Bayesian Inference for Continuous Time Markov Chains

A thesis submitted to the University of Glasgow in fulfilment of the requirements for the award of the degree of Doctor of Philosophy in the College of Science and Engineering

Randa Alharbi

School of Mathematics and Statistics

University of Glasgow

December 2018

Abstract

Continuous time Markov chains (CTMCs) are a flexible class of stochastic models that have been employed in a wide range of applications from timing of computer protocols, through analysis of reliability in engineering, to models of biochemical networks in molecular biology. These models are defined as a state system with continuous time transitions between the states. Extensive work has been historically performed to enable convenient and flexible definition, simulation, and analysis of continuous time Markov chains. This thesis considers the problem of Bayesian parameter inference on these models and investigates computational methodologies to enable such inference. Bayesian inference over continuous time Markov chains is particularly challenging as the likelihood cannot be evaluated in a closed form. To overcome the statistical problems associated with evaluation of the likelihood, advanced algorithms based on Monte Carlo have been used to enable Bayesian inference without explicit evaluation of the likelihoods. An additional class of approximation methods has been suggested to handle such inference problems, known as approximate Bayesian computation. Novel Markov chain Monte Carlo (MCMC) approaches were recently proposed to allow exact inference.

The contribution of this thesis is in discussion of the techniques and challenges in implementing these inference methods and performing an extensive comparison of these approaches on two case studies in systems biology. We investigate how the algorithms can be designed and tuned to work on CTMC models, and to achieve an accurate estimate of the posteriors with reasonable computational cost. Through this comparison, we investigate how to avoid some practical issues with accuracy and computational cost, for example by selecting an optimal proposal distribution and introducing a resampling step within the sequential Monte-Carlo method. Within the implementation of the ABC methods we investigate using an adaptive tolerance schedule to maximise the efficiency of the algorithm and in order to reduce the computational cost.

Declaration of Independence

I declare that all the work presented in this thesis has been done by myself under the supervision of Dr. Vladislav Vyshemirsky and Prof. Dirk Husmeier, except where otherwise stated. This thesis represents work completed, between 2014 and 2018 in Statistics in the School of Mathematics and Statistics at the University of Glasgow.

Acknowledgements

A huge thank my supervisor Dr.Vladislav Vyshemirsky for his scientific guidance, support and encouragement over the duration of my PhD. Without him, this work would have never been accomplished. I truly could not have hoped for better supervisor. I am deeply grateful to you, Dr.Vlad.

Many thanks to Prof.Dirk Husmeier, for for discussions, scientific guidance and general support.

I want to thank my examiners, Prof.Guido Sanguinetti and Dr.Vincent Macaulay for providing valuable suggestions.

I also would like to thank the academic staff of the school for providing a good and friendly environment for the Ph.D. students. Special thanks to Prof. Adrian Bowman for his help, support and advice.

I owe thanks to Heather Lambie (Graduate School Manager) and her team for their administrative guidance during my PhD.

A big thanks to my colleague Suzy Whoriskey and Eilidh Jack, they have been amazing company. I also would like to thank my friends Amani, Salihah, Shuhrah for their love and kindness.

A lot of thanks to the Ministry of Higher Education and my employer Tabuk University for funding my work. I would like also to thank the Saudi Arabian Cultural Bureau for their help and support throughout my study.

Many thanks for my parents and family for their love, prayers, support and encouragement. And most of the thanks to my kids and my husband for love, support and being patient.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Introduction and Thesis Statement

This thesis considers the performance of Bayesian Inference of model parameters for Continuous Time Markov Chains (CTMC). CTMCs are a flexible class of stochastic models that consider a discrete state space with stochastic transitions in continuous time. These models have been widely applied in such fields as analysis of communication protocols (Duflot et al., 2006), reliability analysis (Haverkort et al., 2000), power management (Qiu et al., 1999), and modelling biological systems (Calder et al., 2006).

A large part of the literature focusing on an analysis of CMTC considers the model to be completely observed. In more realistic situations, the problem of identifying model parameters to match the behaviour of the studied system is significantly more challenging (Milios et al., 2017).

When modelling a biological system, a set of ordinary differential equations (ODEs) is often used as a deterministic description of the system. These equations involve a concentration of species and parameters such as mRNA, protein, degradation and production rates. These equations provide an accurate description of the biological system when the number of molecules is large. Nevertheless, when the population of molecules is small, the impact of discrete stochastic behaviour is obvious, and the accuracy of this method becomes degraded (Schnoerr et al., 2017).

In addition, in practical studies in life sciences, it is difficult to observe every state and measure all parameters of the complex biological model. This motivates researchers to develop computational and mathematical approaches to help to under-

stand the systems' mechanisms, behaviours and account for stochasticity. Moreover, a variety of inference methods have been proposed to quantify uncertainty relating to such systems.

Probabilistic Bayesian approaches have been used to quantify uncertainties in such systems, including work described in Golightly and Wilkinson (2005) and (Vyshemirsky and Girolami, 2008). Other studies have been considering using maximum likelihood estimation (Baker et al., 2005), (Timmer et al., 2004). When working with state space models, several sequential approaches have been proposed to handle state and parameter estimation problem such as particle filtering (Quach et al., 2007).

We consider the Bayesian approach to performing parameter inference because it allows quantifying uncertainties about inference results. Additionally, in cases when data provide little information for reliable parameter inference, for example, at the beginning of a new study, the Bayesian approach allows one to use pre-existing expert knowledge as an additional source of information via parameter prior distributions.

The core problem of enabling parameter inference for state space models is the difficulty in formulating the likelihood in a convenient form. Because the scale of the system grows, the likelihood function becomes intractable to evaluate. To cope with this, approximate methods can be applied that often rely on either variational inference or simulation. Our primary focus in this thesis is on one particular approach that is based on statistical simulation.

This thesis does not intend to provide a complete overview of other inference approaches for CTMC models, but we will briefly mention some of them. A recent review included the theory and inference methods for a Markov process in a biological modelling context (Schnoerr et al., 2017).

The work presented in this thesis approaches the problem of Bayesian parameter inference using approximate inference methods that avoid direct evaluation of the likelihood. Simulation from the model is used instead to approximate the likelihood via an unbiased estimator based on Monte Carlo integration. We consider two general sampling schemes that implement this approach in two slightly different forms. The Particle Marginal Metropolis Hastings sampler (Wilkinson, 2011) performs likelihood estimation using a simulation based particle filter, while the Approximate Bayesian Computation with Sequential Monte Carlo (ABC SMC) (Sisson et al., 2007a), (Peters et al., 2012) sampler relies on repeated resampling of a particle population along a sequence of gradually improving approximating distributions. A pseudo-marginal sampler based on truncation is new to the statistics field and

provided exact inference for such systems (Georgoulas et al., 2017). We employ this method as a reference for our case study.

We implement these sampling schemes, study their properties and tuning parameters, and perform a comparison of their performance in two complex case studies. The accuracy of the results obtained from these approaches is verified by comparison to the exact method.

1.2 Thesis Contribution

In this thesis, we demonstrate that CTMC can be applied to modelling complex stochastic systems. Uncertainty quantification for those stochastic systems is studied. In particular, parameter estimation with tuning algorithmic parameters demonstrates how these methods work in practice.

We provide two extensive studies where CTMC are used to model molecular reaction networks in biology. The main contributions of this thesis can be summarised as:

- We give an up-to-date review of the state-of-the-art methods for Bayesian inference without explicit likelihood.
- We study such methods in detail and discuss the tuning of these algorithms. Specifically, the choice of the number of particles in the particle filter, selection of the optimal proposal distribution and convergence diagnostics in the Particle Marginal Metropolis Hastings sampler are studies to improve the exploration of the parameter space and hence the efficiency of the algorithm.
- An extensive comparison between the exact inference approach and approximation approaches involving the accuracy, complexity of the implementation and computational costs was performed.
- We evaluate these methods by making inference for a challenging stochastic model with intractable likelihood. We utilise the Particle Marginal Metropolis Hastings sampler and Approximate Bayesian Computation on the Lotka Volterra model. Moreover, we resort to the recently developed exact method, which is known as a Gibbs sampler based on truncation to quantify the accuracy of approximation methods.
- Motivated by the high variability of the system behaviour, the Repressilator system is chosen to utilise the proposed inference methods involving both

synthetic and real data.

1.3 Outline of the Thesis

This thesis is divided into seven chapters. A brief overview of each chapter and a description of the general thesis structure is given below.

In Chapter 2: We provide the reader with an overview of Markov processes, covering the main concepts and definitions. We also introduce the Continuous Time Markov Chain models relevant to this study and define how CTMC can be used to model biochemical reaction systems.

In Chapter 3: We consider the Bayesian inference framework and also discuss the main issues to deal with when working within this framework. A general review of Monte Carlo methods is given. Possible inference problems, such as the intractability of the likelihood term, are illustrated.

In Chapter 4: We describe the Approximate Bayesian Computation scheme, introduce the ABC SMC sampler, and discuss tuning parameters for this method.

In Chapter 5: We consider the first case study concerning modelling of the stochastic Lotka-Volterra system, perform parameter inference for this case study using approximate sampling algorithms, and compare them to the exact method.

In Chapter 6: We consider the second case study concerning a more complex biochemical system known as the Repressilator (Elowitz and Leibler, 2000). First, we investigate how the algorithm tuning parameters impact the inference results. Then we perform a validation study using a real data set.

In Chapter 7: We review and discuss the results obtained in this work. Possible further studies and their direction are suggested.

Chapter 2

Markov Chains

Most natural phenomena are difficult to observe directly. Statistical models, on the other hand, can be used to understand and predict the behaviour of such systems. Often, these phenomena involve randomness, which can be quantified through probability theory. A random phenomenon can be described by evaluation of a random variable in time. Numerous probabilistic models are applied to explain the system's behaviour. Some stochastic processes have a memoryless property, meaning that the future state of the system can rely only on its present state, independent of its whole past history. Such a stochastic process having a memoryless property was introduced and studied by a Russian mathematician, Andrey Markov. This stochastic process is called a Markov process. Markov processes can be classified based on their timing and state. The main types of the Markov process are a discrete time Markov Chain (DTMC) and a continuous time Markov Chain (CTMC) (Bhattacharya and Waymire, 2009). There are several applications of Markov chains in various scientific fields. In this thesis, in an attempt to understand the Markov chain, we begin with a brief introduction to the Markov process, covering the basic concepts, giving a definition, and describing the theorems and essential properties. We continue this chapter by considering a DTMC, its structure and main properties, where a DTMC is characterised by a discrete state and time, which are assumed to be homogeneous (see section 2.2). In addition, essential features of the Markov chain necessary for studying the chain's long term behaviour are discussed and explained. A CTMC will also be considered in detail as it serves as the main practical model in this thesis (Norris, 1998), (Ross, 2014).

2.1 Stochastic Process

A stochastic process is a vector of random variables, which can be indexed by discrete nonnegative integer e.g. $\{X_1, X_2, \cdots\}$ or $\{X_t : t \in \mathbb{N}\}$. The state of the system at discrete time t is denoted as X_t and takes values in a countable and finite state space $\mathcal{X}$. The random variable can also be indexed by continuous time e.g $\{X(t) : t \in \mathbb{R}\}$.

2.2 Markov Chains in a Discrete State Space

Definition 2.2.1. (Markov Chain)

A stochastic process $\{X_t : t \in \mathbb{N}\}$ is defined as a discrete time Markov chain if it satisfies the Markov property:

$$P\left(X_{t+1} = j | X_0 = x_0, \cdots, X_t = i\right) = P(X_{t+1} = j | X_t = i),$$

where $i, j, x_0, \cdots, x_{t-1} \in \mathcal{X}$ and $t \in \mathbb{N}$.

The previous equation means that the whole past history before state X_t has been forgotten, which is known as the memoryless property. Thus, the conditional probability of state X_{t+1} is independent of all past states $X_0, X_1, \cdots, X_t$ and depends only on the current state X_t. A process with such properties is referred to as a Markov process and is commonly called a Markov chain.

Definition 2.2.2. (Homogeneous Markov chain)

A homogeneous Markov chain can be described as a conditional probability that does not depend on the current time,

$$\forall m \in \mathbb{N} : P(X_{t+m} = j | X_{t+m-1} = i) = P(X_t = j | X_{t-1} = i).$$

$P(X_{t+1} = j | X_t = i)$ is the probability of the process moving from state i to the next state j in a unit time and is known as a one-step transition probability, denoted as: p_{ij}.

For a homogeneous DTMC, let us assume that there is a probability governing the transition between the states, denoted as p_{ij}. Thus, a one-step transition probability is given as:

$$P(X_{t+1} = j | X_t = i) = p_{ij}, \quad \text{where} \quad i, j = 1, 2, 3, \cdots . \tag{2.1}$$

All possible transition probabilities p_{ij} can be defined as matrix $\mathbf{P}$.

2.2.1 Transition Matrix

The transition probability p_{ij} is interpreted as the probability that the process can change randomly from one state to another one, and it can be presented as a matrix.

Definition 2.2.3. (Probability transition matrix)

Given the fact that a state space $\mathcal{X}$ contains N states, the transition matrix is defined as an $N \times N$ matrix with nonnegative entries, where all rows add up to 1, denoted as $\mathbf{P}$:

$$\mathbf{P} = \begin{pmatrix} p_{11} & \cdots & p_{1N} \\ p_{21} & \cdots & p_{2N} \\ \vdots & \ddots & \vdots \\ p_{N1} & \cdots & p_{NN} \end{pmatrix},$$

where the ith row of the transition matrix $\mathbf{P}$ represents the probabilities of moving out from state i to another state. The column j of $\mathbf{P}$ expresses the transition probability into state j.

2.2.2 Chapman-Kolmogorov Equation

A 1-step transition probability p_{ij} was defined in 2.1. However, in order to understand the path of transition in a Markov chain, it is useful to describe the probability of jumping from state i to another state j in m steps, making use of intermediate states. To illustrate this, we begin with a simple case when $m = 2$, so the probability of transition between states i and j can be calculated in two steps via a third intermediate state r:

$$
\begin{aligned}
p_{ij}^2 &= P(X_2 = j | X_0 = i) \\
&= \sum_r P(X_2 = j, X_1 = r | X_0 = i) \quad \text{By law of total probability} \\
&= \sum_r P(X_2 = j | X_1 = r, X_0 = i) P(X_1 = r | X_0 = i) \quad \text{By product rule} \\
&= \sum_r P(X_2 = j | X_1 = r) P(X_1 = r | X_0 = i) \quad \text{By Markov property} \\
&= \sum_r p_{rj} p_{ir} \\
&= \sum_r p_{ir} p_{rj}.
\end{aligned} \tag{2.2}
$$

This can be generalised to compute a m-step transition probability as:

$$
p_{ij}^m = \sum_r p_{ir}^{m-1} p_{rj}. \tag{2.3}
$$

These are known as the Chapman-Kolmogorov equations that are used to compute the probabilities, implying that the process arrives into a specific state after m steps through intermediate steps.

Theorem 2.2.1. (The Chapman-Kolmogorov equations)

In a finite DTMC, given the two states i at time 0 and j at time $m+t$ with time m and t, the transition probability between the states is given as:

$$
p_{ij}^{m+t} = \sum_r p_{ir}^m p_{rj}^t.
$$

Proof. In order to prove the Chapman-Kolmogorov equations, we use the law of total probability, the product rule and the Markov property, which results in the following:

$$\begin{aligned} p_{ij}^{m+t} &= P(X_{m+t} = j | X_0 = i) \\ &= \sum_r P(X_{m+t} = j, X_m = r | X_0 = i) \\ &= \sum_r P(X_{m+t} = j | X_m = r, X_0 = i) P(X_m = r | X_0 = i) \\ &= \sum_r P(X_{m+t} = j | X_m = r) P(X_m = r | X_0 = i) \\ &= \sum_r p_{rj}^t p_{ir}^m = \sum_r p_{ir}^m p_{rj}^t. \end{aligned}$$

□

The Chapman-Kolmogorov equations can be defined in a matrix form e.g. $\mathbf{P} \cdot \mathbf{P} = \mathbf{P}^2$ which is equivalent to $p_{ij}^2 = \sum_r p_{ir} p_{rj}$. In a more general way, the Chapman-Kolmogorov equations can be written in a matrix form as:

$$\mathbf{P}^{m+t} = \mathbf{P}^m \cdot \mathbf{P}^t.$$

2.3 State Probabilities

We have already considered the conditional probabilities of transition between states. In this section, we aim at considering the unconditional probability of any state at a given time n. It is important to begin by defining the probability of the initial state.

Definition 2.3.1. (Initial distribution)

For a time-homogeneous Markov chain $\{X_t\}$, the probability distribution of the initial state at time 0 is defined as:

$$\forall i \in \mathcal{X} : \pi_i^0 = P(X_0 = i),$$

where

$$\sum_{i \in \mathcal{X}} \pi_i^0 = 1.$$

The definition of the probability of the initial state is necessary in the evaluation of the unconditional probability for the state. Suppose we want to compute the following unconditional probability:

$$\begin{aligned}\pi_j^t &= P(X_t = j) \\ &= \sum_{i=1}^{N} P(X_t = j, X_0 = i) \\ &= \sum_{i=1}^{N} P(X_t = j | X_0 = i) P(X_0 = i) \\ &= \sum_{i=1}^{N} p_{ij}^t \pi_i^0,\end{aligned}$$

where $P(X_t = j|X_0 = i) = p_{ij}^t$ and $P(X_0 = i) = \pi_i^0$ then, the t-step transition matrix can be built as:

$$\mathbf{P}^t = \begin{pmatrix} p_{11}^t & \cdots & p_{1N}^t \\ p_{21}^t & \cdots & p_{2N}^t \\ \vdots & \ddots & \vdots \\ p_{N1}^t & \cdots & p_{NN}^t \end{pmatrix}.$$

Let us consider the simplest case when $t = 0$ and $t = 1$, then:

$$p_{ij}^0 = P(X_0 = j|X_0 = i) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

which means that: $\mathbf{P^0} = \mathbf{I}$. In the case of $t = 1$, we have:

$$p_{ij}^1 = P(X_1 = j|X_0 = i) = p_{ij}.$$

and hence $\mathbf{P}^1 = \mathbf{P}$.

The state probabilities in t-steps can be calculated as:

$$\pi^1 = \pi^0 \mathbf{P}$$
$$\pi^2 = \pi^1 \mathbf{P} = \pi^0 \mathbf{P}^2$$
$$\pi^3 = \pi^2 \mathbf{P} = \pi^0 \mathbf{P}^3.$$

The above equations can be generalised as:

$$\pi^t = \pi^0 \mathbf{P}^t,$$

where π^t is the row vector containing π_j^t, where $j = 1, \cdots, N$.

A DTMC can be identified by two main components: the initial distribution and the transition probability.

2.3.1 Important Properties and Classification of States in a Markov Chain

In this section, in the context of a discrete state Markov chain, the essential properties are briefly introduced. In addition, we will list the classification of Markov chains according to the transition probabilities. As a rule, a Markov chain converges to a stationary distribution. In order to reach this stationary distribution, the chain must satisfy certain conditions such as irreducibility and aperiodicity. These properties are required to ensure that a Markov chain visits any region of the state space at any unit of time. We begin with a definition of these properties.

Definition 2.3.2. (Accessibility)

Let i and j be two states in a discrete state space Markov chain. It can be said that state j is reachable or accessible from state i, denoted as $i \to j$ if:

$$\inf\{t : P(X_t = j | X_0 = i) > 0\} < \infty,$$

or, in other words, it can be expressed through the transition probability matrix as $\inf\{t : p_{ij}^t > 0\} < \infty$.

This definition can be used to introduce another concept, known as Communication, which considers the relationship between states.

Definition 2.3.3. (Communication)

It can be said that states i and j are communicating if each is reachable from the other. This can be written in the form:

$$i \leftrightarrow j \text{ iff } i \to j \text{ and } j \to i.$$

This property allows us to define the important concept of irreducibility:

Definition 2.3.4. (Irreducibility)

A Markov chain is irreducible if all states are communicating with each other in that way: $\forall i, j \in \mathcal{X} : i \leftrightarrow j$.

Definition 2.3.5. (Periodicity)

Given a state i in a DTMC, the period of state i, denoted as d_i, is defined as follows:

$d_i = gcd\{t \geq 1 : p_{ii}^t > 0\}$,

where gcd represents the greatest common divisor. The state is named periodic if $d_i > 1$, otherwise, if $d_i = 1$ it is aperiodic.

Another important concept needs to be studied to understand the behaviour of the Markov chain states, and this is based on the number of visits to a particular state if a Markov chain runs infinitely.

2.3.2 Recurrence and Transience

In a chain, there are states that will be visited several times and sometimes others infinitely. To illustrate this concept, let us define the following:

$$\eta_i(t) = \begin{cases} 1 & \text{if } X_t = i \\ 0 & \text{if } X_t \neq i. \end{cases}$$

We can define the number of visits to a state i by $V_i = \sum_{t=0}^{\infty} \eta_i(t)$. The expected number of visits (given that the chain in state i) is:

$$\mathbb{E}(V_i) = \sum_{t=0}^{\infty} \mathbb{E}(\eta_i(t)) = \sum_{t=0}^{\infty} P(X_t = i | X_0 = i) = \sum_{t=0}^{\infty} p_{ii}^t.$$

The expected number of visits to a state will be used to classify the state as recurrent or transient.

Definition 2.3.6. (Recurrence)

The state i in a DTMC is recurrent if $\sum_{t=0}^{\infty} p_{ii}^t = \infty$, otherwise, the state is called transient if $\sum_{t=0}^{\infty} p_{ii}^t < \infty$.

Let us assume that the initial passing time to return to state i is presented as:

$$T_i = \inf\{t > 1; X_t = i\}.$$

Definition 2.3.7. (Positive Recurrent)

In a DTMC, if the state i is recurrent, then, i can be considered as a positive recurrent if:

$$\mathbb{E}(T_i|X_0 = i) < \infty.$$

Definition 2.3.8. (Null Recurrent)

In a DTMC, if the state i is recurrent, then, it can be considered as a null recurrent if:

$$\mathbb{E}(T_i|X_0 = i) = \infty.$$

Definition 2.3.9. (Ergodic)

An irreducible DTMC can be considered ergodic if all states are positive recurrent and aperiodic.

A proof is given by (Gilks et al., 1995). We have considered a Markov chain and its properties. In the light of these properties, an important concept will be discussed in the following section. This is the invariant or the so called stationary distribution, which is often utilised in a Markov chain as Monte Carlo methods to build a sample targeting a particular distribution.

Definition 2.3.10. In an irreducible, ergodic DTMC, the limiting distribution exists and can be defined as follows :

$$\pi_j = \lim_{t\to\infty} p_{ij}^t, \quad \forall i \in \mathcal{X},$$

where π_j is the unique solution of:

$$\pi_j = \sum_{i=0}^{N} \pi_i p_{ij}, \quad \sum_{j=0}^{N} \pi_j = 1.$$

Definition 2.3.11. Given a DTMC with a transition matrix $\mathbf{P} = p_{ij}$, where $i, j \in \mathcal{X}$. A distribution π_j can be considered as a stationary distribution or invariant distribution of a Markov chain $(X_t, t \geq 0)$ if the following is satisfied:

$$\pi_j = \sum_{i=1}^{N} \pi_i p_{ij}, \quad \text{such that} \quad \boldsymbol{\pi} = \boldsymbol{\pi}\mathbf{P}.$$

In order to ensure that a Markov chain has a stationary distribution, a specific condition must be met. This condition is known as a detailed balance equation. This type of Markov chain also exhibits reversibility and is known as a reversible Markov chain.

2.3.3 Time Reversible Markov Chain

Let us suppose that an ergodic Markov chain exists with the transition probability p'_{ij} and the stationary distribution π_j. Let us further assume that the state is moving backwards so that a sequence of states $X_{t+1}, X_t, X_{t-1}, \ldots$ is in the reverse order, which means the distribution of X_t is conditional rather on the future than on the past. Hence, the transition probability can be written as:

$$\begin{aligned} p'_{ij} &= P(X_t = j | X_{t+1} = i) \\ &= \frac{P(X_t = j, X_{t+1} = i)}{p(X_{t+1} = i)} \\ &= \frac{P(X_{t+1} = i | X_t = j) p(X_t = j)}{p(X_{t+1} = i)} \\ &= p_{ji} \frac{\pi_j}{\pi_i}. \end{aligned}$$

Then, a time reversed Markov chain is considered as a Markov chain with:

$$p'_{ij} = p_{ji} \frac{\pi_j}{\pi_i}.$$

Definition 2.3.12. (Reversibility) A Markov chain is called reversible if it satisfies the condition:

$$\pi_i p_{ij} = \pi_j p_{ji} \quad \forall i, j \in \mathcal{X}.$$

This condition is also called the detailed balance equation, where for both states i and j, the movement from i and j occurs at the rate $\pi_i p_{ij}$. It is exactly similar to the transition rate from j and i, which is $\pi_j p_{ji}$.

More details about a DTMC and its properties, theorems, proofs and definitions can be found in (Chung, 1967), (Kemeny et al., 1960) and (Karlin and Taylor, 1981).

A discrete time Markov chain has been widely used to model different phenomena. A DTMC also plays a key role in the Metropolis-Hastings class of the sampling al-

gorithm considered in this thesis. However, most real phenomena rely on continuous time, while a DTMC is limited to discrete time. This motivates us to consider a Markov chain with continuous time.

2.4 Continuous Time Markov Chains

In this section, we provide details of other important type of a Markov chain, namely, the continuous time Markov chain (CTMC) with its essential properties. We have already described the main concepts of a DTMC when properties of a Markov chain and the classification of states have been investigated. We will now investigate a CTMC. A CTMC can be distinguished from a DTMC by the state index t, which is a real number $t \in \mathbb{R}$. In addition, in CTMC, the notations are slightly different from DTMC, the transient probabilities matrix will be $\mathbf{P}(t)$ instead of $\mathbf{P}^t$, the state probability in DTMC is assumed to be $\pi_i^t = P(X_t = i)$ while in CTMC is presented as: $\pi_i(t) = P(X(t) = i)$.

A continuous-time stochastic process $\{X(t) : t \geq 0\}$ can be considered as a Markov process when the future state relies only on the current state and is independent from all history.

Definition 2.4.1. (CTMC)

The tuple $(X, x_{init}, \mathbf{Q})$ is CTMC where:

- state space $\mathcal{X}$ is a finite set of states.
- The initial state $x_{init} \in \mathcal{X}$.
- The transition rate matrix is $\mathbf{Q} : \mathcal{X} \times \mathcal{X} \rightarrow q_{ij} \geq 0$.

In a CTMC, the transition between states is governed by the rate matrix $\mathbf{Q}$ for each pair i and j. A transition from state i to state j can occur when the matrix index is $q_{ij} > 0$.

The transition probabilities and state waiting are determined by the matrix $\mathbf{Q}$. Assume that the exit rate of state i is given as $E(i) = \sum_{j \in \mathcal{X}, j \neq i} q_{ij}$, then, the mean of the waiting time (which follows an exponential distribution) for state i is $\frac{1}{E(i)}$.

The probability of transition from state i occurring within time t is given as $1 - e^{-E(i) \cdot t}$. The probability that the transition fires from state i to state j is $\frac{q_{ij}}{E(i)}$. If

the transition rate is $q_{ij} = 0$, this implies there is no transition from i to j. This matrix is known as the generator matrix $\mathbf{Q}$ and can be defined as:

Definition 2.4.2. (Generator matrix)

Let us assume that the generator matrix associated with a CTMC is denoted by $\mathbf{Q}$, then off diagonal entries are presented as q_{ij} and the diagonal entries are $q_{ii} = -E(i)$. The generator matrix satisfies the following:

- $0 \leq -q_{ii} < \infty \quad \forall i.$
- $q_{ij} \geq 0 \quad \forall i \neq j.$
- $\sum_j q_{ij} = 0.$

A CTMC follows the same property of a DMTC. For instance, a state j is accessible or reachable i if $q_{ij} \geq 0$.

Let us assume that we have a CTMC with N states, when the transient state probability is defined:

$$\boldsymbol{\pi} = (\pi_0(t), \pi_1(t), \cdots, \pi_N(t)),$$

where the probability of a CTMC being in particular state i at unit time t is $\pi_i(t) = P(X(t) = i)$. The dynamics of a CTMC are described by:

$$\frac{d}{dt}\boldsymbol{\pi} = \boldsymbol{\pi}\mathbf{Q}.$$

The Chapman-Kolmogorov equation can be expressed in the matrix form as:

$$\frac{d}{dt}\mathbf{P}(t) = \mathbf{P}(t)\mathbf{Q}.$$

where $\mathbf{P}(t)$ represents the transition matrix that is defined in section 2.2.1.

The stationary behaviour can be obtained via solving the following system of equations:

$$\boldsymbol{\pi}\mathbf{Q} = 0, \quad \sum_{i=1}^{N} \pi_i = 1.$$

Solving this equation requires a CTMC to be irreducible and finite. In this thesis, we focus on transient probability, while reader interested in stationary behaviour is referred to (Drake, 1967), (Cox, 2017).

2.5 An Overview of Modelling Biological Systems

Stochasticity exists in most biological systems. A biological process can involve randomness due to the random collisions and interactions between different system components such as molecules inside cells. Stochastic chemical kinetics provides a description of the dynamic behaviour of such networks and account for stochasticity, the randomness of which has a significant influence on the behaviour of the model (McQuarrie, 1967), (Zheng and Ross, 1991).

A reaction network can be defined as a chemical reaction system including several species and reactions. Each reaction in the system occurs at a stochastic rate which implies that a stochastic model is needed. The reaction systems are modelled as a discrete state representing the number of molecules of species over continuous time resulting in a CTMC. A CTMC enables us to evaluate a biochemical process and its uncertainty over time by estimating the probability of the system being in a specific state at a given time. The probability that the system is at a certain state through certain time can be determined by the Chapman-Kolmogorov equations also known as the Chemical Master Equation (CME). However, the state space of a CTMC increases exponentially in terms of the number of molecules, which results in a large state space of a CTMC. Thus, solving the CME analytically or numerically turns out to be a difficult task (Munsky and Khammash, 2006).

Instead of determining the probability distribution over the various states of the system at each time by solving this equation, a sample can be drawn from their distribution. The generation of a sample trajectory is straightforward due to the proposed stochastic simulation algorithm (Doob and Doob, 1953). The advantage of using simulation of a stochastic model is about having simpler performance. Moreover, many sample paths can be simulated despite the size of the state space of a CTMC. Several exact and approximate methods have been proposed in the literature to solve the CME, we review some of these methods briefly in section 2.6.

2.5.1 Stochastic Biochemical Kinetic

A brief description of the stochastic behaviour and the function of a biological system through a biochemical reaction network using a Markov process will be given in this section (Gillespie, 1991), (Gillespie, 1996), (Wilkinson, 2011).

Consider a constant reaction volume Ω, which is assumed to be well-stirred and in thermal equilibrium. A collision between molecules can occur randomly which consequently results in a specific reaction. To illustrate these reactions, consider N different species denoted as $(X_1, \cdots, X_N)$ with their N populations which can be the number of molecules of species, denoted as $(x_1, \cdots, x_N)$.

A biochemical reaction between two distinct species X_i and X_j can be unimolecular or bimolecular reactions. This can occur if they collide while they are moving randomly and produce another species, as follows:

1. A bimolecular reaction can occur between any two distinct types of species X_i and X_j is presented as:

$$X_i + X_j \xrightarrow{c_i} Products.$$

2. A n unimolecular reaction is when a molecule of the species X_i is transferred to a molecule of another species like that:

$$X_i \xrightarrow{c_i} Products.$$

3. An unimolecular reaction: a production and degradation of a chemical species can be presented, respectively, as:

$$\emptyset \xrightarrow{c_i} Products.$$

$$X_i \xrightarrow{c_i} \emptyset,$$

where $\emptyset$ in the previous equation refers to a species which are not considered in the system. Now, a biochemical reactions system will be described in terms of a Markov process.

Each reaction occurs with a specific constant kinetic rate $c_1, \ldots, c_i$ and is associated with the hazard function, sometimes called the stochastic rate law which will be defined as follows:

Definition 2.5.1. (Hazard function)

Hazard function, denoted by $h(t)dt$, is the conditional probability that a specific type of reaction takes place in the infinitesimal interval $(t, t + dt]$, given that this reaction has not occurred at time t:

$$h(t)dt = P(t < T \leq t + dt | T > t),$$

where T is a nonnegative random variable representing the time of occurrence of the reaction (Steward, 2009).

As described in (Wilkinson, 2011), the hazard function relies only on the reactant population $X = (x_1, \cdots, x_n)$ and a constant rate c_i. In the case of unimolecular reaction or zero order reaction, the hazard function is the constant rate of the reaction:

$$h_i(x, c_i) = c_i.$$

While in a first order reaction, only one molecule x_i of species X_i is required for a reaction to take place, and, the hazard function is:

$$h_i(x, c_i) = c_i.x_i.$$

For a reaction of the second order, two distinct molecules x_d and x_j from two different species X_d and X_j are required, and the total combination of $x_j \cdot x_d$ with constant rate c_i can form the hazard function:

$$h_i(x, c_i) = c_i \cdot x_j \cdot x_d.$$

2.5.2 The Markov Description of Biochemical Reaction Network

Consider a well-mixed system of N species, with populations $(X_1, \cdots, X_N)$. This population can interact with distinct species and result in L reactions, denoted by $(R_1, \cdots, R_L)$ which can be expressed using either unimolecular or bimolecular reactions. The most widely used method to represent a biochemical network is a set of chemical reaction equations which take the form of (Wilkinson, 2011):

$$
\begin{aligned}
R_1 &: p_{11}X_1 + p_{12}X_2 + \cdots + p_{1N}X_N \xrightarrow{c_1} q_{11}X_1 + q_{12}X_2 + \cdots + q_{1N}X_N \\
R_2 &: p_{21}X_1 + p_{22}X_2 + \cdots + p_{2N}X_N \xrightarrow{c_2} q_{21}X_1 + q_{22}X_2 + \cdots + q_{2N}X_N \\
&\vdots \\
R_L &: p_{L1}X_1 + p_{L2}X_2 \cdots + p_{LN}X_N \xrightarrow{c_L} q_{L1}X_1 + q_{L2}X_2 + \cdots + q_{LN}X_N, \qquad (2.4)
\end{aligned}
$$

where the reactants p_{ij} and the products $q_{ij} \in \mathbb{N}_0$ are known as the stoichiometric coefficients.

It can be presented in a matrix form $PX \to QX$, where the entire p_{ij} and q_{ij}, the stoichiometric coefficients, form this matrix.

This matrix specifies the number of consumed and produced molecules in each species based on the occurrence of the reaction. When reaction $i, \text{where } i = 0, \cdots, L$ fires, the number of molecules of $x_j, \text{where } j = 0, \cdots, N$ can decrease by p_{ij} and increase by q_{ij}, then the overall change will be $v_{ij} = p_{ij} - q_{ij}$.

The reaction R_i is described as the stochastic vector v_{ij} that represents the molecules population after a reaction occurs. The second important quantity that characterises the system is the hazard function, where, a set of chemical reactions R_i takes place at specific hazard rate constants $c_1, \ldots, c_i$ and hazard function h_i and this process is known as chemical kinetics (Gillespie, 2007).

In a stochastic framework, a random location of the reaction is considered, and thus the number of molecules in each space is a random variable. Consequently, the hazard function h_i is also a random variable.

As we are working with a probabilistic mathematical model, let us assume that the system is characterised by a set of time dependent states vector $\boldsymbol{x} = (x_1(1), \cdots, x_N(t))^T$ which represents the molecular number of each species $(X_1, \cdots, X_N)$ at given time t. For instance, if the system is currently at state x and whenever a reaction R_i takes place, then the system state jumps to another state $x + v_i$. Since we assume that the system is well mixed meaning that the diffusion and locations of molecules are not modelled, continuous-time Markov chains (CTMC) can be used to model the dynamics of the system.

In particular, the biochemical reaction can take place randomly, resulting in the change of the molecules count and hence lead to a discrete state space $X(t)$ Markov chain with a continuous time (CTMC) (Anderson and Kurtz, 2011), (Gillespie, 1992), (Gardiner, 1986).

A CTMC is defined as the integer-valued state path describing the transition between the state if a reaction fires. The transitions of a CTMC are identified by the probabilities of occurrence of distinct reactions with the infinitesimal interval $(t, t + dt]$. Let now describe the probability of the possible transition of a CTMC according to the distinct reaction occurring such way:

1. If the system is in state $X_t = x$, the probabilities of occurrence of particular reaction R_i within a infinitesimal time interval $(t, t + dt]$ are given as:

$$P(X(t + dt) = x + v_i | X(t) = x) = h_i(x, c_i)dt + O(dt),$$

   where $h_i(x, c_i) = c_i \cdot x$ is the hazard function and the term $\frac{O(dt)}{dt}$ goes to zero as $dt \to 0$.

2. The probability that the system remains in the current state and no more reactions take place is:

$$P(X(t + dt) = x | X(t) = x) = 1 - \sum_{i=1}^{L} h_i(x, c_i)dt + O(dt).$$

3. The probability that multiple reactions will occur in an infinitesimal interval $(t, t + dt]$ is $O(dt)$.

The kinetic law of the system can be obtained through the evaluation of the system's probability. The Chemical Master Equation can provide an evaluation of the system's probability distribution (Van Kampen, 1992).

2.5.3 The Chemical Master Equation

In this section we are aim at evaluating the stochastic reaction network via computing the probabilities of states at any particular time t. Let us assume that we are interested in computing the probability of the system being in state x at a particular time t, given the fact that the system was in a state x_0 at time $t = 0$. The probability $P(X(t+dt) = x|X(0) = x_0)$ after a period of time dt can be decomposed into two parts:

$$
\begin{aligned}
&P(X(t+dt)=x|X(0)=x_0)\\
&\qquad=\sum_{i=1}^{L}\Big(h_i(x-v_i,c_i)dt+O(dt)\Big)P(X(t)=x-v_i|X(0)=x_0)\\
&\qquad\qquad+\Big[1-\sum_{i=1}^{L}\Big(h_i(x,c_i)dt+O(dt)\Big)\Big]P(X(t)=x|X(0)=x_0).
\end{aligned}
$$

The first term represents the probability that one reaction fires in the time interval $[t, t+dt]$ multiplied by the probability that the system is jumping from x to state $x-v_i$. The second term is the probability that the system remains in a state x multiplied by the probability that no reactions take place in the time interval $[t, t+dt]$.

To obtain the CME, subtracting $P(X(t)=x|X(0)=x_0)$, taking the limit $\lim_{dt\to 0}$ and dividing by dt, we get:

$$
\begin{aligned}
&\frac{d}{dt}(P(X(t)=x|X(0)=x_0))\\
&\qquad=\lim_{dt\to 0}\frac{P(X(t+dt)=x|X(0)=x_0)-P(X(t)=x|X(0)=x_0)}{dt}\\
&\qquad=\lim_{dt\to 0}\frac{\Big(\sum_{i=1}^{L}h_i(x-v_i,c_i)dt+O(dt)\Big)P(X(t)=x-v_i|X(0)=x_0)}{dt}\\
&\qquad\qquad-\frac{\Big(\sum_{i=1}^{L}h_i(x_i,c_i)dt+O(dt)\Big)P(X(t)=x|X(0)=x_0)}{dt}\\
&=\sum_{i=1}^{L}h_i(x-v_i,c_i)P(X(t)=x-v_i|X(0)=x_0)-\sum_{i=1}^{L}h_i(x,c_i)P(X(t)=x|X(0)=x_0).
\end{aligned}
$$

For simplicity, we will assume that $P(X(t)=x|X(t_0)=x_0)=P_t(x)$, then the CME can be rewritten as follows:

$$
\frac{d}{dt}P_t(x)=\sum_{i=1}^{L}h_i(x-v_i,c_i)P_t(x-v_i)-\sum_{i=1}^{L}h_i(x,c_i)P_t(x). \tag{2.5}
$$

The equation (2.5) is a coupled system of linear ordinary differential equations. In a Markov framework, the previous equation is commonly known as the Kolmogorov's forward equation that is used to evaluate the probability of a transition between states (Gillespie, 1992),(Gardiner and Zoller, 2004) and (Gardiner, 1986).

CMEs have mostly been used for modelling stochastic biological systems and generally are difficult to solve. A distribution $P_t(x)$ is known as a steady state solution of the equation 2.5 if it satisfies the condition $\frac{d}{dt}P_t(x) = 0$.

However, despite the simplicity of this system, computing CMEs is often intractable because the state space size increases once the number of molecules in each species grows. This consequently leads to a large number of ordinary differential equations. Therefore, the analytical solution of the CME is available only for some specific cases (Munsky and Khammash, 2006). Hence, different approaches have been considered in the literature to approximate the CME, and they are described briefly in the section (2.6). In addition, a stochastic simulation can be used to study the behaviour of such a biological system.

2.5.4 Stochastic Simulation

A stochastic simulation algorithm provides a way to sample exact realisation $X(t)$ of the stochastic system defined by the CME. A stochastic simulation was initially suggested by Gillespie (1976) in the chemical kinetics context, and then different variants were proposed in the literature (Mauch and Stalzer, 2011).

The algorithm simulates a stochastic reaction system as a CTMC which consists of discrete states with continuous time based on drawing an exponential waiting times for all reactions and selecting the smallest waiting time for the next reaction. A standard and widely used approach to simulate such stochastic reaction systems is the Gillespie's algorithm (Doob and Doob, 1953), (Gillespie, 2007), described below:

2.6 Related Works

The first part of this section provides a brief review of existing classes of the system that can be evaluated analytically, and in the second part of this section; some of an existing approximation methods in the literature reviewed.

2.6.1 Exact Methods

The analytic solution to the CME is known only for a restrictive class of systems and few simple special cases. In practice, it is difficult to derive an exact solution for many systems of interest. However, a stochastic simulation algorithm can be used

Algorithm 1 CTMC Simulation algorithm

1: Initialise the time $t = 0$ and the state of the system $X(t) = x_0$.
2: Calculate the hazard function for each reaction i of the system $h_i(x_t, c)$.
3: Draw a waiting time to the next j reactions from the exponential distribution such that:

$$t'_1 \sim Exp(h_1(x, c)), t'_2 \sim Exp(h_2(x, c)), \cdots, t'_i \sim Exp(h_i(x, c)).$$

4: Select the reaction j associated with minimum waiting time:

$$t' = min\{t'_i > 0 : x \neq x_t\}.$$

5: Update the state of the system to be:$x_{t+t'} = x_t + v_j$, and update current time to be: $t = t + t'$.
6: Repeat the steps $2 - 6$ while $t < T$, outputting current t and x_t.

to simulate exact samples for the stochastic process. This also forms the basis of approximate approaches. This section briefly introduces both exact and approximate methods to solve the CME as described by Schnoerr et al. (2017). The cases when the CME can be computed analytically are listed below.

1. Finite state space:

   In a state space when $\boldsymbol{x}$ is finite with N elements, let us assume that each state is associated with probability $\pi_i(t), i = 1, \cdots, N$. If we assume that the matrix of transition probabilities is $\mathbf{P}(t)$, then, the CME can be presented in a matrix form as:

$$\frac{d}{dt}\mathbf{P}(t) = \mathbf{Q}\mathbf{P}(t), \tag{2.6}$$

   where $\mathbf{Q}$ represents the generator matrix. The solution of equation (2.6) is as follows:

$$\mathbf{P}(t) = e^{(\mathbf{Q}\cdot t)}, \tag{2.7}$$

   which can be presented as the matrix exponential and $\mathbf{P}(t)$ can be evaluated as a power series:

$$\mathbf{P}(t) = e^{(\mathbf{Q}\cdot t)} = \sum_{i=0}^{\infty} \frac{(\mathbf{Q}\cdot t)^i}{i!}. \tag{2.8}$$

Hence, the solution of the CME for any system with finite state can be calculated through (2.7).

Yet, even for some small finite state space systems, the evaluation of equation (2.7) is computationally expensive due to matrix exponentiation. Several numerical approaches have been proposed to overcome this limitation (Moler and Van Loan, 1978), (Moler and Van Loan, 2003), but it still remains a difficult task in practice for many biological systems.

Uniformisation can be used to compute these probabilities. The method builds a DTMC by assuming highest exit rate, denoted as q, between all states. Hence, CTMC is transformed into a uniformised system in which the transition occurs at uniform rate (Kwiatkowska et al., 2007). This implies that all waiting times are simulated from the exponential distribution with uniform rate. If we assume that the chain M is CTMC, it will be redefined as:

Definition 2.6.1. The CTMC $M = (\mathcal{X}, x_{init}, \mathbf{Q})$ can be defined after uniformisation as $unif(M) = (\mathcal{X}, x_{init}, \mathbf{P}^*)$, where:

- Set of states and initial state remain the same.
- $\mathbf{P}^* = \mathbf{I} + \frac{\mathbf{Q}}{q}$.
- $\mathbf{I}$ is the identity matrix.
- Uniformisation rate is q and defined as: $q \geq \max\{E(x)|x \in \mathcal{X}\}$.

Using uniformisation is key to compute the transient probability as:

$$\begin{aligned}\mathbf{P}(t) = e^{(\mathbf{Q}\cdot t)} = e^{q\cdot(\mathbf{P}^*-\mathbf{I})\cdot t} = e^{(q\cdot t)\cdot\mathbf{P}^*}\cdot e^{-q\cdot t} \\ &= e^{-q\cdot t}\cdot\left(\sum_{i=0}^{\infty}\frac{(q\cdot t)^i}{i!}\cdot(\mathbf{P}^*)^i\right) \\ &= \sum_{i=0}^{\infty}\left(e^{-q\cdot t}\cdot\frac{(q\cdot t)^i}{i!}\right)\cdot\left(\mathbf{P}^*\right)^i \\ &= \sum_{i=0}^{\infty}\gamma_{q\cdot t,i}\cdot\left(\mathbf{P}^*\right)^i,\end{aligned}$$

where $(\mathbf{P}^*)^i$ contains the probabilities of transition between two states in i steps and $\gamma_{q\cdot t,i}$ represents the Poisson mass function with the parameter rate $q \cdot t$.

$$\mathbf{P}(t) = \sum_{i=0}^{\infty} \gamma_{q \cdot t, i} \cdot \left(\mathbf{P}^*\right)^i. \tag{2.9}$$

Evaluating the transient probabilities using the uniformised DTMC is numerically stable (Gross and Miller, 1984). This is because the evaluation of equation (2.8) relies on the matrix $\mathbf{Q}$ in which negative diagonal entries result in round off errors. In contrast, the uniformisation method minimises this round off error because it only works with a positive number on stochastic matrix $\mathbf{P}^*$. This method will be used later in the Repressilator case study to evaluate the transient probabilities for the system.

2. Linear systems:

   For a linear system, suppose we are not aiming to evaluate the complete distribution of the CME, but we are only interested in the first few moments, for instance, mean and variance. These can be obtained through the time evaluation of the moments of the CME.

   For example, the derivative of the first moment of the CME is the average number of species r and can be obtained by multiplying both sides of equation (2.5) by the rth component x_r, then, summing over all vector of states (molecule numbers) $\boldsymbol{x}$, where $\boldsymbol{x} = x_r, \cdots$, obtaining:

$$\sum_{\boldsymbol{x}} x_r \frac{d}{dt} P_t(x_r) = \sum_{\boldsymbol{x}} x_r \big(\sum_{i=1}^{L} h_i(x - v_i, c_i) P_t(x - v_i) - \sum_{i=1}^{L} h_i(x, c_i) P_t(x) \big). \tag{2.10}$$

   Since

$$\sum_{\boldsymbol{x}} x_r \frac{d}{dt} P_t(x_r) = \frac{d}{dt} \mathbb{E}(x_r),$$

   it can be seen that:

$$\frac{d}{dt} \mathbb{E}(x_r) = \sum_{\boldsymbol{x}} \sum_{i=1}^{L} \big(x_r h_i(x - v_i, c_i) P_t(x - v_i) - x_r h_i(x, c_i) P_t(x) \big). \tag{2.11}$$

By assuming that $x_r = x - v_i$, we obtain:

$$\begin{aligned}\frac{d}{dt}\mathbb{E}(x_r) &= \sum_{i=1}^{L}\sum_{\boldsymbol{x}}\big((x_r + v_i)h_i(x, c_i)P_t(x) - x_r h_i(x, c_i)P_t(x)\big) \\ &= \sum_{i=1}^{L}\sum_{\boldsymbol{x}}(x_r - x_r + v_i)h_i(x, c_i)P_t(x) \\ &= \sum_{i=1}^{L}\sum_{\boldsymbol{x}} v_{ri}h_i(x, c_i)P_t(x),\end{aligned} \tag{2.12}$$

where the right hand side represents the average of h_i. Then, the dynamics of the first moment can be described by the following ODE:

$$\frac{d}{dt}\mathbb{E}(x_r) = \sum_{i=1}^{L} v_{ri}\mathbb{E}(h_i(\boldsymbol{x}, c_i)),$$

where v_{ri} represents the number of rth species produced or consumed by the ith reactions (Schnoerr et al., 2017).

When the system is linear and includes only unimolecular reactions, these moment equations can be evaluated explicitly. For more details see (Schnoerr et al., 2017), (Gardiner, 1986).

However, in the case of a nonlinear system composed of bimolecular reactions, the equation for a specific moment relies on higher order moments and thus results in equations that cannot be solved analytically.

3. Non-linear systems:

   In the previous case, the analytical solution is available for linear systems that correspond to reactions that include only one product molecule. However, with nonlinear systems or linear systems that involve reactions with more than one product molecule, the analytical solution is not available. Nevertheless, it is possible to obtain a steady state solution which satisfies $\frac{d}{dt}P_t(x) = 0$ for a specific class of systems such as reversible systems that satisfy detailed balance (Schnoerr et al., 2017).

   The steady state condition in the CME implies that the probability of exiting a state corresponding to the second part of the equation (2.5) is equal to the probability of entering in a state representing the first part of the equation (2.5) which means that the process is in its stationary behaviour. The system is said to be detailed balance if for a pair of reactions i and i^* which are reversible, $i \leftrightarrows i^*$, we have:

$$h_{i^*}(x + v_i)P_t(x + v_i) = h_i(x)P_t(x),$$

which requires to be satisfied for all reactions and states. Once the detailed balance condition is satisfied, the CME can be solved. The reversible system with mass action kinetics (reaction rate is proportional to the product of concentrations of molecules) has a steady state solution consisting of the product of Poisson distributions multipled by a function that takes into account the conservation laws in molecule counts (Anderson et al., 2010). For more details see (section 3.4.3 in (Schnoerr et al., 2017)) and (Anderson et al., 2010).

4. Exact results for some special cases:

   Despite the fact that many interesting systems are not linear or the detailed balance conditions are not satisfied, there are some cases for which the CME can be solved. For more details and examples concerning these cases see (Schnoerr et al., 2017), (Gardiner, 1986).

2.6.2 Approximate Methods

As the exact solution of the CME is available only for a specific class of systems and restrictive to some cases, a recent effort has been made in the literature to introduce several approximation approaches.

The first approach focuses on approximating a stochastic process with a diffusion process, which uses the Fokker-Planck equation with a chemical Langevin equation as an approximation to the CME (Golightly and Wilkinson, 2005). The construction of the diffusion method which is described by nonlinear stochastic differential equation (SDE) corresponds to a CTMC was illustrated by Golightly and Wilkinson (2005). The method relies on the Fokker-Planck equation which is considered as an approximation to the CME. Let us assume that variables $\boldsymbol{x} = (x_1, \cdots, x_N)$ in the CME equation (2.5) are continuous, where x_i represents the molecules population of X_i. As derived in section 2.3 in (Golightly and Wilkinson, 2005) and section 4.1 in (Schnoerr et al., 2017), applying a Taylor expansion to the second order around $\boldsymbol{x}$ in the first part of the CME equation (2.5) leads to the following Fokker-Planck equation:

$$\frac{d}{dt}P_t(\boldsymbol{x}) = -\sum_{i=1}^{N}\frac{d}{dx_i}\mu_i(\boldsymbol{x})P_t(\boldsymbol{x}) + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{d^2}{dx_i dx_j}\mathbf{B}_{ij}(\boldsymbol{x})P_t(\boldsymbol{x}), \tag{2.13}$$

where $\boldsymbol{\mu}$ is the drift vector and $\mathbf{B}$ is the diffusion matrix, and both of them do not rely on time. The variables x in the CME equation (2.5) represents the discrete molecule counts while in the Fokker-Planck equation (2.13) represents continuous real numbers. Golightly and Wilkinson (2005) and Schnoerr et al. (2017) stated that the equation (2.13) is equivalent to:

$$d\boldsymbol{x} = \boldsymbol{\mu}(\boldsymbol{x})dt + \mathbf{C}(\boldsymbol{x})d\boldsymbol{W}, \quad \mathbf{C}(\boldsymbol{x})\mathbf{C}(\boldsymbol{x})^T = \mathbf{B}(\boldsymbol{x}), \tag{2.14}$$

where $d\boldsymbol{W}$ represents Brownian motion. The equation (2.14) can be interpreted as a generator of simulation of the stochastic process described by the equation (2.13). More details about this method can be found in (Golightly and Wilkinson, 2005).

The second approach aims at approximating the moment of the process and is known as a moment closure approximation. This method includes the artificial truncation of this infinite hierarchy at the specific order to get a finite set of equations. This truncation replaces all moments with high orders by the function of the moment with lower orders. This method has been considered for analysing a CTMC in (Schnoerr et al., 2014). The moment closure approximation has been used to approximate a stochastic kinetic process by Milner et al. (2013). Milner et al. (2013) show a successful application of this method on the Lotka Volterra model and a Prokaryotic autoregulatory gene network (for more details see (Milner et al., 2013)).

If the interest lies in approximating the first few moments only, it would be computationally expensive to evaluate the whole process. The system size expansion can be considered as a suitable method to approximate the moments of a process. The system size expansion method assumes that the CME equation (2.5) can be expanded about its deterministic limit (which is given by the rate equation as described by Schnoerr et al. (2017) in section 3.1).

It is based on splitting the particle number x_i in the CME equation into two parts: deterministic part ϕ_i which is the solution of the deterministic rate equation and the second part is the fluctuations about the deterministic mean which is presented by ϵ_i as follows:

$$\frac{x_i}{\Omega} = \phi_i + \Omega^{-\frac{1}{2}}\epsilon_i,$$

where Ω represents the volume of the system and ϕ_i represents the deterministic solution of the rate equation. In order to proceed the system size expansion, the CME equation will be transformed to consider these variables ϵ_i (as described by

Schnoerr et al. (2017) in section 4.1). The details of proceeding the system size expansion through different steps can be found in ((Schnoerr et al., 2017) section 4.1)

An expansion to the CME can be obtained by applying a Taylor expansion. If this expansion is truncated to zero order, it will result in a linear noise approximation where the CME can be approximated by a Fokker-Planck equation. This method can only provide an accurate approximation when the volume is large. In addition, the linear noise approximation can be considered as an appropriate approach when the molecules account of all species is large in which the stochastic effect is very small. More details can be found in (Ruttor and Opper, 2009), (Fearnhead et al., 2014),(Thomas et al., 2012), (Schnoerr et al., 2017).

The main advantage of the previous approximation method is that no prior information of the system is required, which makes the implementation much easier and provides more accurate approximations in reasonable computational time. Despite the successful application of these approximation methods in the literature (see for example the simulation study of Prokaryotic auto regulatory gene network in (Golightly and Wilkinson, 2005)), there are still many cases where those approaches show a poor performance and result in an inaccurate approximation, especially for a system with species associated with a low molecule number (Schnoerr et al., 2017). Hence, several other approximation approaches have been developed, such as state space truncation and tau-leaping. The tau-leaping approach is an approximate way (relying on the Gillespie algorithm) for the simulation of a stochastic process to be more efficient compared to the exact simulations. This method works by assuming time step τ to be as large as possible to allow more reactions to occur in this step. The accuracy of this method will be ensured by designing constraint that should be satisfied. The τ should be chosen in which the proportional change in all of the hazard is small. This means, after the leap of τ, the term $|h_i(x', c) - h_i(x, c)|$ should be very small for every i, where x is the state of the system before the leap and x' after the leap (Gillespie, 2001), (Wilkinson, 2011), (Schnoerr et al., 2017). However, these methods only applied to a specific type of system and require special tuning. A detailed explanation of both the exact and the approximation methods, their properties, a comparison of these approaches, as well as their advantages and disadvantages can be found in (Schnoerr et al., 2017).

This section focuses on approximating the marginal distributions of the process which is called the forward problem. However, these distributions rely on some parameters where the various values of these parameters exhibit different behaviour.

Therefore, it would be beneficial to take into account the uncertainty by considering the inverse problem, several approaches have been considered to address the inverse problem for Markov process, such as a variational method (Opper and Sanguinetti, 2008), particle filtering based on MCMC method (Opper and Saad, 2001), (Zechner et al., 2014) and method based on an auxiliary variable (Rao and Teh, 2013), which will be discussed in detail in the next chapter.

2.7 Summary

In this chapter, we have provided a review of a Markov chain and its main properties. We have focused on a popular class of probabilistic model CTMC. In addition, we have given a general overview of stochastic modelling, biochemical reactions and the hazard function.

We have shown that the time evaluation of the probability distribution of a biological system can be obtained by solving the CME, which is analytically challenging for many systems of interest. This is due to the fact that the model can involve many different species which results in a high dimensional state space. Hence, this is resulting in a system of many ordinary differential equations that cannot be evaluated explicitly, and many efforts have been made by researchers to solve this problem. We briefly reviewed some of these existing exact and approximation approaches of the CME. A stochastic simulation which has been introduced to overcome this problem is investigated too.

We have considered all essential concepts that are required to construct a CTMC model of the interesting stochastic system. Before we start applying this modelling approach to our case study, we first consider the uncertainties related to this model and how it can be inferred within a Bayesian framework in the next chapter.

Chapter 3

Bayesian Inference Methods

This chapter provides an introduction to the Bayesian Inference method. The main goals are to review essential concepts of probability theory and to define the notation that we will be using. Modelling a stochastic system using a probabilistic model can provide a description of the system's behaviour. This model, however, is associated with some uncertainty which requires inference. The Bayesian inference framework quantifies the uncertainty involved in the model as a random variable and each parameter of the model is then supposed to be distributed according to a certain density. Bayesian inference thus makes use of the probability concept by assigning a prior probability to those random variables. The inference about unknown quantity is based on the prior knowledge of the model. The belief, however, can be updated if a new observation is made or further information becomes available. This can be done through the Bayes' theorem, which was devised by Thomas Bayes (Bayes et al., 1763):

$$\pi(A|B) = \frac{\pi(B|A)\pi(A)}{\pi(B)}, \tag{3.1}$$

where $\pi(A)$ is the probability that represents the prior knowledge about the event A. The belief concerning the event A can be updated; taking into account the occurrence of event B, this is denoted by the conditional probability $\pi(A|B)$. Within a probabilistic model, A can be presented as a set of quantities of interest. This prior belief $\pi(A)$ can be updated to the posterior belief $\pi(A|B)$ in the light of the new observation of B (Bernardo and Smith, 2001),(Smets, 2008),(Armitage et al., 2008). When inference is performed on a probabilistic model, the aim is to infer an unknown parameter associated with this model. In a probabilistic modelling setting, it can be useful to replace a general event A by an unknown model parameter θ and

the event B by y, then, the inference about the model parameter is given by the posterior,

$$\pi(\theta|y) = \frac{\pi(y|\theta)\pi(\theta)}{Z}, \tag{3.2}$$

where

$$Z = \pi(y) = \int_{\Theta} \pi(y|\theta)\pi(\theta)d\theta$$

is the normalisation constant and also can be interpreted as the prior predictive distribution.

The posterior predictive distribution for future observations $\tilde{y}$ is $\pi(\tilde{y}|y)$ which can be evaluated using:

$$\pi(\tilde{y}|y) = \int_{\Theta} \pi(\tilde{y}|\theta, y)\pi(\theta|y)d\theta.$$

The normalisation constant Z of the posterior is a function of y only, and it can be dropped but then the relation will change from equality to proportion, which is called un-normalised posterior density:

$$\pi(\theta|y) \propto \pi(y|\theta)\pi(\theta). \tag{3.3}$$

The main issue to be faced in Bayesian inference, especially when using a complex model, arises from the dimension of parameter θ, which results in high dimensional and complicated distribution that cannot be evaluated analytically.

This is the motivation research to consider alternative approximation methods based on statistical simulation to carry out the inference.

This chapter focuses only on the important common methods of parameter inference, which are based on statistical simulation. In particular, we start with standard Monte Carlo approaches. This refers to a general stochastic way that uses a collection of sampled random variables to approximate a certain complicated integral (Gilks et al., 1995). As the direct sampling from the target distribution is often not possible; we resort to another approach that relies on drawing a sample from a different distribution, known as a proposal density. This method is called rejection sampling (RJ), which is described in section 3.1.1 (MacKay, 2003). The main limitation of this method is that, if the proposal density is not similar to the target,

this will affect the efficiency of the estimation. An alternative method that can be employed to obtain a more efficient approximation is importance sampling (IS) which was extended in a sequential setting later. Gordon et al. (1993) and Stewart (1991) introduced a Sequential form of Monte Carlo (SMC), often called a particle filter and it was developed by Andrieu and Doucet (2003) and Del Moral (1997). A complete review of the SMC method can be found in (Doucet and Johansen, 2009).

The main goal is to provide the reader with a general description of a different methodology that allows parameter inference to be performed with a complicated target distribution. As we are working with a stochastic model, the inference approach is reinvestigated in the light of a state space model with emphasis on the Markovian model. Section 3.2.4 defines the state space model and outlines an appropriate method that can be performed to carry out the inference.

Section 3.3 provides details of the Monte Carlo approach that makes use of a Markov chain to generate a sample from the posterior and hence approximate the intractable distribution, namely, Markov chain Monte Carlo (MCMC). The MCMC becomes an important tool to approximate the target distribution for various statistical problems. We focus on the most powerful class of MCMC: the Metropolis-Hastings algorithm which was first introduced in statistical physics (Metropolis et al., 1953),(Hastings, 1970). The main concepts and efficient design of the algorithm are illustrated in section 3.4.

Two powerful approaches: sequential Monte Carlo (SMC) and MCMC are combined to form a new Particle MCMC method which was proposed by Beaumont (2003) and analysed by Andrieu et al. (2010). Finally, we provide a discussion and comments about these approaches.

3.1 Monte Carlo Methods

The main aim of Monte Carlo is to overcome the numerical problem of the intractability of the target distribution in Bayesian inference. Monte Carlo approaches can be used instead of evaluating intractable distribution by drawing samples from them (Metropolis and Ulam, 1949),(Robert, 2004). Let us assume that the interest is to evaluate the complicated target distribution $\pi(x)$. This distribution cannot be evaluated directly but, instead, it can be estimated according to a classical Monte Carlo approach.

The main idea of Monte Carlo is that any complicated distribution can be approxi-

mated by sampling identical independently distributed random samples $x_1, \cdots, x_N$ from the target $\pi(x)$ where in a Bayesian setting, the probability density $\pi(x)$ corresponds to the posterior density $\pi(\theta|y)$.

Monte Carlo provides an empirical approximation to $\pi(x)$ as follows:

$$\hat{\pi}^N_{MC}(x) \approx \frac{1}{N}\sum_{i=1}^{N}\delta_{x^i}(x), \tag{3.4}$$

where $\{x^i\}_{i=1}^N$ is the collection of iid samples (particles) generated from the target $x^i \sim \pi(x)$, and δ_{x^i} represents the Dirac function, with a unit mass at x^i. For further information see Robert (2004).

However, the direct sample from the distribution of interest is not possible in many inference problems. Therefore, a different sampling technique has been developed to generate a sample from the target distribution within the Monte Carlo framework. One of these methods is rejection sampling or, as it is sometimes called, Accept-Reject sampling.

3.1.1 Rejection Sampling

One of the basic Monte Carlo sampling methods is the rejection sampling technique, which aims to draw a sample from an alternative distribution, as the direct sampling from the target distribution $\pi(x)$ is difficult because of the complexity of distribution. We, therefore, make use of another known similar distribution, denoted by $q(x)$, the proposal distribution $q(x)$ should satisfy the following condition:

$$\pi'(x) < Mq(x), \quad \forall x \quad \text{under the support of } \pi'(x), \tag{3.5}$$

where $\pi'(x)$ is non-normalised distribution of the target:

$$\pi(x) = \frac{\pi'(x)}{Z} \propto \pi'(x),$$

and the constant M is determined to bound the ratio $\frac{\pi'(x)}{q(x)}$.

The rejection sampling algorithm works by drawing two random samples: x from the proposal density $q(x)$, which corresponds to taking a random location x and sampling u from the uniform distribution which represents a random y location. If $u < \frac{\pi'(x)}{Mq(x)}$, we accept the proposed sample; otherwise we reject it.

Algorithm 2 Acceptance-Rejection Algorithm

1: Sample $x \sim q(x)$
2: Sample $u \sim U(0,1)$
3: **If** $u < \frac{\pi'(x)}{Mq(x)}$
4: Accept x
5: **else**
6: Reject x

To prove that this algorithm yields samples from the distribution of interest is as follows:

$$S = \{(x,u) : u \leq \frac{\pi'(x)}{Mq(x)}\}, \quad S_0 = \{(x,u) : x \leq x_0, u < \frac{\pi'(x)}{Mq(x)}\}.$$

To confirm that samples from this algorithm follow the target distribution $\pi'(x)$, we calculate the cumulative distribution function of the accepted samples as follows:

$$\begin{aligned} P(x \leq x_0 | x \text{ is accepted}) &= \frac{P(x \leq x_0,\ x \text{ is accepted })}{P(x \text{ is accept})} \\ &= \frac{\int\int \mathbb{1}((x,u) \in S_0) q(x) du dx}{\int\int \mathbb{1}((x,u) \in S) q(x) du dx} \\ &= \frac{\int_{-\infty}^{x_0} \{(\pi'(x)/Mq(x)) q(x) du\} dx}{\int_{-\infty}^{\infty} \{(\pi'(x)/Mq(x)) q(x) du\} dx} \\ &= \frac{\int_{-\infty}^{x_0} \pi'(x) dx}{\int_{-\infty}^{\infty} \pi'(x) dx}. \end{aligned}$$

This confirms that the distribution of the sampled values is the distribution corresponding to the cumulative distribution function of the target $\pi'(x)$ (Murphy, 2012).

The probability of acceptance is:

$$\pi(\text{accept}) = \int \frac{\pi'(x)}{Mq(x)} q(x) dx = \frac{1}{M} \int \pi'(x) dx,$$

In practice, M should be chosen as small as possible subject to the condition defined by the inequality 3.5. The algorithm becomes inefficient when M is large, as most of the proposed values will be rejected. For more details about the rejection sampling methods, see (MacKay, 2003) and (Robert, 2004). As a consequence of the shortcomings of the rejection scheme, other techniques have been developed to solve such inference problems.

3.1.2 Importance Sampling

The main limitation of the rejection sampling method is that it can be inefficient in cases of generating samples from complex and multidimensional distributions. Thus, the importance sampling method will be considered as a means of overcoming the efficiency issues associated with the rejection algorithm. The main idea of the importance sampling method is to make use of the arbitrary density $q(x)$ to generate an *iid* sample x^i to approximate the target $\pi(x)$ and then make use of a weight to correct the discrepancy between the target and proposal densities. The proposal density or importance function is often chosen to be simple, which makes the sampling easy. In addition, the support of the importance density should contain the support of the target $\pi(x)$. The discrepancy between the distributions $\pi(x)$ and $q(x)$ is measured by the importance weight, as denoted $w(x)$. These samples are used to estimate the target distribution $\pi(x)$ by:

$$\hat{\pi}_{IS}^{N}(x) = \frac{1}{N}\sum_{i=1}^{N}\frac{\pi(x^i)}{q(x^i)}\delta_{x^i}(x), \tag{3.6}$$

where $\frac{\pi(x)}{q(x)}$ is the importance weight $w(x)$, used as a correction step for the fact that the drawn are samples from the proposal distribution $q(x)$ not from the target distribution. That is,

$$\hat{\pi}_{IS}^{N}(x) = \frac{1}{N}\sum_{i=1}^{N} w(x^i)\delta_{x^i}(x). \tag{3.7}$$

For simplicity, we refer to $w(x^i)$ as $w^{(i)}$.

Unfortunately, in most cases the target distribution cannot be evaluated due to the unknown normalisation constant Z (Marshall, 1954), (Robert, 2004). Hence, an importance sampling scheme can be define alternatively in light of an unnormalised target $\pi(x) \propto \pi'(x)$, the estimator can be defined as follows:

$$\hat{\pi}_{SNIS}^{N}(x) = \frac{1}{N}\sum_{i=1}^{N}\frac{w'(x^i)}{\sum_{j=1}^{N} w'(x^j)}\delta_{x^i}(x) \tag{3.8}$$

$$\hat{\pi}_{SNIS}^{N}(x) = \frac{1}{N}\sum_{i=1}^{N} w^{(i)}\delta_{x^i}(x).$$

This estimator is known as the self normalised importance sampling (SNIS).

Algorithm 3 Self Normalised Importance Sampling (SNIS) Algorithm

1: Sample $x^i \sim q(x)$ where $i = 0, \cdots, N$
2: Compute the weight $w'(x^i) = \frac{\pi'(x^i)}{q(x^i)}$
3: Compute the normalised weight $w^{(i)} = \frac{w'(x^i)}{\sum_{j=1}^{N} w'(x^j)}$.

The main advantage of the importance sampling scheme is that it allows using samples from another arbitrary density instead of having a complex target distribution which can only be known up to the normalisation constant. A suitable selection of the proposal is such that the proposal density mimics the target density so such that if $\pi(x) > 0$, then $q(x) > 0$ and the support of the proposal $q(x)$ should include the support of $\pi(x)$ such that: support$(\pi) \subset$ support(q). In addition, care should be taken when the proposal function has heavy tails because most of the proposed samples can lie in the low posterior probability region with a small weight. In contrast, if the proposal density is lightly tailed in the area of high posterior probability, the weight of the proposal sample from the tail of the proposal density will be large which can result in poor estimation of the target distribution.

In general, the importance sampling method can provide consistent and efficient estimation, compared to the classical Monte Carlo method and the rejection sampling method (Murphy, 2012). This is due to the fact that the samples only focus on the important regions of space which implies that less number of samples are needed compared to sample from the exact distribution $\pi(x)$.

Yet, in some cases, the algorithm can be inefficient due to the high difference between importance and target densities which result in high variability of the weights. In addition, in case of high dimensional and complex target distribution $\pi(x)$, employing importance sampling can be challenging as the choice of proposal density require some knowledge about the target distribution which is not available. More details and an explanation about importance sampling can be found in (Glynn and Iglehart, 1989), (Hastings, 1970) and (Robert and Casella, 2010).

3.2 Sequential Monte Carlo Methods

The importance sampling method was improved by sampling from a sequence of intermediate distributions to approximate the target distribution. Such a method is known as the Sequential Monte Carlo Method (SMC). The main idea behind SMC is to sample sequentially from a sequence of distributions $\{\pi_m(x_{0:m})\}_{m=0}^{M}$, where,

m represents the discrete stages. The target distribution within the sequence of distributions can be denoted by $\pi_m(x_{0:m}) = \frac{\pi'_m(x_{0:m})}{Z_m}$, where $\pi'_m(x_{0:m})$ represents the unnormalised target distribution. The normalisation constant of the target π_m is represented by Z_m.

The more efficient way is to construct the proposal density sequentially as suggested in (Del Moral et al., 2006) as follows:

$$q_m(x_{0:m}) = q_{m-1}(x_{0:m-1})q_m(x_m|x_{m-1}) = q_0(x_0)\prod_{s=1}^{m} q_s(x_s|x_{0:s-1}). \tag{3.9}$$

The particles $x^i_{0:m} \sim q_m(x_{0:m})$ can be obtained at time m by sampling $x^i_0 \sim q_0(x_0)$ firstly at the initial time $m = 0$. Secondly, at time s, where $s = 1, \cdots, m$, the x^i_s can be sampled from the proposal such that $x^i_s \sim q_s(x_s|x^i_{0:s-1})$. The associated weight can be evaluated sequentially by:

$$w'_m(x_{0:m}) = \frac{\pi'_m(x_{0:m})}{q_m(x_{0:m})} = \frac{\pi'_{m-1}(x_{0:m-1})}{q_{m-1}(x_{0:m-1})} \quad \frac{\pi'_m(x_{0:m})}{\pi'_{m-1}(x_{0:m-1})q_m(x_m|x_{0:m-1})}. \tag{3.10}$$

The method result from the previous sequential setup is commonly known as sequential importance sampling (SIS) which is considered as a particular case of SMC (Doucet and Johansen, 2009). This scheme can be considered as an extension of the standard importance sampling approach. Details of sequential importance sampling are described in the following section.

3.2.1 Sequential Importance Sampling

Suppose that at stage $m-1$, the obtained weighted samples $\{x^i_{0:m-1}, w^i_{0:m-1}\}_{i=1}^N$ are used to approximate the target π_{m-1} by an empirical approximation as:

$$\hat{\pi}^N_{m-1.SIS}(x_{0:m-1}) \approx \sum_{i=1}^{N} \frac{\pi'(x^i_{0:m-1})}{q(x^i_{0:m-1})}\delta_{x^i_{0:m-1}}(x_{0:m-1}) \approx \sum_{i=1}^{N} w_m(x^i_{0:m-1})\delta_{x^i_{0:m-1}}(x_{0:m-1}). \tag{3.11}$$

These samples then propagate to the following distribution $\pi_m(x)$ by making use of the proposal distribution q_m in order to have a collection of samples $\{x^i_{0:m}\}$. Then, the unnormalised weight is computed as:

$$w'_m(x^i_{0:m}) = w^i_{m-1}\frac{\pi'_m(x^i_m)}{q_m(x^i_m)}.$$

Algorithm 4 Sequential Importance Sampling Algorithm

1: For $i = 1, \cdots, N$, Initialise sample

$$x_0^i \sim q_0(x_0)$$

Assign initial weight:

$$w_0^{'}(x_0^i) = \frac{\pi'(x_0^i)}{q_0(x_0^i)}$$

$$w_0^i = \frac{w_0^{'}(x_0^i)}{\sum_{j=1}^{N} w_0^{'}(x_0^j)}$$

2: At the next time $m = 1, \cdots, M$ and for $i = 1, \cdots, N$ propagate:

$$x_m^i \sim q_m(x_m | x_{m-1}^i)$$

3: Compute the importance weight:

$$w_m^{'}(x_{0:m}^i) = w_{m-1}^i \frac{\pi_m'(x_{0:m})}{\pi_{m-1}'(x_{0:m-1}) q_m(x_m | x_{0:m-1})}.$$

4: Compute the normalise weight:

$$w_m^i = \frac{w_m'(x_{0:m}^i)}{\sum_{j=1}^{N} w_m^{'}(x_{0:m}^i)}$$

The obtained weighted samples $\{x_{0:m-1}^i, w_{0:m-1}^i\}_{i=1}^N$ at the final stage m then we used to estimate the target distribution π_m as follows:

$$\hat{\pi}_{m.SIS}^N(x_{0:m}) \approx \sum_{i=1}^{N} w_m(x_{0:m}^i) \delta_{x_{0:m}^i}(x_{0:m}). \tag{3.12}$$

The procedure of SIS is outlined in algorithm 4.

The main problem with SIS is that as the number of stages m increases, the discrepancy between the target and the proposal distribution will increase too which result in particle degeneracy.

3.2.2 Particle Degeneracy

The main idea of SIS is that the samples (particles) are placed into important regions in the space where there is a high mass of the target distribution. The best case is when the importance distribution is proportional to the target distribution where all

samples have a similar weight. Thus, the variance of the weighted sample can reflect the quality of the estimation. However, the major problem with the SIS algorithm is that after a few steps only a few particles are associated with high weight while the other particles have a negligible weight (Doucet et al., 2000), (Cappé et al., 2007). As a consequence, the variance of the weights highly increases as the number of stages m increases, and this produces what is known as particle degeneracy (Kong et al., 1994; Liu and Chen, 1998). The degeneracy of particles means that only a few samples have a relatively high importance weight and the level of degeneracy can be quantified through the effective sample size (ESS) through iterations, which is defined as:

$$ESS = \frac{1}{\sum_i^N (w_m^i)^2}$$

To mitigate the issue of sample degeneracy an additional resampling step was introduced to minimise the variance between weights. The accompaniment of SIS with resampling results in a method known as sequential importance resampling (SIR) (Rubin, 1987), (Rubin, 1988). The procedure of SIR essentially depends on multiplying the particles with high weight and abandoning the particles with low weight. Carrying out this procedure at every step can result in sample impoverishment, which means that only a few single particles carry high weight while the particles with small weights are discarded during the resampling step. To overcome this, a resampling step is applied only at some stages, when ESS drops below a predefined threshold.

3.2.3 Sequential Importance Resampling

Sequential importance resampling was investigated by Gordon et al. (1993) and introduced to overcome the degeneracy problem described in 3.2.2. The algorithm has the following three main stages: resampling, propagation and weighting. The basic idea behind the resampling stage is that sampled particles with high weights are replicated while particles with small weights are discarded. Therefore, only a particle with significant weight will be used. There are different strategies for resampling particles, such as multinomial sampling, residual resampling and stratified sampling. More details are provided by Hol et al. (2006), Carpenter et al. (1999) and Kitagawa and Sato (2001). In this thesis, multinomial sampling is considered as the simplest resampling scheme, which is based on sampling with replacement from the currently weighted particle set (Gordon et al., 1993).

The process works by sampling a parent particle at the previous step $m-1$ where the current particle i originates from the particle denoted by a_m^i, often known as the ancestor index. This process can be performed through multinomial resampling, which is sampling from a multinomial distribution and the probability of sampling is obtained by:

$$\pi(a_m^i = j) = w_{m-1}^{(j)} \quad j = 1, \cdots, N.$$

However, performing resampling at each stage can result in losing sample diversity of the particles, which is known as impoverishment (Carpenter et al., 1999). This implies that it is not desirable to resample at each stage. So, it is preferable to define a specific threshold on the ESS, and when the ESS is less than a defined threshold, resampling should be carried out. In order to obtain a particle at stage m, a simulation carried out from a stage $m-1$ using a proposal distribution to propagate each particle as follows:

$$x_m^i \sim q_m(x_m | x_{m-1}^{a_m^i}),$$

and the complete particle history $x_{0:m}^i$ can be found as:

$$x_{0:m}^i = \{x_{0:m-1}^{a_m^i}, x_m^i\}.$$

The importance weight is computed as described in equation (3.9). The summary of the algorithm is given in algorithm 4.

The previous sections aim to provide a summary of the methods that are used to approximate the intractable distribution in general. In Bayesian inference, the posterior distribution we aim to infer can be more difficult due to the intractability in the likelihood term where the evaluation of the posterior requires an explicit and known form of the likelihood. Thus, the likelihood is required to carry out the Bayesian inference. However, in a Markov chain, the likelihood can be doubly intractable as it contains a latent variable, such as the state of the model. Therefore, a Monte Carlo approximation technique will be reconsidered for the approximation of the likelihood but with an emphasis specifically on the Markov chain Model.

Algorithm 5 Sequential Importance Sampling Resampling

1: For $i = 1, \cdots, N$, Initialise sample

$$x_0^i \sim q_0(x_0)$$

Assign initial weight:

$$w_0^{'i} = \frac{\pi(x_0^i)}{q_0(x_0^i)}$$

$$w_0^i = \frac{w_0^{'i}}{\sum_{j=1}^N w_0^{'j}}$$

2: At the next time $m = 1, \cdots, M$ and for $i = 1, \cdots, N$

3: Sample the index a_m^i from $j = 1, \cdots, N$ with probabilities w_{m-1}^j and set $x_{m-1}^i = x_{m-1}^{a_m^i}$

4: Propagate:

$$x_m^i \sim q_m(x_m | x_{m-1}^{a_m^i}).$$

5: Compute the importance weight:

$$w_m^{'}(x_{0:m}^i) = w_{m-1}^i \frac{\pi_m'(x_{0:m})}{\pi_{m-1}'(x_{0:m-1}) q_m(x_m | x_{0:m-1})}.$$

6: Normalise the weight.

3.2.4 Bayesian Inference with a State Space Model

We have introduced important inference methods that aim to estimate the target distribution in general. In this section, these approaches will be reconsidered in the light of a state space model e.g. the Markov Chain Model. Let us first describe the state space model, involving the Markov process and other relevant notations. Let us assume that we are aiming to model a system using a homogeneous continuous-time Markov model, which is characterised by a set of states $\{x_m\}_{m>0}$ where $x_m \in \chi$. Let us further assume that the model is parametrised by $\theta \in \Theta$, which is unknown and our interest is to make an inference about the model parameter, given the available data. The Markov process, as we described in chapter 2, is associated with the transition kernel $f_\theta(x_m|x_{m-1})$ which represents the current state, given only the previous one. The observation $\{y_m\}_{m>0}$ of data is supposed to be conditionally independent given $\{x_m\}_{m>0}$ and governed by the distribution $\pi_\theta(y_m|x_m)$. The probabilistic description of the model can be defined as:

$$x_0 \sim \pi_\theta(x_0), \tag{3.13}$$

$$x_m|x_{m-1} \sim f_\theta(x_m|x_{m-1}), \tag{3.14}$$

$$y_m|x_m \sim \pi_\theta(y_m|x_m). \tag{3.15}$$

The term $\sim$ implies distribution according to, $\pi_\theta(x)$ denotes a probability density function and $f_\theta(x_m|x_{m-1})$ is the probability density represent the transient from state x_{m-1} to state x_m. The density of the observation given the state is $\pi_\theta(y_m|x_m)$ (Doucet and Johansen, 2009).

Within modelling context and in the Bayesian framework, equation (3.13) and equation (3.14) represent the prior distribution of the process $\{x_m\}_{m>0}$. The equation (3.15) represents the likelihood, then; we can define the following:

$$\pi_\theta(x_{0:m}) = \pi_\theta(x_0) \prod_{s=1}^{m} f_\theta(x_s|x_{s-1}) \tag{3.16}$$

$$\pi_\theta(y_{0:m}|x_{0:m}) = \prod_{s=1}^{m} \pi_\theta(y_s|x_s). \tag{3.17}$$

In a Bayesian framework, the inference about $x_{0:m}$ given the observation $y_{0:m}$ can be obtained through the posterior distribution as follows:

$$\pi_\theta(x_{0:m}|y_{0:m}) = \frac{\pi_\theta(x_{0:m}, y_{0:m})}{\pi_\theta(y_{0:m})}, \tag{3.18}$$

where the numerator is:

$$\pi_\theta(x_{0:m}, y_{0:m}) = \pi_\theta(y_{0:m}|x_{0:m})\pi_\theta(x_{0:m}), \tag{3.19}$$

and the denominator is:

$$\pi(y_{0:m}) = \int \pi_\theta(x_{0:m}, y_{0:m})dx_{0:m} = \int \pi_\theta(y_{0:m}|x_{0:m})\pi_\theta(x_{0:m})dx_{0:m}. \tag{3.20}$$

The main interest is to approximate the marginal likelihood $\pi(y_{0:m})$.

Recall the likelihood defined in the equation (3.17) and the prior which is defined in (3.16). The unnormalised posterior distribution $\pi(x_{0:m}, y_{0:m})$ defined in equation (3.18) satisfies:

$$\pi_\theta(x_{0:m}, y_{0:m}) = \pi_\theta(x_{0:m-1}, y_{0:m-1})f_\theta(x_m|x_{m-1})\pi_\theta(y_m|x_m). \tag{3.21}$$

Hence, the posterior can satisfy the recursion as a follow:

$$\pi_\theta(x_{0:m}|y_{0:m}) = \pi_\theta(x_{0:m-1}|y_{0:m-1})\frac{f_\theta(x_m|x_{m-1})\pi_\theta(y_m|x_m)}{\pi_\theta(y_m|y_{0:m-1})}. \tag{3.22}$$

where

$$\pi_\theta(y_m|y_{0:m-1}) = \int f_\theta(x_m|x_{m-1})\pi_\theta(y_m|x_m)\pi_\theta(x_{m-1}|y_{0:m-1})dx_m dx_{m-1}, \tag{3.23}$$

where $\pi_\theta(x_{m-1}|y_{0:m-1})$ represents unknown filtering distribution of the current state of the model given the available information (Doucet and Johansen, 2009; Del Moral et al., 2013). The marginal distribution $\pi_\theta(x_m|y_{0:m})$ can be obtained by integrating $x_{0:m-1}$ out in equation (3.22):

$$\pi_\theta(x_m|y_{0:m}) = \frac{\pi_\theta(y_m|x_m)\pi_\theta(x_m|y_{0:m-1})}{\pi_\theta(y_m|y_{0:m-1})}, \tag{3.24}$$

the previous equation is called the updating step, where

$$\pi_\theta(x_m|y_{0:m-1}) = \int f_\theta(x_m|x_{m-1})\pi_\theta(x_{m-1}|y_{0:m-1})dx_{m-1}. \tag{3.25}$$

The equation (3.25) is called the prediction step. If $\{\pi_\theta(x_{0:m}|y_{0:m})\}$ and $\{\pi_\theta(x_m|y_{0:m})\}$ can be computed sequentially, then the marginal likelihood $\pi(y_{0:m})$ can be evaluated using:

$$\pi(y_{0:m}) = \pi(y_0)\prod_{s=1}^{m}\pi_\theta(y_s|y_{0:s-1}),$$

where the term $\pi_\theta(y_s|y_{0:s-1})$ is defined in equation (3.23) (Doucet and Johansen, 2009).

Evaluating such distributions analytically can be difficult. Alternatively, they can be estimated through a Monte Carlo sampling technique such as importance sampling. For more details about the filtering sampling algorithm see (Doucet et al., 2000), (West, 1996) and (Gordon et al., 1993).

In SIS, the desired distribution can be approximated by N particles. Then, IS is employed to propagate these particles through the stags using importance density. At initial stage $m = 0$, the importance density assumed to be $q_0(x_0)$, hence; the importance weight can be defined as:

$$w_0^{'i} = \frac{\pi_\theta(x_0^i)\pi_\theta(y_0|x_0^i)}{q(x_0^i)}.$$

At stage $m > 1$, the importance weight is computed as:

$$w_m^{'i} = \frac{f_\theta(x_m^i|x_{m-1}^i)\pi_\theta(y_m|x_m^i)}{q(x_m^i|x_{m-1}^i)}.$$

Then, the weight can be normalised as:

$$w_m^i = \frac{w_m^{'i}}{\sum_j^N w_m^{'j}}.$$

The set of $\{x_m^i\}$ and the corresponding weights $\{w_m^i\}$ represent the SIS for the state space model, and is called a particle filter.

The likelihood in equation (3.23) can then be approximated as:

Algorithm 6 Sequential Importance Sampling for state space model

1: For $i = 1, \cdots, N$, initialise sample

$$x_0^i \sim q(x_0)$$

Assign initial weight:

$$w_0^{i'} = \frac{\pi_\theta(x_0^i)\pi_\theta(y_0|x_0^i)}{q(x_0^i)}$$

$$w_0^i = \frac{w_0^{i'}}{\sum_{j=1}^N w_0^{j'}}$$

2: At the next stage $m = 1, \cdots, M$ and for $i = 1, \cdots, N$ propagate:

$$x_m^i \sim q(x_m|x_{m-1}^i)$$

3: Compute the importance weight:

$$w_m^{i'} = w_{m-1}^i \frac{f_\theta(x_m^i|x_{m-1}^i)\pi_\theta(y_m|x_m^i)}{q(x_m^i|x_{m-1}^i)}$$

4: Compute the normalised weight:

$$w_m^i = \frac{w'_m(x_{0:m}^i)}{\sum_{j=1}^N w'_m(x_{0:m}^i)}.$$

$$\hat{\pi}_\theta(y_m|y_{0:m-1}) \approx \frac{1}{N}\sum_{i=1}^{N} w_m^i.$$

The procedure of this approach is described in algorithm 6. Different types of proposal density can be chosen. One possible option is to choose the state transition, this results in a particular case of the particle filter known as a bootstrap particle filter.

3.2.5 Bootstrap Particle Filter

The bootstrap particle filter, suggested by Gordon et al. (1993), is a recursive approach that enables us to perform Bayesian inference for a model. This method considers the transition distribution of the Markov model as the importance distribution:

$$q(x_m|x_{m-1}) = f_\theta(x_m|x_{m-1}). \tag{3.26}$$

Then, the weight can be defined as:

$$w_m^{i'} = w_{m-1}^i \frac{f_\theta(x_m^i|x_{m-1}^i)\pi_\theta(y_m|x_m^i)}{f_\theta(x_m^i|x_{m-1}^i)}.$$

By setting the importance in (3.26), the weight is simplified to be:

$$w_m^{'i} = \pi_\theta(y_m|x_m^i)w_{m-1}^i.$$

A resampling step based on a multinomial sample of the particles can be employed in both Sequential Importance Sampling for a state space model and Bootstrap particle filter when it is required.

3.3 Markov Chain Monte Carlo

The Markov Chain Monte Carlo method (MCMC) is widely used for drawing samples from complicated probability distributions of interest. The idea of this algorithm is based on constructing a specific type of Markov chain that represents a distribution of interest as its stationary distribution. The stationary distribution is the essential property that should be satisfied as we are aiming at drawing a sample from a target

distribution by making use of a Markov chain with a certain stationary distribution. A distribution π is considered to be stationary to the Markov chain if:

$$\pi(\theta') = \int_{\Theta} \pi(\theta) T(\theta'|\theta) d\theta, \tag{3.27}$$

where we assume that $T(d\theta'|\theta)$ is the Markov transition kernel $T : \Theta \times \Theta \to [0, 1]$ that admits a density $T(\theta'|\theta)$ which assigns a probability density to any state θ' given the present state θ.

If a Markov chain is ergodic (aperiodic, irreducible and positive recurrence), then it can be said that the Markov chain converges to a unique limiting distribution:

$$\pi(\theta') = \lim_{n \to \infty} T^n(\theta'|\theta),$$

which means that the distribution over states will converge to $\pi(\theta')$ when the number of stages n tends to infinity. If the limiting distribution exists, it can be considered as the stationary distribution and no other stationary distribution exists.

Another sufficient condition that should be satisfied to have a stationary distribution which is considered as the target distribution is known as detailed balance:

$$\pi(\theta) T(\theta'|\theta) = \pi(\theta') T(\theta|\theta') \quad \forall \theta', \theta \in \Theta, \tag{3.28}$$

which is also known as reversibility. To confirm the stationary property (equation 3.27), we integrate both side of equation 3.28, such that:

$$\begin{aligned} \int_{\Theta} \pi(\theta) T(\theta'|\theta) d\theta &= \int_{\Theta} \pi(\theta') T(\theta|\theta') d\theta \\ &= \pi(\theta') \int_{\Theta} T(\theta|\theta') d\theta \\ &= \pi(\theta'). \end{aligned} \tag{3.29}$$

The property $\int_{\Theta} T(\theta|\theta') d\theta = 1$ is used in the third step. The stationary property is recovered. More details of Markov chain concepts, their essential properties and examples are given by Robert (2004).

3.4 Metropolis-Hastings

The Metropolis-Hastings approach was originally introduced in statistical physics (Metropolis et al., 1953),(Hastings, 1970). The Metropolis-Hastings scheme was then applied within the Bayesian inference framework in image analysis by Geman and Geman (1984). It is a specific case of the MCMC method. The Metropolis method makes use of a proposal distribution to sample from target distribution $\pi(\theta)$ by building a Markov chain whose stationary distribution is considered as the target distribution. In a Bayesian framework, the target distribution $\pi(\theta)$ is the posterior distribution for model parameters $\pi(\theta|y)$.

As mentioned in section 3.3, the MH algorithm works by generating a Markov chain $(\theta_1, \cdots, \theta_N)$, which converges towards its unique stationary distribution under the ergodicity condition.

The sufficient condition for ensuring the chain converges to a unique stationary distribution is detailed balance (equation (3.28)), which must be satisfied:

$$\pi(\theta)T(\theta'|\theta) = \pi(\theta')T(\theta|\theta') \quad \forall\theta, \theta', \tag{3.30}$$

Let us start with an algorithm that performs a random walk through the sample space Θ, using proposal distribution $q(\theta'|\theta)$. The proposal distribution follows the Markov property which means it only depends on the last value of the state variable and is independent of all its previous values. The proposal density $q(\theta'|\theta)$ represents the probability density that, given that the current state is θ, the proposed value will be θ'. We can find for instance, that the movement from the state θ to state θ' can occur more probably, that is:

$$\pi(\theta)q(\theta'|\theta) > \pi(\theta')q(\theta|\theta'). \tag{3.31}$$

The inequality (3.31) shows that the average number of movements from state θ to θ' are occurred more likely. Chib and Greenberg (1995) state that it is possible to reduce the average number of movements from the state θ to θ'. This can be done through defining a probability of movement (denoted a) to be $a(\theta, \theta') < 1$. However, in case of movement is not occurred, the process returns to state θ as a value from our target distribution. The transitions from state θ to θ' are occurring according to:

$$T(\theta'|\theta) = q(\theta'|\theta)a(\theta, \theta'), \quad \theta \neq \theta',$$

where a is to be defined.

As the inequality

$$\pi(\theta)q(\theta'|\theta) > \pi(\theta')q(\theta|\theta')$$

shows the average of movement from θ' to θ is less frequent than θ to θ'. Hence, $a(\theta, \theta')$ should be specified to be as large as possible, it is assumed to be 1 as the upper limit of probability is 1.

To determine the probability of movement $a(\theta, \theta')$ the reversibility condition must be satisfied on $T(\theta'|\theta)$ as the following:

$$\pi(\theta)q(\theta'|\theta)a(\theta, \theta') = \pi(\theta')q(\theta|\theta')a(\theta', \theta).$$

$$\pi(\theta)q(\theta'|\theta)a(\theta, \theta') = \pi(\theta')q(\theta|\theta'),$$

and so

$$a(\theta, \theta') = \frac{\pi(\theta')q(\theta|\theta')}{\pi(\theta)q(\theta'|\theta)}. \tag{3.32}$$

Thus, the probabilities $a(\theta, \theta')$ and $a(\theta', \theta)$ are defined to guarantee that the both sides of inequality (3.31) are in balanced, which implies that $T(\theta'|\theta)$ satisfied the reversibility. Chib and Greenberg (1995) stated that the probability of move should be set to:

$$a(\theta, \theta') = min\left\{\frac{\pi(\theta')q(\theta|\theta')}{\pi(\theta)q(\theta'|\theta)}, 1\right\}.$$

The chain is initialised arbitrarily from some θ_1, then a new candidate value θ' is proposed from the chosen proposal distribution $q(\theta'|\theta)$, which depends on the current sample value θ. The proposed θ' can be either accepted into our sampling chain or rejected according to the acceptance probability $a(\theta', \theta)$:

$$a(\theta', \theta) = min\left\{1, \frac{\pi(\theta')q(\theta|\theta')}{\pi(\theta)q(\theta'|\theta)}\right\} = min\left\{1, \frac{\pi'(\theta')Zq(\theta|\theta')}{\pi'(\theta)Zq(\theta'|\theta)}\right\}. \tag{3.33}$$

It is evident from equation (3.33) that the normalisation constant term Z in the acceptance probability ratio cancels. Hence, evaluation of the acceptance probability only requires computing the unnormalised target distribution.

Algorithm 7 Metropolis-Hastings Algorithm

1: Initialise the starting value randomly θ
2: Propose a candidate value to move $\theta' \sim q(\theta'|\theta)$
3: Accept the proposed move θ' with probability

$$a(\theta', \theta) = min\left\{1, \frac{\pi(\theta')q(\theta|\theta')}{\pi(\theta)q(\theta'|\theta)}\right\}$$

4: **else**
5: Reject θ' and remain at current state θ.

If the proposal distribution choice is symmetric such that $q(\theta'|\theta) = q(\theta|\theta')$, then the algorithm becomes a special case of MH and is known as a Metropolis algorithm (Metropolis et al., 1953).

The resulting Markov chain from the MH algorithm explores the space of target distribution by taking a small advantage from the previously accepted value. It can be seen that the MH algorithm as described in the algorithm 7 accepts the proposed value θ' more likely when the proposed move is assigned with the largest probability under the target distribution, compared to the previous state θ.

The efficiency of the MH scheme relies on selecting a suitable proposal distribution, which is determined by the user and the obtained acceptance rate. There are two main common types of proposal distribution. The first choice can be an independent normal distribution, which is given by:

$$q(\theta'|\theta) = q(\theta') = N(\theta'; \mu, \sigma), \tag{3.34}$$

where the candidate's proposal θ' is independent of the current state θ, which means that the proposal cannot use the previously accepted value as guidance to reach the target distribution. Performing an MH algorithm within an independent proposal can be inefficient as it works in a similar way to the basic accept-reject method described in 3.1.1.

The other type of proposal which is widely employed in MH algorithms is the normal random walk, which is defined as follows:

$$q(\theta'|\theta) = N(\theta'; \theta, \sigma), \tag{3.35}$$

where this type of proposal depends on the current state (parameter value) of the chain θ_i and tuning the covariance σ can influence the performance of the algorithm. As a consequence, if σ is small, then the algorithm accepts a small movement in a particular area of the state space, which results in a high acceptance rate and strong autocorrelation. On the other hand, if σ is large, the candidate proposed value can lie in the area of low probability mass, and most of the proposed values are rejected, which makes the convergence towards stationary distribution slow. Both extreme values of σ eventually will result in a bad mixing of the resulting Markov chain from the MH algorithm.

Further details about the Metropolis-Hastings algorithm and the best way of tuning the proposal summarised in section 3.5 (Robert, 2004), (Roberts et al., 1997), (Gelman et al., 2013). The proposal should be chosen carefully for it can be misleading in some cases. The influence of the proposal distribution on the MH algorithm performance is illustrated in example1.

Example1: ***Sampling from the Beta Distribution Using MCMC***

In this section we will demonstrate how the Metropolis-Hastings algorithm works. Let us assume that our target distribution is $\pi(x) = \beta(0.5, 0.5)$ with density:

$$\pi(x; \alpha, \beta) \propto x^{\alpha-1}(x-1)^{\beta-1}.$$

We are aiming to draw samples from this distribution to show how the Metropolis-Hastings (MH) algorithm works.

The MH starts with an initial value x, which is drawn randomly from the support of our target distribution. Next, we propose a new value x' from a proposal density $q(x'|x)$. In this example, we use the standard normal distribution as our proposal density. The new value will be accepted or rejected from our sample according to the acceptance probability $a(x', x) = \frac{\pi(x')}{\pi(x)}$. Since the proposal density is symmetric, the ratio $\frac{q(x'|x)}{q(x|x')}$ will cancel out in the acceptance probability.

Figure 3.1 shows the relative frequency of samples produced by this algorithm and the red line represents the probability density of that beta distribution. These two distributions match well, and this demonstrates that our MH algorithm is working correctly.

Despite the MH algorithm giving a reasonable sample, there is still a problem as many of the proposed points are rejected from the target because the target distribution and the proposal density have different support.

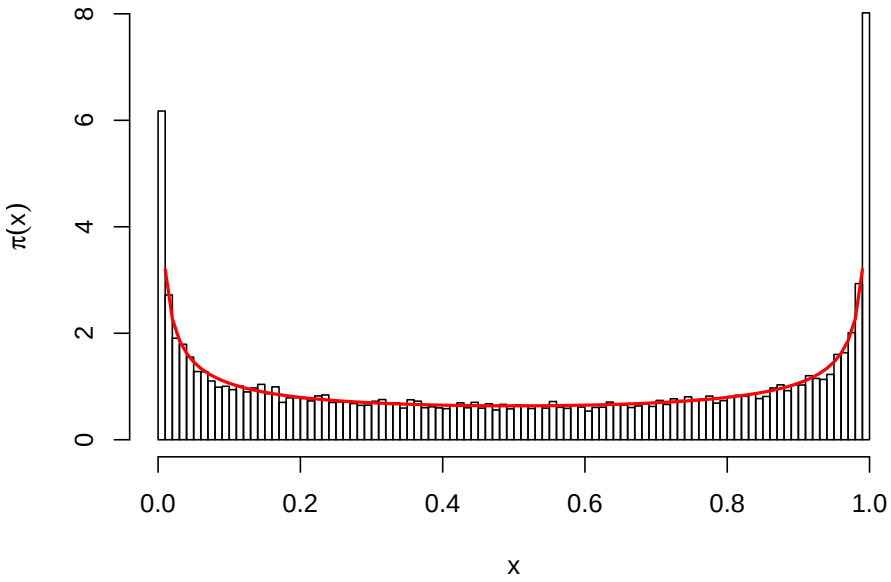


Figure 3.1: The plot shows the histogram of samples from the beta distribution produced with the MH algorithm and the red line shows the probability density of our target distribution. The histogram matches the target distribution reasonably well.

For instance, at iteration n, the current state is $x = 0.01$, the standard normal distribution is used to propose a new sample at iteration $n + 1$, a large proportion of our proposed values x' will be negative. Therefore, the proposed sample will be rejected, the past state is returned and recorded as current state and does not augment the counter, which implies that the proposal density is no longer symmetric. This also causes the undesirable auto-correlation in the produced sample. A common solution to this problem is to resample the proposal, if the previous value falls outside of the target support. However, this approach introduces bias to the MH algorithm.

The other unbiased case is that the MH algorithm proposes a new sample x', if the proposed sample is negative, it will be rejected and recorded as the current state. Thus, the counter augmented which means that the proposal density is symmetric. This results in the Metropolis algorithm with a high rejection rate.

Figure 3.2 shows the relative frequency of samples produced by this algorithm and the red line represents the probability density of that beta distribution. However, these two distributions do not match well, and this demonstrates that there is a problem.

The idea of resampling from the proposal density effectively truncates the proposal density to the support of our target distribution. This mean that the proposal

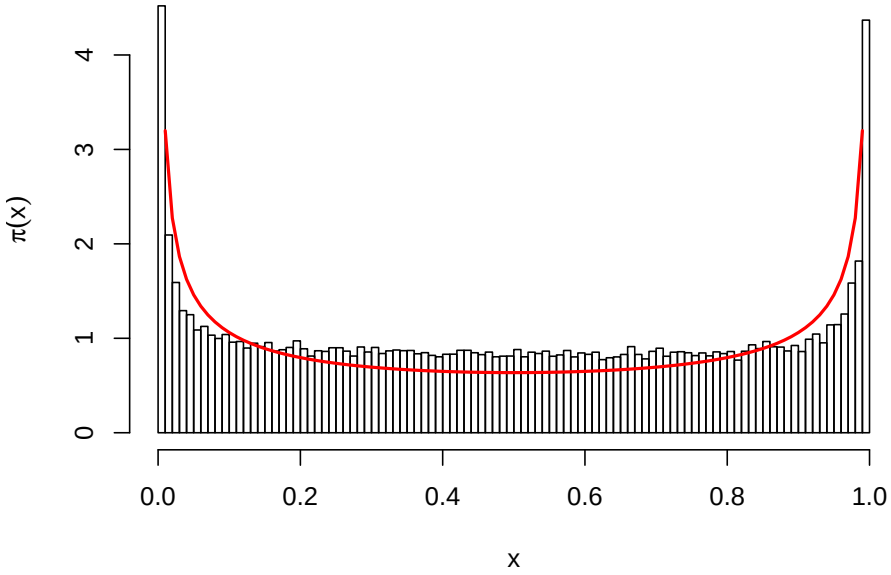


Figure 3.2: The plot shows the histogram of samples from the beta distribution produced with the MH algorithm with a tweaked proposal; and the red line shows the probability density of our target distribution. The histogram does not match the target distribution anymore. Our proposal correction introduces bias to the algorithm.

density is no longer symmetric.

Therefore, we will account for the proposal density in our acceptance probability:

$$a(x', x) = \frac{q(x|x')\pi(x')}{q(x'|x)\pi(x)},$$

where

$$q(x'|x) = \frac{\frac{1}{\sigma}p(\frac{x'-x}{\sigma})}{P(\frac{1-x}{\sigma}) - P(\frac{0-x}{\sigma})}\mathbb{1}_{[0,1]}(x),$$

and

$$q(x|x') = \frac{\frac{1}{\sigma}p(\frac{x'-x}{\sigma})}{P(\frac{1-x'}{\sigma}) - P(\frac{0-x'}{\sigma})}\mathbb{1}_{[0,1]}(x),$$

where p denotes a probability density function of standard normal, P is cumulative standard normal distribution, σ is the standard deviation of the proposal density. The indicator function is:

$$\mathbb{1}_{[0,1]}(x) = \begin{cases} 1 & \text{if } 0 \leq x \leq 1 \\ 0 & \text{otherwise.} \end{cases}$$

The ratio of $\frac{q(x|x')}{q(x'|x)}$ will be:

$$\frac{q(x|x')}{q(x'|x)} = \frac{P(\frac{1-x}{\sigma}) - P(\frac{0-x}{\sigma})}{P(\frac{1-x'}{\sigma}) - P(\frac{0-x'}{\sigma})}.$$

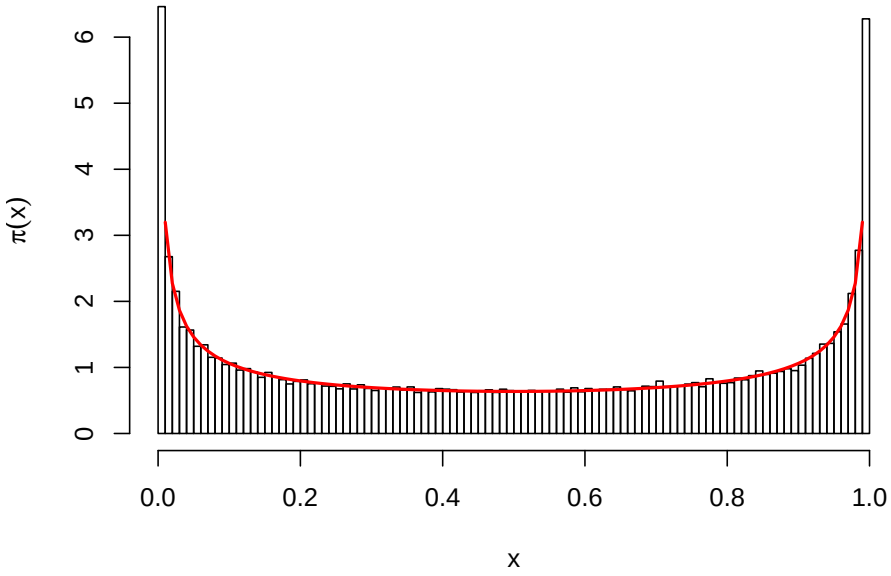


Figure 3.3: The plot shows the histogram of samples from the beta distribution produced with the MH algorithm using a correctly truncated proposal; and the red line shows the probability density of our target distribution. The histogram matches the target distribution again. Therefore, we fixed the bias problem.

Finally, in Figure 3.3 the histogram matches the target distribution again; this means we have fixed our bias problem.

3.5 Optimal Choice of Proposal for the MH

The efficiency of the Metropolis-Hastings approach relies on choosing an appropriate proposal distribution. The well-known choice of the proposal density is a normal random walk proposal $N(\theta, \mathbf{\Sigma})$. However, the main issue with the random walk proposal is how to scale the movement length which is based on the covariance matrix. If a small movement occurred in each MH iterations, the chain moves slowly towards the target distribution, and the acceptance rate will be high, which

means the parameter space is not explored well. In contrast, a large move decreases the acceptance rate. Thus, the chain can be stuck in a specific region. It is very important to avoid the extreme value of the movement steps. A suitable proposal move will be made with a plausible acceptance probability that is far away from 0 and 1.

It is possible to monitor the acceptance rate of the MH algorithm in order to avoid these extremes. But we do not yet know what optimal acceptance rate should be achieved. Despite the limitations of the theoretical result in this, Roberts et al. (1997) state that the reasonable acceptance rate for MCMC is 0.23. As the performance of the MH algorithm relies on selecting a suitable proposal, the proposal should be chosen carefully, exploring the parameter space and controlling the acceptance rate. In order to overcome this issue, an adaptive proposal was introduced to be used within the Metropolis algorithm to tune the proposal during the iterations. The proposal is tuned according to the evaluation of the target covariance from a previous state value, which ensures that the Markov chain is adapted to the target distribution.

Therefore, the appropriate scaling for a random walk is achieved by using an adaptive normal proposal $\mathbf{\Sigma}$ that is designed to adapt the proposal distribution during the run to keep the acceptance rate within the optimal acceptance range (Gelman et al., 2013).

The Metropolis-Hastings algorithm can make use of symmetric normal proposal, which takes the following form:

$$q(\theta'|\theta_i) = N(\theta; \theta', c^*\hat{\mathbf{\Sigma}}), \tag{3.36}$$

An efficient scale in practice is $c^* = (\frac{2.4}{d})^2$ where d is the model parameter number. The suggested steps in an adaptive sampling algorithm to obtain an efficient Metropolis algorithm are:

1. start sampling the Markov chain using a fixed proposal distribution that is applying the Metropolis-Hastings algorithm scheme.
2. After a few iterations, the Metropolis-Hastings proposal distribution is updated as follows:
   (a) The covariance of the proposal distribution is adjusted to be proportional to the target covariance matrix estimated from the samples.

(b) Based on the acceptance rate, adjust the scale of proposal distribution c^2 either increasing or decreasing this scale, in order to keep an acceptance rate in the range between 0.23 and 0.44.

However, the main question arising is under which conditions the adaptive method preserves the stationary distribution of the Markov Chain. Gelman et al. (2013) proposed a scheme that initially made use of the Markov chain but with a different updating proposal to obtain the best parameter values; the samples were then obtained from the final run of the Markov chain, assumed to be convergent with the target distribution and hence the chain should be restarted after the proposal has been fixed. For more details about the cases of using this adaptive method with the MCMC algorithm, see (Andrieu et al., 2006) and (Atchadé et al., 2005).

3.5.1 Methods for Monitoring Convergence

A critical issue when performing the MCMC algorithm is to decide when the algorithm should be stopped and to estimate the target of interest based on this sample. In practice, it is impossible to obtain a Markov chain, the invariant distribution of which is equivalent to the target from early iterations. Hence, the Markov chain can run for a period of time until it reaches its invariant distribution from which we have been aiming to sample. As the initial proposed samples of the chain can be a poor representative of the target distribution, it would be better to discard these initial samples. This is known as a burn-in. All obtained samples after the burn-in phase are assumed to be drawn from the invariant distribution. A typical question is how long the burn-in phase should be. It is possible to run multiple chains with distinct initial values and visualise the chain and then explore whether all chains converged to the same stationary distribution or not. Based on the pilot run, the length of the burn-in can be specified, but in general, it is difficult to determine the burn-in period precisely. However, it is beneficial to perform convergence diagnosis test to ensure the Markov chain obtained after the burn-in period has converged to its equilibrium distribution. Various methods are highlighted and described for convergence monitoring in (Cowles and Carlin, 1996).

In this thesis, we are interested in investigating a particular convergence test method that was suggested by Gelman et al. (2013). This test is designed to run several different chains in parallel initialised from distinct values, and then a statistical comparison is performed on the output from these chains. Each dimension of the parameter space is monitored separately (Gelman et al., 2011).

Assuming that we run m chains in parallel, each producing n samples, we label the individual draws as θ_{ij} $(i = 1, \cdots, n, \quad j = 1, \cdots, m)$. These draws are considered separately for each scalar dimension of the parameter space.

Then we compute between B and within-sequence W variances for each θ as in the following:

$$B = \frac{n}{m-1} \sum_{j=1}^{m} (\bar{\theta}_{.j} - \bar{\theta}_{..})^2 \tag{3.37}$$

where

$$\bar{\theta}_{.j} = \frac{1}{n} \sum_{i=1}^{n} \theta_{ij}$$

and

$$\bar{\theta}_{..} = \frac{1}{m} \sum_{j=1}^{m} \bar{\theta}_{.j}$$

$$W = \frac{1}{m} \sum_{j=1}^{m} s_j^2$$

where

$$s_j^2 = \frac{1}{n-1} \sum_{i=1}^{n} (\theta_{ij} - \bar{\theta}_{.j})^2.$$

The marginal target variance $var(\theta)$ of θ is estimated by the averaging of B and W:

$$\widehat{var}(\theta|y) = \frac{n-1}{n} W + \frac{1}{n} B, \tag{3.38}$$

which can be considered to be an unbiased estimator of the marginal posterior variance $var(\theta|y)$ when the initial values of the chains were sampled from the target distribution. In the case when the initial distribution is overdispersed, the estimator of $var(\theta|y)$ can be considered as an overestimation.

For any finite sequence n, the variance W can be an underestimate of $var(\theta|y)$. This is due to the finite sample: where each sequence has not had enough time to explore all the target distribution, and therefore less variability is achieved. In the limit as $n \to \infty$, the expectation of within-sequence variance W approximates $var(\theta|y)$.

Gelman et al. (2013) state that monitoring convergence can be performed by estimating the potential scale factor, by means of which the scale of the current distribution for θ can be further reduced when the simulation is carried out in the limit $n \to \infty$. The potential scale reduction $\widehat{R}$ can be expressed by the following:

$$\widehat{R} = \sqrt{\frac{\widehat{var}(\theta|y)}{W}}.$$

If $\widehat{R}$ is large, more simulation is required in order to decrease the variance $\widehat{var}(\theta|y)$ where $\widehat{R}$ is decreased to 1 as $n \to \infty$. If $\widehat{R}$ is below a set of the threshold, e.g. 1.1, then the chains have converged to the stationary distribution.

Example2: ***Metropolis-Hasting for Quadratic Model***

In this section we have demonstrated how the MH algorithm can be applied to simple models within the Bayesian framework. Usually we want to infer the posterior which is:

$$\pi(\theta|y) = \frac{\pi(y|\theta)\pi(\theta)}{\pi(y)} = \frac{\pi(y|\theta)\pi(\theta)}{\int \pi(y|\theta)\pi(\theta)d\theta}.$$

However, evaluating the integral in the denominator can indeed be intractable. To avoid this problem, we can use MH to sample from the posterior $p(\theta|y)$. When we sample from the posterior, the acceptance probability is:

$$a(\theta', \theta) = \frac{\pi(\theta'|y)}{\pi(\theta|y)} \times \frac{q(\theta|\theta')}{q(\theta'|\theta)} = \frac{\frac{\pi(y|\theta')\pi(\theta')}{\int \pi(y|\theta')\pi(\theta')d\theta'}}{\frac{\pi(y|\theta)\pi(\theta)}{\int \pi(y|\theta)\pi(\theta)d\theta}} \times \frac{q(\theta|\theta')}{q(\theta'|\theta)} = \frac{\pi(y|\theta')\pi(\theta')}{\pi(y|\theta)\pi(\theta)} \times \frac{q(\theta|\theta')}{q(\theta'|\theta)},$$

and the intractable integral cancels out completely. Now, we will demonstrate how this algorithm works on simple polynomial models. We will consider the quadratic model :

$$y_i = a + bx_i + cx_i^2 + \epsilon_i,$$

where, $\epsilon_i, \overset{iid}{\sim} N(0, \sigma^2)$ and we want to make inference about the model parameters $\theta = (a, b, c, \sigma^2)$.

Assume that we have n observations $y = \{(x_i, y_i), i = 1, \ldots, n\}$ that we want to explain using our model.

In a Bayesian framework we are interested in drawing a sample from the posterior distribution which can be defined as:

$$\pi(a, b, c, \sigma^2|y) \propto \pi(y|a, b, c, \sigma^2)\pi(a, b, c, \sigma^2),$$

where the prior is assumed to be independent as:

$$\pi(a, b, c, \sigma^2) = \pi(a)\pi(b)\pi(c)\pi(\sigma^2),$$

and we assumed a uniform prior for a, b, c and a gamma prior for σ^2 such that:

$$\begin{aligned}\pi(a) &= U(-2, 2)\\ \pi(b) &= U(-2, 2)\\ \pi(c) &= U(-2, 2)\\ \pi(\sigma^2) &= \Gamma(1, 3).\end{aligned}$$

The likelihood is:

$$\pi(y|a, b, c, \sigma^2) = \prod_{i=1}^{n} \pi(y_i|a, b, c, \sigma^2, x_i) = \prod_{i=1}^{n} \frac{1}{\sigma\sqrt{2\pi}} exp\left\{\frac{-(y_i - a - bx_i - cx_i^2)^2}{2\sigma^2}\right\}.$$

Our aim is to draw samples from our desired posterior $p(a, b, c, \sigma^2|y)$. Sampling is performed by using Markov chain Monte Carlo with the MH algorithm. We initialise the chain by drawing a sample randomly from the prior. Then, at each iteration, we propose a new candidate value from a proposal density q. In our case, the proposal density is set to be a normal distribution. As the proposal distribution is symmetrical, the term $\frac{q(\theta|\theta')}{q(\theta'|\theta)}$ will be cancelled from the ratio a. Then, the proposed value is accepted with probability a:

$$a(\theta', \theta) = \frac{\pi(y|\theta')\pi(\theta')}{\pi(y|\theta)\pi(\theta)}.$$

If the proposed point is accepted then we set this point to be our current value in the chain. If this point is rejected, the chain will stay at the previous value.

As we initialise our chain by drawing a start-point from the prior and performing the random walk, the chain will take some steps before converging to its stationary distribution. In order to eliminate the effect of these starting values, we discard the first part of the sample from the chain: this process is known as burn-in. To decide how long burn-in is sufficient, we will use a method that assesses the convergence.

The approach that we applied in our example is known as the potential scale reduction method (Gelman et al., 2013). This method is based on comparing different simulated chains. In this example, we start by simulating two independent chains for 10000 steps. For each parameter, we compute the potential scale reduction factor

$\hat{R}$. If the $\hat{R}$ is below 1.1 we will stop the burn-in, otherwise, we will continue the burn-in for another 10000 steps and repeat the convergence evaluation and we aim to obtain acceptance rate between 20% and 60%.

The result of the marginal posterior density for each parameter of the quadratic model is depicted in Figure 3.4.

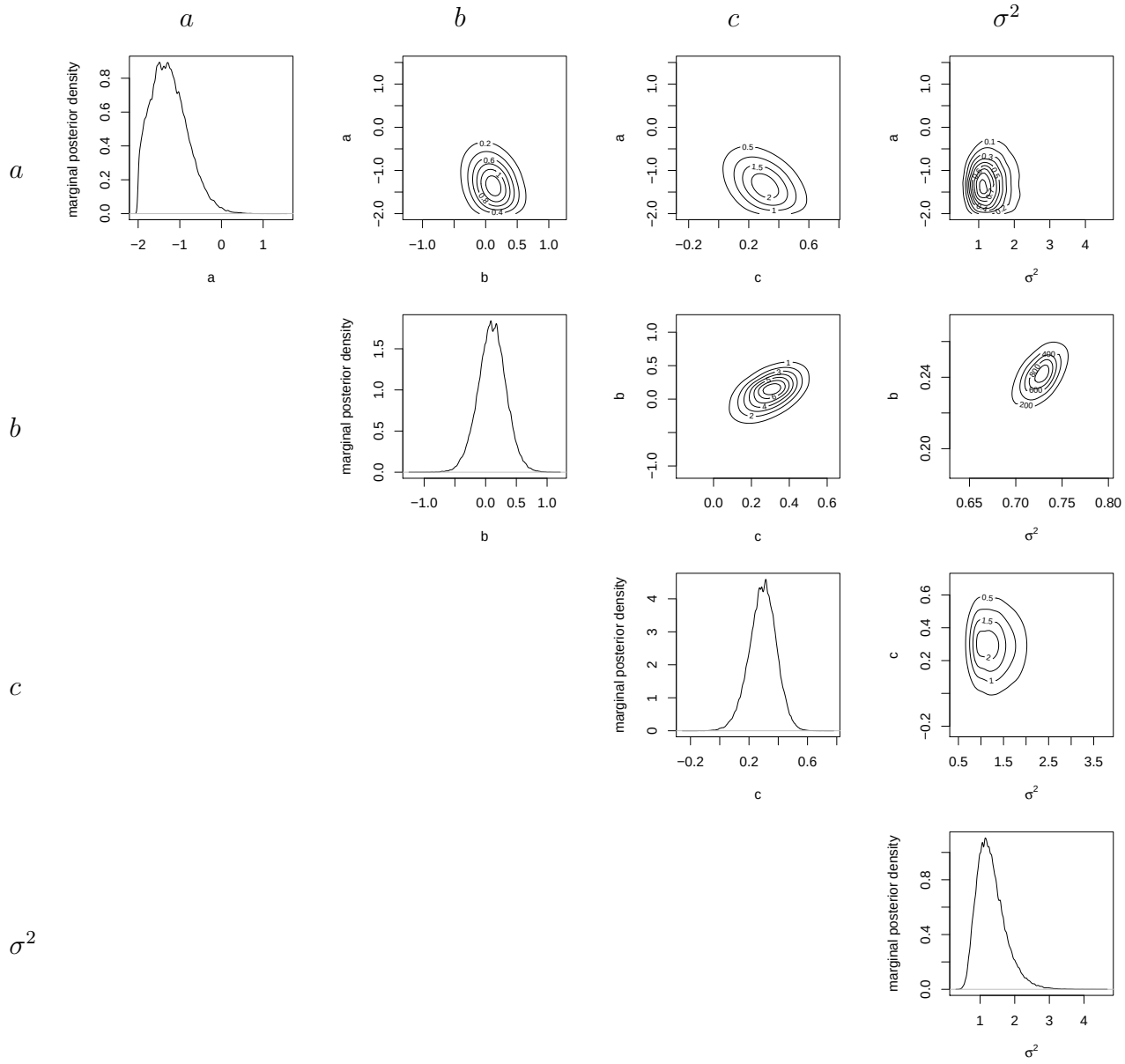


Figure 3.4: The posterior density of parameters of the quadratic model. The posterior is a four dimensional probability density function, therefore we depict it with a set of marginal posterior density plots.

Once we have inferred the joint parameter posterior, we can then find the distribution of unobserved observations $\tilde{y}$ conditional on the observed data. This is called the posterior predictive distribution

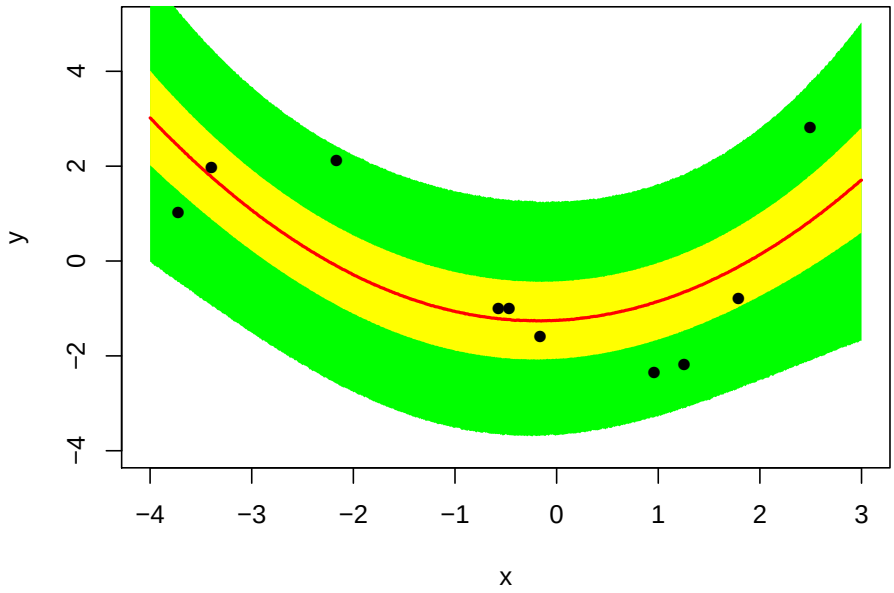


Figure 3.5: The posterior predictive distribution for the quadratic model is compared to the experimental data. The data are depicted with black points. The green interval is the 95% credibility interval of the predictions. The yellow interval is the 50% credibility interval of the predictions. The red line is the median of the predictions.

Looking at Figure 3.5, we can see that the predictive posterior matches the data reasonably well. However, it appears that a more complex model is needed to explain the data better.

Now, we will perform the same procedure using a cubic model:

$$y_i = a + bx_i + cx_i^2 + dx_i^3 + \epsilon_i,$$

where, $\epsilon_i \overset{iid}{\sim} N(0, \sigma^2)$.

considering the same data $y = (x_i, y_i)$. The parameter posterior was inferred in the same way as for the quadratic model. We also show the predictive posterior plot depicted in Figure 3.6. The predictive posterior distribution still matches the data very well, while providing a more reasonable fit.

Standard MCMC methods can not be applied in some systems that are described with a Markov process as it requires the evaluation of the likelihood function which is intractable or infeasible. In order to overcome such an issue, two approaches based on Pseudo-Marginal MCMC are investigated. The first one is considered as an exact approximation which approximates the likelihood and used it in MH algorithm. The second one is a novel Bayesian method proposed by Georgoulas et al. (2017) which

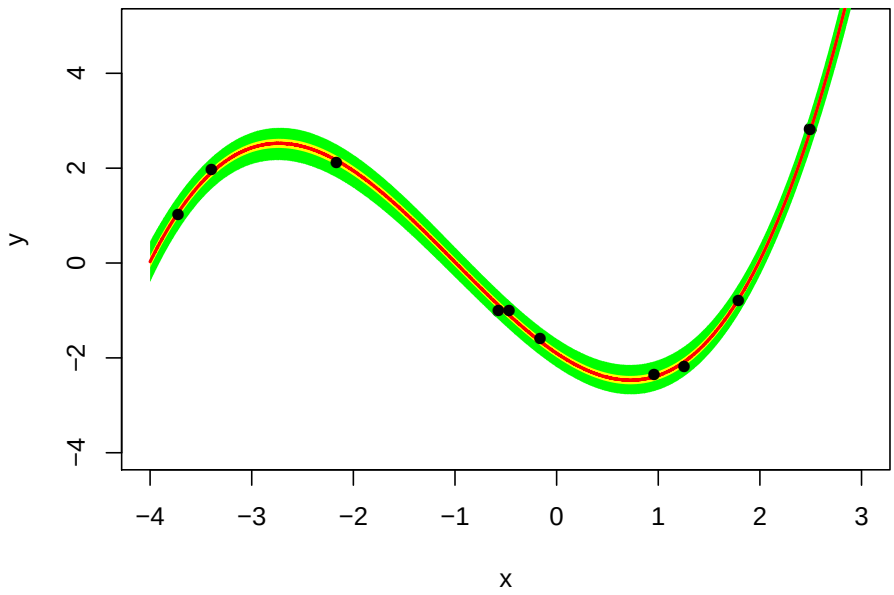


Figure 3.6: The posterior predictive distribution for the cubic model gives a better match to the data. The data are depicted with black points. The green interval is the 95% credibility interval of the predictions. The yellow interval is the 50% credibility interval of the predictions. The red line is the median of the predictions.

provides a way to perform parameter inference for Markov processes. The approach is discussed in section 3.8.

3.6 Pseudo-Marginal MCMC

Performing inference using the MH algorithm requires the evaluation of the acceptance probability ratio (3.33), which relies on the computation of the likelihood $\pi(y|\theta)$. However, the MH method cannot be applied due to the intractability of the likelihood term $\pi(y|\theta)$. Alternatively, the unknown likelihood function $\pi(y|\theta)$ can be estimated using an unbiased and non-negative Monte Carlo estimator $\hat{\pi}(y|\theta)$ and then substituted in the acceptance probability ratio (3.33). The resulting algorithm is known as a pseudo-marginal MCMC, which was introduced by Beaumont (2003) and generalised and extended by Andrieu and Roberts (2009).

The likelihood function we aim to approximate is:

$$\pi(y|\theta) = \prod_{m=1}^{M} \pi(y_m|\theta),$$

which is intractable. Instead, we can sample from the augmented distribution

$\hat{\pi}_N(\theta, u|y)$, where u represents the auxiliary variable. A prior $\pi(u)$ is defined for the random variable u. In addition, suppose that θ and u are independent. Therefore, their joint prior can be expressed as:

$$\pi(\theta, u) = \pi(\theta)\pi(u).$$

Let us assume that the marginal likelihood can be approximated with introducing the auxiliary variable u is:

$$\hat{\pi}_N(y|\theta) = \int \hat{\pi}_N(y, u|\theta)du = \int \hat{\pi}_N(y|u, \theta)\pi(u|\theta)du = \int \hat{\pi}_N(y|u, \theta)\pi(u)du,$$

where, because of the assumption of a prior independence between θ and u, we set $\pi(u|\theta) = \pi(u)$. Then, we can say the likelihood estimator is unbiased if:

$$\mathbb{E}(\hat{\pi}_N(y|\theta, u)) = \int \hat{\pi}_N(y|\theta, u)\pi(u)du = \pi(y|\theta). \tag{3.39}$$

The joint density for both random variables can be defined as:

$$\pi_N(\theta, u|y) = \frac{\pi(\theta)\hat{\pi}_N(y|\theta, u)\pi(u)}{\pi(y)},$$

$$\pi_N(\theta, u|y) \propto \pi(\theta)\hat{\pi}_N(y|\theta, u)\pi(u).$$

where

$$\pi(y) = \int\int \hat{\pi}_N(y|\theta, u)\pi(u)\pi(\theta)dud\theta = \int \pi(\theta)\{\hat{\pi}_N(y|\theta, u)\pi(u)du\}d\theta = \int \pi(\theta)\pi(y|\theta)d\theta.$$

The marginal distribution of the joint $\pi_N(\theta, u|y)$ is $\pi(\theta|y)$ which can be obtained using the equation (3.39):

$$\begin{aligned}
\int \pi_N(\theta, u|y)du &= \int \frac{\pi(\theta)\hat{\pi}_N(y|\theta, u)\pi(u)}{\pi(y)}du \\
&= \frac{\pi(\theta)}{\pi(y)} \int \hat{\pi}_N(y|\theta, u)\pi(u)du \\
&= \frac{\hat{\pi}(y|\theta)\pi(\theta)}{\pi(y)} \\
&= \pi(\theta|y).
\end{aligned}$$

This implies that the generated samples θ from the desired distribution $\pi(\theta|y)$ by sampling from $\pi_N(\theta, u|y)$.

The advantage of the pseudo-marginal MCMC incorporated with SMC that it enables new powerful approaches, namely, PMCMC (Andrieu et al., 2010),(Wilkinson, 2010a).

3.7 Particle Markov Chain Monte Carlo

In this section, we focus on a particular type of MCMC method, one which enables us to perform inference when the likelihood is intractable. The MH is applied for sampling from the distribution of interest $\pi(\theta|y)$ by constructing a Markov chain on the parameter space Θ as explained in section 3.4. The candidate parameter θ' is drawn during the iteration step i given the current parameter value θ from a density $q(\theta'|\theta)$. The proposed value is accepted with probability $a(\theta', \theta)$, which is defined in equation (3.33).

But, in the case of performing Bayesian inference on a stochastic Markovian model, constructing MH can be a challenging task as the MH approach requires the evaluation of the likelihood.

Hence, in many situations especially in complex and high dimensional models, the acceptance probability in (3.33) cannot be evaluated due to the difficulty of computing the likelihood that occurs if the model contains latent variables x (e.g. states in CTMC model). Instead, Andrieu and Roberts (2009) present an approach that can deal with such a crucial problem. This method is known as pseudo-marginal MCMC. The idea of this approach is illustrated in section 3.6, which basically approximates the acceptance probability ratio (Andrieu and Roberts, 2009) and (Andrieu et al., 2010). In an MH algorithm, the intractable likelihood $\pi(y|\theta)$ in acceptance probability (3.33) can be replaced with a non-negative and unbiased estimator, denoted as $\hat{\pi}(y|\theta)$.

Then, the intractable likelihood $\pi(y|\theta)$ term in the acceptance probability (3.33) can be replaced with a non-negative and unbiased estimator $\hat{\pi}(y|\theta)$ such that:

$$a(\theta', \theta) = min\left\{1, \frac{\pi(\theta')q(\theta|\theta')\hat{\pi}(y|\theta')}{\pi(\theta)q(\theta'|\theta)\hat{\pi}(y|\theta)}\right\}. \tag{3.40}$$

In order to obtain a stationary distribution from a Markov chain that is exactly similar to the desired distribution, the necessary condition, namely, a non-negative and unbiased estimator, must hold. The bootstrap particle algorithm, as illustrated in section 3.2.5 can be used to estimate the likelihood of the model, and then plug it

Algorithm 8 Particle Marginal Metropolis-Hastings Algorithm

1: Initialise the starting value arbitrarily θ
2: Run a particle filter (SMC) to obtain an unbiased estimate of the marginal likelihood $\hat{\pi}(y|\theta)$.
3: Sample θ' from $q(\theta'|\theta)$
4: Run a particle filter (SMC) to obtain an unbiased estimate of the marginal likelihood $\hat{\pi}(y|\theta')$.
5: Accept the proposed parameter θ', the marginal likelihood estimator $\hat{\pi}(y|\theta')$ with probability:

$$a = min\left\{1, \frac{\hat{\pi}(y|\theta')\pi(\theta')q(\theta|\theta')}{\hat{\pi}(y|\theta)\pi(\theta)q(\theta'|\theta)}\right\}$$

and set $(\theta, \hat{\pi}(y|\theta)) = (\theta', \hat{\pi}(y|\theta'))$
6: **else**
7: reject θ', and $\hat{\pi}(y|\theta')$, remain at current state.

into acceptance probability (Andrieu and Roberts, 2009) and (Andrieu et al., 2010). The resulting acceptance probability after substituting $\hat{\pi}(y|\theta')$ is shown in (3.40).

Regardless of the approximation in the acceptance probability, the method can target the exact posterior when the resulting likelihood estimator is non-negative and unbiased. The algorithm is considered to be an exact approximation (Andrieu et al., 2010), (Wilkinson, 2011).

Within some state space model, PMMH algorithm is aiming to target the full joint distribution of interest e.g $\pi(\theta, x|y)$. However, in this thesis, only a specific case of PMMH, combined with pseudo marginal is addressed. More details about the PMCMC that targets the joint distribution can be found in (Andrieu et al., 2010), (Golightly and Wilkinson, 2011), (Pitt and Shephard, 1999)

The general properties of the PMMH scheme such as: selecting an appropriate proposal density, the acceptance rate and convergence diagnosis are similar to MH.

Nevertheless, the performance efficiency of this algorithm usually relies on the number of particles N that are used to approximate the target distribution. Yet a small number of N can result in significant variance in the estimates and, as a result, the Markov chain can be stuck with a low acceptance rate. Moreover, the number of particles has a direct influence on the computational time of the target approximation.

3.8 Exact Inference

The MCMC method relies on likelihood evaluation which can be intractable for a CTMC especially when the state of Markov chain is unbounded or large. Due to state space size, the direct inference is difficult. Yet, adopting a pseudo-marginal method, which is based on a random truncation, can provide an exact Bayesian inference to such difficult problem (Georgoulas et al., 2017). The method is constructed on the auxiliary variable Gibbs scheme that is proposed in (Rao and Teh, 2013). Then, the likelihood is formulated to permit the deployment of a Russian Roulette random truncation procedure as described in (Lyne et al., 2015) and (Filippone and Girolami, 2014). This results in two novel developed pseudo-marginal sampling methods for Markov process. The first one is a Metropolis-Hastings pseudo-marginal approach and the second one is an auxiliary variable pseudo-marginal Gibbs sampler. In this thesis, we focus on the second scheme, and it will be performed on one of the biological case studies considered in this thesis.

3.8.1 Sampling a Trajectory

As we mentioned in section 2.6.1, the analytical analyses of a CTMC through the CME can be difficult or computationally expensive. An alternative method is to use uniformisation as described in section 2.6.1. The main idea behind the uniformisation is to build a DTMC by assuming a common exit rate for all Markov states, denoted q. This results in a sample path (X, T) from the discrete process. Standard methods can address the resulting discrete system. Hence, a new sample path of the process, given observations is drawn according to a forward filtering-backward sampling scheme (Rao and Teh, 2013), (Georgoulas et al., 2017).

3.8.2 Gibbs Sampling for Finite State

In this section, an exact Gibbs sampler for the unknown parameter will be considered (following the explanation in section 3.1 in (Georgoulas et al., 2017)). Let us consider a particular case when the reaction of the system is associated with the functional form of the stochastic law is:

$$h_i(x) = \theta_i \rho_i(x), \tag{3.41}$$

where $\rho_i(x)$ represents a fixed function of the state x and θ_i is an unknown parameter.

Then, a conjugate prior for θ_i is suggested to simplify the sampling step in the Gibbs algorithm.

Let us assume that $X = \{x_0, x_1, \cdots, x_N\}$ is a sequence of states at a sequence of times $T = \{t_0, t_1, \cdots, t_N\}$ which is form the full sampling path of the process. Each reaction is associated with a different update vector v. The reaction at time t_{n+1} is denoted by v_n, where the update vector is $(x_{n+1} - x_n)$. Then, the total rate of the state x_n based on equation (3.41) is given by:

$$h(x_n) = \sum_{i=1}^{L} \theta_i \rho_i(x_n),$$

where L represents distinct reaction types. As we assumed that the waiting time for the CTMC follows the exponential distribution and it can be defined as:

$$\pi(t_{n+1}|t_n, x_n) = h(x_n)e^{-dt_n h(x_n)}, \quad \text{where} \quad dt_n = t_{n+1} - t_n. \tag{3.42}$$

Then, we can define the probability of the following state x_{n+1} is $\frac{\theta_{v_n}\rho_{v_n}(x_n)}{h(x_n)}$, and hence the likelihood can be formulated as follows:

$$\pi(X, T|\theta) = \pi(X|\theta)\pi(T|X, \theta). \tag{3.43}$$

By substituting the equation (3.42) in (3.43), we obtain:

$$\pi(X, T|\theta) = \prod_{n=0}^{N-1} \frac{\theta_{v_n}\rho_{v_n}(x_n)}{h(x_n)} h(x_n)e^{-dt_n h(x_n)}$$

$$\pi(X, T|\theta) = \prod_{n=0}^{N-1} \theta_{v_n}\rho_{v_n}(x_n)e^{-dt_n h(x_n)}. \tag{3.44}$$

If we assume a Gamma distribution for each parameter

$$\pi(\theta_i) = \frac{b_i^{a_i}}{\Gamma(a_i)}\theta_i^{a_i-1}e^{-b_i\theta_i}, \tag{3.45}$$

The following distribution can be evaluated using equation (3.45) and equation (3.44):

$$\pi(\theta_i|X, T) \propto \pi(\theta_i)\pi(X, T|\theta)$$

$$\propto \theta_i^{a_i+K_i-1} e^{-(b_i+\sum_{n=0}^{N-1} dt_n \rho_i(x_n))\theta_i}, \tag{3.46}$$

which implies that the parameters are Gamma distributed with rate $b_i+\sum_{n=0}^{N-1} dt_n \rho_i(x_n)$ and shape $a_i + K_i$, where K_i represents the number of times that the i reaction is observed. This results in an exact Gibbs sampler for the unknown parameter.

However, uniformsation does not work for many interesting systems with infinite state spaces. Alternatively, there is a method based on introducing a random truncation which is used to provide an unbiased estimator of the likelihood. The estimated likelihood then uses pseudo-marginal MCMC (Andrieu and Roberts, 2009) and (Beaumont, 2003). This results in two methods relying on random truncation namely Metropolis-Hastings, which targets the marginal likelihood, and an auxiliary variable Gibbs scheme.

3.8.3 Russian Roulette

Let us suppose that we want to estimate the following infinite sum:

$$f = \sum_{N=0}^{\infty} f_N,$$

which can be approximated by picking one term f_k, where k can be chosen from a chosen discrete probability $\pi_0, \pi_1, \cdots$. Then, the estimator of f is $\hat{f} = \frac{f_k}{\pi_k}$. The estimator is unbiased because $\hat{f}$ has the expectation $\mathbb{E}[\hat{f}] = \sum_{N=0}^{\infty} \frac{f_N}{\pi_N}\pi_N = f$. This method is importance sampling, and the main problem with this method is that variance of the estimator $\hat{f}$ can be large or infinite (Georgoulas et al., 2017), (Lyne et al., 2015).

The variance estimator can be reduced through Russian Roulette sampling procedure. The method relies on approximating f with a partial sum where each term j can be chosen randomly. The probability of stopping the sum (as described in section 3.2.2 in (Georgoulas et al., 2017)) is $1 - q_j$, else, we continue the process to form the partial sum $\hat{f} = \sum_{N=0}^{j} \frac{f_N}{\pi_N}$, the term $\pi_N = \prod_{j=1}^{N-1} q_j$. This process results in an unbiased estimator of the full sum (where the proof the unbiased estimator can be found in (Lyne et al., 2015)). This method is known as Russian Roulette, For further details (see (Lyne et al., 2015), (Georgoulas et al., 2017)).

3.8.4 Expanding the Likelihood

The Russian Roulette method requires expressing the likelihood as an infinite sum. Hence, the likelihood for a Markov process can be formalised as a set of an infinite series and then the space of processes path can be decomposed into "a nested sum over a subspace of trajectories which differ by at most N from the observations" (Georgoulas et al., 2017).

As it is explained in (section 3.2.1 in (Georgoulas et al., 2017)), the likelihood for a single observation (x_t, t) at one dimensional process can be written as:

$$\pi(x_t|x_0,\theta) = \sum_{N=0}^{\infty} \pi_t(x_t, \max(x_{0:t} - x^v) = N|x_0,\theta), \tag{3.47}$$

supposing that at time 0, the initial state is known as x_0 and assume that $x^v = \max(x_0, x_t)$. The possible states can be visited in the interval $[0, t]$ is denoted by $x_{0:t}$ and the maximum value of the process is represented by $\max(x_{0:t}) = N$. The variable to sum over N is assumed to be the maximum state's value achieved in that time. The constraint assumption for $x_{0:t}$ does not mean that state space is defined. Therefore, it will be not considered as a solution of the CME.

If we have:

$$f_t^N(x', x) = \pi_t(x', \max(x_{0:t} - x^v) \leq N|x,\theta),$$

every term in the equation (3.47) decomposes as:

$$\pi_t^N(x_t, x_0) = \pi_t(x_t, \max(x_{0:t} - x^v) = N|x_0,\theta) = f_t^N(x_t, x_0) - f_t^{N-1}(x_t, x_0), \tag{3.48}$$

where each term in equation (3.48) can be considered as the transient probability for a finite Markov process. For each finite state space, a generator matrix can be defined and then a transient probability can be computed using (2.7) Georgoulas et al. (2017).

3.8.5 Modified Gibbs Sampler

It is only possible to sample trajectories directly when there is a bounded state space. This is because a finite number of states is required in the uniformisation process. An alternative approach is proposed in (Georgoulas et al., 2017) to avoid this issue which works by sampling a truncation point through the Russian Roulette method.

This truncation defines a finite state space and thereafter sample a parameter and path as described in 3.8.2. An auxiliary variable (truncation point z^*) is proposed in the sampler.

Therefore, we are sampling from the conditional posterior over the chosen truncation z^* e.g $\pi(X, T|\theta', Y, z^*)$ instated of the correct conditional posterior $\pi(X, T|\theta', Y)$.

Hence, an acceptance ratio should be introduced as not each drawn path and parameter sample are accepted. The summary of the modified Gibbs sampler (as described in (Georgoulas et al., 2017)) is given in Algorithm 9.

Algorithm 9 Auxiliary variable Gibbs sampler for finite state Markov

1: Draw a parameter θ' from $\pi(\theta|X, T)$ according to equation (3.46).

2: Draw a sample $X^*|\theta', Y$:

   (a) Select a truncation point z^* through Russian Roulette method.

   (b) Draw a trajectory (X^*, T^*) from $\pi(X, T|\theta', Y, z^*)$ according to section (3.8.1).

3: The acceptance ratio is calculated as a follow:

   (a) Calculate $\pi^{i+1}(X^*|\theta', Y)$ and $\pi^i(X^*|\theta', Y)$, the conditional posterior of the proposed trajectory under the new and old truncations.

   (b) Calculate $\pi^{i+1}(X_i|\theta', Y)$ and $\pi^i(X_i|\theta', Y)$, the conditional posterior of the old trajectory under the proposed and old truncations.

   (c) Setting $a = \frac{\pi^{i+1}(X^*|\theta',Y)\pi^{i+1}(X_i|\theta',Y)}{\pi^i(X^*|\theta',Y)\pi^i(X_i|\theta',Y)}$

4: Accept the new sample with the acceptance probability a and then set $\theta' = \theta_{i+1}$ and $(X^*, T^*) = (X_{i+1}, T_{i+1})$, otherwise set $\theta_i = \theta_{i+1}$ and $(X_i, T_i) = (X_{i+1}, T_{i+1})$.

The probabilities that are required to compute the acceptance ratio a can be evaluated through the forward-backwards algorithm, the outline of this algorithm is given in the section A.1 of Appendix A.

3.9 Related Work

The main difficulty in performing Bayesian inference in biological systems is how to formalise the likelihood, which relies on the calculation of the transition probability and therefore the solution of the CME. As we mentioned in section 3.9, it is not possible to solve CME analytically for most biological systems, which is required to

compute the likelihood in Bayesian inference. Several approaches have been proposed to solve the inference problem, such as MCMC and ABC. These approaches are based on statistical simulations. A different approximate inference method has been proposed based on variational inference (Murphy, 2012) and (Opper and Sanguinetti, 2008). The construction of a variational approximation for CTMC, in particular, has been illustrated by Opper and Sanguinetti (2008).

The approach relies on picking an approximation from a family of distributions $q(x) \in Q$. It attempts to make the approximation as close as possible to the probability distribution for the system $\pi(x)$.

The main advantage of using the variational inference for Markov processes is speed as it is faster than MCMC algorithms. Variational inference can be performed with large data with different forms and scenarios, while MCMC requires intensive computational time even for small data sets. Therefore, there is a trade- off between accuracy and speed. MCMC provides an exact approximation, variational inference is suggested when speed is important.

Assume that $\pi(x)$ is the true posterior distribution we are interested in but this distribution is intractable. Assume further that $\pi(x)$ is approximated by $q(x)$, which can be chosen from the tractable distribution family. The distribution q has free parameters which can be optimised

The posterior then can be approximated by minimising the KL divergence between two Markov processes, which are defined through their probabilities path $\pi(x)$ and $q(x)$ as:

$$KL(q(x), \pi(x)) = \sum_x q(x) \log \frac{q(x)}{\pi(x)},$$

where π represents the posterior of the Markov process and q represents the approximate distribution.

One of the common choices of form of variational inference is the mean field approximation, which assumes that the posterior approximation has the form:

$$q(x) = \prod_{j=1} q_j(x_j).$$

Then, the aim is to find the member of that family which minimises the KL divergence to the true distribution,

$$q^*(x) = \min_{q_1,\cdots,q_Q} KL(q(x)|\pi(x)),$$

where each marginal distribution q_j is optimised over its parameters. Then, the true distribution is approximated with the optimised member of $q^*(x)$.

More details about this method, how to perform Bayesian inference and applications can be found in (Opper and Saad, 2001), (Opper and Sanguinetti, 2008), (Cohn et al., 2009), (Murphy, 2012).

Pseudo-marginal approaches which rely on random truncation are recently developed methods in statistics (Filippone and Engler, 2015), (Lyne et al., 2015). Georgoulas et al. (2017) proposed a novel random truncation method for unbounded state spaces that ensure unbiasedness of the result as is detailed in section 3.8.

In addition, (Boys et al., 2008) evaluates various MCMC methods in different data scenarios and applied to the Lotka- Volterra system. The paper shows how inference can be made given a complete, regular and partially observed data sets. Based on the investigation (Boys et al., 2008), the partially observed data set case was more challenging compared to other scenarios.

3.10 Summary

In this chapter, an overview of Monte Carlo methods which use a set of samples to approximate the target distribution is provided. We began with rejection sampling and importance sampling. These methods can be inefficient in a complex model because it requires a careful selection of suitable proposal distribution. In contrast, the SMC and MCMC can be more efficient to deal with complex and high dimensional problems. We have discussed these methods as a means of estimating the generally complicated target distribution.

In addition, these algorithms are described with reference to Bayesian inference in Markovian models. MCMC, and in particular the MH algorithm, is presented with an efficient adaptive scheme, based on the evolution of an unknown covariance matrix through the iterations. The main obstacle to the direct application of the standard MH algorithm is the performance of this MH algorithm mainly relies on the evaluation of the acceptance probability, which itself requires an explicit evaluation of the likelihood.

The role of PMMH was detailed in carrying out the Bayesian inference when the

likelihood is estimated with SMC. The challenge is to formulate this approach with model involve latent variable which can limit it to their computational complexity. A recent effort that considered the inference in a model with intractable likelihood results in an exact inference method known as pseudo-marginal Gibbs approaches which reducing the computational cost investigated. The main advantage of Gibbs sampler is that there is no need to select a proposal distribution while in PMMH, the efficiency of this method is relying on the choice the proposal distribution.

Chapter 4

Approximate Bayesian Computation

4.1 Introduction

One of our main goals is to enable Bayesian parameter inference for CTMC models. The main problem in performing such inference lies in the lack of a closed form expression for the likelihood $\pi(y|\theta)$, because it depends on the transient probability of the latent states of the CTMC. Potentially, the state space of the CTMC model may be very large, and working out a transient probability of individual state will be infeasible if not impossible. A range of approximate inference methods exists that does not require an explicit evaluation of the likelihood, and employs simulation from the likelihood instead. We consider a family of the likelihood-free inference methods called Approximate Bayesian Computation (ABC) methods. In this section, we review state of the art in ABC.

In an attempt to provide the reader with a full understanding of the ABC, we begin with a brief overview of the methodology that was developed for a popular class of Bayesian inference algorithms and that demonstrated how the challenging problem could be solved.

4.2 Review of Likelihood-Free Methods

The likelihood free approach is a common method that can be used within the Bayesian context to deal with an intractable model. A stochastic model may be used to model a dynamical system; this implies that if we are able to generate data from this model, given the model parameter has some value $\theta \in \Theta$, then the output

of such a system is given by:

$$y^* \sim \pi(\cdot|\theta),$$

where $y^* \in \mathcal{X}$ is the state of the model and the likelihood for this model is $\pi(\cdot|\theta)$. The main aim is to infer the unknown model parameter within a Bayesian perspective where the posterior can provide all the information about the parameter by using the following joint distribution:

$$\pi(\theta, y^*) \propto \pi(y^*|\theta)\pi(\theta). \tag{4.1}$$

A possible method that can be used when the likelihood is not available is Approximate Bayesian Computation. The basic idea of this likelihood-free method is that the target posterior can be obtained through an augmented model. An auxiliary variable is introduced in the model; thus, the posterior $\pi(\theta|y) \propto \pi(y|\theta)\pi(\theta)$ can be adopted within this augmentation to be:

$$\pi(\theta, y^*|y) \propto \pi(y|y^*, \theta)\pi(y^*|\theta)\pi(\theta), \tag{4.2}$$

where auxiliary $y^* \in \mathcal{X}$ represents ith simulated dataset among simulated data from the model likelihood $y^* \sim \pi(y^*|\theta)$, where y^* is on the same state space as y. The function $\pi(y|y^*, \theta)$ is defined to weight the intractable posterior $\pi(\theta|y)$, where the high value of the weight in regions indicates the similarity between y^* and y. In the case that $y^* = y$, the function $\pi(y|y^*, \theta)$ is considered to be constant with respect to some parameter θ, thus, $\pi(y|y^*, \theta) = c$. On the other hand, the low values of weight in the regions mean that the datasets y^* and y are dissimilar. The main interest lies in the marginal augmented posterior:

$$\pi_{ABC}(\theta|y) \propto \int_{\mathcal{X}} \pi(y|y^*, \theta)\pi(y^*|\theta)\pi(\theta)dy^*. \tag{4.3}$$

The marginalisation can be performed by integrating out of the auxiliary simulated data y^*. The resulting posterior $\pi_{ABC}(\theta|y)$ is considered as an approximation to the exact posterior $\pi(\theta|y)$ and can be illustrated by the following:

$$\pi_{ABC}(\theta|y) \propto \int_{\mathcal{X}} \pi(y|y^*,\theta)\pi(y^*|\theta)\pi(\theta)dy^* \tag{4.4}$$

$$= \pi(\theta)\mathbb{E}_{(y^*|\theta)}[\pi(y|y^*,\theta)] \tag{4.5}$$

$$\approx \frac{\pi(\theta)}{N}\sum_{i=1}^{N}\pi(y|y_i^*,\theta). \tag{4.6}$$

Here, $y_i^* \sim \pi(y^*|\theta)$ represents N simulated datasets from the model. The approximated expectation in (4.6) was initially introduced by Marjoram et al. (2003); then, it was studied and used by Toni et al. (2009).

Typically, there are two main approaches present in the literature to simulate the posterior $\pi_{ABC}(\theta|y)$ underlying a likelihood-free concept. The first approach is aiming to simulate from the augmented model $\pi(\theta, y^*|y)$, where the joint samples (y^*, θ) are obtained before marginalisation of the posterior. The other approach works on marginalising space directly $\pi_{ABC}(\theta|y^*)$ through Monte Carlo integration, as shown in (4.6). Throughout this thesis, this second approach, in particular, will be used.

The ABC approaches have been developed and gained popularity rapidly since they were first introduced in population genetics by Pritchard et al. (1999) and Tavaré et al. (1997). Then, the research focused on the development of ABC within the likelihood-free method. The likelihood-free approaches were introduced in the Bayesian literature by Tavaré et al. (1997). A recent development was an adaption of the ABC algorithm, which was used to handle an inference issue in statistical genetics. Then, a sequence of developments occurred within a basic likelihood-free setting, a significant development in ABC methodology replacing the direct comparison of the datasets in weighting function by tolerance level; this introduced the concept of approximation and the sample obtained was considered to be from the approximated posterior (Beaumont et al., 2002), (Marjoram et al., 2003),(Marin et al., 2012).

Further developments were introduced by Wilkinson (2013) who showed that ABC can provide an exact result under a model error assumption; more details of this methods can be found in (Wilkinson, 2013). A major effective development in the ABC methodology was made by Sisson et al. (2007a) and Peters et al. (2012). They proposed a novel approach using SMC within the ABC method. Several algorithms were developed to obtain a more efficient scheme for the ABC method, compared to the standard rejection technique (Del Moral et al., 2006). There have also been various developments focusing on improving the efficiency of the ABC SMC, taking into consideration the setting of the tolerance level and the choice of the type of

perturbation kernel (Filippi et al., 2013)(Del Moral et al., 2012), (Sisson et al., 2007a). An illustration of the SMC ABC and its recent development, as well as the design of the algorithm, will be given in detail in this chapter.

The concept of the ABC approach is based on proposing a new candidate parameter θ'; given this candidate's value, a new dataset will be simulated from the model $y^* \sim \pi(y^*|\theta')$. If the simulated dataset is equal or "close enough" to the observed one $y^* \approx y$, then the candidate parameter will be accepted into the posterior sample $\pi(\theta|y)$. If the equality condition is satisfied, the proposed parameter is considered to be drawn from the exact posterior distribution; when the datasets are "close enough" but not equal, the proposed parameter is drawn from the approximated posterior distribution. To be able to carry out this inference procedure, the following essential ingredients are required:

1. Ability to simulate a data set: the key requirement for applying ABC or the likelihood- free approach is the ability to generate a synthetic data y^* from the model $\pi(y^*|\theta')$. The essential term in Bayesian inference is computing the likelihood $\pi(y|\theta')$ which is replaced by an approximation of the closeness of the simulated data set to the observed dataset. Hence, the main component for the ABC method is to be able to simulate from the model.

2. Distance metrics: after having obtained a synthetic data set from the model, the ABC then makes use of the distance metrics function to measure the distance between the simulated and observed data. A popular choice of the distance is the Euclidean distance, which is defined as follows:

$$\rho(y, y^*) = \sqrt{\sum_{i=1}^{N}(y_i - y_i^*)^2}.$$

   The metric $\rho(\cdot, \cdot)$ must satisfy the standard properties:

   (a) $\forall(y^*, y), \rho(y^*, y) \geq 0$

   (b) $\forall(y^*, y), \rho(y^*, y) = 0 \quad \text{iff} \quad y^* = y$

   (c) $\forall(y^*, y), \rho(y^*, y) = \rho(y, y^*)$

   (d) $\forall(y, y^*, z), \rho(y^*, z) \leq \rho(y^*, y) + \rho(y, z).$

   If $\rho(y, y^*) = 0$, this implies the simulated data exactly matches the observed data, and the corresponding sample θ' is coming from the exact posterior distribution.

3. Weighting function: having specified the distance function and measured the difference between the observed and synthetic datasets, the following step is to define the weighting function or indicator function based on $\pi(y|y_i^*, \theta)$. Numerous definitions have been proposed using a direct comparison of both datasets as:

$$\pi(y|y_i^*, \theta) \propto \begin{cases} 1 & \text{if } y^* = y \\ 0 & \text{if } y^* \neq y \end{cases} \tag{4.7}$$

However, an exact comparison may be impossible in most cases as it requires a huge computational cost to obtain at least one acceptable sample. Therefore, in order to obtain a practical sampler, the above equation is redefined by using of the distance function and introducing a tolerance level ϵ, so that it can be expressed as follows:

$$\pi(y|y_i^*, \theta) \propto \begin{cases} 1 & \text{if } \rho(y^*, y) \leq \epsilon \\ 0 & \text{if } \rho(y^*, y) > \epsilon \end{cases} \tag{4.8}$$

4. Tolerance schedule: the choice of the tolerance schedule plays an important role in the efficiency of the ABC. Using the distance metric, these approximations tend to the exact posterior as ϵ goes to zero;

$$\lim_{\epsilon \to 0} \pi(\theta|\rho(y^*, y) < \epsilon) = \pi(\theta|y).$$

A small ϵ contributes to increasing the accuracy of the approximation but the computational cost will increase. In contrast, large values of ϵ lead to imprecise approximation. The resulting approximated posterior will converge to the correct one as:

$$\text{if} \quad \epsilon \to 0, \quad \pi_{ABC}(\theta|y) \to \pi(\theta|y)$$

while

$$\text{if} \quad \epsilon \to \infty, \quad \pi_{ABC}(\theta|y) \to \pi(\theta).$$

It is clear from the above that the accuracy of the resulting approximated posterior and its correct convergence will be based on the value of ϵ. The deterministic tolerance can be employed with rejection algorithm and the Sequential

Approximate Bayesian Computation (ABC SMC) algorithm (described later in this chapter). We can also use a deterministic and fixed sequence of tolerances ϵ_t. However, an alternative effective choice of the ϵ has been used where the tolerance is adaptive through the iterations. This adaptive tolerance method will be discussed in more detail in section 4.5.1.

4.3 ABC Rejection

The ABC rejection process is based on two steps: first, a new value is proposed for the model parameter θ' from the prior distribution; then data y^* is simulated from the model given the sampled parameter θ', and compared to the observed y. The proposed value can be accepted if the equality between the simulated and the observed data is satisfied, or otherwise rejected. Finally, the accepted candidate value is assumed to be a part of the exact posterior distribution. This procedure is known as the likelihood free rejection algorithm or Perfect rejection sampling algorithm and presented in the following terms:

Algorithm 10 Perfect rejection sampling algorithm

1: Propose a parameter θ' from the prior: $\theta' \sim \pi(\theta)$.
2: Simulate data y^* from the model using the sampled parameter θ', $y^* \sim \pi(\cdot|\theta')$.
3: If $y^* = y$, then a sampled parameter is accepted, or otherwise rejected.

This algorithm is exact. To illustrate this, let us assume that there is a joint distribution for the accepted simulated data y^* and proposed parameter θ', denoted $\pi_{ABC}(\theta', y^*)$, so that:

$$\pi_{ABC}(\theta', y^*) = \pi(y^*|\theta')\pi(\theta')\mathbb{1}_y(y^*),$$

where the indicator function $\mathbb{1}$ equivalent to the weighting function $\pi(y|y^*, \theta)$ in the equation (4.7).

In the case of the equality between the simulated and the actual data being satisfied, the indicator function is $\mathbb{1}_y(y^*) = 1$. As we are interested in the marginal parameter space, the simulated data y^* is marginalised as:

$$\pi_{LF}(\theta') = \int_{\mathcal{X}} \pi(y^*|\theta')\pi(\theta')\mathbb{1}_y(y^*)dy^*$$

$$= \pi(y|\theta')\pi(\theta') \propto \pi(\theta'|y^*),$$

and this is a proof that all the accepted candidate θ' are sampled from the exact posterior distribution.

The likelihood free rejection sampling algorithm can be performed only if equality between observed data and simulated data is obtained with a certain probability. However, finding simulated data that exactly matches the observed data with non-zero probability is a rare case and can occur only if the data are discrete variables. When data is continuous, it is indeed very challenging to obtain non-zero probability equality. The main disadvantage of this algorithm is that the acceptance rate will be small and a large number of simulations from the model is needed to obtain a reasonable sample size which can represent the posterior distribution.

Due to the restrictions of the perfect rejection sampling, another ABC approach was proposed to mitigate the problems associated with low acceptance rates by modifying the acceptance condition. The algorithm is based on introducing a distance function that measures the similarity between simulated and observed data, and the equality in the previous algorithm is replaced by a certain tolerance level $\epsilon > 0$ to measure the closeness between the datasets. If the distance is below the defined tolerance level, the proposed parameter θ' will be accepted as a sample from the approximate posterior distribution. Otherwise, the sample proposed from the prior θ' will be discarded. A summary of this algorithm is detailed below:

Algorithm 11 ABC rejection sampling algorithm

1: Propose a parameter θ' from the prior: $\theta' \sim \pi(\theta)$
2: Simulate data y^* from the model using the sampled parameter θ', $y^* \sim \pi(\cdot|\theta')$.
3: If $\rho(y^*, y) < \epsilon$; then the sampled parameter is accepted, or otherwise rejected.

The accepted proposed parameter θ' and simulated data can form the joint distribution $\pi_{ABC}(\theta', y^*|y)$ as:

$$\pi_{ABC}(\theta', y^*|y) = \pi(\theta', y^*|\rho(y^*, y) < \epsilon)\pi(\theta')$$

$$\propto \pi(y^*|\theta')\pi(\theta')\mathbb{1}_{\rho(y^*,y)<\epsilon},$$

and here the indicator function corresponds to the weighting function that was defined previously in equation (4.8). Then, the resulting approximate marginal

posterior distribution after the simulated data are marginalised out is given by:

$$\pi_{ABC}(\theta') = \int_{\mathcal{X}} \pi_{ABC}(\theta', y^*|y)dy^*.$$

In this algorithm, the simulated data are marginalised out, which implies that these simulated data do not need to be kept in order to obtain a sample from the parameter posterior distribution. The performance of the ABC scheme depends on the setting of the tolerance ϵ: setting them ϵ to be small improves the approximation of the posterior but in the meantime results in expensive computational time. However, a large tolerance setting results in a poor approximation of the posterior as most proposed values from the prior are accepted. We, therefore, conclude that a suitable setting for the tolerance is such that can balance between the accuracy and the computational time (McKinley et al., 2009). Wilkinson (2013) suggests a method to improve the inference in light of considering an error ϵ, which is either an error on data y or an error in the model $\pi(\cdot|\theta)$. Let us assume that the error has a density $\epsilon \sim \pi_\epsilon(\cdot)$. Then, it can be possible to describe the posterior in terms of errors, with the ABC rejection algorithm becoming:

Algorithm 12 Generalised ABC (GABC)

1: Draw θ' from the prior $\pi(\theta)$
2: Simulate the data y^* from the model $\pi(\cdot|\theta')$
3: Accept θ to the posterior sample with probability $\pi_\epsilon\big(\frac{y-y^*}{c}\big)$.

As proposed in (Wilkinson, 2013), the constant c is chosen to ensure that the term $\frac{\pi_\epsilon(y-y^*)}{c}$ defines a probability. When $\epsilon = 0$, we have $c = \pi_\epsilon(0)$ which maximises the acceptance rate of the algorithm. For more information on this method, see (Wilkinson, 2013).

However, Toni et al. (2009) states that the possible drawback of using the ABC rejection method is that it still suffers from a low acceptance rate and poor accuracy, especially when the prior and the posterior have a significantly different form. Moreover, Wilkinson (2013) mentions that sampling from the prior repeatedly can lead to an inefficient rejection algorithm.

4.4 Sequential Approximate Bayesian Computation Approach

ABC approaches can be formulated based on different sampling schemes, such as rejection sampling, and can also be based on SMC. SMC is considered to be the most efficient scheme as it can provide an accurate and consistent approximation to the posterior. Particular attention, however, must be paid when we are designing the ABC SMC scheme. In this section, we discuss different settings of the ABC SMC in order to obtain good performance from this algorithm.

The ABC in this section will be performed within the SMC perspective. The underlying concept for the ABC SMC approach is that instead of performing an ABC algorithm with a specific tolerance value that directly affects the performance of the algorithm, a sequence of decreasing tolerances $\{\epsilon_m\}$will be used. In addition, in order to obtain a more efficient approximation, the proposal distribution can be constructed based on the current particle from which a sample can be drawn rather than sampling from the prior.

The SMC method relies on a sequence of target distributions. The ABC SMC approach also makes use of a series of distributions with a sequence of tolerances. The sequence of the joint distributions can be defined by:

$$\pi_{ABC}(\theta', y^*|y, \epsilon_m) \propto \pi(y^*|\theta')\pi(\theta')\mathbb{1}_{\rho(y^*,y)<\epsilon_m},$$

where $m = 1, \cdots, M$, then the marginal posterior can be defined as follows:

$$\pi_{ABC}(\theta'|y, \epsilon_m) \propto \int_{\mathcal{X}} \pi(y^*|\theta')\pi(\theta')\mathbb{1}_{\rho(y^*,y)<\epsilon_m} dy^*.$$

The method works by setting a sequence of tolerances $\epsilon_1 > \epsilon_2 > \cdots > \epsilon_m$ and aiming to draw samples from a more efficient distribution rather than the prior. The method is similar to the SMC algorithm, which is described in Chapter 3 with a small modification for the ABC framework. At the initial stage $m = 0$, a number of particles $\{\theta^1, \cdots, \theta^N\}$ are sampled from the prior distribution $\pi(\theta')$. These sampled particles are propagated via a series of intermediate distributions $\pi(\theta|\rho(y^*, y) < \epsilon_m)$.

At stage m, a sample θ' is drawn from the set of weighted particles $\{\theta_i^{m-1}, w_i^{m-1}\}$ from the previous population (stage $m-1$). The set of particles is propagated through a kernel to obtain $\theta^{**} \sim K_m(\cdot|\theta')$. Then, data is simulated from the model,

given the parameter θ^{**}, such that: $y^* \sim \pi(\cdot|\theta^{**})$. The essential step is the acceptance criterion based on the distance measure, where the algorithm will keep simulating from the model until this criterion is satisfied, $\rho(y, y^*) < \epsilon_m$. This process is targeting an approximation to the posterior distribution.

Performing this process on the marginal space, we obtain the following weight:

$$w_m \propto \frac{\int_{\mathcal{X}} \pi(y^*|\theta_m)\pi(\theta_m)\mathbb{1}_{\rho(y^*,y)<\epsilon_m} dy^*}{\int_{\theta_{m-1}} \pi(\theta_{m-1}|y, \epsilon_{m-1})K_m(\theta_m|\theta_{m-1})d\theta_{m-1}},$$

which can be approximated using the Monte Carlo method. The numerator can be approximated by a single sample from the likelihood and the denominator uses the previous set of particles which was approximately distributed as $\pi(\theta_{m-1}|y, \epsilon_{m-1})$; based on this, we obtain w_m as:

$$w_m \propto \frac{\pi(\theta_m)}{\sum_{j=1}^{N} w_{m-1}^j K_m(\theta_m|\theta_{m-1}^j)}.$$

Since the simulation is repeated until the acceptance criteria, according to the distance measure, is satisfied, the indicator function is not needed in the weight in the equation. The procedure of ABC SMC is summarised in the following algorithm:

Algorithm 13 ABC- SMC algorithm

1: Initialise the sequence of tolerances as: $\epsilon_1 > \epsilon_2 > \cdots > \epsilon_m$.
2: For $m = 1, \cdots, M$ and for $i = 1, \cdots, N$, do
3: At stage $m = 1$, draw θ^{**} from the prior $\pi_0(\cdot)$
4: If $m > 1$, draw a sample θ' from the set of weighted particles $\{\theta_i^{m-1}, w_i^{m-1}\}$, then perturb the particles through the kernel to obtain $\theta^{**} \sim K_m(\cdot|\theta')$
5: If $\pi_0(\theta^{**}) = 0$, return to 4.
6: Simulate a dataset from the model, given a proposed parameter, $y^* \sim \pi(\cdot|\theta^{**})$
7: If $\rho(y, y^*) > \epsilon_m$, repeat the previous steps, otherwise, if $\rho(y, y^*) < \epsilon_m$, set: $\theta_m^i = \theta^{**}$
8: Calculate the weight:

$$w_m'^i = \begin{cases} 1 & \text{if } m = 1 \\ \frac{\pi(\theta_m^i)}{\sum_{j=1}^{N} w_{m-1}^j K_m(\theta_m^i|\theta_{m-1}^i)} & \text{if } m > 1 \end{cases}$$

9: Normalise the weights as: $w_m^i = \frac{w_m'^i}{\sum_{j=1}^{N} w_m'^j}$.

The performance of ABC SMC mainly relies on the choice of tolerance levels ϵ_m and

the setting of the perturbation kernel $K_m(\cdot|\cdot)$. It is therefore important to consider a choice which maximises the accuracy of the approximation and minimises the computational time.

4.5 ABC SMC Algorithm Tuning

The efficiency of the ABC SMC algorithm in terms of its quality of estimation and computational cost relies on the algorithm parameters, such as the selection of the tolerance schedule and perturbation kernel. In this section, we shall describe how the algorithm can be carefully tuned to obtain satisfactory estimation in a reasonable time.

4.5.1 Tolerance Level

The choice of an appropriate tolerance is considered to be a critical part in setting ABC. In a basic ABC rejection algorithm, setting the tolerance value to be $\epsilon = 0$ implies that the real data are equal to the simulated data which is impossible in practice. Thus, obtaining a large sample size from the algorithm that can represent the approximate posterior with a small value of tolerance is computationally expensive. In contrast, setting a large value of ϵ indicates that most of the proposed values from the prior will be accepted, resulting in an inaccurate approximation of the posterior but in the meantime yielding a faster algorithm. The Euclidian function can measure the discrepancy between simulated data and real data.

In an SMC algorithm, the tolerance schedule can be tuned manually, relying on the available knowledge about the model. Defining the sequence of the thresholds $\epsilon_1, \cdots, \epsilon_m$ is a challenging task. Slowly reducing ϵ leads to a slow convergence to the posterior distribution and thus is computationally intensive (Wilkinson, 2010b), while a quick decrease of the tolerance level is computationally convenient but results in a poor approximation. Therefore, the sequence of thresholds $\epsilon_1, \cdots, \epsilon_m$ should be chosen very carefully. In ABC SMC, rather than using deterministic tolerance values, the tolerance level can be set to decrease through algorithmic steps adaptively. This ensures that the algorithm can explore the state space of the parameters sufficiently.

The adaptive tolerance level, based on the α^{th} quantile scheme of the distance between the real data y and the simulated data y^*, produced at the previous algorithmic step (Del Moral et al., 2012), (Drovandi and Pettitt, 2011b) and (Drovandi and

Pettitt, 2011a) can be a convenient approach for selecting a suitable level and can be applied to any ABC SMC scheme (Lenormand et al., 2013).

The adaptive rate assumes the value $0 < \alpha^{th} < 1$; yet the choice of quantile value is a difficult task, as it has an influence on exploring the state space and thus the accuracy and the computational time. To ensure that we obtain a consistent posterior approximation in a reasonable time, different choices of quantile setting will be tested in our first case study.

4.5.2 Perturbation Kernel

The perturbation kernel is another essential part of the algorithm parameters which influences exploring the parameter state space. The particles produced from the intermediate distribution through the ABC SMC steps denoted $\{\theta^i_{m-1}\}$, need to be perturbed through the perturbation kernel $K_m(\cdot|\cdot)$. In the literature, several kernels were proposed where the common choice is a Gaussian or uniform distribution (Sisson et al., 2007a), (Toni et al., 2009), (Liepe et al., 2010). In this thesis, we make use of a component-wise random walk normal kernel to construct the perturbation kernel that produces uncorrelated particles. The perturbation kernel takes the form:

$$K_m(\cdot|\cdot) \sim N(\theta^i_{m-1}, \mathbf{\Sigma})$$

where the diagonal elements of covariance matrix ($\mathbf{\Sigma}$) are the empirical variances of the current population which can be estimated though the iterations (Beaumont et al., 2009). The variance is recommended to be adaptive through the steps (Beaumont et al., 2009), (Jasra et al., 2011). The empirical variance of the parameter sample is:

$$var(\theta) = \mathbb{E}[\theta^2_m] - (\mathbb{E}[\theta_m])^2, \tag{4.9}$$

and the expectation can be defined as:

$$\mathbb{E}[\theta_m] = \sum_i w^i_m \theta^i_m \tag{4.10}$$

The variance after substituting the equation 4.10 into equation 4.9 can be written as:

$$var(\theta) = \sum_i (w_m^i \theta_m^i)^2 - (\sum_i w_m^i \theta_m^i)^2.$$

For more details about the choice of perturbation kernel see (Filippi et al., 2013).

4.5.3 Number of Particles

The approximation of the obtained posterior distribution from the ABC SMC algorithm can be improved by increasing the number of particles. It is therefore important to have enough particles to provide a sample that can represent the posterior well. However, a large number of particles can result in huge computational time until the target posterior is obtained. There is no standard method for selecting the number of particles but it can be measured by the ESS. If the produced sample does not represent the approximate posterior well, the number of particles should be increased.

4.5.4 Target Tolerance and the Number of Stages

The number of stages is an essential algorithmic parameter that should be chosen by the user. However, it is very difficult to determine if the predefined number of stages is enough to achieve a small tolerance value. In some cases, the number of stages is set to be large, hence; the algorithm keeps simulating from the model without further reduction of the approximation error. However, this requires an expensive computational time. Therefore, the number of stages will be tuned automatically according to predefined target tolerance.

The trade-off is between the tolerance level and the computational time. If the final tolerance value tends to be high, the approximated posterior can be far from the target distribution. In contrast, a small tolerance value results in expensive computational time that might not be required to obtain a reasonable estimation of the posterior. To save the computational effort while performing the ABC SMC, it is useful to predefine the target tolerance. The choice of the target tolerance can be made through several simulations from the model, where the simulated data generated from the model must have the same size as the real data, and then the distance between these datasets can be measured by Euclidian distance. Finally, some low percentile achieved from these trial runs can be used as the criterion to decide whether this is the right stage to stop the algorithm or not. The steps are:

1. Several data sets y_i^* are simulated from the model with respect to the fixed arbitrary parameter value θ, where the number of data points simulated from the model should be equal to the number of real data points T.

2. The distance between simulated data y_i^* and y_j^* will be calculated as:

$$\rho(y_i^*, y_j^*), \quad i \neq j$$

3. The target tolerance can be defined as some low percentile (e.g the 1^{st}) over a set of distances, which can be obtained from the previous step can be set as the criterion to decide if the number of stages is sufficient or not. If the final tolerance value is equal to or less than target tolerance, we can accept the number of stages to be enough to approximate the posterior; otherwise, the number of stages can be automatically increased to reach the minimal target tolerance.

4.5.5 Effective Sample Size

ABC SMC can provide a reasonable estimation of model parameters. However, a critical issue that can arise when performing this sequential sampling method is weight degeneracy after a few iterations, but it can be overcome by adding a resampling step to the algorithm. In this way, the particles with negligible weight are eliminated and those with a high weight are replicated.

However, performing resampling at each step means that only particles with a high weight are used and diversity in the particle population can thus be lost and can add computational cost to the algorithm. Therefore, it would be more efficient to perform a resampling step, only when needed, to avoid this problem. In this case, the Effective Sample Size (ESS) of the weighted particle set is needed to decide whether a resampling step is required or not. When the sample size of the weighted samples falls below the defined threshold, a resampling step should be applied. ESS can be defined as:

$$ESS = \frac{1}{\sum_{i=1}^{N} w_i'^2},$$

where the wights w_i' are unnormalised. We can define $w_i = \frac{w_i'}{\sum_{i=1}^{N} w_i'}$, then:

$$ESS = \frac{1}{\sum_{i=1}^{N} w_i^2} = \frac{(\sum_{i=1}^{N} w_i')^2}{\sum_{i=1}^{N} w_i'^2},$$

where ESS takes values between 1 and N (Robert and Casella, 2010).

4.6 Summary

In this chapter, we discussed the family of ABC algorithms concerning parameter inference for a complex stochastic model with intractable likelihood. The ABC method has proven its ability as a powerful tool for applying inference in a wide variety of intractable model situations. We began by introducing the basic technique of a likelihood-free method and then considered the development of a perfect rejection method to the ABC rejection method. We identified the important factor that can influence the quality of estimation (ϵ), in the case of the rejection method, and we also described the impact of setting ϵ on the accuracy and computational cost. In order to obtain a more efficient approximation of the posterior, ABC with SMC was introduced.

In this chapter, we discussed the ABC SMC technique, and we investigated a possible algorithmic setting that can improve the efficiency of the algorithm. Within an ABC algorithm the main goal is to achieve an accurate estimation with a lower computational time, compared to other Bayesian inference methods. We also identified that the choice of ϵ plays an essential role in performing the ABC SMC algorithm. An adaptive choice of the tolerance level, based on quantiles, was introduced instead of having to set the tolerance values manually. Yet, it is hard to determine which quantile should be used, but we conclude that the use of a higher quantile resulted in a more efficient estimation but adding to the computational cost.

We also outlined a possible issue that can arise in practice: the main difficulty in performing ABC SMC is to determine the approximate number of required simulations to provide a reasonable estimation. In some situations, it is unpractical to match precisely the simulated and observed data; hence, there is always the smallest tolerance level that will not be exceeded anymore. It might be beneficial to be able to approximately specify a target tolerance level to avoid intensive running time without any improvement in estimation. It may, therefore, be beneficial to identify the tolerance level based on a pilot run of the model, and hence an appropriate stopping time for the algorithm could then be determined. The main advantage of the ABC approach is that it applies to any complex model without any restriction,

as long as one can simulate from this model.

Chapter 5

Application to the Lotka-Volterra Model

Lotka-Volterra (LV) is considered as stochastic reaction network which was developed within the modelling context in ecology (Lotka, 1932). This system has been modelled the process several times, and many inference methods have been applied to this model. For example, ABC SMC (Toni et al., 2009), PMCMC (Golightly and Wilkinson, 2011), (Owen et al., 2015) and pseudo-marginal sampler based on truncation (Georgoulas et al., 2017). These methods will be applied with a slightly different setting. A comprehensive comparison between these approaches in terms of accuracy and the computational cost will be performed. The ABC SMC scheme of (Toni et al., 2009) was performed with a deterministic tolerance schedule, and in this thesis, this method will be performed with an adaptive tolerance schedule tuning. In addition, a different choice of tolerance schedule, a predefined target tolerance and the number of particles will be investigated. Owen et al. (2015) consider the ABC algorithm to initialise the PMCMC algorithm to obtain a faster convergence. In contrast, in this thesis, the PMCMC will be applied to different synthetic data sets that are generated at different parameter settings to assess the performance of the algorithm. In a pseudo-marginal sampler based on a random truncation of (Georgoulas et al., 2017), the LV model comprises four reactions while the LV model in this thesis is designed to make it comprise of three reactions. This method provides an exact result for the model, which makes it an attractive approach to be used to assess the accuracy of other approximation methods.

This chapter aims to build a stochastic simulation of the LV model. Afterwards, the time series data that are generated from the model are used as our synthetic data. Also, this chapter will demonstrate the performance of inference of reaction rates

of the LV model from a Bayesian perspective over continuous time Markov chain models with no available explicit likelihood. In addition, the comparisons will be carried out between these inference methods in terms of accuracy and computational expensiveness.

5.1 The Lotka-Volterra Model (LV)

The LV model can be defined in terms of a stochastic kinetic model consisting of two (nonnegative integer values) species, x_1 for the prey and x_2 for the predators. The interaction between them within the population can be modelled by the following reaction equations:

$$\begin{aligned} R_1 &: x_1 \rightarrow 2x_1 && \textit{prey reproduction} \\ R_2 &: x_1 + x_2 \rightarrow 2x_2 && \textit{predator reproduction} \\ R_3 &: x_2 \rightarrow \phi && \textit{predator death} \end{aligned} \tag{5.1}$$

The LV model includes biochemical reactions relying on hazard functions which depend on the current state of the system as defined in section 2.5.1. The hazard function for the first order reaction R_1 is defined by the hazard:

$$h_1(x, c_1) = c_1 x_1.$$

The hazard function of a second order reaction R_2 can be defined by the number of combinations of two species x_1 and x_2, it is given as:

$$h_2(x, c_2) = c_2 x_1 x_2.$$

For the reaction R_3, the hazard is:

$$h_3(x, c_3) = c_3 x_2.$$

Having identified the reaction equations and the stochastic hazard functions of the system, the system can be defined as the LV model using CTMC. The states of our CTMC are labelled with pairs of values (prey and predator counts), so that: $\boldsymbol{x}(t) = (x_1(t), x_2(t))$. The transitions of the CTMC correspond to the reaction equations in (5.1). There can be more than one outgoing transition from a given state. The transition to the next state will be determined by the minimum of

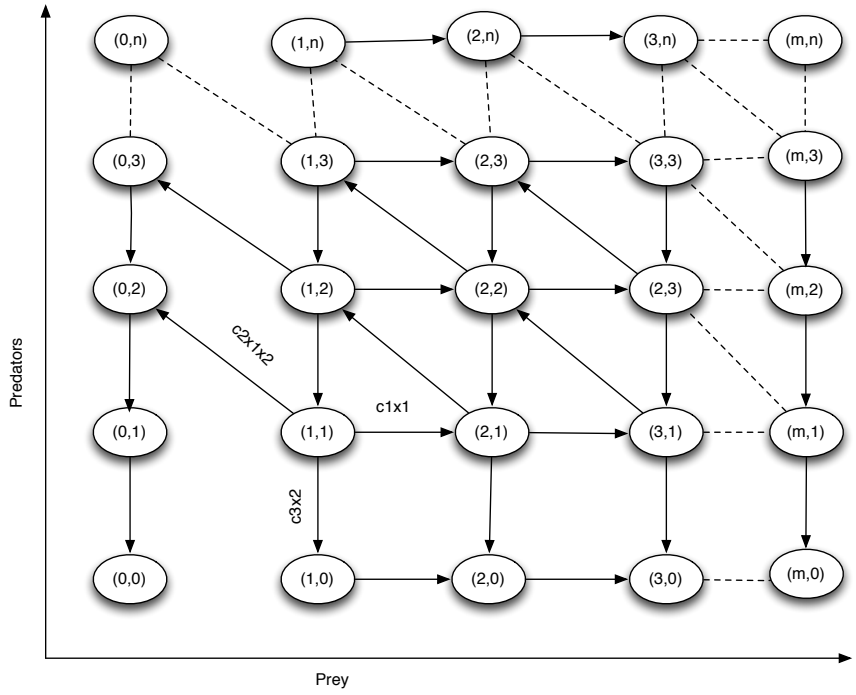


Figure 5.1: The CTMC that describes the LV model. States are labeled with the number of prey and predators. Also, the transitions are associated with corresponding reactions. All the transitions represented with horizontal arrows will have the transition rate $h_1 = c_1 x_1$, the transitions represented with vertical arrows will have the transition rate $h_3 = c_3 x_2$ and the transitions represented with diagonal arrows will have the transition rate $h_2 = c_2 x_1 x_2$.

the exponentially distributed waiting times such as: waiting time $= min(t_1, t_2, t_3)$, where $t_1 \sim Exp(h_1)$, $t_2 \sim Exp(h_2)$ and $t_3 \sim Exp(h_3)$. Figure 5.1 shows how the interactions between prey and predators can be modelled by CTMC.

5.2 Simulation of the Lotka-Volterra Model

The properties of the stochastic biological system are such that the behaviour of the system changes as the reaction rate parameter is changed. To understand the behaviour of the LV system, we simulate the LV model at different stochastic constant rates (c_1, c_2 and c_3)to produce different synthetic data sets. The LV system exhibits several behaviours such as the convergence to a stable fixed point, a periodic limit cycle behaviour, exponential growth of the prey or an exponential decay of the predators. In this thesis, we mainly focus on inferring those reaction rate parameters, and the analytical solution to this system is beyond the scope of this work. Therefore, a brief description of the behaviour of the dynamical system of the LV model will be given and refer the interested reader to (Liu et al., 2005) for more information.

In addition to displaying the different behaviour of the system by simulating from the mathematical model, the characteristics of the solution curves will be determined to understand how the populations can change over time. Several approaches to observe the stability of the system have been proposed, such as linearisation and nullclines. The fixed point (equilibrium point) of this system is denoted by (x^*, y^*) and located where the nullclines intersect. Population stability can occur when $\frac{dx_1}{dt} = 0, \frac{dx_2}{dt} = 0$, which means the population level will not change. The fixed points can be obtained by solving the system of differential equations. Each equilibrium point can be analysed individually and determined through the Jacobian matrix (Kot, 2001).

The following set of equations can describe the change of the prey and the predators with respect to the time:

$$\begin{aligned} \frac{dx_1}{dt} &= ax_1 - bx_1x_2 = x_1(a - bx_2) \\ \frac{dx_2}{dt} &= \delta x_1x_2 - cx_2 = -x_2(c - \delta x_1), \end{aligned} \tag{5.2}$$

where x_1 represents the number of prey, x_2 represents the number of predators, a is the prey production rate, b is the interaction rate, c is the predator death rate and

δ is the predator population production rate.

The stability of the system occurs when the number of prey and predator are not changed through time. That means the derivatives of equations (5.2) are equal to zero as follows:

$$\begin{aligned} \frac{dx_1}{dt} &= 0 = x_1(a - bx_2) \\ \frac{dx_2}{dt} &= 0 = -x_2(c - \delta x_1). \end{aligned} \tag{5.3}$$

When the system of equations is solved, the first points (equilibrium points) $(x^*, x_2^*) = (0, 0)$ and the second point $(\frac{c}{\delta}, \frac{a}{b})$ are obtained.

The stability analysis of the equilibrium point requires a linearisation of the LV equation using partial derivatives. The Jacobian matrix of the LV model is given by:

$$\mathbf{J}(x^*, x_2^*) = \begin{pmatrix} a - bx_2 & -bx_1 \\ \delta x_2 & \delta x_2 - c \end{pmatrix}.$$

At the equilibrium point $(x^*, x_2^*) = (0, 0)$, the Jacobian matrix $\mathbf{J}$ is:

$$\mathbf{J}(0, 0) = \begin{pmatrix} a & 0 \\ 0 & -c \end{pmatrix}.$$

which result in the characteristic equation $\lambda^2 - (a - c)\lambda - ac = 0$, with the following eigenvalues:

$$\lambda_1 = a, \quad \lambda_2 = -c.$$

In the LV model, both a and c are always positive, the sign of the eigenvalues will always differ. Hence, the first point (equilibrium) at the origin is a saddle point. This fixed point is the point of the extinction of the prey and the predators (Kot, 2001), (Liu et al., 2005).

The Jacobian matrix of the second point $(\frac{c}{\delta}, \frac{a}{b})$ is:

$$\mathbf{J}\left(\frac{c}{\delta}, \frac{a}{b}\right) = \begin{pmatrix} 0 & -\frac{b\delta}{\delta} \\ \frac{a\delta}{b} & 0 \end{pmatrix},$$

which result in the characteristic equation $\lambda^2 + (ac) = 0$, with the following eigenvalues:

$$\lambda_1 = i\sqrt{ac}, \quad \lambda_2 = -i\sqrt{ac}.$$

Because the eigenvalues are imaginary, the second fixed point is elliptic. Hence, the solutions are periodic limit cycle around this point. The prey and predators curves are a closed orbit that oscillates around the fixed point with a period $w = \sqrt{ac}$ without damping. The solutions can be visualised as orbits in "phase space" (where x_1 and x_2 are plotted against the others without time being shown in the plot). This plot can represent a closed trajectory of a stable coexistence of both populations either at a limit cycle or the fixed point (Kot, 2001), (Liu et al., 2005).

The other possible behaviour is that the prey grows exponentially in the absence of the predators $x_2 = 0$. Then, the first equation of the system 5.2 become:

$$\frac{dx_1}{dt} = ax_1, \tag{5.4}$$

and this is known as the exponential growth of prey. When there is no prey left in the system $x_1 = 0$, the predators can exponentially decay, that is:

$$\frac{dx_2}{dt} = -\delta x_2, \tag{5.5}$$

where the model reaches extinction. More detail about analysing the LV system can be found in (Kot, 2001), (Liu et al., 2005). In the following section, the simulation of several data sets and their behaviour will be presented.

5.2.1 Changing the Reaction Rate Parameters

This section considers different data sets to explore the influence of varying reaction rates on LV model behavior. To simulate the behavior of the LV model, we initialise the number of prey as $x_1 = 700$ and the number of predators as $x_2 = 200$. A finite state space is assumed, therefore; a maximum population for both species is assumed to be 1000.

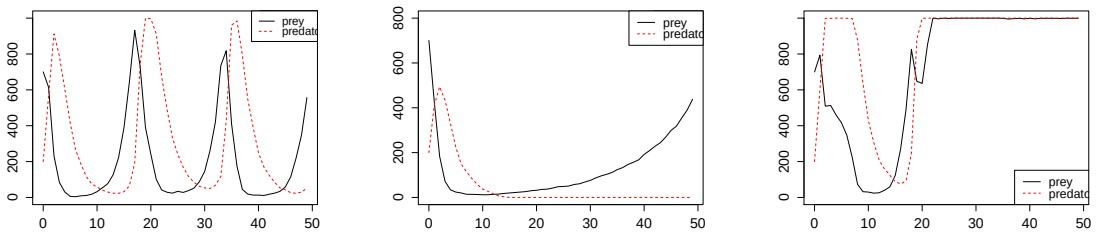
5.2.2 The Effect of Varying the Prey Production Rate c_1

In order to explore the effect of changing the rate at which the prey is produced, we simulate three synthetic data sets at three different values of c_1. In the first case,

a data set is simulated at the reaction rate values: $(c_1 = 0.6, c_2 = 0.002, c_3 = 0.4)$ and the initial states for the prey and the predators are: $x_1 = 700$, $x_2 = 200$. The resulting trace of both the prey and the predators from the simulation is shown in Figure 5.2 (a). There is a relationship between the count of both species predators and prey, the predators count increases because there is plenty of the prey and hence the prey population decline due to the interaction. Since the predators decrease to a lower level, the prey can grow again with the rate $c_1 = 0.6$. This is result in a stable coexistence at a periodic limit cycle as described in section 5.2.

In the second case, the prey reproduction rate is low $(c_1 = 0.1, c_2 = 0.002, c_3 = 0.4)$. It can be seen that, when the prey reproduction rate is low, the numbers of alive prey are decreased. If there are not plenty of the prey, the predators could not reproduce and subsequently reach zero as shown in Figure 5.2 (b). It turns out that the predators become extinct and hence the prey population would increase exponentially with respect to the time. Equation 5.4 describes the exponential growth rate of the prey population.

In the third case, the prey growth rate is assumed to be high $(c_1 = 0.8, c_2 = 0.002, c_3 = 0.4)$. The prey count begins with a population 700, but then dropped as it is consumed by the predators (see Figure 5.2 (c)). Then, the prey population is able to reproduce as the production rate is high and hence the number of predators can increase. It can be seen in Figure 5.2 (c) that the system converges to the fixed stable point (the maximum population point 1000) as both the prey and the predators are not changed. Hence, when the prey growth rate $c_1 = 0.8$ is high, and the interaction level is relatively small $c_2 = 0.002$, a stable coexistence of both species is obtained.



(a) $c_1 = 0.6$ (b) $c_1 = 0.1$ (c) $c_1 = 0.8$

Figure 5.2: The figure shows the way in which the behaviour of the stochastic LV model can vary when the prey production reaction $c_1 = \{0.6, 0.1, 0.8\}$ varies.

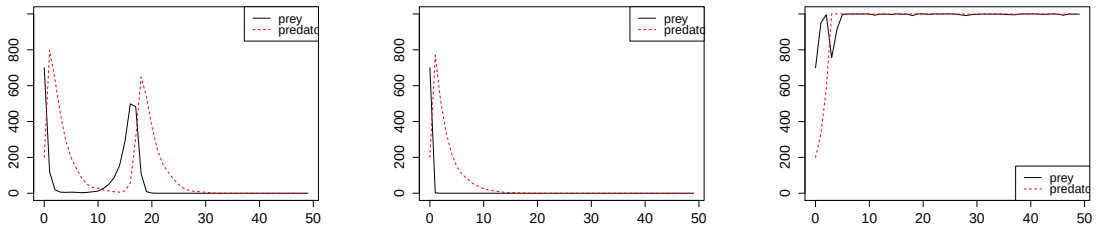
5.2.3 The Effect of Varying the Predator Death Rate c_2

The predator reproduction rate represents the interaction level between both species at which the prey is consumed by predators. The rate c_2 will be changed, and all other rates of reaction and the initial numbers of prey and predators are kept the same ($c_1 = 0.6, c_2, c_3 = 0.4$), just as the previous setting.

When the interaction level is assumed to be $c_2 = 0.004$, the model shows unstable periodic cycle behaviour within a period. Then, this limit cycle is damped which leads to extinction as shown in Figure 5.3 (a).

When the interaction is high $c_2 = 0.008$, the predators consume the prey. Due to the high rate of interaction, the prey will die out first, and this is followed by the exponential decay of the predators as described in equation 5.5. Hence, both the prey and the predator populations reach extinction as depicted in Figure 5.3 (b).

A low rate $c_2 = 0.001$ results in diminished the interaction between the prey and the predator. As the growth rate is relatively high $c_1 = 0.6$, the number of prey remains high, and only an insignificant number of prey is consumed by the predator. Therefore, both of them reach the maximum value in the system and stay stable as shown in Figure 5.3 (c). This results in a stable coexistence of both populations at equilibrium point as described in section 5.2.



(a) $c_2 = 0.004$ (b) $c_2 = 0.008$ (c) $c_2 = 0.001$

Figure 5.3: The figure shows the way in which the behaviour of the stochastic LV model can vary when the prey production reaction $c_2 = \{0.004, 0.008, 0.001\}$ varies.

5.2.4 The Effect of Varying the Predator Death Rate c_3

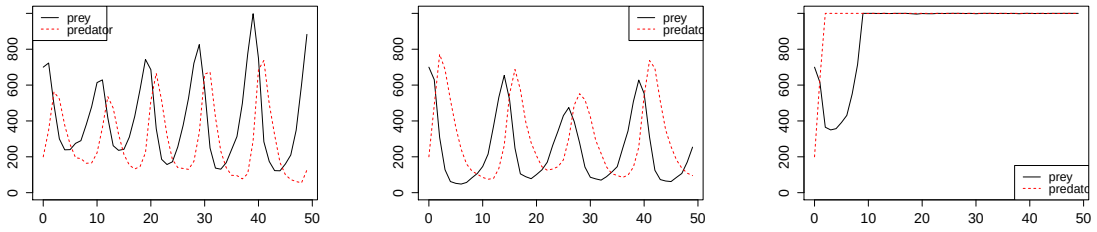
The rate, c_3 represents the rate of the predator's death. To investigate the effect of changing this parameter, we keep all initial populations and the parameter rates are the same and vary ($c_1 = 0.6, c_2 = 0.002, c_3$).

In case of a high predator death rate $c_3 = 0.8$, the prey number increases and reaches a higher population value as compared to that of the predator. Consequently, the predators can reproduce again due to the interaction. This tendency results in fluctuations (growth- decline) in the LV model and neither the prey nor the predators die out completely as shown in Figure 5.4 (a). The limiting behaviour of the system is a stable limit cycle.

When the death rate is $c_3 = 0.5$, neither of species die out, and the system shows a stable coexistence of both populations at limit cycle period as shown in 5.4 (b).

Finally, when the death rate of the predator is low $c_3 = 0.2$, and the interaction rate is quite low $c_2 = 0.002$, the prey count will reach to the maximum value as it is able to produce at rate 0.6. This is result in a stable coexistence at high population levels as depicted in 5.4 (c).

‘



(a) $c_3 = 0.8$ (b) $c_3 = 0.5$ (c)$c_3 = 0.2$

Figure 5.4: The figure shows the way in which the behaviour of the stochastic LV model can vary when the prey production reaction $c_3 = \{0.8, 0.5, 0.2\}$ varies.

5.3 Parameter Inference

This section demonstrates the proposed inference method described in chapters 3 and 4 when the direct inference is infeasible due to the intractable likelihood. In the case of intractable likelihood, ABC and PMMH approaches allow performing Bayesian inference in the LV model. The ABC is an approximate method that does not require the evaluation of the likelihood but instead relies on simulations from the model. PMMH is considered as an exact approximation, where PMMH is based on an unbiased estimate of the likelihood. To quantify the quality of the approximation obtained from both methods, the recently proposed Pseudo-marginal method (the Gibbs sampler) based on random truncation is used to perform an exact inference

for the LV model. The performance of the two approximate approaches on the LV model is illustrated and then compared with the exact method. The accuracy of estimating the posterior distribution and the computational cost for all inference methods are involved in the comparison. Once the capability of ABC is assessed, it will be applied on larger datasets and the algorithmic parameter will be investigated.

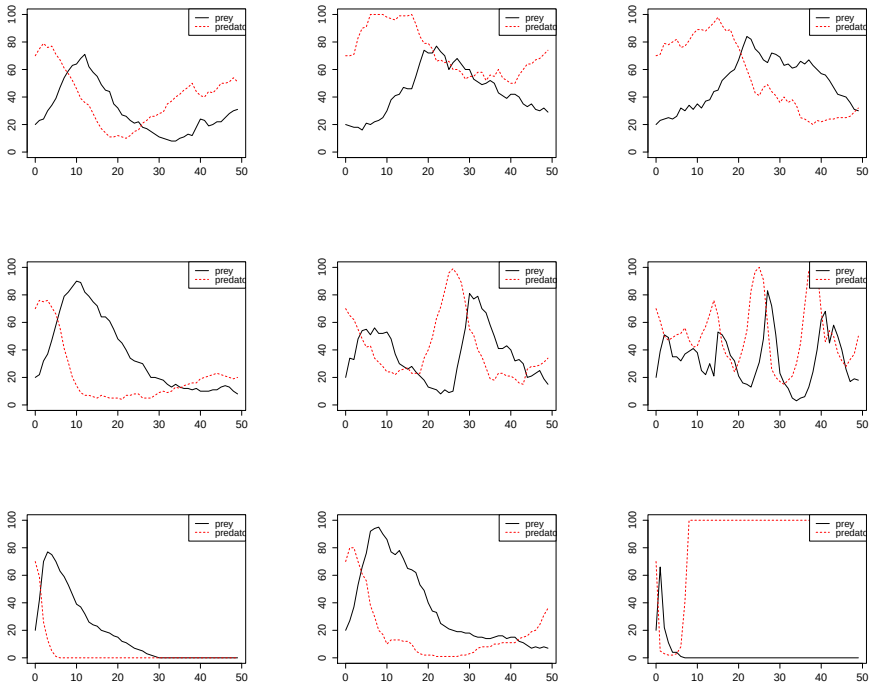
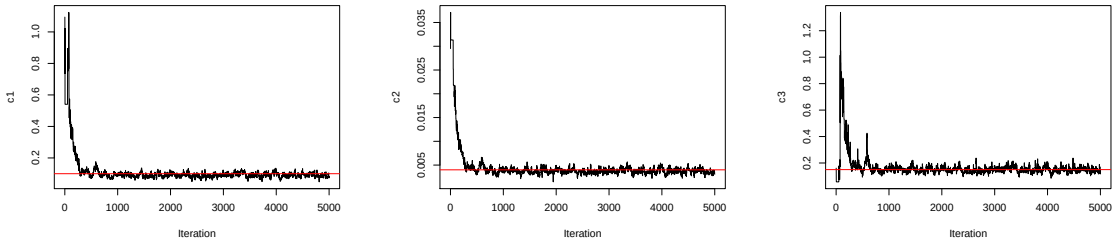


Figure 5.5: The figure illustrates several behaviours of the LV model, which are simulated at different parameter values, as presented in Table 5.1.

5.4 Bayesian Inference for the LV Model

The performance of Bayesian inference methods will be demonstrated in this section. The Gibbs sampler based on random truncation is first applied. The performance of this method relies on the size of the state space. The computational cost will be intensive when the state space is large. Consequently, instead of performing inference using the simulated data set involving a high number of species (see section 5.2), the LV system will be described with low counts of species. The low number of species makes the inference computationally feasible. According to section 5.2, the data sets are simulated with the maximum population count, which is assumed to be 100 and the initial count for prey is $x_1 = 20$ and for the predator is $x_2 = 70$.

We simulate several synthetic data sets from the LV model using different values



(a)c_1 (b)c_2 (c)c_3

Figure 5.6: The figure shows the trace plot for each LV model parameter obtained from the Gibbs sampler, where the Gibbs sampler applied to the synthetic data simulated at: $(c_1 = 0.1, c_2 = 0.004, c_3 = 0.15)$. The red line indicates the true parameter values.

of kinetic rate parameters, c_1, c_2, c_3, for each of them as shown in Table 5.1. The observations (noise free) are simulated over 50 equally spaced fixed time points $t = (0, 1, 2, 3, \cdots, 49)$. The results exhibit different behaviours of the system as is depicted in Figure 5.5.

The proposed inference methods will be applied to all synthetic data sets included in Figure 5.5. Each inference method is applied to a single data set and is considered as one experiment. Each independent run applied to the single data set is repeated 10 times. The results from all these experiments will be recorded and then discussed.

The Gibbs sampler is performed to estimate the exact posterior of the LV model parameters by considering the data sets in Figure 5.5. Each experiment is performed independently with a number of 5000 samples. The conjugate gamma prior is assumed for each model parameter. The choice of ranges of the parameters for priors is made based on the previous knowledge given in (Georgoulas et al., 2017) (Toni et al., 2009) as follows:

$$\pi(c_1) \sim \Gamma(2, 4)$$

$$\pi(c_2) \sim \Gamma(2, 100)$$

$$\pi(c_3) \sim \Gamma(2, 4).$$

The resulting 5000 samples from the Gibbs sampler based on truncation is depicted in the trace plot in Figure 5.6. It is possible to assess the mixing and convergence of the Gibbs sampler through Gelmen and Rubin as is described 3.5.1. We quantify the mixing of the resulting Markov chain using the visual inspection for each

Table 5.1: Computational time of the Gibbs sampler and the ABC SMC algorithm.

Experiments	Parameter setting	Gibbs sampler	ABC SMC
Experiment 1	$(c_1 = 0.1, c_2 = 0.004, c_3 = 0.15)$	6.2 Hours	22 minutes
Experiment 2	$(c_1 = 0.1, c_2 = 0.002, c_3 = 0.15)$	8.1 Hours	57 minutes
Experiment 3	$(c_1 = 0.08, c_2 = 0.002, c_3 = 0.1)$	8.1 Hours	49 minutes
Experiment 4	$(c_1 = 0.11, c_2 = 0.004, c_3 = 0.1)$	9.5 Hours	23 minutes
Experiment 5	$(c_1 = 0.2, c_2 = 0.006, c_3 = 0.25)$	15.3 Hours	24 minutes
Experiment 6	$(c_1 = 0.5, c_2 = 0.01, c_3 = 0.7)$	18.4 Hours	27 minutes
Experiment 7	$(c_1 = 0.29, c_2 = 0.017, c_3 = 0.14)$	11.1 Hours	19 minutes
Experiment 8	$(c_1 = 0.09, c_2 = 0.07, c_3 = 0.42)$	8.50 Hours	22 minutes
Experiment 9	$(c_1 = 0.2, c_2 = 0.005, c_3 = 0.11)$	10.6 Hours	17 minutes

parameter. According to Figure 5.6, the Markov chain for each parameter exhibits a good mixing, as for example, the first parameter c_1 is although initialised around one moves from one to the reasonable state space of the true parameter quickly after approximately 500 iterations as shown in Figure 5.6 (a). Therefore, the initial part of the Markov chain is discarded (burn in). The benefit of using this method over other alternative MCMC approaches is that it converges to the reasonable state space of the parameter quickly after a small length of iterations.

These samples are used to form the exact marginal posterior of the LV model parameters as depicted in Figure 5.7. This inference method is repeated on the second synthetic data set that is generated with parameters $(c_1 = 0.1, c_2 = 0.002, c_3 = 0.15)$ and result in a similar estimation as shown in Figure 5.7.

The exact posteriors in Figure 5.7 are used to assess the accuracy of its corresponding approximation posteriors obtained from the ABC SMC algorithm.

The ABC SMC algorithm (described in chapter 4) is performed on the same data set that is used with the Gibbs sampler. For a fair comparison, the same setup in terms of priors and initial conditions are used. One of the main objectives is to investigate how much the quality of the approximation and the computational time have been affected by various choices of N. Thus; the ABC SMC is performed with different particle numbers $N = \{500, 1000, 2000, 4000, 8000\}$. The investigation of tuning the ABC SMC algorithmic parameter such as the choice of the number of particles, the target tolerance and the adaptive tolerance schedule will be considered in section 5.8; for more information see Appendix B.

Figure 5.8 shows that the posteriors resulting from the ABC SMC with $N = 8000$

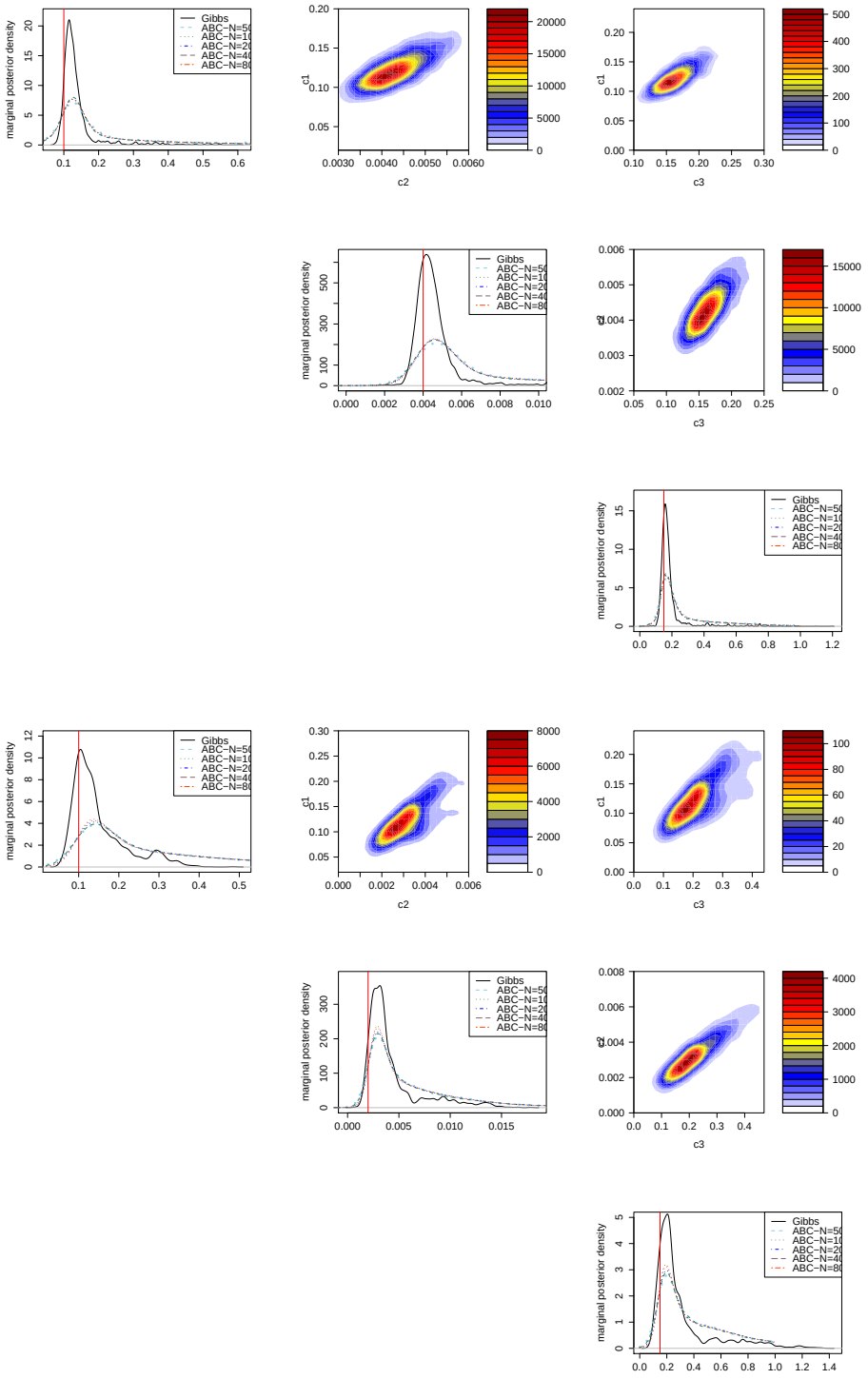


Figure 5.7: The marginal posterior densities of the parameters of the LV model that are obtained by performing Gibbs algorithm is compared with the approximated marginal posteriors obtained from the ABC SMC with different N using the synthetic data sets which are simulated at: $(c_1 = 0.1, c_2 = 0.004, c_3 = 0.15)$ (top) and $(c_1 = 0.1, c_2 = 0.002, c_3 = 0.15)$ (bottom) respectively. The contour plots show the joint densities which are estimated using a kernel density estimate (KDE).

are quite tightly distributed compared to the prior. This implies that we learn from combining the simulated data with the priors about the unknown LV parameters.

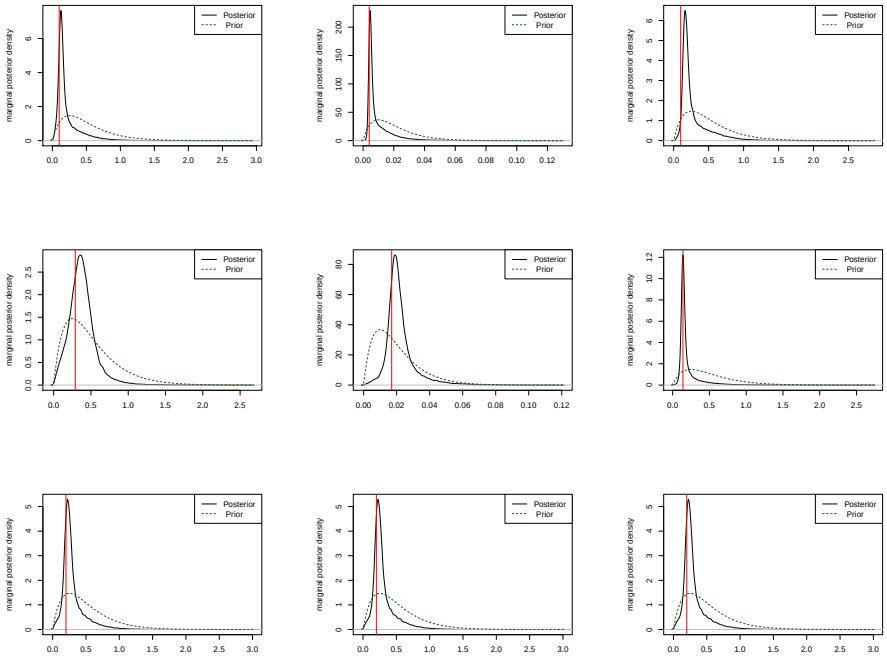


Figure 5.8: Approximation posterior distributions of the LV parameters obtained from three individual runs (columns) of the ABC SMC using three different simulated data sets (rows). The prior distributions for each parameter are depicted by the green dashed line and the true parameters are indicated by the red one. The approximate posterior distributions do not resemble the prior distributions, meaning that we are learning from the ABC SMC algorithm about those parameters.

The results indicate that increasing the number of particles that are used in the ABC SMC algorithm have a negligible effect on improving the approximation. Figure 5.7 shows that using a few numbers of particle $N = 500$ still provides a reasonable approximation to the posterior concentrates around the exact posterior mode. In addition, Figure 5.9 shows the ESS for experiment 1; it is clear that the ESS is large for all numbers of particles.

The posterior distributions resulting from performing all numerical experiments using exact and approximate methods given several synthetic data sets are comparable. For example, according to Figure 5.7, it is evident that the posterior distributions of the LV model parameters generated from the ABC SMC algorithm cover the whole exact posterior distributions, even the tails of the distribution, which means it fully explores the state space. However, the approximate posteriors are a little wider compared to the exact posterior distributions. Additional results from replicating

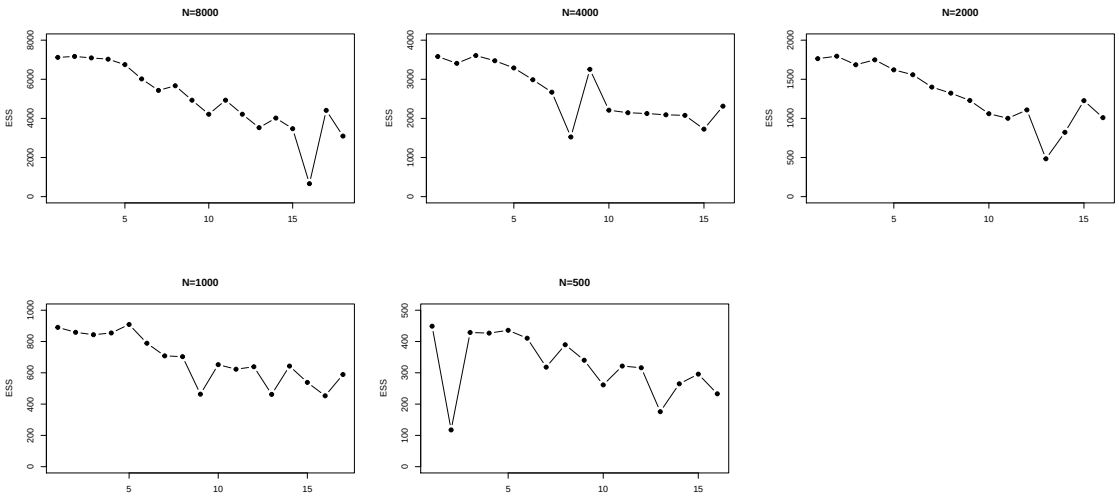


Figure 5.9: The figure shows the ESS resulting from the performance of the ABC SMC algorithm with different $N = \{8000, 4000, 2000, 1000, 500\}$ values respectively. The x axis represents the number of ABC SMC stages.

the comparison are presented in Appendix A.

In terms of the computational cost, the computational time for each run would be recorded. The overall computational cost of all experiments are compared and displayed in Figure 5.10. Table 5.1 represents the comparison of runtimes between Gibbs sampler and ABC SMC with $N = 8000$. For example, the estimation of posterior densities of the LV model parameters is obtained in 6.2 hours on average while the ABC SMC gave the approximations in 22 minutes. The results indicate that the Gibbs sampler appears to be more expensive compared to the ABC SMC. The computational cost of the Gibbs sampler results from the long uniformised sampling paths.

Figure 5.10 shows that the ABC SMC that was performed with $N = 500$ consumes less runtime compared to the ABC SMC was performed with $N = 8000$. The most computational time of the ABC SMC algorithm is caused by the perturbation kernel which evaluates the weighted variances based on current particles as described in section 4.5.2. Hence, the required computational time to evaluate the weighted variances increases as N increases. In general, ABC SMC with a variant setting of N is significantly faster and results in a substantial reduction in the computational cost compared to the computational time associated with the Gibbs sampler.

In conclusion, the Gibbs sampler is easy to apply and does not require special tuning such as selecting the proposal distribution; this gives it an advantage over other MCMC methods. The ABC SMC algorithm has been applied successfully to the LV

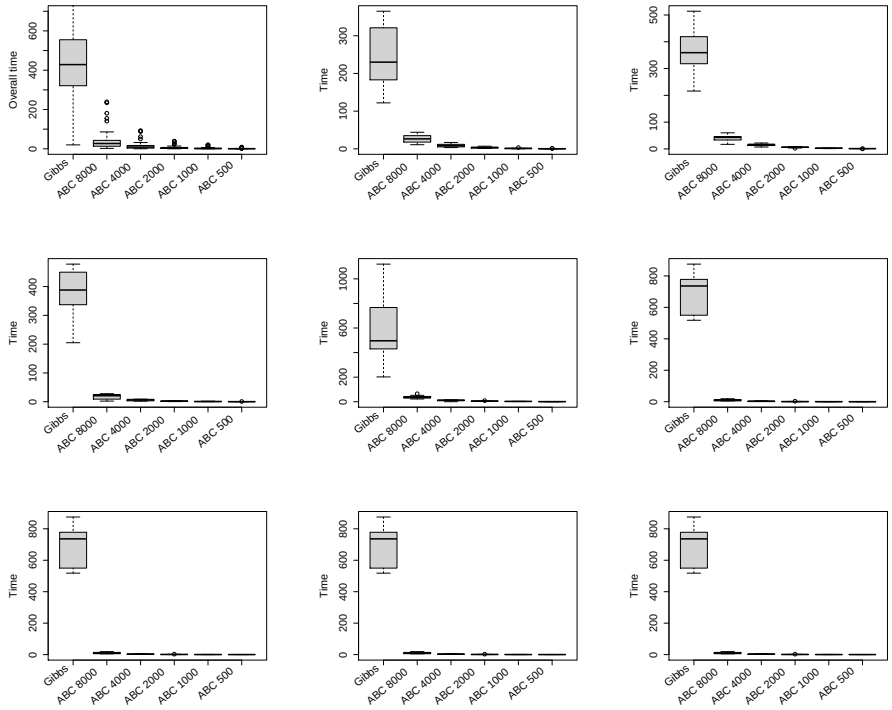


Figure 5.10: The box plots illustrate the distribution of time across the experiments running for each individual data set presented in Figure 5.5. It is obvious that the ABC SMC algorithms are cheaper than the Gibbs sampler.

model as it explores the true parameter space with different parameter settings. The ABC SMC algorithm provides a consistent approximation to the exact posterior with different parameter settings which make it a promising approach. This motivates us to carry out an empirical investigation of the effectiveness of tuning the important algorithmic parameters of the ABC sampler in the next section.

5.5 Further Investigation

The efficiency of the ABC SMC algorithm relies on the user setting, where the user settings are defined as the tuning of the adaptive tolerance schedule, the number of stages and the number of particles. More accurate approximation of the posterior distribution can be obtained when the final tolerance value is set to be very small. However, at some stage, the tolerance will be lowered to a value where sampling becomes expensive. Therefore, it would be useful to find a convenient way of selecting the final target tolerance.

Therefore, we suggest predefining the target tolerance, which must be satisfied for each individual algorithm based on a pilot run that is explained in section 4.5.1. The strategy relies on the way of defining the target tolerance and the number of simulations that are used to estimate the target tolerance.

We will perform two further experiments that can provide an idea of how the target tolerance varies as the way of selecting the percentile and the number of simulations varies. These experiments will be performed to ensure the appropriate choice of the target tolerance.

In the previous ABC SMC experiments, we have used different numbers of particles $N = \{8000, 4000, 2000, 1000, 500\}$, and the adaptive tolerance schedule is reduced according to $\alpha = 0.5$ of the distances between the observed and the simulated data at each stage until the chosen target tolerance is achieved. The target tolerance is estimated as the first percentile of the range of distances between simulated data sets. We have to stop the algorithm when the target tolerance is achieved. The further experiments are designed to derive the efficiency of the ABC SMC algorithm when a lower tolerance level can be achieved.

The experiment assesses the influence of variations of the target tolerance on the accuracy of approximation and the computational cost.

In the first experiment, an estimation of the target tolerance can be evaluated by

performing a pilot run to measure the discrepancy between the observed data y and the simulated dataset y_i^*, where $i = 20000$, represents the number of traces that are generated at a drawn parameter value from the prior. In the first run of the first experiment, the target tolerance is chosen to be 5 percentile of the range of the distances. The ABC SMC algorithm is performed 10 times and terminates when this target tolerance is achieved.

This procedure is repeated for other experiments to estimate the target tolerance based on different percentiles $Q = \{1, 0.5, 0.1\}$. The results of the target tolerances were different among data sets. This procedure is repeated 40 times, where it is applied 10 times with each $Q = \{5, 1, 0.5, 0.1\}$. For the purpose of comparison, the Gibbs sampler is performed by considering those data sets.

In order to determine the accuracy of this approximation, the Jensen-Shannon divergence (JSD) is used to quantify the discrepancy between the resulting exact and approximate posterior distributions that are obtained from each experiment, which is defined as:

$$\text{JSD}(\pi_i|\pi_i^*) = \frac{1}{2}\sum_i \left(\pi_i \text{log}\frac{\pi_i}{\frac{1}{2}(\pi_i + \pi_i^*)}\right) + \frac{1}{2}\sum_i \left(\pi_i^* \text{log}\frac{\pi_i^*}{\frac{1}{2}(\pi_i + \pi_i^*)}\right),$$

where the term π_i^* indicates the approximate density obtained from ABC SMC and π_i represents the exact density obtained from the Gibbs sampler (Mishtal and Arel, 1012). We observe that using a larger percentile $Q = 5$ results in a large JSD divergence. This is expected because it supports the claim that the approximation tends to be better as the tolerance decreases.

The results of the JSD divergences for each parameter are recorded and visualised in Figure 5.11. In Figure 5.11, each plot consists of four box plots which represent the distribution of JSD divergence values. These values are obtained from 40 runs. It can be observed from Figure 5.11 that the JSD values tend to decrease as Q decreases. This result indicates that when the target tolerance decreased the accuracy of the approximation increased.

The computational cost of these experiments is recorded and represented in Figure 5.12. From Figure 5.12, it can be seen that for the first experiment, the ABC SMC algorithm proceeding with respect to $Q = 5$ has a significantly lowered computational cost compared to the Gibbs sampler. In the second experiment ($Q = 1$), the ABC SMC sampler is still relatively cheaper than the Gibbs sampler. In the third experiment ($Q = 0.5$), the target tolerance is smaller because it requires more

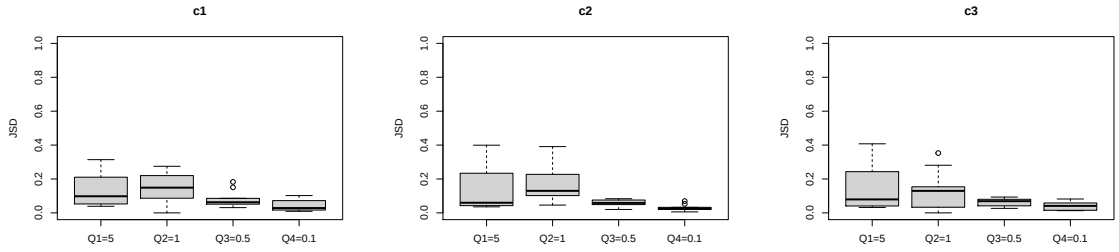


Figure 5.11: The figure of box plots show the JSD divergence of the inferred LV model parameters obtained from the Gibbs sampler and the ABC SMC algorithms with different settings of the target tolerance over 40 experiments. This figure indicates that the accuracy of the approximation increased as the target tolerance decreased.

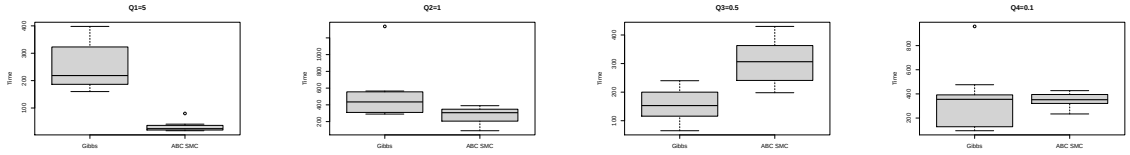


Figure 5.12: Box plots show the total required computational time for the Gibbs sampler and the ABC SMC algorithm across 40 experiments. The result indicates that the computational cost of the ABC SMC algorithm increased as the target tolerance decreased.

simulations to reach it. The small target tolerance makes the computational time of the ABC SMC greater than the Gibbs sampler. Finally, when ($Q = 0.1$), the algorithm requires high numbers of simulations to achieve a target tolerance. This case makes the ABC SMC algorithm extremely expensive.

The second experiment of this section is to verify the claim that increasing the number of simulation traces can lead to lower estimating of the target tolerance. This experiment evaluates the target tolerance using three different ways (e.g: the minimum value, the first percentile and the fifth percentile of the range of distances) for a sequence of 15 different numbers of simulations. In the first case, we estimate the target tolerance based on the fifth percentile of the range of distances using $\{200, 400, \cdots, 3276800\}$ simulation traces. According to Figure 5.13, it can be seen that the target tolerance decreases slightly when the number of simulations increase for the first five evaluations. After that, it remains approximately stable. In the second case, when the target tolerance is estimated based on the first percentile of the range of distances over different sets of the numbers of simulations, the last 10

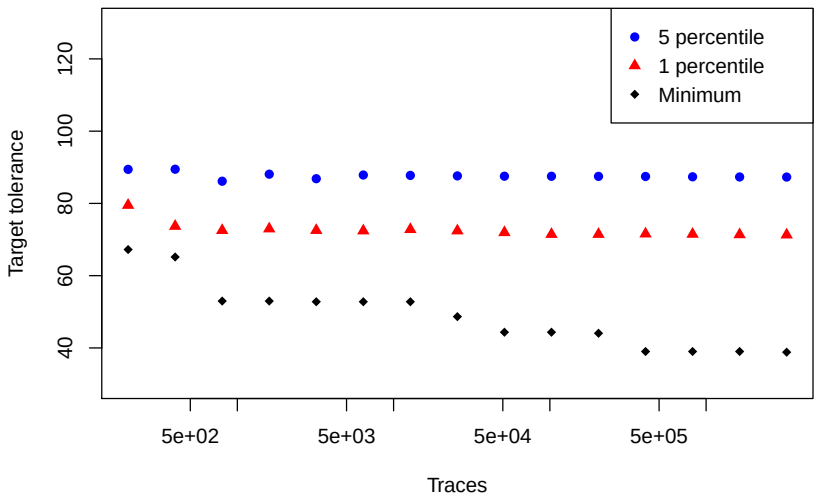


Figure 5.13: Target Tolerance estimation based on using different number of traces and different ways of estimating the target tolerance.

target tolerances become stable. Finally, when the target tolerance is estimated at a minimum value over a set of distances, it can be seen that the distance is decreasing as the number of traces increased.

In conclusion, for the first experiment, the accuracy of approximation increased as a small divergence value is achieved. This is because the target tolerance becomes low, but this contributes to increasing the computational time. The computational cost of ABC SMC relies on the required number of simulations to satisfy the target tolerance. The ABC SMC algorithm still provides a reasonable approximation to the exact posterior and a reasonable computational time, when the target tolerance is estimated based on $Q = 1$. In order to balance the accuracy and the computational cost, according to Figure 5.11 and Figure 5.12 estimating the target tolerance based on the first percentile is used in this work and suggested. The ABC SMC can benefit from parallel hardware computing, which can be used to increase the precision of the approximation.

For the second experiment, we found that the target tolerance becomes approximately similar at some point and no further reduction can be obtained when percentiles are used to define the target tolerance. The minimum value over set distances is reduced as the number of simulations increased. We have observed that using a larger number of simulation traces has not reduced the estimated target tolerance significantly.

5.6 Application of the PMMH on the LV Model

In this section, we provide illustrations of the PMMH procedure to estimate the LV parameters $\theta = (c_1, c_2, c_3)$ given the data set presented in Figure 5.5. The performance of the PMMH algorithm requires an estimate of the likelihood. We employ the bootstrap particle filter described in section 3.2.5 to estimate the likelihood. However, the crucial part of the PMMH algorithm is how algorithmic parameters are chosen, especially the number of particles that are used in the bootstrap particle filter. We note that based on several pilots runs that the variance of the likelihood estimator decreases, as the number of particles N increases. Hence, there is a trade-off between the number of particles and the variance of the likelihood estimator. A small number of particles used in the bootstrap particle filter leads to a high variance in the likelihood estimator. In our implementation, the number of particles is chosen to be $N = 300$. In order to increase the probability of hitting the data and obtaining a non-negligible acceptance rate, a noise in the measurement process will be assumed. Therefore, our simulated time series data sets (depicted in Figure 5.5) consisting of 50 equally spaced observations are subject to Gaussian measurement error with a standard deviation of 20.

Afterwards, we start the PMMH algorithm (8) by initialising the Markov chain with sampling the reaction rate parameters from a Gamma prior. The range of the parameters of the prior distributions are selected based on the knowledge in (Georgoulas et al., 2017) and (Toni et al., 2009), as follows:

$$\pi(c_1) \sim \Gamma(2, 4)$$
$$\pi(c_2) \sim \Gamma(2, 100)$$
$$\pi(c_3) \sim \Gamma(2, 4),$$

then the bootstrap particle filter (as described in section 3.2.5) is applied to estimate the likelihood of the sampled values from the prior. Then, this likelihood estimator is used in a standard MH sampler as described in the PMMH algorithm (8). In order to propose new parameters, a proposal distribution must be selected. The choice of the proposal plays an essential role in exploring the parameter space. A small movement step results in a high correlation between successive samples and therefore a bad mixing. This motivates us to design an adaptive proposal that can be tuned by an unknown covariance $\mathbf{\Sigma}$, which is computed using the current sample during the run of the algorithm. Thus, an adaptive normal random walk (as described in

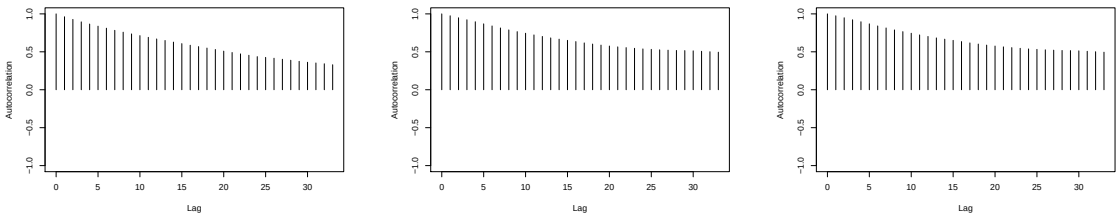
section 3.5) is chosen as follows:

$$q(\theta'|\theta) = N(\theta, c^*\mathbf{\Sigma}).$$

The step length of the proposal is initialised to be $c^* = 0.1$. The algorithm works in two stages. The first one is the adaptive stage (burn in), where the step length c^* is adapted during the run, either by increasing or decreasing c^*. This is to ensure that the acceptance rate is between 20% and 80%. The second stage is the stationary stage in which the step length of the proposal c^* is fixed, and the algorithm will be performed and result in two sets of samples.

The algorithm also assesses the convergence of two chains by using Gelman-Rubin convergence diagnostic test, as described in section 3.5.1. This can be done by running two parallel chains, each chain has a length of iterations, which start from different points in the parameter space.

If the resulting potential scale reduction (shrink factor) $\hat{R}$ of each parameter from this test is $\hat{R} \leq 1.1$, this suggests that two Markov chains are mixed well and reach its stationary distribution. Otherwise, the burn in phase will be increased.



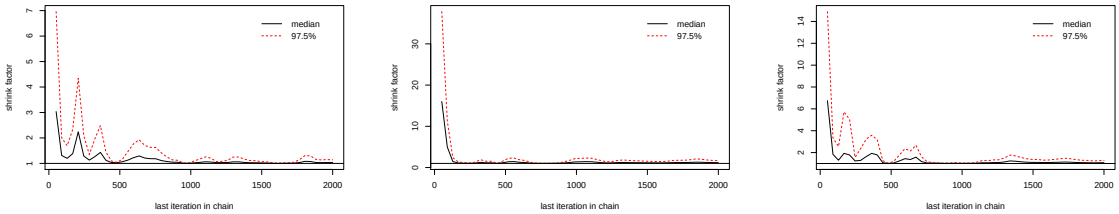
(a)c_1 (b)c_2 (c)c_3

Figure 5.14: The figure shows the strong autocorrelation between samples for the LV model parameters.

The proposed parameters can be either accepted or rejected based on the acceptance probability $a(\theta'|\theta)$. This can be carried out by sampling a single random variable from a uniform distribution such that:

$$u \sim \text{Uniform}(0, 1).$$

If $u < a(\theta'|\theta)$, we accept the proposed parameter θ' and update the Markov chain with its corresponding likelihood estimator as the current state of the Markov chain. Otherwise, it will be rejected, and the chain will return to the state obtained from the previous iteration.



(a)c_1 (b)c_2 (c)c_3

Figure 5.15: The figure shows that all model parameters are converged to their stationary distributions as the resulting potential scale reduction (shrink factor) for all parameters are $\hat{R} \leq 1.1$.

In order to obtain independent samples from the posterior distribution, the samples can be thinned by keeping θth sample from the chain and discarding the rest. The thinning factor can be determined by inspecting the autocorrelation function (acf) plot from the burn in stage. Then, in a stationary stage, the thinning factor is chosen at which the autocorrelation is close to zero. However, Link and Eaton (2012) claimed that thinning can affect the precision of the estimation where more accurate estimation can be obtained if all iterations are used.

In our implementation, the autocorrelation function produced from the PMMH algorithm is strongly autocorrelated as shown in Figure 5.14. The resulting Markov chain from the PMMH algorithm is thinned every 10th of samples from the stationary stages. The autocorrelation between samples is high, which is to be expected given that the PMMH algorithm is sampling more likely from the high posterior region and ignoring the tail of the posterior. This is a typical problem in the PMMH algorithm which will be discussed in section 5.7.1.

The resulting acceptance rate of this algorithm is 74%. The potential scale reduction (shrink factor) for all parameters is $\hat{R} < 1.1$ as is shown in Figure 5.15.

The posterior distributions of the LV model parameters are constructed based on the two sets of 2000 samples with $ESS = 125$ from the stationary stages, after discarding $108,000$ samples from several adaptive stages as burn in.

The resulting marginal posterior densities estimation of the three LV model parameters are presented in Figure 5.16 with their corresponding kernel densities estimation of the pair of parameters. To ensure the consistency of the PMMH method, the PMMH algorithm is repeated several times on different simulated data sets. We initialise the Markov chain from the prior distribution, and we have found that repeating the process with different initial settings yield a similar consistent estima-

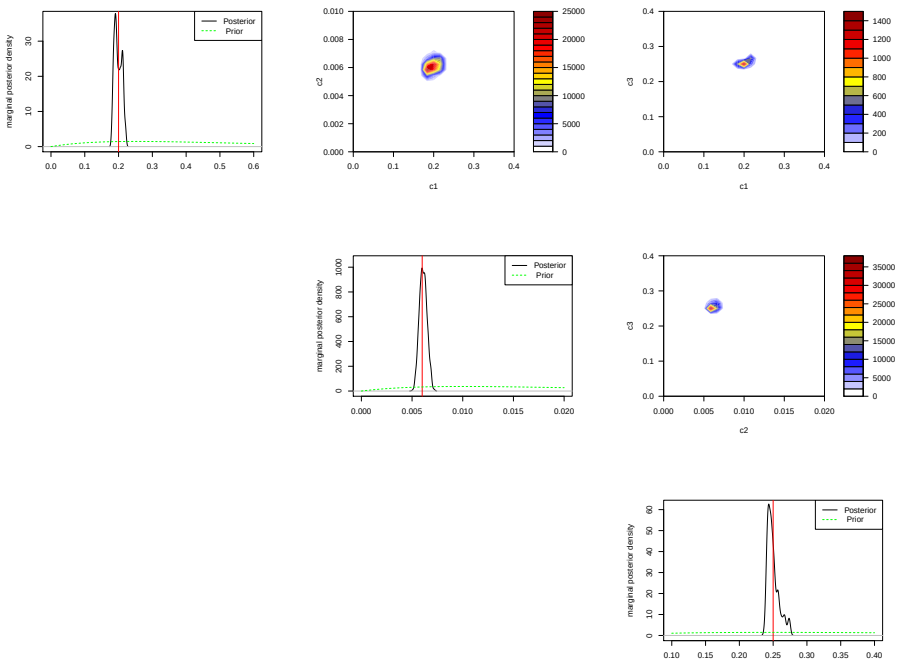


Figure 5.16: The posterior densities of the parameters of the LV model obtained from the PMMH1 (solid line) with its corresponding prior distributions (dashed line). The posteriors are three dimensional probability densities function. Therefore, we depict it with a set of marginal posterior densities. The contour plots show the joint densities which are estimated using a kernel density estimate (KDE).

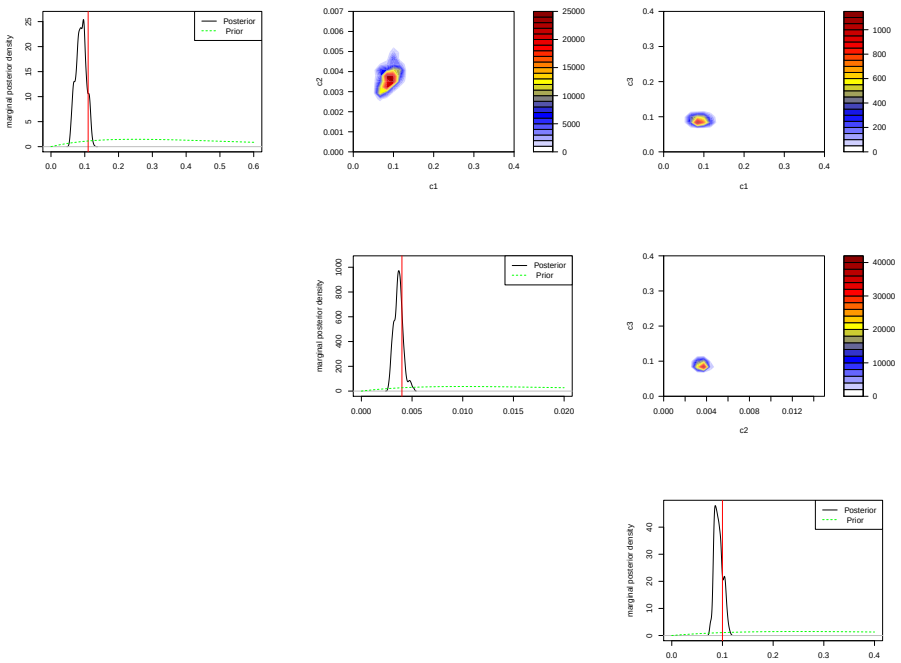


Figure 5.17: The posterior densities of the parameters of the LV model obtained from PMMH2 (solid line) with its corresponding prior distributions (dashed line). The joint posterior densities are estimated using a kernel density estimate (KDE) and are depicted by contour plot.

tion of posterior distributions. However, we have experienced difficulty with slow convergence as it discards around $108,000$ samples before reaching the stationary stage.

Due to the fact that all the reaction rate parameters of the LV model are positive, this motivates the use of a proposal distribution with positive support. Hence, we replicate the application of the PMMH algorithm using a log-normal proposal on the same simulated data sets depicted in Figure 5.5. To distinguish between the two algorithms, we denote the PMMH with a normal proposal algorithm by PMMH1 and the PMMH with a log-normal proposal algorithm by PMMH2.

The output of the PMMH2 algorithm is the two sets of 2000 samples with $ESS = 107$ and the acceptance rate is 64.2% after burn-in $87,000$ samples. It is thinned every 10th of samples. The resulting marginal posterior densities of the LV model parameters are depicted in Figure 5.17.

The resulting posteriors obtained from both the PMMH1 and PMMH2 algorithms are relatively tight around the true values. However, the lower part of the state space is poorly explored. This is due to the property of the local moves of the Markov chain where the state space of the posterior distribution is explored by using the previously accepted sample. Therefore, when the posterior distribution assigns a high probability in a particular region of the state space, most of the proposed samples in this part are likely to accept it. This results in high autocorrelation between samples.

The PMMH1 algorithm requires more time to explore and sample from the correct state space as it discards $108,000$ samples from the burn in phase, while the PMMH2 algorithm converges faster after discarding $87,000$ samples from the burn in phase. This is expected as the proposal density of the PMMH2 only supports the positive state space of the model parameters. The computational efficiency of PMMH1 and PMMH2 algorithms is measured by the ESS, we have found that the PMMH2 has a lower of ESS compared to the PMMH1 algorithm as some of the results are shown in Table 5.2.

5.7 Comparison

In this section, the resulting posterior distributions of the LV model parameters obtained from the ABC-SMC algorithm, the PMMH1 and the PMMH2 algorithms, are compared to the exact posterior distributions obtained from the Gibbs sampler.

Table 5.2: The ESS obtained from various runs of PMMH1 and PMMH2.

	PMMH1	**PMMH2**
Run1	125	107
Run2	284	78
Run3	262	130
Run4	170	68
Run5	116	87

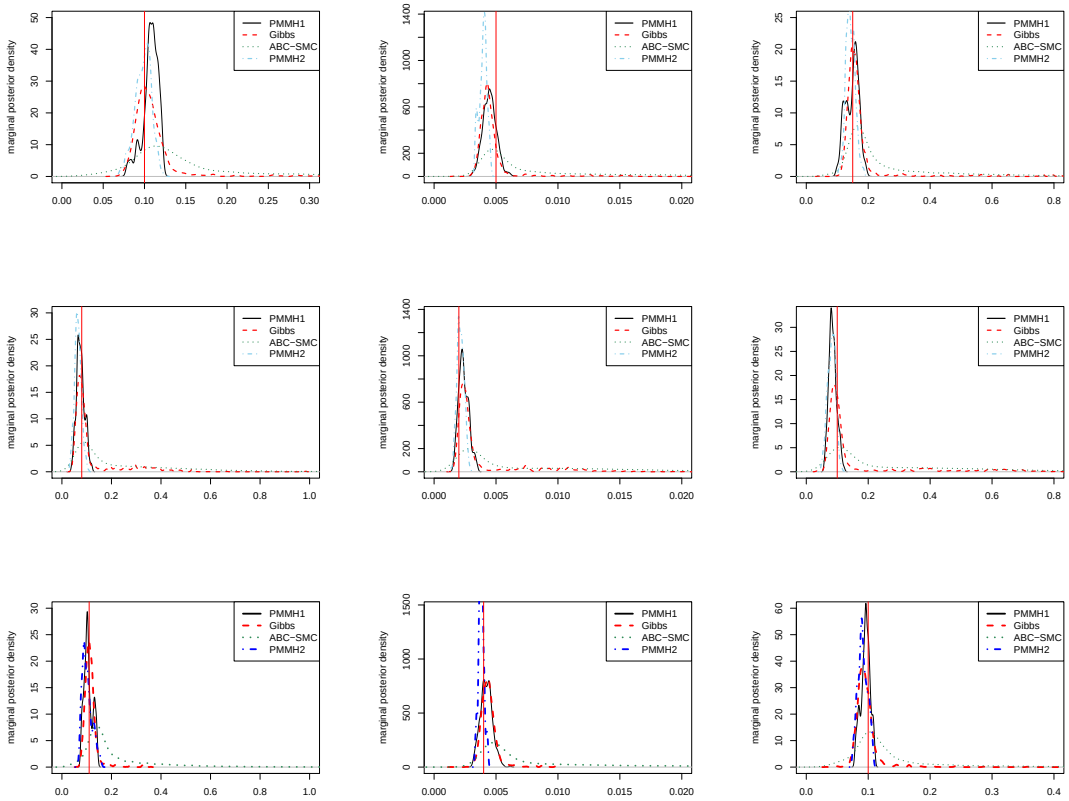


Figure 5.18: The figure shows the approximate posterior densities resulting from applying the ABC SMC and the PMMH algorithms with its corresponding exact densities several times. Each column of this figure represents a specific model parameter that has been inferred, column one represents the posterior densities of c_1, column two represents the posterior densities of c_2 and column three represents the posterior densities of c_3. It can be seen that the estimate posterior distributions (resulting from PMMH1 and PMMH2) are distributed around the exact posterior mode. The approximate posterior distributions are overdispersed and this is due to the ABC approximation where the quality of approximation depends on the level of the target tolerance ϵ, but still, have similar support as the exact posterior densities.

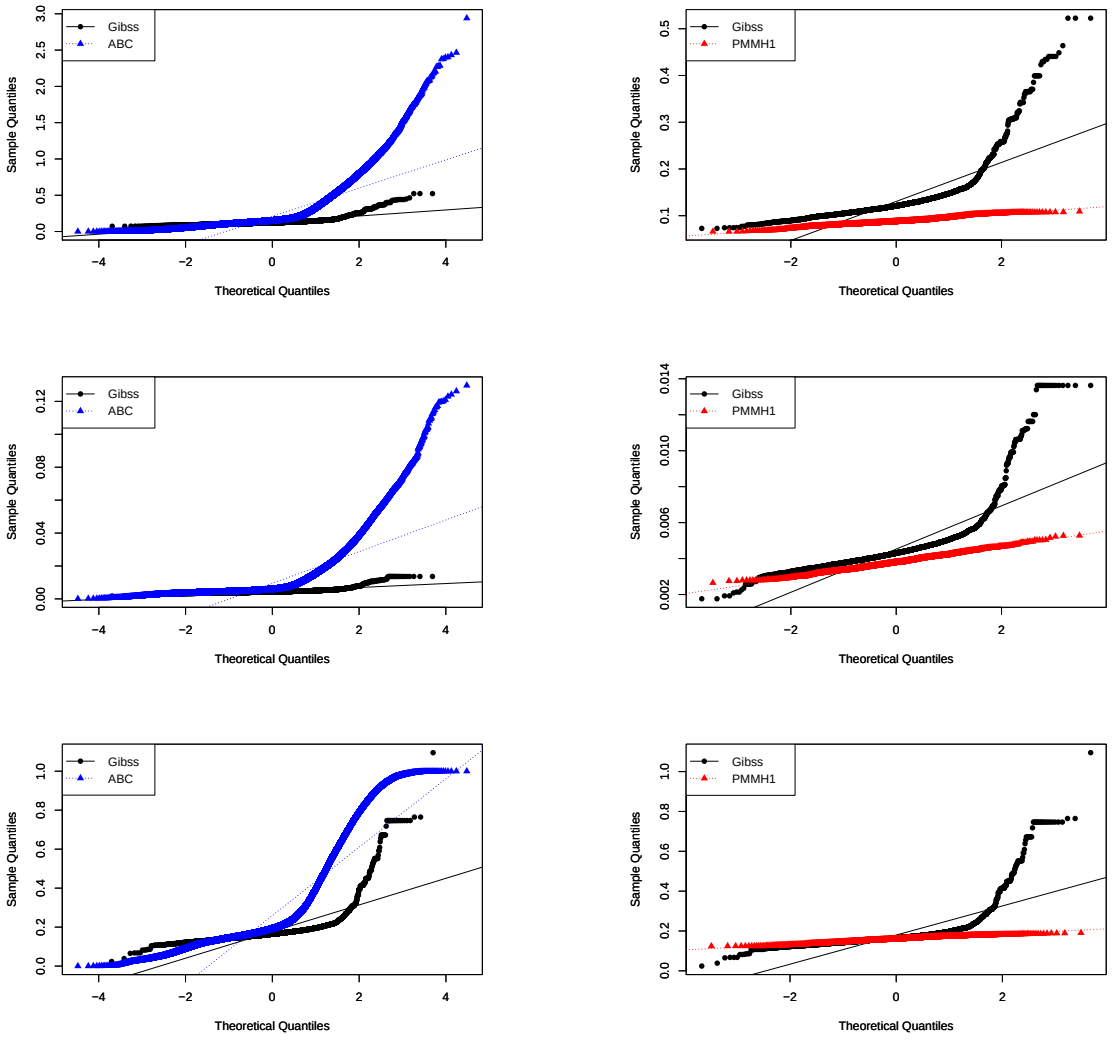


Figure 5.19: Q-Q plots of the estimation of LV model parameters obtained from the exact inference (black line) and approximate inference methods (ABC SMC, PMMH). The estimates shown in the first column are based on the ABC SMC and the second column are based on the PMMH algorithm. The first row represents the parameter c_1, the second row represents the parameter c_2, and the third row represents the parameter c_3.

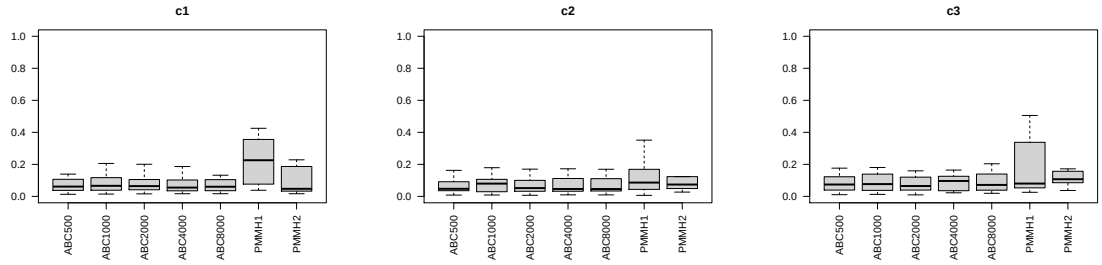


Figure 5.20: The box plot displays the distribution of the resulting JSD divergence values over several independent runs of ABC SMC and both PMMH algorithms over different synthetic data sets (presented in Figure 5.1) which calculate the similarity between the approximate and the exact probability distributions for each LV model parameter.

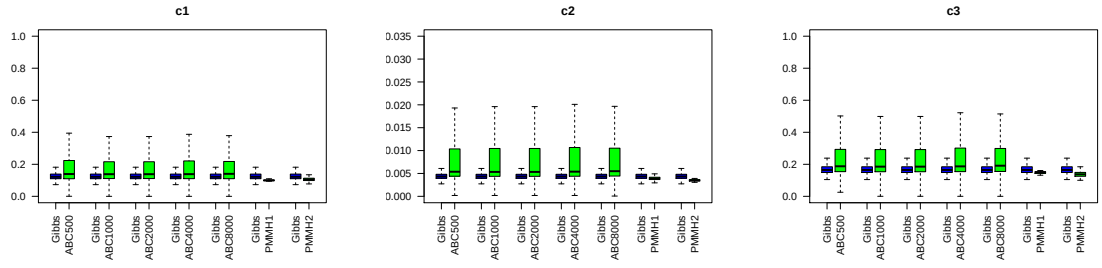


Figure 5.21: Box plots of the exact posterior samples (blue box plot) which are obtained from performing the Gibbs sampler versus the corresponding approximate distributions (green box plot) which are obtained from performing the ABC SMC with five settings of N, where the term ABC500, ABC1000, ABC2000, ABC4000 and ABC8000 means that the ABC SMC algorithm is performed with 500, 1000, 2000, 4000 and 8000 number of particles respectively. The term PMMH1 represent the performance of the PMMH algorithm with the normal proposal while the PMMH2 with log normal proposal density. It is clear that the approximate posterior distributions resulting from the ABC SMC algorithm are widely distributed compared to the exact posterior distributions. While the resulting posterior distributions from both PMMH algorithms are tightly distributed around the exact posterior mode.

The comparison of these inference methods is presented in Figure 5.18. The exact posterior distributions which are depicted in the red dash line give evidence that the PMMH1 and the PMMH2 methods provide an estimation of the model parameters around the posterior mode.

Both the PMMH1 and the PMMH2 methods seem to give a more accurate estimation of the exact posterior compared to the ABC SMC method. However, the posterior distributions resulting from the PMMH1 and the PMMH2 algorithms only cover a narrow range of parameters. This is due to the Markov property, where the algorithm explores the state space by local moves using the previous state. When the posterior is assigned in the region of a large probability mass, the movement will be made in this area more likely and leads to high autcorrelation between samples. Consequently, the tail of the posterior cannot be explored well. This is problematic if the exact posterior is multi-modal or has a heavier tail as the tail is poorly explored.

The poor exploration of the tail might be caused by the variance of the likelihood estimator (Wilkinson, 2011). Wilkinson (2011) states that the variance of the log likelihood estimator is not constant, but it varies significantly as the parameter value varies. The variance becomes small in the vicinity of the maximum likelihood and large in other parts of the parameter space as Wilkinson (2011) illustrated in Figure 5.23. This might lead to the poor exploration of the tails of the posterior density as it can be observed in Figure 5.18. Q-Q plots are the additional graphical way that can be used to visualise the difference between the approximate and the exact posterior distributions as shown in Figure 5.19. It can be clearly seen from the first column in Figure 5.19 that both the approximate distributions obtained from the ABC SMC sampler (blue) and the exact (black) posteriors are right skewed distribution and the difference between these distributions is gradually increase towards the tail as the approximate posterior densities have longer right tail (positively skewed) than the exact distributions. The second column in Figure 5.19 shows that the posterior distributions obtained from the PMMH algorithm (red) do not have probability mass in the right tail while the exact distortions have a probability mass in the right tail. Most of the probability mass is allocated in the high posterior mode which means that the PMMH algorithm has not fully explored the lower part of the parameter state as we expected. Several practical attempts to solve this problem and to improve the performance of the PMMH algorithm will be discussed in section 5.7.1.

The ABC SMC algorithm seems to provide a less confident approximation compared to the exact posterior distributions. In addition, the resulting posterior distributions

obtained from the ABC SMC have the same support as the exact posterior distributions. It also covers the tails of the distributions as shown in Figure 5.18.

The JSD divergence test is used to measure the dissimilarity between the exact and the approximate posterior distributions. Figure 5.20 provides a result from carrying experiments that aim to measure the discrepancy between the exact posterior with its corresponding approximation posterior for each LV parameter. This experiment involves the resulting posterior distributions from the ABC SMC algorithm with five different numbers of particles and both of the PMMH algorithms. According to Figure 5.20, it can be seen that the ABC SMC algorithm with different values of N has a lower divergence compared to both PMMH algorithms. This is caused by the poor exploration of the state space as the tail of the posterior distribution is not fully explored by the PMMH algorithm. The JSD divergence for the ABC SMC with different N is relatively similar.

Figure 5.21 visualises the difference between the resulting approximate and exact posterior samples for each individual algorithm for all model parameters given a single data set (Experiment1). The resulting approximate posterior densities from the ABC SMC algorithms seem to be wider (over dispersed) when is compared to the exact posterior distribution, while the PMMH1 and PMMH2 are much tighter and concentrate around the high posterior mode compared to the exact posterior distribution.

The ABC SMC has a much smaller computational time. For example, the single run of the Gibbs sampler has a running time approximately of 7 hours, while the single run of the ABC SMC algorithm is much quicker and takes approximately 22 minutes until the stopping criteria (target tolerance) is achieved. The PMMH algorithm requires approximately between 16 to 72 hours to terminate after the potential scale reduction factor became $\hat{R} < 1.1$.

The main aim of performing all these experiments is to compare the accuracy and computational time of the four inference algorithms. Figure 5.22 summarises the comparison for all experiments. According to Figure 5.22, the ABC SMC algorithm with different settings of N has a low computational cost compared to the PMMH algorithms and the Gibbs sampler algorithm. Also, it can be seen that the computational cost of the ABC SMC algorithm increases as N increases, while the accuracy remains approximately the same as is shown in Figure 5.22.

The PMMH1 and PMMH2 algorithms are much more expensive compared to the ABC SMC algorithm and the Gibbs sampler. The main computational cost in the

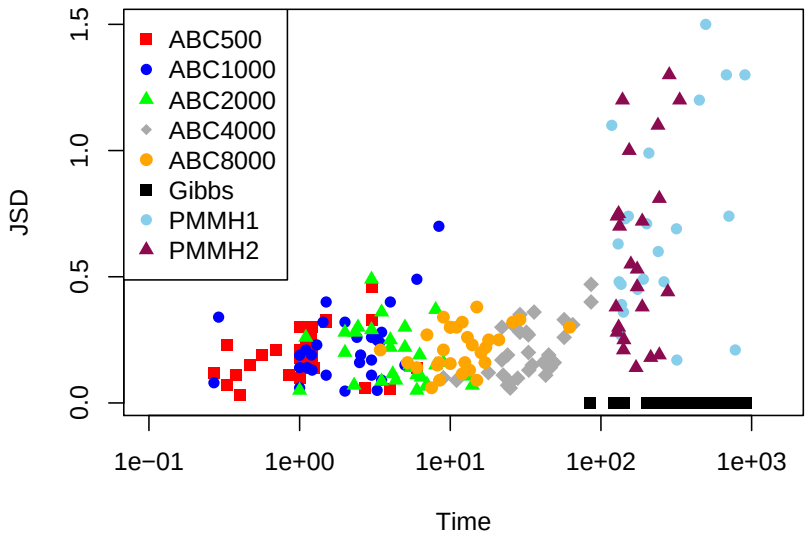


Figure 5.22: Accuracy of estimating the posterior density which is quantified through the JSD divergence between the exact posterior distributions (Gibbs sampler) with its corresponding approximate posterior distributions versus the computational time for each algorithm across several runs.

PMMH1 and PMMH2 is gained from the bootstrap particle filter which is used to estimate the likelihood at each iteration.

In conclusion, the Gibbs sampler gives the exact result, does not require special tuning such as selecting the proposal density. It also converges to the true distribution very fast and is computationally cheap compared to the PMMH algorithm. It is more expensive compared to the ABC SMC approximation approach. The large state space size can limit the computational feasibility of the Gibbs sampler method. Also, the Gibbs sampler cannot benefit from the parallel hardware. This empirical comparison suggests using the Gibbs sampler based on truncation if it is available. When the Gibbs sampler cannot be employed, the ABC SMC would be preferable over the PMMH algorithm. This is because the PMMH algorithm is expensive and exhibits a higher divergence compared to the ABC SMC algorithm. This divergence results from losing the tail of the posterior distribution which is completely ignored. This problem can be a consequence of the variance of the likelihood estimator.

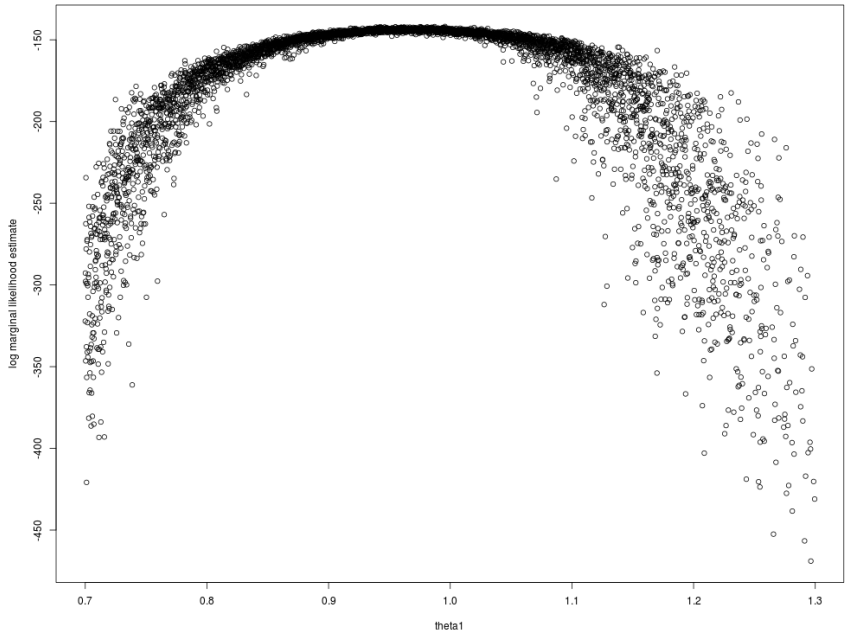


Figure 5.23: The figure shows how the variance of the log likelihood estimator obtained from the particle filter that is used in the PMMH algorithm to estimate the likelihood varies as the parameter values vary, where the true value in this example is 1 (Wilkinson, 2011).

5.7.1 Improving the Performance of the PMMH Algorithm

This section considers some possible solutions that can overcome the problem of the poor exploration of the parameter space. We will investigate the increasing of the number of particles used in the bootstrap particle filter algorithm, increasing the length of iterations in the PMMH algorithm and the population MCMC method and how these attempts can improve the exploration of the parameter state.

It is difficult to tune the number of particles that are used in the bootstrap particle filter to estimate the likelihood. We know that the variance of the likelihood estimator is not constant, it changes as the parameter value changes (see Figure 5.23), but in general, the variance will be decreased as the number of particles N increase. In an attempt to obtain a better exploration of the parameter space, the number of particles will be increased from $N = 300$ to $N = 500$.

The resulting posterior distributions from performing the PMMH1 with $N = 500$ that used in the particle filter to estimate the likelihood are depicted in Figure 5.24.

We note that the resulting posterior distributions obtained from the PMMH1 with $N = 500$ are better as it includes a larger range of the parameters and the shapes

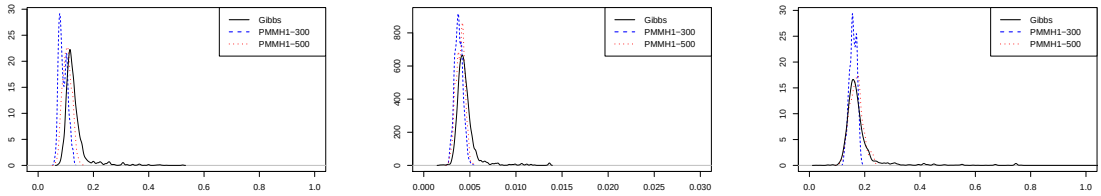


Figure 5.24: The resulting posterior densities from performing the PMMH1 algorithm two times with different tuning of the number of particles that used in particle filter algorithm to obtain the likelihood estimator. It can be seen that the posterior distributions obtained from the PMMH1 with $N = 500$ (red) are more similar to the exact posterior densities than the original posterior densities (blue), in particular for parameter c_3.

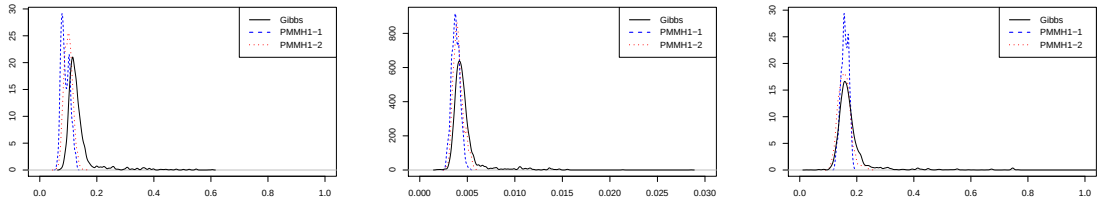


Figure 5.25: The resulting posterior distributions from applying the PMMH algorithm with a larger number of iterations in the burn in stage 3000 and stationary stage 5000 (red). It can be noticed that the tail of the posterior distributions of the parameters c_2 and c_3 include more parameter values compared to the original one (blue).

are more similar to the exact posteriors than the original posterior distributions obtained from the PMMH1 with $N = 300$. This improvement is expected as the variance of the likelihood estimator decreases, but it does not solve the problem completely as it can be clearly seen in Figure 5.24 that there is no probability mass on the tails for the parameter c_1 and c_2.

The result also shows a better exploration of the posterior distribution for the parameter c_3 as there is a probability mass on the right tail, but still does not fully explore the tail of the distribution. In terms of efficiency, the effective sample size ESS of the PMMH1 with $N = 300$ was 125.21 and it increases to be $ESS = 208.81$ when the PMMH1 algorithm is performed with $N = 500$.

The second attempt is to increase the length of the iterations of the adaptive (burn in) and the stationary stages in the PMMH algorithm to give the Markov chain a

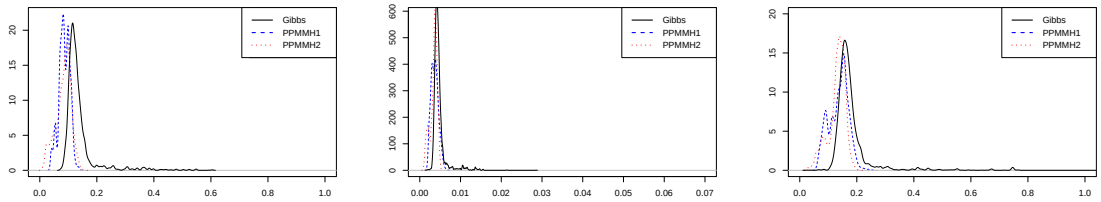
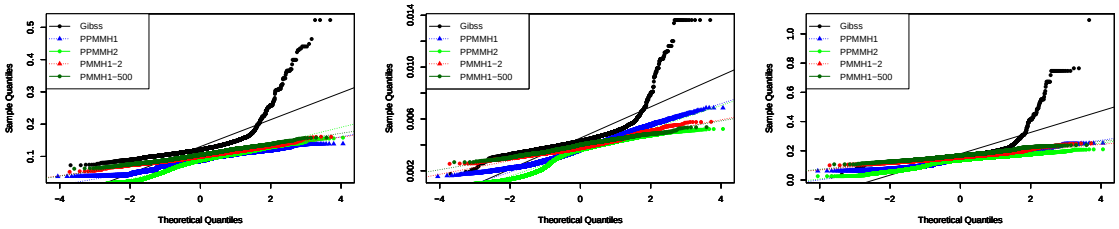


Figure 5.26: The resulting posterior distributions from combining 7 parallel runs of the PMMH1 algorithm based on the normal proposal (blue) and 7 parallel runs of the PMMH2 algorithm based on the log normal proposal (red).



(a)c_3 (b)c_2 (c)c_3

Figure 5.27: Q-Q plots of the estimation of the LV model parameter obtained from four PMMH algorithms and corresponding exact posteriors (Gibbs sampler). It is clear that the tails have not been fully explored by the PMMH algorithm.

chance to visit all areas of the posterior distribution. We, therefore, run the PMMH algorithm with the length of iterations 3000 in burn in stage instead of 2000 and the length of iterations 5000 instead of 2000 in the stationary stage.

We have found that increasing the length of iterations helps to improve the efficiency of the algorithm as a better $ESS = 278.38$ is obtained. It seems that performing the PMMH algorithm with a larger length of iterations would result in a small improvement of the exploration of the parameter space for the parameter c_2 and c_3 as shown in Figure 5.25.

The other possible way that can address the problem of exploring the parameter space is to use multiple MCMC chains and combine the independently produced result. We ran seven parallel Markov chains and combined the samples produced in the stationary stage of PMMH algorithm. The results are shown in Figure 5.26.

The resulting posterior densities obtained by combining seven MCMC samples cover more range of the parameters compared to the original PMMH1 algorithm as shown in Figure 5.26. Figure 5.26 also shows that both the PMMH1 and PMMH2 converge to the same stationary distribution. However, we still observe that the right tail of the posterior distribution has not been sufficiently explored.

All results from the possible attempts to improve the performance of the PMMH algorithm are visualised in Q-Q plots in Figure 5.27. Figure 5.27 demonstrates that MCMC estimates of the posterior distributions from the new attempts underrepresent the right tail of the exact posterior distribution.

We have identified that a common problem in the PMMH algorithm is the poor exploration of the tail of the posterior distribution. The resulting Markov chain from the PMMH algorithm is strongly autocorrelated, which corresponds to a trapped chain in a high posterior mode in which the variance of the likelihood estimator is small. Hence, the PMMH algorithm produces an underspread estimate of the posterior distribution. We also have found that neither increases the number of particles N nor the number of iterations can solve the problem completely but they can improve the exploration of the parameter space.

Golightly and Wilkinson (2011) state that the mixing of the PMMH algorithm relies on the variance of the likelihood estimator. The variability of the likelihood estimator relies on the efficiency of the particle filter algorithm which depends on the measurement error. In the case of using forward simulation from the model as a proposal in the particle filter method (the bootstrap particle filter), the method can break as the measurement error tends to zero. In the case of low measurement error,

the probability of hitting the data and hence accepting the proposed value will be very small, and therefore lead to poor mixing of the MCMC. Golightly and Wilkinson (2011) suggest a strategy of using a diffusion process to improve the performance of PMCMC method in the case of low measurement error. This method is based on sampling diffusion bridges to perform inference for the parameters of a diffusion approximation as described in section 2.6.2. This approach is known as marginal Metropolis-Hastings using diffusion bridges. More details and applications of this method can be found in (Golightly and Wilkinson, 2011).

It would be beneficial to investigate the increase of both the number of particles used in the particle filter and the number of iterations used in the PMMH algorithm as suggested by Doucet et al. (2015). However, this requires a huge computational time because each iteration of the PMMH algorithm becomes very slow. Despite the fact that the PMMH algorithm can benefit from parallel computing (using multiple resources), however, we do not currently have resources available to explore this further.

5.8 Application of Approximate Bayesian Computation on a Larger Data Set

This section considers an application of the ABC SMC method on a larger data set to ensure the applicability of the ABC SMC scheme. The implementation of this method depends on an algorithmic setting chosen by the user (e.g. the number of particles, the adaptive tolerance schedule and the number of stages). The main aim of this section is to investigate a different tuning of the algorithmic parameter and its effect on the quality of the approximation and the computational time. These experiments give an idea of how the different tunings of the algorithmic parameters can influence the performance ABC SMC algorithm.

In order to accept the proposed parameter in the ABC SMC algorithm, the distance between both the simulated and the observed data sets must be less than a defined tolerance at each stage. The sequence of the tolerances are chosen by the user and should be chosen as small as possible. However, it is generally difficult to have the observed data exactly match the simulated data from a model, especially in stochastic systems. Therefore, it would be useful to perform a pilot run as guidance to specify the target tolerance for data set presented in section 4.5.4.

In our experiment, the target tolerance is defined as the first quantile of the range of

the distances and is set as a criterion (target tolerance) to terminate the algorithm. The number of simulation traces is chosen as ($i = 2000$) based on experiments carried out in section 5.5.

We implement the ABC SMC algorithm described in chapter 4 which allows us to obtain an approximate posterior for the LV model parameters. As we are interested in noticing how the approximation of posterior relies on tuning the number of particles N, we perform ABC SMC with the different choices of the number of particles, that are $N = (500, 1000, 2000, 4000, 8000)$ for each run.

The implementation of the ABC SMC algorithm requires the user to define the number of stages to perform the algorithm. Many numbers of steps can result in a more accurate approximation as it has more chance to explore the state space. It can be computationally expensive. Conversely, a small number of steps is computationally cheaper but can ignore some part of parameter state space. In our experiments, instead of using a certain number of stages, the number of stages are automatically tuned which will be increased until the target tolerance is achieved. This technique can save computational effort.

The adaptive tolerance rate is chosen to be $\alpha^{th} = 0.5$, where the sequence of the tolerance schedule is decreased by 50% of the discrepancy between the observed and the simulate data set until the target tolerance (691.1) is achieved. The perturbation kernel is chosen to be a componentwise normal distribution. The prior densities are assumed to be uniform distributions. The choice of the prior densities is made based on the previous suggestion in the literature (Toni et al., 2009), such that:

$$\pi(c_1) \sim \text{Uniform}(0, 1)$$

$$\pi(c_2) \sim \text{Uniform}(0, 0.01)$$

$$\pi(c_3) \sim \text{Uniform}(0, 1).$$

We consider these settings and perform Bayesian inference using the ABC SMC algorithm given the data sets depicted in Figure 5.2, Figure 5.3 and Figure 5.4 in section 5.2.

The stage's number required to achieve the predefined target tolerance for each independent run using five different settings of N is different. It also varies from data are set to another as shown in Table 5.3. It can be seen that for each single data set, the required number of steps to reach the target tolerance does not vary significantly as varying the number of particles N. However, the time has changed significantly as

Table 5.3: Summary of performing ABC SMC on synthetic data (presented in 5.2 (a)).

N	**Number of steps**	**Goal achieved**	**Time (in minutes)**
500	24	674.2	14.16
1000	21	687.56	32.92
2000	21	654.46	86.81
4000	21	658.88	172.85
8000	21	656.12	376.06

the number of particles N varies as is shown in Table 5.3. A large number of particles requires more computational time to achieve the target. The cost of running time is not only caused by the simulation from the model, but also is impacted by the size of the population (Number of particles).The proposal distribution density (i.e the density of the perturbation kernel) needs to be evaluated for each proposed particle as a mixture of normal distributions, with the same number of components as the number of particles employed. The cost of doing it at each stage is therefore $O(N^2)$.

The resulting approximated posterior densities for all unknown model parameters given four different data sets are shown in Figure 5.28. It can be noticed that making use of only 500 particle number still provides a reasonable approximation to the posterior density for the parameters c_1 and c_3. However, for the second parameter c_2, there is a noticeably different between the distributions with higher numbers of particles and the distributions with higher numbers of particles as shown in the second column of Figure 5.28.

For a theoretical perspective, it is recommended to use a large number of particles when performing ABC to ensure the diversity of the sampling path, a better performance, and a more accurate approximation.

However, the experiments we have carried out illustrate that using a small number of particles has a small effect on the approximation's accuracy for parameter c_1 and c_3 while it has huge effect on approximating the parameter c_2 (see Figure 5.28). This is because the ESS for all algorithms remains high at most stages (see Figure 5.29). When ESS falls below the defined threshold $\frac{N}{2}$, the resampling step is performed. In our implementation, the ESS dropped once when $N = 500$. A similar case occurs when ABC SMC is applied with $N = 1000$, $N = 4000$, as is shown in Figure 5.29. It can be seen that ESS is maintained to be high at all stages when the algorithm uses the number of particles $N = 8000$. In our experiments, when $N = 2000$,

Table 5.4: Summaries of the experiments of running the ABC SMC under a different tolerance schedules settings (based on three different quantiles).

Adaptive tolerance rate	Number of step	Time (in minutes)
$\alpha^{th} = 0.3$	12	3.05 hours
$\alpha^{th} = 0.5$	21	6.2 hours
$\alpha^{th} = 0.75$	47	13 hours
$\alpha^{th} = 0.3$	14	36.6 mins
$\alpha^{th} = 0.5$	23	42.5 mins
$\alpha^{th} = 0.75$	53	1.53 hours
$\alpha^{th} = 0.3$	12	8.9 hours
$\alpha^{th} = 0.5$	21	13.9 hours
$\alpha^{th} = 0.75$	74	23.3 hours

the resampling step is carried out twice. This means that the resampling step in the ABC SMC algorithm will not add additional computational cost as it is not performed at each stage.

5.8.1 Further Analysis

We further analyse the performance of the ABC SMC sampler under different choices of the adaptive tolerance schedule. Therefore, the ABC SMC algorithm will be repeated with the settings presented in section 5.8 and the adaptive tolerance schedule α^{th} will be varied. The target tolerance is the final tolerance of the adaptive tolerance schedule.

In this experiment, we have found that the performance of the ABC SMC sampler with three different choices of the adaptive tolerance schedule provides a similar approximation to the posterior distributions (see Figure 5.30). The required stage's number to reach the target tolerance for each independent run is compared and summarised in Table 5.4. From the comparisons carried out, we have found that the ABC SMC algorithm with the adaptive decreasing tolerance schedule based on $\alpha^{th} = 0.75$ requires 47 steps to achieve the target tolerance. While the algorithm requires less numbers of stages (21) when $\alpha^{th} = 0.5$. This number of stages is reduced to 12 when $\alpha^{th} = 0.3$ as is shown in Table 5.4. The performance of ABC SMC with a lower α^{th} requires a few numbers of stages to achieve the target tolerance and thus the computational time will be decreased.

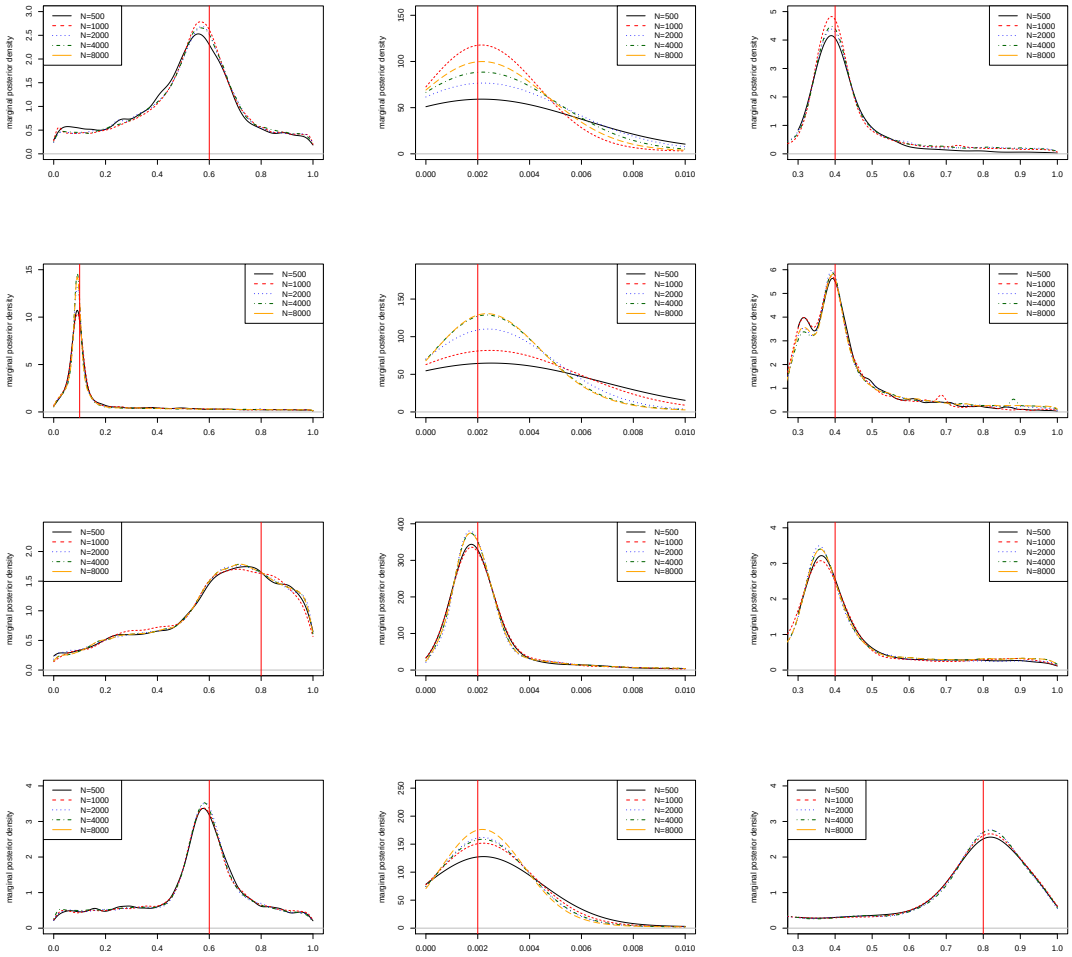


Figure 5.28: The figure shows the approximate posterior densities for the LV model parameters, the first row represents the inference given the data set presented in Figure 5.2 (a), the second row represents the inference given the data set presented in Figure 5.2 (b), the third row represents the inference given the data set presented in Figure 5.2(c), and the fourth row represents the inference given the data set presented in Figure 5.4(a). Each column represents specific model parameter that has been inferred using the ABC SMC algorithm, column one represents the posterior densities of c_1, column two represents the posterior densities of c_2, and column three represents the posterior densities of c_3.

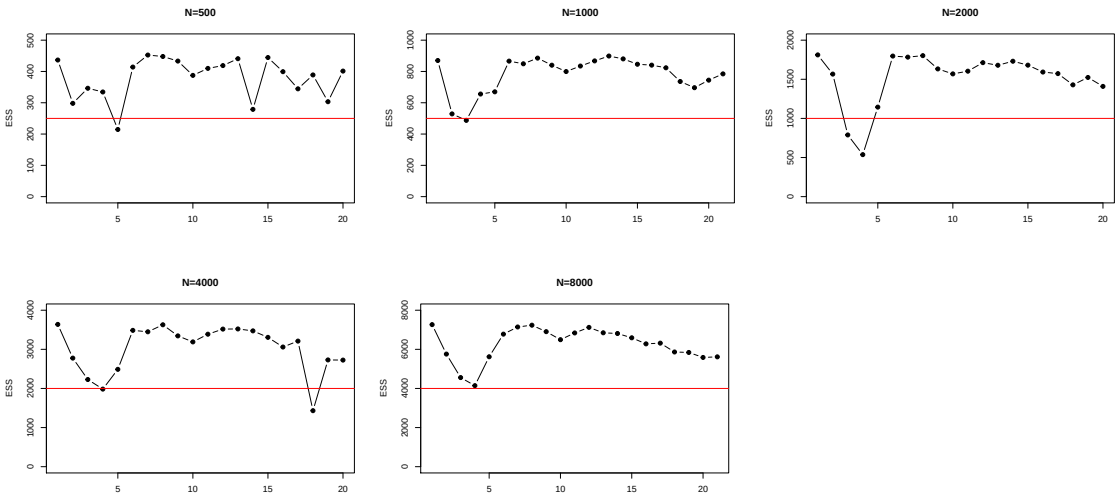


Figure 5.29: The plot represents the ESS values from performing the ABC SMC algorithm, given the synthetic data set presented in Figure 5.2 (a). The ESS values for all runs have not dropped to very small value near to zero.

A comparative investigation of choosing the adaptive decreasing tolerance schedule based on the different settings of α^{th} suggests that performing algorithm with $\alpha^{th} = 0.3$ converges to the approximate posterior around the true parameter value which is far from the prior distribution after a few stages. However, this quick convergence can possibly ignore some part of the entire parameter state space. It can be seen from Figure 5.30 that the approximate posteriors resulting from using $\alpha^{th} = 0.75$, exploring a higher region of the parameter space, especially for the parameter c_2. Hence, there is some benefit of choosing $\alpha^{th} = 0.75$ compared with $\alpha^{th} = 0.3$ and $\alpha^{th} = 0.5$. This is because performing the algorithm a larger number of stages can result in a better exploration of the parameter space.

When the computational time is essential (e.g. large data), $\alpha^{th} = 0.5$ can be a suitable choice in practice as it can provide a similar approximation in less computational time. (See Appendix C and D for more experiments regarding the further implementation of the ABC SMC.)

5.9 Summary

This chapter considered the LV system, which is modelled by a CTMC. This model is governed by the reaction rate parameters. Each parameter value results in different behaviour of the dynamic of the system. Several data sets are simulated from the model at different parameter values to show how the model behaviour varies

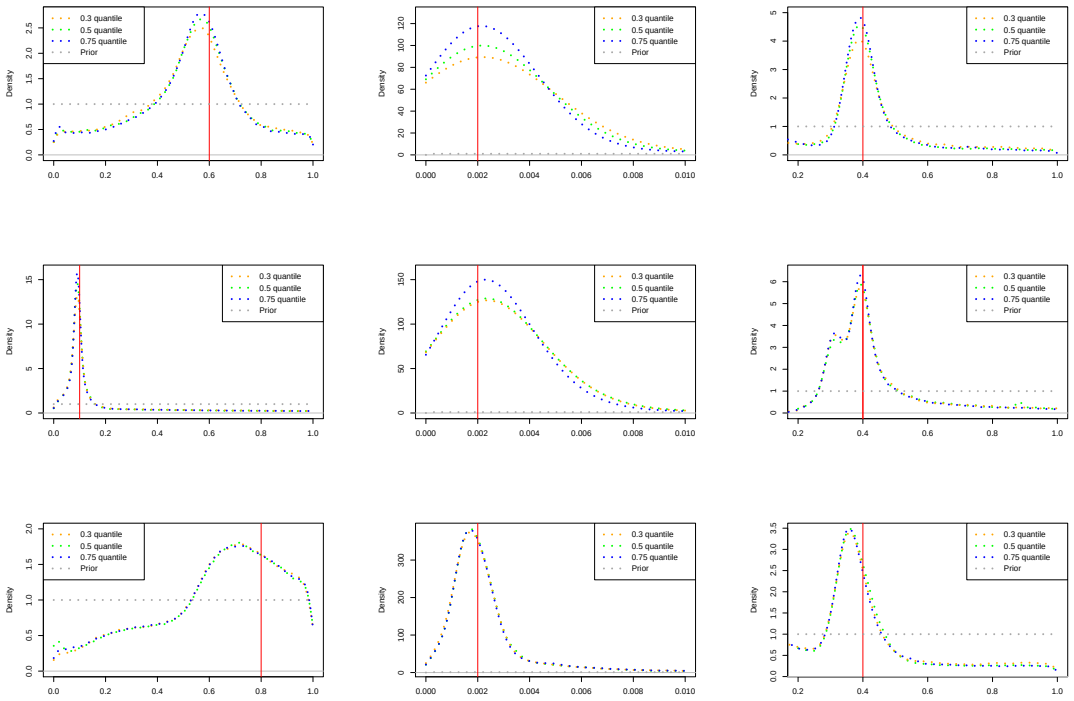


Figure 5.30: The figure shows the approximate posterior densities which are resulting from applying the ABC SMC algorithm with three different choices of the adaptive tolerance schedule. The first row represents the inference given the data set presented in Figure 5.2 (a), the second row represents the inference given the data set presented in Figure 5.2 (b), the third row represents the inference given the data set presented in Figure 5.2(c). Each column represents specific model parameter that has been inferred using the ABC SMC algorithm, column one represents the posterior densities of c_1, column two represents the posterior densities of c_2, and column three represents the posterior densities of c_3.

as the reaction rates vary as shown in section 5.2. The likelihood is, in general, intractable for some stochastic models. The intractability of the likelihood makes the direct inference prohibitive and motivates us to consider different inference methods which are based on simulation and approximation, such as ABC SMC and PMMH approaches. A novel pseudo-marginal approach was proposed recently by Georgoulas et al. (2017) and is considered as an exact inference method. This method addresses the issue of Bayesian parameter inference in a case when the likelihood is intractable. In this thesis, this method is used to quantify the accuracy of other approximation methods.

The effectivness of the Bayesian inference approaches described in sections 3 and 4 is demonstrated on the LV model. Section 5.4 demonstrates that the pseudo marginal Gibbs sampler allowed for exact inference for the LV model. The ABC SMC was performed on various data sets. The Gibbs sampler method is used to check the accuracy of the ABC SMC algorithm. All the experiments we applied in section 5.4 show that the ABC SMC algorithm gave a reasonable approximation to the exact marginal posterior resulting from the Gibbs sampler. Both methods are easy to design and do not require extensive special tuning.

A further analysis regarding the ABC SMC algorithm was carried out in section 5.5. The analysis investigates the impact of the different ways of estimating the target tolerance on the accuracy and computational cost. The effect of the number of simulations used to estimate the target tolerance is also considered. The results from this analysis suggest that more computational effort is required to obtain a more accurate approximation.

The chapter continues with the application of the PMMH algorithm to the LV model in section 5.6. The posterior estimations from the PMMH with different choices of proposal density converge to the exact posterior in all experiments. The efficiency of this algorithm relies on the number of particles to estimate the likelihood at each iteration. It is important to balance the number of iterations and the number of particles to obtain an efficient estimation in reasonable computational time. Otherwise, PMMH can perform poorly, and this can be problematic in practice.

The major concern with the PMMH is that the chain can get stuck in a specific area of the state space. This results in a bad mixing and, hence, fail to converge to its stationary distribution. In addition, the local extreme (tail) can be poorly explored, and this is caused due to the Markov property. In our application, it is clear that the tail of the estimated posterior distributions is lost. It appears from measuring the difference between the approximate and the exact posteriors using the JSD

divergence test that some information is lost due to this problem. This highlights the requirement for investigating a different proposal type that can overcome such a problem. The Gibbs sampler and ABC SMC do not suffer from being trapped in a particular region in the state space.

An extensive comparison between the four inference approaches was carried out. To obtain a fair comparison, each method was run several times. The results are visualised in Figure 5.22 in section 5.7. It can be seen that the ABC SMC with various numbers of particles has the best performance in term of computational effort. In terms of accuracy, the Gibbs sampler approach provides the exact estimation of the LV model parameter. The ABC SMC with different numbers of particles has a small divergence compared to both PMMH. This is due to the fact that the ABC SMC covers a wide range of parameter. Yet, the Gibbs sampler method cannot be applied to many models, especially when a model involves large counts of species, which makes the computational time heavy.

For many interesting, challenging models, the PMMH approach is computationally infeasible. This can limit the use of the PMMH in practice. The PMMH algorithm should be designed carefully where, in case of using a small number of particles, a short length of chain, or selecting an inefficient proposal, the algorithm performs poorly. In addition, the performance of the PMMH algorithm can be influenced by the data settings (e.g., the noise level) which are not explored in this thesis (we only explore the algorithmic settings). In particular, the performance of the PMMH algorithm can be improved when the measurement error is large (Golightly and Wilkinson, 2011). Therefore, it would be useful to investigate the performance of the PMMH algorithm with different noise levels.

Besides, this gives an advantage to the ABC SMC, which provides a reasonable approximation to the exact posterior. The ABC SMC is also easy to apply and its efficiency relies on the choice of the target tolerance. The ABC SMC permits a tractable computational inference compared to the PMMH method, which makes it applicable to a challenging problem.

Chapter 6

Application to Repressilator System

This chapter aims to demonstrate the performance of the considered inference methods on a more complex example - a stochastic model of a biochemical network known as the Repressilator (Elowitz and Leibler, 2000). This model describes behaviour of a synthetic biological network implemented by Elowitz and Leibler (2000). This model is more challenging than the LV model considered in the previous chapter, as it considers the behaviour of a larger number of biochemical species, has a slightly larger parameter space, and exhibits a greater variation of possible stochastic behaviours. We employ CTMC to model stochastic interactions performing parameter inference for this model using the methods considered in this thesis. The greater complexity of this model in comparison to the LV model results in a significantly larger state space of the CTMC, and therefore limits the analytical analysis of this model using explicit methods, such as solving the CME.

In this chapter, we consider performing parameter inference using synthetic data sets first, and once the adequate performance of inference algorithms is achieved, we move on to performing the most efficient algorithm for analysis of real data.

The explicit inference for this model is feasible only for some very limited cases where the state space is artificially restricted. Solving the CME in such cases is still problematic, however, the likelihood can be evaluated using the uniformisation method (see section 2.6.1), which converts the CTMC model to a DTMC that has the same behaviour on the set of the observed time points. We exploit this option to obtain the "gold standard" results for our simulated data study, against which the approximate posterior is compared.

Unfortunately, even the uniformisation becomes unfeasible once we consider the CTMC with realistic scales. Therefore, we pilot approximate inference algorithms

on a simple case where explicit results are known and then employ these approximate methods for at the larger study on a realistic scale. We consider both the PMMH and the ABC approach to approximate inference. We investigate different choices of the algorithm parameters in the simulation study, compare the obtained approximate posterior distributions to the exact posterior, and compare the computational time required to obtain each result. Once the best performing approximate method is selected, we apply this method to a realistically scaled example using three different scales of the model.

6.1 Repressilator System

The Repressilator is the first synthetic genetic regulatory network introduced by (Elowitz and Leibler, 2000). It involves three genes denoted (G_1, G_2, G_3) with their corresponding proteins (p_1, p_2, p_3). In this network, each gene can repress (inhibit the expression) of the next gene and consequently affect the production of the corresponding protein. The three proteins are $LacI$, $TetR$, and cl which are expressed by their corresponding genes $TetR$, obtained from tetracycline resistance transposon $TetR$, cl from cl and $LacI$ from $E.coli$. These genes are connected in a feedback loop as shown in Figure 6.1. The first protein $LacI$ inhibits the transcription of the second gene $TetR$. Likewise, the protein product from $TetR$ inhibits the expression of the gene cl. Then, the cl protein inhibits $LacI$ expression, thus completing the negative feedback cycle. The green fluorescent protein (gfp) is the fourth protein in the system which is known as a reporter as shown in Figure 6.1 (right).

The original model for the Repressilator is described by (Elowitz and Leibler, 2000) and the concentration of each species is described by ordinary differential equations (ODEs):

$$\frac{dm_i}{dt} = -m_i + \frac{\alpha}{1 + (p_j)^n} + \alpha_0 \tag{6.1}$$

$$\frac{dp_i}{dt} = -\beta(p_i - m_i), \tag{6.2}$$

where $i = TetR, LacI, cl, gfp$ and $j = LacI, cl, TetR, gfp$. So, (p_i) are proportional to the concentration of the three proteins $TetR$, cl and $LacI$ and (m_i) are the concentrations of mRNA corresponding to the gene expression products. The negative feedback loop leads to oscillation in the system's behaviour.

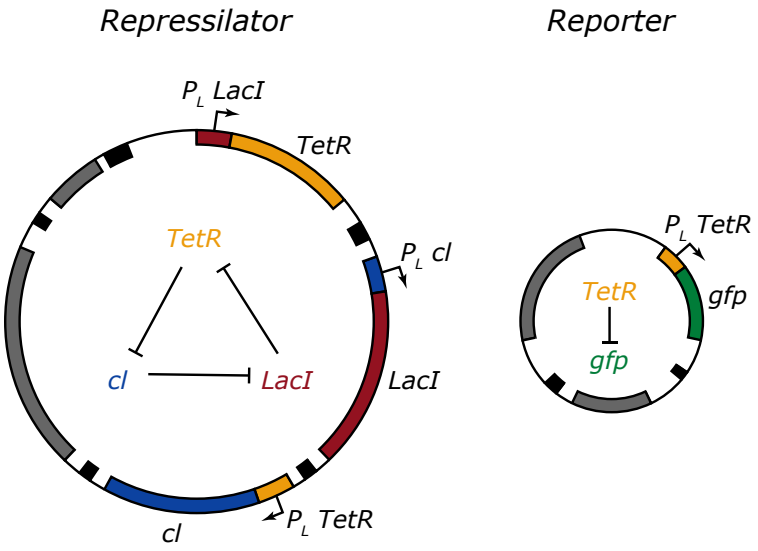


Figure 6.1: The original Repressilator network.

The dynamic behaviour of the Repressilator system is changing over time. An ordinary differential equation (ODE) is often used to describe the dynamical behaviour of the system and its function. However, modelling a biological system using ODEs does not account for the natural randomness and stochastic behaviour of a biological system. In such cases, using a suitable model to account for the stochasticity and uncertainty is crucial.

A CTMC is suitable to model a Repressilator system with discrete events over continuous time, as it is based on stochastic kinetics. We begin with a definition of the kinetic reaction equations for the Repressilator system, which describe possible transitions in the system and interaction between the system's components. Since the Repressilator system can be considered as a biochemical stochastic network, it can, therefore, be characterised by a set of reaction equations, which are introduced in the next section.

6.2 Model Reactions

For stochastic kinetics, a reaction in a biological network can occur when molecules collide, and the probability that a specific reaction i occurs at a specific time interval dt is given by the hazard function $h_i(x, \theta)$ (as defined in section 2.5.1). This function depends on the current state of the system x, which represents the concentration level of mRNAs and proteins and some rate θ.

All three proteins and mRNAs can participate in model reactions, which occur over time. There are three main reactions in this network: the transcription of a gene producing mRNA with inhibition by any protein, the translation of mRNA resulting in protein production and the degradation of both, mRNA and proteins. The reaction rate of the model can be derived from (6.1) and (6.2), each equation including positive and negative terms, the positive one representing the production rate of mRNA ($\frac{\alpha}{1+(p_j)^n} + \alpha_0$) while the negative term ($-m_i$) is the decay rate. The degradation rate of a protein is represented by βp_i and βm_i is the production rate for that protein.

We transform the Repressilator deterministic system described in (6.2) and (6.1) into a stochastic model using a set of reaction equations. We assume there are 8 species and also assume the number of copies of genes is constant. Hence, the dynamics of the system can be described by the following chemical reaction scheme:

- Transcription

$$\begin{aligned} R_1 : G_1 \to G_1 + m_1 \quad &\text{with hazard} \quad h_1 = \frac{\alpha}{1+(p_j)^n} + \alpha_0 \\ R_2 : G_2 \to G_2 + m_2 \quad &\text{with hazard} \quad h_2 = \frac{\alpha}{1+(p_j)^n} + \alpha_0 \\ R_3 : G_3 \to G_3 + m_3 \quad &\text{with hazard} \quad h_3 = \frac{\alpha}{1+(p_j)^n} + \alpha_0 \\ R_4 : G_3 \to G_{gf} + m_{gf} \quad &\text{with hazard} \quad h_4 = \frac{\alpha}{1+(p_j)^n} + \alpha_0 \end{aligned} \tag{6.3}$$

- Translation

$$\begin{aligned} R_5 : m_1 \to m_1 + p_1 \quad &\text{with hazard} \quad h_5 = \beta m_1 \\ R_6 : m_2 \to m_2 + p_2 \quad &\text{with hazard} \quad h_6 = \beta m_2 \\ R_7 : m_3 \to m_3 + p_3 \quad &\text{with hazard} \quad h_7 = \beta m_3 \\ R_8 : m_{gf} \to m_{gf} + p_{gf} \quad &\text{with hazard} \quad h_8 = \beta m_{gf} \end{aligned} \tag{6.4}$$

- Degradation

$$\begin{aligned} R_9 : m_1 \to \emptyset \quad &\text{with hazard} \quad h_9 = m_1 \\ R_{10} : m_2 \to \emptyset \quad &\text{with hazard} \quad h_{10} = m_2 \\ R_{11} : m_3 \to \emptyset \quad &\text{with hazard} \quad h_{11} = m_3 \\ R_{12} : m_{gf} \to \emptyset \quad &\text{with hazard} \quad h_{12} = m_{gf} \end{aligned} \tag{6.5}$$

$$\begin{aligned} R_{13} : p_1 \to \emptyset \quad \text{with hazard} \quad h_{13} = \beta p_1 \\ R_{14} : p_2 \to \emptyset \quad \text{with hazard} \quad h_{14} = \beta p_2 \\ R_{15} : p_3 \to \emptyset \quad \text{with hazard} \quad h_{15} = \beta p_3 \\ R_{16} : p_{gf} \to \emptyset \quad \text{with hazard} \quad h_{16} = \beta p_{gf}, \end{aligned} \tag{6.6}$$

where α_0 is the production rate of the mRNAs, α is the main production rate of the mRNAs that may be inhibited, β is the production and degradation rate of the proteins, and n is the Hill coefficient which corresponds to the fraction of the binding site saturated by the protein as a function of protein concentration.

6.3 Simulating CTMC Trajectories

In order to construct a CTMC model for the Repressilator system, it is necessary to define the model's states along with all possible transitions. The Repressilator system is defined as a network of biochemical interactions between species using the previous model equation reactions, described in section 6.2. For the simulation study we assume that the concentrations of all proteins and mRNAs can be observed directly, and therefore we do not consider the gfp reporter. The variables considered for simulation study are m_1, m_2, m_3 for the concentrations of $TetR, LacI, cl$ mRNAs corresponding; and p_1, p_2, p_3 for $TetR, LacI, cl$ protein concentrations. We describe the transitions of this model over a discrete level of concentrations over the species, e.g. "low concentration", "high concentration", considering the number of the discrete concentration levels to be pre-defined (constant) parameter of the model guarding model complexity, e.g. $m_1 \in \{0, 1, 2, 3, 4, 5\}$, and so on.

For simulation study we consider six concentration levels $(0, 1, 2, 3, 4, 5)$ for all the species. Each state of the CTMC model is therefore labelled with a vector $\boldsymbol{x} = (m_1, p_1, m_2, p_2, m_3, p_3)$. The initial state for the CTMC in the simulation study is chosen to be $\boldsymbol{x_0} = (m_1 = 0, p_1 = 1, m_2 = 0, p_2 = 3, m_3 = 0, p_3 = 2)$. We simulate nine synthetic data sets from this model using different values of kinetic parameters, $\alpha, \alpha_0, n, \beta$, for each of them as shown in Table 6.1. The observations are simulated over 61 equally spaced fixed time points $t = (0, 0.5, 1, 1.5, \cdots, 30)$.

This results in a CTMC model with 46656 states, and approximately half a million transitions. Simulation and exact inference for a model of this size are challenging, and it is at the limit of our computing capacity. This is the reason why we cannot

perform inference for larger, realistically scaled studies. Only approximate methods will be considered for larger cases.

Figure 6.2 depicts the simulated data sets. We are using these data to perform parameter inference in section 6.4.

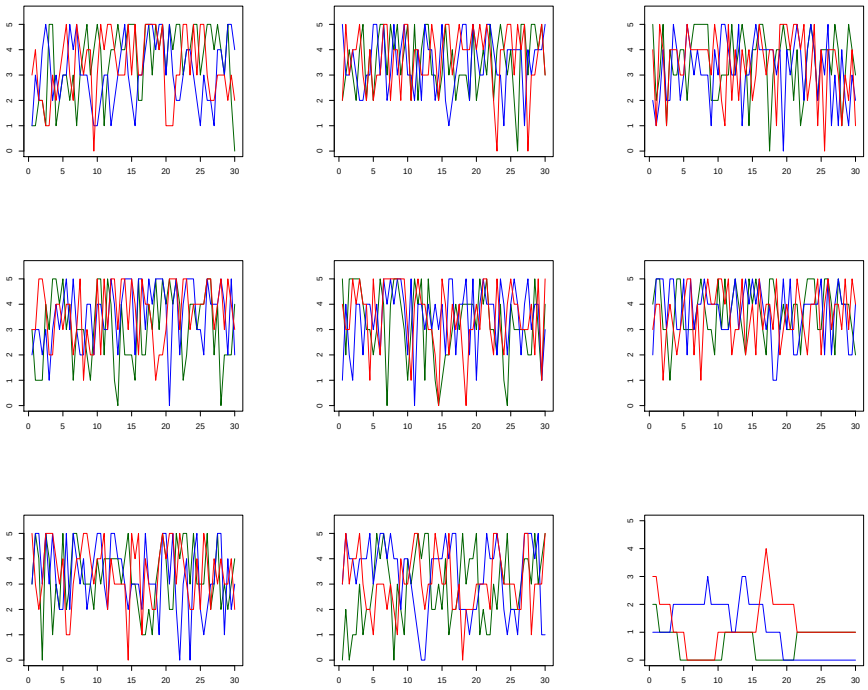


Figure 6.2: An illustration of the nine different synthetic data sets simulated from the stochastic model using different parameter values. Only protein concentrations are plotted as differently coloured lines. mRNAs concentrations were also simulated and used in inference, but they are omitted from these plots to avoid cluttering the visualisation.

6.4 Inference Based on Synthetic Data

In this section, we evaluate approximate inference methods based on synthetic data described in section 6.3. As relatively small for this simulation study, and complete observation of all variables is assumed over discrete time points, we are also able to perform parameter inference exactly, therefore providing the exact posteriors against which to measure the results of approximate inference.

We consider two likelihood-free approaches to approximate inference: the PMMH, as described in section 3.7, and the ABC SMC as described in section 4.4. We perform

Table 6.1: Parameter settings for simulations from the Repressilator model.

Experiments	Parameter setting
Experiment 1	$\theta = \{\alpha = 1000, \alpha_0 = 1, n = 2, \beta = 1\}$
Experiment 2	$\theta = \{\alpha = 1200, \alpha_0 = 1, n = 1, \beta = 3\}$
Experiment 3	$\theta = \{\alpha = 1400, \alpha_0 = 1.5, n = 2, \beta = 4\}$
Experiment 4	$\theta = \{\alpha = 2000, \alpha_0 = 1, n = 2, \beta = 2\}$
Experiment 5	$\theta = \{\alpha = 2200, \alpha_0 = 1, n = 2, \beta = 3\}$
Experiment 6	$\theta = \{\alpha = 500, \alpha_0 = 0.1, n = 2, \beta = 5\}$
Experiment 7	$\theta = \{\alpha = 100, \alpha_0 = 1, n = 2, \beta = 1\}$
Experiment 8	$\theta = \{\alpha = 50, \alpha_0 = .01, n = 2, \beta = 2\}$
Experiment 9	$\theta = \{\alpha = 1, \alpha_0 = 0.001, n = 2, \beta = 0.2\}$

these methods several times using different settings of algorithm tuning parameters. Inference is performed separately on each of the nine data sets simulated in section 6.3 to evaluate their performance.

6.4.1 Exact Inference

In this section, we perform exact inference over one of the synthetic data sets using uniformisation of the CTMC model as described in section 2.6.1. This inference was performed on the computing cluster, and required about 7137 hours (297 days) to complete. As we only have limited access to the computing cluster, we did not have the resources to perform exact inference using all of the data sets. For this reason, the comparison of approximate posteriors to the exact result will be made only for one of the data sets. Other cases will be considering only the approximate posterior.

Computing the transient state distribution (as described in section 2.6.1.) using the uniformisation method, we can define the likelihood as:

$$\pi(y|\theta) = \prod_{i=0}^{60} p(y_i|t_i),$$

where $t_i = 0.5.i$, y_i are the corresponding data, and $p(y_i|t_i)$ is the probability of the CTMC state labelled with y_i.

We selected the following prior distributions for the kinetic parameter of this model based on the previous suggestion in the literature (Toni et al., 2009):

$$\pi(\alpha) \sim \text{Uniform}(0, 2500)$$

$$\pi(\alpha_0) \sim \text{Uniform}(0, 2)$$

$$\pi(n) \sim \text{Uniform}(1, 3)$$

$$\pi(\beta) \sim \text{Uniform}(0, 8).$$

Unlike Toni et al. (2009) we do not consider negative proper support for the kinetic parameters, as this conflicts the physical definition of the kinetic model.

We perform posterior inference using the SIS algorithm described in section 3.2. At every stage of the sequential sampler we are using $N = 1000$ particles, and we consider the following of the intermediate distributions for the SIS sampler:

$$\pi_i(y, \theta) = \pi(y|\theta)^{\tau_i}\pi(\theta), \quad i = 0, \cdots, 35, \quad \tau_i = \left(\frac{i}{35}\right)^4,$$

when $i = 0$, the samples are coming from the prior, and when $i = 35$, the samples are coming from the posterior distribution. The proposal density $q_i(\theta)$ is chosen to be an adaptive normal distribution with variance estimated with a population variance of the previous SIS stage.

The ESS is monitored during the run at each stage to ensure the diversity of the model population of particles. If the ESS falls below $\frac{N}{2} = 500$ samples, we resample the population of particles according to their weights. According to Figure 6.3, we needed to perform such resampling at every stage of the SIS, however, our population never degraded to a complete collapse into a singular point.

Figure 6.4 illustrates the samples at the very first stage, where all the particles are coming from the prior distribution. Figure 6.5 illustrates one of the intermediate distributions (where $i = 19$), and Figure 6.6 depicts the density of the obtained exact posterior distribution.

Note that despite the marginal posterior distribution $\pi(n|y)$ being quite wide, it is highly correlated with parameter α, and jointly they define a reasonably tight posterior ridge.

We use this exact posterior as a target to evaluate approximate posteriors produced by the ABC SMC and the PMMH algorithms.

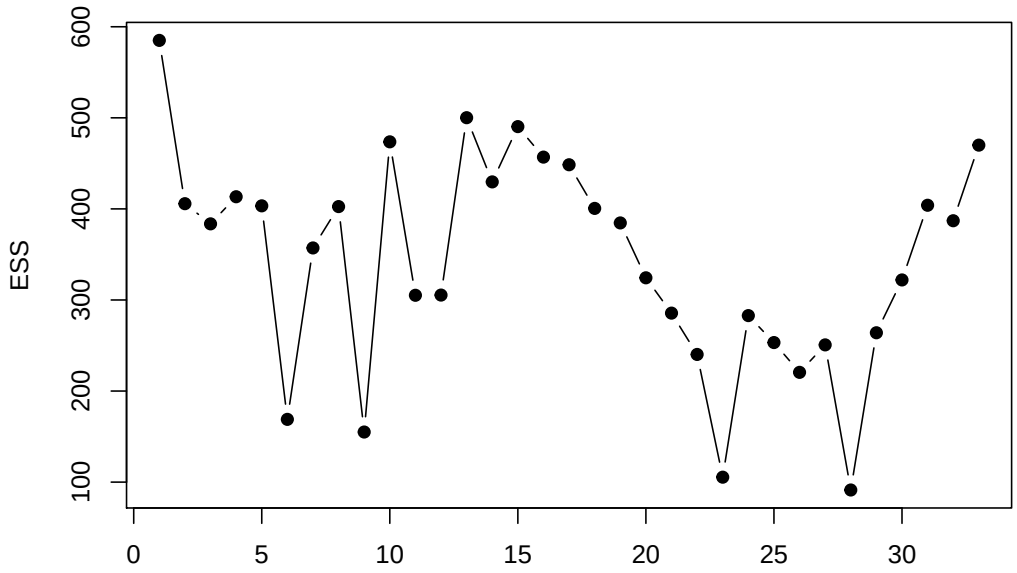


Figure 6.3: The plot shows the ESS which is obtained from performing the SIS algorithm on the first synthetic data set (Experiment 1), the number of SIS stages 35 are represented in X-axis.

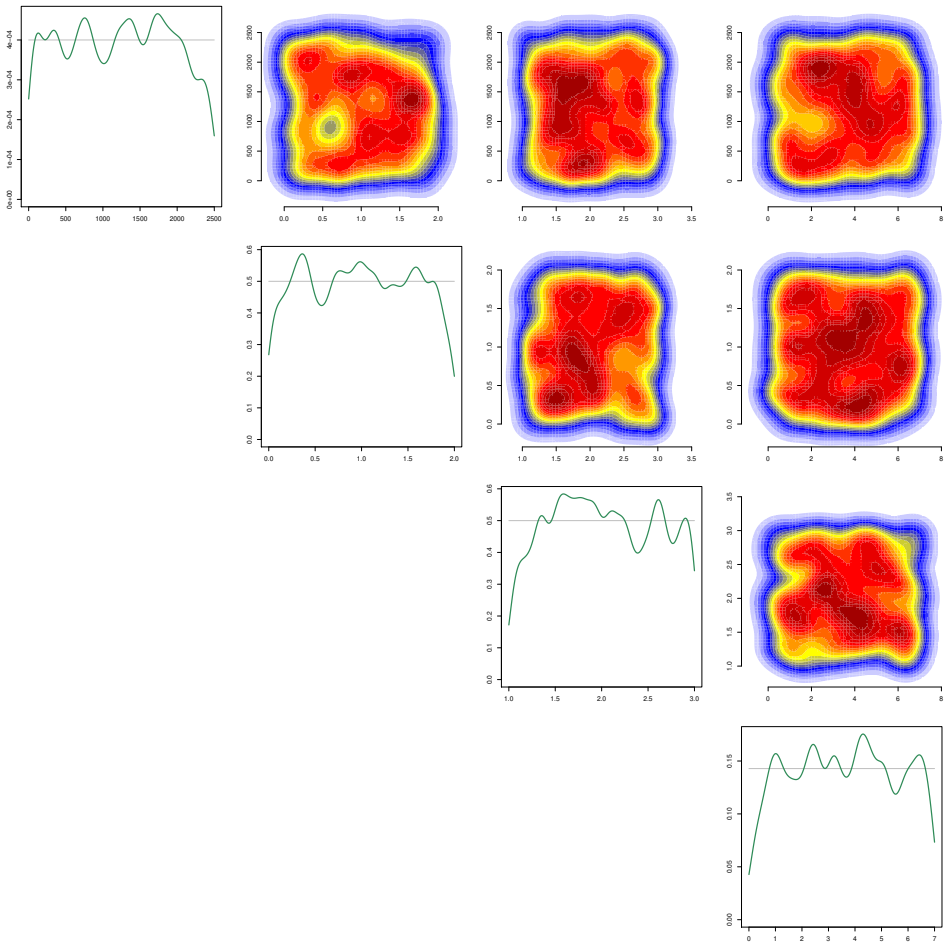


Figure 6.4: The distribution of model parameters $(\alpha, \alpha_0, n, \beta)$ at the first stage of SIS sampler. All the particles at this stage are coming from the prior distribution.

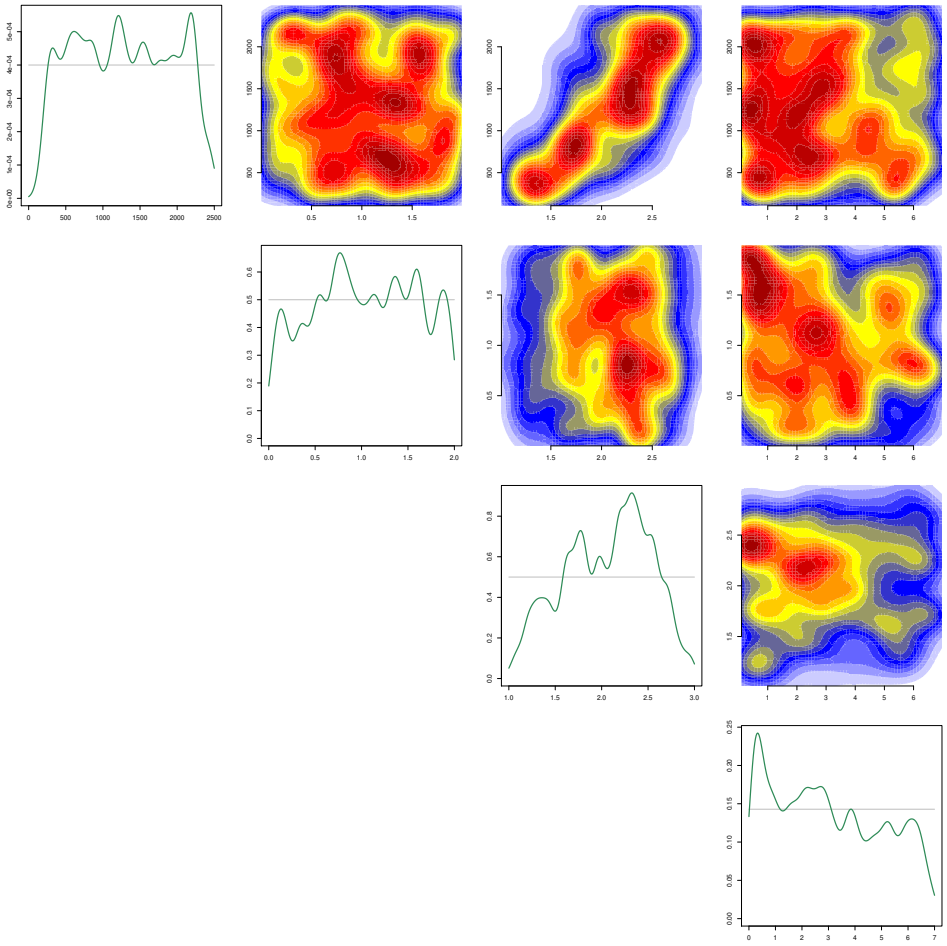


Figure 6.5: The distribution of model parameters at one of the intermediate stages ($i = 19$) of the SIS sampler. This distribution has already diverged from the prior distribution, but it is still different to the posterior distribution.

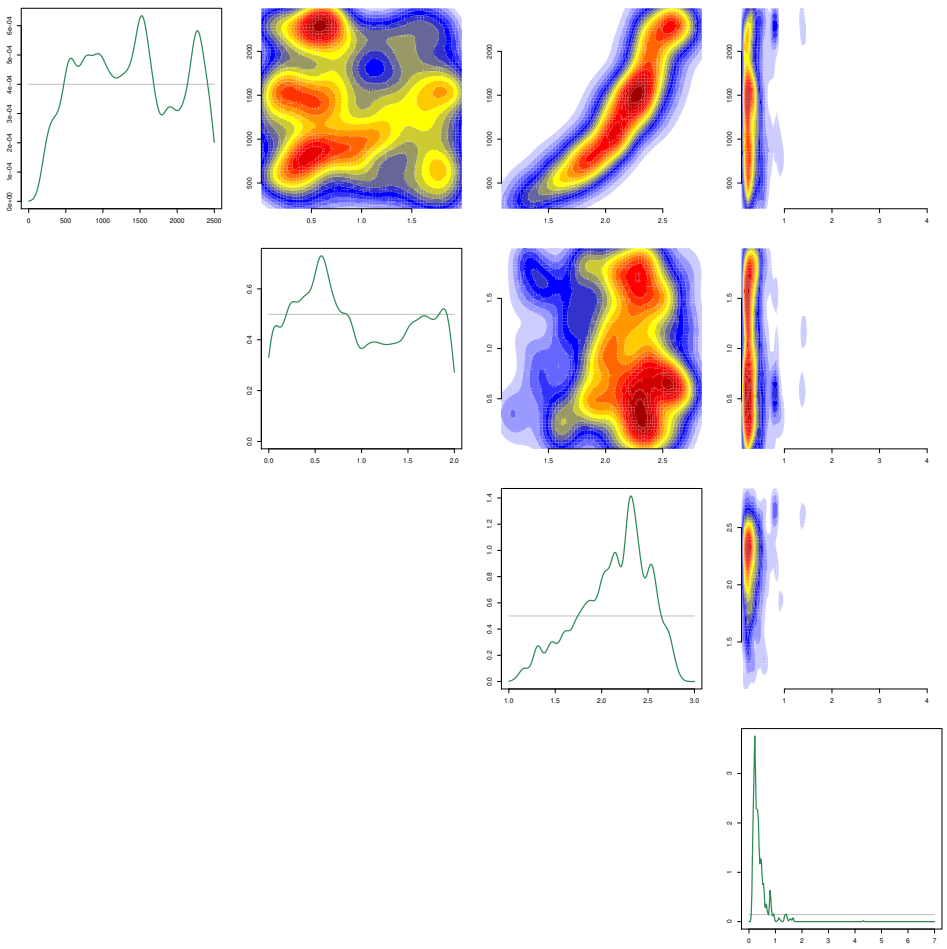


Figure 6.6: The posterior distribution of the Repressilator model parameters $(\alpha, \alpha_0, n, \beta)$ obtained using the SIS sampler. It is illustrated as a set of marginal posterior densities as the posterior defined in four dimensions.

6.4.2 Application of the ABC SMC

The ABC SMC algorithm (described in chapter 4) is performed for each data set generated from the Repressilator model according to Table 6.1. The target tolerance for the ABC SMC algorithm was selected as described in section 4.5.4. We randomly selected parameter values from the prior and simulated 2000 data sets from the model. Then, we compute Euclidian distances between these data sets, and choose 1% percentile of these distances (for one fixed value of model parameters) as our target tolerance. The resulting target tolerance for this study is $\epsilon = 20.9$.

In all experiments, the tolerance schedule was chosen to be adaptive as the $\alpha^{th} = 0.5$ quantile of the discrepancy between the observed and the simulated data sets. The prior distributions for each parameter are defined as previously:

$$\pi(\alpha) \sim \text{Uniform}(0, 2500)$$

$$\pi(\alpha_0) \sim \text{Uniform}(0, 2)$$

$$\pi(n) \sim \text{Uniform}(1, 3)$$

$$\pi(\beta) \sim \text{Uniform}(0, 8).$$

In the first experiment, we consider the first data sets simulated in section 6.3. It is the same data set as used for exact inference in section 6.4.1. The data were simulated using the kinetic parameters $\theta = \{\alpha = 1000, \alpha_0 = 1, n = 2, \beta = 1\}$.

The resulting approximate posteriors obtained using the ABC SMC algorithm for the first synthetic data set are depicted in Figure 6.7. Being an over dispersed approximation to the exact posterior, the ABC SMC result is certainly wider, but it still has substantial posterior support for the parameter value used for data generation. The marginal posteriors for α, α_0 and β are recovered reasonably well, while the posterior for n is biased in comparison to the true posterior. This may be due to nontrivial and nonlinear contribution of parameter n to the behaviour of the system, as it contributes as the power parameter controlling inhibition affinity. The overlap of the approximate and the exact posteriors not empty, but maximum a posteriori estimate are still quite different. Most importantly, the approximate posterior is wider than the exact one, as the ABC SMC produces an over dispersed approximation.

Figure 6.8 depicts the ESS for the ABC SMC algorithm with $N = 8000$ particles. The ESS never drops below the threshold of 4000 particles, so population resam-

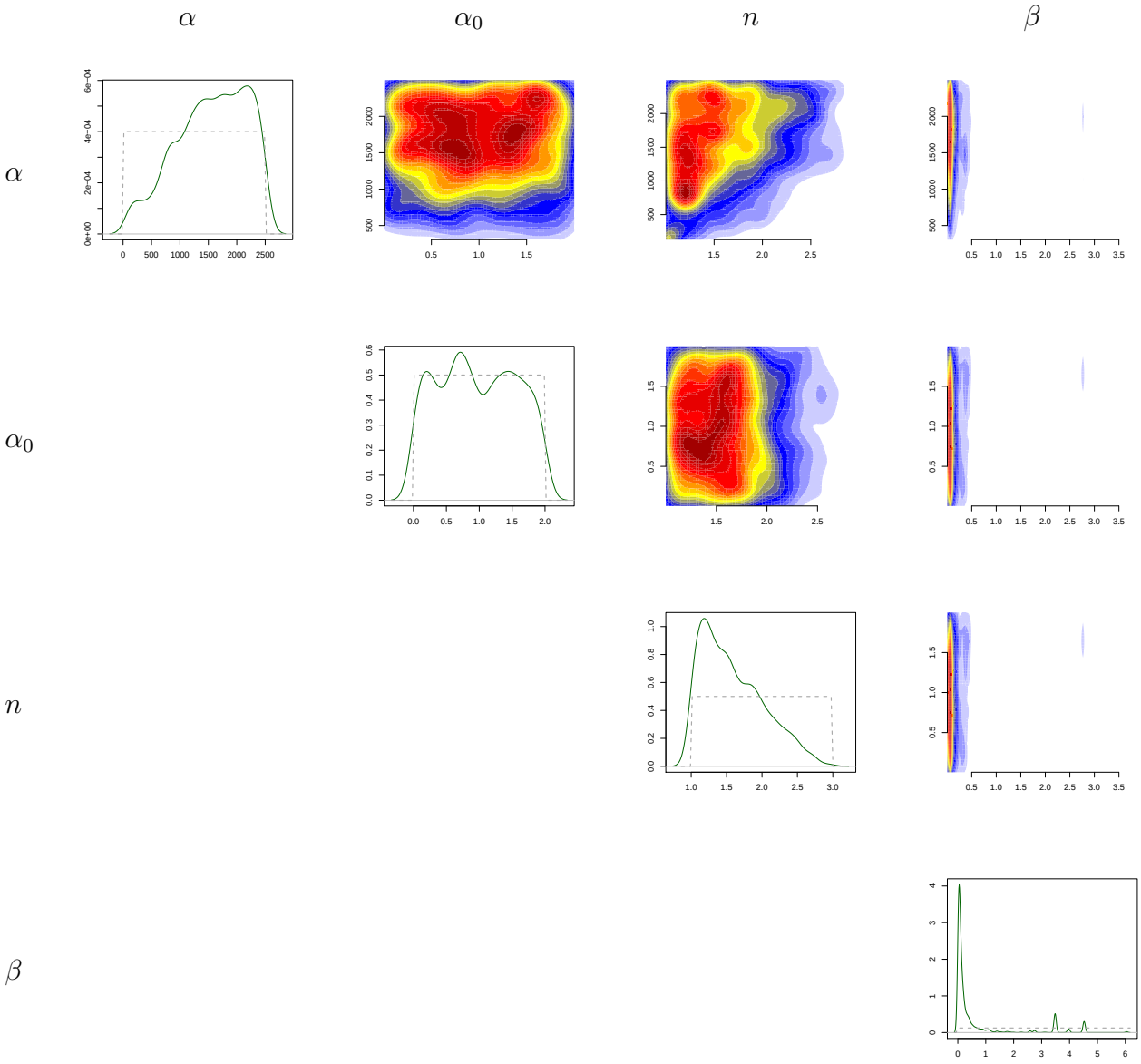


Figure 6.7: The posterior distribution of the Repressilator model parameters obtained using the ABC SMC algorithm. The dashed line represents the prior distribution.

pling was never needed. This demonstrates that a healthy population is maintained throughout the run of the samples.

For the second experiment we used a synthetic data set generated using very small parameter values $\theta = \{\alpha = 1, \alpha_0 = 0.001, n = 2, \beta = 0.2\}$. It was done to investigate whether the ABC SMC would explore the extremes of the parameter space well using a wide prior to begin.

The approximate posterior distributions for this case was obtained through a sequence of 24 distributions, and are depicted in Figure 6.9 (initial distribution), Figure 6.10 (intermediate distribution) and Figure 6.11 (posterior distribution). Figure 6.12 demonstrates how marginal posteriors evolve along the sequence of ABC approximations in the ABC SMC algorithm. This posterior is remarkably different to the prior. Marginal posterior distributions parameters α and β collapse to a tight distribution around the values used for data generation. The marginal posterior distribution for α_0 demonstrates moderate divergence from the prior, while the marginal posterior for n hardly diverges from the prior at all emphasising the difficulty of inferring n in this nonlinear model.

Figure 6.13 depicts the ESS for this experiment using a population of $N = 8000$ particles the ESS dropped below the threshold of 4000 particles only once, and we resampled the population in that case. Again, this shows that a reasonable diversity is maintained in the population, and we do not observe any population degeneracy problems.

The population size for ABC SMC is one of the key parameters of this algorithm. We investigate the impact of selecting the population size at different levels: $N = \{4000, 2000, 1000, 500\}$, independently. In the case, $N = 1000$ the ESS was dropping below the resampling threshold only a few times (5 times at worst), while for $N = 500$ resampling was required at the majority at the stages as shown in Figure 6.14. Judging by the performed diagnostic, it will be reasonable to use $N = 1000$ particles as it maintains a healthy population while requiring less computational time than our original setting $N = 8000$. The algorithm required 25 stages to reach the target tolerance for $N = \{8000, 4000\}$, and it took 24 stages for $N = \{2000, 1000, 500\}$.

The marginal posterior distribution obtained using different population sizes are compared in Figure 6.15 and Figure 6.16. The results are remarkably similar, which confirms that the algorithm converges to the same distribution. In the case $N = 500$, the result is still acceptable despite the worse performance of the ESS. This confirms

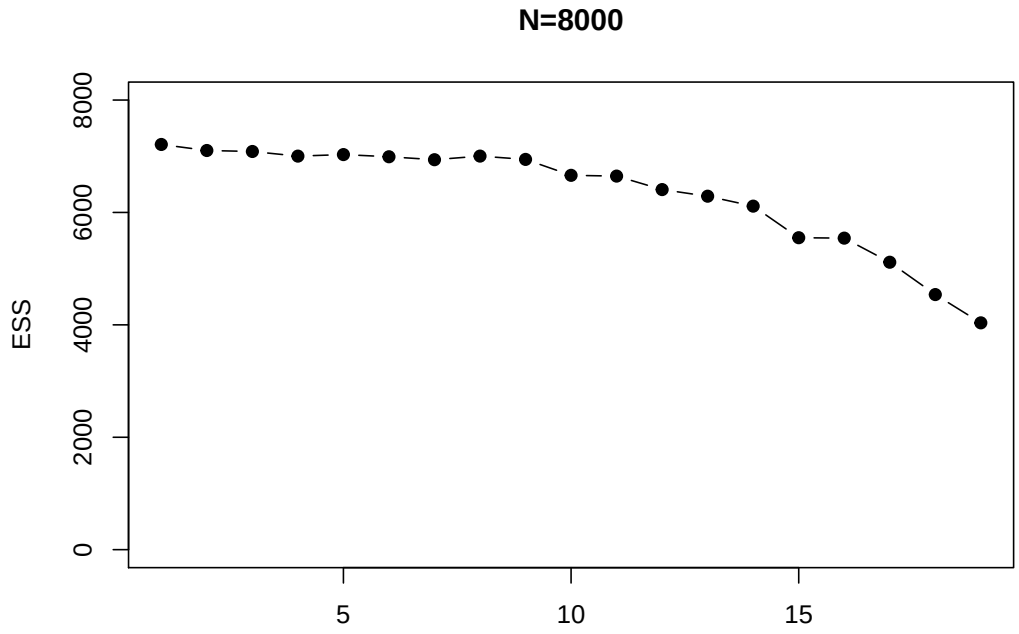


Figure 6.8: The plot shows the ESS obtained from performing the ABC SMC algorithm on the first synthetic data set (Experiment 1).

that resempling stage of the algorithm is mitigating the problems correctly, and the population never collapses to a single (or a few) particles.

Additionally, we performed parameter inference for all the nine synthetic data sets discussed in section 6.3. The results obtained are similar to the case discussed above. For example, for experiment $\theta = \{\alpha = 100, \alpha_0 = 1, n = 2, \beta = 1\}$, we repeated inference independently three times using different random number generator seeds, and the resulting marginal posteriors are depicted in Figure 6.17. It demonstrates that the sampler converges to the same target distribution every time. The same convergence property was observed for all nine data sets.

The computational cost of performing ABC SMC is significantly impacted by the size of the population. The importance distribution density (i.e. the density of the perturbation kernel) needs to be evaluated for every proposed particle as a mixture of normal distributions, with the same number of components as the number of particles employed. The cost of doing it at every stage is, therefore, $O(N^2)$. The time required to run the ABC SMC with different population sizes is compared in figure 6.18. The box plots are produced using the results of all nine data sets considered in this chapter. Obviously, the largest populations require a larger time

to sample.

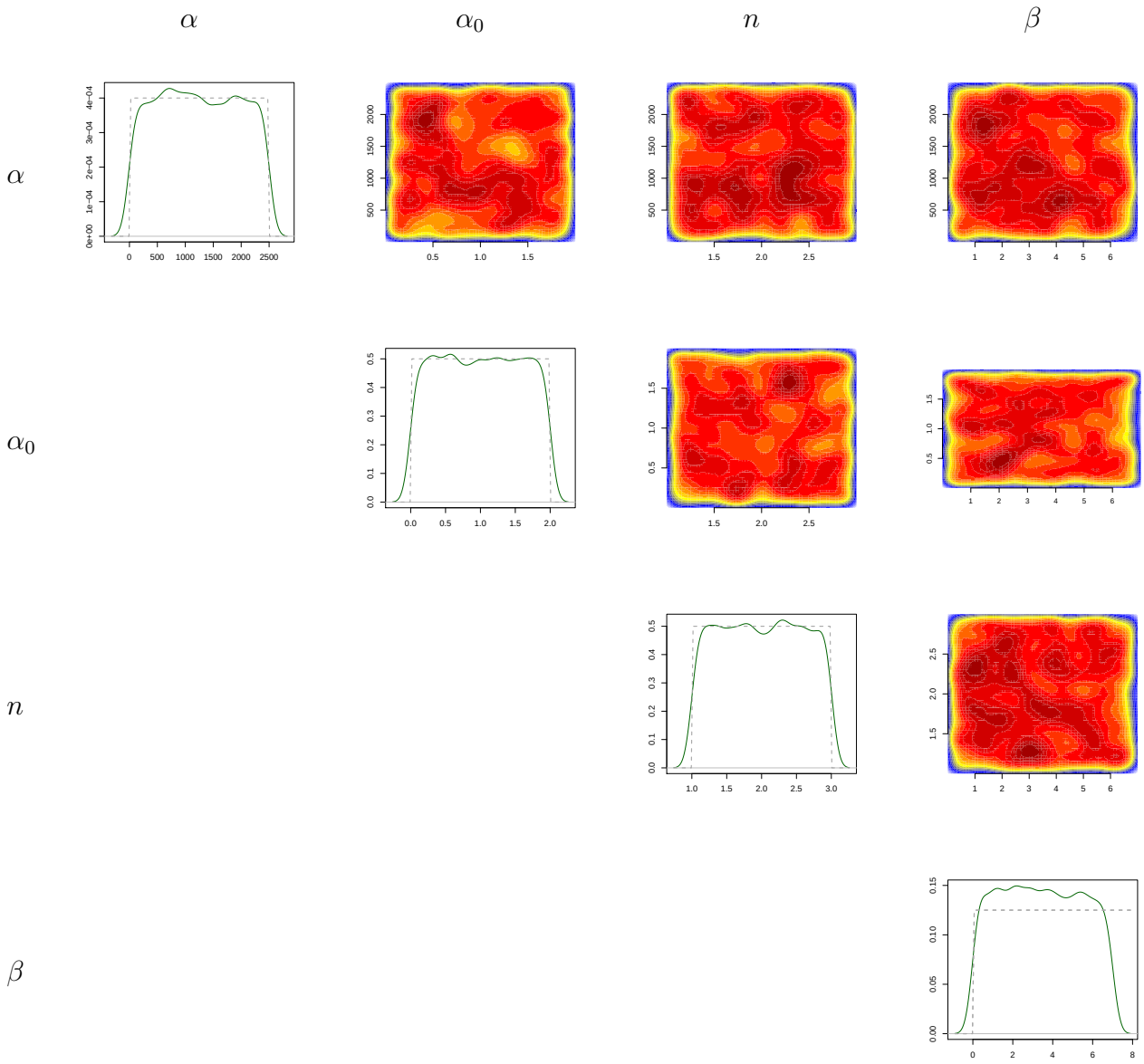


Figure 6.9: The distribution of model parameters at the first stage of ABC SMC algorithm. All the samples at this stage are coming from the prior distribution.

6.4.3 Application of the PMMH to the Repressilator Model

The PMMH algorithm is applied for each synthetic data set, some of the results will be presented in this section, and other will be given in Appendix E. The bootstrap particle filter is used to estimate the likelihood with the number of particles $N = 200$. The chains were initialised from the same prior distributions as before:

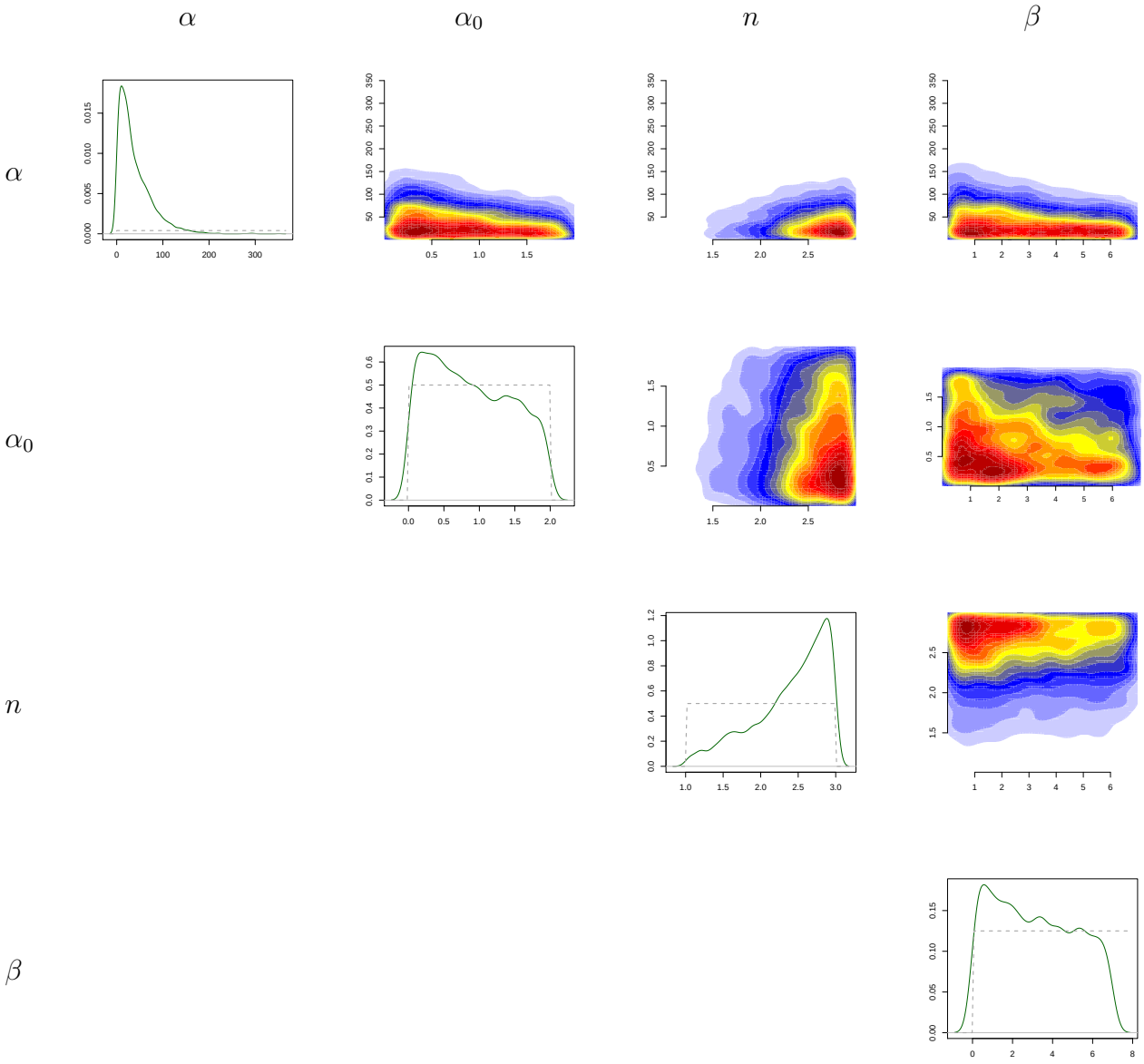


Figure 6.10: The distribution of model parameters at one of the intermediate stages (i = 11) of ABC SMC algorithm. This distribution has slightly diverged from the prior distribution, but it is still different to the posterior distribution.

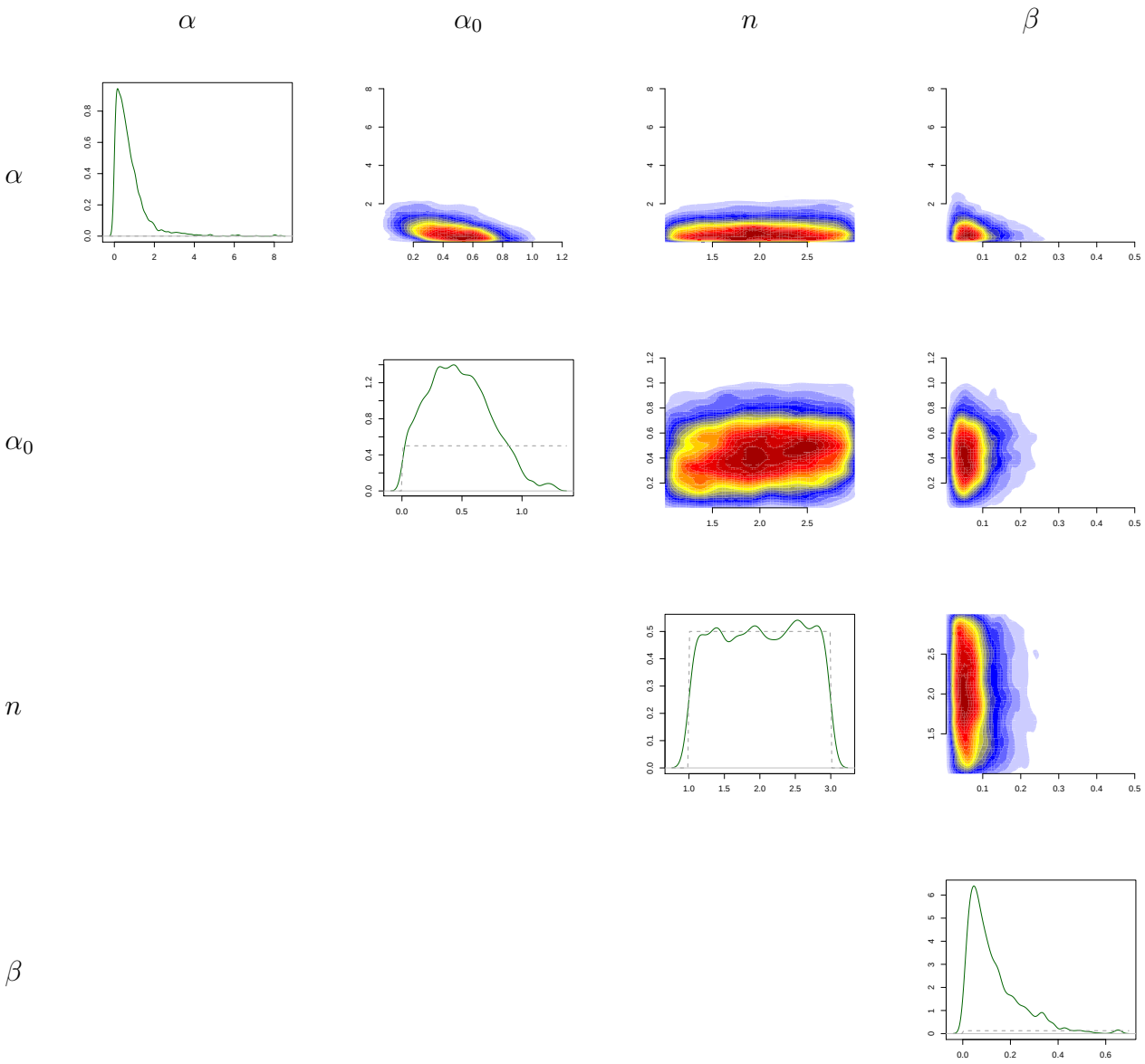


Figure 6.11: The posterior density of parameters of the Repressilator model given the first synthetic data set, the dashed line represents the prior distribution.

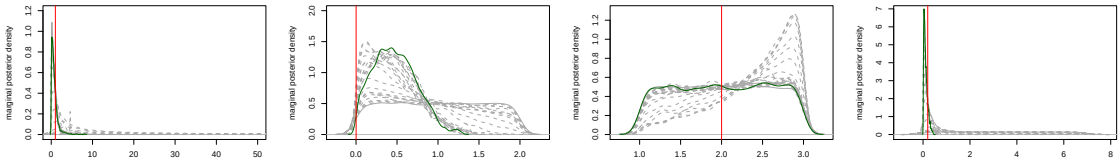


Figure 6.12: Approximate posterior distributions of the Repressilator model parameter obtained using ABC SMC giving a single synthetic data set. Dashed lines represent the sequence of distributions, solid lines show the final approximation posterior distributions, whereas the red vertical line shows the parameter values that have been used to simulate the synthetic data.

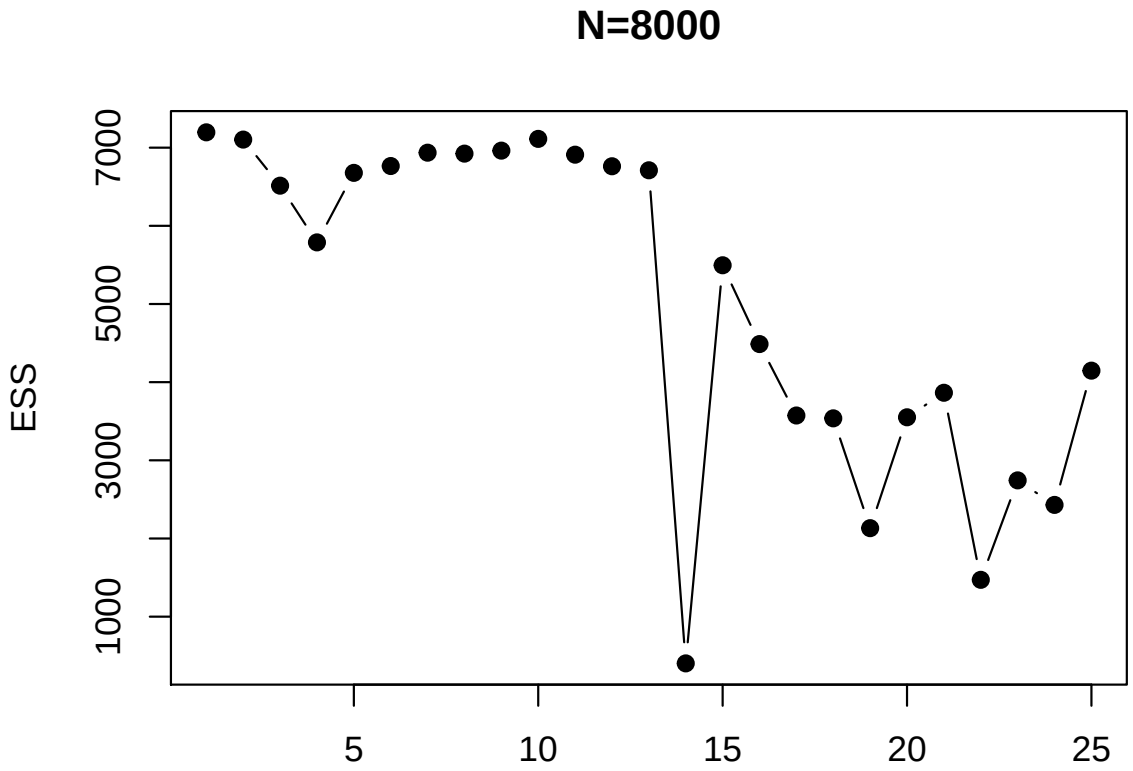


Figure 6.13: The plot represents the ESS associates with experiment 9 where the ABC SMC applied with $N = 8000$, where X- axis represents the number of ABC SMC stages25.

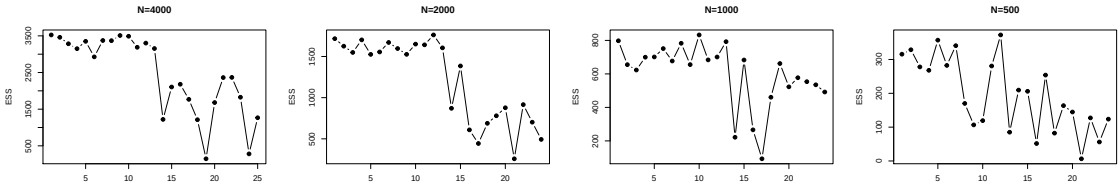


Figure 6.14: The plot represents the ESS corresponding to several runs of experiment9, in which ABC SMC applied with five different N given the same synthetic data.

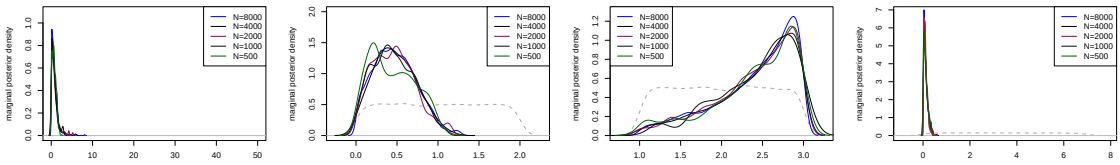


Figure 6.15: Approximate posterior distributions of the Repressilator model parameters obtained from performing ABC SMC algorithm five times independently with different values of N, given the same synthetic data set (Experiment 9). The five approximate posterior distributions are very similar.

$$\alpha \sim \text{Uniform}(0, 2500)$$

$$\alpha_0 \sim \text{Uniform}(0, 2)$$

$$n \sim \text{Uniform}(1, 3)$$

$$\beta \sim \text{Uniform}(0, 8).$$

The proposal distribution is selected to be an adaptive normal random walk with covariance matrix being estimated from the history of the current PMMH trace.

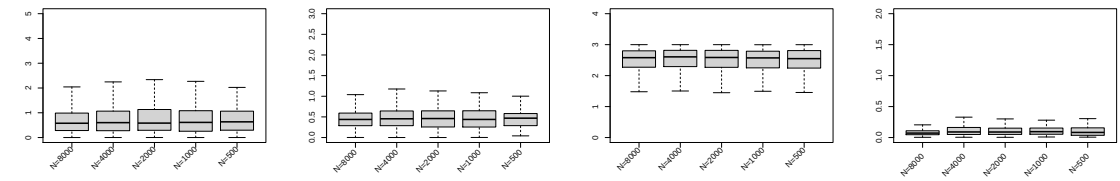


Figure 6.16: The box plot represents the approximate posterior distributions of the Repressilator model parameter obtained from performing ABC SMC 5 times independently with different values of N, giving the same synthetic data set (Experiment 9). The five approximate posterior distributions were very similar.

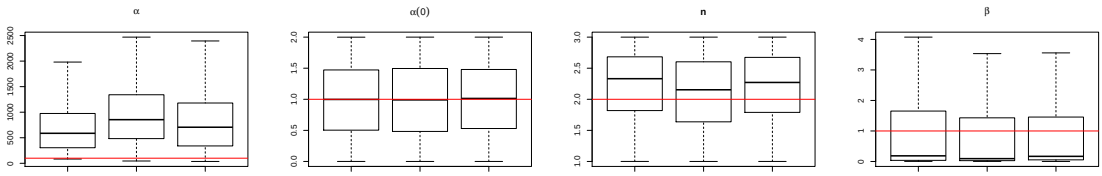


Figure 6.17: The box plot represents the approximate posterior distributions of the Repressilator model parameter obtained from repeating ABC SMC algorithm three times independently given the same synthetic data set.

The proposal is then scaled using the movement length c^*. It is initialised with $c^* = 0.1$, and then adjusted to keep the acceptance rate between 20% and 70%. The algorithm includes two stages: adaptive and stationary. At the adaptive stage sampling is performed in windows at 3000 iterations, after each window we adjust the covariance and the movement length of the proposal distribution. After convergence was achieved, the proposal distribution was fixed, and we call the next stage to be the stationary stage of the algorithm. 2000 samples were collected from the stationary stage to construct the posterior distributions. Due to the enormous computational cost of running this algorithm, we could not afford simulating multiple independent chains for formal convergence monitoring, and the convergence was judged visually by observing current traces until they reach stationarity.

The posteriors for all nine of the data sets introduced in section 6.3 were obtained using this algorithm. It can be observed in Table 6.2 that in every case a large number of samples had to be discarded to burn in. This table also shows the huge computational on the cost of running this sampler.

Figures 6.21 and 6.22 show two independent runs of the sampler using the data for experiment1. Both of those traces are produced from the stationary stage of the samples after discarding 156000 samples for the burn in. The proposal distributions were adapted 52 times during the burn in. It is concerning that after running these chains for about 60 hours each, there is still little agreement in their resulting distributions. We were using 200 particles in the likelihood estimator using the bootstrap particle filter. It might be the effect of the variance of the likelihood estimator that causes poor convergence performance, as discussed in chapter 5. The computational cost of running this algorithm limits our ability to investigate longer burn in periods or larger population sizes in the bootstrap filter.

We applied the sample to all nine of the datasets, and observed similar poor performance in terms of convergence when inference is repeated, despite chains looking

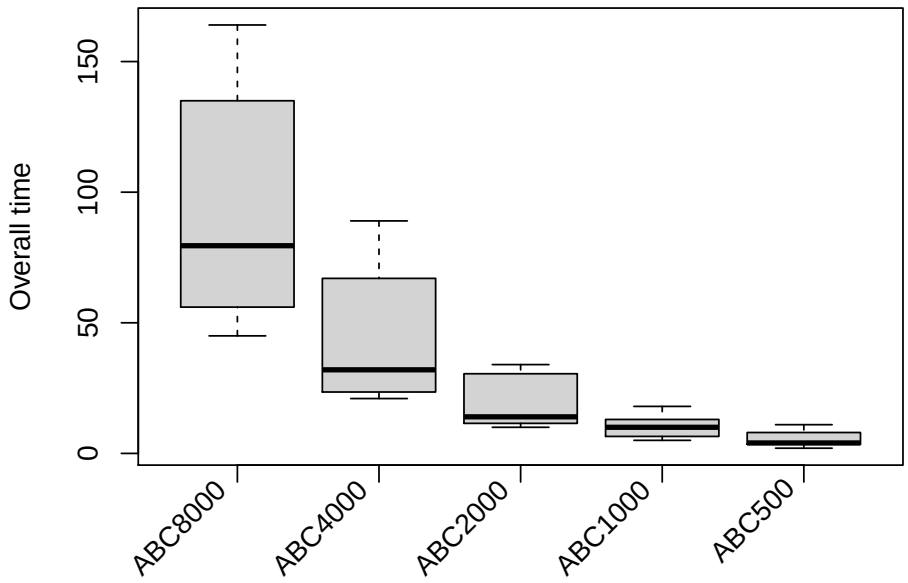


Figure 6.18: The box plot shows the computational time for several runs of the ABC SMC algorithm, given a different synthetic data sets that are depicted in Figure 6.2. It is obvious that the inference based on $N = 8000$ requires more computational time compared to $N = 500$.

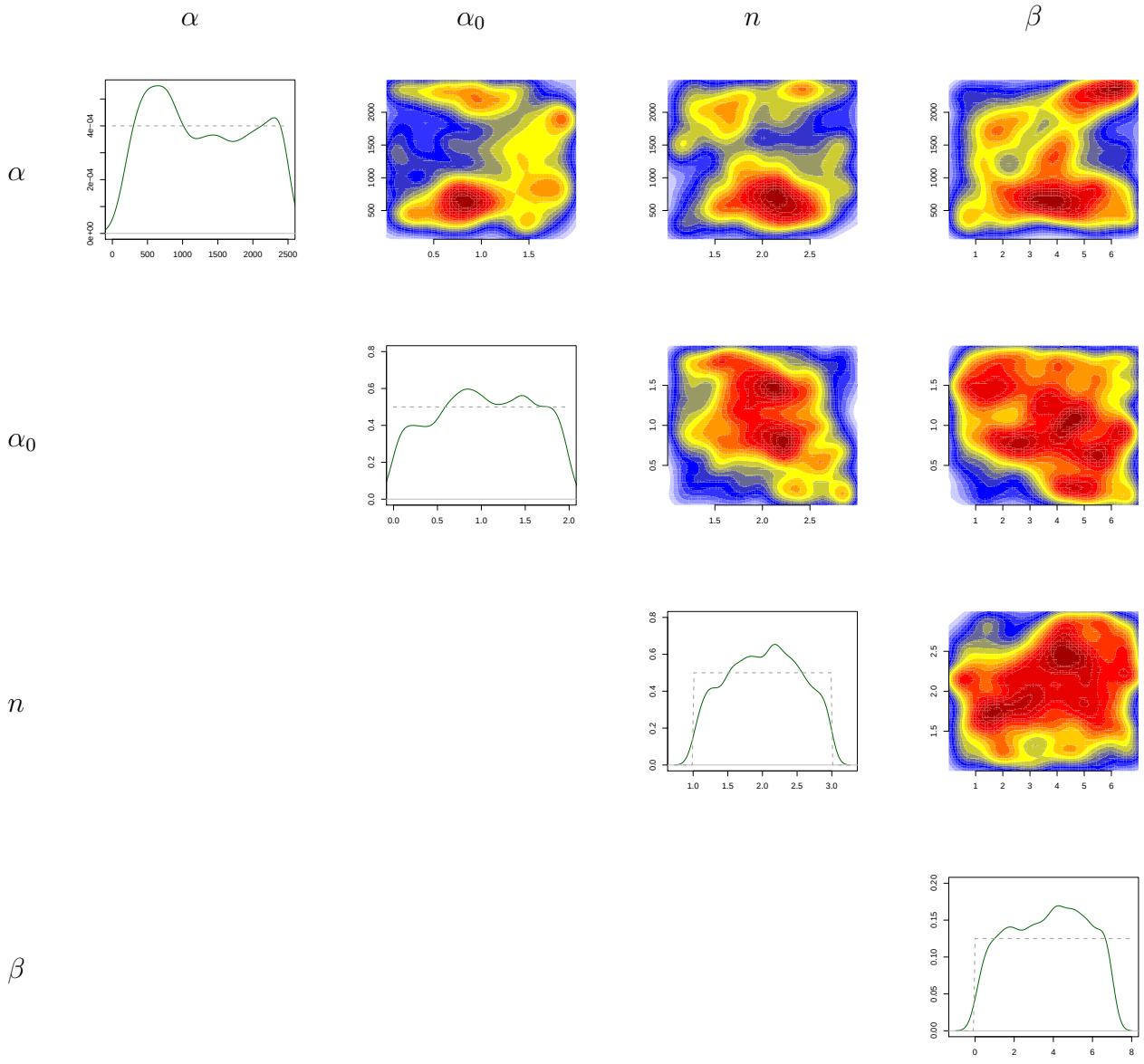


Figure 6.19: The posterior density of parameters of the Repressilator model obtained from performing the PMMH, given the first synthetic data set, the dashed line represents the prior distribution.

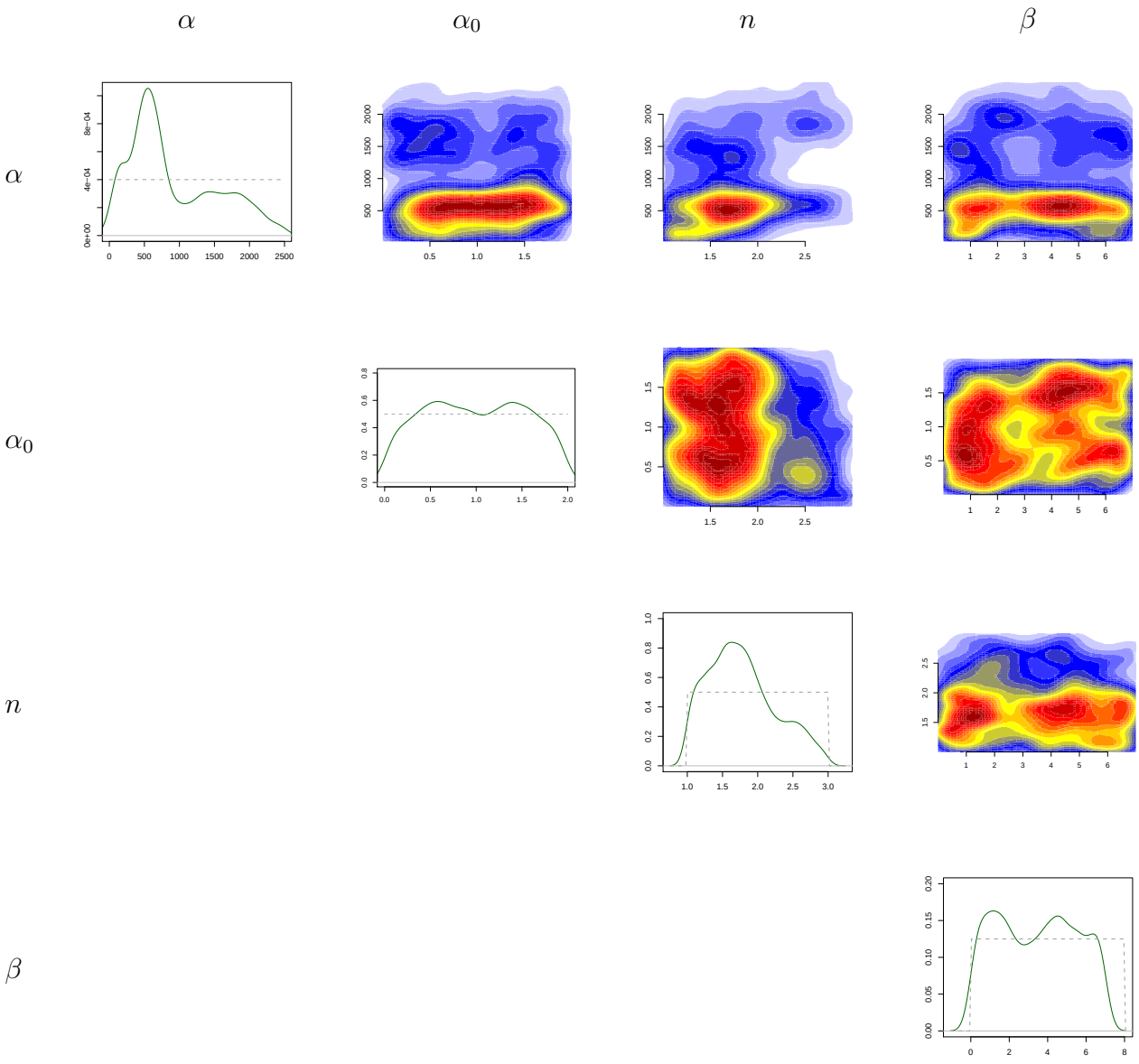


Figure 6.20: The posterior density of parameters of the Repressilator model given the first synthetic data set and the priors are shown in dashed line.

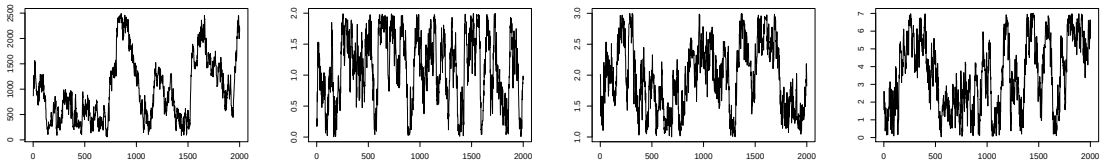


Figure 6.21: The trace plot represents the chain after burn in period for the Repressilator model parameter obtained from performing the PMMH algorithm using the first synthetic data set (Experiment 1).

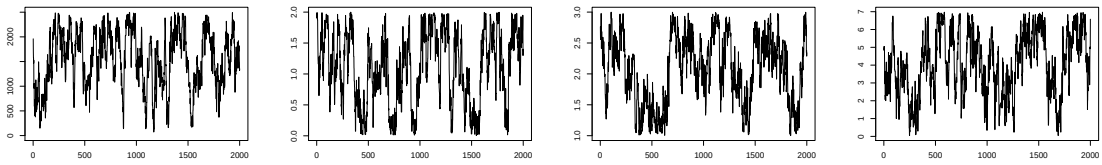


Figure 6.22: The trace plot represents the chain after burn in period for the Repressilator model parameter obtained from repeating the PMMH algorithm (Experiment 1).

stationary at some stage of the adaptive sampling. This means that the sample becomes trapped in local posterior modes. This is not an unusual problem with MCMC in general, and the situation is made even more difficult by the variance of the likelihood estimator in this setup.

The marginal posterior distributions for two of the datasets obtained using the PMMH were similar to the results obtained with ABC SMC as depicted in Figures 6.23 and 6.24, while the posterior distributions for some other datasets hardly diverge from the prior as in Figure 6.25 (additional results are given in Appendix E).

Comparing the approximate posteriors for the first dataset depicted in Figure 6.19 and Figure 6.20 to the exact posterior depicted in Figure 6.6, we see that the approximation obtained with the PMMH algorithm is far from the exact result. The PHHM sampler appears to be trapped in some local mode.

Since the PMMH algorithm has explored the parameter space poorly, we were encouraged to replicate the algorithm using the adaptive log normal random walk as a proposal distribution. The PMMH algorithm with the log normal proposal was running for more than two weeks without any improvement in terms of convergence. We were not able to observe stationarity of the traces.

Such poor performance of the PMMH algorithm for the Repressilator model moti-

Table 6.2: Summaries of total time for several runs.

Experiment	Time (in minutes)	Number of adaptive phases
Experiment 1	3389	52
Experiment 2	2597	38
Experiment 3	2221	34
Experiment 4	1802	28
Experiment 5	2471	36
Experiment 6	2437	34
Experiment 7	4016	52
Experiment 8	2030	30
Experiment 9	769	24

vated considering only the ABC SMC method for inference using the real data set discussed in the next section, which is significantly less informative and the model is even more complex.

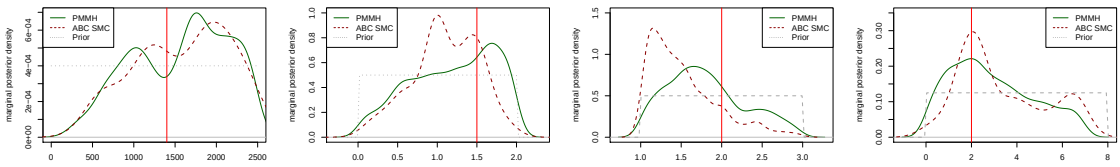


Figure 6.23: The approximate posterior distributions for the Respressilator model parameter obtained from performing the ABC SMC and the PMMH given the synthetic data set generated at $\theta = \{\alpha = 1400, \alpha_0 = 1.5, n = 2, \beta = 0.2\}$.

6.5 Parameter Inference for Repressilator Model Using Real Data

We perform approximate parameter inference for the Repressilator model using the ABC SMC algorithm. The PMMH sampler will not be considered due to its poor performance in the study on synthetic data. The original paper on the Repressilator (Elowitz and Leibler, 2000) reports only indirect measurements of the green fluorescent protein activity over time depicted in Figure 6.26. Therefore, we need to consider the complete Repressilator model involving four mRNAs and four proteins. However, in this case, the initial concentrations of the mRNAs and the proteins

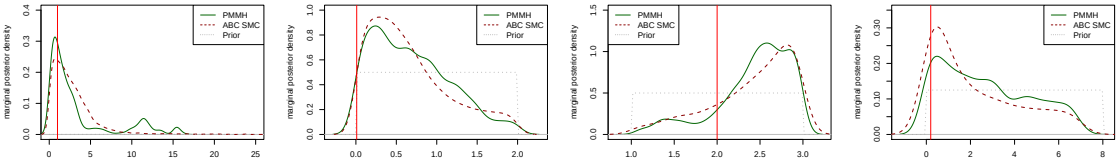


Figure 6.24: The approximate posterior distributions for the Respressilator model parameter obtained from performing the ABC SMC and the PMMH given the synthetic data set generated at $\theta = \{\alpha = 1, \alpha_0 = 0.001, n = 2, \beta = 0.2\}$.

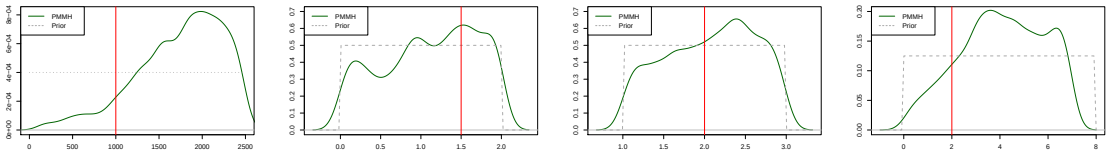


Figure 6.25: The approximate posterior distributions for the Repressilator model parameter obtained from performing the PMMH.

are unknown. We will treat these initial concentrations as additional unknown parameters in our inference problem. This results in considering inference over 12 parameters in total the multitude of uncertainties in the Repressilator system motivates the preference for using the Bayesian approach over the traditional maximum likelihood estimation. This allows using prior information, in theory, to constrain the uncertainty. In the case, where only limited data are available, and the model is large and complex, the likelihood becomes infeasible to estimate directly. However it is relatively simple to simulate from the model for a given value of model parameters. Therefore, the ABC method should perform well in this situation.

We will study the Repressilator model defined using three different scales for the number of discrete concentration levels (denoted ML) $ML = 5$, $ML = 20$ and $ML = 200$. The prior for the initial mRNA and protein values are chosen to be uniform:

$$\pi(m_1) \sim \text{Uniform}(0, ML)$$

$$\pi(m_2) \sim \text{Uniform}(0, ML)$$

$$\pi(m_3) \sim \text{Uniform}(0, ML)$$

$$\pi(m_{gf}) \sim \text{Uniform}(0, ML)$$

$$\pi(p_1) \sim \text{Uniform}(0, ML)$$
$$\pi(p_2) \sim \text{Uniform}(0, ML)$$
$$\pi(p_3) \sim \text{Uniform}(0, ML)$$
$$\pi(p_{gf}) \sim \text{Uniform}(0, ML),$$

and the priors for the kinetic parameters are:

$$\pi(\alpha) \sim \text{Uniform}(0, 2500)$$
$$\pi(\alpha_0) \sim \text{Uniform}(0, 10)$$
$$\pi(n) \sim \text{Uniform}(0.5, 3)$$
$$\pi(\beta) \sim \text{Uniform}(0, 20).$$

To produce the target tolerance level, we take a sample of the model parameters from the prior, then produced 2000 simulated data sets from the model. Then, we compute Euclidean distances between them and take the 1% percentile of those distances as the target tolerance level. The perturbation kernel is choosing to be a component wise normal distribution with variances estimated from the previous population of particles.

At every stage of the ABC SMC, we select the next tolerance level as the $\alpha^{th} = 0.5$ quantile of the distances between the simulated and the real data and the population of particles.

The first experiment in the section considers the largest model setup with $ML = 200$. The second one considers $ML = 20$, and the third one uses $ML = 5$. In every case, the data in Figure 6.26 has to be scaled to the number of discrete levels. We assume that the largest measurement and the data set should correspond to the 59.5% of ML. This assumes that for the first experiment the data are translated into concentration levels directly by rounding numbers, while for smaller models they are appropriately scaled.

The ABC SMC algorithm is performed using $N = 3000$ particles in every case. We observed that using $\alpha^{th} = 0.5$ quantile for improving the tolerance level from stage to stage performed well. In the first experiment, 1.11% of the samples were accepted on average. The target tolerance for the first experiment was 209.1, while the first tolerance level in ABC SMC setup was $\epsilon = 326$. The target tolerance level

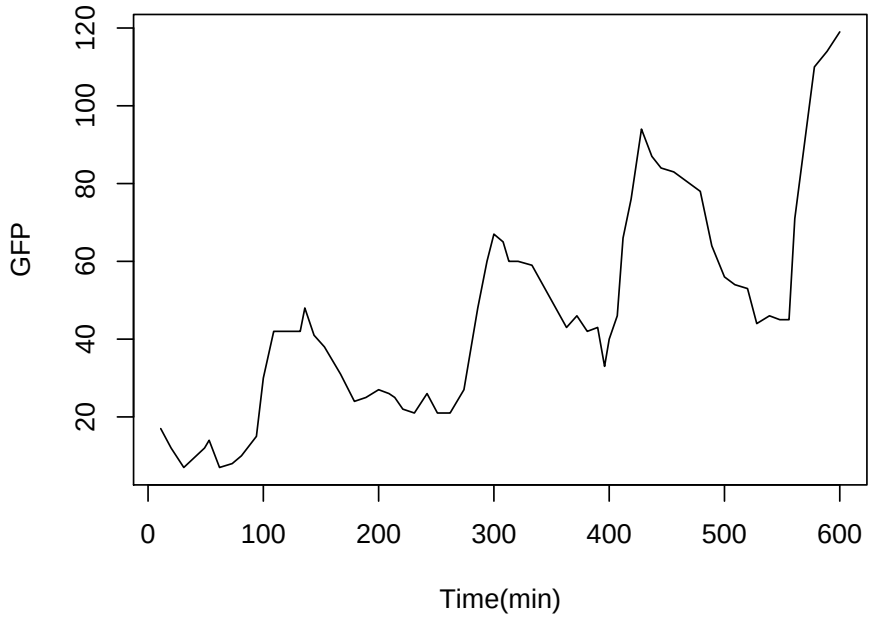


Figure 6.26: The plot shows the GF protein concentration level for the Repressilator model.

was achieved in 9 stages as shown in Figure 6.27. For the second experiment with $ML = 20$ the target tolerance of 26 was achieved in 15 stages with the average acceptance rate of 0.012%. For the third experiment with $ML = 5$ target tolerance of 13 was achieved in 12 stages with the average acceptance rate of 0.002%. We observed that the target tolerance level and the acceptance rates decrease for smaller models, due to the smaller scale of the data values. The sequence of tolerance levels for all three of our experiments are shown in Figure 6.27.

The ESS was measured for all of the experiments to monitor population diversity in the ABC SMC. In the first experiment, the ESS stays constantly high as shown in Figure 6.28. While in the second and third experiments ESS demonstrated population collapse to a very small number of particles at one of the stages as shown in Figure 6.28. As the result we recommend relying on the posteriors from the first (and largest) experiment, as the other two may be underestimating the posterior variance.

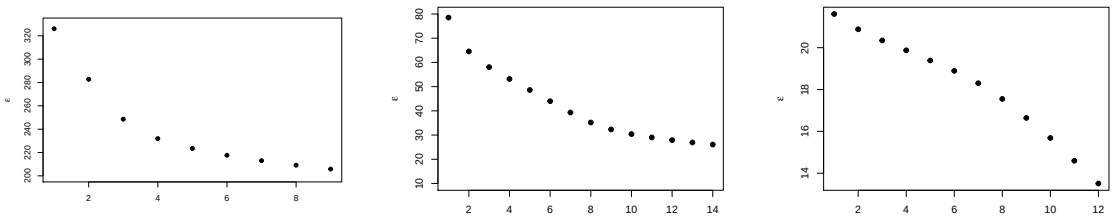
Performing computations for each of these experiments takes approximately a month, as the acceptance rates drop dramatically at later stages. Considering larger particle population was not feasible due to limited availability of computing resources.

Every posterior produced in these three experiments is a distribution in 12 dimensions. We only plot the marginal posteriors for the kinetic parameters in Figures 6.29, 6.30 and 6.31.

The posterior from the first experiment is the most reliable one as particle population never collapsed in the run. It demonstrates that the data are not particularly informative for three of the kinetic parameters, while there is still a significant divergence from the prior and a non trivial correlation identified for one of the parameters. This emphasises that there is a wide range of parameter value combinations that explain the observed data equally well. This demonstrates the futility of the maximum likelihood point of estimation approaches to model fitting. At least with the Bayesian approach we know how big the uncertainty of the estimates is.

The root of the problems is in the type and quantity of real data available in this case. The available data provides only very limited information about the underlying model.

The posteriors for the second and the third experiments are remarkably tighter, however, we argue that this is an artefact due to the population collapse in ABC SMC, as indicated by the EES values plotted in Figure 6.28.

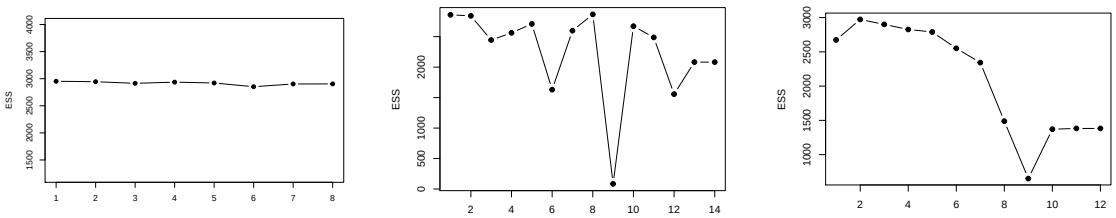


(a) (b) (c)

Figure 6.27: The sequence of tolerance values for each experiments.

6.6 Summary

In this chapter, we have investigated Bayesian inference for the Repressilator model with both synthetic and real data. We have found that applying Bayesian inference for the Repressilator model is a challenging task. This is due to the fact that the Repressilator model includes unobserved states and thus the formalisation of the likelihood of the Repressilator model becomes intractable. In addition to that, the



(a) (b) (c)

Figure 6.28: The plot shows the ESS for each experiments.

Repressilator model involves a large number of states, resulting in large state space which limits the applicability of direct approaches, such as a Gibbs sampler based on a truncation to evaluate the exact posterior. Approximation approaches were suggested in the recent development of the inference methods as a practical way to handle this problem, such as ABC and PMMH algorithms.

The first set of experiments in this chapter considered an application of inference methods on the Repressilator system using synthetic data sets. It would be beneficial to consider an approach that allows us to obtain exact inference for this model. One possible approach is to compute the probability of the transition is based on uniformisation. Still, when the CTMC model includes high concentration levels, the number of states in the Markov chain becomes large, and hence the matrix exponential becomes extremely expensive. In this sense, the CTMC is constrained to 5 levels only to obtain the feasible likelihood. The exact inference can be used to quantify the quality of the performance of the approximation approaches.

We have designed a simulation study to examine the performance of the suggested approximation inference approaches, specifically ABC SMC and PMMH algorithms.

The ABC SMC gave consistent results with the exact method. We demonstrated in the simulation study that the proposed inference approaches are able to infer the Repressilator model parameter from a set of synthetic data.

The main complication we have faced while performing these methods was the computational cost. The exact method was computationally expensive even though the model was relatively small. Nevertheless, we have designed the algorithm to be run on parallel hardware to achieve our goal. This is a serious problem when a larger model is considered.

The main challenge in performing sampling using the PMMH algorithm is achieving convergence of the underlying Markov chains in a reasonable time. Even for the

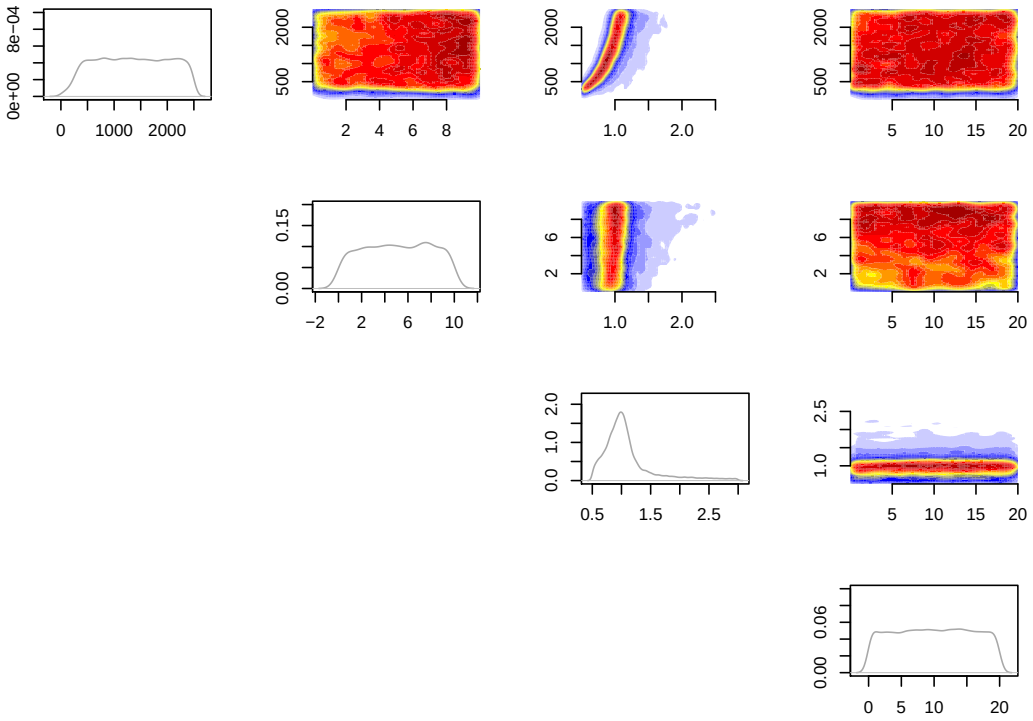


Figure 6.29: Posterior distributions for the Repressilator model parameters obtained from performing the ABC SMC sampler, when $ML = 200$.

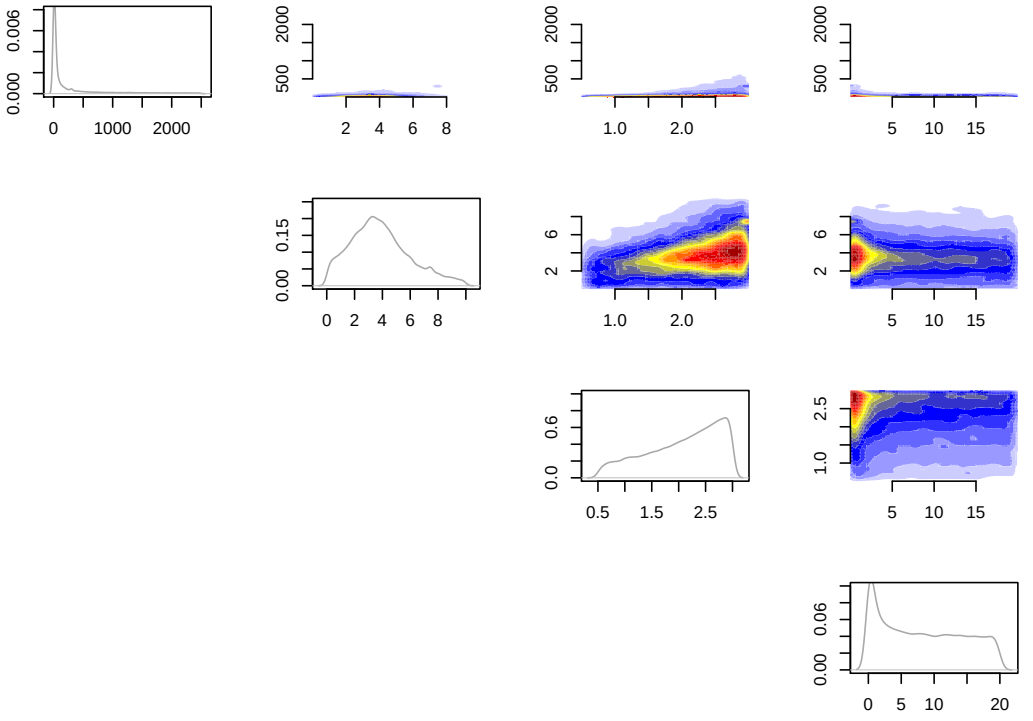


Figure 6.30: Posterior distributions for the Repressilator model parameters obtained from performing the ABC SMC sampler, when $ML = 20$.

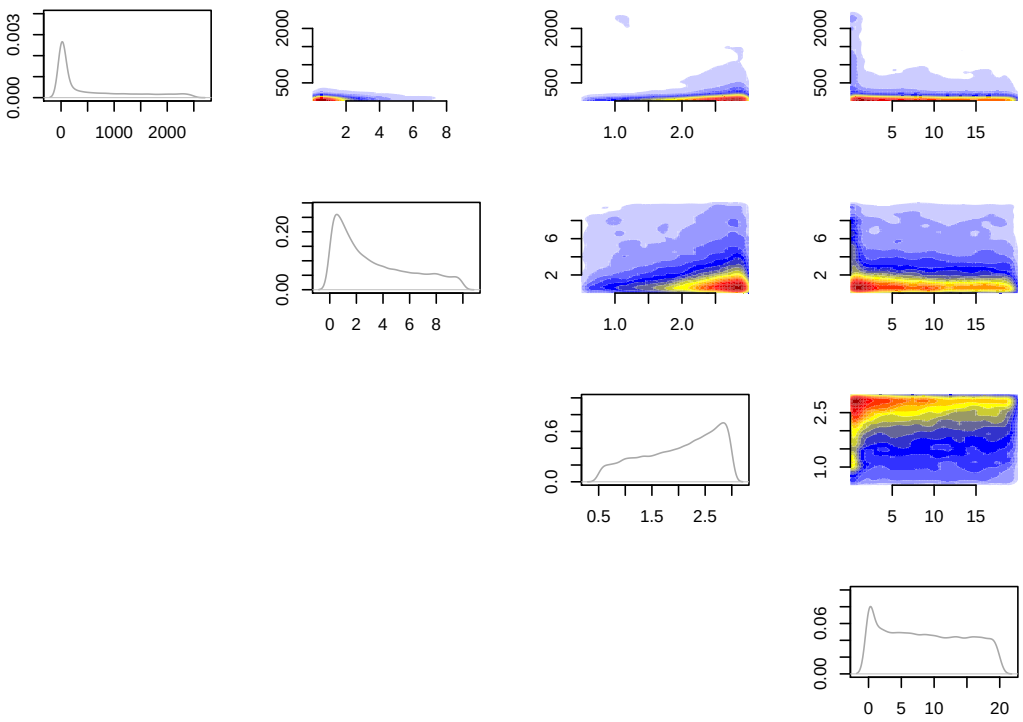


Figure 6.31: Posterior distributions for the Repressilator model parameters obtained from performing the ABC SMC sampler, when $ML = 5$.

simplest model configurations we did not achieve reliable convergence after hundreds of thousands of iterations. It is also extremely expensive computationally to use larger particle populations in the likelihood estimator for PMMH.

In the ABC SMC algorithm, the results can be obtained more reliably, however careful monitoring of the ESS is required to avoid population collapse as illustrated in two of our experiments using the real data.

The second part of this chapter illustrated approximate inference using the ABC SMC on the real data set. The additional challenge to the intractability of the likelihood, in this case, is that the number of unknown parameters was significantly larger, and the data set was not particularly informative for this inference. The obtained result, while having large posterior variance, demonstrates the key benefit of using the Bayesian approach to the parameter inference problem. We properly quantify the uncertainty of our results, rather than being falsely confident about one a solution to the parameter estimation problem.

Chapter 7

Summary and Conclusion

Many important phenomena in life sciences demonstrate stochastic behaviour. A particular example considered in this thesis is biochemical system ubiquitous in molecular biology. The traditional approach of modelling such systems using ordinary differential equations sacrificed the stochastic nature of these systems altogether and considers their behaviour on average. This is not a huge problem when the populations of biochemical species are large. However, when modelling single cell gene expression, stochasticity of molecular interactions may lead to significant variation in observed behaviours of the system. In such cases ODE modelling becomes inadequate, and stochastic models of molecular interactions must be employed. One of the frameworks to model such stochastic systems is using CTMC. A huge problem, however, arises when working with large CTMC models, evaluating the likelihood for these models becomes infeasible in practice.

We focus particularly on the problem of Bayesian parameter inference for CTMC. The problem of evaluating likelihoods for this inference is further complicated by the fact that practical datasets are far from being complete. Only few of the species are typically observed and observations are sparse.

Several methods have been suggested and developed to overcome the issue of the intractability of the likelihood. These methods are typically based on approximating the likelihoods. In this thesis, we considered two such methods; the approximate Bayesian computation (ABC) approach and the Particle Marginal Metropolis-Hastings (PMMH) sampler. Both of them are using simulations from the model to approximate the likelihood in slightly different ways.

We demonstrated application of these approaches in two case studies that considered models of stochastic biological system. The main contribution of this work is in

exploration and comparison of performance and effectiveness characteristic of those two approaches to approximate Bayesian inference.

We compared the quality of posterior approximations obtained with both of these methods and the computational costs of running them. The extensive evolution of these methods allows drawing important conclusions about using them in practice.

The most challenging problem in performing approximate Bayesian inference using these methods is tuning the parameters of the corresponding algorithms to balance the quality of the results with the computational costs required.

We replicated inference on multiple synthetic datasets using both methods, measuring the quality of approximations and the running time in different configurations of the algorithms. To do this we first considered simpler model configurations for which exact inference results were available. We employee the Gibbs sampler based on random truncation to obtain exact posteriors in our first case study, and a sequential importance sampling algorithm using model uniformisation in the second case study. We demonstrated that ABC SMC sample produces approximate results better or equal to the results using PMMH at significantly lower computational cost.

In the case of ABC SMC we investigated the impact of the size of the population and the choice of tolerance schedule on the quality and the cost of the result.

We observe that the size of the population, as long as it is larger than a few hundred particles has little impact on the quality of approximation. The computational cost of simulating larger populations keeps growing while the approximations' quality improves only marginally. It is critical however to monitor the effective sample size of particle populations during ABC SMC runs, as sometimes, especially when using larger and complete models, the particle population may collapse into a singular point. The situation must be detected and avoided.

We also observed that using adaptive tolerance schedules in ABC SMC is effective, and the tolerance reduction quantile parameter, α^{th}, has significant impact on the approximation quality and performance. If the quantile chosen to be small, the algorithm usually converges in a small number of stages. However, it creates a higher risk of population degeneracy, and population collapse into a singularity. Larger values of α^{th} cause the distributions in the sequence of approximations to be more similar, and therefore the sampling algorithm becomes more resiliant to population degeneracy problems. This, however, means that more sample and stages would be required to achieve the target approximation tolerance. In our case studies we found that values of α^{th} between 0.5 and 0.75 balance the performance and the

approximation quality well, while using even $N = 1000$ particles is usually adequate.

The PMMH sampler has also been trialled, and demonstrated to require significantly greater computational resources to perform. It was critical to uses large population particles as feasible in the bootstrap filter use for likelihood estimation. Even in the most favourable of setups this algorithm may struggle to achieve convergence. In our first case study we managed to achieve convergence, however the resulting posteriors seem to have lost the tails of the target distribution. It is an important disadvantage of a Metropolis-Hastings type of algorithm: they may fail to explore the parameter space efficiently and therefore underrepresent the tails of the posterior. We argue that in practice it is more important to cover a complete range of posterior support rather than to be falsely over confident around the mode.

The PMMH algorithm failed to converge to the stationary distribution in the majority of examples considered in the second case the study, despite being run for weeks. This motivates our recommendation to use ABC methods for performing inference over most stochastic models.

The ABC SMC approach demonstrates easily controllable characteristic that allow balancing the approximate quality and computational costs we demonstrate its performance in a realistically sized case study.

7.1 Limitations and Further Improvement

Approximate inference usually become more and more computationally expensive as smaller approximation error is desired. Population-based samplers (such as SMC) seem to be better suited for scaling up to more complex problems. Metropolis-Hastings based approaches are problematic to use on realistically sized studies. An important advantage of the ABC SMC is a potential for parallel computing implementation, that allows scaling it to larger models.

Future applications to complex stochastic models with large state spaces would certainly require such a parallel implementation. PMMH is inherently limited in scalability as the transitions of the underlying Markov chains are conditioned on their current state and therefore massive parallelisation is not possible for this family of approaches.

Performance of ABC methods may be significantly improved by employing summary statistics over the data and simulated traces at least at the early stages of analysis.

Particular attention and care must be exercised when selecting suitable summary statistics. A separate study might be required to evaluate appropriate choices of such summaries.

7.2 Software Implementation

The algorithms employed in this work were implemented using C and R. We have our own implementations of the ABC SMC, the PMMH and uniformization-based exact inference algorithms that are available from the author on request.

We used (Georgoulas et al., 2017) implementation of the Gibbs sampler for the Lotka-Volterra case study written in MATLAB scripting language. This code was obtained from GitHub at: https://github.com/ageorgou/roulette. We used bootstrap particle filter by Wilkinson (2011) in our PMMH implementation. This filter is available in the SMFS package for R.

Appendices

Appendix A

Additional Result from the Gibbs Sampler

A.1 Forward-Backward Algorithm

The forward-Backward algorithm is an inference method which calculates the posterior marginal distributions of all states X given a sequence of observations Y. This method computes the posterior distributions in two messages. The first message goes forward in time and known as the forward message, while the second message goes backward in time and known as the backward message. We follow the forward-Backward algorithm described in (Georgoulas et al., 2017).

Let us suppose that we have y_i observations at time $t_i, i = 1, \cdots, K$. A finite state space is described by X_n. Then, the forward and backward can be defined as vectors of size $|X|$, where only one message can be observed at each time unit. Hence, the forward message at the time unit t_i, denoted by a^i is:

$$a_n^i = \pi(y_1, \cdots, y_{i-1}, X_n),$$

where the previous equation represents the joint probability of the observations before the time t_i and the state at t_i is X_n. Hence, the probability of the observed time can be obtained from the backward messages b^i as:

$$b_n^i \propto \pi(X_n | y_i, \cdots, y_K)$$

These probabilities are then used in computing the acceptance ratio a where the

Algorithm 14 Forward-Backward algorithm

1: Calculate the transition probability between states based on parameter θ
2: For $i = 1, \cdots, K$, do
3: Calculate the forward message a^i.
4: end
5: Initialise $\pi \leftarrow 1$
6: Calculate the backward message b^i.
7: Search index n of observation y_i in X.
8: $\pi = \pi \cdot b_n^i$
9: end
10: Return π

acceptance step is required, as we are not sampling path from the exact posterior that was explained in section 3.8.

A.2 Additional Results

In this appendix, we present several results from repeating the experiments that compare the approximate marginal posterior obtained from ABC SMC algorithm to the exact posterior obtained from Gibbs sampler. For variety, each algorithm is performed given a synthetic data set that generated at different parameter rate (Figure 5.5), using the same set up for algorithm described in chapter 5.

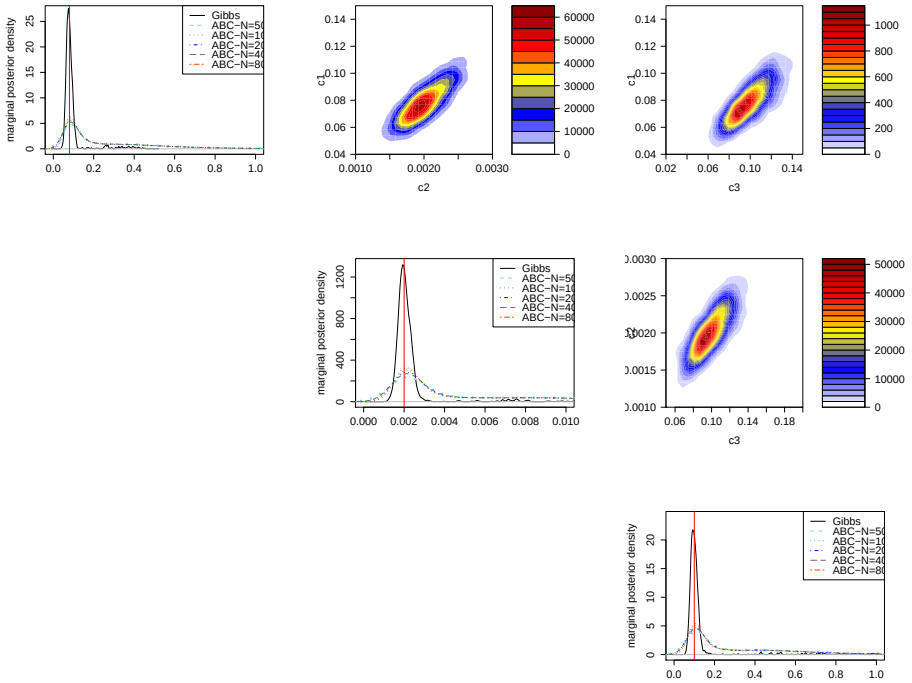


Figure A.1: The marginal posterior density of parameters of the LV model which are obtained by performing the Gibbs algorithm compared with approximated marginal posterior obtained from ABC SMC using the synthetic data set 3 (diagonal plots).

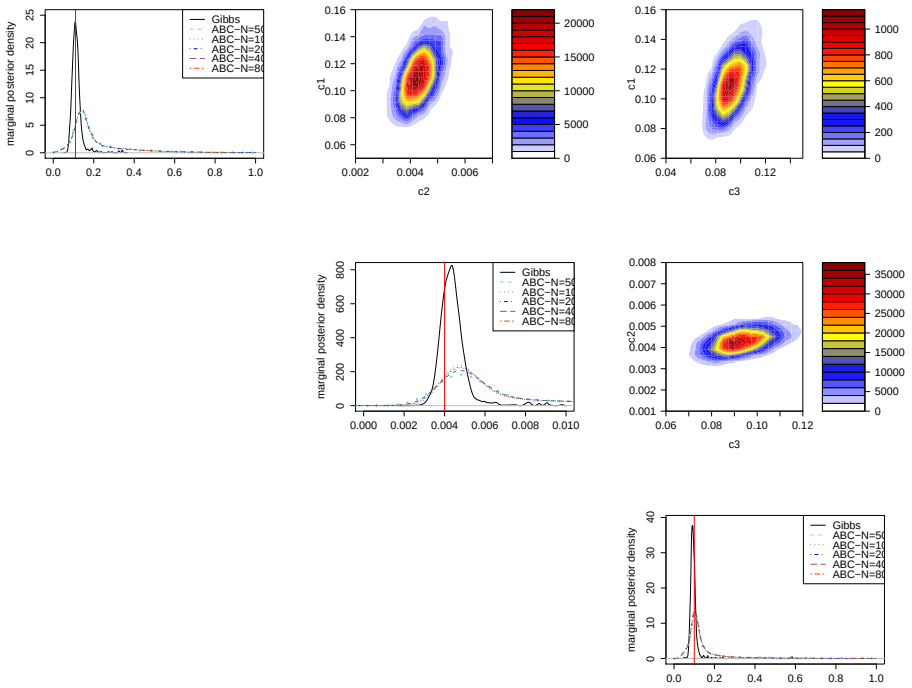


Figure A.2: The figure shows the marginal posterior density of parameters of the LV model which are obtained by performing the Gibbs algorithm compared with approximated marginal posterior obtained from ABC SMC using the synthetic data set 4 (diagonal plots).

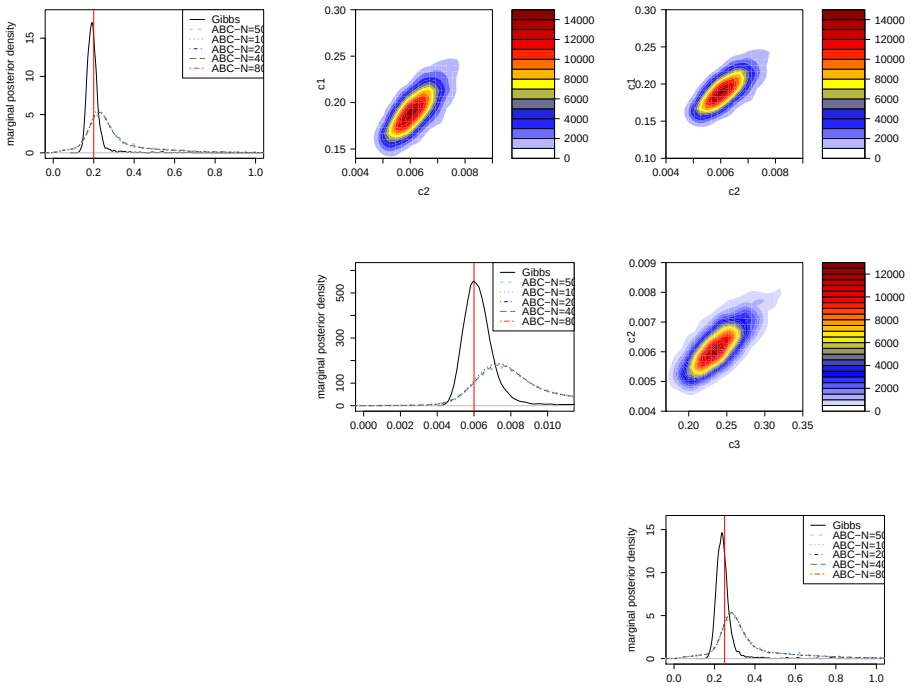


Figure A.3: The marginal posterior density of parameters of the LV model which are obtained from applying the Gibbs algorithm compared with approximated marginal posterior obtained from ABC SMC using the synthetic data set 5 (diagonal plots).

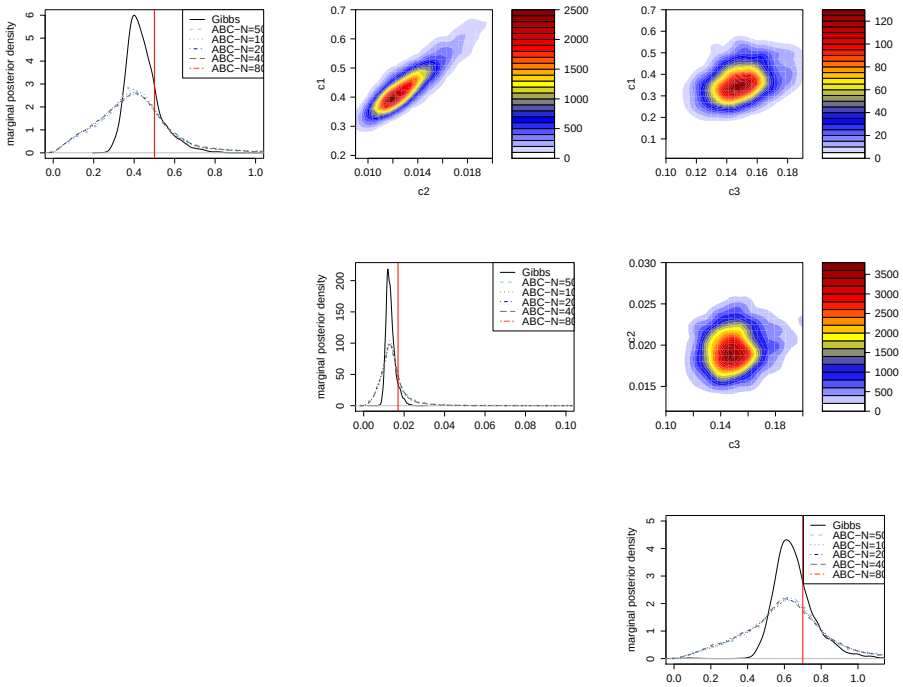


Figure A.4: The marginal posterior density of parameters of the LV model which are obtained by performing the Gibbs algorithm compared with approximated marginal posterior obtained from ABC SMC using the synthetic data set 6 (diagonal plots).

Appendix B

Implementation Details

In this Appendix, we provide some additional result from the experimental run that presented in section 5.4. The performance of the ABC SMC algorithm requires a pilot run to predefine the target tolerance for each individual synthetic data set (Figure 5.5). Each run of the ABC SMC with different N setting seems to require the different number of stages to achieve the goal. However, here we only present the result when the ABC SMC applied with $N = 8000$, the adaptive tolerance schedule for all experiments is chosen to be $\alpha^{th} = 0.5$. To maintain the diversity of the sampling path, the resampling step is carried out only when the ESS less a defined threshold which chooses to be $N/2$.

Table B.1: Summaries of running the ABC SMC sampler.

Experiments	Number of required stages	Time
Experiments 1	16	22 minutes
Experiments 2	18	57 minutes
Experiments 3	21	49 minutes
Experiments 4	18	23 minutes
Experiments 5	14	24 minutes
Experiments 6	11	27 minutes
Experiments 7	16	19 minutes
Experiments 8	12	22 minutes
Experiments 9	17	17 minutes

Appendix C

Additional Results for LV Model Parameter Inference

In this appendix, further performance to infer larger LV model parameter is carried out using the ABC SMC algorithm following a similar setting in section 5.8, given the different synthetic data set presented in Figure 5.3 and Figure 5.4. We provide list of figures represent the resulting marginal posterior distribution for each parameter C.2.

Additional experiments have been carried out to investigate the different choice of adaptive tolerance schedule. The performance of the ABC SMC algorithm with different choices of the adaptive tolerance schedule provides similar approximate posterior distribution as shown C.3

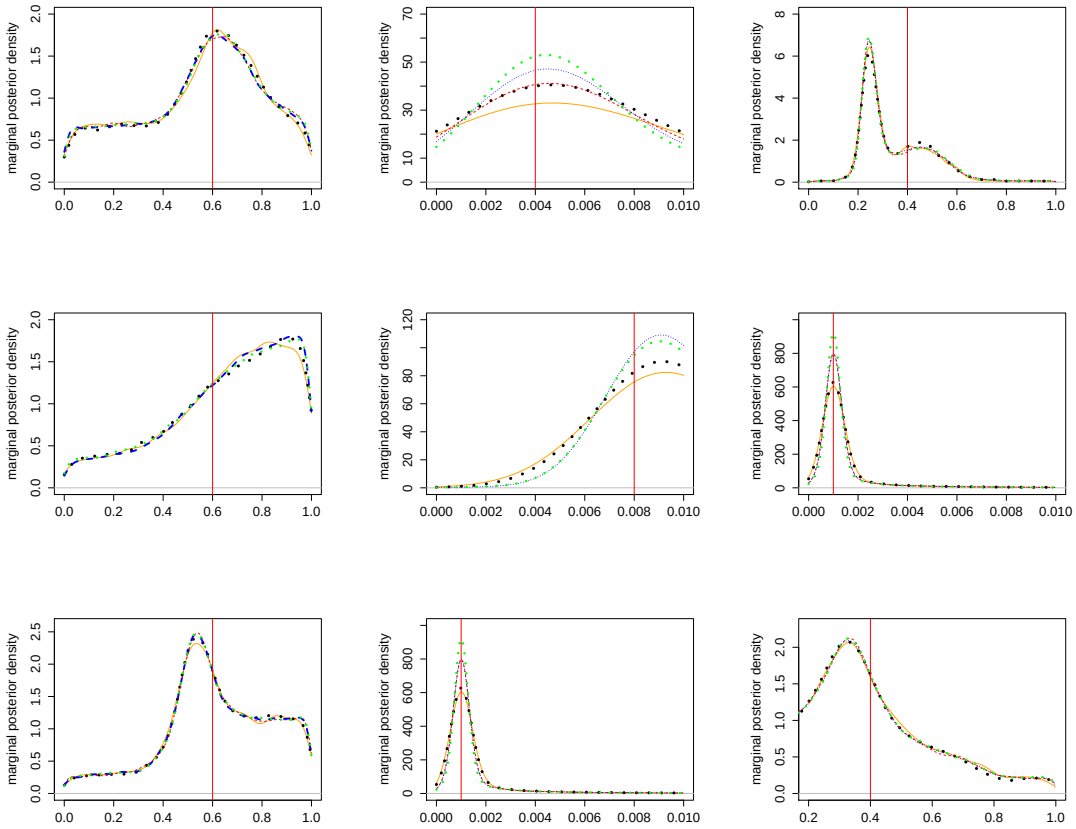


Figure C.1: The figure repressnts the marginal posterior density for the parameter of the LV model for 5 population size using data presented Figure 5.3, resulting from performing ABC SMC algorithm.

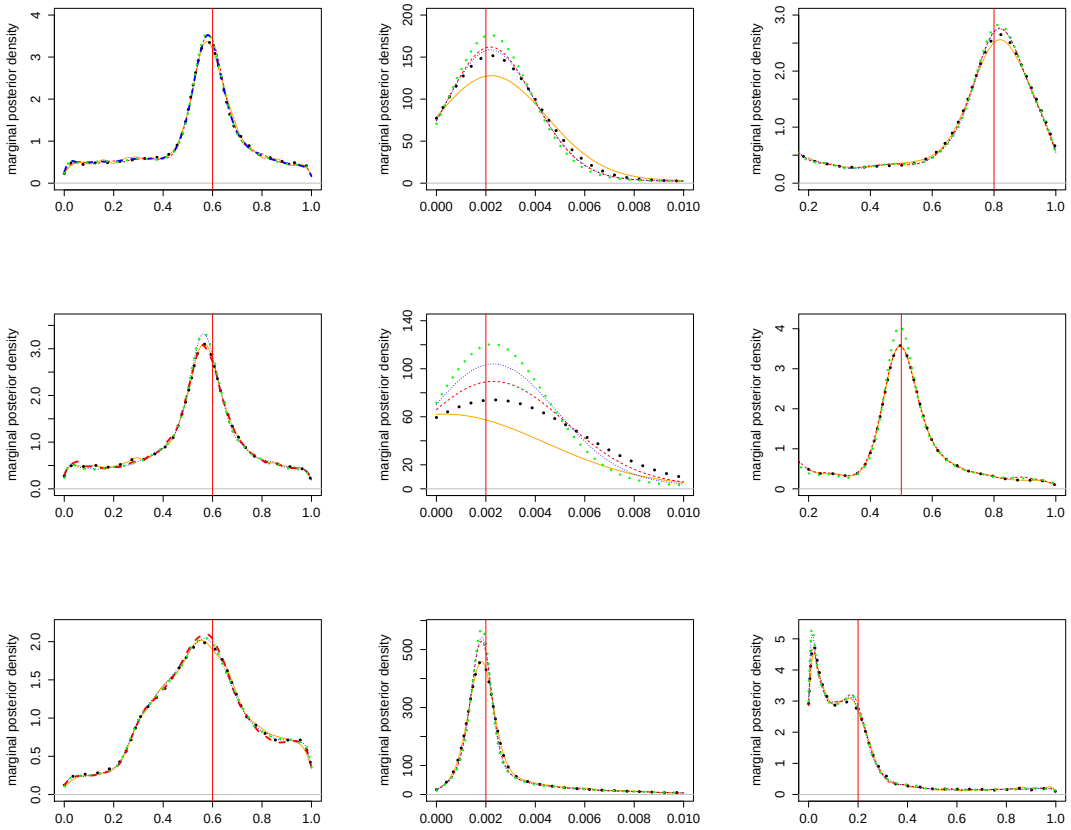


Figure C.2: The figure shows the marginal posterior density for the parameter of the LV model for 5 population size using data presented Figure 5.4, resulting from applying ABC SMC algorithm.

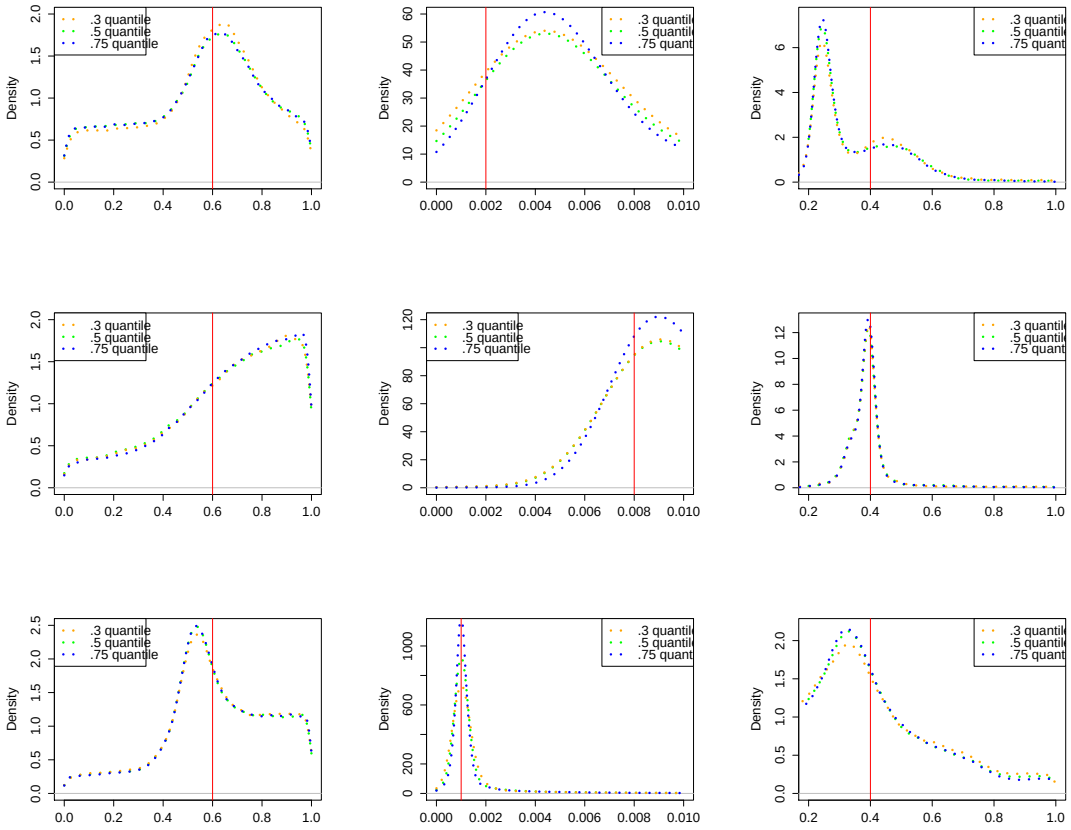


Figure C.3: The figure represents the approximated posterior density resulting from applying ABC SMC algorithm with three different quantile setting, the first row represent the inference given data set 4, the second row represents the inference given data set 5 and the third row represent the inference given data set 6.

Appendix D

Several Simulation from the LV Model

We continue this appendix by investigating the further implementation of the ABC SMC algorithm in a different range of data set. We start by simulating several data traces at the same parameter rate. According to Figure D.1 (first column), it is clear that the traces are not identical even though it simulated at the same reaction rate. This variability affects the ABC SMC method as it mainly relays on the distance between the simulated and observed data sets.

Then, one single data set will be used to perform inference. The algorithmic parameter is chosen based on previous experiments we have discussed in section 5.8.

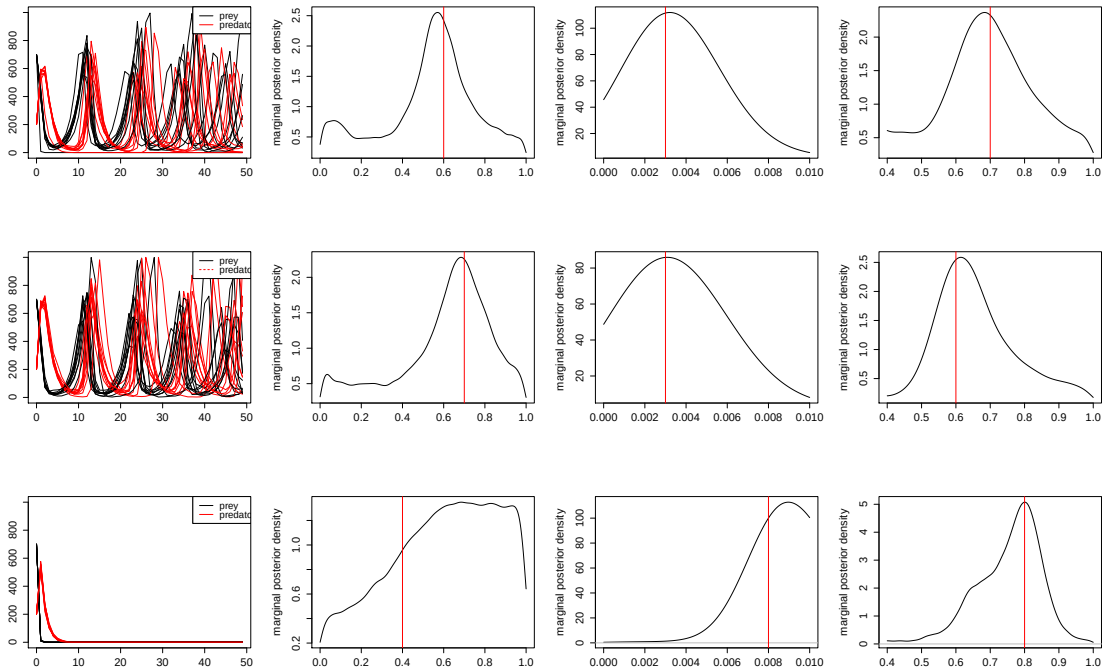


Figure D.1: The figure shows the approximated posterior density for LV model parameter with its corresponding data set.

Appendix E

Additional Result for the Repressilator Model

In this Appendix, additional plot from performing the PMMH for the Repressitor model is presented. We have found that the posterior distributions obtained from the PMMH algorithm are hardly diverge from the prior distributions.

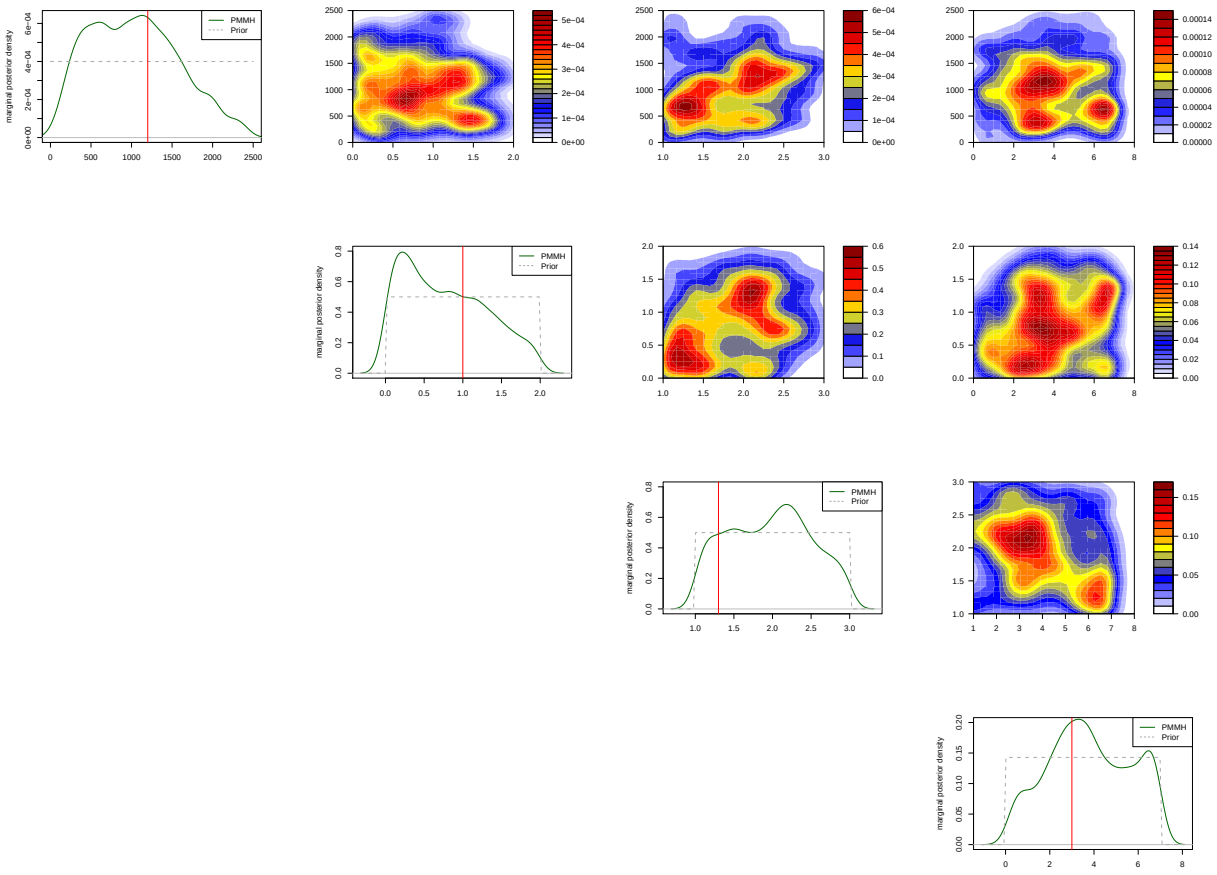


Figure E.1: The posterior densities of the Repressilator model parameters obtained from the PMMH sampler, given the synthetic data set generated at parameter $\theta = \{\alpha = 1200, \alpha_0 = 1, n = 1, \beta = 3\}$.

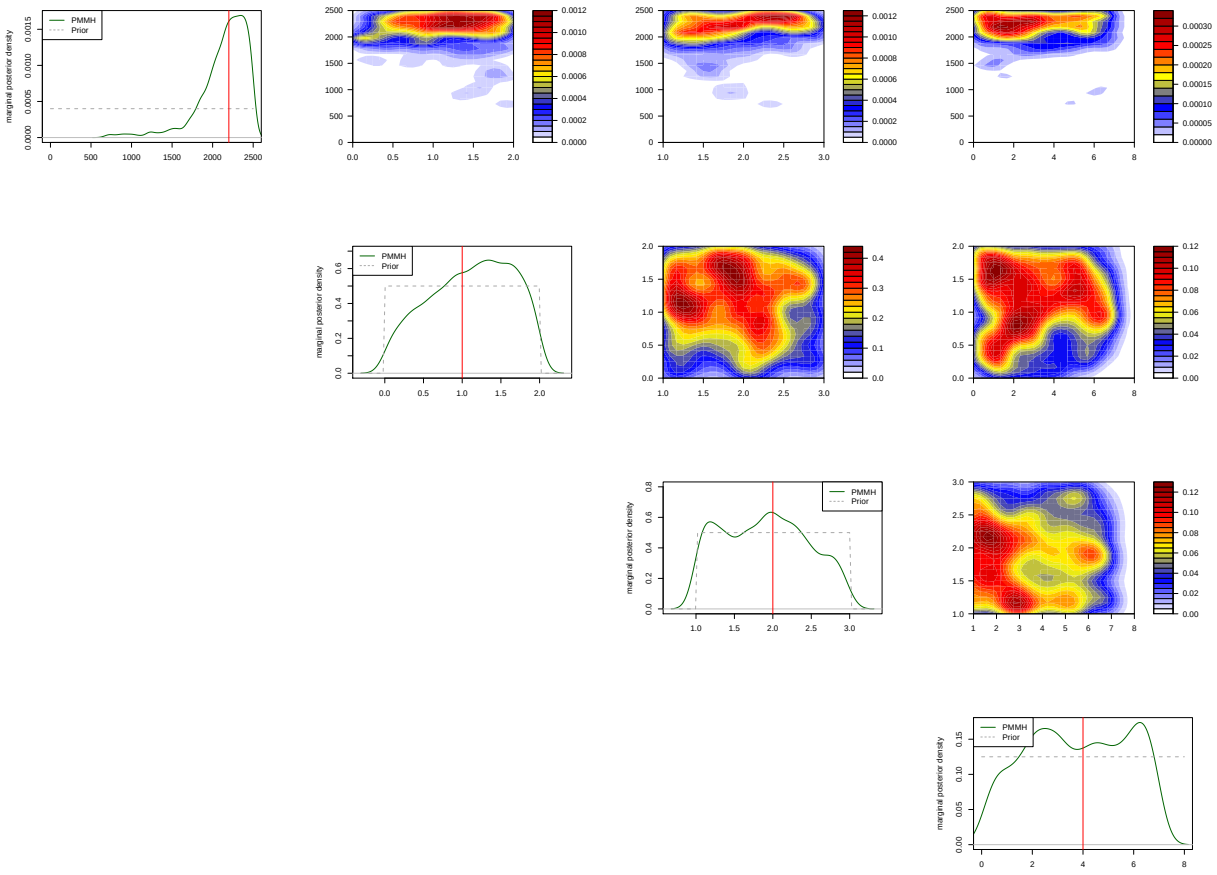


Figure E.2: The figure shows the posterior densities of the Repressilator model parameters obtained from performing the PMMH sampler, given the synthetic data set simulated with parameter $\theta = \{\alpha = 2400, \alpha_0 = 1, n = 2, \beta = 4\}$.

Bibliography

Al-Mohy, A. H. and N. J. Higham (2011). Computing the action of the matrix exponential, with an application to exponential integrators. *SIAM journal on scientific computing 33*(2), 488–511.

Anderson, D. F., G. Craciun, and T. G. Kurtz (2010). Product-form stationary distributions for deficiency zero chemical reaction networks. *Bulletin of mathematical biology 72*(8), 1947–1970.

Anderson, D. F. and T. G. Kurtz (2011). Continuous time markov chain models for chemical reaction networks. In *Design and analysis of biomolecular circuits*, pp. 3–42. Springer.

Andrieu, C. and Y. F. Atchadé (2006). On the efficiency of adaptive mcmc algorithms. In *Proceedings of the 1st international conference on Performance evaluation methodolgies and tools*, pp. 43. ACM.

Andrieu, C. and A. Doucet (2003). Online expectation-maximization type algorithms for parameter estimation in general state space models. In *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on*, Volume 6, pp. VI–69. IEEE.

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

Andrieu, C., É. Moulines, et al. (2006). On the ergodicity properties of some adaptive mcmc algorithms. *The Annals of Applied Probability 16*(3), 1462–1505.

Andrieu, C. and G. O. Roberts (2009). The pseudo-marginal approach for efficient monte carlo computations. *The Annals of Statistics*, 697–725.

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

Armitage, P., G. Berry, and J. N. S. Matthews (2008). *Statistical methods in medical research*. John Wiley & Sons.

Atchadé, Y. F., J. S. Rosenthal, et al. (2005). On adaptive markov chain monte carlo algorithms. *Bernoulli 11*(5), 815–828.

Bailey, N. T. (1990). *The elements of stochastic processes with applications to the natural sciences*, Volume 25. John Wiley & Sons.

Baker, C. T., G. Bocharov, J. M. Ford, P. M. Lumb, S. J. Norton, C. Paul, T. Junt, P. Krebs, and B. Ludewig (2005). Computational approaches to parameter estimation and model selection in immunology. *Journal of computational and applied mathematics 184*(1), 50–76.

Bayes, T., R. Price, and J. Canton (1763). An essay towards solving a problem in the doctrine of chances. *C. Davis, Printer to the Royal Society of London London, U. K*.

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

Beaumont, M. A., J.-M. Cornuet, J.-M. Marin, and C. P. Robert (2009). Adaptive approximate bayesian computation. *Biometrika 96*(4), 983–990.

Beaumont, M. A., W. Zhang, and D. J. Balding (2002). Approximate bayesian computation in population genetics. *Genetics 162*(4), 2025–2035.

Bernardo, J. M. and A. F. Smith (2001). Bayesian theory.

Bhattacharya, R. N. and E. C. Waymire (2009). *Stochastic processes with applications*. SIAM.

Boys, R. J., D. J. Wilkinson, and T. B. Kirkwood (2008). Bayesian inference for a discretely observed stochastic kinetic model. *Statistics and Computing 18*(2), 125–135.

Brooks, S. P. and A. Gelman (1998). General methods for monitoring convergence of iterative simulations. *Journal of computational and graphical statistics 7*(4), 434–455.

Calder, M., V. Vyshemirsky, D. Gilbert, and R. Orton (2006, Springer). Analysis of signalling pathways using continuous time Markov chains. *Transactions on Computational Systems Biology VI 4220*, 44–67.

Calderhead, B., M. Girolami, and N. D. Lawrence (2009). Accelerating bayesian inference over nonlinear differential equations with gaussian processes. In *Advances in neural information processing systems*, pp. 217–224.

Cappé, O., S. J. Godsill, and E. Moulines (2007). An overview of existing methods and recent advances in sequential monte carlo. *Proceedings of the IEEE 95*(5), 899–924.

Carpenter, J., P. Clifford, and P. Fearnhead (1999). Improved particle filter for nonlinear problems. *IEE Proceedings-Radar, Sonar and Navigation 146*(1), 2–7.

Chib, S. and E. Greenberg (1995). Understanding the metropolis-hastings algorithm. *The american statistician 49*(4), 327–335.

Chung, K. L. (1967). *Markov Chains with Stationary Transition Probabilities: 2d Ed.* Springer.

Cohn, I., T. El-Hay, N. Friedman, and R. Kupferman (2009). Mean field variational approximation for continuous-time bayesian networks. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pp. 91–100. AUAI Press.

Cowles, M. K. and B. P. Carlin (1996). Markov chain monte carlo convergence diagnostics: a comparative review. *Journal of the American Statistical Association 91*(434), 883–904.

Cox, D. R. (2017). *The theory of stochastic processes.* Routledge.

Csilléry, K., M. G. Blum, O. E. Gaggiotti, and O. François (2010). Approximate bayesian computation (abc) in practice. *Trends in ecology & evolution 25*(7), 410–418.

DeGroot, M. H. (2005). Optimal statistical decisions. *John Wiley & Sons 82*.

Del Moral, P. (1997). Nonlinear filtering: Interacting particle resolution. *Comptes Rendus de l'Académie des Sciences-Series I-Mathematics 325*(6), 653–658.

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

Del Moral, P., A. Doucet, and A. Jasra (2012). An adaptive sequential monte carlo method for approximate bayesian computation. *Statistics and Computing 22*(5), 1009–1020.

Del Moral, P., G. W. Peters, and C. Vergé (2013). An introduction to stochastic particle integration methods: with applications to risk and insurance. In *Monte Carlo and Quasi-Monte Carlo Methods 2012*, pp. 39–81. Springer.

Doob, J. L. and J. L. Doob (1953). *Stochastic processes*, Volume 7. Wiley New York.

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

Doucet, A., S. Godsill, and C. Andrieu (2000). On sequential monte carlo sampling methods for bayesian filtering. *Statistics and computing 10*(3), 197–208.

Doucet, A. and A. M. Johansen (2009). A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of nonlinear filtering 12*(656-704), 3.

Doucet, A., M. Pitt, G. Deligiannidis, and R. Kohn (2015). Efficient implementation of markov chain monte carlo when using an unbiased likelihood estimator. *Biometrika 102*(2), 295–313.

Drake, A. W. (1967). *Fundamentals of applied probability theory*. Mcgraw-Hill College.

Drovandi, C. C. and A. N. Pettitt (2011a). Estimation of parameters for macroparasite population evolution using approximate bayesian computation. *Biometrics 67*(1), 225–233.

Drovandi, C. C. and A. N. Pettitt (2011b). Likelihood-free bayesian estimation of multivariate quantile distributions. *Computational Statistics & Data Analysis 55*(9), 2541–2556.

Duflot, M., M. Kwiatkowska, G. Norman, and D. Parker (2006). A formal analysis of bluetooth device discovery. *International journal on software tools for technology transfer 8*(6), 621–632.

Elowitz, M. B. and S. Leibler (2000). A synthetic oscillatory network of transcriptional regulators. *Nature 403*(6767), 335–338.

Fearnhead, P., V. Giagos, and C. Sherlock (2014). Inference for reaction networks using the linear noise approximation. *Biometrics 70*(2), 457–466.

Filippi, S., C. P. Barnes, J. Cornebise, and M. P. Stumpf (2013). On optimality of kernels for approximate bayesian computation using sequential monte carlo. *Statistical applications in genetics and molecular biology 12*(1), 87–107.

Filippone, M. and R. Engler (2015). Enabling scalable stochastic gradient-based inference for gaussian processes by employing the unbiased linear system solver (ulisse). *arXiv preprint arXiv:1501.05427*.

Filippone, M. and M. Girolami (2014, IEEE). Pseudo-marginal bayesian inference for gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence 36*(11), 2214–2226.

Gardiner, C. (1986). Handbook of stochastic methods for physics, chemistry and the natural sciences. *Applied Optics 25*, 3145.

Gardiner, C. and P. Zoller (2004). *Quantum noise: a handbook of Markovian and non-Markovian quantum stochastic methods with applications to quantum optics*, Volume 56. Springer Science & Business Media.

Gelman, A., J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin (2013). *Bayesian data analysis*. CRC press.

Gelman, A., K. Shirley, et al. (2011). Inference from simulations and monitoring convergence. *Handbook of markov chain monte carlo*, 163–174.

Geman, S. and D. Geman (1984). Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on pattern analysis and machine intelligence* (6), 721–741.

Georgoulas, A., J. Hillston, and G. Sanguinetti (2017, Elsevier). Unbiased bayesian inference for population markov jump processes via random truncations. *Statistics and computing 27*(4), 991–1002.

Gilks, W. R., S. Richardson, and D. Spiegelhalter (1995). Markov chain monte carlo in practice. *Gilks, Walter R and Richardson, Sylvia and Spiegelhalter, David*.

Gillespie, D. T. (1976). A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *Journal of computational physics 22*(4), 403–434.

Gillespie, D. T. (1991). *Markov processes: an introduction for physical scientists*. Elsevier.

Gillespie, D. T. (1992). A rigorous derivation of the chemical master equation. *Physica A: Statistical Mechanics and its Applications 188*(1-3), 404–425.

Gillespie, D. T. (1996). Exact numerical simulation of the ornstein-uhlenbeck process and its integral. *Physical review E 54*(2), 2084.

Gillespie, D. T. (2001). Approximate accelerated stochastic simulation of chemically reacting systems. *The Journal of Chemical Physics 115*(4), 1716–1733.

Gillespie, D. T. (2007). Stochastic simulation of chemical kinetics. *Annu. Rev. Phys. Chem. 58*, 35–55.

Girolami, M. and B. Calderhead (2011). Riemann manifold langevin and hamiltonian monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 73*(2), 123–214.

Glynn, P. W. and D. L. Iglehart (1989). Importance sampling for stochastic simulations. *Management Science 35*(11), 1367–1392.

Golightly, A. and D. J. Wilkinson (2005). Bayesian inference for stochastic kinetic models using a diffusion approximation. *Biometrics 61*(3), 781–788.

Golightly, A. and D. J. Wilkinson (2011). Bayesian parameter inference for stochastic biochemical network models using particle markov chain monte carlo. *Interface Focus*, rsfs20110047.

Gordon, N. J., D. J. Salmond, and A. F. Smith (1993). Novel approach to nonlinear/non-gaussian bayesian state estimation. In *IEE Proceedings F (Radar and Signal Processing)*, Volume 140, pp. 107–113. IET.

Grimmett, G. and D. Stirzaker (2001). *Probability and random processes*. Oxford university press.

Gross, D. and D. R. Miller (1984). The randomization technique as a modeling tool and solution procedure for transient markov processes. *Operations Research 32*(2), 343–361.

Hajiaghayi, M., B. Kirkpatrick, L. Wang, and A. Bouchard-Côté (2014). Efficient continuous-time markov chain estimation. In *International Conference on Machine Learning*, pp. 638–646.

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

Haverkort, B., H. Hermanns, and J.-P. Katoen (2000, October). On the use of model checking techniques for dependability evaluation. *Proc. 19th IEEE Symposium on Reliable Distributed Systems (SRDS'00)*, 228–237.

Hol, J. D., T. B. Schon, and F. Gustafsson (2006). On resampling algorithms for particle filters. In *Nonlinear Statistical Signal Processing Workshop, 2006 IEEE*, pp. 79–82. IEEE.

Jasra, A., D. A. Stephens, A. Doucet, and T. Tsagaris (2011). Inference for lévy-driven stochastic volatility models via adaptive sequential monte carlo. *Scandinavian Journal of Statistics 38*(1), 1–22.

Karlin, S. and H. E. Taylor (1981). *A second course in stochastic processes*. Elsevier.

Kemeny, J. G., J. L. Snell, et al. (1960). *Finite markov chains*, Volume 356. van Nostrand Princeton, NJ.

Kenkre, V., E. Montroll, and M. Shlesinger (1973). Generalized master equations for continuous-time random walks. *Journal of Statistical Physics 9*(1), 45–50.

Kitagawa, G. (1996). Monte carlo filter and smoother for non-gaussian nonlinear state space models. *Journal of computational and graphical statistics 5*(1), 1–25.

Kitagawa, G. and S. Sato (2001). Monte carlo smoothing and self-organising state-space model. In *Sequential Monte Carlo methods in practice*, pp. 177–195. Springer.

Komorowski, M., B. Finkenstädt, C. V. Harper, and D. A. Rand (2009). Bayesian inference of biochemical kinetic parameters using the linear noise approximation. *BMC bioinformatics 10*(1), 343.

Kong, A., J. S. Liu, and W. H. Wong (1994). Sequential imputations and bayesian missing data problems. *Journal of the American statistical association 89*(425), 278–288.

Kot, M. (2001). *Elements of mathematical ecology*. Cambridge University Press.

Kwiatkowska, M. (2003). Model checking for probability and time: from theory to practice. In *Logic in Computer Science, 2003. Proceedings. 18th Annual IEEE Symposium on*, pp. 351–360. IEEE.

Kwiatkowska, M., G. Norman, and D. Parker (2007). Stochastic model checking. In *International School on Formal Methods for the Design of Computer, Communication and Software Systems*, pp. 220–270. Springer.

Lenormand, M., F. Jabot, and G. Deffuant (2013). Adaptive approximate bayesian computation for complex models. *Computational Statistics 28*(6), 2777–2796.

Liepe, J., C. Barnes, E. Cule, K. Erguler, P. Kirk, T. Toni, and M. P. Stumpf (2010). Abc-sysbio—approximate bayesian computation in python with gpu support. *Bioinformatics 26*(14), 1797–1799.

Link, W. A. and M. J. Eaton (2012). On thinning of chains in mcmc. *Methods in ecology and evolution 3*(1), 112–115.

Liu, B., Y. Zhang, and L. Chen (2005). The dynamical behaviors of a lotka–volterra predator–prey model concerning integrated pest management. *Nonlinear Analysis: Real World Applications 6*(2), 227–243.

Liu, J. S. and R. Chen (1998). Sequential monte carlo methods for dynamic systems. *Journal of the American statistical association 93*(443), 1032–1044.

Lotka, A. J. (1932). The growth of mixed populations: two species competing for a common food supply. *Journal of the Washington Academy of Sciences 22*(16/17), 461–469.

Lyne, A.-M., M. Girolami, Y. Atchadé, H. Strathmann, D. Simpson, et al. (2015, Institute of Mathematical Statistics). On russian roulette estimates for bayesian inference with doubly-intractable likelihoods. *Statistical science 30*(4), 443–467.

MacKay, D. J. (2003). *Information theory, inference and learning algorithms*. Cambridge university press.

Marin, J.-M., P. Pudlo, C. P. Robert, and R. J. Ryder (2012). Approximate bayesian computational methods. *Statistics and Computing*, 1–14.

Marjoram, P., J. Molitor, V. Plagnol, and S. Tavaré (2003). Markov chain monte carlo without likelihoods. *Proceedings of the National Academy of Sciences 100*(26), 15324–15328.

Marshall, A. W. (1954). The use of multi-stage sampling schemes in monte carlo computations. Technical report, RAND CORP SANTA MONICA CALIF.

Mauch, S. and M. Stalzer (2011). Efficient formulations for exact stochastic simulation of chemical systems. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB) 8*(1), 27–35.

McKinley, T., A. R. Cook, and R. Deardon (2009). Inference in epidemic models without likelihoods. *The International Journal of Biostatistics 5*(1).

McQuarrie, D. A. (1967). Stochastic approach to chemical kinetics. *Journal of applied probability 4*(3), 413–478.

Metropolis, N., A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller (1953). Equation of state calculations by fast computing machines. *The journal of chemical physics 21*(6), 1087–1092.

Metropolis, N. and S. Ulam (1949). The monte carlo method. *Journal of the American statistical association 44*(247), 335–341.

Milios, D., G. Sanguinetti, and D. Schnoerr (2017). Probabilistic model checking for continuous time markov chains via sequential bayesian inference. *arXiv preprint arXiv:1711.01863*.

Milner, P., C. S. Gillespie, and D. J. Wilkinson (2013). Moment closure based parameter inference of stochastic kinetic models. *Statistics and Computing 23*(2), 287–295.

Mishtal, A. and I. Arel (21012). Jensen-shannon divergence in ensembles of concurrently-trained neural networks. In *Machine Learning and Applications (ICMLA), 2012 11th International Conference on*, Volume 2, pp. 558–562. IEEE.

Moler, C. and C. Van Loan (1978). Nineteen dubious ways to compute the exponential of a matrix. *SIAM review 20*(4), 801–836.

Moler, C. and C. Van Loan (2003). Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM review 45*(1), 3–49.

Munsky, B. and M. Khammash (2006). The finite state projection algorithm for the solution of the chemical master equation. *The Journal of chemical physics 124*(4), 044104.

Murphy, K. P. (2012). Machine learning: A probabilistic perspective. adaptive computation and machine learning. MIT press.

Neal, R. M. (2003). Slice sampling. *Annals of statistics* (JSTOR), 705–741.

Neal, R. M. et al. (2011). Mcmc using hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo 2*(11), 2.

Norris, J. R. (1998). *Markov chains*. Number 2. Cambridge university press.

Opper, M. and D. Saad (2001). *Advanced mean field methods: Theory and practice*. MIT press.

Opper, M. and G. Sanguinetti (2008). Variational inference for markov jump processes. In *Advances in Neural Information Processing Systems*, pp. 1105–1112.

Owen, J., D. J. Wilkinson, and C. S. Gillespie (2015, Springer). Scalable inference for markov processes with intractable likelihoods. *Statistics and Computing 25*(1), 145–156.

Peters, G. W., Y. Fan, and S. A. Sisson (2012). On sequential monte carlo, partial rejection control and approximate bayesian computation. *Statistics and Computing 22*(6), 1209–1222.

Pitt, M. K., R. dos Santos Silva, P. Giordani, and R. Kohn (2012). On some properties of markov chain monte carlo simulation methods based on the particle filter. *Journal of Econometrics 171*(2), 134–151.

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

Pritchard, J. K., M. T. Seielstad, A. Perez-Lezaun, and M. W. Feldman (1999). Population growth of human y chromosomes: a study of y chromosome microsatellites. *Molecular Biology and Evolution 16*(12), 1791–1798.

Qiu, Q., Q. Wu, and M. Pedram (1999). tochastic modeling of a power-managed system: construction and optimization. pp. 194–199. ACM Press.

Quach, M., N. Brunel, and F. d'Alché Buc (2007). Estimating parameters and hidden variables in non-linear state-space models based on odes for biological networks inference. *Bioinformatics 23*(23), 3209–3216.

Raiffa, H. and R. Schlaifer. Applied statistical decision theory, harvard university, boston, 1961. *RaiffaApplied Statistical Decision Theory1961*.

Rao, V. and Y. W. Teh (2013, JMLR. org). Fast mcmc sampling for markov jump processes and extensions. *The Journal of Machine Learning Research 14*(1), 3295–3320.

Robert, C. and G. Casella (2010). *Introducing Monte Carlo Methods with R*. Springer Science & Business Media.

Robert, C. P. (2004). *Monte carlo methods*. Wiley Online Library.

Roberts, G. O., A. Gelman, W. R. Gilks, et al. (1997). Weak convergence and optimal scaling of random walk metropolis algorithms. *The annals of applied probability 7*(1), 110–120.

Roberts, G. O., J. S. Rosenthal, et al. (2001). Optimal scaling for various metropolis-hastings algorithms. *Statistical science 16*(4), 351–367.

Ross, S. M. (2014). *Introduction to probability models*. Academic press.

Rubin, D. B. (1987). The calculation of posterior distributions by data augmentation: Comment: A noniterative sampling/importance resampling alternative to the data augmentation algorithm for creating a few imputations when fractions of missing information are modest: The sir algorithm. *Journal of the American Statistical Association 82*(398), 543–546.

Rubin, D. B. (1988). Using the sir algorithm to simulate posterior distributions. *Bayesian statistics 3*, 395–402.

Ruttor, A. and M. Opper (2009). Efficient statistical inference for stochastic reaction processes. *Physical review letters 103*(23), 230601.

Schnoerr, D., R. Grima, and G. Sanguinetti (2016). Cox process representation and inference for stochastic reaction–diffusion processes. *Nature communications 7*, 11729.

Schnoerr, D., G. Sanguinetti, and R. Grima (2014). Validity conditions for moment closure approximations in stochastic chemical kinetics. *The Journal of chemical physics 141*(8), 08B616_1.

Schnoerr, D., G. Sanguinetti, and R. Grima (2017). Approximation and inference methods for stochastic biochemical kinetics?a tutorial review. *Journal of Physics A: Mathematical and Theoretical 50*(9), 093001.

Sisson, S. A., Y. Fan, and M. M. Tanaka (2007a). Sequential monte carlo without likelihoods. *Proceedings of the National Academy of Sciences 104*(6), 1760–1765.

Sisson, S. A., Y. Fan, and M. M. Tanaka (2007b). Sequential monte carlo without likelihoods. *Proceedings of the National Academy of Sciences 104*(6), 1760–1765.

Smets, P. (2008). Belief functions: the disjunctive rule of combination and the generalized bayesian theorem. In *Classic Works of the Dempster-Shafer Theory of Belief Functions*, pp. 633–664. Springer.

Steward, W. (2009). Probability, markov chains, queues, and simulation. Princeton University Press, UK.

Stewart, W. J. (1991). *Numerical solution of Markov chains*, Volume 8. CRC press.

Tavaré, S., D. J. Balding, R. C. Griffiths, and P. Donnelly (1997). Inferring coalescence times from dna sequence data. *Genetics 145*(2), 505–518.

Thomas, A., D. J. Spiegelhalter, and W. Gilks (1992). Bugs: A program to perform bayesian inference using gibbs sampling. *Bayesian statistics 4*(9), 837–842.

Thomas, P., H. Matuschek, and R. Grima (2012). Computation of biochemical pathway fluctuations beyond the linear noise approximation using ina. *arXiv preprint arXiv:1207.1631*.

Timmer, J., T. Müller, I. Swameye, O. Sandra, and U. Klingmüller (2004). Modeling the nonlinear dynamics of cellular signal transduction. *International Journal of Bifurcation and Chaos 14*(06), 2069–2079.

Toni, T., D. Welch, N. Strelkowa, A. Ipsen, and M. P. Stumpf (2009). Approximate bayesian computation scheme for parameter inference and model selection in dynamical systems. *Journal of the Royal Society Interface 6*(31), 187–202.

Van Kampen, N. G. (1992). Stochastic processes in physics and chemistry. *Elsevier 1*.

Vyshemirsky, V. and M. Girolami (2008). Biobayes: a software package for bayesian inference in systems biology. *Bioinformatics 24*(17), 1933–1934.

West, M. (1996). *Bayesian forecasting*. Wiley Online Library.

Whittle, P. (1986). *Systems in stochastic equilibrium*. John Wiley & Sons, Inc.

Wilkinson, D. (2010a). The pseudo-marginal approach to "exact approximate" mcmc algorithms.". *Web (url: http://darrenjw. wordpress. com/2010/09/20/the-pseudo-marginal-approach-to-exactapproximate-mcmc-algorithms/)*.

Wilkinson, D. J. (2010b). Parameter inference for stochastic kinetic models of bacterial gene regulation: a bayesian approach to systems biology. In *Proceedings of 9th Valencia International Meeting on Bayesian Statistics*, pp. 679–705.

Wilkinson, D. J. (2011). *Stochastic modelling for systems biology*. CRC press.

Wilkinson, R. D. (2013). Approximate bayesian computation (abc) gives exact results under the assumption of model error. *Statistical applications in genetics and molecular biology 12*(2), 129–141.

Zechner, C., M. Unger, S. Pelet, M. Peter, and H. Koeppl (2014). Scalable inference of heterogeneous reaction kinetics from pooled single-cell recordings. *Nature methods 11*(2), 197.

Zheng, Q. and J. Ross (1991). Comparison of deterministic and stochastic kinetics for nonlinear systems. *The Journal of chemical physics 94*(5), 3644–3648.