Sequential Bayesian Learning for State Space Models

Patrick Aschermayr

Department of Statistics
London School of Economics and Political Science

This dissertation is submitted for the degree of
Doctor of Philosophy

London School of Economics and
Political Science

June 2023

I would like to dedicate this thesis to my wonderful wife, Kayla. From the bottom of my heart, thank you for your unparalleled support throughout all this time. Your very presence in my life has made me a better person, immer mit dir! This dissertation is also dedicated to my parents, Alfred and Silvia. Thank you for your constant encouragement and loving me unconditionally. Your good examples have taught me to be diligent and to strive toward my goals with conviction!

Acknowledgements

This thesis marks the end of my third chapter at university and the beginning of another. It also provides an opportunity to thank my supervisor, Dr. Konstantinos Kalogeropoulos, for his continuing guidance and valuable advice during my time at the London School of Economics and Political Science. I am deeply grateful to have engaged in a countless number of discussions over the last few years and to have been given the time to learn, make mistakes and improve my research studies. I also want to sincerely thank you for believing in me and introducing me to many other fantastic researchers and projects during the course of my PhD.

Furthermore, I would like to express my genuine appreciation to the whole LSE administrative staff, especially Penny Montague, for providing excellent support and guidance throughout my time in London.

I also want to thank my close friends, in particular Davide Marchini and Maximilian Schleritzko. Davide, thank you for always being there, trying out new pizza restaurants around London and biking all the way to Hyde Park to spend time with me. Max, thank you for all the chats and discussions over the years; I challenge you to find anyone with a longer chat history than ours.

I would like to express my sincere gratitude to my caring parents, Silvia and Alfred, for their constant encouragement and all those weekly Skype calls that kept us close over the years. Thank you so much for visiting us in London, getting to know our favorite places in the city and hosting us at your lovely home in Klaus an der Pyhrnbahn during the final months of my PhD journey.

Last but not least, I want to thank my extraordinary wife, Kayla. Thank you for your unparalleled support throughout all this time. I am so grateful that you moved to a new country and created an exciting (and sometimes stressful) life in London with me. Doing a PhD is never easy and, particularly during the COVID-19 lockdown periods, life could feel quite lonely. You made life so much brighter, I love you!

This work has been accomplished with the financial support of the Economic and Social Research Council (ESRC). I am very grateful for this opportunity; without it, I would not have been able to pursue this degree.

Declaration

Patrick Aschermayr
June 2023

Statement of Co-Authored Work

I confirm that Chapter 2 was jointly co-authored with Dr. Konstantinos Kalogeropoulos.

I confirm that Chapter 3 was jointly co-authored with Dr. Konstantinos Kalogeropoulos and Dr. Nikolaos Demiris.

I confirm that Chapter 4 was jointly co-authored with Dr. Konstantinos Kalogeropoulos, Prof. Alexandros Beskos and Dr. Aristidis Nikolopoulos.

Patrick Aschermayr
June 2023

Abstract

This thesis explores the topic of sequential inference on a variety of novel model classes. Chapter 2 focuses on a class of discrete State Space Models (SSM) known as Hidden Semi-Markov Model (HSMM), a versatile generalization of the famous Hidden Markov Model (HMM) in which the underlying stochastic process follows a semi-Markov chain. In a case study on the VIX Index, efficient batch as well as sequential Bayesian parameter estimation schemes are contributed and validated. We benchmark HSMMs against popular discrete SSM alternatives and show how by-products that arise during the estimation process can be used for model selection and clustering. Chapter 3 centers on a new class of Susceptible-Exposed-Infected-Recovered (SEIR) type models to analyze and detect regime switches in the SARS-CoV-2 pandemic. We propose an epidemic model with the transmission rate between susceptible and infected individuals being time varying and piecewise constant. At any point in time, this parameter is linked to a latent variable that follows a HSMM. We define this model in state space formulation and demonstrate the latent states can be efficiently estimated using the Particle MCMC (PMCMC) and Sequential Monte Carlo Squared (SMC2) machinery. Moreover, a case study is conducted on the reported infection and fatalities data in the United Kingdom, during which we benchmark models with varying observation distribution specifications and determine the number of latent regimes in the data. Chapter 4 addresses Stochastic Volatility (SV) models and employs a variety of carefully selected copulas to explore the dependency structure between stocks and their volatility. This new class of models can reconstruct stylised empirical behaviours that cannot be captured by standard symmetric Gaussian innovations. In a case study on the S&P 500 and the VIX index, we examine the marginal distributions and joint dependency structure of the error terms in our proposed model. Moreover, batch and sequential Bayesian model selection are applied to analyze the suitability of the separate copula choices against standard modelling techniques.

Contents

List of Figures

List of Tables

Acronyms

R_t Effective Reproductive Number. xviii, 38, 51, 54

APF Auxiliary Particle Filter. 17

AR Autoregressive. 7, 9, 13, 27

ARHMM Autoregressive Hidden Markov Model. xx, 28, 31, 101

ARHSMM Autoregressive Hidden Semi-Markov Model. xx, xxiii, 28, 29, 31, 101

BF Bayes Factor. 19, 22, 47, 67

BMI Basic Marginal Likelihood Identity. 15

BPF Bootstrap Particle Filter. 17

CI Credible Interval. xviii–xxi, 20, 24, 51, 55, 95, 98, 108

CLPBF Cumulative Log Predictive Bayes Factor. xvii, xxi, 22, 28, 30, 47, 50, 67, 73, 109, 113

Copula Copula. xix, xxiv, xxv, 4, 60, 62–65, 67–75

CPF Conditional Particle Filter. 17, 44

DAG Directed Acyclic Graph. 42, 63

DIC Deviance Information Criterion. 46, 51, 52, 66, 67, 70, 73, 74

EDHMM Explicit-duration Hidden Markov Model. 11, 13, 42

ESS Effective Sample Size. 16

GBM Geometric Brownian Motion. 38

HM Heston Model. 61

HMC Hamiltonian Monte Carlo. 4, 18, 44, 65

HMM Hidden Markov Model. xi, 2, 3, 7, 8, 10, 11, 13, 14, 27, 31, 33, 39, 41, 51

HSMM Hidden Semi-Markov Model. xi, xviii–xx, xxiii, 3, 7–11, 13–15, 20, 22, 24, 26, 27, 31, 33, 37, 38, 41–43, 47, 50–52, 92–96, 98–100

HSMM-EM HSMM-driven epidemic Model. xviii, xx, xxi, xxiv, 37, 39, 42, 43, 45, 48, 53, 56, 102, 103, 106–109

IBIS Iterated Batch Importance Sampling. 4, 65–67

ifr Infection Fatality Ratio. xviii, 41, 43, 49, 50

IS Importance Sampling. 15

LPPD Log Pointwise Predictive Density. 46, 52, 66, 70, 74

MC Monte Carlo. 20

MCMC Markov Chain Monte Carlo. xxi, xxiv, 1, 2, 4, 9, 14, 17–21, 33, 44–46, 59, 60, 65–69, 72, 73, 81, 111

NUTS No U-Turn Sampling. 2, 4, 18, 44, 65

ODE Ordinary Differential Equation. xviii, xxiv, 3, 39–41, 44, 45, 48–51, 53, 54, 56

PACF Partial Autocorrelation Function. xix, 20, 91

PF Particle Filter. xix, 2, 3, 8, 9, 15–17, 19–22, 33, 39, 40, 44, 45, 52, 65, 90, 91

PGAS Particle Gibbs with ancestors sampling. 18, 21

PGIBBS Particle Gibbs. 9, 44, 45

PL Predictive Likelihood. xvii, xxiii–xxv, 22, 24, 25, 28, 31, 46, 47, 51, 52, 66, 70, 73, 74

PMCMC Particle MCMC. xi, xviii–xxi, 2, 9, 14, 15, 18–23, 33, 37, 39, 44, 45, 47, 51, 52, 54, 91, 93, 94, 103, 106

PMH Particle Metropolis Hastings. 18, 21, 44

SEEIIR Susceptible-Exposed-Exposed-Infected-Infected-Recovered. xxiv, 39, 47, 48, 52, 53

SEIR Susceptible-Exposed-Infected-Recovered. xi, 3, 38–40

SIR Sequential Importance Resampling. 16

SIS Sequential Importance Sampling. 15, 16

SMC Sequential Monte Carlo. xx, xxi, 1–4, 8, 9, 19, 37, 45, 59, 60, 70, 74, 81, 96, 98, 99, 107–109, 112

SMC2 Sequential Monte Carlo Squared. xi, xviii–xx, xxiii, 9, 10, 19–24, 28, 29, 31–33, 39, 45–47, 51, 52, 65, 95, 100, 101

SSM State Space Model. xi, 1–3, 8, 9, 14, 16, 18, 19, 24, 27, 31, 38, 39, 41, 45, 47, 52

SV Stochastic Volatility. xi, xxiv, xxv, 4, 27, 59–62, 70, 72–75

SVC Stochastic Volatility Model with Copula Dependencies. xix, xxi, 64, 67, 71, 111, 112

VI Variational Inference. 1

WAIC Watanabe–Akaike Information Criterion. 46, 51, 52, 66, 67, 70, 73, 74

Chapter 1

Introduction

Bayesian Inference has grown increasingly popular in the last decade as inference algorithms that were previously difficult to implement have become more accessible through the immense effort of organizations including the Stan Development Team (2021), Ge et al. (2018) or Salvatier et al. (2016) and the open source policy of their software libraries. In particular, classic batch estimation techniques such as Markov Chain Monte Carlo (MCMC) and Variational Inference (VI) have seen many new research areas evolve. There is an abundance of new applications for these techniques emerging frequently, in large part due to the ease of ready-made tools requiring minimal coding effort from the user. Another area that has seen vast amounts of research effort is known as Sequential Monte Carlo (SMC), inference methods that are particularly suited for times series and sequential data, which often arise naturally in real applications. My own doctoral research primarily evolved around combining and improving SMC with MCMC methods. SMC methods can be broadly classified into two categories: density tempering or annealing, which involves expanding in the density region, and data tempering or annealing, which involves expanding in the data dimension. A short overview of this is given in (Gunawan et al., 2021, Dai et al., 2020). Within this framework, I mainly worked with State Space Models. I extensively explored the concepts proposed by Chopin et al. (2013) in Chapters 2 and 3. In Chapter 4, I ventured into non-latent variable models based on the ideas presented by Chopin (2002). For each project, a brief overview with the most important literature contributions is provided in Section 1.3.

1.1 Challenges

While both SMC and MCMC methods have excellent software tools available, very rarely are such packages inter-operable and, consequently, less effort has been conducted on the interconnection between MCMC and SMC techniques. This can lead to challenges when working with complex models typically used in research projects that require advanced MCMC methods. A big challenge during my research was thus the ability to use state-of-the-art tools of both research directions jointly, which ultimately led me to create and open source libraries that do exactly this, described more in detail in Section 1.2.

Modelling wise, the major challenge for State Space Models and more general Latent Variable Models is working around the in general intractable likelihood function. While basic models such as the popular Hidden Markov Model (HMM) have reasonable computational complexity for the likelihood function calculation, more advanced models can have intractable or computationally extremely challenging costs. An in-depth discussion on this topic is provided in Section 2.

1.2 Open Source Contributions

While pursuing my PhD, a significant amount of time has been invested into developing and open sourcing modular software pieces in which advanced MCMC kernels can be used efficiently within SMC techniques. For instance, Baytes.jl is a Julia package to perform batch and sequential Bayesian inference and provides valuable tools to summarize and analyze the corresponding algorithm output. It consists of several sub-libraries such as the MCMC library BaytesMCMC.jl that contains implementations for, e.g., the No U-Turn Sampling (NUTS) kernel and the fully auto-differentiable Particle Filter library BaytesFilters.jl. In the Particle MCMC (PMCMC) package BaytesPMCMC.jl, one can freely combine kernels from the BaytesMCMC.jl and the BaytesFilters.jl libraries; the SMC module BaytesSMC.jl provides methods to mix all sub-libraries listed above. Moreover, as a by-product of expanding these software components, tools with a broader span of applications have been open sourced, including libraries to perform Automatic Differentiation on (nested) model parameter structures, see ModelWrappers.jl and BaytesDiff.jl. These software packages mark a major achievement in my work and have been the backbone of each project within this thesis. They are intended as a tool for researchers working with exotic models, which are more common in research applications and are otherwise hard to implement with standard probabilistic programming languages.

1.3 Thesis Structure and Literature Overview

The projects discussed in Chapters 2, 3 and 4 were completed in chronological order but are self-contained, so each section can be read independently. Between each Chapter, a notable quote from a book I read or a series I watched during my time at the London School of Economics and Political Science (LSE) is shared; they serve as pleasant memories when reflecting on my work.

Chapter 2 explores the class of Hidden Semi-Markov Models (HSMM), see (Murphy, 2002, Yu, 2010, 2016), a generalization of the popular discrete state space Hidden Markov Model, see (Baum and Petrie, 1966). In this section, we construct efficient computational schemes that operate on the joint space of latent states and parameters based on the ideas from (Andrieu et al., 2010, Andrieu and Roberts, 2009) and Chopin et al. (2013). Notably, we implement an efficient method to sample the latent state trajectory via a Particle Filter (PF), see Johansen and Doucet (2008) for an excellent review on this topic. These Sequential Monte Carlo (SMC) methods also facilitate techniques for Bayesian model selection and predictive performance assessment based on the ideas of the aforementioned authors and the terminology of Kastner (2016). We display the suitability of this framework by modelling sequential financial data through Auto-regressive Models that are linked to HSMMs.

In Chapter 3, we formulate and provide methods to efficiently estimate a novel Susceptible-Exposed-Infected-Recovered (SEIR)-type discrete State Space Model with time-varying, piecewise constant transmission rate parameter between susceptible and infected individuals. The need for time-varying parameter in the general SEIR structure, see (Cohen, 1992, Diekmann et al., 2012), is provided by e.g. Flaxman et al. (2020b). To facilitate our estimation framework, we reuse the SMC machinery developed in the first project, but have to adapt the Particle Filter machinery to account for the Ordinary Differential Equation (ODE) arising from the SEIR structure at each transition in the PF propagation. This make the computational complexity significantly more challenging. A detailed model examination is provided in Chapter 3. We display the suitability for this class of models for decision making in a case study on reported COVID-19 data in the United Kingdom, and provide fatalities and infections predictions as well as COVID-19 relevant diagnostics in a sequential setting.

Chapter 4 introduces a new class of Stochastic Volatility Models, see (Engle, 1982, Bera and Higgins, 1993). Standard SV Models have the capacity to capture numerous significant stylized effects, such as leptokurtic and heavy-tailed marginal return distributions, volatility clustering, and the phenomenon known as the leverage effect, see Shephard (1996), Ghysels et al. (1996). In this project, we use Copula (Copula) Models, see (Joe, 1997), to depict the joint dependency structure between stock prices and their volatility. Several recent works have empirically illustrated the presence of non-linear, asymmetric structures at the joint distribution of prices and volatility, and suggested the use of classes of Copulas for capturing such effects (see e.g. Ning et al. (2008)). Copula theory provides an extremely flexible modelling framework for a multitude of shapes for asymmetric joint distributions, see, e.g., Joe (2014)) and Cherubini et al. (2004), Czado et al. (2019), Krupskii and Joe (2020), Bladt and McNeil (2022) for more general Copula usage in finance. Model parameter are estimated using advanced Markov Chain Monte Carlo (MCMC) and Sequential Monte Carlo (SMC) kernels for each selected copula. As there is no latent state space involved, a slightly different SMC method known as Iterated Batch Importance Sampling (IBIS), see Chopin (2002), can be used. The corresponding MCMC kernel used in all projects is known as No U-Turn Sampling (NUTS), an Hamiltonian Monte Carlo (HMC) kernel with automatic hyper-parameter tuning, which was proposed by Hoffman and Gelman (2014). Notable improvements for this algorithm are suggested in Betancourt (2016). In this project, batch as well as sequential Bayesian model choice are used to determine the applicability for each copula, see e.g. (Spiegelhalter et al., 2002, Watanabe, 2010). Our results support the more flexible Copula approach over the use of Gaussian innovations that are standard in the financial modelling research.

Chapter 5 concludes the thesis with some reflection on the past few years at the London School of Economics and Political Science as well as the future direction of my work. While all projects tackle distinct economic problems, they are unified in their sequential nature. The broader contribution of our work is thus to demonstrate the capabilities and advantages of working with sequential inference methods and to incentivize their use.

Taking on a challenge is a lot like riding a horse, isn't it?
If you're comfortable while you're doing it, you're probably doing it wrong.

Ted Lasso (*Ted Lasso*, 2020)

Chapter 2

Sequential Bayesian Learning for Hidden Semi-Markov Models

In this Chapter, we explore the class of the Hidden Semi-Markov Model (HSMM), a flexible extension of the popular Hidden Markov Model (HMM) that allows the underlying stochastic process to be a semi-Markov chain. HSMMs are typically used less frequently than their basic HMM counterpart due to the increased computational challenges when evaluating the likelihood function. Moreover, while both models are sequential in nature, parameter estimation is mainly conducted via batch estimation methods. Thus, a major motivation of this Chapter is to provide methods to estimate HSMMs (1) in a computationally feasible time, (2) in an exact manner, i.e. only subject to Monte Carlo error, and (3) in a sequential setting. We provide and verify an efficient computational scheme for Bayesian parameter estimation on HSMMs. Additionally, we explore the performance of HSMMs on the Volatility index (VIX) time series using Autoregressive (AR) models with hidden semi-Markov states and demonstrate how this algorithm can be used for regime switching, model selection and clustering purposes.

2.1 Introduction

Discrete State Space Models (SSM) provide a flexible class of models with applications in ecology, economics, finance, robotics and signal processing (Bulla and Bulla, 2006, Lindsten and Schön, 2013, Chopin and Papaspiliopoulos, 2020, Corenflos et al., 2021), among others. They can handle structural breaks, shifts, or time-varying parameters and still have an interpretable structure. Moreover, such models are generative and allow for multi-step forecasting. However, analytical forms of the likelihood function are only available in special cases, and standard parameter optimization routines are often challenging to implement. Given the observed data $e_{1:T} = (e_1, \ldots, e_T)$ and parameter θ, the major challenge in the estimation of SSMs is thus the generally intractable likelihood function $p_\theta(e_{1:T})$, which integrates over the latent state trajectory $s_{1:T}$ such that $p_\theta(e_{1:T}) = \int p_\theta(e_{1:T}, s_{1:T}) ds_{1:T}$.

A flexible discrete SSM on which we focus in this Chapter is known as a Hidden Semi-Markov Model. HSMMs have a flexible state duration distribution, well suited for processes that remain in any particular state for an extended period of time, and can be considered as generalizations of the well-known basic Hidden Markov Model introduced in Baum and Petrie (1966). A further review on HMMs can be found in Cappé et al. (2005). HSMMs have been employed in ecology, epidemiology, finance (Bulla and Bulla, 2006, Pohle et al., 2021, Visani et al., 2021) and many other fields (Yu, 2016), but are typically used more sporadically than their standard HMM counterparts because the likelihood function is significantly more costly to evaluate. In the HMM case, the likelihood has computational complexity of $\mathcal{O}(K^2T)$, where K = number of latent states, T = number of data points, see Baum and Petrie (1966). For the HSMM, this is a much more expensive operation of order $\mathcal{O}(K^2(d_{max} - d_{min})^2T)$ (Murphy, 2002, Dewar et al., 2012), where d_{min} and d_{max} denote the minimal and maximal state duration in a latent regime. In practice, $(d_{max} - d_{min}) >> K$, as described in more detail in Section 2.2, which often leads to computationally expensive inference algorithms. Such considerations have led to the use of approximate methods in applications where HSMMs provide valuable models, see for example Hadj-Amar et al. (2022) and Xiao et al. (2018).

In this Chapter, we follow an alternative route aiming to construct efficient computational schemes that operate on the joint space of latent states and parameters using Sequential Monte Carlo (SMC) methods that are exact, in the sense that they are only amenable to Monte Carlo error. The fundamental building block of the proposed schemes is the Particle Filter (PF), see for example Doucet and Johansen (2011) and the references therein. Traditionally, SMC samplers such as PFs have been used to estimate the underlying state sequence of SSMs, while standard Markov Chain Monte Carlo (MCMC) samplers facilitate Bayesian inference for the model parameters. More recently, combining these methods is becoming increasingly popular, see Daviet (2018) and Buchholz et al. (2020). A natural computational framework that jointly infers the latent state sequence and model parameter is known as Particle MCMC (PMCMC) (Andrieu et al., 2010, Andrieu and Roberts, 2009). To our knowledge, PMCMC has not been used for HSMMs, so we work within this framework aiming to construct an efficient implementation. In particular, we focus on the Particle Gibbs (PGIBBS) version to implement parameter updates, conditional on the latent state trajectory, via Hamiltonian MCMC (Neal, 2012) variants. Given that SSMs are typically used in applications with data of sequential nature, it is essential to explore techniques where previous parameter estimates can be reused once the data is updated. An example for that is provided by the Sequential Monte Carlo Squared (SMC2) algorithm, introduced in Chopin et al. (2012), which can be viewed as an extension of the main SMC framework of Chopin (2002) and Del Moral et al. (2006); see also Dai et al. (2020) for some recent work that includes a survey of applications in different contexts. Other similar approaches include Fearnhead and Taylor (2010) and Crisan and Miguez (2017). More information on these methods is provided in the Section 2.3.

The major motivation of this Chapter is thus to develop methods to estimate HSMMs (1) in a computationally feasible time and (2) in a sequential manner. The contribution of this Chapter is two-fold: First, we offer Sequential Monte Carlo schemes on Hidden Semi-Markov Models by tailoring ideas from Andrieu et al. (2010) and Chopin et al. (2013) for batch and sequential estimation. This offers several benefits over standard deterministic filtering techniques, including computational efficiency. The developed SMC schemes can also facilitate Bayesian model choice and assessment of predictive performance in an efficient manner. Second, we propose a novel class of models by linking Autoregressive-type models with HSMMs to better describe data consisting of financial and econometric time series. Sequential estimation of such models is particularly important as AR HSMMs have the potential to detect substantial changes in the data, which we illustrate in a case study on data that evolves rapidly during the Covid-19 pandemic.

The Chapter is organized as follows: Section 2.2 formally introduces HSMMs via a suitable formulation to apply sequential Monte Carlo methods such as Particle Filtering. It also provides justification for the use of PFs instead of deterministic filtering techniques. In Section 2.3, the developed methodology of this Chapter is presented, which includes the model choice criteria available from by-products of the estimation process. Section 2.4 explores the performance of the developed methods via simulation based experiments. In Section 2.5, we focus on the performance of HSMMs, estimated with the developed methodology of this Chapter, on real-world applications such as financial time series of the VIX index. Comparisons of different HSMMs as well as benchmark HMMs are conducted. Model selection, and in particular choice of the number of states using SMC2, is also put into test. Finally, Section 2.6 concludes with some relevant discussion.

2.2 Hidden Semi-Markov Model

A standard Hidden Markov Model may be specified via a bivariate stochastic process $\{e_t, s_t\}_{t=1,2,\dots}$, where s_t is an unobserved Markov chain and e_t is an observed sequence of independent random variables, conditional on s_t. The model is fully specified by the transition distribution f_θ, $s_t \sim f_\theta(s_t \mid s_{t-1}),\ \ t \geq 2$, the corresponding initial distribution π_θ, $s_1 \sim \pi_\theta(s_1)$, and the observation distribution g_θ, $e_t \sim g_\theta(e_t \mid s_t),\ \ t \geq 1$. Directly computing the likelihood function of this model involves summing up over all possible state sequences,

$$p(e_{1:T}) = \sum_{s_{1:T}} p(e_1 \mid s_1) p(s_1) \prod_{t=2}^{T} p(e_t \mid s_t) p(s_t \mid s_{t-1}).$$

Hence, various filtering techniques have been proposed that take into account the memory of the latent state variable to reduce the computational costs to $\mathcal{O}(K^2 T)$. One shortcoming of HMMs is their explicit distributional assumption regarding the duration in any particular state. To give insight into this issue, we denote $p(s_{t+k} = j,\ s_{t+1:t+k-1} = i \mid s_t = i)$, the probability a state remains in any current state until it switches, as state duration distribution. In the HMM case, this probability is implicitly geometric. Set $p(s_t = i \mid s_{t-1} = i) = \mathcal{T}_{ii}$, and assume there are only 2 states, then for a homogeneous Markov chain, using the chain rule and the Markov assumption, it holds:

$$\begin{aligned} p(s_{t+3} = j, s_{t+2} = i, s_{t+1} = i \mid s_t = i) &= p(s_{t+3} = j \mid s_{t+2} = i) p(s_{t+2} = i, \mid s_{t+1} = i) p(s_{t+1} = i \mid s_t = i) \\ &= (1 - \mathcal{T}_{ii}) * \mathcal{T}_{ii}^2 \end{aligned}$$

In general, for $t + k$ steps, it holds that

$$\begin{aligned} p(s_{t+k} = j, \ldots, s_{t+1} = i \mid s_t = i) &= (1 - \mathcal{T}_{ii}) * \mathcal{T}_{ii}^{k-1} \\ &= Geometric_{\mathcal{T}_{ii}}, \end{aligned}$$

where the geometric distribution has to be interpreted as the length of state duration up to and including the transition to the other state. For processes that tend to stay in any particular state for a long-time horizon, this may be a poor modelling choice. Alternatively, the state duration could be explicitly modelled. A Hidden Semi-Markov Model, see (Murphy, 2002, Yu, 2010, 2016), is a generalization of an HMM, which may be viewed as a HSMM with Geometric state duration distribution. A graph structure and a comparison to the standard HMM can be seen in figures 2.1a and 2.1b. A specific formulation of the HSMM that explicitly defines the duration distribution is known as Explicit-duration Hidden Markov Model (EDHMM). Transitions are allowed only at the end of each state, resulting in the following definition:

Definition 2.2.1. *Hidden semi-Markov Model (HSMM) A hidden semi-Markov model is a bivariate stochastic process $\{e_t, z_t\}_{t=1,2,\ldots}$, where $z_t = \{s_t, d_t\}$ is an unobserved semi-Markov chain and, conditional on z_t, e_t is an observed sequence of independent random variables. The model is fully specified by the transition distribution $f_\theta(s_t \mid s_{t-1}, d_{t-1})$ of s_t*

$$s_t \sim \begin{cases} \delta(s_t, s_{t-1}) & d_{t-1} > 0 \\ f_\theta(s_t \mid s_{t-1}, d_{t-1}) & d_{t-1} = 0 \end{cases},$$

the duration distribution h_θ of d_t

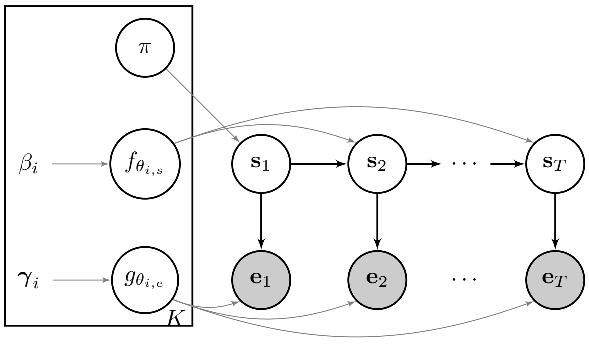
$$d_t \sim \begin{cases} \delta(d_t, d_{t-1} - 1) & d_{t-1} > 0 \\ h_\theta(d_t \mid s_t, d_{t-1}) & d_{t-1} = 0 \end{cases},$$

the corresponding initial distribution π_θ of z_t, and the observation distribution g_θ, $e_t \sim g_\theta(e_t \mid s_t)$,

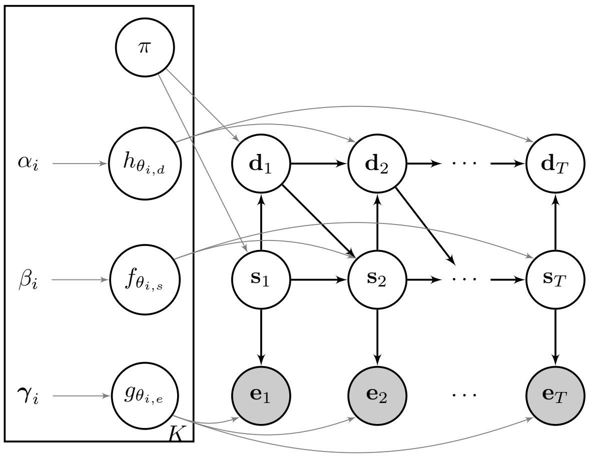
$$e_t \sim g_\theta(e_t \mid s_t).$$

where $\delta(a, b)$ is and indicator function and equals 1 if $a = b$ and 0 otherwise.

Popular choices for the duration distribution h_θ are the Poisson or the Negative Binomial distribution, for greater flexibility at the cost of an additional model parameter per state. The observation distribution g_θ can be set according to the specifics of the application at hand, which includes higher order data dependency such as



(a)



(b)

Figure 2.1 Figure 2.1a depicts a K-state Bayesian HMM, parameter θ and hyper-parameter $\{\beta, \gamma\}$. The shaded nodes e_t denote the observed data at time t, while the unshaded nodes indicate the latent state s_t. $\theta_{i,s}$ denotes the parameter at state i for latent state s, and β_i the corresponding hyper-parameter. f is the transition distribution s, g is the observation distribution for e. π denotes the initial distribution for s. Figure 2.1b depicts aK-state Bayesian HSMM, parameter θ and hyper-parameter $\{\alpha, \beta, \gamma\}$. The shaded nodes e_t denote the observed data at time t, while the unshaded nodes indicate latent duration d_t and state s_t. $\theta_{i,d}$ denotes the parameter at state i for latent duration d, and α_i the corresponding hyper-parameter. h and f are the transition distributions for d and s, g is the observation distribution for e. π denotes the initial distribution for d and s.

$g_\theta(e_t \mid s_t, e_{t-k:t-1})$, for $t \geq 1$. An example for an excellent data dependency use case is provided in Section 2.5.1, where AR(1) models are used in each latent regime. The joint distribution of an EDHMM given the parameter can be stated as

$$p_\theta(s_{1:T}, d_{1:T}, e_{1:T}) = \pi_\theta(s_1)\pi_\theta(d_1)g_\theta(e_1 \mid s_1)\prod_{t=2}^{T} f_\theta(s_t \mid s_{t-1}, d_{t-1})h_\theta(d_t \mid s_t, d_{t-1})g_\theta(e_t \mid s_t)$$

The likelihood can be obtained by integrating out both $s_{1:T}$ and $d_{1:T}$. Due to the additional latent variables $d_{1:T}$, this is a much more computationally expensive operation than in the standard HMM case. In order to gain more insight on this, we can shrink the graphical model structure of a HMM and HSMM to a single time step. In order to compute the likelihood of observation e_{t+1} given the current state s_t, a sum over all possible state transitions has to be taken, as can be seen in equation (2.1). This resembles a standard mixture model computation, with the transition matrix of the HMM replacing the mixture component weights.

$$\begin{aligned} p(e_{t+1} \mid s_t = k) &= \sum_{s_{t+1}} p(e_{t+1}, s_{t+1} \mid s_t = k) \\ &= \sum_{s_{t+1}} p(s_{t+1} \mid s_t = k)p(e_{t+1} \mid s_{t+1}) \end{aligned} \tag{2.1}$$

For the HSMM, however, the duration variable means an additional sum over a random variable that has at worst an infinite number of terms in the case of duration distributions with countably infinite support, shown in equation (2.2).

$$\begin{aligned} p(e_{t+1} \mid s_t = k, d_t = j) &= \sum_{s_{t+1}}\sum_{d_{t+1}} p(e_{t+1}, s_{t+1}, d_{t+1} \mid s_t = k, d_t = j) \\ &= \sum_{s_{t+1}}\sum_{d_{t+1}} p(s_{t+1} \mid s_t = k, d_t = j)p(d_{t+1} \mid s_{t+1}, d_t = j)p(e_{t+1} \mid s_{t+1}) \end{aligned} \tag{2.2}$$

Note that $\sum_{z_{t+1}} = \sum_{d_{t+1}}\sum_{s_{t+1}}$, which sums up all possible durations over all states, has at worst an infinite number of terms if the duration distributions have countably infinite support, and at best a large number of terms for long sequences, see Dewar et al. (2012). The standard approach to tackle this problem is to set up a minimum and maximum duration d_{min} and d_{max}, where the computational complexity of the forward-backward algorithm reduces to $O(T(K(d_{max} - d_{min})^2))$, compared to the original $O(TK^2)$ in the HMM, see Murphy (2002). Choosing an appropriate maximum duration varies depending on the underlying data. If the truncation is too small, then inference will typically fail, if is too large then calculations might become infeasible. Hence, $(d_{max} - d_{min})$ may increase the computational complexity to burdensome levels, which requires the modeler to set d_{max} too small. Other approaches include (Johnson and Willsky, 2013, Johnson, 2014), who decrease computational complexity by censoring the initial or end time. To give a numerical example, in order to appropriately estimate a HSMM, we assume that at least a single state transition has to occur, hence $d_{max} < T$, but $(d_{max} - d_{min}) >> K$. The computational costs for a 5-state HMM and 1000 data points would be $\mathcal{O}(5^2 \times 1000)$, while for the HSMM, assuming $d_{max} = 500$ and $d_{min} = 0$, $\mathcal{O}(5^2 \times 500^2 \times 1000)$ for a single likelihood call.

Alternatively, a particle filter can be used for the likelihood computation in $\mathcal{O}(NT)$ operations, even if the model has HSMM dynamics. Exact inference is retained, subject to Monte Carlo error, using the Particle MCMC algorithm (Andrieu et al., 2010). N denotes the number of particles used in the filter, and we observed that it is sufficient to set $N = \frac{T}{2}$ for sequential Monte Carlo schemes on HSMMs, see Section 2.3 for more detail.

2.3 Bayesian Inference on Hidden Semi-Markov Models

In a Bayesian framework, the typical goal is to infer the posterior distribution of the model parameter θ given the observed data $e_{1:T}$, $p(\theta \mid e_{1:T}) = \frac{p_\theta(e_{1:T})\ p(\theta)}{p(e_{1:T})}$. This is a challenging task for SSMs as it involves integrating $s_{1:T}$ over the likelihood $p_\theta(e_{1:t}) = \int p_\theta(e_{1:T}, s_{1:T})\ ds_{1:T}$, which is typically intractable or costly to evaluate. Hence, usually the full posterior distribution

$$p(s_{1:T}, \theta \mid e_{1:T}) = \frac{p_\theta(e_{1:T} \mid s_{1:T})\ p_\theta(s_{1:T})\ p(\theta)}{p(e_{1:T})}$$

is inferred. If $s_{1:T}$ is continuous, the target distribution $p(s_{1:t}, \theta \mid e_{1:t}) \propto p(e_{1:t} \mid s_{1:t}, \theta) \times p(s_{1:t} \mid \theta) \times p(\theta)$ can theoretically be estimated via MCMC. However, in this case, a state trajectory of $p(s_{1:T} \mid \theta)$ has to be sampled while evaluating the target function, usually resulting in very poor outcomes of this strategy. Alternatively, classic Gibbs sampling strategies could be applied, which iterate the estimation process between sampling the latent states given the continuous model parameter and vice versa. However, for the latent state trajectory proposal step, a forward-backward algorithm would have to be employed again, as other choices such as one-at-a-time updates or overlapping blocks are known to cause slow mixing (Kalogeropoulos et al., 2010, Golightly, 2009) of the Markov chain.

A more general attempt to jointly target the full joint posterior $p(s_{1:T}, \theta \mid e_{1:T})$ can be shown as follows:

- 1. propose $\theta^\star \sim f(\theta^\star \mid \theta)$ and $s^\star_{1:T} \sim p_{\theta^\star}(s^\star_{1:T} \mid e_{1:T})$,
- 2. accept $(\theta^\star, s^\star_{1:T})$ with acceptance probability

$$a((s^\star_{1:T}, \theta^\star), (s_{1:T}, \theta)) = \frac{p_{\theta^\star}(e_{1:T})}{p_\theta(e_{1:T})} \frac{p(\theta^\star)}{P(\theta)} \frac{q(\theta \mid \theta^\star)}{q(\theta^\star \mid \theta)}. \tag{2.3}$$

The last term in equation (2.3) has been simplified by using the Basic Marginal Likelihood Identity (BMI) of Chib (1995). This framework allows to jointly sample θ and $s_{1:t}$, but the in general intractable likelihood function is still contained in step 3. While this term can be computed analytically for the discrete HSMM via the so-called forward-backward algorithms, they are prohibitively expensive to run, as described in Section 2.2. Going forward, we introduce the algorithmic machinery known as Particle MCMC (Andrieu et al., 2010), which replaces the likelihood evaluation with an estimate $\hat{p}_\theta(e_{1:T})$ from a Particle Filter.

2.3.1 Particle Filtering

Particle Filters are often used to solve filtering equations in the form of $\pi_t(x_{1:t}) = \frac{\tau_t(x_{1:t})}{z_t}$. The goal is to sequentially sample a sequence of random variables, $x_t, t \in (1, ..., T)$ that come from a sequence of target probabilities $\pi_t(x_{1:t})$ with the same computational complexity at each time step. If it impossible to directly sample from π_t, a similar proposal distribution q_t can be used, s.t. $\pi_t(x_{1:t}) > 0 \Rightarrow q_t(x_{1:t}) > 0$. The fraction of $\tau_t(x_{1:t})$ and $q_t(x_{1:t})$ is known as un-normalized weight function $w_t(x_{1:t}) = \frac{\tau_t(x_{1:t})}{q_t(x_{1:t})}$, s.t. the target distribution can be rewritten as $\pi_t(x_{1:t}) = \frac{w_t(x_{1:t})q_t(x_{1:t})}{z_t}$. This method is recognized as Importance Sampling (IS). Often, the variable of interest is the normalizing constant $z_t = \int \tau_t(x_{1:t})dx_{1:t} = \int w_t(x_{1:t})q_t(x_{1:t})dx_{1:t}$, which can be approximated via the un-normalized weight functions

$$\hat{z}_t = \frac{1}{t} \sum_{i=1}^{K} \frac{\tau_t(x_{1:t})}{q_t(x_{1:t})} = \frac{1}{t} \sum_{i=1}^{K} w_t(x^i_{1:t}).$$

This technique does, unfortunately, rapidly degenerate as t becomes larger. A technique to sequentially sample from such distributions is called Sequential Importance

Sampling (SIS), which keeps the computational costs fixed given additional time steps by decomposing the joint distribution as $\tau_t(x_{1:t}) = \tau_{t-1}(x_{1:t-1})\tau_t(x_t \mid x_{1:t-1})$. Similarly, the importance distribution can be decomposed as $q_t(x_{1:t}) = q_1(x_1)\prod_{n=2}^{t} q_n(x_n \mid x_{1:n-1})$. The associated un-normalized weights can then be computed recursively via $w_t(x_{1:t}) = w_1(x_1)\prod_{k=1}^{t} \alpha_k(x_{1:k})$, where the incremental importance weights $\alpha_t(x_{1:t})$ are given as $\alpha_t(x_{1:t}) = \frac{\tau_t(x_t|x_{1:t-1})}{q_t(x_t|x_{1:t-1})}$.

In most state space models, the memory for $q_t(x_t \mid x_{1:t-1})$ is limited, so the target distribution can be evaluated and sampled from via the methods above at fixed computational costs. The only freedom in this framework is choosing an appropriate q_t. Unfortunately, it can be shown that the variance of the corresponding weights grows with t, and we refer to this problem as weight degeneracy. Due to weights degeneracy, the variance for the estimator of the normalizing constant $\hat{z}$ is also increasing with t. To alleviate this obstacle, a resampling step for the particle trajectories $x_{1:t}$ that normalizes the corresponding weights can be applied. Algorithms that do so at each iteration are known as Sequential Importance Resampling (SIR). However, this creates a new challenge known as sample path degeneracy, which refers to the problem that continuously resampling particle paths ultimately ends with very few unique trajectories. Balancing weight and sample path degeneracy is an ongoing research topic, and the most common method is to resample trajectories only at specific iterations, for example, if the Effective Sample Size (ESS)

$$ESS_t = \frac{1}{\sum_{n=1}^{N}\left(\frac{w_t(x_{1:t}^n)}{\sum_{i=1}^{N} w_t(x_{1:t}^i)}\right)^2}$$

of the particles is less than an a priori set threshold. For a discussion on several different resampling techniques, see Douc and Cappe (2005). Adaptive resampling mitigates the exploding variance of the particle weights and keeps sample path degeneracy in check. Algorithms that apply this machinery are commonly referred as Particle Filters. They have a fixed computational complexity that is both linear in time T and in number of particles N, $\mathcal{O}(NT)$, and return an estimate of the normalizing constant $\hat{z}_t$ and a particle path of $\hat{\pi}(x_{1:t})$. In the SSM case, the joint distribution and the normalizing constant are of the form $\tau_t(x_{1:t}) = p_\theta(s_{1:t}, e_{1:t})$ and $z_t = p_\theta(e_{1:t})$. An approximation for the likelihood can be computed via the weights $\hat{z}_t = \frac{1}{N}\sum_{n=1}^{N} w_t(s_{1:t}^n, e_{1:t})$, which can be decomposed in the following recursive form:

$$w_t(s_{1:t}, e_{1:t}) = w_1(s_1, e_1) \prod_{k=1}^{t} \alpha_k(s_{1:k}, e_{1:k}).$$

The incremental weight α is defined as $\alpha_t(s_{1:t}, e_{1:t}) = \frac{p_\theta(e_t|s_{1:t}, e_{1:t-1})\ p_\theta(s_t|s_{1:t-1}, e_{1:t-1})}{q(s_t|s_{1:t-1}, e_{1:t})}$, and the full likelihood estimate can be expressed as $\hat{z_T} = \frac{1}{N}\sum_{n=1}^{N}\prod_{k=1}^{T}\alpha_k(s^n_{1:k}, e_{1:k})$, which is usually the preferred method in particle filter software implementations as this avoids memory allocations. This permits the estimation of the incremental likelihood $p_\theta(e_t \mid e_{1:t-1}) \approx \frac{1}{N}\sum_{n=1}^{N}\alpha_t(s^n_{1:t}, e_{1:t})$ as well, which becomes relevant in the model selection Section 2.3.5. $p_\theta(e_t \mid s_{1:t}, e_{1:t-1})$ and $p_\theta(s_t \mid s_{1:t-1}, e_{1:t-1})$ are model distributions, and have usually limited memory. The only free distribution to choose is $q(s_t \mid s_{1:t-1}, e_{1:t})$, which should ideally look like $p_\theta(s_t \mid s_{1:t-1}, e_{1:t})$. A common and simple choice is known as Bootstrap Particle Filter (BPF), which takes $q(s_t \mid s_{1:t-1}, e_{1:t}) = p_\theta(s_t \mid s_{1:t-1}, e_{1:t-1})$, reducing the incremental weight to $\alpha_t(s_{1:t}, e_{1:t}) = p_\theta(e_t \mid s_{1:t}, e_{1:t-1})$. Another popular approach is the so called Auxiliary Particle Filter (APF) (Pitt and Shephard, 1999), which assumes $\alpha_t(s_{1:t}, e_{1:t})$ to be independent of s_t (in the Bootstrap Particle Filter, this does not hold!). See (Kantas et al., 2015, Doucet and Johansen, 2011) for a more in-depth review. A pseudo algorithm implementation for a standard PF can be found in Algorithm 1, where the auxiliary variable a^i_t refers to the ancestor path of a particular particle s^i at time t. Hence a particle trajectory can be recursively defined as $s^i_{1:t} = (s^{a^i_t}_{1:t-1}, s^i_t)$. Resampling the whole particle trajectory is equivalent to sampling a new ancestor path. Note that it is usually much faster to sample ancestors one at a time and then recursively recover the resampled particle path than to resample the whole particle trajectory at each iteration. A variant of this algorithm is known as Conditional Particle Filter (CPF), where a single particle path, $s'_{1:T}$, is chosen a priori as reference trajectory. This implementation tracks a slightly different target distribution, $\hat{p}(s_{1:T} \mid s'_{1:T}, e_{1:T})$, and is used in Section 2.4 and 2.5. A pseudo algorithm for this Conditional Particle Filter (CPF) with ancestor sampling can be found in Algorithm 2, and a more thorough review can be read in (Lindsten et al., 2014, 2015).

Once a particle filter has been run, observed and latent data can be forecasted by first sampling a new state $s_{T+1} \sim p_\theta(\cdot \mid s_{1:T}, e_{1:T})$ and then a new data point given this state $e_{T+1} \sim p_\theta(\cdot \mid s_{1:T+1}, e_{1:T})$. s_{T+1} can be sampled by forward propagating algorithm 1 or 2 from T to $T+1$, thereby reusing particles from 1 to T. This can be repeated for multiple time steps as well, resulting in a very fast procedure to sample from predictive distributions.

2.3.2 Particle Markov Chain Monte Carlo

The most common Bayesian inference technique for model parameter θ is known as Markov Chain Monte Carlo. Basic familiarly with this concept is assumed, and we refer to Craiu and Rosenthal (2014) for a more detailed review about standard MCMC techniques. A pseudo algorithm for a basic Metropolis step can be found in Algorithm 3. The major difficulty in algorithm 3 is finding a good proposal distribution f, which often results in slow mixing. A MCMC kernel that automatically tunes its proposal distribution at the cost of additional tuning hyper-parameter is known as Hamiltonian Monte Carlo (HMC), see (Neal, 2012) for an introduction and (Betancourt, 2018) for a review on this topic. A pseudo algorithm can be seen in Algorithm 4. The additional tuning hyper-parameter can be configured on the fly in the famous extension No U-Turn Sampling (NUTS), which was proposed by Hoffman and Gelman (2014). Notable improvements for this algorithm are suggested in Betancourt (2016).

Note that both the HMC and NUTS kernel require the target density to be fully differentiable with respect to the model parameter θ. In the SSM case, targetting the marginal posterior distribution $p(\theta \mid e_{1:T}) \propto p_\theta(e_{1:T})\ p(\theta)$ is difficult or impossible via MCMC, as the latent variables in $p_\theta(e_{1:T}) = \int_{s_{1:T}} p_\theta(e_{1:T}, s_{1:T})$ have to be integrated out in the proposal ratio. However, $p_\theta(e_{1:T}, s_{1:T})$ is usually computable pointwise, so $p(s_{1:T}, \theta \mid e_{1:T})$ can be targeted. In the Particle Metropolis Hastings (PMH) case, formally introduced in (Andrieu et al., 2010) and shown in pseudo Algorithm 5, a particle filter is used to obtain approximations for $p_\theta(e_{1:T})$ and $p_\theta(s_{1:T} \mid e_{1:T})$ as substitutes for the analytical solutions to target $p(s_{1:T}, \theta \mid e_{1:T})$ jointly. In this setting, (Andrieu and Roberts, 2009) have shown the puzzling result that one can do so and still target the exact posterior distribution of interest. A major difficulty for this method is finding a good MCMC kernel $K_{mcmc}(e_{1:T}, \theta)$, because the θ proposal will be accepted based on the particle filter likelihood estimate, so tuning might be very noisy. Moreover, gradient based MCMC sampler do not work in this case as $p_\theta(e_{1:T})$ typically cannot be evaluated pointwise. A common critique on PMH is thus that this algorithm is ill-suited for a higher dimensional model parameter θ. A PMCMC variant that can mitigate this is known as Particle Gibbs with ancestors sampling (PGAS), see (Lindsten et al., 2014, 2015). A pseudo algorithm is shown in Algorithm 6. To account for sampling from an approximation via a particle filter and to preserve the invariance principle, a slightly adjusted $\hat{p}_{\theta^\star}(s^\star_{1:T} \mid s_{1:T}, e_{1:T})$ distribution is used to sample from the state trajectory. This method does not jointly estimate the state sequence and model parameter, but the state sequence is fixed when the new model parameter $\theta^\star \sim p_\theta(\theta^\star \mid s_{1:T}, e_{1:T})$ are sampled. In this step, more advanced HMC style MCMC kernels can be used as $p_{\theta^\star}(e_{1:T} \mid s_{1:T})$, which is easy to evaluate, replaces $p_{\theta^\star}(e_{1:T})$ in the acceptance ratio. Hence, more advanced MCMC kernels, such as the NUTS sampler, can be used to estimate model parameter θ in the PGAS setting.

2.3.3 Sequential Monte Carlo Squared

In a times series setting, forecasting is of major relevance. The standard way for prediction in a Bayesian setting is simple: obtain the posterior predictive distribution by integrating out the model parameter θ and, in the SSM case, the state trajectories $s_{1:T}$,

$$\begin{aligned} p(e_{T+1} \mid e_{1:T}) &= \int p(e_{T+1}, s_{T+1}, s_{1:T}, \theta \mid e_{1:T}) \; ds_{T+1}, s_{1:T}, \theta \\ &= \int p_\theta(e_{T+1} \mid s_{T+1}, s_{1:T}, e_{1:T}) \; p_\theta(s_{T+1} \mid s_{1:T}, e_{1:T}) \; p(s_{1:T}, \theta \mid e_{1:T}) \; ds_{T+1}, s_{1:T}, \theta. \end{aligned}$$

Once a sample for $p(s_{1:T}, \theta \mid e_{1:T})$ is obtained, the predictive distributions for $s_{T+1} \mid s_{1:T}, e_{1:T}, \theta$ and $e_{T+1} \mid s_{T+1}, s_{1:T}, e_{1:T}, \theta$ are trivial to sample from. A major drawback of the PMCMC machinery is that, even though this algorithm primarily works for models suited to times series settings, it only works as batch estimation. Once additional data is observed, the algorithm needs to be run again to target $p(s_{1:T+1}, \theta \mid e_{1:T+1})$. A method that uses PMCMC in a sequential setting is known as Sequential Monte Carlo Squared, see Chopin et al. (2012). SMC based algorithm often expand in the density region, known as density tempering or annealing, or the data dimension, known as data tempering or annealing, see Gunawan et al. (2021). SMC2 is a data tempering algorithm and moves a collection of particles that consist of the model parameters and latent states by incrementally adding data to the estimation process. At the beginning, particles are drawn from the prior. Subsequent particles are explored iteratively by using multiple Particle Filter and particle MCMC sampler. At each iteration, N Particle Filter are used to obtain the incremental likelihood estimates $\hat{p}_{\theta^n}(e_{1:t} \mid e_{1:t-1}) = \hat{Z}_t = \frac{1}{N} \sum_{n=1}^{N} \alpha_{\theta^n}(s^n_{1:t}, e_{1:t})$ and state trajectories $s^n_{1:t} \sim \hat{p}_{\theta^n}(s^n_{1:t} \mid e_{1:t})$ for $n = 1, \ldots, N$. If the estimates $\hat{p}_{\theta^n}(e_{1:t} \mid e_{1:t-1})$ are getting too noisy, the particles are jittered via PMCMC. Note that either the Particle Gibbs or the Particle Metropolis Hastings variant can be chosen for the PMCMC kernel. The density at the final iteration is the posterior distribution of interest. A pseudo algorithm can be seen in Algorithm 8.

A powerful feature of the SMC2 algorithm is that at each iteration, an unbiased estimate of the incremental marginal likelihood $\hat{p}(e_t \mid e_{1:t-1})$ is obtained at practically no extra costs, see equation (B.1). From here on, it is straight forward to obtain an estimate for the marginal likelihood, $\hat{p}(e_{1:T}) = \prod_{t=1}^{T} \hat{p}(e_t \mid e_{1:t-1})$, for model comparison or Bayes Factor (BF) calculations. Moreover, this machinery is highly parallelizable, a feature that is typically difficult to include in standard MCMC techniques, and most SMC2 iterations can be performed online as the particle filter can be propagated forward if no resampling step has been taken at the previous iteration. Additionally, during the propagation step, samples for s_{t+1} and e_{t+1} for posterior predictive distribution analysis can be obtained at each time index.

2.3.4 Tuning Configurations

Particle Filter tuning

In our setup, we used a bootstrap particle filter with transition distribution equal to the model dynamics as defined in Section 2.2. The particle resampling method was chosen to be systematic, and the resampling threshold for the ESS calculation was set to 75%. The only free tuning parameter in this case is the number of particles N. The higher N, the lower the variance for the log likelihood estimate, but the higher the computational costs per PMCMC iteration. We note that ultimately, the Particle MCMC samples are drawn from the correct target distribution, only subject to Monte Carlo error, independent of N. However, the mixing of the MCMC chains might be slower if less particles are used. In this case, a larger N might lead to better results at a fixed computational time horizon. Often, people set N equal to the number of data points received, but in practice, significantly less particles may be used. As a sanity check, we performed two experiments to provide insight. First, we computed a likelihood estimate for a range of parameter values for $\theta = \{\mu, \sigma, r, \phi\}$ of a 2-state HSMM for 1000 data points. μ and σ represent the parameter for a Normal distribution given a latent state, r and ϕ are the parameter of a Negative Binomial duration distribution. The transition distribution has no unknown parameter in the 2-state HSMM case, as the diagonal elements of the transition matrix are separately modeled by the duration. The true parameter can be seen as vertical grey lines on each subplot in Figure A.1, which shows the log likelihood estimate for a range of each parameter that was chosen based on the 95% Credible Interval (CI) of a PMCMC run in Section 2.4, keeping all other parameter fixed. It can be seen that the variance is reasonably similar for $N = 500$, 1000 or 2000. Second, we directly examined the particle filter performance during a PMCMC run for the model defined above. The Partial Autocorrelation Function (PACF) plot of the log likelihood PF estimates for a varying number of particles can be seen in Figure A.2. As there is little difference in the likelihood estimate variance between PFs with 500 and 2000 particles and the PACF look reasonably similar for the same particle range, we will use $N = \frac{T}{2}$ for our analysis going forward.

PMCMC tuning

Once a particle filter is designed, only a suitable MCMC kernel has to be chosen. As discussed in Section 2.3, more advanced gradient based MCMC sampler do not work in the Particle Metropolis Hastings case, as $p_\theta(e_{1:T})$ typically cannot be evaluated pointwise. Thus, we will use Particle Gibbs with ancestors sampling and target $\theta^\star \sim p_\theta(\theta^\star \mid s_{1:T}, e_{1:T})$ in the MCMC step. As gradients for $p_{\theta^\star}(e_{1:T} \mid s_{1:T})$ can easily be calculated in this case, we choose the NUTS MCMC variant (Hoffman and Gelman, 2014, Betancourt, 2016) as MCMC kernel, as other kernels often take significantly more proposal steps to move toward the typical set. This is especially relevant for PMCMC on our model, as we will run a comparatively expensive particle filter after each MCMC proposal.

SMC2 tuning

As described in pseudo-algorithm 8, SMC2 has a particle filter and a PMCMC algorithm assigned for each particle. These particles may be propagated in parallel, so the number of SMC2 are typically chosen to be a multiple of the available computer cores. Initial model parameter drawn from the prior distributions, and the associated PF and PMCMC algorithm will be initiated based on the starting data of length $t_0 << T$. Another tuning parameter is the number of jittering steps in the resampling step. As a guideline, we will continue jittering until the maximum parameter correlation is below 75%.

2.3.5 Model Selection

Once parameter are estimated, how should the performance of a model be evaluated? A powerful method for comparison is to validate models based on their marginal likelihood $p(e_{1:T}) = \int p(e_{1:T}, \theta)\, d\theta = \prod_{t=1}^{T} p(e_t \mid e_{1:t-1})$. This distribution is typically intractable or very costly to evaluate, but during an SMC2 run, an estimate for $p(e_t \mid e_{1:t-1})$ can be obtained at practically no extra cost at each time step. In the particle filter propagation step in algorithm 8, the incremental weight given current model parameter θ_m is computed via

$$\hat{p}_{\theta_n}(e_t \mid e_{1:t-1}) = \frac{1}{M} \sum_{m=1}^{M} \alpha_{t,\theta_n}(s_{1:t}^m, e_{1:t}), \tag{2.4}$$

where α_{t,θ_n} is defined as in Section 2.3.1 and M is the number of particles that are used for the particle filter associated to continuous parameter vector θ_n. After all particles have been propagated forward, a Monte Carlo estimator for $p(e_t \mid e_{1:t-1})$ is obtained by weighting these likelihood increments from equation (2.4) with the corresponding normalized particle weight $w_n \propto w_{n-1}\hat{p}_{\theta_n}(e_t \mid e_{1:t-1})$ associated to parameter θ_n,

$$\hat{p}(e_t \mid e_{1:t-1}) = \sum_n w_n \times \hat{p}_{\theta_n}(e_t \mid e_{1:t-1}).$$

Moving forward, we refer to $p(e_{t+1} \mid e_{1:t})$ as (one step ahead) Predictive Likelihood (PL) at $t+1$, PL_{t+1}, see Kastner (2016). After the final iteration, the marginal likelihood estimate can then be computed as $\hat{p}(e_{1:t}) = \prod_{i=1}^{t} \hat{PL}_t$. Based on the cumulative sums of log PLs, model choice can be performed via the so called Cumulative Log Predictive Bayes Factor (CLPBF). To compare model A and B, for $u > 0$, the CLPBF is defined as

$$CLPBF_{t+1:t+u} = log\left[\frac{p_A(e_{t+u} \mid e_{1:t})}{p_B(e_{t+u} \mid e_{1:t})}\right] = \sum_{i=t+1}^{u} log\left[PL_i(A) - PL_i(B)\right].$$

A positive CLPBF indicates evidence in favor for model A and, if $t = 0$ and $u = T$, this factor is known as log Bayes Factor. Note that in Section 2.3.4, we mentioned that we usually use $t_0 > 1$ data points to initialize the jitter kernels and then record all future PL increments going forward, hence the resulting estimate will be the slightly different $\hat{p}(e_{t_0+1:T} \mid e_{1:t_0}) = \prod_{t=t_0+1}^{T} \hat{PL}_t$. The t_0 data points can be seen as training data for the jitter kernels to be initialized in a reasonable parameter region.

2.4 Simulation and Experimental Results

This section consists of simulation experiments conducted to study the performance of the PF, the PMCMC and the SMC2 algorithm on Hidden Semi-Markov Models. We first generate 1000 data points from a HSMM with Negative Binomial state duration distributions. A plot with sampled observed and latent data is shown in figure A.3, which also depicts a PF estimate of the latent parameter for known continuous model parameter. The advantages of explicitly modelling duration are visible on the third subplot, in which durations in each state vary drastically. The last sub-section shows how the number of hidden states in the data can be estimated using SMC2.

2.4.1 Model Dynamics and Prior Assignments

The model consists of parameter: $\theta = \{\mu, \sigma, p, r, \phi\}$, where the data $e_t \sim N(\mu_{s_t}, \sigma_{s_t})$ has a normal distribution given the latent state. The latent states have the same dynamics as explained in Section 2.2, latent state and duration

$$s_t \sim \begin{cases} \delta(s_t, s_{t-1}) & d_{t-1} > 0 \\ Categorical(p_{s_{t-1}}) & d_{t-1} = 0 \end{cases}, \; d_t \sim \begin{cases} \delta(d_t, d_{t-1} - 1) & d_{t-1} > 0 \\ NegativeBinomial(r_{s_t}, \phi_{s_t}) & d_{t-1} = 0 \end{cases}.$$

The μ parameter have truncated Normal priors with equal variance and different means, $\mu_1 \sim Normal_{(-100,0)}(\mu = -2, \sigma = 10^5)$, $\mu_2 \sim Normal_{(0,100)}(\mu = 2, \sigma = 10^5)$. We assigned a truncated Normal prior for the variances $\sigma \sim Normal_{(0,10)}(\mu = 2, \sigma = 10^5)$ and for the Negative Binomial parameter, $r \sim Normal_{(0,100]}(\mu = 10, \sigma = 10^5)$. The second duration distribution parameter, ϕ, has equal mass from 0 to 1, $\phi \sim Beta(\alpha = 1, \beta = 1)$. Similarly, we assigned a Dirichlet prior for the transition probabilities, p, that favors equal weights, $p \sim Dirichlet(\alpha_1 = \alpha_2 = ... = \alpha_k = k)$, where $k =$ number of latent states.

2.4.2 Estimation

If parameter θ are unknown, both the PMCMC and SMC2 machinery can be used. As the latter builds on the former, we first show estimation results for the Particle MCMC run. The traceplots of four chains for the continuous model parameter can be seen in figure A.4, and for the latent state sequence in figure A.5. After parameter are initialized from the prior distributions, they rapidly converge to the values used to generate the sample data. Common MCMC output statistics are summarized in Table 2.1. SMC2 results can be seen in figure A.6 for the model parameter and in figure A.7 for the latent variables. Just as in the PMCMC case, samples converge fast toward the typical set. Common MCMC output statistics of the final SMC2 iteration for the continuous model parameter are displayed in Table 2.2.

θ	**True**	**Mean**	**MCSE**	**SD**	**Rhat**	**Q2.5**	**Q25.0**	**Q50.0**	**Q75.0**	**Q97.5**
μ_1	-2.0	-2.03	0.01	0.39	1.00	-2.50	-2.18	-2.0	-1.85	-1.55
μ_2	2.0	1.78	0.00	0.22	1.00	1.59	1.72	1.78	1.84	1.96
σ_1	4.0	4.01	0.00	0.27	1.00	3.72	3.91	4.00	4.11	4.32
σ_2	2.0	2.07	0.00	0.17	1.00	1.93	2.014	2.06	2.11	2.23
r_1	10.0	13.64	0.01	4.06	1.00	5.46	10.64	14.01	17.09	19.72
r_2	15.0	13.18	0.23	8.04	1.00	4.12	8.03	11.26	15.99	32.96
ϕ_1	0.3	0.37	0.00	0.08	1.00	0.18	0.32	0.38	0.43	0.48
ϕ_2	0.3	0.25	0.00	0.09	1.00	0.10	0.18	0.24	0.31	0.48

Table 2.1 Posterior output statistics of 4 PMCMC chains for a HSMM on simulated data in Section 2.4. 2000 iterations were run with 1000 iterations burnin, resulting in 4000 total samples. Initial parameter have been sampled from the prior distribution.

θ	**True**	**Mean**	**MCSE**	**SD**	**Q2.5**	**Q25.0**	**Q50.0**	**Q75.0**	**Q97.5**
μ_1	-2.0	-2.04	0.01	0.42	-2.86	-2.28	-2.05	-1.82	-1.06
μ_2	2.0	1.7	0.00	0.17	1.42	1.63	1.73	1.83	2.10
σ_1	4.0	4.03	0.00	0.22	3.61	3.90	4.02	4.16	4.48
σ_2	2.0	2.00	0.00	0.18	1.42	1.94	2.02	2.1	2.30
r_1	10.0	9.38	0.19	5.99	0.09	4.94	9.68	14.19	19.40
r_2	15.0	23.68	0.63	22.03	1.00	8.50	16.03	31.50	87.74
ϕ_1	0.3	0.26	0.00	0.13	0.02	0.16	0.27	0.36	0.48
ϕ_2	0.3	0.41	0.00	0.21	0.06	0.24	0.37	0.55	0.88

Table 2.2 Posterior output statistics for 100 SMC chains on a HSMM at final iterations on simulated data in Section 2.4.

2.4.3 Determining the number of states

Before estimating model parameter for a given data set, the researcher has to choose the number of hidden states in a discrete SSM, which is often challenging a priori. To tackle this problem, we choose methods from the overfitting mixtures literature, see Rousseau and Mengersen (2011), Fruehwirth-Schnatter (2006), Fruehwirth-Schnatter et al. (2018). In particular, Rousseau and Mengersen (2011) show that the posterior distribution of a mixture model is much more stable if the prior weights of the mixture components are concentrated on the boundary regions of the parameter space. We borrow this concept for SSMs and show that by assigning appropriate prior weights on the transition matrices of a HSMM, we can choose more states than necessarily describe the data and still have an interpretable structure, as superfluous latent states will never be visited during the estimation process. As an experiment, we fit a 2-, 3-, and 5-state HSMM to data that was generated by a 3-state HSMM. The parameter used to generate the data can be seen in Table 2.3, and the corresponding data in plot 2.2. In Section 2.3, we mentioned that the Predictive Likelihood can be estimated as a by-product of the inference procedure in SMC2. Hence, we tracked the cumulative log PL for each model at each iteration, and plotted the results in figure 2.2. It can be seen that the 3- and 5-state HSMMs Predictive Likelihood perfectly align, and we conclude that it is possible to determine the number of states via SMC2, and even check for differences during the data propagation. All estimates for the model parameter can be seen in Table 2.3. In this table, state 2 and 4 are almost never visited and act as superfluous regimes. The remaining state estimates contain the correct parameter in the 95% Credible Intervals.

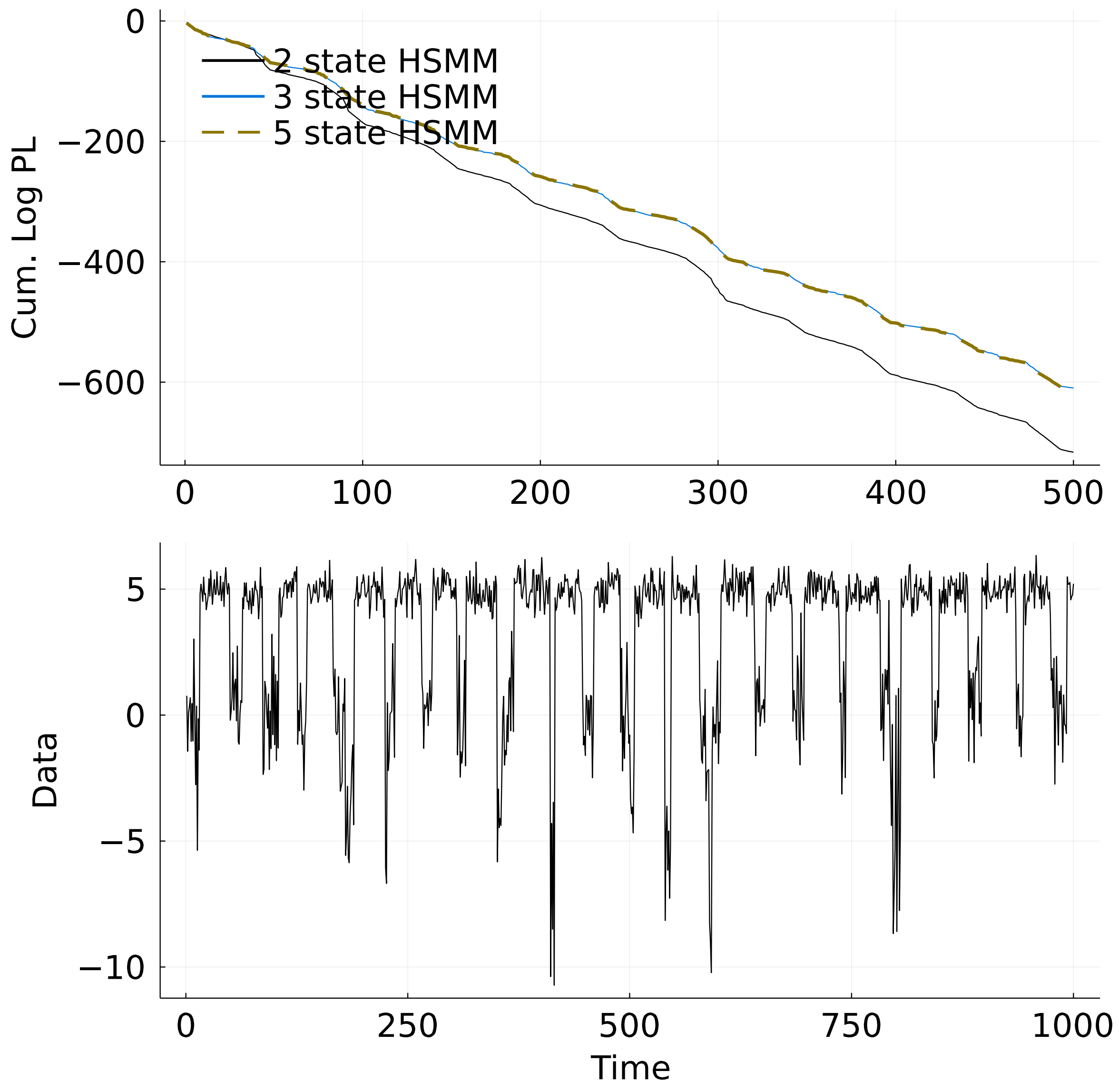


Figure 2.2 Cumulative log Predictive Likelihood of 2-, 3- and 5-state HSMM, as discussed in Section 2.3. The underlying data has been generated by a 3-state HSMM, and the 5-state HSMM returns PL levels as the HSMM that generated the data, while the 2-state HSMM is not flexible enough to detect the data structure. Hence, we can "overfit" the number of states and let the algorithm decide the number of states itself in case no prior knowledge is available. The bottom plot shows the whole data sequence, while the bottom graph depicts the cumulative log PL as defined in Section 2.3.5 of the last 500 iterations.

θ	**True**	**Mean**	**MCSE**	**SD**	**Q2.5**	**Q25.0**	**Q50.0**	**Q75.0**	**Q97.5**
μ_1	-5.0	-4.73	0.03	1.48	-8.21	-5.43	-4.66	-3.97	-1.6
μ_2	-	-1.5	0.1	4.18	-8.8	-4.46	-1.04	0.09	8.64
μ_3	0.0	-0.44	0.07	2.06	-6.06	-0.19	-0.05	0.07	2.93
μ_4	-	-1.56	0.11	5.36	-9.25	-5.65	-3.19	2.56	9.35
μ_5	5.0	4.95	0.0	0.03	4.9	4.94	4.96	4.97	5.01
σ_1	2.5	2.95	0.03	1.38	1.13	2.26	2.7	3.2	7.69
σ_2	-	3.27	0.08	2.32	0.76	1.56	2.42	4.11	9.24
σ_3	1.5	1.92	0.05	1.33	1.27	1.46	1.55	1.67	6.9
σ_4	-	4.43	0.06	2.68	0.46	2.4	3.59	6.68	9.62
σ_5	0.5	0.52	0.0	0.06	0.45	0.49	0.51	0.53	0.69
λ_1	5.0	4.53	0.04	1.7	1.38	3.4	4.46	5.53	8.36
λ_2	-	10.14	0.17	6.71	1.55	5.08	9.33	12.23	27.51
λ_3	10.0	10.76	0.1	3.58	3.16	9.89	10.8	11.65	21.89
λ_4	-	11.72	0.23	8.77	1.36	4.01	8.9	18.78	29.02
λ_5	30.0	30.92	0.04	1.59	27.72	29.84	30.98	32.03	33.89
$p_{1,1}$	-								
$p_{1,2}$	-	0.15	0.01	0.22	0.0	0.0	0.05	0.26	0.77
$p_{1,3}$	0.2	0.27	0.01	0.26	0.0	0.05	0.25	0.46	0.87
$p_{1,4}$	-	0.08	0.0	0.18	0.0	0.0	0.01	0.09	0.74
$p_{1,5}$	0.8	0.50							
$p_{2,1}$	-	0.19	0.01	0.25	0.0	0.01	0.07	0.27	0.9
$p_{2,2}$	-								
$p_{2,3}$	-	0.2	0.01	0.26	0.0	0.0	0.07	0.31	0.92
$p_{2,4}$	-	0.13	0.01	0.23	0.0	0.0	0.02	0.14	0.89
$p_{2,5}$	-								
$p_{3,1}$	0.2	0.17	0.0	0.17	0.0	0.03	0.13	0.26	0.59
$p_{3,2}$	-	0.1	0.01	0.16	0.0	0.0	0.03	0.14	0.59
$p_{3,3}$	-								
$p_{3,4}$	-	0.07	0.0	0.14	0.0	0.0	0.01	0.07	0.49
$p_{3,5}$	0.8	0.67							
$p_{4,1}$	-	0.23	0.01	0.29	0.0	0.01	0.08	0.37	0.95
$p_{4,2}$	-	0.22	0.01	0.29	0.0	0.0	0.07	0.36	0.94
$p_{4,3}$	-	0.24	0.01	0.29	0.0	0.01	0.11	0.4	0.94
$p_{4,4}$	-								
$p_{4,5}$	-								
$p_{5,1}$	0.2	0.2	0.0	0.15	0.0	0.07	0.19	0.3	0.51
$p_{5,2}$	-	0.18	0.01	0.26	0.0	0.0	0.04	0.24	0.82
$p_{5,3}$	0.8	0.57	0.01	0.28	0.0	0.47	0.67	0.78	0.89
$p_{5,4}$	-								
$p_{5,5}$	-								

Table 2.3 Parameter estimates for 5-state HSMM. The first column depicts the parameter for the 3 state HSMM that were used to generate the data. State 2 and 4 are hardly visited, and are the superfluous states. $p_{i,j}$ denotes the transition probability from state i to j. $p_{i,i}$ is separately modeled by the duration distribution. The transition probabilities p form a simplex, so the probability to transition to the final state is determined by all other states.

2.5 Applications on the VIX Times Series Data

The section contains a case study on model selection and prediction for a financial data set. Moreover, we show how SSMs can be used for clustering purposes.

2.5.1 Prediction and Model Selection on Financial Volatility

In this Section, we are modeling the VIX directly via HSMMs and other popular benchmark models. The VIX is derived from options with near-term expiration dates on the US major equity S&P 500 index, and is a popular indicator for future short term volatility expectations. Volatility is often modeled indirectly based on (log) stock prices via Stochastic Volatility (SV) models and its autoregressive nature is well recognized, see e.g. (Hull and White, 1987, Kim et al., 1998, Kastner, 2016). We incorporate this behaviour in our model by assigning autoregressive weights to the location parameters in each latent regime. The corresponding model can be viewed as a separate AR(1) model in each state, where a latent variable following a semi-Markov chain governs the regime changes and durations. Multiple duration distribution choices, such as the negative Binomial and Poisson distribution, are tested and benchmarked against HMM and AR(1) models.

For real data collection, we first obtain 1000 daily end-of-day data points of this instrument, which can be seen in figure A.8, as of January 1st 2022 from the Thomson Reuters database. Notably, the Covid-19 epidemic is included in the data, which causes the evidence to be much more volatile from the start of 2020 and to transition to a different regime from there onward. Common ideas to address this issue are to assign change points across the times series before estimating the model parameter. Such methods have an easily interpretable structure, but lack the information from a stochastic process governing the model dynamics, which enhances inference capabilities for the data. For example, due to the discrete state space formulation of our proposed model, parameter interpretation is straightforward, while the HSMM dynamics allow to incorporate significantly more decision-making tools, such as regime change and duration forecasting.

Model Dynamics and Prior Assignments

The AR(1) HSMM with Negative Binomial duration consists of parameter: $\theta = \{\mu, \sigma, w, p, r, \phi\}$, where the data $e_t \sim N(w_{s_t} \times e_{t-1} + \mu_{s_t}, \sigma_{s_t})$ has a normal distribution given the latent state and the previous data point. The latent states have the same dynamics as explained in Section 2.2, latent state and duration

$$s_t \sim \begin{cases} \delta(s_t, s_{t-1}) & d_{t-1} > 0 \\ Categorical(p_{s_{t-1}}) & d_{t-1} = 0 \end{cases}, \; d_t \sim \begin{cases} \delta(d_t, d_{t-1} - 1) & d_{t-1} > 0 \\ NegativeBinomial(r_{s_t}, \phi_{s_t}) & d_{t-1} = 0 \end{cases}.$$

The μ parameter have truncated Normal priors with equal variance and means for each state, $\mu_i \sim Normal_{(0,1)}(\mu = 0.2, \sigma = 10^5)$. Similarly, we assigned a truncated Normal prior for the variances $\sigma_i \sim Normal_{(0,1)}(\mu = 0.2, \sigma = 10^5)$ and for the Negative Binomial parameter, $r_i \sim Normal_{(0,100]}(\mu = 10, \sigma = 10^5)$. The second duration distribution parameter ϕ has equal mass from 0 to 1, $\phi_i \sim Beta(\alpha = 1, \beta = 1)$. Similarly, we assigned a Dirichlet prior for the transition probabilities p that favors equal weights, $p \sim Dirichlet(\alpha_1 = \alpha_2 = ... = \alpha_k = k)$, where k = number of latent states. The autoregressive parameter, w, is bounded between -1 and 1 by assigning a truncated prior, $w_i \sim Normal_{(-1,1)}(\mu = 0, \sigma = 10^5)$. The AR(1) HSMM with Poisson duration only differentiates with respect to the duration parameter, $\theta = \{\mu, \sigma, w, p, \lambda\}$. In this case, the latent duration has a Poisson distribution,

$d_t \sim \begin{cases} \delta(d_t, d_{t-1} - 1) & d_{t-1} > 0 \\ Poisson(\lambda_{s_t}) & d_{t-1} = 0 \end{cases}$, where $\lambda_i \sim Normal_{(0,100]}(\mu = 20, \sigma = 10^5)$. The AR(1) HMM with parameter $\theta = \{\mu, \sigma, w, p\}$, has the same data dynamics and latent state dynamics $s_t \sim Categorical(p_{s_{t-1}})$. All prior configurations are assigned from the previous models. The AR(1) Model with parameter $\theta = \{\mu, \sigma, w\}$ does not have latent variables, and data dynamics $e_t \sim Normal(w \times e_{t-1} + \mu, \sigma)$. The priors for μ are set to $\mu \sim Normal_{(0,10)}(\mu = 2.0, \sigma = 10^5)$. σ and w have the same prior as the other models.

Results

A detailed methodology for model comparison can be found in section 2.3.5. The SMC2 machinery with the tuning configurations discussed in Section 2.3 is used to estimate model parameter. The Cumulative Log Predictive Bayes Factor as discussed in Section 2.3 is shown in figure 2.3, which contains diagnostics for ARHSMMs and ARHMMs with a different number of states. We kept increasing the number of latent states until the cumulative Predictive Likelihood started to decrease. As can be seen in table 2.5, which contains diagnostics at the final time index, the 3 state models performed best within each model family, and the Negative Binomial duration distribution performed best among all models. Figure 2.3 shows the CLPBF of this model against all other benchmarks, which is positive and in favor of the proposed model consistently over time against all other models. Results for this particular model can be seen in Figure A.9 and Table 2.4 for a model parameter summary. The traceplots show all model parameter estimates at each point in time with the corresponding 95% credible interval.

There is a clear distinction between a lower and higher volatility state. Moreover, an additional regime is introduced at the time COVID-19 makes significant headlines in the global markets with much larger volatility estimates that either of the other latent regimes. During this period, a clear distinction between the different regimes forms, which is captured very fast during the estimation process, and would be impossible for standard batch estimation methods. Figure A.10 shows the filtered state trajectory of the latent variables at each time step. A re-scaled posterior mean of the latent variable at each time index is shown against the real data in the bottom sub-plot, which displays the clear distinction between two volatile and a stable state in the times series. There is more variation at the initial stages before more data is added to the algorithm. Figure A.11 depicts model predictions against the realized data at each time step.

Parameter	**Mean**	**MCSE**	**StdDev**	**Q2.5**	**Q25.0**	**Q50.0**	**Q75.0**	**Q97.5**
μ_1	0.36	0.2	0.01	0.11	0.2	0.3	0.47	0.88
μ_2	0.23	0.05	0.0	0.1	0.21	0.25	0.28	0.3
μ_3	0.13	0.17	0.01	0.02	0.05	0.07	0.09	0.69
σ_1	0.17	0.06	0.0	0.05	0.14	0.17	0.2	0.28
σ_2	0.09	0.02	0.0	0.05	0.08	0.09	0.1	0.15
σ_3	0.06	0.03	0.0	0.04	0.05	0.05	0.05	0.19
w_1	0.92	0.07	0.0	0.76	0.88	0.94	0.98	1.0
w_2	0.92	0.02	0.0	0.89	0.9	0.91	0.93	0.97
w_3	0.95	0.06	0.0	0.78	0.96	0.97	0.98	0.99
$p_{1,2}$	0.72	0.2	0.01	0.18	0.62	0.77	0.87	0.97
$p_{2,1}$	0.49	0.23	0.01	0.07	0.3	0.52	0.68	0.89
$p_{3,1}$	0.46	0.29	0.02	0.03	0.18	0.46	0.71	0.93
r_1	4.99	5.4	0.15	0.03	0.66	2.74	7.83	18.35
r_2	6.12	5.46	0.15	0.29	1.59	4.23	9.56	18.89
r_3	4.75	4.33	0.12	0.36	1.72	3.21	6.25	17.13
ϕ_1	0.68	0.3	0.01	0.07	0.45	0.79	0.95	1.0
ϕ_2	0.56	0.28	0.01	0.05	0.34	0.62	0.8	0.94
ϕ_3	0.24	0.18	0.01	0.04	0.12	0.19	0.31	0.74

Table 2.4 Posterior output statistics for 100 SMC chains on ARHSMM in chapter 2.5 at final iterations in SMC2 run. $p_{i,j}$ denotes the transition probability from state i to j. $p_{i,i}$ is separately modeled by the duration distribution. The transition probabilities p form a simplex, so the probability to transition to the final state is determined by all other states. r and ϕ are parameters for the Negative Binomial duration probabilities. w are the autoregressive coefficients for the ARHSMM.

Once parameter have been estimated, the latent state trajectory estimates can be used to cluster the log VIX data into different regimes. A re-scaled state posterior

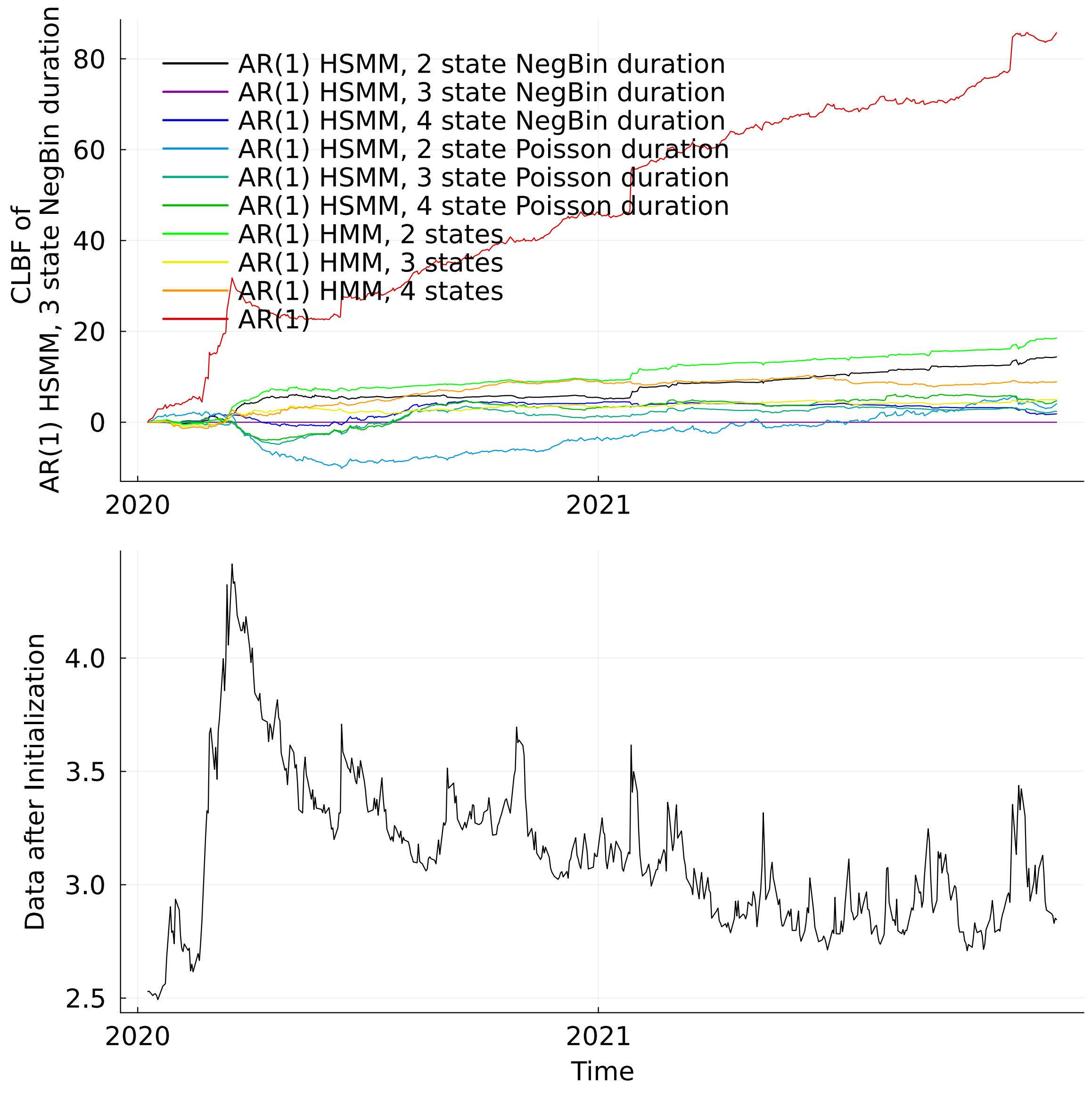


Figure 2.3 The top plot depicts the Cumulative Log Predictive Bayes Factor as defined in Section 2.3.5 of the winning model in Section 2.5 at each iteration. At the bottom, the corresponding log VIX index data is shown over time.

Names	**Cum Log PL.**	**Difference to Winner**
AR(1) HSMM, 2 state NegBin duration	573.25	14.39
AR(1) HSMM, 3 state NegBin duration	587.64	0.00
AR(1) HSMM, 4 state NegBin duration	585.80	1.84
AR(1) HSMM, 2 state Poisson duration	583.55	4.09
AR(1) HSMM, 3 state Poisson duration	585.23	2.41
AR(1) HSMM, 4 state Poisson duration	582.92	4.73
AR(1) HMM, 2 states	569.09	18.55
AR(1) HMM, 3 states	582.58	5.06
AR(1) HMM, 4 states	578.74	8.90
AR(1)	501.88	85.76

Table 2.5 Cumulative log Predictive Likelihood as defined in Section 2.3.5 for various discrete state space models fitted via SMC2 on data described in chapter 2.5.

mean of the latent trajectory at the last SMC2 iteration against the actual data can be seen at the top plot in Figure 2.4. State 1 corresponds to a short duration state with jumps and drastic changes in levels, while state 2 and 3 corresponds to the more typical high and low volatility regimes that are observed for the majority of times. Based on the posterior mean, we clustered the changes in the index for each state, which can be seen in the bottom plot of Figure 2.4. Here, state 1 is clearly associated with drastic movements in either direction. To compare the ARHMM clusters with the results obtained from our proposed model, we generated a plot that showcases both models side by side in Figure A.12. Upon examination, it becomes evident that the HMM variant exhibits significantly shorter durations in the high volatility state. Instead, it predominantly switches back and forth between the most frequent state, which accounts for the lower marginal likelihood observed in the tests. An intriguing observation is that the separation of the low- and medium volatility states appears to be based on scale in the Autoregressive Hidden Markov Model (ARHMM), whereas in the Autoregressive Hidden Semi-Markov Model (ARHSMM), the separation occurs on a scale basis.

2.6 Conclusions

In this Chapter, we discussed sequential parameter estimation techniques for State Space Models with a focus on Hidden Semi-Markov Models. We compared the forecasting accuracy of various models on financial data and concluded that the HSMM has a superior predictive performance against other popular discrete SSMs. Moreover,

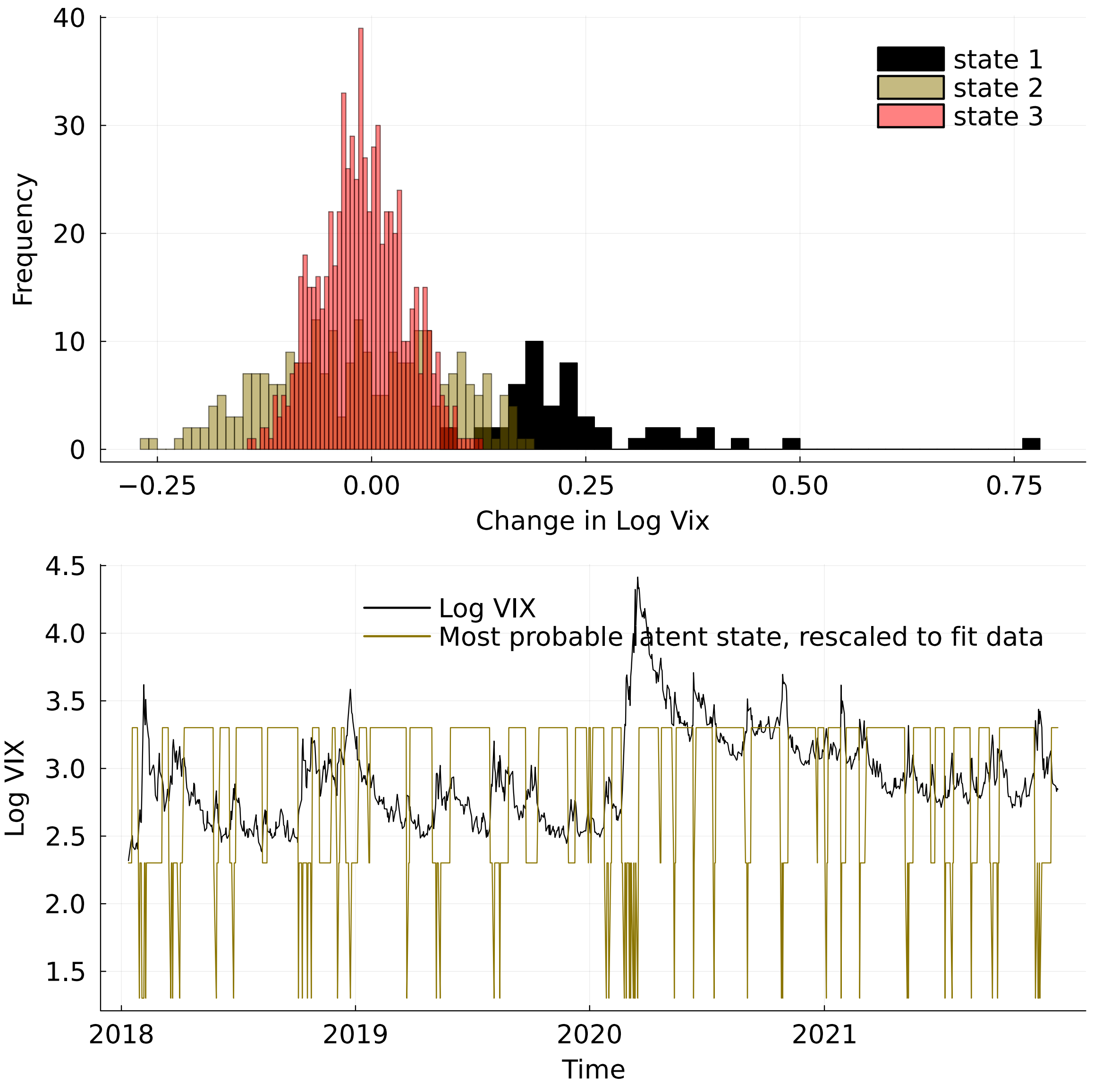


Figure 2.4 The top plot depicts a histogram for changes in log VIX data, conditioned on the most probable posterior latent state from the final SMC2 iteration. The bottom graph displays the log VIX data over time (black) alongside the most probable latent state, rescaled to the underlying data.

we demonstrated how by-products emerging from a SMC2 estimation run can be used to determine the number of latent regimes governing such models. While the additional duration variable in the HSMM typically causes parameter inference to be more challenging due to the increased computational costs of the likelihood function, it adds significantly more flexibility in modelling the latent process. Our proposed inference technique has the same computational costs for both the basic HMM and the HSMM and is particularly suitable for sequential data. As for future research topics, more optimized techniques to adaptively select the number of particles in a PF may lead to faster runs and improved mixing for both the PMCMC and SMC2 algorithm. Furthermore, as tuning the individual MCMC and PF kernels was handled independently during the SMC2 runs, adding information from all chains may drastically increase the tuning process for the individual jitter kernels.

2.7 Software

The data and code used to run the algorithms in this Chapter can be be seen in https://github.com/paschermayr/Publish_SequentialHSMM. For more detailed information about the implementations for running all algorithms and computing all tables can be found in https://github.com/paschermayr/Baytes.jl and its sub-libraries. The corresponding plots are defined in https://github.com/paschermayr/BaytesInference.jl.

The most important step a man can take. It's not the first one, is it? It's the next one.

Brandon Sanderson (*The Way of Kings*, 2010)

Chapter 3

SIR-type State Space Models with Piecewise Constant Transmission Rates

The SARS-CoV-2 pandemic has seen multiple resurgences due to evolving virus variants, making it difficult to analyze these kinds of phenomena with classic epidemic models, and thus challenging decision makers to successfully counteract new infection waves in time. In this Chapter, we propose an epidemic model with the transmission rate between susceptible and infected individuals, β, being time varying and piecewise constant. At any point in time, β is linked to a latent variable that follows a Hidden Semi-Markov Model (HSMM). This HSMM-driven epidemic Model (HSMM-EM) structures the data into multiple regimes, which greatly enhances decision-making capabilities, while the limited number of continuous model parameter guarantees straightforward model interpretation. In a case study on COVID-19 numbers in the United Kingdom, reported fatalities as well as infections are used in the observation model specification to account for the uncertainty in either of the reporting methodologies. We show that this is preferable to using only fatalities based on Bayesian model choice derived from Particle MCMC (PMCMC) and Sequential Monte Carlo (SMC) runs.

3.1 Introduction

Since the emergence of the SARS-CoV-2 virus in late 2019, policy makers find it challenging to set intervention policies to prevent COVID-19 disease outbreaks in time. Due to the high reproduction rate of the SARS-CoV-2 virus, it is of utmost importance to monitor the current number of infections in an economy in order to avoid the virus to spread to uncontrollable levels and cause havoc on both citizens and the economy. Petrosillo et al. (2020) and Liu et al. (2020) provide estimates for various coronavirus spread rates, placing even the early SARS-CoV-2 variants among the top. The standard techniques to model such phenomena are known as compartmental models, which assign the target population to different departments. For example, the well-known Susceptible-Exposed-Infected-Recovered (SEIR) model, see e.g. (Cohen, 1992, Diekmann et al., 2012), divides the population into susceptible, exposed, infectious, or recovered individuals. A major advantage of this class of models is the ease of interpretability for the model parameter and the established approaches on government intervention policy analysis.

An important factor that can be computed based on the parameter from a SEIR style model is known as Effective Reproductive Number (R_t), the number of secondary infections that one infected person would produce through the entire duration of the infectious period, $R_t = \frac{\beta S(t)}{\gamma N(t)}$. In this case, N is the total population. In the standard SEIR model case, the β parameter, the transmission rate between susceptible and infected individuals, is kept constant across the time horizon. However, in the SARS-CoV-2 case, it has been abundantly clear that this parameter is time varying, see, e.g., Flaxman et al. (2020b). Consequently, traditional SEIR type models are ill-suited to provide accurate parameter estimates and predictions for future COVID-19 outbreaks. Common ideas to alleviate this problem is to assign multiple β parameters across the time horizon. Most propositions evolve around either assigning change points to different β parameters or using continuous latent states, i.e. a Geometric Brownian Motion (GBM), that are linked to this parameter, see, e.g., Chatzilena et al. (2022). The former methods are easy to interpret but have no stochastic process governing the dynamics and no memory associated to the parameter. This makes interpretation of intervention impact and prediction more difficult. The latter method does have these attributes, but model parameter, in particular the GBM part, are more challenging to interpret.

We propose a new class of SEIR type models that accommodates both features by having a time-varying, piecewise constant β that is dependent on a latent process that follows a Hidden Semi-Markov Model, a flexible State Space Model (SSM) that is an extension of the popular Hidden Markov Model (HMM). Piecewise constant β parameter are easier to interpret and base decisions on. They are a natural progression to change-points and more parsimonious than continuous latent state trajectories. Moreover, using a state space model formulation instead of change points has the significant advantage that at any point in time, transition probabilities of regime changes can be computed, which provides significant more information for decision-making. We will denote all SEIR type models with such dependencies HSMM-driven epidemic Model (HSMM-EM) from here on onwards. However, attaching a latent process to the transmission rate between susceptible and infected individuals significantly increases the computational complexity of the proposed model. In order to estimate the discrete latent state trajectory, we design a tailored Particle Filter (PF) that is also used for batch estimation of all model parameter via a Particle MCMC kernel. Additionally, a Sequential Monte Carlo Squared (SMC2) algorithm is used for sequential estimation. Note that estimating the latent state trajectory involves significant computational challenges as the memory of the proposed model is much higher than for typical SSMs.

Our main contributions in this Chapter are formulating and efficiently estimating a novel discrete State Space Model for epidemic models that is more flexible than its alternatives but still convenient to interpret. This includes designing a Particle Filter that can be used for SSMs that have Ordinary Differential Equation (ODE)s at each transition in the PF propagation step. We show that this model performs excellently in terms of predictive accuracy on reported COVID-19 infections and fatalities in the United Kingdom and identify the number of latent regimes for the provided data. Last but not least, we show that combining both reported fatalities with the reported infections increases predictive performance versus models that only use reported fatalities as data source.

This Chapter is organized as follows: Section 3.2 formally describes the proposed Susceptible-Exposed-Exposed-Infected-Infected-Recovered (SEEIIR) model in more detail. In Section 3.3, we present the methodology that is used in this Chapter for parameter estimation and model comparison. Section 3.4 explores the validity of the developed methods via simulation based experiments. In Section 3.5, we perform a case study on COVID-19 fatalities and infections in the United Kingdom. Finally, Section 3.6 concludes with some relevant discussion.

3.2 Model Specification

In our framework, model implied infections and fatalities are linked to the corresponding reported data. Throughout this Chapter, we will use a superscript r for the reported data, and a superscript i for the model implied data. The model implied cases are obtained from the following Ordinary Differential Equation:

$$\begin{aligned}
\frac{\mathrm{d}S_t}{\mathrm{d}t} &= -\beta_t S_t \frac{(I_{1,t} + I_{2,t})}{N} - \rho\nu_{t-U},\\
\frac{\mathrm{d}E_{1,t}}{\mathrm{d}t} &= \beta_t S_t \frac{(I_{1,t} + I_{2,t})}{N} - \epsilon E_{1,t},\\
\frac{\mathrm{d}E_{2,t}}{\mathrm{d}t} &= \epsilon E_{1,t} - \epsilon E_{2,t},\\
\frac{\mathrm{d}I_{1,t}}{\mathrm{d}t} &= \epsilon E_{2,t} - \gamma I_{1,t},\\
\frac{\mathrm{d}I_{2,t}}{\mathrm{d}t} &= \gamma I_{1,t} - \gamma I_{2,t},\\
\frac{\mathrm{d}R_t}{\mathrm{d}t} &= \gamma I_{2,t} + \rho\nu_{t-U},
\end{aligned} \tag{3.1}$$

which is an extension of a standard SEIR Model, see Dureau et al. (2012). The total population N in our model is divided into four groups: Susceptible (S), Exposed (E), Infected (I), and Recovered (R). New infections occur at a rate of $\beta_t S_t \frac{(I_{1,t}+I_{2,t})}{N}$. This implies that susceptible individuals make effective contacts at a rate of β, which represents the transmission rate between susceptible and infected individuals. The parameters ϵ and γ indicate the average waiting times for individuals in the Exposed and Infected compartments, respectively. These waiting times are given by $\frac{1}{\epsilon}$ and $\frac{1}{\gamma}$. ν_t represents the number of individuals who have received their first COVID-19 vaccination at time t. We set U, the number of days until the vaccination is in full effect, to 45 and assume a constant vaccine efficiency ratio ρ of 0.5. The introduction of additional compartments, namely E_2 and I_2, allows for more realistic waiting time periods, as described in Wearing et al. (2005). One crucial distinction in our model is that the transmission rate parameter between susceptible and infected individuals, β, is associated with a latent discrete state, making it time-varying and piecewise constant.

In addition to the model parameter stated in Equation (3.1), a full discrete state trajectory of length T, where T is the number of data points, has to be estimated. We use a Particle Filter to sample the full state trajectories $s_{1:T}$ linked to β in the estimation process. Once the ODE has been solved, the model implied cases can be computed. The model implied deaths are a function of the model implied cases, see

Flaxman et al. (2020a), Chatzilena et al. (2022), and are defined as

$$d_t^i = ifr_t * \sum_{\tau=max(1,t-28)}^{t-1} c_\tau^i f_{t-\tau},$$

where d_0^i is set to 0. c are the model implied cases from the ODE, f the distribution of time from infection to death, which is based on the work of Verity et al. (2020). Infection Fatality Ratio (ifr) denotes the probability of death for an infected individual, which is also known as the infection fatality ratio and is based on Levin et al. (2020) and adjusted from findings of Chatzilena et al. (2022). The model implied deaths can occur only up to 28 days after infection, which is the number of days the United Kingdom uses to compute reported cases, see GOV.UK (2022). The observation probability for the reported cases is defined as

$$c_t^r \sim \text{Negative Binomial}_{\mathit{Alternative}} \left(c_t^*, c_t^* + \frac{c_t^{*2}}{\phi_c} \right), \tag{3.2}$$

where $c_t^* = c_t^i * ur_t$ are the model implied cases adjusted by an under-reporting score for the additional noise during the initial period of data gathering. The corresponding values can be seen in Figure A.16. Negative Binomial$_{\mathit{Alternative}}(\mu, \phi)$ is an alternative parameterization of the Negative Binomial distribution with $\mathbb{E}(Y) = \mu$ and $Var(Y) = \mu + \frac{\mu^2}{\phi}$. Similarly, the observation probability for the reported deaths is defined as

$$d_t^r \sim \text{Negative Binomial}_{\mathit{Alternative}} \left(d_t^i, d_t^i + \frac{d^{i2}_t}{\phi_d} \right) \tag{3.3}$$

where d_t^i are the model implied deaths. The model implied deaths are a function of the model implied cases, which in turn are obtained from solving the ODE stated in Equation (3.1). In order to solve these equations, a vector of the piecewise constant β parameter equal to the number of data points is needed. We obtain these by linking each β to a latent variable that follows a HSMM, a flexible extension of the popular HMM that allows the underlying stochastic process to be a semi-Markov chain. A particular advantage of the HSMM is that it allows its state duration distribution, the probability a latent state remains in any current state until it switches to a different state, to be fully customizable, while the HMM counterpart is implicitly geometric. Given the tendency of rapid switching between COVID-19 outbreaks and longer time horizons of lockdowns, HSMMs are better suited for our purpose. In addition, assigning a State Space Model allows us to compute probabilities for regime changes given the current state, which is extremely useful for prediction purposes. The HSMM formulation used throughout this Chapter is known as the Explicit-duration Hidden Markov Model (EDHMM), i.e. a HSMM that explicitly defines the duration distribution. Transitions are allowed only at the end of each state, resulting in the following definition. A hidden semi-Markov model is a bivariate stochastic process $\{e_t, z_t\}_{t=1,2,\ldots}$, where $z_t = \{s_t, d_t\}$ is an unobserved semi-Markov chain and, conditional on z_t, e_t is an observed sequence of independent random variables. The model is fully specified by the transition distribution $f_\theta(s_t \mid s_{t-1}, d_{t-1})$ of s_t

$$s_t \sim \begin{cases} \delta(s_t, s_{t-1}) & d_{t-1} > 0 \\ f_\theta(s_t \mid s_{t-1}, d_{t-1}) & d_{t-1} = 0 \ , \end{cases}$$

the duration distribution h_θ of d_t

$$d_t \sim \begin{cases} \delta(d_t, d_{t-1} - 1) & d_{t-1} > 0 \\ h_\theta(d_t \mid s_t, d_{t-1}) & d_{t-1} = 0 \ , \end{cases}$$

the corresponding initial distribution π_θ of z_t, and the observation distribution g_θ, $e_t \sim g_\theta(e_t \mid s_t)$,

$$e_t \sim g_\theta(e_t \mid s_t).$$

where $\delta(a, b)$ is an indicator function and equals 1 if $a = b$ and 0 otherwise. For additional flexibility, we choose a Negative Binomial distribution to model the state durations. A more detailed review of HSMMs can be seen in see Murphy (2002), Yu (2010, 2016), and additional applications can be found in Bulla and Bulla (2006), Lindsten and Schön (2013), Chopin and Papaspiliopoulos (2020), Corenflos et al. (2021). The general structure of the HSMM-EM can be seen as a Directed Acyclic Graph (DAG) in Figure 3.1.

3.3 Bayesian Inference

3.3.1 Parameter estimation

Inference on the posterior distribution of the model parameter θ given the observed data $e_{1:T}$, $p(\theta \mid e_{1:T}) = \frac{p_\theta(e_{1:T})\ p(\theta)}{p(e_{1:T})}$ is a standard goal in Bayesian parameter estimation. For our particular model, we want to emphasize that, in addition to the model parameter, a latent state trajectory has to be estimated that assigns a piecewise

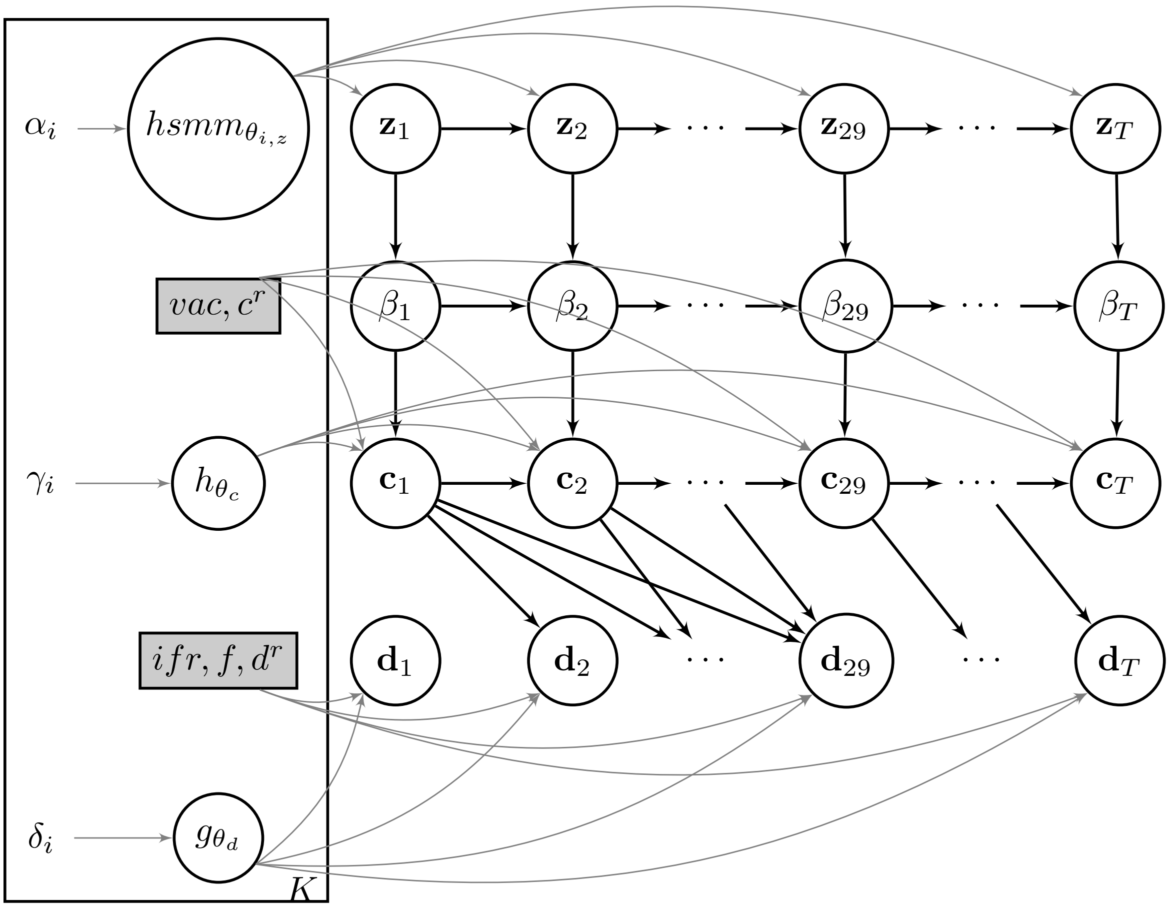


Figure 3.1 A HSMM-driven epidemic Model, parameter θ and hyper-parameter $\{\alpha, \gamma, \delta\}$. z_t follows a Hidden Semi-Markov Model and determines the transmission rate β_t at each time index. The unshaded nodes c_t (cases) and d_t (deaths) denote the unobserved true data at time t. $\theta_{i,z}$ denotes the model parameter for the latent state z in regime i. h denotes all model functions and θ_c all parameter for the model implied cases c, while g denotes all model functions and θ_d all parameter for the model implied deaths d. vac, c_r, d_r denote the reported vaccinations, cases and deaths. ifr denotes the infection-fatality ratio, f the distribution of time from infection to death.

constant transmission rate between susceptible and infected individuals, β_t, that is linked to a latent variable s_t at each time index. Additionally inferring the latent states $s_{1:T}$ is a challenging task as they have to be integrated out in the likelihood computation, $p_\theta(e_{1:t}) = \int p_\theta(e_{1:T}, s_{1:T})\ ds_{1:T}$. This is typically intractable or costly to evaluate and, consequently, usually the full posterior distribution $p(s_{1:T}, \theta \mid e_{1:T}) = \frac{p_\theta(e_{1:T}|s_{1:T})\ p_\theta(s_{1:T})\ p(\theta)}{p(e_{1:T})}$ is inferred.

In this Chapter, we focus on the Particle MCMC machinery, see Andrieu et al. (2010), which is a framework that jointly infers the latent state sequence and model parameter. In a PMCMC step, a PF and a MCMC kernel are used iteratively to sample a state trajectory $s_{1:T}$ and continuous model parameter θ. Typically, for a standard PMCMC sampler like the Particle Metropolis Hastings (PMH) kernel, it is difficult to find a good MCMC kernel $K_{mcmc}(e_{1:T}, \theta)$, because the θ proposal will be accepted based on the Particle Filter likelihood estimate. Additionally, $p_\theta(e_{1:T})$ typically cannot be evaluated pointwise or is at least prohibitively expensive to do so, so gradient based MCMC kernels are unavailable. An alternative PMCMC method is known as Particle Gibbs (PGIBBS). During a PGIBBS step, first a Conditional Particle Filter (CPF) is used to obtain a sample from $\hat{p}_{\theta^\star}(s^\star_{1:T} \mid s^{'}_{1:T}, e_{1:T})$, and then a MCMC kernel is used to obtain a sample for all other continuous model parameter θ. $s^{'}_{1:T}$ is commonly referred as reference trajectory and used to preserve the invariance principle in the slightly altered Particle Filter target distribution. A major advantage of the PGIBBS kernel is that the likelihood function for the MCMC step, $p_\theta(y_{1:T} \mid s_{1:T})$, is usually computable pointwise, so a greater variety of more advanced MCMC kernels can be used. Consequently, we use the PGIBBS variant of this algorithm in order to use more advanced gradient based MCMC algorithm to update the continuous model parameter. A pseudo algorithm for the standard PMCMC variant can be seen in Algorithm 5, and for the PGIBBS variant in Algorithm 6. We refer to Kantas et al. (2015), Doucet and Johansen (2011) for a thorough Particle Filter review, and to Lindsten et al. (2014, 2015) for an introduction to the PGIBBS version. Craiu and Rosenthal (2014) provides a detailed guide on the topic of MCMC. In our computations, we used a fully automated version of the MCMC variant Hamiltonian Monte Carlo (HMC), see Neal (2012), Betancourt (2018), known as No U-Turn Sampling (NUTS) algorithm, which was proposed by Hoffman and Gelman (2014) and later on improved in Betancourt (2016).

While the Particle Filter permits the estimation of the log likelihood function by incrementally evaluating data points, the ODE in Equation (3.1) has to be solved at each time step as the log weights for the particles are based on the sum of Equations (3.2) and (3.3). Note that for most SSMs, the probability of observing data e_t depends only on the current state s_t, and the transition probability for s_t only depends on s_{t-1}. In our model, data $e_t = \{c_t^r, d_t^r\} \sim \text{Neg. Bin.}_{Alt}\left(c_t^*, c_t^* + \frac{c_t^{*2}}{\phi_c}\right) \times \text{Neg. Bin.}_{Alt}\left(d_t^i, d_t^i + \frac{{d^i}_t^2}{\phi_d}\right)$, where ϕ_d is dependent on the past 28 model implied cases, see Section 3.2. This time frame is based on the U.K. methodology to compute the reported deaths after a person has been inflicted with the COVID-19 disease. For our particular Particle Filter proposal, this means that the model distribution $p_\theta(e_t \mid s_{1:t}^n, e_{1:t-1})$ used to compute the incremental weights $\alpha_t(s_{1:t}^n, e_{1:t}) = \frac{p_\theta(e_t|s_{1:t}^n, e_{1:t-1})\ p_\theta(s_t^n|s_{1:t-1}^n, e_{1:t-1})}{q(s_t^n|s_{1:t-1}^n, e_{1:t})}$ for particle index n in pseudo Algorithm 1 has a significantly higher memory. While for most SSMs, this density collapses to $p_\theta(e_t \mid s_t^n)$, it defaults to $p_\theta(e_t \mid s_{t-28:t}^n, e_{t-28:t-1})$ for the HSMM-EM. Consequently, at any point in time, the previous 28 particle ancestors have to be resampled. We note that the computational complexity of this PF implementation is still linear with respect to the time index T, but a single proposal step is computationally more expensive than for standard PFs. This strengthens the need for a good MCMC kernel and provides further support for the Particle Gibbs case.

Given that data for our model are typically acquired sequentially, it is prudent to explore methods that facilitate such parameter estimation. A method that uses the PMCMC machinery and in which parameter estimates can be reused once new data arrives is known as Sequential Monte Carlo Squared, see Chopin et al. (2012). SMC2 is a method to sequentially estimate state trajectories as well as model parameter by using a Particle Filter as well as a Particle MCMC kernel for multiple chains in parallel. This algorithm is built on the Sequential Monte Carlo framework of Chopin (2002) and Del Moral et al. (2006). A pseudo algorithm for this implementation can be seen in Algorithm 8. For a further review that also includes multiple other applications, see Dai et al. (2020).

3.3.2 Prediction

A convenient feature for SMC2 algorithms is that parameter are estimated at each time index. Consequently, prediction for future cases and deaths can be performed at each time step as well. This is particularly useful for model comparison, see Section 3.3.3. In the SMC2 context, prediction comes naturally by using the Particle Filter. For particle n, given the state trajectory $s_{1:t}^n$, parameter θ and ODE state O_t, the predicted state $s_{t+1}^n \mid s_{1:t}^n, \theta$ can be sampled via the PF. Next, the ODE of Equation (3.1) can be solved with initial state O_t, transmission rate $\beta_{t+1} \mid s_{t+1}^n$ and otherwise fixed parameter θ. Finally, the model implied cases and deaths at $t+1$ can be computed as shown in Section 3.2.

3.3.3 Model Comparison

In Section 3.5, model comparison is performed to assess models with different observation distribution specifications as well as to detect the number of latent regimes governing the data. We will use methods from a classic batch as well as from a sequential estimation setting. In the batch estimation case, we use the Deviance Information Criterion (DIC), introduced in Spiegelhalter et al. (2002) and the Watanabe–Akaike Information Criterion (WAIC), introduced in Watanabe (2010). Both criteria have methods to account for the number of parameter in each model and provide a single numerical comparison value; the smaller the associated number, the better. Both criteria can be computed with the parameter samples from a MCMC run. For the DIC computation, we follow the notation from Gelman et al. (2013):

$$\mathrm{DIC} = -2 log\ p(e_{1:T} \mid \hat{\theta}_{Bayes}) + 2\,\mathrm{p_{DIC}},$$

where $\hat{\theta}_{Bayes} = \mathbb{E}(\theta \mid e_{1:T})$ is the posterior mean given data $e_{1:T}$. $\mathrm{p_{DIC}}$ is defined as the variance across all likelihood estimates. For the WAIC calculation, we follow the guidance of Vehtari et al. (2016):

$$\mathrm{WAIC} = -2\,\mathrm{lppd} + 2\,\mathrm{p_{WAIC}},$$

where the Log Pointwise Predictive Density (LPPD) can be interpreted as a training error and is computed as $\mathrm{lppd} = \sum_t^T log\left(\frac{1}{N}\sum_n^N p(e_t \mid \theta_n)\right)$. T is the number of data points and N the number of MCMC samples. $\mathrm{p_{WAIC}}$ can be computed as $\mathrm{p_{WAIC}} = \sum_t^T \mathrm{Var_{t,post}}\,(log\ p(e_t \mid \theta))$, where $\mathrm{Var_{t,post}}\,(log\ p(e_t \mid \theta))$ is the variance of $p(e_t \mid \theta)$ at time t accross all θ samples. $\mathrm{p_{DIC}}$ and $\mathrm{p_{WAIC}}$ serve as penalizer for additional model parameter. In a sequential model comparison context using SMC2, there are several techniques that can be used from by-products that arise during the estimation run. A highly sought after but typically unobtainable method is the analysis of each model's marginal likelihood $p(e_{1:T}) = \int p(e_{1:T}, \theta)\ d\theta$. This distribution is typically intractable, but in the SMC2 case, an estimate for $p(e_t \mid e_{1:t-1})$ can be computed at each time step at practically no extra costs. We denote $p(e_{t+1} \mid e_{1:t})$ as (one step ahead) Predictive Likelihood (PL) at $t+1$, PL_{t+1}, see Kastner (2016), and note that it is straight forward to compute $\hat{p}(e_{1:t}) = \prod_{i=1}^t \hat{PL}_t$ after the SMC2 run has finished. However, given the observation distribution specifications are different for our models, a direct comparison via this method is difficult. An alternative approach to compute the PL terms is based on the prediction step, which is described in Section 3.3.2. For

SSMs, the posterior predictive distribution is defined as $p(e_{t+1} \mid e_{1:t}) = \int p_\theta(e_{t+1} \mid s_{t+1}, s_{1:t}, e_{1:t})\ p_\theta(s_{t+1} \mid s_{1:t}, e_{1:t})\ p(s_{1:t}, \theta \mid e_{1:t})\ ds_{t+1}, s_{1:t}, \theta$. $p_\theta(e_{t+1} \mid s_{t+1}, s_{1:t}, e_{1:t})$ can be evaluated and sampled from, and a Monte Carlo estimate can be obtained for the Predictive Likelihood, $\hat{PL}_{t+1} = \hat{p}(e_{t+1} \mid e_{1:t}) \approx \frac{1}{N} \sum_{n=1}^{N} p_{\theta_n}(e_{t+1} \mid e_{1:t}, s^n_{1:t+1})$, where N is the number of samples. Note that in this case, $p_{\theta_n}(e_{t+1} \mid e_{1:t}, s^n_{1:t+1})$ has to be evaluated and is not a pure by-product of the actual algorithm, which permits us to evaluate only parts in the observation distribution specification that are shared among all models. Direct model comparison can be conducted by computing the so called Cumulative Log Predictive Bayes Factor (CLPBF), which is derived from the cumulative sums of log PLs. Given model A and B, for $u > 0$, this factor can be stated as

$$CLPBF_{t+1:t+u} = log \left[\frac{p_A(e_{t+u} \mid e_{1:t})}{p_B(e_{t+u} \mid e_{1:t})} \right] = \sum_{i=t+1}^{u} log \left[PL_i(A) - PL_i(B) \right],$$

where a positive value for the CLPBF indicates evidence in favor for model A. In this context, if $t = 0$ and $u = T$, the CLPBF defaults to the log Bayes Factor (BF). Note that when fitting real data to the SMC2 algorithm in Section 3.5, we usually use $t_0 > 1$ data points as training period to initialize the jitter kernels in a reasonable parameter region. This is especially useful in our case study as the reporting standard for the COVID-19 data at the beginning of the times series has been extremely noisy. The resulting estimate for the comparison criteria will be the slightly different $\hat{p}(e_{t_0+1:T} \mid e_{1:t_0}) = \prod_{t=t_0+1}^{T} \hat{PL}_t$.

3.4 Simulations

In this section, simulation experiments are performed to verify the PMCMC and SMC2 algorithms. In order to do so, we use the posterior mean parameter of the PMCMC run of the winning model in Section 3.5 according to the model comparison criteria stated in Section 3.3.3. 500 data points are generated from a SEEIIR model with piecewise constant β that follows a HSMM. The corresponding values used for data generation are shown in the second column in Table 3.1, and the corresponding generated data can be seen in Plot A.13. The estimation results and common output statistics can be seen Table 3.1. The initial parameter are sampled from the priors, defined in Section 3.5, and traceplots are shown in Figure A.14. Both the traceplots and $\hat{R}$ indicate excellent mixing, and the posterior mean of each parameter is close to the true value, which is contained in the credible interval for all parameter.

θ	**True**	**Mean**	**MCSE**	**SD**	**Rhat**	**Q2.5**	**Q50.0**	**Q97.5**
β_1	-1.72	-1.71	0.0	0.08	1.01	-1.86	-1.71	-1.55
β_2	-1.36	-1.36	0.01	0.09	1.02	-1.51	-1.36	-1.17
β_3	-0.81	-0.81	0.0	0.09	1.0	-0.99	-0.82	-0.63
β_4	0.45	0.46	0.01	0.25	1.0	-0.11	0.47	0.95
γ_1	0.45	0.43	0.0	0.04	1.0	0.36	0.43	0.51
γ_2	0.46	0.48	0.0	0.04	1.0	0.41	0.48	0.55
ϵ	0.94	1.01	0.0	0.1	1.0	0.82	1.01	1.23
p_1	0.87	0.84	0.0	0.11	1.0	0.58	0.87	0.98
p_2	0.50	0.53	0.0	0.15	1.01	0.25	0.53	0.82
p_3	0.17	0.26	0.0	0.16	1.0	0.03	0.23	0.62
$p_{thirdstate,1}$	0.35	0.35	0.0	0.15	1.0	0.09	0.34	0.67
$p_{thirdstate,2}$	0.35	0.34	0.0	0.15	1.0	0.08	0.32	0.65
r_1	36.11	36.03	0.13	6.01	1.0	25.15	35.72	48.59
r_2	24.19	25.12	0.11	5.31	1.0	15.85	24.64	36.88
r_3	14.19	15.83	0.1	4.18	1.0	8.6	15.43	25.06
r_4	28.12	27.93	0.12	5.48	1.0	18.2	27.68	39.95
ψ_1	0.76	0.75	0.0	0.06	1.01	0.62	0.75	0.86
ψ_2	0.74	0.78	0.01	0.08	1.04	0.64	0.78	0.94
ψ_3	0.54	0.62	0.0	0.08	1.0	0.45	0.62	0.77
ψ_4	0.50	0.5	0.0	0.15	1.0	0.21	0.5	0.79
ϕ_{cases}	5.00	5.32	0.0	0.12	1.0	5.00	5.33	5.56
ϕ_{deaths}	5.30	5.42	0.0	0.11	1.0	5.2	5.41	5.63

Table 3.1 Posterior output statistics of four PMCMC chains on a HSMM-EM with SEEIIR style ODE, four states and Negative Binomial duration distribution for simulation experiments in Section 3.4. During the run, 1500 iterations have been used with 700 burnin steps, resulting in 3200 total samples. Data has been generated from a model with parameter equal to the values in the second column of table, and can be seen in Figure A.13. Initial parameter have been sampled from the prior distributions.

3.5 Applications

In this section, a case study on the SARS-CoV-2 epidemic in the United Kingdom is performed. In particular, we analyze the number of regimes that are needed to accurately describe this phenomena with the model defined in Section 3.2. Moreover, we consider if expanding the observation model specification yields a predictive advantage. The data source for the proposed observation model includes fatalities, which we define as the reported deaths due to COVID-19 as interpreted by the U.K. government, as well as infections, which we depict as the reported cases in this country, see GOV.UK (2022). Observation model specifications often include either infections, which can be noisy and include lagging reporting periods, or fatalities, which are less noisy but typically need additional parameter in the observation model. We will combine both sources and test if this yields a competitive advantage with respect to predictive performance against models that use fatalities only.

3.5.1 Data Processing

All data for the number of reported deaths, cases and vaccinations in the United Kingdom have been obtained since the starting date of recording from a publicly available website, see GOV.UK (2022). The initial data point for the estimation was taken at the first time index corresponding to at least ten reported deaths. Vaccination data have been set to zero where unavailable. The number of data points were chosen to be around 600. This period includes several waves of COVID-19 spikes with respect to the reported infections and had estimated Infection Fatality Ratio values available at the time of the case study, see Brazeau et al. (2022) for more details on the ifr.

3.5.2 Model Dynamics and Prior Assignments

The model consisting of both fatalities and infections in the observation distribution has parameter: $\theta = \{\beta, \gamma, \epsilon, p, p_{thirdstate}, r, \psi, \phi_{cases}, \phi_{deaths}\}$. β, γ, ϵ are ODE parameter in Equation (3.1). The other parameter, ρ, which can be interpreted as vaccine efficiency parameter, has been fixed to 0.5. p is the transition distribution parameter and $p_{thirdstate}$ is the transition distribution parameter from a non-recurring initial state. This separate state considers the additional uncertainty at the beginning of the reporting process and significantly improves inference for all other parameter going forward. r and ψ are the Negative Binomial distribution parameters for the duration distribution. ϕ_{cases} and ϕ_{deaths} are noise parameter for the corresponding data, which is described in more detail in Section 3.2. The data $e_t = \{c_t^r, d_t^r\} \sim$ Neg. Bin.$_{Alt}\left(c_t^*, c_t^* + \frac{c_t^{*2}}{\phi_c}\right) \times$ Neg. Bin.$_{Alt}\left(d_t^i, d_t^i + \frac{d_t^{i2}}{\phi_d}\right)$ depends on the model implied deaths, which are a function of the model implied cases, which in turn are obtained from the ODE in Equation (3.1). This ODE depends on the latent state trajectory up to time t, and each latent state follows a HSMM. The ifr values used to compute the model implied deaths defined in Section 3.2 are set to $\{0.01035, 0.0095, 0.007245, 0.004, 0.002\}$ and the change point dates have been set to 2020-07-18, 2020-10-01, 2021-01-30 and 2021-06-01. Our choices are based on the results of Chatzilena et al. (2022) and have been adjusted for the limited memory of the model implied death computations. The same holds for the under-reporting score that is used in the observation model for the cases, which can be seen in Figure A.16. The β parameter are modeled in log space and are a vector of length equal to the number of latent states, and their prior is defined as ordered Multivariate Normal distribution. The four-state model has a mean vector of $\{log(0.15), log(0.4), log(0.6), log(1.2)\}$ with a unit diagonal covariance matrix. The γ parameter is of length two, and the elements have Gamma priors for shape and scale of Gamma(1600, $\frac{1}{4000}$) and Gamma(2500, $\frac{1}{5000}$). Similarly, Gamma priors have been chosen for $\epsilon \sim$ Gamma(1000, $\frac{1}{10000}$), $\phi_{cases} \sim$ Gamma(2500, $\frac{1}{500}$) and $\phi_{deaths} \sim$ Gamma(2500, $\frac{1}{500}$). For the transition distribution parameter p and $p_{thirdstate}$, Dirichlet prior have been assigned that favor equal weights, i.e., $p \sim Dirichlet(\alpha_1 = \alpha_2 = ... = \alpha_k = k)$, where $k =$ number of latent states. $\psi \sim Beta(0.5, 0.5)$ for each state, and r has Gamma priors with shape $\{40, 30, 20\}$ in the 4 state case for the recurring states, and $shape = 28$ for the non-recurring state. 28 has been chosen as this is the number of lookback days that assigns a death due to COVID-19 after being infected in the UK reporting methodology. We want to emphasize that initial test runs have clearly shown that three or fewer regimes have very slow mixing properties for the MCMC part and struggle to forecast realistic data, hence we will start with at least four available states, and check if adding more regimes is favourable with respect to the model comparison criteria mentioned in Section 3.3.3.

3.5.3 Results

In our analysis, models that include fatalities and infections in the observation distribution specification vastly outperform models that include fatalities only with respect to predictive power. Daily and weekly forecast performances with respect to the Cumulative Log Predictive Bayes Factor as defined in Section 3.3.3 can be seen in Figure A.20, which illustrates a consistent performance difference that is increasing over time. To visualize the discrepancy in predictive accuracy, daily forecasts for all models can be seen in Figure 3.3, in which the fatalities-only model has continuously larger Credible Intervals across all time horizons versus the combined fatalities-infections models. As for the number of regimes, the four-state model does perform slightly better for daily forecasts, and similarly on the weekly scale in comparison to its five state counterpart. A summary for all model comparison criteria is shown in Table 3.2. For batch estimation comparison, DIC and WAIC results are shown in the two top sub-tables, in which both model choice schemes favor the four-state fatalities-infections model. The bottom two sub-plots depict the daily and weekly cumulative log PL as defined in Section 3.3.3 from the SMC2 runs at the final time index. We conclude that the four-state fatalities-infections variant, while having less parameter, performed at least equally well with respect to predictive power and the DIC/WAIC model comparison criterion. Consequently, all analysis going forward will default to this model.

A traceplot for the continuous model parameter of the PMCMC run is shown in Figure A.17, and output diagnostics for all samples can be seen in Table 3.3. All parameter have been initiated from the prior distributions and converge quickly to the typical set. Summary plots for the model implied against reported infections and fatalities can be seen in Figure 3.2, which depicts COVID-19 relevant diagnostics given data available up to the current time on the x-axis. In the two top plots, both the model implied deaths and cases, adjusted for under-reporting, match the reported data excellently. The third plot from the top shows the posterior mode of the latent state at each time point. Each state represents a different regime, and after the time horizon shown at the blue dashed line, which shows the first date when vaccination data is taken into account for the ODE in our model, there is a clear regime change. The model transitions from being in regime one and two to being exclusively in regime two and three, which have higher β transmission rates. Furthermore, the bottom plot shows the R_t values over time, which are consistently higher after the vaccination effect. In order to have such a large number of cases after taking the vaccination effect into account, new, more dangerous, SARS-CoV-2 variants such as the alpha and omega variations have to have emerged at that time. This regime change has been immediately identified during the sequential estimation run, which would be challenging for classic batch estimation techniques.

We note that if the HSMM with Negative Binomial distribution exhibited a failure rate close to 1, it would resemble a Geometric distribution. In such a scenario, the HSMM would collapse into a Hidden Markov Model. However, as can be seen in the parameter estimates in Table 3.3, it was found that the failure rate in all states of the winning model was significantly higher than that.

DIC computations			
Model	**DIC**	**LPPD**	$\mathbf{p_{DIC}}$
Deaths and Cases - 4 states	41011.8	-20043.5	462.4
Deaths and Cases - 5 states	42093.1	-19272.0	1774.5

WAIC computations			
Model	**WAIC**	**LPPD**	$\mathbf{p_{WAIC}}$
Deaths and Cases - 4 states	15705.3	-7768.9	83.7
Deaths and Cases - 5 states	15837.2	-7749.2	169.4

Daily sequential model choice	
Model	**Daily cumulative log PL**
Deaths - 4 states	-26394
Deaths and Cases - 4 states	-16523
Deaths and Cases - 5 states	-16601

Weekly sequential model choice	
Model	**Weekly cumulative log PL**
Deaths - 4 states	-1985
Deaths and Cases - 4 states	-1845
Deaths and Cases - 5 states	-1843

Table 3.2 Model comparison criteria for various models estimated on real data in Section 3.5. The 2 top tables depict batch model selection rules - the lower the performance criterion, the better. The 2 bottom tables display sequential model selection rules - the larger the performance criterion, the better.

3.6 Conclusions

In this Chapter, we explored a new class of SEEIIR models with a piecewise constant transmission rate parameter depending on a latent variable that follows a HSMM. The SSM formulation permits regime change and duration forecasting, which is extremely valuable for policy decision-making. Simultaneously, the discrete structure of the latent variables keeps the number of continuous parameters limited and the interpretation of the model straightforward. In order to estimate the latent HSMM, we designed a tailored PF that can be used efficiently in the PMCMC and SMC2 machinery. The computational schemes were validated on simulated data and, in a case study on reported COVID-19 fatalities and infections in the United Kingdom, the proposed models were tested for multiple regimes and various specifications of the observation

θ	**Mean**	**MCSE**	**SD**	**Rhat**	**Q2.5**	**Q25.0**	**Q50.0**	**Q75.0**	**Q97.5**
$\log \beta_1$	-1.72	0.0	0.08	1.01	-1.89	-1.78	-1.72	-1.67	-1.57
$\log \beta_2$	-1.36	0.01	0.08	1.03	-1.5	-1.41	-1.36	-1.31	-1.2
$\log \beta_3$	-0.81	0.0	0.09	1.0	-0.99	-0.87	-0.81	-0.76	-0.63
$\log \beta_4$	0.45	0.01	0.22	1.01	0.02	0.3	0.44	0.59	0.9
γ_1	0.45	0.0	0.04	1.0	0.37	0.42	0.45	0.48	0.54
γ_2	0.46	0.0	0.04	1.0	0.38	0.43	0.45	0.48	0.53
ϵ	0.94	0.0	0.1	1.0	0.76	0.87	0.94	1.0	1.13
p_1	0.87	0.0	0.09	1.01	0.66	0.83	0.89	0.94	0.98
p_2	0.5	0.01	0.14	1.01	0.23	0.38	0.5	0.6	0.77
p_3	0.17	0.01	0.1	1.01	0.03	0.09	0.15	0.23	0.4
$p_{thirdstate,1}$	0.35	0.01	0.15	1.0	0.09	0.23	0.34	0.45	0.67
$p_{thirdstate,2}$	0.35	0.01	0.15	1.0	0.08	0.24	0.34	0.46	0.67
r_1	36.12	0.18	6.53	1.0	24.69	31.63	35.71	39.77	50.58
r_2	24.19	0.15	5.29	1.0	14.72	20.7	23.91	27.49	35.39
r_3	14.19	0.17	4.32	1.0	7.22	11.0	13.72	16.96	23.56
r_4	28.13	0.12	5.33	1.0	19.08	24.22	27.75	31.69	39.0
ψ_1	0.76	0.01	0.07	1.04	0.62	0.72	0.77	0.81	0.88
ψ_2	0.75	0.01	0.07	1.03	0.6	0.71	0.75	0.8	0.87
ψ_3	0.55	0.01	0.09	1.01	0.36	0.48	0.55	0.61	0.73
ψ_4	0.5	0.01	0.15	1.0	0.21	0.4	0.5	0.61	0.81
ϕ_{cases}	4.91	0.0	0.11	1.0	4.68	4.83	4.91	4.99	5.13
ϕ_{deaths}	5.25	0.0	0.12	1.0	5.02	5.17	5.26	5.33	5.49

Table 3.3 Posterior output statistics of four PMCMC chains on a HSMM-EM with SEEIIR style ODE, 4 states and Negative Binomial duration distribution for applications in Section 3.5. 1200 iterations have been used with burnin set to 700, resulting in 2000 total samples. Real data can be seen in Figure A.15, and is described in more detail in Section 3.5.1. Initial parameter have been sampled from the prior distributions.

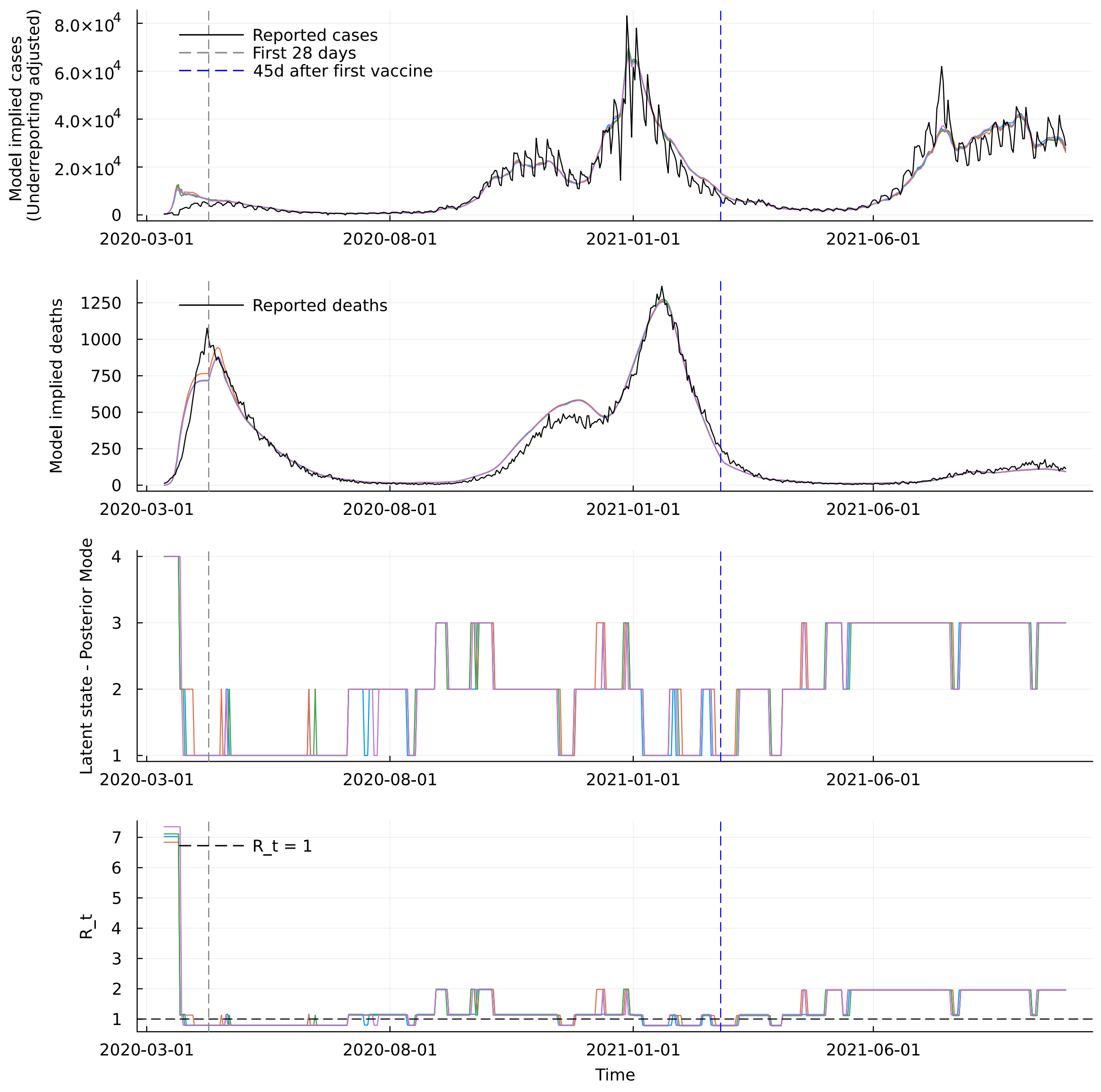


Figure 3.2 PMCMC Model implications of the best performing model for real data in Section 3.5. All computations are based on data available up to the current time on the x-axis. The upper plot shows the reported infections against the model implied infections, adjust for an under-reporting score. The second plot from the top shows the corresponding fatalities. The third plot from the top shows the posterior mode of the latent state at each time point. Each state represents a different regime. The bottom plot shows the corresponding R_t value at each time index. The grey horizontal line represents the first 28 days, for which no full history for the model implied deaths as stated in Section 3.2 is available. The blue horizontal line shows the date when vaccination data is taken into account for the ODE in our model.

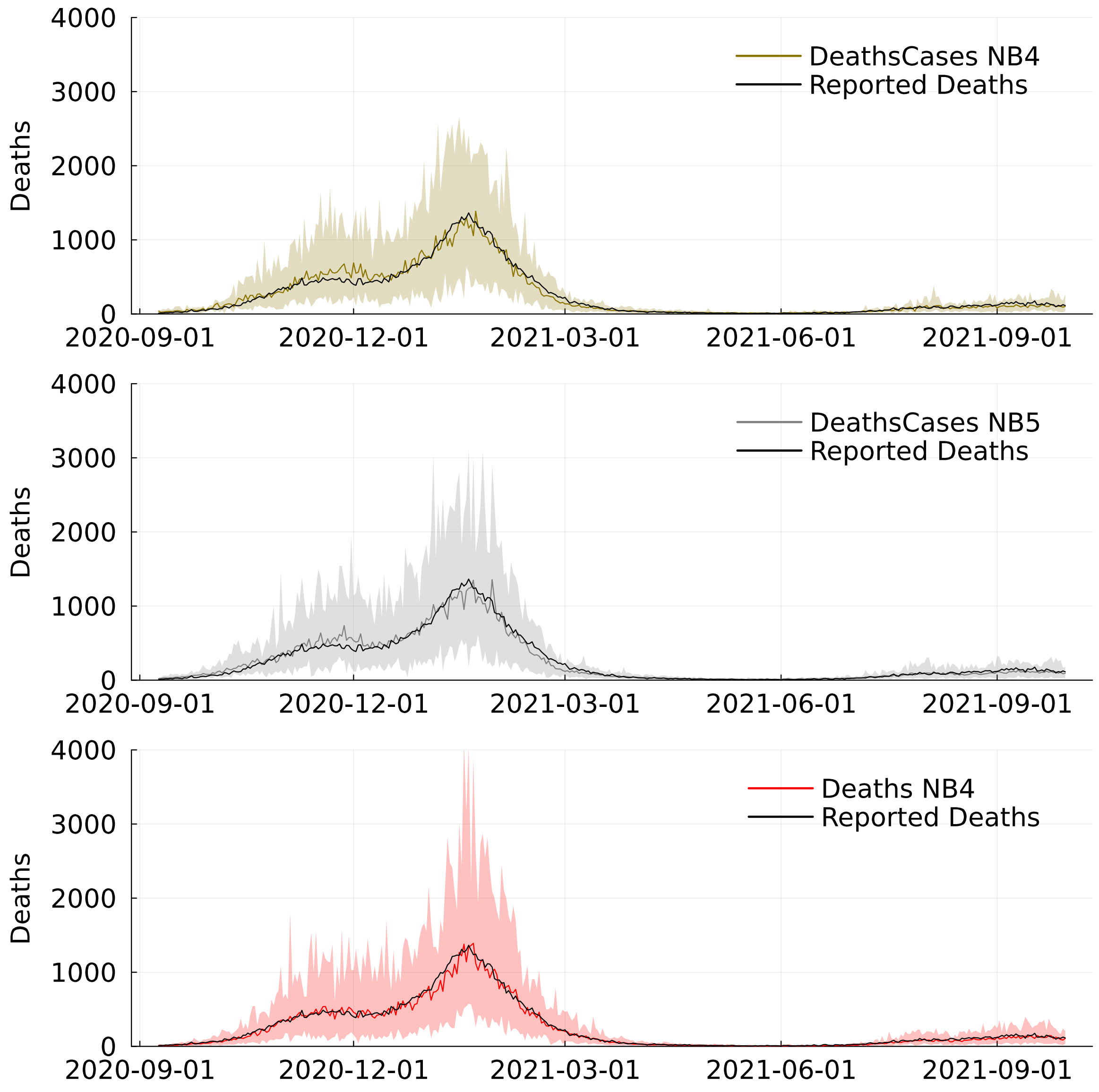


Figure 3.3 This plot shows daily predictions of each model described in Section 3.5 and their corresponding 95% CI against realized data. NB4 stands for models using a Negative Binomial Duration distribution with 4 different regimes. In this plot, only predicted fatalities are shown to compare all models equally.

model. We conclude that combining infections and fatalities adds significant value to the predictive performance of HSMM-EMs and that utilizing more than four different regimes does not improve their predictive power for the underlying data. For future research, open projects include linking the latent state duration probabilities explicitly to the data. Possible inference about the timing and length of government policy actions, i.e., initiating a lockdown given the current regime, could be deduced from here. Furthermore, new models could be added and compared to our proposals via the comparison criteria and inference algorithms used in this Chapter. Different data sources could be added to our proposed observation model specification as well. For example, a useful addition would be hospital admissions, which were unavailable at the time we conducted our case study.

3.7 Software

The data and code used to run the algorithms in this Chapter can be be seen in https://github.com/paschermayr/Publish_Covid19SSM. More detailed information about the implementations for running all algorithms and computing all tables can be found in https://github.com/paschermayr/Baytes.jl and its sub-libraries. The corresponding plots are defined in https://github.com/paschermayr/BaytesInference.jl. For all ODE computations used within the Baytes.jl packages, the Julia library Differentialequations.jl, see Rackauckas and Nie (2017), has been used.

Sometimes I'll start a sentence and I don't even know where it's going. I just hope I find it along the way.

Michael Scott (*The Office*, 2005)

Chapter 4

A Class of Stochastic Volatility Models with Copula Dependencies

Stochastic Volatility (SV) models are a popular class of models to analyze the dependency structure between stocks and their volatility. We develop a new class of SV models by incorporating carefully selected copula structures to reconstruct stylised empirical behaviours that cannot be captured by symmetric Gaussian innovations. To estimate model parameter for each copula setting, modern Markov Chain Monte Carlo (MCMC) and Sequential Monte Carlo (SMC) methods are applied. We use batch as well as sequential Bayesian model selection to provide insights into the suitability of different copula choices on the US equity S&P 500 and an associated volatility index. Our results provide strong evidence against the common choice of Gaussian innovations for typical SV models proposed in the financial modelling literature.

4.1 Introduction

Time-varying volatility modelling has appeared in the financial community at least since the early work in Clark (1973). Formulated statistical models are broadly grouped into conditional volatility (i.e. ARCH, GARCH and their extensions, see e.g. (Engle, 1982, Bera and Higgins, 1993) for a review) and Stochastic Volatility classes. The latter treats the underlying volatility as a latent process. Several works in empirical finance and econometrics have tried to compare models in-between the two groups or within the same group, see e.g. Kim et al. (1998). A lot of research has focused on approximate or computationally intensive methods for fitting SV models, as they fall into the intractable likelihood class of models, in contrast to the conditional volatility models mentioned above, which typically have explicit likelihoods. Within the SV family, another degree of separation involves continuous versus discrete time models, the former class being particularly popular in the stochastic finance and asset pricing communities (see e.g. Hull and White (1987)). We will focus on the latter and aim to enrich this class to better capture empirical stylised effects in the co-movement of prices and volatilities.

SV models can capture important stylised effects, including leptokurtic, heavy-tailed marginal return distributions, volatility clustering and the so called leverage effect, see Ghysels et al. (1996), Shephard (1996). Our interest lies in the important matter of dependency structures within SV modelling. Several recent works have empirically illustrated the presence of non-linear, asymmetric structures at the joint distribution of prices and volatilities, and suggested the use of classes of *Copulas* for capturing such effects (see e.g. Ning et al. (2008)). Copula theory provides an extremely flexible modelling framework for a multitude of shapes for asymmetric joint distributions, see, e.g., Joe (2014)).

Our first aim in this work is to suggest useful models for log-prices and volatilities that naturally incorporate Copula dependencies in their development, thus moving beyond the classical Heston-type models of symmetric Gaussian structures for capturing leverage effects. After the modelling aspect has been clarified, we will make use of recent advances in Monte-Carlo methodology to carry out model choice and determine the most appropriate Copulas from a collection of reasonable candidates based on Markov Chain Monte Carlo as well as Sequential Monte Carlo runs. During these steps, we will also discuss methods that permit *online* model choice procedures.

The rest of the Chapter is structured as follows. In Section 4.2, we show some popular continuous-time SV models for scalar time series and set up the framework for adding Copula effects. Section 4.2.2 gives a brief introduction of Copula theory, with a focus on aspects relevant to the modeling context of this work. In Section 4.3, we present the computational approach for *simultaneously* fitting the models and calculating the model evidence. We show numerical applications on real data in Section 4.5, and conclude with some discussion in Section 4.6.

4.2 Modelling framework

4.2.1 Diffusions and their Discretisation

For a standard Stochastic Volatility model in scalar form, a general structure for the instantaneous log-price S_t and volatility V_t can be depicted as:

$$\begin{aligned} dS_t &= f_\theta(S_t, V_t)dt + g_\theta(V_t)dB_t \ , \quad S_0 = s_0 \in \mathbb{R} \ ; \\ dV_t &= a_\theta(V_t)dt + b_\theta(V_t)dW_t \ , \quad V_0 \sim p_\theta(v_0) \ , \end{aligned}$$

for mappings $f_\theta : \mathbb{R}^2 \mapsto \mathbb{R}$, $g_\theta : \mathbb{R} \mapsto \mathbb{R}^+$, $a : \mathbb{R} \mapsto \mathbb{R}$, $b : \mathbb{R} \mapsto \mathbb{R}^+$ depending on some parameter $\theta \in \Theta \subseteq \mathbb{R}^p$, correlated Brownian innovations dB_t, dW_t to allow for a leverage effects, and some initial distribution $p_\theta(v_0)$ for the underlying velocity process. A famous specification of this model is known as Heston Model (HM) and has the following dynamics:

$$\begin{aligned} dS_t &= (\alpha + \beta V_t)dt + V_t^{1/2} dB_t \ ; \\ dV_t &= \kappa\,(\mu - V_t)dt + \sigma V_t^{1/2} dW_t \ , \end{aligned} \tag{4.1}$$

where B_t, W_t denote standard Brownian motions such that $\langle dB_t, dW_t\rangle = \rho \in (-1, 1)$. Model parameters include $\theta = (\alpha, \beta, \kappa, \mu, \sigma)$ and the noise distribution adds the extra parameter ρ. As discussed in Section 4.1, we will sometimes have to work with a discrete-time version of such models. It follows that, after using a Euler scheme,

$$\begin{aligned} S_i &= S_{i-1} + f_\theta(S_{i-1}, V_{i-1})\delta + g_\theta(V_{i-1})\sqrt{\delta}\,\epsilon_i \ , \quad S_0 = s_0 \in \mathbb{R} \ ; \\ V_i &= V_{i-1} + a_\theta(V_{i-1})\delta + b_\theta(V_{i-1})\sqrt{\delta}\,\zeta_i \ , \quad V_0 \sim p_\theta(v_0) \ , \end{aligned}$$

with the innovation terms $\{E_i\}_i = (\epsilon_i, \zeta_i)_i$ being independent over $i \geq 1$. In standard SV models, the error terms are typically specified as

$$(\epsilon_i, \zeta_i) \sim N\Big(0, \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}\Big),$$

for $\rho \in (-1, 1)$. Here, we will allow for a much more flexible dependence framework by referring to the Copula machinery. Thus, for marginal distributions F_S and F_V, our model for the distribution of the transformed noises $(\tilde{\epsilon}_{ii}, \tilde{\zeta}_i) := (F_S(\epsilon_i), F_V(\zeta_i))$ will be

$$(\tilde{\epsilon}_{ii}, \tilde{\zeta}_i) \sim C(\tilde{\epsilon}_{ii}, \tilde{\zeta}_i; \theta)$$

independently over $i \geq 1$, where C is a Copula and F_S and F_V the marginal distributions of choice. The original noises can be easily reconstructed via $(\epsilon_i, \zeta_i) = (F_S^{-1}(\tilde{\epsilon}_i), F_V^{-1}(\tilde{\zeta}_i))$.

4.2.2 Copulas

Copulas provide a flexible approach for building up multivariate dependencies and have been used extensively in finance, see Cherubini et al. (2004), Czado et al. (2019), Krupskii and Joe (2020), Bladt and McNeil (2022). We provide a brief summary of basic properties of Copula theory, and refer to Joe (1997) for a more thorough review on this topic. In our setting we focus on bivariate laws, as that suffices for the development of our Stochastic Volatility models. A (bivariate) Copula $C = C(u, v)$ is a distribution function on $[0, 1]^2$ with uniform marginals on $[0, 1]$. We denote the corresponding pdf (which we assume exists) as $c(u, v)$ and assume that marginal cdfs F_X, F_Y have been determined for some univariate random variables X, Y respectively. Following common practice, for uniform random variables (U, V), the following transformation can be stated:

$$(U, V) =: (F_X(X), F_Y(Y)) \mapsto \left(F_X^{-1}(F_X(X)), F_Y^{-1}(F_Y(Y)) \right) =: (X, Y).$$

Thus, the choice of Copula function refers to the specification of the joint law of $(F_X(X), F_Y(Y))$. Stylised effects explored for the selection of Copulae are, e.g., tail symmetries/asymmetries, positive/negative dependence or the strength of tail dependency. We will provide results for the lower and upper tail dependency as well as estimates for the correlation structure of stock and volatility for each Copula choice in Section 4.5. More discussion on the selection of Copulas can be found directly in the Applications Section 4.5.4.

4.2.3 Model specification

In our proposed model, we consider the transformation $X_t = \log V_t$ and apply Ito's Lemma on (4.1) to get

$$\begin{aligned} dS_t &= \Big(\mu_S - \exp(X_t)/2\Big)dt + \exp(X_t/2)dB_t, \\ dX_t &= \frac{\kappa\Big(\mu_V - \exp(X_t)\Big) - \frac{1}{2}\sigma^2}{\exp(X_t)}dt + \sigma\exp(-X_t/2)dW_t \end{aligned} \tag{4.2}$$

We will consider a discrete approximation of (4.2) based on the Euler-Maruyama scheme

$$\begin{aligned} S_t &= S_{t-1} + \Big(\mu_S - \exp(X_t)/2\Big)\delta + \sqrt{\delta}\,\exp(X_t/2)\,\epsilon_t, \\ X_t &= X_{t-1} + \frac{\kappa\Big(\mu_V - \exp(X_{t-1})\Big) - \frac{1}{2}\sigma^2}{\exp(X_{t-1})}\delta + \sigma\sqrt{\delta}\exp(-X_{t-1}/2)\,\zeta_t\ . \end{aligned} \tag{4.3}$$

The dependency structure between ϵ_t and ζ_t is modeled via a Copula. Note that when estimating Copulae, it is typically standard to formulate the model with respect to the uniform data U instead of real data Y. This step decouples the modeling of marginals from the dependency between them and is recommended if the researcher is only interested in the dependency structure. In our model, however, parameters are used to transform data from the real to the uniform domain, and the model has to be specified with respect to Y. Assuming $Y_t^v = V_t$ and $Y_t^s = S_t$, the likelihood $f(Y^s, Y^v \mid \Theta)$ for the unknown parameter vector Θ, can then be written with respect to the error terms $(\epsilon_i, \zeta_i)_{i=1,\dots,T}$ by inverting (4.3) to obtain $E = h_\Theta(Y^s, Y^v)$. The Jacobian of this transformation is denoted by J_Θ. Using $h_\Theta(\cdot)$ and J_Θ, we get

$$f(Y^s, Y^v \mid \Theta) = f\left(E \mid J_\Theta\right)|J_\Theta| = \sigma^{-T}\prod_{t=1}^{T} f_S(\epsilon_t)f_V(\zeta_t)c(F_S(\epsilon_t), F_V(\zeta_t) \mid \theta),$$

with $f_S(\epsilon_t)$ and $f_V(\zeta_t)$ being the pdf and $F_S(\epsilon_t)$ and $F_V(\zeta_t)$ the cdf of the marginal distributions of choice. c is the density of the Copula choice. The derivation of $|J_\Theta|$ can be seen in Appendix C. The model structure can be seen as a Directed Acyclic Graph (DAG) in Figure 4.1.

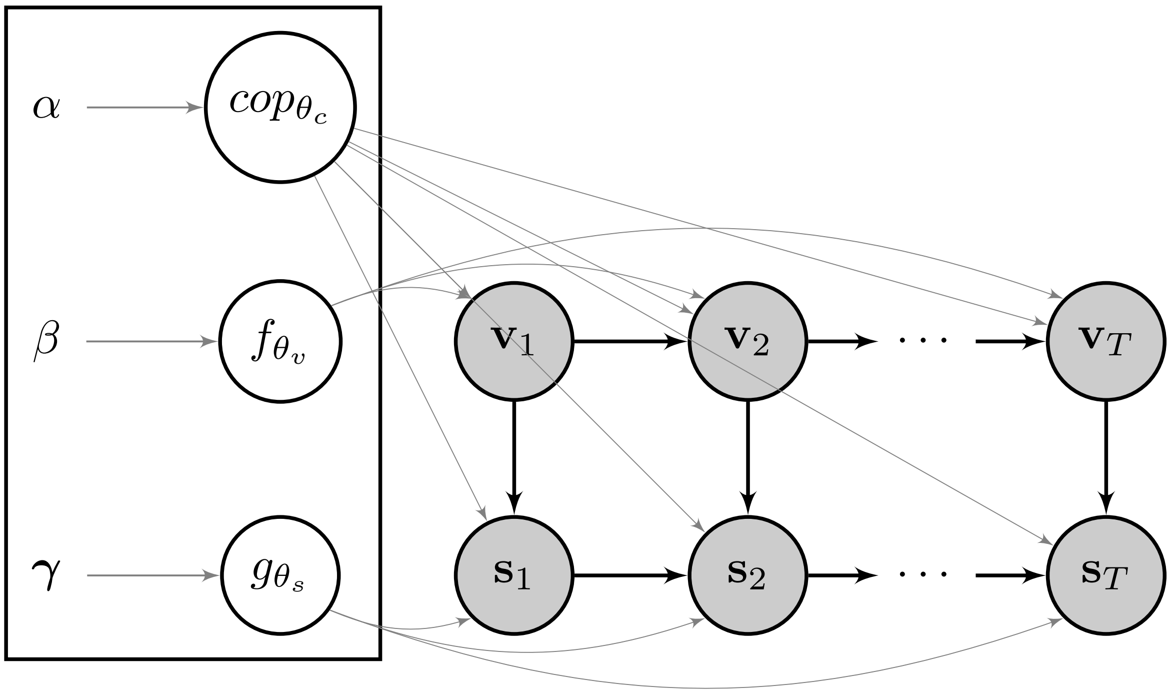


Figure 4.1 A Stochastic Volatility Model with Copula Dependencies (SVC), parameter θ and hyper-parameter $\{\alpha, \beta, \gamma\}$. The shaded nodes v_t (volatility) and s_t (stock) denote the observed data at time t. *cop* defines the Copula choice and dependence structure between s_t and v_t. f denotes all model functions and θ_s all parameter for s, while g denotes all model functions and θ_v all parameter for v. θ_c denotes all parameters required for the Copula choice.

4.3 Bayesian Inference

4.3.1 Parameter estimation

The typical Bayesian parameter estimation goal is to infer the posterior distribution of the model parameter θ given the observed data $e_{1:T}$, $p(\theta \mid e_{1:T}) = \frac{p_\theta(e_{1:T})\ p(\theta)}{p(e_{1:T})}$. For this model, standard MCMC estimation can be applied, for which we assume basic familiarity and refer to Craiu and Rosenthal (2014) for a more detailed guide on the topic. For all batch estimation computations in Section 4.4 and 4.5, a fully automatic version of the MCMC variant Hamiltonian Monte Carlo (HMC), see Neal (2012), Betancourt (2018), known as No U-Turn Sampling (NUTS) algorithm, which was proposed by Hoffman and Gelman (2014) and later on improved in Betancourt (2016), has been used.

Given the underlying data is usually observed in a times series setting, sequential parameter estimation techniques are particularly interesting. A method that explores the parameter space sequentially and in which the MCMC machinery insight from the batch estimation part can be reused is known as Iterated Batch Importance Sampling (IBIS), see Chopin (2002) and Del Moral et al. (2006). During each iteration over time index t, the incremental likelihood $p_{\theta_n}(e_{1:t} \mid e_{1:t-1})$ and therefore $p_{\theta_n}(e_t)$ is computed for each of n a priori chosen particles. If the variance of these increments is too large, the parameter are jittered using a MCMC kernel. Some advantages of this method are discussed in Section 4.3.3, and a pseudo algorithm implementation for the IBIS algorithm can be seen in Algorithm 7. For a more detailed review, including multiple other applications, see Dai et al. (2020). It is important to note that this inference challenge differs from the ones addressed in the previous two chapters. Unlike before, there is no involvement of an unobserved latent state sequence, and as a result, there is no need to employ a Particle Filter to compute incremental likelihood approximations. Consequently, the IBIS algorithm generally exhibits faster computational speed compared to the SMC2 variant utilized in the previous chapters.

4.3.2 Prediction

While the IBIS algorithm iterates through the data, predictions can be performed at each time step at practically no extra costs. This convenient feature permits the use of additional model comparison techniques, see Section 4.3.3. At time t, given parameter θ, (u_{t+1}, v_{t+1}) can be simulated from the attached Copula of choice. The uniform data can then be transformed to the real space given the marginals associated to the model, which will yield the innovation terms $(\epsilon_{t+1}, \zeta_{t+1})$ from section 4.2. $(\epsilon_{t+1}, \zeta_{t+1})$ can then be transformed to the stock and volatility prediction via Equation (4.3) given the observed data and parameter θ.

4.3.3 Model Comparison

We will use model comparison methods from a standard batch as well as from a sequential estimation view point. For the former, we use the Deviance Information Criterion (DIC), see Spiegelhalter et al. (2002) and the Watanabe–Akaike Information Criterion (WAIC), see Watanabe (2010). In these methodologies, a single numerical value is provided to compare different models - the smaller the value, the better. The DIC as well as the WAIC account for the number of parameter in different models by assigning a penalizer, p, for using more parameter. Both methods can be used based on the samples from the MCMC run alone. We use the Gelman et al. (2013) notation for the DIC computation:

$$\mathrm{DIC} = -2 log\ p(e_{1:T} \mid \hat{\theta}_{Bayes}) + 2\, \mathrm{p}_{\mathrm{DIC}},$$

where $\hat{\theta}_{Bayes} = \mathbb{E}(\theta \mid e_{1:T})$ is the posterior mean given data $e_{1:T}$. $\mathrm{p}_{\mathrm{DIC}}$ is defined as the variance of all likelihood evaluations of the parameter samples. The WAIC calculation is based on Vehtari et al. (2016):

$$\mathrm{WAIC} = -2\,\mathrm{lppd} + 2\,\mathrm{p}_{\mathrm{WAIC}},$$

where the Log Pointwise Predictive Density (LPPD) is commonly referred to as training error and can be computed as $\mathrm{lppd} = \sum_t^T log \left(\frac{1}{N} \sum_n^N p(e_t \mid \theta_n) \right)$. T is the number of data points and N the number of MCMC samples. The penalizer term $\mathrm{p}_{\mathrm{WAIC}}$ can be computed as $\mathrm{p}_{\mathrm{WAIC}} = \sum_t^T \mathrm{Var}_{\mathrm{t,post}} \left(log\ p(e_t \mid \theta) \right)$, where $\mathrm{Var}_{\mathrm{t,post}} \left(log\ p(e_t \mid \theta) \right)$ is the variance of $p(e_t \mid \theta)$ at time t across all θ samples. After the sequential IBIS estimation is finished, multiple model comparison criteria are available. An often difficult to obtain but highly sought after method is to compare the marginal likelihood $p(e_{1:T}) = \int p(e_{1:T}, \theta)\ d\theta$ of each model. This distribution is typically unavailable, but an estimate for $p(e_t \mid e_{1:t-1})$ can be computed at each time step as a by-product of the IBIS run. Going forward, we refer to $p(e_{t+1} \mid e_{1:t})$ as (one step ahead) Predictive Likelihood (PL) at $t+1$, PL_{t+1}, see Kastner (2016). An estimate for the marginal likelihood can then be easily computed after the IBIS run has finished via the corresponding PL estimates, $\hat{p}(e_{1:T}) = \prod_{t=1}^T \hat{PL}_t$. The cumulative sums of log PLs also allow for direct model comparison via the Cumulative Log Predictive Bayes Factor (CLPBF). To compare model A and B, for $u > 0$, this factor can be computed as

$$CLPBF_{t+1:t+u} = log\left[\frac{p_A(e_{t+u} \mid e_{1:t})}{p_B(e_{t+u} \mid e_{1:t})}\right] = \sum_{i=t+1}^{u} log\left[PL_i(A) - PL_i(B)\right],$$

where a positive value indicates evidence in favor for model A. If $t = 0$ and $u = T$, then the CLPBF defaults to the log Bayes Factor (BF). Note that for all IBIS computations, $1 < t_0 < T$ data points have been used as training period to initialize the jitter kernels in a reasonable parameter region, so the resulting comparison criteria is the slightly different $\hat{p}(e_{t_0+1:T} \mid e_{1:t_0}) = \prod_{t=t_0+1}^{T} \hat{PL}_t$.

4.4 Simulation

4.4.1 Parameter Estimation

In this Section, simulation experiments for all choices of the Stochastic Volatility Model with Copula Dependencies have been performed. Specifically, 1000 data points have been generated and a MCMC sampler as described in Section 4.3 has been used to estimate model parameter for each corresponding Copula. The resulting output statistics are summarized in Table 4.1. It is worth noting that all choices converge rapidly and are close to the parameter used to generate the data.

4.4.2 Model Choice

We established several model selection criteria to determine the most suitable Copula for our proposed application. In this section, we validate these criteria by conducting a simulation using data from a Frank Copula. Subsequently, we estimate all the Copulas proposed in Section 4.5 using this simulated data. We evaluate the performance of each Copula using both batch model comparison criteria, namely the Deviance Information Criterion and Watanabe–Akaike Information Criterion. Additionally, we employ our sequential criterion, the marginal likelihood, to assess all the models. It is noteworthy that the resulting order consistently favors the Frank Copula across all model choices, and even the order of the remaining choices stays consistent across all criteria. Table 4.3 provides a clear demonstration of that.

Clayton										
θ	**True**	**Mean**	**MCSE**	**SD**	**Rhat**	**Q2.5**	**Q25.0**	**Q50.0**	**Q75.0**	**Q97.5**
μ_s	0.0	-0.06	0.0	0.07	1.0	-0.21	-0.11	-0.06	-0.01	0.08
μ_v	0.03	0.03	0.0	0.0	1.0	0.03	0.03	0.03	0.03	0.03
κ	20.0	20.24	0.02	1.11	1.0	18.07	19.51	20.24	20.99	22.42
σ	0.5	0.5	0.0	0.01	1.0	0.49	0.5	0.5	0.5	0.51
c	4.0	3.79	0.0	0.16	1.0	3.47	3.68	3.78	3.89	4.11

Frank										
θ	**True**	**Mean**	**MCSE**	**SD**	**Rhat**	**Q2.5**	**Q25.0**	**Q50.0**	**Q75.0**	**Q97.5**
μ_s	0.0	-0.07	0.0	0.08	1.0	-0.22	-0.12	-0.07	-0.02	0.08
μ_v	0.03	0.03	0.0	0.0	1.0	0.03	0.03	0.03	0.03	0.03
κ	20.0	21.96	0.04	2.83	1.0	16.2	20.08	21.96	23.91	27.44
σ	0.5	0.46	0.0	0.01	1.0	0.44	0.45	0.46	0.46	0.48
c	4.0	3.53	0.0	0.22	1.0	3.1	3.39	3.52	3.67	4.00

Gumbel										
θ	**True**	**Mean**	**MCSE**	**SD**	**Rhat**	**Q2.5**	**Q25.0**	**Q50.0**	**Q75.0**	**Q97.5**
μ_s	0.0	-0.53	0.0	0.09	1.0	-0.71	-0.6	-0.53	-0.47	-0.34
μ_v	0.03	0.04	0.0	0.0	1.0	0.04	0.04	0.04	0.04	0.05
κ	20.0	20.09	0.02	0.9	1.0	18.3	19.48	20.09	20.71	21.86
σ	0.5	0.48	0.0	0.0	1.0	0.47	0.48	0.48	0.49	0.49
c	4.0	4.11	0.0	0.12	1.0	3.88	4.03	4.11	4.19	4.33

Joe										
θ	**True**	**Mean**	**MCSE**	**SD**	**Rhat**	**Q2.5**	**Q25.0**	**Q50.0**	**Q75.0**	**Q97.5**
μ_s	0.0	-0.08	0.0	0.08	1.0	-0.24	-0.13	-0.08	-0.03	0.06
μ_v	0.03	0.03	0.0	0.0	1.0	0.03	0.03	0.03	0.03	0.03
κ	20.0	19.33	0.03	1.3	1.0	16.8	18.45	19.33	20.19	21.94
σ	0.5	0.49	0.0	0.01	1.0	0.47	0.48	0.49	0.49	0.5
c	4.0	3.92	0.0	0.14	1.0	3.66	3.82	3.91	4.01	4.2

BB1										
θ	**True**	**Mean**	**MCSE**	**SD**	**Rhat**	**Q2.5**	**Q25.0**	**Q50.0**	**Q75.0**	**Q97.5**
μ_s	0.0	0.07	0.07	0.0	1.0	-0.07	0.02	0.07	0.13	0.22
μ_v	0.03	0.03	0.0	0.0	1.0	0.02	0.03	0.03	0.03	0.03
κ	20.0	20.31	1.2	0.02	1.0	17.93	19.52	20.29	21.09	22.72
σ	0.5	0.5	0.0	0.0	1.0	0.49	0.5	0.5	0.5	0.51
c_{lower}	2.0	1.99	0.17	0.0	1.0	1.67	1.87	1.99	2.1	2.33
c_{upper}	2.0	1.98	0.1	0.0	1.0	1.79	1.91	1.97	2.04	2.18

BB7										
θ	**True**	**Mean**	**MCSE**	**SD**	**Rhat**	**Q2.5**	**Q25.0**	**Q50.0**	**Q75.0**	**Q97.5**
μ_s	0.0	0.01	0.07	0.0	1.0	-0.13	-0.03	0.01	0.06	0.14
μ_v	0.03	0.03	0.0	0.0	1.0	0.02	0.03	0.03	0.03	0.03
κ	20.0	18.11	1.56	0.03	1.0	15.13	16.99	18.11	19.19	21.08
σ	0.5	0.5	0.01	0.0	1.0	0.49	0.5	0.5	0.51	0.52
c_{lower}	2.0	1.95	0.1	0.0	1.0	1.76	1.89	1.95	2.02	2.15
c_{upper}	2.0	2.08	0.13	0.0	1.0	1.84	1.99	2.08	2.17	2.35

Table 4.1 MCMC summary diagnostics for simulated data with Archimedean Copula dependencies. θ summarizes all model parameter, and c the copula specific parameter.

Gaussian										
θ	**True**	**Mean**	**MCSE**	**SD**	**Rhat**	**Q2.5**	**Q25.0**	**Q50.0**	**Q75.0**	**Q97.5**
μ_s	0.0	0.09	0.0	0.07	1.0	-0.05	0.04	0.09	0.13	0.22
μ_v	0.03	0.02	0.0	0.0	1.0	0.02	0.02	0.02	0.02	0.03
κ	20.0	22.73	0.04	2.03	1.0	18.84	21.32	22.74	24.13	26.57
σ	0.5	0.51	0.0	0.01	1.0	0.49	0.5	0.51	0.51	0.53
c	-0.75	-0.74	0.0	0.01	1.0	-0.77	-0.75	-0.74	-0.74	-0.72
T (2 degrees of freedoom)										
θ	**True**	**Mean**	**MCSE**	**SD**	**Rhat**	**Q2.5**	**Q25.0**	**Q50.0**	**Q75.0**	**Q97.5**
μ_s	0.0	-0.07	0.0	0.07	1.0	-0.2	-0.12	-0.07	-0.02	0.08
μ_v	0.03	0.03	0.0	0.0	1.0	0.03	0.03	0.03	0.03	0.03
κ	20.0	18.74	0.04	1.71	1.0	15.33	17.57	18.69	19.91	22.02
σ	0.5	0.51	0.0	0.01	1.0	0.5	0.5	0.51	0.51	0.52
c	-0.75	-0.75	0.0	0.01	1.0	-0.78	-0.76	-0.75	-0.74	-0.72

Table 4.2 MCMC summary diagnostics for simulated data with Elliptical Copula dependencies. θ summarizes all model parameter, and c the copula specific parameter.

4.5 Applications

4.5.1 Data Processing

We will consider the case where, in addition to the observed log-prices S_t, information can also be made available for V_t through option prices, and assume here for a moment that we work in continuous-time. A typical example is the case of S&P 500 and VIX index pairs, which we use to illustrate our methodology. It can be shown that

$$Y_t := VIX_t^2 \times 10^{-4} = \frac{1}{\tau} E_{\mathbb{Q}} \left(\int_t^T V_s ds \right), \tag{4.4}$$

where $\mathbb{Q}$ is the risk neutral measure and τ denotes the time to expiry for the option prices used in the construction of VIX, known to be equal to 30 days, see for example Lin (2007). Hence, the specification of $\mathbb{Q}$ provides a link between the VIX observations and V_t. Under the Black & Scholes model we get $Y_t^v = V_t$, which we will use going forward. Reverting back to discrete-time, we assume that the complete dataset consists of the log-returns $S = (S_1, \ldots, S_T)$ of assets of interest, and the VIX-informed observations $V = (V_1, \ldots, V_T)$, for some time-horizon $T \geq 1$. For the corresponding evidence used in the estimation runs, 1000 data points for the S&P 500 Total Return and VIX Index have been obtained from Investing.com (2022). Both data sources were obtained as of end-of-day, and are shown in Figure A.21. In a pre-processing step, the stock data has

DIC computations			
Copula	**DIC**	**LPPD**	$\mathbf{p_{DIC}}$
Frank	2559.4	-1275.0	4.7
Gaussian	2603.6	-1297.2	4.5
2-reflected BB1	2660.6	-1324.8	5.5
2-reflected BB7	2677.8	-1333.3	5.6
2-reflected Clayton	2689.4	-1340.1	4.6
2-reflected Gumbel	2735.2	-1363.1	4.5
2-reflected Joe	2863.0	-1427.0	4.5
T (2 degrees of freedoom)	2923.1	-1456.9	4.7

WAIC computations			
Copula	**WAIC**	**LPPD**	$\mathbf{p_{WAIC}}$
Frank	2560.5	-1274.5	5.7
Gaussian	2605.0	-1296.6	5.9
2-reflected BB1	2662.8	-1323.7	7.6
2-reflected BB7	2680.3	-1332.2	8.0
2-reflected Clayton	2691.2	-1339.2	6.3
2-reflected Gumbel	2736.8	-1362.3	6.1
2-reflected Joe	2865.2	-1425.9	6.7
T (2 degrees of freedoom)	2925.1	-1455.9	6.6

SMC model choice	
Copula	**Cum Log Predictive Likelihood**
Frank	-1457.9
Gaussian	-1473.7
2-reflected BB1	-1478.0
2-reflected BB7	-1495.1
2-reflected Clayton	-1500.0
2-reflected Gumbel	-1506.4
2-reflected Joe	-1547.6
T (2 degrees of freedoom)	-1548.9

Table 4.3 Comparison diagnostics as defined in Section 4.3.3 for all SV Models with different Copula choices on simulated data from the favored Copula choice in Section 4.5. The lower tables depicts estimates for the cumulative log PL as defined in Section 4.3.3.

been transformed into log space, $\ell S = \log(S\&P\ 500)$ and VIX data has been rescaled to $X = \log((\frac{VIX}{100})^2)$, see Equation (4.4) as reference.

4.5.2 Model Prior Assignments

The Stochastic Volatility Model with Copula Dependencies has the following parameter: $\theta = \{\mu_s, \mu_x, \kappa, \sigma, copula\}$. The corresponding parameter boundaries for μ_s and μ_x are tackled via prior assignments. The drift parameter μ_s for the transformed stock data has a truncated Normal prior with large variance, $\mu_s \sim Normal_{(-1,1)}(\mu = 0, \sigma = 10^2)$. μ_x, the mean parameter for the variance $\mu_x \sim Normal_{[0,1)}(\mu = 0, \sigma = 10^2)$. The mean reversion parameter κ has a Normal prior distribution as well, $\kappa \sim Normal_{(0,100)}(\mu = 10, \sigma = 10^4)$ and the strictly positive noise term σ has $\sigma \sim Normal_{(0,2)}(\mu = 0.5, \sigma = 10^4)$. The Copula parameter depends on the model corresponding Copula of choice. The parameter for the T and Gaussian Copula was bounded between -1 and 1, the parameter for the Archimedean types were set to be positive. More details can be found directly on https://github.com/paschermayr/Publish_SVCopula.

4.5.3 Marginals Discussion

A factor to note in our model specification from Section 4.2 is that the marginals are based on Gaussian distributions. Given the presence of possible jumps in the price movement as well as the measurement error for the volatility via the VIX index, we apply a sensitivity analysis on the marginals in line with common practice in Copula modelling. To do so, we assign T distributed marginals with unknown degrees of freedom for both stock and volatility, and estimate these separately. Copula specific parameter as well as either stock or volatility parameter, respectively, are ignored while estimating the marginal distribution of the other. We transform the degrees of freedom parameter to $\ell df = \log(df - 2) \sim Normal_{(-10,25)}(\mu = 0.0, \sigma = 10^3)$. The output diagnostics for S and X can be seen in Table 4.4. Note that in this case, we re-scaled the noise term σ in order to use the scaled version of the T distribution. We conclude that the degrees of freedom (in log space) for the stock data converges around 15, and for the volatility data around 3. Consequently, our sensitivity analysis suggests that the Gaussian marginal innovations are suitable enough for the price and volatility process analysis going forward.

S&P 500									
θ	**Mean**	**MCSE**	**SD**	**Rhat**	**Q2.5**	**Q25.0**	**Q50.0**	**Q75.0**	**Q97.5**
μ_s	0.09	0.0	0.07	1.0	-0.03	0.05	0.1	0.14	0.23
$\ell\nu$	16.05	0.2	5.31	1.01	6.95	11.47	16.02	20.71	24.64

VIX									
θ	**Mean**	**MCSE**	**SD**	**Rhat**	**Q2.5**	**Q25.0**	**Q50.0**	**Q75.0**	**Q97.5**
μ_v	0.02	0.0	0.0	1.0	0.02	0.02	0.02	0.02	0.02
κ	22.16	0.06	3.02	1.0	16.15	20.08	22.2	24.25	27.97
σ	0.39	0.0	0.01	1.0	0.37	0.38	0.39	0.4	0.41
$\ell\nu$	2.91	0.0	0.17	1.0	2.58	2.79	2.9	3.01	3.26

Table 4.4 MCMC summary diagnostics on transformed S&P 500 data and VIX data for marginal distribution analysis in Section 4.5.3.

4.5.4 Copula Selection

Choosing the number of available Copulae is based upon empirical considerations. In particular, assuming data information about each of the involved times series data, estimates of the residuals $(\hat{\epsilon}_i, \hat{\zeta}_i)_{i=1,\dots,T}$ can be obtained to perform pairwise exploratory analysis in order to inform appropriate choices of bivariate Copulas. The objective is to propose a reasonable number of different options and carry out model choice via estimation of the model evidence. Preliminary analysis that fit univariate series and investigated pairwise structure of estimated residuals has identified strong negative dependencies between pairs of (residuals of log) prices and volatilities. Going forward, we include all standard Archimedean Copulae such as the Clayton, Gumbel, Frank and Joe variant that feature tail behavior on either side. In addition, we include the 2-parameter Archimedean Copulas BB1 and BB7, which exhibit different tail behavior on both ends. To replicate the negative dependency structure between stock and volatility, the Clayton, Gumbel, Joe, BB1 and BB7 Copulas undergo a process of double reflection (2-reflection). This process is also often referred to as rotation, and a 2-reflection is equivalent to a 270-degree rotation. To account for the difference in tail behavior, we also perform a 1-reflection/90-degree rotation on all Copulas that undergo such a transformation. On the other hand, the Frank copula, despite also belonging to the Archimedean family, does not require any reflection and can be used directly as it is, as it possesses reflection invariance. Next, the elliptical Gaussian and T Copula are included as suitable candidates. The Gaussian Copula, in particular, serves as a benchmark for standard modelling choices of Stochastic Volatility model error terms against our proposed methods. For a review on the corresponding cdf and pdf functions as well as upper and lower tails of any aforementioned Copulae, see Joe and Hu (1996).

4.5.5 Results

We adopted the Gaussian Copula as the benchmark model for all the comparison criteria, as it aligns with the standard framework for modeling error terms in SV models. The sequential model comparison criteria, introduced in Section 4.3.3, are depicted in Figure A.24, which illustrates the Cumulative Log Predictive Bayes Factor at each iteration during the IBIS run. Throughout the entire process, the Frank Copula consistently outperforms all other Copula choices. The values for the final iteration, along with the cumulative log Predictive Likelihood, are presented in Table 4.6, which also includes the results for the batch estimation criteria DIC and WAIC. Notably, the Frank Copula, with a higher concentration of mass towards both tail ends, yields a significantly higher estimate for the marginal likelihood and consistently outperforms other Copula choices in terms of DIC and WAIC. Additional Copula diagnostics, presented in Table 4.7, further validate our intuition regarding the negative relationship between stock prices and their volatility. The trace plots for the favored model, displaying all parameter estimates, can be seen in Figure A.22. These plots are generated after discarding the initial 1000 burn-in iterations and demonstrate rapid convergence. Convergence is further confirmed by the MCMC diagnostics shown in Table 4.5.

To conclude, the Frank as well as the T Copula perform consistently better than the benchmark Gaussian Copula. Notably, the Gumbel and BB1 Copula with asymmetric tail behavior also outperform the benchmark with respect to most comparison criteria.

θ	**Mean**	**MCSE**	**SD**	**Rhat**	**Q2.5**	**Q25.0**	**Q50.0**	**Q75.0**	**Q97.5**
μ_s	-0.1	0.06	0.0	1.0	-0.23	-0.14	-0.1	-0.06	0.03
μ_v	0.03	0.0	0.0	1.0	0.03	0.03	0.03	0.03	0.04
κ	13.04	1.53	0.03	1.0	10.06	11.98	13.05	14.07	16.02
σ	0.49	0.01	0.0	1.0	0.47	0.48	0.49	0.49	0.5
copula	-12.56	0.39	0.01	1.0	-13.32	-12.81	-12.57	-12.29	-11.79

Table 4.5 MCMC summary diagnostics for SV Model with Frank Copula dependencies on real data in Section 4.5.

DIC computations			
Copula	**DIC**	**LPPD**	**p_{DIC}**
Frank	890.2	-440.6	4.5
T	1293.5	-642.0	4.7
1-reflected BB1	1357.8	-673.9	5.0
1-reflected Gumbel	1350.4	-671.0	4.2
Gaussian	1457.7	-724.1	4.8
2-reflected BB1	1494.7	-742.1	5.2
2-reflected Gumbel	1495.9	-743.5	4.5
1-reflected BB7	1625.3	-807.2	5.5
1-reflected Joe	1632.7	-811.6	4.8
2-reflected Clayton	1640.9	-815.9	4.6
2-reflected Joe	1688.4	-897.7	-53.4
1-reflected Clayton	1696.2	-922.0	-73.9

WAIC computations			
Copula	**WAIC**	**LPPD**	**p_{WAIC}**
Frank	895.3	-438.3	9.4
T	1299.5	-638.8	11.0
1-reflected Gumbel	1356.4	-667.8	10.3
1-reflected BB1	1373.2	-668.5	18.1
Gaussian	1473.6	-715.8	21.0
2-reflected Gumbel	1513.2	-735.9	20.7
2-reflected BB1	1520.2	-730.3	29.8
1-reflected Joe	1646.2	-805.2	17.9
2-reflected Clayton	1654.5	-809.4	17.9
1-reflected BB7	1659.9	-792.9	37.0
2-reflected Joe	1768.6	-855.3	29.0
1-reflected Clayton	1801.5	-871.0	29.8

SMC model choice	
Copula	**Cum Log Predictive Likelihood**
Frank	-839.8
T	-914.3
2-reflected BB1	-948.4
1-reflected BB1	-949.7
Gaussian	-950.2
2-reflected Gumbel	-950.3
1-reflected Gumbel	-951.4
2-reflected Joe	-1019.3
1-reflected Clayton	-1029.1
1-reflected BB7	-1032.9
2-reflected Clayton	-1074.2
1-reflected Joe	-1074.3

Table 4.6 Comparison diagnostics as defined in Section 4.3.3 for all SV Models with different Copula choices on real data in Section 4.5. The lower tables depicts estimates for the cumulative log PL as defined in Section 4.3.3.

Model	**LT Q50**	**UT Q50**	**Spearman's ρ Q50**	**Kendall's τ Q50**
1-reflected BB7	0.77	0.41	-0.61	-0.79
2-reflected BB1	0.27	0.73	-0.62	-0.8
1-reflected BB1	0.74	0.02	-0.61	-0.8
2-reflected Gumbel	0.0	0.75	-0.62	-0.8
1-reflected Gumbel	0.75	0.0	-0.61	-0.79
2-reflected Clayton	0.79	0.0	-0.6	-0.78
1-reflected Clayton	0.0	0.8	-0.61	-0.79
2-reflected Joe	0.0	0.81	-0.62	-0.79
1-reflected Joe	0.8	0.0	-0.6	-0.79
Frank	0.0	0.0	-0.62	-0.8
T	0.65	0.65	-0.62	-0.8
Gaussian	0.0	0.0	-0.62	-0.8

Table 4.7 Median estimates of Model implied Copula diagnostics on real data in Section 4.5. LT refers to Lower Tail dependence, and UT refers to Upper Tail dependence, respectively.

4.6 Conclusions

In this Chapter, we explored a new class of Stochastic Volatility Models with Copula dependencies to express the relationship between stock prices and their volatility. We incorporated carefully selected Copula choices and benchmarked them based on different model comparison criteria and their predictive performance, in which the Frank and T Copula functioned consistently better than the other candidates. While Copulas with asymmetric tail behavior did not continually perform better than standard approaches, the aforementioned (symmetric) Copulas outperformed the common choice of Gaussian innovations for SV models normally used in the financial modelling literature. Moving forward, open research topics include proposing a framework to jointly estimate the marginal distributions and joint densities of the data. In particular, non-parametric approaches for the marginals might be suitable in combination with our copula framework. Latent factors could also be imposed to describe the joint dependency structure in the data and could be incorporated seamlessly in our model specification in the form of factor Copulas. Furthermore, considering our objective of accurately describing the (asymmetric) joint dependency structure between stock returns and volatility, it is worth exploring new model choice criteria that specifically emphasize the tails of the distribution. Last but not least, the stochastic volatility model itself could be extended to include jumps.

4.7 Software

Data and code used to run the algorithms in this Chapter can be be seen in https://github.com/paschermayr/Publish_SVCopula. More detailed information about the implementations for running all algorithms and computing all tables can be found in https://github.com/paschermayr/Baytes.jl and its sub-libraries. The corresponding plots are defined in https://github.com/paschermayr/BaytesInference.jl.

There aren't any magical words, really. Words just hold the magic.

Jim Butcher (*Grave Peril*, 2001)

Chapter 5

Concluding Remarks

5.1 Results

This Chapter marks the end of my PhD journey and the beginning of another. It presents a valuable occasion for self reflection, sharing insights gained from my research experience and to offer recommendations to fellow researchers venturing into academia. Moreover, I aim to provide my perspective on the promising avenues of future research within my field, while acknowledging that only time will reveal the (in)accuracy of my predictions.

5.2 Self Reflection

Overall, I am content with my accomplishments. However, there are numerous aspects I would tackle differently a posteriori. Although I have greatly enhanced my coding abilities and contributed to multiple open source packages, these endeavors often demand a substantial amount of time. If you intend to pursue research in a computationally intensive field, it is crucial to recognize that a significant portion of your research time might be dedicated to programming — a resource that could otherwise be allocated to actual research or publishing. While this is not necessarily a drawback and can be invaluable in other domains, it may lead to slower initial progress during your PhD studies. Moreover, recognizing when certain projects are futile is crucial. I have devoted considerable amounts of time to projects that ultimately led nowhere. While this is a common occurrence in academia, I believe it's important to establish a predetermined time frame to tackle a problem. If no significant progress is made within that time frame, it's best to pivot and redirect your efforts. Time passes swiftly, and in order to earn your degree, you must generate results. Thus, maximizing your productivity becomes essential. Looking back, I regret not fully seizing the multitude of opportunities available to me as a PhD student in London. In particular, I wish I had actively participated in more seminars, meetings, and general lectures. I can't help but feel that I missed out on valuable experiences. In retrospect, the benefits of forging new contacts and friendships would have far outweighed the late-night work sessions. While I can't alter the past, I am determined to prioritize such engagements moving forward.

5.3 Advice to new Research Students

Regardless of where this journey leads, I have learned a tremendous amount during the past four years. This growth extends beyond my research topic and transformed my skills as a student, teacher, colleague, and problem-solver. Failure is a crucial part of the research process. Though this can be frustrating in the moment, such experiences contain valuable lessons for the future. At the beginning of my studies, setbacks were particularly daunting but, over the years, I have come to understand this is often one step of the process and fresh ideas can emerge from it. You will develop resilience in the face of failures, although it may be challenging in the beginning. Moreover, it is acceptable to be intimidated by new research tasks. Pursuing a PhD revolves around a deep dive into exciting and unexplored research areas, which can be simultaneously exhilarating and frightening. Even with objectives that seem overly ambitious at first, taking everything one step at a time makes each project more feasible. I found overwhelming tasks a lot more bearable when I split them into smaller pieces and tried to solve them one at a time, it also makes asking questions about things you are uncertain about much clearer. Additionally, it is perfectly fine to not have all the answers. When venturing into a new research area, nobody expects you to be an instant expert. While it's important to thoroughly prepare for meetings, don't hesitate to seek clarification by asking questions if something is unclear. Furthermore, I highly advise aiming to publish your first project as swiftly as feasible. I acknowledge that early projects can greatly benefit from the insights gained later in your research journey and thus be published in better journals. However, as a student, having something already publicly available can alleviate a substantial amount of pressure. Additionally, it enhances your academic visibility and proves beneficial during job interviews when discussing your track record.

5.4 Future Outlook

One objective of this dissertation is supporting the application and awareness of Sequential Monte Carlo algorithms. In particular, automatic model selection and sequential prediction seamlessly integrated in the SMC machinery establish these methods as exceptionally suitable for sequential data. Historically, I believe the two biggest drawbacks preventing widespread use of SMC kernels are the substantial computational complexity and the relative scarcity of available software tools in comparison to MCMC methods. I anticipate a significant reduction in these challenges in the upcoming years, primarily driven by the continuous advancement in computational processing power and the growing influx of open source contributions from both industry and academia. The widespread adoption of Markov Chain Monte Carlo methods, following the influential paper by Hoffman and Gelman (2014) and the introduction of STAN, serves as a testament to their mainstream success. I envision a similar level of achievement for sequential Monte Carlo SMC methods if a team of esteemed experts undertakes their development. Nevertheless, one significant challenge that I encountered in my own research was the necessity to tune the more advanced MCMC kernels while using the SMC framework. It remains unclear a priori how to accomplish this task effectively. While in theory tuning parameter should be fixed after the adaption period in the MCMC kernel, a major strength of Sequential Monte Carlo methods is the adaptability over time if the underlying data process changes. However, this intriguing research area holds great promise and is something I am eager to explore further in the future. Moving forward, I intend to continue working within this research area and cannot wait to discover and explore future developments throughout this domain. Though I had little experience with SMC algorithms when this PhD journey began, I grew to enjoy working with this machinery due to the inference and diagnostics tools that are inaccessible with other methods.

Bibliography

Andrieu, C., Doucet, A., and Holenstein, R. (2010). Particle markov chain monte carlo methods. *Journal of the Royal Statistical Society Series B*, 72(3):269–342.

Andrieu, C. and Roberts, G. O. (2009). The pseudo-marginal approach for efficient monte carlo computations. *Ann. Statist.*, 37(2):697–725.

Baum, L. and Petrie, T. (1966). Statistical inference for probabilistic functions of finite state markov chains. *The Annals of Mathematical Statistics*, 37:1554–1563.

Bera, A. K. and Higgins, M. L. (1993). ARCH models: properties, estimation and testing. *Journal of Economic Surveys*, 7(4):305–366.

Betancourt, M. (2016). Identifying the optimal integration time in hamiltonian monte carlo.

Betancourt, M. (2018). A conceptual introduction to hamiltonian monte carlo.

Bladt, M. and McNeil, A. J. (2022). Time series copula models using d-vines and v-transforms. *Econometrics and Statistics*, 24:27–48.

Brazeau, N. F., Verity, R., Jenks, S., Fu, H., Whittaker, C., Winskill, P., Dorigatti, I., Walker, P. G. T., Riley, S., Schnekenberg, R. P., Hoeltgebaum, H., Mellan, T. A., Mishra, S., Unwin, H. J. T., Watson, O. J., Cucunubá, Z. M., Baguelin, M., Whittles, L., Bhatt, S., Ghani, A. C., Ferguson, N. M., and Okell, L. C. (2022). Estimating the covid-19 infection fatality ratio accounting for seroreversion using statistical modelling. *Commun Med*, 54(2).

Buchholz, A., Chopin, N., and Jacob, P. E. (2020). Adaptive tuning of hamiltonian monte carlo within sequential monte carlo.

Bulla, I. and Bulla, J. (2006). Stylized facts of financial time series and hidden semi-markov models. *Computational Statistics & Data Analysis*, 51:2192–2209.

Cappé, O., Moulines, E., and Ryden, T. (2005). *Inference in Hidden Markov Models (Springer Series in Statistics)*. Springer-Verlag, Berlin, Heidelberg.

Chatzilena, A., Demiris, N., and Kalogeropoulos, K. (2022). A modelling framework for the analysis of the transmission of sars-cov2.

Cherubini, U., Luciano, E., and Vecchiato, W. (2004). *Copula Methods in Finance*. Wiley.

Chib, S. (1995). Marginal likelihood from the gibbs output. *Journal of the American Statistical Association*, 90(432):1313–1321.

Chopin, N. (2002). A sequential particle filter method for static models. *Biometrika*, 89(3):539–552.

Chopin, N., Jacob, P. E., and Papaspiliopoulos, O. (2012). SMC 2: an efficient algorithm for sequential analysis of state space models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 75(3):397–426.

Chopin, N., Jacob, P. E., and Papaspiliopoulos, O. (2013). Smc2: an efficient algorithm for sequential analysis of state-space models.

Chopin, N. and Papaspiliopoulos, O. (2020). *An Introduction to Sequential Monte Carlo*. Springer International Publishing.

Clark, P. K. (1973). A subordinated stochastic process model with finite variance for speculative prices. *Econometrica: Journal of the Econometric Society*, pages 135–155.

Cohen, J. E. (1992). Infectious Diseases of Humans: Dynamics and Control. *JAMA*, 268(23):3381–3381.

Corenflos, A., Thornton, J., Deligiannidis, G., and Doucet, A. (2021). Differentiable particle filtering via entropy-regularized optimal transport.

Craiu, R. V. and Rosenthal, J. S. (2014). Bayesian computation via markov chain monte carlo. *Annual Review of Statistics and Its Application*, 1(1):179–201.

Crisan, D. and Miguez, J. (2017). Nested particle filters for online parameter estimation in discrete-time state-space markov models.

Czado, C., Ivanov, E., and Okhrin, Y. (2019). Modelling temporal dependence of realized variances with vines. *Econometrics and Statistics*, 12:198–216.

Dai, C., Heng, J., Jacob, P. E., and Whiteley, N. (2020). An invitation to sequential monte carlo samplers.

Daviet, R. (2018). Inference with hamiltonian sequential monte carlo simulators.

Del Moral, P., Doucet, A., and Jasra, A. (2006). Sequential monte carlo samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(3):411–436.

Dewar, M., Wiggins, C., and Wood, F. (2012). Inference in hidden markov models with explicit state duration distributions. *IEEE Signal Processing Letters*, 19:235–238.

Diekmann, O., Heesterbeek, H., and Britton, T. (2012). *Mathematical Tools for Understanding Infectious Disease Dynamics*. Princeton University Press.

Douc, R. and Cappe, O. (2005). Comparison of resampling schemes for particle filtering.

Doucet, A. and Johansen, A. (2011). A tutorial on particle filtering and smoothing: Fifteen years later.

Dureau, J., Kalogeropoulos, K., and Baguelin, M. (2012). Capturing the time-varying drivers of an epidemic using stochastic dynamical systems.

Engle, R. F. (1982). Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica*, 50(4):987–1007.

Fearnhead, P. and Taylor, B. M. (2010). An adaptive sequential monte carlo sampler.

Flaxman, S., Mishra, S., Gandy, A., Unwin, H. J. T., Mellan, T. A., Coupland, H., Whittaker, C., Zhu, H., Berah, T., Eaton, J. W., et al. (2020a). Estimating the effects of non-pharmaceutical interventions on covid-19 in europe. *Nature*, 584(7820):257–261.

Flaxman, S., Mishra, S., and Gandy, A. e. a. (2020b). Estimating the effects of non-pharmaceutical interventions on covid-19 in europe. *Nature*, 584(257-261).

Fruehwirth-Schnatter, S. (2006). *Finite mixture and Markov switching models*. Springer, 1st edition.

Fruehwirth-Schnatter, S., Celeux, G., and Robert, C. P. (2018). *Handbook of Mixture Analysis*. Chapman & Hall/CRC Handbooks of Modern Statistical Methods. CRC Press.

Ge, H., Xu, K., and Ghahramani, Z. (2018). Turing: a language for flexible probabilistic inference. In *International Conference on Artificial Intelligence and Statistics, AISTATS 2018, 9-11 April 2018, Playa Blanca, Lanzarote, Canary Islands, Spain*, pages 1682–1690.

Gelman, A., Hwang, J., and Vehtari, A. (2013). Understanding predictive information criteria for bayesian models.

Ghysels, E., Harvey, A. C., and Renault, E. (1996). Stochastic volatility. *Handbook of Statistics*, 14:119–191.

Golightly, A. (2009). Bayesian filtering for jump-diffusions with application to stochastic volatility. *Journal of Computational and Graphical Statistics*, 18(2):384–400.

GOV.UK (2022). Gov.uk coronavirus (covid-19) in the uk. https://coronavirus.data.gov.uk/, Last accessed on 2022-06-30.

Gunawan, D., Kohn, R., and Tran, M. N. (2021). Robust particle density tempering for state space models.

Hadj-Amar, B., Jewson, J., and Fiecas, M. (2022). Bayesian Approximations to Hidden Semi-Markov Models for Telemetric Monitoring of Physical Activity. *Bayesian Analysis*, pages 1 – 31.

Hoffman, M. D. and Gelman, A. (2014). The no-u-turn sampler: Adaptively setting path lengths in hamiltonian monte carlo. *The Journal of Machine Learning Research*, 15(1):1593–1623.

Hull, J. and White, A. (1987). The pricing of options on assets with stochastic volatilities. *The Journal of Finance*, 42(2):281–300.

Investing.com (2022). Investing.com. https://www.investing.com/, Last accessed on 2022-10-30.

Joe, H. (1997). *Multivariate Models and Multivariate Dependence Concepts*. Chapman and Hall/CRC, 1st edition.

Joe, H. (2014). *Dependence modeling with copulas*. CRC Press.

Joe, H. and Hu, T. (1996). Multivariate distributions from mixtures of max-infinitely divisible distributions. *Journal of multivariate analysis*, 57(2):240–265.

Johansen, A. and Doucet, A. (2008). A note on auxiliary particle filters. *Statistics & Probability Letters*, 78(12).

Johnson, M. J. (2014). *Bayesian Time Series Models and Scalable Inference*. PhD thesis, MIT.

Johnson, M. J. and Willsky, A. S. (2013). Bayesian nonparametric hidden semi-markov models. *Journal of Machine Learning Research*, 14:673–701.

Kalogeropoulos, K., Roberts, G. O., and Dellaportas, P. (2010). Inference for stochastic volatility models using time change transformations. *The Annals of Statistics*, 38(2):784 – 807.

Kantas, N., Doucet, A., Singh, S. S., Maciejowski, J., and Chopin, N. (2015). On particle methods for parameter estimation in state-space models. *Statistical Science*, 30(3):328–351.

Kastner, G. (2016). Dealing with stochastic volatility in time series using the r package stochvol. *Journal of Statistical Software, Articles*, 69(5):1–30.

Kim, S., Shephard, N., and Chib, S. (1998). Stochastic volatility: likelihood inference and comparison with ARCH models. *The Review of Economic Studies*, 65(3):361–393.

Krupskii, P. and Joe, H. (2020). Flexible copula models with dynamic dependence and application to financial data. *Econometrics and Statistics*, 16:148–167.

Levin, A. T., Hanage, W. P., Owusu-Boaitey, N., Cochran, K. B., Walsh, S. P., and Meyerowitz-Katz, G. (2020). Assessing the age specificity of infection fatality rates for covid-19: systematic review, meta-analysis, and public policy implications.

Lin, Y.-N. (2007). Pricing VIX futures: Evidence from integrated physical and risk-neutral probability measures. *Journal of Futures Markets*, 27(12):1175–1217.

Lindsten, F., Bunch, P., Singh, S. S., and Schön, T. B. (2015). Particle ancestor sampling for near-degenerate or intractable state transition models.

Lindsten, F., Jordan, M. I., and Schön, T. B. (2014). Particle gibbs with ancestor sampling.

Lindsten, F. and Schön, T. (2013). Backward simulation methods for monte carlo statistical inference. *Foundations and Trends in Machine Learning*, 6:1–142.

Liu, J., Xie, W., Wang, Y., Xiong, Y., Chen, S., Han, J., and Wu, Q. (2020). A comparative overview of covid-19, mers and sars: Review article. *International journal of surgery*, 81(1-8).

Murphy, K. (2002). Hidden semi-markov models (hsmms).

Neal, R. M. (2012). Mcmc using hamiltonian dynamics.

Ning, C., Xu, D., and Wirjanto, T. S. (2008). Modeling the leverage effect with copulas and realized volatility. *Finance Research Letters*, 5(4):221–227.

Petrosillo, N., Viceconte, G., Ergonul, O., Ippolito, G., and Petersen, E. (2020). Covid-19, sars and mers: are they closely related? *Clinical Microbiology and Infection*, 26(6).

Pitt, M. and Shephard, N. (1999). Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, 94(446):590–599.

Pohle, J., Adam, T., and Beumer, L. T. (2021). Flexible estimation of the state dwell-time distribution in hidden semi-markov models.

Rackauckas, C. and Nie, Q. (2017). Differentialequations.jl–a performant and feature-rich ecosystem for solving differential equations in julia. *Journal of Open Research Software*, 5(1):15.

Rousseau, J. and Mengersen, K. (2011). Asymptotic behaviour of the posterior distribution in overfitted mixture models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(5):689–710.

Salvatier, J., Wiecki, T. V., and Fonnesbeck, C. (2016). Probabilistic programming in python using PyMC3. *PeerJ Computer Science*, 2:e55.

Shephard, N. (1996). Statistical aspects of ARCH and stochastic volatility. *Monographs on Statistics and Applied Probability*, 65:1–68.

Spiegelhalter, D. J., Best, N. G., Carlin, B. P., and Van Der Linde, A. (2002). Bayesian measures of model complexity and fit. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 64(4):583–639.

Stan Development Team (2021). Stan modeling language users guide and reference manual.

Vehtari, A., Gelman, A., and Gabry, J. (2016). Practical bayesian model evaluation using leave-one-out cross-validation and WAIC. *Statistics and Computing*, 27(5):1413–1432.

Verity, R., Okell, L. C., Dorigatti, I., Winskill, P., Whittaker, C., Imai, N., Cuomo-Dannenburg, G., Thompson, H., Walker, P. G., Fu, H., et al. (2020). Estimates of the severity of coronavirus disease 2019: a model-based analysis. *The Lancet infectious diseases*, 20(6):669–677.

Visani, G. M., Lee, A. H., Nguyen, C., Kent, D. M., Wong, J. B., Cohen, J. T., and Hughes, M. C. (2021). Approximate bayesian computation for an explicit-duration hidden markov model of covid-19 hospital trajectories.

Watanabe, S. (2010). Asymptotic equivalence of bayes cross validation and widely applicable information criterion in singular learning theory.

Wearing, H., Rohani, P., and Keeling, M. (2005). Appropriate models for the management of infectious diseases.

Xiao, Q., Fang, Y., Liu, Q., and Zhou, S. (2018). Online machine health prognostics based on modified duration-dependent hidden semi-markov model and high-order particle filtering. *The International Journal of Advanced Manufacturing Technology*, 94(1):1283–1297.

Yu, S. (2016). *Hidden Semi-Markov Models: Theory, Algorithms and Applications*. Elsevier, Boston.

Yu, S.-Z. (2010). Hidden semi-markov models. *Artificial Intelligence*, 174(2):215–243. Special Review Issue.

Appendix A

Plots

Chapter 2 Plots

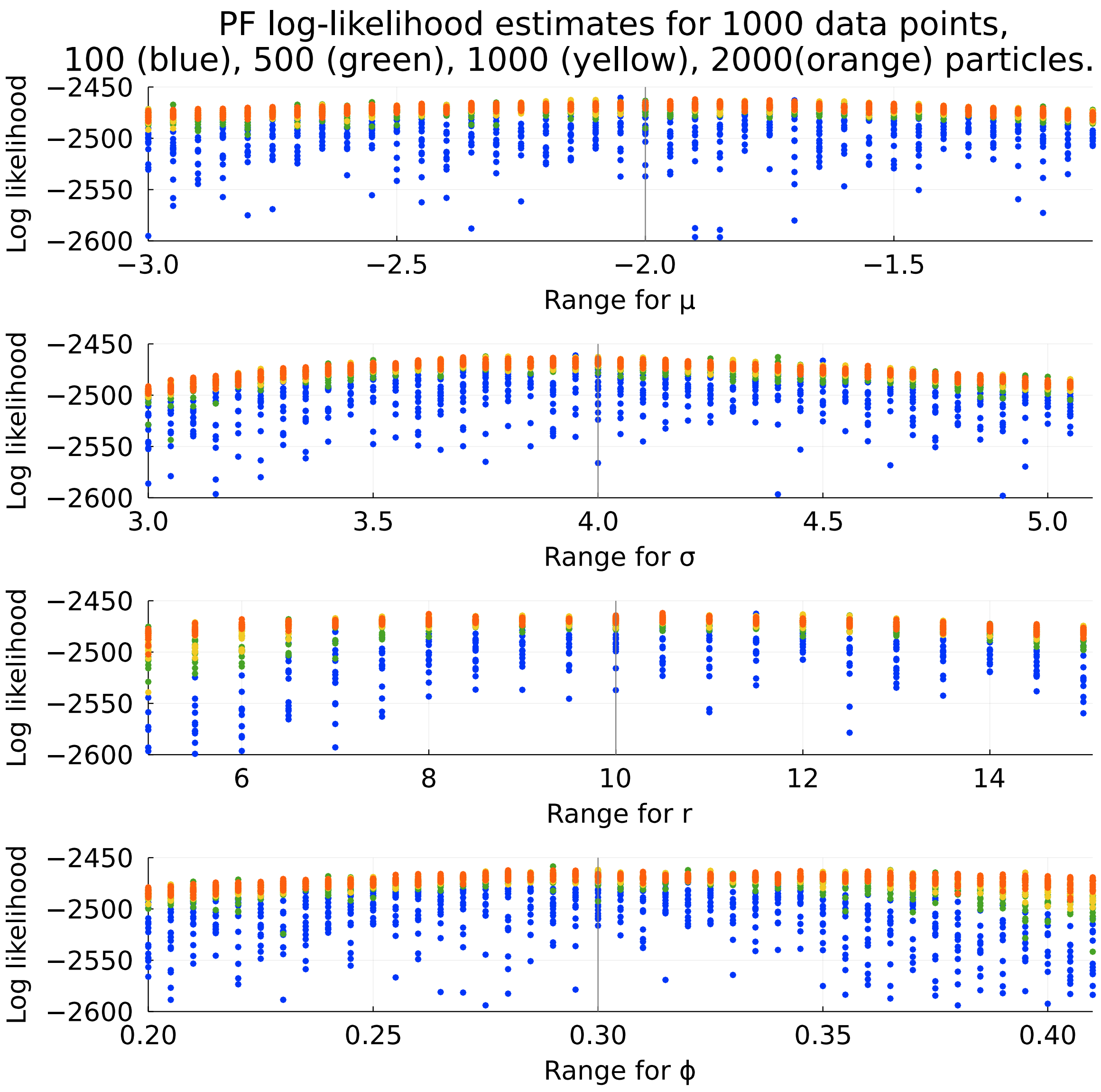


Figure A.1 This graph shows Particle Filter likelihood estimates for a range of different parameter values. At each column, all parameter were kept constant except the labeled parameter at the x-axis. The different colors depict various amount of particles used for the computations: 100 (blue), 500 (green), 1000 (yellow), 2000 (orange) particles for sample data of size 1000.

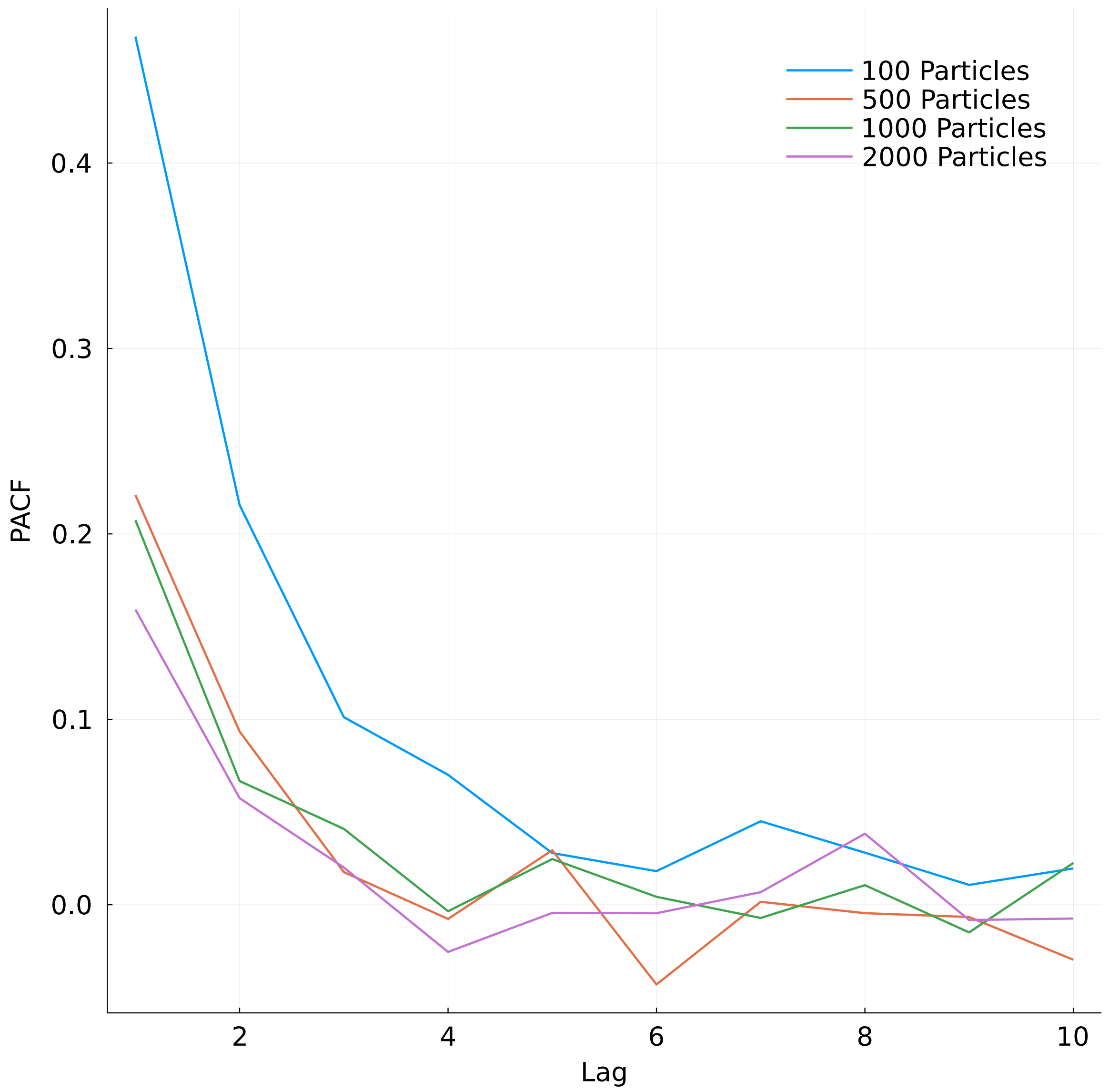


Figure A.2 This graph shows the Partial Autocorrelation Function of Particle Filter likelihood estimates that have been obtained during 1000 PMCMC steps after burnin. The Particle Filter in the PMCMC kernels were set to have a different amount of particles for each run.

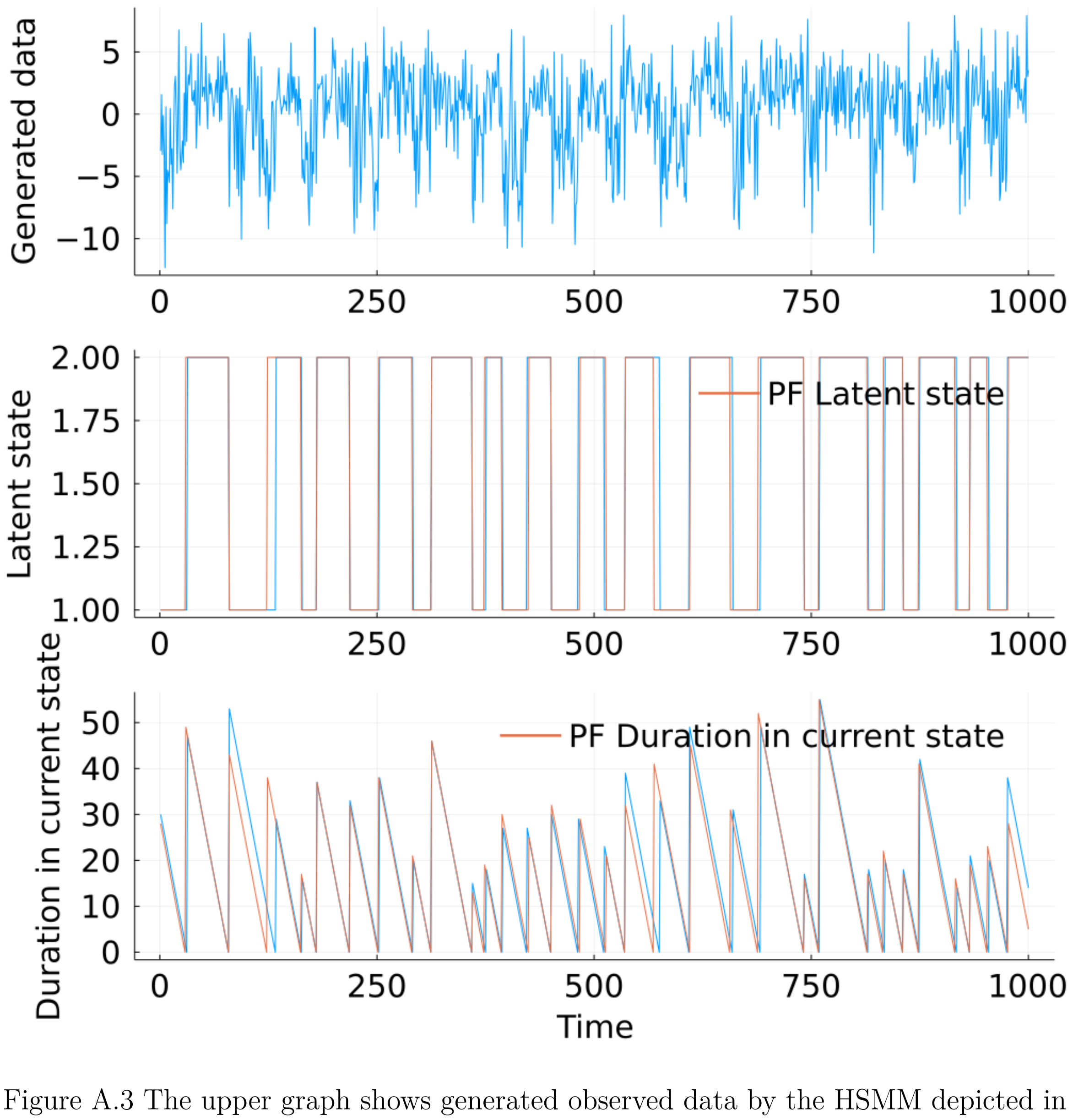


Figure A.3 The upper graph shows generated observed data by the HSMM depicted in section 2.4. The middle plot shows the hidden state (blue) and a sample of a filtered trajectory from a particle filter. The lower plot shows the remaining duration given the current state, and a sample of a filtered particles from a particle filter.

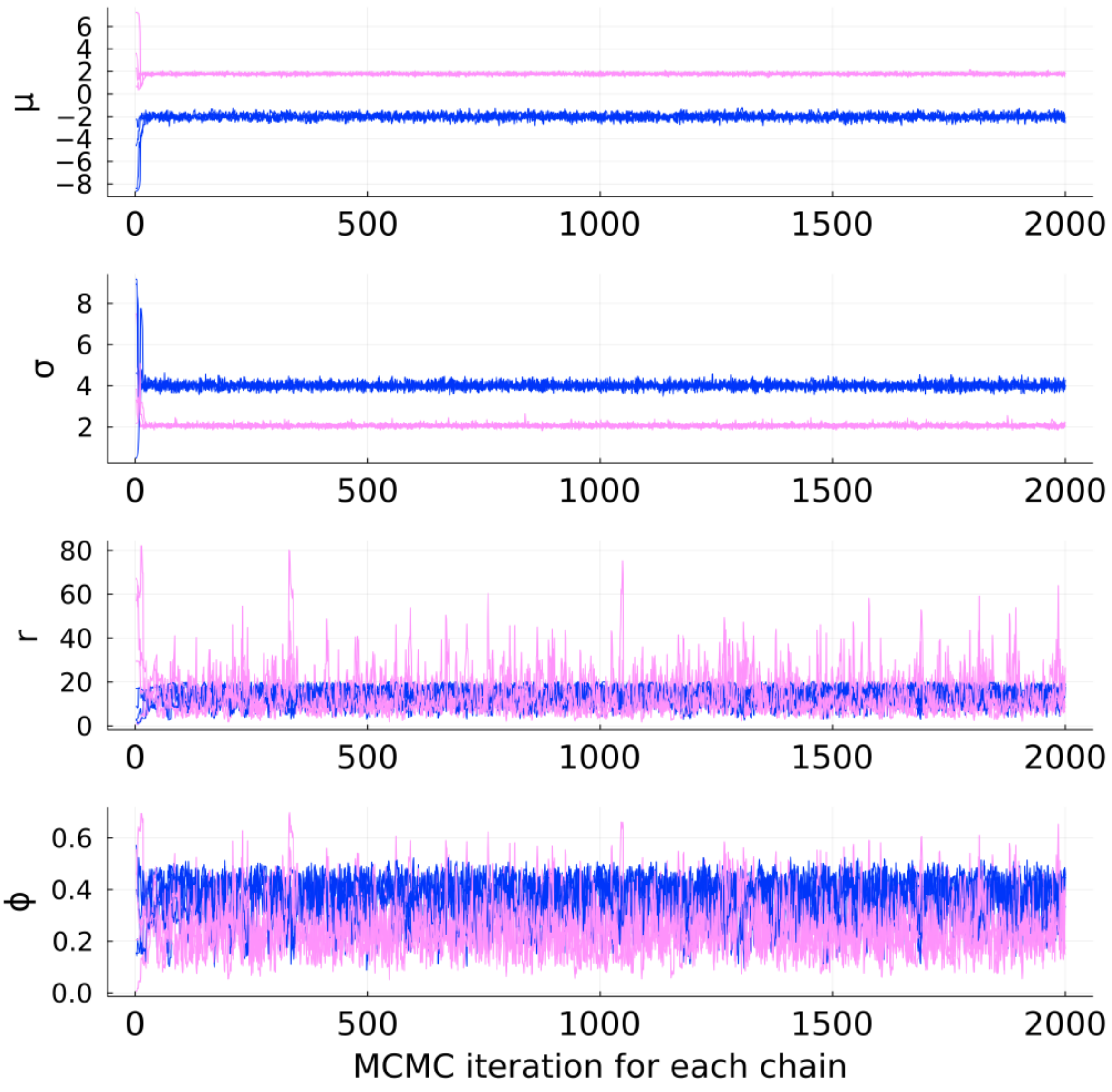


Figure A.4 Traceplots of four Particle MCMC chains for continuous model parameter of a HSMM in section 2.4.

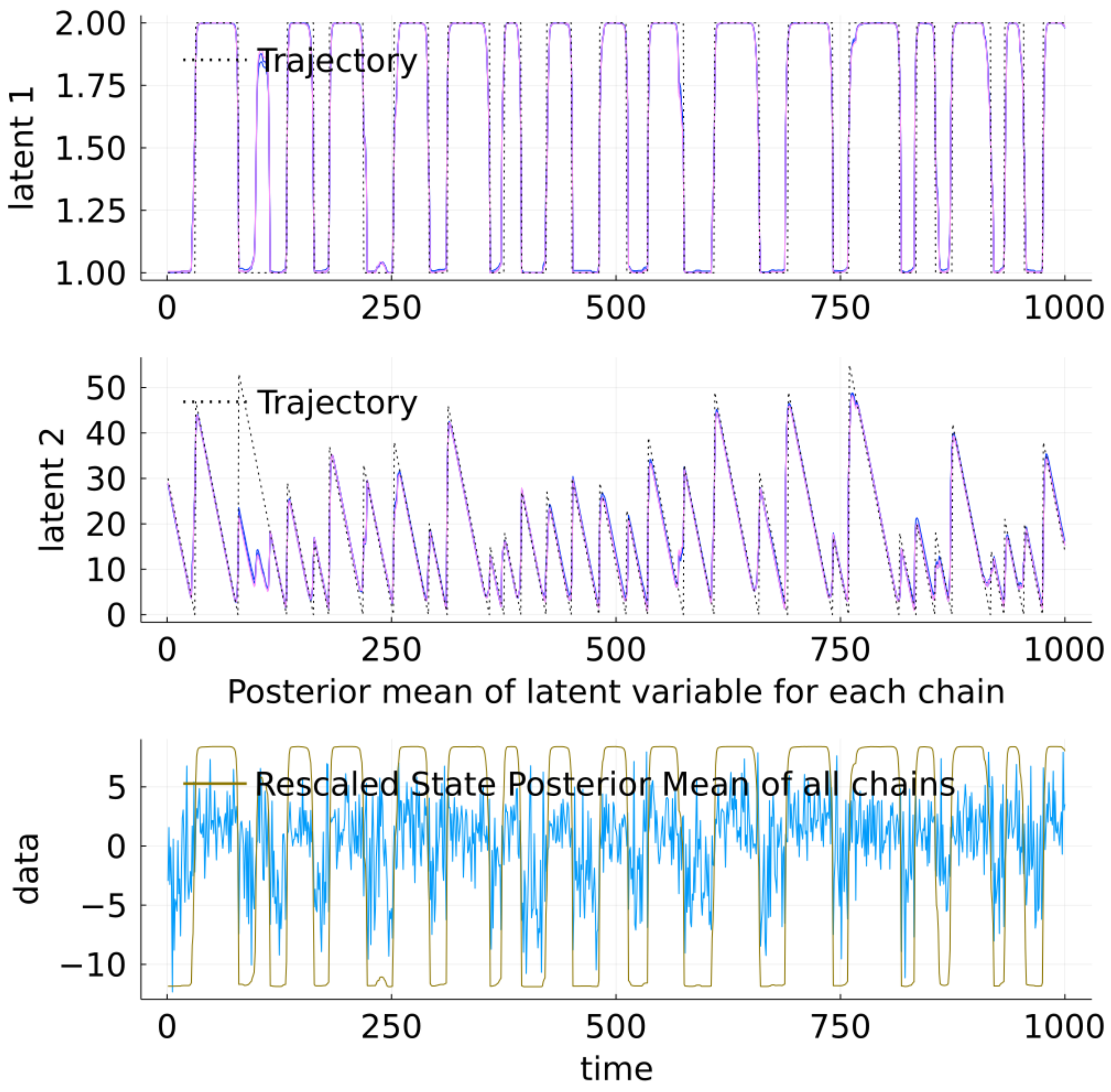


Figure A.5 Particle MCMC posterior estimates of the filtered latent state trajectory of four chains for the HSMM in section 2.4. Parameter used to generate sample data are shown as dashed lines. The bottom plot shows re-scaled posterior means and the observed data.

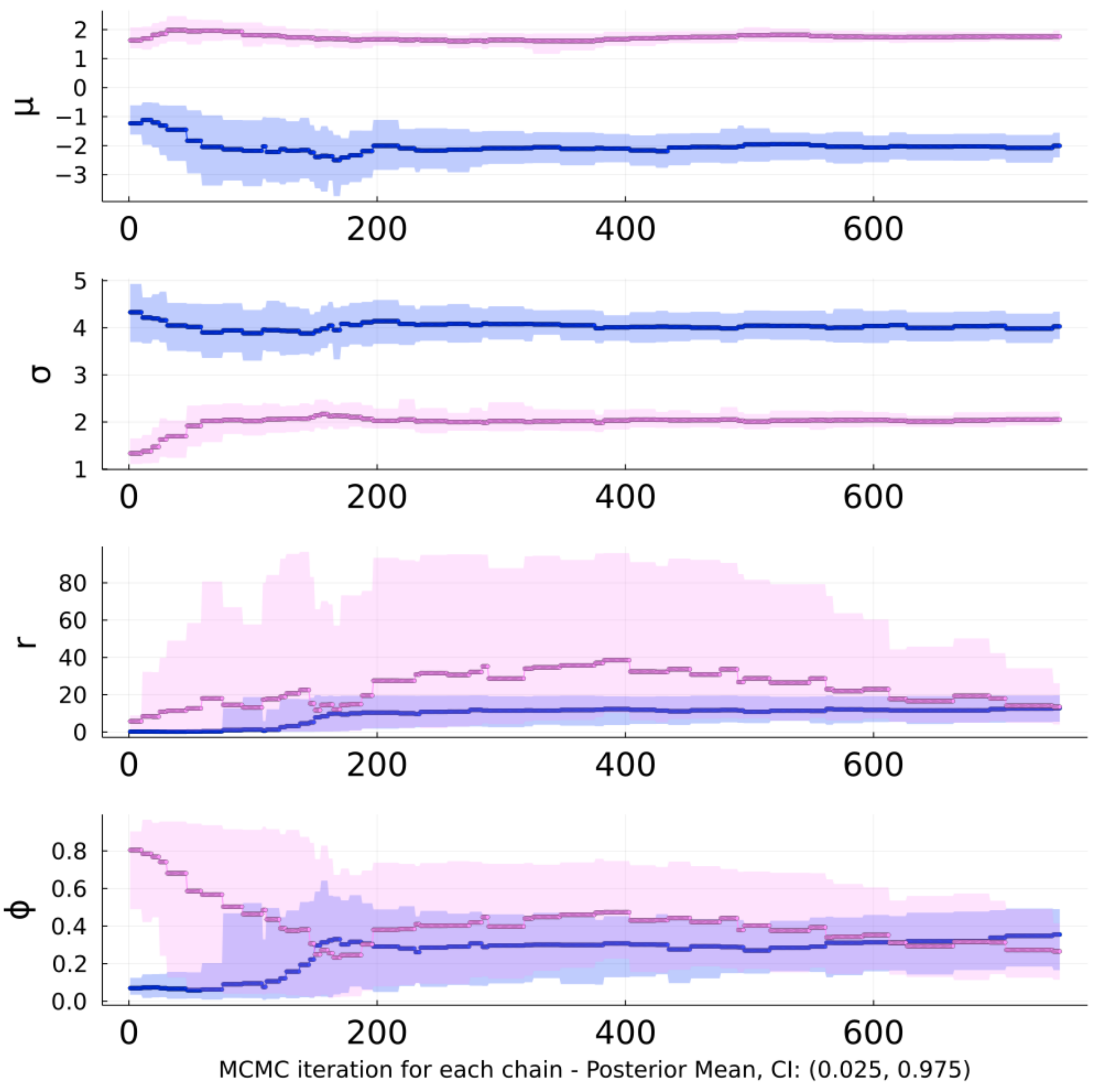


Figure A.6 Sequential Monte Carlo Squared posterior estimates of the continuous model parameter of 100 chains for the HSMM in section 2.4. The posterior mean and a 95% Credible Interval are provided for each parameter at each time index.

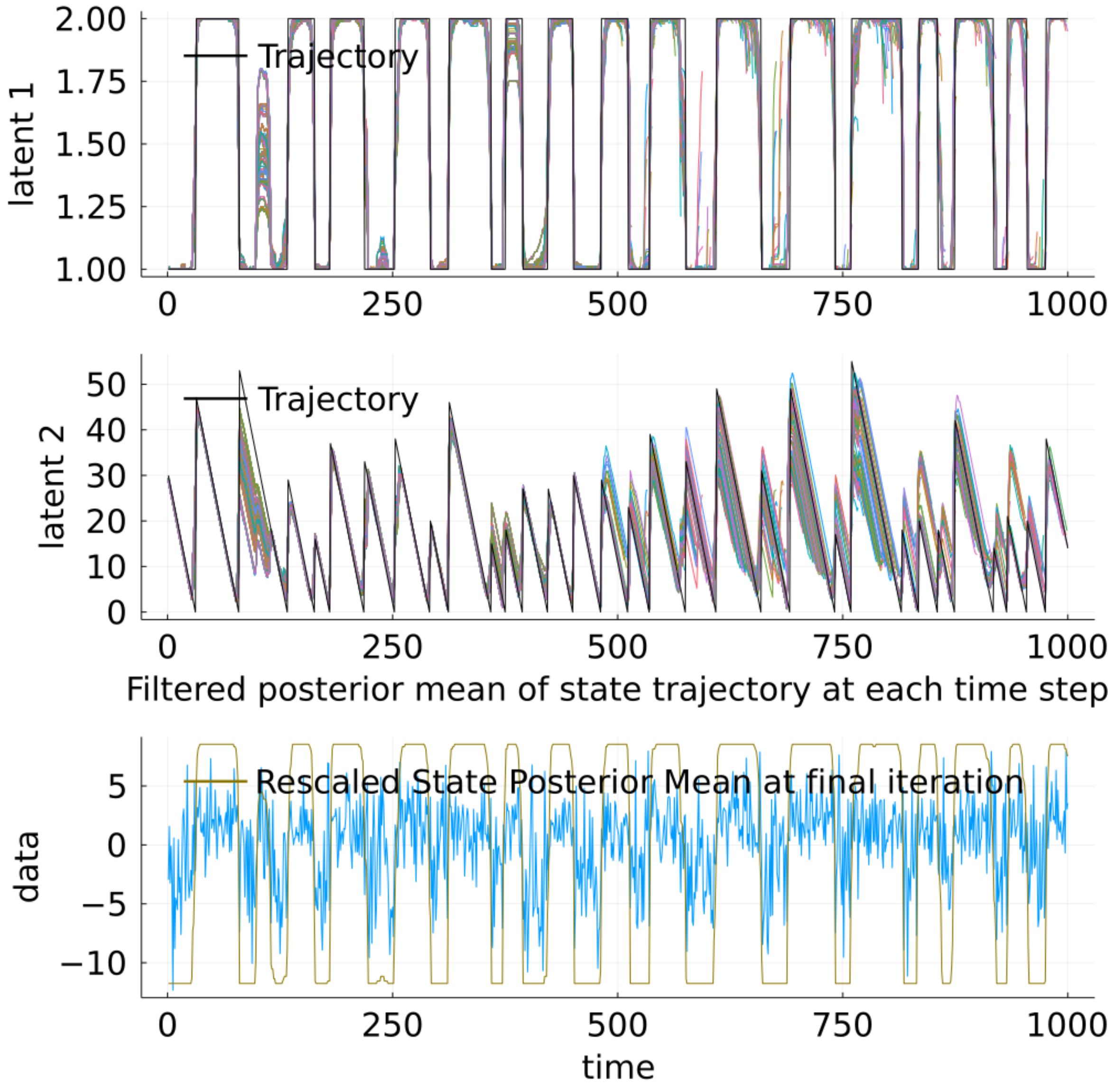


Figure A.7 Sequential Monte Carlo posterior estimates of the filtered latent state trajectory of 100 chains for the HSMM in section 2.4 at each time index. The bottom plot shows the underlying observed data and re-scaled posterior mean of the latent state at the final iteration. Parameter used to generate sample data are shown as dashed horizontal lines.

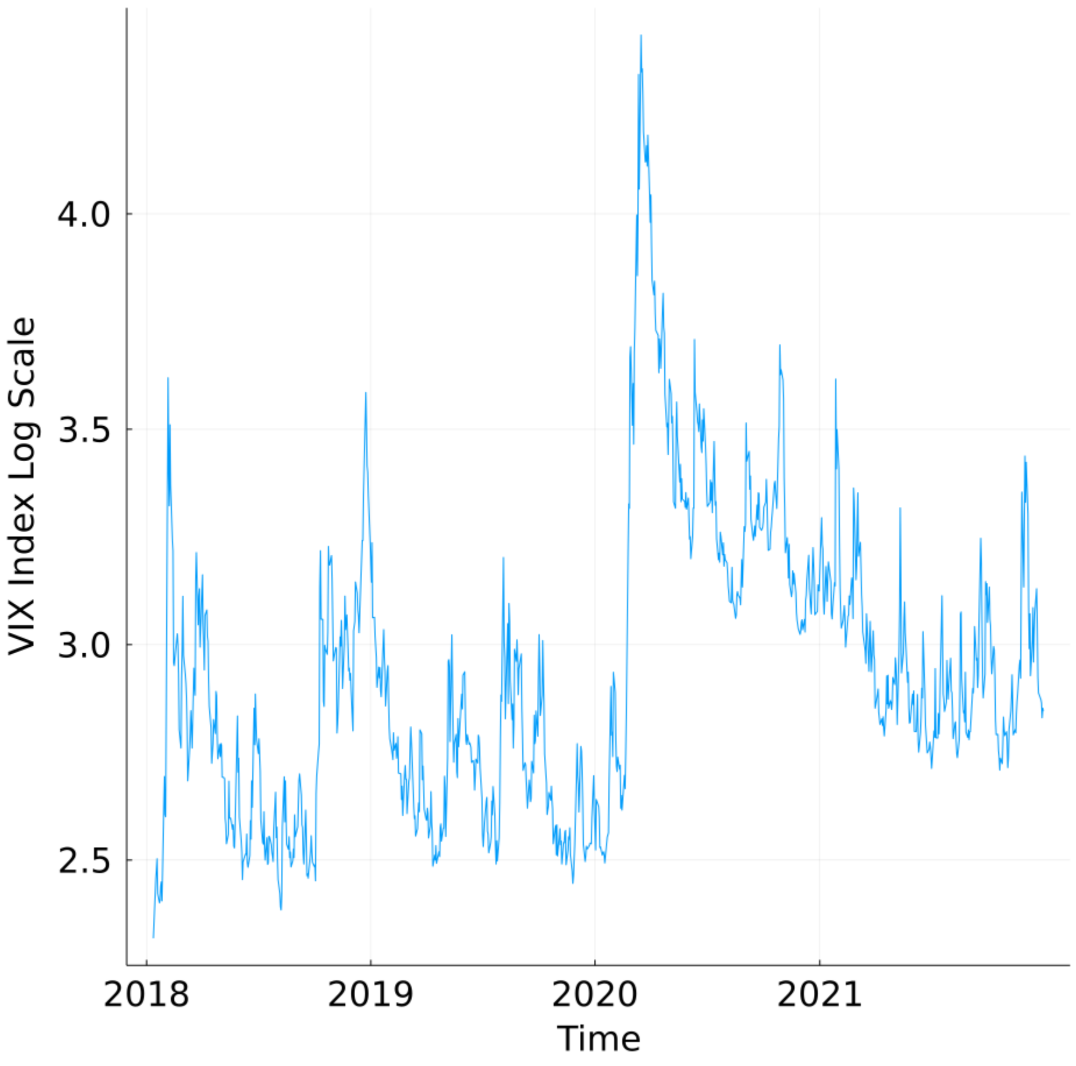


Figure A.8 Last 1000 end-of-day data points of the VIX Index used as data in Section 2.5. Data as of January 1st 2022 from the Thomson Reuters database.

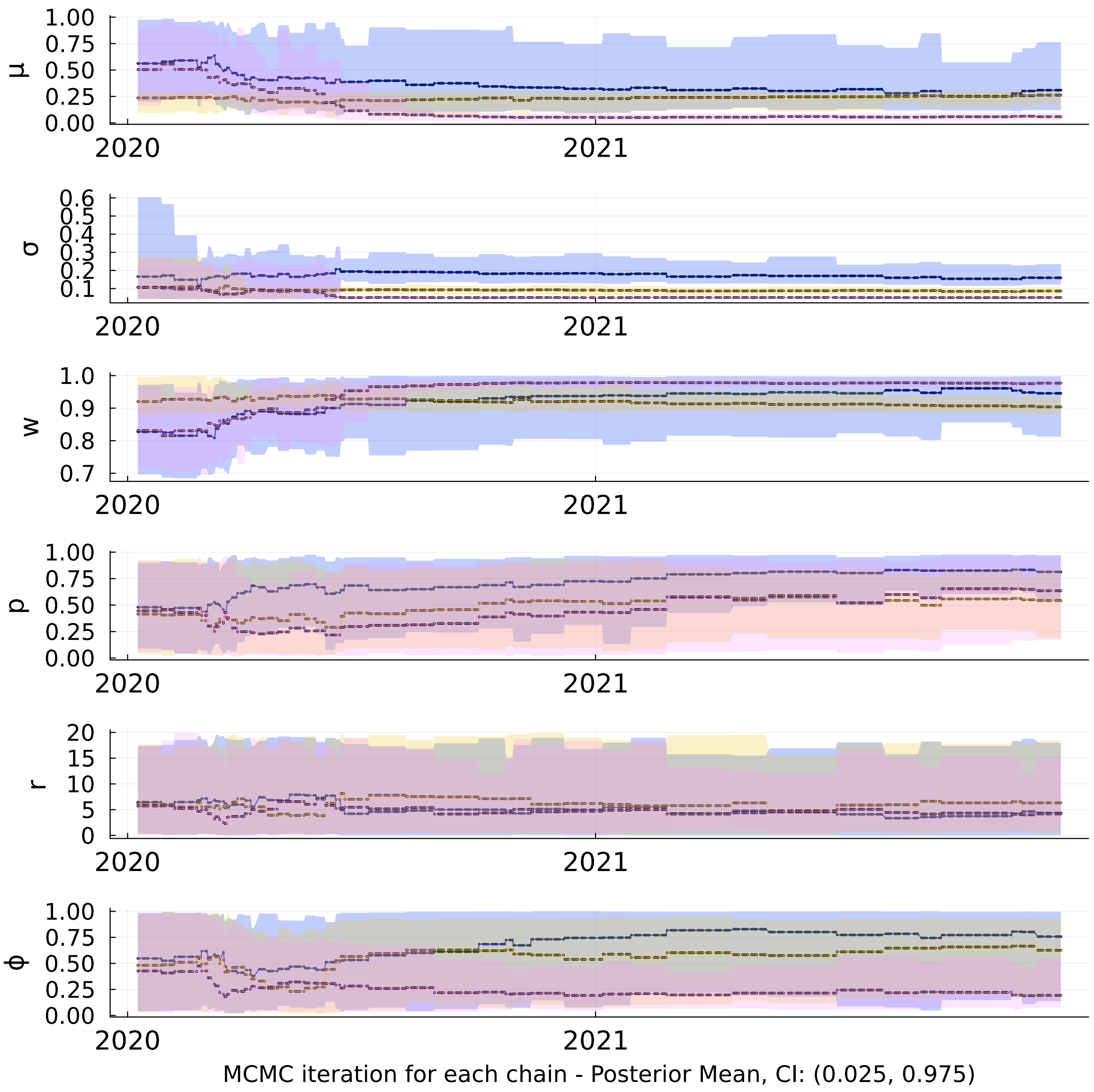


Figure A.9 Sequential Monte Carlo posterior estimates of the continuous model parameter of 100 chains for the AR(1) HSMM with Negative Binomial duration distribution, discussed in section 2.5. The posterior mean and a 95% Credible Interval are provided for each parameter at each time index.

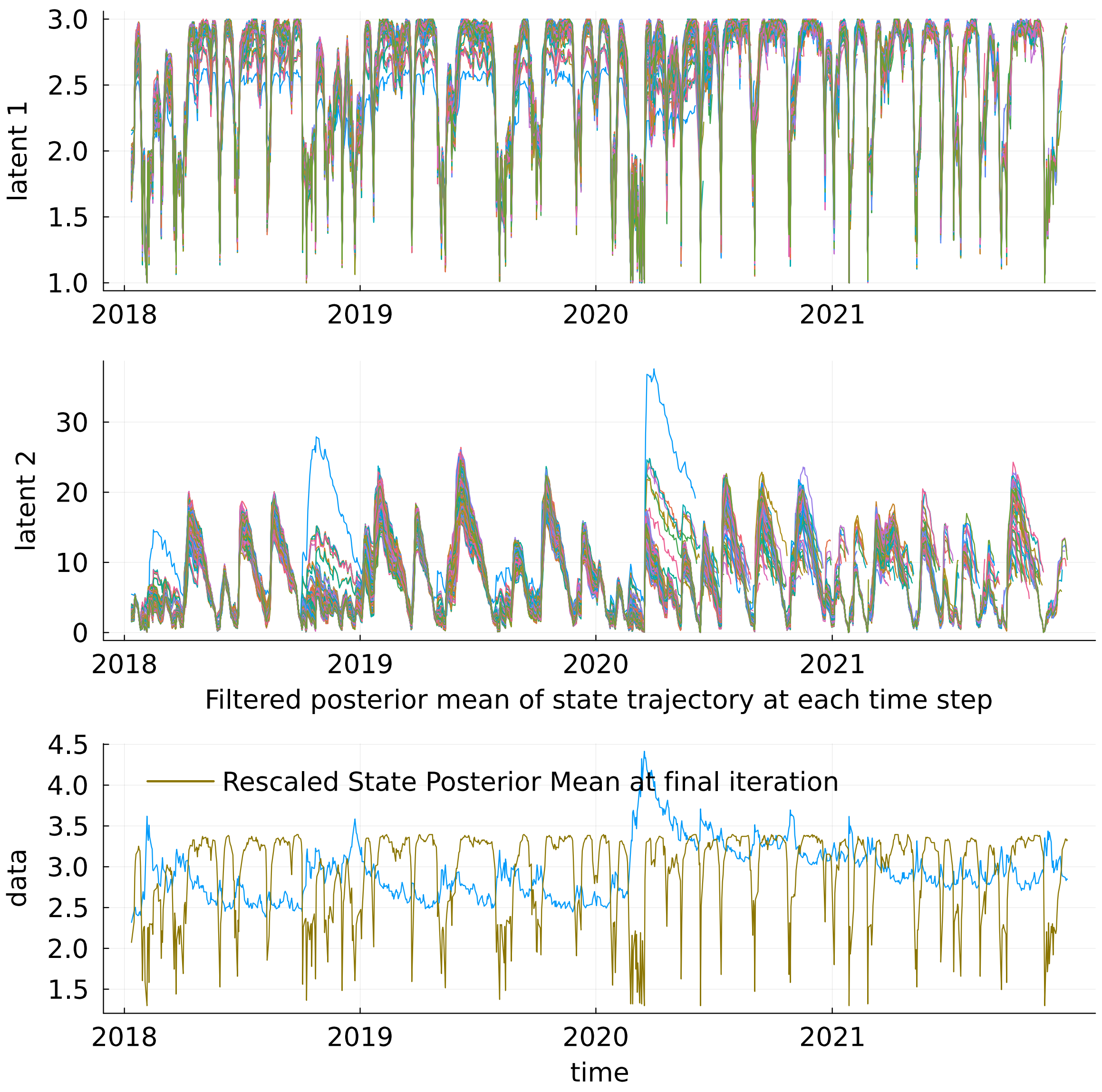


Figure A.10 Sequential Monte Carlo posterior estimates of the filtered latent state trajectory of 100 chains for the AR(1) HSMM with Negative Binomial duration distribution, discussed in section 2.5. The bottom plot shows the underlying observed data and re-scaled posterior mean of the latent state (rounded to closest integer) at the final iteration.

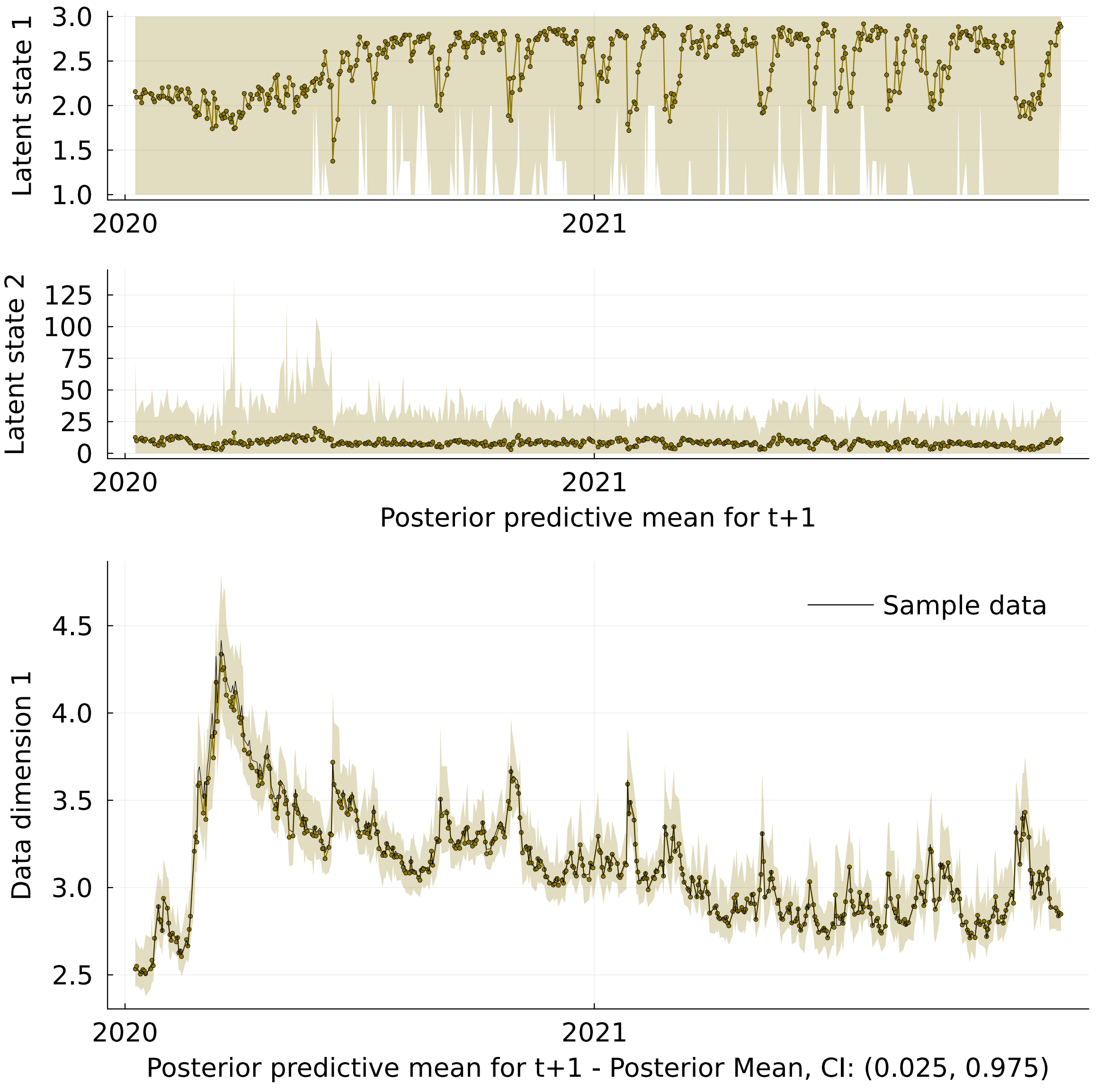


Figure A.11 Sequential Monte Carlo Squared posterior predictive samples for the AR(1) HSMM with Negative Binomial duration distribution, discussed in section 2.5. The Black line at the bottom table depicts the realized future value against predictions in gold. The top 2 graph are predictions for the state and duration variables, and the bottom plot shows predictions for the observed data.

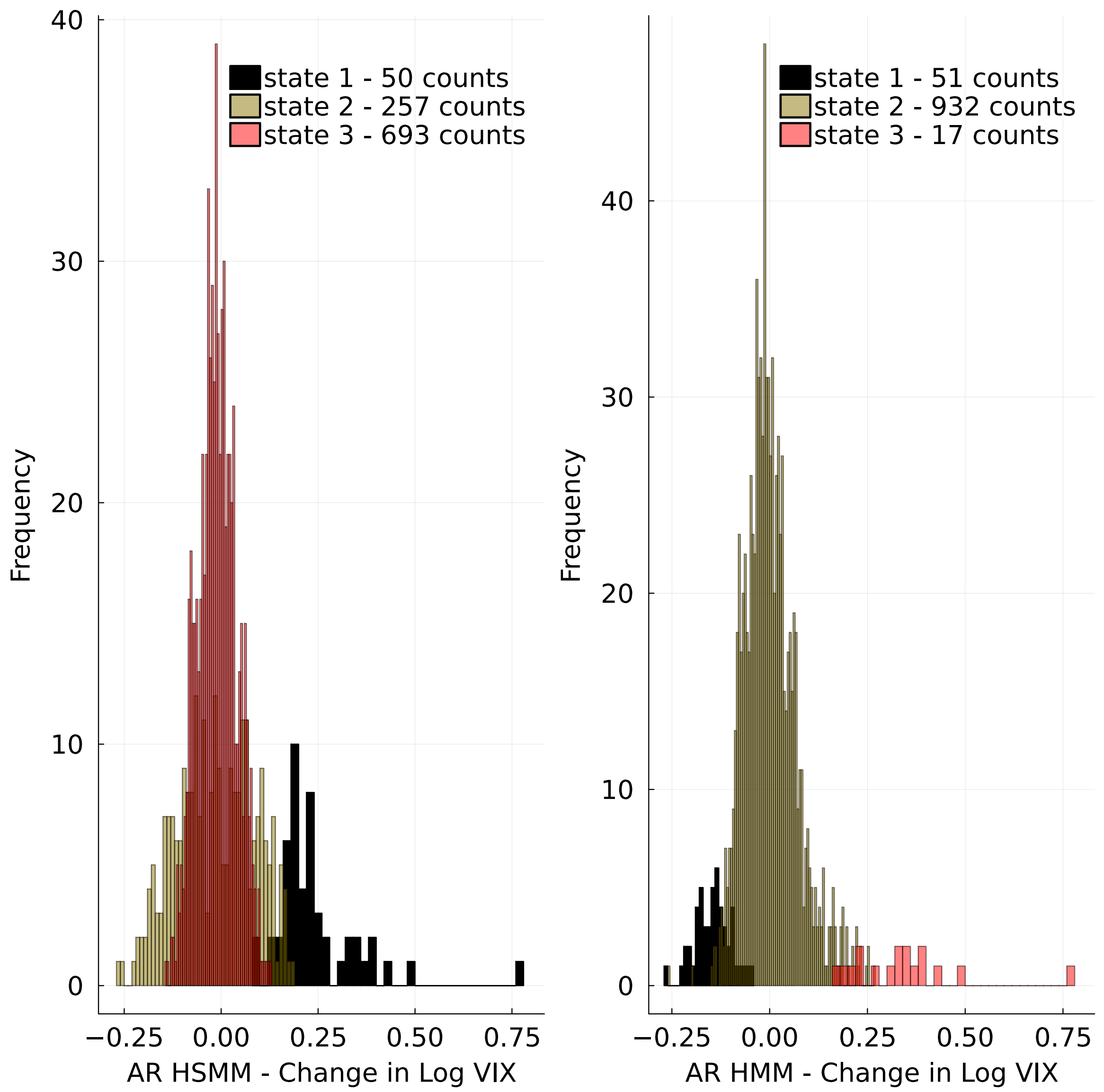


Figure A.12 Both plot depict a histogram for changes in log VIX data, conditioned on the most probable posterior latent state from the final SMC2 iteration. The left uses the states from the winning ARHSMM, while the right plot uses the corresponding ARHMM data.

Chapter 3 Plots

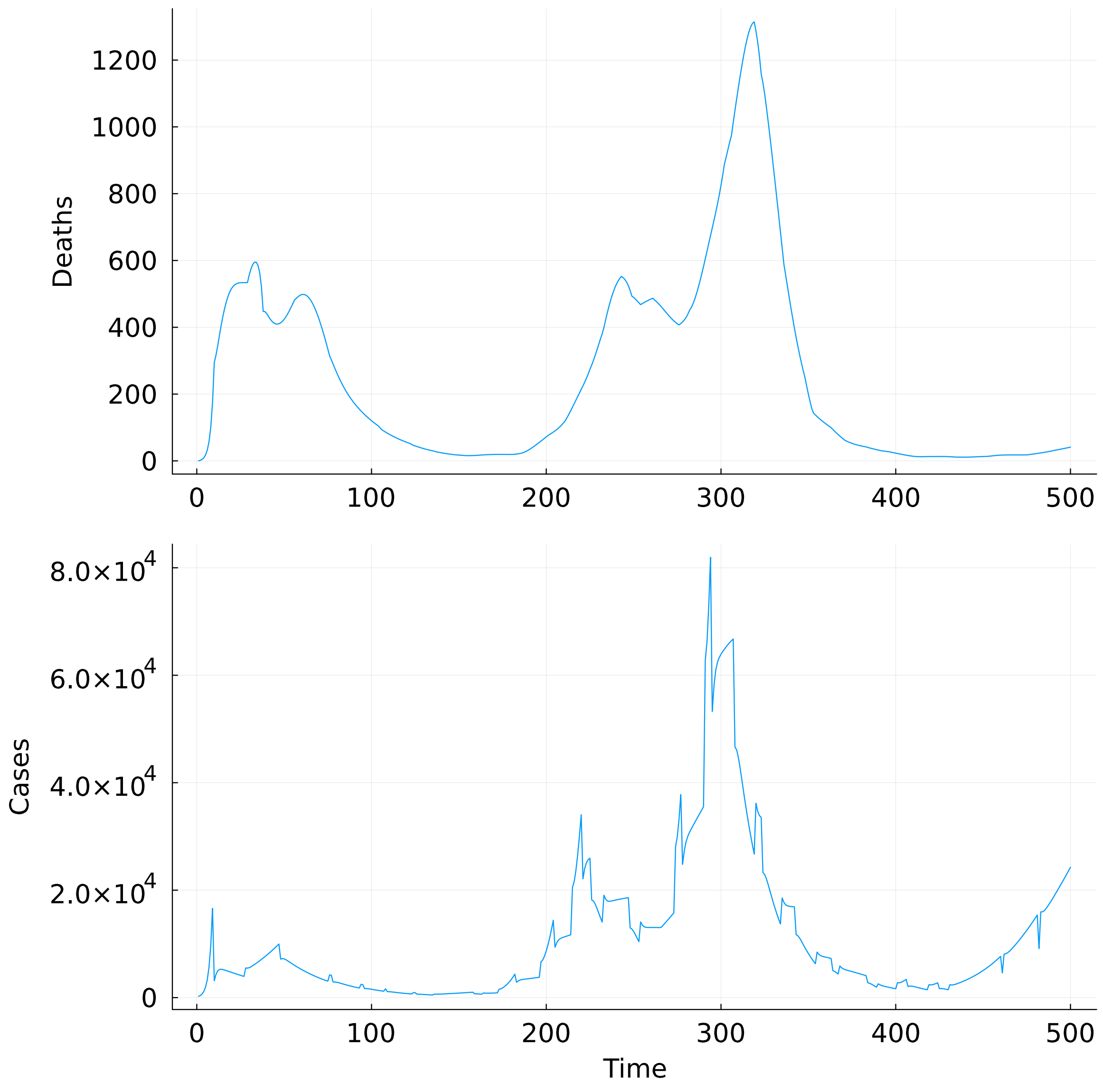


Figure A.13 Generated data for simulated HSMM-EM in Section 3.4. The upper graph shows generated model implied deaths, computed as described in Section 3.2. The lower plot shows the implied cases, computed as described in Section 3.2.

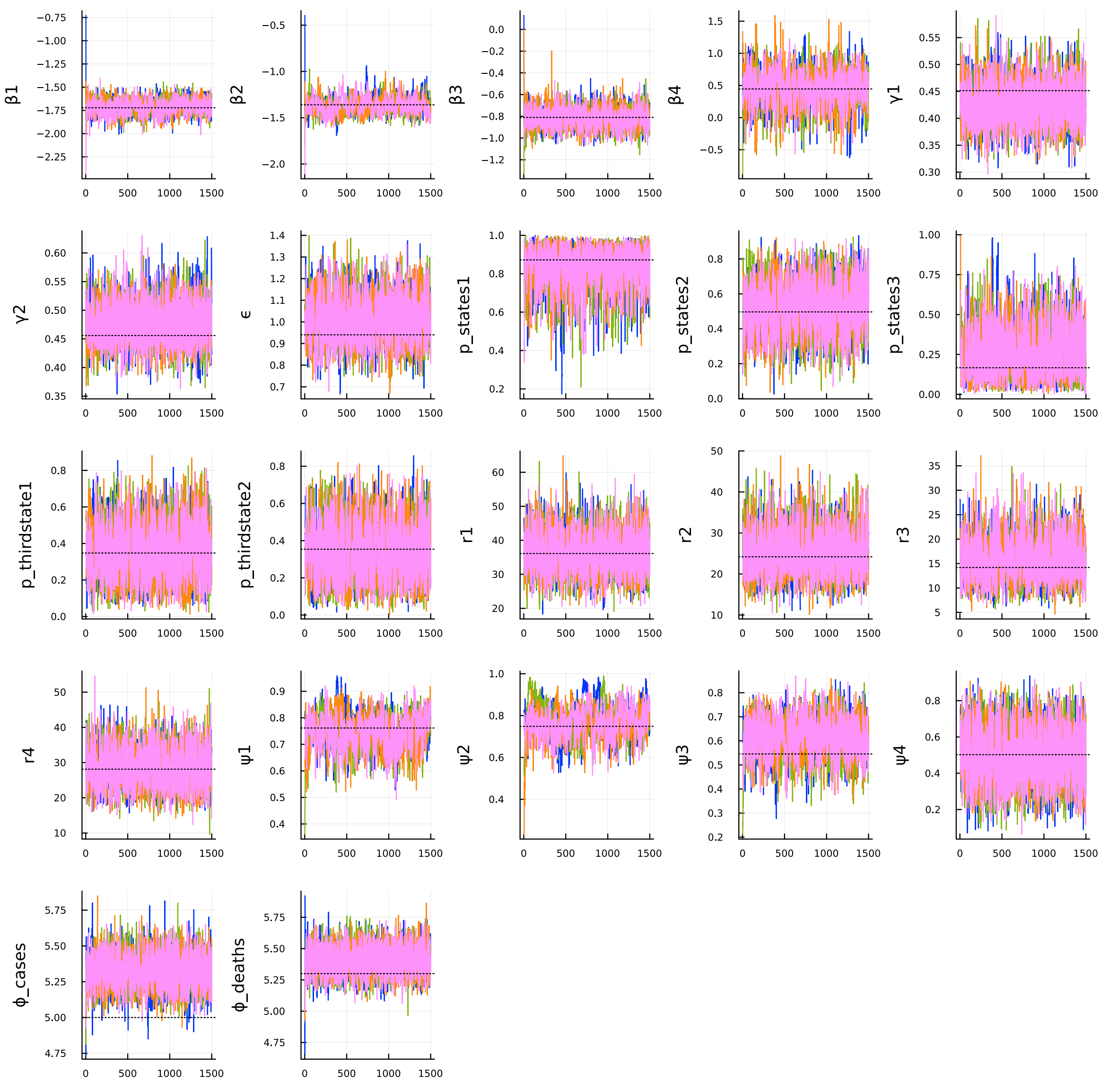


Figure A.14 Full PMCMC trace plots of a HSMM-EM with 4 states and Negative Binomial distribution for simulated data in Section 3.4. Parameter used to generate sample data are shown as dashed horizontal lines.

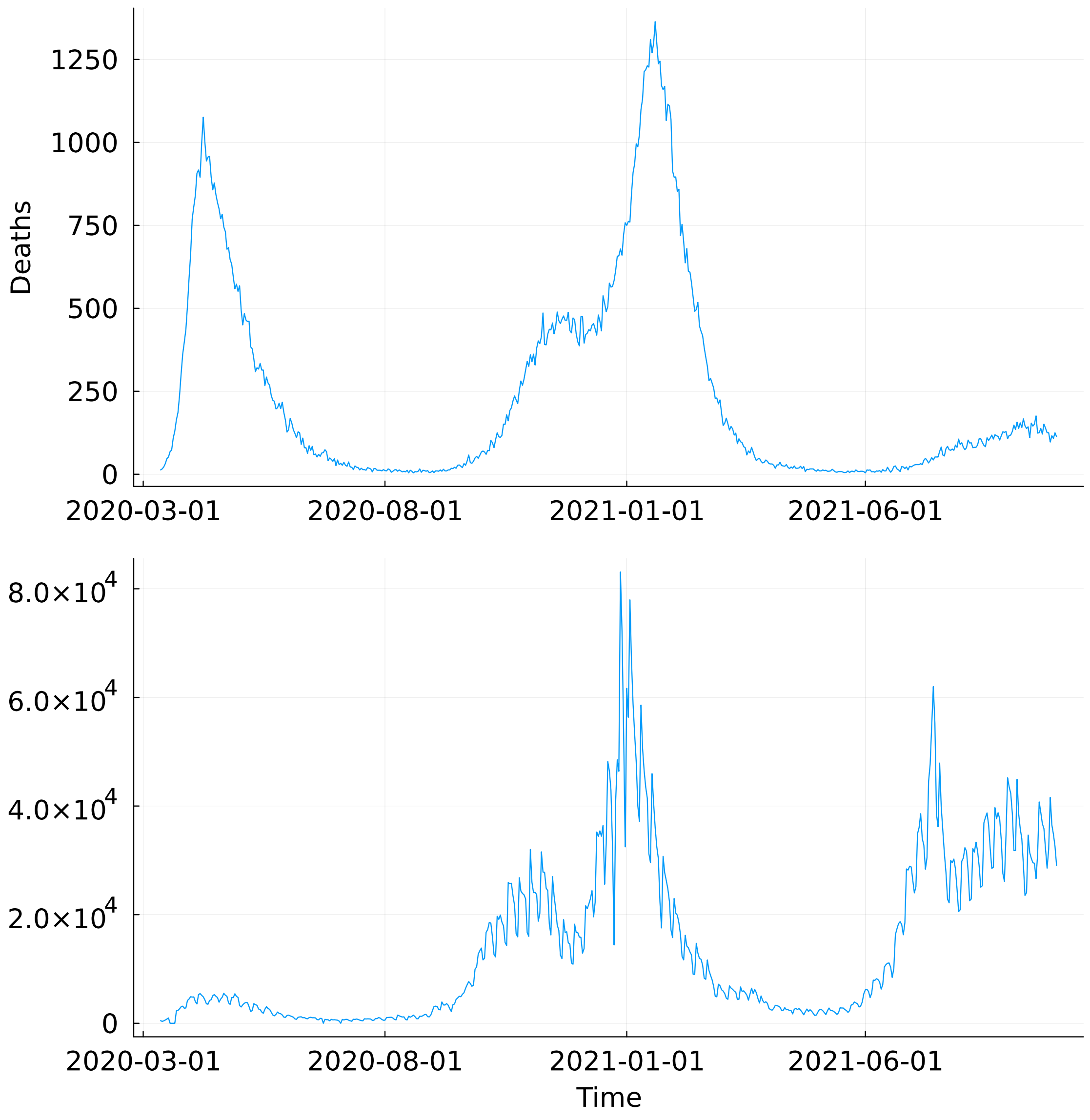


Figure A.15 Real data used in Section 3.5. The upper plot shows the reported deaths in the UK, the lower plot the reported cases. A more detailed explanation for the data processing is provided in Section 3.5.1.

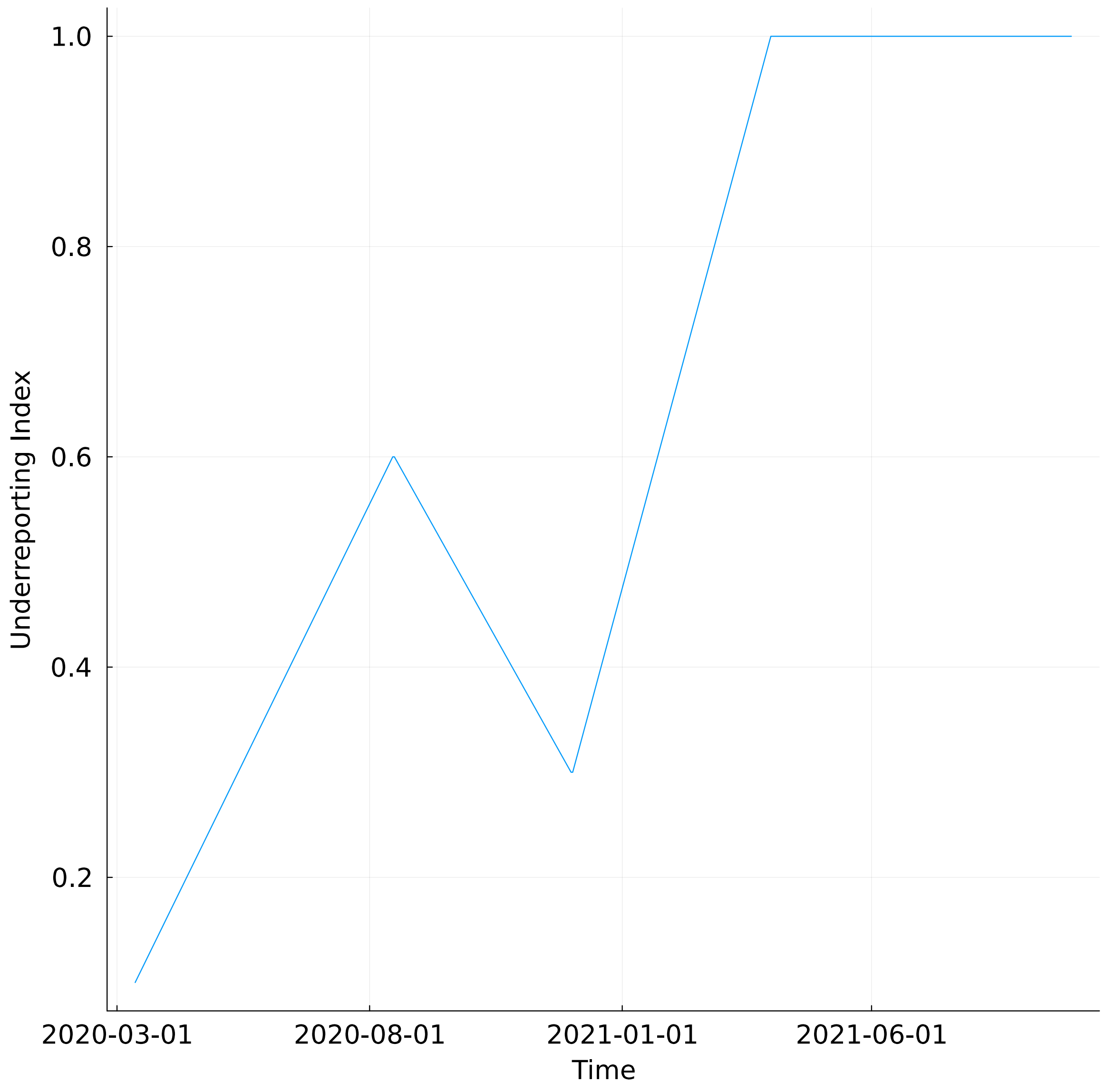


Figure A.16 The Underreporting index that is used in the observation model for the reported infections, see Section 3.2. Individual values are determined based on the work of Chatzilena et al. (2022).

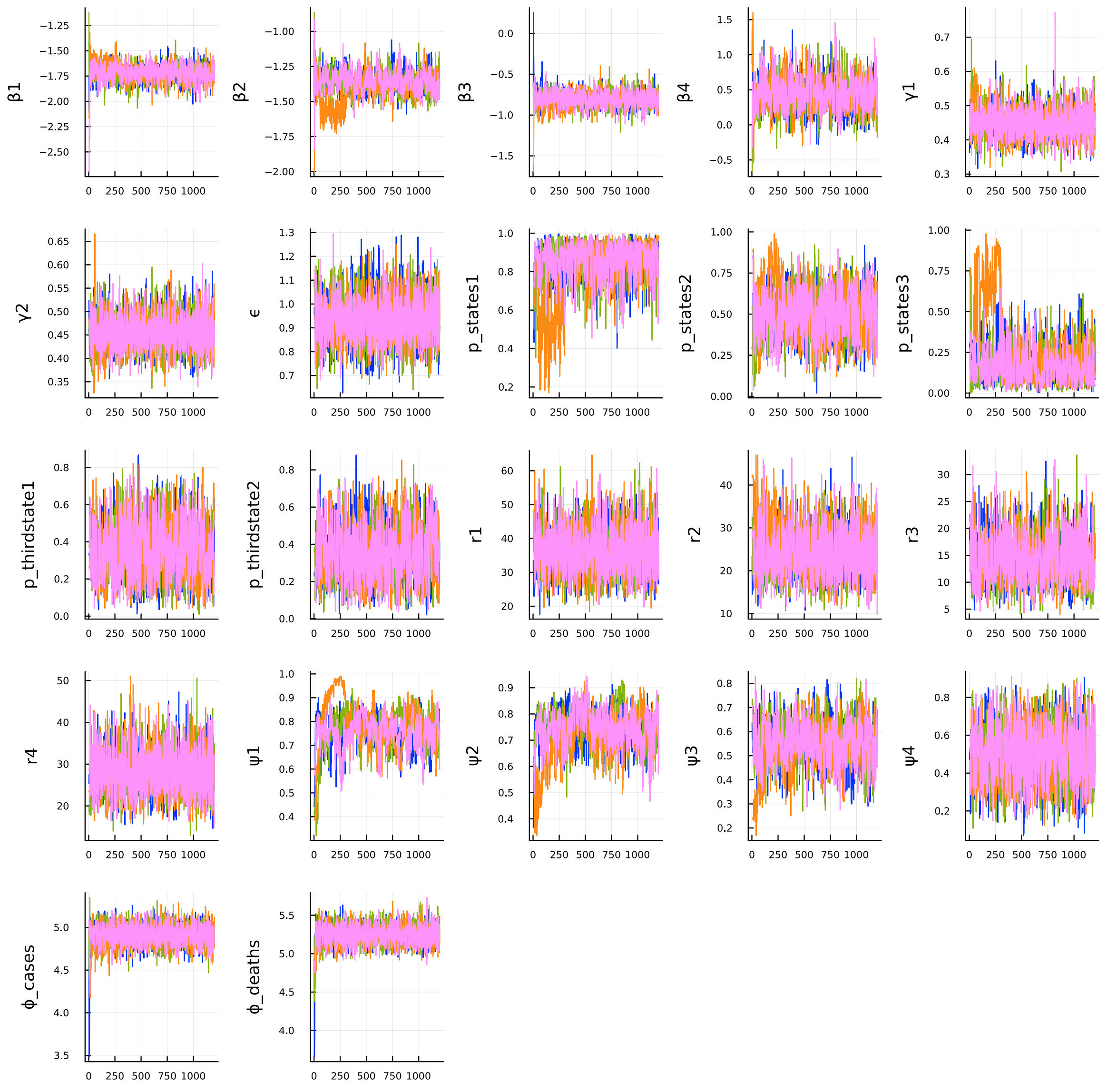


Figure A.17 Full PMCMC trace plots of a HSMM-EM with 4 states and Negative Binomial distribution estimated on real data in Section 3.5.

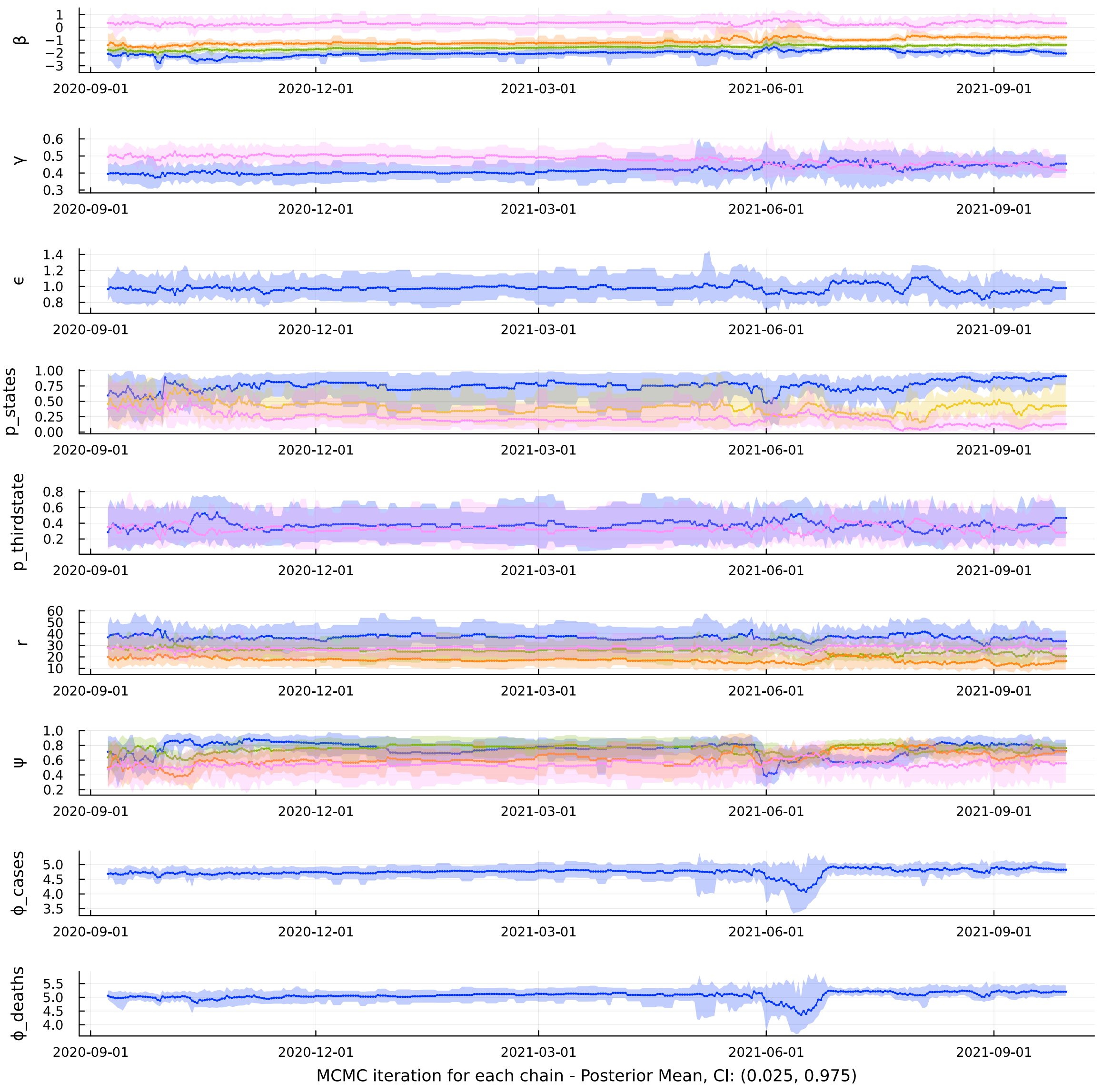


Figure A.18 SMC posterior estimates of the continuous model parameter of 48 chains for the best performing HSMM-EM defined in section 3.2 and used on real data in Section 3.5.

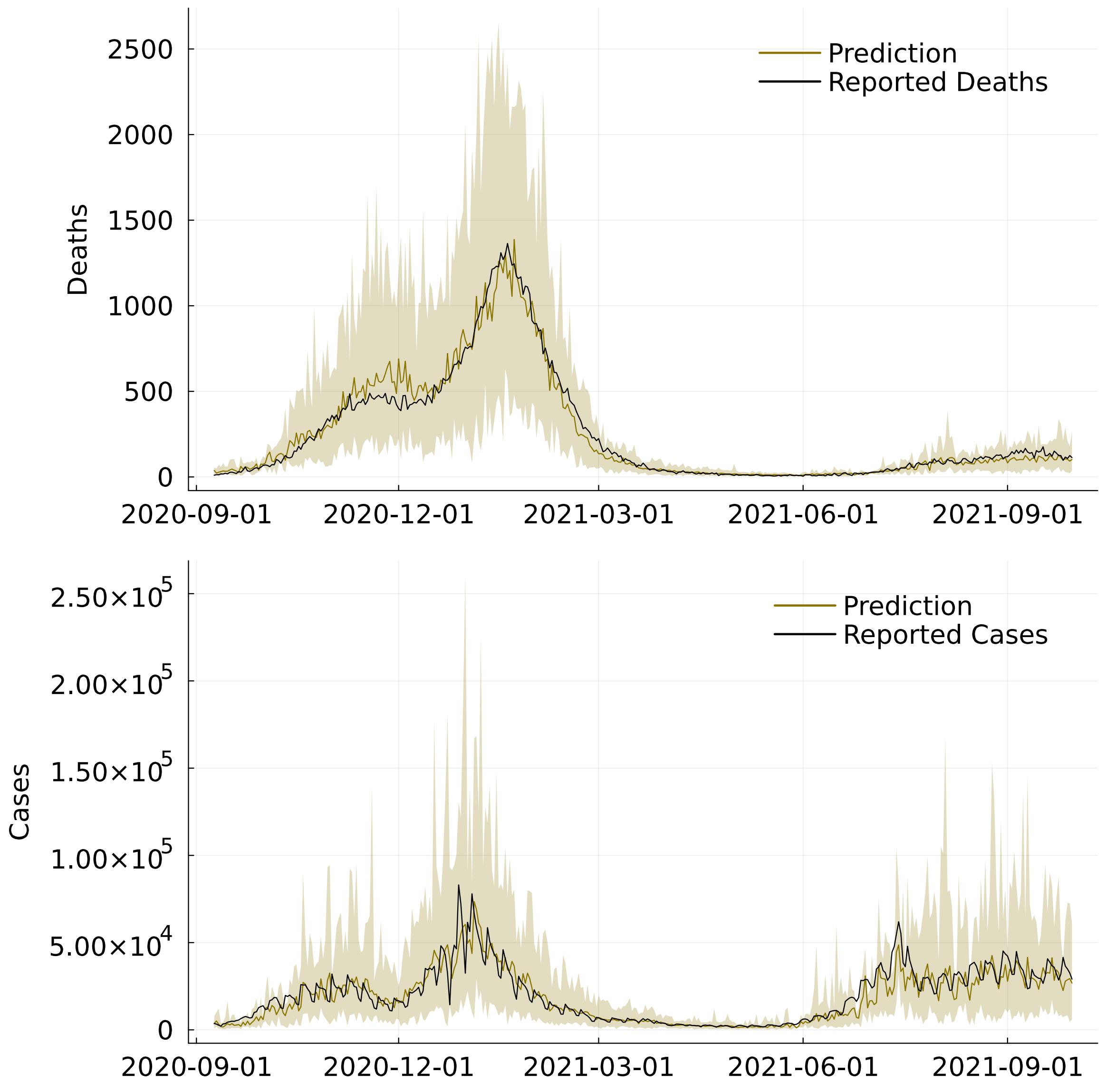
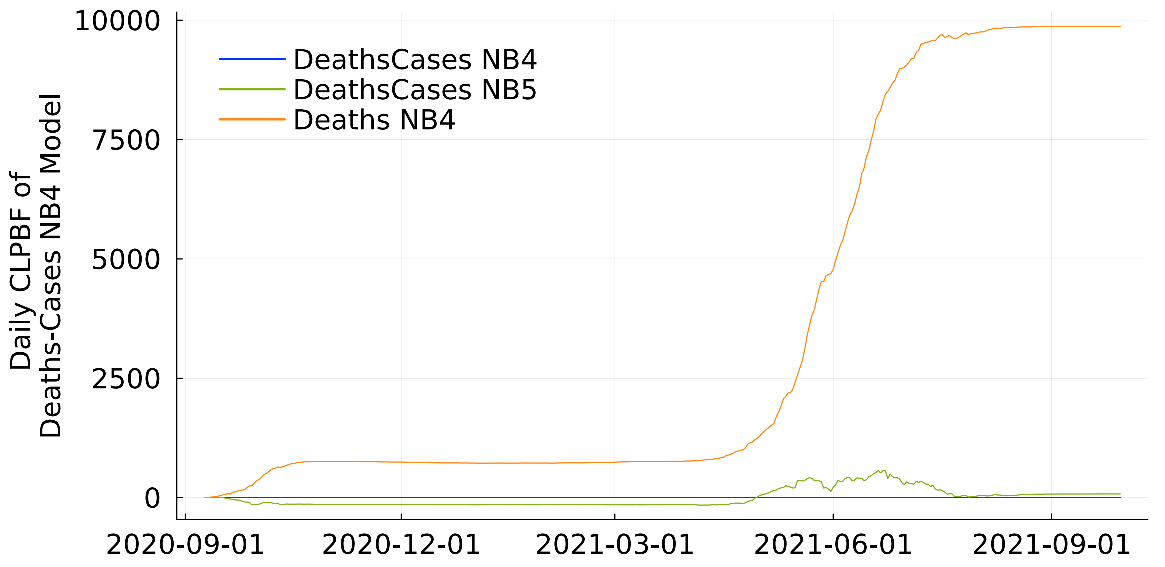


Figure A.19 SMC prediction plot for the HSMM-EM with 4 states and Negative Binomial distribution estimated on real data in Section 3.5. A 95 % Credible Interval (CI) is included at each iteration. The upper plot shows the reported fatalities against the predicted ones, the lower the corresponding cases.



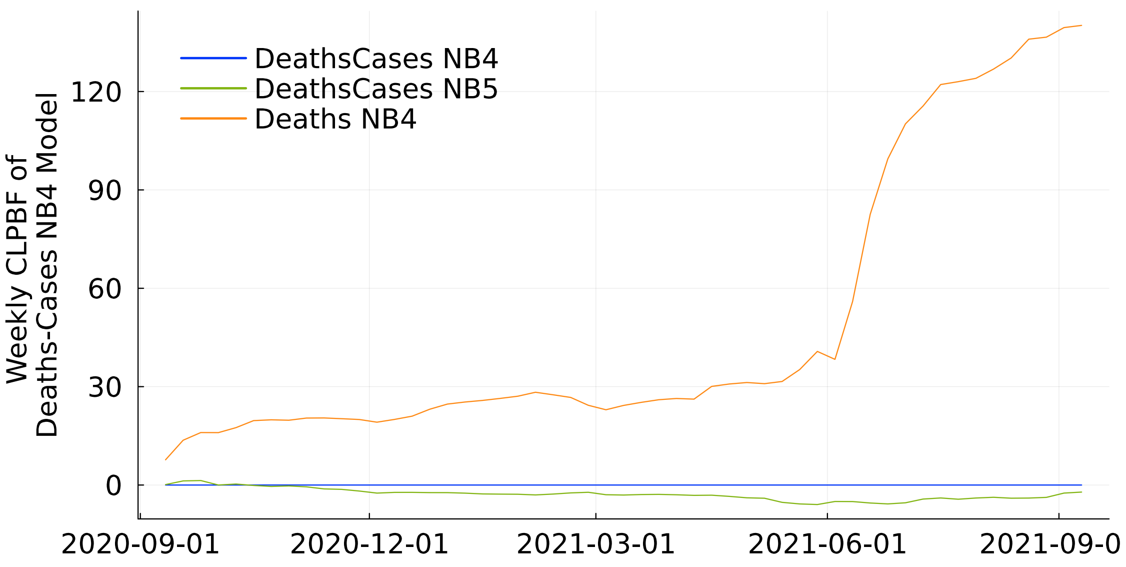


Figure A.20 SMC prediction comparison: this plot shows the Cumulative Log Predictive Bayes Factor of the HSMM-EM with 4 states and Negative Binomial distribution against all other models for daily predictions (sub-plot 1) as well as for weekly predictions (sub-plot 2). NB4 stands for models using a Negative Binomial Duration distribution with 4 different regimes.

Chapter 4 Plots

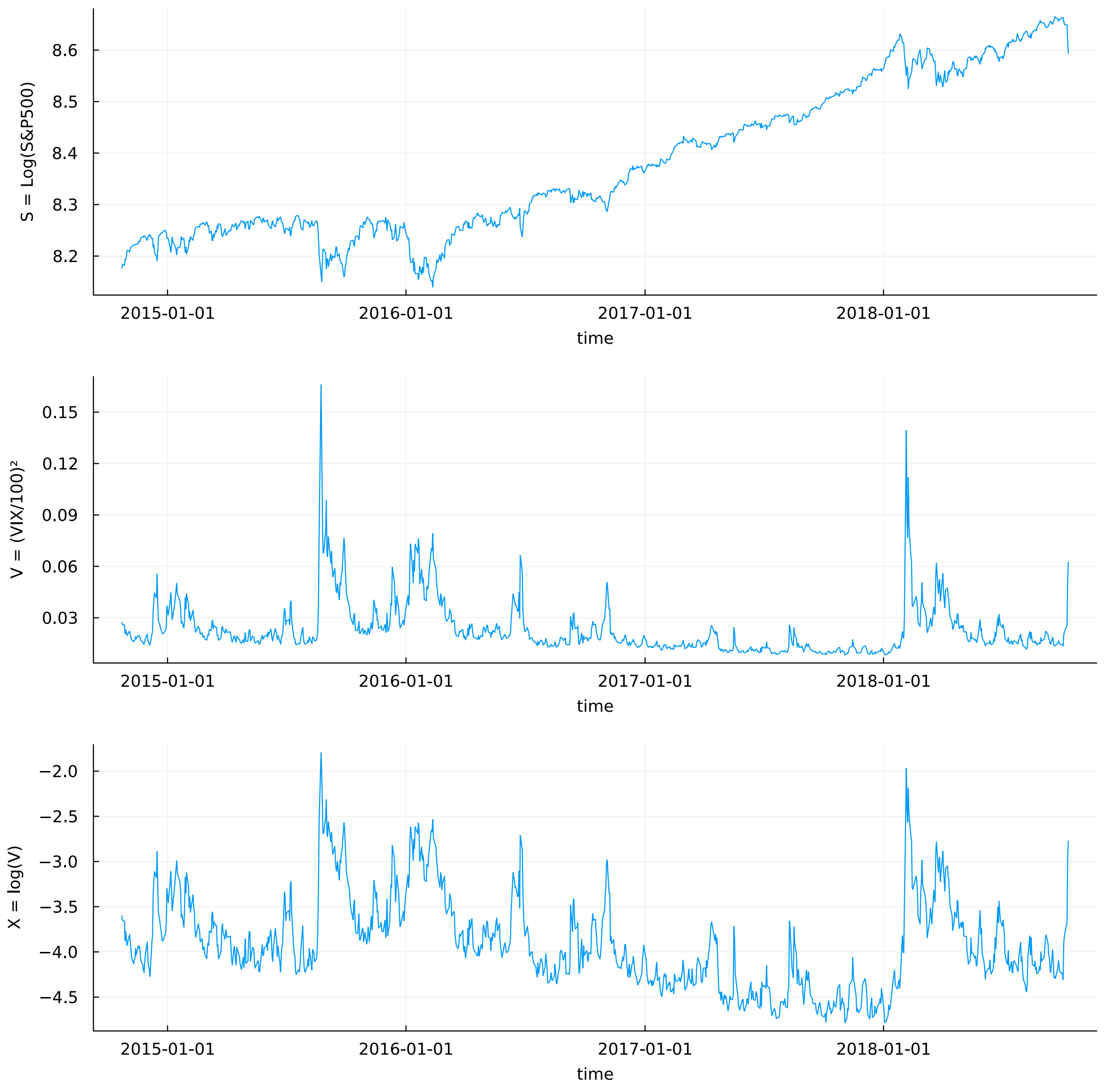


Figure A.21 Real data used in Section 4.5. The upper plot shows the S&P 500 Index in log space, the lower plot the transformed data based on VIX Index. A more detailed explanation for the data processing is provided in Section 4.5.1.

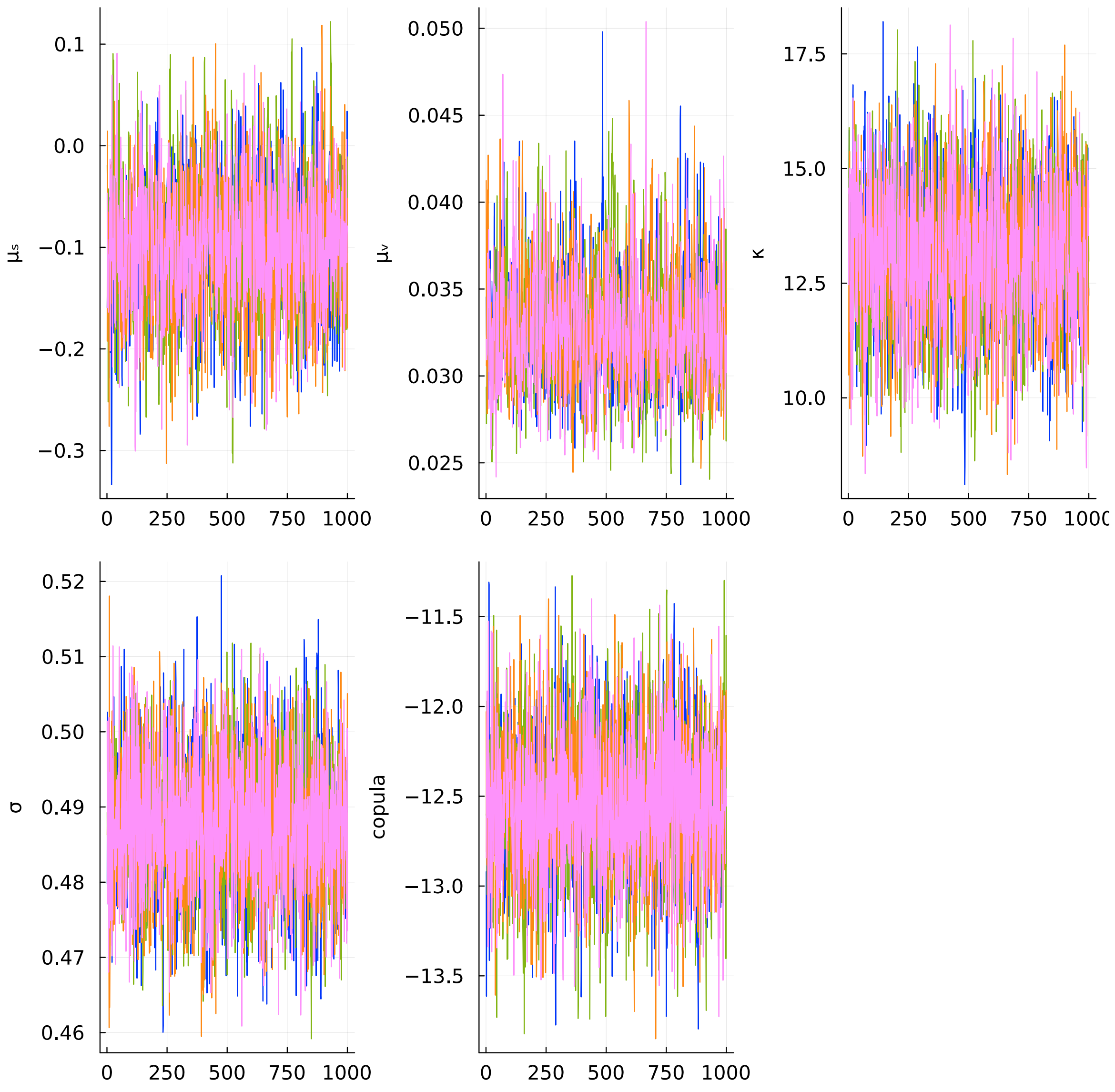


Figure A.22 Full MCMC trace plots of the best SVC according to the model comparison criteria described in Section 4.3.3 for real data in Section 4.5.

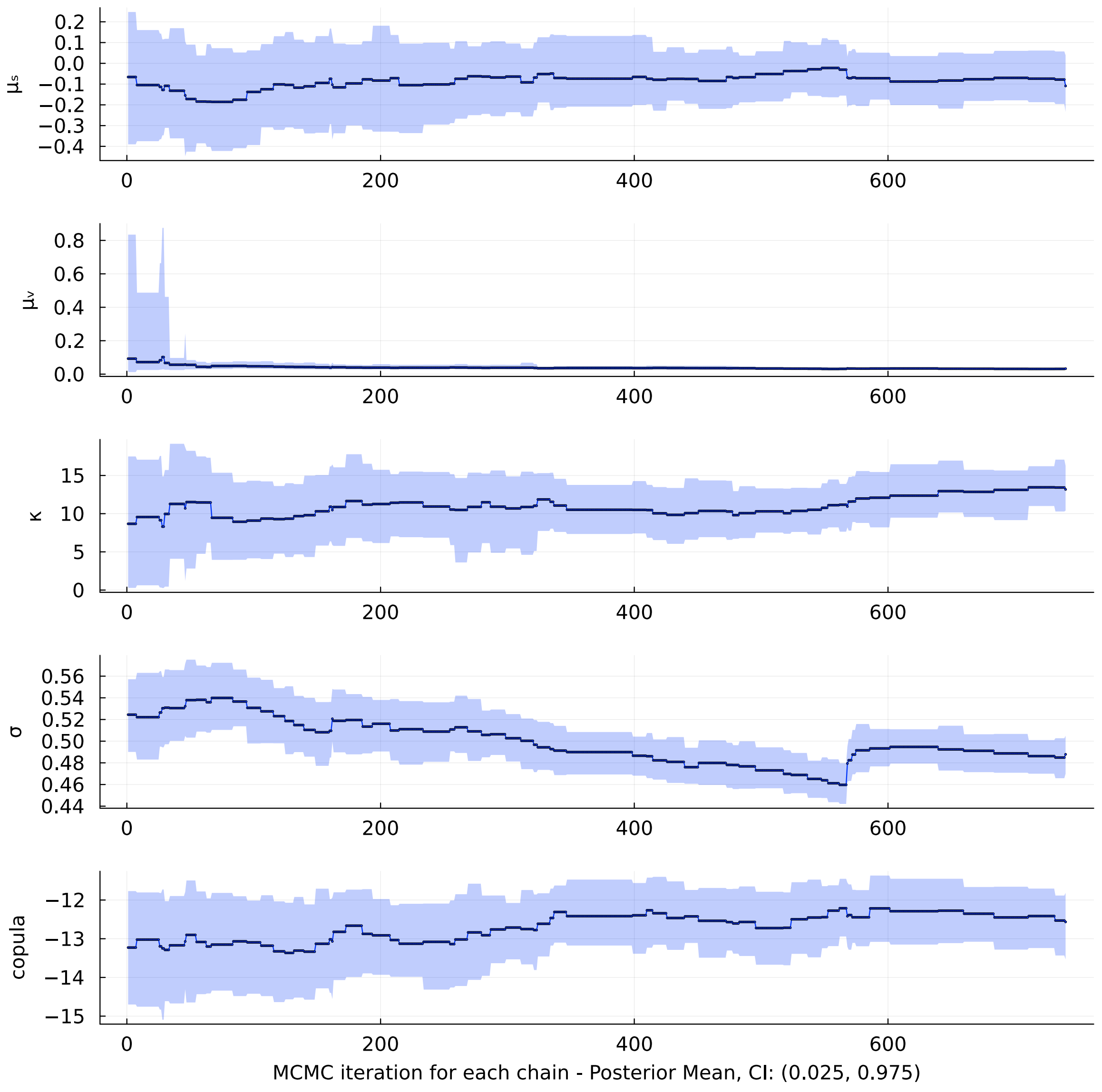
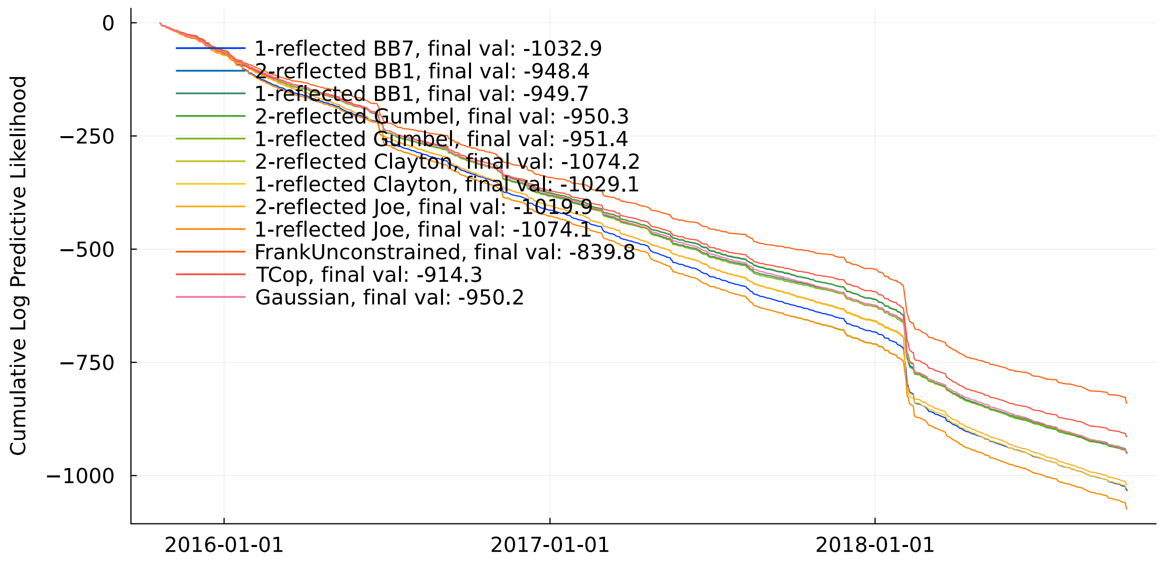


Figure A.23 Full SMC trace plots of the best SVC according to the model comparison criteria described in Section 4.3.3 for real data in Section 4.5.



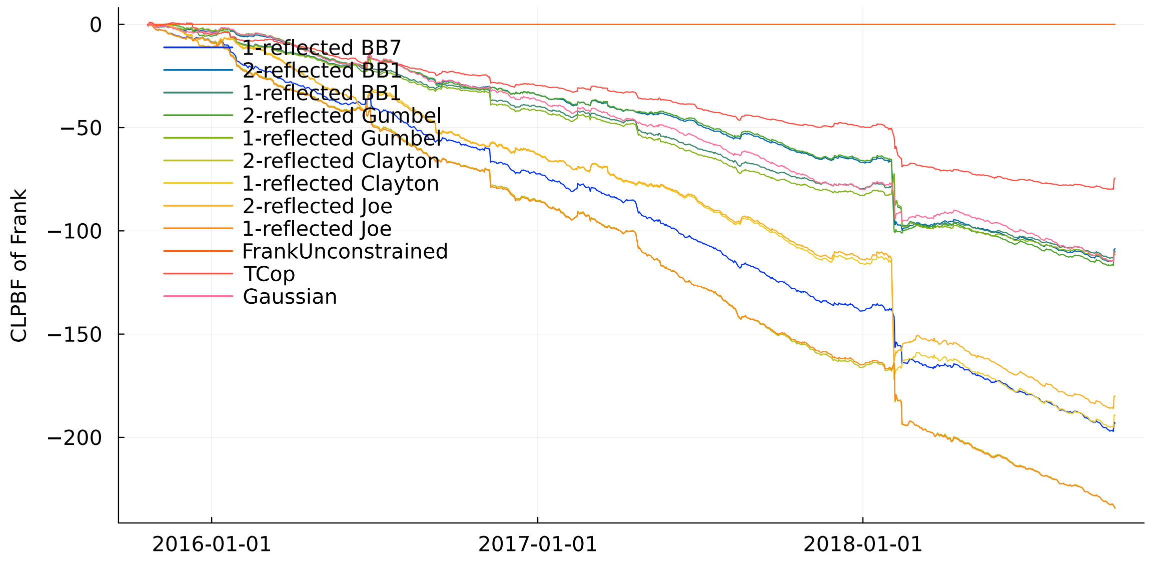


Figure A.24 This plot shows the Cumulative Log Predictive Bayes Factor as defined in Section 4.3.3 of the Frank Copula against the rest. All computations are based on daily data.

Appendix B

Pseudo Algorithms

Particle Filter

Algorithm 1: Standard particle filter

```
input            : data $e_{1:T}$, model parameter $\theta$
output           : log-likelihood estimate $\hat{\ell}(\theta) = \log \hat{p}_\theta(e_{1:T})$ and sample
                   $s_{1:T} \sim \hat{p}(s_{1:T} \mid e_{1:T})$
tuning parameter : proposal distribution $q$, number of particles $N$
function         : particle filter $pf(e_{1:T}, \theta)$

   // Initialization:
1  for $n \leftarrow 1$ to $N$ do
2      Initiate particle $s_1^n \sim \pi_\theta(s_1)$.
3      Compute $\alpha_1^n(s_1^n, e_1) = \frac{p_\theta(e_1|s_1^n)\ p_\theta(s_1^n)}{q(s_1^n|e_1)}$.
4  Normalize weights $\tilde{\alpha}_1^i \propto \alpha_1^i(s_1^n, e_1)$ for $i = 1 : N$, s.t. $\sum_{i=1}^N \tilde{\alpha}_1^i = 1$.
5  Compute log-likelihood increment $\hat{\ell}(\theta) = \log \frac{1}{N} \sum_{i=1}^N \alpha_1^i(s_1^i, e_1)$
   // Forward propagation:
6  for $t \leftarrow 2$ to $T$ do
7      if Resampling required then
8          Sample ancestor $a_t^n$ for particle trajectory $s_{1:t-1}^n$ for $n = 1$ to $N$
             according to normalized weights $\tilde{\alpha}_{t-1}$.
9      else
10         Set $a_t^n = n$ for $n = 1$ to $N$.
11     for $n \leftarrow 1$ to $N$ do
12         Sample $s_t^n \sim q(s_t^n \mid s_{1:t-1}^{a_t^n}, e_{1:t})$.
13         Set $s_{1:t}^n := (s_{1:t-1}^{a_t^n}, s_t^n)$.
14         Calculate incremental weight:
               $\alpha_t(s_{1:t}^n, e_{1:t}) = \frac{p_\theta(e_t \mid s_{1:t}^n, e_{1:t-1})\ p_\theta(s_t^n \mid s_{1:t-1}^n, e_{1:t-1})}{q(s_t^n \mid s_{1:t-1}^n, e_{1:t})}$
15     Normalize weights $\tilde{\alpha}_t^i \propto \alpha_t^i(s_{1:t}^i, e_{1:t})$ for $i = 1$ to $N$, s.t. $\sum_{i=1}^N \tilde{\alpha}_t^i = 1$.
16     Add incremental weights to log-likelihood:
         $\hat{\ell}(\theta) = \hat{\ell}(\theta) + \log \frac{1}{N} \sum_{i=1}^N \alpha_t^i(s_{1:t}^i, e_{1:t})$.
   // Return log-likelihood estimate and particle trajectories:
17 Draw k with $P(k = i) \propto \tilde{\alpha}_T^i$.
18 return $\hat{\ell}(\theta)$ and $s_{1:T}^k$.
```

Algorithm 2: Conditional particle filter with ancestor sampling

```
input              : Reference $s'_{1:T}$, data $e_{1:T}$, model parameter $\theta$
output             : Log-likelihood estimate $\hat{\ell}(\theta) = \log \hat{p}_\theta(e_{1:T})$ and sample
                     $s_{1:T} \sim \hat{p}(s_{1:T} \mid s'_{t:T}, e_{1:T})$
tuning parameter   : proposal distribution $q$, number of particles $N$
function           : particle filter $cpf(s'_{1:T}, e_{1:T}, \theta)$
// Initialization:
1  for $n \leftarrow 1$ to $N-1$ do
2      Initiate particle $s_1^n \sim \pi_\theta(s_1)$.
3      Compute $\alpha_1^n(s_1^n, e_1) = \frac{p_\theta(e_1|s_1^n)\ p_\theta(s_1^n)}{q(s_1^n|e_1)}$.
4  Set $s_1^N := s'_1$ and compute $\alpha_1^N(s_1^N, e_1) \propto \frac{p_\theta(e_1|s_1^N)\ p_\theta(s_1^N)}{q(s_1^N|e_1)}$.
5  Normalize weights $\tilde{\alpha}_1^i \propto \alpha_1^i(s_1^n, e_1)$ *for* $i = 1 : N$, *s.t.* $\sum_{i=1}^N \tilde{\alpha}_1^i = 1$.
6  Compute log-likelihood increment $\hat{\ell}(\theta) = \log \frac{1}{N} \sum_{i=1}^N \alpha_1^i(s_1^i, e_1)$
// Forward propagation:
7  for $t \leftarrow 2$ to $T$ do
8      if *Resampling required* then
9          Sample ancestor $a_t^n$ for particle trajectory $s_{1:t-1}^n$ for $n = 1$ to $N-1$
             according to normalized weights $\tilde{\alpha}_{t-1}$.
10         Set $a_t^N = k$, with
             $P(k = i) \sim \alpha_{t-1}^i(s_{1:t-1}^i, e_{1:t-1})\ p_\theta(s'_{t:T}, e_{t:T} \mid s_{1:t-1}^i, e_{1:t-1})$.
11     else
12         Set $a_t^n = n$ for $n = 1$ to $N$.
13     for $n \leftarrow 1$ to $N$ do
14         Sample $s_t^n \sim q(s_t^n \mid s_{1:t-1}^{a_t^n}, e_{1:t})$.
15         Set $s_{1:t}^n := (s_{1:t-1}^{a_t^n}, s_t^n)$.
16         Calculate incremental weight:
               $$\alpha_t(s_{1:t}^n, e_{1:t}) = \frac{p_\theta(e_t \mid s_{1:t}^n, e_{1:t-1})\ p_\theta(s_t^n \mid s_{1:t-1}^n, e_{1:t-1})}{q(s_t^n \mid s_{1:t-1}^n, e_{1:t})}$$
17     Normalize weights $\tilde{\alpha}_t^i \propto \alpha_t^i(s_{1:t}^i, e_{1:t})$ for $i = 1$ to $N$, s.t. $\sum_{i=1}^N \tilde{\alpha}_t^i = 1$.
18     Add incremental weights to log-likelihood:
         $\hat{\ell}(\theta) = \hat{\ell}(\theta) + \log \frac{1}{N} \sum_{i=1}^N \alpha_t^i(s_{1:t}^i, e_{1:t})$.
// Return log-likelihood estimate and particle trajectories:
19 Draw k with $P(k = i) \propto \tilde{\alpha}_T^i$.
20 return $\hat{\ell}(\theta)$ and $s_{1:T}^k$.
```

MCMC

Algorithm 3: Metropolis Hastings (MH) Kernel

```
input            : data $e_{1:T}$, current model parameter $\theta$
output           : model parameter $\theta \sim p(\theta \mid e_{1:T})$
tuning parameter : proposal distribution $f$
function         : MCMC Kernel $K_{mh}(e_{1:T}, \theta)$
1 Propose $\theta^\star \sim f(\theta^\star \mid \theta)$.
2 Set $\theta := \theta^\star$ with acceptance probability $min(1, a(\theta^\star, \theta))$, where
      $$a(\theta^\star, \theta) = \frac{p(\theta \mid e_{1:T})\ f(\theta \mid \theta^\star)}{p(\theta \mid e_{1:T})\ f(\theta^\star \mid \theta)} = \frac{p_{\theta^\star}(e_{1:T})\ p(\theta^\star)\ f(\theta \mid \theta^\star)}{p_\theta(e_{1:T})\ p(\theta)\ f(\theta^\star \mid \theta)}.$$
3 return $\theta$.
```

Algorithm 4: Hamiltonian Monte Carlo (HMC) Kernel

```
input            : data $e_{1:T}$, current model parameter $\theta$
output           : model parameter $\theta \sim p(\theta \mid e_{1:T})$
tuning parameter : Mass matrix $M$, stepsize $\epsilon$, number of leapfrog steps $L$.
function         : MCMC Kernel $K_{HMC}(e_{1:T}, \theta)$
1  Sample $\rho \sim MvNormal(0, M)$ and set $(\theta^\star, \rho^\star) := (\theta, \rho)$.
2  for $i \leftarrow 1$ to $L$ do
3      $\theta^\star, \rho^\star = Leapfrog(\theta^\star, \rho^\star, M, \epsilon)$
4  Set $\theta := \theta^\star$ with acceptance probability $min(1, a(\theta^\star, \theta))$, where
       $a(\theta^\star, \theta) = exp(H(\rho, \theta) - H(\rho^\star, \theta^\star))$
5  return $\theta$.
6  Function Leapfrog($\theta_t, \rho_t, M, \epsilon$):
7      $\rho_{t+\frac{\epsilon}{2}} \leftarrow \rho_t + \frac{\epsilon}{2} \frac{\partial log\ p(\theta|e_{1:T})}{\partial \theta}(\theta_t)$
8      $\theta_{t+\epsilon} \leftarrow \theta_t + \epsilon M^{-1} \rho_{t+\frac{\epsilon}{2}}$
9      $\rho_{t+\epsilon} \leftarrow \rho_{t+\frac{\epsilon}{2}} + \frac{\epsilon}{2} \frac{\partial log\ p(\theta|e_{1:T})}{\partial \theta}(\theta_{t+\epsilon})$
10     return $\theta_{t+\epsilon}, \rho_{t+\epsilon}$.
```

Particle MCMC

Algorithm 5: Particle Metropolis Hastings Kernel

```
input            : current state trajectory $s_{1:T}$, data $e_{1:T}$, current model
                   parameter $\theta$
output           : model parameter $\theta$ and state trajectory $s_{1:T}$,
                   $(\theta, s_{1:T}) \sim p(\theta, s_{1:T} \mid e_{1:T})$
tuning parameter : MCMC kernel $K_{mcmc}$, particle filter $pf$, number of
                   iterations $N$
function         : PMCMC kernel $K_{pmh}(e_{1:T}, s_{1:T}, \theta)$
1 Propose $\theta^\star \sim K_{mcmc}(e_{1:T}, \theta)$
2 Run particle filter $pf$ to obtain $\hat{p}_{\theta^\star}(e_{1:T})$ and $s^\star_{1:T} \sim \hat{p}_{\theta^\star}(s^\star_{1:T} \mid e_{1:T})$.
3 Set $(\theta, s_{1:T}) := (\theta^\star, s^\star_{1:T})$ with acceptance probability $min(1, a(\theta^\star, \theta))$, where
      $$a(\theta^\star, \theta) = \frac{\hat{p}_{\theta^\star}(e_{1:T})\ p(\theta^\star)\ f_{mcmc}(\theta \mid \theta^\star)}{\hat{p}_\theta(e_{1:T})\ p(\theta)\ f_{mcmc}(\theta^\star \mid \theta)}$$
4 return $(\theta, s_{1:T})$
```

Algorithm 6: Particle Gibbs Kernel

```
input            : reference trajectory $s_{1:T}$, data $e_{1:T}$, current model
                   parameter $\theta$
output           : model parameter $\theta$ and state trajectory $s_{1:T}$,
                   $(\theta, s_{1:T}) \sim p(\theta, s_{1:T} \mid e_{1:T})$
tuning parameter : conditional particle filter $cpf$
function         : PMCMC kernel $K_{pgibbs}(e_{1:T}, s_{1:T}, \theta)$
1 Propose $\theta^\star \sim p_\theta(\theta^\star \mid s_{1:T}, e_{1:T})$
2 Run a conditional particle filter $cpf$ to obtain $s^\star_{1:T} \sim \hat{p}_{\theta^\star}(s^\star_{1:T} \mid s_{1:T}, e_{1:T})$.
3 Set $(\theta, s_{1:T}) := (\theta^\star, s^\star_{1:T})$
4 return $(\theta, s_{1:T})$
```

SMC

Algorithm 7: Iterative Batch Importance Sampling Algorithm

```
input            : Data $e_{1:T}$
output           : model parameter $\theta$, $(\theta^i)_{i=1:N} \sim p(\theta^i \mid e_{1:t})$ for $t = 1$ to $T$
tuning parameter : number of particles $N$, N MCMC kernel $(K_{mcmc,i})_{i=1:N}$
function         : IBIS sampler $ibis(e_{1:T})$

   // Initialization:
1  for $n \leftarrow 1$ to $N$ do
2      Initiate parameter vector $\theta_n \sim p(\theta_n)$.
   // Transition:
3  for $t \leftarrow t_0 + 1$ to $T$ do
4      if Resampled at t-1 then
5          for $n \leftarrow 1$ to $N$ do
6              Compute $p_{\theta^n}(e_{1:t} \mid e_{1:t-1})$.
7      else
8          for $n \leftarrow 1$ to $N$ do
9              Compute $p_{\theta^n}(e_t \mid e_{1:t-1})$ and set
                $p_{\theta^n}(e_{1:t} \mid e_{1:t-1}) = p_{\theta^n}(e_t \mid e_{1:t-1}) p_{\theta^n}(e_{1:t-1} \mid e_{1:t-2})$
10     Normalize incremental weights $\tilde{\alpha}_t^n \propto p_{\theta^n}(e_{1:t} \mid e_{1:t-1})$ for $n = 1$ to $N$, s.t.
        $\sum_{n=1}^N \tilde{\alpha}_t^n = 1$.
11     Compute an estimate for the incremental marginal likelihood:
            $\hat{L}_t = \hat{p}(e_t \mid e_{1:t-1}) = \sum_{n=1}^{N} \tilde{\alpha}_t^n p_{\theta^n}(e_{1:t} \mid e_{1:t-1}).$
12     if Resampling required then
13         for $n \leftarrow 1$ to $N$ do
14             Draw k with $P(k = i) \propto \tilde{\alpha}_t^i$.
15             Propose $\theta^\star \sim K_{mcmc,k}(e_{1:t}, \theta^k)$.
16             Set $\theta^n := \theta^\star$
17 return $\theta^i_{i=1:N}$ for $t = 1$ to $T$.
```

Algorithm 8: Sequential Monte Carlo Squared algorithm

```
input            : Data $e_{1:T}$
output           : model parameter $\theta$ and state trajectory $s_{1:t}$,
                   $(\theta^i, s^i_{1:t})_{i=1:N} \sim p(\theta^i, s^i_{1:t} \mid e_{1:t})$ for $t = 1$ to $T$
tuning parameter : number of particles $N$, N particle filter $(pf_i)_{i=1:N}$, N
                   PMCMC kernel $(K_{pmcmc,i})_{i=1:N}$
function         : SMC2 sampler $smc^2(e_{1:T})$

   // Initialization:
1  for $n \leftarrow 1$ to $N$ do
2      Initiate parameter vector $\theta_n \sim p(\theta_n)$.
3      Run particle filter $pf_n$ to obtain $s^n_{1:t_0} \sim \hat{p}_{\theta_n}(s^n_{1:t_0} \mid e_{1:t_0})$.
   // Transition:
4  for $t \leftarrow t_0 + 1$ to $T$ do
5      if Resampled at t-1 then
6          for $n \leftarrow 1$ to $N$ do
7              Run particle filter $pf_n$ to obtain $\hat{p}_{\theta^n}(e_{1:t} \mid e_{1:t-1})$ and
                $s^n_{1:t} \sim \hat{p}_{\theta^n}(s^n_{1:t} \mid e_{1:t})$.
8      else
9          for $n \leftarrow 1$ to $N$ do
10             Propagate particle filter $pf_n$ forward to obtain $\hat{p}_{\theta^n}(e_{1:t} \mid e_{1:t-1})$ and
                $s^n_t \sim \hat{p}_{\theta^n}(s^n_t \mid s^n_{1:t-1}, e_{1:t})$.
11             Set $s^n_{1:t} := (s^n_{1:t-1}, s^n_t)$.
12     Normalize incremental weights $\tilde{\alpha}^n_t \propto \hat{p}_{\theta^n}(e_{1:t} \mid e_{1:t-1})$ for $n = 1$ to $N$, s.t.
        $\sum_{n=1}^{N} \tilde{\alpha}^n_t = 1$.
13     Compute an estimate for the incremental marginal likelihood:

           $$\hat{L}_t = \hat{p}(e_t \mid e_{1:t-1}) = \sum_{n=1}^{N} \tilde{\alpha}^n_t \hat{p}_{\theta^n}(e_{1:t} \mid e_{1:t-1}). \qquad (B.1)$$

14     if Resampling required then
15         for $n \leftarrow 1$ to $N$ do
16             Draw k with $P(k = i) \propto \tilde{\alpha}^i_t$.
17             Propose $(\theta^\star, s^\star_{1:t}) \sim K_{pmcmc,k}(e_{1:t}, s^k_{1:t}, \theta^k)$.
18             Set $(\theta^n, s^n_{1:t}) := (\theta^\star, s^\star_{1:t})$
19 return $(\theta^i, s^i_{1:t})_{i=1:N}$ for $t = 1$ to $T$.
```

Appendix C

Derivations

Chapter 4 Derivations

Derivation of $|J_\Theta|$

We can write

$$|J_\Theta| = \left|\frac{dE}{dY}\right|,$$

$|\cdot|$ denotes the determinant, Y is a vector containing Y^s and Y^v and

$$\frac{dE}{dY} = (\partial E_i)/\partial Y_j)_{ij} \in \mathbb{R}^{2T\times 2T}.$$

The model in Section 4.2 can be re-written as

$$E_t = \begin{bmatrix} \epsilon_t \\ \zeta_t \end{bmatrix} = \begin{bmatrix} \frac{Y_t^s - Y_{t-1}^s - \left(\mu_S - \exp(X_t)/2\right)\delta}{\sqrt{\delta}\,\exp(X_t/2)}, \\ \frac{X_t - X_{t-1} - \frac{\kappa\left(\mu_V - \exp(X_{t-1})\right) - \frac{1}{2}\sigma^2}{\exp(X_{t-1})}\delta}{\sigma\sqrt{\delta}\exp(-X_{t-1}/2)} \end{bmatrix}$$

where $t = 1, 2, \ldots T$, and $X_t = \log Y_t^v$. Notice that dE/dY is a bi-diagonal matrix and therefore its determinant is given by the product of its diagonal entries that are

$$\frac{\partial E_t}{\partial Y_t} = \begin{cases} \frac{1}{\sqrt{\delta}\,\exp(X_t/2)} & \text{if } \ t = 1, \ldots, T, \\ \frac{\exp(-X_t)}{\sigma\sqrt{\delta}\exp(-X_{t-1}/2)} & \text{if } \ t = T+1, \ldots, 2T. \end{cases}$$

Hence we can write

$$\left|\frac{dE}{dY}\right| = \prod_{t=1}^{T} \left\{\sqrt{\delta}\, \exp(X_t/2)\right\}^{-1} \left\{\sigma\, \sqrt{\delta}\, \exp(X_{t-1}/2)\right\}^{-1} \propto \sigma^{-T}.$$