algorithm to the PMCMC framework is therefore an exciting opportunity for future research.

We conclude this section with an comparison between the three MCMC algorithms previously discussed. In Example 4.3, we replicate and extend an illustration from Neal (2010) to compare the different methods when sampling from a high-dimensional target distribution.

— 4.3 Example: Parameter inference in a 100-dimensional Gaussian dist. —

Consider the problem of sampling from a non-isotropic 100-dimensional Gaussian distribution with zero-mean and covariance matrix $\Sigma = \text{diag}(1.00, 0.99, \dots, 0.01)$. In this problem, we use random step sizes where $\epsilon_{\text{RW}} \sim \mathcal{U}(0.018, 0.026)$ for MH-RW and $\epsilon_{\text{HMC}} \sim \mathcal{U}(0.010, 0.016)$ for HMC with $L = 150$. These values follow the suggestions by Neal (2010). For MALA, we use the step length $\epsilon_{\text{LMC}} \sim \mathcal{U}(0.008, 0.013)$, which results in an acceptance rate of about 60%. The HMC algorithm is executed for 1000 iterations and the RW and MALA algorithms are executed for 150000 iterations but only every 150th iteration is considered to make a fair comparison with the HMC. The resulting acceptance probabilities are 0.26, 0.60 and 0.88, for each proposal respectively.

The trace plots and ACF for the first coordinate (with standard deviation 1) are presented in Figure 4.3. The estimated mean and standard deviations of the target distribution are presented in Figure 4.4. The HMC algorithm gives almost

Figure 4.3: The trace of x_1 (left) and the corresponding estimated ACF (right) for the RW-MH, MALA and HMC algorithms.

Figure 4.4: The estimated parameters obtained for each coordinate from the RW-MH, MALA and HMC algorithms. The estimates of the mean vector (left) and the diagonal of the covariance matrix (right) are presented for each coordinate. The dashed lines indicate the true parameter values for each coordinate.

independent samples from the target. The ACF falls off quicker for the MALA compared with the RW-MH, which indicates a more efficient exploration of the posterior. However, the estimates of the covariance matrix are biased for the MALA, which could be the result of that the Markov chain gets stuck somewhere in the parameter space. We conclude that there is a large gain in using the HMC algorithm for inference in models with a high-dimensional parameter vector.

4.3 Particle Metropolis-Hastings

As previously discussed in Chapter 2, the likelihood of an SSM is analytically intractable and therefore we cannot make use of the MH algorithm directly for parameter inference. In Section 3.3.4, we reviewed how to construct an unbiased estimator for the likelihood based on the APF. A natural solution to this problem could therefore be to replace the intractable quantities in the acceptance probability with unbiased estimates.

This idea is first used by Beaumont (2003), where the authors replace an analytical intractable likelihood with an unbiased estimate for a genetics application of the MH algorithm. The first use of this idea for parameter inference in nonlinear SSMs is found in Fernández-Villaverde and Rubio-Ramírez (2007), where the intractable likelihood is replaced with an estimate from the APF. Subsequent work by Andrieu and Roberts (2009) and Andrieu et al. (2010) analyse the resulting PMH algorithm and prove that this is a valid approach to solve the problem. To see why this method works, we review the derivation of the PMH algorithm following the presentation in Flury and Shephard (2011).

Consider the problem of using the MH algorithm to sample the parameter posterior (2.14), i.e. $\pi(\theta) = p(\theta|y_{1:T})$. This implies that the acceptance probability (4.3) depends explicitly on the intractable likelihood $p_\theta(y_{1:T})$, preventing direct application of the MH algorithm to this problem. Instead, assume that there exists an unbiased, non-negative estimator of the likelihood $\hat{p}_\theta(y_{1:T}|u)$, i.e.

$$\mathbb{E}_{u|\theta} [\hat{p}_\theta(y_{1:T}|u)] = \int \hat{p}_\theta(y_{1:T}|u) m_\theta(u) \mathrm{d}u = p_\theta(y_{1:T}) \quad (4.7)$$

where $u \in \mathcal{U}$ denotes the multivariate random variable (vector) used to construct this estimator. Here, $m_\theta(u)$ denotes the probability density of u on $\mathcal{U}$. When the APF is used to construct the estimator of the likelihood, the random variable u is the particles and their ancestors $\{x_{0:T}^{(i)}, a_{1:T}^{(i)}\}_{i=1}^N$.

The PMH algorithm can then be seen as a standard MH algorithm operating in a non-standard extended space $\Theta \times \mathcal{U}$, with the *extended target* given by

$$\pi(\theta, u|y_{1:T}) = \frac{\hat{p}_\theta(y_{1:T}|u) m_\theta(u) p(\theta)}{p(y_{1:T})} = \frac{\hat{p}_\theta(y_{1:T}|u) m_\theta(u) p(\theta|y_{1:T})}{p_\theta(y_{1:T})},$$

and the proposal distribution $m_{\theta''}(u'')q(\theta''|\theta')$. As a result, we can recover the

Algorithm 5 Particle Metropolis-Hastings (PMH) for Bayesian inference in SSMs

INPUTS: Algorithm 2, $M > 0$ (no. MCMC steps), $q(\theta'', \theta')$ (proposal) and θ_0 (initial parameters).

OUTPUT: $\theta = \{\theta_1, \dots, \theta_M\}$ (samples from the parameter posterior).

```

1: Initialise using  $\theta_0$ .
2: for  $k = 1$  to  $M$  do
3:   Sample  $\theta'$  from the proposal  $\theta' \sim q(\theta'|\theta_{k-1})$ .
4:   Estimate the likelihood  $\hat{p}_{\theta'}(y_{1:T})$  using Algorithm 2 and (3.19).
5:   Sample  $\omega_k$  from  $\mathcal{U}(0, 1)$ .
6:   if  $\omega_k < \alpha(\theta', \theta_{k-1})$  given by (4.8) then
7:      $\{\theta_k, \hat{p}_{\theta_k}(y_{1:T})\} \leftarrow \{\theta', \hat{p}_{\theta'}(y_{1:T})\}$ . {Accept the parameter}
8:   else
9:      $\{\theta_k, \hat{p}_{\theta_k}(y_{1:T})\} \leftarrow \{\theta_{k-1}, \hat{p}_{\theta_{k-1}}(y_{1:T})\}$ . {Reject the parameter}
10:  end if
11: end for

```

parameter posterior by marginalisation of the extended target,

$$\int \pi(\theta, u|y_{1:T}) du = \frac{p(\theta|y_{1:T})}{p_{\theta}(y_{1:T})} \underbrace{\int \hat{p}_{\theta}(y_{1:T}|u) m_{\theta}(u) du}_{=p_{\theta}(y_{1:T})} = p(\theta|y_{1:T}),$$

using the unbiasedness property (4.7) of the likelihood. Samples from the parameter posterior can therefore be obtained as a byproduct by simulating from $\pi(\theta, u|y_{1:T})$. By selecting the proposal distribution as $q(\theta''|\theta', u)$, the acceptance probability is given by

$$\alpha(\theta'', \theta') = 1 \wedge \frac{\hat{p}_{\theta''}(y_{1:T}|u'')}{\hat{p}_{\theta'}(y_{1:T}|u')} \frac{p(\theta'')}{p(\theta')} \frac{q(\theta''|\theta', u')}{q(\theta'|\theta'', u'')}. \quad (4.8)$$

Note, that the acceptance probability is the same as for the MH algorithm, but replacing the intractable likelihood with an unbiased estimator and including an *extended proposal*. As previously discussed, the random variable u contains the entire particle system generated by the APF algorithm. From Section 3.4.2, we know that this information can be used in combination with the FL smoother to estimate the score and information matrix, which can be used to construct particle versions of the MH1 and MH2 algorithms. This is the main idea behind the proposed PMH1 and PMH2 algorithms in Paper A, to which we refer interested readers for more information.

It turns out that the acceptance rate of the PMH algorithm is closely connected with the number of particles used to estimate the log-likelihood. If N is too small, then the variance of the log-likelihood estimates are large and therefore we often get stuck with the Markov chain with a resulting low acceptance rate. We also know that N is connected with the computational cost of the APF algorithm. Therefore, we have a trade-off between the number of MCMC iterations and the number of

particles in the APF, where we would like to minimise the total computational cost of the PMH algorithm. This problem is analysed and discussed by Doucet et al. (2012), Pitt et al. (2010) and Pitt et al. (2012). From this work, it is recommended to use a value of N such that the variance of the log-likelihood estimates is between 0.25 and 2.25. Consequently, we can determine the *optimal* number of particles by a pilot run.

Two other common versions of the PMH algorithm are the *particle independent MH* (PIMH) algorithm and the *particle marginal MH* (PMMH or PMH0) algorithm. The Gaussian versions of these proposals are obtained by using $q(\theta'') = \mathcal{N}(\theta''; 0, \epsilon^2)$ and $q(\theta''|\theta') = \mathcal{N}(\theta''; \theta', \epsilon^2)$, respectively. The resulting general version of the PMH algorithm that incorporates the PMH0 and PIMH as special cases is presented in Algorithm 5. The full procedure for PMH1 and PMH2 are found in Algorithm 1 in Paper A.

4.4 Example: PMH0 for parameter inference in the GARCH(1,1) model

Consider the parameter inference problem for the GARCH(1,1) model (2.3) using the NASDAQ OMX Stockholm 30 Index data from Section 2.2.2. We make use of the PMH0 algorithm and the RW proposal with the step length $\epsilon = 0.005$ for all elements in the parameter vector. We run the algorithm for $M = 20\,000$ iterations (discarding the first 5000 as burn-in) with the faPF algorithm and $N = 3\,000$ particles for estimating the likelihood.

The resulting acceptance rate is 0.06 (after burn-in) with ESS $\{39, 21, 21, 11\}$ for the four parameters, respectively. Note that the poor ESS are due to that we for simplicity make use of the same step length for all parameters. In Figure 4.5, we present the trace plots and posterior estimates obtain from the run. We see that the mixing is rather poor and longer runs with smaller step lengths are needed. Also, we could make use of PMH1 or PMH2 from Paper A to improve the mixing. The posterior means are $\hat{\theta}_{\text{PMH}} = \{0.0088, 0.14, 0.86, 0.63\}$, which can be used as point estimates of the parameters in the model.

4.4 Bayesian optimisation

BO (Jones, 2001; Boyle, 2007; Lizotte, 2008; Osborne, 2010) is a popular (global) derivative-free optimisation method, which is currently studied extensively in the machine learning community. Here, we consider the use of BO to solve

$$x_{\max} = \operatorname{argmax}_{x \in \mathcal{B}} f(x), \quad (4.9)$$

where $\mathcal{B} \subset \mathbb{R}^d$ denotes some compact set. Here, we assume that we cannot directly evaluate the real-valued function $f(x)$, but can obtain noisy estimates (samples) modelled as

$$y_k = f(x_k) + z_k, \quad z_k \sim \mathcal{N}(0, \sigma_z^2), \quad (4.10)$$

Figure 4.5: The trace plots (left) and the corresponding posterior estimates (right) of the GARCH(1,1) model using the Nasdaq OMX Stockholm 30 Index data from Section 2.2.2. The estimates are computed using the PMH0 algorithm with 20 000 iterations and discarding the first 5 000 as burn-in.

where z_k denotes some zero-mean Gaussian noise with unknown variance σ_z^2 . Therefore, the BO algorithm is useful when we can only estimate the value of the objective function with some simulation based algorithm. It also turns out that the BO algorithm requires less estimates of the objective function than other optimisation algorithms (Brochu et al., 2010). As a consequence, BO is useful when the noisy estimates of the objective function are computationally costly to obtain. In the following, we make use of BO for ML based parameter inference and for input design in nonlinear SSMS. In these cases, the objective function corresponds to the log-likelihood and the logarithm of the determinant of the expected information matrix, respectively. In both these cases, these quantities are analytically intractable but we can obtain noisy estimates from the objective function by the use of computationally costly particle filtering and smoothing. Hence, these are two applications which fits the BO algorithm well.

The BO algorithm operates by constructing a *surrogate function* also called a *response surface* that emulates the objective function. In BO, this surrogate function is modelled as a probabilistic function with some prior form selected by the user. The name comes from that the samples from the objective function are used together with the prior to update the model using Bayes' theorem. Using the updated posterior, we can predict the value of the objective function anywhere in the space of interest and also obtain an uncertainty of the predicted value.

Using the predictive distribution, the algorithm can analyse where the optimum of the objective function could be located and focus the sampling to that area. Also, the algorithm can choose to explore areas where there are large uncertainties in the predicted value of the objective function. We refer to these two situations as *exploitation* and *exploration*, respectively. In the following, we discuss this part of the algorithm in more detail and the *acquisition rules* that is used to make the decisions regarding exploitation and exploration. To conclude this overview of the BO algorithm, we present the three steps that are carried out during the k th iteration:

- (i) Sample the objective function in x_k to obtain y_k .
- (ii) Use the data collected $\mathcal{D}_k = \{x_i, y_i\}_{i=1}^k$ to construct a *surrogate function*.
- (iii) Use an *acquisition rule* and the surrogate function to select x_{k+1} , i.e. where to sample the objective function in the next iteration of the algorithm.

We now proceed by discussing Steps (ii) and (iii) in more detail. Step (i) depends on the specific optimisation problem that we would like to solve and for the two previously discussed applications correspond to the APF and the FL smoother in Algorithms 2 and 3, respectively.

In this thesis, we make use of GPs (Rasmussen and Williams, 2006) as the surrogate function. Therefore, we devote the following section to introducing the GP and discuss its structure and how to combine it with the obtained samples from the objective function. Then, we discuss some different acquisition functions and compare their properties. Finally, we combine the GP with the BO algorithm to

obtain the *Gaussian process optimisation* (GPO) algorithm. We conclude this section by discussing some applications of GPO in connection with SSMs. For more information regarding BO, see Lizotte (2008), Boyle (2007), Brochu et al. (2010) and Snoek et al. (2012).

4.4.1 Gaussian processes as surrogate functions

GPs are an instance of *Bayesian nonparametric models* and has its origins in *kriging* (Cressie, 1993; Matheron, 1963) methods from spatial statistics. An application of kriging is to interpolate between elevation measurements sampled in some terrain to build a map of the elevation in an area. The underlying assumption is that the elevation should vary smoothly between sampled points. This is the property of the GP that we would like to use for interpolating the objective function between the values in which we sample it.

A GP (Rasmussen and Williams, 2006) can be seen as a generalisation of the multivariate Gaussian distribution to an infinite dimension. As such, it is a collection of random variables, where each finite subset is jointly distributed according to a Gaussian distribution. A realisation drawn from a GP can therefore be seen as an infinitely long vector of values, which can be seen as a function over the real space $\mathbb{R}^d$. This is why the GP is considered by some to be a *prior over functions* on $\mathbb{R}^d$.

As the GP is a Gaussian distribution of infinite dimension, we cannot characterise it using a mean vector and covariance matrix. Instead, we introduce a *mean function* $m(x)$ and a *kernel* (or covariance function) $\kappa(x, x')$ defined as

$$m(x) = \mathbb{E}[f(x)], \quad (4.11a)$$

$$\kappa(x, x') = \mathbb{E}\left[(f(x) - m(x))(f(x') - m(x'))\right]. \quad (4.11b)$$

To construct the surrogate function, we assume a priori that the objective function can be modelled as a GP,

$$f(x) \sim \mathcal{GP}(m(x), \kappa(x, x')), \quad (4.12)$$

with the mean function and kernel defined by (4.11). Here, the mean function specifies the average value of the process and the kernel specifies the correlation between (nearby) samples. Both functions are considered to be prior choices to the algorithm and are used to encode the beliefs about the data before it is observed.

Consequently, we have that both the prior (4.12) and the data likelihood (4.10) are distributed according to Gaussian distributions. Hence, the posterior resulting from Bayes' theorem is a Gaussian distribution with some mean and covariance that can be calculated using standard results. From this posterior, we can con-

struct the *predictive distribution* at some *test point* $x_\star$ given the data $\mathcal{D}_k$ by

$$f(x_\star) | \mathcal{D}_k \sim \mathcal{N}\left(x_\star; \mu_f(x_\star | \mathcal{D}_k), \sigma_f^2(x_\star | \mathcal{D}_k)\right), \quad (4.13a)$$

$$\mu_f(x_\star | \mathcal{D}_k) = \kappa_\star^\top \left[\kappa(x_{1:k}, x_{1:k}) + \sigma_z^2 I_k \right]^{-1} y_{1:N}, \quad (4.13b)$$

$$\sigma_f^2(x_\star | \mathcal{D}_k) = \kappa(x_\star, x_\star) - \kappa_\star^\top \left[\kappa(x_{1:k}, x_{1:k}) + \sigma_z^2 I_k \right]^{-1} \kappa_\star + \sigma_z^2, \quad (4.13c)$$

where $\kappa_\star = \kappa(x_\star, x_{1:k})$ denotes the covariance between the test value and the sampling points. Here, $\kappa(x_{1:k}, x_{1:k})$ denotes a matrix where the element at (i, j) is given by $\kappa(x_i, x_j)$ for $i = 1, \dots, k$ and $j = 1, \dots, k$.

To obtain the GP posterior, we need to select a kernel function. Note that, it is possible to include the assumption of a non-zero mean function into the kernel function by adding an appropriate constant kernel. Therefore, we only make use of a zero mean function in this thesis and focus on the *kernel design problem*, where several kernels can be combined by different operations to encode the prior beliefs of the structure in the data. Here, we only consider the combination of a constant covariance function and three different popular kernels: the squared exponential (SE), the Matérn 3/2 and the Matérn 5/2. See Rasmussen and Williams (2006) for other kernels and for a discussion of how they can be combined.

The SE kernel is also known as radial basis function (RBF) and has the form

$$\kappa_{\text{SE}}(x, x') = \sigma_\kappa^2 \exp\left(-\frac{(x - x')^2}{2l^2}\right), \quad (4.14)$$

where the hyperparameters are $\alpha = \{\sigma_\kappa^2, l\}$. Here, l is called the *characteristic length scale* as it scales the Euclidean distance between the two points x and x' . Two other kernels are the Matérn 3/2 and the Matérn 5/2 with the form

$$\kappa_{3/2} = \sigma_\kappa^2 \left(1 + \frac{\sqrt{3}(x - x')}{l}\right) \exp\left(-\frac{\sqrt{3}(x - x')}{l}\right), \quad (4.15a)$$

$$\kappa_{5/2} = \sigma_\kappa^2 \left(1 + \frac{\sqrt{5}(x - x')}{l} + \frac{5(x - x')^2}{3l^2}\right) \exp\left(-\frac{\sqrt{5}(x - x')}{l}\right), \quad (4.15b)$$

where the hyperparameters are $\alpha = \{\sigma_\kappa^2, l\}$. The main difference between the three kernels are their smoothness properties. The SE kernel is the smoothest and it has infinitely many continuous derivatives. The $\kappa_{3/2}$ and $\kappa_{5/2}$ kernels only have one and two continuous derivatives, respectively. To illustrate the different kernels, we present some simulated realisations from each in Example 4.5.

4.5 Example: GP kernels

In Figure 4.6, we present realisations from the GP prior (4.12) using three different kernels with two different length scales. We see that the smoothness of the prior decreases from top to bottom and this verifies the previous discussion. Also, we see that the length scale determines the rate of change in the realisations.

Figure 4.6: Realisations simulated from the GP prior using three different kernels: SE (upper), Matérn 5/2 (middle) and Matérn 3/2 (lower) and length scales 1 (left) and 3 (right).

Finally, we need to determine suitable values for the hyperparameters in the kernel. This can be done using an ML procedure called *empirical Bayes* (EB), where the marginal likelihood of the data is optimised with respect to the hyperparameters α . Note that this is not a pure Bayesian approach as the data is used to determine the properties of the prior. Nevertheless, it is a popular approach in the GP literature and therefore we make use of it here. The marginal likelihood can be computed by a marginalisation,

$$p(y|x, \alpha) = \int p(y|f, x, \alpha) p(f|x, \alpha) df,$$

where we drop the subscript on $y_{1:k}$, $x_{1:k}$ and $f_{1:k}$ for brevity. The *log-marginal likelihood* can be obtained in closed form using results for the Gaussian distribution as

$$\log p(y|x, \alpha) \propto -\frac{1}{2} y^\top \left[\kappa(x, x) + \sigma_z^2 I_N \right]^{-1} y - \frac{1}{2} \log \left| \kappa(x, x) + \sigma_z^2 I_N \right|,$$

where we have neglected the terms that are independent of α (independent of the kernel). The gradient of the log-marginal likelihood for α_j can be computed using

$$\frac{\partial}{\partial \alpha_j} \log p(y|x, \alpha) = \frac{1}{2} \text{tr} \left(\left(\beta \beta^\top - \kappa(x, x) \right)^{-1} \frac{\partial}{\partial \theta_j} \kappa(x, x) \right), \quad \beta = \kappa(x, x)^{-1} y.$$

Therefore, the hyperparameters can be estimated by maximising the log-marginal likelihood using a gradient-based search algorithm (4.1). In Example 4.6, we make use of the EB method for GP regression using the different kernels discussed in this section.

4.6 Example: GP regression

Consider the GP regression problem where we would like to recover the underlying function $f(x)$ given by

$$f(x) = \left[3 \cos(0.5x) + \sin(2 + 0.25x) \right]^2,$$

from the noisy measurements y generated by (4.10) with $\sigma_z^2 = 2$. Here, we make use of the three kernels in (4.14) and (4.15) with an added constant kernel to account for the non-zero mean of the data. The hyperparameters of the resulting kernels are estimated using the EB procedure.

In the left part of Figure 4.7, we use $N = 5$ observations and present the resulting predictive mean (solid line), the underlying function $f(x)$ (dashed) and the 95% CI of the predictive distribution (gray area). We see that most of the samples are located in the left part of the region and as a result the predictive means of the GPs follows the underlying function.

In the right part of Figure 4.7, we present the same setup but using $N = 15$ sampled points instead. Here, the overall fit is much better and the three GP predictive distributions recover the underlying function pretty well across the region. As the underlying function is smooth (have infinity many continuous derivatives), it is well captured by the SE kernel.

Figure 4.7: The GP regression problem in Example 4.6 using $N = 5$ (left) and $N = 15$ (right) data points with three different kernels. The mean of the predictive distributions (solid lines) and the corresponding 95% CI (gray area) are presented together with the true function (dashed line) and the noisy observations (black dots).

4.4.2 Acquisition rules

We now proceed by discussing Step (iii) of the BO algorithm, i.e. the acquisition rule and how it operates. The main idea with this rule is to make use of the predictive mean and its uncertainty to decide on a good parameter to sample the objective in during the next iteration. We would like the algorithm to explore the parameter space to find all the peaks, but still focus the samples around the maximum to decrease the number of samples required to solve the problem. This is called the *exploration and exploitation trade-off* as we would like to explore the space, but also exploit the information encoded in the surrogate function about about where the maxima are located.

Using a general acquisition rule $AQ(x|\mathcal{D}_k)$, we would like to select x_{k+1} in Step (iii) as the maximising argument,

$$x_{k+1} = \operatorname{argmax}_{x \in \mathcal{B}} AQ(x|\mathcal{D}_k), \quad (4.16)$$

which in itself is an optimisation problem. We discuss some different methods for solving this in Paper B and now instead proceed with discussing three different acquisition rules that are popular and widely used in GPO (Brochu et al., 2010). These are: *the probability of improvement* (PI), *the expected improvement* (EI) and *the upper confidence bound* (UCB).

The PI and EI make use of the fact that the predictive distribution is Gaussian and that we can use the predictive mean and covariance. From this, we can use the probability density function (PDF) and the cumulative distribution function (CDF) of the Gaussian distribution to calculate the PI and EI. This can be done by introducing the highest predicted value of the surrogate function (or the incumbent),

$$\mu_{\max} = \max_{x_i \in x_{1:k}} \mu_f(x_i|\mathcal{D}_k), \quad (4.17)$$

then the PI (Kushner, 1964) can be computed using

$$PI(x|\mathcal{D}_k) = \mathbb{P}(\mu_f(x|\mathcal{D}_k) \geq \mu_{\max} + \xi) = \Phi(Z_k), \quad (4.18a)$$

$$Z_k = \frac{\mu_f(x|\mathcal{D}_k) - \mu_{\max} - \xi}{\sigma_f(x|\mathcal{D}_k)}, \quad (4.18b)$$

where $\Phi(\cdot)$ denotes the Gaussian CDF. Here, $\xi \geq 0$ denotes a user-defined coefficient proposed in Lizotte (2008) to balance the exploitation and exploration behaviour. From the form of the PI expression, we note that the variable Z_k can be seen as a standard Gaussian random variable. Hence, Z_k assumes a (large) positive value if the predictive mean is close to or larger than $\mu_{\max}$. Therefore, we obtain a value of the PI close to one and it is probable that the GPO algorithm will sample the objective function in this region during its next iteration. Instead, we obtain a small value of the PI if the predictive mean is much smaller than $\mu_{\max}$ and/or the uncertainty is very large. A small value of the PI results in that it is unlikely that the GPO algorithm will sample the objective function in this region during its next iteration.

However, the PI only takes into account the probability of an improvement and not its size. To include this information, we consider the EI rule (Mockus et al., 1978; Jones et al., 1998) of the form

$$\text{EI}(x|\mathcal{D}_k) = (\mu_f(x|\mathcal{D}_k) - \mu_{\max} - \xi)\Phi(Z_k) + \sigma_f(x|\mathcal{D}_k)\phi(Z_k), \quad (4.19)$$

where $\phi(\cdot)$ denotes the Gaussian PDF. The interpretation of the EI follows in analogue with the PI rule. If the predictive mean is close to or larger than $\mu_{\max}$ then $\Phi(Z_k)$ assumes a value close to one, which scales the expected gain in the objective function. Here, we also take the uncertainty into account by the second term in (4.19). This term can be seen as a scaling of the uncertainty in the predictive distribution. The scaling is large if Z_k is close to zero and decreases for larger value. This means that we get an extra contribution to the EI if the predictive mean is close to $\mu_{\max}$, which means that it could be interesting to explore that area. For more information, see Paper B where we review the derivation of the EI rule.

The third acquisition rule, the UCB rule follows from that we can construct a CI using the predictive distribution. The intuition for this is that if the predictive mean is high in an area of the parameter space, the resulting UCB is also large. Moreover, uncertainty in a region increases the predictive covariance, which also increases the value of the UCB. By this rule, we therefore explore areas where peaks have been found and where the uncertainty is large. As the name suggests, we are interested in the upper bound which for a Gaussian distribution is given by

$$\text{UCB}(x|\mathcal{D}_k) = \mu_f(x|\mathcal{D}_k) + \epsilon\sigma_f(x|\mathcal{D}_k). \quad (4.20)$$

Here, $\epsilon \geq 0$ denotes a coefficient determining the confidence level of the interval. As the predictive distribution is Gaussian, we would choose $\epsilon = 1.96$ to obtain a 95% CI and $\epsilon = 2.58$ to obtain a 99% CI.

4.7 Example: GPO using different acquisition rules

Consider again the problem in Example 4.6 using the Matérn 5/2 kernel with $N = 3$, $N = 5$ and $N = 15$ samples. In the left of Figure 4.8, we present the predictive distributions together with the underlying function as before. In the right part of the figure, we present the normalised value of the three different acquisition functions previously discussed: the PI (green), the EI (red) and the UCB (blue). Here, we use the recommended value of $\xi = 0.01$ in Lizotte (2008) and $\epsilon = 1.96$, resulting in that the UCB corresponds to a 95% CI.

We note that the three acquisition functions have quite different behaviours in the three situations. In the first situation (upper), the PI and the UCB have two peaks that are located in the left and right part of the region to exploit the current information and to explore the region better, respectively. The EI would like to sample the left end of the region to exploit the current information and to reduce the uncertainty in that area.

In the second situation (middle), the EI again would like to exploit the current information by placing a sample near the peak of the predictive mean. The other two acquisition functions would like to explore the right part of the region better.

Figure 4.8: The GP regression problem in Example 4.6 using $N = 3$ (upper), $N = 5$ (middle) and $N = 10$ (lower) data points with the Matérn 5/2 kernel. In the left part, we present the mean of the predictive distributions (solid lines) and the corresponding 95% CI (gray area) together with the true function (dashed line) and the noisy observations (black dots). In the right part, we present normalised values of three different acquisition functions for each situation.

Finally, in the third situation (lower) the three functions agree and would like to exploit the current peak in the predictive mean. Hence, we conclude that the three acquisition functions have rather different behaviour and we return in a following example to investigate how this affects the resulting GPO algorithm.

4.4.3 Gaussian process optimisation

The full procedure for using GPO to estimate the solution to (4.9) is presented in Algorithm 6. As previously discussed, the user choices include the kernel for the GP prior and the acquisition function. Again, we remind the reader that the choice of kernel is crucial for the performance of the algorithm. Furthermore, an optimisation method is needed to optimise the acquisition function in (4.16). Here, we make use of a global derivative-free optimisation algorithm called DIRECT (Jones et al., 1993), but we discuss other possible choices in Paper B.

The GPO algorithm is often initialised by some randomly selected parameters sampled from the parameter prior. These samples are used to estimate the hyperparameters of the GP kernels so that the AQ function can operate. The number of these samples varies with the dimension of the problem but between 5 and 50 are reasonable numbers for small problems. Also, as the EB procedure is computationally costly, it is beneficial to estimate the GP hyperparameters every 5th or 10th iteration to save computations.

We end this section by discussing three different examples of where GPO and/or surrogate function modelling is useful. In Examples 4.8 and 4.9, we use the GPO algorithm for solving the ML parameter inference problem in an SSM and compare the different possible choices of AQs. Note that more examples of this application are found in Papers B and C.

Finally in Example 4.10, we illustrate the use of GPO for creating an input that maximises the logarithm of determinant of the expected information in an SSM. Remember that this corresponds to maximising the accuracy of the ML parameter estimate, which is the objective in input design.

Algorithm 6 Gaussian process optimisation (GPO)

INPUTS: $\kappa(\cdot, \cdot)$ (GP kernel), AQ (acquisition func.), K (no. iterations) and x_1 (init. value).
 OUTPUT: $\hat{x}_{\max}$ (estimate of the maximising argument of (4.9)).

- 1: Initialise the algorithm by random sampling.
 - 2: **for** $k = 1$ to K **do**
 - 3: Sample $f(x_k)$ to obtain the noisy estimate y_k .
 - 4: Compute (4.13) to obtain $\mu_f(x|\mathcal{D}_k)$.
 - 5: Compute (4.17) to obtain $\mu_{\max}$.
 - 6: Compute (4.16) to obtain x_{k+1} .
 - 7: **end for**
 - 8: Compute the maximiser of $\mu_f(x|\mathcal{D}_K)$ to obtain the estimate $\hat{x}_{\max}$ of (4.9).
-

4.8 Example: GPO for ML inference in the GARCH(1,1) model

Consider the ML parameter inference problem in the GARCH(1,1) model (2.3) using synthetic data with $\theta = \tau$ and $\Theta = (0, 0.25) \in \mathbb{R}$. Here, we make use of the GPO algorithm for solving this problem, by first rewriting (4.10) as

$$\hat{\ell}(\theta_k) = \ell(\theta_k) + z_k, \quad z_k \sim \mathcal{N}(0, \sigma_z^2),$$

which is similar to the CLT established in Section 3.3.4. As a consequence, (4.9) turns into the maximum likelihood maximisation problem discussed in Section 2.3. The resulting procedure follows from Algorithm 6 by plugging in the APF from Algorithm 2 for log-likelihood estimation.

Here, we use an one dimensional parameter vector to be able to compare the three different AQS discussed in the previous section. We generate $T = 250$ observations from the model using $\{\alpha, \beta, \gamma, \tau\} = \{0.1, 0.8, 0.05, 0.3\}$. We make use of the faPF with $N = 100$ to estimate the log-likelihood during each iteration. Here, we use the recommended value of $\xi = 0.01$ in Lizotte (2008) and $\epsilon = 1.96$ for the acquisition rule.

In Figure 4.9, we present the procedure at five consecutive iterations using three different acquisition rules. The procedure is initialised with two randomly selected samples of the log-likelihood, which are not shown in the figure. Here, we see that the behaviour of the three acquisition rules are rather different but the resulting mode of the log-likelihood is almost the same. The parameter estimate is obtained around $\hat{\theta}_{\text{ML}} = 0.12$ for all three choices of the acquisition rule. Note, that this small toy example is probably not complex enough to show the real differences between the acquisition rules. Remember, that more extensive evaluations by Lizotte (2008) show that the EI rule is often a good choice in many different applications.

Figure 4.9: Five steps of the GPO algorithm for ML parameter inference in Example 4.8 using three different acquisition rules: PI (left), EI (center), UCB (right). The predictive mean and the resulting value of the acquisition rule are presented with coloured solid and dotted lines, respectively. The black dots and gray areas indicate the samples obtained from the log-likelihood and the 95% predictive CI, respectively.

4.9 Example: GPO for ML inference in the earthquake count model

We return to the setting considered in Example 4.9 for ML parameter inference in the earthquake count model (2.6) using the real-world data discussed in Section 2.2.3. Here, the parameter vector is given by $\theta = \{\phi, \sigma_v, \beta\}$ and the parameter space is $\Theta = (0, 1) \times (0, 1) \times (10, 20) \in \mathbb{R}^3$. We make use of the bPF with $N = 1000$ particles to estimate the log-likelihood. The procedure is initialised with 50 samples from the log-likelihood at randomly selected parameters and continues with 150 iterations of the GPO algorithm.

In Figure 4.10, we present the current ML parameter estimate at each iteration of the GPO algorithm. We see that the parameter estimates and the predicted log-likelihood stabilise after about 150 iterations. This is rather fast compared with e.g. SPSSA which in Paper B requires an order of magnitude more samples of the log-likelihood. The final parameter estimate is obtained as $\hat{\theta}_{\text{ML}} = \{0.88, 0.15, 17.65\}$. This shows that the underlying intensity is rather slowly varying and that the mean number of major earthquakes each year is 17.65.

4.10 Example: Input design in the LGSS model using GPO

Consider the LGSS model (2.2) with the parameters $\theta^* = \{0.5, 1, 0.1, 0.1\}$, $T = 250$ and an input $u_{1:T}$ generated by

$$u_t = \begin{cases} -1, & \text{with probability } (1 - \alpha_1)(1 - \alpha_2), \\ 0, & \text{with probability } \alpha_1, \\ 1, & \text{with probability } (1 - \alpha_1)\alpha_2, \end{cases}$$

for $t = 1, \dots, T$. Here, the aim is to find the input parameters $\alpha^* = \{\alpha_1^*, \alpha_2^*\}$ such that

$$\alpha^* = \underset{\alpha \in [0,1] \times [0,1]}{\operatorname{argmax}} \log \det \left(\hat{\mathcal{I}}(\theta^*, u_{1:T}(\alpha)) \right),$$

where $\hat{\mathcal{I}}(\theta^*, u_{1:T}(\alpha))$ denotes the estimated expected information matrix using the input $u_{1:T}(\alpha)$ generated using the input parameters α . We make use of the second formulation in (2.13) to estimate the expected information matrix. This is done by first estimating the score function with the FL smoother in Algorithm 3 for $M = 100$ different data realisations from the model using the same input $u_{1:T}$ and the faPF with $N = 100$. Finally, the expected information matrix is estimated using the sample covariance matrix of the score function estimates.

We integrate this problem into the GPO procedure outlined in Algorithm 6, where the estimation of the expected information matrix is Step (i). Furthermore, we make use of the EI as the acquisition rule and initialise the procedure with 10 uniform random samples for α . In Figure 4.11, we present the resulting predictive mean for the input parameters and the sampling points for the algorithm. The input parameter estimates are obtained as $\hat{\alpha} = \{0.26, 0.50\}$. We see that the algorithm converges quickly to the estimates, which this shows that the GPO algorithm can be a useful alternative in input design. This as, it requires a limited number of computationally costly estimates of the expected information matrix.

Figure 4.10: The ML parameter estimate and the resulting predicted log-likelihood at each iteration of the GPO algorithm in Example 4.9.

Figure 4.11: Upper: the estimated mean of the expected information matrix as a function of the input parameters. Middle: the sampling points of the GPO algorithm. Lower: a realisation of the estimated optimal input.

5

Concluding remarks and future work

In this chapter, we give a summary of the contributions in the papers included in this thesis and discuss some avenues for future work.

5.1 Summary of the contributions

Broadly speaking, the main contributions of this thesis are mainly within two areas. The first contribution is to *develop new methods and improve existing methods for efficient parameter inference in SSMs*. Here, we are concerned with computational efficiency, which means that we would like to reduce the number of particles or iterations needed to reach a certain accuracy in the parameter estimates. This contribution is contained within Papers A-C. The second contribution is to make use of SMC and MCMC to *extend existing methods for parameter inference and input design to nonlinear problems*. This contribution is discussed in Papers D and E.

In Paper A, we develop a novel PMCMC algorithm that combines the PMH algorithm from Section 4.3 with the Langevin dynamics discussed in Section 4.2.2. The key idea here is to include the particle system within the proposal of the PMH algorithm. With this information, we can make use of the FL smoother from Section 3.4.2 to estimate the score function and the observed information matrix. These quantities are then used to construct the PMH1 and PMH2 algorithms in Paper A in analogue with the MH1 and MH2 algorithms discussed in Section 4.2.2. The resulting algorithm is efficient as it explores the posterior distribution better and this results in a higher ESS compared with the PMH0 algorithm. Furthermore, the added information makes the algorithm invariant to affine transformations of the parameter vector and reduces the length of the the

burn-in. As a consequence, the proposed algorithm require less iterations than the PMH0 algorithm in some settings, which makes it more computationally efficient as the computational complexity of the two algorithm are the same.

In Paper B, we develop a novel algorithm for ML parameter inference by combining ideas from GPO in Section 4.4 with log-likelihood estimation using the APF from Section 3.3.4. The resulting algorithm is computationally efficient as it requires less samples from the log-likelihood compared with some other optimisation methods. As these estimates are computationally costly to obtain this results in an overall decreased computational cost. Compared with SPSA, the gain is about one order of magnitude, see Paper B for a comparison on the HWSV model. In Paper C, we extend the combination of GPO and SMC to parameter inference in nonlinear SSMs with intractable likelihoods. Computationally costly ABC methods are used to approximate the intractable likelihood. Therefore, there could be substantial gains in using this algorithm for inference in this type of models.

In Paper D, we develop a novel algorithm for input design in nonlinear SSMs, which can handle amplitude constraints on the input. The proposed method combines results from Valenzuela et al. (2013) with SMC from Section 3.4.2 for estimating the expected information matrix. In Paper E, we propose two algorithms for parameter inference in ARX models with Student- t innovations which includes automatic model order selection by two different methods. These methods makes use of the MH algorithm discussed in Section 4.2 with reversible jump and the Gibbs sampler together with sparseness priors.

5.2 Outlook and future work

In this section, we summarise some ideas for future work and extensions of the contributions presented within this thesis. We discuss three different areas; the PMH-algorithm, the GPO-SMC algorithm and input design in SSMs.

5.2.1 Particle Metropolis-Hastings

The proposed contributions to the PMH algorithm are mainly methodological developments of existing methods. Therefore, it would be interesting to examine the theoretical properties of the PMH1 and the PMH2 algorithms. This includes questions regarding the convergence rate of the algorithm and how its properties scale with the dimension of the parameter space. Similar analysis has previously been done for MH0 (Roberts et al., 1997), MH1 (Roberts and Rosenthal, 1998) and PMH0 (Sherlock et al., 2013). A possible first step for the PMH analysis is to consider the situation where the number of observations is large. By the discussion in Section 2.4, we know that the Bayesian CLT would give a roughly Gaussian posterior, which simplifies the analysis.

Further methodological developments could also be interesting in the PMCMC framework. This includes the development of adaptive PMH1 and PMH2, which automatically determines suitable step sizes and the number of particles. It could

also be possible to relax the reversibility constraint of the Markov chain during the burn-in phase of the algorithm. This could decrease the hitting time of the posterior mode by the Markov chain. The adaptive mechanism could then decide when the chain has reached the mode and then turn on the reversibility condition, so that the chain admits the target as its stationary distribution. Relevant work for this idea is found in Andrieu and Thoms (2008), Peters et al. (2010) and Pitt et al. (2012). Another approach to reduce the length of the burn-in is to make use of GPO to estimate the location of the posterior mode in a pilot run. This is similar to the work by Owen et al. (2014), where the authors make use of some pilot runs of the ABC algorithm to initialise the PMH0 algorithm in SSMs with intractable likelihoods.

Also, it would be interesting to develop a particle HMC algorithm as suggested in the discussions (Doucet et al., 2011) following Girolami and Calderhead (2011). The challenge with this idea is how to handle that multiple APFs are run within each PMH iteration, which might require some additional developments to the PMCMC framework. Better particle smoothers could also be useful as they could improve the estimates of the score function and the information matrix. This results in that larger step lengths can be used in the PMH2 algorithm and could lead to even larger increases in mixing of the Markov chain.

Online methods for Bayesian inference would also be of great interest, especially for the many big data problems that are likely to be faced in the future. Also, graphical processing units (GPUs) and other multicore architectures can be used to decrease the computational cost as some parts of the SMC and MCMC algorithms are possible to run in parallel. Interested readers are referred to Beam et al. (2014), Neiswanger et al. (2013), Henriksen et al. (2012) and Murray (2012) for more information.

5.2.2 Gaussian process optimisation using the particle filter

As we have demonstrated in this thesis, the performance of the GPO algorithm depends on the kernel function in the GP prior and the choice of acquisition function. Therefore it would be interesting to develop new acquisition functions that could make use of ideas from sparse GPs, Newton methods and/or proximal point algorithms (Rasmussen and Williams, 2006; Nocedal and Wright, 2006). The main challenge is to construct a rule that keeps exploring the objective function, but still keeps the fast convergence that we have illustrated in the examples in Section 4.4.3 and in Papers B and C.

Another possible improvement to the algorithm is to remove the bias in the log-likelihood estimate. This could be done by the bias compensation discussed in Example 3.4. Also, it would be interesting to develop online methods for this algorithm, perhaps by using some kind of stochastic approximation scheme. Finally, we think that there are many interesting applications of GP models for estimating the score and information matrix for an SSM. As these are computationally costly to evaluate with good accuracy, perhaps the ideas from *probabilistic numerics* could be helpful. This is an emerging field in machine learning, where GPs also

are used to estimate derivatives and integrals as well as to solve ordinary differential equations. For more information, see Hennig (2013), Osborne et al. (2012), Osborne (2010) and Boyle (2007).

5.2.3 Input design in SSMs

The main drawback of the input design method proposed in Paper D is that the expected information matrix is computationally costly to evaluate. It is possible to make a perfect parallel implementation of this method to reduce the computational cost. However, it would be even more interesting to develop the idea from Example 4.10 and make use of GPO for input design. This could be useful when considering that GPO does not require many evaluations of the objective function, which corresponds to estimates of the expected information matrix.

Also, it would be interesting to consider methods that infer the parameters of the model at the same time as the optimal input. This would relax the unrealistic assumption that we need to know the true parameters to be able to construct an optimal input. There are mainly three approaches for solving this problem. The first is to pose the problem in a robust optimisation setting and only assume that the true parameter is located within some set. By this construction, the resulting optimal input would be an average (in some sense) over this set.

The second approach is to construct a sequential algorithm that first infer the parameters and then the optimal input. This is repeated over many iterations by exciting the system with the input constructed in the last iteration. However, it is difficult to prove that this approach would converge to the true optimal input and the true parameters. Finally, we could pose this problem in a Bayesian manner and marginalise over the parameters of the model. The resulting optimal input would therefore be a marginalisation over the parameter posterior. Therefore, we would not need to know the true parameters of the system. See Müller et al. (2007), Kuck et al. (2006) and Bubeck and Cesa-Bianchi (2012) for more information.

5.3 Source code and data

Source code written in Python and R for recreating most of the examples in Part I are available from the author's homepages at: <http://users.isy.liu.se/en/rt/johda87/> and <http://code.johandahlin.com/>. Furthermore, source code for recreating some of the numerical illustrations from Papers A, B, C and E are also available from the same homepages. The source code and data are provided under the MIT license with no guaranteed support and no responsibility for its use and function.

Bibliography

- M. Adolfson, S. Laséen, J. Lindé, and M. Villani. RAMSES – a new general equilibrium model for monetary policy analysis. *Sveriges Riksbank Economic Review*, 2, 2007a.
- M. Adolfson, S. Laséen, J. Lindé, and M. Villani. Bayesian estimation of an open economy DSGE model with incomplete pass-through. *Journal of International Economics*, 72(2):481–511, 2007b.
- M. Adolfson, S. Laséen, L. Christiano, M. Trabandt, and K. Walentin. RAMSES II – Model Description. *Occasional Paper Series*, 12, 2013.
- G. Amisano and O. Tristani. Euro area inflation persistence in an estimated nonlinear DSGE model. *Journal of Economic Dynamics and Control*, 34(10):1837–1858, 2010.
- S. An and F. Schorfheide. Bayesian analysis of DSGE models. *Econometric reviews*, 26(2-4):113–172, 2007.
- B. D. O. Anderson and J. B. Moore. *Optimal filtering*. Courier Publications, 2005.
- C. Andrieu and G. O. Roberts. The pseudo-marginal approach for efficient Monte Carlo computations. *The Annals of Statistics*, 37(2):697–725, 2009.
- C. Andrieu and J. Thoms. A tutorial on adaptive MCMC. *Statistics and Computing*, 18(4):343–373, 2008.
- C. Andrieu, A. Doucet, and R. Holenstein. Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342, 2010.
- A. L. Beam, S. K. Ghosh, and J. Doyle. Fast Hamiltonian Monte Carlo Using GPU Computing. *Pre-print*, 2014. arXiv:1402.4089v1.
- M. A. Beaumont. Estimation of population growth or decline in genetically monitored populations. *Genetics*, 164(3):1139–1160, 2003.
- J. O. Berger. *Statistical decision theory and Bayesian analysis*. Springer, 1985.

- M. J. Betancourt and M. Girolami. Hamiltonian Monte Carlo for Hierarchical Models. *Pre-print*, 2013. arXiv:1312.0906v1.
- C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, USA, 2006.
- T. Björk. *Arbitrage theory in continuous time*. Oxford University Press, 2004.
- F. Black and M. Scholes. The pricing of options and corporate liabilities. *The Journal of Political Economy*, pages 637–654, 1973.
- T. Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, 31(3):307–327, 1986.
- P. Boyle. *Gaussian processes for regression and optimisation*. PhD thesis, Victoria University of Wellington, 2007.
- M. Briers, A. Doucet, and S. Maskell. Smoothing algorithms for state-space models. *Annals of the Institute of Statistical Mathematics*, 62(1):61–89, 2010.
- E. Brochu, V. M. Cora, and N. De Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *Pre-print*, 2010. arXiv:1012.2599v1.
- P. J. Brockwell and R. A. Davis. *Introduction to time series and forecasting*. Springer, 2002.
- S. Bubeck and N. Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning*, 5(1):1–122, 2012.
- P. Bunch and S. Godsill. Improved particle approximations to the joint smoothing distribution using Markov Chain Monte Carlo. *IEEE Transactions on Signal Processing*, 61(4):956–963, 2013.
- D. Burke, A. Ghosh, and W. Heidrich. Bidirectional importance sampling for direct illumination. In *Proceedings of the 16th Eurographics Symposium on Rendering Techniques*, pages 147–156, Konstanz, Germany, June 2005.
- O. Cappé, E. Moulines, and T. Rydén. *Inference in Hidden Markov Models*. Springer, 2005.
- O. Cappé, S. J. Godsill, and E. Moulines. An overview of existing methods and recent advances in sequential Monte Carlo. *Proceedings of the IEEE*, 95(5): 899–924, 2007.
- C. M. Carvalho, M. S. Johannes, H. F. Lopes, and N. G. Polson. Particle learning and smoothing. *Statistical Science*, 25(1):88–106, 2010.
- R. Casarin. Bayesian inference for generalised Markov switching stochastic volatility models, 2004. CEREMADE Journal Working Paper 0414.
- G. Casella and R. L. Berger. *Statistical Inference*. Duxbury Press, 2 edition, 2001.

- K. S. Chan and J. Ledolter. Monte Carlo EM estimation for time series models involving counts. *Journal of the American Statistical Association*, 90(429):242–252, 1995.
- T. Chen, E. B. Fox, and C. Guestrin. Stochastic Gradient Hamiltonian Monte Carlo. *Pre-print*, 2014. arXiv:1402.4102v1.
- S. Chib, F. Nardari, and N. Shephard. Markov chain Monte Carlo methods for stochastic volatility models. *Journal of Econometrics*, 108(2):281–316, 2002.
- N. Chopin, P. E. Jacob, and O. Papaspiliopoulos. SMC²: an efficient algorithm for sequential analysis of state space models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 75(3):397–426, 2013.
- R. Cont. Empirical properties of asset returns: stylized facts and statistical issues. *Quantitative Finance*, 1:223–236, 2001.
- J-M. Cornuet, J-M. Marin, A. Mira, and C. P. Robert. Adaptive Multiple Importance Sampling. *Pre-print*, 2011. arXiv:0907.1254v5.
- N. Cressie. *Statistics for spatial data*. Wiley, 1993.
- D. Crisan and A. Doucet. A survey of convergence results on particle filtering methods for practitioners. *IEEE Transactions on Signal Processing*, 50(3):736–746, 2002.
- J. Dahlin and F. Lindsten. Particle filter-based Gaussian process optimisation for parameter inference. In *Proceedings of the 19th IFAC World Congress*, Cape Town, South Africa, August 2014. (accepted for publication).
- J. Dahlin and P. Svenson. A Method for Community Detection in Uncertain Networks. In *Proceedings of 2011 European Intelligence and Security Informatics Conference*, Athens, Greece, August 2011.
- J. Dahlin and P. Svenson. Ensemble approaches for improving community detection methods. *Pre-print*, 2013. arXiv:1309.0242v1.
- J. Dahlin, F. Johansson, L. Kaati, C. Mårtensson, and P. Svenson. A Method for Community Detection in Uncertain Networks. In *Proceedings of International Symposium on Foundation of Open Source Intelligence and Security Informatics 2012*, Istanbul, Turkey, August 2012a.
- J. Dahlin, F. Lindsten, T. B. Schön, and A. Wills. Hierarchical Bayesian ARX models for robust inference. In *Proceedings of the 16th IFAC Symposium on System Identification (SYSID)*, Brussels, Belgium, July 2012b.
- J. Dahlin, F. Lindsten, and T. B. Schön. Particle Metropolis Hastings using Langevin dynamics. In *Proceedings of the 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vancouver, Canada, May 2013a.
- J. Dahlin, F. Lindsten, and T. B. Schön. Inference in Gaussian models with missing data using Equalisation Maximisation. *Pre-print*, 2013b. arXiv:1308.4601v1.

- J. Dahlin, F. Lindsten, and T. B. Schön. Second-order particle MCMC for Bayesian parameter inference. In *Proceedings of the 19th IFAC World Congress*, Cape Town, South Africa, August 2014a. (accepted for publication).
- J. Dahlin, F. Lindsten, and T. B. Schön. Particle Metropolis-Hastings using gradient and Hessian information. *Pre-print*, 2014b. arXiv:1311.0686v2.
- J. Dahlin, T. B. Schön, and M. Villani. Approximate inference in state space models with intractable likelihoods using Gaussian process optimisation. Technical Report LiTH-ISY-R-3075, Department of Electrical Engineering, Linköping University, Linköping, Sweden, April 2014c.
- P. Debevec. Rendering synthetic objects into real scenes: bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 189–198, Orlando, FL, USA, jul 1998. ACM.
- P. Del Moral. *Feynman-Kac Formulae - Genealogical and Interacting Particle Systems with Applications*. Probability and its Applications. Springer, 2004.
- P. Del Moral. *Mean field simulation for Monte Carlo integration*. CRC Press, 2013.
- P. Del Moral, A. Doucet, and A. Jasra. Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(3):411–436, 2006.
- P. Del Moral, A. Doucet, and S. Singh. Forward smoothing using sequential Monte Carlo. *Pre-print*, 2010. arXiv:1012.5390v1.
- M. Del Negro and F. Schorfheide. Priors from General Equilibrium Models for VARS. *International Economic Review*, 45(2):643–673, 2004.
- A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1): 1–38, 1977.
- R. Douc and O. Cappé. Comparison of resampling schemes for particle filtering. In *Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis (ISPA)*, pages 64–69, Zagreb, Croatia, September 2005.
- R. Douc, E. Moulines, and D. S Stoffer. *Nonlinear Time Series: theory, methods and applications with R examples*. CRC Press, 2014.
- A. Doucet and A. Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. In D. Crisan and B. Rozovsky, editors, *The Oxford Handbook of Nonlinear Filtering*. Oxford University Press, 2011.
- A. Doucet, S. Godsill, and C. Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and computing*, 10(3):197–208, 2000.
- A. Doucet, P. Jacob, and A. M. Johansen. Discussion on Riemann manifold

- Langevin and Hamiltonian Monte Carlo methods. *Journal of the Royal Statistical Society: Series B Statistical Methodology*, 73(2), p 162, 2011.
- A. Doucet, M. K. Pitt, and R. Kohn. Efficient implementation of Markov chain Monte Carlo when using an unbiased likelihood estimator. *arXiv.org*, arXiv:1210.1871, October 2012.
- S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth. Hybrid Monte Carlo. *Physics letters B*, 195(2):216–222, 1987.
- C. Dubarry and R. Douc. Particle approximation improvement of the joint smoothing distribution with on-the-fly variance estimation. *Pre-print*, 2011. arXiv:1107.55241.
- E. Ehrlich, A. Jasra, and N. Kantas. Static Parameter Estimation for ABC Approximations of Hidden Markov Models. *Pre-print*, 2012. arXiv:1210.4683v1.
- R. F. Engle. Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. *Econometrica: Journal of the Econometric Society*, pages 987–1007, 1982.
- P. Fearnhead, D. Wyncoll, and J. Tawn. A sequential smoothing algorithm with linear computational cost. *Biometrika*, 97(2):447–464, 2010.
- J. Fernández-Villaverde and J. F. Rubio-Ramírez. Estimating macroeconomic models: A likelihood approach. *The Review of Economic Studies*, 74(4):1059–1087, 2007.
- R. A. Fisher. Theory of statistical estimation. *Mathematical Proceedings of the Cambridge Philosophical Society*, 22(05):700–725, 1925.
- T. Flury and N. Shephard. Bayesian inference based only on simulated likelihood: particle filter analysis of dynamic economic models. *Econometric Theory*, 27(5): 933–956, 2011.
- K. Fokianos, A. Rahbek, and D. Tjøstheim. Poisson autoregression. *Journal of the American Statistical Association*, 104(488):1430–1439, 2009.
- A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin. *Bayesian data analysis*. Chapman & Hall/CRC, 3 edition, 2013.
- S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.
- A. Ghosh, A. Doucet, and W. Heidrich. Sequential sampling for dynamic environment map illumination. In *Proceedings of the 17th Eurographics conference on Rendering Techniques*, pages 115–126, Nicosia, Cyprus, June 2006.
- P. Giordani and R. Kohn. Adaptive independent Metropolis-Hastings by fast estimation of mixtures of normals. *Journal of Computational and Graphical Statistics*, 19(2):243–259, 2010.

- M. Girolami and B. Calderhead. Riemann manifold Langevin and Hamiltonian Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(2):1–37, 2011.
- P. Glasserman. *Monte Carlo methods in financial engineering*, volume 53. Springer, 2004.
- S. J. Godsill, A. Doucet, and M. West. Monte Carlo smoothing for nonlinear time series. *Journal of the American Statistical Association*, 99(465):156–168, March 2004.
- N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEEE Proceedings of Radar and Signal Processing*, 140(2):107–113, 1993.
- A. G. Gray. *Bringing tractability to generalized n-body problems in statistical and scientific computation*. PhD thesis, Carnegie Mellon University, 2003.
- W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.
- E. Hecht. *Optics*. Pearson, 4 edition, 2013.
- P. Hennig. Fast probabilistic optimization from noisy gradients. In *Proceedings of the 30th International Conference on Machine Learning*, Atlanta, GA, USA, jun 2013.
- S. Henriksen, A. Wills, T. B. Schön, and B. Ninness. Parallel implementation of particle MCMC methods on a GPU. In *Proceedings of the 16th IFAC Symposium on System Identification (SYSID)*, Brussels, Belgium, July 2012.
- J. D. Hol, T. B. Schön, and F. Gustafsson. On resampling algorithms for particle filters. In *Proceedings of the Nonlinear Statistical Signal Processing Workshop*, Cambridge, UK, September 2006.
- J. Hull. *Options, Futures, and other Derivatives*. Pearson, 7 edition, 2009.
- J. Hull and A. White. The pricing of options on assets with stochastic volatilities. *The Journal of Finance*, 42(2):281–300, 1987.
- D. Hultqvist, J. Roll, F. Svensson, J. Dahlin, and T. B. Schön. Detection and positioning of overtaking vehicles using 1D optical flow. In *Proceedings of the IEEE Intelligent Vehicles (IV) Symposium*, Dearborn, MI, USA, June 2014. (accepted for publication).
- E. Jacquier, N. G. Polson, and P. E. Rossi. Bayesian analysis of stochastic volatility models with fat-tails and correlated errors. *Journal of Econometrics*, 122(1):185–212, 2004.
- A. H. Jazwinski. *Stochastic processes and filtering theory*, volume 63. Academic press, 1970.

- M. Johannes and N. Polson. MCMC Methods for Continuous-time Financial Econometrics. In Y. Ait-Sahalia and L. Hansen, editors, *Handbook of Financial Econometrics, Vol. 1: Tools and Techniques*, volume 2, pages 1–72. North-Holland, 2009.
- D. R. Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization*, 21(4):345–383, 2001.
- D. R. Jones, C. D. Perttunen, and B. E. Stuckman. Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181, 1993.
- D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492, 1998.
- T. Kailath, A. H. Sayed, and B. Hassibi. *Linear Estimation*. Prentice Hall, Upper Saddle River, NJ, USA, 2000.
- N. Kantas, A. Doucet, S.S. Singh, and J.M. Maciejowski. An overview of sequential monte carlo methods for parameter estimation in general state-space models. In *IFAC Symposium on System Identification (SYSID)*, Saint-Malo, France, July 2009.
- M. J. Keeling and P. Rohani. *Modeling infectious diseases in humans and animals*. Princeton University Press, 2008.
- S. Kim, N. Shephard, and S. Chib. Stochastic volatility: likelihood inference and comparison with ARCH models. *The Review of Economic Studies*, 65(3): 361–393, 1998.
- G. Kitagawa and S. Sato. Monte Carlo smoothing and self-organising state-space model. In A. Doucet, N. de Freitas, and N. Gordon, editors, *Sequential Monte Carlo methods in practice*, pages 177–195. Springer, 2001.
- M. Klaas, M. Briers, N. de Freitas, A. Doucet, S. Maskell, and D. Lang. Fast particle smoothing: if I had a million particles. In *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, USA, June 2006.
- P. E. Kloeden and E. Platen. *Numerical solution of stochastic differential equations*, volume 23. Springer, 4 edition, 1992.
- J. Kronander and T. B. Schön. Robust auxiliary particle filters using multiple importance sampling. In *Proceedings of the 2014 IEEE Statistical Signal Processing Workshop (SSP)*, Gold Coast, Australia, July 2014. (accepted for publication).
- J. Kronander, J. Dahlin, D. Jönsson, M. Kok, T. B. Schön, and J. Unger. Real-time Video Based Lighting Using GPU Raytracing. In *Proceedings of the 2014 European Signal Processing Conference (EUSIPCO)*, Lisbon, Portugal, September 2014a. (submitted, pending review).

- J. Kronander, T. B. Schön, and J. Dahlin. Backward sequential Monte Carlo for marginal smoothing. In *Proceedings of the 2014 IEEE Statistical Signal Processing Workshop (SSP)*, Gold Coast, Australia, July 2014b. (accepted for publication).
- H. Kuck, N. de Freitas, and A. Doucet. SMC samplers for Bayesian optimal nonlinear design. In *Proceedings of the 2006 Nonlinear Statistical Signal Processing Workshop*, pages 99–102, Cambridge, UK, sep 2006.
- H. J. Kushner. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *Journal of Basic Engineering*, 86(1): 97–106, 1964.
- R. Langrock. Some applications of nonlinear and non-Gaussian state-space modelling by means of hidden Markov models. *Journal of Applied Statistics*, 38(12): 2955–2970, 2011.
- R. Langrock and W. Zucchini. Hidden Markov models with arbitrary state dwell-time distributions. *Computational Statistics & Data Analysis*, 55(1):715–724, 2011.
- E. L. Lehmann and G. Casella. *Theory of point estimation*. Springer, 1998.
- F. Lindsten. An efficient stochastic approximation EM algorithm using conditional particle filters. In *Proceedings of the 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vancouver, Canada, May 2013.
- F. Lindsten and T. B. Schön. Backward simulation methods for Monte Carlo statistical inference. In *Foundations and Trends in Machine Learning*, volume 6, pages 1–143, August 2013.
- J. S. Liu. *Monte Carlo Strategies in Scientific Computing*. Springer, 2008.
- S. Livingstone and M. Girolami. Information-geometric Markov Chain Monte Carlo methods using Diffusions. *Pre-print*, 2014. arXiv:1403.7957v1.
- D. J. Lizotte. *Practical Bayesian optimization*. PhD thesis, University of Alberta, 2008.
- L. Ljung. *System identification: theory for the user*. Prentice Hall, 1999.
- T. A. Louis. Finding the observed information matrix when using the EM algorithm. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 44(02):226–233, 1982.
- A. Marshall. The use of multi-stage sampling schemes in Monte Carlo simulations. In M. Meyer, editor, *Symposium on Monte Carlo Methods*, pages 123–140. Wiley, 1956.
- G. Matheron. Principles of geostatistics. *Economic Geology*, 58(8):1246–1266, 1963.

- G. J. McLachlan and T. Krishnan. *The EM algorithm and extensions*. Wiley-Interscience, second edition, 2008.
- N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.
- S. P. Meyn and R. L. Tweedie. *Markov chains and stochastic stability*. Cambridge University Press, 2009.
- T. P. Minka. Expectation propagation for approximate Bayesian inference. In *Proceedings of the 17th conference on Uncertainty in Artificial Intelligence*, pages 362–369, Seattle, WA, USA, aug 2001.
- S. Mitra. A review of volatility and option pricing. *International Journal of Financial Markets and Derivatives*, 2(3):149–179, 2011.
- J. Mockus, V. Tiesis, and A. Zilinskas. The application of Bayesian methods for seeking the extremum. In L. C. W. Dixon and G. P. Szego, editors, *Toward Global Optimization*, pages 117–129. North-Holland, 1978.
- P. Müller, D. A. Berry, A. P. Grieve, M. Smith, and M. Krams. Simulation-based sequential Bayesian design. *Journal of Statistical Planning and Inference*, 137(10):3140–3150, 2007.
- K. P. Murphy. *Machine learning: a probabilistic perspective*. The MIT Press, 2012.
- L. Murray. GPU acceleration of the particle filter: The Metropolis resampler. *Pre-print*, 2012. arXiv:1202.6163v1.
- C. A. Naesseth. Nowcasting using Microblog Data. Bachelor thesis, Linköping University, sep 2012. LiTH-ISY-EX-ET-12/0398.
- R. M. Neal. MCMC using Hamiltonian dynamics. In S. Brooks, A. Gelman, G. Jones, and X-L. Meng, editors, *Handbook of Markov Chain Monte Carlo*. Chapman & Hall/ CRC Press, June 2010.
- W. Neiswanger, C. Wang, and E Xing. Asymptotically Exact, Embarrassingly Parallel MCMC. *Pre-print*, 2013. arXiv:1311.4780v2.
- B. Ninness and S. Henriksen. Bayesian system identification via Markov chain Monte Carlo techniques. *Automatica*, 46(1):40–51, 2010.
- J. Nocedal and S. Wright. *Numerical Optimization*. Springer, 2 edition, 2006.
- J. Nolan. *Stable distributions: models for heavy-tailed data*. Birkhauser, 2003.
- B. Øksendal. *Stochastic differential equations*. Springer, 6 edition, 2010.
- J. Olsson, O. Cappé, R. Douc, and E. Moulines. Sequential Monte Carlo smoothing with application to parameter estimation in nonlinear state space models. *Bernoulli*, 14(1):155–179, 2008.

- M. Osborne. *Bayesian Gaussian Processes for Sequential Prediction, Optimisation and Quadrature*. PhD thesis, University of Oxford, 2010.
- M. A. Osborne, R. Garnett, S. J. Roberts, C. Hart, S. Aigrain, and N. Gibson. Bayesian quadrature for ratios. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 832–840, La Palma, Canary Islands, SP, apr 2012.
- J. Owen, D. J. Wilkinson, and C. S. Gillespie. Scalable Inference for Markov Processes with Intractable Likelihoods. *Pre-print*, 2014. arXiv:1403.6886v1.
- G. W. Peters, G. R. Hosack, and K. R. Hayes. Ecological non-linear state space model selection via adaptive particle Markov chain Monte Carlo. *Pre-print*, 2010. arXiv:1005.2238v1.
- M. Pharr and G. Humphreys. *Physically based rendering: From theory to implementation*. Morgan Kaufmann, 2010.
- M. K. Pitt. Smooth Particle Filters for Likelihood Evaluation and Maximisation. Technical Report 651, Department of economics, University of Warwick, Coventry, UK, July 2002. Warwick economic research papers.
- M. K. Pitt and N. Shephard. Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, 94(446):590–599, 1999.
- M. K. Pitt, R. S. Silva, P. Giordani, and R. Kohn. Auxiliary particle filtering within adaptive Metropolis-Hastings sampling. *Pre-print*, 2010. arXiv:1006.1914v1.
- M. K. Pitt, R. S. Silva, P. Giordani, and R. Kohn. On some properties of Markov chain Monte Carlo simulation methods based on the particle filter. *Journal of Econometrics*, 171(2):134–151, 2012.
- G. Poyiadjis, A. Doucet, and S. S. Singh. Particle approximations of the score and observed information matrix in state space models with application to parameter estimation. *Biometrika*, 98(1):65–80, 2011.
- C. R. Rao. *Linear Statistical Inference and Its Applications*. Wiley, 1965.
- C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.
- H. E. Rauch, F. Tung, and C. T. Striebel. Maximum likelihood estimates of linear dynamic systems. *AIAA Journal*, 3(8):1445–1450, August 1965.
- C. P. Robert. *The Bayesian choice*. Springer, 2007.
- C. P. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer, 2 edition, 2004.
- G. O. Roberts and J. S. Rosenthal. Optimal Scaling of Discrete Approximations to Langevin Diffusions. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 60(1):255–268, 1998.

- G. O. Roberts, A. Gelman, and W. R. Gilks. Weak convergence and optimal scaling of random walk Metropolis algorithms. *The Annals of Applied Probability*, 7(1):110–120, 1997.
- S. M. Ross. *Simulation*. Academic Press, 5 edition, 2012.
- H. Rue, S. Martino, and N. Chopin. Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(2):319–392, 2009.
- T. B. Schön, A. Wills, and B. Ninness. System identification of nonlinear state-space models. *Automatica*, 47(1):39–49, 2011.
- C. Sherlock, A. H. Thiery, G. O. Roberts, and J. S. Rosenthal. On the efficiency of pseudo-marginal random walk Metropolis algorithms. *Pre-print*, 2013. arXiv:1309.7209v1.
- R. H. Shumway and D. S. Stoffer. *Time series analysis and its applications*. Springer, 3 edition, 2010.
- J. Snoek, H. Larochelle, and R. P. Adams. Practical Bayesian Optimization of Machine Learning Algorithms. In *Advances in Neural Information Processing Systems 25 (NIPS 2012)*, pages 2951–2959. Curran Associates, Inc., 2012.
- J. C. Spall. A stochastic approximation technique for generating maximum likelihood parameter estimates. In *American Control Conference*, pages 1161–1167, Minneapolis, MN, USA, June 1987.
- R. Srikanthan and T. A. McMahon. Stochastic generation of annual, monthly and daily climate data: A review. *Hydrology and Earth System Sciences*, 5(4): 653–670, 1999.
- E. Taghavi, F. Lindsten, L. Svensson, and T. B. Schön. Adaptive stopping for fast particle smoothing. In *Proceedings of the 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vancouver, Canada, May 2013.
- L. Tierney. Markov chains for exploring posterior distributions. *The Annals of Statistics*, 22(4):1701–1728, 1994.
- R. S Tsay. *Analysis of financial time series*. John Wiley & Sons, 2 edition, 2005.
- J. Unger, J. Kronander, P. Larsson, S. Gustavson, J. Löw, and A. Ynnerman. Spatially varying image based lighting using HDR-video. *Computers & graphics*, 37(7):923–934, 2013.
- P. E. Valenzuela, C. R. Rojas, and H. Hjalmarsson. Optimal input design for dynamic systems: a graph theory approach. In *Proceedings of the IEEE Conference on Decision and Control (CDC)*, Florence, Italy, dec 2013.
- P. E. Valenzuela, J. Dahlin, C. R. Rojas, and T. B. Schön. A graph/particle-based method for experiment design in nonlinear systems. In *Proceedings of the 19th*

- IFAC World Congress*, Cape Town, South Africa, August 2014. (accepted for publication).
- E. Veach and L. J. Guibas. Optimally combining sampling techniques for monte carlo rendering. In *Proceedings of the 22nd Annual Conference on Computer Graphics*, pages 419–428, Los Angeles, CA, USA., aug 1995. ACM.
- E. Veach and L. J. Guibas. Metropolis light transport. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 65–76, 1997.
- D. J. Wilkinson. *Stochastic modelling for systems biology*. CRC press, 2 edition, 2011.
- D. A. Woolhiser. Modeling daily precipitation – progress and problems. In *Statistics in the Environmental and Earth Sciences 5*, pages 71–89. Halsted Press New York, 1992.
- S. Yildirim, S. S. Singh, T. Dean, and A. Jasra. Parameter Estimation in Hidden Markov Models with Intractable Likelihoods Using Sequential Monte Carlo. *Preprint*, 2013. arXiv:1311.4117v1.
- S. L. Zeger. A regression model for time series of counts. *Biometrika*, 75(4): 621–629, 1988.

Part II

Publications

Publications

The articles associated with this thesis have been removed for copyright reasons. For more details about these see:

<http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-106752>