THE UNIVERSITY *of* EDINBURGH

Efficient Monte Carlo methods for Bayesian state-space model inference

Mary Llewellyn

Doctor of Philosophy
University of Edinburgh
January 2024

Declaration

I declare that I wrote this thesis myself under the guidance of my supervisors. Chapters 3 and 4 are based on papers for which I am first of multiple co-authors. The associated papers and their contributions to the thesis are as follows.

> Llewellyn, M., King, R., Elvira, V., Ross, G. (2023). A point mass proposal method for Bayesian state-space model fitting. *Statistics and Computing*, 33(111).
> Available at https://doi.org/10.1007/s11222-023-10268-6.

Chapter 3 is an extended version of this article.

> Llewellyn, M., Ross, G., Ryan-Saha, J. (2023). COVID-era forecasting: Google trends and window and model averaging. *Annals of Tourism Research*, 103:103660.
> Available at https://doi.org/10.1016/j.annals.2023.103660.

The tourism demand case study in Section 4.4.2 of Chapter 4 is an extended version of the model proposed in this publication.

> Llewellyn, M., King, R., Elvira, V., Ross, G. Grid particle Gibbs with ancestor sampling for efficient state-space model inference.

This article has been submitted for publication. Chapter 4 is an extended version of this article. The work in this thesis has not been submitted for any other degree or professional qualification.

Mary Llewellyn

Acknowledgements

First and foremost, I would like to thank my supervisors: Professor Ruth King, Professor Víctor Elvira and Dr Gordon Ross. Ruth, your calibre of expertise, unwavering support and sense of humour have been instrumental to my PhD. I have learned so much from you and thank you for always encouraging me. Thank you to Víctor for the long and formative discussions throughout my PhD. Your enthusiasm for discussing ideas and pushing the boundaries of my research has been invaluable. And thank you Gordon for always pushing the way I think and write about problems, and for letting me loose on real data. Your advice has fundamentally shaped this thesis. I feel incredibly lucky to have learned from three such accomplished academics, and for the constructive and friendly academic atmosphere they have provided.

I also feel incredibly lucky for my big family and the role they have played in my academic journey. Thank you to my mum, Elaine. Your strength and work ethic are a constant inspiration for me, and thank you for always believing that I can achieve anything. You have always tried to make my life easier and have sacrificed a lot in doing so. Thank you to my stepdad, Mark, who has supported and encouraged me since I was small, always pushing me to question ideas. You have also been incredibly pragmatic throughout my academic journey and your guidance has been invaluable. I would like to acknowledge my dad, Nigel. Despite the challenges that have marked his life, his story has been a part of my own. While your path has been difficult, you have taught me strength and empathy. Thank you to my brother and sister, James and Molly, for your wisdom, constant belief in me and relentless sarcasm, and to Kate and Lucy, my stepsisters, who have been great role models. Finally, thank you to my partner James, and to my dog, Frankie, who make me feel lighter and are anchors of emotional support (and walks).

The research in this thesis was funded by the Engineering & Physical Sciences Research Council (EPSRC).

Lay summary

It is often useful to understand how aspects of our lives change over time. Weather updates may help us plan travel; reviewing exercise and sleep patterns over time may help us improve our general health; and tracking spending may help us budget effectively. On a greater scale, monitoring changes in the prevalence of a disease may improve interventions; following emerging market trends may improve business strategies; and tracking variations in income or inflationary pressure may improve economic policy. To understand changing situations and make informed decisions, measuring relevant quantities over time can be useful, often capturing temporal patterns objectively. Such measurements of given quantities over time are referred to as time series data.

While time series data can aid decision making, it can be challenging to understand useful patterns simply by inspecting the data. Statistical or mathematical models for time series data can improve the quality of inferences drawn by formalising our understanding of the underlying system and the data-generating processes. For example, models for the weather may combine our scientific understanding of temperature and precipitation with the observed historical patterns, and provide better forecasts than those based on intuition. When considering suitable models for data, the accuracy of a model (and inferences made) can often be improved by accounting for unobserved factors, such as error in the temperature measurements or seasonal fluctuations.

In this thesis, we focus on a class of models useful for describing the additional hidden processes that may be underlying the time series data, referred to as *state-space models*. When applying state-space models to time series data, a key step is estimating an appropriate model from the information in the data. However, the computational power required to estimate a model is often large and is increasing as both data and models become increasingly complex. This thesis proposes frameworks aimed at accelerating the computation process. We focus specifically on a class of computational techniques using random simulation, where existing approaches can be highly inefficient. We address these inefficiencies by intro-

ducing several methods that incorporate non-random techniques, improving the distribution of computational power while maintaining the favourable properties and theoretical guarantees of the random approaches.

We demonstrate the efficiency and versatility of the proposed framework when applied to various computationally demanding scenarios, including near-chaotic blowfly population growth and an extended case study of COVID-era tourism demand. In these scenarios, we show that the new proposed techniques provide robust and accelerated inference compared to state-of-the-art methods. In addition, we explore how the techniques can be implemented for efficient inference in other general scenarios.

Abstract

State-space models are widely used to model time series data where the observations depend on a latent process. The latent process consists of a sequence of latent states that evolve according to a specified system process. The observed time series data are then modelled as a function of the latent states via an observation process. Estimating the parameters of a state-space model within a Bayesian framework can be challenging. In this thesis, we consider these challenges and develop efficient Monte Carlo algorithms to aid inference. We propose two classes of method that, as a central theme, leverage approximate hidden Markov models for efficient inference.

In the first part of this thesis, we propose that approximate hidden Markov models can be used to design efficient Markov chain Monte Carlo proposal distributions, defined such that the usual theoretical guarantees apply. We discuss how the hidden Markov models are constructed under the proposed approach, the associated generality arising from the tuning parameters, and how these tuning parameters can be chosen efficiently in practice. We demonstrate that this proposed algorithm provides an efficient and robust alternative method for fitting state-space models, even for those that exhibit near-chaotic behaviour.

In the second part of this thesis, we develop an approximate hidden Markov model approach to designing efficient particle Markov chain Monte Carlo algorithms. In particular, we propose an approach to particle Gibbs with ancestor sampling that leverages approximate hidden Markov models to combat impoverishment within the sequential Monte Carlo steps of the original algorithm. We additionally propose that fixed approximations to the hidden Markov model can be used to substantially reduce the computational cost of the hidden Markov model and particle Gibbs with ancestor sampling algorithm. We demonstrate the efficiency of this proposed approach in several traditionally challenging examples, focusing on state-space models with regime switching.

List of acronyms

COVID, COVID-19	Coronavirus disease 2019
CSMC	Conditional sequential Monte Carlo
CSMC-AS	Conditional sequential Monte Carlo with ancestor sampling
ESS	Effective sample size
GPGAS	Grid particle Gibbs with ancestor sampling
HMM	Hidden Markov model
MCMC	Markov chain Monte Carlo
M-H	Metropolis-Hastings
PCA	Principal component analysis
PGAS	Particle Gibbs with ancestor sampling
PMPMH	Point mass proposal Metropolis-Hastings
RRMSE	Relative root mean squared error
SMC	Sequential Monte Carlo
SMC^2	Sequential Monte Carlo squared
SSM	State-space model
TPM	Transition probability matrix

Contents

Chapter 1

Introduction

Understanding how a system changes over time is of interest in many real-world scenarios. Communities at risk of extreme weather may be interested in changes in weather patterns; supermarkets may be interested in seasonal product fluctuations to plan optimal stocking strategies; and businesses may be interested in how tourism is recovering after COVID-19. Data collected sequentially over time are referred to as time series data and can be useful to record and understand temporal patterns. To learn more about the underlying system, statistical models for time series data formalise current knowledge of the system and data-generating processes. For example, a model may reflect our understanding of how an underlying system evolves, as well as extra sources of variability in the observed data from human or machine measurement error (Durbin and Koopman, 2012, Chapter 2).

State-space models (SSMs) account for the extra dynamic sources of variability in time series data via an additional unobserved latent process. The latent process consists of a sequence of unobserved states that evolve through time, forming a system process; with observed time series data modelled as a function of these latent states via an observation process. This distinction between the two processes and the resulting flexibility has led to the application of SSMs across various domains. Examples range from the management of threatened animal populations (where population estimates are recorded over time; Buckland et al., 2004; King et al., 2010; King, 2012; Newman et al., 2014) to the modelling of inflation rates with indirectly observed but time-varying features and heteroscedasticity (Koopman and Bos, 2004; Nadal-De Simone, 2000). However, despite the flexibility and range of applications of SSMs, inferring the unknown components of an SSM can be challenging in practice, particularly from a computational perspective. Existing computational approaches can be complex and

require practically infeasible run times. In this thesis, we propose new efficient computational methods for fitting SSMs to time series data. We therefore begin by defining SSMs, the associated notation, and the computational challenges.

1.1 State-space models

We consider SSMs for time series data observed at discrete (regular) time points up to a final time point, T, denoted by $y_{1:T} = (y_1, \ldots, y_T)$. We assume that the data can take continuous values and define a latent process of unobserved states, denoted $x_{1:T} = (x_1, \ldots, x_T)$. These latent states can also be continuous, taking values in some set $x_t \in \chi$ for each $t \in \{1, \ldots, T\}$ (Durbin and Koopman, 2012, Chapter 2). Both the observations and latent states can be a vector at each time point, for example, for each time $t \in \{1, \ldots, T\}$, $y_t \in \mathbb{R}^k$ for some integer k and $x_t \in \chi \subseteq \mathbb{R}^l$ for some integer l. An SSM models the observed time series data in terms of the latent states via two processes:

(i) a system process for the latent states, $x_{1:T}$, describing the evolution of the latent states via initial state and transition distributions. The system process is typically first-order Markov, i.e., the latent state(s) at time $t \in \{2, \ldots, T\}$ only depend on the state(s) at times $t - 1$;

(ii) an observation process linking the latent process with the observed data, $y_{1:T}$, through an observation distribution. The data at time $t \in \{1, \ldots, T\}$, y_t, are conditionally independent of the other states and observations given underlying latent state(s) at time t, x_t.

We illustrate the structure of an SSM via the graphical representation in Figure 1.1.

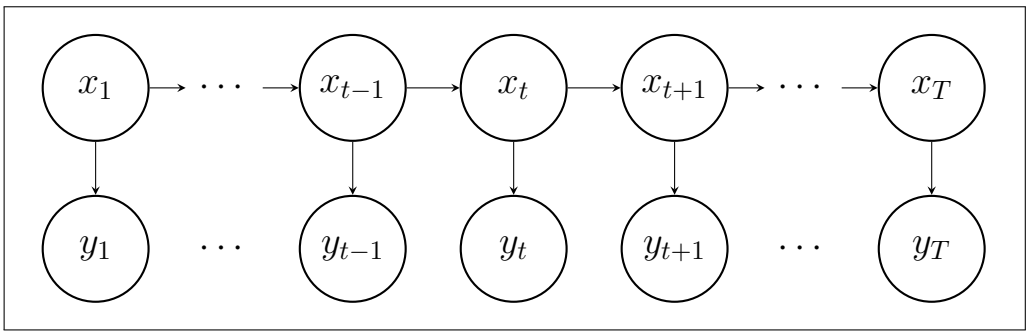


Figure 1.1: Graphical representation of a discrete-time SSM with (i) a system process of latent states, $x_{1:T}$, in the first layer and (ii) an observation process relating the latent states to the time series observations, $y_{1:T}$, in the output layer.

To define the SSM mathematically, we specify the system and observation processes determining the evolution of the latent states and observed time series data. Letting θ denote the associated static model parameters, the SSM can be written mathematically as

$$\begin{aligned} \textit{Initial state distribution}: &\quad p(x_1|\theta), \\ \textit{State transition distribution}: &\quad p(x_t|x_{t-1},\theta), \quad t = 2,\dots,T, \\ \textit{Observation distribution}: &\quad p(y_t|x_t,\theta), \quad t = 1,\dots,T, \end{aligned} \tag{1.1}$$

where we use 'density' and 'distribution' interchangeably. For $t = 2,\dots,T$, $p(x_t|x_{1:t-1},\theta) = p(x_t|x_{t-1},\theta)$ due to the Markov assumption in the state process. Further, for $t = 1,\dots,T$, $p(y_t|x_{1:T},\theta) = p(y_t|x_t,\theta)$ since the observations depend only on the latent state(s) at time t.

Hidden Markov models (HMMs) are often described as a particular case of an SSM in which the state space is discrete and finite (Rabiner and Juang, 1986), for example, $\chi = \{1,\dots,N\}$. We use this definition of an HMM consistently throughout this thesis. That is, we define the HMM by restricting the SSM in Equation (1.1) to discrete states, $k, n = 1,\dots,N$:

$$\begin{aligned} \textit{Initial state probabilities}: &\quad P(x_1 = n|\theta), \\ \textit{State transition probabilities}: &\quad P(x_t = n|x_{t-1} = k,\theta), \quad t = 2,\dots,T, \\ \textit{Observation distribution}: &\quad p(y_t|x_t = n,\theta), \quad t = 1,\dots,T. \end{aligned} \tag{1.2}$$

Both SSMs and HMMs are widely used to model time series observations that depend on an unobserved latent process (King, 2012; Lin et al., 2019; Newman

et al., 2022; Zeng and Wu, 2013). However, inference about the latent process and model parameters of an SSM can be particularly challenging. We focus on inference within a Bayesian framework and proceed by defining the inferential problem.

1.2 Fitting state-space models

For the general continuous-state SSMs defined in Equation (1.1), inference usually refers to estimating the model parameters (θ) or the latent states and model parameters (both $x_{1:T}$ and θ). An exception is the known model parameter case, for example in Kalman (1960). For Bayesian inference about the model parameters only, the target posterior distribution is $p(\theta|y_{1:T})$. The associated likelihood, conditional on the model parameters, is the integral of the joint distribution of the latent states and observations conditional on the model parameters, $p(x_{1:T}, y_{1:T}|\theta)$, over the latent states:

$$\begin{aligned} p(y_{1:T}|\theta) &= \int_{\chi^T} p(x_{1:T}, y_{1:T}|\theta) dx_{1:T} \\ &= \int_{\chi^T} p(x_1|\theta) \prod_{t=2}^{T} p(x_t|x_{t-1}, \theta) \prod_{t=1}^{T} p(y_t|x_t, \theta) dx_{1:T}, \end{aligned} \tag{1.3}$$

where χ^T is the product space of the state spaces at each time point. A closed-form expression for this (marginal) likelihood typically only exists when either the state space is discrete and finite (i.e., that of an HMM), or when the SSM is linear and Gaussian. In the discrete HMM case, calculating the integral in Equation (1.3) amounts to a summation over the state space. This summation can be calculated recursively using forward filtering to reduce computational cost, described in Rabiner (1989) and Rabiner and Juang (1986). In the linear Gaussian case, the marginal likelihood can also be calculated recursively using the Kalman filter (Durbin and Koopman, 2012, Chapter 2; Kalman, 1960). In brief, the Kalman filter calculates $p(x_t|y_{1:t}, \theta)$ for each $t = 1, \ldots, T$ using the conjugacy of the linear Gaussian distributions. Each factor of the likelihood decomposition $p(y_{1:T}|\theta) = p(y_1|\theta) \prod_{t=2}^{T} p(y_t|y_{1:t-1}, \theta)$ can be similarly obtained using the conjugacy of the Gaussian distributions. However, the marginal likelihood does not admit a closed-form expression for general continuous-state SSMs. Thus, most approaches to approximate inference in the Bayesian case are intractable (see Andrieu and Roberts, 2009; Beaumont, 2003 for exceptions).

We focus on approaches that infer θ by targeting the joint (Bayesian) posterior distribution of the model parameters *and* latent states (Hobert, 2011; Tanner and Wong, 1987):

$$p(x_{1:T}, \theta | y_{1:T}) \propto p(\theta) p(x_{1:T}, y_{1:T} | \theta), \tag{1.4}$$

This is often referred to as the data augmentation approach. The target posterior distribution now uses the joint distribution, $p(x_{1:T}, y_{1:T}|\theta)$ in Equation (1.3), directly and thus generally admits a closed-form expression up to proportionality. Standard random sampling methods are tractable in this case and can be applied to approximate this joint posterior distribution (Equation (1.4)). Samples approximating the posterior distribution of the model parameters, $p(\theta|y_{1:T})$, can then be obtained by simply disregarding the latent state samples and retaining the samples for θ, effectively marginalising the joint posterior distribution. The fact that sampling approaches are tractable when approximating the joint distribution is, implicitly, also useful when both the latent states and the model parameters are of interest.

A key challenge to inference using this joint distribution is inferring the many (often highly-correlated) latent states. The rest of this thesis describes approaches to inference via SSMs targeting $p(x_{1:T}, \theta|y_{1:T})$, the challenges associated with updating the latent states, and proposes efficient solutions.

1.3 Contributions and thesis outline

This thesis contributes to the existing literature on SSMs by providing new computationally efficient approaches to estimating both the latent states and model parameters of an SSM within a Bayesian framework. We introduce reliable partially deterministic approaches to estimation and demonstrate the approaches in several case studies.

The main material in this thesis is contained in one further background and literature review chapter (Chapter 2), two further chapters containing new research contributions (Chapters 3 and 4), and a concluding chapter discussing the contributions of the thesis and possible avenues for future research (Chapter 5). We provide a short description of each chapter below.

Chapter 2: Computational approaches to state-space model-fitting

In this chapter, we describe current approaches to estimating the latent states and model parameters of an SSM, including importance sampling, Markov chain Monte Carlo, and particle Gibbs algorithms.

The algorithms defined in this chapter are referred to throughout the rest of the thesis. In addition, we review the computational efficiency of each approach and motivate the proposed algorithms.

Chapter 3: Point mass proposal Metropolis-Hastings

We develop a novel block Metropolis-Hastings proposal distribution on the joint latent state and parameter space, referred to as the point mass proposal Metropolis-Hastings algorithm. The proposal distribution for updating the latent states is informed by a deterministic HMM, defined such that the usual theoretical guarantees of Markov chain Monte Carlo algorithms apply. We discuss how the HMMs are constructed, the generality of the approach from the tuning parameters, and how these tuning parameters can be chosen efficiently in practice. Finally, we demonstrate that the proposed algorithm provides an efficient and robust alternative method for fitting SSMs, even those that exhibit near-chaotic behaviour.

Chapter 4: Grid particle Gibbs with ancestor sampling

This chapter proposes an efficient approximate HMM method within the particle Gibbs framework to infer the latent states and model parameters, referred to as the grid particle Gibbs with ancestor sampling algorithm. We build efficient importance distributions for the latent states by discretising the SSM to an approximate but efficient HMM. We focus on the use of the deterministic HMM-importance distribution within particle Gibbs with ancestor sampling methods, where the proposed algorithm is particularly well-suited and efficient. We demonstrate this efficiency by focusing on challenging regime-switching models, including a post-COVID tourism demand model using high-dimensional Google Trends data.

Chapter 5: Conclusions

We summarise and conclude the thesis, discussing the main research contributions of the proposed approaches to the scientific field. We also discuss promising avenues for future research developing the proposed HMM approximation framework for improved and efficient inference using SSMs.

Chapter 2

Computational approaches to fitting state-space models

As discussed in Chapter 1, the marginal likelihood associated with a state-space model (SSM) in Equation (1.3) typically only admits a closed-form expression in the special cases when the SSM is linear and Gaussian, and in the discrete hidden Markov model (HMM) case. In the general non-linear or non-Gaussian case, alternative approaches to Bayesian inference are required. We focus on approaches that target the joint posterior distribution of both the latent states and model parameters, $p(x_{1:T}, \theta|y_{1:T})$, allowing for tractable approximations of the posterior distribution of the model parameters and/or latent states.

Examples of recent approaches to approximating the joint distribution include the use of optimisation-based variational inference (Blei et al., 2017; Wang et al., 2022) or approximate Bayesian computation (ABC; Ebert et al. (2019)). However, in general, these approaches lack theoretical guarantees, such as the consistency in the posterior estimates (studied, for example, by Frazier et al., 2022; Tran and Kohn, 2015; Wang and Titterington, 2004). We focus on arguably the most common approach to approximating the joint distribution of an SSM: Monte Carlo methods, which sample from the joint posterior distribution and yield consistent posterior estimates with well-studied theoretical guarantees (Durbin and Koopman, 2012, Chapter 11; Frühwirth-Schnatter, 2004; Tanner and Wong, 1987).

Given M independent samples, x^m for $m = 1, \ldots, M$, from a general distribution of interest, $p(x)$, Monte Carlo methods approximate $p(x)$ by

$$\hat{p}(x) = \frac{1}{M} \sum_{m=1}^{M} \delta_{x^m}(x), \tag{2.1}$$

where $\delta_{x^m}(x)$ denotes the Dirac function at $x = x^m$. Given this Monte Carlo approximation of $p(x)$, we can approximate expectations of a function $g(x)$ with respect to $p(x)$ by

$$\begin{aligned} E_p(g(x)) &= \int g(x)p(x)dx \\ &\approx \int g(x)\hat{p}(x)dx = \frac{1}{M}\sum_{m=1}^{M} g(x^m), \end{aligned}$$

since the integral of the Dirac function over the space of x is 1. The approximation on the right-hand side corresponds to the average of $g(x)$ evaluated at each sample and ensures almost sure convergence to $E_p(g(x))$ in the number of samples, and unbiasedness and consistency with a convergence rate of $\mathcal{O}(1/M)$ (Chopin and Papaspiliopoulos, 2020, Chapter 8; Doucet and Johansen, 2009). This is fundamental to Monte Carlo approaches and their widespread use. However, this direct Monte Carlo approach samples from the target distribution exactly, which is infeasible for the intractable joint posterior distribution of general SSMs, $p(x_{1:T}, \theta|y_{1:T})$.

Instead of sampling directly from $p(x)$, modern Monte Carlo methods generally sample from a *different* distribution, $q(x)$, and correct the samples using weights that only require a closed-form expression for the target distribution up to proportionality. One approach is rejection sampling (for example, in Martino, 2017), which samples from an 'envelope' distribution designed to closely capture the characteristics of the target distribution. The samples from the envelope distribution are then weighted and retained such that they approximate the target distribution (Rao et al., 2016). However, rejection sampling approaches are often inefficient since identifying envelope distributions can be challenging, particularly for multi-dimensional target spaces (Deligiannidis et al., 2020; Johansen, 2010). Two alternative approaches have become prominent and widely applied for Bayesian inference using the joint distribution of an SSM: importance sampling and Markov chain Monte Carlo (MCMC) (Auger-Méthé et al., 2021; Triantafyllopoulos, 2021, Chapter 6). In this chapter, we focus on importance sampling and MCMC approaches and describe each approach in turn.

The structure of this chapter is therefore as follows. First, we describe importance sampling and MCMC methods in turn and describe their use for inference using the joint posterior distribution of an SSM. We then describe the combination of importance sampling and MCMC methods in this context, typically

referred to as particle MCMC, before discussing the efficiency of each approach in practice. This motivates the new methodology proposed in the rest of this thesis.

2.1 Importance sampling

Importance sampling, described in Geweke (1989), is a Monte Carlo method that approximates a distribution, $p(x)$, by weighting samples from an alternative distribution, $q(x)$, defined in the same space as $p(x)$ (i.e., $p(x) > 0 \implies q(x) > 0$). Samples from $q(x)$ ($\{x^m\}_{m=1}^M$ for M samples) first approximate $q(x)$ by

$$q(x) \approx \frac{1}{M} \sum_{m=1}^{M} \delta_{x^m}(x). \tag{2.2}$$

Thus, the target distribution can be approximated using these samples by correcting for the alternative sampling distribution, $q(x)$:

$$p(x) = \frac{p(x)}{q(x)} q(x) \approx \frac{1}{M} \sum_{m=1}^{M} \frac{p(x^m)}{q(x^m)} \delta_{x^m}(x), \tag{2.3}$$

where the Monte Carlo approximation in Equation (2.2) is substituted for $q(x)$ (though not the reciprocal $q(x)$ term) and we assume that $p(x)$ can be evaluated at x^m, $m = 1, \ldots, M$. By defining the ratio of the true density to the alternative importance sampling density as weights, i.e., $w(x^m) = p(x^m)/q(x^m)$, $m = 1, \ldots, M$, we obtain a weighted Monte Carlo approximation of $p(x)$:

$$\hat{p}(x) = \frac{1}{M} \sum_{m=1}^{M} w(x^m) \delta_x^m(x), \quad w(x^m) = \frac{p(x^m)}{q(x^m)}.$$

Importance sampling is typically used when sampling directly from a distribution of interest is not possible or is inefficient. However, this form of importance sampling requires that the target distribution can be evaluated exactly to calculate the importance weights and thus cannot be applied for inference via SSMs in the general case (Chapter 1).

In cases when the target distribution admits a closed form up to proportionality (such as the joint posterior distribution of the latent states and model parameters considered throughout this thesis), the importance sampling framework can be extended to additionally approximate the normalising constant (Liu, 2004, Chapter 2). Suppose that we can only evaluate the target distribution, $p(x)$, up to proportionality, i.e., that $p(x)$ can be re-written as $p(x) = \tilde{p}(x)/\int \tilde{p}(x)dx$,

where we can evaluate $\tilde{p}(x)$ but not the denominator (normalising constant). The importance sampling Monte Carlo approximation of the form Equation (2.3) can be applied to approximate both the numerator and normalising constant of $p(x)$:

$$p(x) = \frac{\tilde{p}(x)}{\int \tilde{p}(x)dx} \approx \sum_{m=1}^{M} \frac{\tilde{p}(x^m)/q(x^m)}{\sum_{k=1}^{M} \tilde{p}(x^k)/q(x^k)} \delta_{x^n}(x).$$

Equivalently, the Monte Carlo approximation can be re-written in terms of *normalised* weights, $W(x^m)$, as

$$\hat{p}(x) = \sum_{n=1}^{N} W(x^m)\delta_x^m(x),$$
$$W(x^m) = \frac{w(x^m)}{\sum_{k=1}^{M} w(x^k)}, \quad w(x^m) = \frac{\tilde{p}(x^m)}{q(x^m)},$$

giving the weighted Monte Carlo approximation of $p(x)$. Note that the normalising constant, $\int \tilde{p}(x)dx$, is estimated by the normalising constant of the weights, $\frac{1}{M}\sum_{k=1}^{M} w(x^k)$. This estimator of the normalising constant is unbiased (Elvira and Martino, 2021).

The challenges in designing importance distributions for SSMs largely originate from the dimensionality required and approximation error. The rest of this section considers improved importance distributions using the temporally-sequential structure of an SSM so that lower-dimensional importance distributions are required.

2.1.1 Sequential Monte Carlo

Sequential Monte Carlo (SMC) methods (Chopin and Papaspiliopoulos, 2020, Chapter 10; Doucet and Johansen, 2009) can be applied to SSMs to approximate the posterior distribution of the latent states conditional on the model parameters, $p(x_{1:T}|y_{1:T}, \theta)$. To approximate $p(x_{1:T}|y_{1:T}, \theta)$, SMC methods apply importance sampling sequentially targeting each $p(x_{1:t}|y_{1:t}, \theta)$, $t = 1, \ldots, T$, in turn until time $t = T$. The sequential steps are derived by first noting that, for the general SSM defined in Equation (1.1), $p(x_{1:t}|y_{1:t}, \theta)$ can be written recursively as

$$p(x_{1:t}|y_{1:t}, \theta) = \frac{p(x_{1:t-1}|y_{1:t-1}, \theta)p(x_t, y_t|x_{t-1}, \theta)}{p(y_t|y_{1:t-1}, \theta)}, \;\; t = 1, \ldots, T. \tag{2.4}$$

At $t = 1$, the components on the right-hand side of Equation (2.4) are defined as $p(x_{1:0}|y_{1:0}, \theta) = 1$, $p(x_1, y_1|x_0, \theta) = p(x_1, y_1|\theta)$, and $p(y_1|y_{1:0}, \theta) = p(y_1|\theta)$. The

right-hand side of Equation (2.4) follows by the definition of conditional probability, the first-order Markov property of x_t, and the conditional independence of x_t and y_t from $y_{1:t-1}$ given x_{t-1}:

$$\begin{aligned} p(x_{1:t}|y_{1:t},\theta) &= \frac{p(x_{1:t},y_t|y_{1:t-1},\theta)}{p(y_t|y_{1:t-1},\theta)} \\ &= \frac{p(x_{1:t-1}|y_{1:t-1},\theta)p(x_t,y_t|x_{t-1},\theta)}{p(y_t|y_{1:t-1},\theta)}. \end{aligned}$$

Suppose, at time $t \in \{2,\ldots,T\}$, we have an importance sampling approximation of $p(x_{1:t-1}|y_{1:t-1},\theta)$ at the previous time point:

$$\hat{p}(x_{1:t-1}|y_{1:t-1},\theta) = \sum_{m=1}^{M} W_{1:t-1}(x_{1:t-1}^m)\delta_{x_{1:t-1}^m}(x_{1:t-1}),$$

for a set of M samples ('particles') and associated normalised importance weights, $\{x_{1:t-1}^m, W_{1:t-1}(x_{1:t-1}^m)\}_{m=1}^M$. We can extend this approximation of the conditional distribution at time t via a low-dimensional importance density of the form $q(x_t|y_t,x_{t-1},\theta)$. The particles are propagated to time t by sampling a set of particles from the importance distribution, i.e., $x_t^m \sim q(x_t|y_t,x_{t-1}^m,\theta)$ for $m = 1,\ldots,M$. Combined with the sequential decomposition of $p(x_{1:t}|y_{1:t},\theta)$ in Equation (2.4), we obtain the approximation:

$$\begin{aligned} &\hat{p}(x_{1:t}|y_{1:t},\theta) = \sum_{m=1}^{M} W_{1:t}(x_{1:t}^m)\delta_{x_{1:t}^m}(x_{1:t}), \\ &W_{1:t}(x_{1:t}^m) = \frac{w_{1:t}(x_{1:t}^m)}{\sum_{k=1}^m w_{1:t}(x_{1:t}^k)}, \;\; w_{1:t}(x_{1:t}^m) \propto w_{1:t-1}(x_{1:t-1}^m)\frac{p(x_t^m|x_{t-1}^m,\theta)p(y_t|x_t^m,\theta)}{q(x_t^m|y_t,x_{t-1}^m,\theta)}. \end{aligned} \tag{2.5}$$

For all time points, $t = 2,\ldots,T$, $w_{1:t}^{1:M} = \{w_{1:t}(x_{1:t}^m)\}_{m=1}^M$ denotes the sequence of unnormalised weights and $W_{1:t}^{1:M} = \{W_{1:t}(x_{1:t}^m)\}_{m=1}^M$ the sequence of normalised weights such that $\sum_{m=1}^M W_{1:t}(x_{1:t}^m) = 1$. Both the particles and weights are defined as a function of the particles and weights at the previous time point. The initial calculations at time $t = 1$ use $w_{1:0}(x_{1:0}) = 1$ and $q(x_1|y_1,x_0,\theta) = q(x_1|y_1,\theta)$ in Equation (2.5). Thus, we obtain a recursive approximation of $p(x_{1:t}|y_{1:t},\theta)$ given by the set of particle trajectories and (normalised) weights, $\{x_{1:t}^m, W_{1:t}^m\}_{m=1}^M$. A common choice for the importance density is the latent state distribution, i.e., $q(x_1|y_1,\theta) = p(x_1|\theta)$ and $q(x_t|y_t,x_{t-1}\,\theta) = p(x_t|x_{t-1},\theta)$ at time $t \in \{2,\ldots,T\}$. The resulting SMC algorithm is often referred to as a bootstrap particle filter (Gordon et al., 1993).

Particle degeneracy occurs in the SMC algorithm when many particles have low weights, eliminating their effective use for posterior estimation. Moreover, degeneracy is inevitable for almost all particle paths as the number of SMC recursions increases (Doucet and Johansen, 2009). To prevent particle degeneracy, an SMC algorithm typically incorporates an additional resampling step into its recursions, eliminating particles with low weights and replicating those with high weights. Before the importance sampling step at each time point $t = 2, \ldots, T$, M new particle trajectories are sampled from the distribution of the existing particle trajectories (conditional on their weights), denoted $r(a_t|W_{t-1}^{1:M})$. That is, we sample trajectory indices from

$$a_t^m \sim r(a_t|W_{1:t-1}^{1:M}), \;\; m = 1, \ldots, M, \tag{2.6}$$

and set $x_{1:t-1}^m = x_{1:t-1}^{a_t^m}$ for $m = 1, \ldots, M$. However, resampling reduces the diversity in the particles. Thus, resampling steps are often only executed when they are deemed necessary, for example, resampling when the effective sample size of the particles falls below a certain threshold (Del Moral et al., 2012). Throughout this thesis, we assume standard multinomial resampling, i.e., that $r(a_t|W_{1:t-1}^{1:M})$ is a multinomial distribution with probabilities equal to the normalised weights for each $t = 2, \ldots, T$, and explore different thresholds for resampling based on the effective sample size of the particles. Other resampling methods are discussed, for example, in Douc et al. (2005) and Liu (2004, Chapter 2). We present the SMC algorithm with both the sequential importance sampling and resampling steps at each time step in Algorithm 1.

2.1.2 Joint parameter and state inference

SMC provides a flexible framework for sampling from the conditional posterior distribution of the latent states, $p(x_{1:T}|y_{1:T}, \theta)$, for known and fixed model parameters, θ. However, in this thesis, we consider θ also unknown: the inferential target is the joint posterior distribution of $x_{1:T}$ and θ, $p(x_{1:T}, \theta|y_{1:T})$. SMC and other importance sampling methods can be applied for inference targeting this joint distribution. In the rest of this section, we describe these approaches and discuss some of the challenges in practice.

For linear Gaussian SSMs, (Durbin and Koopman, 2012, Chapter 13) describe an importance sampling method for inference targeting the joint distribution. Using the Kalman filter, the marginal likelihood $p(y_{1:T}|\theta)$ is evaluated for use in importance sampling steps targeting $p(\theta|y_{1:T})$ and the Kalman filter is applied

Algorithm 1 Sequential Monte Carlo (SMC)

Input:

- M = number of particles
- $q(x_1|y_1,\theta)$, $\{q(x_t|y_t,x_{t-1},\theta)\}_{t=2}^{T}$ = importance distributions
- θ = known model parameters

for $m = 1,\ldots,M$ **do**
  sample $x_1^m \sim q(x_1|y_1,\theta)$
calculate $w_1^{1:M}$ and $W_1^{1:M}$ ▷ Equation (2.5)

for $t = 2,\ldots,T$ **do**
  for $m = 1,\ldots,M$ **do**
    sample $a_t^m \sim r(a_t|W_{1:t-1}^{1:M})$ ▷ Equation (2.6)
    sample $x_t^m \sim q(x_t|y_t, x_{t-1}^{a_t^m},\theta)$
    set $x_{1:t}^m = (x_{1:t-1}^{a_t^m}, x_t^m)$
  calculate $w_{1:t}^{1:M}$ and $W_{1:t}^{1:M}$ ▷ Equation (2.5)

return $\{x_{1:T}^m, W_{1:T}^m\}_{m=1}^{M}$

targeting $p(x_{1:T}|y_{1:T},\theta)$, obtaining samples from the joint posterior $p(x_{1:T},\theta|y_{1:T})$. For general SSMs, when the marginal likelihood cannot be evaluated exactly, several importance sampling approaches have been proposed that augment model parameter sampling into SMC recursions. A first approach samples particles for θ from their prior, $p(\theta)$, and augments the SMC recursions with the parameters to target $p(x_{1:t},\theta|y_{1:t})$ at each recursion $t = 1,\ldots,T$. Eventually, at time $t = T$, samples from the joint posterior distribution, $p(x_{1:T},\theta|y_{1:T})$ can be obtained via weighted pairs of samples for θ and $x_{1:T}$. However, since the parameter samples are not updated dynamically, each resampling step of the SMC recursions reduces the number of unique samples for θ (Triantafyllopoulos, 2021, Chapter 6). Artificial dynamics can be added to the parameters (often referred to as jittering) to reduce impoverishment of the parameter samples, for example in Gilks and Berzuini (2002), but can have a high computational cost or introduce bias into the posterior estimates (Flury and Shephard, 2009; Kitagawa, 1998; Liu and West, 2001). An alternative, related approach is SMC^2 (Chopin et al., 2012; Chopin and Papaspiliopoulos, 2020, Chapter 18; Fulop and Li, 2013), which runs separate SMC algorithms for each parameter particle initially sampled from $p(\theta)$. However, such approaches can be computationally costly in practice (Kantas et al., 2015).

An additional but key challenge to inference via SMC is sample impoverishment, which is highly related to particle degeneracy, occurring when particles

degenerate in the SMC algorithm and are discarded upon resampling. When there is a high rate of sample impoverishment, many particle trajectories with low weights are discarded, reducing the number of unique particles in the SMC approximation of the latent state distribution. In turn, this increases the variance of the Monte Carlo estimates of the posterior distribution. Several approaches have been proposed that, for example, reduce sample impoverishment via importance distributions that more accurately approximate each target distribution of the SMC recursions, $p(x_{1:t}|y_{1:t}, \theta)$, reducing the number of low-weight particle trajectories (Li et al., 2014; Wang et al., 2017). These approaches, and a new proposed approach, are discussed in Chapter 4. We now discuss alternatives to importance sampling and SMC for inference targeting the joint posterior distribution of an SSM, MCMC and particle Gibbs.

2.2 Markov chain Monte Carlo

MCMC is an alternative approach to approximating intractable distributions that admit a closed-form expression up to proportionality. The first known work on MCMC appears in the physics literature in the 1950s in Metropolis et al. (1953), but was popularised in statistics in the 1990s due, in part, to advances in computational power (Gelfand et al., 1989). MCMC methods have continued to develop and are still widely applied to solve problems that involve intractable distributions (Brooks et al., 2011; Jones and Qin, 2022).

In contrast to the formulation of Monte Carlo approximations within the importance sampling frameworks, MCMC methods construct a Markov chain that admits the target distribution as its limiting distribution. MCMC methods therefore yield dependent samples as opposed to the independent samples achieved from importance sampling. We define a Markovian transition kernel, i.e., the probability of some x' in the target space given x, denoted $T(x, x')$ (Geyer, 2011). The transition kernel is defined such that the associated Markov chain converges to the target distribution, $p(x)$. Values are then sampled sequentially from this Markov transition kernel. Once the Markov chain has converged to its limiting equilibrium distribution, subsequent samples can be regarded as (dependent) samples from the target distribution, providing a Monte Carlo approximation of the form Equation (2.1).

Sampling from $p(x)$ directly would result in a Markov chain converging to $p(x)$, but we have assumed that this distribution cannot be sampled from. Instead, the use of an intermediate transition kernel, $T(x, x')$, allows values to be sampled

from an alternative distribution. In this section, we describe standard MCMC approaches to defining $T(x, x')$ and the associated challenges.

2.2.1 The Metropolis-Hastings algorithm

The Metropolis and Metropolis-Hastings algorithms (in Metropolis et al., 1953 and Hastings, 1970 respectively) sample from a target distribution, $p(x)$, by defining Markovian transition kernels to sample from an alternative 'proposal distribution' and correct the samples via an accept/reject step. In this section, we describe the more general Metropolis-Hastings (M-H) algorithm, which uses general proposal distributions defined conditionally on the current Markov chain sample, x, denoted $q(x'|x)$. This proposal distribution is then used to define a transition kernel from x to x' (of the form $T(x, x')$) as follows:

$$T(x, x') = q(x'|x)\alpha(x, x'), \quad \alpha(x, x') = \min\left(1, \frac{p(x')q(x|x')}{p(x)q(x'|x)}\right). \tag{2.7}$$

Values are sampled according to this transition kernel by first sampling $x' \sim q(x'|x)$ from the proposal distribution. We then accept or reject the sampled value, x', according to some specified acceptance probability, $\alpha(x, x')$, and continue to sample values in this way over many iterations. It can be shown that under certain conditions on the proposal distribution (see below), the stationary distribution of this Markov chain is $p(x)$. Thus, once the chain has reached its stationary distribution, samples can be regarded as distributed according to $p(x)$. The M-H algorithm is summarised in Algorithm 2.

Algorithm 2 Metropolis-Hastings (M-H)

Input:
- S = number of MCMC iterations
- $q(x'|x)$ = proposal distribution
- $x^{(0)}$ = initial value(s)

for $s = 1, \dots S$ **do**
  sample $x' \sim q(x'|x^{(s-1)})$
  calculate $\alpha(x^{(s-1)}, x')$ ▷ Equation (2.7)
  sample $u \sim U(0, 1)$
  if $u \leq \alpha(x^{(s-1)}, x')$ **then**
    $x^{(s)} \leftarrow x'$
  else $x^{(s)} \leftarrow x^{(s-1)}$

return $\{x^{(s)}\}_{s=1}^{S}$

Using standard Markov chain theory, we provide a sketch of a proof that the M-H samples converge to $p(x)$ and provide the necessary conditions on the proposal distribution. If the proposal distribution, $q(x'|x)$, is defined such the Markov chain can simulate any value with $p(x) > 0$ in a finite number of steps, and the chain is not periodic or transient, then the M-H transition kernel in Equation (2.7) admits $p(x)$ as its limiting distribution.

Sketch proof. Ensuring that any value such that $p(x) > 0$ can be simulated in a finite number of steps ensures that the Markov chain is irreducible. Combined with the Markov chain not being periodic or transient, the Markov chain has a unique stationary distribution (Gelman et al., 2013). To show that this unique stationary distribution is the target distribution, $p(x)$, a standard approach uses the more general detailed balance property as in (Tierney, 1994), also called reversibility (Geyer, 2011). The detailed balanced property is that

$$p(x)T(x, x') = p(x')T(x', x),$$

i.e., the density of x and moving from x to x' is the same as the reverse. If the detailed balance property holds then the Markov chain admits $p(x)$ as its unique stationary distribution. For the M-H kernel, we have that

$$\begin{aligned} p(x)T(x, x') &= p(x)q(x'|x)\alpha(x, x') \\ &= \min\Big(p(x)q(x'|x), p(x')q(x|x')\Big) \\ &= p(x')q(x|x') \min\Big(\frac{p(x)q(x'|x)}{p(x')q(x|x')}, 1\Big) = p(x')T(x', x). \end{aligned}$$

Thus, the detailed balance condition holds and the M-H kernel admits $p(x)$ as its limiting distribution.

Subject to the proposal distribution, $q(x'|x)$, ensuring that the Markov chain is irreducible, aperiodic, but not transient, the M-H algorithm produces samples from $p(x)$ upon convergence by design. An additional important feature of the M-H algorithm is that $p(x)$ only needs to be evaluated up to proportionality since the normalising constants cancel in the acceptance probability (Equation (2.7)). Thus, the M-H algorithm, similar to some forms of importance sampling, is used widely when the normalising constant of a distribution does not admit a closed-form expression.

The design of the proposal distribution is critical in the performance of the M-H algorithm. Ideally, proposal distributions are designed to ensure good 'mixing': fast exploration of the space of $p(x)$. This leads to an M-H algorithm that

converges quickly while minimising the correlation between samples, producing posterior estimates with low Monte Carlo error. Proposal distributions highly dependent on the previous value often have high acceptance probabilities, ensuring frequent moves are made around the space, but these moves are often small and may not explore the space sufficiently to achieve good mixing. Larger moves can be made by increasing the variance of the proposal distribution but this can lead to poor mixing by reducing the acceptance probability. Conversely, independent proposal distributions of the form $q(x'|x) = q(x')$ may yield poor acceptance probabilities if they are a poor approximation of $p(x)$, leading to poor mixing.

For multi-dimensional target spaces, efficient proposal distributions are particularly challenging to design due to the complexity required in the proposal distribution. A common alternative approach implements M-H steps over lower-dimensional sub-spaces of the target distribution since efficient low-dimensional proposal distributions are typically simpler to design. This is often referred to as single-site updating if individual parameters are updated in separate M-H steps or block updating if multiple parameters are updated in each M-H step (Fearnhead, 2011). In fact, the target distributions on the lower-dimensional spaces may now be formed of conjugate densities and admit tractable sampling distributions.

2.2.2 Gibbs sampling

We have assumed that we cannot sample directly from the target distribution. However, we may be able to sample from nested conditional distributions of the target distribution, removing the possibility of poor mixing due to low acceptance probabilities.

Suppose that the target variable, x, can be divided into D dimensions, $x = (x_1, \ldots, x_D)$, such that we can sample directly from each $\{p(x_d|x_{-d})\}_{d=1}^{D}$, where x_{-d} is the components of x except for x_d. The Gibbs sampler (Geman and Geman, 1993) produces samples jointly distributed according to $p(x) = p(x_1, \ldots, x_D)$ by sampling from each conditional distribution in turn. To define a valid Markov transition kernel, each x_d is sampled conditional on the most recent Markov chain samples for x_{-d}. The Gibbs sampler is summarised in Algorithm 3.

The Gibbs sampler can be seen as a special case of the M-H algorithm with proposal distribution $q(x'_d|x_{-d}) = p(x'_d|x_{-d})$, leading to a conditional M-H algorithm with acceptance probability 1 from Equation (2.7). It follows that when we can only sample some of the components of x directly from their conditional distribution, it is possible to combine Gibbs updates with conditional M-H updates targeting $p(x)$. This is frequently referred to as a Metropolis-within-Gibbs

Algorithm 3 Gibbs sampler

Input:

- S = number of MCMC iterations
- $x^{(0)}$ = initial value(s)

for $s = 1, \dots S$ **do**

set $x^{(s)} \leftarrow x^{(s-1)}$

for $d = 1, \dots, D$ **do**

sample $x_d^{(s)} \sim p(x_d | x_{-d}^{(s)})$

return $\{x^{(s)}\}_{s=1}^{S}$

algorithm (Gamerman and Lopes, 2006; Latuszyński et al., 2013). However, in either a full Gibbs or Metropolis-within-Gibbs sampler, the chain may explore the space slowly if the components of x are highly correlated since each component is essentially 'anchored' to the others (Fearnhead, 2011).

2.2.3 Markov chain Monte Carlo for state-space models

We now focus on inference using SSMs, for which the challenge of designing efficient proposal distributions is a key focus of the rest of this chapter and thesis. As discussed in previous sections, the joint distribution of a general SSM, $p(x_{1:T}, \theta | y_{1:T})$, only admits a closed-form expression up to proportionality. The M-H algorithm is therefore widely applied for inference under this joint distribution (Triantafyllopoulos, 2021, Chapter 6).

Within each iteration of the M-H algorithm, the latent states and model parameters can be updated simultaneously from their joint distribution. However, in practice, the latent states and model parameters are updated in turn from their full conditional distributions (in Metropolis-within-Gibbs-style updates), since updating the model parameters is typically straightforward within this framework (Fearnhead, 2011). We summarise this updating strategy in Algorithm 4. In this case, the model parameter updates target the full conditional distribution, $p(\theta | y_{1:T}, x_{1:T})$, which can often be efficiently sampled from using standard proposal distributions or Gibbs updates (Carter and Kohn, 1994; Durbin and Koopman, 2012, Chapter 13; Frühwirth-Schnatter, 2004; Tanner and Wong, 1987). However, efficiently updating the latent states from $p(x_{1:T} | y_{1:T}, \theta)$, can be challenging. We first focus on the case where the SSM is a discrete HMM

and the latent states can be sampled directly from their conditional distribution, before considering approaches to designing M-H updates when the conditional distribution cannot be sampled from directly.

Algorithm 4 Conditional update Metropolis-Hastings algorithm

Input:

- S = number of MCMC iterations
- $x_{1:T}^{(0)}$, $\theta^{(0)}$ = initial values

for $s = 1, \dots S$ **do**

update $x_{1:T}^{(s)}$ targeting $p(x_{1:T}|\theta^{(s-1)}, y_{1:T})$

update $\theta^{(s)}$ targeting $p(\theta|x_{1:T}^{(s)}, y_{1:T})$

return $\{x_{1:T}^{(s)}, \theta^{(s)}\}_{s=1}^{S}$ targeting $p(x_{1:T}, \theta|y_{1:T})$

2.2.4 Hidden Markov models

When the SSM is an HMM (i.e., the state space is discrete and finite; Equation (1.2)), the conditional distribution of the latent states, $p(x_{1:T}|y_{1:T}, \theta)$, is tractable and can be sampled from directly. We can sample latent states from this distribution at a computational cost of $\mathcal{O}(NT)$ (for N discrete states) using the forward-filtering backward-sampling algorithm introduced in Rabiner and Juang (1986). We first initialise the algorithm by calculating a 'filtering' distribution at time $t = 1$:

$$P(x_1 = n|y_1, \theta) \propto P(x_1 = n|\theta)p(y_1|x_1 = n, \theta), \quad n = 1, \dots, N. \tag{2.8}$$

For $t = 2, ..., T$, we then calculate the *forward-filtering* probability mass functions recursively for each state value $n = 1, \dots, N$:

$$P(x_t = n \mid y_{1:t}, \theta) \propto p(y_t|x_t = n, \theta) \times \sum_{j=1}^{N} P(x_{t-1} = j|y_{1:t-1}, \theta)P(x_t = n|x_{t-1} = j, \theta). \tag{2.9}$$

Values for the latent states, $x'_{1:T}$, are sampled from their conditional distribution, $P(x_{1:T}|y_{1:T}, \theta)$ using *backward-sampling* steps. We first sample an x'_T from

$$\{n, P(x_T = n \mid y_{1:T}, \theta)\}_{n=1}^N,$$

and for $t = T-1, ..., 1$, we sample an x'_t from

$$\{n, P(x_t = n \mid x_{t+1} = x'_{t+1}, y_{1:t}, \theta)\}_{n=1}^N,$$

where, for $n = 1, \ldots, N$,

$$P(x_t = n \mid x_{t+1} = x'_{t+1}, y_{1:t}, \theta) \propto P(x_t = n \mid y_{1:t}, \theta) P(x_{t+1} = x'_{t+1} | x_t = n, \theta). \tag{2.10}$$

We summarise the forward-filtering backward-sampling for HMMs in Algorithm 5. Although we focus on sampling from the conditional distribution of the latent states, the forward-filtering recursions also calculate the likelihood, $p(y_{1:T}|\theta)$, and backward steps can be used to find likely states under the marginal state distributions, $p(x_t|y_{1:T}, \theta)$ for each time $t = 1, \ldots, T$, referred to as local state decoding (McClintock et al., 2020).

Algorithm 5 Forward-filtering backward-sampling

Input:
- $\{1, \ldots, N\}$ = discrete HMM states
- θ = known model parameters

for $t = 1, \ldots, T$ **do**
  for $n = 1, \ldots, N$ **do**
    calculate $P(x_t = n|y_{1:t}, \theta)$ ▷ Equations (2.8) and (2.9)

sample x'_T from $\{n, P(x_T = n|y_{1:T}, \theta)\}_{n=1}^N$
for $t = T-1, \ldots, 1$ **do**
  sample x'_t from $\{n, P(x_t = n|x_{t+1} = x'_{t+1}, y_{1:t}, \theta)\}_{n=1}^N$ ▷ Equation (2.10)

return $x'_{1:T} \sim P(x_{1:T}|y_{1:T}, \theta)$

2.2.5 Proposal distributions for the latent states

In general, we cannot sample directly from the conditional distribution of the latent states, $p(x_{1:T}|y_{1:T}, \theta)$, and must specify a proposal distribution to implement the M-H algorithm. Designing an efficient proposal distribution for the latent states is often particularly challenging. From Sections 2.2.1 and 2.2.2, the ideal

proposal distribution with respect to the mixing and convergence of the M-H algorithm:

1. updates all latent states simultaneously to minimise the correlation between samples for different states,

2. proposes values for the latent states independently of the previous samples to minimise the correlation between samples for the same latent state,

3. accepts all proposed values to minimise the correlation between the sampled values for the states.

Clearly, these conditions are satisfied if the latent states are sampled directly from $p(x_{1:T}|y_{1:T}, \theta)$ but we have assumed that this is not possible. We therefore consider approaches to designing efficient proposal distributions for the latent states targeting $p(x_{1:T}|y_{1:T}, \theta)$. We suggest that these approaches can be broadly categorised by which of conditions 1-3 are prioritised and we discuss some of the main approaches in turn.

Approximations of $p(x_{1:T}|y_{1:T}, \theta)$

It is possible to update all of the latent states simultaneously and independently of previous chain values, for example, via a proposal distribution of the form $q(x_{1:T}|y_{1:T}, \theta)$. This proposal distribution fulfils conditions 1-2 but can be challenging to design to achieve a high acceptance probability, fulfilling condition 3. Typically, there are several latent states (at least one for each time point) and the latent states are often highly correlated (Fearnhead, 2011). Thus, this approach often requires a high-dimensional proposal distribution that accurately approximates the state dynamics.

Gaussian approximation methods, such as Laplace approximations (Kristensen et al., 2016) or extensions to the Kalman filter (van der Merwe et al., 2004), can often be applied efficiently when the SSM is well-approximated by Gaussian distributions but can perform poorly for general nonlinear or non-Gaussian SSMs (Carter and Kohn, 1994). Alternatively, Hamiltonian Monte Carlo uses Hamiltonian dynamics to propose values for the latent states that aim to move in the direction of high probability mass (Duane et al., 1987; Hirt et al., 2021) but can incur a high computational cost due to expensive gradient calculations (Newman et al., 2022). Several other approaches to designing proposal distributions for the latent states have been proposed, including the use of variational approximations of the posterior conditional distribution (Ghahramani and Beal, 1999)

and weighted expectation maximisation (Barra et al., 2016). Further reviews are given in Auger-Méthé et al. (2021) and Newman et al. (2022). In Section 2.3, we discuss an additional general approach that uses SMC to design proposal distributions for the latent states.

Adaptive proposal distributions

It is possible to use information from the history of the MCMC iterations to improve upon the acceptance probabilities associated with independent proposal distributions (relaxing condition 2). Such state-adaptive strategies are called adaptive MCMC algorithms. Broadly, adaptive MCMC algorithms learn the hyper-parameters of proposal distributions from the history of the MCMC iterations to optimise their performance (Gilks et al., 1994; Haario et al., 2001; Rosenthal, 2011). Subject to certain ergodicity conditions on the nature of the adaptation outlined in Andrieu and Thoms (2008), adaptive MCMC methods can be applied to inference targeting the conditional distribution of the latent states and can perform well in practice. However, these approaches can result in poor mixing in the MCMC iterations if the proposal distribution is highly dependent on the previous state samples. Maire et al. (2019) and Niemi and West (2010) propose adaptive mixture approaches and Titsias and Dellaportas (2019) show an approach to optimising the parameters of a general proposal distribution based on the previous samples and a maximum entropy regularised objective function. For a further review of general adaptive MCMC approaches see Jin et al. (2019).

Block updating strategies

To reduce the required dimension of the proposal distribution, an alternative approach updates each latent state individually from its full conditional distribution. However, this approach relaxes condition 1 and typically leads to poor mixing due to correlation in the state process (Fearnhead, 2011). Instead, it is often possible to update the latent states in low-dimensional blocks. Further, the correlation between the state samples at the block boundaries can be reduced by updating the states in blocks that overlap temporally, or by randomly sampling the blocks to update at each iteration (Chib and Ramamurthy, 2010). Although block-updating strategies require simpler proposal distributions for each block when compared to the global update approach, proposal distributions can still be challenging to design, requiring an efficient proposal distribution that accounts for the correlation between the states (de Freitas et al., 2001; Shephard and Pitt, 1997).

2.3 Particle Gibbs

In this section, we discuss a final approach to efficient inference targeting the joint posterior distribution of an SSM, $p(x_{1:T}|y_{1:T}, \theta)$, that combines sequential Monte Carlo (Section 2.1.1) and MCMC (Section 2.2). Methods that combine the importance sampling and MCMC frameworks have been proposed widely, for example in Radivojević and Akhmatskaya (2020) and Xu (2019). A popular approach is particle MCMC (Andrieu et al., 2010; Kantas et al., 2015), which applies SMC to the challenging latent state inference component and corrects the approximation using MCMC steps.

Here, we discuss the particle Gibbs algorithm in Andrieu et al. (2010) that designs MCMC updates for the states, $x_{1:T}$, conditional on the model parameters, θ, observations, $y_{1:T}$, and the previously-generated trajectories of state values, which yields samples from $p(x_{1:T}|y_{1:T}, \theta)$ (Chopin and Singh, 2015; Murphy and Godsill, 2016). Crucially, the particle Gibbs algorithm uses SMC-type approximations of the latent states conditional on the current chain values, resulting in proposed values for the entire latent state vector that are always accepted. The particle Gibbs approach therefore fulfils conditions 1 and 3 of the ideal M-H proposal distribution in Section 2.2.3 and somewhat relaxes condition 2.

To describe the particle Gibbs algorithm in detail, we first note that an approximation of $p(x_{1:T}|y_{1:T}, \theta)$ is derived using a variant of the SMC algorithm (Algorithm 1), the conditional SMC (CSMC) algorithm. At each MCMC iteration, the CSMC algorithm first conditions on the current latent states by fixing a 'reference trajectory' to their values. The remaining particles are then sampled via standard SMC steps and all particles are weighted as in Equation (2.5). We assume that the last particle trajectory is the reference trajectory, i.e., $x_{1:T}^{M} = x_{1:T}^{(s-1)}$ for MCMC iteration s and M particles. However, any trajectory can be chosen as the reference trajectory provided that the same trajectory index is chosen for all time points.

Example CSMC particle dynamics are illustrated in Figure 2.1 using a similar representation to (Andrieu et al., 2010). We show the particles sampled over time, their ancestor particles, and the fixed reference trajectory. We present the steps of the CSMC algorithm with resampling at each time step in Algorithm 6.

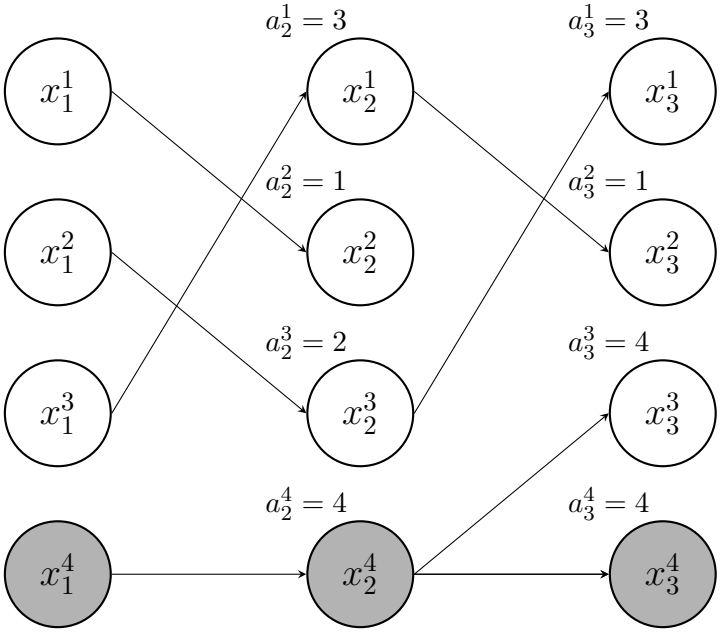


Figure 2.1: Example CSMC algorithm with $M = 4$ particles and $T = 3$ time points, with the fixed reference trajectory shown in grey. Each node represents a particle and the ancestor indices are annotated above the corresponding node.

Algorithm 6 Conditional sequential Monte Carlo (CSMC)

Input:

- M = number of particles
- $q(x_1|y_1, \theta)$, $\{q(x_t|y_t, x_{t-1}, \theta)\}_{t=2}^T$ = importance distributions
- θ = known model parameters
- $x_{1:T}^{(s-1)}$ = trajectory of latent states at MCMC iteration $s-1$

set $x_1^M = x_1^{(s-1)}$

for $m = 1, \ldots, M-1$ **do**

sample $x_1^m \sim q(x_1|y_1, \theta)$

calculate $w_1^{1:M}$ and $W_1^{1:M}$ ▷ Equation (2.5)

for $t = 2, \ldots, T$ **do**

set $x_t^M = x_t^{(s-1)}$, $a_t^M = M$

for $m = 1, \ldots, M-1$ **do**

sample $a_t^m \sim r(a_t|W_{1:t-1}^{1:M})$ ▷ Equation (2.6)

sample $x_t^m \sim q(x_t|y_t, x_{t-1}^{a_t^m}, \theta)$

set $x_{1:t}^m = (x_{1:t-1}^{a_t^m}, x_t^m)$

calculate $w_t^{1:M}$ and $W_{1:t}^{1:M}$ ▷ Equation (2.5)

return $\{x_{1:T}^m, W_{1:T}^m\}_{m=1}^M$

Once the CSMC recursions have been completed, MCMC proposed values for the latent states are sampled from the resulting approximation of $p(x_{1:T}|y_{1:T}, \theta)$, $\{x^m_{1:T}, W^m_{1:T}\}^M_{m=1}$. For example, from the trajectories in Figure 2.1, we sample an MCMC proposed trajectory of latent states, either (x^2_1, x^3_2, x^1_3), (x^3_1, x^1_2, x^2_3), (x^4_1, x^4_2, x^3_3), or (x^4_1, x^4_2, x^4_3), depending on their weights, $W^{1:4}_{1:3}$. The proposed values are always accepted, resulting in Gibbs steps. Finally, the model parameters are updated using standard and often low-dimensional M-H or Gibbs steps targeting $p(\theta|x_{1:T}, y_{1:T})$. This particle Gibbs algorithm results in MCMC samples converging to the joint distribution $p(x_{1:T}, \theta|y_{1:T})$ and is given in Algorithm 7.

Algorithm 7 Particle Gibbs

Input:

- M = number of particles
- S = number of MCMC iterations
- $q(x_1|y_1, \theta)$, $\{q(x_t|y_t, x_{t-1}, \theta)\}^T_{t=2}$ = importance distributions
- $x^{(0)}_{1:T}$, $\theta^{(0)}$ = initial values
- a Gibbs or Metropolis-Hastings sampling scheme to update θ from $p(\theta|x_{1:T}, y_{1:T})$

for $s = 1, \dots S$ **do**

run Algorithm 6 with $q(x_1|y_1, \theta)$, $\{q(x_t|y_t, x_{t-1}, \theta)\}^T_{t=2}$, $x^{(s-1)}_{1:T}$, and $\theta = \theta^{(s-1)}$

sample $x^{(s)}_{1:T}$ from $\{x^m_{1:T}, W^m_{1:T}\}^M_{m=1}$

update $\theta^{(s)}$ from $p(\theta|x^{(s)}_{1:T}, y_{1:T})$

return $\{x^{(s)}_{1:T}, \theta^{(s)}\}^S_{s=1}$ approximating $p(x_{1:T}, \theta|y_{1:T})$

Andrieu et al. (2010) and Chopin and Singh (2015) establish that the particle Gibbs state samples are distributed according to $p(x_{1:T}|y_{1:T}, \theta)$ upon convergence. The authors show that the algorithm samples from an extended target distribution that admits $p(x_{1:T}|y_{1:T}, \theta)$ as a marginal distribution due to a corrective unbiased estimate of the likelihood term. Thus, the particle Gibbs algorithm converges to $p(x_{1:T}|y_{1:T}, \theta)$ but latent state samples are also always 'accepted' in the MCMC steps.

2.3.1 Particle Gibbs with ancestor sampling

The mixing of the particle Gibbs algorithm can be poor when sample impoverishment occurs in the CSMC approximation (Chopin and Singh, 2015; Rainforth et al., 2016; Wigren et al., 2019). In severe cases of sample impoverishment, the reference trajectory is nearly always proposed (and accepted) in the particle Gibbs steps since it is fixed. The MCMC algorithm therefore remains at the

same values for the latent states for many iterations, leading to poor mixing. To improve the mixing of particle Gibbs methods, Lindsten et al. (2014) proposed the particle Gibbs with ancestor sampling (PGAS) algorithm, which uses CSMC with ancestor sampling (CSMC-AS) to artificially recompose the particle Gibbs reference trajectory, ensuring that unique values for the latent states are proposed at each MCMC iteration.

The CSMC-AS algorithm recomposes the reference trajectory at each CSMC forward recursion by artificially re-assigning its particle history. As such, unique latent states are proposed at each MCMC iteration. We re-assign the particle history by first noting that in the CSMC algorithm, the reference trajectory at each time $t = 2, \ldots, T$ is indexed by a_t^M. Thus, to recreate the history of the reference trajectory, the CSMC-AS algorithm samples new values for a_t^M at each time t. New values for a_t^M are sampled according to the probability that the associated trajectory generated the reference particle, denoted $\tilde{w}_t^m$ for time $t = 2, \ldots, T$, and given by

$$\tilde{w}_t^m \propto W_{1:t-1}^m p(x_t^{(s-1)} | x_{t-1}^m, \theta), \;\; m = 1, \ldots, M, \tag{2.11}$$

where $x_t^{(s-1)}$ denotes the reference particle (state sample at iteration $(s-1)$) at time t. The weights are normalised so that they sum to one, giving normalised weights $\tilde{W}_t^m = \tilde{w}_t^m / \sum_{k=1}^M \tilde{w}_t^k$ and the new ancestor is sampled using these weights, i.e., a_t^M is sampled from $\{m, \tilde{W}_t^m\}_{m=1}^M$ at each time t. Finally, the reference trajectory at time $t = 2, \ldots, T$ is recreated by attaching the current reference particle to its new likely history, $x_{1:t}^M = (x_{1:t-1}^{a_t^M}, x_t^{(s-1)})$. We illustrate the CSMC-AS particle dynamics in Figure 2.2, where the re-assigned reference trajectory histories are shown by dashed arrows. The other particle dynamics are identical to the pure CSMC illustration in Figure 2.1, allowing for direct comparison. We also summarise each step of the CSMC-AS algorithm in Algorithm 8.

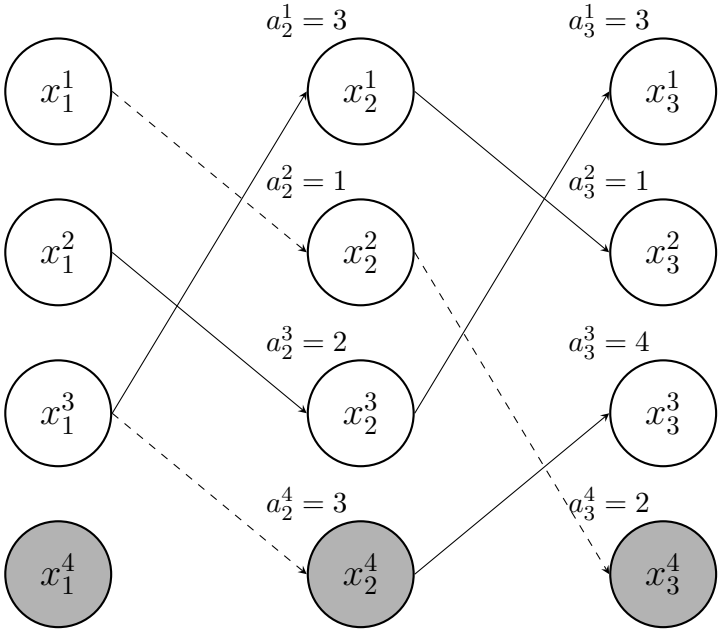


Figure 2.2: Illustration of a CSMC-AS algorithm with $M = 4$ particles and $T = 3$ time points. As in the illustration for the CSMC algorithm (Figure 2.1), the fixed reference trajectory is shown in grey, each node represents a particle and the ancestor indices are annotated above the corresponding node. However, the particle history of the reference trajectory is now re-assigned in ancestor sampling steps, shown by the dashed arrows.

Once the CSMC-AS recursions have been completed, the full PGAS algorithm samples new values for the latent states $x_{1:T}^{(s)}$ at iteration s, targeting $p(x_{1:T}|y_{1:T}, \theta)$ and the parameters are updated targeting $p(\theta|y_{1:T}, x_{1:T})$. The full PGAS algorithm is given in Algorithm 9.

PGAS methods are shown to improve upon the mixing of particle Gibbs algorithms both theoretically and in a wide range of examples (Berntorp and Di Cairano, 2017; Chopin and Singh, 2015; Nonejad, 2015; Wigren et al., 2019). However, PGAS methods can still be inefficient when there is a high rate of sample impoverishment in the SMC algorithm. As shown in Rainforth et al. (2016), if sample impoverishment is particularly prevalent, the pool of trajectories at each CSMC recursion may represent the posterior distribution poorly and the MCMC sampler may not explore the space sufficiently even if the history of the reference trajectory is recomposed in ancestor sampling steps.

Alternatively, the mixing of particle Gibbs methods can be improved by simulating particles in high posterior regions, tackling the initial sample impoverishment problem. Several approaches have been proposed, including the use of Gaussian approximations of the SSM (Andrieu et al., 2003), deterministic

Algorithm 8 Conditional sequential Monte Carlo with ancestor sampling (CSMC-AS)

Input:

- M = number of particles
- $q(x_1|y_1, \theta)$, $\{q(x_t|y_t, x_{t-1}, \theta)\}_{t=2}^{T}$ = importance distributions
- θ = known model parameters
- $x_{1:T}^{(s-1)}$ = trajectory of latent states at MCMC iteration $s-1$

set $x_1^M = x_1^{(s-1)}$

for $m = 1, \ldots, M-1$ **do**
  sample $x_1^m \sim q(x_1|y_1, \theta)$

calculate $w_1^{1:M}$ and $W_1^{1:M}$ ▷ Equation (2.5)

for $t = 2, \ldots, T$ **do**
  set $x_t^M = x_t^{(s-1)}$
  for $m = 1, \ldots, M-1$ **do**
    sample $a_t^m \sim r(a_t|W_{1:t-1}^{1:M})$ ▷ Equation (2.6)
    sample $x_t^m \sim q(x_t|y_t, x_{t-1}^{a_t^m}, \theta)$
  calculate $\tilde{W}_t^{1:M}$ ▷ ancestor sampling, Equation (2.11)
  sample a_t^M from $\{m, \tilde{W}_t^m\}_{m=1}^M$
  set $x_{1:t}^m = (x_{1:t-1}^{a_t^m}, x_t^m)$, $m = 1, \ldots, M$
  calculate $w_{1:t}^{1:M}$ and $W_{1:t}^{1:M}$ ▷ Equation (2.5)

return $\{x_{1:T}^m, W_{1:T}^m\}_{m=1}^M$

optimisation-based approximations in annealing schemes (Donnet and Robin, 2017), and variational approximations of the posterior (He et al., 2023). A popular approach is the auxiliary particle filter (Carpenter et al., 1999; Pitt and Shephard, 1999, 2001). However, additional simulation steps are often required within the auxiliary particle filter to direct particles to high posterior density areas (Elvira et al., 2018), thus the computational cost of such approaches can accumulate quickly when used within particle Gibbs algorithms.

Algorithm 9 Particle Gibbs with ancestor sampling (PGAS)

Input:

- M = number of particles
- S = number of MCMC iterations
- $q(x_1|y_1,\theta)$, $\{q(x_t|y_t,x_{t-1},\theta)\}_{t=2}^T$ = importance distributions
- $x_{1:T}^{(0)}$, $\theta^{(0)}$ = initial values
- a Gibbs or Metropolis-Hastings sampling scheme to update θ from $p(\theta|x_{1:T},y_{1:T})$

for $s = 1,\dots S$ **do**

run Algorithm 8 with $q(x_1|y_1,\theta)$, $\{q(x_t|y_t,x_{t-1},\theta)\}_{t=2}^T$, $x_{1:T}^{(s-1)}$, and $\theta = \theta^{(s-1)}$

sample $x_{1:T}^{(s)}$ from $\{x_{1:T}^m, W_{1:T}^m\}_{m=1}^M$

update $\theta^{(s)}$ from $p(\theta|x_{1:T}^{(s)},y_{1:T})$

return $\{x_{1:T}^{(s)},\theta^{(s)}\}_{s=1}^S$ approximating $p(x_{1:T},\theta|y_{1:T})$

2.4 Discussion on efficiency

Designing efficient methods for inference using SSMs can be challenging. Importance sampling-only approaches using SMC are flexible but can be computationally expensive when the model parameters are unknown and may suffer from sample impoverishment in the state dimension. On the other hand, MCMC approaches provide a natural conditional-updating mechanism for both the latent states and model parameters but standard proposal distributions for the latent states may lead to poor performance when there are many, highly correlated latent states. It is often possible to update the latent states in lower-dimensional blocks, requiring lower-dimensional and simpler proposal distributions for each block. However, block updating can still lead to poor mixing when the states are highly correlated and the multi-dimensional proposal distributions can still be challenging to design efficiently.

SMC and MCMC approaches can be combined to infer the latent states and model parameters efficiently. The particle Gibbs algorithm proposes values for the latent states that are always accepted in the MCMC steps and thus avoids poor mixing from low acceptance probabilities. Further, the PGAS algorithm can be used to partially circumvent the poor mixing exhibited when sample impoverishment occurs in the SMC steps. However, the PGAS algorithm can still require a large computational cost if sample impoverishment occurs extensively before the reference trajectory can be recreated. The design of efficient MCMC proposal distributions for the latent states of general SSMs remains an open challenge.

Chapter 3

Point mass proposal Metropolis-Hastings

3.1 Introduction

In this chapter, we address the challenge of designing efficient Markov chain Monte Carlo (MCMC) algorithms for state-space models (SSMs) discussed throughout Chapter 2 and propose a novel and efficient approach to Metropolis-within-Gibbs sampling (Section 2.2). In particular, we focus on the design of efficient updates of the latent states conditional on the model parameters as part of a Metropolis-within-Gibbs approach targeting the joint posterior distribution of the latent states and model parameters, $p(x_{1:T}, \theta|y_{1:T})$. The new approach designs efficient and general-use block Metropolis-Hastings (M-H) proposal distributions via approximate hidden Markov models (HMMs; Equation (1.2)), and is referred to as the *point mass proposal Metropolis-Hastings* (PMPMH) algorithm.

Proposed values for (blocked) M-H latent state updates are proposed in two steps. The first step discretises the state space into a set of pre-defined intervals, forming a grid. We then deterministically approximate an HMM in the grid cell indices, conditional on the current MCMC values for the model parameters and latent states, yielding a deterministic HMM approximation to the distribution of the latent states given the model parameters. The approximate discrete latent state distribution is now that of a tractable HMM. Therefore, to finish this first step, we propose a sequence of intervals from the discrete latent state distribution, applying the recursive forward-filtering backward-sampling algorithm for HMMs described in Rabiner and Juang (1986) and Algorithm 5. In the second step of the algorithm, we sample values for the latent states from within the proposed intervals using a continuous proposal distribution. Once the candidate values have

been proposed using the two-step procedure, we correct for the discretisation error imposed by the approximate HMM using an M-H acceptance step. The model parameters are then updated conditionally on the latent states using standard Gibbs or Metropolis-within-Gibbs updates.

The first step of the proposed algorithm is related to embedded HMMs and point mass filtering, which both use the tractable discrete case of an SSM, an HMM, to sample latent states. Embedded HMMs (Finke et al., 2016; Neal, 2003; Shestopaloff and Neal, 2013, 2018) construct an HMM in the indices of stochastically-generated 'pool' states, and sample states such that they target the correct posterior distribution. Here, we use the embedding of an HMM via deterministic grid cells (for fixed parameters and current latent states), binning areas of the state space and approximating an HMM to ensure that latent states are proposed with reasonable posterior mass. The binning approach and formulation of deterministic approximations to the HMM are also related to spatial aggregation in some animal movement models (Newman, 1998). Point mass filters, similarly to the proposed approach, use a deterministic grid to reduce the SSM to an HMM (Bucy and Senne, 1971; Kitagawa, 1987; Langrock et al., 2012; Langrock and King, 2013). The HMM can be used to approximate the likelihood or posterior distribution of the model parameters directly. As such, point mass filters can be (relatively) computationally inexpensive to implement, but generally produce biased estimators. Further, the bias and variance of estimators often depend on the grid (for example, the number and location of the grid cells and their coverage of the state space), and the integration method used to approximate the HMM (de Valpine and Hastings, 2002; Matoušek et al., 2020). Instead, we use the grid cells simply to inform an M-H proposal distribution and are thus able to define the HMM and grid cells coarsely (and inexpensively), while being able to achieve convergence to the correct posterior distribution, correcting for the bias introduced by the discretisation.

The rest of the chapter is structured as follows. We start by describing the proposed HMM approximation strategy in Section 3.2, the PMPMH algorithm, in which we use a grid-based HMM approximation within a proposal distribution for the latent states. In Section 3.3, we discuss practical implementation and provide three particular cases of the algorithm, each given by a different method for defining the grid cells, and describe when each can be usefully applied. Further, we discuss, in detail, several other tuning parameters that can be used to improve the performance of the algorithm. Such tuning parameters include the block size for the latent states (i.e. the number of latent states to update simultaneously), the number of intervals used in the HMM approximation, and the distribution

of the intervals in the state space. Finally, we illustrate our approach with two case studies in Section 3.4, including a near-chaotic system, where our proposed approach substantially improves upon the performance of alternative approaches. We conclude in Section 3.5 with a discussion of our proposed algorithm and further possible avenues for research.

3.2 Point mass proposal Metropolis-Hastings

We describe the proposed *point mass proposal Metropolis-Hastings* (PMPMH) algorithm. The setting is a Metropolis-within-Gibbs algorithm, described in Section 2.2.1, targeting the joint posterior distribution, $p(x_{1:T}, \theta|y_{1:T})$. The model parameters, θ, are updated conditional on $x_{1:T}$ using a standard M-H or Gibbs update. The latent states, $x_{1:T}$, are then updated conditional on the parameters, θ, using the proposed PMPMH algorithm.

To update the latent states conditional on the model parameters, Step 1 of the PMPMH algorithm initially converts the SSM into an HMM of the form Equation (1.2) by discretising the state space at each time point, forming a grid, and the HMM is approximated. Grid cells are then sampled from the associated (discretised and approximate) posterior distribution, conditional on the model parameters. For this, we employ the standard forward-filtering backward-sampling algorithm for HMMs at a computational cost of $\mathcal{O}(NT)$ for N grid cells. Given the sampled grid cells, Step 2 proposes values for the latent states from within the cells using some specified (bounded) continuous proposal distribution. We then correct for the approximate sampling distribution using an M-H acceptance step targeting the correct posterior distribution.

We derive the algorithm for one-dimensional state spaces. Extensions to higher-dimensional state spaces are possible; we demonstrate the algorithm in the two-dimensional case with an example in Section 3.4 and discuss scalability further in Section 3.5.

3.2.1 Step 1: sampling a grid cell trajectory

In this initial step, we formulate a discrete HMM representation of the SSM, approximate the HMM, and sample a trajectory from the discrete approximation. First, we propose a partition of the whole state space into a grid; the state space at each time t, χ, is partitioned into intervals, forming grid cells. The N grid cells at time t are denoted $I_t(n)$, $n = 1, \ldots, N$ and cover χ with no overlap. The respective outer grid cells have infinite length if the state space is

unbounded. For example, if the state space at each time point is one-dimensional with range $(-\infty, \infty)$, three grid cells may partition the space at time t into the intervals $\{I_1(1), I_1(2), I_1(3)\} = \{(-\infty, -2), [-2, 2], (2, \infty)\}$. We show a further example partition of the state space for some time t in Figure 3.1. See also further discussion and examples of defining the grid cells in Section 3.3.1.

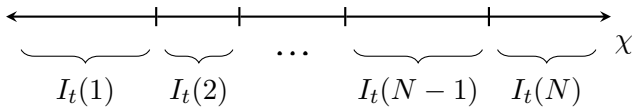


Figure 3.1: Example partition of the state space, χ, into N intervals at time t.

The partition could be applied to the state space at all time points, forming a grid. For example, partitioning the state space at time $t = 1$ as in Figure 3.1 and defining further partitions of the state space at all other time points results in a grid similar to Figure 3.2.

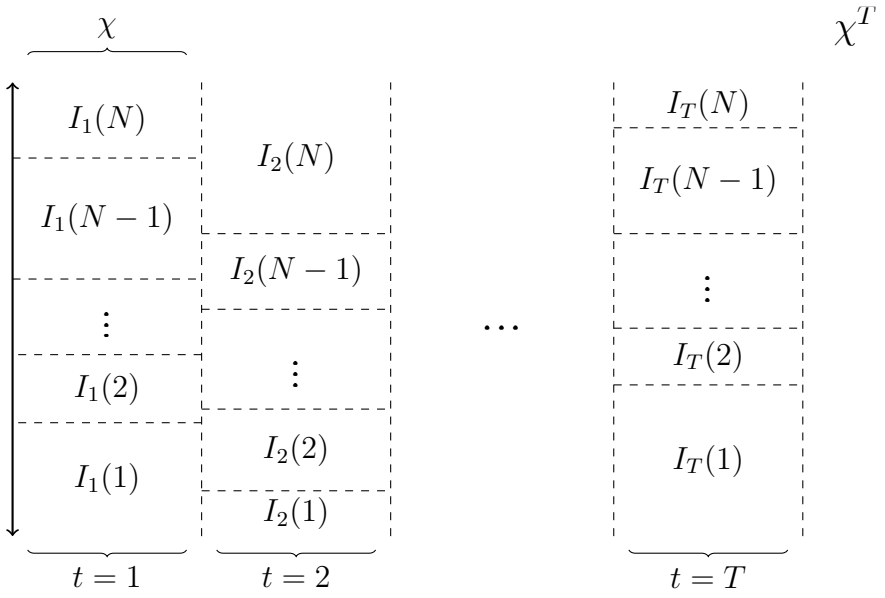


Figure 3.2: Example partition of a state space, χ^T, for all time points, forming a grid. Each grid cell is labelled by the corresponding interval.

The grid cells discretise the state space. Thus, the grid cell indices are the discrete states of an HMM. For example, for the intervals $\{I_t(1), \ldots I_t(N)\}$, the indices, $\{1, \ldots N\}$, are the discrete HMM states. To define the dynamics of the HMM, we first let B_t denote the random variable of the grid cell indices under the specified grid cell boundaries so that $B_t \in \{1, \ldots, N\}$ for $t = 1, \ldots, T$. Then, the dynamics

of the HMM are defined in terms of the SSM densities in each grid cell (Kitagawa, 1987; Langrock, 2011). Mathematically, the HMM probabilities are given by

$$\begin{aligned}
&\textit{Initial state probabilities}\text{: } P(B_1 = n|\theta),\ n = 1, \ldots, N,\\
&\textit{State transition probabilities}\text{: } P(B_t = n|B_{t-1} = k, \theta),\ k, n = 1, \ldots, N, t = 2, \ldots, T,\\
&\textit{Observed state distribution}\text{: } p(y_t|B_t = n, \theta),\ n = 1, \ldots, N, t = 1, \ldots, T.
\end{aligned}$$

The HMM state transition probabilities at time t over the combinations of discrete states form a transition probability matrix (TPM). The HMM probabilities under the original SSM are defined by integrals of the state space equations over the given intervals, for states $k, n \in \{1, \ldots, N\}$:

Initial state probabilities:

$$P(B_1 = n|\theta) = \int_{I_1(n)} p(x_1|\theta)dx_1,$$

State transition probabilities:

$$P(B_t = n|B_{t-1} = k, \theta) = \int_{I_t(n)} \int_{I_{t-1}(k)} p(x_t|x_{t-1}, \theta)dx_{t-1}dx_t,\ t = 2, \ldots, T,$$

Observed state distribution:

$$p(y_t|B_t = n, \theta) = \int_{I_t(n)} p(y_t|x_t, \theta)dx_t,\ t = 1, \ldots, T. \tag{3.1}$$

In general, the probabilities associated with this HMM are not available in closed form, thus we apply integral approximations. To formulate an approximation under the SSM, we use simple (fast) deterministic methods, for example, the mid-point rule or other Riemann sum methods. More complex integral approximation methods can be used and are discussed in Matoušek et al. (2023).

To approximate Equation (3.1) via Riemann sum methods, we define $L_t(n)$ as the length of the interval $I_t(n)$, with the length of any infinite grid cells set at some arbitrary (finite) length (discussed in Section 3.3.2). Within each grid cell, we choose a set of node points, $\{\xi_t^d(n)\}_{d=1}^D$. In the simple case where $D = 1$ (one node in each grid cell such as the mid-point of finite cells), we have that $\{\xi_t^d(n)\}_{d=1}^D = \xi_t(n)$. The approximation to the HMM (Equation (3.1)) is given for grid cell indices $k, n = 1, \ldots, N$ by

$$\begin{aligned}
\hat{P}(B_1 = n|\theta) &\propto L_1(n)p(\xi_1(n)|\theta),\\
\hat{P}(B_t = n|B_{t-1} = k, \theta) &\propto L_t(n)L_{t-1}(k)p(\xi_t(n)|\xi_{t-1}(k), \theta),\ t = 2, \ldots, T,\\
\hat{p}(y_t|B_t = n, \theta) &\propto L_t(n)p(y_t|\xi_t(n), \theta),\ t = 1, \ldots, T.
\end{aligned} \tag{3.2}$$

Thus, the probability of moving from grid cell k to grid cell n at time t is approximately proportional to the product of the lengths of both intervals and the conditional density function, reflecting transition from $\xi_{t-1}(k)$ to $\xi_t(n)$. Each approximate probability is normalised so that the probabilities sum to one and are thus valid probability mass functions. Note that these calculations are related to spatial aggregation in some animal movement models and the mid-point approximation error depends on the uniformity of the densities within each grid cell (Newman, 1998). Once the probability mass function approximations have been obtained, grid cell indices are proposed from

$$\hat{P}(B_{1:T}|y_{1:T},\theta) \propto \hat{P}(B_1|\theta)\hat{p}(y_1|B_1,\theta)\prod_{t=2}^{T}\hat{P}(B_t|B_{t-1},\theta)\hat{p}(y_t|B_t,\theta).$$

We sample indices from this distribution, denoted $b'_{1:T}$, using the forward-filtering backward-sampling method for HMMs in Algorithm 5.

3.2.2 Step 2: sampling a point trajectory

Step 2 proposes continuous values for the latent states which will be subsequently corrected in the M-H step. Given the discrete grid cell indices sampled previously, we propose values for $x_{1:T}$ conditional on the corresponding intervals $\{I_1(b'_1),\ldots,I_T(b'_T)\}$. This amounts to sampling $x_{1:T}$ from the domain $I_1(b'_1)\times I_2(b'_2)\times\cdots\times I_T(b'_T)$. For simplicity, we consider proposal distributions for each x_t independently of the other states and θ, giving proposal distributions of the form $q(x_t|x_t\in I_t(b'_t)) = q(x_t|B_t = b'_t)$, for $t = 1,\ldots,T$. Although any distribution with domain $I_t(b'_t)$ is applicable, we opt for standard distributions, proposing each x_t from a uniform proposal distribution over $I_t(b'_t)$ for finite $I_t(b'_t)$, or a truncated Gaussian distribution if $I_t(b'_t)$ has infinite length.

The PMPMH proposal distribution

The process of sampling a grid cell trajectory and sampling a value for $x_{1:T}$ defines an independent proposal distribution on χ^T. The density function of the proposed trajectories is therefore the probability of selecting the grid cell indices from a specified set of grid cell boundaries, combined with the density of the (continuous) values within those grid cells. Let $B_{1:T}$ denote the random variables of the grid cell indices under the proposed grid with intervals $I_t(n),\ \ n = 1,\ldots,N,\ \ t = 1,\ldots,T$.

Then the density of the subsequently proposed state values, $x'_{1:T}$, is given by

$$q(x'_{1:T}|y_{1:T},\theta) = \hat{P}(B_{1:T} = b'_{1:T}|y_{1:T},\theta)\prod_{t=1}^{T} q(x'_t|B_t = b'_t), \tag{3.3}$$

i.e., the product of the probability of the grid cells under the HMM approximation and the probability of selecting the points within each grid cell. The resulting proposed trajectory, $x'_{1:T}$, is retained according to the M-H acceptance probability (see Section 2.2.1), given by

$$\alpha(x_{1:T}, x'_{1:T}|\theta) = \min\left(1, \frac{p(x'_{1:T}|y_{1:T},\theta)q(x_{1:T}|y_{1:T},\theta)}{p(x_{1:T}|y_{1:T},\theta)q(x'_{1:T}|y_{1:T},\theta)}\right), \tag{3.4}$$

where $q(x'_{1:T}|y_{1:T},\theta)$ is given in Equation (3.3) and $q(x_{1:T}|y_{1:T},\theta)$ uses the same definition for the grid cells, that is, their indices, $b_{1:T}$, are such that $x_t \in I(b_t)$ for all t. In addition, $p(x_{1:T}|y_{1:T},\theta)$ denotes the posterior conditional distribution of $x_{1:T}$ given $y_{1:T}$ and θ, which can be evaluated up to proportionality in the acceptance probability, i.e., we evaluate the right-hand side of

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

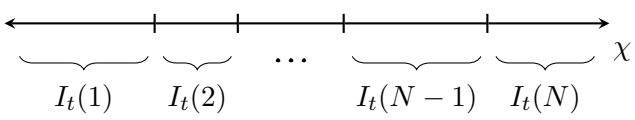
Once the latent states have been updated, the model parameters θ can be updated using standard M-H or Gibbs updates conditional on the current chain value of the states.

3.2.3 Summary

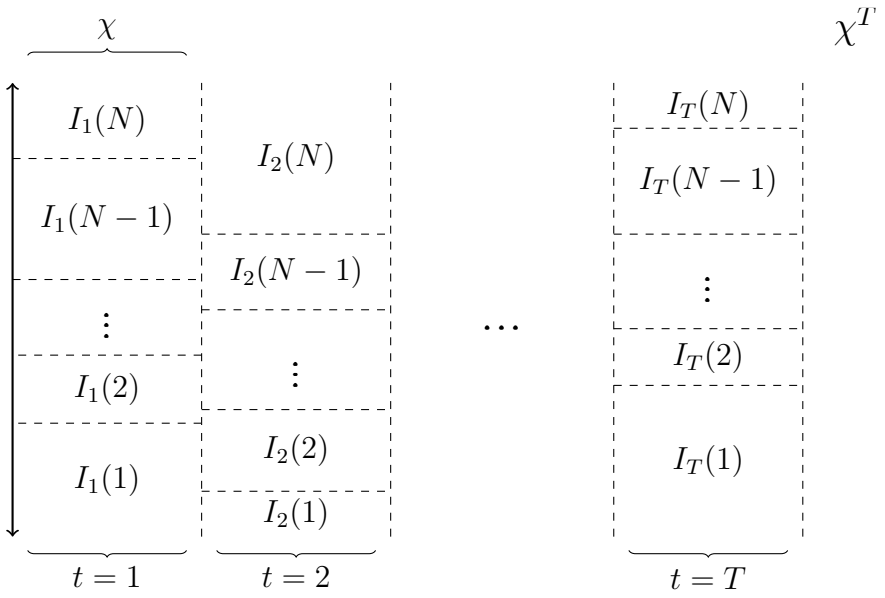
We first present a largely pictorial version of the PMPMH algorithm with a one-dimensional state space to summarise each step and aid implementation. We then present pseudocode in Algorithm 10. This supplements the full descriptions in Sections 3.2.1 and 3.2.2.

Step 1(a): discretise the state space

Partition the state space, χ, into N intervals at time point t, for example:

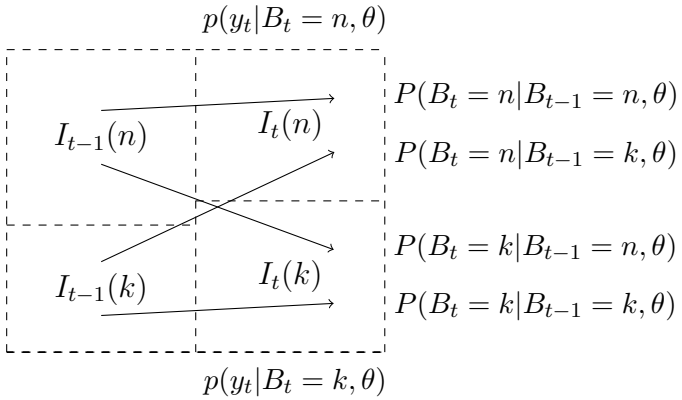


Form a grid by partitioning the state space at all time points:



Step 1(b): approximate an HMM in the grid cell indices

The dynamics of the HMM in the grid cell index random variables, $B_t \in \{1, \ldots, N\}$ at time t, are defined in terms of the underlying SSM. For $n, k \in \{1, \ldots, N\}$:



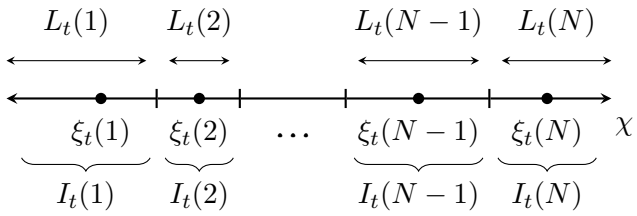
These HMM probabilities are integrals of the SSM equations over the given intervals:

$$P(B_1 = n|\theta) = \int_{I_1(n)} p(x_1|\theta)dx_1,$$

$$P(B_t = n|B_{t-1} = k, \theta) = \int_{I_t(n)} \int_{I_{t-1}(k)} p(x_t|x_{t-1}, \theta)dx_{t-1}dx_t, \;\; t = 2, \ldots, T,$$

$$p(y_t|B_t = n, \theta) = \int_{I_t(n)} p(y_t|x_t, \theta)dx_t, \;\; t = 1, \ldots, T.$$

To approximate these integrals using the mid-point rule, at time t, define the length and node point of each grid cell, $L_t(1), \ldots, L_t(N)$ and $\xi_t(1), \ldots, \xi_t(N)$:



For such node points and grid cell lengths, the approximate HMM is given by

$$\begin{aligned}
\hat{P}(B_1 = n|\theta) &\propto L_1(n)p(\xi_1(n)|\theta), \\
\hat{P}(B_t = n|B_{t-1} = k, \theta) &\propto L_t(n)L_{t-1}(k)p(\xi_t(n)|\xi_{t-1}(k), \theta), \;\; t = 2, \ldots, T, \\
\hat{p}(y_t|B_t = n, \theta) &\propto L_t(n)p(y_t|\xi_t(n), \theta), \;\; t = 1, \ldots, T.
\end{aligned}$$

Normalise these probabilities so that they sum to one, then impose a lower bound, for example, 0.01, and re-normalise to ensure that all grid cells have reasonable probability mass (and can be proposed later).

Step 1(c): sample a grid cell trajectory

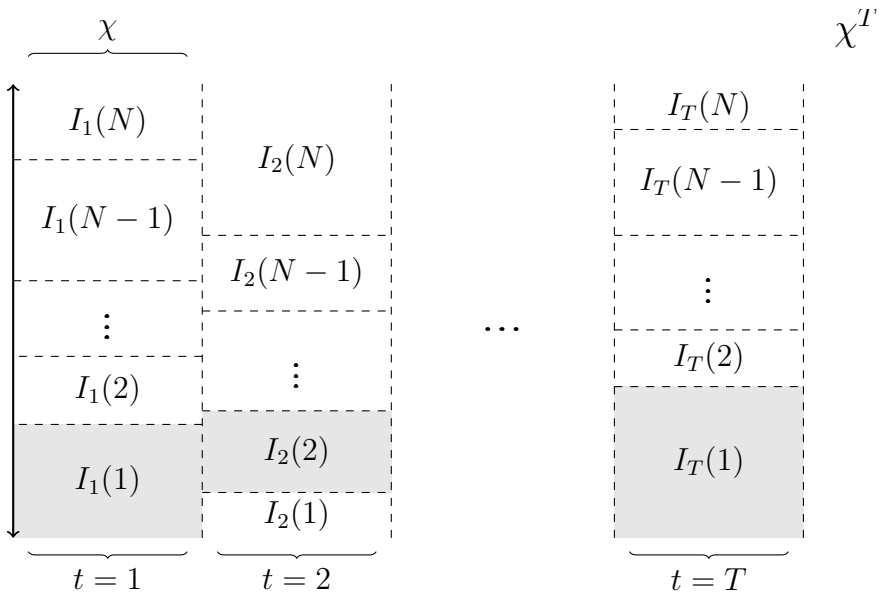
Sample grid cell indices from $\hat{P}(B_{1:T}|y_{1:T}, \theta)$ using forward-filtering backward-sampling (Algorithm 5). That is, first find $\hat{P}(B_1|y_1, \theta) \propto P(B_1|\theta)p(y_1|B_1, \theta)$, then for $t = 2, \ldots, T$,

$$P(B_t \mid y_{1:t}, \theta) \propto p(y_t|B_t, \theta) \sum_{j=1}^{N} P(B_{t-1} = j|y_{1:t-1}, \theta)P(B_t|B_{t-1} = j, \theta).$$

Sample b'_T from $P(B_T|y_{1:T}, \theta)$. Then, sample each b'_t, $t = 1, \ldots, T-1$ backward recursively using

$$P(B_t \mid B_{t+1} = b'_{t+1}, y_{1:t}, \theta) \propto P(B_t \mid y_{1:t}, \theta)P(B_{t+1} = b'_{t+1}|B_t, \theta).$$

The sampled indices, $b'_{1:T}$, represent a grid cell trajectory. For example, sampled indices $b'_{1:T} = \{1, 2, \ldots, 1\}$ may correspond to the following grey regions:



Step 2(a): sample a point trajectory

Sample $x'_{1:T}$ from independent proposal distributions defined in each sampled grid cell, i.e., of the form $q(x_t|B_t = b'_t)$. The sampled $x'_{1:T}$, may correspond to:

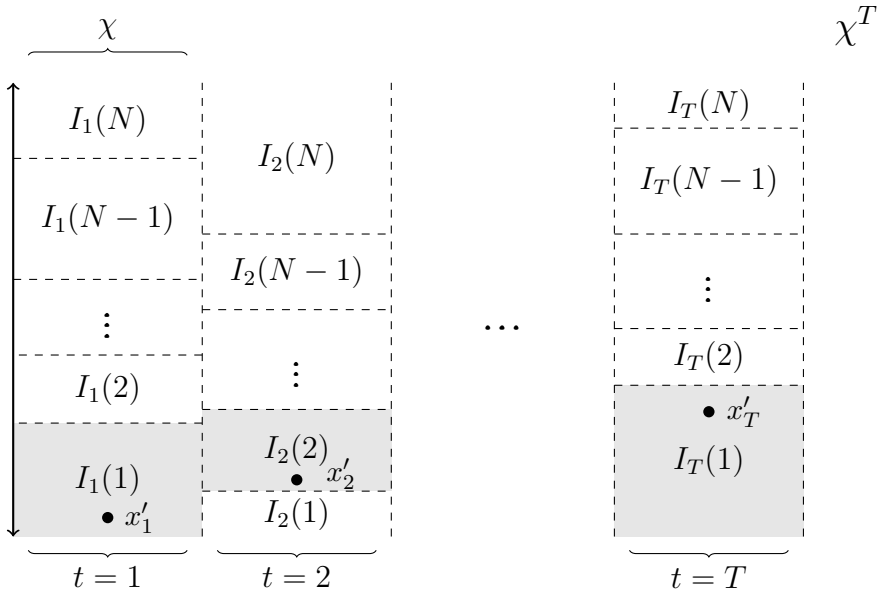


Figure 3.3: Illustration of PMPMH proposed state values. The sampled grid cells are shown in grey. The nodes represent points proposed within each grid cell.

For example, $x_1 \sim \text{TN}_{I_1(1)}(\xi_1(1), \sigma^2)$ (a truncated Gaussian distribution over $I_1(1)$ with mean $\xi_1(1)$ and variance σ^2), $x_2 \sim \text{Uniform}(I_2(2))$, and $x_T \sim \text{TN}_{I_T(1)}(\xi_T(1), \sigma^2)$.

Step 2(b): Metropolis-Hastings step
Accept or reject the sampled $x'_{1:T}$ with probability

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

where

$$q(x'_{1:T}|y_{1:T}, \theta) = \hat{P}(B_{1:T} = b'_{1:T}|y_{1:T}, \theta) \prod_{t=1}^{T} q(x'_t|B_t = b'_t).$$

We can then update the model parameters, θ, conditionally using standard M-H or Gibbs updates.

The PMPMH method for obtaining a sample of $x_{1:T}$ conditional on θ is given in full in Algorithm 10. For completeness, we also include the updates of θ conditional on $x_{1:T}$.

Algorithm 10 Point mass proposal Metropolis-Hastings (PMPMH)

Input:

- N = number of grid cells
- S = number of MCMC iterations
- $x_{1:T}^{(0)}$, $\theta^{(0)}$ = initial values
- a Gibbs or Metropolis-Hastings sampling scheme to update θ from $p(\theta|x_{1:T}, y_{1:T})$

for $s = 1, \dots S$ **do**

Sample a grid cell trajectory (Step 1)

define a grid with indices $B_{1:T}$ ▷ Step 1(a)

for $t = 1, \dots, T$ **do**

for $k = 1, \dots, N$ **do**

calculate $\hat{P}(B_t = n|y_{1:t}, \theta^{(s-1)})$ ▷ Step 1(b), Step 1(c)

sample b'_T from $\{n, \hat{P}(B_T = n|y_{1:T}, \theta^{(s-1)})\}_{n=1}^N$

for $t = T-1, \dots, 1$ **do**

sample b'_t from $\{n, \hat{P}(B_t = n|B_{t+1} = b'_{t+1}, y_{1:t}, \theta^{(s-1)})\}_{n=1}^N$

Sample a point trajectory (Step 2)

for $t = 1, \dots, T$ **do**

sample x'_t from $q(x_t|B_t = b'_t)$ ▷ Step 2(a)

M-H update

calculate $\alpha(x_{1:T}^{(s-1)}, x'_{1:T}|\theta^{(s-1)})$ ▷ Equation (3.4), Step 2(b)

sample $u \sim U(0, 1)$

if $u \leq \alpha(x_{1:T}^{(s-1)}, x'_{1:T}|\theta^{(s-1)})$ **then**

$x_{1:T}^{(s)} \leftarrow x'_{1:T}$

else $x_{1:T}^{(s)} \leftarrow x_{1:T}^{(s-1)}$

update $\theta^{(s)}$ from $p(\theta|x_{1:T}^{(s)}, y_{1:T})$

return $\{x_{1:T}^{(s)}, \theta^{(s)}\}_{s=1}^S$ approximating $p(x_{1:T}, \theta|y_{1:T})$

3.2.4 Block updates of the latent states

Updating all of the states simultaneously may be inefficient if there are many latent states. We can, instead, update the states in smaller blocks: for a block of ℓ states starting at time t, we sample $x_{t:t+\ell-1}$ from a proposal distribution of the form $q(x_{t:t+\ell-1}|x_{t-1}, x_{t+\ell}, y_{t:t+\ell-1}, \theta)$.

A block PMPMH proposal distribution is given by a simple adaptation of the case where all states are updated simultaneously, conditioning on the current

states at either side of the block; x_{t-1} and $x_{t+\ell}$. Therefore, in Step 1, we condition on the current grid cell indices, b_{t-1} and $b_{t+\ell}$, such that $x_{t-1} \in I(b_{t-1})$ and $x_{t+\ell} \in I(b_{t+\ell})$ under the current grid definition. Note that the exact values for the current states at either side of the block can be used in the HMM calculations but this approach requires additional TPM calculations when the states are updated.

We use forward-filtering backward-sampling to sample proposed indices from the grid cells in the block. In Step 2, we simply define the proposal distribution, as before, for each latent state in the given block, $q(x_i|B_i = b_i')$, for $i = t, \ldots, t + \ell - 1$. To complete the PMPMH step, we define the acceptance probability for a proposed set of ℓ states starting at time t, $x'_{t:t+\ell-1}$, by

$$\alpha(x_{t:t+\ell-1}, x'_{t:t+\ell-1}|\theta) = \min\left(1, \frac{p(x'_{t:t+\ell-1}|x_{t-1}, x_{t+\ell}, y_{t:t+\ell-1}, \theta)}{p(x_{t:t+\ell-1}|x_{t-1}, x_{t+\ell}, y_{t:t+\ell-1}, \theta)} \frac{q(x_{t:t+\ell-1}|x_{t-1}, x_{t+\ell}, y_{t:t+\ell-1}, \theta)}{q(x'_{t:t+\ell-1}|x_{t-1}, x_{t+\ell}, y_{t:t+\ell-1}, \theta)}\right), \tag{3.5}$$

where in the first block:

$$p(x_{1:\ell}|x_0, x_{\ell+1}, y_{1:\ell}, \theta) = p(x_{1:\ell}|x_{\ell+1}, y_{1:\ell}, \theta),$$

and similarly for the proposal density. In the last block:

$$p(x_{T-\ell+1:T}|x_{T-\ell}, x_{T+1}, y_{T-\ell+1:T}, \theta) = p(x_{T-\ell+1:T}|x_{t-\ell}, y_{T-\ell+1:T}, \theta).$$

The PMPMH algorithm for block updates is summarised in Algorithm 11. It is possible to parallelise the updates of the latent states to improve computational efficiency. For example, blocks of states can be updated in parallel provided that the states to be updated and those being conditioned on do not overlap. For the sake of comparison with other methods, we do not parallelise our computation but simply note its possibility. See also King et al. (2023) for further discussion on parallelisation within an SSM-fitting context.

Algorithm 11 Block point mass proposal Metropolis-Hastings (block PMPMH)

Input:

- N = number of grid cells
- S = number of MCMC iterations
- $x_{1:T}^{(0)}$, $\theta^{(0)}$ = initial values
- a Gibbs or Metropolis-Hastings sampling scheme to update θ
- blocks of size ℓ with starting points for each block, $\{1, \ldots, D\} \subset \{1, \ldots, T\}$

For $t = 1, \ldots, \ell$, (block $d = 1$), define

$$\hat{P}(B_t = k | B_0 = b_0, y_{1:t}, \theta) = \hat{P}(B_t = k | y_{1:t}, \theta),$$

$$\hat{P}(B_t = k | B_0 = b_0, B_{t+1} = b_{t+1}, y_{1:\ell}, \theta) = \hat{P}(B_t = k | B_{t+1} = b_{t+1}, y_{1:\ell}, \theta),$$

and define

$$\hat{P}(B_T = k | B_{D-1} = b_{D-1}, B_{T+1} = b_{T+1}, y_{D:T}, \theta) = \hat{P}(B_T = k | B_{D-1} = b_{D-1}, y_{D:t}, \theta).$$

for $s = 1, \ldots S$ **do**

Sample a grid cell trajectory (Step 1)

for $d = 1, \ldots, D$ **do**

define grid cells with indices $B_{\max(1,d-1):\min(d+\ell,T)}$ ▷ Step 1(a)

for $t = d, \ldots d + \ell - 1$ **do**

for $k = 1, \ldots, N$ **do**

calculate $\hat{P}(B_t = k | B_{d-1} = b_{d-1}, y_{d:t}, \theta^{(s-1)})$ ▷ Step 1(b), Step 1(c)

sample $b'_{d+\ell-1}$ from

$\{n, \hat{P}(B_{d+\ell-1} = n | B_{d+\ell} = b_{d+\ell}, B_{d-1} = b_{d-1}, y_{d:d+\ell-1}, \theta^{(s-1)})\}_{n=1}^{N}$

for $t = d + \ell - 2, \ldots, d$ **do**

sample b'_t from

$\{n, \hat{P}(B_t = n | B_{t+1} = b'_{t+1}, B_{d-1} = b_{d-1}, y_{d:t}, \theta^{(s-1)})\}_{n=1}^{N}$

Sample a point trajectory (Step 2)

for $t = d, \ldots, d + \ell - 1$ **do**

sample x'_t from $q(x_t | B_t = b'_t)$ ▷ Step 2(a)

M-H update

calculate $\alpha(x_{d:d+\ell-1}^{(s-1)}, x'_{d:d+\ell-1} | \theta^{(s-1)})$ ▷ Equation (3.5), Step 2(b)

sample $u \sim U(0, 1)$

if $u \leq \alpha(x_{d:d+\ell-1}^{(s-1)}, x'_{d:d+\ell-1} | \theta^{(s-1)})$ **then**

$x_{d:d+\ell-1}^{(s)} \leftarrow x'_{d:d+\ell-1}$

else $x_{d:d+\ell-1}^{(s)} \leftarrow x_{d:d+\ell-1}^{(s-1)}$

update $\theta^{(s)}$ from $p(\theta | x_{1:T}^{(s)}, y_{1:T})$

return $\{x_{1:T}^{(s)}, \theta^{(s)}\}_{s=1}^{S}$ approximating $p(x_{1:T}, \theta | y_{1:T})$

3.2.5 Conditions for convergence

A benefit of the proposed method is that estimators of integrable functions are consistent under some mild conditions ensuring the validity of the proposal distribution in the M-H steps. We prove the validity of the proposal distribution under these conditions by showing that the constructed Markov chains are irreducible and the detailed balance condition is satisfied.

Theorem *The PMPMH algorithm provides consistent ergodic average estimators of integrable functions with respect to* $p(x_{1:T}, \theta|y_{1:T})$ *if,*

1. *for all* $b_t \in \{1, \ldots, N\}$, $t \in \{1, \ldots, T\}$, *the HMM probabilities are defined such that* $\hat{P}(B_{1:T} = b_{1:T}|y_{1:T}, \theta) > 0$ *and the proposal distributions within each grid cell are defined such that* $q(x_t|B_t = b_t) > 0$ *for all* $x_t \in I_t(b_t)$, *and*
2. *(a) the grid cells are defined and fixed at the start of the algorithm or*
   (b) the grid cells are defined as a function of the current MCMC iteration parameter or state values.

Proof. We assume that the transition kernel on the parameter space is constructed such that it admits $p(\theta|x_{1:T}, y_{1:T})$ as its limiting distribution. By standard results from Metropolis-within-Gibbs algorithms (Gamerman and Lopes, 2006; Latuszyński et al., 2013), proving that the PMPMH algorithm provides consistent estimators of integrable functions reduces to ensuring that the transition kernel on the blocks of ℓ states converge to the correct conditional posterior distribution. We therefore show that irreducibility and detailed balance are satisfied by the proposal distributions with respect to the correct conditional posterior distribution for each block of ℓ states.

By construction, we have that $\cup_{n=1}^{N} I_t(n) = \chi$ for all $t = 1, \ldots, T$. Condition 1 implies that $q(x_{1:T}|y_{1:T}, \theta) > 0$ for all $x_{1:T} \in \chi^T$, thus irreducibility is satisfied since any value in the state space can be proposed at each iteration of the algorithm. Note that Condition 1 can be ensured by thresholding the HMM probabilities above zero and proposing values via many standard distributions that allow any value within each grid cell to be proposed.

It is immediate that detailed balance holds under Condition 2(a) if the grid cells are defined and fixed at the start of the M-H algorithm. It is also straightforward to show that detailed balance holds under Condition 2(b) if the grid cells are defined as a function of the current MCMC iteration parameter or state values.

Thus, Condition 1 and Condition 2(a) or 2(b) ensure that irreducibility and detailed balance are satisfied by the PMPMH proposal distribution, and the algo-

rithm provides consistent ergodic average estimators of integrable functions with respect to $p(x_{1:T}, \theta | y_{1:T})$. □

3.3 Practical considerations

We have, so far, given a broad framework for using an approximate HMM as an M-H proposal distribution. There are, however, many different ways to define the grid cells, the deterministic integration method and the within-cell proposal distributions, possibly resulting in substantially different proposal distributions and computational costs. In this section, we discuss suggested practical choices and their efficiency, including three approaches to defining the grid cells.

3.3.1 Defining the grid cells

The method fundamentally relies on the use of a grid to discretise the state space, that is deterministic given the current MCMC values for the model parameters and latent states. We will see in Section 3.4 that the efficiency of the algorithm is highly dependent on the choice of grid cell definition (the size and location of grid cells) and resulting proposal distribution. Since the optimal choice depends on the application, we provide the reader with three approaches to defining the grid cells and describe when each can be usefully and efficiently applied.

Approach 1: equal grid cells

When the state space is infinite at each time point, we define outer, infinite grid cells and partition the remaining space into finite grid cells. For example, if $\chi = (-\infty, \infty)$, we may define infinite grid cells over the intervals $(-\infty, L)$ and (U, ∞). We then define $N - 2$ *finite* grid cells over the remaining interval $[L, U]$.

For a fixed number of grid cells, N, this approach sets all finite cells equally sized and the same across all time points. That is, $I_t(n) = I_i(n)$ for all $t, i = 1, \ldots, T$ and $n = 2, \ldots, N-1$ (assuming an infinite lower and upper bound on the latent states). We assume that these equally sized grid cells are centred around the mean of the data, $\frac{1}{T}\sum_{t=1}^{T} y_t$, over a range of the state space denoted by $\mathcal{S}$ (for example $\mathcal{S} = [L, U]$), though this approach could be adapted by centring the grid cells around another function of the data.

If the model parameters do not vary with time, only one TPM needs to be calculated per MCMC iteration when the model parameters are updated. For a fixed number of grid cells, this approach has the lowest computational cost of the approaches considered. However, since the grid cells are the same for each time

point but the regions of high posterior density may change at each time point, the grid cells should cover a large range of the state space to ensure coverage of these high-density regions and good mixing. This may require many grid cells and a greater computational cost if the high posterior regions of the SSM are highly non-uniform over time, and hence this approach is most efficient for SSMs with uniform high posterior regions over time.

Approach 2: data-driven quantiles (conditioning on $y_{1:T}$)

If the high posterior density regions vary over time, a time-inhomogeneous approximation to the TPM may be more efficient than the previous approach. Hence, we set grid cells in this approach using the observed data at each time point. In particular, in the implementations of this thesis, we set the grid cell boundaries at the quantiles of one-dimensional Gaussian distributions centred around each observed data point. For each time point, t, we set the boundaries at the quantiles of X where $X \sim N(y_t, {\sigma_y}^2)$. The variance of the Gaussian distribution, ${\sigma_y}^2$, is set at a scalar value, or as a function of the current observation process variance within the MCMC iterations and can be used to control the range that the majority of the grid cells cover via pilot tuning. The resulting quantiles are simply rounded if integer values are required. Note that simple extensions to this approach may include using a different distribution to set the quantiles, provided that its domain is in the state space.

When the model parameters are updated, a TPM needs to be recalculated for each time point. It is, however, possible to reduce the computational cost of this approach by using the same TPM across several time points, centring, for example, on a single data point or the mean of the corresponding sub-series. By using the observed data at each time point, we aim to approximately concentrate the grid cells in areas that are likely to have high posterior mass at each time point. This method may be most efficient if the observed data used in the centring is a good proxy for the underlying state process and its dependencies.

Approach 3: latent state quantiles (conditioning on $x_{1:T}$)

The previous approaches define the grid cells independently of the state process, which may result in slow mixing if the latent states of the SSM are highly correlated over time. The accuracy of the proposal distribution and mixing of the latent states can be improved through the definition of the grid cells. Matoušek et al. (2020) show empirically that the accuracy of deterministic grid cell approximations to the posterior distribution can be improved by placing the grid

cells according to rough approximations to the posterior distribution. Thus, here we set the grid cells using the current latent states in the MCMC iterations at each time t. The underlying intuition is that, after the MCMC algorithm has converged, the current states reflect a sample from the posterior distribution. Thus, over several iterations, grid cells centred around the current state lead to proposed values that are distributed similarly to the (conditional) posterior distribution (Haario et al., 1999), improving the accuracy of the proposal distribution and mixing of the MCMC steps.

To formulate grid cells using the current state, we let ${x_t}^{(s-1)}$ denote the current state in the MCMC iterations at time t, then similarly to the previous approach, we define the grid cell boundaries at the quantiles of a Gaussian distribution X for each t, where $X \sim N({x_t}^{(s-1)}, {\sigma_x}^2)$. As with the previous approach, the variance of the Gaussian distribution used to define the grid cells, ${\sigma_x}^2$, controls the range that the finite cells cover. This variance could be set, for example, as a fixed value (chosen via pilot-tuning), as a function of the current estimate of the system process variance or as a function of the current state within the MCMC iterations. We note that setting the majority of the grid cells to cover a small range relative to the high posterior density ranges worked well in practice (see Section 3.4).

In this approach, the TPMs need to be recalculated at each time point for every change in the model parameters and in order to calculate the reverse proposal probability in the M-H acceptance step, but will typically require fewer grid cells, smaller N, than the other approaches to achieve good mixing.

In general, there is a trade-off between the extent to which the grid cells are *well-placed* and the associated computational expense. For example, equally-sized grid cells are computationally fast but may give lower acceptance probabilities if they give coarse approximations in high-density regions. Where equally-sized grid cells give poor acceptance probabilities, for example, due to non-uniformity in high posterior regions over time, acceptance probabilities may be improved using Approach 2 or 3 at a greater computational expense.

3.3.2 Deterministic integration method

In Section 3.2.1, we focus on the simple case for approximating the HMM probabilities in each grid cell: mid-point integration using $D = 1$. However, the Riemann sum integral approximation method scales simply for higher order polynomials where $D \geq 2$, or these methods can be easily replaced by more complex nu-

merical integration strategies. However, the complexity of these methods should be balanced with their associated computational cost to ensure efficiency in this step. We therefore test the efficiency of the algorithm using mid-point integration ($D = 1$).

The *mid-point* and length of the cells must be defined to apply mid-point integration, even when the $n = 1$ or $n = N$ grid cells have an infinite range. In these infinite cells, we simply set this artificial length at the average length of the finite cells, and we set the artificial mid-point at half that length away from the corresponding upper or lower boundary of the finite cells.

Further, to ensure that all HMM probabilities are sufficiently high to avoid the proposal distribution resulting in a 'near-reducible' Markov chain, we set a lower bound on the HMM probabilities. In all the implementations of this chapter, we use a lower bound of 0.01 and renormalise the transition probabilities so that they sum to one.

3.3.3 Proposal distributions within the grid cells

Once we have sampled a set of grid cells, indexed by $b'_{1:T}$ from Section 3.2.2, we sample point values from within the grid cells using simple proposal distributions for each $t = 1, \ldots, T$. In all implementations, we sample from uniform distributions with domain in the finite grid cells.

To sample values for the state in infinite grid cells in the first case study of Section 3.4.1, we sample from a truncated Gaussian distribution with mean equal to the finite boundary and a variance of 5. The variance of the infinite-cell distribution is relatively low so that proposals in this grid cell are mostly concentrated around the boundary of the finite cells, increasing the density in the tails of the proposal distribution. In the second case study, we sample from a shifted Poisson distribution with mean parameter equal to 2 (again to ensure a relatively heavy-tailed proposal distribution), shifted to the lower bound of the upper (infinite) grid cell.

3.3.4 Toy examples

We provide worked examples to show the process of defining the grid cells under each approach, and how the HMMs are subsequently approximated and used to formulate M-H updates of the latent states. For illustrative purposes, we consider

a simple linear Gaussian SSM over a small number of time points:

$$\begin{aligned} x_1 &\sim N(x_0, \sigma_\eta^2), \\ x_t &\sim N(x_{t-1}, \sigma_\eta^2), \quad t = 2, 3, \\ y_t &\sim N(a x_t, \sigma_\epsilon^2), \quad t = 1, 2, 3, \end{aligned} \tag{3.6}$$

where x_0 is the mean parameter of the initial state distribution, a is an observation scaling parameter, and $\sigma_\eta^2, \sigma_\epsilon^2 > 0$ are the state and observation distribution variances respectively. The set of model parameters is denoted collectively by $\theta = (x_0, a, \sigma_\eta^2, \sigma_\epsilon^2)$. The data we consider are simulated from this model using $\theta = (1, 2, 0.5, 1)$. So that the reader can reproduce the analyses in this section, we use the R programming language and set a random number seed of 1234 (via `set.seed(1234)`). We simulate the SSM in the same order as Equation (3.6), resulting in simulated data $y_{1:3} = (-2.052746, 1.114420, 2.724983)$.

Set up We assume that the latent states, $x_{1:T}$, and model parameters, θ are unknown, and we aim to design a PMPMH algorithm targeting the joint posterior distribution, $p(x_{1:T}, \theta | y_{1:T})$. As discussed in Section 3.2, we sample from this joint distribution by updating the latent states and model parameters in turn from their full conditional distributions, $p(x_{1:T} | y_{1:T}, \theta)$ and $p(\theta | x_{1:T}, y_{1:T})$, at each M-H iteration. To focus our exploration here, we demonstrate the PMPMH algorithm in a single M-H iteration. That is, at the beginning of M-H iteration s, we assume that the current M-H values for the latent states and model parameters are $x_{1:T}^{(s-1)} = (0.01, 0.16, 1.45)$ and $\theta^{(s-1)} = (-0.54, 0.66, 0.35, 0.67)$, and show the process of updating the latent states to $x_{1:T}^{(s)}$ targeting $p(x_{1:T} | y_{1:T}, \theta^{(s-1)})$. Note that the model parameters are then updated to $\theta^{(s)}$ via standard Gibbs or Metropolis-Hastings updates targeting $p(\theta | x_{1:T}^{(s)}, y_{1:T})$, and the entire process is repeated for the desired number of M-H iterations as in Algorithm 10. Within the worked example for each approach, we use a small number of grid cells, $N = 5$, for illustrative purposes.

Approach 1

Under Approach 1 (discussed in Section 3.3.1), the finite grid cells are equally sized and do not change over time. Thus, in Step 1 (Section 3.2.1; Step 1(a) in Section 3.2.3), we aim to cover a large range of the state space to capture potentially varying high posterior density regions over time, and set the finite

grid cells to cover at least the minimum and maximum values of the data. We set the finite grid cell boundaries up to 3 units on either side of the mean of the data, over the interval $[-2.40, 3.60]$, as in Figure 3.4.

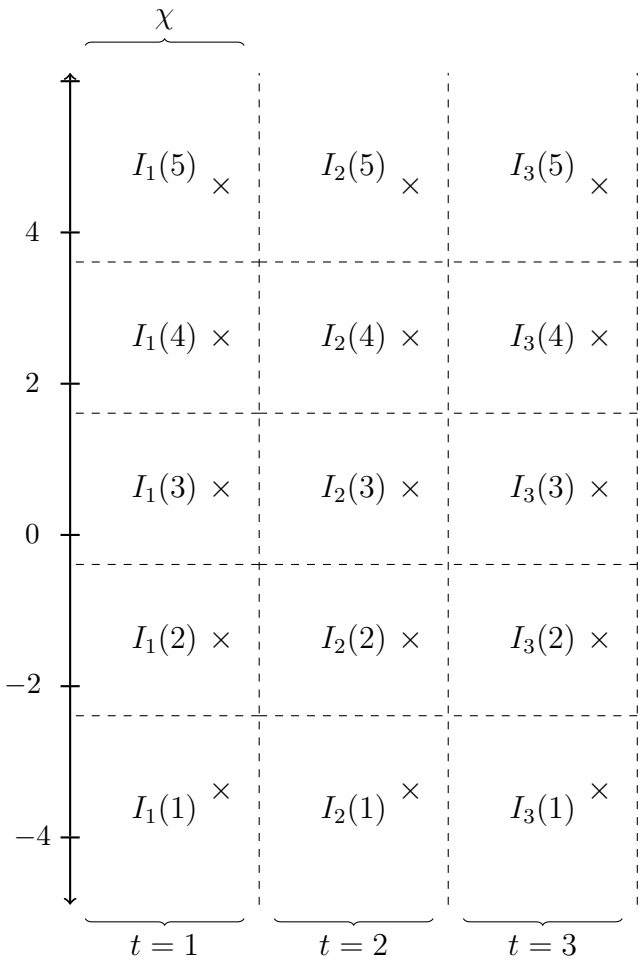


Figure 3.4: Illustration of the grid cells defined under Approach 1 for the toy example with data $y_{1:3}$. The grid cell boundaries are defined for each time point at approximately $-2.40, -0.40, 1.60, 3.60$. Each grid cell is labelled by the corresponding interval. Mid-points within each grid cell are shown by crosses.

Once the state space has been partitioned into grid cells, we approximate an HMM in the grid cell indices (Step 1(b) in Section 3.2.3). Using mid-point integration to approximate the SSM density within each grid cell, we set the 'lengths' of the infinite cells equal to the lengths of the finite cells, and set the 'mid-points' half that length from the respective finite boundaries, shown in Figure 3.4. We then calculate the approximate HMM in the grid cell indices (with random variables $B_t \in \{1, 2, 3, 4, 5\}$ for each t) using Equation (3.2). We may calculate the

approximate transitions from grid cell 3 to 1 up to proportionality as

$$\hat{P}(B_t = 1|B_{t-1} = 3, \theta^{(s-1)}) \propto 2 \times 2 \times p(dx_t = -3.40|dx_{t-1} = 0.60, \theta^{(s-1)}),$$

for $t = 2, 3$, approximately 3.2×10^{-10}. Once we have calculated such state transition probabilities for all $B_t \in \{1, 2, 3, 4, 5\}$, we then normalise the probabilities so that they sum to one. We then further threshold the probabilities to avoid a near-reducible Markov chain. As in the implementations in this chapter, we lower bound the HMM probabilities by 0.01 and renormalise. For example, in this case, $\hat{P}(B_t|B_{t-1} = 3, \theta^{(s-1)}) = (0.01, 0.01, 0.96, 0.01, 0.01)$ approximately. Note that the state TPMs are the same for all time points $t > 1$ since the grid cells are the same for all time points. We similarly approximate the entire HMM, including the observed state distribution and initial state probabilities as in Equation (3.2). At iteration s, we calculate one $1 \times N$ (1×5) matrix of initial state probabilities, one $N \times N$ (5×5) TPM and one $N \times T$ (5×3) observation matrix.

To sample a grid cell trajectory, we perform forward-filtering backward-sampling. First, we recursively calculate the forward-filtering probabilities, substituting the HMM probabilities as in Step 1(c) in Section 3.2.3. We then sample a grid cell index backward recursively. For example, if $\hat{P}(B_3|y_{1:3}, \theta^{(s-1)}) = (2.8 \times 10^{-3}, 0.10, 0.03, 0.67, 0.20)$, we may sample the grid cell index $b'_3 = 4$. We may then sample

$b'_2 = 4$ with

$$\hat{P}(B_2|B_3 = 4, y_{1:3}, \theta^{(s-1)}) = (6.7 \times 10^{-4}, 0.05, 0.01, 0.94, 7.9 \times 10^{-4})$$

$b'_1 = 2$ with

$$\hat{P}(B_1|B_2 = 4, y_{1:3}, \theta^{(s-1)}) = (0.03, 0.93, 3.6 \times 10^{-3}, 0.04, 3.6 \times 10^{-4}),$$

resulting in a proposed trajectory of grid cell indices, $b'_{1:3} = (4, 4, 2)$, each corresponding to a grid cell.

In Step 2 (Step 2(a) in Section 3.2.3), we sample a point trajectory for the latent states: we sample point values for each $x'_{1:3}$ independently from within the grid cells associated with the indices $b'_{1:3}$. For example, for the sampled indices $b'_{1:3} = (4, 4, 2)$, we sample $x'_1 \sim U(I_1(4))$, i.e., $x'_1 \sim \text{Uniform}(1.60, 3.60)$. Similarly, $x'_2 \sim \text{Uniform}(1.60, 3.60)$ and $x'_3 \sim \text{Uniform}(-2.40, -0.40)$.

We then accept or reject this sample according to its M-H acceptance probability (Equation 3.4 and Step 2(b), Section 3.2.3). In this example,

$$\alpha(x_{1:3}^{(s-1)}, x'_{1:3}|\theta^{(s-1)}) = \min\left(1, \frac{p(x'_{1:3}|y_{1:3}, \theta^{(s-1)})q(x_{1:3}^{(s-1)}|y_{1:3}, \theta^{(s-1)})}{p(x_{1:3}^{(s-1)}|y_{1:3}, \theta^{(s-1)})q(x'_{1:3}|y_{1:3}, \theta^{(s-1)})}\right)$$

where $p(x_{1:3}|y_{1:3}, \theta)$ is calculated up to proportionality as

$$p(x_{1:3}|y_{1:3}, \theta) \propto p(x_1|\theta)\prod_{t=2}^{3} p(x_t|x_{t-1}, \theta)\prod_{t=1}^{3} p(y_t|x_t, \theta),$$

and $q(x_{1:3}|y_{1:3}, \theta) = \hat{P}(B_{1:3} = b_{1:3}|y_{1:3}, \theta)\prod_{t=1}^{3} q(x_t|B_t = b_t)$ with $\hat{P}(B_{1:3} = b_{1:3}|y_{1:3}, \theta)$ is the probability of $b_{1:3}$ calculated with respect to the current grid cell boundary definition and $q(x_t|B_t = b_t)$ is the point value proposal distribution defined within the grid cell with index b_t.

Approach 2

We now show the steps of the PMPMH algorithm with the grid cells defined according to the data-driven quantiles of Approach 2 (Section 3.3.1). To illustrate this, in Step 1 (Step 1(a) in Section 3.2.3), we set the finite grid cell boundaries at each time point, $t \in \{1, 2, 3\}$, using the relationship between the observations and states in the observation distribution. We set the grid cells at the quantiles of X where $X \sim N(y_t/a^{(s-1)}, \sigma_y^2)$ and $a^{(s-1)}$ and denotes the current M-H parameter sample for a. We set all finite cells between the 10% and 90% quantiles of this distribution with $\sigma_y^2 = 0.87$ so that the finite cells cover approximately 50% of the range of the data. This method for defining data-driven grid cells is shown in Figure 3.5. In practice, as discussed in Section 3.3.1, the parameters of the quantile distribution used to set the grid cells can also be found using pilot tuning.

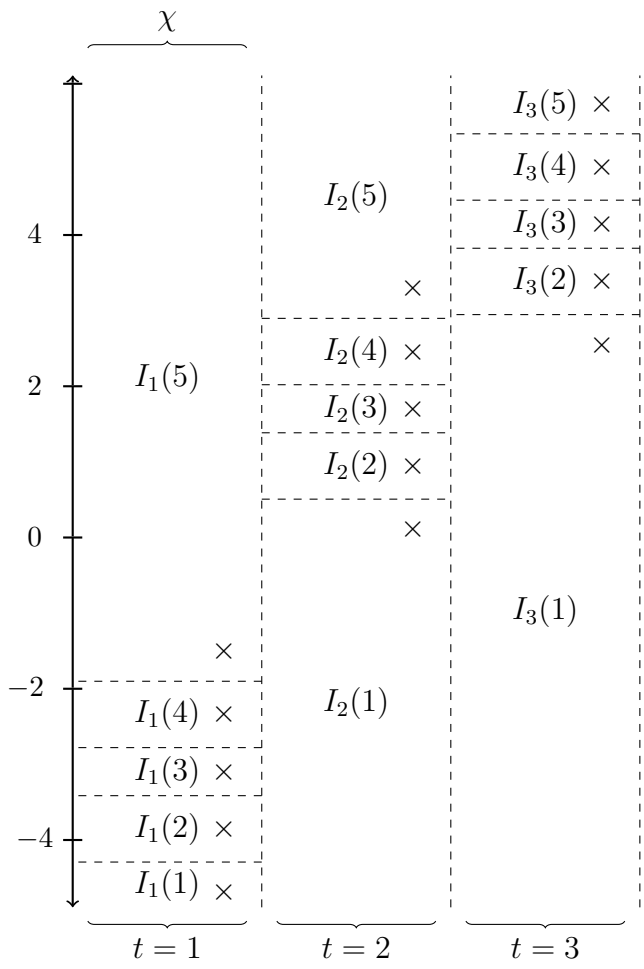


Figure 3.5: Illustration of the grid cells defined under Approach 2 for the toy example with data $y_{1:3}$. The grid cell boundaries are set at approximately $-4.31, -3.43, -2.79, -1.91$ for $t = 1$, $0.49, 1.37, 2.01, 2.88$ for $t = 2$, and $2.93, 3.81, 4.45, 5.32$ for $t = 3$. Each grid cell is labelled by the corresponding interval. Mid-points within each grid cell are shown by crosses.

Similarly to Approach 1, we now approximate the HMM in the grid cell indices using mid-point integration as in Equation (3.2) (Step 1(b) in Section 3.2.3). We set the lengths of the outer, infinite cells artificially at the average lengths of the finite cells (approximately 0.80) and the mid-points at half the respective length from the finite boundary. Under the grid definition here, for example,

$$\hat{P}(B_2 = 1 | B_1 = 3, \theta^{(s-1)}) \propto 0.80 \times 0.64 \times p(dx_2 = 0.09 | dx_1 = -3.11, \theta^{(s-1)}),$$

approximately 1.4×10^{-7}. Once the probabilities for $B_2 \in \{1, 2, 3, 4, 5\}$, have been calculated, we normalise this row of the TPM, impose a lower bound of 0.01 and re-normalise, resulting in $\hat{P}(B_2 | B_1 = 3, \theta^{(s-1)}) = (0.96, 0.01, 0.01, 0.01, 0.01)$ approximately. We calculate TPMs for each $t = 2, 3$ since the grid cells differ at

each time point. Calculating and normalising all HMM probabilities at iteration s results in one $1 \times N$ (1×5) matrix of initial state probabilities, $T-1$ $N \times N$ TPMs (two 5×5 TPMs) and one $N \times T$ (5×3) observation matrix. We then sample a grid cell trajectory and point values, and accept or reject the sample as in the worked example for Approach 1 (Step 1(c) onwards).

Approach 3

Under Approach 3 (Section 3.3.1), the grid cells are defined as a function of the current latent states, $x_{1:3}^{(s-1)}$, in order to approximately locate the high posterior density regions at each time point. In Step 1 (Step 1(a) in Section 3.2.3), we use quantiles of a Gaussian distribution centred around the current latent state at each time point. That is, the quantiles of X where $X \sim N(x_t^{(s-1)}, \sigma_x^2)$ for each $t = 1, 2, 3$. We set the finite grid cells to cover 40% of the range of the data: between the 10% and 90% quantiles with $\sigma_x^2 = 0.56$. The resulting current state-centred grid cells are shown in Figure 3.6.

As in Approaches 1 and 2, we approximate the HMM (Step 1(b) in Section 3.2.3), by setting the length of the infinite cells at the average length of the finite cells (approximately 0.64), and define the mid-points at half the respective length from the finite boundary. We may then calculate the HMM probabilities,

$$\hat{P}(B_2 = 1 | B_1 = 3, \theta^{(s-1)}) \propto 0.64 \times 0.51 \times p(dx_t = -1.11 | dx_{t-1} = 0.01, \theta^{(s-1)}),$$

approximately 0.04. Once all HMM probabilities have been calculated and normalised, the probabilities are lower bounded (here by 0.01) and re-normalised, resulting in, for example, $\hat{P}(B_2 | B_1 = 3, \theta^{(s-1)}) = (0.07, 0.36, 0.34, 0.21, 0.02)$ approximately. The TPMs are calculated for each $t > 1$ since the grid cells vary across time points. We then perform Step 1(c) onwards as in Approaches 1 and 2. However, we also calculate the reverse state probability for the proposal probability of the current states, i.e., the HMM probability of the current states centred on the proposed states. In total, at iteration s, we therefore calculate two $1 \times N$ (1×5) sets of initial state probabilities, $2(T-1)$ $N \times N$ TPMs (four 5×5 TPMs) and two $N \times T$ observation matrices (two 5×3 observation matrices).

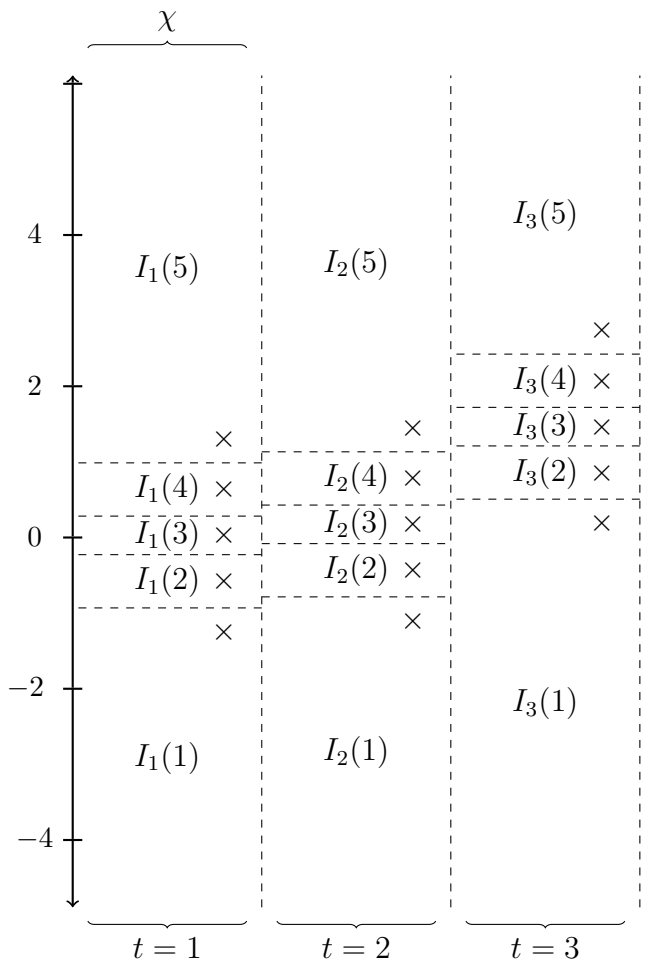


Figure 3.6: Illustration of the grid cells defined under Approach 3 for the toy example with data $y_{1:3}$. The grid cell boundaries are set at approximately $-0.95, -0.23, 0.26, 0.97$ for $t = 1$, $-0.80, -0.09, 0.41, 1.12$ for $t = 2$, and $0.49, 1.20, 1.70, 2.41$ for $t = 3$. Each grid cell is labelled by the corresponding interval. Mid-points within each grid cell are shown by crosses.

3.4 Case studies

We demonstrate the proposed PMPMH algorithm via two case studies. The first is an SSM with a simple one-dimensional Gaussian mixture state process, demonstrating how the algorithm can be implemented, and the properties of the algorithm when different practical decisions are made. We then show how a similar PMPMH implementation can be used to efficiently sample the latent states of a more challenging 2-dimensional population growth model that can display near-chaotic behaviour.

In each case study, we compare the performance of the algorithm to two particle Gibbs algorithms: the particle Gibbs sampler (Section 2.3) and the particle

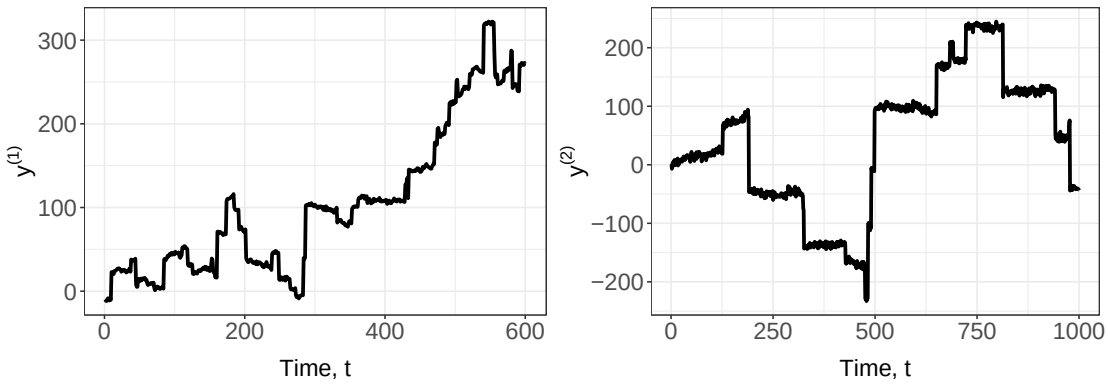
Gibbs with ancestor sampling (PGAS) algorithm (Section 2.3.1). The PGAS algorithm, in particular, is a state-of-the-art method that often improves upon the mixing properties of the particle Gibbs algorithm by reducing sample impoverishment (Kantas et al., 2015; Meent et al., 2015; Nonejad, 2015), though at an increased computational cost.

3.4.1 Gaussian mixture state-space model

We consider a simple one-dimensional Gaussian mixture state process:

$$\begin{aligned} x_1 &\sim w_1 N(1, \sigma^2_{\eta_1}) + (1 - w_1) N(1, \sigma^2_{\eta_2}), \\ x_t | x_{t-1} &\sim w_t N(x_{t-1}, \sigma^2_{\eta_1}) + (1 - w_t) N(x_{t-1}, \sigma^2_{\eta_2}), \;\; t = 2, \ldots, T, \\ w_t &\sim \text{Bernoulli}(p), \;\; t = 1, \ldots, T, \end{aligned}$$

where $p \in [0, 1]$ denotes the probability of selecting each mixture component of the state distribution (the Gaussian distributions with respective variances $\sigma^2_{\eta_1}$ or $\sigma^2_{\eta_2}$). Data, $y_{1:T}$, are observed according to $y_t | x_t \sim N(x_t, \sigma^2_\epsilon)$ and the model parameters are given by $\theta = (p, \sigma^2_{\eta_1}, \sigma^2_{\eta_2}, \sigma^2_\epsilon)$. We simulate two data sets from this model, $y^{(1)}_{1:T}$ using $\theta = (0.9, 1, 700, 1)$ and $T = 600$, and $y^{(2)}_{1:T}$ using $\theta = (0.99, 1, 10000, 10)$ and $T = 1000$, shown in Figure 3.7.



(a) (b)

Figure 3.7: Simulated data from the Gaussian mixture SSM: $y^{(1)}_{1:T}$ using (a) $\theta = (0.9, 1, 700, 1)$, $T = 600$, and (b) $y^{(2)}_{1:T}$ using $\theta = (0.99, 1, 10000, 10)$, $T = 1000$

Both data sets are simulated with a high-variance second mixture component, selected with a low probability, resulting in infrequent but large *jumps* in the state process. The parameter values used to simulate $y^{(2)}_{1:T}$ result in more occasional and larger jumps in the state process than the first. We compare the properties of the algorithm in both cases.

PMPMH implementation

We sample from the joint posterior distribution of the states and parameters, updating $x_{1:T}$ and θ from their conditional distributions. We use a single-site updating strategy to target both $p(\theta|x_{1:T}, y^{(1)}_{1:T})$ and $p(\theta|x_{1:T}, y^{(2)}_{1:T})$ and we assign vague priors to the parameters to enable diagnostic comparisons with the simulated parameters. For $y^{(1)}_{1:T}$:

$$\sigma^2_\epsilon \sim \text{InvGamma}(2, 2),$$

where InvGamma denotes an inverse gamma distribution and results in Gibbs updates for σ^2_ϵ. We assign further high-variance priors to the other parameters:

$$\begin{aligned} p &\sim \text{Uniform}(0, 1), \\ \sigma^2_{\eta_1} &\sim \text{InvGamma}(2, 2), \\ \sigma^2_{\eta_2} &\sim \text{InvGamma}(2, 700). \end{aligned}$$

At each iteration, proposed values for these parameters are sampled from uniform random walk M-H proposal distributions. We use intervals of length 0.3, 2, and 160 for p, $\sigma^2_{\eta_1}$, and $\sigma^2_{\eta_2}$ respectively. For $y^{(2)}_{1:T}$, we similarly assign the independent priors:

$$\begin{aligned} \sigma^2_\epsilon &\sim \text{InvGamma}(2, 10), \\ p &\sim \text{Uniform}(0, 1), \\ \sigma^2_{\eta_1} &\sim \text{InvGamma}(2, 2), \\ \sigma^2_{\eta_2} &\sim \text{InvGamma}(2, 700), \end{aligned}$$

where the prior for σ^2_ϵ again results in Gibbs steps for this parameter. We propose p, $\sigma^2_{\eta_1}$ and $\sigma^2_{\eta_2}$ at each iteration from a uniform random walk proposal distribution over 0.02, 0.5, and 20,000 units respectively. All intervals for the random walk proposal distributions are set using pilot tuning.

We apply the two stages of the PMPMH algorithm to update $x_{1:T}$ conditional on θ. The PMPMH framework presented can be adapted in several ways, for example, by changing the deterministic integration method used to approximate the HMM probabilities, the number of grid cells, and the distribution used to propose values from within grid cells. However, we show in this section that we achieve stable and efficient performance by fixing a number of these decisions and opting for the simple choices given in Section 3.3.

We focus on the practical decisions that are relevant to the efficiency of the algorithm: (a) the choice of Approach 1, 2 or 3 of Section 3.3.1 (including the range of the state space covered by the finite grid cells, $\mathcal{S}$), (b) the number of grid cells, N, and (c) the temporal block sizes in which the states are updated, ℓ. We test the efficiency of the algorithm under various combinations of these tuning parameters:

(a) the finite grid cells in Approach 1 over a range of $\mathcal{S} = 150, 250, 350, 450, 550$ units for both data sets (compared to the range of each set of observations, $y_{1:T}^{(1)}$ and $y_{1:T}^{(2)}$, equal to 334 and 478 units respectively). In Approaches 2 and 3, we use $\mathcal{S} = 1, 3, 5, 7, 9, 11, 13, 15$ units for both models,

(b) $N = 5, 10, 20$,

(c) $\ell = 1, 4, 10$, overlapping blocks by one state to improve the mixing of the states at the "boundaries" of each block as suggested by Fearnhead (2011).

Results

For each combination of the tuning parameters listed above, the results are based on 10 separate runs of 10,000 iterations each, taking $1.3 - 7.5$ hours using one core and a 1.6 GHz CPU under $y_{1:T}^{(1)}$ (depending on the number of grid cells and the approach chosen) and $2.2 - 11$ hours under $y_{1:T}^{(2)}$. We present the results for each model in Tables 3.1 and 3.2 respectively.

Increasing the number of grid cells, N, results in a more accurate approximation to $p(x_{1:T}|y_{1:T}, \theta)$ in the region covered by the finite grid cells, improving mixing but at a higher computational cost. The equally-spaced grid cells of Approach 1, defined in Section 3.3.1, performed poorly on both simulated models compared to the quantile-based methods of Approaches 2 and 3, requiring a large finite cell range for both models (350 and 550 units respectively) due to a large range of values in the high posterior regions, and thus large numbers of grid cells and a large computational cost, to provide a reasonable HMM approximation.

$\mathbf{y}_{1:T}^{(1)}$						
		% **of range**	**Approach 2**		**Approach 3**	
N	$\mathcal{S}$	**of the data**	**ESS**	**ESS/s**	**ESS**	**ESS/s**
5	1	0.3	100	0.02	-	-
	3	0.9	900	0.18	1000	0.20
	5	1.5	700	0.14	100	0.02
	7	2.1	300	0.06	-	-
	9	2.6	200	0.04	-	-
	≥ 11	≥ 3.3	-	-	-	-
10	1	0.3	100	0.01	-	-
	3	0.9	1000	0.10	1700	0.17
	5	1.5	2400	0.24	200	0.02
	7	2.1	2200	0.22	-	-
	9	2.6	2000	0.20	-	-
	≥ 11	≥ 3.3	-	-	-	-
20	1	0.3	100	0.00	-	-
	3	0.9	700	0.03	2200	0.08
	5	1.5	2700	0.10	1000	0.04
	7	2.1	3000	0.11	-	-
	9	2.6	3000	0.11	-	-
	≥ 11	≥ 3.3	-	-	-	-

Table 3.1: Average effective sample size (ESS) and effective sample size per second (ESS/s) for Approaches 2 and 3 of the PMPMH algorithm, defined in Section 3.3.1, with temporal blocks of size $\ell = 4$. The range of the finite cells is denoted by $\mathcal{S}$, the number of grid cells, N. Dashed lines indicate that convergence to the target distribution had not occurred within 10,000 iterations. The ESS is rounded to the nearest 100 to account for variation between chains. Since at least one model parameter was updated at each iteration for each run of the algorithm, the computational times are the same for Approaches 2 and 3: 5,000 seconds for $N = 5$; 10,000 seconds for $N = 10$; and 27,000 seconds for $N = 20$, where the computational time is rounded to the nearest 1000 seconds to account for variations in computing speeds.

$\mathbf{y}_{1:T}^{(2)}$						
		% of range	Approach 2		Approach 3	
N	$\mathcal{S}$	**of the data**	**ESS**	**ESS/s**	**ESS**	**ESS/s**
5	≤ 3	≤ 0.6	-	-	-	-
	5	1	-	-	200	0.03
	7	1.5	-	-	500	0.06
	≥ 9	≥ 1.9	-	-	-	-
10	≤ 3	≤ 0.6	-	-	-	-
	5	1	-	-	200	0.02
	7	1.5	100	0.01	500	0.04
	9	1.9	200	0.02	200	0.02
	11	2.3	100	0.01	-	-
	≥ 13	≥ 2.7	-	-	-	-
20	≤ 3	≤ 0.6	-	-	-	-
	5	1	-	-	300	0.01
	7	1.5	100	0.00	600	0.02
	9	1.9	200	0.01	400	0.01
	11	2.3	200	0.01	400	0.01
	≥ 13	≥ 2.7	-	-	-	-

Table 3.2: Average effective sample size (ESS) and effective sample size per second (ESS/s) for Approaches 2 and 3, defined in Section 3.3.1, of the PMPMH algorithm with temporal blocks of size $\ell = 4$. The range of the finite cells is denoted by $\mathcal{S}$, the number of grid cells, N. Dashed lines indicate that convergence to the target distribution had not occurred within 10,000 iterations. The ESS is rounded to the nearest 100 to account for variation between chains. Since at least one model parameter was updated at each iteration for each run of the algorithm, the computational times are the same for Approaches 2 and 3: 8,000 seconds for $N = 5$; 13,000 seconds for $N = 10$; and 38,000 seconds for $N = 20$, where the computational time is rounded to the nearest 1000 seconds to account for variations in computing speeds.

Under the quantile-based Approaches 2 and 3 (Section 3.3.1), using blocks of size $\ell = 10$ or a range for the finite cells greater than 11 units required more than $N = 20$ grid cells for convergence. In addition, these tuning parameters resulted in much more costly and less efficient implementations than the other sets of tuning parameters. Further, using single-site updates ($\ell = 1$) gave poor mixing when compared to blocks of size $\ell = 4$ due to the correlation between consecutive states. The results when using $\ell = 1, 10$, a finite grid cell ranges greater than 11 units and $N > 20$ are thus excluded from Tables 3.1 and 3.2.

The results for Approaches 2 and 3 in Table 3.1 use blocks of size $\ell = 4$, and where convergent, converge within $1000 - 5000$ iterations using the Brooks-Gelman-Rubin (BGR) diagnostics in Brooks and Gelman (1998); Gelman and Rubin (1992). For the implementations using simulated data $y_{1:T}^{(1)}$, both quantile-

based Approaches 2 and 3 for defining the grid cell boundaries yield similar levels of efficiency when considering the effective sample size (ESS) per second, likely since the small observation error means that both approaches focus grid cells in roughly the same region of the state space around the current state. This is potentially also the reason that both methods required a small range for the finite cells and a small number of grid cells for convergence relative to the range of the data ($1-9$ vs. 344 units, around $0.3-2.6\%$ of the range of the data): as is true of proposal distributions that make local moves, grid cells can be focused over a smaller range of the data compared to global-move approaches and still achieve good acceptance probabilities and mixing. The smaller range also means that as few as 5 grid cells can be used to achieve good convergence properties (compared to 50 grid cells when applying Approach 1) via a good HMM approximation over the region, reducing the computational cost of the approaches.

The data $y_{1:T}^{(2)}$ are simulated with large observation process variance, thus the proposal distribution under Approach 2 is now different to the local-move proposal distribution defined under Approach 3. In this case, the proposal distribution defined under Approach 2 results in slower mixing and convergence than that under Approach 3, which exhibits relatively stable performance even for $N = 5$. The efficient ranges for the finite grid cells under Approach 3 are similar to those for the first model (here, $1-2.3\%$ of the range of the data).

We also fitted both of the models using the particle Gibbs and PGAS algorithms with 5 to 1000 particles, and various combinations of resampling thresholds based on the standard percentage ESS criterion (Cappé et al., 2005, Chapter 7). The particle Gibbs sampler did not converge for either model using as many as 1000 particles, resulting in computational times of around 14 hours for $y_{1:T}^{(1)}$ and 28 hours for $y_{1:T}^{(2)}$ on one core and a 1.6 GHz CPU. Conversely, the PGAS sampler converged using as few as 5 particles for both models. On the whole, the PGAS sampler gave greater levels of efficiency than the PMPMH algorithm for both of the models, achieving an average ESS per second of around 4.45 for $y_{1:T}^{(1)}$ and 1.49 for $y_{1:T}^{(2)}$. However, this efficiency was not uniform across all states. The average ESS per second of the states simulated according to the second mixture was 0.25 to 0.41 for $y_{1:T}^{(1)}$ and 0.004 to 0.008 for $y_{1:T}^{(2)}$. In contrast, where convergent, the PMPMH algorithm was more robust to the mixture associated with the state, with the ESS per second of states in the second mixture minimally 99% of those quoted in Tables 3.1 and 3.2 (the average ESS per second in the second mixture-distributed states ranging from 0.01 to 0.24 for $y_{1:T}^{(1)}$ and 0.01 to 0.06 for $y_{1:T}^{(2)}$).

Overall, the PMPMH algorithm provides a more consistent approach to in-

ference about the latent states and model parameters under the SSM considered here. The latent state-centred Approach 3 typically required low grid cell coverage of the state space, leading to a reduction in the computational cost of the HMM approximation. This approach also appeared the most robust to the underlying model. We now investigate the performance of the algorithm on a challenging model that can display near-chaotic behaviour.

3.4.2 Nicholson's blowfly model

We consider Nicholson's blowfly model for chaotic population growth described by Wood (2010). The population counts over time, denoted by $N_{1:T}$, arise from two correlated survival and birth processes, denoted $S_{1:T}$ and $R_{\tau+1:T}$, $\tau > 0$, respectively. Following Wood (2010), we let $\exp(-\gamma\epsilon_t)$ denote the daily survival probability with associated environmental error term ϵ_t, such that $\epsilon_t \sim \text{Gamma}(\beta_\epsilon, \beta_\epsilon)$, $\beta_\epsilon > 0$. The survival component of the system process for $t = 1, \ldots, T$ is given by

$$S_t \sim \text{Binom}(N_{t-1}, \exp(-\gamma\epsilon_t)).$$

Letting e_t denote an environmental noise term in the reproductive process such that $e_t \sim \text{Gamma}(\beta_e, \beta_e)$, $\beta_e > 0$, the reproductive component of the system process for $t = \tau + 1, \ldots, T$, is given by

$$R_t \sim \text{Poisson}\left(PN_{t-\tau-1} \exp\big(-\frac{N_{t-\tau-1}}{N_0}\big)e_t\right),$$

where $N_t = S_t + R_t$ for $t = \tau+1, \ldots, T$ and $N_t = S_t$ for $t = 1, \ldots, \tau$. We let $\tau = 5$ be the known time lag between population count and subsequent birth count, and $N_0 = 50$ is the known initial population count. The survival and birth processes, $S_{1:T}$ and $R_{\tau+1:T}$, are unknown latent states, with observed population counts $y_t \sim \text{Poisson}(\phi N_t)$, for all t. Further, the model parameters $\theta = (\gamma, P, \beta_\epsilon, \beta_e, \phi)$, $\epsilon_{1:T}$, and $e_{\tau+1:T}$ are unknown. Figure 3.8 shows the simulated data used in this case study, $y_{1:T}$, using $\theta = (0.7, 50, 1, 0.1, 1)$ (unbiased mapping from state to observation since $\phi = 1$) and $T = 300$.

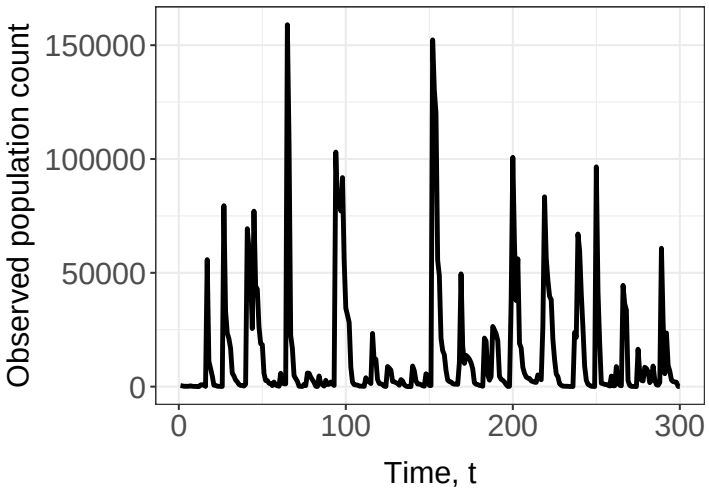


Figure 3.8: Simulated blowfly population count data using $T = 300$ and $\theta = (0.7, 50, 1, 0.1, 1)$

PMPMH implementation

We develop an M-H updating strategy targeting the joint posterior distribution of the unknown latent states and model parameters, $p(S_{1:T}, R_{\tau+1:T}, \theta|y_{1:T})$. The model parameters are updated from their full conditional distribution, $p(\theta|S_{1:T}, R_{\tau+1:T}, y_{1:T})$, making use of conjugate priors for single-site updates of θ where possible, and assigning vague priors to help diagnose convergence:

$$\begin{aligned}
\gamma &\sim \text{Gamma}(0.007, 0.01), \\
P &\sim \text{Gamma}(50, 1), \\
\beta_\epsilon &\sim \text{InvGamma}(100, 100), \\
\beta_e &\sim \text{InvGamma}(10, 1), \\
\phi &\sim \text{Gamma}(0.01, 0.01).
\end{aligned}$$

This results in single-site Gibbs updates for P and ϕ. For γ, β_ϵ and β_e, we use a random walk M-H step with uniform proposal distributions over intervals of length 0.03, 0.5, and 0.05 respectively. The parameters of the proposal distributions were chosen via pilot tuning.

We now describe how the PMPMH algorithm can efficiently update the latent states, $S_{1:T}$ and $R_{\tau+1:T}$, conditional on θ. First, the state process is now two-dimensional at each time point. To simplify the design of the grid used within the PMPMH algorithm, we build separate PMPMH proposal distributions targeting the full conditional distributions in each state dimension, $p(S_{1:T}|R_{\tau+1:T}, y_{1:T}, \theta)$

and $p(R_{\tau+1:T}|S_{t:T}, y_{1:T}, \theta)$, respectively. To sample from each of the full conditional distributions in turn, a number of the practical decisions are needed, as in Section 3.4.1. We first fix several of the practical decisions to those in Section 3.3 and use blocks of size $\ell = 4$, overlapping blocks by one state to reduce the correlation between states at the block boundaries. We test the performance of the algorithm for $N \in \{10, 20, 50\}$ grid cells.

There are two main differences in this implementation compared to the previous example. The first accounts for the near-chaotic state processes and the large range of the data (1.59×10^5 units). We apply the current state-centred Approach 3 of Section 3.3.1, which requires fewer grid cells for good mixing properties due to the lower range for the finite cells required to achieve efficient (more local) moves. Within this approach, we permit large variability around large values for the state process by adjusting the variance of the Gaussian distributions used to set the boundaries of the grid cells: we set the variance proportional to the current state at each time point. We try factors of proportionality of $0.1, 0.25, 0.5, 1$, with finite grid cells between the q and $1 - q$ quantiles with $q = 0.01, 0.1, 0.2$, resulting in (average) ranges for the finite cells $\mathcal{S} = 52 - 400$. The second difference is that we ensure the grid cells are over a discrete space bounded at zero by rounding the quantiles used to determine the grid cells, setting the lower boundary to zero if needed.

Results

We assess the performance of the PMPMH algorithm using 10 independent MCMC simulations of 50,000 iterations and present the performance metrics in Table 3.3. Where the PMPMH algorithm converges, convergence was achieved within $4{,}000 - 18{,}000$ iterations, taking $11 - 27$ hours using one core and a 1.6 GHz CPU. Fairly consistent and efficient performance (in terms of ESS per second; Table 3.3) is achieved using $20 - 50$ grid cells with the finite cells covering a region of the state space $0.05 - 0.25\%$ of the range of the data ($\mathcal{S} \in [82, 400]$).

Max N	Average S			
	$\mathcal{S}$	**R**	**ESS**	**ESS/s**
10	≤ 76	≤ 70	-	-
	82-100	75-90	2500	0.05
	100-200	90-180	3200	0.07
	> 200	> 180	-	-
20	≤ 76	≤ 70	-	-
	82-100	75-90	4400	0.09
	100-200	90-180	5400	0.09
	200-300	180-260	7000	0.11
	300-400	260-300	6700	0.10
50	≤ 76	≤ 70	-	-
	82-100	75-90	4400	0.06
	100-200	90-180	5700	0.06
	200-300	180-260	7300	0.08
	300-400	260-300	6900	0.07

Table 3.3: Average ESS and ESS/s across the state and parameter samples for various sets of tuning parameters under Approach 3 with blocks of size $\ell = 4$. Since the range of the finite cells varies across time points, we provide ranges for each range calculated from the MCMC output. The ESS has been rounded to the nearest 100 to account for the variability within the chains

Approach 3 mixes effectively by producing a reasonable HMM approximation in a relatively small region around the current state and is also computationally cheap to implement. The efficiency of the algorithm reduces at an upper bound of 50 grid cells, indicating that the improved mixing properties of the algorithm are not justified by the extra cost when compared with using fewer grid cells. Conversely, 10 grid cells, although lower in computational cost, do not produce a sufficiently accurate approximation to the posterior in the region of the current state. We also note that, similarly to the previous case study, implementations relying on *extremely* small moves around the state space (finite cells $< 0.05\%$ of the range of the data) gave poor mixing and convergence properties.

The PGAS sampler, when applied to the model using both joint updating of the latent states and updating each latent state process in turn from their full conditional distributions, did not converge in 50,000 iterations with 1000 particles, remaining in a range of the state space unrepresentative of the posterior due to sample impoverishment and taking 160 hours with one core and a 1.6 GHz CPU.

3.5 Discussion

We provide a novel and efficient approach to fitting general SSMs to observed data. The approach uses tractable HMM approximations to efficiently update the unobserved latent states in an M-H algorithm. We demonstrate the generality of the proposed approach by its application to two problems, including a challenging near-chaotic problem. The proposed PMPMH algorithm is demonstrated to provide reliable posterior estimates within reasonable computational time frames, especially when compared to a state-of-the-art method, which did not converge within a reasonable time frame for the near-chaotic problem. The flexibility of the PMPMH approach via the tuning parameters, including the location, size, and number of grid cells, and the temporal block size, provides an adaptable and efficient algorithm. Using simple methods for placing the grid cells, such as equally-sized grid cells, can be efficient since they are computationally cheap. However, such simple methods may require a large number of grid cells to achieve good mixing if the range of the data is large. A current state-centred approach for approximating the HMM is useful, especially when the state space covers a large range, and using a data-centred approach is useful provided the range of the data is small.

The PMPMH approach motivates several interesting points for future research. One avenue for research is to consider the computational efficiency of parallelisation when applied to different components of the algorithm. The computational efficiency of algorithms using block-updating strategies can be improved via parallelisation (King et al., 2023). Within the framework of the PMPMH algorithm specifically, serially independent blocks of states can be updated in parallel within each M-H iteration. Further, for the state-centred grid cell approach presented here, computing the approximate HMM probabilities in each block in parallel can lead to an improvement in the execution time of the approximate HMM computations. However, as with parallelisation schemes for particle MCMC methods, such as in Henriksen et al. (2012), the user must balance memory limitations and the computational cost from re-synchronisation with the computational gains from parallel implementation.

Within the sequential (i.e., not parallel) computing context of this chapter, one way to reduce the overall computational cost of the algorithm is to reduce the number of times the HMM approximation is calculated across iterations. That is, periodically update the HMM approximation at fixed MCMC iteration numbers and fixing the HMM approximations in the iterations between updates. This is valid for static grid cell approximation methods that do not depend on the current

state or model parameters. However, for grid definitions that adapt to previous samples (for example, Approaches 2 and 3), the resulting posterior estimate may become biased since the chain is no longer Markovian (Haario et al., 1999). However, Haario et al. (1999) also show that the bias introduced is negligible in some applications and that unbiased samples can be obtained by introducing some predetermined stopping criterion for updating the grid, that is, specifying a number of iterations beyond which the grid is no longer updated. The computational gains from updating the HMM approximation less frequently should, however, be balanced with potentially reduced mixing properties. For example, local-move grids using a small range for finite cells are more sensitive to the frequency of the HMM approximation since they rely on proposed moves being made in the region of the current state. However, updating the grid less frequently may give an efficient approach when applied to more global-move samplers. We could also consider the combination of approaches to defining the grid cells, for example, defining a data-driven or state-centred approach (Approach 2 or 3) depending on the magnitude of the current observation variance sample. An alternative approach to reducing the computational cost of the HMM approximations could apply an extra corrective importance sampling step to proposed latent states, possibly reducing the initial accuracy and computational cost required in the HMM approximation. We explore such a method further in Chapter 4.

Finally, although we demonstrated the scalability of the proposed algorithm to higher-dimensional spaces by updating each state dimension conditionally on the remaining state dimensions, high-dimensional spaces are a particular challenge if individual state dimensions are highly correlated, resulting in poor mixing. In this case, if low-dimensional sets of state dimensions are independent, for example, factorial SSMs (Ghahramani and Jordan, 1997; Rimella and Whiteley, 2022), or if dimensions of the state space are partially integrable (Borowska and King, 2023), the poor mixing from highly-correlated state dimensions could be improved by performing joint updates using lower-dimensional TPM approximations. However, in general, the issue of the scalability of grid-based methods to high-dimensional SSMs is a challenge to the proposed algorithm and an active area of research.

Chapter 4

Grid particle Gibbs with ancestor sampling

4.1 Introduction

This chapter expands the point mass proposal Metropolis-Hastings (PMPMH) framework of the previous chapter and introduces an efficient deterministic hidden Markov model (HMM) approximation method within the particle Gibbs framework (Section 2.3, and in particular Section 2.3.1). The approach is motivated by the use of deterministic HMM approximations to improve the efficiency of sequential Monte Carlo (SMC) steps while generally reducing the computational cost of HMM approximations of the state-space model (SSM).

We introduce a novel particle Gibbs algorithm targeting the joint distribution of the latent states and model parameters, referred to as the *grid particle Gibbs with ancestor sampling* (GPGAS) algorithm. First, we approximate the posterior distribution of the latent states conditional on the model parameters by a new particle filter. This particle filter uses efficient HMM approximations to formulate importance distributions, allowing us to propose particles in high posterior areas of the state space. This improves particle degeneracy, leading to a particle filter with many fewer particles (for the same precision) or more accurate approximations of the conditional latent state distribution (for the same number of particles). We then use the HMM particle filter within particle Gibbs with ancestor sampling (PGAS) steps to update the latent states conditional on the model parameters, and the model parameters are updated using standard Gibbs or Metropolis-within-Gibbs steps. While the proposed HMM approximation can be used within other parameter estimation methods or even as a stand-alone fil-

ter, we focus on its use within PGAS methods, where the proposed algorithm is particularly well-suited and efficient.

We demonstrate the efficiency of the GPGAS algorithm by focusing on a class of models that remain challenging to fit: regime-switching SSMs. These models embed an additional latent state process allowing the observation and latent state transition models to change abruptly. However, despite their widespread use (Haimerl and Hartl, 2023; Hamilton, 1989; Liang-qun et al., 2009), current computational methods for fitting the latent states and model parameters of general regime-switching SSMs can be inefficient due to the abrupt changes in the state process. We test the performance of the proposed GPGAS algorithm when applied to such models, including a challenging real data case study focusing on tourism demand recovery in Edinburgh.

The rest of this chapter is structured as follows. In Section 4.2, we describe optimal approaches to inference via PGAS algorithms and motivate the proposed GPGAS algorithm, before introducing the new GPGAS algorithm in Section 4.3. We then apply the proposed GPGAS algorithm to challenging regime-switching SSMs in Section 4.4, first focusing on a simulated data example of stochastic volatility with leverage. We then investigate the efficiency of the proposed method when applied to the challenging tourism demand case study. Finally, we discuss the proposed method and potentially interesting avenues for future research in Section 4.5.

4.2 Optimal importance distributions

We revisit the PGAS algorithm described in detail in Chapter 2 (Algorithm 9) and applied in Chapter 3, Section 3.4. The PGAS algorithm is an alternative MCMC approach to inference about the latent states and model parameters of an SSM using conditional SMC with ancestor sampling (CSMC-AS, defined in Algorithm 8). In particular, by sampling reference trajectory ancestors in the CSMC-AS steps, the PGAS algorithm addresses the poor mixing observed in standard particle Gibbs methods (Berntorp and Di Cairano, 2017; Chopin and Singh, 2015; Nonejad, 2015; Wigren et al., 2019). In contrast to the PMPMH approach introduced in Chapter 3, the PGAS algorithm requires fewer tuning parameters and benefits from the sequential correction of state samples in the SMC steps (improving their accuracy), resulting in proposals that are always accepted. However, as observed in Rainforth et al. (2016) and in Section 3.4, PGAS methods can be inefficient when there is a high rate of sample impover-

ishment in the CSMC-AS algorithm. If sample impoverishment is particularly prevalent, the pool of trajectories at each CSMC-AS recursion may represent the posterior distribution poorly and the MCMC sampler may not explore the space sufficiently.

We therefore turn our attention to efficient solutions to the initial sample impoverishment problem to improve the mixing of the PGAS algorithm. Ideally, the SMC importance distributions generate particles that exactly represent the posterior distribution, thus producing uniformly distributed weights and maximising the number of particles that survive the resampling steps of the CSMC-AS algorithm. As in Equation (2.5), the SMC steps first sample particles from the importance distribution, $x_t^m \sim q(x_t|y_t, x_{t-1}^m, \theta)$, $m = 1, \ldots, M$, and then approximate the conditional distribution of the latent states by

$$\begin{aligned} \hat{p}(x_{1:t}|y_{1:t}, \theta) &= \sum_{m=1}^{M} W_{1:t}^m \delta_{x_{1:t}^m}(x_{1:t}), \\ W_t^m &= \frac{w_{1:t}^m}{\sum_{k=1}^m w_{1:t}^k}, \quad w_{1:t}^m \propto w_{1:t-1}^m \frac{p(x_t^m|x_{t-1}^m, \theta) p(y_t|x_t^m, \theta)}{q(x_t^m|y_t, x_{t-1}^m)}, \end{aligned} \tag{4.1}$$

for $t = 1, \ldots, T$, where $\delta_{x_{1:t}^m}(x_{1:t})$ denotes the Dirac function at $x_{1:t} = x_{1:t}^m$; and $w_{1:t}^{1:M} = \{w_{1:t}^m\}_{m=1}^M$ and $W_{1:t}^{1:M} = \{W_{1:t}^m\}_{m=1}^M$ denote the set of unnormalised weights and normalised $w_t^{1:M}$ weights at time t, respectively. To minimise sample impoverishment at each recursion (i.e., produce uniformly distributed weights), the optimal approach samples particles from $p(x_{1:t}|y_{1:t}, \theta)$ directly. This approach also approximates the likelihood of all previous observations (Branchini and Elvira, 2021; Chopin and Papaspiliopoulos, 2020, Chapter 10; Elvira et al., 2019). An alternative approach is to sample particles in a locally-optimal manner and sample from the target distribution at each time point, $p(x_t|y_t, x_{t-1}, \theta)$. This results in new multiplicative weight terms at each time point (Equation (4.1)) that are uniformly distributed. This is typically referred to as the 'optimal' importance distribution (Doucet and Johansen, 2009) but is a function of the current observation and does not admit a tractable sampling distribution for general SSMs.

Several approaches have been proposed to approximate the optimal importance distributions, including via Gaussian approximations of the given SSM (Andrieu et al., 2003), deterministic optimisation-based approximations in annealing schemes (Donnet and Robin, 2017), and variational approximations of the posterior (He et al., 2023). A popular approach is the auxiliary particle filter (Carpenter et al., 1999; Pitt and Shephard, 1999, 2001) which approximates the optimal importance distribution for general SSMs. At each resampling step of

the SMC recursions, the auxiliary particle filter accounts for the current observation, often via a simulated approximation of the optimal importance distribution (Elvira et al., 2018). However, since the CSMC steps of a particle Gibbs algorithm are simply used to formulate proposal distributions for the latent states, the computational cost associated with the use of the auxiliary particle filter within each CSMC sweep of the MCMC algorithm can accumulate quickly. In this chapter, we propose novel optimal-type importance distributions using discrete HMM approximations of the SSM. In addition, we introduce deterministic HMM approximations that can be used to reduce computational cost in the particle Gibbs iterations and produce a computationally efficient approach.

4.3 Grid particle Gibbs with ancestor sampling

In this section, we introduce the proposed grid PGAS (GPGAS) algorithm. For general SSMs, the optimal PGAS importance densities are not available in closed form. We therefore propose general-use importance densities that use HMM approximations of the SSM, resulting in tractable approximations. In Step 1, we present the approximate HMM construction related to the approach of Chapter 3, and we introduce the novel tractable discrete approximations of the optimal importance distribution at each time point. In Step 2, we describe how particles can be sampled from the approximate discrete importance distributions and the CSMC-AS specification using these approximations for use within a PGAS algorithm (Algorithm 9).

4.3.1 Step 1: approximating the SSM with an HMM

We present the algorithm for one-dimensional state spaces and note that extensions to higher-dimensional spaces are often possible and are illustrated and discussed further in Sections 4.4 and 4.5. To approximate the SSM by a deterministic HMM, the state space is first partitioned into grid cells. That is, at each time point, we partition the state space, χ, into N intervals that cover the space with no overlap. For notational simplicity, we assume that the grid cells are the same for all time points and denote them by $I(n)$, $n = 1, \ldots, N$, for example in Figure 4.1, but this can be easily relaxed. The intervals form grid cells when the state space is partitioned for all time points, for example in Figure 4.2.

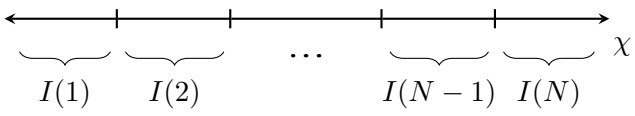


Figure 4.1: Example partition of the state space, χ, into N intervals for any time point t.

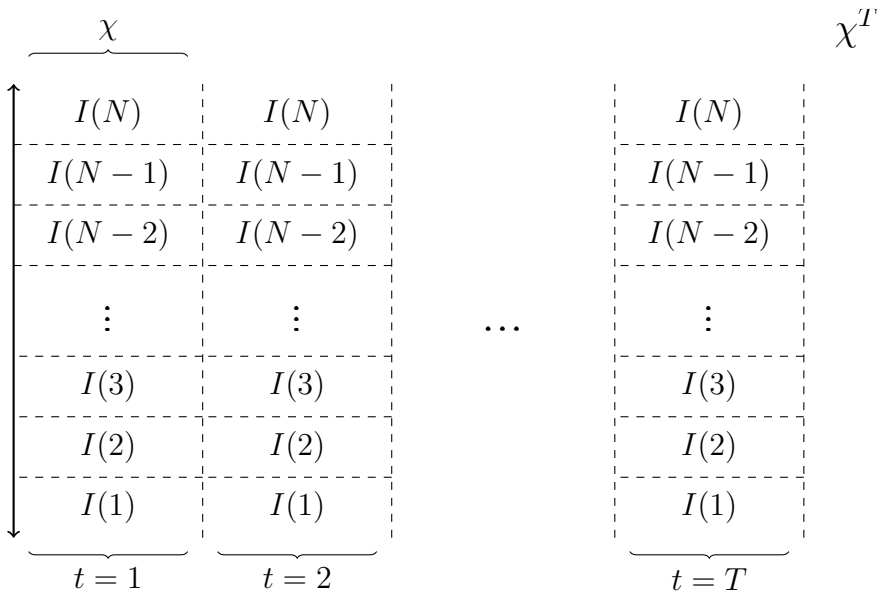


Figure 4.2: Example grid on χ^T formed by partitioning the state space at all time points into grid cells.

In this chapter, we assume that the non-infinite grid cells (Figure 4.2) are equally sized (i.e., the grid cells cover the same amount of the state space at each time point). Note that this is a special case of the grid cell construction defined in Chapter 3 (see, for example, Figures 3.2 and 3.4-3.6), which encompasses more complex grid structures. Additional grid cell definitions are described by Matoušek et al. (2020). However, here, extra attention may be required in relation to the additional computational cost and its trade-off with efficiency. This is discussed further in Section 4.5.

As in Chapter 3, we interpret the grid cell indices, $\{1, \ldots, N\}$, as the discrete states of an HMM, with dynamics defined by the SSM. Since, in this chapter, we define the grid cells in the same way for all time points, the HMM state transition

probability matrices (TPMs) are constant over time. Letting B_t denote the random variable of the grid cell indices at time t, we define the HMM as

Initial state probabilities:

$$P(B_1 = n|\theta) = \int_{I(n)} p(x_1|\theta)dx_1, \;\; n = 1, \ldots, N,$$

State transition probabilities:

$$P(B_t = n|B_{t-1} = k,\theta) = \int_{I(n)} \int_{I(k)} p(x_t|x_{t-1}, \theta)dx_{t-1}dx_t, \;\; k, n = 1, \ldots, N, t = 2, \ldots, T,$$

Observed state distribution:

$$p(y_t|B_t = n, \theta) = \int_{I(n)} p(y_t|x_t, \theta)dx_t, \;\; n = 1, \ldots, N, t = 1, \ldots, T. \tag{4.2}$$

In general, these HMM probabilities do not admit a closed-form expression. Thus, we apply a deterministic mid-point integration approach to approximate the HMM (as in Chapter 3). However, the grid cells are fixed (static) for all time points. We can therefore apply time-invariant deterministic approximations of the state probabilities. Let the length and mid-point of the n^{th} interval $I(n)$ be denoted by $L(n)$ and $\xi(n)$ respectively. We approximate the HMM in Equation (4.2) for states $k, n \in \{1, \ldots, N\}$ by

$$\begin{aligned} \hat{P}(B_1 = n|\theta) &\propto L(n)p(\xi(n)|\theta), \\ \hat{P}(B_t = n|B_{t-1} = k, \theta) &\propto L(n)L(k)p(\xi(n)|\xi(k), \theta), \;\; t = 2, \ldots, T, \\ \hat{p}(y_t|B_t = n, \theta) &\propto L(n)p(y_t|\xi(n), \theta), \;\; t = 1, \ldots, T, \end{aligned} \tag{4.3}$$

for grid cells indices $k, n = 1, \ldots, N$, similarly to Equation (3.2) but with constant state TPMs and grid cell lengths. Each of these probabilities is bounded to ensure that they are non-zero, and normalised over the grid cells so that they sum to one for each $t = 1, \ldots, T$.

4.3.2 Step 2: HMM importance distributions

We use the HMM approximation to formulate SMC importance distributions to improve particle distribution and sample impoverishment. We start by defining the following discrete approximation of the optimal importance distribution using the approximate HMM:

$$\hat{P}(B_1 = n|y_1, \theta) \propto \hat{P}(B_1 = n|\theta)\hat{p}(y_1|B_1 = n, \theta),$$
$$\hat{P}(B_t = n|y_t, B_{t-1} = k, \theta) \propto \hat{P}(B_t = n|B_{t-1} = k, \theta)\hat{p}(y_t|B_t = n, \theta), \quad t = 2, \ldots, T, \tag{4.4}$$

for grid cells indices $k, n = 1, \ldots, N$ since $\hat{P}(B_1 = n|y_1, \theta) \propto \hat{P}(B_1 = n, y_1|\theta)$ and $\hat{P}(B_t = n|y_t, B_{t-1} = k, \theta) \propto \hat{P}(B_t = n, y_t|B_{t-1} = k, \theta)$. At time t, we sample M grid cell indices (one for each SMC particle trajectory) from the associated discrete approximation. That is, at $t = 1$, we sample b_1^m for $m = 1, \ldots, M$ from

$$\{n, \hat{P}(B_1 = n|y_1, \theta)\}_{n=1}^N.$$

At time $t = 2, \ldots, T$, we sample b_t^m from

$$\{n, \hat{P}(B_t = n|y_t, B_{t-1} = b_{t-1}^m, \theta)\}_{n=1}^N,$$

for each $m = 1, \ldots, M$. Given a set of sampled grid cell indices at time t, $b_t^{1:M}$, we propose continuously-valued state particles, $x_t^{1:M}$, by sampling from within the grid cells associated with these indices. We therefore define continuous importance distributions over the space of each grid cell, similarly to the proposal distributions defined in Section 3.2.2. We assume that the importance distributions within each grid cell are defined independently of θ and the data, i.e., the importance distributions are of the form $q(x_t|B_t = n) = q(x_t|x_t \in I(n))$, for $n = 1, \ldots, N$. Examples of such within-cell distributions include uniform distributions for bounded grid cells and truncated Gaussian distributions for infinite grid cells.

At time t, sampling a grid cell from the optimal importance distribution conditional on the grid cell at time $t-1$ and sampling a particle from within the associated grid cell results in values b_t^m and x_t^m respectively. When repeated for the specified number of particles, we obtain a set of grid cells and particles, $\{b_t^m, x_t^m\}_{m=1}^M$, from the importance distributions:

$$q(x_1, B_1|y_1, \theta) = \hat{P}(B_1|y_1, \theta)q(x_1|B_1),$$
$$q(x_t, B_t|y_t, B_{t-1}, \theta) = \hat{P}(B_t|y_t, B_{t-1}, \theta)q(x_t|B_t), \quad \text{for } t = 2, \ldots, T. \tag{4.5}$$

In addition, we have that $q(x_1, B_1 = b_1|y_1, \theta) = q(x_1|y_1, \theta)$ and $q(x_t, B_t = b_t|y_t, B_{t-1} = b_{t-1}, \theta) = q(x_t|y_t, B_{t-1} = b_{t-1}, \theta)$ for all $t = 2, \ldots, T$ since we sample particles such that $x_t \in I(b_t)$ for all t. These distributions are

defined over the state space since the grid cells and within-cell distributions assign non-zero probability everywhere in the space.

Grid particle Gibbs with ancestor sampling

Within the CSMC-AS steps of the PGAS algorithm, the GPGAS algorithm samples grid cells and particles according to Equation (4.5), denoted $\{b_t^m, x_t^m\}_{m=1}^M$ for $t = 1, \ldots, T$. Thus, the SMC approximation of $p(x_{1:t} \mid y_{1:t}, \theta)$ under the proposed importance distribution is given by

$$\begin{aligned}
\hat{p}(x_{1:t}|y_{1:t}, \theta) &= \sum_{m=1}^{M} W_{1:t}^m \delta_{x_{1:t}^m}(x_{1:t}), \\
W_1^m &\propto \frac{p(x_1^m|\theta)p(y_1|x_1^m, \theta)}{q(x_1^m, B_1 = b_1^m|y_t, \theta)}, \\
W_{1:t}^m &\propto \frac{W_{1:t-1}^m p(x_t^m|x_{t-1}^m, \theta)p(y_t|x_t^m, \theta)}{q(x_t^m, B_t = b_t^m|y_t, x_{t-1}^m, \theta)}, \quad t = 2, \ldots, T,
\end{aligned} \tag{4.6}$$

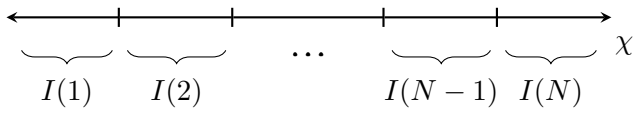
where $\delta_{x_{1:t}^m}(x_{1:t})$ is a Dirac mass at $x_{1:t} = x_{1:t}^m$ and $W_{1:t}^m$ is the weight associated with $(b_{1:t}^m, x_{1:t}^m)$, defined recursively. To use the proposed importance distribution within a CSMC-AS algorithm, we simply replace the importance distributions and weights in Algorithm 9 with those defined above, resulting in the GPGAS algorithm.

4.3.3 Summary

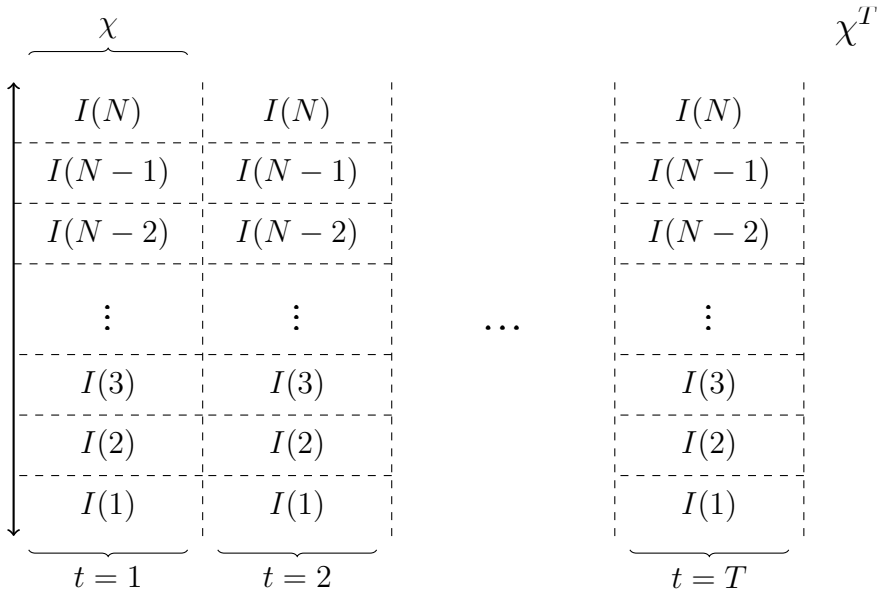
We provide a pictorial summary of the steps of the GPGAS algorithm with a one-dimensional state space at each time point. These pictorial representations are formulated to allow for comparison with the PMPMH summary in Section 3.2.3, and the CSMC and CSMC-AS algorithms in Figures 2.1 and 2.2. Pseudocode for the GPGAS algorithm is then presented in Algorithm 12.

Step 1(a): discretise the state space

Partition the state space at all T time points, $\chi_t = \chi$, $t = 1, \ldots, T$, into N equally-sized intervals:

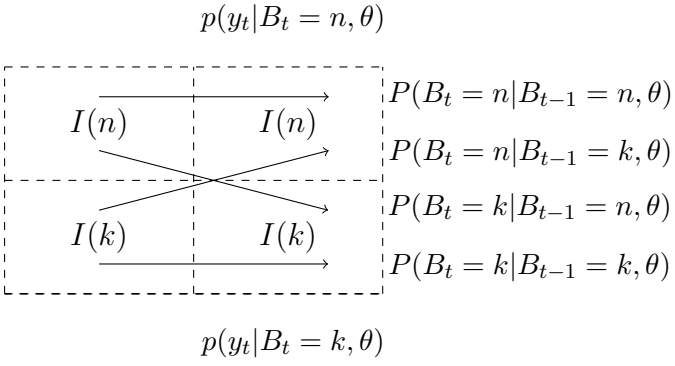


Form a grid by combining the partitions at all time points $t = 1, \ldots, T$:



Step 1(b): approximate an HMM in the grid cell indices

For grid cell index random variables, $B_t \in \{1, \ldots, N\}$ at time t, define an HMM in terms of the underlying SSM. For $n, k \in \{1, \ldots, N\}$:



Integrating the SSM over the given intervals gives the HMM probabilities:

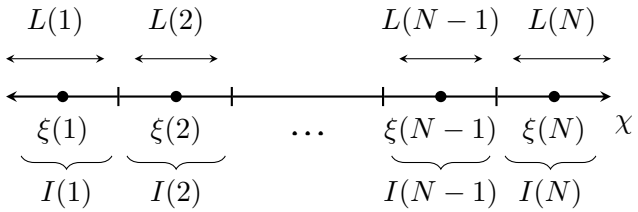
$$P(B_1 = n|\theta) = \int_{I(n)} p(x_1|\theta)dx_1,$$

$$P(B_t = n|B_{t-1} = k, \theta) = \int_{I(n)} \int_{I(k)} p(x_t|x_{t-1}, \theta)dx_{t-1}dx_t, \;\; t = 2, \ldots, T,$$

$$p(y_t|B_t = n, \theta) = \int_{I(n)} p(y_t|x_t, \theta)dx_t, \;\; t = 1, \ldots, T,$$

where the HMM transition probabilities are the same for all time points, i.e., the TPM is constant over time.

To approximate the HMM, find the length and mid-point of the n^{th} interval $I(n)$, denoted by $L(n)$ and $\xi(n)$ respectively, for example:



Approximate the HMM via mid-point integration using

$$\begin{aligned}
\hat{P}(B_1 = n|\theta) &\propto L(n)p(\xi(n)|\theta), \\
\hat{P}(B_t = n|B_{t-1} = k, \theta) &\propto L(n)L(k)p(\xi(n)|\xi(k), \theta), \;\; \text{for all } t = 2, \ldots, T, \\
\hat{p}(y_t|B_t = n, \theta) &\propto L(n)p(y_t|\xi(n), \theta), \;\; t = 1, \ldots, T.
\end{aligned}$$

These probabilities are normalised, bounded to ensure that they are non-zero and then re-normalised to sum to one.

Step 2(a): formulate HMM importance distributions
Calculate

$$\hat{P}(B_1|y_1, \theta) \propto \hat{P}(B_1|\theta)\hat{p}(y_1|B_1, \theta),$$

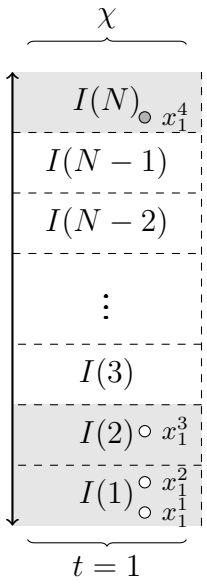
$$\hat{P}(B_t|y_t, B_{t-1}, \theta) \propto \hat{P}(B_t|B_{t-1}, \theta)\hat{p}(y_t|B_t, \theta).$$

Step 2(b): sample grid cells and particles at time t = 1

Fix the M^{th} particle to the current MCMC latent state value at time $t = 1$ (CSMC), i.e., $x_1^M = x_1^{(s-1)}$. For the other particles:

- Sample grid cell indices, $b_1^m \sim \hat{P}(B_1|y_1, \theta)$, $m = 1, \ldots, M-1$,
- and particles, $x_1^m \sim q(x_1|B_1 = b_1^m)$, $m - 1, \ldots, M-1$.

For example, for $M = 4$ particles, suppose that the sampled grid cells for the first three particles are $b_1^{1:3} = (1, 1, 2)$, and the corresponding state values within each grid cells are selected as $x_1^1, x_1^2 \sim \text{TN}_{I(1)}(\xi(1), \sigma^2)$ (a truncated Gaussian distribution over $I(1)$ with mean $\xi(1)$ and variance σ^2) and $x_1^3 \sim \text{Uniform}(I(2))$. If $x_1^4 \in I(N)$ (the reference particle):



Step 2(c): weight the particles at time t = 1

Calculate weights using

$$W_1^m \propto \frac{p(x_1^m|\theta)p(y_1|x_1^m, \theta)}{q(x_1^m, B_1 = b_1^m|y_t, \theta)}, \quad m = 1, \ldots, M,$$

where $q(x_1, B_1|y_t, \theta) = \hat{P}(B_1|y_1, \theta)q(x_1|B_1)$ and b_1^M is such that $x_1^M \in I(b_1^M)$.

Step 2(d): sample grid cells and particles at time t = 2, . . . , T

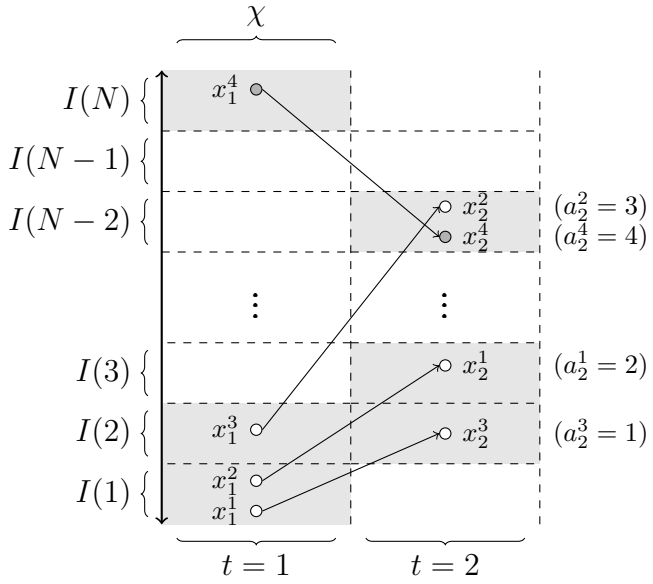
Fix $x_t^M = x_t^{(s-1)}$ (CSMC). For the other particles:

- Resample $a_t^m \sim r(a_t|W_{1:t-1}^{1:M})$, $m = 1, \ldots, M-1$ (for example, using a multinomial distribution).

- Sample grid cell indices, $b_t^m \sim \hat{P}(B_t|y_t, B_{t-1} = b_{t-1}^{a_t^m}, \theta)$, $m = 1, \ldots, M-1$, where $b_{t-1}^{a_t^m}$ is such that $x_{t-1}^{a_t^m} \in I(b_{t-1}^{a_t^m})$.

- Sample particles, $x_t^m \sim q(x_t|B_t = b_t^m)$, $m = 1, \ldots, M-1$ (independent proposal distributions).

For $M = 4$ particles and $t = 2$, the fixed reference particle, resampling, and sampled grid cells and particles may correspond to:



Note that the fixed reference particle, x_2^4, has a value such that $x_2^4 \in I(N-2)$, with ancestor here fixed to the reference particle at time $t = 1$, i.e., $a_2^4 = 4$.

Step 2(e): sample an ancestor and weight the particles at time $\mathbf{t = 2, \ldots, T}$
Sample the ancestor of the reference particle, a_t^M, (CSMC-AS) using the weights:

$$\tilde{W}_t^m \propto W_{1:t-1}^m p(x_t^M|x_{t-1}^m, \theta), \;\; m = 1, \ldots, M.$$

Calculate weights for each particle using

$$W_{1:t}^m \propto \frac{W_{1:t-1}^m p(x_t^m|x_{t-1}^m, \theta) p(y_t|x_t^m, \theta)}{q(x_t^m, B_t = b_t^m|y_t, x_{t-1}^m, \theta)}, \;\; m = 1, \ldots, M,$$

where $q(x_t, B_t|y_t, x_{t-1}, \theta) = \hat{P}(B_t|y_t, B_{t-1}, \theta) q(x_t|B_t)$, b_t^M is such that $x_t^M \in I(b_t^M)$.

Repeat Steps 2(d) and 2(e) recursively until time $t = T$.

If $M = 4$ and $T = 3$, then the CSMC-AS steps may result in the following particle trajectories, with the re-assigned reference trajectory history shown by the dashed arrows:

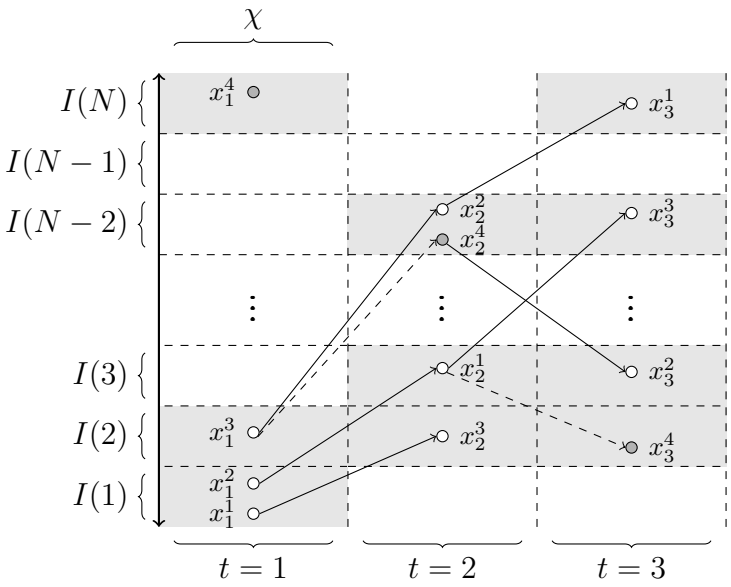


Figure 4.3: Illustration of the CSMC-AS steps of the GPGAS algorithm with $M = 4$ particles and $T = 3$ time points. The dotted partitions form grid cells, labelled by their associated interval, and the sampled grid cells are shown in lighter grey. Each node represents a particle within each grid cell, and the fixed reference trajectory is shown by the nodes in darker grey. The reference trajectory history is re-assigned in the ancestor sampling steps, shown by the dashed arrows. For example, the ancestor of the reference particle x_2^4 is sampled as $a_2^4 = 3$, thus the ancestor particle is x_1^3, and the ancestor of the reference particle x_3^4 is sampled as $a_3^4 = 1$, thus the ancestor particle is x_2^1.

Step 2(f): update the latent states and model parameters

Update the latent states by sampling particles, $x_{1:T}^{(s)}$, from $\{x_{1:T}^m, W_{1:T}^m\}_{m=1}^M$. Then, update the model parameters, θ, conditional on the latent states conditionally using standard M-H or Gibbs updates.

Figure 4.3 highlights the similarities and differences between the algorithms presented in this thesis. Firstly, note that grid cells are imposed similarly to the PMPMH algorithm (Figure 3.3), but the grid cells are equally sized and the GPGAS algorithm proposes multiple grid cells and particles at each time point according to a CSMC-AS algorithm. The nodes and particle trajectories in the GPGAS Figure 4.3 are comparable to the nodes in the CSMC-AS algorithm illustration in Figure 2.2. However, the GPGAS algorithm presents the particles relative to their location in the state space due to the imposition of the grid cells.

Algorithm 12 Grid particle Gibbs with ancestor sampling (GPGAS)

Input:

- M = number of particles
- N = number of grid cells
- S = number of MCMC iterations
- $\{q(x_t|x_t \in I(n))\}_{n=1}^N$, $t = 1, \ldots, T$ = importance distributions
- $x_{1:T}^{(0)}$, $\theta^{(0)}$ = initial values
- a Gibbs or Metropolis-Hastings sampling scheme to update θ from $p(\theta|x_{1:T}, y_{1:T})$
- A grid with indices $B_{1:T}$

for $s = 1, \ldots S$ **do**

Approximate HMM (Step 1)

calculate $\hat{p}(y_1|B_1 = n, \theta^{(s-1)})$, $\hat{P}(B_1 = n|\theta^{(s-1)})$, $n = 1, \ldots, N$ ▷ Step 1(b)

calculate $\hat{P}(B_t = n|B_{t-1} = k, \theta^{(s-1)})$, $k, n = 1, \ldots, N$

for $t = 2, \ldots, T$ **do**

calculate $\hat{p}(y_t|B_t = n, \theta^{(s-1)})$, $n = 1, \ldots, N$

Formulate importance distributions (Step 2)

calculate $\hat{P}(B_1 = n|y_1, \theta^{(s-1)})$, $n = 1, \ldots, N$ ▷ Step 2(a)

for $t = 2, \ldots, T$ **do**

calculate $\hat{P}(B_t = n|y_t, B_{t-1} = k, \theta^{(s-1)})$, $k, n = 1, \ldots, N$

▷ Step 2(b)-(e)

Run a PGAS step (Algorithm 9) with M particles, $\theta = \theta^{(s-1)}$, $x_{t-1}^{(s-1)}$, and importance distributions $q(x_1, B_1|y_1, \theta^{(s-1)})$, $\{q(x_t, B_t|y_t, B_{t-1}, \theta^{(s-1)})\}_{t=2}^T$ defined in Equation (4.5)

sample $x_{1:T}^{(s)}$ from $\{x_{1:T}^m, W_T^m\}_{m=1}^M$ ▷ Step 2(f)

update $\theta^{(s)}$ from $p(\theta|x_{1:T}^{(s)}, y_{1:T})$

return $\{x_{1:T}^{(s)}, \theta^{(s)}\}_{s=1}^S$ approximating $p(x_{1:T}, \theta|y_{1:T})$

4.3.4 Computational and practical considerations

In this section, we describe the associated computational and practical aspects of the GPGAS algorithm that affect its efficiency. We first note some important computational strategies that can be implemented universally, independent of the model considered:

1. As illustrated in Algorithm 12, only one approximate TPM needs to be calculated per MCMC iteration since the grid cells are the same for all time points.

   In contrast to the time-invariant HMM approximation, a time-dependent HMM approximation can be derived, as in Chapter 3. In this case, the exact values of the particles at the previous time point could be used to approximate the transition probabilities. That is, we may formulate a proposal distribution of the form $q(x_t, B_t|y_t, x_{t-1}, \theta)$ (replacing Equation (4.5)) using a TPM of the form $\hat{P}(B_t|y_t, x_{t-1}, \theta)$ (replacing Equation (4.3)). Although this may improve the accuracy of the HMM approximation, additional transition probability calculations are required (for each unique particle and each time point). Thus, the potentially improved mixing properties need to be balanced with the associated increase in computational cost.

2. The discrete approximations to the optimal importance distributions at time $t \geq 2$, only need to be calculated for grid cells containing particles at the previous time point. This computational strategy reduces the computational cost of each GPGAS iteration from $\mathcal{O}(N^2T)$ to $\mathcal{O}(N^2+N\sum_{t=2}^{T}\tilde{N}_{t-1})$, where $\tilde{N}_{t-1}$ denotes the number of grid cells containing particles at time $t-1$ of the GPGAS iteration. For the stochastic volatility with leverage example in Section 4.4.1, this computational strategy reduces the computational cost of the first 1000 MCMC iterations by around 50% for $N = 100$ grid cells and $M = 100$ particles.

3. Given the sampled grid cells at time t, many of the particles at time t are identically distributed according to the importance distributions within each grid cell. Thus, we can sample multiple particles from the same importance distribution simultaneously to reduce computational cost.

Aside from the computational adjustments that can be made universally, there are model-dependent practical considerations, particularly with respect to how the grid cells are defined. We provide general guidance in relation to these. We note that for the examples considered in Section 4.4, performance was robust

within these general guidelines and efficient decisions were made in relation to each point, where appropriate, using crude pilot tuning over a small number of MCMC iterations.

a) The computational cost of the HMM approximations across iterations (Equation (4.3)) can be reduced by fixing the approximations after a given number of iterations. We consider that a sensible approach assumes the posterior mean estimates for the parameters are sufficiently stable when calculated using several GPGAS iterations past a certain number of iterations, $\tilde{s}$. Thus, we fix the HMM approximations using the parameter mean estimates of several samples after iteration $\tilde{s}$. We note that this value should be chosen to balance the reduction in computational cost with the accuracy of the importance distributions.

b) To ensure that any value in the state space with reasonable posterior mass can be proposed, the majority of finite grid cells should be set to ensure that areas of the state space with significant probability are covered. Excessively large ranges should be avoided to ensure that computational cost is not spent in effectively zero-density areas of the state space.

c) We sample particles within each grid cell using standard (and computationally inexpensive) distributions, for example, uniform distributions in the finite grid cells and truncated Gaussian distributions in the outer (infinite) grid cells. In the implementations of Section 4.4, we parameterised the truncated Gaussian distributions by setting their mean equal to the 'mid-point' of the associated grid cell, defined at a distance from the finite boundary equal to the distance between the mid-points and finite boundaries in the finite grid cells.

4.3.5 Toy example

We demonstrate the steps of the GPGAS algorithm, and the computational and practical considerations, using a toy example. This illustration uses the same SSM and simulated data as the toy examples in Section 3.3.4 so that the approaches can be compared:

$$\begin{aligned} x_1 &\sim N(x_0, \sigma_\eta^2), \\ x_t &\sim N(x_{t-1}, \sigma_\eta^2), \quad t = 2, 3, \\ y_t &\sim N(a x_t, \sigma_\epsilon^2), \quad t = 1, 2, 3. \end{aligned}$$

where, $\sigma_\eta^2, \sigma_\epsilon^2 > 0$ are the state and observation process variance respectively, a is an observation scaling parameter and $\theta = (x_0, a, \sigma_\eta^2, \sigma_\epsilon^2)$ denotes the set of model parameters collectively. The data are simulated using a random number seed of 1234, `set.seed(1234)` in R, and $\theta = (1, 2, 0.5, 1)$, resulting in simulated data $y_{1:3} = (-2.052746, 1.114420, 2.724983)$.

Set up We focus on the conditional state updates, $p(x_{1:T}|y_{1:T}, \theta)$, in a single MCMC iteration, s, as part of an MCMC algorithm targeting the joint distribution $p(x_{1:T}, \theta|y_{1:T})$. We use $N = 6$ grid cells and $M = 4$ particles for illustrative purposes, and we assume that the current M-H values for the latent states and model parameters are $x_{1:T}^{(s-1)} = (0.01, 0.16, 1.45)$ and $\theta^{(s-1)} = (-0.54, 0.66, 0.35, 0.67)$ respectively. Note that, when applying the GPGAS algorithm over several MCMC iterations, rather than a single iteration, the grid cells can also be fixed after a specified number of iterations as in computational point (a) in Section 4.3.4.

In Step 1 (Step 1(a) in Section 4.3.3), we partition the state space into equally-sized grid cells at each time point. To ensure areas of the state space with significant posterior mass are approximated, we aim to locate these regions using the observation distribution and current parameter samples ($y/a^{(s-1)}$ where $a^{(s-1)}$ is the current MCMC sample for a). This indicates a potential range for the finite grid cells of approximately $[-3.11, 4.13]$. We extend this and set the finite cells in $[-8, 8]$ to ensure coverage of the state space, resulting in the equally-sized grid cells in Figure 4.4. In practice, the finite cell range can be set using pilot tuning.

Under this definition for the grid cells, we approximate the HMM (Step 1(b) in Section 4.3.3) in the grid cell indices $B_t \in \{1, \ldots, 6\}$. To approximate the HMM using mid-point integration, we artificially set the length of the outer, infinite cells equal to the length of the finite cells (4), and the mid-point half that length from the finite boundary. We then calculate the HMM in the grid cell indices using Equation (4.3). That is, for transitions from grid cell 1 to 2,

$$\hat{P}(B_t = 2|B_{t-1} = 1, \theta^{(s-1)}) \propto 4 \times 4 \times p(dx_t = -6|dx_{t-1} = -10, \theta^{(s-1)}),$$

for $t = 2, 3$, approximately 1.3×10^{-9}. After calculating similar probabilities for all $B_t \in \{1, ..., 6\}$, we then normalise to sum to one, bound the probabilities to ensure that they are non-zero (here by 1×10^{-10} due to the large range of the finite cells), and renormalise. This results in $\hat{P}(B_t|B_{t-1} = 1, \theta^{(s-1)}) = (1.00, 1.2 \times 10^{-10}, 1.0 \times 10^{-10}, 1.0 \times 10^{-10}, 1.0 \times 10^{-10}, 1.0 \times 10^{-10})$ approximately. We similarly

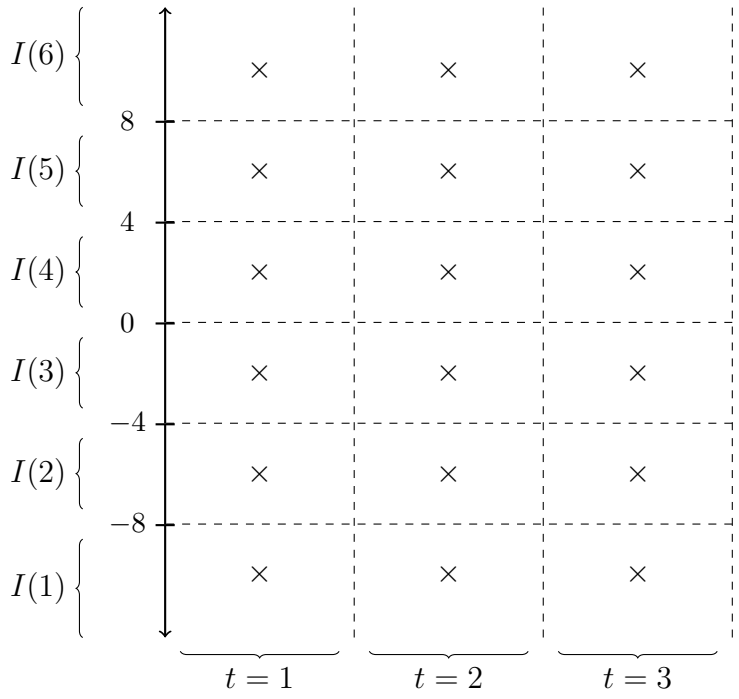


Figure 4.4: GPGAS grid cells for the toy example with data $y_{1:3}$. The grid cell boundaries are defined for each time point at $-8, -4, 0, 4, 8$. Each grid cell is labelled by the corresponding interval. Mid-points within each grid cell are shown by crosses.

approximate the initial state and observation probabilities of the HMM using Equation (4.3), and the TPMs are constant over time since the grid cells are the same for all time points. We therefore calculate one set of $1 \times N$ (1×6) initial state probabilities, one $N \times N$ (6×6) TPM and one $N \times T$ (6×3) observation matrix at iteration s.

In Step 2 (Step 2(a) in Section 4.3.3), we formulate the HMM importance distributions in Equation (4.4). For example,

$$\hat{P}(B_1|y_1, \theta^{(s-1)}) \propto \hat{P}(B_1|\theta^{(s-1)})\hat{P}(y_1|B_1, \theta^{(s-1)}),$$

approximately $(2.9 \times 10^{-17}, 9.9 \times 10^{-12}, 1.00, 6.8 \times 10^{-12}, 1.1 \times 10^{-20}, 1.1 \times 10^{-20})$. We then fix the reference trajectory in a PGAS step as in Step 2(b) and sample the remaining particles. First, we sample 3 grid cells ($M - 1$, one for each remaining particle), from the HMM distribution. For example, we may sample $b_1^{1:3} = (3, 4, 3)$ from $\hat{P}(B_1|y_1, \theta^{(s-1)})$. We then sample a particle from within each associated grid cell: for $m = 1, 3$, $x_1^m \sim \text{Uniform}(I(3))$, i.e., $x_1^m \sim \text{Uniform}(-4, 0)$, for $m = 2$, $x_1^m \sim \text{Uniform}(0, 4)$. Noting the computational points in Section 4.3.4, we can sample x_1^1 and x_1^3 simultaneously from the same distribution. The particles are

then weighted as in the PGAS steps of Equation (4.6) (and Step 2(c) in Section 4.3.3).

At time $t = 2$ (Step 2(d) in Section 4.3.3), we first fix the reference particle and resample the remaining particles according to the weights. We then sample grid cells and particles conditional on these resampled particles. For example, if the resampling steps result in ancestors to the particles at time $t = 2$, $a_2^{1:3} = (4, 4, 2)$, we sample grid cell indices, b_2^m from $\hat{P}(B_2|y_t, B_1 = b_1^{a_2^m}, \theta^{(s-1)})$ for $m = 1, 2, 3$:

$$\begin{aligned}
b_2^1 \quad &\text{from} \quad \hat{P}(B_2|y_2, B_1 = b_1^{a_2^1}, \theta^{(s-1)}) = \hat{P}(B_2|y_2, B_1 = 4, \theta^{(s-1)}) \\
b_2^2 \quad &\text{from} \quad \hat{P}(B_2|y_2, B_1 = b_1^{a_2^2}, \theta^{(s-1)}) = \hat{P}(B_2|y_2, B_1 = 4, \theta^{(s-1)}) \\
b_2^3 \quad &\text{from} \quad \hat{P}(B_2|y_2, B_1 = b_1^{a_2^3}, \theta^{(s-1)}) = \hat{P}(B_2|y_2, B_1 = 4, \theta^{(s-1)}),
\end{aligned}$$

we therefore sample all $b_2^{1:3}$, from $\hat{P}(B_2|y_2, B_1 = 4, \theta^{(s-1)}) = (1.0 \times 10^{-20}, 4.7 \times 10^{-19}, 1.5 \times 10^{-12}, 1.00, 2.9 \times 10^{-13}, 1.8 \times 10^{-20})$ (Equation (4.4)). Note that, as in computational point 2 in Section 4.3.4, the HMM importance distribution is only calculated for $B_1 = 4$, rather than for all grid cells at $t = 1$. Given a sample $b_2^{1:3} = (4, 4, 4)$, we then sample particles simultaneously such that $x_2^m \sim \text{Uniform}(I(4))$, i.e., $x_2^m \sim \text{Uniform}(0, 4)$, for $m = 1, 2, 3$. Finally, we sample the reference particle ancestor, a_2^4, in the ancestor sampling steps, and weight the particles using Equation (4.6) (and Step 2(e) in Section 4.3.3). Once the GPGAS recursions have been completed for all time points, (up to $t = 3$), we sample a new MCMC value for the latent state, $x_{1:3}^{(s)}$ from $\{x_{1:3}^m, W_{1:3}^m\}_{m=1}^4$.

4.4 State-space models with regime switching

We investigate the performance of the GPGAS algorithm when applied to the challenging case of SSMs with regime switching. Regime-switching SSMs allow the observation or transition models of an SSM to change abruptly between a set of discrete 'regimes', labelled $1, \ldots, K$. At each time point, the choice of regime determines the observation and transition model. However, the regime label at time t, denoted $s_t \in \{1, \ldots, K\}$, is unobserved and is assumed to be first-order Markovian. A regime-switching SSM can be described mathematically as a standard SSM (Equation (1.1)) with an additional discrete latent process of regime labels, $s_{1:T} = (s_1, \ldots, s_T)$, determining the system and observation process parameters:

$$
\begin{aligned}
\textit{Initial state distribution}: &\quad p_{s_1}(x_1|\theta^{s_1}),\\
\textit{State transition distribution}: &\quad p_{s_t}(x_t|x_{t-1},\theta^{s_t}),\ \ t=2,\ldots,T,\\
\textit{Observed state distribution}: &\quad p_{s_t}(y_t|x_t,\theta^{s_t}),\ \ t=1,\ldots,T,
\end{aligned}
$$

$$
\begin{aligned}
\textit{Initial regime probabilities:} &\quad P(s_1=j)=\pi_j^0,\\
\textit{Regime transition probabilities}: &\quad P(s_t=j|s_{t-1}=i)=\pi_{ij},\ \ t=2,\ldots,T,
\end{aligned}
$$

where $\{\theta^1,\ldots,\theta^K\}$ denote the model parameters associated with each regime, the probability density functions can vary across regimes, and π_j^0 and π_{ij} denote the initial probabilities for each regime and transition probabilities between regimes with labels $i,j\in\{1,\ldots,K\}$ respectively. For further discussion see, for example, Kim and Nelson (1999).

Regime-switching SSMs have several well-known applications, including tracking manoeuvring targets (Bar-Shalom et al., 2001; Karlsson and Bergman, 2000; Liang-qun et al., 2009) and modelling economic and financial data (Frühwirth-Schnatter, 2001; Hamilton, 1989; Kim and Nelson, 1999; Kim and Cho, 2022). Despite their widespread application, computational methods for fitting the latent states and model parameters of a regime-switching SSM can be inefficient. Since the SSM is non-linear, a natural approach often applied is particle Gibbs sampling. However, standard SMC steps within the particle Gibbs algorithm are known to degenerate when the state switches, requiring many particles and a high computational cost to combat sample impoverishment (Doucet et al., 2001; Driessen and Boers, 2004). Other approaches, including the use of multiple SMC implementations across regimes (the interacting multiple models filter; Blom and Bar-Shalom (1988); Boers and Driessen (2003); McGinnity and Irwin (2000)) and the auxiliary particle filter (Andrieu et al., 2003), can require a high computational cost to mix sufficiently when the state switches (El-Laham et al., 2020; Elvira et al., 2018).

Various strategies have been proposed to merge deterministic techniques with importance sampling to combat sample impoverishment for regime-switching SSMs, including the deterministic allocation of particles to regimes heuristically or using posterior model probability approximations (El-Laham et al., 2020; Martino et al., 2017; Urteaga et al., 2016). The GPGAS algorithm is intuitively similar to these approaches but provides an approach to joint state and parameter inference.

We investigate the performance of the proposed GPGAS algorithm when applied to two classes of regime-switching model. In the first example of this section, we focus on a simulated stochastic volatility model with regime-switching to investigate the performance of the GPGAS algorithm relative to several current efficient approaches: the PGAS algorithm using both the bootstrap and auxiliary particle filters (Lindsten et al., 2014; Pitt and Shephard, 1999), and the PMPMH algorithm proposed in Chapter 3. In this example, we also explore the performance of each method under two different model parameterisations that are known to impact the efficiency of traditional SMC methods, understanding some of the settings in which each algorithm can be applied efficiently. The second example then applies the best-performing algorithms in the previous example to a challenging real-world regime-switching model for COVID-era tourism demand in Edinburgh. This section serves as an extended case study, demonstrating a practical and efficient implementation of the GPGAS algorithm on a new regime-switching model for COVID-era tourism demand.

4.4.1 Stochastic volatility with leverage

We initially focus on the model for stochastic volatility described by Kim (2015) and So et al. (1998). In this model, a two-state regime process, denoted $s_{1:T} = (s_1, \ldots, s_T)$, $s_t \in \{1, 2\}$ for each t, captures switching in the level of U.S. stock market log volatility over time. Transitions from the first and second regimes occur with probability π_{12} and π_{21} respectively, and the regime labels, $s_{1:T}$, each correspond to a parameter, γ_1 or γ_2. The level of the latent log volatility process, $x_{1:T}$, is a function of these parameters and determines the variance of the observations, $y_{1:T}$. Mathematically, this stochastic volatility SSM with regime switching can be written as

State transition distribution:

$$x_t = \gamma_{s_t} + \phi(x_{t-1} - \gamma_{s_{t-1}}) + \eta_t, \quad \eta_t \sim N(0, \sigma_\eta^2),$$

Observed state distribution:

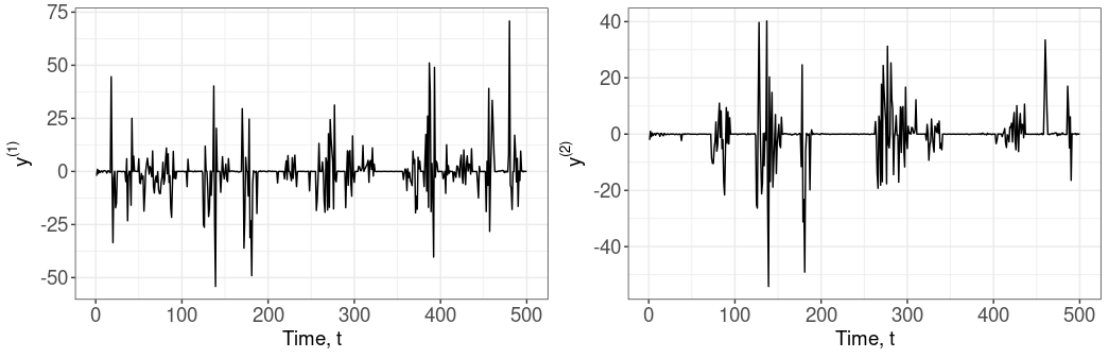
$$y_t = \exp\left(\frac{x_{t-1}}{2}\right)\epsilon_t, \quad \epsilon_t \sim N(0, 1),$$

Regime transition probabilities:

$$P(s_t = j \mid s_{t-1} = i) = \pi_{ij}, \quad i, j \in \{1, 2\},$$

for $t = 1, \ldots, T$ where ϕ is an autoregressive scaling parameter and $\sigma_\eta^2 > 0$ is the system process variance. The observation variance parameter is fixed at 1 to focus exploration on the latent processes and associated parameters. In addition, the initial continuous latent state is defined as $x_0 = \mu$, the initial regime label is $s_0 = 1$, and the expected duration in each regime is the same, i.e, $\pi_{22} = \pi_{11}$. We consider that $x_{1:T}$, $s_{1:T}$, and the set of model parameters, $\theta = (\gamma_1, \gamma_2, \phi, \sigma_\eta^2, \mu, \pi_{11})$, are unknown.

The duration of the regimes (persistence) is determined by π_{11} and can influence how well an SMC algorithm approximates the posterior distribution of the latent states. In general, degeneracy rates increase when the true state switches. Thus, increasing the number of states that switch generally increases degeneracy rates and reduces the accuracy of the SMC approximation of the posterior distribution of the latent states. We therefore investigate the performance of the proposed GPGAS algorithm under different levels of regime persistence, resulting in different sets of simulated data: $y_{1:T}^{(1)}$, simulated using $\pi_{11} = 0.85$, and $y_{1:T}^{(2)}$, using $\pi_{11} = 0.95$ (reducing the number of state switches). Each data set is simulated using $T = 500$ time points and remaining model parameters $\theta = (\gamma_1, \gamma_2, \phi, \sigma_\eta^2, \mu) = (-5, 5, 0.95, 0.1, 1)$. We present each set of simulated data in Figure 4.5.



(a) (b)

Figure 4.5: Simulated data from the stochastic volatility model: $y_{1:T}^{(1)}$ using $\pi_{11} = 0.85$, $T = 500$, and $y_{1:T}^{(2)}$ using $\pi_{11} = 0.95$, $T = 500$.

The associated (independent) priors are given for both data sets by

$$\begin{aligned}
\gamma_1 &\sim N(-5, 10), \\
\gamma_2 &\sim N(5, 10), \\
\phi &\sim N(0.95, 1), \\
\sigma_\eta^2 &\sim \text{InvGamma}(2.01, 0.101), \\
\mu &\sim N(1, 1), \\
\pi_{11} &\sim \text{Beta}(9.9875, 1.7625),
\end{aligned}$$

where InvGamma denotes an inverse gamma distribution and the Gaussian distributions are parameterised by their variance. Each unknown model parameter is sampled in the same way for each model parameterisation using conditional Gibbs updates.

Computational decisions

We detail the exact computational and practical decisions required to implement the GPGAS algorithm. Since the regime labels are discrete, we use the regime labels to determine grid cells in the discrete state space. The following relates to the grid cells used in the space of the continuous states, χ.

We first note that the GPGAS algorithm is implemented using the computational strategies in points (1-3) of Section 4.3.4. The practical choices with respect to points (a-c) in Section 4.3.4 are as follows:

a) In all implementations, we fix the HMM approximations using the posterior mean of each parameter estimated using 1000 samples after iteration $\tilde{s} = 2000$. These values were found to be reasonable from short pilot tuning runs.

b) Using these pilot tuning runs, we establish that a range of $[-12, 12]$ for the finite grid cells ensures that any value in the state space with reasonable posterior mass can be proposed.

c) To sample within each grid cell, we use uniform and truncated Gaussian distributions as described in Section 4.3.4, and note that the truncated Gaussian distributions have variance 2.4 (10% of the finite grid cell range).

Results

We present the results for the GPGAS, PGAS, PGAS with the auxiliary particle filter, and PMPMH algorithms with various numbers of grid cells and particles. A resampling threshold of 25% of the effective sample size in the SMC recursions is universally favourable for both the GPGAS and PGAS algorithms in this case. Each implementation is executed 10 times for 10,000 iterations on one core and a 1.6 GHz CPU and we compare the performance of each implementation to 'ground truth' runs. These ground truth runs consist of the PGAS algorithm with $M = 5000$ particles, taking around 89 hours to complete 10,000 iterations under both sets of simulated data, $y_{1:T}^{(1)}$ and $y_{1:T}^{(2)}$.

The auxiliary particle filter implementation uses simulation of the auxiliary weights and requires a large computational cost for sufficiently accurate approximation of the auxiliary weights to prevent sample impoverishment: at least 30 particles to approximate each auxiliary weight and 50 particles in the CSMC-AS recursions, taking around 3 hours to reach errors comparable to the cheapest standard PGAS implementation. Similarly, the PMPMH algorithm requires the calculation of many TPMs, and a large computational cost, to achieve comparable errors in the posterior estimates. Thus, we focus the remaining results on the PGAS algorithm and proposed GPGAS algorithm and present the results in Tables 4.1 and 4.2 and Figures 4.6 and 4.7.

PGAS					
	M	**RRMSE mean**	**RRMSE var**	**ESS**	**Time (s)**
	≤ 25	-	-	-	-
	50	0.09	4.03	4200	3600
	100	0.05	2.39	4400	6600
	200	0.03	1.30	4500	12000
GPGAS					
N	**M**	**RRMSE mean**	**RRMSE var**	**ESS**	**Time (s)**
25	≤ 25	-	-	-	-
	50	0.03	0.17	4800	4200
	100	0.03	0.13	4900	5900
	200	0.02	0.10	4900	9400
50	≤ 10	-	-	-	-
	25	0.04	0.14	4600	3900
	50	0.02	0.10	4800	5300
	100	0.01	0.10	4900	6900
	200	0.01	0.10	5800	10700
100	≤ 10	-	-	-	-
	25	0.04	0.12	4700	6500
	50	0.02	0.10	5500	7800
	100	0.01	0.09	5600	8900
	200	0.01	0.09	6000	12800
200	≤ 10	-	-	-	-
	25	0.04	0.12	5700	8500
	50	0.02	0.10	6200	13500
	100	0.01	0.09	6200	18000
	200	0.01	0.08	6300	24000

Table 4.1: Relative root mean squared errors in the posterior mean (RRMSE mean) and posterior variance (RRMSE var) of the continuous latent states, $x_{1:T}$, effective sample size (ESS), and computational time (Time (s)). Shown for implementations of the PGAS and GPGAS algorithms under $y_{1:T}^{(1)}$ (simulated at with $\pi_{11} = 0.85$) with N grid cells and M particles, grouped by computational times comparable to the PGAS implementations. Dashes indicate non-convergent implementations.

PGAS					
	M	**RRMSE mean**	**RRMSE var**	**ESS**	**Time (s)**
	≤ 10	-	-	-	-
	25	0.18	4.08	3500	2000
	50	0.12	3.48	4700	3400
	100	0.07	2.53	4800	6000
	200	0.03	1.58	5500	11500
GPGAS					
N	**M**	**RRMSE mean**	**RRMSE var**	**ESS**	**Time (s)**
25	≤ 10	-	-	-	-
	25	0.05	0.35	3700	3800
	50	0.03	0.16	5000	6000
	100	0.02	0.13	5400	9200
	200	0.01	0.10	5900	10200
50	≤ 10	-	-	-	-
	25	0.02	0.14	5300	3900
	50	0.02	0.10	5500	6300
	100	0.02	0.08	5800	9200
	200	0.01	0.08	6100	12200
100	≤ 10	-	-	-	-
	25	0.02	0.12	5600	5100
	50	0.01	0.10	5800	6800
	100	0.01	0.08	5800	9200
	200	0.01	0.08	6400	13000
200	≤ 10	-	-	-	-
	25	0.02	0.10	5700	8900
	50	0.01	0.10	6300	11600
	100	0.01	0.09	6400	18700
	200	0.01	0.07	6400	23900

Table 4.2: Relative root mean squared errors in the posterior mean (RRMSE mean) and posterior variance (RRMSE var) of the continuous latent states, $x_{1:T}$, effective sample size (ESS), and computational time (Time (s)). Shown for implementations of the PGAS and GPGAS algorithms under $y_{1:T}^{(2)}$ (simulated at with $\pi_{11} = 0.95$) with N grid cells and M particles, grouped by computational times comparable to the PGAS implementations. Dashes indicate non-convergent implementations.

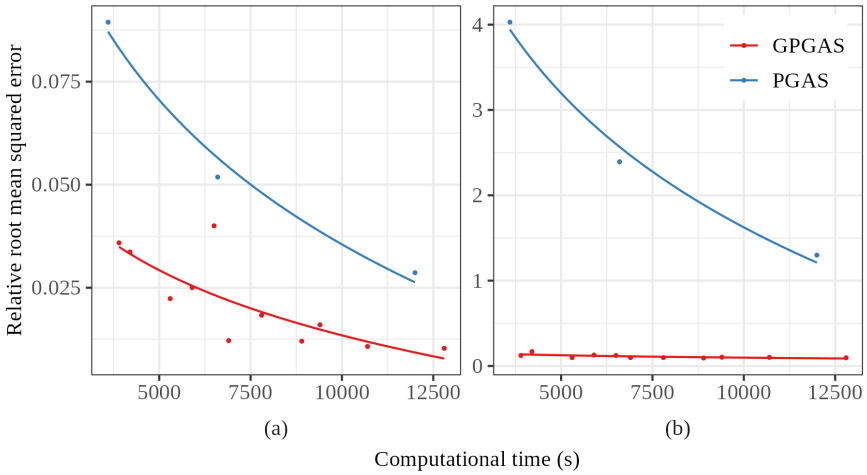


Figure 4.6: Relative root mean squared errors in the (a) posterior mean and (b) posterior variance estimates of the continuous latent states, $x_{1:T}$, with computational time for $y_{1:T}^{(1)}$ with $\pi_{11} = 0.85$. Each point represents a different combination of $N \in \{10, 25, 50, 100\}$ grid cells and $M \in \{10, 25, 50, 100, 200\}$ particles; non-convergent implementations are excluded. Computational time is measured as the time in seconds taken to complete the 10,000 iterations.

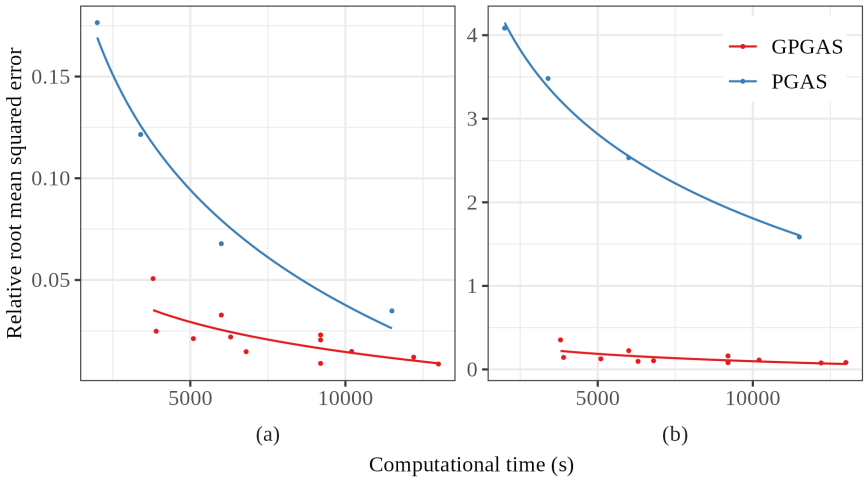


Figure 4.7: Relative root mean squared errors in the (a) posterior mean and (b) posterior variance estimates of the continuous latent states, $x_{1:T}$, with computational time for $y_{1:T}^{(2)}$ with $\pi_{11} = 0.95$. Each point represents a different combination of $N \in \{10, 25, 50, 100\}$ grid cells and $M \in \{10, 25, 50, 100, 200\}$ particles; non-convergent implementations are excluded. Computational time is measured as the time in seconds taken to complete the 10,000 iterations.

The GPGAS algorithm leads to improved posterior mean and variance estimates for a fixed computational time, with particularly notable improvements in the relative root mean squared errors in Tables 4.1 and 4.2. In addition, the GPGAS algorithm appears to scale well both in the number of grid cells and the number of particles for the regime-switching implementations considered in this section. This demonstrates the potential gains in efficiency from combining deterministic approximations of the SMC importance distributions with the computational strategies presented in points 1-3 of Section 4.3.4.

All GPGAS and PGAS implementations appear more efficient according to the effective sample size per second when implemented using more persistent regimes ($\pi_{11} = 0.95$ resulting in more persistent regimes than $\pi_{11} = 0.85$), indicating the impact of degeneracy when the states switch on efficiency. The reason for the differences in performance between the algorithms is likely related to the associated degeneracy rates. For approximately equivalent run times, the GPGAS algorithm reduces the average number of states not updated in the SMC steps by $11 - 50\%$ depending on the implementation. To see the effect of regime switching on the sample impoverishment and accuracy of posterior estimates, we present additional results in Figures 4.8 and 4.9. The figures summarise the performance of each algorithm according to switching and non-switching states and demonstrate the efficiency of the GPGAS algorithm to estimate both the switching and non-switching states. However, it is worth noting that the differences in efficiency appear most noticeable when the state switches.

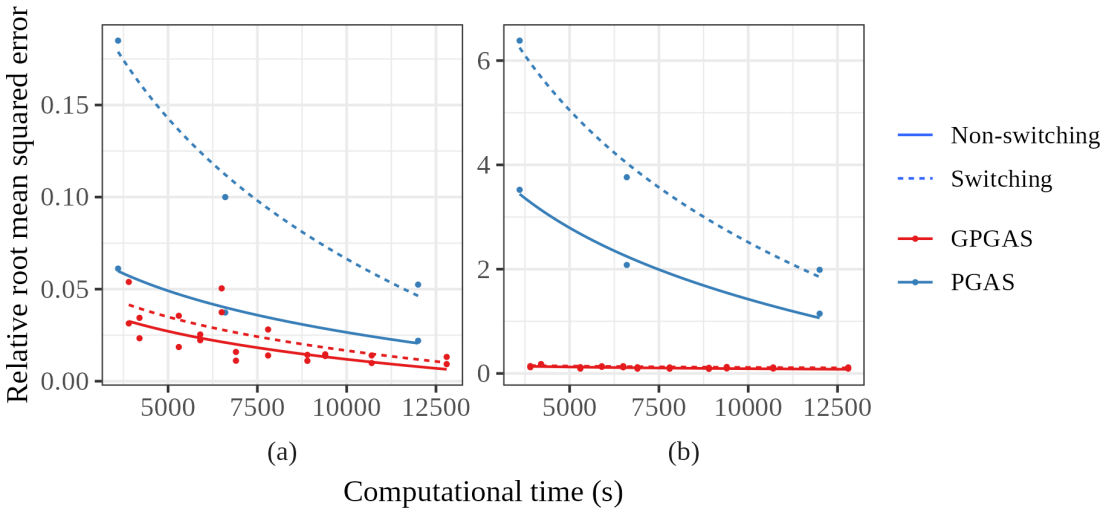


Figure 4.8: Relative root mean squared errors for the (a) posterior mean and (b) posterior variance estimates of the continuous latent states, $x_{1:T}$, by non-switching and switching states with computational time for $y_{1:T}^{(1)}$ (simulated with $\pi_{11} = 0.85$). Each point represents a different combination of $N \in \{10, 25, 50, 100\}$ grid cells and $M \in \{10, 25, 50, 100, 200\}$ particles, non-convergent implementations are excluded. Computational time is measured as the time in seconds taken to complete the 10,000 iterations.

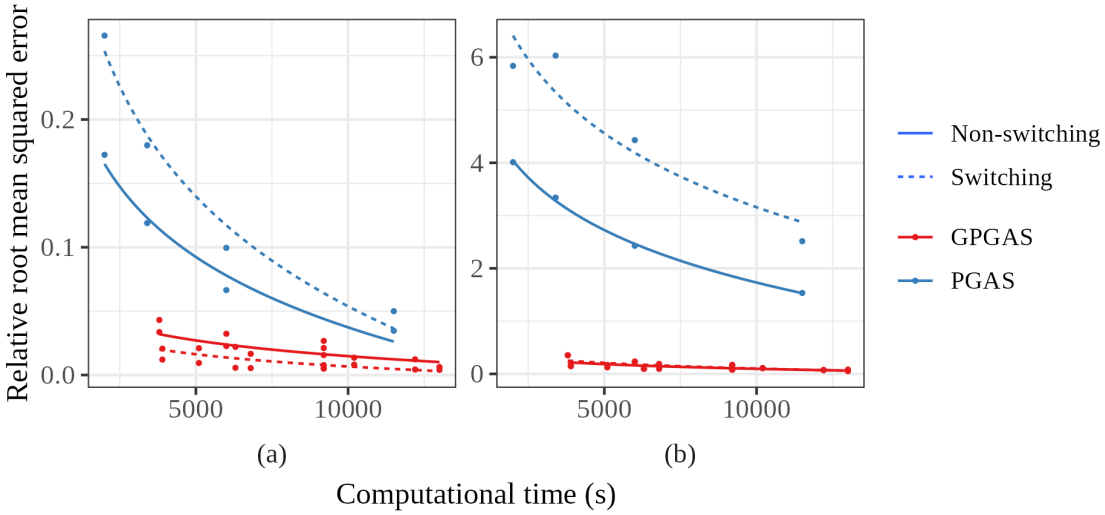


Figure 4.9: Relative root mean squared errors for the (a) posterior mean and (b) posterior variance estimates of the continuous latent states, $x_{1:T}$, by non-switching and switching states with computational time for $y_{1:T}^{(2)}$ (simulated with $\pi_{11} = 0.95$). Each point represents a different combination of $N \in \{10, 25, 50, 100\}$ grid cells and $M \in \{10, 25, 50, 100, 200\}$ particles, non-convergent implementations are excluded. Computational time is measured as the time, in seconds (s), taken to complete the 10,000 iterations.

4.4.2 Case study: Edinburgh tourism demand

We now focus on an extended case study demonstrating the performance of the GPGAS algorithm on a real data regime-switching example. The example is motivated by the impact of the COVID-19 pandemic on Edinburgh's tourism industry, the biggest contributor (32%, Llewellyn et al., 2023) to Scottish tourism revenue in pre-COVID times. In post-COVID recovery plans, understanding the nature of recovery is essential for business communities and policymakers to formulate appropriate policy responses (Lawrence, 2020; OECD, 2020).

In this section, we first outline a model for COVID-era tourism demand in Edinburgh motivated by the work (and data) in Llewellyn et al. (2023). This is accompanied by a detailed discussion of the data and model selection process in Appendix A. The main focus of this section is on efficient model-fitting approaches and we describe, in detail, how the GPGAS algorithm can be implemented efficiently on such a challenging example.

Data

We consider two data sets: response data measuring aggregate hotel revenue (a proxy for tourism demand) and additional covariate data from Google Trends. The hotel revenue (response) data relate to the weekly revenue of over 300 hotels in Edinburgh pre and post-COVID and are shown in Figure 4.10. The covariate data we consider are weekly search query volumes from Google Trends over the same period as the hotel revenue data. The 254 Google Trends series, published by Google LLC (2024), represent the volume of searches over time for particular terms entered into the Google search engine, normalised on a scale of $0-100$ with respect to the search query location and time period. These data are hypothesised to measure the relative popularity of a single search term and aim to capture behavioural responses to the pandemic in the absence of systematic patterns, discussed further in Appendix A.1.2. We present the data for two search terms in Figure 4.11. Each series is lagged to account for a time lag in searches and actual tourist activity and a full discussion of both data sets, including the use of the tourism demand data as a proxy for tourism demand and the pre-processing applied, is provided in Appendix A.1.1.

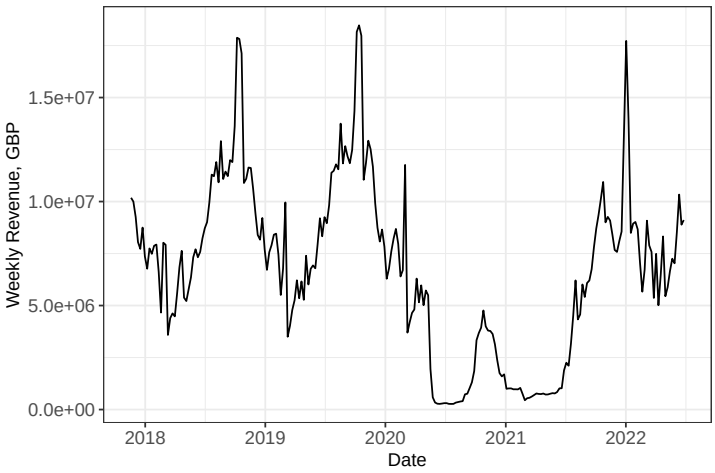


Figure 4.10: Weekly revenue of over 300 hotels in Edinburgh in Great British Pounds (GBP) vs. date.

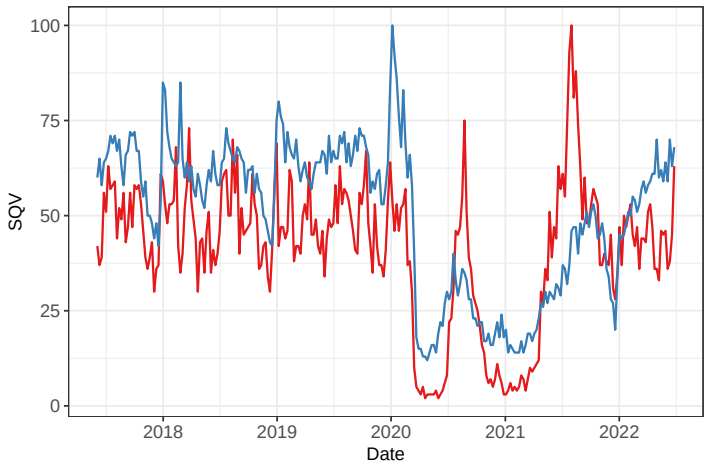


Figure 4.11: Normalised weekly Google search query volumes (SQVs) for UK searches for 'things to do in Edinburgh' (red) and Global searches for 'flights to Edinburgh' (blue) vs. date.

Tourism demand regime-switching state-space model

We model tourism demand (hotel revenue) via a regime-switching SSM to capture dynamic model uncertainty in the COVID period. The model uses a library of structural components similar to those identified in Rivera (2016) and shrinkage priors (Douwes-Schultz et al., 2023; Fischer et al., 2022) to identify the suitable components explaining the tourism demand data. The derivation of the model is discussed further in Appendix A.2 and we simply present the selected model here.

We denote the weekly tourism demand data up to time T by $y_{1:T} = (y_1, \ldots, y_T)$ and the Google Trends data by $\mathbf{G}_{1:T} = (g^1_{1:T}, \ldots, g^{254}_{1:T})$. The tourism demand data,

$y_{1:T}$, are modelled by an SSM with structural time series components (trend and seasonality) and seasonally adjusted, dimension-reduced Google Trends components:

- The high-dimensional Google Trends data, $\mathbf{G}_{1:T}$, are projected to a lower-dimensional space of principal components, $PC_{1:T} = (PC_1, \ldots, PC_T)$, using Bayesian principal component analysis (PCA) (Bishop, 1998). The principal components map to the Google Trends data through a 254-dimensional vector of weights, $\mathbf{W}_{s_t}$ for each $t = 1, \ldots, T$, and the remaining variability in the Google Trends data is modelled via independent Gaussian distributions with variance $\sigma^2_{\eta_{s_t}} > 0$. That is,

$$\mathbf{G}_t \sim N(\mathbf{W}_{s_t} PC_t, \sigma^2_{\eta_{s_t}}), \quad t = 1, \ldots, T. \tag{4.7}$$

- The tourism demand data are log-linearly related to the regime-dependent annual seasonal components, $\lambda^i_t \in \{\lambda^i_1, \ldots, \lambda^i_{52}\}$, $i \in \{1, 2\}$, trend terms, $\mu_{1:T}$, and Google Trends principal components, adjusted by annual seasonal components $\omega^i_t \in \{\omega^i_1, \ldots, \omega^i_{52}\}$, $i \in \{1, 2\}$, and trend terms, $u_{1:T}$. The tourism demand *observation process* is

$$y_t|\mu_t \sim \text{logNormal}(\lambda^{s_t}_t + \mu_t - a_{s_t}(PC_t - \omega^{s_t}_t - u_t), \sigma^2_{\epsilon_{s_t}}), \quad t = 1, \ldots, T, \tag{4.8}$$

  where $\sigma^2_{\epsilon_{s_t}} > 0$ denotes the tourism demand observation process variance. Note that the formulation of the observation process, including the assumptions underlying the seasonal components, is discussed in Appendix A.2.

- We assume that the trend terms, in this case the latent processes, evolve according to linear auto-regressive processes. Mathematically,

$$\begin{aligned} \mu_t|\mu_{t-1} &\sim N(\mu_{t-1} + b_{s_t}, \sigma^2_{\mu_{s_t}}), \quad t = 1, \ldots, T, \\ u_t|u_{t-1} &\sim N(u_{t-1} + c_{s_t}, \sigma^2_{u_{s_t}}), \quad t = 1, \ldots, T, \end{aligned} \tag{4.9}$$

  where μ_0 and u_0 are additional unknown parameters, and b_{s_t} and c_{s_t} are the unknown gradients of the linear trend. In addition, the processes exhibit random fluctuations with variances $\sigma^2_{\mu_{s_t}}, \sigma^2_{u_{s_t}} > 0$, and form the *system processes* of latent states.

- Finally, the parameterisation of the model can vary according to the regime label at each time point $s_{1:T} = (s_1, \ldots, s_T)$. We hypothesise that the current regime is more likely to be consistent with the previous regime. Thus, the

regime evolves according to the following first-order Markov process:

$$P(s_t = j | s_{t-1} = i) = \pi_{ij}, \quad i, j \in \{1, 2\}. \tag{4.10}$$

We assume that the initial regime is arbitrarily set to $s_0 = 1$ and note that the unknown parameters are $\theta = (\lambda_1^1, \ldots, \lambda_{52}^1, \lambda_1^2, \ldots, \lambda_{52}^2, \omega_1^1, \ldots, \omega_{52}^1, \omega_1^2, \ldots, \omega_{52}^2, PC_{1:T}, a_{1:2}, \sigma_{\epsilon_{1:2}}^2, \mathbf{W}_{1:2}, \sigma_{\eta_{1:2}}^2, b_{1:2}, \sigma_{\mu_{1:2}}^2, c_{1:2}, \sigma_{u_{1:2}}^2, \pi_{11}, \pi_{22}, \mu_0, u_0)$, with 104 seasonal revenue components and 104 seasonal Google Trends components. Finally, we assign independent priors to each parameter, with shrinkage priors for several parameters, discussed further in Appendix A.2:

$$\begin{aligned}
&\lambda_t^1 \sim N(16, 0.5), \quad t = 1, \ldots, 52, \\
&\lambda_t^2 \sim N(0, 1), \quad t = 1, \ldots, 52, \\
&\omega_t^1 \sim N(0.5, 0.5), \quad t = 1, \ldots, 52, \\
&\omega_t^2 \sim N(0, 1), \quad t = 1, \ldots, 52, \\
&PC_t, a_i, b_i, c_i, \mu_0, u_0 \sim N(0, 1), \quad t = 1, \ldots, T, \quad i = 1, 2, \\
&W_i^k \sim N(0, 1), \quad i = 1, 2, \quad k = 1, \ldots, 254, \\
&\sigma_{\epsilon_i}^2, \sigma_{\eta_i}^2, \sigma_{\mu_i}^2, \sigma_{u_i}^2 \sim \text{InvGamma}(2, 1), \quad i = 1, 2, \\
&\pi_{ii} \sim \text{Beta}(9.9875, 1.7625), \quad i = 1, 2.
\end{aligned}$$

The Gaussian distributions are parameterised by their variance. These priors give conditional Gibbs updates for $b_{1:2}$, $c_{1:2}$, μ_0, u_0, $\mathbf{W}_{1:2}$, $\sigma_{\eta_{1:T}}$, $\sigma_{\mu_{1:T}}$, $\sigma_{u_{1:T}}$, π_{11}, and π_{22}. The remaining parameters are independently sampled from Gaussian random walk proposal distributions: the λ_t^1, $t \in \{1, \ldots, 52\}$, with variance 0.5, the λ_t^2 and ω_t^2, $t \in \{1, \ldots, 52\}$, with variance 1, the ω_t^1, $t \in \{1, \ldots, 52\}$, with variance 0.5, the PC_t, $t \in \{1, \ldots, T\}$, with variance 0.01, the $a_{1:2}$ with variance 1×10^{-5}, the $\sigma_{\epsilon_1}^2$ with variance 1×10^{-3}, and the $\sigma_{\epsilon_2}^2$ with variance 5×10^{-3}.

Computational decisions

Given the regime-switching SSM in Equations (4.7)-(4.10), we now focus on the computational GPGAS approach to inferring the latent states $(\mu_{1:T}, u_{1:T}, s_{1:T})$ and model parameters, θ. To reduce the computational cost associated with defining grid cells in a three-dimensional latent space, we first sample $(s_{1:T}, \mu_{1:T})$ jointly from their full conditional distribution, followed by $u_{1:T}$ from its full conditional distribution. In both cases, the GPGAS algorithm is implemented following the computational strategies in points (1-3) of Section 4.3.4. To define the grid cells, we use the exact regime labels in the discrete state space of regime labels,

$s_{1:T}$, and we make the following decisions with respect to points (a-c) (Section 4.3.4) in the continuous spaces of $\mu_{1:T}$ and $u_{1:T}$:

a) The HMM approximations are fixed after iteration $\tilde{s} = 1500$ using the posterior mean of each parameter in iterations $1000 - 1500$. This is a lower value than Section 4.4.1 to address the computational cost associated with HMM approximation in two continuous state dimensions, and was found to be reasonable via pilot tuning.

b) Using this pilot tuning run, the finite grid cells were found to cover a reasonable posterior mass over ranges $[-5, 20]$ in the state space of each μ_t and $[-1, 360]$ in the state space of each u_t.

c) As in Section 4.4.1, we sample from within each grid cell using uniform distributions over the finite cells, and otherwise we sample values for μ_t and u_t using truncated Gaussian distributions with variances 2.5 and 36.1 respectively (10% of the range of the finite grid cells).

Results

We compare the performance of four GPGAS and PGAS updating strategies. Due to the large range of the finite cells required to update $u_{1:T}$, and the accuracy of the HMM approximation required, we present an additional approach that updates $(s_{1:T}, \mu_{1:T})$ using the GPGAS algorithm and $u_{1:T}$ using the PGAS algorithm (referred to as *GPGAS + PGAS*). For a fair comparison, we compare the performance of these approaches to a PGAS algorithm using such conditional updates (*conditional PGAS*), as well as a PGAS algorithm jointly updating $(s_{1:T}, \mu_{1:T}, u_{1:T})$ at each iteration (*joint PGAS*). The SMC resampling threshold for all algorithms is set at the optimal level of 50% of the effective sample size, and we test the efficiency of each algorithm using $M = 10, 25, 50, 100, 200, 300, 400$ particles and $N = 25, 50, 100, 200, 300, 400$ grid cells.

We execute each implementation (combination of tuning parameters) 10 times for 1 hour on one core and a 1.6 GHz CPU and compare the results to a joint updating PGAS algorithm with $M = 5000$ particles (the 'ground truth'). Each ground truth run takes around 97 hours to complete its 25,000 iterations. The results for the most efficient implementations of each algorithm are presented in Table 4.3 and are defined as those achieving the lowest mean squared errors compared to the ground truth. Since there are several model parameters and three latent state processes, we summarise each approach by the errors in posterior predictive estimates under each algorithm compared to the ground truth, i.e.,

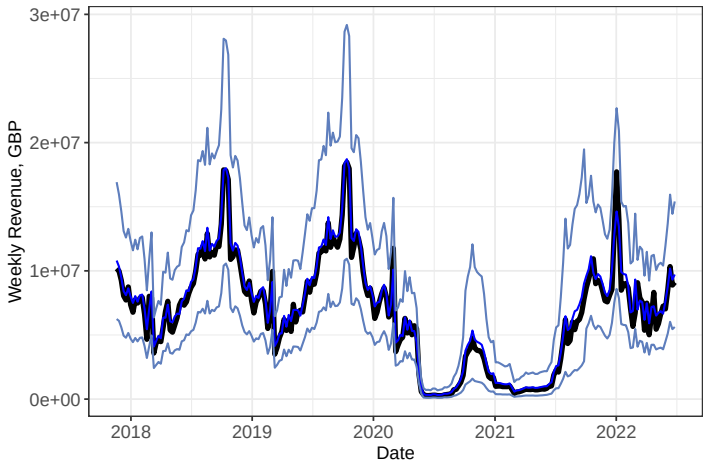
those estimated from samples from the marginal distribution $p(\tilde{y}_{1:T}|y_{1:T})$ for new observations $\tilde{y}_{1:T}$.

				RRMSE				Itera-
						50%	90%	-tions
	N	M	ESS	Mean	Var	CrI	CrI	/hour
Joint PGAS	-	200	900	0.042	0.850	0.049	0.102	6275
Conditional PGAS	-	200	1400	0.028	0.254	0.034	0.057	5435
GPGAS	100	100	1800	0.026	0.331	0.029	0.070	5878
GPGAS + PGAS	200	100	3000	0.017	0.208	0.021	0.051	7714

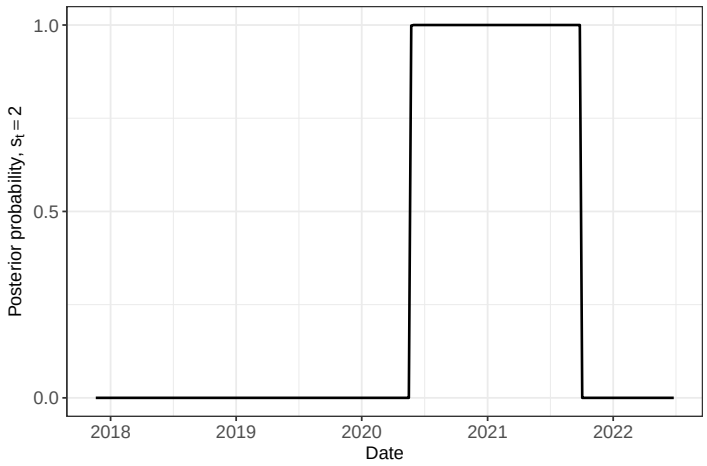
Table 4.3: Effective sample size (ESS), relative root mean squared error (RRMSE) of the estimated posterior predictive mean, variance (Var), and credible intervals (CrI), and the number of iterations completed within 1 hour (Iterations per hour). The RRMSEs are calculated using the posterior predictive estimates compared to those under the ground truth. Shown for the most efficient implementations of each algorithm.

Overall, the results indicate that the GPGAS updates of $\mu_{1:T}$ and $s_{1:T}$, and PGAS updates of $u_{1:T}$ (GPGAS + PGAS) are the most efficient approach. Further, the GPGAS algorithm is arguably as efficient as the conditional PGAS algorithm when updating $u_{1:T}$ but is less efficient than the GPGAS and PGAS combined approach due to the large high posterior density range requiring many grid cells to achieve reasonable HMM approximation error. However, the GPGAS algorithm appears to improve efficiency at switching points, increasing the number of unique particles at these points by $5 - 7\%$ on average, and thus provides an efficient approach for updating $\mu_{1:T}$ and $s_{1:T}$.

We present additional model-based results in Figure 4.12 illustrating the overall fit of the model according to its posterior predictive distribution and the regimes inferred by the model over time. The regime-switching SSM appears to fit the data well, modelling stable period tourism demand before COVID, the data during the COVID lockdown, and the patterns in the upward trend in tourism demand post-lockdown. As well as generally representing the data well, the model effectively switches from the first to second models around the first lockdown (Figure 4.12b), indicating that the regimes correspond to a stable period tourism demand model and COVID disruption model. The model switches back to the first regime (pre-COVID stable model) from 3$^{\text{rd}}$ October 2021, possibly indicating recovery under the assumed interpretation of each regime and subject to the assumptions of each model.



(a) Revenue data over time (black, also shown in Figure 4.10), and associated estimated posterior predictive mean and 90% credible intervals (blue).



(b) Estimated posterior probability of regime 2 over time. The variability in the posterior regime label at each time point ranges from effectively 0 to 0.002 (around the switching points).

Figure 4.12: Posterior summaries for the regime-switching SSM for tourism demand estimated by the GPGAS + PGAS algorithm with 200 grid cells and 100 particles.

The posterior estimates of the parameters provide extra insight into the differences between the models for each period. We highlight the main differences via the posterior estimates in Table 4.4. Several of the parameters appear to undergo shrinkage, defaulting to the shrinkage priors assigned, and could possibly be removed from the model. Notably, the effect size of the Google Trends components in the first (stable period) regime (a_1) indicates that the Google Trends components have negligible explanatory power, especially compared to the second (lockdown) regime (with effect size a_2). Arguably, there are no seasonal effects in the lockdown period, and a possible model extension could explore the effect of

removing the seasonal components in both periods due to, additionally, the low variability in the seasonal components for the stable period. There also appears to be a greater amount of unexplained variability in the lockdown period compared to the stable period from the posterior distributions of $\sigma^2_{\epsilon_1}$ and $\sigma^2_{\epsilon_2}$, indicating a higher inherent level of variability or systematic components not captured by the model in the lockdown period.

	Posterior estimate of		
Parameter	**Mean**	**5% quantile**	**95% quantile**
a_1	1.515×10^{-4}	-3.247×10^{-4}	7.710×10^{-4}
a_2	-0.031	-0.035	-0.029
$\sigma^2_{\epsilon_1}$	0.047	0.035	0.061
$\sigma^2_{\epsilon_2}$	0.242	0.124	0.404
λ^1	16.008	15.630	16.388
λ^2	0.042	-0.797	0.876

Table 4.4: Posterior mean and credible intervals estimates for several parameters estimated by the GPGAS + PGAS algorithm with 200 grid cells and 100 particles. The estimates presented for λ^1 and λ^2 are the average of the posterior mean and credible intervals for λ^1_t and λ^2_t, $t = 1, \ldots, 52$.

4.5 Discussion

We present an efficient computational approach to fitting general SSMs using deterministic grid approaches within the SMC framework. For a range of common regime-switching SSMs, we show that the GPGAS approach gives more efficient performance than current SMC-based approaches by efficiently reducing sample impoverishment. Further, we show that GPGAS updates can improve efficiency when applied to a challenging COVID-era tourism demand model motivated by real data. By combining a deterministic grid with SMC steps, we have capitalised on the accuracy of grid-based approaches while reducing their overall computational cost and improving their scalability in the number of grid cells, and the scalability of SMC steps in the number of particles. Further, the SMC corrections have reduced the number of tuning parameters associated with current grid-based approaches, and their sensitivity, improving their utility in practical applications.

A possible point for further consideration is the parallelisation of the GPGAS algorithm. As with other SMC approaches, trajectories of particles can be sampled in parallel, potentially reducing the computational time of the approach. Further approaches to parallelisation can also be considered and are discussed, for example, in Vergé et al. (2013). Note that a potentially efficient approach

could group parallel computations by the grid cells containing particles from the previous time point, thus avoiding the additional computational cost from relaxing computational strategy 2 of Section 4.3.4. However, as with the different parallelisation strategies in Section 3.5, the computational cost associated with re-synchronisation should also be considered (Henriksen et al., 2012).

In this chapter, we considered an additional interesting point for research: the combination of the GPGAS and PGAS methods. As in Section 4.4.2, and observed earlier in Chapter 3, equally-sized grid cell HMM approximations can have a high computational cost when applied to states with a large high posterior density range. It may be possible to improve the efficiency of the GPGAS algorithm using a state-centred or a similar approach, as in Chapter 3 (Section 3.3.1), provided that this still provides a valid particle Gibbs algorithm. However, this approach relies on being able to adapt the centring of the HMM approximation and may perform poorly if the HMM approximation is fixed in future iterations to reduce computational cost. A further avenue for research could explore the efficiency of a deterministic grid on the space of the observations, reducing the number of observed state probability matrix calculations in the HMM approximations. That is, an additional partition is imposed on the space of possible observations and the observed state distribution is now calculated for each state grid cell and each new interval (rather than each state grid cell and each time point). The observed state distribution at each time point is then given by the probabilities associated with the interval containing the observation.

The proposed grid-based importance distribution could be extended to other SMC-based methods. In particular, the grid importance distribution could be applied to improve sample impoverishment in standard filtering applications with known model parameters. In this case, the grid-based approach does not require multiple transition and observed state probability matrix approximations (across iterations) and is thus computationally inexpensive. However, for online parameter inference, using for example the nested particle filter (Crisan and Míguez, 2018; Pérez-Vieites and Míguez, 2021), the method may be computationally costly, requiring many transition and observation probability matrix approximations for different model parameter values. One possibility is to calculate HMM approximations for groups of similar model parameter samples, reducing the number of HMM approximations required. This presents a particularly interesting avenue for future research, extending the grid importance distribution to other SMC-based methods to combat sample impoverishment efficiently.

Chapter 5

Conclusions

5.1 Conclusions and contributions

This thesis addresses the challenge of efficient Bayesian inference on the latent states and model parameters of a state-space model. Markov chain Monte Carlo methods are widely applied when the joint posterior distribution of the latent states and model parameters only admits a closed-form expression up to proportionality, providing theoretical guarantees such as the consistency of posterior estimates. However, updating the latent states within a Markov chain Monte Carlo algorithm can be inefficient, leading to practically infeasible computational times to achieve sufficiently low error in the posterior estimates.

In Chapter 3, we introduced the point mass proposal Metropolis-Hastings (PMPMH) framework which uses deterministic hidden Markov model approximations to design efficient Metropolis-Hastings proposal distributions. We showed that the algorithm provides a reliable approach to posterior estimation and is especially efficient for a traditionally challenging near-chaotic population growth model where state-of-the-art approaches are computationally costly. In this chapter, we also highlighted the flexibility of deterministic grid proposal distributions and explored how the number of grid cells, and their location and size, can be chosen efficiently depending on the application. In addition, we provided three approaches to defining the grid cells and demonstrated that a current-state-centred approach can be particularly efficient when the range of the data or the observation process variance is large.

In Chapter 4, we presented the grid particle Gibbs with ancestor sampling (GPGAS) algorithm, proposing the use of deterministic hidden Markov model approximations to design particle Gibbs with ancestor sampling importance distributions. The approach is motivated by the combination of hidden Markov model

approximations with corrective sequential Monte Carlo steps that reduce the computational cost and number of tuning parameters associated with the hidden Markov model approximations while addressing the sample impoverishment problem observed in particle Gibbs with ancestor sampling algorithms. We showed that the GPGAS algorithm efficiently handles challenging regime-switching state-space models and can substantially improve the accuracy of posterior estimates for a fixed computational time when compared to state-of-the-art approaches. We additionally showed that combining GPGAS and particle Gibbs with ancestor sampling updates can be particularly efficient, applying the GPGAS algorithm to state dimensions that are likely to suffer from sample impoverishment and the particle Gibbs with ancestor sampling algorithm to state dimensions with large high posterior ranges.

5.2 Future work and extensions

The combination of deterministic grid-based hidden Markov model approximations and Markov chain Monte Carlo methods presents several interesting points for future research. We first note that it is relatively straightforward to parallelise components of both the PMPMH and GPGAS algorithms and that this may improve the computational time of each approach. Within the PMPMH framework, parallelisation strategies can be imposed to update serially independent blocks of states and compute the hidden Markov model probabilities within each block under the state-centred grid cell approach. Within the GPGAS approach, trajectories of particles can be sampled in parallel, particularly across groups of particles within different grid cells. Although additional consideration should be given to memory limitations and the computational cost associated with re-synchronising each algorithm (Henriksen et al., 2012), parallelisation is a relatively straightforward addition that may improve computational times. There are, however, a number of additional but more complex avenues for future research that could be explored. We describe each avenue in turn.

5.2.1 HMM approximation accuracy

A particularly interesting avenue for future research considers efficient improvements to the hidden Markov model approximations used by each algorithm. A first consideration is the use of more accurate numerical integration strategies, for example, increasing the number of points in each grid cell used for the deterministic integration strategy, the use of simple linear functions joining evaluations at

mid-points, or the use of other fast evaluations that include the gradient in each grid cell. Example approaches are discussed in Ballreich (2017, Chapter 3) and Davis and Rabinowitz (2007). There may also be ways to improve the efficiency of the within-cell proposal distributions when sampling point values, including approaches that approximate the posterior density in each grid cell where the posterior conditional distributions are irregular or vary in an unsystematic way depending on the value of the parameter, such as Gaussian mixture approximation (Plataniotis and Hatzinakos, 2001) and variational inference (Blei et al., 2017; Hirt et al., 2019). These approaches to improving the mixing of each algorithm may permit the use of fewer grid cells, decreasing the overall computational cost which may also permit greater scalability of the HMM to higher dimensional latent spaces. However, with any such adaptations, there is a trade-off between the improved mixing properties and the computational cost arising from the complexity of the method and these approaches could be most valuable to explore for more complex state-space models.

Within the GPGAS framework, when the high posterior mass range of the state space is large, it may be possible to improve efficiency using a state-centred approach as in the PMPMH approach in Chapter 3. That is, the grid cell boundaries could vary through time according to the empirical quantiles of the particles at each time point or the current states in the Markov chain Monte Carlo iterations as in Approach 3 of the PMPMH algorithm. However, compared to the PMPMH approach, the equally-sized grid cells of the GPGAS algorithm scale well with the state dimension, requiring fewer transition matrix calculations in the Markov chain Monte Carlo steps. Therefore, approaches that improve the hidden Markov model approximation for large high posterior mass ranges whilst maintaining a small number of transition matrix calculations could be explored. A possible approach could define the grid cells in the same way for all or several time points, setting the grid cells according to coarsely-approximated quantiles of the true posterior distribution via, for example, variational Bayes approximations (Onizuka et al., 2023). However, as with the previously-described approaches to improving the hidden Markov model approximations and reducing the number of grid cells required, the computational gains should be balanced with the computational cost of the chosen approach.

5.2.2 Computational cost of the HMM approximations

Exploring methods to reduce the computational cost of the hidden Markov model approximations whilst retaining the mixing properties of each approach may also

be a fruitful avenue for future research, reducing the computational cost of both the PMPMH and GPGAS methods. For example, one may consider a deterministic grid on the space of the observations, reducing the number of observed state probability matrix calculations in the HMM approximations. It may also be possible to reduce computational cost whilst retaining the mixing properties of the algorithms by adapting the number of grid cells at each time point, reducing the number of grid cells when there is little uncertainty in the latent states. Further, the computational cost of the hidden Markov model approximation can be reduced by updating the grid cells less frequently across the MCMC iterations. Although updating the grid cells periodically across iterations may produce biased estimates of the posterior distribution (Haario et al., 1999), fixing the grid cell definition after a specified number of iterations is valid and we showed that this is an efficient approach within the GPGAS framework. This approach could be explored further for the PMPMH method and for state-centred approaches to defining the grid cells. However, we note that state-centred grid cells may be more sensitive to the frequency with which the grid cells are updated and may exhibit reduced mixing properties.

5.2.3 Grid design on high-dimensional spaces

Defining grid cells on high-dimensional spaces is non-trivial (Duník et al., 2018; Smidl and Gasperin, 2013), resulting in challenges when it is inefficient to sample lower-dimensional state dimensions conditional on other state dimensions. One possibility could explore the efficiency of joint updating strategies when partial state dimensions are sampled under the particle Gibbs with ancestor sampling framework and others are sampled from the grid-based importance distributions of the GPGAS algorithm. Other possible approaches include projection and grid definition on lower-dimensional manifolds (Tidefelt and Schön, 2009). This is a challenging and active area for future research.

We propose approaches that are efficient for traditionally challenging state-space models. As more data are collected and these data become more complex, more complex models are required. In turn, more complex model-fitting algorithms may be required for practical inference. Efficient model-fitting algorithms are likely to be an active area of ongoing research.

Appendix A

Further materials for the tourism demand case study

In this appendix, we provide supplementary information for the tourism demand model-fitting case study in Section 4.4.2. We first describe each data set, pre-processing, and any further evaluations of the data in Appendix A.1. In Appendix A.2, we derive the regime-switching model presented in Section 4.4.2.

A.1 Data

A.1.1 Response data: hotel revenue

The response data are given by the weekly aggregate revenue of hotels in Edinburgh. These data relate to the revenue of 357 hotels in Edinburgh in Great British Pounds (GBP) and were collected by Smith Travel Research, Inc. (STR) from 4$^{\text{th}}$ June 2017 to 30$^{\text{th}}$ April 2022.

The hotel revenue data are complete but indirectly measure tourism demand. Therefore, to assess the hotel revenue data as a proxy for tourism demand, the hotels listed in the data set are first compared to those registered with Edinburgh Council (the local government authority for the city of Edinburgh), confirming similarity in their spatial distributions. Additionally, the hotel data are compared to footfall counts for Edinburgh Castle. Edinburgh Castle is one of the primary tourist attractions in the city, and, as a direct measure of tourism demand, the castle data may reflect tourism demand in Edinburgh more accurately than aggregate hotel revenue. To compare these data, the aggregate hotel revenue data and Edinburgh Castle are scaled by dividing each data point by their respective all-time maximum values, shown in Figure A.1.

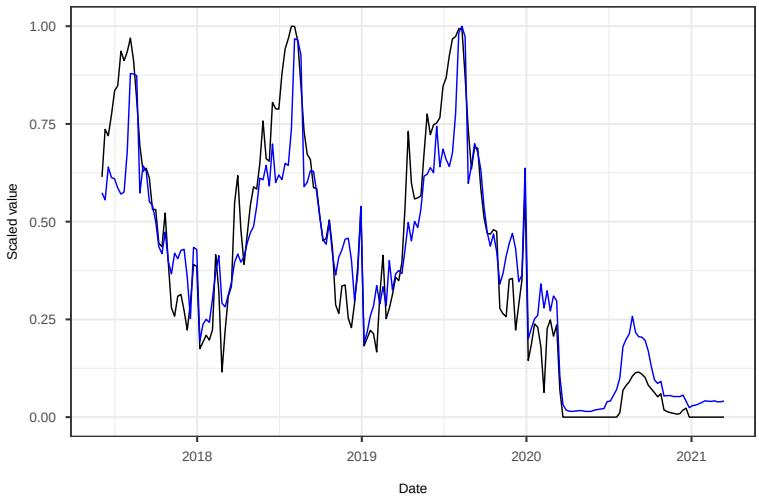


Figure A.1: Scaled Edinburgh Castle visitor count data (black) and scaled hotel revenue data (blue) from 2017.

The scaled data are similar before the first UK lockdown (16$^{\text{th}}$ March 2020). After the first lockdown in the UK, the scaled hotel data follow the same overall trend as the Edinburgh Castle visitor data but at higher levels. At the beginning of the COVID pandemic, the number of visitors to the castle was limited as a COVID prevention measure, even after tourist activity was permitted by the UK and Scottish governments. Therefore, the castle data were censored and are not used as the proxy for tourism demand directly but we conclude that the hotel revenue data serve as a sensible proxy for tourism demand in Edinburgh.

A.1.2 Covariate data: Google Trends

Search engine 'big data' have proven effective in improving tourism demand model-based inferences during the COVID-19 pandemic, hypothetically capturing large-scale behavioural responses to the pandemic (Li et al., 2021; Ma et al., 2022). The underlying idea is that individuals typically retrieve information about an area they would like to visit from a search engine, thus aggregate search query volumes over time may indicate how collective interest in visiting an area changes over time. In addition to their potential for providing behavioural information based on a large sample size when historic data cannot be relied upon, search query volume data are also available almost immediately after the time point they were captured, a key practical advantage in quickly-changing scenarios such as COVID. We consider Google Trends data to inform extra covariates and hypothesise that

this may capture additional idiosyncratic information in the tourism demand (hotel revenue) data.

Google LLC (2024) publish indices representing the volume of searches over time for a particular term entered into the Google search engine. Google first take a sample of the billions of search queries entered into its search engine per day and count incidences for each search query by location. The published indices (in the range $0-100$) are the counts normalised with respect to all searches in a given location and time period. Thus, a particular Google Trends series captures the popularity of a search term relative to the other searches in the time period and location and can be used to compare the popularity of search terms.

Using the application programming interface (API) wrapper gtrendsR (Massicotte and Eddelbuettel, 2022) in the R programming language, several Google Trends search query volume series can be obtained. The API wrapper automatically scrapes the weekly search query volume data in a specified location, as well as 'related search terms' for a given period and search term. Following a standard approach in the literature (Li et al., 2017; Xiaoxuan et al., 2016; Yang et al., 2015), search terms are selected by first selecting an original first level of search terms: *Edinburgh hotels*, *Edinburgh Castle*, *Edinburgh Airbnb*, *Edinburgh car rental* and *Edinburgh trains*. The remaining search terms are then all of the associated queries provided by the related search terms. In our acquired data set, this process was repeated for both search query volume data globally and in the UK over the period from 4$^{\text{th}}$ June 2017 to 30$^{\text{th}}$ April 2022, acquiring global search query volume data for 124 search terms and UK search query volume data for 130 search terms after removing duplicated search terms.

As a final pre-processing step, we consider that tourists obtain planning information, for example, information on hotels, travel guides, and places to eat in advance of their actual tourist activity. Thus, we lag each Google Trends series by that which produces the highest Pearson correlation with the tourism demand data. In this case, the Google Trends data are most linearly related to the logarithm of the tourism demand data. Thus, if the correlation between the i^{th} Google Trends series, $d^i_{1:T}$, and the logarithm of the tourism demand series, $\log y_{1:T}$, is maximised at lag l of the Google Trends series, each new series is given by

$$g_t{}^i = d^i_{t-l}, \;\; t = l+1, \ldots .T,$$

Applying this lagging procedure to each Google Trends series, $i = 1, \ldots, 254$, and standardising each series to give uniform variance to ensure that the contribution of each series is treated equally, we obtain the Google Trends covariate data, denoted $\mathbf{G}_{1:T} = (g^1_{1:T}, \ldots, g^{254}_{1:T})$.

A.2 Regime-switching structural components and model development

Stable period tourism demand models (before COVID) generally rely on long-run historic behaviour such as dominant seasonal patterns (Jorge-González et al., 2019; Song and Li, 2008; Wu et al., 2017). However, designing an appropriate model following an unprecedented event such as COVID can be challenging due to the lack of relevant historical data. A variety of approaches have been proposed, often relying on exogenous data or information captured, for example, using irregular components (Fotiadis et al., 2021), counterfactual explanations within causal inference (Zhang et al., 2021) or by incorporating intervention effects (Seong and Lee, 2021). Google Trends data, in particular, have been shown to improve COVID-era tourism demand models (Li et al., 2021; Ma et al., 2022).

Following the immediate effect of the pandemic, tourism demand showed evidence of recovery: return to a period of stability in both the absolute levels of tourism demand and its systematic patterns (UNWTO, 2022; World Travel & Tourism Council, 2022). However, the extent to which the systematic patterns in the recovery data were the same as pre-COVID levels is unclear. It is possible that stable patterns are present alongside some lasting COVID effects during the recovery period. Bhatia et al. (2022) and Zhang et al. (2023) hypothesise that some impacts of COVID are long-lasting, for example, permanent changes in the risk perception of tourists, increased promotion of domestic tourism, increased awareness of environmental issues, and the acceleration of technological advancements (a 'new normal' in tourism demand). Therefore, early switching to a stable period tourism demand model based on the long-run pre-COVID data, and less on exogenous data, may not accurately capture tourism demand. On the other hand, remaining in a disruptive-period model when the data can be modelled using long-run historic behaviour may result in unnecessarily reduced model accuracy.

Regime-switching models (Hamilton, 1989) can be applied to automatically adapt the choice of model to the observed data and account for dynamic model uncertainty, and have been applied widely when the model specification and/or

model switching point is difficult to define. Due to the difficulties in defining singular models for recovery phases, regime-switching models have found a number of post-disruption applications, including in post-war economics (Eo and Kim, 2016), financial market crisis contagion during the Global Financial Crisis (Chan et al., 2018), the modelling of infection rates and stock returns in the aftermath of the COVID pandemic (Bouteska et al., 2023; Haimerl and Hartl, 2023; Kartal et al., 2022), and the modelling of tourism demand following political violence, terrorist attacks or smaller-scale pandemic events (Aronica et al., 2021; Lanouar and Goaied, 2019).

We derive a regime-switching model and begin by considering that, due to the unprecedented nature of the COVID pandemic and the lack of prior information on the suitable variables explaining the tourism demand data, a sensible approach identifies a library of structural components that may be informative of tourism demand in each regime. However, to avoid over-fitting, we assign shrinkage priors (Douwes-Schultz et al., 2023; Fischer et al., 2022) to perform model selection within each regime. Note that an alternative approach defines variables to include or exclude in each regime model a priori, for example, in Haimerl and Hartl (2023), where additional variables are included to account for extra effects in the post-COVID period. However, this approach requires prior information on the negligible effects in each regime. Thus, to derive the library of structural components considered here we note that the variables explaining tourism demand are likely to vary most substantially between the pre and post-COVID periods. We consider these periods separately for the purposes of effect selection. We then describe the process for obtaining the Google Trends covariates, before describing the regime-switching model and shrinkage prior approach.

A.2.1 Components identified in the pre-COVID data

The pre-COVID tourism demand (hotel revenue) data in Figure 4.10 have a strong trend and an annual seasonal pattern. Widely used to provide interpretable models of tourism demand with strong periodicity, we propose a structural time series model (Eugenio-Martin and Perez-Granja, 2020; Harvey, 1990; Scott and Varian, 2014; Song and Li, 2008). In addition, we consider that slight increases in the variability observations can be observed over the pre-COVID period.

Thus, we assume that the observations are log-linearly related to the structural components and the pre-COVID data are described for $t = 1, \ldots, T$ by

$$\begin{aligned} y_t &\sim \text{logNormal}(\mu_t + \lambda_t, \sigma_\epsilon^2), \\ \mu_t &= \mu_{t-1} + b, \end{aligned} \tag{A.1}$$

where μ_t and $\lambda_t \in \{\lambda_1, \ldots, \lambda_{52}\}$ denote the trend and annual-period seasonal components at time t respectively, $\sigma_\epsilon^2 > 0$ denotes the residual variance, and b is the gradient parameter of the trend with intercept μ_0. Note that we have assumed that the levels of trend are correlated over time (Markovian) but, for simplicity, we have assumed that the seasonal components at different time points are uncorrelated. One possible extension to the model could consider incorporating auto-regressive structure in the seasonal components to capture temporal dependence in the weekly effects.

A.2.2 Components identified in the post-COVID period

Noting the lack of prior information on the form of the model in the post-COVID period and the later use of shrinkage priors, we initially consider the model identified in Equation (A.1). However, the post-COVID period data in Figure 4.10 appears to exhibit short-term fluctuations in the level of the trend that may not be explained by systematic components. Therefore, we also consider that the trend in the post-COVID tourism demand may be more variable with the short-term time dependence captured by an autoregressive process. We therefore adapt the model in Equation (A.1) to allow for autoregressive stochastic trend:

$$\begin{aligned} \mu_t | \mu_{t-1} &\sim N(\mu_{t-1} + b, \sigma_\mu^2), \\ y_t | \mu_t &\sim \text{logNormal}(\lambda_t + \mu_t, \sigma_\epsilon^2), \end{aligned} \tag{A.2}$$

for $t = 1, \ldots, T$ where the additional parameter, $\sigma_\mu^2 > 0$ denotes the imposed variability in the trend process. Notably, Equation (A.2) is now a state-space model in the unobserved trend process.

In this post-COVID case, such a model relying on systematic patterns in an unstable period may fit the data poorly. We therefore propose that residual idiosyncratic patterns may be captured by exogenous sources of information, Google Trends data, relating to the behavioural drivers of tourism demand.

A.2.3 Google Trends components

Google Trends search query volumes relating to tourism quantify collective interest in visiting an area and have thus been used widely and successfully to provide additional information on near-term tourism activity (Li et al., 2021). For example, we may expect that individuals considering a visit to Edinburgh would search 'Edinburgh car rental' on Google in advance of their visit. Thus, aggregate search query volumes on such search terms may reflect changes in tourist activity.

However, a key challenge to modelling using Google Trends data is capturing the relevant information in many Google Trends series while avoiding over-fitting (Law et al., 2019; Yang et al., 2015). The Google Trends data considered here relate to 254 search terms, including both global and UK series, and not all of the information in the Google Trends data set may be relevant to the model. A common approach, and that used here, reduces the dimension of the data using principal component analysis (PCA) (Bishop, 1998; Höpken et al., 2020; Jolliffe and Cadima, 2016), summarising the main information in the lagged Google Trends data introduced in Section 4.4.2.

Assuming that one principal component sufficiently summarises the information in the Google Trends data set while avoiding over-fitting, we follow the standard Bayesian PCA approach of Bishop (1998). If $\mathbf{G}_{1:T} = (g^1_{1:T}, \ldots, g^{254}_{1:T})$ denotes the full set of Google Trends data, these data can be written as a projection of the principal component, $PC_{1:T}$, with projection directions, $\mathbf{W}$. In addition, we assume that the residual errors at different time points are uncorrelated and Gaussian, and that the projections are uncorrelated. Thus, the principal component at each time point satisfies

$$\mathbf{G}_t \sim N(\mathbf{W}PC_t, \sigma^2_\eta),$$

where $\sigma^2_\eta > 0$ denotes the residual variance.

To ensure that the Google Trends principal component, $PC_{1:T}$, captures extra idiosyncratic information not captured by the tourism demand data alone, we allow similar seasonal and stochastic trend components in the principal components when regressed on the data. In addition, we later assign shrinkage priors so that these patterns in the data are only enforced if they are present. Combining the principal component model with the components identified in Equation (A.1) and subsequently Equation (A.2), the full state-space model is

now given for $t = 1, \ldots, T$ by

$$\begin{aligned}
&\mu_t|\mu_{t-1} \sim N(\mu_{t-1} + b, \sigma_\mu^2), \\
&u_t|u_{t-1} \sim N(u_{t-1} + c, \sigma_u^2), \\
&\mathbf{G}_t \sim N(\mathbf{W}PC_t, \sigma_\eta^2), \\
&y_t|\mu_t \sim \text{logNormal}(\lambda_t + \mu_t - a(PC_t - \omega_t - u_t), \sigma_\epsilon^2),
\end{aligned} \tag{A.3}$$

where additionally σ_u^2 denotes the variance of the principal component trend process with intercept and gradient u_0 and c respectively. Note that such a state-space model incorporating seasonal components has been applied previously in (Rivera, 2016).

A.2.4 Regime-switching state-space model

Given the state-space model with latent states $\mu_{1:T}$ and $u_{1:T}$ in Equation (A.3), we now account for COVID-era model uncertainty and consider that the relevance of different model components may vary over time. As a motivating hypothesis, we consider that the Google Trends principal components may be more informative of tourism demand in the absence of systematic patterns post-COVID and that this hypothesis may extend to other model components.

We therefore impose a two-state regime-switching process to enable changes in the model parameterisations informed by the data. That is, at time $t \in \{1, \ldots, T\}$, we denote the regime label by $s_t \in \{1, 2\}$ and impose time dynamics by allowing the regime label to depend on that at the previous time point. In other words, the regime label evolves according to a discrete-valued process with probabilities $P(s_t = j|s_{t-1} = i) = \pi_{ij}$, $i, j \in \{1, 2\}$ and the regime-switching model based on Equation (A.3) is given for $t = 1, \ldots, T$ by

$$\begin{aligned}
&\mu_t|\mu_{t-1} \sim N(\mu_{t-1} + b_{s_t}, \sigma_{\mu_{s_t}}^2), \\
&u_t|u_{t-1} \sim N(u_{t-1} + c_{s_t}, \sigma_{u_{s_t}}^2), \\
&\mathbf{G}_t \sim N(\mathbf{W}_{s_t}PC_t, \sigma_{\eta_{s_t}}^2), \\
&y_t|\mu_t \sim \text{logNormal}(\lambda_t^{s_t} + \mu_t - a_{s_t}(PC_t - \omega_t^{s_t} - u_t), \sigma_{\epsilon_{s_t}}^2), \\
&P(s_t = j|s_{t-1} = i) = \pi_{ij}, \quad i, j \in \{1, 2\}.
\end{aligned} \tag{A.4}$$

Finally, we consider that, particularly in the post-COVID case, many aspects of this model may result in over-fitting. For example, seasonal components may not be informative of the post-COVID tourism demand levels and over-estimating

their effects may dilute the information provided by other components such as the Google Trends principal components. We therefore apply Bayesian ridge shrinkage priors (van Erp et al., 2019) to $\lambda_1^2 \ldots, \lambda_{52}^2$, $\omega_1^2 \ldots \omega_{52}^2$, $PC_{1:T}$, a_1, a_2, b_1, b_2, c_1, c_2, μ_0, u_0, $\mathbf{W}_1$, $\mathbf{W}_2$, $\sigma_{\epsilon_1}^2$, $\sigma_{\epsilon_2}^2$, $\sigma_{\eta_1}^2$, $\sigma_{\eta_2}^2$, $\sigma_{\mu_1}^2$, $\sigma_{\mu_2}^2$, $\sigma_{u_1}^2$, and $\sigma_{u_2}^2$ so that they are likely to be near-zero unless there is evidence to the contrary. We note that this also unifies the models in Equations (A.1) and (A.2), i.e., both of these models can be written as a special case of that in Equation (A.4) and we assign the (independent) priors:

$$
\begin{aligned}
&\lambda_t^1 \sim N(16, 0.5), \quad t = 1, \ldots, 52, \\
&\lambda_t^2 \sim N(0, 1), \quad t = 1, \ldots, 52, \\
&\omega_t^1 \sim N(0.5, 0.5), \quad t = 1, \ldots, 52, \\
&\omega_t^2 \sim N(0, 1), \quad t = 1, \ldots, 52, \\
&PC_t, a_i, b_i, c_i, \mu_0, u_0 \sim N(0, 1), \quad t = 1, \ldots, T, \quad i = 1, 2, \\
&W_i^k \sim N(0, 1), \quad i = 1, 2, \quad k = 1, \ldots, 254, \\
&\sigma_{\epsilon_i}^2, \sigma_{\eta_i}^2, \sigma_{\mu_i}^2, \sigma_{u_i}^2 \sim \text{InvGamma}(2, 1), \quad i = 1, 2, \\
&\pi_{ii} \sim \text{Beta}(9.9875, 1.7625), \quad i = 1, 2.
\end{aligned}
$$

noting simple zero-centred priors (ridge priors) for many parameters to avoid over-fitting. The Gaussian distributions are parameterised by their variance. We note the choice of non-zero centred priors for λ_t^1 and ω_t^1, $t \in \{1, \ldots, 52\}$, since we assume prior knowledge that seasonality is present in at least one period (for example, pre-COVID). The prior parameters for λ_t^1, $t \in \{1, \ldots, 52\}$, are chosen to reflect the assumption that average weekly revenue is in the order of 1×10^7 (hence the log average revenue is around 16), and the prior parameters for ω_t^1, $t \in \{1, \ldots, 52\}$, are chosen since the Google Trends data are between 0 and 1. We also assume persistent regimes via the priors for π_{11} and π_{22}, which have expected values of 0.85 and variances of 0.01.

Bibliography

Andrieu, C., Davy, M., and Doucet, A. (2003). Efficient particle filtering for jump Markov systems. *IEEE Transactions on Signal Processing*, 51(7):1762–1770.

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. *Annals of Statistics*, 37(2):697–725.

Andrieu, C. and Thoms, J. (2008). A tutorial on adaptive MCMC. *Statistics and Computing*, 18(4):343–373.

Aronica, M., Pizzuto, P., and Sciortino, C. (2021). COVID-19 and tourism: what can we learn from the past? *The World Economy*, 45(2):430–444.

Auger-Méthé, M., Newman, K., Cole, D., Empacher, F., Gryba, R., King, A. A., Leos-Barajas, V., Mills Flemming, J., Nielsen, A., Petris, G., and Thomas, L. (2021). A guide to state–space modeling of ecological time series. *Ecological Monographs*, 91(4):1–38.

Barra, I., Hoogerheide, L., Koopman, S. J., and Lucas, A. (2016). Joint Bayesian analysis of parameters and states in nonlinear non-Gaussian state space models. *Journal of Applied Econometrics*, 32(5):1003–1026.

Bar-Shalom, Y., Li, X., and Kirubarajan, T. (2001). *Estimation with Applications to Tracking and Navigation: Theory, Algorithms and Software*. Wiley.

Beaumont, M. A. (2003). Estimation of population growth or decline in genetically monitored populations. *Genetics*, 164(3):1139–1160.

Berntorp, K. and Di Cairano, S. (2017). Particle Gibbs with ancestor sampling for identification of tire-friction parameters. In *20th World Congress of the IFAC*, pages 14849–14854.

Bhatia, A., Roy, B., and Kumar, A. (2022). A review of tourism sustainability in the era of COVID-19. *Journal of Statistics and Management Systems*, 25(8):1871–1888.

Bishop, C. (1998). Bayesian PCA. In *Proceedings of the 11th International Conference on Neural Information Processing Systems*, pages 382 – 388.

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2017). Variational inference: a review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877.

Blom, H. and Bar-Shalom, Y. (1988). The interacting multiple model algorithm for systems with Markovian switching coefficients. *IEEE Transactions on Automatic Control*, 33(8):780–783.

Boers, Y. and Driessen, J. N. (2003). Interacting multiple model particle filter. *IEE Proceedings - Radar, Sonar and Navigation*, 150(5):344–349.

Borowska, A. and King, R. (2023). Semi-complete data augmentation for efficient state-space model fitting. *Journal of Computational and Graphical Statistics*, 32(1):19–35.

Bouteska, A., Sharif, T., and Abedin, M. Z. (2023). COVID-19 and stock returns: evidence from the Markov switching dependence approach. *Research in International Business and Finance*, 64(101882).

Branchini, N. and Elvira, V. (2021). Optimized auxiliary particle filters: adapting mixture proposals via convex optimization. In *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, pages 1289–1299.

Brooks, S., Gelman, A., Jones, G. L. J., and Meng, X.-L., editors (2011). *Handbook of Markov Chain Monte Carlo*. Chapman & Hall.

Brooks, S. P. and Gelman, A. (1998). General methods for monitoring convergence of iterative simulations. *Journal of Computational and Graphical Statistics*, 7(4):434–455.

Buckland, S. T., Newman, K., Thomas, L., and Koesters, N. B. (2004). State-space models for the dynamics of wild animal populations. *Ecological Modelling*, 171(1):157–175.

Bucy, R. S. and Senne, K. D. (1971). Digital synthesis of non-linear filters. *Automatica*, 7(3):287–298.

Cappé, O., Moulines, E., and Rydén, T. (2005). *Inference in Hidden Markov Models*. Springer.

Carpenter, J., Clifford, P., and Fearnhead, P. (1999). An improved particle filter for non-linear problems. *IEE Proceedings - Radar, Sonar and Navigation*, 146(1):2–7.

Carter, C. K. and Kohn, R. (1994). On Gibbs sampling for state space models. *Biometrika*, 81(3):541–553.

Chan, J. C. C., Fry-McKibbin, R. A., and Hsiao, C. Y.-L. (2018). A regime switching skew-normal model of contagion. *Studies in Nonlinear Dynamics & Econometrics*, 23(1):20170001.

Chib, S. and Ramamurthy, S. (2010). Tailored randomized block MCMC methods with application to DSGE models. *Journal of Econometrics*, 155(1):19–38.

Chopin, N., Jacob, P. E., and Papaspiliopoulos, O. (2012). SMC2: an efficient algorithm for sequential analysis of state space models. *Journal of the Royal Statistical Society: Series B*, 75(3):397–426.

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

Chopin, N. and Singh, S. S. (2015). On particle Gibbs sampling. *Bernoulli*, 21(3):1855–1883.

Crisan, D. and Míguez, J. (2018). Nested particle filters for online parameter estimation in discrete-time state-space Markov models. *Bernoulli*, 24(4):3039–3086.

Davis, P. J. and Rabinowitz, P. (2007). *Methods of Numerical Integration, Second Edition*. Dover.

de Freitas, N., Højen-Sørensen, P., Jordan, M. I., and Russell, S. (2001). Variational MCMC. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, pages 120–127.

de Valpine, P. and Hastings, A. (2002). Fitting population models incorporating process noise and observation error. *Ecological Monographs*, 72(1):57–76.

Del Moral, P., Doucet, A., and Jasra, A. (2012). On adaptive resampling strategies for sequential Monte Carlo methods. *Bernoulli*, 18(1):252–278.

Deligiannidis, G., Doucet, A., and Rubenthaler, S. (2020). Ensemble rejection sampling. *arXiv*. https://doi.org/10.48550/arXiv.2001.09188.

Donnet, S. and Robin, S. (2017). Using deterministic approximations to accelerate SMC for posterior sampling. *arXiv*. https://doi.org/10.48550/arXiv.1707.07971.

Douc, R., Cappé, O., and Moulines, E. (2005). Comparison of resampling schemes for particle filtering. In *Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis*, pages 64–69.

Doucet, A., Gordon, N., and Krishnamurthy, V. (2001). Particle filters for state estimation of jump Markov linear systems. *IEEE Transactions on Signal Processing*, 49(3):613–624.

Doucet, A. and Johansen, A. (2009). Tutorial on particle filtering and smoothing: fifteen years later. In Crisan, D. and Rozovskii, B., editors, *The Oxford Handbook of Nonlinear Filtering*, pages 656–704. Oxford University Press.

Douwes-Schultz, D., Schmidt, A. M., Shen, Y., and Buckeridge, D. (2023). A three-state coupled Markov switching model for COVID-19 outbreaks across Quebec based on hospital admissions. *arXiv*. https://doi.org/10.48550/arXiv.2302.02488.

Driessen, H. and Boers, Y. (2004). An efficient particle filter for jump Markov nonlinear systems. *IEE Target Tracking 2004: Algorithms and Applications*, pages 19–22.

Duane, S., Kennedy, A. D., Pendleton, B. J., and Roweth, D. (1987). Hybrid Monte Carlo. *Physics Letters B*, 195(2):216–222.

Duník, J., Soták, M., Veselý, M., Straka, O., and Hawkinson, W. (2018). Design of Rao–Blackwellized point-mass filter with application in terrain aided navigation. *IEEE Transactions on Aerospace and Electronic Systems*, 55(1):251–272.

Durbin, J. and Koopman, S. J. (2012). *Time Series Analysis by State Space Methods, Second Edition*. Oxford University Press.

Ebert, A., Pudlo, P., Mengersen, K., Wu, P., and Drovandi, C. (2019). Combined parameter and state inference with automatically calibrated ABC. *arXiv*. https://doi.org/10.48550/arXiv.1910.14227.

El-Laham, Y., Yang, L., Djuric, P., and Bugallo, M. (2020). Particle filtering under general regime switching. In *2020 28th European Signal Processing Conference*, pages 2378–2382.

Elvira, V. and Martino, L. (2021). Advances in importance sampling. *Wiley StatsRef-Statistics Reference Online*. https://doi.org/10.1002/9781118445112.stat08284.

Elvira, V., Martino, L., Bugallo, M. F., and Djurić, P. M. (2018). In search for improved auxiliary particle filters. In *2018 26th European Signal Processing Conference*, pages 1637–1641.

Elvira, V., Martino, L., Luengo, D., and Bugallo, M. F. (2019). Generalized multiple importance sampling. *Statistical Science*, 34(1):129–155.

Eo, Y. and Kim, C.-J. (2016). Markov-switching models with evolving regime-specific parameters: are postwar booms or recessions all alike? *The Review of Economics and Statistics*, 98(5):940–949.

Eugenio-Martin, J. L. and Perez-Granja, U. (2020). Have low-cost carriers crowded out full services and charter carriers in tourism destinations? A trivariate structural time series analysis. *Journal of Travel Research*, 60(4):810–832.

Fearnhead, P. (2011). MCMC for state-space models. In Brooks, S., Gelman, A., Jones, G. L. J., and Meng, X.-L., editors, *Handbook of Markov Chain Monte Carlo*, pages 513–529. Chapman & Hall.

Finke, A., Doucet, A., and Johansen, A. (2016). On embedded hidden Markov models and particle Markov chain Monte Carlo methods. *arXiv*. https://doi.org/10.48550/arXiv.1610.08962.

Fischer, M. M., Hauzenberger, N., Huber, F., and Pfarrhofer, M. (2022). General Bayesian time-varying parameter vector autoregressions for modeling government bond yields. *Journal of Applied Econometrics*, 38(1):69–87.

Flury, T. and Shephard, N. (2009). Learning and filtering via simulation: smoothly jittered particle filters. In *Department of Economics Discussion Paper Series, University of Oxford*, volume 469, pages 1–27.

Fotiadis, A., Polyzos, S., and Huan, T.-C. T. C. (2021). The good, the bad and the ugly on COVID-19 tourism recovery. *Annals of Tourism Research*, 87(3):103–117.

Frazier, D. T., Loaiza-Maya, R., and Martin, G. M. (2022). Variational Bayes in state space models: inferential and predictive accuracy. *Journal of Computational and Graphical Statistics*, 32(3):793–804.

Frühwirth-Schnatter, S. (2004). Efficient Bayesian parameter estimation. In *State Space and Unobserved Component Models: Theory and Applications*, pages 123–151. Cambridge University Press.

Frühwirth-Schnatter, S. (2001). Fully Bayesian analysis of switching Gaussian state space models. *Annals of the Institute of Statistical Mathematics*, 53(1):31–49.

Fulop, A. and Li, J. (2013). Efficient learning via simulation: a marginalized resample-move approach. *Journal of Econometrics*, 176(2):146–161.

Gamerman, D. and Lopes, H. F. (2006). Markov chains. In *Markov Chain Monte Carlo: Stochastic Simulation for Bayesian Inference, Second Edition*, pages 113 – 139. Chapman & Hall.

Gelfand, A. E., Hills, S. E., Racine-Poon, A., and Smith, A. F. M. (1989). Illustration of Bayesian inference in normal data models using Gibbs sampling. *Journal of the American Statistical Association*, 85(412):972–985.

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., and Rubin, D. B. (2013). Basics of Markov chain simulation. In *Bayesian Data Analysis, Third Edition*. Chapman & Hall.

Gelman, A. and Rubin, D. B. (1992). Inference from iterative simulation using multiple sequences. *Statistical Science*, 7(4):457–472.

Geman, S. and Geman, D. (1993). Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *Journal of Applied Statistics*, 20(5):25–62.

Geweke, J. (1989). Bayesian inference in econometric models using Monte Carlo integration. *Econometrica*, 57(6):1317–1339.

Geyer, C. J. (2011). Introduction to Markov chain Monte Carlo. In Brooks, S., Gelman, A., Jones, G. L. J., and Meng, X.-L., editors, *Handbook of Markov Chain Monte Carlo*, pages 3–48. Chapman & Hall.

Ghahramani, Z. and Beal, M. (1999). Variational inference for Bayesian mixtures of factor analysers. In *Advances in Neural Information Processing Systems 12*, pages 449–455.

Ghahramani, Z. and Jordan, M. (1997). Factorial hidden Markov models. *Machine Learning*, 29(2):245–273.

Gilks, W. R. and Berzuini, C. (2002). Following a moving target - Monte Carlo inference for dynamic Bayesian models. *Journal of the Royal Statistical Society: Series B*, 63(1):127–146.

Gilks, W. R., Roberts, G. O., and George, E. I. (1994). Adaptive direction sampling. *Journal of the Royal Statistical Society: Series D*, 43(1):179–189.

Google LLC (2024). Google Trends data for search query volumes. *Google Trends* *https://trends.google.com/trends/*.

Gordon, N. J., Salmond, D. J., and Smith, A. F. M. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *IEE Proceedings - Radar, Sonar and Navigation*, volume 140, pages 107–113.

Haario, H., Saksman, E., and Tamminen, J. (1999). Adaptive proposal distribution for random walk Metropolis algorithm. *Computational Statistics*, 14(3):375–395.

Haario, H., Saksman, E., and Tamminen, J. (2001). An adaptive Metropolis algorithm. *Bernoulli*, 7(2):223–242.

Haimerl, P. and Hartl, T. (2023). Modeling COVID-19 infection rates by regime-switching unobserved components models. *Econometrics*, 11(2):1–15.

Hamilton, J. D. (1989). A new approach to the economics analysis of nonstationary time series and the business cycle. *Econometrica*, 57(2):357–384.

Harvey, A. C. (1990). *Forecasting, Structural Time Series Models and the Kalman Filter*. Cambridge University Press.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109.

He, M., Das, P., Hotan, G., and Purdon, P. L. (2023). Switching state-space modeling of neural signal dynamics. *Public Library of Science Computational Biology*, 19(8):1011395.

Henriksen, S., Wills, A., Schön, T. B., and Ninness, B. (2012). Parallel implementation of particle MCMC methods on a GPU. In *16th IFAC Symposium on System Identification*, pages 1143–1148.

Hirt, M., Dellaportas, P., and Durmus, A. (2019). Copula-like variational inference. In *Advances in Neural Information Processing Systems 32*, volume 32, pages 1–13.

Hirt, M., Titsias, M., and Dellaportas, P. (2021). Entropy-based adaptive Hamiltonian Monte Carlo. In *Advances in Neural Information Processing Systems, 34*, pages 28482–28495.

Hobert, J. P. (2011). The data augmentation algorithm: theory and methodology. In Brooks, S., Gelman, A., Jones, G. L. J., and Meng, X.-L., editors, *Handbook of Markov Chain Monte Carlo*, pages 253–293. Chapman & Hall.

Höpken, W., Eberle, T., Fuchs, M., and Lexhagen, M. (2020). Improving tourist arrival prediction: a big data and artificial neural network approach. *Journal of Travel Research*, 60(5):998–1017.

Jin, S.-S., Ju, H., and Jung, H.-J. (2019). Adaptive Markov chain Monte Carlo algorithms for Bayesian inference: recent advances and comparative study. *Structure and Infrastructure Engineering*, 15(11):1548–1565.

Johansen, A. M. (2010). Monte Carlo Methods. In *International Encyclopedia of Education, Third Edition*, pages 296–303. Elsevier.

Jolliffe, I. T. and Cadima, J. (2016). Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A*, 374(2065):20150202.

Jones, G. L. and Qin, Q. (2022). Markov chain Monte Carlo in practice. *Annual Review of Statistics and Its Application*, 9(1):557–578.

Jorge-González, E., González-Dávila, E., Martín-Rivero, R., and Lorenzo-Díaz, D. (2019). Univariate and multivariate forecasting of tourism demand using state-space models. *Tourism Economics*, 26(4):598–621.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45.

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.

Karlsson, R. and Bergman, N. (2000). Auxiliary particle filters for tracking a maneuvering target. *Proceedings of the 39th IEEE Conference on Decision and Control*, pages 3891–3895.

Kartal, M. T., Ayhan, F., and Kirikkaleli, D. (2022). Regime-switching effect of COVID-19 pandemic on stock market index: evidence from Turkey as an emerging market example. *Macroeconomics and Finance in Emerging Market Economies*, pages 1–18.

Kim, C. J. and Nelson, C. R. (1999). *State-Space Models with Regime Switching: Classical and Gibbs-Sampling Approaches with Applications*. MIT Press.

Kim, J. (2015). Bayesian inference in a non-linear/non-Gaussian switching state space model: regime-dependent leverage effect in the U.S. stock market. *Munich Personal RePEc Archive*. https://mpra.ub.uni-muenchen.de/67153/.

Kim, J. R. and Cho, S. (2022). Developing a regime-switching present value model: switching fundamentals and bubbles. *International Economic Journal*, 36(4):477–490.

King, R. (2012). A review of Bayesian state-space modelling of capture-recapture-recovery data. *Interface Focus*, 2(2):190–204.

King, R., Morgan, B. J. T., Gimenez, O., and Brooks, S. P. (2010). *Bayesian Analysis for Population Ecology*. Chapman & Hall.

King, R., Sarzo, B., and Elvira, V. (2023). When ecological individual heterogeneity models and large data collide: an importance sampling approach. *Annals of Applied Statistics*, 17(4):3112–3132.

Kitagawa, G. (1987). Non-Gaussian state-space modeling of nonstationary time series. *Journal of the American Statistical Association*, 82(400):1032–1041.

Kitagawa, G. (1998). A self-organizing state-space model. *Journal of the American Statistical Association*, 93(443):1203–1215.

Koopman, S. J. and Bos, C. S. (2004). State space models with a common stochastic variance. *Journal of Business and Economic Statistics*, 22(3):346–357.

Kristensen, K., Nielsen, A., Berg, C. W., Skaug, H., and Bell, B. M. (2016). TMB: automatic differentiation and Laplace approximation. *Journal of Statistical Software*, 70(5):1–21.

Langrock, R. (2011). Some applications of nonlinear and non-Gaussian state–space modelling by means of hidden Markov models. *Journal of Applied Statistics*, 38(12):2955–2970.

Langrock, R. and King, R. (2013). Maximum likelihood estimation of mark-recapture-recovery models in the presence of continuous covariates. *Annals of Applied Statistics*, 7(3):1709–1732.

Langrock, R., MacDonald, I. L., and Zucchini, W. (2012). Some nonstandard stochastic volatility models and their estimation using structured hidden Markov models. *Journal of Empirical Finance*, 19(1):147–161.

Lanouar, C. and Goaied, M. (2019). Tourism, terrorism and political violence in Tunisia: evidence from Markov-switching models. *Tourism Management*, 70:404–418.

Latuszyński, K., Roberts, G. O., and Rosenthal, J. S. (2013). Adaptive Gibbs samplers and related MCMC methods. *Annals of Applied Probability*, 23(1):66–98.

Law, R., Li, G., Fong, D. K. C., and Han, X. (2019). Tourism demand forecasting: a deep learning approach. *Annals of Tourism Research*, 75:410–423.

Lawrence, P. (2020). Tourism and hospitality sector recovery plan - follow up. *Policy and Sustainability Comittee, City of Edinburgh Council*. https://democracy.edinburgh.gov.uk/documents/s24704/Item%206.5%20-%20Tourism%20and%20Hospitality%20Sector%20Recovery%20Plan%20Follow%20Up.pdf.

Li, T., Sun, S., Sattar, T. P., and Corchado, J. M. (2014). Fight sample degeneracy and impoverishment in particle filters: a review of intelligent approaches. *Expert Systems with Applications*, 41(8):3944–3954.

Li, X., Law, R., Xie, G., and Wang, S. (2021). Review of tourism forecasting research with internet data. *Tourism Management*, 83:104245.

Li, X., Pan, B., Law, R., and Huang, X. (2017). Forecasting tourism demand with composite search index. *Tourism Management*, 59:57–66.

Liang-qun, L., Wei-xin, X., Jing-xiong, H., and Jianjun, H. (2009). Multiple model Rao-Blackwellized particle filter for manoeuvring target tracking. *Defence Science Journal*, 59(3):197–204.

Lin, A., Zhang, Y., Heng, J., Allsop, S. A., Tye, K. M., Jacob, P. E., and Ba, D. (2019). Clustering time series with nonlinear dynamics: a Bayesian non-parametric and particle-based approach. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Laplace Approximation*, pages 2476–2484.

Lindsten, F., Jordan, M. I., and Schön, T. B. (2014). Particle Gibbs with ancestor sampling. *Journal of Machine Learning Research*, 15(63):2145–2184.

Liu, J. and West, M. (2001). Combined Parameter and State Estimation in Simulation-Based Filtering. In Doucet, A., de Freitas, N., and Gordon, N., editors, *Sequential Monte Carlo Methods in Practice*, pages 197–223. Springer.

Liu, J. S. (2004). *Monte Carlo Strategies in Scientific Computing*. Springer.

Llewellyn, M., Ross, G., and Ryan-Saha, J. (2023). COVID-era forecasting: Google trends and window and model averaging. *Annals of Tourism Research*, 103:103660.

Ma, X., Yang, Z., and Zheng, J. (2022). Analysis of spatial patterns and driving factors of provincial tourism demand in China. *Scientific Reports*, 12(1):2260.

Maire, F., Friel, N., Mira, A., and Raftery, A. E. (2019). Adaptive incremental mixture Markov chain Monte Carlo. *Journal of Computational and Graphical Statistics*, 28(4):790–805.

Martino, L. (2017). Parsimonious adaptive rejection sampling. *Electronics Letters*, 53(16):1115–1117.

Martino, L., Read, J., Elvira, V., and Louzada, F. (2017). Cooperative parallel particle filters for online model selection and applications to urban mobility. *Digital Signal Processing*, 60:172–185.

Massicotte, P. and Eddelbuettel, D. (2022). gtrendsR: perform and display google trends queries. *CRAN, R package version 1.5.1*. https://CRAN.R-project.org/package=gtrendsR.

Matoušek, J., Duník, J., and Brandner, M. (2023). Design of efficient point-mass filter with application in terrain aided navigation. *arXiv*. https://doi.org/10.48550/arXiv.2303.05100.

Matoušek, J., Duník, J., and Straka, O. (2020). Density difference grid design in a point-mass filter. *Energies*, 13(16):1–18.

McClintock, B. T., Langrock, R., Gimenez, O., Cam, E., Borchers, D. L., Glennie, R., and Patterson, T. A. (2020). Uncovering ecological state dynamics with hidden Markov models. *Ecology Letters*, 23(12):1878–1903.

McGinnity, S. and Irwin, G. (2000). Multiple model bootstrap filter for maneuvering target tracking. *IEEE Transactions on Aerospace and Electronic Systems*, 36(3):1006–1012.

Meent, J.-W., Yang, H., Mansinghka, V., and Wood, F. (2015). Particle Gibbs with ancestor sampling for probabilistic programs. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, pages 986–994.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21(6):1087–1092.

Murphy, J. and Godsill, S. J. (2016). Blocked particle Gibbs schemes for high dimensional interacting systems. *IEEE Journal of Selected Topics in Signal Processing*, 10(2):328–342.

Nadal-De Simone, F. (2000). Forecasting inflation in Chile using state-space and regime-switching models. In *Working Papers of the International Monetary Fund*. https://www.imf.org/en/Publications/WP/Issues/2016/12/30/Forecasting-Inflation-in-Chile-Using-State-Space-and-Regime-Switching-Models-3835.

Neal, R. M. (2003). Markov chain sampling for non-linear state space models using embedded hidden Markov models. *arXiv*. https://doi.org/10.48550/arXiv.math/0305039.

Newman, K., Buckland, S. T., Morgan, B. J. T., King, R., Borchers, D. L., Cole, D. J., Besbeas, P., Gimenez, O., and Thomas, L. (2014). *Modelling Population Dynamics: Model Formulation, Fitting and Assessment using State-space Methods*. Springer.

Newman, K., King, R., Elvira, V., de Valpine, P., McCrea, R. S., and Morgan, B. J. T. (2022). State-space models for ecological time series data: practical model-fitting. *Methods in Ecology and Evolution*, 14(1):26–42.

Newman, K. B. (1998). State-space modeling of animal movement and mortality with application to salmon. *Biometrics*, 54(4):1290–1314.

Niemi, J. and West, M. (2010). Adaptive mixture modeling Metropolis methods for Bayesian analysis of nonlinear state-space models. *Journal of Computational and Graphical Statistics*, 19(2):260–280.

Nonejad, N. (2015). Particle Gibbs with ancestor sampling for stochastic volatility models with: heavy tails, in mean effects, leverage, serial dependence and structural breaks. *Studies in Nonlinear Dynamics & Econometrics*, 19(5):561–584.

OECD (2020). Mitigating the impact of COVID-19 on tourism and supporting recovery. In *OECD Tourism Papers*. https://doi.org/10.1787/23071672.

Onizuka, T., Hashimoto, S., and Sugasawa, S. (2023). Fast and locally adaptive Bayesian quantile smoothing using calibrated variational approximations. *Statistics and Computing*, 34(15):1–16.

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

Pitt, M. K. and Shephard, N. (2001). Auxiliary variable based particle filters. In *Sequential Monte Carlo Methods in Practice*, pages 273–293. Springer.

Plataniotis, K. N. and Hatzinakos, D. (2001). Gaussian mixtures and their applications to signal processing. In *Advanced Signal Processing Handbook*. CRC Press.

Pérez-Vieites, S. and Míguez, J. (2021). Nested Gaussian filters for recursive Bayesian inference and nonlinear tracking in state space models. *Signal Processing*, 189:108295.

Rabiner, L. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, volume 77, pages 257–286.

Rabiner, L. and Juang, B. (1986). An introduction to hidden Markov models. *IEEE ASSP Magazine*, 3(1):4–16.

Radivojević, T. and Akhmatskaya, E. (2020). Modified Hamiltonian Monte Carlo for Bayesian inference. *Statistics and Computing*, 30(2):377–404.

Rainforth, T., Naesseth, C. A., Lindsten, F., Paige, B., van de Meent, J.-W., Doucet, A., and Wood, F. (2016). Interacting particle Markov chain Monte Carlo. In *Proceedings of the 33rd International Conference on Machine Learning*, pages 2616–2625.

Rao, V., Lin, L., and Dunson, D. B. (2016). Data augmentation for models based on rejection sampling. *Biometrika*, 103(2):319–335.

Rimella, L. and Whiteley, N. (2022). Exploiting locality in high-dimensional factorial hidden Markov models. *Journal of Machine Learning Research*, 23(4):1–34.

Rivera, R. (2016). A dynamic linear model to forecast hotel registrations in Puerto Rico using Google Trends data. *Tourism Management*, 57:12–20.

Rosenthal, J. S. (2011). Optimal proposal distributions and adaptive MCMC. In Brooks, S., Gelman, A., Jones, G. L. J., and Meng, X.-L., editors, *Handbook of Markov Chain Monte Carlo*, pages 93 – 111. Chapman & Hall.

Scott, S. and Varian, H. (2014). Predicting the present with Bayesian structural time series. *International Journal of Mathematical Modelling and Numerical Optimisation*, 5(1):4–23.

Seong, B. and Lee, K. (2021). Intervention analysis based on exponential smoothing methods: applications to 9/11 and COVID-19 effects. *Economic Modelling*, 98:290–301.

Shephard, N. and Pitt, M. K. (1997). Likelihood analysis of non-Gaussian measurement time series. *Biometrika*, 84(3):653–667.

Shestopaloff, A. Y. and Neal, R. M. (2013). MCMC for non-linear state space models using ensembles of latent sequences. *arXiv*. https://doi.org/10.48550/arXiv.1305.0320.

Shestopaloff, A. Y. and Neal, R. M. (2018). Sampling latent states for high-dimensional non-linear state space models with the embedded HMM method. *Bayesian Analysis*, 13(3):797–822.

Smidl, V. and Gasperin, M. (2013). Rao-Blackwellized point mass filter for reliable state estimation. *Proceedings of the 16th International Conference on Information Fusion*, pages 312–318.

So, M. K. P., Lam, K., and Li, W. K. (1998). A stochastic volatility model with Markov switching. *Journal of Business & Economic Statistics*, 16(2):244–253.

Song, H. and Li, G. (2008). Tourism demand modelling and forecasting - A review of recent research. *Tourism Management*, 29(2):203–220.

Tanner, M. A. and Wong, W. H. (1987). The calculation of posterior distributions by data augmentation. *Journal of the American Statistical Association*, 82(398):528–540.

Tidefelt, H. and Schön, T. B. (2009). Robust point-mass filters on manifolds. In *15th IFAC Symposium on System Identification*, pages 540–545.

Tierney, L. (1994). Markov Chains for exploring posterior distributions. *The Annals of Statistics*, 22(4):1701–1728.

Titsias, M. and Dellaportas, P. (2019). Gradient-based adaptive Markov chain Monte Carlo. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 15730–15739.

Tran, M. N. and Kohn, R. (2015). Exact ABC using importance sampling. *arXiv*. https://doi.org/10.48550/arXiv.1509.08076.

Triantafyllopoulos, K. (2021). *Bayesian Inference of State Space Models: Kalman Filtering and Beyond*. Springer.

UNWTO (2022). Tourism recovery gains momentum as restrictions ease and confidence returns. *United Nations World Tourism Organization*. Retrieved 3rd February 2023 from the UNWTO: https://www.unwto.org/news/tourism-recovery-gains-momentum-as-restrictions-ease-and-confidence-returns#:~:text=Tourism%20Recovery%20Gains%20Momentum%20as%20Restrictions%20Ease%20and%20Confidence%20Returns,-All%20Regions&text=Tourism%20continues%20to%20recover%20at,Europe%20leading%20the%20sector's%20rebound.

Urteaga, I., Bugallo, M. F., and Djurić, P. M. (2016). Sequential Monte Carlo methods under model uncertainty. In *2016 IEEE Statistical Signal Processing Workshop*, pages 1–5.

van der Merwe, R., Wan, E., and Julier, S. (2004). Sigma-point Kalman filters for nonlinear estimation and sensor-fusion applications to integrated navigation. In *Proceedings of the AIAA Guidance, Navigation & Control Conference*, pages 1–30.

van Erp, S., Oberski, D. L., and Mulder, J. (2019). Shrinkage priors for Bayesian penalized regression. *Journal of Mathematical Psychology*, 89:31–50.

Vergé, C., Dubarry, C., Del Moral, P., and Moulines, E. (2013). On parallel implementation of sequential Monte Carlo methods: The island particle model. *Statistics and Computing*, 25(2):243–260.

Wang, B. and Titterington, D. M. (2004). Lack of consistency of mean field and variational Bayes approximations for state space models. *Neural Processing Letters*, 20(3):151–170.

Wang, H., Bhattacharya, A., Pati, D., and Yang, Y. (2022). Structured variational inference in Bayesian state-space models. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, pages 8884–8905.

Wang, X., Li, T., Sun, S., and Corchado Rodríguez, J. (2017). A survey of recent advances in particle filters and remaining challenges for multitarget tracking. *Sensors*, 17(12):2707.

Wigren, A., Risuleo, R. S., Murray, L. M., and Lindsten, F. (2019). Parameter elimination in particle Gibbs sampling. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 1–12.

Wood, S. N. (2010). Statistical inference for noisy nonlinear ecological dynamic systems. *Nature*, 466(7310):1102–1104.

World Travel & Tourism Council (2022). Travel & tourism: economic impact 2022. In *Global Economic Impact Reports, World Travel & Tourism Council*. https://wttc.org/Portals/0/Documents/Reports/2022/EIR2022-Global%20Trends.pdf.

Wu, D., Song, H., and Shen, S. (2017). New developments in tourism and hotel demand modeling and forecasting. *International Journal of Contemporary Hospitality Management*, 29(1):507–529.

Xiaoxuan, L., Qi, W., Geng, P., and Benfu, L. (2016). Tourism forecasting by search engine data with noise-processing. *African Journal of Business Management*, 10(6):114–130.

Xu, D. (2019). Learning nonlinear state space models with Hamiltonian sequential Monte Carlo sampler. *arXiv*. https://doi.org/10.48550/arXiv.1901.00862.

Yang, X., Pan, B., Evans, J., and Lv, B. (2015). Forecasting Chinese tourist volume with search engine data. *Tourism Management*, 46:386–397.

Zeng, Y. and Wu, S., editors (2013). *State-Space Models: Applications in Economics and Finance*. Springer.

Zhang, H., Song, H., Wen, L., and Liu, C. (2021). Forecasting tourism recovery amid COVID-19. *Annals of Tourism Research*, 87(4):103149.

Zhang, S., Sun, T., and Lu, Y. (2023). The COVID-19 pandemic and tourists' risk perceptions: tourism policies' mediating role in sustainable and resilient recovery in the new normal. *Sustainability*, 15(2):1323.