Scalable Bayesian inference for stochastic epidemic processes

Martin Joseph Burke

PhD

University of York

Environment

September 2021

Abstract

- The research reported in this thesis is motivated by the goal of using mathematical models to better understand the within-herd disease dynamics of Bovine tuberculosis (BTB) in UK cattle.

- This led to the development of new Bayesian methods and tools, including an open-source software package for Bayesian data analysis. In particular, those applicable to Discrete-state-space Partially Observed Markov Processes (DPOMP models).

- These were applied to the problem of model and parameter inference for a sample of individual herds selected from UK BTB surveillance records.

- Those findings led to the alternative models and methods utilised in the penultimate chapter, where we present a large scale, system-of-herds model and report novel parameter estimates for BTB. The latter include those that relate to disease detection; regional background risk; and farmer behaviour (specifically, the trading of live cattle).

- The work goes beyond previous, similar published research in three ways: it incorporates individual herd records, not just aggregated data; it includes formal methods of model assessment; that work is (partially) extended to systems comprising up to thousands of herds.

Contents

List of Tables

List of Figures

Acknowledgements

I'd like to thank my supervisors Mike Hutchings, Ross Davidson, Piran White, and most especially Glenn Marion who has supported me at every turn in a long and interesting journey. (All grammatical errors are mine despite Glenn's best efforts.) Also my mentor Giles Innocent for his time, patience and good humour. Jamie Prentice is another colleague who deserves a special mention. Like Glenn, he has reviewed and tested multiple iterations of the software presented in Chapter 5 and given much helpful guidance and advice besides. I'm also grateful to other colleagues at Biomathematics and Statistics Scotland (BioSS) for hosting me, as well as those based nearby in the University of Edinburgh maths department for the tea, biscuits, sympathy, advice and opportunities.

I'm indebted to the University of York for permitting me to postpone submission of this thesis in order to focus on a COVID-related project as part of my role at BioSS. Although my own contribution was modest, their patience and understanding afforded me the opportunity to apply knowledge gained over the course of the project to work of vital and current importance, for which I'm grateful.

I'd like to acknowledge authors such as Pooley, Bishop, Marion, Chopin, Doucet, Johansen, Gelman and others for contributing tools and other literature which have been of tremendous utility to this project and my personal learning [4, 9, 10, 11]. I'm also grateful to Art B Owen for delivering a series of lectures on Quasi Monte Carlo [12] at CompStat 2018[1] in which (I think) he suggested the essential idea for the ARQ-MCMC algorithm presented in Chapter 3 to me and many others. Any errors and misconceptions in the attempted implementation of that idea are mine.

Finally I'm very grateful to my examiners, Dr Julia Touza-Montero and Dr TJ McKinley, for their kind words, encouragement and contributions throughout, but especially in Chapter 6.

[1] And to Murray Pollock and others at the University of Warwick for hosting CompStat.

Declaration

I declare that this thesis is a presentation of original work and I am the sole author. This work has not previously been presented for an award at this, or any other, University. All sources are acknowledged as References.

Chapter 1

Introduction

"The movement of events is often as wayward and incomprehensible as the course of human thought, and this is why we ascribe to chance whatever belies our calculation."

-Pericles

Summary

- The research reported in this thesis is motivated by the goal of using mathematical models to better understand the within-herd disease dynamics of Bovine tuberculosis (BTB) in UK cattle.
- This led to development and application of methodology and software for statistical inference for such models and in particular for Discrete-state-space Partially Observed Markov Processes (DPOMP models).
- This brief introductory chapter begins by providing background information concerning this primary motivation. It then describes the overall aims and objectives, and outlines the structure of the remainder of the main text.

1.1 Problem statement and motivation

"Scientific findings indicate that the rising incidence of disease can be reversed, and geographical spread contained, by the rigid application of cattle-based control measures alone."

- Professor John Bourne, Chairman of the [UK] Independent Scientific Group on Cattle Tb [13]

1.1.1 Background

Bovine tuberculosis (BTB) is a bacterial infectious disease caused by *Mycobacterium bovis*, which for the benefit of the uninitiated is a slow-growing, nonchromogenic acid fast bacillus [14]. It affects cattle and a wide range of other mammals including humans, badgers, deer, goats and pigs. The primary routes of infection are through the respiratory and gastrointestinal tracts. There is no test that can reliably deduce the route by which an animal was infected, though this is sometimes indicated post-mortem by the presence of lesions in specific internal organs. In terms of financial losses BTB has been reported to account annually for an estimated USD 3 billion globally, of which approximately USD 130 million was attributed to the UK in c.2006 [14, 15].

BTB in cattle is an endemic disease that presents significant [statistical] modelling challenges, in part because the nature of transmission is poorly defined. Other sources of uncertainty include trade; other extraneous factors such as the possible presence of wildlife reservoirs; and the aforementioned (i.e. 'slow-growing') long generation time associated with *M. Bovis* –approximately 16 to 20 hours [16]. The latter complicates the task of clearly identifying patterns of infection due to the time elapsed between exposure and [at least indirectly] observable indicators of disease, such as 'shedding' [17, 18].

The control of BTB is an interesting and often controversial topic. However it is important to note that the 'controversy' surrounding BTB does not appear to be related any significant discord in the scientific community, at least concerning the overall importance of wildlife reservoirs (and by extension, the relevance of measures such as badger culling). On the contrary, while much about the nature and role of inter-species transmission (in both directions) remains unknown, intra-species (cattle-to-cattle) transmission is generally accepted to be far more common.

In his report to the Secretary of State for Environment, Food and Rural Affairs in 2007 (as quoted at the top of this chapter) Professor John Bourne of the Independent Scientific Group on Cattle TB noted the implications of this fact with respect to control measures: those that target cattle-to-cattle transmission, *alone*, would likely be sufficient to systematically reduce BTB incidence on UK cattle farms. This implies, at the least, that the dynamics of BTB cannot be adequately addressed *without* considering how cattle-to-cattle transmission (and relevant control measures) feature in the problem.

1.1.2 Modelling within-herd BTB dynamics

Within-herd BTB transmission is the most obvious form of cattle-to-cattle transmission, since herds are usually managed so as to keep them largely isolated from one another. An important exception is inadvertent import/export (domestically and otherwise) of disease through trade. Nevertheless the primary focus of initial chapters is within-herd transmission, with consideration of external sources of infection (and related farmer behaviour) postponed until Chapter 7.

The difficulties associated with statistical analysis of within-herd transmission of BTB (i.e. the sources of uncertainty noted above) are evidenced by the wide range of estimates reported for important epidemiological parameters like the 'latent' period [17, 2, 15]. This is discussed further in Chapter 6. Details of the models and methods developed to address this problem are also described in subsequent chapters. For now, key aspects of the broad approach adopted throughout are illustrated in Figure 1.1. The image is easily recognisable as a (somewhat) standard epidemiological model.

The rates that govern state transitions in the model, which we can think of as the migration of individuals between 'compartments', are shown as mathematical expressions (between the boxes) that depend on the numbers of individuals in each state, and are given for each type of transition, or 'event' (this overall process is referred to throughout as 'migration'). The rates are also partially determined by 'model parameters', represented by letters of the Greek alphabet. For example the 'contact' parameter is denoted by β and influences the rate of new infections.

A useful application of *Bayesian inference* (see Chapter 2) is to estimate, or *infer*, [the likely distribution of] such model parameters from data. For example, if we ignore the possibility of extraneous migration (other than the presumed removal of test-positive animals – not shown) then we could 'fit' the illustrated model using within-herd BTB 'surveillance' data. Then, in as far as the model is adequate, the inferred parameters could be said to characterise *within-herd BTB dynamics*.

$$\begin{tikzcd}
\boxed{\mathrm{S}} \arrow[r, "\beta \mathrm{SI}"] & \boxed{\mathrm{E}} \arrow[r, "\gamma_1 \mathrm{E}"] & \boxed{\mathrm{T}} \arrow[r, "\gamma_2 \mathrm{I}"] & \boxed{\mathrm{I}}
\end{tikzcd}$$

Figure 1.1: **Compartmental modelling.** The Susceptible-Exposed-Test sensitive-Infectious (SETI, also referred to as the 'SORI' [2]) model is one of a number of similar compartmental models proposed for BTB. Conceptualised as a continuous-time, discrete state-space model, individuals transition in sequence between states according to specified event rates. The states correspond to: susceptible to the disease S; exposed but neither infectious or detectable by available diagnostic tests E; test-sensitive but not yet infectious T; and infectious I. Furthermore, model state-space is the vector describing the numbers of individuals in each state $\{S, E, T, I\}$.

1.1.3 Study design

In contrast to standard experimental design, that typically poses a specific scientific question of data that we *intend* to collect, the nature of observational studies leads to a more general question:

> *'What interesting scientific question[s] can be addressed, given the data that are already available?'*

The design of such a study is thus emergent, beyond the need to evaluate the data; clean it; and process it for statistical inference. Another important consideration is the selection of appropriate models and methods. For example, the broad approach adopted in this thesis is referred to as *Bayesian data analysis* [19, 11].

The overall objectives that have been identified for this project, including how they relate to specific portions of the thesis, are laid out in the next section. The [observational] data available primarily consists of BTB surveillance records that (purportedly) include the number of tests, positive reactors (i.e. animals that are test positive for BTB) herd size and other certain other pertinent information. The initial aims for the project include fitting models such as the one shown in Figure 1.1

to data on *individual* herds, which to the best of our knowledge this has not yet been done by others who have analysed this data. One benefit of analysing individual herds is that it ought to allow a more detailed characterisation of the disease process compared with less fine-grained epidemiological modelling approaches that have been utilised to date to analyse the same or similar data [17, 2, 15].

1.2 Aims and objectives

We now enumerate the specific aims of this work and explain the layout of the thesis.
The key aims and objectives are:

1. To characterise within-herd BTB dynamics in UK cattle herds using the available data. In the first instance, to carry out Bayesian parameter inference based on conventional epidemiological models for a selection of individual herds.

2. To extend (1) to the problem of multi-model inference; to discover which models best fit the data, and formally evaluate those results using the appropriate Bayesian methods.

3. Ultimately, to apply the findings of (1) and (2) towards to a large-scale system-of-herds model, up to and including national level (i.e. many thousands of herds).

4. Building on the above to develop [Bayesian] tools and methods that are of potential value to the wider scientific community.

The extent to which any of these aims were met is summarised in the concluding chapter of the thesis. However the first and second are at least *addressed* directly in Chapter 6. The third, in Chapter 7. The last is arguably addressed in two different senses, first by the algorithms and Bayesian workflow presented in Chapters 3 and 4. Secondly by the implementation of those methods as an open-source software package, as discussed in Chapter 5. Both those units of work rely heavily on the methods and concepts introduced in Chapter 2.

1.3 Thesis structure

The organisation of the thesis is informed (to some extent) by the way the project itself was planned and executed, with the selection, development, and implementation of methods covered in early chapters, and the application of those methods to the motivating problem covered in later ones. This is summarised visually in Figure 1.2.

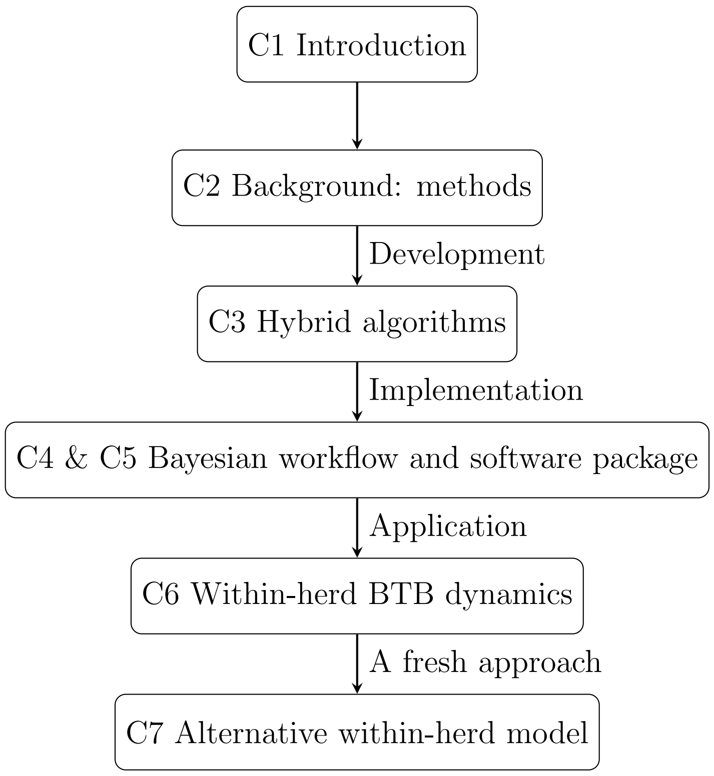


Figure 1.2: Thesis structure. This overview essentially represents the original plan w.r.t. to the initial chapters. Additional work was conducted as a result of the findings presented up to Chapter 6, as laid out in a subsequent chapter.

Chapter 2 introduces key underlying methods and notation relied on throughout, covering both development of stochastic models for disease transmission and a wide range of approaches and specific methods for statistical inference that can and have been applied to such models.

Chapter 3 introduces two novel methods for statistical inference that derive their strengths through combination of distinct approaches to inference normally considered separately. The following chapter incorporates them with the concept of Bayesian Workflows. This is intended to ensure robust analysis of available data by detecting potential pitfalls and enabling the practitioner to correct them.

In Chapter 5 a software package is introduced that enables users to define, simulate and perform inference on a wide range of stochastic models using a simple syntax and interface. This package provides access to a wide range of inference methods introduced in previous chapters and can be used to implement the Bayesian Workflow described in Chapter 4.

The computational and inferential limits of these methods are tested in Chapter 6 through application to data sets describing within-herd BTB testing data from UK cattle farms. In response to the computational challenges encountered, Chapter 7 introduces inference for a more tractable model formulation of within-herd BTB dynamics and applies this to data at regional scale. Finally Chapter 8 reviews achievements and lessons learned and suggests avenues for further research.

Chapter 2

Background: methods

"The combination of Bayes and Markov Chain Monte Carlo has been called 'arguably the most powerful mechanism ever created for processing data and knowledge"'

- Sharon Bertsch McGrayne [20]

Summary

- This chapter introduces key concepts and notation used throughout the thesis.
- It presents the algorithms that are applied and further developed to obtain the results and new models and methods presented in later chapters.
- It introduces DPOMP models of within-herd disease dynamics in UK cattle herds and methods for *Bayesian inference* applicable to such models.
- Finally, it shows relationships between such methods and highlights those directly relevant to the generalised implementations (i.e. software packages), produced throughout the course of the project, and described in Chapter 5.

Overview

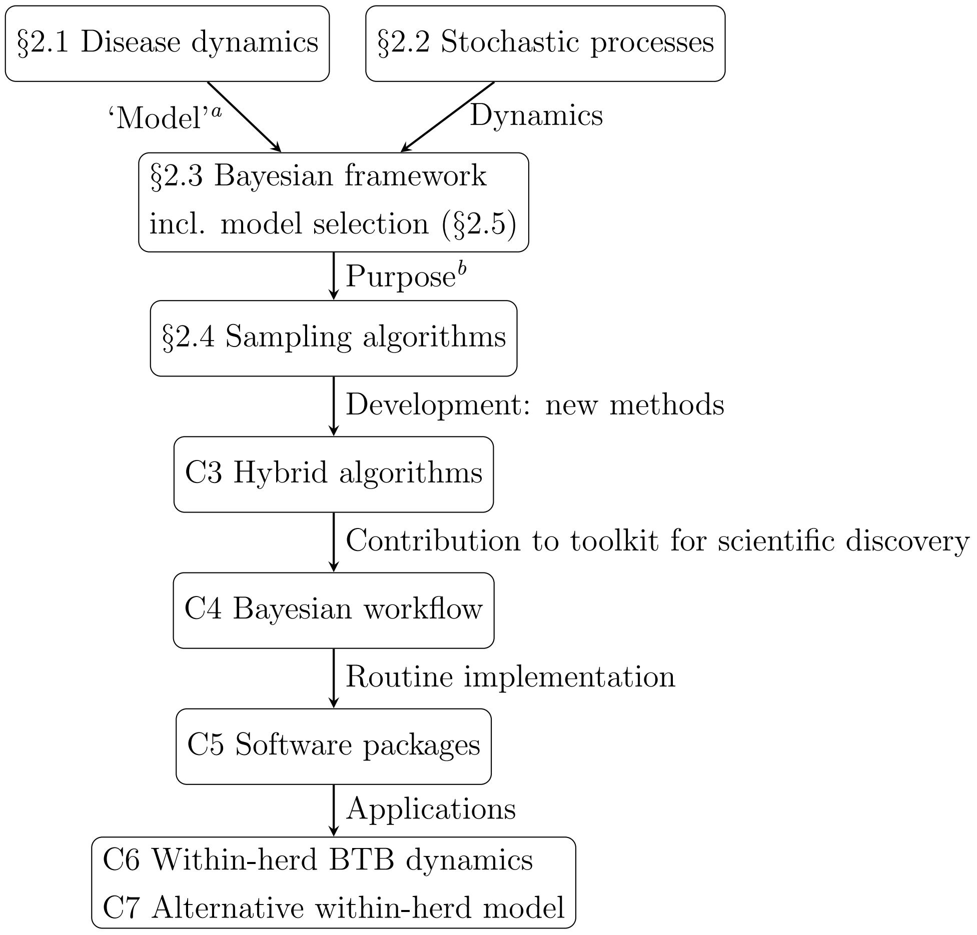
The purpose of this chapter is to introduce the concepts and notation required to motivate and understand the technical developments and applications presented later in this thesis (See Figure 2.1 for an overview). Section §2.1 introduces key concepts, terminology and notation used in dynamic epidemiological modelling of disease transmission. Section §2.2 introduces relevant material on stochastic processes. The Bayesian framework for single-model inference is described in §2.3. However, the greater part of the chapter is devoted to a description of a wide range of computational approaches to sampling algorithms used to implement Bayesian inference in practice (see §2.4). Finally in Section §2.5 describe how the Bayesian framework is readily extensible to multi-model inference and show how some of the algorithms introduced in §2.4 enable implementation of this.

2.1 Introduction: epidemiological modelling

Using the canonical example of the Kermack-McKendrick SIR model [3] this section introduces the key ideas of pathogen transmission and the dynamics of progression through discrete disease states. Together with the concepts stochastic process models introduced in §2.2 –which provide a mathematical foundation for the dynamics of the models introduced in this section– these ideas provide the basis for epidemiological applications of the Bayesian framework for statistical inference (see §2.3) described in the latter sections of this chapter and beyond.

2.1.1 Key concepts in compartmental modelling

Models like the standard susceptible-infectious-recovered (SIR) model depicted in Figure 2.2 are easily recognisable due to their ubiquity in science and statistics. They are sometimes referred to using the general term ‘compartmental models, or else as *state-space models* (SSM) especially within population modelling and ecology [21, 22, 23]. In a *discrete* SSM, individuals are said to take one of n discrete state values and transition between them, according to the mathematical (e.g. probabilistic) statements, which are therefore held to govern dynamics within the modelled system.

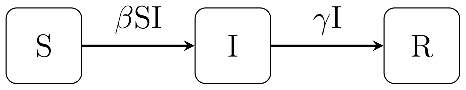


[a]implemented as deterministic or (as here) stochastic process
[b]i.e. functional purpose – the technical motivation here.

Figure 2.1: **Key concepts and their relation to thesis**. This chapter begins with a brief introduction to the class of epidemiological *state-space-models* considered throughout the thesis, along with the *stochastic processes* used to simulate them. Next Bayesian inference and the algorithms used to implement are introduces. Together these provide the foundational basis for the methods and applied problems considered later in the thesis.

Deterministic vs. stochastic models

Model 'dynamics' is used throughout simply to refer to the time-evolution of the system. In other words, the change in the overall system state, over time. In mathematical terms this could be described for the SIR model using a set of ordinary



$$\begin{tikzcd} \boxed{S} \arrow[r, "\beta SI"] & \boxed{I} \arrow[r, "\gamma I"] & \boxed{R} \end{tikzcd}$$

Figure 2.2: **Example of an epidemiological state-space model**. The susceptible-infectious-recovered, or SIR, model [3]. The rate quantities given here can be interpreted in different ways –probabilistic or deterministic– but in either sense they are mathematical expressions that govern the time-evolution of the system. For example, in a discrete-state-space models *individuals* take only one value of a discrete set of states $\{S, I, R\}$. Allowed transition between states (arrows) occur according to rates determined by the system state-space vector (S, I, R) representing the numbers of individuals in these states at a given time and model parameters. Here the parameter β is the transmission or *contact* rate and γ is the *recovery* rate.

differential equations:

$$\frac{dS}{dt} = -\beta SI \tag{2.1}$$

$$\frac{dI}{dt} = \beta SI - \gamma I \tag{2.2}$$

$$\frac{dR}{dt} = \gamma I \tag{2.3}$$

This is a deterministic representation of the model. Alternatively, model dynamics may be expressed in probabilistic terms – a 'stochastic' model (i.e. with a probabilistic element). For example, this can be achieved by developing *stochastic differential equations* through the addition of so-called noise terms to the system of equations. In the Euler discretisation of the above equations with time step Δt these noise terms would be proportional to i.i.d. Gaussian random variables (one for each time-step) with mean zero and variance Δt. The (tricky) limit as $\Delta t \to 0$ leads to the formal definition of stochastic differential equations. An alternative probabilistic approach, that is perhaps more intuitive, are discrete-state stochastic models that count individuals in a finite number of states (or compartments) and define the probabilities that individuals make a transition from one state to another. These models can be in discrete or continuous time and are described in detail in §2.2.

Discrete vs continuous time models

Finally, time-evolution of a system can be modeled on the basis of discrete-time intervals (e.g. a *Markov model*) or else represented in continuous-time (e.g. *Markov process*). The Markov property (see box) is widely used throughout this thesis and within the field of Bayesian inference more generally; as in *'Markov chain Monte Carlo'* (see §2.4.1). However, it is important to note that non-Markovian models are also possible, and often more parsimonious e.g. than a Markovian model with expanded state space. An example of a non-Markovian model is the stochastic process described in §2.2.3 that was used by [24] in conjunction with an SIR model to analyse the behaviour of social media users. The same is true of the 'STR' model that is utilised to good effect in the penultimate chapter of this thesis.

Markov property: *When the evolution of a modelled system is dependent only on the 'current' state, as opposed to past states or even the entire history of the system, it is said to be* ***Markovian*** *(or 'memoryless').*

2.1.2 Modelling 'latent' infection (S*E*IR)

The SIR model depicted in Figure 2.2 can be arbitrarily adapted or extended to include any number of states and state-transition possibilities. This is useful for representing a diverse range of real-world systems, which naturally includes many different varieties of epidemiological model. One of the most well known –the SEIR model as depicted in Figure 2.3– extends the standard SIR model to include an 'exposed' state. It is typically intended to represent a latent non-infectious phase of disease, where 'latent' simply means 'unobserved'. The precise meaning implied by this usage (i.e. in an epidemiological *modelling* context, e.g. this section of the chapter) is similar to the way it is used in *clinical* epidemiological literature to mean 'asymptomatic'. However, note that this usage is subtly distinct from the way it is used to describe the concept of *'latent processes'* in applied statistics (see e.g. section §2.3.2, of this chapter) where it is used to mean unobserved.

Latency and *reinfection* in standard epidemiological models

Standard epidemiological models that account for a latent non-infectious period, typically assume that individuals within the ['exposed'] state are no longer suscep-

$$\begin{tikzcd}\boxed{\mathrm{S}} \arrow[r, "\beta \mathrm{SI}"] & \boxed{\mathrm{E}} \arrow[r, "\gamma_1 \mathrm{E}"] & \boxed{\mathrm{I}} \arrow[r, "\gamma_2 \mathrm{I}"] & \boxed{\mathrm{R}}\end{tikzcd}$$

Figure 2.3: **The susceptible-exposed-infectious-recovered SEIR model**. An extension of the basic SIR model, which accounts for an 'exposed' state to represent a latent, non-infectious phase of disease. As before, individuals can assume one of a discrete set of states $\{S, E, I, R\}$, and transition between states proceed according to the specified rates that depend on the numbers of individuals in each state, denoted (S, E, I, R).

tible to infection. This precludes the possibility of *reinfection*, where by latently infected individuals *are* still susceptible to [further] infection.

Of course, the simpler (no-reinfection) assumption may provide an adequate representation of disease dynamics even in situations where we believe there is (possibly widespread, or 'endemic') latent infection *with* the possibility of reinfection (such as BTB, or the COVID-19 virus [6]). We can rationalise this choice by interpreting the 'exposed' state to reflect only individuals who have been sufficiently 'exposed' such that their chances of progressing to clinical disease [in the reasonably near-term future] is fairly high, rather than to account for *all* primary infections. This working definition lies somewhere beyond clinical usage of the phrase 'latently infected', which refers specifically and more simply to the presence of a pathogen without indication[1].

Note that the implicit assumptions involved in the standard no-reinfection modelling approach could conceivably give rise to situations where a population (e.g. a cattle herd) with some clinically-latent infection, is judged to be *totally* free of infection. To more accurately model such a case, we might prefer instead to assign a small but non-zero probability to the outcome that one or more 'susceptible' (but in truth latently infected) individuals may eventually progress to clinical disease and infectiousness [2].

In summary, under the standardly used modelling approach there is a trade-off between capturing the true extent of infection, and the need to adequately account for the possibility of *reinfection* in clinically latent individuals who might otherwise

[1]See: https://www.merriam-webster.com/dictionary/latent.

[2]NB. this is intended as a thought experiment on the basis of intuitive reasoning – not a claim to the effect that the methodological approach is sound.

be high unlikely to progress to *symptomatic* infection (i.e. infectiousness, essentially). This fact motivates the consideration of alternative modelling approaches in Chapters 6 and 7.

2.1.3 Epidemiological modelling data

Epidemiological data sets vary greatly in their nature and characteristics. A key characteristic of the epidemiological models considered here is that they tend to rely on *longitudinal* data: where many observations of the same variable(s) are made over time. Here we focus in particular on the nature of observations times and the reliability of the data recorded.

Informative and uninformative timing of observations

The timing of observations can be *informative* (e.g. a patient who initiates contact with medical services), or *uninformative* (e.g. a scheduled appointment). Such information can be related to the SIR and SIR-like models, where informative observation times refer to cases where transition times are observed. For example, contact with medical services can be interpreted as a good proxy for the time of onset of recovery in cases where subsequent transmission can be ignored [8]. On the other hand data associated with uninformative observation times, such as a population-wide diagnostic screening carried out at fixed intervals (see the applied example of [4]) still provide valuable information e.g. on the number of individuals that are infected (actually we observe test positive cases, see below) at a given time. Our treatment of DPOMPs allows for both informative and uninformative observation times though we are mainly focused on the latter. For example, a *partially observed Hawkes process* is used to describe a model with uninformative observation times. Note that in this thesis at least, the term *latent Hawkes processes* is used to distinguish similar models that rely on *informative* observation times instead (see Chapter 7).

The 'observation model'

Other notable features of epidemiological models can derive from the nature of the observation, diagnostic test or other data sources. In the work presented here all such considerations are usually encapsulated within a single mathematical construct, referred to as the *'observation [likelihood] model'* in Bayesian statistics. Depending on the situation, these properties may be expressed as distinctive e.g. probabilistic measures (such as standard error, sensitivity and specificity) conditional on the 'true' system state, as represented within the model. As such, observation models described herein (and elsewhere) are associated with a probability distribution, e.g. the Binomial. In those cases the observation model is essentially defined by the probability density function associated with that distribution (and parametrised by the 'true' model state). This is only a cursory explanation for the characterisation (i.e. modelling) of what is also sometimes referred to as the *observation process* in Bayesian statistics. The pertinent details are given for each model in due course.

2.2 Stochastic processes

In the previous section we introduced the concept of *epidemiological models* and data, with particular emphasis on aspects relevant to later chapters. In this section we introduce the mathematical concept of stochastic processes, that define the 'dynamics' of the model i.e. that describes how to run the model forwards in time to produce outputs. This will complete the definition of the *process* model framework used in this thesis. However, what will remain to state is the Bayesian statistical framework and methods laid out in §2.3 and §2.4 respectively, for eventual application to the BTB data set (as addressed directly in the penultimate two chapters). Of necessity, only a narrow selection of stochastic processes are addressed here; those directly pertinent to the aforementioned work. We begin with a brief introduction to the general concept, before moving on to some more specific examples.

Introduction: stochastic process models

Stochastic, or random, processes[3] can be intuitively understood as probabilistic mathematical models that describe the time-evolution of a random phenomenon, such as a biological or biochemical process [25]. More formally, they are defined as an ensemble of random variables $X = \{X_t : t \in T\}$ on a common probability space, where the set T can be discrete or continuous [26]. In many (perhaps most) practical applications T is said to be one-dimensional and used to represent time.

[3]https://en.wikipedia.org/wiki/Stochastic_process

The scope of this section is limited to introducing the specific process models pertinent to this work, since the topic of stochastic processes is too vast to provide even a brief general overview. In each model presented the index set $T = (t_0, t_N]$ is one-dimensional, continuous and used to represent time. For practical reasons t_N usually corresponds to the final observation time (for a given realisation or set of observations). The initial time, t_0, may be known (e.g. the time of the first observation) or else treated as an unknown parameter for the purposes of inference – this is discussed further in §2.4.

2.2.1 Poisson point processes

The *Poisson [point] process*[4] has different forms and interpretations, but is often interpreted as a *counting process* – a random process that describes the number and distribution of points (or 'events') in a given time interval. The model is so-named because the number of points that occur in the unit-interval, i.e. $(0, 1]$, is a (discrete) Poisson random variable $|X| \sim Pois(\lambda)$ where $|X|$ is used to denote the cardinality of set X. Somewhat equivalently, $|X|$ is the counting process associated with X and its probability mass function is given by:

$$p(|X| = k) = \frac{\lambda^k e^{-\lambda}}{k!} \tag{2.4}$$

The single parameter λ is also known as the 'rate' parameter, owing to the fact that $E(|X|) = \lambda$. In informal terms, we would expect to observe roughly λ events per unit-time interval.

However it is more useful in this instance to consider the continuous random variable that represents the inter-event time Δt, and is exponentially distributed with probability density function:

$$p(\Delta t = \delta t) = \lambda e^{-\lambda \delta t} \tag{2.5}$$

These definitions are based on the standard (or homogeneous) Poisson process in which λ is held to be constant over time. They can however, be extended to situations where $\lambda(t)$ is given as function of time. Such models are typically referred to as *inhomogeneous* (or *nonhomogeneous*) Poisson processes.

[4]https://en.wikipedia.org/wiki/Poisson_point_process

2.2.2 Inhomogeneous Poisson processes

A *inhomogeneous Poisson point process*[5] can be intuitively understood as a Poisson process with rate $\lambda(t)$ such that it varies over time. More formally it is defined as a Poisson point process where parameter $\lambda(t)$ is a location-dependent function and t is a d-dimensional point located in T. In this case the expected number on points in T is given by:

$$E(|X|) = \int_T \lambda(t)dt \tag{2.6}$$

with probability mass function:

$$p(|X| = k) = \frac{E(|X|)^k e^{-E(|X|)}}{k!} \tag{2.7}$$

Inhomogeneous Poisson processes are a useful extension to the homogeneous model, because many real-world phenomena, from natural [e.g. biological] processes to financial markets, exhibit time-varying behaviour. In the case of epidemics within small populations it seems realistic, for example, to intuit that the rate at which new infections occur depends (at least partially) on the disease status of the underlying population, e.g. the present number of infectious individuals.

All of the relevant models presented in this thesis (i.e. those that incorporate nonhomogeneous-Poisson processes) are formulated on this basis, such that λ is a 'step-wise' function of the underlying system state rather than a continuous function of time. That is, rate parameters are held constant between events, with the benefit that the approach used to compute (2.5) for the homogeneous model, remains applicable in this case too[6]. In view of this, it would arguably be more correct to write the rate function as $\lambda(\eta)$ where η denotes the current system state [for any given time t]. However in order to maintain consistency with the descriptions already provided a subscript is used henceforth such that (where applicable) the rate function is written as $\lambda_\eta(t)$. Note also that the *Markov* ('memoryless') property is preserved since the time-evolution of the system depends only on the 'current' system state at any given time t and not on past events.

Another useful natural extension to the standard univariate Poisson process introduced in §2.2.1 are *multivariate* models, which allow for two or more types of

[5]https://en.wikipedia.org/wiki/Poisson_point_process#Inhomogeneous_Poisson_point_process

[6]An important caveat is that 'event' here is taken to mean any change in the underlying system state, whether or not it is an aspect of the model that happens to be governed by Poisson-like dynamics. The point lacks context here, but becomes more relevant in later chapters

possible event. These can be understood as a set of coupled[7] [interacting] processes, with separately defined rate parameters (or functions) for each event type. Such models can be used to represent more complicated systems, such as chemical reactions or epidemics, where they are judged to be comprised of two or more underlying processes – *infection* and *recovery*, for example. In such cases, the probability density associated with the inter-event time is given (for all event types) instead by:

$$p(\Delta t = \delta t) = Re^{-R\delta t} \tag{2.8}$$

$$R = \sum_{\xi \in \Xi} \lambda_\xi \tag{2.9}$$

where each λ_ξ corresponds to a given event type $\xi \in \Xi$.

2.2.3 Hawkes' self-exciting process

Another stochastic process model of interest to work presented in latter sections of the thesis, are 'self-exciting' *Hawkes processes* [27, 28]. These are so-described because of their defining property: each event (temporarily) increases rate at which new events occur. They are therefore another useful tool that can be productively applied to systems which exhibit time-variant or 'clustered' behaviour, such as market events, social media activity, earthquakes and epidemics. A more generalised description which incorporates Hawkes and Cox stochastic process models with 'Poisson noise'[8] has also been proposed for applications within numerical finance and related fields as the *'dynamic contagion process'* [29].

The event rate of a Hawkes process can be written as:

$$\lambda(t) = \mu + \sum_{t_\xi < t} \nu(t - t_\xi) \tag{2.10}$$

with an exponential kernel function chosen for ν, such as:

$$\nu(\tau) = \alpha e^{-\beta\tau} \tag{2.11}$$

where α is a scalar and β parametrises a time-dependent exponential decay that characterises the impact of past events on the event rate at a given time t. This

[7]I.e. they are 'coupled' by their joint dependence on the overall system state.

[8]Also known as: https://en.wikipedia.org/wiki/Shot_noise

model is non-Markovian; the time-evolution of the system at any given time t depends on the entire history of the system up to that time. The probability density associated with a given realisation x is written as:

$$p(X = x) = \prod_{i=1}^{|x|} \lambda(t_i) e^{-\int_T \lambda(t)dt} \tag{2.12}$$

where i indexes the i^{th} event in x. This is a somewhat generic formulation, and the feasibility of directly computing (2.12) can depend greatly on the choice of kernel function ν. It was noted from the literature, that exponential kernel functions seem to be quite often preferred, presumably for their mathematical simplicity and convenience. At least, compared to another (slightly less) popular choice – power law distributions[9]. Here we choose the relatively simple exponential 'parametrisation' (as they are sometimes referred to) given in (2.11) for the work presented in Chapter 7 of this thesis.

Hawkes processes within finite populations

While Hawkes processes provide an interesting alternative to inhomogeneous Poisson processes, for situations where event times are thought to be clustered – it is not immediately obvious that they are applicable to finite (especially small) populations. This is because –unlike the inhomogeneous Poisson process example described above– the event rate is unaffected by the system state, including the total size of the population, or any given sub-population. In simple terms, the total number of infected individuals within an epidemiological model, for example, would not be bound by the total number of [susceptible] individuals available to infect. However Rizoiu *et al.,* [24] propose a method of overcoming this problem, with a generalisation of the Hawkes process for models with finite populations of size N, in which the event rate is defined instead by:

$$\lambda^H(t) = \left(1 - \frac{N_t}{N}\right) \left[\mu + \sum_{t_\xi < t} \nu(t - t_\xi)\right] \tag{2.13}$$

where N_t is the associated counting process, such that when $N_t = N$ the event rate is zero. They also demonstrate application to an epidemiological model by analysing overdispersed, or 'viral', social media data using an SIR model.

[9]https://en.wikipedia.org/wiki/Power_law

2.2.4 Simulation of stochastic processes

Here we discuss the practical endeavour of sampling from the corresponding statistical distributions defined by the stochastic processes described above; in simpler terms, the *simulation of stochastic processes*. Algorithms that permit such simulations can provide both a convenient way to conduct exploratory analysis, and numerically 'solve' (i.e. integrate) models. They are also a powerful tool for statistical inference[10] and as a source of simulated observational data. For example, the latter is one means by which methods and algorithms were validated throughout the course of this work. As with previous sections, the methods described are limited in scope to the two classes of stochastic process that are directly pertinent to the work presented in later chapters.

Simulation of inhomogeneous Poisson processes

Step-wise *inhomogeneous Poisson processes* of the kind described above can be easily simulated using the using the Doob-Gillespie direct method algorithm which was developed for exact simulation of coupled processes to model chemical reactions [30].

Let X denote a process of the form given in §2.2.2 distributed according to $f(X = x)$ (2.7) and assume that we wish to sample f (i.e. simulate) to obtain a single realisation of the model denoted by x. For the sake of simplicity, it is assumed (for now) that any model parameters (e.g. $\theta = \{\beta, \gamma\}$ for the SIR example illustrated in Figure 2.2) are known constants. Finally let η_{ic} represent the state of the system at $t = 0$ (also referred to as the 'initial condition'). Algorithm 1 can then be applied to obtain a single realisation of the model denoted by x up to any chosen t_{max} which is distributed according to the desired probability density.
As noted above our interpretation of sampling from the initial state density μ depends somewhat on circumstance. One approach in epidemiological modelling, which has already been briefly discussed, is to assume that epidemics are seeded by a single infectious individual at some unknown time, represented by a random variable – i.e. an element of parameter vector θ. Different approaches may be called for when disease spread is believed to be endemic, or else with other population modelling problems that typically arise within ecology. Since the examples used

[10]Since an intuitive and well known guiding principal of random sampling generally is that where we cannot sample directly, the optimal choice of proposal distribution is as similar to the target distribution as practicalities will allow.

Algorithm 1 Doob-Gillespie direct method algorithm (DGA)

```
Require: $\theta$, $\eta_{ic}, t_{max}$
  Set $\eta \leftarrow \eta_{ic}$, $t = 0$
  loop
    Evaluate (2.8), i.e. compute rate vector $r = \lambda_\eta(t)$.
    Sample (2.5), i.e. determine the next event time: $t_\xi = t - \frac{ln(u)}{\sum r}$ where $u \sim U(0,1]$.
    if $t_\xi < t_{max}$ then
      Choose event type $\xi$ with $pr(\xi) = \frac{r_\xi}{\sum r}$
      Append the event to the trajectory: $x \leftarrow x_\xi$ and set $t \leftarrow t_\xi$
    else
      return $x$
    end if
  end loop
```

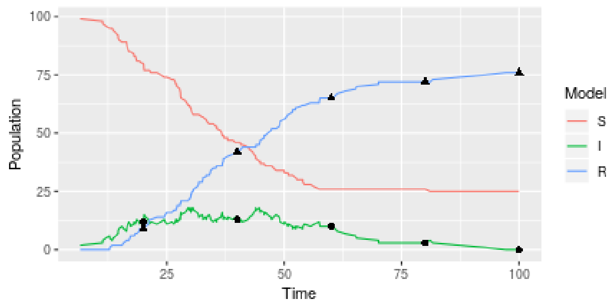
here are all epidemiological we treat μ in the manner described above.

Simulation of Hawkes processes

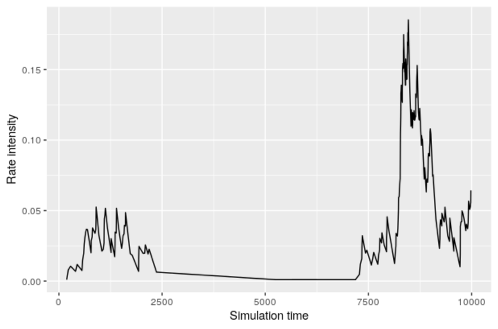
There are also methods available for exact simulation of Hawkes processes [31, 32]. For the purpose of exposition, and to aid in the construction of a general framework, we can consider a relatively straightforward algorithmic approach referred to as 'thinning' [33]. As with the previous example the parametrisation of the model (which may vary according to the precise implementation) is denoted simply by the vector θ but for simplicity it is assumed that the model is univariate such that x is a vector of arrival times (events). Any event history prior to $t = 0$ is disregarded. The broad approach described for Algorithm 2 can also be regarded as a more general framework that incorporates both algorithms, since it is precisely equivalent to the DGA in the special case of a multivariate model, in which rate parity ensures that the acceptance probability will always evaluate to $p_a = 1$. Sample realisations of both types of process are illustrated in Figure 2.4.

Algorithm 2 Thinning algorithm for Hawkes processes

```
Require: $\theta$, $t_{max}$
  Set $t = 0$
  loop
    Evaluate (2.10) and set upper bound $\lambda^* = \lambda(t)$
    Sample the (provisional) next event time: $t_\xi = t - \frac{ln(u)}{\lambda^*}$ where $u \sim U(0,1]$.
    if $t_\xi < t_{max}$ then
      Set $t \leftarrow t_\xi$
      Compute the acceptance probability: $p_a = \frac{\lambda(t)}{\lambda^*}$
      Append the event time to $x \leftarrow t_\xi$ with probability $p_a$ (else reject)
    else
      return $x$
    end if
  end loop
```



(a) DGA simulation (SIR model).



(b) Univariate Hawkes process.

Figure 2.4: Sample realisations produced using Algorithms 1 and 2. The parameter sets used were $\theta = \{0.1, 0.002\}$ and $\theta = \{0.1, 0.002\}$ respectively.

2.3 Bayesian inference

Hitherto in this chapter, we have laid out fundamental methodological concepts that inform the construction of models throughout the thesis. The rest of it is devoted to the methods that allow for their utilisation in the analysis of data. First we describe the broad statistical framework that allows us to use models to learn from data; a task more formally referred to as *inference*. Among other things, this provides necessary context for the inference methods described subsequently, in §2.4

and elsewhere throughout the remainder of the thesis.

Frameworks for statistical inference are typically organised according to the overall mathematical approach taken. For example:

1. Maximum Likelihood Estimation (MLE)
2. Bayesian inference
3. Approximate Bayesian Computation (i.e. distance-based measures)

In general, the objective of all forms of statistical inference –which includes *frequentist* approaches to hypothesis testing, familiar to most because of their widespread usage in practical scientific applications– is to infer the properties of an underlying population, or *probability distribution*. The section begins with a general introduction, which contextualises the *Bayesian* approach described here by comparing it with a well-known application of *frequentist* statistical inference. Section §2.3.2 goes on to develop the framework, as it applies to the concepts already introduced in previous sections of this chapter.

2.3.1 Introduction

Bayesian methods are so-called because the distribution of interest is derived according to Bayes' theorem:

$$P(H|y) = \frac{P(y|H)P(H)}{P(y)} \propto P(y|H)P(H) \tag{2.14}$$

The equation applies the rules of conditional probability to describe the probability (or 'likelihood') of an event[11]. (We note that the familiar canonical definition of Bayes' theorem often given in terms of $\{A, B\}$ instead of $\{H, y\}$.) This highlights an important feature of Bayesian thinking: probabilities are used to describe the degree of belief in a given outcome.

The quantity $P(y|H)$ is referred to as the *likelihood function*, because it describes how likely H is given y. It often has an implicit dependence on some underlying model. Note that the *marginal likelihood*, denoted $P(y)$, is the same for all H. This leads to the proportional quantity on the RHS of the equation.

[11]See https://en.wikipedia.org/wiki/Bayes'_theorem for a more complete definition.

An example: scientific hypothesis testing

Consider an illustrative example, in which $\{H, y\}$ is used to represent a hypothesis, and (new) evidence, or 'data' in conversational terms. In this context, (2.14) can be parsed as describing the degree of conviction in our beliefs about H –expressed as a probability– *after* accounting for y. This is referred to in Bayesian statistics as the *posterior probability distribution*.

Similarly, $P(H)$ is used to denote our beliefs *before* accounting for the new evidence, y. More formally, $P(H)$ is the *prior probability distribution*.

H could also be used to represent [at least] two hypotheses. In standardly used [frequentist] approaches to hypothesis testing then, we would usually label these $\{H_0, H_1\}$, and only choose to reject H_0 in favour of H_1 when the *p-value* exceeds some given confidence level, say 0.95.

In contrast, the conventional approach within Bayesian statistics is to compute the ratio of posterior probabilities $P(H_1|y)/P(H_0|y)$, referred to as the *Bayes factor*. Interpretation of Bayes' factors is discussed in due course but it is essentially the same as for common frequentist approaches. We 'reject' H_0 in favour of H_1 (or at least, say that the evidence in favour of H_1 is 'strong') when the Bayes factor exceeds a predetermined level, say 10 (according to the Kass-Raftery scale of interpretation [1]).

Beyond standard hypothesis testing

However, H could equally well represent competing hypotheses arbitrarily labelled $\{H_1, ..., H_n\}$ and we could disregard the acceptance/rejection step altogether. For example, if we are interested in $P(H|y)$ merely as a mathematical expression of our beliefs (as opposed to a binary decision-making tool). Practical examples of y might include the outcomes of an experiment, or a longitudinal data set of observations $y = \{y_1, ..., y_n\}$ from a non-experimental setting. It is the latter that is of primary relevance here and throughout the thesis.

A short note on notation The symbol $P(...)$ has been used here to represent *probability mass functions*, since we are notionally dealing with discrete set of values (i.e. hypotheses). However H could instead represent our beliefs about the distribution of a continuous [random] variable, such as the average height of children in a school. In that case, the posterior probability is a continuous distribution and it is conventional to denote a *probability density function* as $\pi(...)$ instead. In practice though, consistency is preferable to strict adherence, in the case of models which incorporate *both* continuous and discrete probability distributions. Almost without exception, the main target (i.e. posterior) probability distributions of interest considered through out this thesis, are functions of a vector of random variables (parameters) on a finite-dimensional subset of Euclidean space. Accordingly, the symbol π is used henceforth to denote both density and mass functions alike.

2.3.2 Bayesian framework

Bayesian methods are useful for working with large or complex scientific data sets, because the laws of conditional probability allow models to be extended to accommodate almost arbitrarily complicated features, in order to represent a given dynamic [e.g. natural] system. In general such models are sometimes known by the evocative term 'scientific domain models', particularly in the practical context of software engineering[12]. This highlights an important benefit of the general framework for Bayesian inference in practice: it is amenable to bespoke model definitions and other situations, for which standard, e.g. frequentist, methods may be less well suited. This section therefore builds on the framework loosely described above, introducing the concept of parameter inference. First, an illustrative example of a relevant model is introduced using an abstract notion of the [stochastic process] model laid out in §2.2.2.

A generic stochastic process model for Bayesian inference

Bayesian models that incorporate *stochastic processes* like those briefly described in §2.2 are fairly common across a range of scientific applications, where completely observing the process of interest is difficult or impossible. In such cases the dynamics of the system can be thought of as comprising a hidden (or 'latent'[13]) part (or process) and an observed one. This class of model are often known as *hidden Markov*

[12]For example, see Chris Rackaukas' talk on Domain Models with Integrated Machine Learning at the 2019 JuliaCon in Baltimore. Also, Michael Betancourt's lecture series 'Efficient Bayesian inference with Hamiltonian Monte Carlo', delivered at the Machine Learning Summer School in Iceland in 2014 – both available on YouTube.

[13]Note (once again but in reverse) that this use of the term is distinct from the way it is used in the context of epidemiological modelling to refer to a state or phase of disease which asymptomatic or otherwise undetectable.

models (HMM) and *partially observed Markov processes*, for discrete and continuous [e.g. -time] models respectively. In either case, latent and observation processes are treated as random variables, distributed according to probability distributions. For example:

$$X|(\Theta = \theta) \sim f_\theta(x) \tag{2.15}$$

$$Y|(X = x, \Theta = \theta) \sim g_\theta(y|x) \tag{2.16}$$

where x represents the latent process; y the observed process (or data) and '$\sim$' means 'distributed according to'. The latent process is sometimes known as the *signal process*, depending on context[14]. The probability densities associated with each are labelled f_θ and g_θ. When y represents a dataset of observations it is convenient to refer to the latter simply as the *observation model*, since it describes the probability distribution of those observations y [conditional on x]. Finally, the subscripts denote the fact that either or both may be parametrised. For example f is parametrised by $\theta = \{\beta, \gamma\}$ for the SIR example given in 2.2.

Disregarding the latter for now (i.e. assume that θ is known) it is self-evident that:

$$\pi(x|y) = \frac{\pi(x)\pi(y|x)}{\pi(y)} = \frac{f_\theta(x)g_\theta(y|x)}{\pi(y)} \tag{2.17}$$

Parameter inference

In order to perform statistical inference where θ is *unknown*, however, we must express knowledge about θ probabilistically. In other words we must derive an alternative expression for (2.17) that does incorporate θ. To do so recall that the variable x represents the time-evolution of a stochastic process (described in §2.2) chosen so as to mathematically accommodate our *possible*[15] beliefs about an underlying [e.g. biological] process. As above, y can be used to denote a set of *longitudinal*[16] measurements, at intervals indexed by $1, ..., n$. Let θ denote a vector of (unknown) random variables which govern the behaviour of the system, as described above.

[14]See https://en.wikipedia.org/wiki/Signal_processing#Statistical.

[15]Though tautological, it is still noteworthy that the model can do nothing to inform our beliefs concerning situations or scenarios outwith the constraints (or assumptions) of the model, since we have by-definition deemed them *impossible*. This is an inescapable limitation; it must instead be accounted for in any interpretation of the results from an analysis.

[16]Taken over time.

The goal is to discover the *joint [posterior] distribution* of θ and x, based on measurements from a real-life system. That can be written as:

$$\pi(\theta, x|y) = \frac{\pi(\theta)\pi(x|\theta)\pi(y_{1:n}|x,\theta)}{\pi(y_{1:n})} \tag{2.18}$$

where $\pi(\theta)$ is the prior probability density; the *marginal likelihood* $\pi(y_{1:n})$ serves as a normalising constant – leading to the RHS of equation (2.14); and the remaining terms are defined by:

$$\pi(x|\theta) = f_\theta(x) = \prod_{i=1}^{n} f_\theta(x_i|x_{i-1}) \tag{2.19}$$

$$\pi(y_{1:n}|x,\theta) = g_\theta(y_{1:n}|x) = \prod_{i=1}^{n} g_\theta(y_i|x) \tag{2.20}$$

where the equivalence implied in (2.20) is derived by the chain rule on the assumption that observations are statistically independent given x [10].

Note that in some situations for practical reasons it make senses to partition x on intervals defined by the time of each observation, with the first segment written as x_1, the second as x_2 and so on.

Model notation and interpretation

The precise definitions of f and g depend on the underlying model. For example in the case of the stochastic process described in §2.2.2 the precise definition of (2.19) is given by (2.7). Their interpretation in the text, meanwhile, is somewhat dependent on context. The former can be notionally regarded in the algorithms described as either the likelihood function, or else as a simulation protocol. In other words, per context it can be understood as either a method for computing the likelihood of a given x, or else drawing a random sample ('simulation') from that same probability distribution. For now the observation model, g, is left undefined but its computation is typically trivial, and dominated by the challenge of sampling efficiently from f, in the inference schemes described in the next section. Similar considerations apply to the observation model though g is a function that computes the likelihood of a given y, conditional on an also given x. Again, depending on context it could also be a distribution we wish to sample from, conditional on the same. In casual terms, we may wish to simulate a set of observations data (e.g. for testing and verification), given a known value of x.

2.4 Sampling methods

As we have seen, Bayesian statistics provides a broad framework for inference – the objective of which is to infer the properties of (equivalently) an underlying population; probability distribution; or random variable. This has been partially illustrated using the general concept of stochastic processes, as described above. The purpose of this section is to show how that framework can be implemented in practice, and the focus is on computational methods that allow samples to be drawn from the posterior. In particular, we consider three approaches to this problem:

1. Data-augmented Markov chain Monte Carlo (**DA-MCMC**)

2. Sequential Monte Carlo (**SMC**)

3. Quasi Monte Carlo (**QMC**)

where 'Monte Carlo' is a commonly used term for methods that rely on random sampling. In most practical settings actual random numbers are not available, thus we *simulate* them using an algorithm that generates pseudorandom numbers. The section begins with a basic introduction to MCMC, and in particular the Metropolis-Hastings algorithm, aspects of which are also relevant to each of the methods applied in later chapters. It goes on to describe the selected methods, organised according to the three broad classes noted above.

The first two classes of method for [Bayesian] inference described, are *data augmented* (DA) MCMC and *sequential Monte Carlo* (SMC). Standard implementations of these algorithms are conceptually quite distinct. For example methods that rely on SMC combine a sequential sampling step with a resampling step over a population of *'particles'* [34]. Each particle represents distinct system trajectory – an entire realisation (i.e. 'simulation') of the model. For this reason they are sometimes placed within one of four broad classes of method applicable to stochastic epidemiological models referred to as being *simulation* based [35]. In this taxonomy, the other three are given as DA; *approximation* based methods; and *Martingale* methods [36, 8, 37, 38, 39]. Approximation based and Martingale approaches are not directly addressed in this thesis. The three types of method that are addressed fall under either simulation-based (SMC and QMC) or data-augmentation (DA-MCMC). Only an overview of these three classes is provided, since with rare exception only these methods are employed throughout.

Algorithm	Class	Kernel	Type	Section
Metropolis-Hastings	MCMC	n/a	Rejection sampler	§2.4.1
Data-augmented MCMC	DA-MCMC	Generic	Rejection sampler	§2.4.2
Model based proposal (MBP)	Seq. IS	n/a	Importance sampler	§2.4.2
MBP-MCMC	DA-MCMC	MBP	Rejection sampler	§2.4.2
Particle filter	SMC	n/a	Importance sampler	§2.4.3
Particle MCMC	MCMC	SMC	Rejection sampler	§2.4.3
SMC^2	IBIS	SMC	Importance sampler	§2.4.3
Quasi Monte Carlo	QMC	n/a	Importance sampler	§2.4.4

Table 2.1: Overview of the inference methods addressed in this section. The 'kernel' in MCMC rejection sampling schemes is the method used to generate new proposals based on the current sample. It plays a similar role in *iterative batch importance samplers*; a collection of particles is perturbed by using the kernel to propose new ones during the mutation step. These features are described in more detail in due course. For now, note that many of the algorithms and methodological concepts are interrelated; this is better illustrated in Figure 2.5.

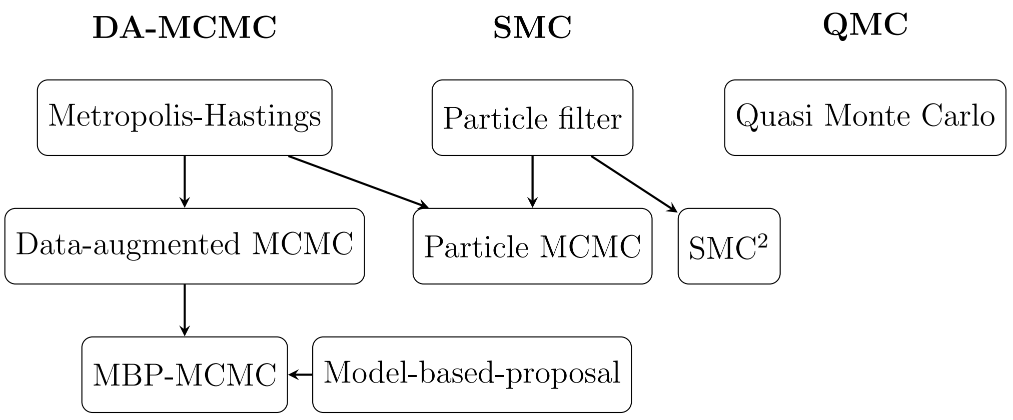


Figure 2.5: Algorithm dependency map. The diagram shows how [subsets of] the methods and algorithms introduced throughout this section interrelate. An arrow from a to b indicates that a provides a modular component of, or methodological basis for, algorithm b. The same broad principle is extended in Chapter 4 to provide the basis for two new algorithms.

The second part of the section covers useful sundry methods for performance and diagnostics that are directly relatable to MCMC but that also have a direct bearing on the algorithm introduced in §3.3. The section and chapter conclude with an overview of methods for multi-model inference (i.e. inference with multiple candidate models) and a brief summary, in §2.5 and §2.6 respectively.

2.4.1 Introduction to MCMC

As with standard Monte Carlo methods, *Markov chain Monte Carlo*[17] is a class of algorithms designed for drawing samples from a probability distribution. Unlike standard methods, that rely on the statistical independence of samples, MCMC algorithms are designed to produce a sequence of *correlated* samples which nonetheless has the target (i.e. posterior) distribution as its *equilibrium distribution.*

Similar to use of the term in §2.1, *Markov chains* are so-called, because the i^{th} sample x_i is contingent only on the previous sample x_{i-1}. As an interesting aside, MCMC algorithms can also be placed within the class of stochastic processes referred to as 'random-walks'[18]. More meaningfully, MCMC also belongs to a class of random sampling algorithm known as 'rejection samplers', because not all [proposed] samples are accepted. In fact, it is essentially this step that ensures the Markov chain will reliably converge on the target distribution, though the efficiency with which it does so may vary according to circumstance, as discussed in due course.

A proper treatment of *Markov chains* would, among other things, enumerate and explain important mathematical properties, such as *irreducibility*; *aperiodicity*; and *detailed balance* which are required to ensure that the samples generated are distributed (asymptotically) according to the target density. By contrast the remainder of this section focuses on their practical application here, mostly by exposition of a simple example. We then move on to a more advanced class of MCMC algorithm which is more directly applicable, in §2.4.2

[17]See https://en.wikipedia.org/wiki/Markov_chain_Monte_Carlo

[18]This is interesting only in that it demonstrates the broad nature and ubiquity of the terminology and concepts invoked, such as 'stochastic process'. It is not particularly useful in this case to regard MCMC as a stochastic process, at least in the narrow sense of the Bayesian framework established in §2.3.2.

Example: the Metropolis-Hastings algorithm

The canonical example of MCMC, is the *Metropolis-Hastings* algorithm [40]. A basic implementation is illustrated in Algorithm 3. An important practical feature of any Metropolis-Hastings implementation, is the choice of *transition kernel* (or density) that generates proposed samples based on the current sample. For the time being though, this concept is treated in abstract terms, and denoted simply as $X_f|X_i = x_i \sim q(x_f|x_i)$, where the i^{th}) sample x_i is drawn by making 'proposals'. X_f are conditionally distributed according to an arbitrary chosen 'proposal' probability density, denoted $q(.)$.

The proposed samples are accepted (or rejected) with Metropolis-Hastings probability:

$$p_{mh} = \min\left\{1, \frac{\pi(x_f)}{\pi(x_i)}\frac{q(x_i|x_f)}{q(x_f|x_i)}\right\} \tag{2.21}$$

where the target density is given simply as $\pi(x)$, and the subscripts are interpretable as above – i for the i^{th} sample and f for proposed samples. In cases where proposals are not accepted, the 'current' sample value is drawn instead, such that $x_{i+1} = x_i$. Note that in practice, a judicious choice of $q(.)$ is symmetrical (e.g. Gaussian) such that those two terms cancel. Pseudo-code for a basic Metropolis-Hastings algorithm is given in Algorithm 3.

Algorithm 3 Metropolis-Hastings algorithm

```
Require: $N$, $q$
  Choose (arbitrarily) initial sample $x_1$
  for $i := 1$ to $N - 1$ do
    Sample $x_f \sim q(x_f|x_i)$, $u \sim U(0,1]$
    Compute $p_{mh}$, i.e. evaluate (2.21)
    if $u < p_{mh}$ then
      Set $x_{i+1} \leftarrow x_f$ //accept sample
    else
      Set $x_{i+1} \leftarrow x_i$ //reject
    end if
  end for
```

Note: The Metropolis-Hastings acceptance step can be productively applied in other algorithms, outwith traditional MCMC methods. In some contexts, namely that of a *transition kernel* it is more natural to encapsulate and refer to a given procedure that incorporates 'MCMC moves' (as they are sometimes referred to in literature[19]) as a [type of] *Markov kernel*. For example, the method presented in §3.2.

The 'burn-in' period

As per the pseudo-code given above, Markov chains are initialised at an 'arbitrarily' (in practice, quite carefully[20]) chosen location. An important step not disclosed in that description is the standard practice of discarding an initial number of samples, so as to allow the Markov chain to converge on the stationary distribution. This is commonly referred to as the 'burn-in' period (or 'adaptation period' in finite-adaptive MCMC schemes).

2.4.2 Data augmented MCMC

As noted in the introduction to this section, there is an appreciable difference in the conceptual approach of data-augmented (DA) methods for parameter inference, and 'particle' (SMC) methods, such as the one described in §2.4.3. The latter are so-called because each particle represents a distinct system trajectory –where such [trajectory-denoting] variables are labelled throughout as x. Essentially, the purpose of the algorithm is to integrate over *all* x for a given theta tuple[21], which can be written as $\pi(x|y, \theta)$.

By contrast, DA methods can be understood by interpreting x as a [random variable which represents a] *singular* system trajectory, where the goal is to draw samples from the *augmented*, joint [posterior] distribution of θ and x, denoted $\pi(x, \theta|y)$. Accordingly, the acceptance probability equivalent to (2.21) is expressed here instead as:

$$p_{mh} = \min\left\{1, \frac{q(\theta_i, x_i|\theta_f, x_f)\pi(\theta_f, x_f|y)}{q(\theta_f, x_f|\theta_i, x_i)\pi(\theta_i, x_i|y)}\right\} \tag{2.22}$$

The purpose of this is to allow the construction of a Markov chain with the joint distribution of θ and x (i.e. the posterior) as its equilibrium distribution, in essentially the same way as Algorithm 3 but based on the model defined by (2.18).

[19]A search for the phrase "MCMC moves" (inclusive of quotes) yielded 797 results on Google scholar – conducted in 19th July 2020.

[20]This apparent contradiction is resolved in due course.

[21]More accurately, it provides an unbiased estimator: $\hat{\pi}(x|y, \theta)$.

In a sense, this conceptual approach allows for the formulation of schemes based more directly on traditional methods such as the basic Metropolis-Hastings algorithm. In this way, parameter inference can be performed without the need for additional algorithmic 'components', such as particle filters. However, as we will see in due course this advantage must be set against the challenge of generating proposed samples in the typically high-dimensional measure-space[22] on which [values of] x are defined. By illustrative comparison – this is precisely the challenge that particle filters serve to overcome, albeit at the computational cost of independently generating ('simulating') many distinct values of x.

As noted above, DA schemes employ an additional (i.e. *augmented*) variable to represent one or more realisations of a [state] process. For consistency with the notation already established these augmented variables are labelled here as x. As per the discussion of Algorithm 5 that follows, recent innovations blur the distinction between SMC and DA methods a little, by employing sequential 'partial simulation' methods for making proposals within data augmented schemes [4, 35]. Another scheme that combines features from existing SMC and DA methods, which we refer to as *MBP-IBIS*, is presented in Chapter 3.

Proposals in data-augmented MCMC

For the purpose of exposition, it is simplest to consider $q(\theta)$ and $q(x)$ on an individual basis for the time being: choosing a proposal distribution for the model parameters θ is relatively trivial because it is defined on a continuous number space $\mathbb{R}^{|\theta|}$. In practice a simple multivariate Gaussian distribution is a convenient choice for the latter. This can be improved by adaptively re-parametrising $q(\theta)$ based on the covariance of samples as they are obtained (so as to reflect correlation between parameters in the target density).

A more pressing challenge for the construction of efficient DA-MCMC schemes lay in choosing an adequate proposal density $q(x)$ for x, which is defined on its own measure space – denoted where applicable as Ω. In general though, a *poor* choice for $q(.)$ is likely to result in inadequate 'mixing'. That is, the Markov chain may take an infeasibly long time to converge on its equilibrium distribution. Alternatively

[22]See https://en.wikipedia.org/wiki/Measure_space for a brief but formal definition of 'measure space'. Loosely speaking, if x represents a 'trajectory' –a sequence of events in continuous time– then the corresponding measure space is a set, which contains all possible trajectories.

(and equivalently) q must be chosen such that Ω is adequately 'explored' in order to obtain a reliable [posterior] estimate. Some recent innovations –that incorporate knowledge of model dynamics in a way that could be loosely described as 'partial simulation'– have been proposed to accomplish this. One such method, applicable to the stochastic process model given in §2.2.2, is described in Algorithm 5.

Generic example

The simplest conceptual implementation of data-augmented MCMC can be considered as one in which proposals are made to *either* θ or x, conditional on the other (i.e. Gibbs sampling [21]). This is not the approach that is generally relied on in this thesis, but it nonetheless (notionally) allows for correlated (i.e. dependent) samples to be drawn from the *joint distribution* of the model parameters and state process in the form of a Markov chain: $\{\theta_i, X^i\} \sim \pi(\theta, x|y)$. It therefore serves to illustrate the basic concept, and is applicable to stochastic processes in general. Recalling the framework laid out in (2.18) we have:

$$\pi(\theta, x|y) \propto \pi(\theta)\pi(x|\theta)\pi(y|x, \theta) = \pi(\theta)f_\theta(x)g_\theta(y|x) \tag{2.23}$$

We shall also allow that the prior density $\pi(\theta)$ is straightforward to compute. In conjunction with (2.22) we have:

$$p_{mh} = \min\left\{1, \frac{q(.)\pi(\theta_f)f_\theta(x_f)g_\theta(y|x_f)}{q(.)\pi(\theta_i)f_\theta(x_i)g_\theta(y|x_i)}\right\} \tag{2.24}$$

where the proposal densities are shortened for brevity and the parametrisation of f and g (i.e. the model parameters) when applicable may be inferred by the parameter passed to the function, e.g. $f_\theta(x_f)$ implies θ_f. A generic pseudo-code description based on the Metropolis-Hastings algorithm is provided in Algorithm 4.

DA-MCMC is reasonably straightforward to implement in practice, so long as new values of x are proposed in a 'naive' fashion; that is, without overdue consideration of model dynamics or other other important factors relating to the target distribution. However while this can be effective in some cases, this naive approach does not scale well to other types of model. For example [4] demonstrate how the standard data augmented MCMC method yields highly correlated samples even in a relatively simple model that has auto regulatory dynamics. Such models lack a natural upper bound on the number of events that may occur in a single realisation. This manner of making proposals may be problematic if it leads to poor 'mixing'

Algorithm 4 Generic DA-MCMC

```
Require: $N$, $q$
  Choose initial sample: $\{\theta_1, x_1\}$
  for $i := 1$ to $N - 1$ do
    Sample $\{\theta_f, x_f\} \sim q(\theta, x|\theta_i, x_i)$, $u \sim U(0, 1]$
    Evaluate (2.24) to obtain $p_{mh}$
    if $u < p_{mh}$ then
      Set $\{x_{i+1}, \theta_{i+1}\} \leftarrow \{x_f, \theta_f\}$ //accept sample
    else
      Set $\{x_{i+1}, \theta_{i+1}\} \leftarrow \{x_i, \theta_i\}$ //reject
    end if
  end for
```

in the algorithm with respect to x. This in turn naturally leads to correspondingly high auto correlation in the θ samples obtained. In practical terms, the Markov chain may take a long time to converge.

'Partial simulation' methods

The issues described above are motivation for recent improvements to DA methods, which have also narrowed the distinction with simulation based methods –as briefly described in the introduction to this section– by incorporating a sequential sampling (or 'partial simulation') step.

For example [4] describe a method for model based proposals (MBP) – sequentially sampling from a proposal distribution constructed based on the discrepancy between the current and proposed values of θ, that incorporates model dynamics. Notionally this amounts to 'partially' simulating new events and probabilistically deleting existing ones, and is applicable specifically to the stochastic process model described in §2.2.1. This method is illustrated with pseudo-code in Algorithm 5, where f_Δ is shorthand for the DGA algorithm parametrised by the Poisson rate vector $\theta = r_\Delta$.

When combined with an appropriate parameter proposal density (as already discussed) the MBP algorithm provides a complete (and highly efficient) choice of $q(.)$ in the form required by Algorithm 4. The advantage of the approach can be understood by considering that, for a reasonably well-designed model, θ and

Algorithm 5 Model based proposals (Pooley *et al.,*)

```
Require: $f$, $\theta_f$, $\theta_i$, $x_i$, $\eta_{ic}$
  Set $\eta_f$, $\eta_i \leftarrow \eta_{ic}$, $\alpha \leftarrow 1$
  for $\xi := 1$ to $|x_i|$ do
    // 'partial simulation' step:
    Evaluate (2.8) for both systems to obtain rate vectors $r_f$, $r_i$
    Compute the partial rate vector $r_\Delta = \max\{0, r_f - r_i\}$
    Sample $x_\Delta \sim f_\Delta(x)$ using Algorithm 1 where $t_{max}$ := the $\xi^{th}$ event time
    Append $x_f \leftarrow x_\Delta$ and update $\eta_f$
    // keep (or delete) the $\xi^{th}$ event in $x_i$:
    If necessary, update $r_f$ using (2.8)
    Append $x_f \leftarrow x_{i,\xi}$ with $p_{keep} = \max\left\{1, \frac{r_f}{r_i}\right\}$
    Update $\eta_f$, $\eta_i$ accordingly
  end for
```

x are likely to be strongly correlated – else θ would effectively have no [related] meaning. It is therefore helpful if *appropriate* x_f can be paired with θ_f, when making proposals. This is achieved by accounting for model dynamics, encoded by $f(x)$. In addition, the MBP algorithm is constructed in a way that gives rise to a computational convenience, by way of the following equality:

$$q(x_f|\theta_f, \theta_i, x_i)\pi(x_f|\theta_f) = q(x_i|\theta_i, \theta_f, x_f)\pi(x_i|\theta_i) \tag{2.25}$$

where $q(x|.)$ is the probability density associated with making a *model based proposal.* This leads to cancellations in (2.22). When θ proposals are distributed (i.e. generated) such that $q(\theta_f|\theta_i) = q(\theta_i|\theta_f)$, we can rewrite that equation as:

$$p_{mh} = \max\left\{1, \frac{\pi(\theta_f)\pi(y|\theta_f, x_f)}{\pi(\theta_i)\pi(y|\theta_i, x_i)}\right\} \tag{2.26}$$

and proceed to implement Algorithm 4. An applied example is the method used for validation and comparison in Section §3.4. The [Julia] software package used to produce that and others is presented in Chapter 5.

Alternatively, [35] describe a method which also employs a sequential sampling step and could likewise be interpreted as 'partial simulation'. In that case the authors simulate a single 'subject pathway' (i.e. the trajectory of one individual within the model) conditional on θ and all other subject pathways. While this is notionally an individual-based model, it is mathematically equivalent to the formulation usually preferred when only (partial) population count level data are available, as in [4] and other examples presented in this thesis.

In summary, both examples [4, 35] illustrate how such methods can be employed as functional components within other inferential frameworks, such as the one laid out in Algorithm 4. This is similar, in spirit, to the employment of particle filters within algorithms such as *particle-MCMC* and *SMC*^{2}, as described in the next section.

2.4.3 Sequential Monte Carlo

Sequential-Monte Carlo methods (SMC) are another important class of inference algorithms. Their defining feature is that they combine a *sequential sampling* step with a *resampling* step, in this case over a population of *'particles'* [34], where 'sequential sampling' is often referred to more simply as 'simulating'. In this description, each particle represents distinct system trajectory. In relation to the process models described in §2.2, this involves partitioning x in to $n = |y|$ segments[23] (where $|y|$ is the number of observations). The segments are indexed by $j = 1 : n$, and each segment is sampled *sequentially* from the conditional distribution $x_j \sim \pi(x_j|\theta, x_{1:j-1})$. This is largely a matter of semantics and equivalent to simply sampling from the probability density otherwise denoted $f_\theta(x)$. Thus, in keeping with that same notation – expression (2.19) can be reorganised as:

$$\pi(x_{1:n}|\theta) = f_\theta(x_{1:n}) = \prod_{j=1}^{n} f_\theta(x_j|x_{1:j-1}) \tag{2.27}$$

where informally we allow that sampling $f(x_j|x_{1:j-1})$ for $j = 1$ is equivalent to simulating x in the manner already described, i.e. we simply simulate the first segment given some initial condition, known or otherwise.

One benefit of framing the problem in this way, is that it aligns neatly with treatment of the discrete-time variants of similar models, usually known as *hidden Markov models*. This highlights that much of the general framework and methods laid out for SMC as applied to HMM by [10] are also applicable to Markov processes.

[23] Or $n = |y| - 1$ segments, if the first observation time is also taken to be the initial time t_0 of the system trajectory.

‘Online’ vs ‘offline’ algorithms

A useful feature of many SMC schemes is that they can be implemented as ‘online’ algorithms – see https://en.wikipedia.org/wiki/Online_algorithm.

In practical terms: it is not necessary to store the entire history of each particle in memory. DA schemes by contrast are necessarily constructed ‘offline’; the augmented variable x must always be stored.

Thus, despite the additional computational cost of simulating many different particle trajectories –as opposed to partially simulating just one– SMC can provide a highly efficient means of integrating over ‘difficult’ high-dimensional measure spaces Ω, such as those on which $f_\theta(x)$ is often defined for stochastic process models.

The resampling step in SMC accentuates the natural efficiency of online algorithms, by focusing computation on values of x that contribute disproportionately to the posterior mass.

As in the previous section our primary purpose is to estimate the posterior distribution of a given model of the form described but we first consider SMC methods in the context of the *filtering* problem as an intermediate step towards that goal. That is, we wish to sample sequentially from $\pi(x_{1:n}|y_{1:n}, \theta)$ and estimate $\pi(y|\theta)$. Accordingly, the remainder of the subsection is used to address the problem of parameter inference by covering a narrow range of SMC methods, selected for their relevance to the algorithms and results presented in Chapters 4 and 6.

Resampling methods

Resampling is a topic unto itself, only discussed here briefly. A number of different approaches are possible and can be combined in modular fashion with particle filters and other SMC procedures that depend on them. These include *multinomial; systematic; stratified;* and others [41, 42]. In the results presented here we used the systematic approach of [43] in accordance with the advice of [10].

Particle filtering

SMC methods which address the *filtering* problem are usually referred to as *particle filters*. That is, we wish to sequentially sample N trajectories (particles) denoted $X_n^i \sim f_\theta(x_n|x_{n-1})$ – which we could do by using Algorithm 1, for example. Sampling $f_\theta(x)$ directly ensures that there is no need to evaluate (2.27) (which would only be necessary if we employed a separate proposal density). In a basic particle filter, this is combined with a weighting and *resampling* step, which serves to 'focus' computation on the particles that contribute most to our knowledge of the target distribution.

Pseudo-code is given in Algorithm 6, but here we describe each step of the algorithm, only disregarding the initialisation step for sake of parsimony:

1. We begin by sampling $X_i^j \sim f_\theta(x_i|x_{i-1})$.
2. The particles are then (incrementally) weighted according to $g_\theta(y_i|x_i^j)$ and the average of this quantity recorded.
3. Finally the particles are resampled to obtain a set of equally weighted particles, and the process is repeated until time n.

In this case, our primary interest in the method is that it provides the unbiased estimate (2.29) which can then be used to obtain (by the chain rule) an estimate of $\pi(y|\theta) \propto \pi(\theta|y)$ as per (2.28) and (2.32).

$$\hat{\pi}(y|\theta) = \prod_{i=1}^{n} \hat{\pi}(y_i|\theta) \tag{2.28}$$

$$\hat{\pi}(y_i|\theta) = \frac{1}{N}\sum_{j=1}^{N} g_\theta(y_i|x_i^j) \tag{2.29}$$

The most straightforward (but sometimes inefficient) way to exploit this for the purpose of parameter inference is to employ particle filters as a modular component within traditional sampling schemes like the example given in Algorithm 7.

Finally, in the description given for Algorithm 6 we have allowed for the sake of brevity that resampling occurs at every step. However in practice [10] recommend to only resample when a given criteria is met, such as by comparing the *effective sample size* (ESS) with a predetermined threshold:

$$ESS = \frac{1}{\sum W^2} \tag{2.30}$$

Algorithm 6 Standard SMC ('particle filter')

```
Require: $N$, $f$, $g$, $q$
  Choose initial sample: $\{\theta_1, x_1\}$
  Sample $X_1^i \sim \mu_\theta(x_1)$ and set $w^i \leftarrow g_\theta(y_1|X_1^i)$
  Compute the normalised weights: $W_1^i \propto w^i$
  for $j := 2$ to $n$ do
    Resample $\{W_{j-1}^i, X_{j-1}^i\}$ to obtain $N$ equally weighted particles.
    Iterate particles: sample $X_j^i \sim f_\theta(x_j|x_{1:j-1})$
    Set $w^i \leftarrow g_\theta(y_j|x_j)$ and compute the normalised weights: $W_j^i$
  end for
  Optionally resample $\{W_n^i, X_n^i\}$ to obtain $\bar{X}_n^i \sim \hat{\pi}(x|y,\theta)$
```

where W are the normalised particle weights. This ensures that we only resample when the variance of the particle weights is large enough to justify doing so, thus avoiding unnecessary degeneracy. It is also the approach used where reference is made to 'resampling criteria' in Algorithms 8 and 9.

Example: particle-MCMC

In a similar way to Algorithm 5, particle filters provide a convenient modular component for inclusion within sampling schemes, leading to frameworks that are theoretically robust and easy to implement. MCMC methods are a popular and convenient choice, i.e. *particle MCMC* [34]. This is demonstrated using the already familiar Metropolis-Hastings algorithm in Algorithm 7. In this case the acceptance probability can be more simply expressed as:

$$p_{mh} = \max\left\{1, \frac{q(\theta_i|\theta_f)\pi(\theta_f|y)}{q(\theta_f|\theta_i)\pi(\theta_i|y)}\right\} \tag{2.31}$$

and computed based on the estimates obtained by (2.28). Recalling that by simple application of Bayes' theorem:

$$\pi(\theta|y) = \frac{\pi(\theta)\pi(y|\theta)}{\pi(y)} \propto \pi(\theta)\pi(y|\theta) \tag{2.32}$$

we can thus evaluate (2.31) without consideration of $\pi(y)$ in the same way as before. Likewise, selection of a symmetric distribution (such as a multivariate uniform or Gaussian) for proposal density q provides the additional convenience that $q(\theta_i|\theta_f) = q(\theta_f|\theta_i)$, thereby allowing the cancellation of those terms in (2.31).

Algorithm 7 Particle MCMC

```
Require: $N$, $q$
  Choose initial sample: $\theta_1$
  Compute $\hat{\pi}(y|\theta_1)$ using Algorithm 6 //i.e. run particle filter
  for $i := 1$ to $n-1$ do
    Sample $\theta_f \sim q(\theta|\theta_i)$, $u \sim U(0,1]$
    Compute $\hat{\pi}(y|\theta_f)$ using Algorithm 6
    Evaluate $p_{mh}$ as per (2.31)
    if $p_{mh} > u$ then
      Set $\theta_{i+1} \leftarrow \theta_f$
    else
      Set $\theta_{i+1} \leftarrow \theta_i$
    end if
  end for
```

Note that: Algorithm 7 is provided for illustrative purposes only. In practice, and partially due to the nature of problems considered in later chapters, other more computationally efficient methods were deemed preferable throughout for the analyses conducted.

Iterative batch importance sampling

We have seen how *particle filters* can be employed as a modular 'component' in another inferential framework (MCMC) to provide a complete algorithm. In that case, the particle filtering algorithm was nested within a different kind of algorithm – the Metropolis-Hastings, which is a form of *rejection sampling.* However, particle filters can also be employed with *importance sampling* schemes, including other varieties of SMC algorithm. In plain terms: a particle filter, within another particle filter. Chopin *et al.,* [9] demonstrate precisely this approach: the use of particle filters within the framework of another SMC method to address the parameter inference problem. The algorithm is appropriately named by the authors as SMC^2. As with PMCMC, SMC^2 employs particle filters in a modular fashion, by embedding them within a more general framework –in this case, that of the so-called *iterative batch importance sampling* (IBIS) method outlined for static models in [44].

The purpose of the 'inner' particle filtering procedure is to weight (and 'mutate') a collection of 'outer' θ particles[24]. The weights of the [outer] particle set is then rebalanced through a resampling step, analogous[25] with the same step in Algorithm 6. A notable feature of the IBIS framework is the 'mutation' step, with the important requirement that m (the *Markov kernel*) must leave the target distribution invariant. A novel application of the IBIS framework is described in §3.2, in which Pooley's model-based-proposal [partially] fulfils this role, i.e. Algorithm 5 is essentially m in that context. In either case, the requirement is fulfilled [in this case] by employing a Metropolis-Hastings step to make MCMC moves. In a helpfully illustrative comparison, courtesy of Chopin himself: MCMC is predicated on the application of a mutating step (or 'kernel') applied to a single 'particle', many times. Here, we are instead applying the same kernel to *many particles*, but only a *single time*[26]. The broad conceptual approach is illustrated using an example, as per the pseudo-code and commentary provided with Algorithm 8.

Example: SMC^2

The mutation step in the general IBIS framework is easily explained by using the example of SMC^2, as per Algorithm 8. Note that, as per the algorithm's original [and far more complete] description [9], new values of θ can be proposed, either independently of the 'current' particle, or conditionally (similar to the manner already described for MCMC). The particle filter is then used to compute an estimate of the marginal likelihoods of the mutated θ particle[s], and evaluate a vector of acceptance probabilities (one for each particle) based on the ancestral weights (denoted α) of the marginal likelihoods computed up to the current time step j:

$$p^i_{mh} = \frac{\alpha^{q,i}_j}{\alpha^i_j} \tag{2.33}$$

$$\alpha_j = \hat{\pi}(y_{1:j}|\theta) \tag{2.34}$$

[24]Depending on one's background and familiarity with the topic, however, it is perhaps less confusing to conceive of this idea, by disregarding the 'inner' procedure as an SMC scheme altogether, and instead regarding it merely as a function that computes [an unbiased estimate of] the marginal density of interest: $\hat{\pi}(x|y,\theta)$ – which is then employed by a *singular* (alternatively, 'outer') SMC procedure.

[25]As already noted in the discussion of particle filters, it is advisable to only resample when elected degeneracy criteria are met, such as the one defined by (2.30).

[26]Please note that this commentary is deliberately simplistic, merely intended to be illustrative.

where $\alpha^{q,i}$ are the ancestral weights of the proposed particles. Recall the illustrative comparison provided by the original authors, with the Metropolis-Hastings step, as used within MCMC: where in the latter N proposals (or 'mutations') are applied to one particle, the mutation step in the IBIS method involves the application of one [proposed] mutation to N particles.

Algorithm 8 Particle IBIS, i.e. *SMC*^{2}

```
**Require:** initial parameter samples $\theta_i$, number of particles $p$
  Set $w^i \leftarrow 1$
  **for** $j := 1$ **to** $n$ **do**
    Run the particle filter to compute the incremental weights associated with the $j^{th}$ observation, and obtain $X^{i,p}_{1:j}$
    Set $w^i \leftarrow w^i\ \hat{\pi}(y_j|\theta_i)$
    Compute the normalised weights: $W^i_j \propto w^i$
    Set $\alpha^i \leftarrow \alpha^i w^i$
    **if** resampling criteria met **then**
      Resample $\{W^i_j, \{\theta_i, X^{i,p}_{1:j}\}\}$ to obtain $\{\frac{1}{N}, \{\bar{\Theta}^i, \bar{X}^{i,p}_{1:j}\}\}$
      Mutate: $m_{pf}\{\bar{\Theta}^i, \bar{X}^{i,p}_{1:j}, \alpha^i\}$
      Set $w^i \leftarrow 1, \{\theta_i, X^{i,p}_{1:j}\} \leftarrow \{\bar{\Theta}^i, \bar{X}^{i,p}_{1:j}\}$
    **end if**
  **end for**
```

Algorthm output: The output of the algorithm is a set of weighted θ samples $\{W^i, \theta_i\}$ which can be resampled to obtain unweighted samples, [approximately] $\{\frac{1}{N}, \bar{\Theta}^i\} \sim \pi(\theta|y)$. The expected value of a given function h can be estimated by:

$$\hat{E}[h(\theta)|y_{1:n}] = \frac{\sum w^i h(\theta_i)}{\sum w^i} \tag{2.35}$$

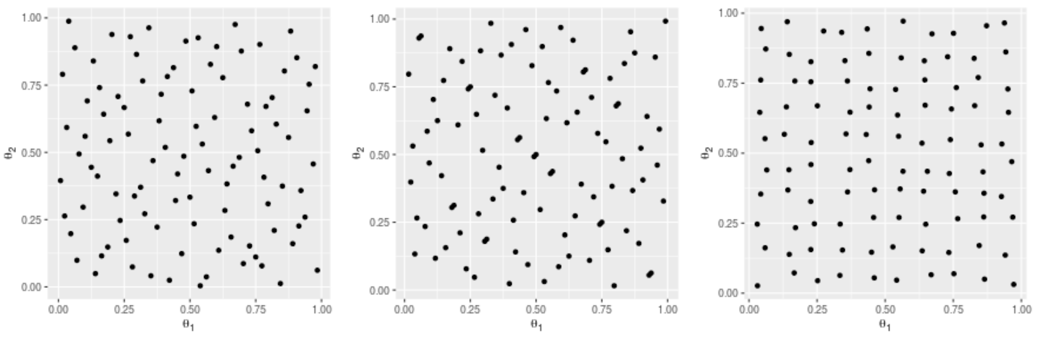
and [44] show that it is consistent and asymptotically normal for all integrable h. The algorithm makes efficient use of the particle filter (compared to PMCMC) and also provides an unbiased estimate of the marginal likelihood $\pi(y_{1:n})$ which can be obtained for a trivial amount of additional computation as described in §2.5.

Efficient use of particle filters is also the key motivation for the algorithm described in §3.3 which is based on a technique broadly described as *quasi-Monte Carlo*, but also inspired by other methods for improving efficiency in MCMC schemes more directly. The latter are discussed further in §2.4.5.

2.4.4 Quasi Monte Carlo

The third and final broad class of inference algorithms described in this chapter are Quasi Monte Carlo (QMC) methods[27]. They are a class of importance sampling methods which involve the use of deterministic or random (RQMC) somewhat evenly spaced (low discrepancy, LDS) sequences in place of the pseudorandom numbers employed in other Monte Carlo methods [12]. The most basic example can be imagined as a uniformly spaced 'grid' on an d dimensional continuous parameter space. The preferential use of LDS as opposed to uniformly spaced samples is accounted for by the specific nature of certain problems, but not addressed in any detail here. Well known examples of deterministic LDS include Latin hypercubes, Sobol and Halton sequences, the latter of which is seeded from a vector of arbitrarily chosen prime numbers [45, 46, 47].

QMC schemes can also be randomised, as in the case of Poisson disc sampling[28]. This involves randomly sampling with a minimum distance between samples to prevent clustering. An approach which is perhaps more conceptually straightforward is to add random noise to sample locations in a uniformly spaced grid, sometimes known as 'jittered sampling' [48]. This also has the advantage of being easy to implement and inexpensive enough to be practical, while allowing for a scalable degree of 'jittering'. Some examples of LDS are illustrated in Figure 2.6.



(a) Halton (b) Sobol (c) Jittered

Figure 2.6: Examples of low-discrepancy sequences. The Halton sequence was initialised with prime sequence $\{2, 3\}$.

QMC schemes tend to require that the parameter space of the function being

[27]https://en.wikipedia.org/wiki/Quasi-Monte_Carlo_method

[28]https://en.wikipedia.org/wiki/Supersampling#Poisson_disc

integrated can be mapped on to the unit hypercube which limits their scope. Unlike MCMC they suffer from the 'curse of dimensionality' [49]. However, at least compared to standard MCMC, they can still achieve rates of convergence that are significantly faster [50], and they have become increasingly popular in fields such as numerical finance for bounded integration problems, even in relatively high dimension (e.g. $d \approx 256$). Intuitively, QMC can be understood to circumvent certain issues that may arise with MCMC, particularly when the target distribution is multi-modal or otherwise difficult to adequately 'explore'.

However if the target density is concentrated in a relatively small region of the space then standard importance sampling methods (including QMC) are likely to waste significant amounts of computation on regions that are of little interest, while failing to sample sufficiently from those that contribute significantly to the target density. In other words we may have reintroduced the very problem that MCMC is used in part to solve! In summary then (and in practice) much appears to depend on the problem at hand.

Quasi MCMC

QMC can also be combined with MCMC in the broadly same manner; replacement of standardly used pseudorandom number generators with LDS, for algorithms including the Metropolis-Hastings [51]. Randomised-QMC variants and those that which rely on the use of randomly permuted (i.e shuffled) QMC sequences are also possible [52, 53]. QMC sequences have also been used to good effect in MCMC schemes [54] which bear conceptual similarity to methods for improving MCMC performance known as 'waste recycling' discussed in §2.4.5, such as the concept described by Frenkel in [55]. However the authors note that the benefit of tends to diminish as the dimensionality of the problem increases. The algorithm introduced in the next chapter (see §3.3) is somewhat basic compared to some of the advanced examples cited here, but was nonetheless found to be highly effective for certain kinds of problem.

2.4.5 Improving MCMC performance

Having laid out the essentials with respect to relevant Bayesian [single-model] inference methods for the class of model described, we now discuss other sundry methods and topics. Namely, performance and validation w.r.t. MCMC algorithms, and multi-model inference (or model comparison). Here again, all are chosen for their relevance to later chapters. We now address the first of those topics; improving MCMC performance. Various methods have been proposed for improving the convergence speed of MCMC, though not all are generally applicable. With respect to SMC methods for example, some could be rationalised in this case on the basis that certain computational tasks –like the one illustrated in Algorithm 6– are computationally expensive, but can be more powerful when used sparingly (compared to standard methods, like Algorithm 7 – better known as *particle-MCMC*).

Delayed acceptance

On a similar basis, [56] propose a *delayed-acceptance MCMC* scheme for models where the target density can inexpensively estimated based on a subset of the available data and provide several practical examples from ecology. Intuitively, the 'expensive' full density computation is only carried out when the sample is reasonably likely to be accepted, by means of this additional step.

Waste recycling

MCMC, and other *rejection sampling* methods are so-called because not all samples are accepted. It has also long been noted of MCMC and rejection sampling methods in general that by discarding information they could justifiably be described as unnecessarily wasteful in many circumstances, giving rise to a class of similar MCMC methods, sometimes referred to as 'waste recycling' [57]. The intuition that using *only* accepted samples is *wasteful*, leads very naturally to the concept of 'waste recycling' in MCMC; the use of rejected samples to better inform our knowledge of the target distribution [57, 58, 59].

The benefit of this approach can also be appreciated by considering (as [56] do) that cases where the acceptance probability is marginal (with respect to $u \sim (0, 1])$ are likely to be relatively rare. In other words, a provisional approximation of the density is likely to lead to the same result (acceptance or rejection) in most cases, and furthermore much information is lost in this step. Thus weight can be added to the power of our statistical estimates by considering the actual likelihood values computed, rather than simply accepting or rejecting the proposed sample.

Recent work has addressed the theoretical foundations of such methods [60]. It has also been noted [55] that, while applicable to any [valid] MCMC algorithm, they may be especially effective in those that utilise *multiple* proposals in parallel. As has already been briefly mentioned, this concept has been combined to good effect with QMC methods by [54] in order to provide more even coverage of the 'local' parameter space, similar to the way envisaged by [55]. Some of these ideas are themselves recycled in Section §3.3, resulting in the development of a novel algorithm that combines aspects of [randomised-] QMC with SMC.

2.4.6 MCMC Convergence diagnostics

The goal of an MCMC analysis is to construct a Markov chain that has the target distribution as its equilibrium distribution. In the case of a single chain analysis we begin with an arbitrarily (but sensibly) chosen sample. In practice it is also sensible to give care and attention to choosing at least an approximately optimal proposal density $q(.)$. That is, while an algorithm may be theoretically valid, and thus *guaranteed* to converge, the *speed* at which a Markov chain converges on (or resolves to) its equilibrium distribution (i.e. the target distribution) depends greatly on $q(.)$. Assessing whether or not the chain has converged to an acceptable degree is naturally somewhat challenging, if we do not already know the target distribution in advance. Visual inspection of the Markov chain for 'mixing' may be sufficient to diagnose problems in some cases but not in all, particularly if the target distribution is multi-modal and irregular.

Formal methods of diagnosing convergence are therefore a vital component of any rigorous MCMC analysis. The rest of this section describes the *Geweke* and *Gelman-Rubin* diagnostics; the tools implemented for single and multiple chains respectively in the software package presented in Chapter 5. Furthermore, in practice many issues can be avoided if many chains are run with widely dispersed initial samples, and that is the approach that has been used throughout this thesis. Therefore only the Gelman-Rubin test is actually pertinent to the results of later chapters.

Geweke test of stationarity

The Geweke statistic tests for *non-stationarity*, by comparing the mean and variance of component-wide parameter estimates, given as θ_i for consistency of notation, for two sections of the Markov chain, denoted here as $\{\alpha, \beta\}$[61, 62]. It is given by:

$$z = \frac{\bar{\theta}_{i,\alpha} - \bar{\theta}_{i,\beta}}{\sqrt{Var(\theta_{i,\alpha}) + Var(\theta_{i,\beta})})} \tag{2.36}$$

A commonly used parametrisation to compute the z statistic is such that, the first 10% of the chain (not including discarded samples from the adaptation period) is compared the final 50%, or $\alpha = 0.1$ and $\beta = 0.5$. Large differences, i.e. $abs(z) > 2$, in the means of the parameters indicate a lack of convergence within at least one of the segments. This would typically suggest that more samples are required, or else another problem with the analysis.

Gelman-Rubin diagnostic

The Gelman-Rubin diagnostic is designed to diagnose convergence of two or more Markov chains by comparing within chain variance to between chain variance [63, 19]. The *estimated scale reduction* statistic (sometimes referred to as *potential scale reduction factor*) is calculated for each parameter in the model.

Let $\bar{\theta}$, W and B be vectors of length P representing the mean of model parameters θ, within chain variance between chain variance respectively for M Markov chains:

$$W = \frac{1}{M} \sum_{i=1}^{M} \sigma_i^2 \tag{2.37}$$

$$B = \frac{N}{M-1} \sum_{i=1}^{M} (\hat{\theta}_i - \hat{\theta})^2 \tag{2.38}$$

The estimated scale reduction statistic is given by:

$$R = \sqrt{\frac{d+3}{d+1} \frac{N-1}{N} + \left(\frac{M+1}{MN} \frac{B}{W} \right)} \tag{2.39}$$

where the first quantity on the RHS adjusts for sampling variance, and d is degrees-of-freedom, estimated using the method of moments. For a valid test of convergence the Gelman-Rubin requires two or more Markov chains with initial samples that are *over-dispersed* (larger variance) relative to the target distribution. As an aside, this resolves the earlier noted 'discrepancy', concerning the claim that Markov chains can be initialised from *arbitrary* locations. In practice, certain choices may impact the [perceived] validity of a [multi-chain] analysis.

2.5 Inference with multiple models

Thus far we have implicitly assumed that we are dealing with the problem of parameter inference from a single model perspective. That is, we assume that we know of an appropriate model and simply wish to fit data to it. However it is not always the case that we know the appropriate model to fit at the outset, and so the problem arises of how to select one from the available candidates.

Alternatively, we may simply have use for a quantitative measure of how well a single model fits some given data. In keeping with the Bayesian framework laid out in §2.3.2, we can utilise the *marginal likelihood*, referred to in this context as the [Bayesian] 'model evidence'[29]. For a given model:

$$\pi(y) = \int \pi(y_{1:n}|x_{1:n}, \theta)\pi(x_{1:n}|\theta)dx_{1:n}d\theta \tag{2.40}$$

In practice it is often convenient to use a log transformation, such as the one used by Pooley and Marion to compare Bayesian model evidence with alternative measures of model fit [64]:

$$-2\ln \pi(y) \tag{2.41}$$

Thus yielding a quantity more relatable to the Akaike information criterion (AIC) Bayesian information criterion (BIC) and similar statistics:

$$BIC = k \ln n - 2(\ln \hat{L}) \tag{2.42}$$

where this k is the number of free parameters in the model; n could be the sample size, depending on the type of model; and $\hat{L}$ is the maximised likelihood value[30].

2.5.1 Model selection using model evidence

Extending the essential Bayesian inferential framework to the problem of model selection (or 'model comparison') we can derive an expression for the different choices

[29]https://en.wikipedia.org/wiki/Marginal_likelihood#Bayesian_model_comparison

[30]Source: https://en.wikipedia.org/wiki/Bayesian_information_criterion

of model available:

$$\pi(y, M_i) = \int \pi(M_i)\pi(y_{1:n}|x_{1:n}, \theta, M_i)\pi(x_{1:n}|\theta, M_i)dx_{1:n}d\theta \tag{2.43}$$

where M_i is the i^{th} model chosen and $\pi(M_i)$ is the corresponding prior distribution, i.e. the probability of selecting that model. Based on this the Bayes factor provides a standardised way to directly compare the model evidence for two candidate models:

$$K_{1,2} = \frac{p(y|m_1)}{p(y|m_2)} \tag{2.44}$$

where, according to the scale originally proposed by Jeffreys [65], $K_{1,2} > 10$ can be considered to be strong evidence for favouring model m_1 over m_2.

$\log_{10}K$	K	Strength of evidence
0 to 1/2	1 to 3.2	Not worth more than a bare mention
1/2 to 1	3.2 to 10	Substantial
1 to 2	10 to 100	Strong
>2	>100	Decisive

Table 2.2: Bayes factor interpretation, from Kass and Raftery [1].

2.5.2 Estimating the model evidence

Here we describe the main method that was used to estimate the model evidence [marginal likelihood] for the algorithms that are carried forward to the BTB analysis presented in Chapter 6. That method concerns SMC algorithms. Techniques applicable to data-augmented MCMC algorithms are also available, including the class of methods described in §2.4.2 [66, 64], but they are not directly relevant to the results presented in later chapters and so have not been addressed here. Moreover, computing the marginal likelihood is a non-trivial (but possible) task in data-augmented MCMC schemes; with the available methods both difficult to implement and computationally taxing. In practice, deviance information criterion (DIC) is often used instead, presumably for those reasons. However that approach has been challenged as being less effective in at least some situations [64]. As discussed in the next chapter, it was ultimately more convenient to adapt the data-augmented [MBP-]MCMC method itself, in order to make it amenable to the technique described here, rather than attempt to solve that particular problem more directly.

Estimating the model evidence using SMC

Directly computing (2.40) is typically just as intractable as directly computing the posterior distribution of the model parameters (for the class of models described herein) but note that by simple application of the chain rule:

$$\pi(y) = \pi(y_{1:n}) = \prod_{i=1}^{n} \pi(y_i | y_{1:i-1}) \tag{2.45}$$

where $y_{1:n}$ are a series of n longitudinal observations (i.e. precisely the kind of data considered here). This is useful because Chopin *et al.,* [44, 9] show how an unbiased and asymptotically normal estimator of the quantity y_i can be computed, using the SMC^2 algorithm, and taking the weighted average of the incremental particle weights:

$$\hat{\pi}(y_i) = \frac{1}{\sum w} \sum_{j=1}^{n} w^j \hat{\pi}(y_i | y_{1:i-1}, \theta^j) \tag{2.46}$$

where w^j is the weight of the j^{th} θ particle and $\hat{\pi}(y_i | y_{1:i-1}, \theta^j)$ is the incremental weight (i.e. the likelihood associated with the i^{th} observation) estimated by means of a particle filter.

2.6 Summary

The purpose of this chapter was to provide a practical review –rather than a comprehensive, or theoretical treatment– of selected fundamental concepts and algorithms, as a prelude to consideration of within-herd disease dynamics in UK cattle herds. In particular, special attention was given to the methods algorithms that were used to develop those introduced in the Chapter 3 and used to implement a robust Bayesian workflow in Chapter 4. A further chapter covers another key output of the project: [somewhat] generalised implementations of selected methods for Bayesian inference, including many of those that were introduced here. Following that, a selection of the algorithms, including SMC^2 (Algorithm 8) are set against the problem of parameter and model inference, as it applies to the aforementioned motivating problem. That is in Chapters 6 and 7.

Chapter 3

Hybrid Algorithms for inference

"everything is a remix"

- Kirby Ferguson [Everything is a Remix (2015)]

Summary

- This chapter develops two novel Bayesian inference methods for discrete state space partially observed Markov Processes (DPOMPs) that address weaknesses of existing algorithms by creating hybrids that exploit the strengths of contrasting approaches.
- (1) MBP-IBIS: leverages the efficiency of data-augmentation model-based-proposals (MBPs - Algorithm 5) as a Markov chain Monte Carlo (MCMC) kernel within Sequential Monte Carlo framework, specifically the Iterated Batch Importance Sampling (IBIS) algorithm (§2.4.3)
- (2) ARQ-MCMC: combines standard particle-MCMC with adaptive Randomised quasi-Monte Carlo (QMC) sampling. It is more generally applicable than MBP-IBIS but here is also applied to DPOMP models.
- In MBP-IBIS use of data augmentation aids posterior sampling by preserving characteristics of state-space trajectories consistent with the data. On the other hand the SMC methodology of IBIS enables calculation of the model evidence which is difficult in standard data-augmented MCMC.

- In contrast ARQ-MCMC inherits such benefits from the SMC toolkit and addresses the problem of 'waste recycling' (see §2.4.5) by better of targeting computational effort expended on evaluation of the likelihood.
- These algorithms are validated and compared, both against each other and existing standard algorithms using simulated data scenarios.
- In light of these results we suggest that both have a role to play in the implementation of practical Bayesian workflows for DPOMPs explored in later chapters.

3.1 Introduction

This chapter describes two novel Bayesian inference algorithms, developed by combining particular strengths of existing approaches described in Chapter 2 and discussed below. The first, MBP-IBIS leverages the efficiency of data-augmentation model-based-proposals (MBPs - Algorithm 5) as a Markov chain Monte Carlo (MCMC) kernel within the Iterated Batch Importance Sampling (IBIS) sequential Monte Carlo (SMC) framework (§2.4.3). The second algorithm, ARQ-MCMC combines standard particle-MCMC with an adaptive Randomised quasi-Monte Carlo (QMC) sampling of parameter space and is more generally applicable than MBP-IBIS which is focused on DPOMPs (this is due to the specific form of model-based proposals considered). ARQ-MCMC addresses the problem of 'waste recycling' (see §2.4.5) by making more efficient use of computational effort in particular by better targeting effort spent on evaluation of the likelihood (see §3.3).

In Chapter 4 we show how these new algorithms can be used in conjunction with existing methods to develop robust workflows that build on their contrasting strengths and weaknesses to address practical difficulties encountered when applying single- and multi-model Bayesian inference to DPOMPs. In addition, both algorithms address a key difficulty with data-augmented MCMC namely estimating the model evidence (marginal likelihood the of the model) thus enabling multi-model inference (see §2.5). As noted in Section §2.5.2, computing the marginal likelihood is a non-trivial (but possible) task in data-augmented MCMC schemes; with available methods both difficult to implement and computationally taxing. In practice, deviance information criteria (DIC) are often used. However, for latent variable models there are several definitions of DIC and these have been found to be inconsistent and unreliable for assessing the structure of epidemiological DPOMP models [64].

Bayesian analysis of scientific data requires judicious choice of the prior distribution $\pi(\theta)$. In practice, a cautious approach may involve repeated analysis of the same model and data with differently chosen prior distributions, in order to identify the one that: optimally encapsulates our prior beliefs; decide which is the most 'objective'; or merely for the purposes of cross-validation and comparison. Thus a secondary motivation and benefit of the *ARQ-MCMC* algorithm is that the nature of its resampling approach can be used to improve the efficiency of multiple quasi-independent MCMC analyses, even with differently selected prior distributions. This is achieved by 'sharing' the computational cost of likelihood evaluations within, and between, different analyses (with possibly different prior distributions). This includes cases where candidate prior distributions are from a different family of distribution altogether, including those that are flat and [partially] unbounded (i.e. 'improper').

This chapter focuses on motivating, describing and then testing these new algorithms.

Motivating MBP-IBIS

The contrast between updates, or mutations, in data-augmentation MCMC and sequential Monte Carlo (SMC) approaches such as IBIS can be described as follows. In DA-MCMC we mutate a single 'particle' N times to form a *Markov chain* whereas in IBIS we mutate a *collection* of N particles a single time [44]. Here mutations means some stochastic update to the current state (i.e. respectively, perturbing the state of the Markov chain or the set of particles i.e. parameters) which will then be accepted or rejected in a way that in some sense guarantees convergence to the distribution of interest: a *Markov kernel*.

A key advantage of the IBIS framework is that it provides an inexpensive way to compute unbiased estimates of the marginal likelihood, which as noted, although possible in DA-MCMC is non-trivial in terms of both implementation and computational cost. A particular strength of data augmentation methods including MBPs is that they naturally preserve the characteristics of state-space trajectories that are consistent with the data. In contrast SMC methods are built on forward time simulations that must be re-weighted to avoid particle degeneracy. The MBP-IBIS algorithm introduced below is designed to leverage both these advantages to create an efficient implementation of Bayesian inference for DPOMPs in an approach known as particle annealing [1] as per the technique described for [discrete-time] hidden Markov models by [10].

To summarise, the key motivation is to combine the effectiveness of (MBP) resampling techniques, whilst mitigating particle degeneracy issues that can arise within SMC schemes as a result of them. In order to accomplish this, particles are simulated in the standard fashion but the MBP algorithm is utilised in construction of a *Markov kernel*, so as to mitigate degeneracy without the need to fully re-simulate particles at each iteration – that is, the 'mutation' step in IBIS (and SMC^2, for which that is a requirement). MBP-IBIS is therefore an SMC version of the data-augmentation method – a hybrid of both approaches. We have already seen how a particle filter can be used as the 'inner' procedure within the IBIS framework: SMC^2. Likewise, the MBP algorithm provides an update 'kernel' that can be applied to similar affect within the same overarching framework of Chopin.

Finally, we note that although we describe MBP-IBIS as a hybrid of data-augmentation and SMC methods, the distinction between 'online' and 'offline' algorithms, is more useful in terms of implementation and understanding performance. That is because understanding whether or not the full history of the state variable needs to be held in memory, conveys useful information about the algorithm's construction. In this parlance MBP-IBIS transforms the online IBIS into an offline algorithm via the introduction of data-augmentation MBPs. The MBP-IBIS algorithm is fully described in Section §3.2 bt first we provide background to motivate the ARQ-MCMC algorithm.

Motivating ARQ-MCMC

Random sampling methods are useful for integrating probability densities (which we denote here as $f(\theta)$ for consistency of notation) when the density can be conveniently sampled [estimated] for a given θ-tupple but directly solving (i.e. integrating) for *all* θ (i.e. using quadrature to integrate over the entire space) is impractical.

[1] Described as PAIS-MBP on pp19 of the BEEPmbp user manual as downloaded on 3rd July 2021. See: https://github.com/ScottishCovidResponse/BEEPmbp for the latest version.

This is often the case when dealing with real-world integration problems, such as Bayesian inference in the context of scientific domain models. Unfortunately, $f(\theta)$ are often computationally intensive to evaluate e.g. because doing so also involves solving *another* difficult integral. In such situations it is usually not practical to compute an exact value for the density, and we may only be able to obtain an [unbiased] estimate $\hat{f}(\theta)$. Furthermore if the estimate suffers from high variance, it exacerbates the overarching computational challenge. The motivating problem for this work are problems that can be solved using particle filters, since they can be used to estimate the likelihood function for the models in consideration. However, the underlying motivation can be stated more plainly: integrating reasonably low ($d < 12$) dimension $\hat{f}(\theta_d)$ that are computationally intensive to evaluate, in particular those with possibly high variance.

Alternatively, recall the computational *efficiency* of particle filters (as 'online' algorithms) for solving certain problems. In that context the motivations for the algorithm can be viewed through the lens of 'waste recycling' for [particle-] MCMC, as discussed in §2.4.5.

The ARQ-MCMC algorithm was developed for Bayesian inference, where $f(\theta)$ defines (i.e. computes) the likelihood function. That is, we wish to use some data, y, to learn about an underlying population [or probability distribution,] denoted by θ. For the purposes of this thesis θ represent the parameters of an underlying disease transmission model (or some other scientific domain model). Recalling the definition for Bayes' theorem, we have:

$$\pi(\theta|y) = \frac{\pi(\theta)\pi(y|\theta)}{\pi(y)} \propto \pi(\theta)\pi(y|\theta) \tag{3.1}$$

$$\pi(y|\theta) := f(\theta) \tag{3.2}$$

where $\pi(\theta)$ expresses our *prior* beliefs about the distribution of the model (i.e. function) parameters, and $\pi(\theta|y)$ is the *posterior* distribution of the model parameters, meaning *after* accounting for y. Note that here we are not explicitly interested in dependence on latent variables (e.g. describing state-space trajectories in dynamic models) and assume these are 'integrated out'. The need for such *inner* integrals is a principle reason why estimates $\hat{f}(\theta_d)$ are typically subject to high variance. The proportionality of the posterior likelihood to the product of prior distribution and the likelihood function, given on the RHS of (3.1), can be exploited to draw posterior samples of θ – this is the approach used here.

The use of [approximately] evenly-spaced samples, lattices, or 'grids' (visualised as points in the parameter space) is a well-established technique for estimating integrals, including probability distributions. The techniques of *quasi*-Monte Carlo (QMC) discussed in §2.4.4 provide tools to deterministically generate so-called low-discrepancy sequences (LDS) as the basis for such grids. Where pseudorandom numbers are used, the sequences may be referred to as *randomised*-QMC (see Figure 2.6 for examples). The adaptive randomised Quasi-Monte-Carlo Markov chain Monte Carlo *ARQ-MCMC* algorithm is so called because it relies on MCMC rejection sampling to adaptively *resample* a *Randomised* QMC sequence. Under this scheme a *quasi* Monte Carlo sequence of θs, is 'resampled' with $f(\theta)$ computed on-the-run and cached for future use.

ARQ-MCMC is a simple design with a discretised MCMC scheme with an abstraction layer consisting of a cache indexed by θ-valued $d-$tuples. The guiding principle is to minimise the number (and maximise the usefulness) of calls to a notionally expensive likelihood function $f(\theta_d)$, or usually estimator functions $\hat{f}(\theta_d)$ (e.g. particle filters) with high variance. It also utilises *all* of the information gleaned from *both* accepted and rejected samples. Thus the algorithm addresses the *waste recycling* problem as it manifests in online particle MCMC (see §2.4.5 for a discussion related to offline MCMC). The adaptive nature of ARQ-MCMC arises from a simple tunable mechanism that adjusts the density of samples judged necessary for sufficiently smooth marginal estimates of the posterior distribution. The benefits of this algorithm diminish rapidly as the dimension d of θ_d increases and the probability of revisiting any location (θ-coordinate) becomes more remote. In practice this means ARQ-MCMC is applicable to models where relatively small number of parameters $d < 12$ need to be estimated.

Chapter overview

The remainder of this chapter is organised as follows. The MBP-IBIS algorithm is described in detail including with pseudo-code in Section §3.2. A simple applied problem is then used to demonstrate use of the algorithm for both single and multi-model inference in §3.2. The ARQ-MCMC algorithm is described in detail including with pseudo-code in Section §3.3. A simple applied problem demonstrates the utility of the algorithm for Bayesian analysis with multiple prior distributions and benefits of the method in cases where estimation of the likelihood is subject to high variance (§3.3). The chapter concludes with a multi-algorithm validation and speed comparison and discussion (§3.4). As noted, both algorithms have been made available (along with others) as a software package described in Chapter 5. That includes applied examples of the algorithms described here, based on real and simulated data sets.

3.2 The MBP-IBIS algorithm

MBP-IBIS takes advantage of the *model based proposal* (MBP) algorithm [4] (Algorithm 5) within an iterative-batch-importance-sampling (IBIS) framework similar to that used in Algorithm 8, i.e. SMC^2 [9]. The MBP method is employed to construct a *Markov kernel* for the 'mutation' step, and the DGA simulation (Algorithm 1) is used for the iteration step. For comparison, both tasks are performed within the SMC^2 [IBIS] algorithm by a *particle filter* [9].

Here we define MBP-IBIS algorithm in detail. The pseudo code is shown in Algorithm 9 and each of the steps are described in more detail below.

Algorithm 9 MBP-IBIS

```
Require: initial parameter samples $\theta_i$
  Set $w^i \leftarrow 1$, $\alpha^i \leftarrow \pi(\theta_i)$
  for $j := 1$ to $n$ do
    Iterate particles: sample $X^i_j \sim f_\theta(x_j|x_{1:j-1})$ to obtain $\{\theta_i,\, X^i_{1:j}\}$
    Set $w^i \leftarrow w^i g_\theta(y_j|x_j)$ and compute the normalised weights: $W^i_j$
    Set $\alpha^i \leftarrow \alpha^i w^i$
    if resampling criteria met then
      Resample $\{W^i_j, \{\theta_i, X^i_{1:j}\}\}$ to obtain $\{\frac{1}{N}, \{\bar{\Theta}^i, \bar{X}^i_{1:j}\}\}$
      Mutate: $m^{\kappa}_{mbp}\{\bar{\Theta}^i, \bar{X}^i_{1:j}, \alpha^i\}$
      Set $w^i \leftarrow 1, \{\theta_i, X^i_{1:j}\} \leftarrow \{\bar{\Theta}^i, \bar{X}^i_{1:j}\}$
    end if
  end for
```

Steps

As with *SMC* ^{2} the algorithm is seeded with a collection of N θ-particles denoted θ_i. The steps of the algorithm are then as follows:

1. **Simulate**: Initial (or next) segments of the trajectory corresponding to θ_i are simulated. Note that for the first step we sample $X_1^i \sim \mu_\theta(x)$. However in accordance with the convention already established for particle filtering $\mu(.)$ is denoted by $f(x_1|x_{1:0})$ in pseudo code.

2. **Particle weights**: At the end of the time step the probability density function associated with the observation process g_θ is used to compute the incremental weights for each particle in $\{\theta_i, X^i\}$; update the particle weights w^i; and compute the normalised particle weights W^i.

3. **Resampling**: Assuming that the resampling condition computed according to (2.30) is satisfied, the particles are resampled in the familiar manner to obtain a set of equally weighted particles: $\{\bar{\Theta}^i, \bar{X}_1^i\}$ approximately $\sim \pi(\theta, x_1|y_1)$.

4. **Mutation**: Candidate parameters are sampled from a chosen distribution and paired with a trajectory generated using the MBP algorithm. The 'mutated' particle is then accepted with Metropolis-Hastings probability:

$$p_{mh} = \max\left\{1, \frac{\alpha_f}{\alpha^i}\right\} \tag{3.3}$$

$$\alpha_f = \pi(\theta_f) g_{\theta_f}(y_t|x_{1:t}^q) \tag{3.4}$$

   where the ancestral weights denoted by α are computed up to the current time step t. Repeat for a total of κ times where κ is predetermined and may be arbitrarily increased as a variance reduction measure.

5. Return to step 1 until $t = n$.

Outputs

The output of the algorithm is a set of weighted particles $\{\theta_i, X^i\}$ that can be resampled to obtain an arbitrary number of particles $\{\bar{\Theta}^i, \bar{X}_1^i\}$ approximately $\sim \pi(\theta, x_{1:n}|y_{1:n})$. As with SMC^2, the expected value of any integrable function h can be estimated by (2.35).

Estimating the model evidence

As noted in §2.5, a useful feature of IBIS is that it is possible to obtain an estimate for the marginal likelihood that is both unbiased and computationally inexpensive.

The corresponding quantity for MBP-IBIS is computed thusly:

$$\hat{\pi}(y_{1:n}) = \frac{1}{\sum w} \sum_{j=1}^{n} w^j \hat{\pi}(y_i | y_{1:i-1}, x_{1:i}^j, \theta^j) \tag{3.5}$$

Tuning the algorithm

Use of the MBP algorithm allows us to perturb the particles in a way that leaves the target distribution invariant, without discarding trajectories that we already to believe are approximately distributed according to the appropriate marginal distribution. As with other types of particle filter the accuracy of the estimates obtained increases with N but is sensitive to numerical stability issues with large N. However smoother estimates can be obtained by increasing the number of iterations in the mutation step, κ, as illustrated in figure 3.1. This is roughly equivalent to increasing N in terms of computational cost, but circumvents the stability issue associated with increasing N beyond a certain threshold, and demonstrates a particular advantage of MBP-IBIS.

Testing the algorithm

Here we test the algorithm for single and multi-model inference, using the model and data set published by Pooley *et al.,*in the paper that introduced the MBP method [4]. This (now familiar) SIS model is illustrated in Figure 3.11. The observations data consist of five equally spaced (in time) observations of the number of individuals in the infectious state up to $t = 100$. The number of individuals is held constant and the initial state is given as $\{100, 1\}$ at time $t = 0$. The observation model density g is not parameterised by θ in this case, since it is assumed to be normally distributed with a (known) error of $\sigma_y = 2$, i.e. $g(y_i) \sim \mathcal{N}(x_i, 4)$.

A number of additional models were also fitted to this data in order to provide further illustration of the ability if MBP-IBIS to infer model parameters and as a demonstration of its ability to support the approach to model selection described in §2.5. The most elaborate of these, the SEIS model, is depicted along with estimates of the model evidence for each model in Figure 3.3. The full model

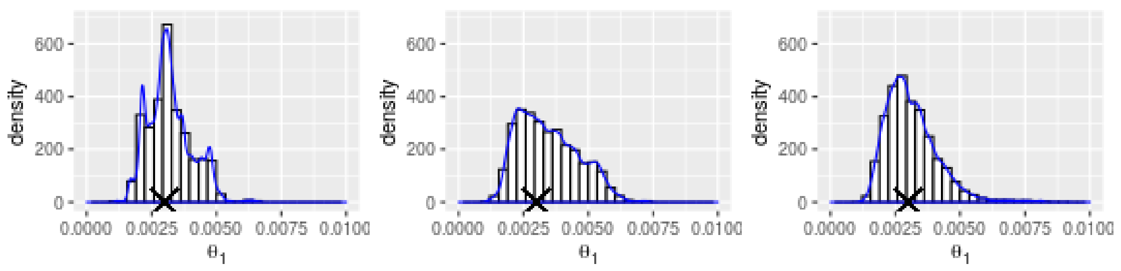
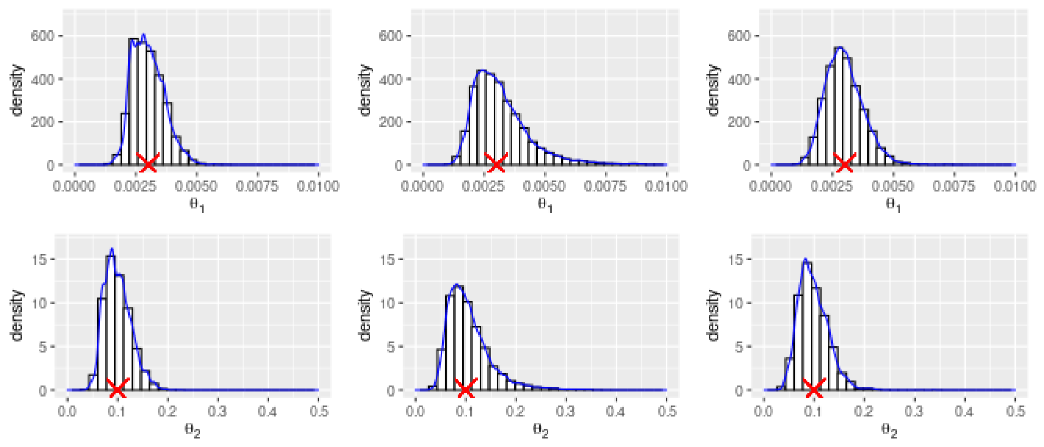


Figure 3.1: **Impact of the MBP mutation step**: Sample output from Algorithm 9 illustrating the impact of the number of iterations in the mutation step, $\kappa = \{1, 3, 10\}$ (left to right). The illustrated samples were obtained from a simulation study, and correspond to the *contact rate* parameter for the SIS model illustrated in Figure 3.11b. κ denotes a user-defined parameter used in construction the *Markov kernel* and determines the number of *model based proposals* used to perturb each particle during the mutation step. It can therefore be tuned to achieve a desired degree of accuracy at the cost of additional algorithm run time (see for example Figure 3.13). As before the values used to generate the simulated data are marked for reference.

specifications are given in the supplementary material but in general the same observation model was applied to the infectious compartment in each model. Both the MBP-IBIS and the SMC^2 algorithms distinguished between models with dynamics that clearly do not appear to fit data such as the *SIR* and one of its derivatives; the *SEIR* model (not shown).

However the estimates obtained for the more similar *SEIS* model (see Fig. 3.3) were slightly lower, suggesting a better fit with that model. Analysis of these results indicated that the posterior mode for both the recovery and the extraneous progression (i.e. $E \rightarrow I$) parameters θ_2, θ_3 was ≈ 0.2. Recalling that inter event times are distributed exponentially for Poisson-like processes we can interpret this as the inverse of the average duration spent by any given individual in either compartment, e.g. if $\theta_n \approx 0.2$ then $\theta_n^{-1} \approx 5$ units of time. Estimates of the contact parameter θ_1 were, likewise, higher for the *SEIS* model than those obtained for the *SIS* model. Unsurprisingly, the estimates obtained for the latter were far more in concert with the parameter values used to simulate the original data set (using the *SIS* model).

While we were roughly able to infer the duration spent in non-infectious com-



(a) SMC^2 IBIS $N = 6000$.

(b) MBP MCMC $N_C = 3$, $N_C = 50000$.

(c) MBP IBIS $N = 10000$, $\kappa = 7$.

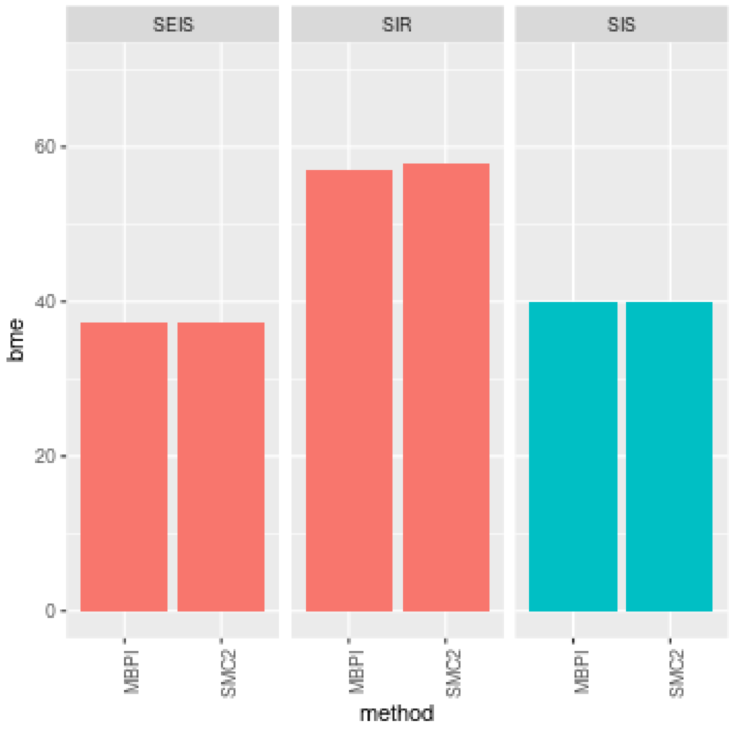
Figure 3.2: **Testing MBP-IBIS algorithm**. Marginal model parameter distributions for SIS model parameters (based on the Pooley data set, see text for details) for selected algorithms with moderate performance configurations (as reflected by the variances in the estimates obtained). The first 10000 samples were discarded from each chain for the MBP MCMC analysis (i.e. the 'burn in' period).

partments as ≈ 10 time units, we were unable to distinguish between the correct dynamics and a more infectious pathogen that involves a significant non-infectious period. This failure draws attention to the fact that unlike some other measures of model deviance, consideration of the model evidence alone does not include a penalty for additional terms. Caution must therefore be applied in its interpretation with models that are distinguished in this manner; a point that has a direct bearing on our interpretation of the results presented in Chapter 6.

Note that in practice, we would likely choose the simplest model (i.e. with the lesser number of free parameters) as our 'null' model, and only proceed to reject it in favour of the complex one (the SEIS in this case) if the *Bayes factor* met some predefined threshold, as described in §2.5. Furthermore, it is notable that the uncertainty in parameter estimates shown in Fig. 3.4 is somewhat larger for the SEIS than for the SIS model. However our purpose here was to evaluate the MBP-IBIS algorithm itself, and thus we have directly compared estimates of the marginal likelihood, or *Bayesian model evidence*, $\pi(y)$.

$$\begin{tikzcd}
\boxed{S} \arrow[r, "\theta_1 SI"] & \boxed{E} \arrow[r, "\theta_2 E"] & \boxed{I} \arrow[ll, bend left, "\theta_3 I"]
\end{tikzcd}$$

(a) SEIS model

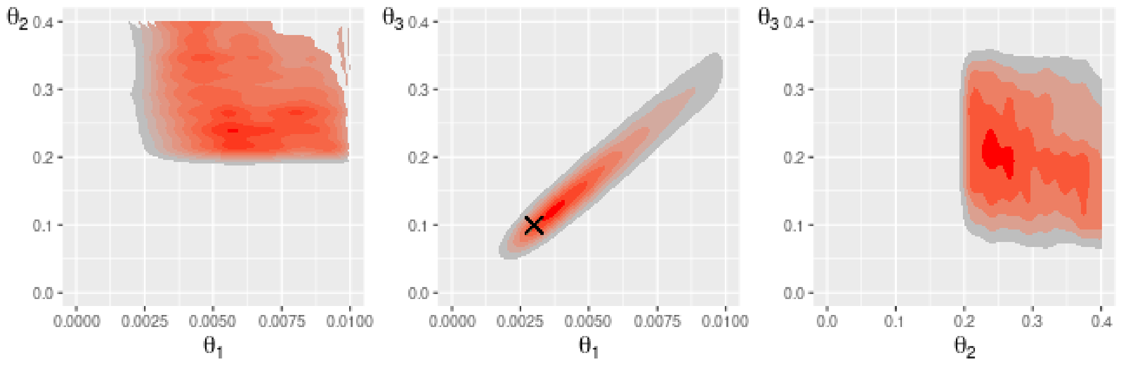


(b) Model evidence

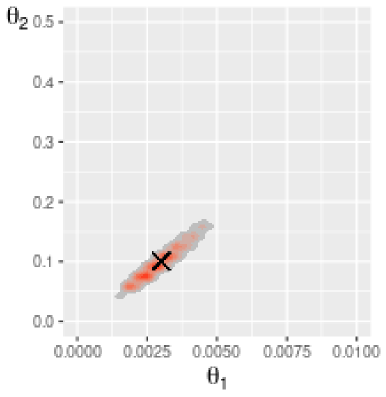
Figure 3.3: **Estimating Bayesian Model Evidence** (i.e. the marginal likelihood, $\pi(y)$) results for the Pooley data set using the method described in §2.5, for the *SMC*^{2} and *MBP-IBIS* algorithms. In each case the algorithms produce similar estimates of $\pi(y)$. The model that generated the data is highlighted (turquoise bars). The analysis was sufficient to accurately distinguish between models that plainly do not fit the pattern of the data such as the SIR and SEIR but estimates of the model evidence for the similar SEIS model (depicted) were lower, suggesting a better fit with that model. This is discussed further in the main text.

3.3 The ARQ-MCMC algorithm

In keeping with the generic Bayesian model laid out in the introduction by equations (3.1) and (3.2), we assume a known [inference] model, including any prior



(a) Marginal sample densities for the SEIS model.



(b) SIS model.

Figure 3.4: **Model assessment**. Comparison of marginal importance sample results for the *SEIS* and *SIS* models as fitted to the simulated SIS data set borrowed from [4].

density function[s] $\pi(\theta)$, and a computable [log] likelihood function $f(\theta) \sim \pi(y|\theta)$. A comprehensive description of the algorithm is provided in due course, but first we lay out some of the key modular components and concepts that inform the algorithm's design in order to illustrate the broad approach. Namely the hypothetical importance sample; *re*sampling; and the adaptive MCMC kernel that is used to accomplish that resampling.

Importance sample Γ: We begin by *notionally* defining an importance sample Γ, using an appropriate QMC sequence construction in the required dimension $d = |\theta|$. For the sake of simplicity, let it be a uniform 'grid'. The number of distinct theta tuples (the cardinality of Γ) is given by $|\Gamma| = S^d$, where S corresponds to $S - 1$ intervals (for each of the d dimensions) and the intervals are user-defined to allow for the scaling of S. Note that the intervals (and thus, the number of samples $|\Gamma|$) may be chosen such that computing $f(\theta)$ for each distinct θ-tuple (i.e. for each sample) in Γ would be impractical.

Resampling kernel: Let q be a 'proposal density' of the form $q(\theta_b|\theta_a)$ (more accurately, a mass function) which relates the probability associated with a random walk, from a to b, on Γ. We now have the basic ingredients for a *discretised* version of Algorithm 7, an MCMC rejection sampler. This allows for Γ to be efficiently *resampled*, with $f(\theta)$ computed 'on-the-run'. Intuitively, we wish to spread computational effort throughout the posterior mass. Furthermore the goal is already partially accomplished by virtue of the fact that samples are at least approximately evenly-spaced.

Adaptive MCMC proposals: Construction of an appropriate proposal density q is an important design consideration. In this case, an adaptive scheme was devised with two automatically tunable components. The first is the distance between and a and b (by the shortest path) which we denote by j. The sampling distribution of j is uniform and discrete. Its parametrisation is adapted at regular intervals according to the [target] MCMC proposal acceptance rate. The second is a vector of probabilistic weights (corresponding to each elements of θ) that are used to generate the random walk to b. This allowed for the random walk to be scaled in accordance with estimates of the posterior variances, based on samples obtained up to a given point. In other words, adaptive MCMC. Both of the adaptive MCMC parameters are reset after each individual Markov chain, in order to maintain that aspect of their quasi-independence.

We now lay out a more detailed description of the *ARQ-MCMC* algorithm, providing pseudo code in the forms of Algorithms 10 and 11. We also provide a detailed step-by-step explanation below this. Note that for ease of exposition Algorithm 10 has been simplified by describing just the inner MCMC procedure as a recursive function with ($N_{MH} - 1$ steps) that accepts (or initialises) the Γ object; updates; and returns it along with the parameter samples. Algorithm 11 serves as a simple intermediate layer (i.e. a 'wrapper') for the target function $f(\theta)$, and also incorporates the prior density function $\pi(\theta)$.

Algorithm 10 Recursive ARQ MCMC procedure

```
Require: $\pi$, $f$, $q$, $N_{MH}$
Require: (Or initialise) $\Gamma$
  Sample $\theta_1 \sim \Gamma$
  Set $p_1 \leftarrow \phi_\Gamma(\theta_1)$ using Algorithm 11
  for $k := 1$ to $N_{MH} - 1$ do
    Sample $\theta_f \sim q(\theta|\theta_k)$, $u \sim U(0, 1]$
    Set $p_f \leftarrow \phi_\Gamma(\theta_f)$ using Algorithm 11
    Evaluate $p_{mh} = \frac{\pi(\theta_f)p_f}{\pi(\theta_f)p_k}$
    if $p_{mh} > u$ then
      Set $\theta_{k+1} \leftarrow \theta_f$, $p_{k+1} \leftarrow p_f$
    else
      Set $\theta_{k+1} \leftarrow \theta_k$, $p_{k+1} \leftarrow p_k$
    end if
  end for
  return $\{\theta, \Gamma\}$
```

Algorithm 11 Function $\phi_\Gamma(\theta)$

```
  //NB.
  // - $\Gamma(\theta)$ is an indexing function on $\Gamma$
  // - $N_f(\theta)$ := the number of times $f$ has been computed for a given $\theta$-tuple.
Require: $\Gamma$, $\pi$, $f$, $\theta$
  if $N_\phi(\theta) < \Gamma_L$ then
    Evaluate $\hat{f}(\theta) := f(\theta)$.
    Update $w_i$ for $i = \Gamma(\theta)$ according to (3.6)
  end if
  return $\pi(\theta)\ w_i$ (for $i = \Gamma(\theta)$)
```

Symbol	Description
Γ	Hypothetical importance sample.
θ_i	The i^{th} distinct θ-tupple in Γ.
q	MCMC proposal density (or function).
π	Prior density function.
$f(\theta)$	Likelihood function (or target density).
ϕ	Modular component that interfaces f with the MCMC resampler – see Algorithms 10 and 11.
w_i	Weight associated with the i^{th} θ-tupple in Γ and given by $\bar{f}(\theta_i)$.
N_C	Config. property: the number of Markov chains.
N_{MH}	Config. property: the number of MCMC [re]samples.
Γ_S	Config. option: number of samples along each dimension of θ – i.e. Γ as a unit cube.
Γ_I	Config. option: manually specify [approximate] sampling intervals in each dimension – alternative to Γ_S.
Γ_L	Config. option: max number of f computations for any given θ-tupple.

Table 3.1: ARQ-MCMC: commonly used notation

Definition 1: mean likelihood $\bar{f}(\theta)$

Assuming that $f(\theta)$ yields an unbiased estimate $\hat{\pi}(y|\theta)$, we can reduce the variance of the estimate obtained (with respect to a single given θ tuple, and up to a given optional limit) by updating the weight associated with that tuple in Γ. That is, each time it is [re]computed:

$$w_i = \frac{\sum_{j=1}^{N_f} f_j(\theta_i)}{N_f} \tag{3.6}$$

where w_i corresponds to the i^{th} θ-tuple in Γ, and w can thus be regarded as a sparse vector of weights. The total number of times f has been evaluated for the i^{th} θ-tuple is (somewhat lazily[2]) denoted simply as N_f.

Definition 2: approximation of the marginal likelihood

Given Γ, [approximately] *evenly spaced* samples, from the region[s] of the parameter space where $\mathrm{E}[f(\theta)] > 0$ by some appreciable margin, we could produce a piecewise approximation of the integral. That is, if we allow that the MCMC resampling procedure has sufficiently enriched the underlying importance sample, we could approximate $\int_\theta f(\theta)$ using a simple summation by parts, over Γ. This is on the intuitive, but somewhat tentative, basis that the (possibly far larger) regions of the parameter space that have not been sampled, would not have contributed significantly to our knowledge of the integrand in any case.

Given Γ, [approximately] evenly spaced spaced samples from the a region broadly covering the posterior mass $\pi(\theta|y)$, we could likewise normalise the weights to obtain a rough (depending on sparsity of coverage in Γ) approximation of the marginal likelihood:

$$\tilde{\pi}(y) = \frac{\sum_{i\in\Gamma} \pi(\theta_i) w_i}{\sum_{i\in\Gamma} \pi(\theta_i)} \tag{3.7}$$

where w_i is the sample mean of the i^{th} θ-tuple in Γ.

[2]Because superscript i is dropped for convenience, though in actual fact N_f corresponds to each individual tuple, θ_i.

Definition 3: Expectations of a function

Similarly, Γ can be used to approximately estimate the expectations for any θ-valued function, such as $g(\theta)$. Thus:

$$\tilde{E}[g(\theta)] = \frac{\sum_{i\in\Gamma} \pi(\theta_i) w_i g(\theta_i)}{\sum_{i\in\Gamma} \pi(\theta_i) w_i} \tag{3.8}$$

where w_i are again the weights associated with each θ-tuple in Γ. This is somewhat redundant as the MCMC rejection [re]samples that constitute the algorithm's primary output for any given analysis provide an easy way to do the same. However in practice the approach was found to yield more accurate estimates in some cases. That is, when Γ was sufficiently enriched and in particular when one or more Markov chains failed to converge, (3.8) was nonetheless found to provide reasonable approximations of elementary g, with respect to both mean and variance.

Detailed description of algorithm steps

Some abbreviations have been introduced in the description here, with $\Gamma(\theta)$ an indexing function on Γ; and $N_{f(\theta)}$ shorthand for the number of times $f(\theta)$ has been computed already for a given θ-tuple in Γ. A concise [recursive] example is illustrated in Algorithm 10 requiring the quantities defined above. The steps of the algorithm are described in more detail as follows:

1. Let Γ denote an *ordered* QMC sequence construction in dimension d and define w_i as a sparse vector of weights corresponding to each tuple (do not compute any, yet).

2. Initialise (sequentially or in parallel) N_C Markov chains, indexed by j, at tuples sampled uniformly from Γ. Compute the *initial* weights using a particle filter or some other given estimator $f(\theta_i)$, and update w_i, i.e. for the very first step: $j = 1$, $k = 1$. Set $w^{\Gamma(\theta_{1,1})} \longleftarrow\!= \phi(\theta_{1,1})$, where the $\Gamma(\theta)$ is an indexing function on Γ. Ignoring the recommended 'burn-in' period for the sake of notational convenience, the number of iterations (and therefore samples) in each Markov chain is denoted N_{MH}, and indexed by k.

3. For each chain: sample $\theta^{j,q} \sim q(\theta|\theta^{j,k-1})$ from Γ. Note that q is used here to represent a probability mass function associated with a random-walk proposal,

as noted above. If $N_\phi(\theta) < \Gamma_L$ compute $\phi(\theta)$ (e.g. run Algorithm 6) and update w_i according to (3.6). As a single operation this function is denoted $\phi_\Gamma(\theta)$, and described using pseudo-code, as Algorithm 11.

4. Accept the proposed move with Metropolis-Hastings probability:

$$p_{mh} = \frac{\pi(\theta^{j,q}) w^{\Gamma(\theta^{j,q})}}{\pi(\theta^{j,k-1}) w^{\Gamma(\theta^{j,k-1})}} \tag{3.9}$$

   or else set $\theta^{j,k} \leftarrow \theta^{j,k-1}$.

5. Return to step three and repeat until $k = N_C$ (for each chain).

Output

The algorithm yields a weighted importance sample $\{w_i, \Theta^i\}$, and $N_C N_{MH}$ rejection samples from the Markov chains (less the number of samples that are discarded during the 'burn in' period) which are approximately distributed according to the target density (or distribution) – assuming that the chains have converged sufficiently. Convergence is 'confirmed' using the Gelman-Rubin test, but it is important to note that this is only an assessment of how well the chains have converged on the underlying importance sample – not the target distribution itself. Practical experimentation suggests that even where this degree of 'convergence' has not been achieved (perhaps because the target distribution is highly multi-modal) the importance sample often still yields reliable estimates, so long as N_C is sufficiently large to ensure at least some coverage of all modes that contribute significantly to the target distribution.

Estimating the model evidence for Algorithm 10

An approximation scheme based on Definition 2 was partially developed but found to be inadequate for all but the most trivial of problems. The details have therefore been relegated to the appendices (A).

Testing the algorithm

Here we test the ARQ-MCMC algorithm using an example, based on an epidemiological model and a particle filter, and focused on Bayesian analysis with multiple candidate prior distributions. The *SEIR* model (see Figure 3.5), was selected as a canonical example of an epidemiological model and inference problem[3]. Algorithm 6 allows for the easy construction of artificially difficult (i.e. high variance) problems, because the number of particles can be arbitrarily reduced to give a sufficiently 'poor' result. This allowed in turn for features of the *ARQ-MCMC* algorithm such as 'sample limiting' (i.e. the optional algorithm configuration parameter Γ_L) to be conveniently explored without resorting to the use of actually difficult problems.

$$\begin{tikzcd}
\boxed{\text{S}} \arrow[r, "\theta_1\text{SI}"] & \boxed{\text{E}} \arrow[r, "\theta_2\text{E}"] & \boxed{\text{I}} \arrow[r, "\theta_3\text{I}"] & \boxed{\text{R}}
\end{tikzcd}$$

Figure 3.5: The [density-dependent] SEIR model used for this simulation study.

Bayesian analysis with multiple prior distributions

As noted in the introduction, a cautious approach to Bayesian analysis of scientific data requires judicious choice of the prior distribution $\pi(\theta)$. In practice, that may require repeated analysis of the same model and data with different prior distributions, in order to identify the one that optimally encapsulates our prior beliefs; decide which is judged to be the most 'objective' in a given context; or for comparison – to understand how the choice of prior distribution affected the posterior distribution. In this example we show how the [partial] resampling feature of the alogrithm laid out the in previous section can be used to improve the efficiency of multiple quasi-independent MCMC analyses, with different prior distributions – three in this case, as laid out in Table 3.2 – so long as they 'share' the same likelihood function.

It is notionally assumed that selection of the prior distribution for onward analyses is qualitatively informed by the most recent set of results. The software package

[3]Note that for the same reason, to emphasise the generality of the approach, the standard notation of epidemiological models has been dropped in favour of a vector of parameters denoted collectively by θ. For example, the contact rate parameter θ_1 is usually denoted by β.

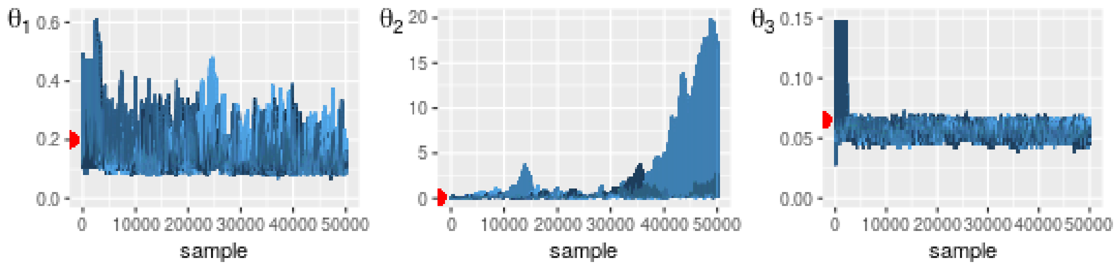
	θ_1	θ_2	θ_3
Analysis 1	Flat and improper		
Analysis 2	U(0, 1)	U(0, 1)	U(0, 1)
Analysis 3	U(0, 1)	Γ(10, 0.07)	U(0.02, 1)

Table 3.2: Prior distribution chosen for each analysis.

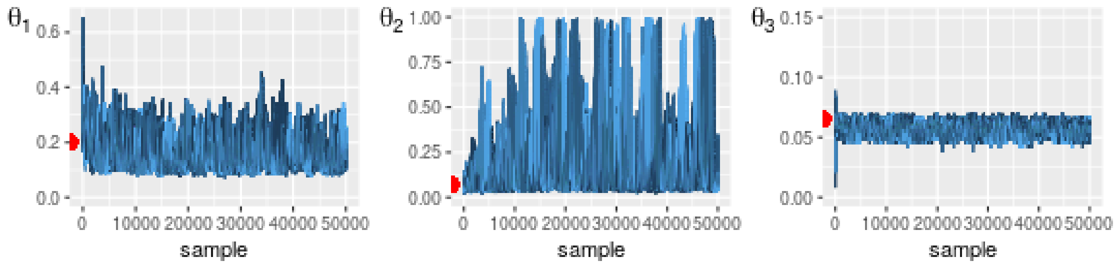
implementation described is organised in this fashion, such that analyses can be 'daisy-chained' in this way if so required (though it is not necessary). The MCMC [re]sample trace plots for each analysis, shown in Figure 3.6, help to illustrate how each analysis was impacted by the choice of prior distribution. A lack of convergence is evident for the first analysis (confirmed by the Gelman-Rubin convergence diagnostic, with results provided in an appendix). That instability was resolved by use of a fairly generically chosen prior distribution, for the second analysis. A somewhat more informative prior distribution was chosen for the third and final analysis. We can imagine here that posterior samples obtained for the second analysis contained implausible values, based on some extraneous knowledge relating to the *average* latent period, given by θ_2^{-1}. We also modify our 'prior' beliefs in that analysis to rule out an average infectious period θ_3^{-1} of more than $50t$, by setting a *lower* bound on the prior distribution of θ_3 of $\frac{1}{50}$.

Inference results: Here we provide an overview of the adapted samples, and the computation expended to obtain them. The posterior samples obtained for each analysis are shown by the joint marginal densities given in Figure 3.7. In conjunction with the traceplots, they help to illustrate how the results of the analyses were impacted by the progressive selection of more informative prior distributions. This is especially true of the contact rate parameter θ_1, which can be seen to converge more on the 'true' (i.e. simulation) value. In truth, the simulated observation data merely represent a [mapping of a] random sample *conditional* on those values (the 'simulation'). In other words, they are only (at best) an approximate guide to the true expectations of the posterior distribution. Thus the benefit –the extent to which the results become 'more accurate'– is largely a result of the subjective approach; using our prior knowledge (or beliefs) to inform the selection of prior distributions. The design of the algorithm merely seeks to facilitate this uniquely Bayesian workflow in as efficient a manner as possible.

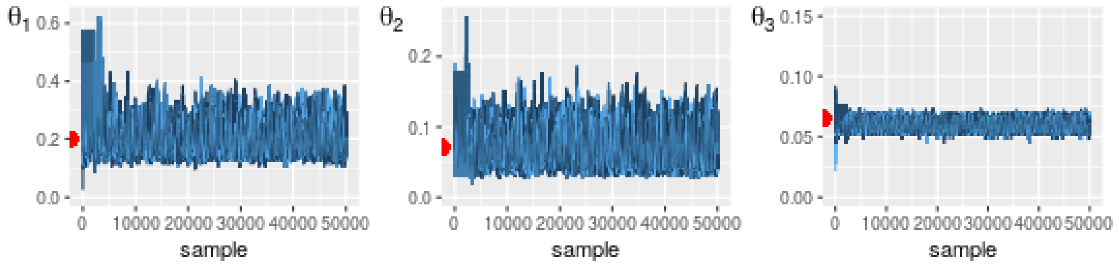
Computation: Finally, the computation plots given in Figure 3.8 illustrate how



(a) Analysis one: flat and improper prior distribution.



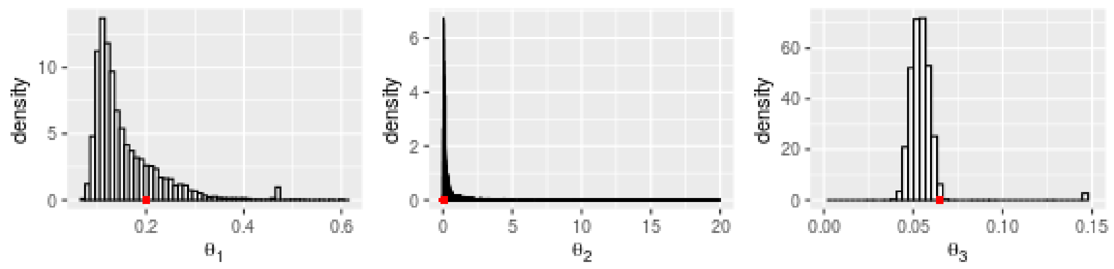
(b) Analysis two: uniform prior distribution.



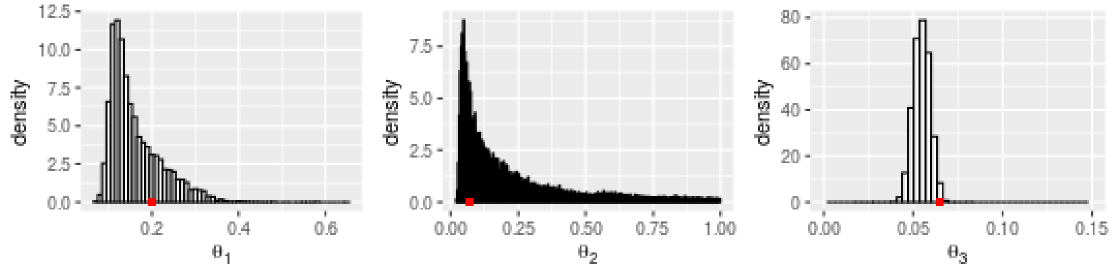
(c) Analysis three: mixed uniform and Gamma prior distribution.

Figure 3.6: MCMC traceplots for the first example. The values used to simulate the underlying 'observational' data have been marked for reference. As indicated by the plots, the first analysis failed to converge due to a lack of identifiability in the second parameter θ_2. The selection of still a not very informative uniform prior (in particular U(0, 1) for θ_2) for the second analysis was sufficient to allow convergence. However we suppose, for the purpose of illustration in the third analysis, that our prior knowledge of the average latent period θ_2^{-1} is such that we can choose a more informative prior distribution for that parameter only, $\Gamma(10, 0.007)$. In order to make the scenario more realistic the latter was parametised such that $\mathrm{E}(\theta_2) = 0.07$ (i.e. slightly different from the true value used in the simulation of 0.065).

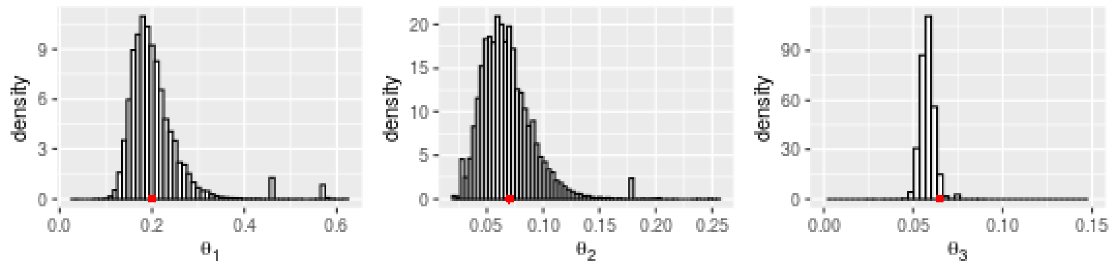
the algorithm became progressively more efficient – i.e. required fewer calls to the



(a) Analysis one: flat and improper prior distribution.



(b) Analysis two: uniform prior distribution.



(c) Analysis three: mixed uniform and Gamma prior distribution.

Figure 3.7: Marginal MCMC [re]sample densities – contact rate $\theta_1 = \beta$; progression (to infectiousness) parameter $\gamma = \theta_2$, etc.

particle filter used to estimate the likelihood function, denoted $f(\theta)$.

Computational impact of sample limiting

It is reasonable to assume that the quantity yielded by (3.6) is asymptotically normal, when is derived from an unbiased estimator (e.g. when estimated by a particle filter). It therefore also makes sense to limit the maximum number of evaluations N_ϕ to some predetermined quantity Γ_L. This automatically 'throttles back' the computation for regions that have already been sufficiently sampled (or computed once where ϕ provides an exact or sufficiently accurate result). The concept illus-

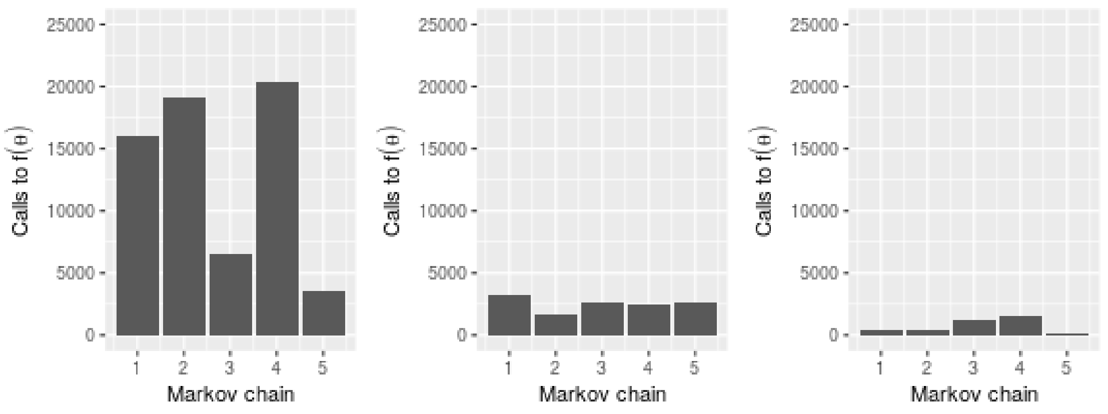
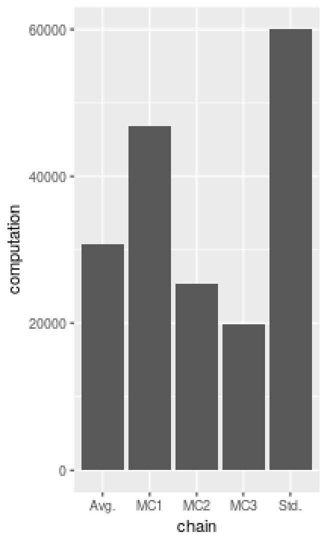


Figure 3.8: Computation required for analysis one, two and three (L-R). Stated in terms of the number of calls to the likelihood function, $f(\theta)$, required. Each analysis consisted of five Markov chains, each yielding 5 x 10^4 resamples.

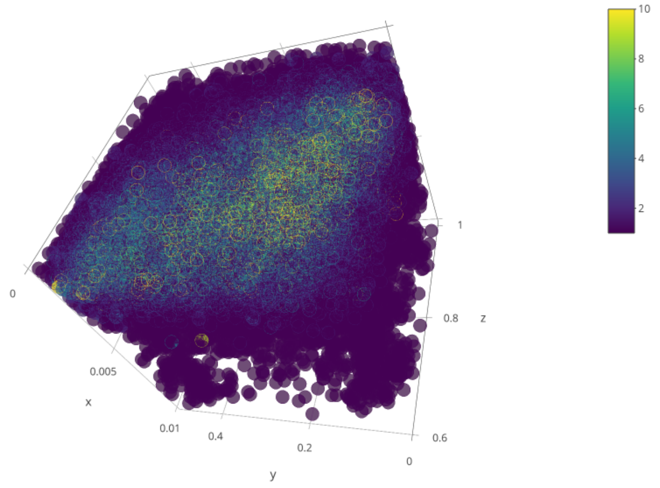
trated using an example in Figure 3.9. Note that the underlying importance sample Γ can be 'recycled' for each chain, such that each iteration (i.e. each Markov chain) requires less computation. However the key benefit of initialising many different [over-dispersed] chains is retained; one is reasonably assured of good 'coverage' of the parameter space, with respect to the posterior mass.

High-variance problems

The 'smoothing' impact of sample limiting is demonstrated (for another example, see Figure 3.2) in Figure 3.10. In this example the number of particles used in the particle filtering procedure was constrained to simulate a high-variance problem in low dimension ($d = 2$). The impact of progressively increasing the sample limit, from one to seven, can be seen in the densities arranged from left to right. At least in this case, that impact is actually *not* in terms of providing a more accurate approximation of the *expectations* (i.e. mean average) – all three are approximately the same in that regard. Instead the algorithm acts to distribute additional computational resources to the periphery of the posterior mass, providing progressively better resolution in the tails.



(a) $f(\theta)$ calls.



(b) Γ exploration ($\Gamma_L = 10$).

Figure 3.9: Computation plots for ARQ MCMC (Algorithm 10). The figure on the LHS illustrates the function calls required for a three chain analysis compared to a standard implementation of PMCMC, Algorithm 7 (parameters $\{\Gamma_S = 50, \Gamma_L = 5\}$). The figure on the RHS is based on the same analysis except that $\Gamma_L = 10$ was used to provide a better visual contrast. It illustrates exploration of the model parameter space and demonstrates the 'throttling' effect of the Γ_L parameter. The brighter, yellow region indicates the bulk of the posterior mass and is visible as a cross section through the regions of lower density.

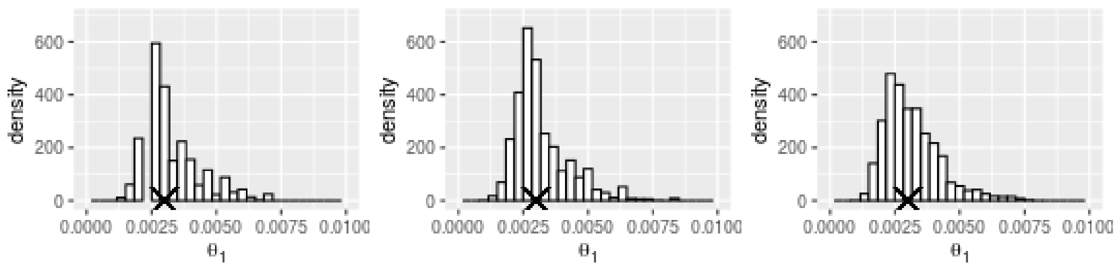


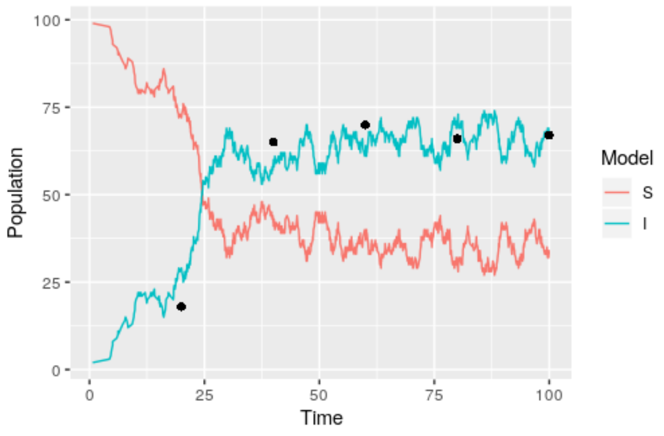
Figure 3.10: Sample output from Algorithm 10 illustrating the impact of (left to right) $\Gamma_L = \{1, 3, 7\}$. The results are marginal distributions for the *contact rate* (i.e. infection) parameter in the model considered in Figure 3.11b, with comparable results given in Figure 3.12. The parametrisation of the algorithm was as follows: $N_{MC} = 5$; $N_{MH} = 50000$; $\Gamma_S = 30$ and Γ_L as noted. The value used in the original simulation [4] has been marked for reference.

3.4 Comparative Performance

The algorithms and implementations presented here were validated by comparison with the method described by [4] (Algorithm 5) – for the same problem and data set described in that paper (and used to demonstrate the *MBP-IBIS* algorithm in the last section). An illustration of that model, and a typical realisation, is given in Figure 3.11. It is similar to the SIR but incorporates autoregulatory dynamics with the result that there is no natural upper bound on the cardinality of the process vector $|x|$ (for a given time period). As a test problem, this is straightforward to set up but computationally challenging. This makes it useful as a test problem, in lieu of the similar but more complex models considered in Chapter 6, for which no published results are directly comparable. Multiple analyses were run for different configurations with comparisons given below, but first sample output for selected analyses are shown in Figure 3.12.

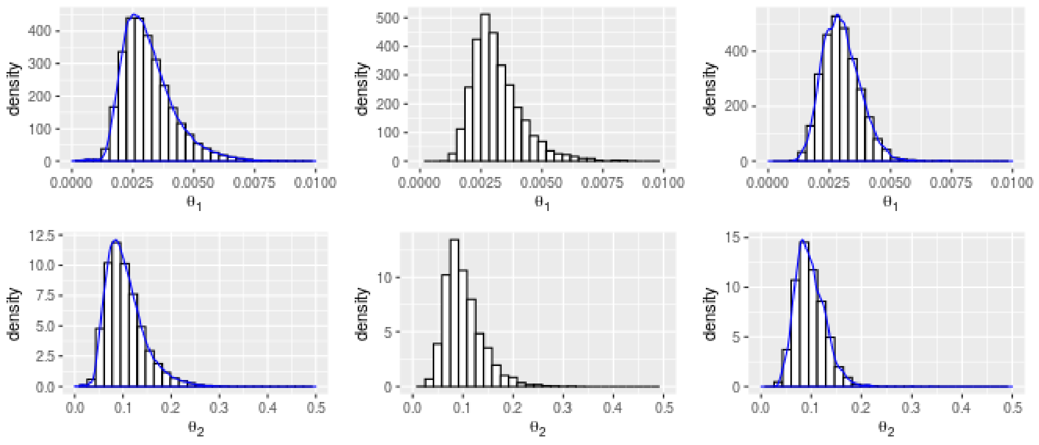
$$\begin{tikzcd} \boxed{S} \arrow[r, shift left, "\theta_1 SI"] & \boxed{I} \arrow[l, shift left, "\theta_2 I"] \end{tikzcd}$$

(a) SIS model



(b) Realised exampled (Algorithm 1)

Figure 3.11: **Benchmark inference problem** used to evaluate algorithm performance. (a) The SIS model is intended to represent the dynamics of infectious diseases which do *not* confer long-lasting immunity. (b) stochastic realisation simulated using Algorithm 1 (see Chapter 2) and the same parameters used by [4] of $\theta = \{\beta := 0.003, \gamma := 0.1\}$. The simulated observations data reported by the authors of that paper are marked for comparison.



(a) MBP MCMC (b) ARQ MCMC (c) MBP-IBIS

Figure 3.12: **Cross-algorithm validation**. Samples results for Algorithm 10 and Algorithm 9, compared with results based on the methods and results of [4] for comparison and validation. The parametrisation used for the latter (essentially Algorithm 4) was $\{N_{MC} = 10;\ N_{MH} = 50000\}$. The first $\frac{1}{5}$ of samples were discarded. The ARQ MCMC parameters were $\{N_{MC} = 10;\ N_{MH} = 50000\}$ with the same 'burn-in' period as the first.

3.4.1 Performance evaluation

A straightforward and informal comparison of performance was made, based on empirical analysis of the algorithms' results for the aforementioned problem – i.e. the methods, model and dataset from [4]. The SMC^2 method (Algorithm 8) was used to provide an additional baseline for comparison, and in fact broadly outperformed both MBP-IBIS and ARQ-MCMC according to the predetermined performance measure; the average normalised error for the posterior expectation values of the model parameters, $E(\theta)$. The true value was estimated based on one thousand runs of the comparatively straightforward *MBP-MCMC* algorithm, repeated twice to ensure the consistency of the result. Those results are illustrated in Figure 3.13.

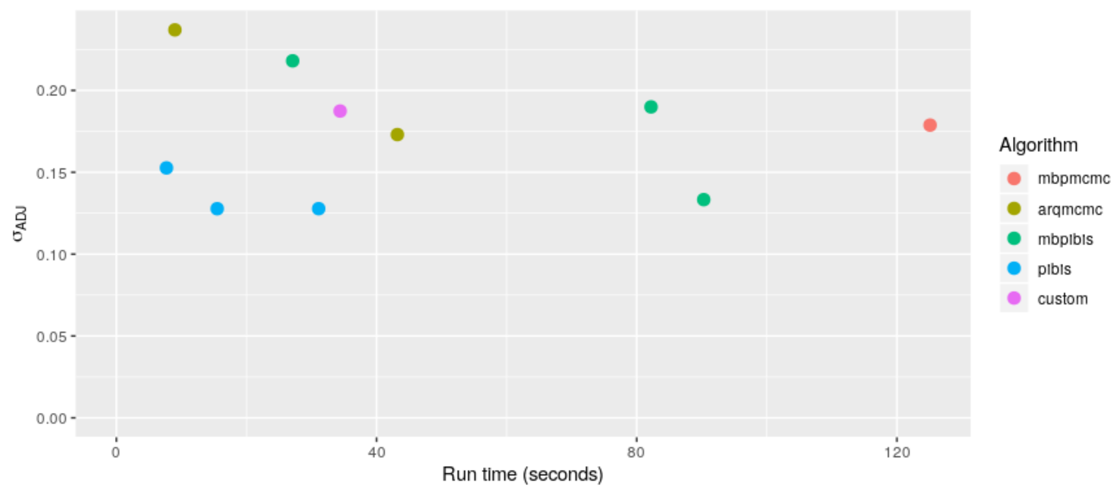


Figure 3.13: **Speed test**. Comparison of algorithm run time (in seconds) and the mean adjusted error for estimates of $E(\theta)$ for the problem illustrated in Figure 3.11b.

It is important to acknowledge that this analysis was limited in scope, as only a single narrow class of problem has been considered. Furthermore, comparisons that rely on benchmarking unavoidably incorporate extraneous factors, such as hardware and language implementation. In terms of overall speed and accuracy the experimental results suggested that SMC^2 (Algorithm 8, labelled as *'pibis'* in Figure 3.13) was the most efficient for the simple problem considered; namely estimating $E(\theta)$. This also proved to be the case for the results presented in Chapter 6. However, the variance reduction features of Algorithm 10 also proved to be useful for applications in Chapter 6 and those features are not clearly revealed by a focus on the posterior expectation alone. In slightly informal terms, the algorithm's design allows for good resolution in the tails of target distribution, leading to smoother marginal sample densities.

3.5 Software implementation

Both algorithms are implemented (and published) in the software package introduced in Chapter 5 for the Julia programming language.

3.6 Discussion

By applying the principles of modular construction [11] to algorithm development, we have utilised a number of distinct methodological concepts in the construction of two new algorithms for Bayesian inference: MBP-IBIS and ARQ-MCMC. These were tested on applications chosen to demonstrate key features and potential, as well as being compared with each other and established algorithms in terms of accuracy and speed. Both algorithms are applied to inference problems in Chapters 5 and 6. They are also jointly compatible with the Bayesian inference modular workflow introduced in the next chapter having been developed in conjunction with that work. We conclude this chapter by discussing each algorithm in the context of the available computational tools for Bayesian inference, highlighting their relative strengths, weaknesses and potential for further development.

3.6.1 MBP-IBIS

The MBP-IBIS algorithm incorporates two highly efficient but somewhat different approaches to inference that have been recently devised for application to similar types of model. Intuitively, such a proposition lends itself to a tantalising prospect; 'the best of both worlds'. However while the algorithm yields an inexpensive estimate of the marginal likelihood, that optimistic view overlooks the sacrifice of 'online' computational efficiency necessary to implement data-augmented schemes. The overhead related to the latter perhaps explains why the SMC^2 was found to outperform this and all other algorithms for the (admittedly narrow) evaluation reported above. However it is known that SMC techniques are ineffective for certain problems, and we can thus speculate that there may be some for which the

MBP-IBIS algorithm *may* outperform it. In (slightly enthusiastic) layman's terms, perhaps for 'tricky' double-integrals, where solving the inner integral is difficult with standard SMC techniques, but where the same does not apply to the outer one. As it was, the SIS-model test problem [4] was selected 'blind' on the basis that it is conceptually simple but [conveniently scalable to be] computationally challenging.

3.6.2 ARQ-MCMC

The second algorithm, ARQ-MCMC addresses the problem that motivates 'waste recycling' techniques (see §2.4.5) by combining adaptive particle-MCMC with randomised, quasi-Monte Carlo (QMC) and is more generally applicable than MBP-IBIS. Despite its generality, the algorithm described here was developed specifically for use in conjunction with an 'inner' particle filtering procedure, in order to leverage the latter algorithm's efficiency as an 'online' algorithm for certain types of inference problem. It was found to perform reasonably well against other algorithms –and outperform standard particle MCMC by some margin– for low dimensional problems, particularly for problems where the posterior mass had low variance relative to the sample interval.

In general, its most useful feature was found to be the scalability of its variance-reduction potential. In plainer terms, it provides a way to focus computational resources more evenly, and out to the 'tails' of the target distribution, which led to 'smooth' [marginal density] estimates – even though it is not particularly efficient for the scale and dimension of model ultimately required for the motivating problem considered in Chapter 6. In summary, ARQ-MCMC proved useful (and reliably consistent) as a validation tool, and also provided the first meaningful results obtained during the course of the project that were based on actual BTB surveillance data.

We conclude this chapter by discussing some of the algorithm's features in more detail, and highlight ideas for future work:

Computing the marginal likelihood

A simple method for approximating the marginal likelihood denoted by convention as $\pi(y)$, was presented in §2.5. Despite the crudeness of the approach, it was found to give results surprisingly close to that of the SMC^2 algorithm for very low dimensional problems ($d < 4$). That approximation might have been improved for other problems with a modest amount of effort. For example, it was noted that MCMC tends to (naturally) weight the importance sample such that remote regions of the target distribution are neglected (except perhaps for the initial 'burn-in' samples). Even sparse stratified sampling of these remote (but non-zero) density regions, combined with a more sophisticated and finely-grained (i.e. integration by parts) approach would likely have extended the usefulness of the algorithm in this regard, and with perhaps only a very modest amount of additional computational cost. However, this remains the subject of future work. In particular the SMC^2 provided a more efficient means of estimating (rather than approximating) the model evidence, such that any additional effort expended on this feature would not have contributed directly to the goals of the project.

Adaptive [harmonic] intervals

As noted above, the algorithm and implementation was found to be a generally useful 'workhorse' throughout the course of this project. More precisely, while the suitability and effectiveness of approaches (i.e. algorithms) for conducting Bayesian inference often seem to depend largely on the nature of the problem, the *ARQ-MCMC* algorithm tended to produce reliable results regardless – though its run-time was still affected by the nature and difficulty of the underlying problem. One obvious exception to this surprisingly general usefulness lay in the necessity to pre-determine the [approximate] interval width between samples. This design decision, or requirement, seemed entirely justified in the early stages of development. In many conceivable circumstances, the decision can be wholly informed by *a priori* knowledge of the problem and parameter space, and the desired resolution in terms of the marginal estimates obtained. For example, given a parameter that represents time duration (or a linear transformation thereof) we would usually be able to grasp these details from our understanding of the phenomena being modelled. In the context of a chemical reaction taking a few minutes, one-second intervals, may seem naturally appropriate. For the initial infection time of the in an epidemic lasting many months, days (or longer) would probably seem to be intuitively more appropriate for most people.

Nonetheless the 'desirable resolution' of estimates (i.e. optimal choice of interval) could more simply (at least, from the user's perspective, especially given limited *a priori* knowledge) be stated in terms of the *variance* of the estimates. That would allow for intervals to be adaptively rescaled based on samples obtained up to a given point. In other words, the 'unit cube' could be adaptively rescaled, such that the posterior mass is approximately concentrated in a cube containing n^d sample nodes. This could naively be accomplished by simply stopping the algorithm intermittently and rescaling the intervals accordingly – effectively, by running the algorithm twice; with intervals used for the second run scaled according to the ratio between the variance of samples obtained by the first run and the initially chosen interval widths. However that would require that the samples obtained in the first run be discarded, and fit poorly in a framework which is largely focused on minimising waste (w.r.t. computation).

It would therefore be more fitting to construct an adaptive rescaling technique so as to encourage harmonic resonances between the initial QMC sequence (largely defined by the interval width), and the next [i.e. adapted] one. In that way, at least some of the computation expended during the 'first' (notional) run could be reused. As a plain illustrative example, consider a single parameter in which the interval was either doubled or halved, according to some adaptive rule. In the first case, half of the samples would still be centred on a valid node, and thus could be reused. In the latter, all of the samples would be centred on a valid node, and the MCMC resampler would silently 'fill in the gaps', in accordance with its normal operation. This could enhance the general usefulness of the algorithm because it would require only an initial guess on the user's part as to the appropriate sample width. It would also provide an automated mechanism to further reduce the *effective dimension* of the underlying problem – and perhaps make the algorithm suitable for problems in higher dimension than was originally intended.

Alternative rules for sample limiting

We have addressed the sample limiting feature of the algorithm, and proposed that it may be useful for circumstances where the target integrand is an estimator with only minimal accuracy, or equivalently, high variance. However we have only addressed one form of limiting rule – the one implemented within *DiscretePOMP.jl*. That rule is intended to ensure that samples are computed, up to the predefined limit as soon as possible, as opposed to say, every n^{th} time a given θ-tuple is sampled from Γ.

In particular, we experimented with one minor adaptive rule change which proved to be useful in some situations. Crudely speaking, if it is detected that a Markov chain is 'stuck', i.e. it has lingered on a given sample for an unusually long period, we may wish to spend one additional function call to ensure that it is not aberrant behaviour in the algorithm, likely induced by 'locking in' an unusually high sample error for a given θ-tuple. Likewise, if a given tuple is rejected n times (without being accepted even once) we might reasonably conclude that it is located in a dense posterior region (to have been visited n times) and that it is worth an additional function call to insure against essentially the same scenario in reverse.

We also note that another option for controlling the flow of computational resources is the *targeted acceptance rate*, implicitly used to adaptively regulate proposal 'distance'. That is, a lower targeted rate will allow that remote [w.r.t. probability density] posterior regions are probed more, and more often. We contend that combining the targeted acceptance rate with different varieties of sample limiting rule, provides a flexible and intuitive tool set for users with specific computational requirements. For example, those who wish to estimate the posterior expectations of a function $g(\theta)$, whose value is only significant for θ corresponding to remote regions of the posterior distribution – sometimes referred to in practical settings simply as 'tail-risk'. In such a case, we would probably wish to spread computation evenly, in a wide envelope around the posterior mass, rather concentrate it on the densest regions.

We could envisage a different scenario based on the particle filter example described. As already noted, that method is an 'online' and thus highly efficient and scalable in terms of accuracy. However if the number of 'particles' were limited, say by the amount of physical memory available for a particularly complex problem, that constraint could conceivably be overcome in the manner described here.

Alternatively, deliberately choosing an artificially low number of particles could serve to [further] *concentrate* computation on the posterior mass, in combination with an appropriately chosen limiting rule. In fact this would emulate, somewhat, another technique for improving accuracy in MCMC, as considered in [56] for similar [i.e. population-]models, and known as *'delayed acceptance'*. This is only applicable when an alternative 'cheaper' means of evaluating the acceptance probability is available. In contrast to the approach used here, in delayed-acceptance MCMC those preliminary means are used to estimate the acceptance probability, with a full evaluation computed only when it is deemed worthy (i.e. reasonably likely to be accepted).

3.6.3 Conclusion

We conclude by noting that a ‘good’ algorithm in practice is one that best fits the user’s specific need. Outwith spectacularly efficient (e.g. analytical) solutions, this may be largely a matter of directing finite computational resources in the best possible way, according to that need. We hope that the methods and open-source implementations we have described here can be of use towards that end, and be improved upon in due course (in the ways suggested herein or otherwise).

Chapter 4

Implementing Bayesian workflow for DPOMP models

"The idea of modular construction goes against a long-term tradition in the statistical literature where whole models were given names and a new name was given every time a slight change to an existing model was proposed. Naming model modules rather than whole models makes it easier to see connections between seemingly different models and adapt them to the specific requirements of the given analysis project."

- Gelman *et al.,* [11]

Summary

- Discrete-state-space Partially Observed Markov Processes (DPOMP models) are a notable subclass of mathematical and statistical models, often referred to by the more general term 'compartmental' models in applied settings such as epidemiology. The Susceptible-Infectious-Recovered, SIR, model is a well known example.

- Bayesian statistics provides an extensible framework for parameter and model inference (or 'model comparison') well suited to scientific domain models (including applications of DPOMPs). For example, allowing, in addition to the primary data used for the analysis, incorporation of findings, beliefs and commonly accepted scientific knowledge about the system under study as the 'prior distribution'. The collection of data itself can be characterised using a

statistical *observation model* allowing for arbitrary complexity and intrinsic uncertainty.

- Recent work towards an integrated, principled Bayesian workflow [11, 67, 68] provides a rigorous framework for full Bayesian analysis of scientific data, including parameter and model inference. Tools such as **Stan** facilitate convenient implementation for *some* classes of models, notably those based on ordinary differential equations (ODEs).

- However implementing a principled Bayesian workflow for other, more complicated classes of scientific domain model is challenging for a variety of reasons. In the case of DPOMPs specifically, estimating the posterior density involves integrating over the parameter and measure space of the underlying stochastic process model (i.e. solving a 'double integral') and can be computationally expensive. The required methods can also be work intensive to implement due to their complexity, even in feature rich languages such as **R** and **Julia**.

- Here we collate algorithms and tools towards the implementation of a scientifically rigorous, computationally efficient (and eventually, automated) Bayesian workflow for the generic class of DPOMP model described herein.

- To enable this we introduce two novel inference algorithms aimed at overcoming common difficulties encountered in application of Bayesian inference for DPOMPs.

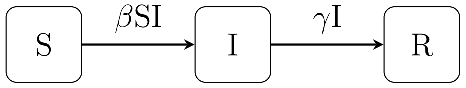
4.1 Introduction

This chapter first provides brief overviews of Discrete-state-space Partially Observed Markov Process (DPOMP) models, and Bayesian inference in the context of DPOMPs, before moving on to the recent topic of principled workflows for Bayesian data analysis. Having laid out these essential concepts, the rest of the chapter is given over to describing the algorithms and other statistical tools necessary for implementation of a multi-phased workflow for Bayesian analysis of DPOMP-modelled data. In particular, we introduce two novel inference algorithms that aid application of Bayesian inference for DPOMPs and showcase them via exemplar Bayesian workflows.

Discrete-state-space Partially Observed Markov Processes

Discrete-state-space **Partially Observed**, **Markov Processes** (DPOMPs) are an important subclass of state-space-model known as 'compartmental' models. The canonical example of which from epidemiology, the susceptible-infectious-recovered, or SIR model, is depicted in Figure 4.1. For DPOMPs, the model is said to have a **discrete-state-space** because individuals take one of n discrete states with stochastic transition events governing their time evolution. In other words (unlike an ODE) the number of *individuals* in a given state at a given time is both a *discrete* quantity and a random variable. Equivalently, the complete system state at any given time can be described by a vector of *integers* e.g. describing the numbers of individuals in each state.

A standard way to construct such models involves a coupled Poisson point processes to represent [the distribution of event times of] X. These are typically distributed according to rate parameters that depend only on the system state at a given time and not its prior history. Another way of saying this is that the random variable X is a **Markovian** [stochastic] continuous time **process**.



$$\begin{tikzcd}\boxed{\text{S}} \arrow[r, "\beta\text{SI}"] & \boxed{\text{I}} \arrow[r, "\gamma\text{I}"] & \boxed{\text{R}}\end{tikzcd}$$

Figure 4.1: **Exemplar compartmental model.** This figure depicts the susceptible-infectious-recovered, or SIR, model [3], in which individuals may only exist in three states: susceptible to the disease S; infectious I; and recovered (and no longer infectious or susceptible) R. For this population-level representation of the SIR model rates are defined in terms of the state-space vector describing the numbers of individuals in each state $\{S, I, R\}$, at a given time, that can be interpreted as probabilistic or deterministic. They govern time-evolution of the system by describing the rate at which individuals from one state transition into other states. Permitted transitions are shown by arrows between states, showing e.g. that the R state is a terminal state and no backward transitions are allowed.

Finally for DPOMPs the system is also said to be **partially observed**. More precisely, the system state as it is [partially] observed and the 'true' underlying (or modelled) system state are treated as distinct but one-way dependent random variables:

$$X|(\Theta = \theta) \sim f_\theta(x) \tag{4.1}$$

$$Y|(X = x, \Theta = \theta) \sim g_\theta(y|x) \tag{4.2}$$

where X represents the complete time evolution of the system and Y the corresponding observations data respectively. They are also sometimes referred to as the *signal process* and the *observation process*. They are distributed according to probability densities functionally denoted as $f_\theta(x)$ and $g_\theta(y|x)$, where θ denotes a vector of model parameters. Typically different components of θ determine the signal and observation processes.

Bayesian inference

In the context of DPOMP models and many others besides, the term *'Bayesian inference'* principally refers to the formulation and computation of probability densities of the form:

$$\pi(\theta, x|y) = \frac{\pi(\theta)\pi(x|\theta)\pi(y|x,\theta)}{\pi(y)} \tag{4.3}$$

where $\pi(\theta, x|y)$ is the *posterior* distribution; $\pi(\theta)$ is the *prior* distribution; and $\pi(y)$ is the *marginal likelihood*. As already effectively noted, the likelihood of the signal process $\pi(x|\theta)$ and observation [model] likelihood $\pi(y|x, \theta)$ are given by:

$$\pi(x|\theta) = f_\theta(x) \tag{4.4}$$

$$\pi(y|x, \theta) = g_\theta(y|x) \tag{4.5}$$

These identities and description of Bayesian inference tacitly assume that we are concerned with only a single model. However they can also be extended for the purpose of model comparison. This is covered in §4.5.

Workflows for Bayesian data analysis

As noted *Bayesian data analysis* includes but is not limited to Bayesian inference. It also includes (initial) model choice, construction and validation. Recent work towards a principled workflow for Bayesian data analysis [11, 67, 68] provides a rigorous framework for scientific research based on Bayesian methods, principally studies that depend on parameter and model inference. This framework can be easily understood by considering the structure of the workflow described by Gelman *et al.,* [11], which consists of three broad phases:

1. Model construction
2. Bayesian inference
3. Model checking and validation.

For simplicity this workflow can be considered initially in the context of parameter inference based on a single model, with formal model comparison added as a fourth stage when needed, or indeed as an extension of phase 3. Although a form of Bayesian inference in its own right, model inference is also a natural form of model validation in that it provides formal metrics that allow us to understand how well (or badly) each competing model fits the data.

Key benefits of Bayesian workflows for scientific research derive firstly in the intrinsic transparency that results from utilisation of a well defined workflow. This facilitates more convenient reproduction of analyses, especially where standardised software tools are available and (well organised) code, and ideally data, is shared openly. Perhaps most importantly, structuring studies in ways that are broadly consistent and standardised where possible, reduces the cognitive burden of attempting to understand a scientific study for readers. In this case, formal convergence diagnostics and other well-established metrics such as 'Bayes factors' can (loosely speaking) play the role that Normality tests and p-values do in Frequentist statistics. That is as commonly understood markers of statistical strength and quality, that aid transparency and rigour.

Implementing Bayesian workflow for DPOMP models

Bayesian workflow are more easily realised through use of standardised software implementations when feasible. Tools such as **Stan**[1] facilitate the convenient implementation of integrated workflows for Bayesian data analysis for *some* classes of models, such as those based on ordinary differential equations (ODEs). However, due to their relatively complexity, DPOMP models are harder to analyse using standardised software tools. More precisely, the underlying stochastic process model gives rise to a nested integral that increases the computational complexity

[1]https://mc-stan.org/

of the problem. This places them somewhat beyond the typical capabilities of software packages designed for standard Bayesian computation, and can also lead to computational difficulties. The main contribution of this chapter is therefore the development of computational statistical tools that overcome such problems while issues of software implementation addressed in Chapter 5.

The rest of this chapter deals directly with implementation of Bayesian workflow and challenges incumbent to the case of DPOMP models detailing each of the three phases: 1) model construction; 2) Bayesian parameter inference; and 3) model checking and validation. These are discussed without regard to any specific realisation of a DPOMP model, however, an example workflow based on analysis of real epidemiological data is provided in the next chapter.

4.2 Phase one: model construction

As noted by [11], initial selection of models and methods is usually guided by pre-existing approaches in the relevant literature e.g. a mathematical model from a published case study. Initially selected models can then be refined to accommodate additional (or fewer) features and complexity, as guided by information obtained from Bayesian parameter inference (phase 2) and model checking and validation (phase 3). This section addresses only the immediate technical considerations and steps involved in defining a DPOMP model. A more complete and generalised treatment of model formulation in the context of principled Bayesian workflows is given by [67].

We begin by noting that the generic class of DPOMP model covered herein is completely described by the following:

- A state vector: specifying the number of individuals in each of N possible states/compartments individuals can occupy within the model, e.g. $\{S, I, R\}$.
- An initial condition: in simple cases, this is the model's state vector at some designated (possibly unknown) initial time t_0 (Other cases discussed below).
- A set Ξ of $|\Xi|$ possible event types
- Transition matrix: a mapping between event types and the resulting changes in state-vector. Typically, it can be represented as an $|\Xi|$ by N matrix.

- The event rate function: a map between the model parameters and state vector, and the rates at which the $|\Xi|$ possible events can occur.
- Observation model: statistical distribution of the observed data conditional on the underlying (not directly observable) system state.

As per the identities laid out in the introduction, *partially observed Markov processes* can be interpreted as statistical models comprised of two random variables, X, Y with the latter conditionally distributed on the former. In most applicable cases the observations data y are longitudinal. In other words, observations are assumed to be made at different points in the temporal evolution of the focal real-world system. It is therefore often convenient to partition x, hereafter referred to as the system *trajectory* [variable,] on the same temporal basis as y. Hence, we have:

$$\pi(x_{1:n}|\theta) = \prod_{1:n}^{i} f_\theta(x_i|x_{i-1}) \tag{4.6}$$

$$\pi(y_{1:n}|x_{1:n}, \theta) = \prod_{1:n}^{i} g_\theta(y_i|x_i) \tag{4.7}$$

Here $f_\theta(x_i|x_{i-1})$ is the distribution under the model of the state of the system x_i at time t_i conditional on it being x_{i-1} at time t_{i-1}.Assuming access to a suitable simulation protocol (the Doob-Gillespie algorithm in this case) we can sample $f_\theta(x_i)$ on a sequential basis, conditional on x_{i-1}. This is particularly useful for the purpose of inference (as discussed in the previous chapter w.r.t. SMC methods).

4.2.1 Initial condition

The above notation implies dependence on initial conditions; initial model state x_0 and corresponding time t_0. Dependent on the available data and assumptions the modeller is willing to make, these may be estimated via Bayesian inference (phase 2). In some cases, it may be appropriate to designate an initial time, e.g. the first observation, and sample the initial state of the model from a given distribution. In others it may seem reasonable to do the reverse and assume that x_0 is known but that t_0 is not. For example, we might assume that an epidemic began with the introduction of one infected individual, but that the time of that introduction t_0 is unknown. For simplicity and notational convenience we disregard the specifics and assume that an appropriate distribution for sampling the initial condition (and computing the corresponding probability density) has been defined. This is written as:

$$X_0|(\Theta = \theta) \sim f_\theta(x_0) \tag{4.8}$$

with the corresponding probability density written as:

$$\pi(x_0|\theta) = f_\theta(x_0) \tag{4.9}$$

Note that any possibly unknown parametrisation of $f_\theta(x_0)$ can be accounted for by introducing an element to the vector of the model parameters θ. Likewise, the initial time t_0 can be similarly accounted for.

4.2.2 Event rates

Given an initial condition (x_0, t_0), time-evolution of the modelled system is governed thereafter by a set of *coupled* inhomogenous Poisson point process (IHPP) with the number of processes corresponding to the number of event types. Inter-event times are exponentially distributed with the rate set by the sum of all event rates. Each event type occurs at a rate which varies according to the state of the system updated by the successively selected events. The corresponding probability density is written as:

$$p(\Delta t = \delta t) = Re^{-R\delta t} \tag{4.10}$$

$$R = \sum_{\xi \in \Xi} \lambda_\xi \tag{4.11}$$

where each rate λ_ξ corresponds to a given event type $\xi \in \Xi$ and Ξ is the set of all possible event types. R is the total rate for all event types combined. When computing these quantities in practice, each λ may be regarded as a separate function that maps (i.e. computes) the rate for the corresponding event type based on model parameters θ and the 'current' (or immediately preceding) state of the model. This simple but extensible framework allows for arbitrary numbers of event types with possibly complicated definitions.

Transition matrix

The simplest way to describe event-driven migration within a model is by using a $|\Xi|$ by N matrix where the cardinality of Ξ gives the number of possible event types and N is the length of the model's discrete state vector. More complex mechanics (such as sampling a new state vector from a given distribution based on the event type) are not addressed here but could nevertheless be easily accommodated within generalised descriptions of DPOMP models by interpreting this step as a [possibly stochastic] function for computing [sampling] new model states.

4.2.3 Observation model

For the purpose of implementing a Bayesian workflow for DPOMPs, the observation model serves two vital purposes:

1. enabling computation of the probability density associated with a given observation y_i: $\pi(y_i|x_i, \theta) = g_\theta(y_i|x_i)$ and also given x_i.
2. allowing sampling of theoretical observation values given some notional system state, x: $Y|(X = x, \Theta = \theta) \sim g_\theta(y|x)$.

Thus the observation model is a statistical distribution from which we wish to both draw samples and compute the probability density associated with given values (and notional system states). As a simple example, we might assume that individuals with a given state S have probability p of being observed. In other words:

$$\pi(y_i|x_i, \theta) = g_\theta(y_i|x_i) \sim Bin(N_S, p) \tag{4.12}$$

where N_S is the number of individuals in state S at the time that observation y_i is taken. Drawing samples in this way is invoked for the purposes of 'predictive checking' as discussed in due course.

4.3 Phase two: Bayesian inference

The second phase of the workflow is Bayesian inference i.e., the formulation and computation of posterior probability densities, stated again here for convenience as:

$$\pi(\theta, x|y) = \frac{\pi(\theta)\pi(x|\theta)\pi(y|x, \theta)}{\pi(y)} \propto \pi(\theta)\pi(x|\theta)\pi(y|x, \theta) \tag{4.13}$$

where the terms and notation are the same as that laid out in §2.3. Note the expression on the RHS (where $\propto$ means 'proportional to') which can be exploited for the purposes of parameter inference without the need to compute the marginal likelihood $\pi(y)$ (algorithms for inference are discussed in §2.4). Here we briefly discuss the prior distribution $\pi(\theta)$, before discussing algorithms for sampling from the posterior. Note that formulation of the observation and process models was discussed under phase one.

4.3.1 Adding prior information

The prior distribution $\pi(\theta)$ provides a formal approach to incorporating into the analysis previous findings, beliefs and commonly accepted scientific knowledge about the system of interest. The selection of an appropriate prior distribution is therefore an important step in any Bayesian analysis. Much like the choice of model structure, selection of an appropriate prior distribution can be an iterative and multi-phased process. One approach is to begin with only a weakly informative prior and progress to a more informative one as our understanding of the problem develops.

4.3.2 Sampling the posterior: algorithm design

As discussed in Chapter 2, directly computing (4.13) tends to infeasible for DPOMPs. A standard approach to Bayesian computation generally is to use random sampling (or 'Monte Carlo') methods to estimate the posterior distribution instead. This is also referred to (in this context) as 'numerical integration'. There are a number of important design choices and options to consider when constructing appropriate algorithms. Firstly, most standard approaches to such pseudo random sampling fall into one of two classes: rejection sampling algorithms, and importance sampling algorithms. Another practical but equally important design choice is mode of computation. 'Online' algorithms are those in which the full system trajectory is not held in memory, greatly reducing memory requirements and increasing speed of computation, at least per sample. In contrast 'offline' algorithms, retain the full system history, which whilst slower, is less restrictive in terms of algorithm design. The additional computational cost can more than offset the cost of 'offline' computation if the enhanced features of the algorithm enable more efficient sampling of the posterior e.g. in generating less correlated samples.

The four specific algorithms for inference that we consider here are selected to give an even coverage of this variety, albeit one that is far from comprehensive. Each pairwise combination of importance/rejection sampling and online/offline computational mode is represented. The first two are established methods from the literature (also described in Chapter 2). They are SMC^2 (an online importance sampler) and model-based-proposal (MBP-) MCMC (an offline rejection sampler). The second two algorithms are the ones introduced in Chapter 3, and were constructed by recombining certain methods and concepts described in Chapter 2, as summarised in Figure 4.2. Much like SMC^2, the MBP-IBIS (offline IS) algorithm can be understood as a version of a technique called iterative-batch-importance sampling (IBIS) proposed by [44]. The ARQ-MCMC (online RS) can be understood as a version of standard [adaptive] particle-MCMC combined with [randomised] quasi-Monte Carlo (QMC). The algorithms and their features are summarised in Table 4.1.

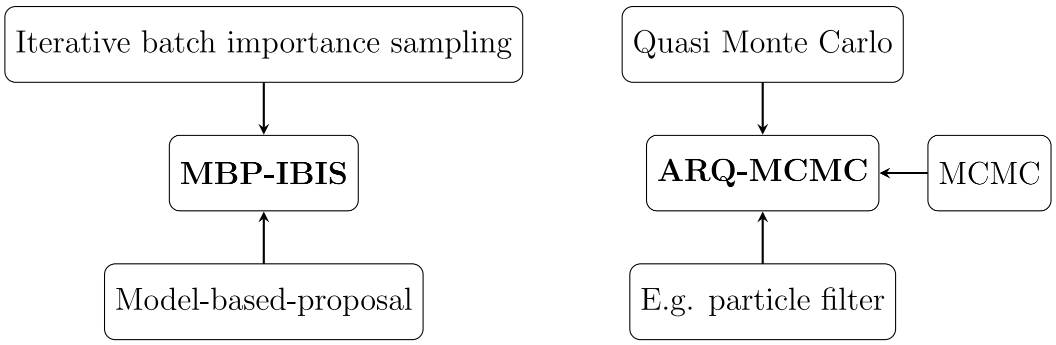


Figure 4.2: **Algorithm dependency map**. The diagram illustrates the key methodological concepts that provide the basis for the two novel algorithms introduced in Chapter 3 and employed in this chapter to implement a practical Bayesian workflow for DPOMP models. An arrow from a to b indicates that a provides a modular component of, or methodological basis for, algorithm b.

4.3.3 Sampling the posterior: algorithm performance

Understanding which of the four algorithms shown in Table 4.1 is 'best' for use with DPOMP models is difficult and it seems likely to be problem dependent. To begin to address such questions, an informal performance evaluation was conducted, based on a simulated inference problem described in [4], with the results reported in the last chapter (§3.4). A visual summary of the results is recapped in Figure 4.3 for the reader's convenience. Recall that the best performing algorithms tending

Algorithm	Mode	Class	Type	Chapter 2
MBP-MCMC	Offline	DA-MCMC	Rejection sampler	§2.4.2
SMC^2	Online	IBIS	Importance sampler	§2.4.3
MBP-IBIS	Offline	IBIS	Importance sampler	
ARQ-MCMC	Online	P-MCMC	Rejection sampler	

Table 4.1: **Algorithm features**. Overview of the inference methods considered here, in terms of characterisation as on- or off-line, and importance or rejection sampling.

towards the bottom left (more accurate / faster) and the worst towards the top right (less accurate / slower). It is important to stress that these results are based on only a single simple inference problem, albeit one chosen to represent a computationally challenge scenario. They are therefore not necessarily representative of the algorithms' capabilities more generally. In particular, the ARQ-MCMC algorithm is particularly well suited to problems in low dimension (with $|\theta| = 2$ in this case). Notwithstanding these caveats, the SMC^2 algorithm (labelled as 'pibis' for particle-IBIS in the figure) was the best performing of all algorithms.

All four algorithms were evaluated based on the same generalised implementation presented in the next chapter. Generalised implementation imposes software design constraints to enable handling of many (at least slightly) distinctive problems and scenarios in order to be useful at scale. To assess the performance impact of such constraints, a customised implementation of the MBP-MCMC algorithm was therefore included (labelled 'custom'). Although it had the same notional configuration and was comparable in terms of accuracy, the customised implementation had an average recorded run time of less than 1/3 that of the generalised implementation.

4.3.4 Sampling the posterior: multi-algorithm workflows

While some (or all) of the named algorithms may be particularly effective for certain classes of problem, it is not necessary, or even perhaps advisable, to use only a single algorithm to obtain estimates.

Here we argue that given all methods have different strengths and weaknesses,

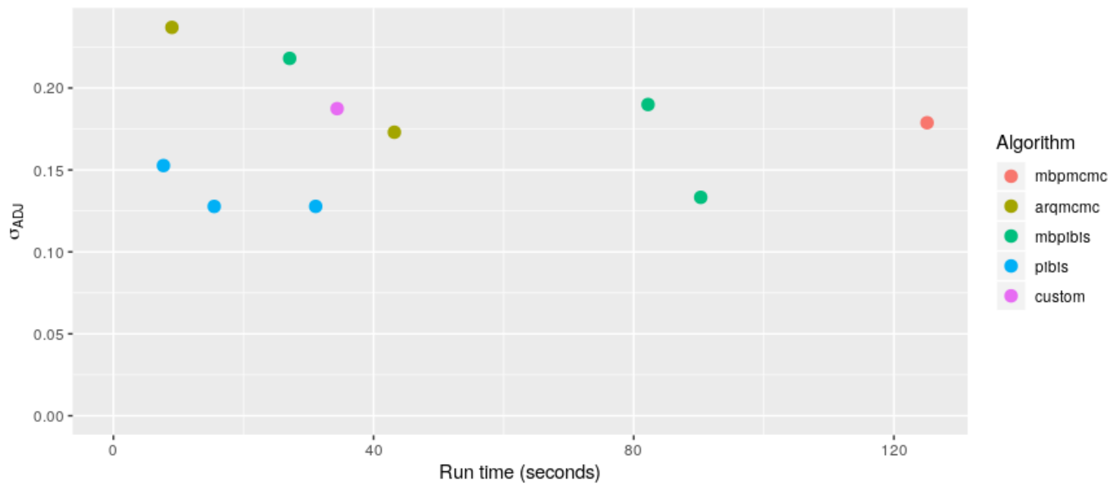


Figure 4.3: **Algorithm performance**. Comparison of algorithm run time (in seconds) and the mean adjusted error for estimates of $E(\theta)$ for the performance evaluation reported in §3.4. Some algorithms permit configurations that trade accuracy for computational effort and a number of approximately optimal configurations were used; hence some algorithms have two or three separate markers. The 'custom' marker is explained in the main text (in this section). See §3.4 for other details of the analysis.

they can be combined as distinct modules of the same analysis (or 'sub-workflow') e.g. for the purposes of independent cross-validation. For example, the two importance sampling algorithms yield an inexpensive and unbiased estimate of the marginal likelihood $\pi(y)$ which is important for formal model assessment and comparison (discussed in §4.5). The estimates obtained could subsequently be cross-validated by comparison with the results of an MCMC analysis, w.r.t. *parameter* estimates (i.e. rather than the marginal likelihood, which is difficult to estimate if not completely intractable using MCMC). A simple but robust chain of validation could then be completed with a standard test of convergence for MCMC, such as the Gelman-Rubin diagnostic. An illustration of such a workflow is given in Figure 4.4.

4.4 Phase three: model validation

The third and final phase of the [single-model] workflow pertains to 'model checking' – assessing the suitability of the model for describing the observations data, y. This

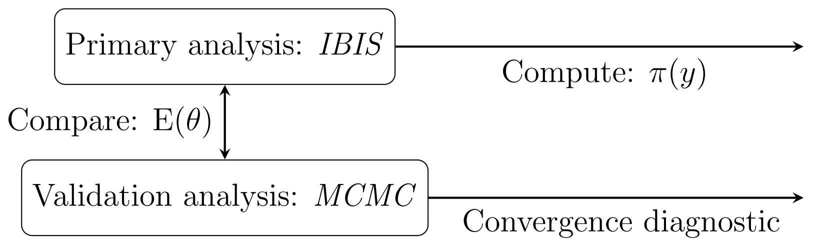


Figure 4.4: **Multi-algorithm workflow**. Schematic depictiobn of a two-algorithm inference workflow for DPOMP models. In this case the choice of IBIS algorithm corresponds to SMC^2 or MBP-IBIS, and MBP-MCMC or ARQ-MCMC for the MCMC validation algorithm.

is distinct from any validation carried out as part of the inference analysis itself as described in the last section. Here assume that inference has been carried out and reliable posterior samples have been obtained. Three somewhat distinct and generally applicable aspects of validation are now described: Simulated inference; Prior predictive and Posterior predictive checking.

4.4.1 Simulated inference

Simulated inference involves applying inference algorithms to observations simulated from a known data generating mechanism (i.e. from a known process and observation model with known parameters). This allows assessment of the ability of inference methods to recover estimates for (the known) model parameters. Furthermore this can be extended to validating model inference and observational or experimental study design by assessing how much and what type of data are needed to conduct reliable inference.

4.4.2 Predictive checks

Predictive checking involves sampling (or resampling) the complete model – including model parameters, system trajectories and [simulated] ‘observations’ data. The distribution of simulated observations data can then be compared with real-world data sets in order to help assess whether typical behaviour in the model is similar to that observed in the actual system of interest. *Prior predictive checks* draw parameter values from the prior and are aimed at assessing the plausibility of the range of models included within this space. The idea is to check for regions of parameter space that give rise to model outcomes that are impossible or highly implausible given what we know about the behaviour of the focal system. *Posterior predictive checks* involve sampling parameter values from the [estimated] posterior and assessing how well the model under the parameter posterior (i.e. the posterior predictive distribution) agrees with observed data.

In the case of DPOMP models, resampling the posterior θ-samples (or estimate) obtained in the second phase of the workflow is a trivial exercise. Likewise for sampling from the prior. The model realisations are obtained by DGA (Algorithm 1). Applied examples of predictive checking are included with the case study described in the next chapter.

4.5 Phase four: Model validation

Model validation contrasts with the validation of methods, such as those suggested for earlier phases of the workflow. In this case the goal is to assess the *suitability* of proposed models for the data in question as opposed to, e.g. confirming that a statistical method has been implemented correctly, or that an algorithm has converged sufficiently.

The marginal likelihood $p(y)$ provides an empirical measure of how well a single model fits the data. In keeping with the Bayesian approach, we can also utilise the *marginal likelihood* for the problem of model validation and comparison (e.g. see §2.5). To recap, application of Bayes' theorem allows us to write an expression for the different choices of model available, as follows:

$$\pi(y, M_i) = \int \pi(M_i)\pi(y_{1:n}|x_{1:n}, \theta, M_i)\pi(x_{1:n}|\theta, M_i)dx_{1:n}d\theta \tag{4.14}$$

where M_i is the i^{th} model chosen, $\pi(M_i)$ is the corresponding prior distribution (i.e. the prior belief in that model), and the observation and process model distributions show explicit dependence on model structure that was previously implicit). The Bayes factor provides a standardised way to directly compare these quantities for two candidate models:

$$K_{1,2} = \frac{p(y|m_1)}{p(y|m_2)} \tag{4.15}$$

where, according to the scale originally proposed by Jeffreys [65], $K_{1,2} > 10$ can be considered to be strong evidence for favouring model m_1 over m_2.

$\log_{10} K$	K	Strength of evidence
0 to 1/2	1 to 3.2	Not worth more than a bare mention
1/2 to 1	3.2 to 10	Substantial
1 to 2	10 to 100	Strong
>2	>100	Decisive

Table 4.2: **Bayes factor interpretation**, from Kass and Raftery[1].

4.6 Discussion

We followed the broad three-phase structure of the Bayesian workflow laid out by [11]: model construction; Bayesian inference; and model validation (plus model comparison). We have described the statistical tools and algorithms necessary for implementation of a principled Bayesian workflow for DPOMP models. We also described a multi-algorithm approach that made use of two novel algorithms for inference introduced in Chapter 3 that, together with the established methods collated in Chapter 2, represent a broad suite of statistical and algorithmic approaches to the problem. These cover online versus offline and importance sampling versus rejection sampling algorithms, with each of the four pairwise combinations of those approaches represented in our analysis.

We note that in practice, the task of conducting a complete Bayesian analysis is almost always one of iteration and refinement. As such, we would not expect any such workflow to be executed precisely and exclusively in the order given. For example, if having executed a complete multi-model inference workflow, we face a choice between a simplistic poorly fitting model, and an elaborate one that seems to better fit the data (but perhaps not sufficiently to warrant rejecting the model according to the Kass-Raftery scale) we might wish to revisit the model construction phase in order to experiment with one or more hybrids. This iterative approach was adopted during the course of the work carried out for Chapter 6 concerning within-herd modelling of BTB dynamics in UK cattle herds and led to development of the 'simple reinfection' model introduced in that chapter. Another example concerns the technique referred to as prior predictive checking. Due to its technical similarity with posterior predicting checking, both methods were presented here as part of the third, model validation phase of the workflow. However it seems quite likely that in at least some situations, prior predictive checking would be carried out beforehand, perhaps so as to inform and validate the selection of the prior distribution (and even model structure) in the first place. A useful and representative illustration of this 'tangled [Bayesian] workflow', as the authors refer to it, is given in a flow diagram in [11] (see Figure 1 in that paper).

We now conclude with some final remarks on further work towards a more complete implementation of a principled Bayesian workflow for DPOMP models.

Consistent presentation of results

The temporal order in which a Bayesian workflow has been executed arguably matters somewhat less than consistency of presentation of results e.g. for the purposes of conducting and publishing scientific research. Consistent treatment lowers the cognitive burden of understanding the results for others, and thus improves the transparency and usefulness of published findings. Such considerations should encompass: practical considerations such as convergence diagnostics and other validation metrics; the format of results tables; and standardised tools for visualisation, only some of which have been narrowly addressed here. Others are addressed in the next chapter. In general we believe that it is advisable to simply follow the conventions established by the best known and curated open-source tools available for Bayesian computation, such as Stan, wherever possible. Where we have failed to do so it is more than likely due to oversight rather than deliberate choice.

Bayesian model averaging

One avenue of investigation not explored in 4.5 is that of *Bayesian model averaging* [69], which essentially circumvents the problem of model selection altogether by providing a weighted average *across* models.

For example, let θ_α denote some common parameter across models (e.g. the 'latent period' in competing disease models). The posterior distribution of θ_α given data y is:

$$\pi(\theta_\alpha|y) = \sum_{i=1}^{n} \pi(\theta_\alpha|M_i, y)\pi(M_i|y) \tag{4.16}$$

where the M_i denotes the i^{th} of n competing models. This approach is self-evidently suited only to models that have important quantities in common, in particular nested models. However a key advantage is that the terms on the RHS can be can be derived from the posterior parameter estimates and marginal likelihoods obtained during earlier phases of the analysis, thus adding substantial value to the results with only a trivial amount of additional computation required.

Model averaging is especially useful when the goal of an analysis includes prediction (i.e. sampling from the posterior, rather than merely characterising it). This is because it incorporates *model* uncertainty in predicted values which, intuitively, is preferable to simply choosing the 'best' model and disregarding model uncertainty thereafter for that purpose.

Algorithm enhancements

The modular multi-algorithm inference workflow proposed in Figure 4.4 calls for manual comparison of posterior estimates for the purposes of cross-validation. However, a computed quantity, such as the Kullback–Leibler divergence would be better, both because it would avoid the need for subjective interpretation and also because it would be more conducive to automation. Alternatively, implementing a method for formal comparison of n independently obtained *weighted* samples would diminish the need for an additional validation analysis altogether.

Generalised software implementation

Perhaps the most useful and obvious way to extend this work would be to produce a generalised software implementation and make it available as an open-source research tool. That challenge is addressed directly in Chapter 5.

Chapter 5

Software for Bayesian analysis of DPOMP-modelled data

"In practice, making code truly open source — in the sense that it can be run, usefully, by a wide variety of people and on a wide variety of platforms — demands a commitment that few scientists are able to make. When building scientific software, researchers usually face the choice to build something that helps answer a specific problem, or to build a more generalized tool and invest in an effort to build a user community around it."

- Professor Steve Easterbrook [70]

Summary

- Bayesian statistics provides an extensible framework for parameter and model inference that is in many ways ideal for scientific research. For example it allows us to incorporate prior findings, beliefs and commonly accepted scientific knowledge as the 'prior distribution' (of e.g. model parameters). It also allows us to mathematically characterise the observations process itself, allowing for arbitrary amounts of complexity and intrinsic uncertainty. In doing so, it allows that the best possible use can be made of whatever experimental or observational data are available.

- The previous chapter laid out the definitions and statistical tools required to implement a multi-phased workflow for discrete-state-space, partially ob-

served Markov process (DPOMP) models, in line with emergent best practice in the field of applied Bayesian statistics.

- This chapter describes a generalised software implementation of that workflow: an open-source package for Bayesian analysis of DPOMP-modelled data in Julia, including model definition and validation. The *BayesianWorkflows.jl* package is designed specifically for the generic class of DPOMP model described in the last chapter and includes automated tools for model construction; simulation; parameter inference; and model inference (i.e. formal model comparison) as well as model validation.

- We demonstrate its use by mirroring a case study on Bayesian workflows for disease transmission data (influenza A, H1N1) as laid out by [5] for ODE-models in Stan.

5.1 Introduction

The practical work conducted throughout the project has relied on a well-organised code base, with particular emphasis on the *Julia* programming language. That includes at least somewhat-generalised implementations of the algorithms and workflows described in previous chapters, including model definition; visualisation; and diagnostics for Bayesian inference based on the class of model laid out in the beginning of Chapter 2 – *discrete-state-space partially observed Markov-processes* (DPOMPs). In accordance with the principles of open science, this work has been organised and documented as an open source software package: a *Julia* package called *BayesianWorkflows.jl*[1]. The remainder of the introduction summarises existing software provision for Bayesian inference, with emphasis on the general class of problem considered in previous chapters. The current availability provides motivation for an additional offering pertinent to DPOMP models specifically. The rest of the chapter is given over to describing the package itself, including key functionality and examples.

[1]Source code and readme: https://github.com/mjb3/BayesianWorkflows.jl

5.1.1 Existing software provision

It is arguably preferable to use, or at least validate against, established statistical software tools (and thereby methods) when conducting scientific research, so as to help assure the accuracy and reproducibility of results. A short review of available software for Bayesian inference reveals that options for the practical implementation of inference methods depend largely on the model and task at hand.

More specifically, for problems where the posterior density, *including* the time-evolution of the state-space-model, can be expressed using a set of differential equations – there is generous provision. Well known examples include *Stan* [71], *JAGS* [72], and *WinBUGS* [73]. Popular tools of this nature can be understood as stand-alone software; usually declarative statistical programming languages in their own right that tend to be accompanied by multiple interfaces, ranging from stand-alone graphical and command-line applications, to software libraries for other programming languages such as Julia and R. There is also overlapping functionality available for the latter on a more language-specific basis. For example, the *'Turing Language'*[2] – a probabilistic programming library for Bayesian inference in *Julia* – includes packages for *Hamiltonian* MCMC. This is a recently popular sampling method that is also used in tools like Stan [71, 74].

Software for DPOMP models

For more sophisticated statistical models –in this case, those that imply the need to solve integrals over the joint parameter and measure space of the model, DPOMP models– the provision of available software (in any language) is less generalised and typically therefore more limited. Again though, it depends greatly on the specific task and methods in contemplation. The simulation of stochastic processes in general is a widely considered problem, with software implementations that cater to specific classes of stochastic process and simulation protocols, and also abstract ones, e.g. [75].

The same is true – at least to an extent – for statistical inference. For DPOMP models specifically, the available provision tends to be based solely on sequential Monte Carlo (SMC) techniques though. For example, the *pomp* [76] package for *R* provides access to *plug and play* approaches including both particle filtering

[2]https://github.com/TuringLang

and full particle-MCMC, but not access to more sophisticated sequential methods such as SMC^2, or to data-augmentation methods. We were not able to identify an equivalent package for *Julia*, but broadly the same functionality is available by other means. For example, direct simulation of inhomogeneous Poisson processes (i.e. Algorithm 1) is available via the *Gillespie.jl* package[3]. Similarly, the *JuliaPOMPD* library for [discrete-time] *'partially observed Markov decision processes'* (POMDP) includes the *ParticleFilters.jl* package for SMC, which by the algorithm's design is just as applicable (at least in theory) to [continuous-time] POMP model-like problems.

5.1.2 Motivation for ***BayesianWorkflows.jl***

We have briefly noted a few of the better known software tools and packages for simulation and Bayesian inference. However, the provision for DPOMP models specifically was found to be too limited to be useful for the task at hand (as considered more directly in Chapter 6 onwards). In particular, we are not aware of any generalised provision for efficient parameter inference using DA methods in conjunction with DPOMP models. At least, none that are widely publicised or used, though bespoke coded examples can sometimes be found in blogs and tutorials (in addition to academic sources).

The lack of generalised implementations for DA methods, compared to SMC, is not particularly surprising though. The modular nature of the latter lends itself quite well to generalised implementation. In particular, the encapsulated nature of particle filters is a convenience that can be exploited for this purpose. By contrast, both the conceptual design and engineering of DA algorithms are naturally more intertwined with the workings of the inner state-space model. This can be understood by considering that, where a particle filter may yield an unbiased estimate $\propto \hat{\pi}(\theta|y)$, DA algorithms are characterised by their reliance on a joint density that includes the augmenting variable x, written herein simply as $\pi(\theta, x|y)$. In crude terms then, the 'challenge' to software provision for advanced Bayesian methods (DA-based, in particular) is to provide an interface that is simple enough to be useful in shorthand. That is, without writing excessive amounts of code, scripts, or long-winded configuration files. But also *scalable* with regard to the nature of the augmented variable (or equivalently, the complexity of the state-space model).

[3]https://github.com/sdwfrost/Gillespie.jl

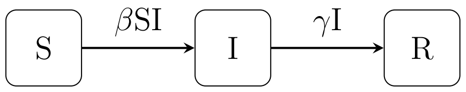
In reality, that challenge has effectively already been met, by the package-based ecosystems of statistical languages like *Julia* and *R*. However even in highly-engineered feature rich programming environments such as these, custom implementations require time, or at least advance knowledge of those resources, along with at least some degree of familiarity with the methods themselves. There may therefore be a useful role for somewhat less generalised intermediate solutions; packages designed to be more accessible to beginners or those with limited time in particular, while still scalable enough to (hopefully!) be useful in a range of situations.

The package we describe is a modest attempt to provide such with an [almost] exclusive focus on DPOMP models, albeit only in one language so far. It follows the principled workflow approach to Bayesian data analysis laid out in the previous chapter, and relies heavily on the *modular construction* approach expounded by Gelman *et al.,* in their paper on Bayesian workflows, and also in the quotation cited at the top of the last chapter. The quoted passage alluded more to modular construction in the context of *model* construction, but the same principles can be applied to implementation of Bayesian workflows too (much as they were for algorithm development in the previous chapter). For example, we treat algorithms as interchangeable modules in the construction of an inference workflow, which is treated in turn as a largely self-contained module (phase two) in the overarching data analysis workflow.

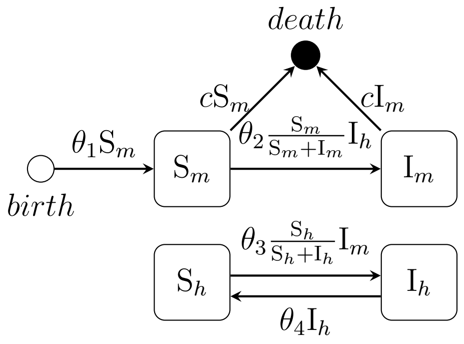
In line with the motivating application for the thesis, the package is presented with a heavy focus on practical examples from epidemiology (e.g. terminology, predefined models, etc). Likewise for the problem used to demonstrate key features in this chapter, which mirrors a case study on Bayesian workflows for ODE-based models of disease transmission [5]. Sample code is provided with the online package documentation, in addition to the snippets given throughout this section, and in Appendix B.

5.2 Case study: models of disease transmission

Here we demonstrate use of the package's main features with help of a relevant case study. It is a Bayesian workflow for analysis of an influenza A epidemic at a British boarding in 1978 by Grinsztajn *et al.,* [5]. That work is based on a mathematical ODE-based model, coded in R and Stan. As with the workflow laid out specifically for DPOMP models in the previous chapter, the first phase is model construction. We begin by describing the observational data and note the key mathematical definitions of the disease transmission model (the SIR, as illustrated in Figure 5.1a). We then directly address implementation (i.e. 'coding') of the model for use in subsequent phases of the workflow.



(a) A standard representation of the *susceptible-infectious-recovered* (SIR) model.



(b) A more complex two-species example, which incorporates both disease and population dynamics. It is based on the SIS-model variant proposed by Ross and MacDonald [77] for modelling Malaria in humans and mosquitoes.

Figure 5.1: *Discrete-state-space* models of varying complexity. Different permutations can be used to describe a large variety of different systems. These two examples illustrate the potential to describe disease dynamics (and the interaction between demography and disease dynamics) in both single- and multi-host disease systems. The worked example described in the text is based on the SIR model.

5.2.1 Data

The data analysed by Grinsztajn *et al.*, come from an influenza (H1N1) outbreak at a British boarding school in 1978, are publicly available and were downloaded using the *Outbreaks*[4] package in R. The data consist of daily observations taken over two weeks, as shown in Table 5.1.

Date	In bed	Convalescent
1978-01-22	3	0
1978-01-23	8	0
1978-01-24	26	0
1978-01-25	76	0
1978-01-26	225	9
1978-01-27	298	17
1978-01-28	258	105
1978-01-29	233	162
1978-01-30	189	176
1978-01-31	128	166
1978-02-01	68	150
1978-02-02	29	85
1978-02-03	14	47
1978-02-04	4	20

Table 5.1: Data reported from an outbreak of influenza at a British boarding school in 1978.

The *In bed* column was interpreted by the authors of the mirrored case study to be a 'messy count' of the number of infectious individuals in an [ODE-based] SIR model. More precisely, they assumed that the number of students *'in bed'* is distributed according to a negative binomial $\sim NB(I, \phi)$, where I is the true number of infectious individuals and the a priori unknown parameter ϕ determines the 'messiness' of the count.

[4]https://github.com/reconverse/outbreaks

5.2.2 Disease transmission model

A DPOMP model is essentially described by:

1. An event rate function (and transition matrix/function).
2. A function for sampling the initial system state.
3. The observation model (i.e. [log-] likelihood and sampling function).

Event rates and state transition

For the SIR model, with frequency-dependent transmission, the event rates are given by:

$$r_1 = \beta SI/N \tag{5.1}$$

$$r_2 = \gamma I \tag{5.2}$$

where the infection and recovery rates are labelled r_1 and r_2 respectively. Note that this will correspond to their positions in the output [Array] variable when coding the model. β and γ parametrise the infection and recovery processes. Lastly S, I, and N, correspond to the number of individuals in the first two compartments, and the total number of individuals, with $R = N - I - S$. The matrix that describes *transition* between states is given as:

$$T = \begin{bmatrix} -1 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix} \tag{5.3}$$

Note that the row vectors are associated with each distinct type of event, and column vectors relate to the model state-space, i.e. S-I-R, from left-to-right in (5.3).

Initial condition

The initial condition of the model encompasses the initial system state and time. In this case it is assumed that the epidemic began with single infectious individual at some unknown time denoted by t_0. The complete model parametrisation is given by $\theta = \{\beta, \gamma, \phi, t_0\}$, where the parameter that has not yet been described, ϕ, is used to parametrise one of two observation models discussed below.

Observation model(s)

The observation model links the observable quantities, in this case the number of students in bed, with the underlying system state, i.e. the number of individuals in each compartment. In their case study, Grinsztajn et al assume that the observable quantity is sampled from the negative binomial distribution. Here this is parametrised as follows:

$$y_{bed} \sim NB(I, \phi^{-1}) \tag{5.4}$$

where y_{bed} is the number of students in bed; I is the number of infected individuals according to the model; and the unknown parameter ϕ is used to account for overdispersion in the counting process. The transformation ϕ^{-1} is a convenience that will be explained in due course.

Note that this choice of distribution can account for both over-estimates and under-estimates w.r.t. the true number of infectious individuals, i.e. $y_{bed} <> I$ is possible. However it can also lead to observations where $y > N$ if the distribution is not truncated, which is plainly absurd. In addition to truncating the first distribution according to population size, we also consider a simple alternative observation model based on the binomial distribution instead.

$$y_{bed} \sim B(I, \phi) \tag{5.5}$$

This has the virtue that $y <= N$ is guaranteed but precludes the possibility of over-estimates in the data, which may be unrealistic. The two variations are therefore compared using formal model comparison methods in §5.5 to establish which is a better fit for the data at hand.

5.2.3 Coding the model

In order to use the automated workflows provided with the package, users must first define the model using the Julia language. The package includes a number of predefined (mostly epidemiological) models including the SIR model, invoked as follows:

```julia
using BayesianWorkflows    # NB. install the package first
# generate model:
initial_condition = [100, 1, 0]
model = generate_model("SIR", initial_condition; freq_dep=true)
```

Listing 5.1: Generate a predefined model instance.

Where applicable, predefined models are *density-dependent* by default so the *frequency-dependent* option has been specified for this example instead. In this case it is more helpful to define the model manually however, so as to provide a more complete demonstration. We therefore begin by defining a set of constants that relate the key properties of our model:

```julia
 MODEL_NAME = "SIR"
 N_EVENT_TYPES = 2
## model state space:
 STATE_S = 1
 STATE_I = 2
 STATE_R = 3
## model parameters:
 PRM_BETA = 1
 PRM_GAMMA = 2
 PRM_PHI = 3
 T_ZERO = 4
## observation model parameters:
 OBS_BEDRIDDEN = 1
 OBS_CONV = 2
```

Listing 5.2: Global constants.

Note: not all of these constants are not actually invoked by the code that follows, but are included anyway for illustration and completeness. The use of constants is good programming practice because it helps to make code more easily maintainable, and also more decipherable for human readers (although in general it is probably not advisable to define redundant ones, as we have done here for the sake of exposition).

Event rate function

The event rate function operates on a predefined output variable. Note that since the function's output variable is predefined (by *Julia* convention) the function name has an exclamation mark appended (as in *'function_name!'*) and operates on the variable that is passed first. In other words the user's only responsibility when defining the custom event rate function, is to populate it with the correct values.

For the SIR model used here this is done as follows:

```
1   # rate function:
2   function sir_rf!(output, parameters::Vector{Float64}, population↩
        ::Vector{Int64})
3       output[1] = parameters[PRM_BETA] * population[STATE_S] * ↩
            population[STATE_I] / sum(population)
4       output[2] = parameters[PRM_GAMMA] * population[STATE_I]
5   end
```

Listing 5.3: Rate function for the SIR model. Note arrows(carriage return symbols) shown on lines 2 and 3 (and latter) denote only the continuation of the line in question and have no algorithmic or other significance.

The order of event types is arbitrary in theory although it can have performance implications in practice[5]. It is however important to *note* the order that is chosen, since the position of a given event type in the rate vector corresponds to the 'event type' argument that is passed as an integer to the transition function.

Transition function

The transition function is a user-defined function that relates the occurrence of events to the actual migration of individuals between compartments (based on the event type, which is passed as an argument to the function). In this case (and many others) it is possible to specify migration completely using an N_r by N matrix, where N_r is the number of distinct types of event. In such cases the software can be used to automatically generate a transition function based on a transition matrix

[5]These are not discussed here for reasons of brevity, suffice to say that events of a more frequent nature (i.e. a usually high rate of occurrence) should be listed first in cases where computational efficiency is a concern.

(‘Array’ in this case) using an appropriate model constructor or helper function, like so:

```
# transition matrix:
tm = [-1 1 0; 0 -1 1]
transition = BayesianWorkflows.generate_trans_fn(tm)
```

Listing 5.4: Transition matrix and function.

where the ‘tm’ variable is given by equation (5.3). Note that for some models it may be appropriate (or simply more efficient) to incorporate more complex logic in the transition function. For example we could treat migration as a random variable, and sample it *conditionally* on the event type instead [6]. In cases such as this, it is necessary to manually specify the transition function. Examples of manually specified functions are included online, in the source code repository.

Initial state function

As noted above, the initial condition of the model is defined by an initial time t_0 and the initial system state. For reasons that are similar to those noted above for the transition function, the initial value of the system state variable is treated as a function rather than a given quantity (as it is assumed to be in this case). In this instance, we require only a simple function that always returns the same value. It is convenient to define it in Julia using in-line syntax for brevity:

```
population_size = 763
initial_state = [population_size - 1, 1, 0]
# initial state (in-line) function:
get_initial_state(parameters::Vector{Float64}) = initial_state
```

Listing 5.5: Transition matrix and function.

The ‘given quantity’ in this case is encoded by the ‘initial_state’ variable, and is returned by the ‘get_initial_state’ [transition] function. Note that the transition

[6]A practical example of this is when we have a process that applies globally (and uniformly) such as births and deaths. In a model with n possible states, it may be more straightforward and parsimonious to define one event type for e.g. deaths and then sample from n states, than it is to define n separate event types with one for each individual state.

function and its argument can be arbitrarily named but that we must use the correct *function signature*, even in cases like this where the 'parameters' variable is not actually invoked by the function. The same applies to all other functional components (or 'modules') of the model.

Observation model

In the context of the package, a realised version of the statistical observation model calls for two functions that are in a sense the inverse of one another. They are the observation sampling *density* (more precisely, log-likelihood) and an observation generating function that permits us to sample from that density.

The first is implemented for the Negative Binomial (i.e. original, but truncated) observation model as follows:

```
1   ## resources: nb. install the Distributions pkg first
2   using Distributions
3   ## observation log likelihood function:
4   function obs_loglike(y::Observation, population::Array{Int64,1},↩
        parameters::Vector{Float64})
5       population[STATE_I] == y.val[OBS_BEDRIDDEN] == 0 && (return ↩
            0.0)
6       population[STATE_I] == 0 && (return -Inf)
7       # truncated -ve binomial distribution:
8       dist = truncated(NegativeBinomial(population[STATE_I], ↩
            parameters[PRM_PHI]^-1), 0, population_size)
9       return logpdf(dist, y.val[OBS_BEDRIDDEN])
10  end
```

Listing 5.6: Observation log likelihood function.

where the 'obs_loglike' function is a relatively simple function that acts as a wrapper for the definition of the observation model distribution, and a call to the 'logpdf' function, both of which depend on the *Distributions.jl* package. The first two lines of the function (i.e. lines 5 and 6) ensure that the call to that function is valid by handling specific scenarios relating to null conditions that would otherwise cause an exception in the program.

The corresponding sampling function for this observation density is defined thusly:

```
function obs_sample!(y::Observation, population::Array{Int64,1},
    parameters::Vector{Float64})
    if population[STATE_I] > 0
        dist = truncated(NegativeBinomial(population[STATE_I],
            parameters[PRM_PHI]^-1), 0, population_size)
        y.val[OBS_BEDRIDDEN] = rand(dist)
    else
        y.val[OBS_BEDRIDDEN] = 0
    end
    y.val[OBS_CONV] = population[STATE_R]
end
```

Listing 5.7: Observation generating function.

Here we have relied heavily on the Distributions.jl package to implement the model. Although this is not strictly necessarily, it is convenient for modular construction of the models, in that both functions can be easily reworked for the alternative, Binomial observation model. This is accomplished by replacing a single line of code in each function: we replace the truncated Negative Binomial distribution with a simple Binomial one. For example:

```
# comment out the -ve binomial obs model
# dist = truncated(NegativeBinomial(population[STATE_I],
    parameters[PRM_PHI]^-1), 0, population_size)
# replace with binomial obs model:
dist = Binomial(population[STATE_I], parameters[PRM_PHI])
```

Listing 5.8: Converting to a Binomial model.

Complete model definition

The full model definition is encapsulated within a Julia Type, with the base constructor invoked as follows:

```
DPOMPModel(MODEL_NAME, N_EVENT_TYPES, sir_rf!, initial_state,
    transition, obs_loglike, obs_sample!, T_ZERO)
```

Listing 5.9: Instantiating the model.

As with the predefined models and alternative constructors, the resulting DPOMP-Model Type is a mutable object ('struct' in the Julia language). In plain terms this means that the package supports the fully modular construction of models. For example, different model variants can be generated from the same pool of modular components, but component modules can also be updated or replaced on an individual basis as required. That includes predefined models: as per the code sample given at the top of this section, it would have been possible to generate a predefined SIR model and simply overwrite components like the observation log-likelihood [function,] whilst retaining those like the event rate and migration functions. Others, like the model 'name' property, can be updated or not as the user desires.

5.3 Fitting the model

Here we describe the second broad phase of the workflow: how to fit the model and check the results. For brevity we focus on the first model variant, based on a truncated Negative Bionomial observation model, with the alternative variant considered during the model comparison exercise described in §5.5. The purpose of fitting the model is to produce posterior estimates for the model parameters θ. As per the explanations given in previous chapters, the posterior distribution is written as:

$$\pi(\theta, x|y) = \frac{\pi(\theta)\pi(x|\theta)\pi(y_{1:n}|x,\theta)}{\pi(y_{1:n})} \tag{5.6}$$

Recall that the *marginal likelihood* is written as $\pi(y_{1:n})$ to indicate that it runs over n [longitudinal] observations. The task of computing (5.6) is the purview of the sampling algorithms discussed in due course. We first discuss the *prior [probability] distribution*, $\pi(\theta)$.

5.3.1 Prior distribution

Selection of a prior distribution of model parameters, denoted $\pi(\theta)$– where $\theta = \beta, \gamma, \phi, t_0$ in this case– is an important part of the overall analysis which we revisit with *prior predictive checking* in due course. For now though, we closely follow the prior distribution used by the authors of the original case study. The distributions and parametrisations reported by those authors are assumed to be independent between parameters as follows:

$$\beta \sim N(2,1) \tag{5.7}$$

$$\gamma \sim N(0.4,0.5) \tag{5.8}$$

$$\phi \sim exp(5) \tag{5.9}$$

The first two are rate quantities and therefore they are truncated at zero. Likewise, ϕ^-1 is a probabilistic quantity in this variant of the model, and so the *lower* bound of ϕ is truncated at one to ensure $\phi^-1 < 1$. A prior distribution for t_0 is not specified for the ODE-version of the model described in the original case study so a Uniform distribution was used, bounded on the dates *1978-01-16* to *1978-01-22*, inclusive. The above can be described in Julia as a multivariate statistical distribution, as follows:

```
beta = truncated(Normal(2.0, 1.0), 0.0, Inf)
gamma = truncated(Normal(0.4, 0.5), 0.0, Inf)
phi = truncated(Exponential(5.0), 1.0, Inf)
t0_lim = ["1978-01-16", "1978-01-22"]
t0_values = Dates.value.(Dates.Date.(t0_lim, "yyyy-mm-dd"))
t0 = Uniform(t0_values...)
prior = Product([beta, gamma, phi, t0])
```

Listing 5.10: Prior distribution.

Unlike before, it is *necessary* to use the Distributions package to define the prior distribution, since that is the Type required by relevant functions when conducting an inference analysis (we could have specified the observation model without using it if we had wanted to).

5.3.2 Selection of sampling algorithm

Implementing workflows for inference (i.e. fitting models to data) is relatively straightforward once a model has been defined. The default [single-model] inference workflow (described in §4.3.4) is based on independent inference analyses carried out using two designated algorithms. The default inference algorithms used in this workflow are the established methods introduced in Chapter 2: *SMC*^{2} and *MBP-MCMC*.

Here we provide code and results for the same essential workflow, but implemented using the algorithms described in Chapter 4 instead. They are:

- Primary: MBP-IBIS
- Validation algorithm: ARQ-MCMC

Note that the second algorithm requires an additional parameter not required for the others: *sample interval*. This is discussed in the previous chapter. A number of optional parameters are also used, some of which are algorithm specific. For example, *n_mutations* which determines the number of steps in the permutation step of the MBP-IBIS algorithm. The complete inference workflow is invoked as follows:

```
# define sample interval for arq mcmc algorithm
sample_interval = [0.05, 0.01, 0.02, 0.5]
# run analysis
results = run_inference_analysis(model, prior, y;
    primary=BayesianWorkflows.C_ALG_NM_MBPI, n_particles=12000, ↩
        n_mutations=7, validation=BayesianWorkflows.C_ALG_NM_ARQ, ↩
        sample_interval=sample_interval(freq_dep), n_mcmc_chains=5)
tabulate_results(results)
save_to_file(results, string(some_file_path, "/", model.name))
```

Listing 5.11: Inference analysis.

The results produced can be analysed in a number of different ways, with perhaps the simplest and most obvious being tabulation, as demonstrated in the code sample. They can also be saved as CSV files, also as demonstrated. These and other means of exploring the results are described next.

5.3.3 Checking inference results

As per the description of the [modular] inference workflow given in §4.3.4, results of the two-part inference analysis are comprised of a weighted *importance* sample and posterior samples obtained by [ARQ-] MCMC *rejection* [re]sampling. The *tabulate_results* function invoked above displays two tables summarising each, with content similar to Tables 5.2 and 5.3, for the MBP-IBIS and ARQ-MCMC algorithms respectively. A further set of results for the latter are given in Table 5.4.

θ	E[θ]	:σ	-ln p(y)
1	1.91	0.166	65.2
2	0.719	0.151	
3	2.67	0.487	
4	722105	0.82	

Table 5.2: Summary of parameter estimates (expectations and corresponding standard deviation, denoted σ) obtained using the MBP-IBIS algorithm. The parameters are listed by their index in θ, i.e. 1 : 4 for $\theta = \beta, \gamma, \phi, t_0$. An estimate of the marginal likelihood is also given, in the form: -ln $\hat{\pi}(y)$.

θ	E[θ]	:σ	E[$f(\theta)$]	:σ	$\hat{R}$	$\hat{R}$97.5
1	1.9	0.168	1.89	0.14	1	1
2	0.731	0.154	0.72	0.123	1	1
3	2.71	0.495	2.66	0.371	1	1
4	722105	0.867	7.22E+05	0.668	1.01	1.01

Table 5.3: Summary of parameter estimates obtained using the ARQ-MCMC algorithm. The parameters are indexed by $\theta_{1:4} = \beta, \gamma, \phi, t_0$ as before, with similar notation as for the MBP-IBIS algorithm. In this case the expectations and corresponding standard deviation, correspond to MCMC resamples. The other estimate given is based on the expectations of the evaluated function, denoted simply in the table as $f(\theta)$, across the entire [sparse] underlying QMC sample Γ. Finally, the parameter estimates are accompanied by the potential scale reduction factors, denoted $\hat{R}$.

The (independently obtained) estimates for the model parameters can be manually checked for similarity using the tables here indicating that the two algorithms generate essentially the same parameter inferences. The potential scale reduction factor (i.e. Gelman-Rubin test statistic) denoted as $\hat{R}$ also provides a means of validation, because it is a formal test of convergence.

Recall from Chapter 3 that the ARQ-MCMC algorithm provides a way for computing estimates directly based on the expectations of the evaluated function, denoted simply in the tables as $f(\theta)$. This can be understood as an importance sample weighted by MCMC. This is redundant in most cases but can be useful in situations where one or more Markov chains fail to converge. An example is given in Table 5.4, based on the same data and algorithm configuration as in Table 5.3.

θ	E[θ]	:σ	E[$f(\theta)$]	:σ	$\hat{R}$	$\hat{R}$97.5
1	1.97	0.565	1.89	0.138	1.33	2.55
2	0.783	0.251	0.712	0.117	1.41	2.42
3	3.69	3.08	2.64	0.345	1.78	7.51
4	7.22E+05	1.24	7.22E+05	0.678	1.18	1.44

Table 5.4: Alternative set of parameter estimates obtained using the ARQ-MCMC algorithm. In this case not all of the Markov chains fully converged, as indicated the potential scale reduction factors, $\hat{R} > 1.2$. However reasonably accurate estimates were nevertheless obtained using the expectations of the evaluated [likelihood] function, denoted in the table as $f(\theta)$, which is computed from the entire [sparse] underlying QMC sample Γ.

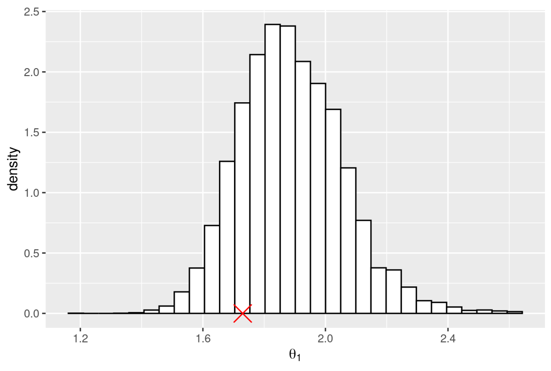
Visual inspection of the parameter inference results is also possible, notably single and pairwise marginal density plots. Traceplots can also be produced for the estimates obtained by MCMC. Examples are given in Figures 5.2 and 5.3, with the estimates for $\{\beta, \gamma\}$ reported by [5] in their case study marked for reference. Note that since that package is designed for use within the Julia REPL (command-line environment) the equivalent plots have been reproduced here using R and GGPlot2 instead.

As a minor aside, the visualisation tools included with the package are native to the Julia programming language and implemented using *UnicodePlots.jl*[7]. Thus, they can be manipulated in the same way as other plots produced using that package. For example:

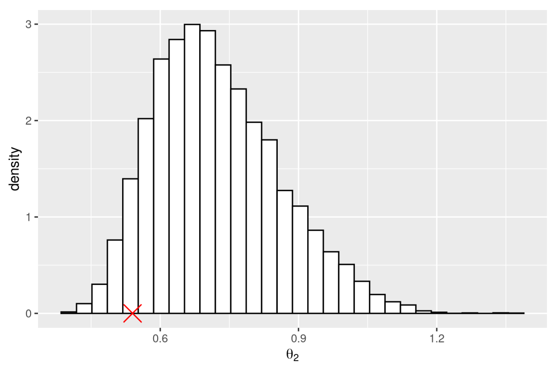
```
import UnicodePlots       # NB. install first, using Pkg.add
p = plot_parameter_heatmap(rs, 1, 2))
UnicodePlots.xlabel!(p, 'new x axis label')
println(p)                # show plot
```

Listing 5.12: Example: alter the axis label of a UnicodePlot.

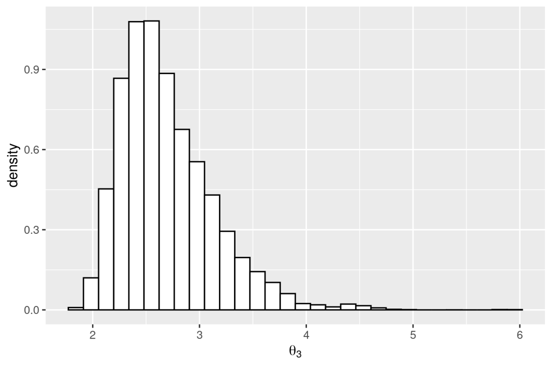
[7]https://libraries.io/julia/UnicodePlots



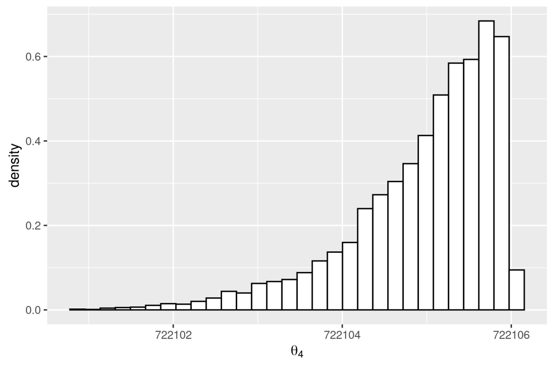
(a) Contact parameter β



(b) Recovery parameter γ



(c) Observation parameter ϕ

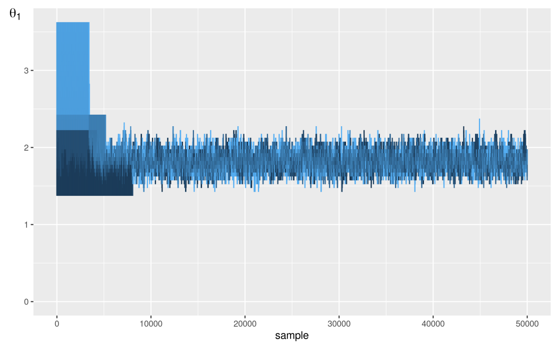


(d) Initial time t_0

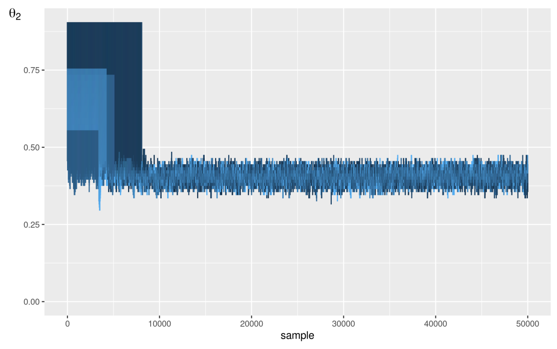
Figure 5.2: Marginal posterior estimates for model parameters $\theta = \{\beta, \gamma, \phi, t_0\}$ obatained using the MBP-IBIS algorithm. The estimates for $\{\beta, \gamma\}$ reported by [5] in their case study have been marked for reference.

5.3.4 Predictive checking

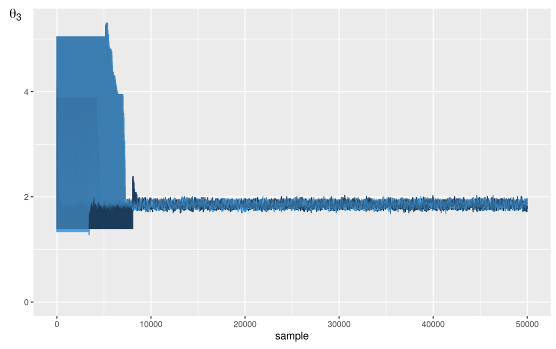
As noted above, the results of the inference analysis consist of a weighted *importance* sample (which can be resampled) and posterior samples obtained by MCMC *rejection* sampling. Either can be utilised for the purpose of full posterior predictive checking. This involves resampling model parameters based on the posterior estimates obtained, and simulating (in particular) new observations data using the data generating function defined earlier. The code for carrying this out is given below.



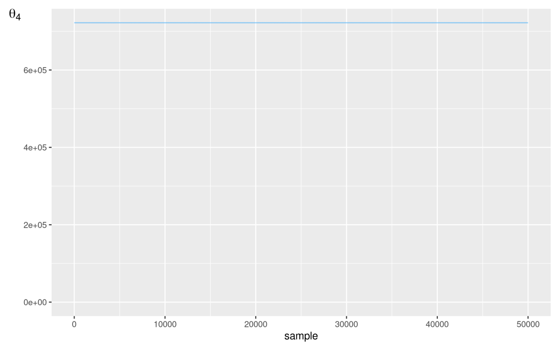
(a) Contact parameter β



(b) Recovery parameter γ



(c) Observation parameter ϕ



(d) Initial time t_0

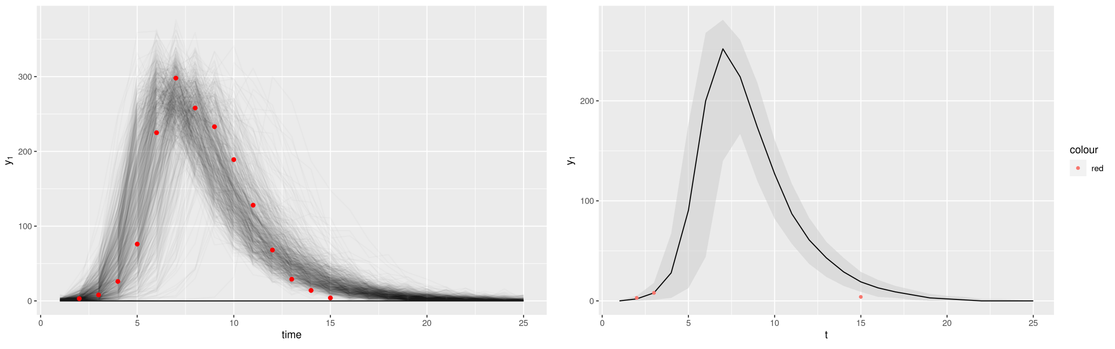
Figure 5.3: MCMC traceplots for model parameters $\theta = \{\beta, \gamma, \phi, t_0\}$ based on samples obtained using the ARQ-MCMC algorithm.

```
# nb. 'results' obtained by call to 'run_inference_analysis' above
model = results.model
parameters = resample(results; n = 1000)
x = gillespie_sim(model, parameters; tmax=max_time, num_obs=↩
    n_observations)
save_to_file(x, string(path_out, model.name, "/predict/"); ↩
    complete_trajectory=false)
plot_observations(x; plot_index=OBS_BEDRIDDEN)
plot_observation_quantiles(x; plot_index=OBS_BEDRIDDEN)
```

Listing 5.13: Posterior predictive checks.

The code sample includes a demonstration of how the results obtained can be passed to two visualisation tools included with the package for this purpose. In this case, to visualise the number of students in bed over one thousand individually simulated scenarios. One shows the simulated quantities themselves, the other as quantiles. Both are replicated in Figure 5.4. The raw observations data from the original epidemic has been overlayed for comparison with the predictive checks, and indicate that the posterior predictions produce results that are reasonably similar to those that were actually observed.

One way to understand the amount of useful information that has been garnered from the inference analysis, is to compare these to the *prior* predictive checks carried out for the next phase of the overall workflow: model checking. That is because, intuitively, our ability to predict outcomes similar to the observed epidemic should be at least somewhat improved by the information gained during inference, assuming that all has gone well.



(a) Simulated trajectories. (b) Quantiles.

Figure 5.4: Posterior predictive checks: simulated trajectories and percentiles $q = \{0.25, 0.5, 0.75\}$ for number of students in bed. The actually observed quantities are overlayed for comparison.

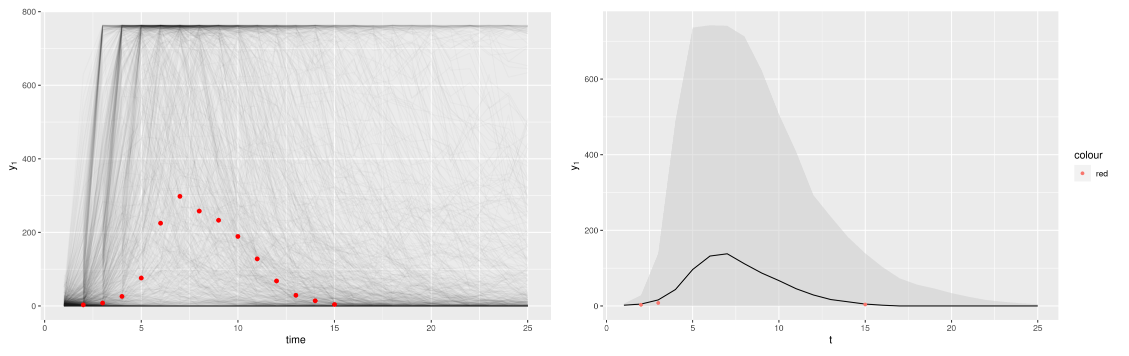
5.4 Model checking

The third and final phase of the single-model workflow involves further validation (or 'checking') in order to assess the suitability and effectiveness of the model (and methods) as well as our choice of prior distribution. Specifically, we cover two simulation-based techniques: *prior predictive checking* and *simulated inference*. We then move directly on to the task of model comparison before a brief summary to conclude the chapter.

5.4.1 Prior predictive checking

Prior predictive checks are carried out in the same manner as posterior ones, except that the parameters used to simulate the model are sampled from the prior distribution instead. Among other things, the checks help to ensure that the selection of prior distribution is appropriate, i.e. that the simulated scenarios are broadly consistent with our prior beliefs and expectations concerning system behaviour. For example, the prior predictions shown in Figure 5.5 show a broad range of possible system trajectories, that are at least *somewhat* centered on the actual observations.

Prior predictive checks can also be compared with posterior ones as a visual summary of the information gleaned about the system at hand, in terms of our ability to (hopefully) predict future system behaviour and outcomes. Comparison of Figures 5.4 and 5.5 suggests that considerable information is added by the posterior



(a) Simulated trajectories. (b) Quantiles.

Figure 5.5: Prior predictive checks: simulated trajectories and percentiles $q = \{0.25, 0.5, 0.75\}$ for number of students in bed. The actually observed quantities are again overlayed for comparison.

estimates obtained by inference, as revealed by comparison of the interquartile ranges in particular.

5.4.2 Simulated inference

The final stage of the single-model inference workflow involves the use of simulated observations data to evaluate the effectiveness of the inference methods used. That is, their ability to accurately recover the model parameters used to simulate new observations data. Much like the prior predictive checks, this stage could also be executed at the beginning of the workflow as pre-validation. However since the actual process of Bayesian data analysis tends to be iterative and cyclical, the precise order of steps listed in any given 'workflow' is a matter of interpretation in any case.

Simulated observations can be generated by choosing an arbitrary set of model parameters, e.g. by sampling from the prior distribution and using the 'gillespie_sim' function. The simulated observations data can then be accessed for analysis as follows:

```
# given a model and prior defined as before:
parameters = rand(prior)
x = gillespie_sim(model, parameters; tmax=max_time, num_obs=↩
    n_observations)
# run analysis and tabulate results:
results = run_inference_analysis(model, prior, x.observations;
    primary=BayesianWorkflows.C_ALG_NM_MBPI, n_particles=12000, ↩
        n_mutations=7)
tabulate_results(results)
```

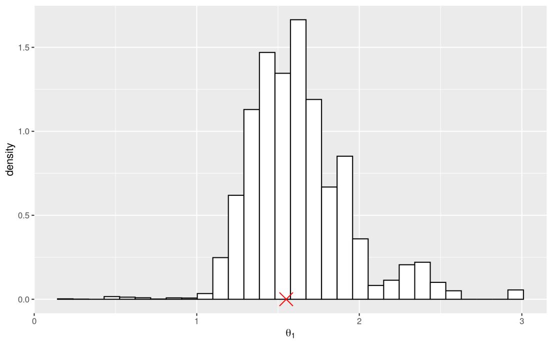
Listing 5.14: Simulated inference analysis.

Note: it is advisable to exercise discretion when selecting simulated scenarios for inference, since not all realisations will be suitable for analysis. In reality, we repeatedly sampled $\pi(\theta)$ and simulated until we obtained a scenario with more than 30 observed infections to conduct this part of the analysis.

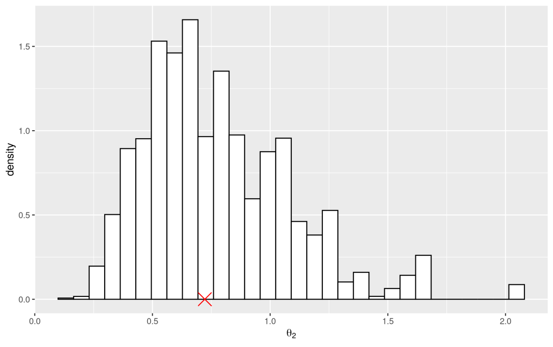
The parameters randomly selected for this analysis (in the manner described above) were $\theta = \{1.5505, 0.7227, 1.0618, 722104.54\}$. The [marginal] posterior estimates obtained are summarised in Figure 5.6, with the tabulated summaries reported in Tables 5.5 and 5.6. The results suggest that the algorithm was reasonably successful in recovering θ as can be seen by comparing them with simulation parameters marked in Figure 5.6.

θ	E[θ]	:σ	-ln p(y)
1	1.63	0.313	36.9
2	0.772	0.312	
3	1.07	0.0752	
4	722103	1.78	

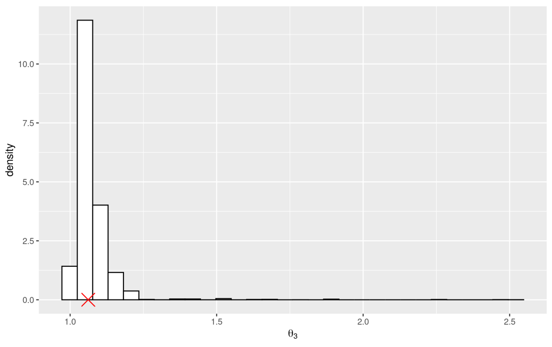
Table 5.5: Summary of parameter estimates (expectations and corresponding standard deviation, denoted σ) obtained using the MBP-IBIS algorithm for the simulated validation analysis. The parameters used to simulate the observations data were $\theta = \beta, \gamma, \phi, t_0 = \{1.55, 0.72, 1.06, 722104.5\}$.



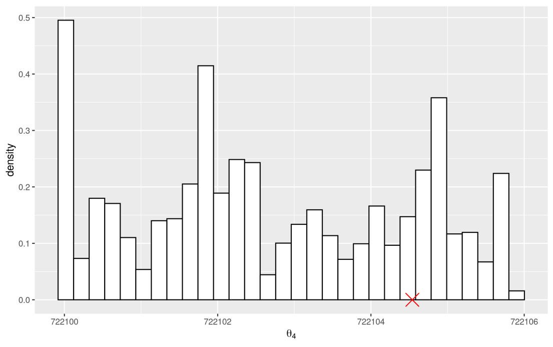
(a) Contact parameter β



(b) Recovery parameter γ



(c) Observation parameter ϕ



(d) Initial time t_0

Figure 5.6: Marginal posterior estimates for model parameters $\theta = \{\beta, \gamma, \phi, t_0\}$ obtained using the MBP-IBIS algorithm for the simulated validation analysis. The estimates used to simulate the observations data were $\theta = \{1.5505, 0.7227, 1.0618, 722104.54\}$ and have been marked for reference.

θ	E[θ]	:σ	E[$f(\theta)$]	:σ	$\hat{R}$	$\hat{R}$97.5
1	1.54	0.82	1.6	0.221	1.21	1.5
2	0.754	0.311	0.788	0.245	1.05	1.09
3	3.11	4.64	1.07	0.0355	1.33	2.23
4	722103	1.75	722103	1.71	1.01	1.03

Table 5.6: Summary of parameter estimates obtained using the ARQ-MCMC algorithm for the simulated validation analysis. The potential scale reduction factors, denoted $\hat{R}$, indicate the convergence was marginal.

5.5 Model comparison

Here we consider both variants of the model in a single analysis, and compare them using the formal Bayesian approach laid out in the previous chapter. In a sense we are therefore inferring the structure of the model itself from the data (i.e. 'model inference'). The two model variants are respectively the one primarily addressed above (termed 'model' below), with observation densities based on the Negative Binomial distribution, and the plain Binomial distribution ('model_b'). To avoid repetition it is assumed that both models have been defined in the manner described above (see the section on Observation Models). It is similarly assumed that an appropriate prior distribution has been chosen. In this case for model_b we have replaced the exponential prior distribution of the Negative Binomial observation parameter with $\phi \sim U(0.4, 1.0)$, where ϕ is the probabilistic quantity that parametrises the corresponding sampling distribution. The inference function is named and invoked exactly as before. The multi-model workflow is invoked automatically based on the input parameters using a vector of models, thusly:

```
# model_b := Binomial [observation] model
# prior_b := prior distribution for model_b
models = [model_b, model]
priors = [prior_b, prior]
# run analysis
results = run_inference_analysis(models, priors, y)
tabulate_results(results)
```

Listing 5.15: Model comparison.

Unlike before, this code sample implies use of the default algorithms and configurations, mainly for the sake of tidyness. However the actual configuration of the analysis carried out in this case was as before (i.e. for the single-model parameter inference). Those results are summarised in Table 5.7.

The results suggest that the data provide strong evidence in favour of the model with observations (specifically, the number of students in bed) distributed according to the Negative Binomial distribution. The estimated marginal likelihood for that model is also broadly comparable with the same estimate given in Table 5.2.

Model	-ln p(y)	BF
SIRfd_nbin	65.0	1.0
SIRfd_bin	80.2	0.0

Table 5.7: Results of the model comparison analysis. The models compared were the same essential [frequency-dependent] SIR model, but different observation models: Negative Binomial ('SIRfd_nbin') and plain Binomial ('SIRfd_bin'). In this case, the difference between the model evidence (given in the form -ln $p(y)$ for ease of analysis) is very large and the Bayes factors (BF) are hardly necessary in order to conclude that the first model is a better fit for data.

5.6 Summary

We have described *BayesianWorkflows.jl*: a package for Bayesian data analysis for DPOMP models in Julia. The software encompasses much of the learning and work developed in previous chapters and provides a framework of methods and tools for application to the motivating problem of the thesis, as addressed in subsequent chapters. That includes methods for simulation; a suite of algorithms for parameter and model inference; and a range of utilities for diagnostics and analysis. In particular, we have made progress towards a low-cost interface for completely generalised access to data-augmented methods for DPOMP models, powered by the feature-rich, functional programming capabilities of the Julia language. To the best of our knowledge this functionality is novel, since as noted in the introduction the applicable software packages that are available tend to rely exclusively on SMC methods. The SIR [state-space] model described here is one of a set of epidemiological models (also the Lotka-Voltera predator-prey model) included with the package as 'predefined' examples.

We express our gratitude in advance for bug reports, other feedback, and especially contributions and improvements to the package itself, from the community we aim to serve.

5.6.1 Source code and documentation

The source code[8] and online package documentation[9] are available on GitHub.

[8]Source code: https://github.com/mjb3/BayesianWorkflows.jl

[9]Package documentation: https://mjb3.github.io/BayesianWorkflows.jl/stable/

Chapter 6

Within-herd parameter estimates

"Infection is increasingly viewed as a continuum of host-pathogen interactions, rather than as the classical binary outcome of active versus latent TB ... and the understanding of latent infection is being revised constantly."

- Skuce *et al.,* [78]

Summary

- This chapter applies selected methods developed in previous chapters to better quantify the the dynamics of within-herd BTB transmission in UK cattle herds.
- We present novel estimates for important BTB parameters, obtained using algorithms, methods and workflows introduced hitherto, with the entirety of the statistical analysis carried out using the software package introduced in Chapter 5.

6.1 Introduction

Bovine tuberculosis (BTB) is a bacterial infectious disease caused by *Mycobacterium bovis*, a slow-growing, nonchromogenic acid fast bacillus [14]. It affects cattle and other mammals including humans, badgers, deer, goats and pigs. In terms of annual financial losses, and as indication with respect to the underlying problem, BTB has been reported to account for estimated of USD 3 billion globally, of which approximately USD 130 million was attributed to the UK in c.2006 [14, 15].

Endemic diseases like bTB present significant statistical modelling challenges due to a number of complicating factors. These include the *possible* role of exogenous reservoirs of infection, for which there is some scientific evidence, but that also remain difficult to characterise, even in terms of the nature (let alone magnitude) of that interaction [79]. The complexity of cattle-to-cattle transmission itself, which is commonly accepted to be the main route of infection, but may be influenced by many factors, including within-herd transmission, spatial location of herds, and live cattle trading [17, 16, 18, 15, 80]. In addition slow replication times associated with growth of *M. Bovis* – approximately 16 to 20 hours. These factors combine to complicate the identification of patterns of infection because a long time may elapse before the presence of infection becomes evident and cases can arise from cattle movements, direct cattle-cattle transmission (within or between herd) and environmental sources.

Furthermore, slow spreading contributes to the highly dispersed (clustered/localised) nature of outbreaks. These difficulties are further exacerbated by the low sensitivity of available diagnostics tests used for standard surveillance [15]. The resultant difficulties in analysis are evidenced in part by the large degree of variation in estimates have been reported in the literature for important epidemiological bTB parameters, even when notionally based on the same [state-space-]model and data [2, 15, 17].

Here we address these difficulties through application of Bayesian methods, workflow and tools presented previous chapters, to directly estimate *within-herd* epidemiological parameters and assess a range of plausible model structures for bTB in UK cattle herds using testing data from observed outbreaks. The key parameters of interest include: the contact or transmission rate; epidemiological parameters that govern within host progression of disease, such the pre-infectious latent period; and the sensitivity and specificity of diagnostic tests. In addition, variation within (e.g. age strata) and between farms should be quantified.

Previous work

We build on previous work that has developed a range of models to describe within-herd bTB disease dynamics, with a focus on the UK. In common with this literature we adopt a categorical approach where individuals are modelled as passing through a series of discrete disease states. It is widely accepted that following exposure to bTB previously susceptible (S state) individuals exhibit a long asymptomatic period represented as an exposed, or E-state that is non-test sensitive and non-infectious. Some studies also recognise an additional test-sensitive state to reflect beliefs that at least some individuals exhibit a detectable immune response, prior to the elevated shed of bacteria that characterises the infectious I state. The test-sensitive state is labelled 'T' in the representation of the model given in Figure 6.1. Elsewhere it has been given equivalent treatment but by slightly different terminology, including the *SORI* model presented by [2], where 'O' and 'R' are labelled for 'occult' and 'reactor' respectively.

$$\begin{tikzcd} \boxed{S} \arrow[r, "\beta SI"] & \boxed{E} \arrow[r, "\gamma_1 E"] & \boxed{T} \arrow[r, "\gamma_2 I"] & \boxed{I} \end{tikzcd}$$

Figure 6.1: SETI model for bovine TB. It is predicated on a similar basis to that of the classic SIR model. This variant includes a test-sensitive 'T' state, which accounts for a non-infectious phase of disease, in which individuals are deemed to exhibit a [sometimes] detectable immune response to standardly used diagnostic tests. As before, individuals take a value of the discrete state vector, e.g. $\{S, E, T, I\}$, and transition between states according to the given event rates. No recovery 'R' state (or removal process) has been accounted for in this version.

As illustrated by such differences in terminology and perhaps driven by the inherently interdisciplinary nature of the topic, previous modelling studies have been presented in a variety of ways that make it difficult to compare results. A key issue is the manner in which models are fitted to data. Many studies use some form of aggregate data, or other meta-characteristics of observed patterns of disease at the aggregate level. These are sometimes combined with expert knowledge, assumptions or constraints, usually justified by reference to a commonly accepted scientific belief. The underlying fitting procedures also vary widely, from the relatively informal:

> *"This was achieved [i.e. parameter estimates were obtained] by varying beta until predicted outputs of reactor rates under simple annual testing regimens* –and *corresponded to those observed in four [cattle-herd 'history'] data sets"* [81],

similar descriptors, such as 'spreadsheet-model' [82], to more formal mathematical constructs, such as the Approximate Bayesian Computation scheme used by O'Hare *et al.,*[17] and Brooks-Pollock *et al.,*[15].

The latter are more readily comparable with the (full or exact) Bayesian inferential framework used here. O'Hare *et al.,* use an arbitrary grouping of ten thousand herds to represent a national system of herds, with migration (e.g. trade) and other population dynamics. The simulation was used to numerically solve quantities that are readily comparable with the data set analysed (the same one analysed here). Those quantities include, for example, the number of reactors who indicated (tested positive) during the first non-clear whole-herd-test. Another recent set of estimates, that rely on both discrete-time simulation and ABC-SMC methods, are those published by Brooks-Pollock *et al.,*[15]. In this case the 'simulation-model' explicitly represented the system of farms in the UK. Posterior estimates of [frequency-dependent] within-herd epidemiological parameters were also reported such as contact parameter $E(\beta) = 0.61$ (95% CI: $0.05 - 1.5$). The authors also explored the likely impact of potential control measures, such as whole-herd culling, and additional or improved testing.

There is a large degree of variance in the parameter estimates obtained in the literature. For example, the estimates for the average latent period –which is implicitly defined by the inferred discrete-state boundaries of the model with respect to the [continuous] real-world disease process– range considerably in literature. Brooks-Pollock, Brooks-Pollock *et al.*, [15] estimate the quantity to be 11.1 years. Others, such as [2] report estimates of as little as two days for the corresponding model (labelled SEI and SOR respectively in each paper). The latter provide an estimate for the SETI/SORI model of $\sim$275 days, which is more in line with that estimated by [17]. The precise numbers are not given for the latter, though we do note that the prior distribution was chosen such that upper bound the entire latent period (including the test-sensitive phase) was approximately one year in any case. Judging by the commentary provided with these papers, the uncertainty surrounding key parameter values for within-herd modelling of BTB is well recognised. For example:

> "There exists considerable uncertainty in the assumed values of key parameters, in particular the occult period, the scaling of transmission rates with herd size [83, 84] and the duration of latency between infection and infectiousness." [2]

Alternatively, *"the natural pathology ... and infectiousness [of BTB] is poorly de-*

fined" [15]. Since the precise definition of these underlying quantities is somewhat fluid[1] –meaning model and data-dependent– this variation is actually not that surprising. However it *could* indicate that the underlying biological process(es) bear little relation to the corresponding mathematical constructs that represent them in epidemiological models of BTB. For example, Brooks-Pollock *et al.,* [15] propose [possible] heterogeneity of transmission as a possible explanation for the 'exceedingly long' estimate they obtained w.r.t. the latent period of 11.1 years (see supplementary materials of [15]). Taken together this provides a strong motivation for going beyond previous attempts to model within-herd transmission of BTB by:

- Making better use of the available BTB surveillance data, e.g. by incorporating within-herd [reactor] arrival times as well as count data.
- Employing more robust [i.e. formal Bayesian] approaches to model validation and comparison.

Exploiting data complexity for better inference

All the examples above rely on model fitting using *distance-based approximation* methods, which tend to rely on *aggregate* measures of disease-spread[2]. When the observation measures are discrete (often referred to as 'count data'), this approach is sometimes described as: 'histogram comparison'[85].

Here we adopt an alternative approach that focusses model fitting on individual herds in order to deal more directly with heterogeneity in the data, making use of the versatility of Bayesian methods (Chapter 2) to *directly* infer epidemiological parameter estimates. This approach preserves more of the information content of the data and is therefore potentially able to extract more meaningful information than estimates based on aggregated data. However, it should be noted that this is also more likely to reveal deficiencies in models. We therefore treat a range of models (described in § 6.2.1) within a consistent Bayesian framework including the comparison of structural difference using model evidence. This is in the spirit of the

[1]Strictly with respect to a statistical modelling paradigm, rather than corresponding clinical definitions.

[2]Michael Betancourt of Warwick University commented in a lecture series on Hamiltonian MCMC https://www.youtube.com/watch?v=pHsuIaPbNbY that a popular approach to dealing with data sets of challenging size and complexity ('big data') is to aggregate them, such that all individual samples are still used, albeit in aggregated form.

approach advocated by Betancourt [3] for complicated or scientific domain models, that can be thought of as using *all*, or as much as possible, of a 'slice' of the data, instead.

Chapter outline

We first present a series of possible models to describe the within-herd disease dynamics of bTB (section §6.2.1). The Bayesian inference framework relevant to fitting and comparing such models and its implementation is presented in Section §6.2.2. The BTB data to which these models are applied is described in Section §6.2.3. Firstly, the data sources used including *VetNet* are outlined, the main challenges encountered in their analysis are detailed and approaches for resolving these along with validation steps are described. Inference methods are then validated in Section 6.3 by application to simulated data generated from models described in §6.2.1. This validation also enables exploration of what inferences might be possible when using data on outbreaks from individual herds and show that inference is possible, but that precise parameter estimation requires observation of a sufficiently large outbreak. However, we also show that in such cases it is possible to correctly distinguish between different model structures.

Section 6.5 describes application of these models and inference tools to data from real outbreaks. Finally we present within-herd epidemiological model and parameter inference results based on selected herds from the *VetNet* dataset for cases where significant outbreaks were observed. Inference was conducted for a range of different models and configurations (e.g. *frequency* vs. *density*-dependant). The results reveal that even when conditioning on herds that experience sizeable outbreaks there is considerable heterogeneity between herds. The significance of these results and the potential for future application and extension of the methods presented is discussed in §6.6. In particular we point to the desirability of hierarchical approaches that could allow fitting of multiple individual herd models in a way that allows for the observed heterogeneity in parameters.

[3]Michael Betancourt of Warwick University in a lecture series on Hamiltonian MCMC: https://www.youtube.com/watch?v=pHsuIaPbNbY

6.2 Methods

6.2.1 Within-herd models for bTB dynamics

Here we describe a series of state-space models for within herd dynamics of bTB. The intent is to unify them with the paradigm and methods for epidemiological models laid out in Chapter 2. Specifically, discrete-state-space, partially observed Markov processes (DPOMPs). The mathematical definitions that unify this paradigm with the event rates denoted for each model are given in §6.2.2. Following an overview of each model and their respective parameterisation we discuss potential uses and interpretation.

However, first we describe a common observation model which describes how infected animals are detected for each of the state-space-models presented.

Observation model

In general, the models are parametrised by $n = |\theta|$ distinct values in the vector of unknown model parameters θ that govern the dynamics of the model including the observations. The observation model (or equivalently, process) is functionally denoted in the diagrams below by $g_\theta(N)$, where N is the number of *test-sensitive* individuals and varies throughout the course of an outbreak. Furthermore which disease states are considered to be test-sensitive is model dependent e.g. in Figure 6.2 both the test sensitive state T and the infectious state I can be detected by diagnostic tests i.e. $N = T + I$ (for a given type or specification of diagnostic test).

It is assumed that each test type is associated with a probability of detection listed in the table of parameters for each model, e.g. Table 6.1. Later these will be inferred along with the other model parameters (as discussed further in §6.2.2). The number of individuals that test positive is therefore described by the following Binomial distribution:

$$g_\theta(N) = B(N, p^* \sigma_T) \tag{6.1}$$

where σ_T is the [inferred] probabilistic quantity that (loosely) represents the sensitivity of test type T. In many cases, not all members of the herd are tested. The probability of being selected for a test p^* is introduced to represent the proportion of the herd that were tested. This may vary from one testing event to the next and between test types. Such information can be computed directly from the available data (see §6.2.3). Note that in the parameter tables SICCT corresponds to the main test type that occurs in the data: the single intradermal comparative cervical tuberculin test [86].

Within the models, testing operations are implemented as discrete events in time where a proportion of the herd is tested with test-postive animals removed. Thus it is assumed that positive detection of an individual results in removal from the herd ('recovery' in standard epidemiological terms). These assumptions are grounded in the common practice of 'culling' infected animals, though they disregard to some extent the operational complexity that no doubt surrounds the corresponding real-world process. For example, in the case of inconclusive tests, or the realistic notion that commercial considerations, farmer attitudes and risk appetite may influence e.g. pre-emptive culling.

Test-determined removal models (SEI tdr/SETI tdr)

The test-determined removal model presented in Figure 6.2 is easily recognisable as an elaboration of the standard SEIR model. A summary of the model's parameters is given in Table 6.1. Although each of the models described below share the share 'test-determined removal', this one is so-named because it is its only distinguishing feature. It can be written in frequency or density-dependent form (see Chapter 2). Figure 6.2 shows the case of density dependent transmission with a test sensitive state, but each of the four possible configurations (with or without state T and with density or frequency dependent transmission) are evaluated against the data for the second set of results presented in §6.5.

$$\begin{tikzcd}
\boxed{\text{S}} \arrow[r, "\beta\text{SI}"'] & \boxed{\text{E}} \arrow[r, "\gamma_T\text{E}"'] & \boxed{\text{T}} \arrow[r, "\gamma_I\text{T}"'] \arrow[rr, bend left, "g_\theta(\text{T})"] & \boxed{\text{I}} \arrow[r, "g_\theta(\text{I})"] & \boxed{\text{R}}
\end{tikzcd}$$

Figure 6.2: Test-determined removal (or 'recovery') SETI-tdR model. The *observation process* is denoted by g, and is a function of both the test sensitivity and the applicable subpopulation size. The standard epidemiological rates and parameters are given for the *density-dependent* configuration of the model. These determine infection; and progression to test-sensitivity; and to full infectiousness.

θ_i	Symbol	Description	Prior: U_a	U_b
1	β	Infection (density dependent)	0.0	0.01
1	β	Infection (frequency dependent)	0.0	0.1
2	γ_T	Progression to T	0.0	0.1
3	γ_I	Progression to I	0.0	0.1
4	t_0	Time of initial onset	-360.0	0.0
5	σ_S	SICCT test sensitivity – standard	0.3	1.0
6	σ_H	SICCT test sensitivity – high	0.3	1.0
7	σ_γ	IFN$_\gamma$ test sensitivity	0.3	1.0

Table 6.1: Test-determined removal model parameters.

Reinfection model with 'super infection'

As discussed in §2.1.2, there is arguably an intrinsic trade-off between accounting for the *full extent* of infection [within a population,] and *reinfection* within commonly used epidemiological model formulations. Here we speculate that this may provide a partial explanation for the large amount of variation in estimates for notionally similar quantities (i.e. estimates of similar epidemiological parameters for BTB) reported by others, as referenced in that part of the Introduction. In order to test the supposition, we utilise the model proposed by [6] (and another partially inspired by it) that accounts for more complex disease dynamics, including *reinfection* and also *'super infection'*. While we find this to be an interesting avenue of investigation, we also note with some regret that the results of that analysis (i.e. as reported subsequently in this chapter) were mostly inconclusive due to our own error (as discussed in due course).

With respect to the aforementioned *reinfection* model, the formulation specified by equations (6.2) through (6.5) is inspired by that of the full model proposed by the original authors (including notation). The state space of the model is denoted by (S, E, Y, Z) for susceptible; exposed; infectious; and recovered. Note that the latter is distinct from the removed state which is denoted for each of the models by 'R'.

$$\frac{\delta S}{\delta t} = \phi - (1 + p')\eta S - \mu S \tag{6.2}$$

$$\frac{\delta E}{\delta t} = \eta S + q\eta Z - p\eta Z - (\mu + \gamma)E \tag{6.3}$$

$$\frac{\delta Y}{\delta t} = p'\eta S + q'\eta Z + p\eta E - (\mu + \delta + \alpha)Y \tag{6.4}$$

$$\frac{\delta Z}{\delta t} = \delta Y - (q + q')\eta Z - \mu Z \tag{6.5}$$

The notation here is as per Table 6.2. The authors noted that this specification is somewhat intractable to [direct] mathematical analysis, and therefore used a simpler construction as the basis for their own analysis.

The original intent in this chapter was to stay with (almost) the full definition of the model for the discrete-state-space variant employed here. Since we are dealing in this case with a situation where population demographics are largely known[4], we effectively set those parameters (denoted by $\{\phi, \mu, \alpha\}) = 0$. Somewhat equivalently, we disregard the assumption used by the original authors in their analysis, that the population size remains constant throughout.

Note: human error led to primary disease transmission being effectively omitted from the model. That is, infectious individuals are (somewhat paradoxically!) decoupled from the disease transmission process. The model as described here; depicted as an SSM in Figure 6.3; and its parametrisation as enumerated in Table 6.2, are therefore specified according to the results that were actually obtained (and thus not an accurate reflection of the original author's model). We acknowledge the error and thank the examiners of the thesis (see Acknowledgements) for bringing it to our attention.

[4]Thanks to the corresponding *CTS* data that was available for this work.

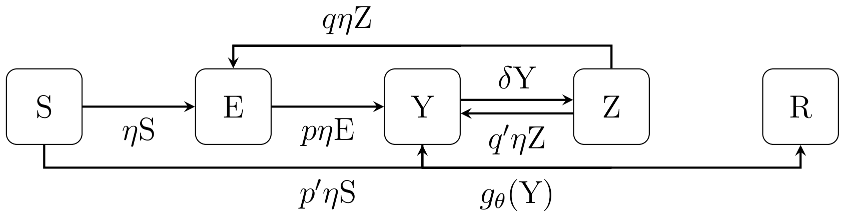


Figure 6.3: Reinfection model based on that proposed by [6]. It was derived from the system of equations described by the authors. Note that this model includes both a 'recovered' state (Z), and a 'removed' state, R. The *observation process* is denoted by g, as before. Progression directly from $S \to Y$ is termed 'super-infection' by the authors of the original model.

Alternative reinfection model

The final model presented has been devised as a blend of the first two approaches. That is, it is similar to the standard formulation whilst attempting to nonetheless account for reinfection as a biological process. In other words, unlike in the standard model it is assumed here that the presence of individuals in the infectious state 'excites' (i.e. increases) the $E \to I$ event rate. Equivalently, it reduces the [average] latent period. This is best illustrated by Figure 6.4 (as before, the model parameters are also summarised, in Table 6.3).

The results from the corresponding analysis are reported alongside the others in §6.5.

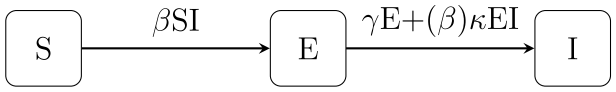


Figure 6.4: Poisson [point] process reinfection model. The compound rate for $E \to I$ incorporates the possibility of reinfection. This is intended to represent the latter as simply as possible and incorporates only two types of event: infection and progression, which is excited by reinfectivity.

6.2.2 Bayesian inference

Recall from Chapter 2 that inference for continuous-time SSM can be understood as *partially observed Markov processes* where the latent and observation processes are

θ_i	Symbol	Description	Prior: U_a	U_b
1	η	Infection	0.000	0.001
2	δ	Recovery	0.000	0.001
3	p'	Super infection	0.0	1.0
4	p	Progression	0.0	1.0
5	q	Loss of temporary immunity	0.0	1.0
6	q'	Remission	0.0	1.0
7	t_0	Time of initial onset	-360.0	0.0
8	σ_S	SICCT test sensitivity – standard	0.3	1.0
9	σ_H	SICCT test sensitivity – high	0.3	1.0
10	σ_γ	IFN$_\gamma$ test sensitivity	0.3	1.0

Table 6.2: Reinfection model parameters.

treated as random variables. That is they are distributed according to the following probability distributions:

$$X|(\Theta = \theta) \sim f_\theta(x) \tag{6.6}$$

$$Y|(X = x, \Theta = \theta) \sim g_\theta(y|x) \tag{6.7}$$

where x represents the latent process; y the observed process (or data) and '$\sim$' means 'distributed according to'. The probability densities associated with each are functionally denoted as f_θ and g_θ. The first is the one associated with [inhomogeneous Poisson point] process described in Chapter 2 and represents the underlying (process) model of disease dynamics being used for the analysis such as those defined in the previous section. The second is the observation models also defined in the previous section. For notational simplicity the vector of $\mathbb{R}$-valued model parameters, θ contains the parameters of both the observation and process models.

Parameter inference

The goal of the initial [single-model] analysis is to infer the *joint [posterior] distribution* of θ and x, conditional on the data y. The likelihood is written as before, as:

$$\pi(\theta, x|y) = \frac{\pi(\theta)\pi(x|\theta)\pi(y_{1:n}|x, \theta)}{\pi(y_{1:n})} \tag{6.8}$$

θ_i	Symbol	Description	Prior: U_a	U_b
1	t_0	Time of initial onset	-360.0	0.0
2	σ_S	SICCT test sensitivity – standard	0.3	1.0
3	σ_H	SICCT test sensitivity – high	0.3	1.0
4	σ_γ	IFN$_\gamma$ test sensitivity	0.3	1.0
5	β	Infection	0.0	0.01
6	γ	Progression	0.0	0.1
7	κ	Reinfection scalar	0.0	10.0

Table 6.3: Alternative reinfection model parameters.

where $\pi(\theta)$ is the prior probability density; $\pi(y_{1:n})$ is the *marginal likelihood*, running over $1:n$ [longitudinal] observations. The remaining terms are defined by:

$$\pi(x|\theta) = f_\theta(x) = \prod_{i=1}^{n} f_\theta(x_i|x_{i-1}) \tag{6.9}$$

$$\pi(y_{1:n}|x,\theta) = g_\theta(y_{1:n}|x) = \prod_{i=1}^{n} g_\theta(y_i|x) \tag{6.10}$$

Note once again that, for practical reasons, it make sense to partition x at intervals, so as to align with the each measurement (or 'observation period') where y_i is the i^th measurement.

The marginal likelihood is obtained by solving the [double] integral:

$$\pi(y) = \int \pi(\theta)\pi(y_{1:n}|x_{1:n},\theta)\pi(x_{1:n}|\theta)dx_{1:n}d\theta \tag{6.11}$$

In this case it is solved by application of the chain rule:

$$\pi(y) = \pi(y_{1:n}) = \prod_{i=1}^{n} \pi(y_i|y_{1:i-1}) \tag{6.12}$$

where the quantity $\pi(y_i|.)$ (and by extension $\pi(y)$) is estimated using the SMC^2 algorithm, as discussed in the section on implementation. The purpose of this exercise is that the marginal likelihood can be utilised as a measure of model fit, as discussed next.

Model comparison

As noted, (6.11) provides an empirical measure of how well a [single] model fits the data. In keeping with the Bayesian framework laid out in §2.3.2, we can also utilise the *marginal likelihood* for the problem of model selection (or 'model comparison').

By application of Bayes' theorem, we write an expression for the different choices of model available, as follows:

$$\pi(y, M_i) = \int \pi(M_i, \theta)\pi(y_{1:n}|x_{1:n}, \theta, M_i)\pi(x_{1:n}|\theta, M_i)dx_{1:n}d\theta \tag{6.13}$$

where M_i is the i^{th} model chosen and $\pi(M_i, \theta)$ is the corresponding prior distribution, i.e. including the probability of selecting that model. The Bayes factor provides a standardised way to directly compare these quantities for two candidate models:

$$K_{1,2} = \frac{p(y|m_1)}{p(y|m_2)} \tag{6.14}$$

where, according to the scale originally proposed by Jeffreys [65], $K_{1,2} > 10$ can be considered to be strong evidence for favouring model m_1 over m_2.

$\log_{10}K$	K	Strength of evidence
0 to 1/2	1 to 3.2	Not worth more than a bare mention
1/2 to 1	3.2 to 10	Substantial
1 to 2	10 to 100	Strong
>2	>100	Decisive

Table 6.4: Bayes factor interpretation, from Kass and Raftery[1].

Bayesian model averaging

One alternative to comparing *between* models is that of is that of *Bayesian model averaging* [69], which essentially circumvents the problem of model selection altogether by providing a weighted average *across* models.

For example, let θ_α denote some common parameter across models (e.g. the 'latent period' in competing disease models). The posterior distribution of θ_α given data y is:

$$\pi(\theta_\alpha|y) = \sum_{i=1}^{n} \pi(\theta_\alpha|M_i, y)\pi(M_i|y) \tag{6.15}$$

where the M_i denotes the i^{th} of n competing models. This approach is self-evidently suited only to models that have important quantities in common, in particular nested models.

We now discuss the implementation of this overarching methodology.

Implementation

The analyses presented in this chapter were carried out in accordance with the Bayesian workflow already laid out in Chapter 4. The configuration of the workflow (i.e. the selection of algorithms) that was used for the main results is given in Figure 6.5. This was found to be the most efficient configuration of the workflow to implement [of the available options], because the same [model-specific] particle filter could be used for both the primary and validation analyses. The 'outer' construction of both algorithms is completely generalised, such that the particle filter was the only component that required manual development effort.

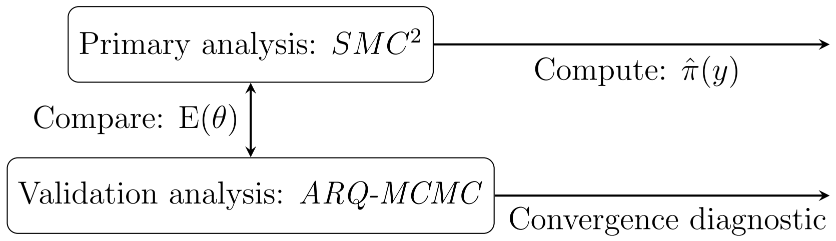

Primary analysis: SMC^2
Compute: $\hat{\pi}(y)$
Compare: $\mathrm{E}(\theta)$
Validation analysis: *ARQ-MCMC*
Convergence diagnostic

Figure 6.5: A depiction of the modular workflow and algorithms utilised for Bayesian inference in this analysis. (The workflow itself is described in Chapter 4.)

6.2.3 Data

Overview

The data available for analysis consisted of routine and incident-related BTB surveillance data from the *VetNet* database. In addition records concerning demography (births; deaths; migration, e.g. trade) were available for [partially] corresponding records from the *CTS* database.

When it comes to the task of performing statistical inference, large operational data sets like *VetNet* pose distinct challenges compared to those collected under controlled experimental conditions. Procedures, legislation and even jurisdiction, relating to the original collection of the data may be expected to vary with time, region, and perhaps down to the level of individual personnel who collected/entered the data originally. Such factors likely account for a certain degree of inconsistency that was observed in especially the *VetNet* data set, that has no ready explanation and usually no obvious systematic solution. This was especially apparent for the Herd Size column as discussed in §6.2.3.

In general though, constraints imposed by the methods were such that it was practical to first fit a relatively small number of identified (seemingly) idealised cases. These are characterised by a reasonably large number of reactors over time, but also other considerations such as having at least one reliable estimate for the herd size. The selection criteria used are laid out in more detail in §6.2.3. Another factor in the *selection* of data sets to analyse that became evident later, throughout the course of the work, was the nature of diagnostic testing. In particular, it was found to be helpful to have two or more distinct variants of diagnostic tests used. Matters related to testing generally are discussed further in the first subsection.

Many minor data quality issues also proved to be entirely surmountable. For example, variations in the test codes recorded (used to both inform the test type, and to provide a guide when sense checking the recorded facts of a given scenario) were coped with by mapping, where possible. In some cases, it was preferable to infer the 'correct' test code based on the number of animals tested compared as a ratio of [estimated] herd size. Care had to be taken with the test dates recorded, since many apparent errors like 'duplicate' records were in fact merely herd tests recorded (and presumably carried out) in partial lots [5]. Here again, knowledge of herd size was important to ensuring that the data made sense, and none of the scenarios selected defied common sense. In some cases the data merely presented overwhelming complexity, such that the true order of events was difficult to establish let alone verify.

Data processing: technical implementation

Data processing was mostly carried out using SQLite on a Linux workstation. The SQLite.jl package was used to interface with the file-based database held on BioSS's secure network drive. This ensured that only the data required for the actual analysis was queried (i.e. retrieved from the database). Furthermore, since a signif-

[5]This practice was confirmed verbally by consultation with academics at SRUC.

icant amount of the initial processing was carried out using SQL, the data were summarised and mostly cleaned of superfluous data at the point of use. Other more complex data processing tasks were generally carried out using the Julia programming language. That is somewhat outside its primary intended usage as a mathematical and statistical language, but it is sufficiently feature rich to allow for control flow-based data processing. Coding in the same language also meant that the data processing layer was more easily integrated with the main model code that depended on it.

Testing practices

'Test' records within *VetNet* pertain to the operational business of the surveillance regime itself, and thus include much that is superfluous, such as 'breakdowns' in herd status that do not relate directly to disease incidence, but may occur as the result of minor administrative matters. For example, some records in the herd test table simply appear to indicate that correspondence has been issued to provide notification that testing is overdue.

The first task was therefore to isolate the records pertinent to diagnostic testing, and group these [test codes] according to perceived overall type, and purpose. Most of the test records relate to single intradermal comparative cervical tuberculin (SICCT) testing, with one important exception. All of them are based on the host's response to bovine purified protein derivative [86]. The tests types are categorised as follows:

1. Routine: routine herd SICCT tests, usually marked RHT or WHT for routine- or whole-herd test, respectively.

2. Trade: also SICCT and, in a sense, routine; the records of most interest are probably those that pertain to pre-movement testing for the purposes of trade.

3. Risk: conducted based on the perception of elevated disease risk. These include non-routine trade-related ('tracing') tests; 'contiguous' testing in herds local or adjacent to infected herds; and short-interval tests of various kinds, typically conducted at higher sensitivity than the routine SICCT test.

4. IFG: Interferon-γ blood test. This is used in some circumstances to supplement the above. It is designed to measure a cytokine (whereas the SICCT

test detects a delayed hypersensitivity reaction) [87].

Interpretation of test type codes (and dates)

The main records of interest in $VetNet$ pertain to two broad types of test, including the interferon-γ blood test as noted above. The other, the SICCT, is the standardly used diagnostic, but has two distinct protocols associated with it: two different sensitivities (interpretations) are used under different circumstances. Together they are the 'Routine' and 'Risk' family of test codes, also as noted above. Most Routine herd tests are recorded in a single row. However some may be conducted over several days –and correspondingly, recorded over several rows– yet considered a single herd test. According to the documentation provided with the dataset, there are two possible scenarios and corresponding sets of [possible] values in the relevant column, *'vtPart'*:

1. A 'complete' test, indicated by a 'C'.
2. 'Partial' testing carried out over N days, i.e. $P_1 \rightarrow ... \rightarrow P_{N-1} \rightarrow C_N$.

A number of additional possibilities were discovered in the data itself:

1. Orphaned partial test records, treated as complete tests: $P_N \approx C$, $C_N \approx C$.
2. The completion of partial testing marked with 'C': $P_1 \rightarrow ... \rightarrow P_{N-1} \rightarrow C$.
3. Disordered combinations of the above, e.g. $P_1 \rightarrow ... \rightarrow C \rightarrow P_x$ where $x < N$.
4. Other irregular partial test records, e.g. $P_1 \rightarrow P_1 \rightarrow C$.

In the end, it was determined that for the purposes of analysis consecutive rows in the test records table with similar[6] dates should simply be grouped, such that this ambiguity was generally less of a problem.

Spatial data

Interfarm distances were approximated based on the *VetNet* dataset in conjunction with the Ordnance Survey National Grid (OSNG) system[7]. However it eventually transpired they were not required for (or included in) the results presented

[6]Less than 14 days difference.

[7]https://en.wikipedia.org/wiki/Ordnance_Survey_National_Grid

here. That is because the methods were not found to be workable at scale, i.e. the problem that essentially provides the motivation for the work presented in Chapter 7. Thus there was little justification in attempting to account for inter-farm disease transmission in a comprehensive way, that may have called for the further consideration of spatial data.

Herd size

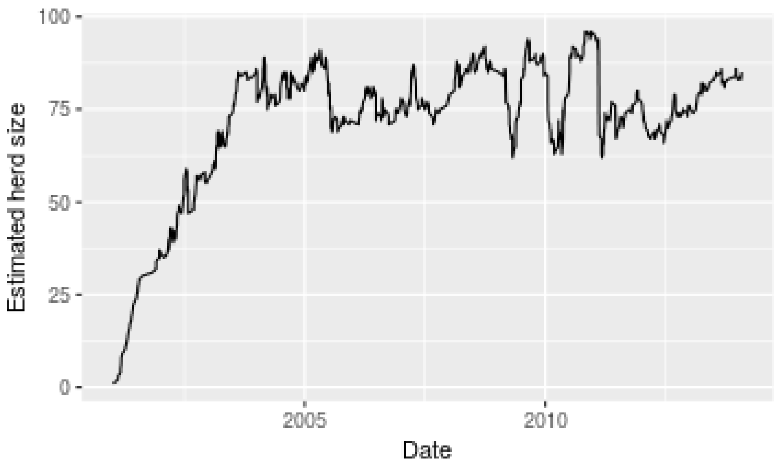
Somewhat surprisingly population [i.e. herd] size was found to be the most challenging aspect of data quality to account for. However, it was necessary to obtain accurate, or at least consistent, estimates of herd size as this has significant impact on disease dynamics within the models of interest. In the raw *VetNet* data, the herd size column was often found to be populated with values that were unrealistically low, or that sometimes conspicuously matched those that merely pertained to the number of animals actually tested. This was found to be the case even in records that related to pre-movement testing, where it would seem reasonable to expect that considerably less than the full herd would usually be subject to a diagnostic test. In general, there is considerable variance in how herd size at the time of testing is reported. For some records the *vtHerdSize* column is populated in way that corresponds with the routine testing scenarios laid out in the previous subsection; with the same value used to populate each linked record. This appears to correspond to a herd test carried out over several days. In other cases the value apparently reflects a subset of the herd, presumably intended to nonetheless represent the same underlying scenario. As noted above, this kind of variance (i.e. pertaining to reporting procedures and data accuracy) was anticipated due to the nature of the dataset, but it was nevertheless necessary to at least approximate the herd size with a reasonable degree of accuracy, in order for a meaningful analysis to proceed. Initially, a simple SQL query was used to compute the average herd size reported for all[8] routine tests over the reported period of interest. These approximations were found to be mostly reasonable, when a subset were compared using the validation method described below.

Herd size validation

It is also possible to estimate average herd size over certain periods using *CTS* data alone. All births, deaths and migration are recorded, so a running total of the net

[8]Except private tests, which were found to be unreliable indicators of the true herd size.

balance converges on the true population size for later records. The principle is illustrated in Figure 6.6. This is evidently only effective for a subset for records, since only later estimates are reasonably likely to incorporate records pertinent to the entire herd. However it was sufficient to provide a degree of comfort, with respect to the accuracy of the approach described above, i.e. the SQL query. One complication that limits this approach to being a validation method, is that *CTS* records are stored at the 'holding' (i.e. farm) level. The pertinent [test] records in *VetNet*, however, are given at the herd-level. The implication is that the method is only valid as a herd-size estimator for holdings that maintain only a single herd. This can be a complicated distinction to make in some cases. The number of herds maintained is not directly recorded in either *VetNet* or *CTS*. Instead it must be inferred from corresponding test records, and in any case, could reasonably be expected to change over time. Otherwise, it might have proved easier to simply limit the data analysed to single-herd farms for which reasonably complete *CTS* records were available.



(a) herd size estimated from *CTS* movement data.

Figure 6.6: Migration data (transactional records from *CTS*) allow for the calculation of a quantity that can be expected to converge over time with the true [holding-wide] herd size. Though the approach can account for even small fluctuations in the population size over the latter portion of that period, its usefulness here was mainly as a means to (approximately) corroborate the herd size recorded in *VetNet*.

Another option that was considered but not pursued, was to group the VetNet records themselves at the holding level, so as to better marry the records in each database. This crude approach has an obvious limitation however. It would implicitly assume the free mixing of individuals *between* herds. It is obviously plausible, perhaps even likely, that disease transmission occurs between herds on the same farm. However in a model this would arguably be better-characterised as a separate process, from those that drive *within-herd* disease dynamics – which is the main focus here. In the end it was not clear that this method would have been less damaging to the objectivity of the analysis, than the [*VetNet* data] interpolation approaches used instead, and described below. The *CTS* validation analyses –which are essentially based on visual comparison of a trend over time– are given for each of the 'individual herd scenarios' presented in §6.4.

Herd size imputation

Issues with the *VetNet* data remained following the pre-processing (e.g. [date-based] grouping) and validation steps described above. The most significant can be (roughly) categorised as follows:

1. Missing data (i.e. null values) – mostly relating to herd size.
2. Implausibly low herd size values for some records – e.g. the issue described above, where in reality the 'herd size' column merely duplicates the data in the 'animals tested' column – that were not resolved by earlier steps in pre-processing (primarily the grouping of rows).

Both classes of problem were resolved in essentially the same manner; missing or suspect values were interpolated based on similar records, and replaced. The first class of problem was often found to affect tests which are routinely given only to a few individuals at a time, for example pre or post-movement testing. These were fairly simple to resolve: the *proportion* of the herd tested was assumed to be similar to that of other records of the same type, based on the test-type column in *VetNet*. More specifically, it was assumed for cases with missing and certain types of suspect data, that the proportion of the herd tested for any given test type matched the average proportion for that type of test. The imputed herd size value can be understood here as a function of: the number of cattle tested, and the [imputed] proportion of the herd tested. This step was carried using short SQL statements generated by script. The following snippet illustrates the approach for pre-movement test records specifically:

```sql
update batch_test_pc
set prop = (select avg(prop)
from batch_test_pc
where test_type = 'VE-PRMT'
and n_vtHerdsize > n_vtTested)
where test_type = 'VE-PRMT'
and n_vtHerdsize <= n_vtTested;
```

Listing 6.1: Example SQL statement: herd size imputation.

The approach is simplistic, at least compared to methods such as 'Bayesian simulation'[88], which involves sampling missing values from the posterior distribution of complete cases. That might have been preferable particularly if more than one column had been used to derive the imputed data. However the information encoded by the test type code column alone was deemed to be sufficient in this case, since it contains over sixty distinct values pertinent to this situation. In most cases the operational usage of different codes was found to be evenly distributed enough, that sufficient samples could be obtained to estimate the average for each individual code. An exception to this was import-test records, which were grouped for this purpose as {'VE-PII','VE-PIO'}. The codes themselves seem to correlate with both the purpose and circumstances in which, e.g. routine herd testing, is carried out. For example, codes that include the character strings 'RHT' and 'WHT'[9], generally seem to relate to four-yearly and one-yearly herd tests, respectively. It is a useful distinction for this purpose, because eligibility criteria for the testing of individual animals may vary according to the surveillance regime, with fewer animals eligible in areas deemed to be at low risk.

The second broad problem described –implausibly low herd size values– was more difficult to resolve. In general terms, it was less obvious in some cases that the data were actually incorrect, rather than just suspect. This was particularly an issue for interferon-γ blood test records. That is largely because their usage did not seem to conform to regular patterns of behaviour evident with other types of test record, reinforced by well-established surveillance regimes. This is unsurprising due

[9]These stand for 'Routine' and 'Whole'-herd test respectively.

to their association non-routine surveillance-testing. In the end a compromise had to be determined, which balanced the need to ensure that the data were plausible, against the risk of replacing accurate data with crude approximations – in this case the average herd size, computed as described above. It was therefore decided that all remaining test records which purportedly recorded both 100% coverage (i.e. the proportion of the herd tested = 1) and also indicated less than $\frac{1}{3}$ of the *average* herd size [over time] were actually tested, would be interpolated. This was achieved simply by recomputing the proportion of the herd tested based on the average herd size as recorded in other (generally more reliable) test records over the pertinent time period – somewhat as described at the beginning of this sub-section for pre-movement test records. A significant number of rows –over fourteen thousand– were affected by this step, unfortunately indicating that many other problematic records may exist beyond the chosen threshold of $\frac{1}{3}$.

Summary of limitations

In particular, perusal of the data suggests that we might expect the final analysis to under-estimate the sensitivity of the interferon-γ test in some cases. That is because some records are likely to indicate that the whole herd was tested, when in reality it might have been as low as [slightly more than] a third. The speculation was reinforced by casual inspection of the data, which *appeared* to suggest that the interferon-γ test records may be particularly affected. Overall though, the approach taken for data processing and validation was conservative, in that adjustments to the underlying data were only made when judged necessary, and with reasonable attempts made to cross-validate where possible.

Data selection

Notwithstanding the steps described above, it was still necessary to filter the data set in order identify cases that were truly suitable for analysis. This is discussed further in §6.2.3. The general problem of systematic bias introduced by these steps; the wider implications; and a possible solution, are also discussed further in 7.

It was noted during the early phases of the project, that individual within-herd analyses were only feasible for reasonably large epidemics. That is, it was only possible to recover reasonable estimates of key parameters under those conditions. For intuitive reasons, this was especially the case when only weakly informative prior distributions were used. Criteria for case selection were iteratively refined during the validation phase, with the final set chosen as:

1. At least ten reactors,
2. spread over six or more [positive] herd tests,
3. where [negative] herd test records cover at least the preceding twelve months, from the first positive test.

It was also found –as demonstrated by the validation analysis illustrated in Figure 6.7– that better estimates could be obtained when more than one different type (or configuration) of diagnostic test was used over the course of an epidemic. Accordingly, the criteria were refined to include only cases where there was at least one reactor per each of the three tests described at the top of §6.2.3.

6.3 Validation: Testing inference with simulated scenarios

The applied methods were first validated using simulation methods, i.e. within-herd epidemics were simulated for different parameter values, and the algorithms were assessed against their ability to consistently recover the parameter set, based on only the simulated observation values.

Four variants of the model presented in §6.2.1 were considered for the purposes of [initially] validating the overall approach. They are:

1. $SEI - tdR$ model, with one diagnostic test.
2. $SEI - tdR$ model, with two diagnostic tests.
3. $SETI - tdR$ model, with one diagnostic test.
4. $SETI - tdR$ model, with two diagnostic tests.

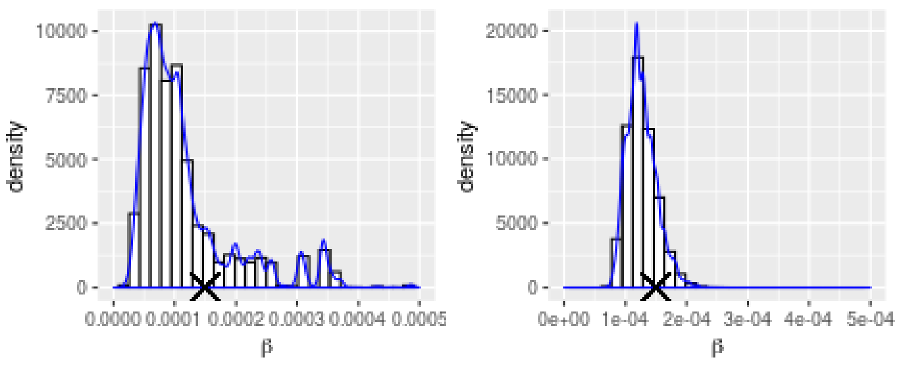
The full results are reported in Appendix C.1. In general the algorithms were moderately successful in recovering parameters. This was easily evaluated using the workflow described, since the *ARQ-MCMC* algorithm (validated using the Gelman-Rubin convergence diagnostic) provided comfort (or not) for the parameters estimates yielded by the others. That in turn provided at least some support for the estimates of the marginal likelihood computed independently by those algorithms (since they are computed from the same samples).

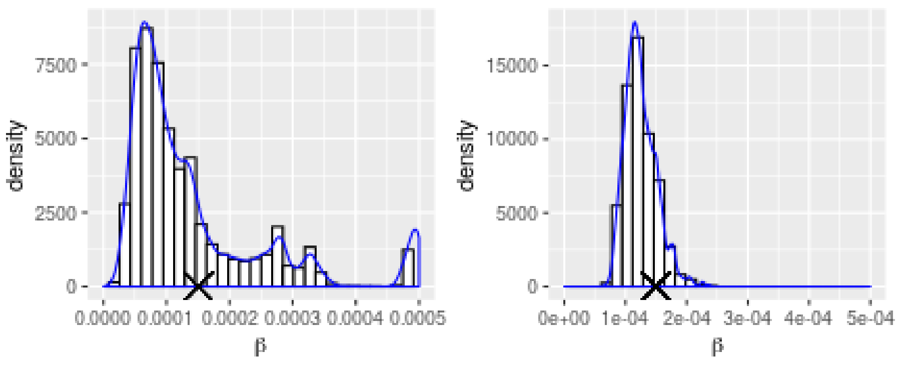
Inference in the context of multiple *types* of observation

Aside from the reasonable efficacy of the algorithms, another interesting point of interest that arose from this work is that the same number of observations (twelve per scenario) tended to yield higher variance estimates when only one test type was assumed. A sample from these results is given in Figure 6.7 to illustrate.

That was found to be somewhat counter-intuitive at first, since it increases the dimensionality of the model without providing extra data [additional observations.] However it could be rationalised by speculating that it mitigates correlation between parameters governing the disease process, and those governing the observation one; a form of 'triangulation'. Whether that speculation is accurate or not, the findings were deemed of sufficient strength to prioritise herds that also participated in the IFN_γ blood test trial for analysis.

With respect to the real world application, that choice certainly seemed to make sense since the reported high sensitivity of the IFN_γ test (reinforced by our findings) provides an otherwise impossible-to-glean glimpse of the 'true' disease status of the herd. Allowing that this happens in reasonably close [temporal] proximity to other, less sensitive tests – it is intuitive to understand why it would like be very helpful when it comes to statistical modelling.


(a) SMC^2 comparison.


(b) *MBP IBIS* comparison.

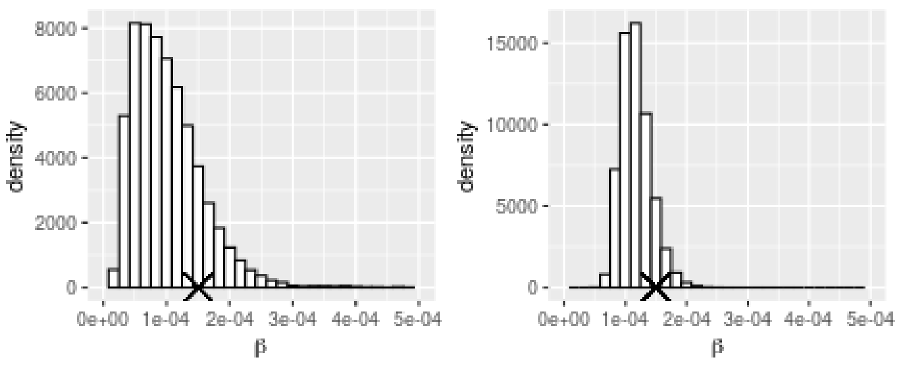

(c) *ARQ MCMC* comparison.

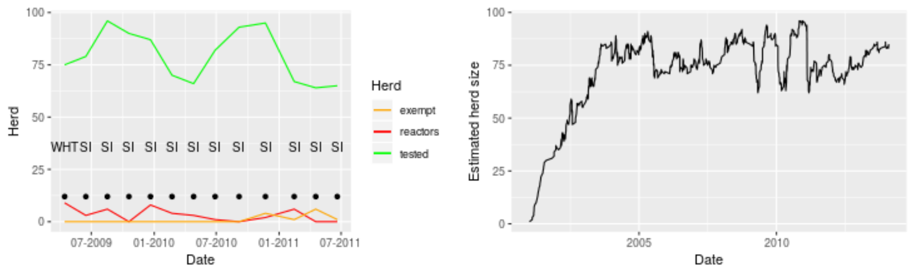
Figure 6.7: Comparison of contact rate parameter estimates from the validation analyses, with one diagnostic test (LHS) and two tests (RHS). The simulation parameters have been marked for reference. The simulated scenarios suggested that having more than one type of test is helpful for reducing variance of estimates for other epidemiological parameters, when test sensitivities are themselves also [assumed to be] unknown.

6.4 Results I : inference of within-herd BTB transmission dynamics

Here we demonstrate that Bayesian inference applied to the models and single-herd BTB data outlined above yields estimates for parameters that characterise individual herd outbreaks. The results shown here a based on a single outbreak, Incident one. Results from two further outbreaks are shown in Appendix C. Rate parameter estimates reported throughout are given w.r.t. days.

6.4.1 Incident one

The first herd analysed was located in Staffordshire with a head count fluctuating between about seventy and one hundred cattle. Details are given in Figure 6.8.



(a) BTB surveillance data from *VetNet*.

(b) herd size estimated from *CTS* movement data.

Figure 6.8: The first epidemic is based on an incident recorded in the *VetNet* database at a farm in Staffordshire which lasted about two years following a breakdown triggered by a routine whole herd test in 2009. Migration data is based on corresponding data as recorded in *CTS*. The latter is also used to corroborate the approximate herd size at the beginning of each modelled scenario.

SEIR model parameter estimates

The first set of results are based on the SEIR variant of the test-determined removal model. The marginal parameter distributions from application of SMC^2 to this data are shown in Figure 6.9. The corresponding results from application of ARQ MCMC are shown in Figure 6.10. Comparing these figures reveals a high-level of consistency between these algorithms in the parameter estimates obtained. This is confirmed by the summary statistics shown on Table 6.5. It is interesting to note that consistent with earlier analyses [15, 2] there is considerable uncertainty in the latent period (progression rate $E \to I$).

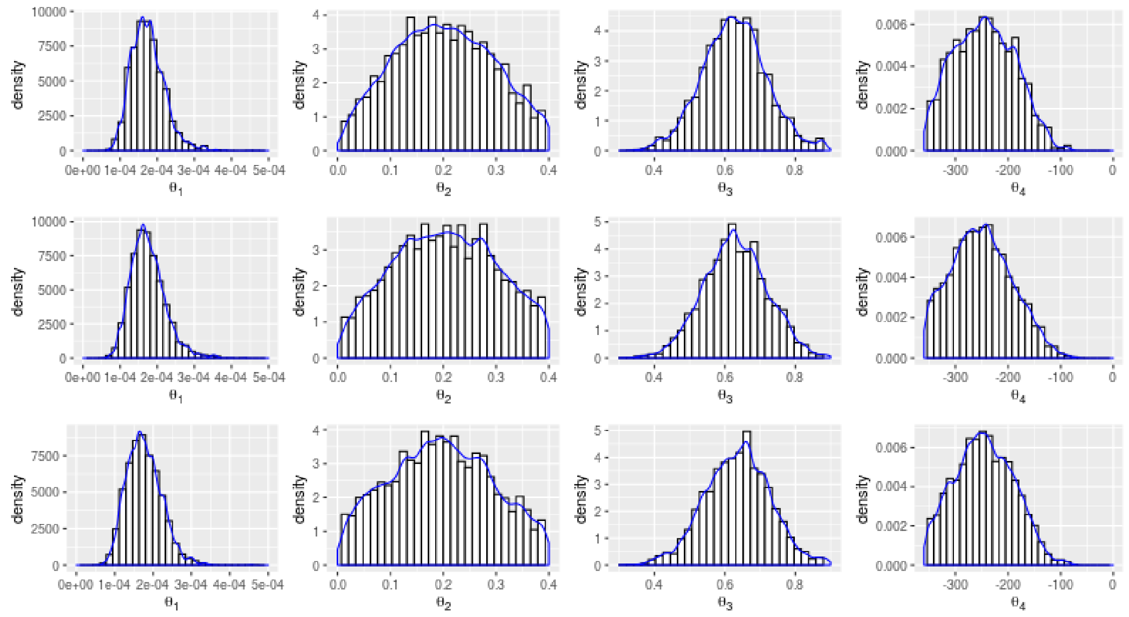


Figure 6.9: Marginal model parameter distributions for the first VetNet data set fitted to the *SEIR* model using the SMC^2 algorithm with $N = \{6000, 8000, 10000\}$ (independent proposals). $\theta 1:4 :=$ contact rate $S \to E$; progression rate $E \to I$; SICCT test sensitivity; and time of onset of infectiousness in the first affected individual. All parameter estimates are given w.r.t. days.

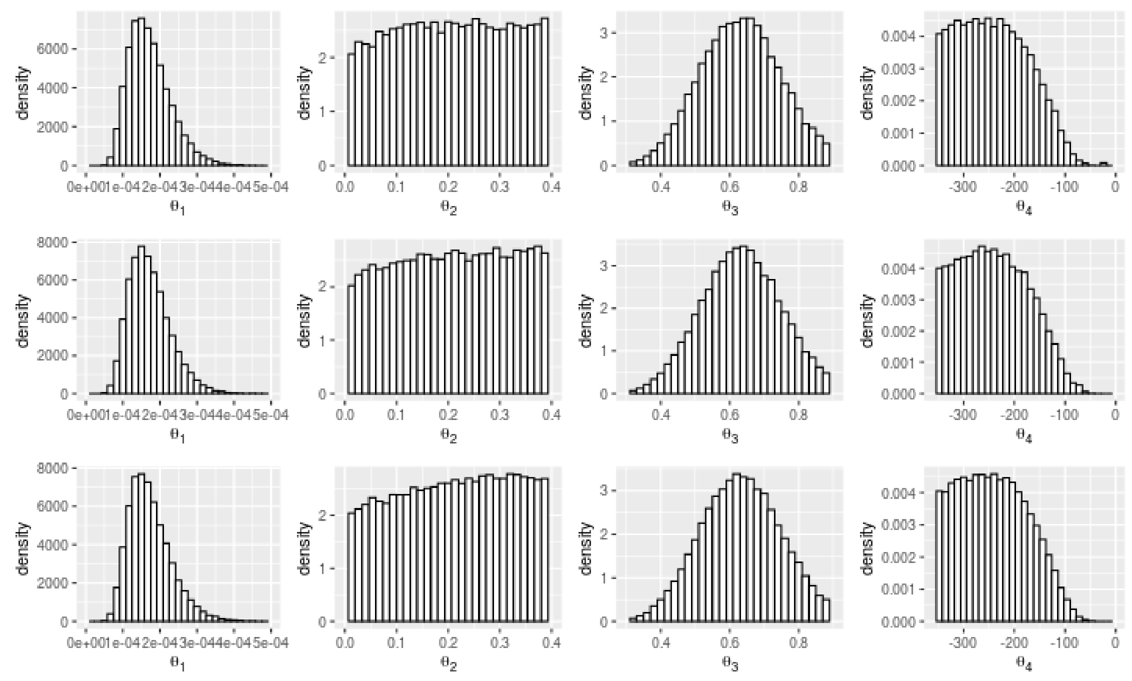
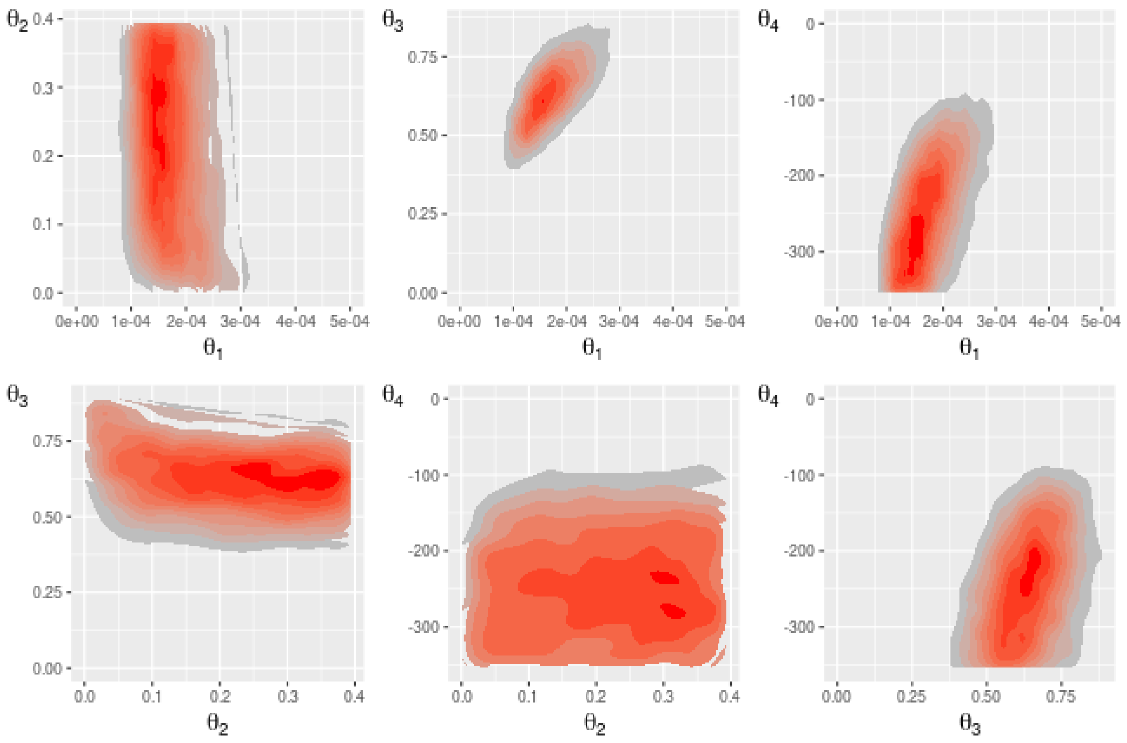


Figure 6.10: Marginal model parameter distributions for the first VetNet data set fitted to the *SEIR* model using the ARQ MCMC algorithm with $\Gamma_L = \{1, 3, 7\}$ ($\Gamma_R = 30$). $\theta 1 : 4 :=$ contact rate $S \to E$; progression rate $E \to I$; SICCT test sensitivity; and time of onset of infectiousness in the first affected individual. All parameter estimates are given w.r.t. days. The algorithm is much less efficient than SMC^2 but tends to produce consistent results.

Theta	iMu	iSD
1	1.80E-04	4.50E-05
2	2.00E-01	9.70E-02
3	6.40E-01	9.20E-02
4	2.50E+02	5.60E+01

Table 6.5: Inference summaries for the SMC^2 *SEIR* analysis (incident one). The model parameters labeled from one to four are as follows: contact parameter $S \to E$; progression rate $E \to I$; SICCT test sensitivity; and onset of infectiousness in the first affected individual, also denoted in the main text as t_0.



(a) Twin parameter marginal densities for the *SEIR* model analysis (ARQ MCMC).

Theta	iMu	iSD	rMu	rSD	SRE	(95%)
1	1.80E-04	5.00E-05	1.80E-04	5.70E-05	1	1
2	2.10E-01	1.10E-01	2.10E-01	1.10E-01	1	1
3	6.30E-01	1.10E-01	6.40E-01	1.10E-01	1	1
4	2.40E+02	-6.70E+01	2.40E+02	7.00E+01	1	1

Table 6.6: Inference summaries for the ARC MCMC *SEIR* analysis (incident one). The model parameters labeled from one to four are as follows: contact parameter $S \to E$; progression rate $E \to I$; SICCT test sensitivity; and onset of infectiousness in the first affected individual.

SETIR model parameter estimates

Inferences for the SETIR model are shown in Figure 6.12 and summarised in Table 6.7. These results also demonstrate a high level of consistency between the analyses conducted and between algorithms applied.

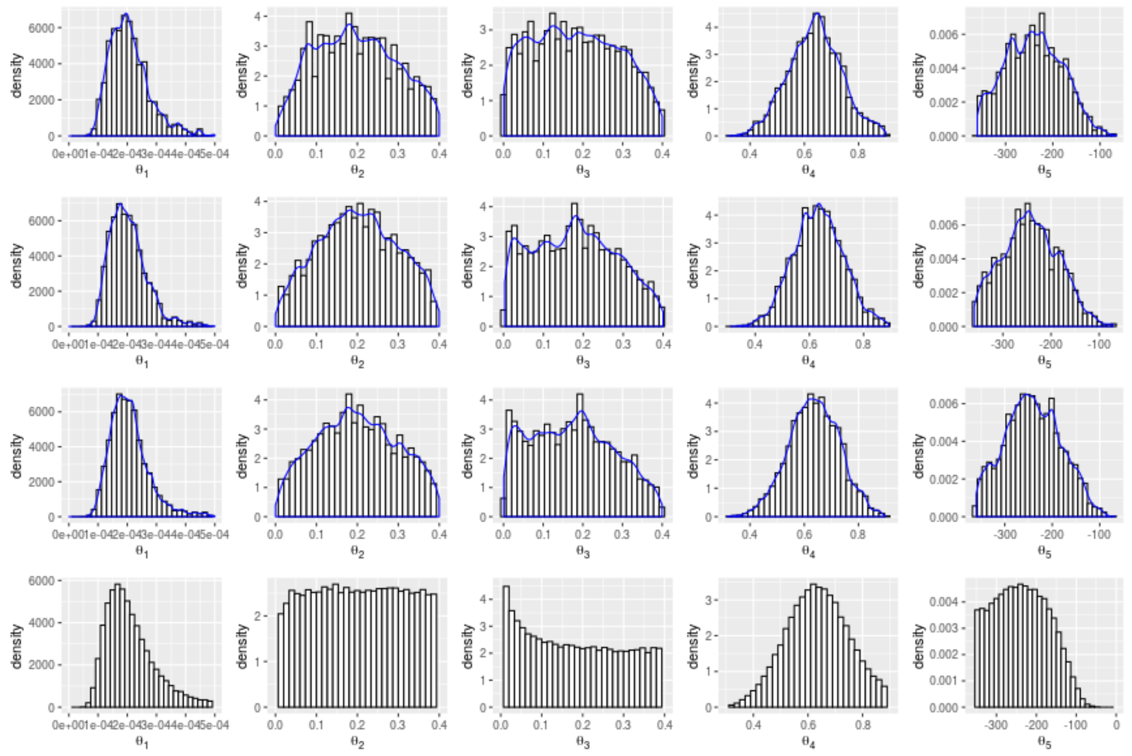


Figure 6.12: Marginal model parameter distributions for *Incident One* set fitted to the *SETIR* model using the SMC^2 algorithm – first three rows – with $N = \{6000, 8000, 10000\}$ (independent proposals) and ARQ MCMC (final row).

6.4.2 Summary

The results presented here demonstrate that it is feasible to use Bayesian inference for population level (DPOMP) models and operational data describing an oubreak to estimate key parameters that characterise the within-herd dynamics of BTB. The results presented for SMC^2 and ARQ MCMC show a high-level both of within-algorithm consistency between different configurations and between algorithms. They also show that it is possible to fit different models to the same data. In comparing the parameter estimates obtained for the SEIR and SETIR model

θ	E(X)	σ
1	0.00021	0.000066
2	0.2	0.099
3	0.17	0.1
4	0.64	0.094
5	240	59

Table 6.7: Inference summaries for the SMC^2 *SETIR* analysis (incident one). The model parameters, labeled from one to five, are as follows: contact parameter $S \to E$; progression rate $E \to T$; progression rate $T \to I$; SICCT test sensitivity; and onset of infectiousness in the first affected individual.

θ	μ_I	σ_I	μ_R	σ_R	PSRF	(95%)
1	0.0002	0.000064	0.00022	0.000083	1	1
2	0.21	0.11	0.2	0.11	1	1
3	0.18	0.11	0.18	0.12	1	1
4	0.63	0.091	0.64	0.11	1	1
5	240	-62	240	70	1	1

Table 6.8: Inference summaries for the ARQ MCMC *SETIR* analysis (incident one). The model parameters are as given in Table 6.7. Importance and rejection sampling estimates for are denoted by subscript, with μ used as shorthand for E(X).

we also observe a high-level of consistency between the estimates obtained for the contact parameter $S \to E$, the SICCT test sensitivity; and the time of onset of infectiousness in the first affected individual. There is less consistency in the latent period which in the SEIR model is the residence time in the E state given by the inverse of the rate of progression $E \to I$. The corresponding figure for the SETIR model derived from the residence times in the E and T states is almost twice as long. However, this is consistent with uncertainty estimates obtained for these parameters, and moreover as noted earlier is consistent with the highly variable estimates for the latent period published in the literature. These results are broadly confirmed through analysis of the other two incidents described in Appendix C and that in this case examination of the model evidence (see §C.5) does not provide sufficient grounds to distinguish between the two models presented here. Finally, although the results obtained here are promising it should be noted that they are derived from data on relatively large outbreaks and therefore may not be representative of all herds. Nonetheless in the next section we build on such results by examining the fits for a larger number of such herds in three geographically distinct regions.

6.5 Results II: meta-analysis of herd outbreaks

We now present the main results of the analysis conducted for this chapter. The preliminary results presented in the previous section and in the appendices (based on both simulated and real data) suggest that the selected methods are reasonably effective for solving the models, in at least some cases. The final data set chosen for analysis contained eighteen distinct herds (although preliminary analyses were carried out for slightly over one hundred.) These were selected by applying the criteria laid out earlier in the chapter to farms drawn from three distinct UK regions:

1. Yorkshire (six herds)
2. East of England (four herds)
3. Scotland (eight herds)

Notwithstanding the efficacy of the Bayesian workflow already described for the purpose of validating the results, additional verification was sought by using multiple independent runs and configurations. These are presented independently for the parameter inference results.

6.5.1 Model comparison

We begin with comparison of models (shown in Table 6.9) w.r.t. to each herd and data set for the models.

These results are reported in full in Appendix D. They are organised by region but should properly considered as the results of a series of independent analyses; there is little meaningful comparison to make between herds for this part of the results (except of course concerning the relative performance of models).

The results show a large degree of variation between herd outbreaks, making it difficult to reach clear findings. However, this is perhaps not particularly surprising. The number of herds that could be practically analysed in the time available was prohibitive, and perhaps clearer indications would have emerged from a larger scale analysis – that after all is the nature of statistics. Nonetheless we attempt to summarise the relative strength of evidence for each model. One reason it was difficult to draw clear conclusions is that it was not always possible to estimate the model evidence sufficiently accurately. In fact taking a cut-off of **Var(BME)**≤ 10

this was achieved in only around a half of all cases for each model except the SRI-SEIR alternative reinfection model, suggesting some merit in considering the role of reinfection of models of BTB. In the East of England group evidence was only calculated accurately for the SRI-SEIR model and once for the TDR-SEYZ model. Therefore we exclude the East of England group and examine rankings using model evidence only where **Var(BME)**≤ 10. Then considering those models with 1^{st} or 2^{nd} lowest estimated evidence we find that the SETIR model with density dependence is ranked top overall with the frequency dependent SETIR variant 2^{nd}. Next highest ranked are jointly the TDR-SEIR frequency dependent and the SEYZ models. There is little to separate frequency v density dependence when combined over TDR-SETIR and TDR-SEIR models. However, the SETIR dynamic (combined across both types of density dependence) is ranked higher than the alternatives including the SEIR dynamic.

Model code	Description
TDR-SEIR (Freq. Dep)	Frequency dependent TDR-SEIR
TDR-SETIR (Freq. Dep)	Frequency dependent TDR-SETIR
TDR-SEIR (Dens. Dep)	Density dependent TDR-SEIR
TDR-SETIR (Dens. Dep)	Density dependent TDR-SETIR
TDR-SEYZ	Reinfection model
SRI-SEIR	Alternative reinfection model

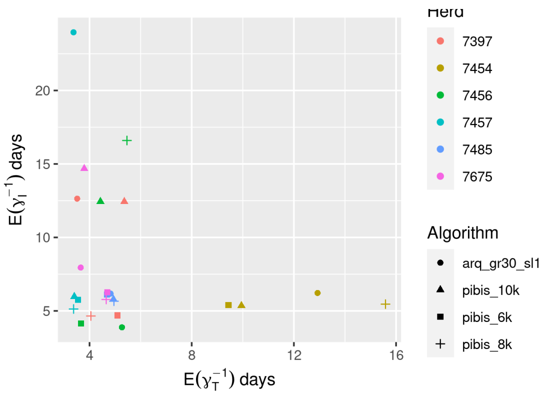
Table 6.9: Enumerated models and configurations, as per §6.2.1.

6.5.2 Epidemiological parameter estimates

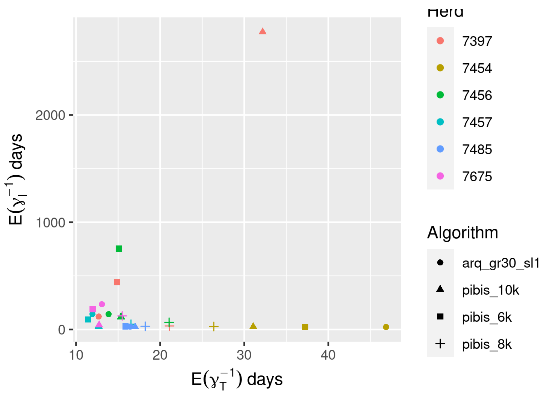
As with the results already reported, several algorithm configurations (and runs) were used to compute parameter estimates for each of the three herd groups. A full set of results for all models and herd-groups is presented in Appendix D.2. Here we present results from the TDR-SETIR models since these were found to be the top ranked in the analysis above, starting with the latent period and then going on to explore the other model parameters.

Estimates for the latent period

Estimates for the latent period in the TDR-SETIR model correspond to the transitions $E \to T$ and $I \to I$ denoted γ_T^{-1} and γ_I^{-1} respectively. The results show considerable variation and we note that in all regions (see Figures 6.13, 6.14, 6.15) the variation under the density dependent model is am order of magnitude or more larger than under the frequency dependent model, which may provide insight into the wide range of latent period estimates obtained in the literature.

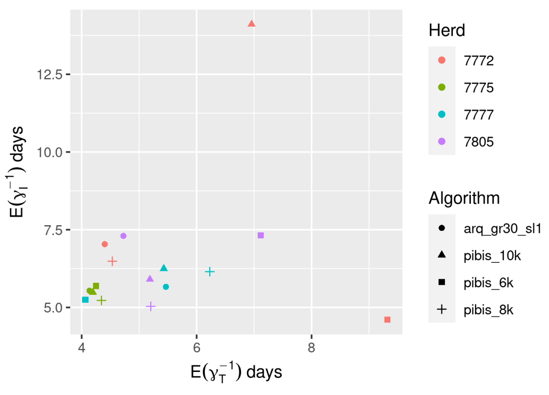


(a) Frequency-dependent

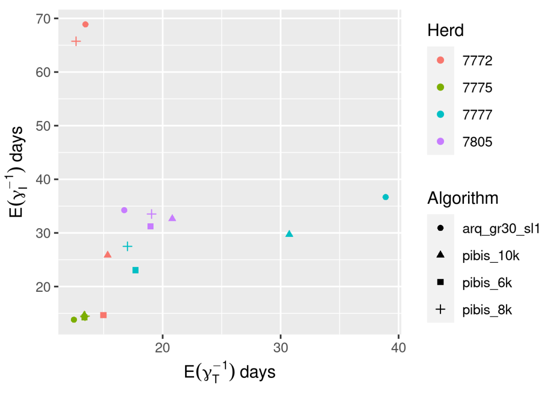


(b) Density-dependent

Figure 6.13: SETIR model estimates for Yorkshire

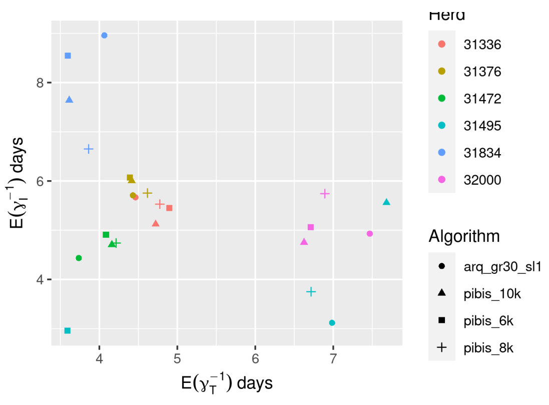


(a) Frequency-dependent

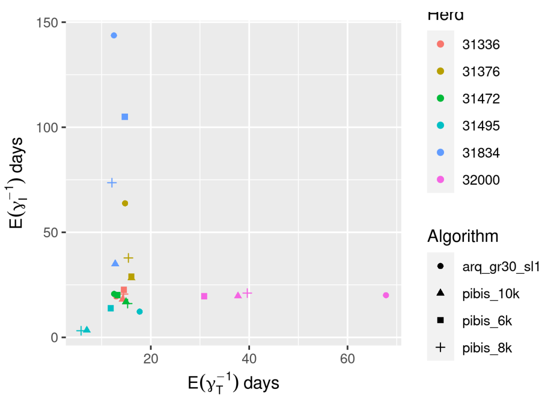


(b) Density-dependent

Figure 6.14: SETIR model estimates for East of England



(a) Frequency-dependent



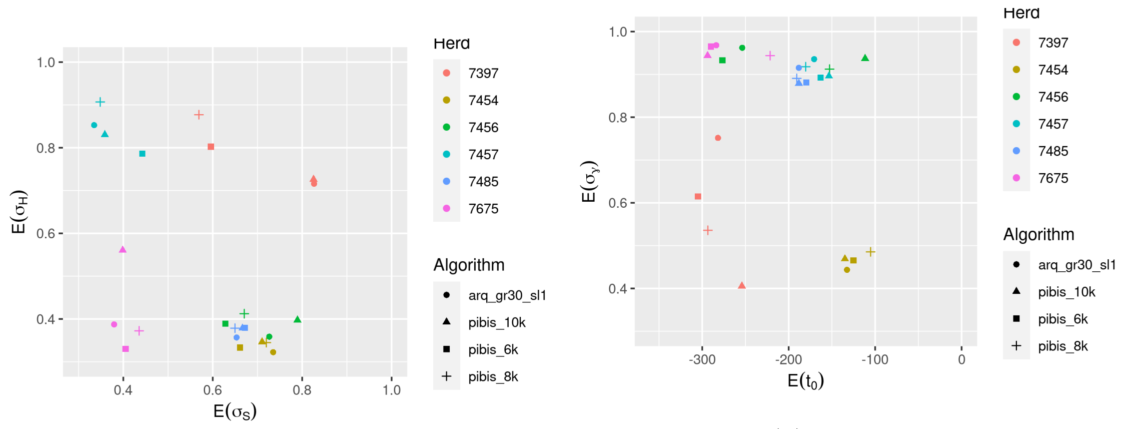
(b) Density-dependent

Figure 6.15: SETIR model estimates for Scotland

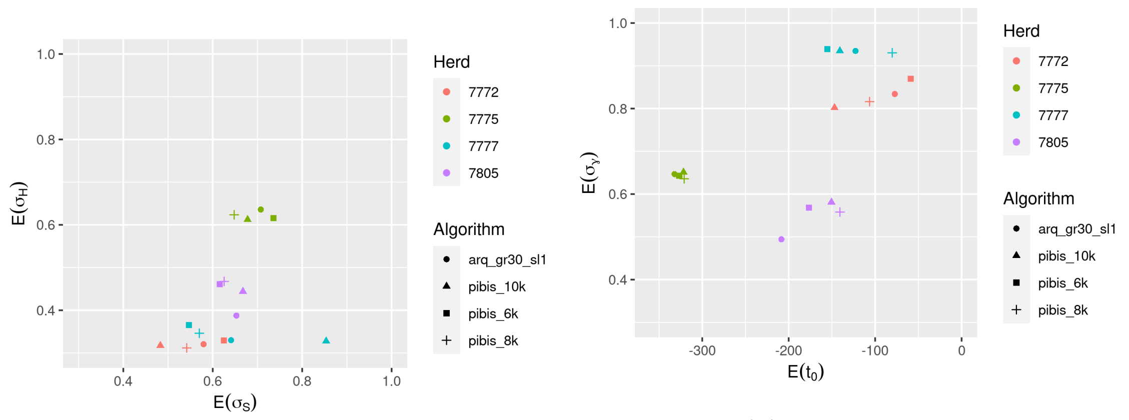
Testing sensitivities: density dependent TDR-SETIR model

Here we visualise the posterior expectations of the model parameters describing the probabilistic quantities associated with detection by different diagnostic tests (and specifications) under the density dependent TDR-SETIR model. They are denoted by σ_S and σ_H for standard and high sensitivity SICCT 'skin' tests, and σ_γ for the IFN-γ blood test. Also included are estimates the time of introduction of the disease t_0, though it is otherwise treated here as a nuisance (not of primary interest) parameter. The results again show the huge variation between herds and between region. In general results from different algorithms produce similar outputs for each herd (same colours cluster together).

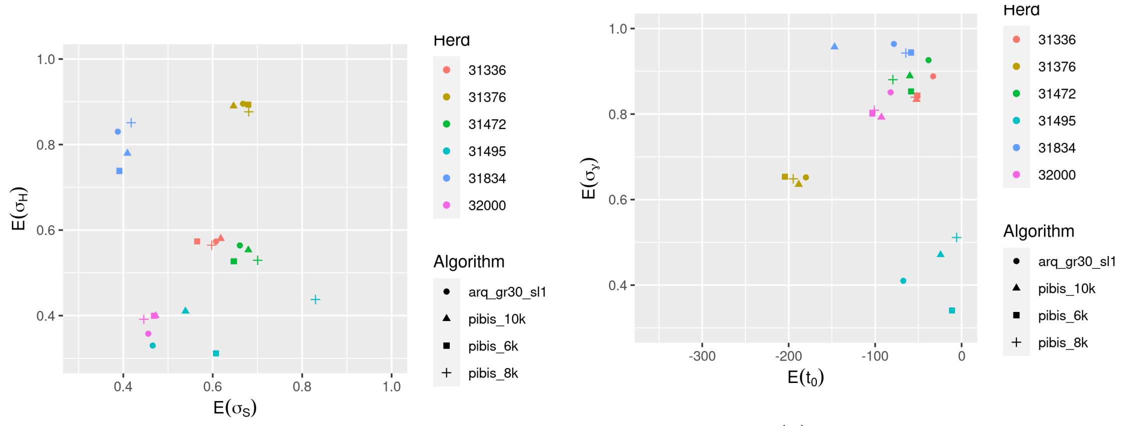
Note that the corresponding variances for these estimates are not reported here but are available in the appendices for each individual set of results.



(a) Yorkshire



(b) East England



(c) Scotland

Figure 6.16: Joint marginal parameter expectations under the Density dependent TDR-SETIR model, Results are shown for Yorkshire group of herds (top), East of England (middle) and Scotland (bottom)

6.6 Discussion

The study proceeded roughly as envisaged at the outset: having identified (and developed) appropriate methods using simulated examples, parameter estimates were obtained for a range of models and herds. This was initially done for a small number of herds as a proof of concept (the individual herd scenarios provided in Appendix C) and then repeated (with iterative minor improvements w.r.t. technique) for a larger number of herds – again, as envisaged at the outset. However the number of herds and data sets that could be analysed fell way short of initial expectations, with only a handful of the thousands available judged suitable for analysis using these methods. The key constraints were data quality and 'identifiability' – in plain terms, there was only sufficient information to provide useful parameter estimates when the total number of reactors observed for a given herd, was large enough. In very rough terms, fifteen or more was found to be a 'good' number. Below we discuss each aspect of this analysis (inference, models and data) before summarising lessons learned and outlining possible ways forward.

Inference algorithms

As a well established method, the SMC^2 algorithm was deemed especially important as a baseline for validation and comparison [9]. This was compared with MBP-IBIS and ARQ MCMC during the validation stage of the analysis, as laid out in §6.3. Ultimately the two that proved most tractable for the problem –SMC^2 (i.e. Algorithm8) and Algorithm 10; the ARQ-MCMC presented in Chapter 4 – were used to implement the workflow, and produce the results presented in onward sections. As an interesting aside, this situation was reversed somewhat during the course of the work presented in Chapter 7, when data-augmented methods proved to be more tractable for the models considered. Another key benefit of the SMC^2 (and *MBP-IBIS*) algorithm is that it provides a convenient and computationally inexpensive estimate of the marginal likelihood. This was used to compute the Bayes factors for model comparison.

Modelling assumptions

The models used in Chapter 6 necessarily incorporate certain assumptions: within herd dynamics only, i.e. no (ongoing) external force of infection; introduction via single infected individual disease status of any given herd; homogeneous test responses, e.g. the possibility of immune deficient individuals is ignored. Though these are strong assumptions, they were judged to be mostly not problematic in the first instance and have been commonly adopted in earlier studies.

Data quality

A range of minor problems, and one major problem, affecting data quality presented during the course of the work. In particular, cases were only selected for inclusion in the final results where herd size could be estimated (or at least, roughly verified) using transactional CTS data. They were accounted for systematically where possible, and other efforts were made throughout the study to ensure a [systematically] 'clean' data set. For example, Scotland became the largest single cohort in a deliberate effort to isolate extraneous factors that might have introduced (for want of a better term) 'noise'. As noted above, it was hoped that the relatively low incidence of BTB in Scotland would help to ensure this. However the results were probably the poorer in the end for the *lack* of BTB incidence, since few observations translates to higher variance in the posterior distribution, and more difficulty with convergence in the algorithms.

Lessons learned

Nevertheless, important lessons were learned during the course of the work. The study served its purpose as a proof of concept. Even though the outcome was not the desired one, it did reinforce the rationale for the corresponding objectives; by highlighting the need for a hierarchical approach, and methods that are scalable beyond a mere handful of herds. It also highlighted that exercises in parameter inference are likely to be of greater scientific value and interest when combined with model-inference, so that we understand the extent to which models fit the data. These observations provide retrospective support for the efforts made the address that problem, and in particular the key outputs from Chapter 4. The results presented in §6.5 also have one other minor strength. Where previous work has focused on (largely approximation-based) aggregate analysis of the overall BTB system, or farm network – the approach employed here provides at least *notionally* independent parameter estimates for each herd, excepting the question of systematic bias introduced by the way in which the herds were selected for analysis. That is, where many others have produced epidemiological parameter estimates for BTB that conflict greatly with one another, we have managed to produce estimates that conflict even with themselves! This highlights the importance of, and the need for, systematic approaches to quantify between-herd variation in BTB dynamics.

Towards a hierarchical, 'system-of-herds' model

The ultimate goal of this thesis is to go beyond previous attempts to characterise such systems for BTB (such as [17]) by incorporating *directly inferred* within-herd disease dynamics for a system of comparable size (ten thousand herds in that case). As discussed, the approach used here here proved computationally and manually unequal to the scale of this particular ambition. That is not because SMC methods are intrinsically unsuited to hierarchical methods. On the contrary, [89] describe a flexible scheme that might have served for this purpose. However it was found that the algorithms required manual attention with respect to each herd, and were computationally intensive, with an average herd analysis taking on the order of 30 to 45 minutes(not including manual attention). More importantly, some data sets were found to be more tractable to analysis than others and in practice it was convenient to simply discard those that proved 'difficult'. This is problematic for a hierarchical analysis because the likelihood is essentially given by the product of many individual 'component' likelihoods – a failure in one analysis essentially ruins all.

It was therefore anticipated that the further development of this particular approach would scale probably at most to a few dozen herds in the data set considered, and that such efforts would be largely without merit. Nevertheless, the knowledge and familiarity with methods obtained throughout course of the entire project did provide the foundation for one final (within the bounds of this work) attempt to solve this problem. That approach and the accompanying analysis, that was successfully carried out for approximately county-wide systems of herds comprised of a few hundred herds, up to region and nation-wide systems comprised of many thousands, is presented in the next chapter.

Chapter 7

Alternative within-herd model

"All models are wrong, but some are useful."

- George Box[1]

Summary

- Endemic diseases like BTB (and also COVID-19 [90, 91, 92]) present significant statistical modelling challenges due to factors such as the *possible* presence of extraneous sources of infection; uncertain routes of transmission; and the highly [statistically] dispersed (or 'clustered') nature of localised outbreaks. This is evidenced in part by the large degree of variation in estimates that have been reported for important epidemiological BTB parameters by others, even those that are notionally based on the same compartmental models and data [2, 15, 17].

- Furthermore our own results reported in the previous chapter tentatively suggest that no single [state-space-]model that was considered in that analysis (including those that have typically been applied to BTB in the past by others) demonstrated a particularly good fit for the data compared to others. At least, not consistently across different herds.

- Here we consider a different approach; indeed, a different type of stochastic process altogether, in a revised model where disease transmission is represented by a *single* [univariate] process for each herd. A key benefit of this less

[1]There is some uncertainty surrounding the provenance of this quote but it is usually attributed to the statistician George Box.

computationally intensive approach is that it allows us to go beyond the work presented in the previous chapter, by directly modelling within-herd transmission for systems comprised of *thousands* of herds, rather than individual ones, thus adding statistical strength to the estimates that are obtained.

- The herd groups analysed range in size from county-wide (typically in the order of a few hundred herds) up to nation-wide (the whole of Scotland, c.4,500 herds) and the results include estimates for parameters relating the regional (county-level) background risk rate; trade-related risk; and the detection probabilities associated with various kinds of diagnostic test.

- We develop and apply three variants of hierarchical Bayesian model, and make substantial progress towards a model and methods capable of estimating parameters for the entire VetNet/CTS data set, i.e. the full national herd system (excl. Northern Ireland).

Background

Scientific domain modelling (and in fact, statistical inference generally) can be informally understood as adding value to the statistical analysis of data, by introducing 'structure' to a 'raw' data set. That structure can be informed by our pre-existing knowledge (or more accurately, our beliefs) about the broad shape of some e.g. biological process or system. Combined with the alchemy of mathematics, our beliefs about the nature of say an epidemic process, i.e. a pathogen spreading through a naive population, give rise to [a mathematical identity for] a *statistical distribution* that links the data to the model. This is referred to as the *posterior distribution* in Bayesian statistics. The 'added value' arises from the fact that we need only fit a handful of observations in order to be able to characterise the entire, mostly unobserved population (or system) more generally. In other words, we can gain a significant amount of knowledge from a relatively small amount of data. That is, assuming that our assumptions (the mathematical manifestation of our beliefs) are *reasonably* correct.

An alternative, and in some ways contrary approach, is to assume the opposite: that we know very little about the pertinent underlying, e.g. biology, and regard the data somewhat more abstractly as a statistical distribution in the first instance.

In this case it would perhaps be better described as a statistical *signal*, since the BTB surveillance and trading data we are dealing with are longitudinal: recorded over time.

Of course, this is a false dichotomy in most practical situations[2]. The approach adopted here is somewhat analogous to the latter, more abstract way of doing things; a deliberate decision to focus on the statistical properties of the underlying empirical data –in particular, the [possibly] *overdispersed* distribution of reactors within and between herds– as a way of identifying promising methods and statistical tools. A key motivating factor was the need to improve upon the results presented in Chapter 7, in order to construct a hierarchical model that could be used to models systems comprised of thousands of individual herds. The solution that resulted from those initial motivating principles is at least simpler than the methods primarily considered heretofore in the thesis. A single stochastic process is used to model the entire disease process within each herd (rather than a coupled system of [IHPP] processes for each herd), and in that sense it could be judged less prescriptive in terms of the underlying biology. However it could just as accurately be stated that it is merely an alternative description of the underlying biology. It is a matter of subjective interpretation.

We now present that work, which can be regarded as an addendum to the (originally planned) main body of the thesis. We begin by revisiting pertinent methodology from Chapter 2, in particular Hawkes processes. We then do the same for the data, which was described more fully in the previous chapter. That is followed by a description of the model, which is fairly distinct from those used to obtain the results presented in the previous chapter, as well as materials and methods, in §7.3 and §7.4 respectively. Details concerning the DA-MCMC algorithm used to solve the model are provided in §7.4.2. As before, results reported in this chapter include those based on both simulated and actual BTB surveillance data, with the latter reported in §7.5. Please note that only a summary of those results are presented in the main text. The bulk, including the most granular representations and inference tables, are laid out in Appendix E. Finally, the chapter concludes with a summary discussion in §7.7, following on to the conclusion of the thesis itself, in Chapter 8. A visual summary of the chapter is provided in Figure 7.2.

[2]Quantum physics is the exception that springs to mind here.

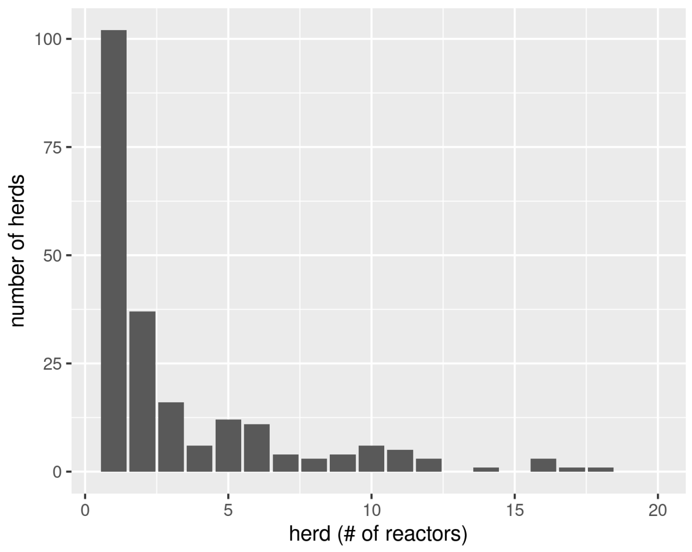
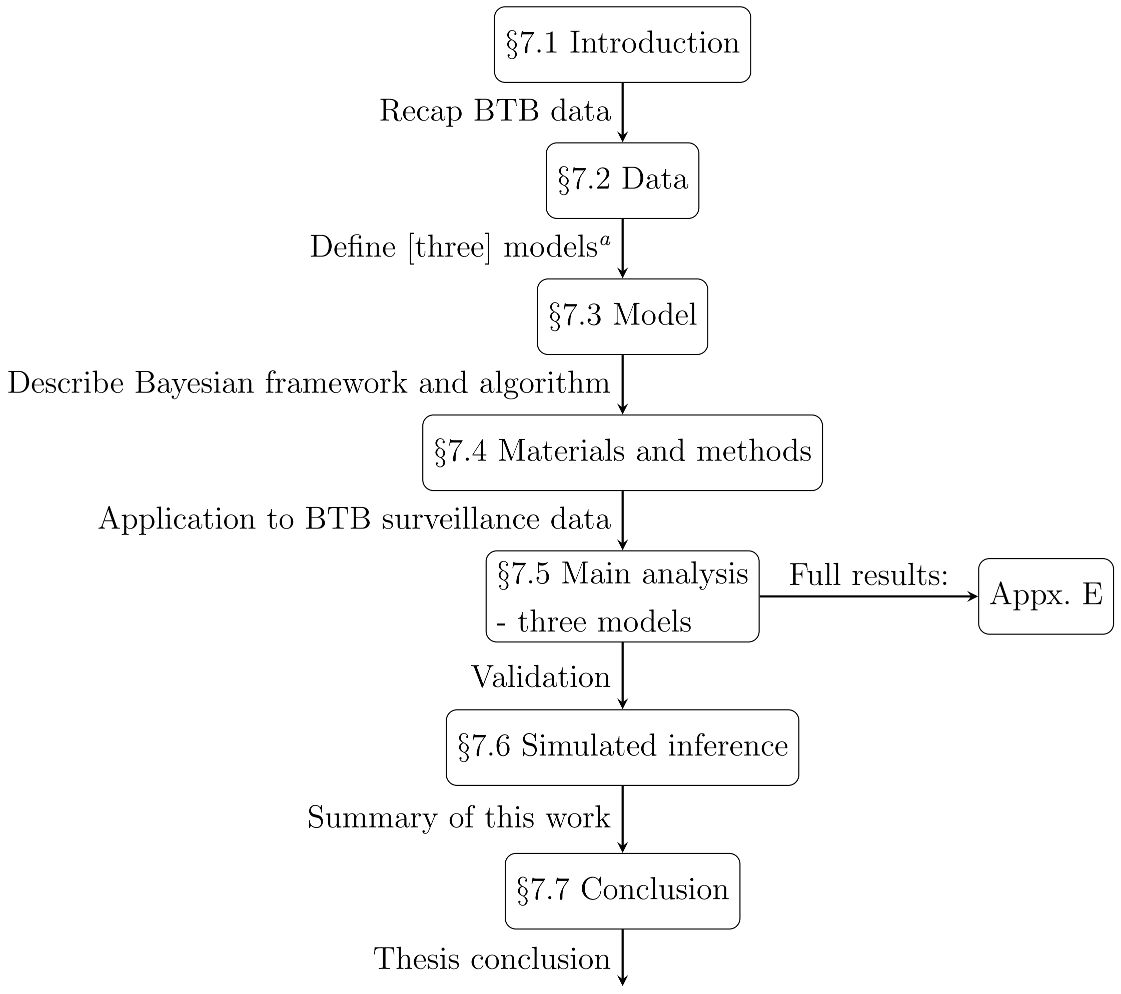


Figure 7.1: Reactor distribution by number of reactors in each herd. This is example is for a data set of herds drawn from the East Midlands over a similar time period. Further details of the corresponding analysis are given in Appendix E. BTB incidents reported in VetNet are 'clustered' (or *overdispersed* – higher than expected variance) with most incidents consisting of a single reactor (and many more herds with none at all) but a handful with more than a dozen. This presents significant modelling challenges that are discussed in due course.



[a]Alternatively, three variations on the same [Hawkes process] model: they are, *homogeneous process model*; *hierarchical model with homogeneous disease processes*; and *hierarchical model with heterogeneous disease processes*. Further details are given in §7.3.

Figure 7.2: Layout of this chapter. It begins with a recapped introduction to Hawkes processes, other relevant concepts and data in §7.1 and §7.2. Those inform the construction of three variants on the same essential model: a partially observed, *non*-Markovian process model, laid out in §7.3. [Bayesian] methods and materials are discussed in §7.4. The algorithm that was used to obtain the results is then described in §7.4.2, with the results themselves reported in §7.6 onwards, for both simulated and real herds. The chapter concludes with a summary discussion in §7.7.

7.1 Introduction

Here we provide a brief overview of Hawkes processes and other concepts pertinent to the work presented in this chapter.

7.1.1 'Self-exciting' Hawkes processes

As noted in §2.2, stochastic epidemiological models can be formulated using point processes other than the standard inhomoengous *Poisson processes* that have been applied thus far in the thesis. The other variant introduced in that section are 'self-exciting' *Hawkes processes* [27]. Note that Hawkes processes are by definition non-Markovian. That is because the time evolution of the system at any given time t depends on the entire history of the system up to that time. Here we briefly discuss the rationale for using them, and also recap the essentials from Chapter 2 to provide a foundation for the [partially observed, *non*-Markovian process] model and methods presented later.

Statistical ***overdispersion***

Overdispersion is a [relative] property of data, that relate it to a given statistical distribution. It is defined by 'excessive' variance. That is, variance that is higher than expected given the assumed distribution. Or equivalently, higher than that predicted by a given statistical model. We can begin to understand the usefulness of Hawkes processes for modelling data that, loosely speaking (i.e. without respect to any specific model) are overdispersed, by considering their application to what could be described as 'clustered phenomena'. For example [24] describe a small (i.e. finite) population model based on an SIR model for analysing social media data, where such [overdispersed] phenomena are often conversationally described as 'viral' content. Another area where Hawkes have been usefully applied is in finance. Here again, it is intuitive to consider the stock market as a system that fluctuates fairly steadily for the most part, but is also prone to occasional flurries of activity, e.g. a 'sell-off', in which the returns for an individual portfolio or else the system behaviour as a whole could reasonably be described as 'overdispersed'. Here again, the concept is a somewhat informal and ambiguous one without reference to a specific statistical model or distribution though.

Within the field of epidemiology, Hawkes processes have been recently applied to the problem of modelling COVID-19 incidence. Similar to the data analysed here, data from that epidemic exhibit what could be described as overdispersion in the form of clustered outbreaks of disease [90, 91, 92]. We do not claim that Hawkes processes are intrinsically a better fit to the data we consider here though, despite their apparently overdispersed or clustered properties. On the contrary, we note in §7.7.5 that we have yet to address that vital question by means of formal model comparison. Furthermore despite being diametrically opposed in one fashion, Markovian and non-Markovian stochastic process models are not necessarily mutually exclusive in terms of their application here, i.e. it is possible to envisage a coupled system of mixed-type stochastic processes.

We now recap the essentials of Hawkes processes, and go on to describe how the concept can be used to model an endemic disease with a relatively stable 'background' latent disease population, in which clustered epidemics of infectious individuals are nevertheless observed (i.e. BTB).

Definitions

First we recall the underlying mathematical model as it applies here. The 'event rate' of a Hawkes process can be written as:

$$\lambda(t) = \mu + \sum_{t_\xi < t} \phi(t - t_\xi) \tag{7.1}$$

with an exponential kernel function chosen for ϕ such that:

$$\phi(\tau) = \alpha\beta e^{-\beta\tau} \tag{7.2}$$

where α is a scalar and β parametrises a time-dependent exponential decay that characterises the impact of past events on the event rate at a given time t.

Hawkes processes within finite populations

Rizoiu *et al.*, [24] provide a generalisation of the Hawkes process for models with finite populations of size N in which the event rate is defined by:

$$\lambda^H(t) = \left(1 - \frac{N_t}{N}\right)\left[\mu + \sum_{t_\xi < t} \phi(t - t_\xi)\right] \tag{7.3}$$

where N_t is the associated counting process, such that when $N_t = N$ the event rate is zero. The application in that was the aforementioned SIR model that was applied to the analysis of social media data.

7.1.2 Partially observed Hawkes processes

N.b. not to be confused with 'latent Hawkes processes' The fundamental modelling concept described in the next section is essentially that of a *partially observed Hawkes process*. We note that this is not the same as the concept referred to elsewhere as 'latent Hawkes processes', where some portion of the 'latent' process is *directly* observed [39]. The distinction can be better understood by invoking the concept of *informative* (and *uninformative*) observation times. In this case, the BTB surveillance data are largely scheduled and thus observation times are *uninformative*. We therefore refer to them as being *'partially observed'* processes. This provides continuity with terminology used heretofore in the thesis. More importantly, it distinguishes them from the (pre-existing) concept of *latent Hawkes processes* with *informative* observation times, as described by others.

7.2 Data

Here we briefly recap the pertinent data and describe the observation model that was used for the models and analyses presented in due course.

7.2.1 BTB surveillance data

The data analysed are the same as that analysed in Chapter 6: VetNet BTB-surveillance records and migration data including trade records drawn from CTS. An important limitation of the analysis conducted for that chapter was the fact that useful estimates could only be obtained from herds and incidents with a reasonably large number of reactors. This is problematic as a possible source of selection bias (but also wasteful because we are throwing away useful data). Other exclusion criteria included holdings with more than one herd (due to differing levels of granularity in the data). The frequent occurrence of informal partial herd testing also led to ad-hoc adjustment or exclusion in some cases due to the difficulty involved in (manual) interpretation of those records.

Such a degree of discrimination and manual scrutiny was not feasible in this case, because the aim was to analyse individual herd records for systems comprised of thousands of herds. In particular, no exclusions were made based on reactor count. This eliminated a significant, likely source of systematic bias and allowed for the inclusion of herds with no reactors. This in turn allowed for those records to contribute a good deal of statistical strength to the analysis w.r.t. important parameters relating to background (including trade-related) transmission risk. The final data selection criteria were determined as follows:

- Herds (and date ranges) with corresponding records in CTS.
- Herds with at least two initial clear herd tests, i.e. so as to exclude cases that were already in the midst of a BTB-related incident and better characterise parameters relating to background risk.
- Holdings with more than one recorded herd (for the same reason as before).

These data were analysed in regional subsets of increasing size, with precise details for each subset given in due course. As before, test records were organised by four major classifications depending the test code used to record them. They are:

1. Routine: routine herd tests, usually marked RHT or WHT for routine- or whole-herd test, respectively.
2. Trade: also, in a sense, routine; the records of most interest are probably those that pertain to pre-movement testing for the purposes of trade.
3. Risk: conducted based on the perception of elevated disease risk. These include non-routine trade-related ('tracing') tests; 'contiguous' testing in herds local or adjacent to infected herds; and short-interval tests of various kinds, typically conducted at higher sensitivity than the routine SICCT test.
4. IFG: Interferon-γ blood test.

Estimates were obtained for the detection probabilities associated with each, as reported in due course.

7.2.2 Observation model

As in Chapter 6, the same observation model was used for all models and scenarios. It is parametrised by n distinct values in the vector of unknown model parameters θ; σ_T is intended to represent a given type (and sensitivity) of diagnostic test. The observation model (or more correctly, process) is functionally denoted in the diagrams below by $g_\theta(\mathrm{N})$ where N is the number of test-sensitive individuals.

σ_i	Symbol	Description	Prior: U_a	U_b
1	σ_S	SICCT test sensitivity – standard	0.3	1.0
2	σ_H	SICCT test sensitivity – high	0.3	1.0
3	σ_R	Risk-induced (variable type)	0.3	1.0
4	σ_γ	IFN$_\gamma$ test sensitivity	0.3	1.0

Table 7.1: Diagnostic test categories.

It is assumed that each test type –they are listed in Table 7.1– is associated with a given probability of detection. The corresponding statistical distribution is the Binomial:

$$g_\theta(N) = B(N, p^*\sigma_T) \tag{7.4}$$

where σ_T is the [inferred] probabilistic quantity intended to represent the sensitivity of test type T. Not all of the herd may be tested on a given occasion[3]. Thus, the quantity p^* is again used to represent the proportion of the herd that were, i.e. the probability of being selected for a test to begin with. This is computed directly from the available data, including the total herd size [in *VetNet*] as verified (and where necessary, inferred) from the corresponding *CTS* data.

It is a (possibly inadequate[4]) feature of the model, that the removal of individuals who test positive does not directly impact disease dynamics, unlike in Chapter 6. Thus the observation process does not have to be accounted for in the same way with respect to the model state space[5] in this case.

[3]At the cost of repetition, it is important to emphasise once again that this does *not* refer to cases where herd tests are carried out over multiple dates.

[4]This is discussed along with a possible remedy in §7.7.

[5]In some of those cases, it was necessary to sample the removed individuals from the possible states, i.e. ‘T’ and ‘I’ for the SETIR model.

7.3 Model

Here we describe a simple state-space model based on the SIR, for an endemic disease with a relatively stable 'background' latent disease population, in which clustered epidemics of infectious individuals are nevertheless observed (i.e. BTB). It is depicted in Figure 7.3.

$$\begin{tikzcd} \boxed{\mathrm{S}_h} \arrow[r, "\lambda(t)"] & \boxed{\mathrm{T}_h} \arrow[r, "g(\mathrm{T})"] & \boxed{\mathrm{R}_h} \end{tikzcd}$$

Figure 7.3: The STR model. The compartments are labelled to provide continuity with previous chapters. For example, individuals judged to be observable take the value T of the discrete state vector, e.g. (S, T, R). However unlike in previous examples, a subscript is used to denote a given herd. Individuals transition only within their own herd (with inter-herd population dynamics accounted for separately). Removal is determined by the observation process itself, similar to the models described in the previous chapter.

One of the main distinguishing features of the model is that it is intended to represent systems of herds, rather than individual ones. That is accounted for in the diagram by the fact that each state (compartment) is indexed by the subscript, h, e.g. S_h.

We now describe each of the three models. The first is the most basic; a 'baseline' for the other two. The latter two (unlike the first) are hierarchical models that otherwise share the same broad construction as the first, but account for the [latent] disease process is slightly different ways. That distinction is described in due course.

7.3.1 Basic model variant: homogeneous processes

The first model is the simplest construction, i.e. the model in its simplest form. It was devised as as little more than a baseline for further development and evaluation of the model against mostly simulated and some real data. To that end, it was used in the analysis of several smaller herd groups ($\sim 500-2000$ herds) and so is included as the first of three variants.

The rate that governs the disease process for each individual herd is identical for all (though it depends only on the history of events for that specific herd).

$$\lambda(t) = \mu + \sum_{t_\xi < t} \phi(t - t_\xi) \tag{7.5}$$

where ϕ is defined and parametrised by $\{\alpha, \beta\}$ as before. A key for the model parameters (for all three models) is given in Table 7.2, but for now note that all parameters including μ (that represents the constant ongoing force of infection, or FOI) are homogeneous for every herd. Thus we refer to this as the 'homogeneous-process' model.

7.3.2 Hierarchical model with homogeneous disease processes

The second model elaborates on the first in two ways. Firstly by introducing a hierarchical structure to the model's parametrisation; the constant external force of infection is sampled for each regional cohort from a distribution also parametrised by θ. That corresponded to county in the main analysis.

Secondly, individual herd population dynamics (alternatively, 'farmer behaviour') are accounted for indirectly by introducing a secondary variable, representing another external force of infection. The parametrisation of that process is not hierarchical; rather, it is scaled by the rate of trade observed for that herd (during each observation period). The complete event rate is thus given by:

$$\lambda(t) = \mu_j + T_i \mu_T + \sum_{t_\xi < t} \phi(t - t_\xi) \tag{7.6}$$

where μ_j is the external force of infection for the j^{th} regional cohort ϕ is defined according to (7.2). The trade-related risk parameter μ_T is scaled for the i^{th} herd by the observed trade rate for each distinct period between observations:

$$T_i = \frac{C_{y_i}}{t_{y_i} - t_{y_{i-1}}} \tag{7.7}$$

where C_i is the number of cattle purchased during the i^{th} observation period, and $t_{y_i} - t_{y_{i-1}}$ is the duration of that period (in days).

It is assumed that the [self-exciting aspect of the] disease process itself is homogeneous – parametrised by the same α and β for all herds, irrespective of cohort.

7.3.3 Hierarchical model with heterogeneous disease processes

Finally, the third model has a more hierarchical structure still. That is, the disease process parameters $\{\alpha, \beta\}$ are sampled from distribution for each [minor-] regional cohort. The model is therefore referred to as having *heterogeneous disease processes*. Trade and other aspects of the model are given the same treatment as for the second model.

Thus, $\lambda(t)$ is given as per (7.6) but the ϕ component is given by:

$$\phi(\tau) = \alpha_j \beta_j e^{-\beta_j \tau} \tag{7.8}$$

I.e. both α and β are sampled for each geographical region from also inferred 'parent' distributions. The (almost) complete parametrisation for each of the three models is given in Table 7.2. The rest are given in Table 7.1.

θ-type	Symbol	Description
1	$\mu_{[j]}$	External FOI
2	μ_T	Trade-related EFOI
3	$\alpha_{[j]}$	Parametrises the disease process
4	$\beta_{[j]}$	As above
5	σ_i	Diagnostic test sensitivity
6	M_a	Hierarchical: $\mu_j \sim \Gamma(\text{M}_a,\text{M}_b)$
7	M_b	As above
8	A_a	Hierarchical: $\alpha_j \sim \Gamma(\text{A}_a,\text{A}_b)$
9	A_b	As above
10	B_a	Hierarchical: $\beta_j \sim \Gamma(\text{A}_a,\text{A}_b)$
11	B_b	As above

Table 7.2: Partially observed Hawkes model parameters. Where applicable, hierarchical parameters μ, α and β are indexed by j for the j^{th} geographical cohort.

In the next section we describe the broad inferential framework that was used to fit the models to BTB surveillance data; i.e. the results presented in due course.

7.4 Materials and methods

Recall from Chapter 2 that the inferential framework described there can be (further) extended using Bayes' theorem. In this case we begin with the now familiar identity for parameter inference for a data-augmented model:

$$\pi(\theta, x|y) = \frac{\pi(\theta)\pi(x|\theta)\pi(y_{1:n}|x)}{\pi(y_{1:n})} \tag{7.9}$$

where the notation is the same as before, although the treatment of the χ-valued variable x is slightly different, because in this case it represents a singular, rather than coupled, Hawkes process (one for each herd).

7.4.1 Hierarchical Bayesian model

(7.9) can be extended to represent a *hierarchical* model as follows:

$$\pi(\theta_H, \theta, x|y) = \frac{\pi(\theta_H)\pi(\theta|\theta_H)\pi(x|\theta)\pi(y_{1:n}|x)}{\pi(y_{1:n})} \tag{7.10}$$

where θ_H can be (informally) referred to as the 'parent' distribution; it parametrises the distributions from which model parameters θ are sampled. Thus by implication the prior distribution is defined only for θ_H, and goal is to infer the *entire* hierarchy of model parameters represented by $\{\theta_H, \theta\}$. Note however that this is a simplistic representation. Not all hierarchical parameters are treated in the same way in the models used here. For example those associated with test sensitivity are hierarchical in one sense because they are stratified according to test type. However they are not sampled from a 'parent' distribution in the way described above, as others are.

As before, a key benefit to this [i.e. Bayesian] approach to parameter inference is that it allows us to characterise the observational data in a manner so as to account for arbitrary degrees of complexity and intrinsic uncertainty in the observation process itself.

Likelihood function for multiple Hawkes processes

Extending the definition given in Chapter 2 for univariate Hawkes processes, the probability density function associated with a system of N_h processes (herds in this case) can be written as:

$$\pi(x|\theta) = \prod_{i=1}^{N_h} \prod_{j=1}^{|x_i|} \lambda(t_j^i) e^{-\int_T \lambda(t)dt} \tag{7.11}$$

where j indexes the j^{th} event for the i^{th} process. The precise definition of the rate quantity denoted by λ (and thus the appropriate method for computing it) depends on the specific parametrisation of the process that is used. In this case the parametrisation and recursive method described by [33] was used to compute (7.11), as per the definitions and description given in §2.2.3.

7.4.2 Implementation: DA-MCMC algorithm

The algorithm used to solve (7.9) for each of the three models was data-augmented MCMC, with a generic description of the technique already given in §2.4.2. As per that section, the Metropolis-Hastings acceptance probability for a generic DA-MCMC scheme is given by:

$$p_{mh} = \min\left\{1, \frac{q(\theta_i, x_i|\theta_f, x_f)\pi(\theta_f, x_f|y)}{q(\theta_f, x_f|\theta_i, x_i)\pi(\theta_i, x_i|y)}\right\} \tag{7.12}$$

where q is the proposal density. This is the probability density associated with the modular component of MCMC algorithms often referred to as the 'kernel'. Recall from Chapter 2 that in MCMC $q(a|b)$ is the probability associated with *proposing* sample a given the 'current' sample, b.

Recalling also that the marginal likelihood $\pi(y_{1:n})$ in (7.10) cancel when computing the acceptance probability, we can write an expression to compute (7.12) for the hierarchical Hawkes model as follows:

$$\pi(\theta_f, x_f|y) = \pi(\theta_H)\pi(\theta_f|\theta_H)\pi(x_f|\theta)\pi(y|x_f) \tag{7.13}$$

where $\pi(\theta_H)$ is the prior distribution and θ_H parametrise the sampling distribution of the model parameters, denoted as always by θ. The precise definition of associated densities naturally depends on the choices of sampling distribution, but in general are trivial to compute. The other terms, the likelihood function and the observation likelihood, are given by (7.11) and (7.4) respectively and are likewise reasonably inexpensive to compute, even for systems comprising thousands of herds.

As with many DA-MCMC algorithms and random sampling schemes in general (certainly, all those contemplated in this thesis) the most important design consideration is the choice of proposal density q. A number of different strategies based on Gibbs sampling were employed, with four major variants and additional configurable properties.

An adaptive, multivariate Gaussian distribution was used for parameter proposals, in the same manner as in previous chapters.

Another strategy employed throughout involved the partioning of the processes (or herds) into those with at least one reactor (across the entire time series) and those without. This allowed for computation (i.e. proposals) to be directed more towards the former. The primary strategy employed was to propose new x (sequentially for each herd) for all herds with observed reactors but only with a given probability for those with no reactors.

The impact this has in terms of improved mixing (w.r.t. θ) can be rationalised by considering that herds with e.g. complicated underlying patterns of infection require a larger number of proposals in order to obtain reasonably good samples.

New x were proposed using a combination of strategies that blended standard MCMC moves with a technique that can be thought of as 'seeded' simulation. This was motivated by the fact that sampling from the density associated with the stochastic process, i.e. 'simulating', is a naturally efficient approach to random sampling in this situation. However naively simulating realisations of x without respect to y led to many samples with no disease history at all (e.g. because the background/trade rate was too low for a given parametrisation). 'Seeding' a single infection time by sampling from a uniform distribution bounded from t_0 to the time of the first, non-zero observation mitigates this problem.

Finally, varying combinations of proposal types (i.e. θ or x) were employed in staggered blocks in order to 'leapfrog' the joint parameter and process (or 'measure') space, somewhat similar to the manner conceptually described by [55] as discussed in §2.4.5 on the topic of 'waste recycling' in MCMC.

7.5 Results

We now present the main results of the inference analyses. They are organised as follows:

1. Case study: Wiltshire (model one only)

2. Model one: homogeneous processes

3. Model two: hierarchical model with homogeneous disease processes

4. Model three: hierarchical model with heterogeneous disease processes

The overall objective of the exercise was to iterate towards a hierarchical model that is scalable up to a nationwide level, which corresponds to systems comprised of many thousands of herds. Due to the experimental nature of the models and methods developed towards that end, there is a joint focus on algorithm performance (i.e. convergence) and the estimates obtained. This is most evident from the meta-analyses provided for each of the three models, where we report key parameter estimates along side the estimated potential scale reduction factors (i.e. the Gelman-Rubin convergence diagnostic) in visual summaries.

Among other things, this helps to elucidate the extent of the algorithms scalability to models with larger numbers of parameters and herd groups. In that respect, the algorithm performed reasonably well for the first model across all groups, but struggled to converge in some other cases, particularly for the third model. These issues are discussed in due course.

Each of the models was used to analyse multiple herd groups, though only a subset of the results have been included here (even within most of the meta-analyses) for practical reasons. The data for each of the reported herd groups are summarised in a table at the top of the results section for each model.

First though, in order to give a flavour of the individual analyses that comprise those results, we provide an individual set for Wiltshire as a case study.

7.5.1 Case study: Wiltshire

We begin with a short summary of the surveillance data recorded in VetNet for the Wiltshire herd group. Similar summaries are provided for each analysis (i.e. model) and herd group, in Appendix E.

Summary of surveillance data

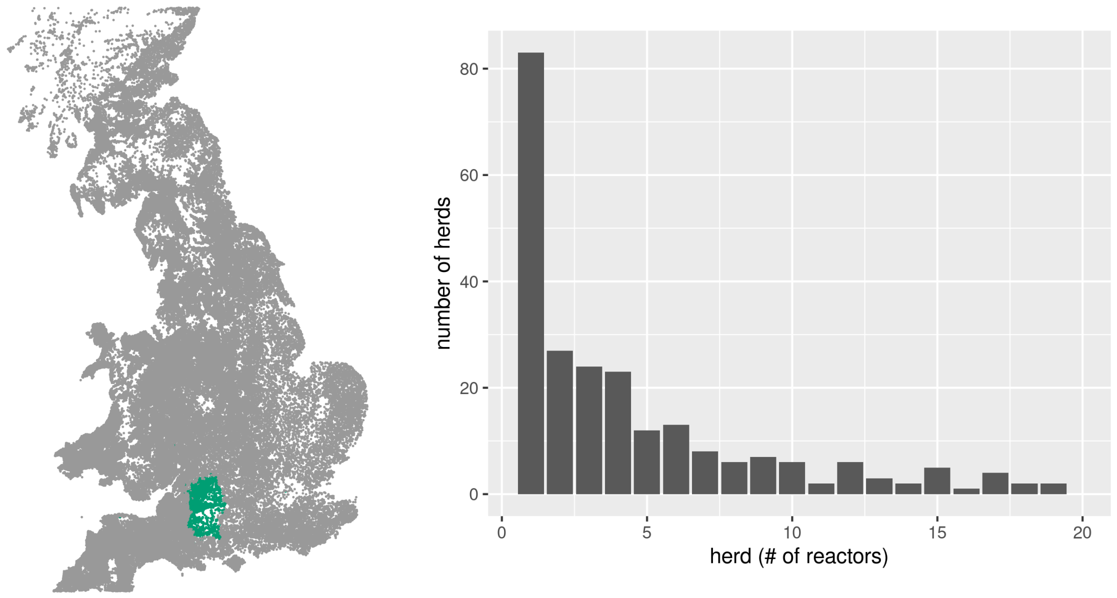


Figure 7.4: Herd and reactor distribution. The latter is given for the total number reactors by herd for the entire herd group.

Number of herds: 851
Reactor herds: 258
Number of reactors, i.e. animals: 1801

σ_i	animals	+ve
1	4371	478
2	1800	35
3	3901	1225
4	36	63

Figure 7.5: VetNet surveillance data selected for the Wiltshire herd group. The diagnostic test data are categorised as follows: standard SICCT herd tests; high-sensitivity SICCT herd tests; risk-based surveillance testing; and the IFN$_\gamma$ blood test trial, denoted σ_i for 1:4 respectively.

Inference summary for the homogeneous process model

Here we provide a set of parameter estimates for the basic seven-parameter model for the Wiltshire herd group. A tabulated summary of them is given in Table 7.3. Multiple independent analyses were carried out for each model and herd group, in part to identify approximately optimal configurations of the algorithm. The results reported here were computed from three MCMC chains, each consisting of one hundred thousand samples (with the first fifth discarded).

θ	Mean	SD	$\hat{\text{R}}$	97.5
μ	0.00001	0.00000	1.0	1.0
α	0.55900	0.18800	1.0	1.0
β	0.42900	0.25800	1.1	1.3
σ_S	0.42100	0.11500	1.0	1.1
σ_H	0.35300	0.09710	1.0	1.0
σ_R	0.46000	0.07500	1.2	1.5
σ_γ	0.82600	0.09310	1.0	1.0

Table 7.3: Homogeneous process model parameter inference results for Gwynedd. Number of herds (reactors) := 1585 (95).

Discussion of the parameter estimates is reserved for the following sections, since the meta-analyses presented for each model give a better indication of likely distribution than a single analysis. However the results given in Table 7.3 are indicative of the fact that the model and algorithm seemed to scale reasonably well for herd systems of this size, as indicated by the potential scale reduction factors being mostly $\hat{\text{R}} < 1.2$, though again this is perhaps easier to judge from the meta-analyses. Finally, marginal distributions for these results are illustrated in Figure 7.6. They, and also the traceplots given in the appendices, highlight that the lower bound of the prior distribution for σ_H may be too low (though not for σ_S which is curious). Somewhat in keeping with the commentary provided on Bayesian workflows in prior chapters (i.e. that they are circular and iterative) this was amended to be 0.2 for analysis of the two additional models.

We now present the aggregated results for this model, followed by the other two.

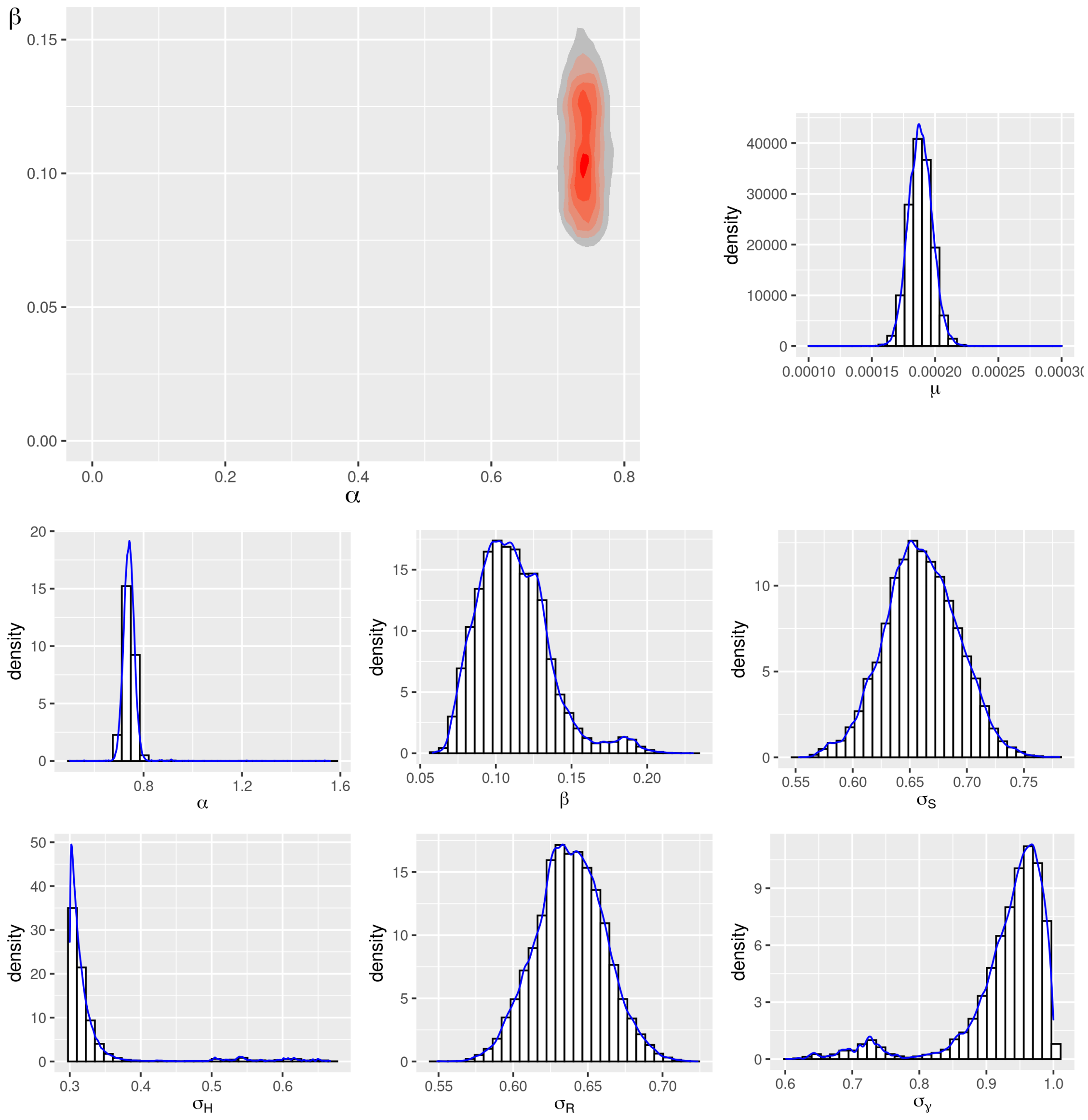


Figure 7.6: Marginal sample densities for Wiltshire.

7.5.2 Homogeneous process model

Here we present aggregated results for the basic, seven-parameter model. The analysed data are summarised for each individual herd group listed in Table 7.4.

Herd group	No. of Herds	Reactor herds	Reactors
North east England	1545	40	135
Cheshire	1364	152	955
Oxfordshire	502	35	242
Avon	632	161	737
Dorset	971	196	849
Wiltshire	851	258	1801
Gwynedd	1585	46	95
Gwent	612	204	1816
Powys	1763	571	3298

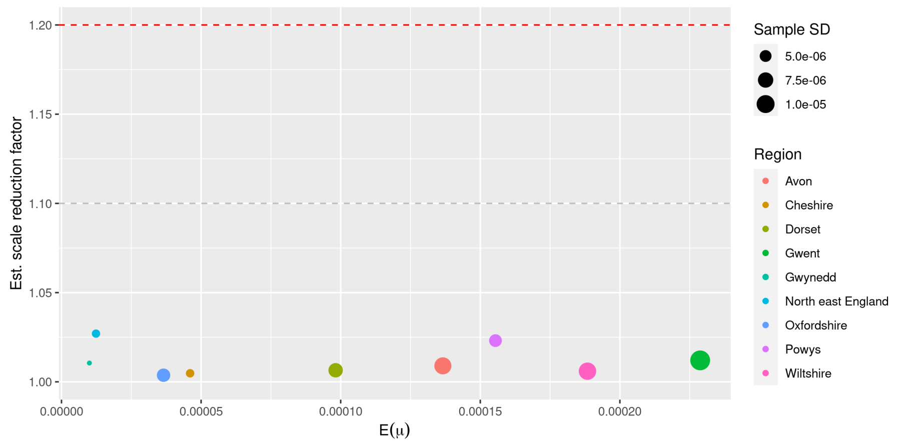
Table 7.4: Herd groups analysed using the homogeneous process model.

This model was developed as a prototype towards a fully hierarchical Bayesian model that was anticipated would be necessary to accommodate herd systems comprised of hundreds of herds (let alone thousands). As noted in §7.5.1 however, fitting the model for moderately large herd systems (c.500 to 1,500 herds) proved reasonably straightforward. That is not to say that the model is a good fit for that data. That is difficult to judge because the work has not yet been developed sufficiently to employ formal methods of model assessment and comparison. However the results were encouraging enough to motivate development of the second and third models. We now report a selection of estimates for each parameter and herd group.

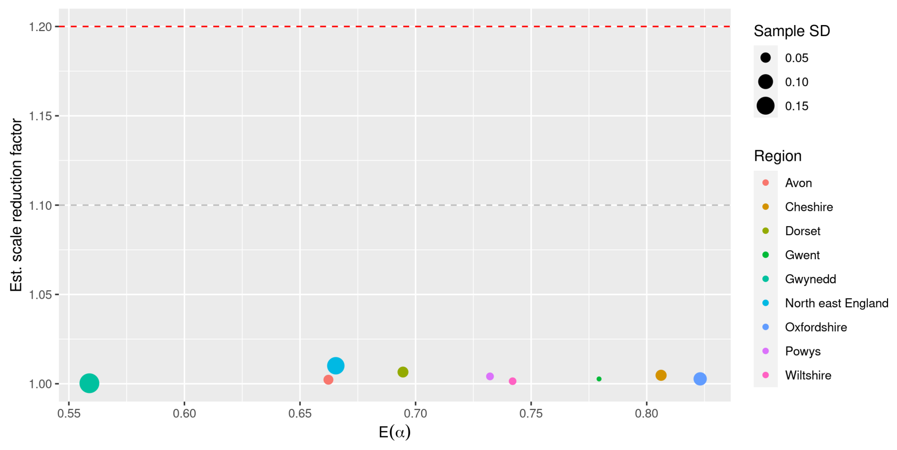
Parameter inference summary

Here we report estimates for each parameter and herd group. The results are given in the form of visual summaries that highlight both the mean and standard deviation of posterior samples (x-axis) and the degree of convergence achieved (y-axis). The latter is also indicated with markers at the conventionally accepted thresholds of $\hat{\mathrm{R}}$= 1.1 and 1.2. The sample standard deviations are indicated by the size of the corresponding marker. The first parameter group, illustrated in Figure 7.7, are the three that directly relate to the hidden disease process, denoted by $\{\mu, \alpha, \beta\}$. As per the model description, μ is the background force-of-infection and $\{\alpha, \beta\}$ parametrise the decay kernel of the Hawkes process for every herd. The [expectations of the] estimates obtained for μ and α are spread fairly evenly w.r.t. their own individual variances but those obtained for β were bunched, with the exception of Gwynedd (which had a low number of reactors for its size in any case). This was noted as possible justification for a partially hierarchical approach to the disease process, in which α could be preferentially selected for stratification over β. However a fully hierarchical model was eventually chosen (for the third model) instead.

The second parameter group relates to the observation model, they are the detection probabilities associated with four diagnostic test categories. They are illustrated in Figures 7.8 and 7.9. There is a marginal amount of additional consistency in those estimates, compared to the first parameter group. The estimates for the IFN_γ in particular are clustered, with the sample mean around 0.9 and mostly comparable in terms of variance. The same is true of σ_H. However the the fact that the samples are (again) bunched against the lower bound of the prior distribution on that parameter perhaps renders that result rather less significant. At any rate, it led (once again) to the selection of a weaker prior distribution still for the subsequent analysis.



(a) Constant FOI parameter estimates.



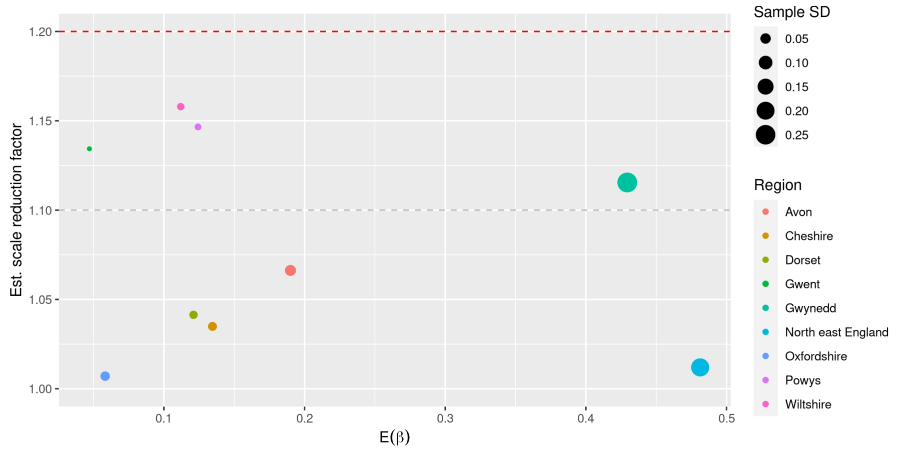
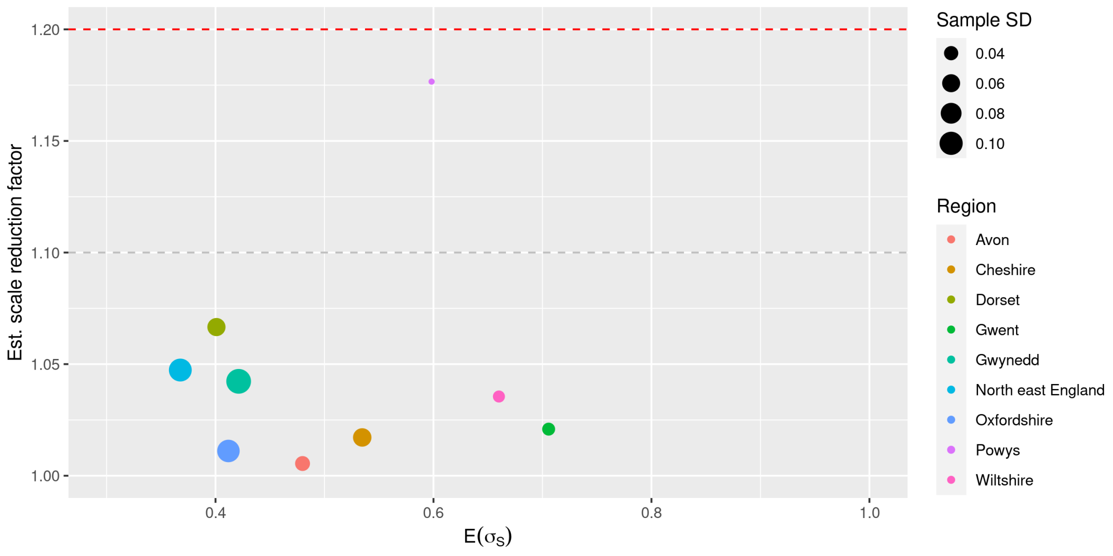
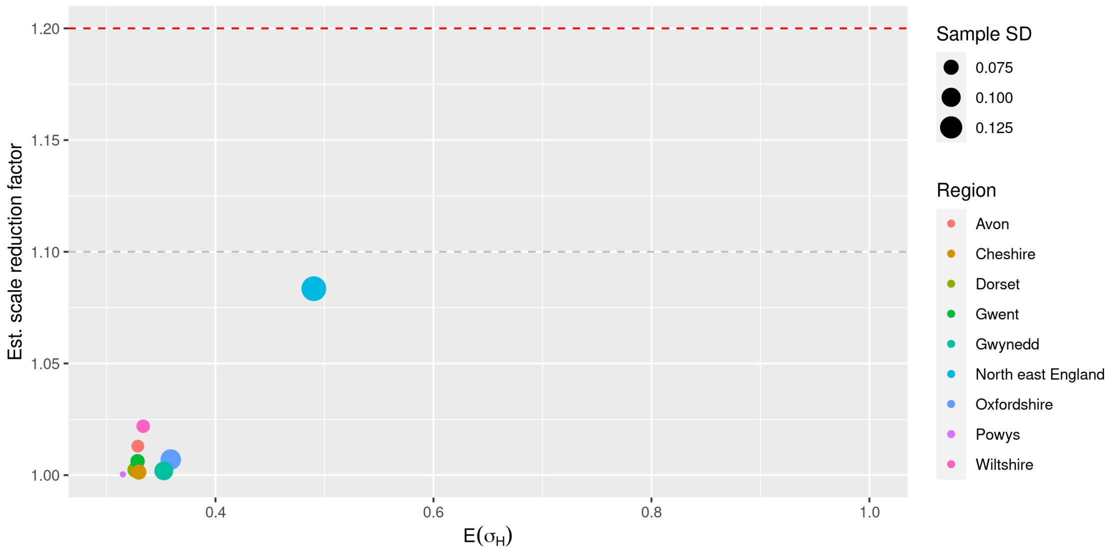


Figure 7.7: Disease process parameter estimates $\{\mu, \alpha, \beta\}$.

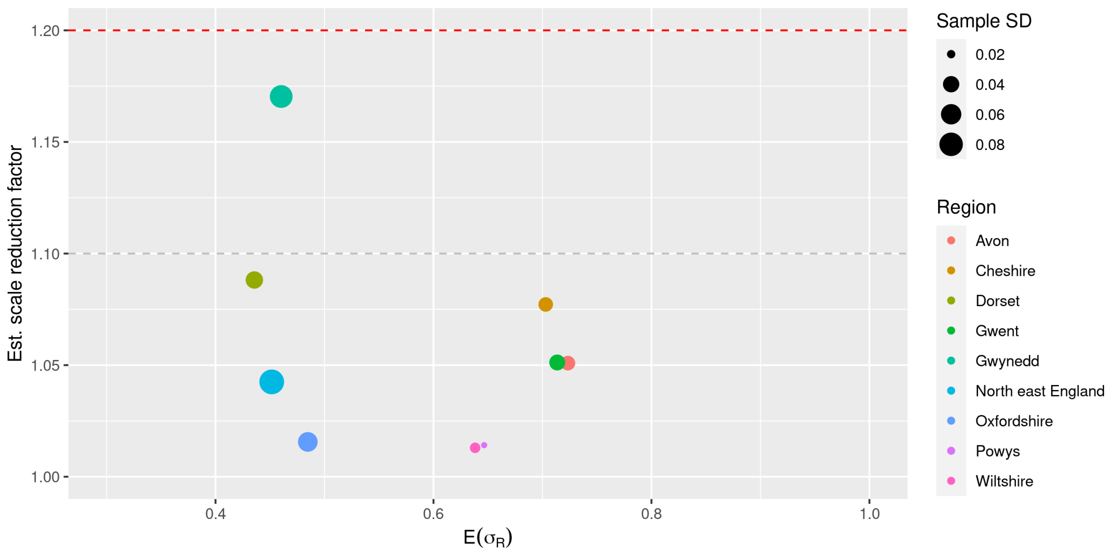


(a) Standard sensitivity.

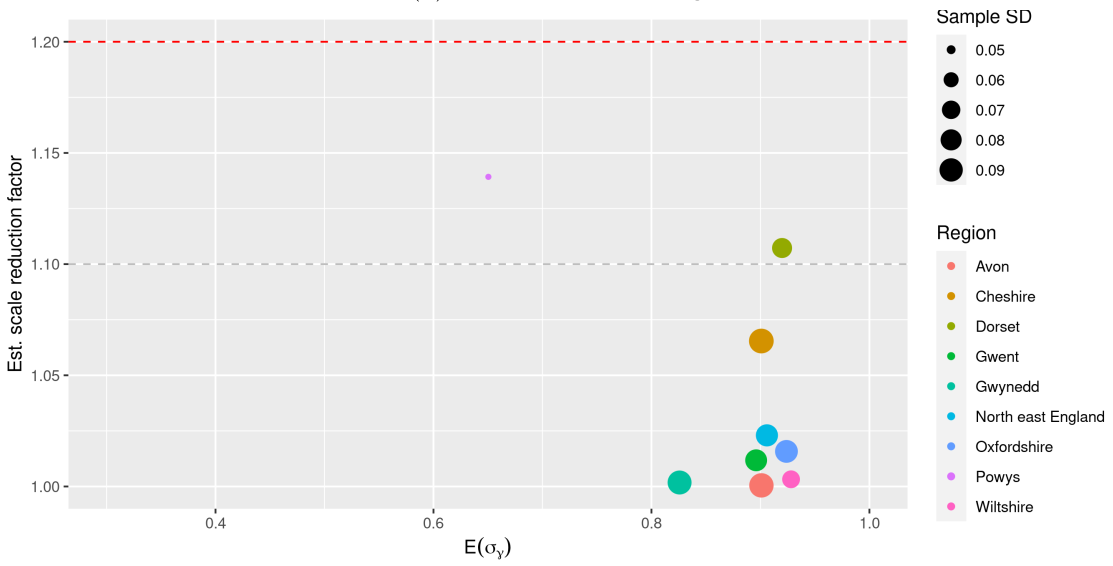


(b) High sensitivity.

Figure 7.8: Diagnostic test sensitivity parameter estimates: SICCT test.



(a) Risk-related testing.



(b) IFN$_\gamma$ blood test.

Figure 7.9: Diagnostic test sensitivity parameter estimates: other.

7.5.3 Homogeneous disease process-hierarchical model

The second model introduces a degree of heterogeneity in the form of county-level background, or external force-of-infection, risk. Another elaboration on background risk in this model (and the subsequent one) is the introduction of an [external] trade force-of-infection risk component. The latter is homogeneous w.r.t. the *parameter*, μ_T, across herds but the corresponding component risk scales with the observed rate of trade. As before, the data analysed are summarised in Table 7.5.

Herd group	No. of Herds	Reactor herds	Reactors
East England	513	13	75
East Midlands	1966	231	1217
North east England	1015	22	103
North west England	3432	190	989
South east England	1393	78	357
Scotland	4510	77	280

Table 7.5: Herd groups analysed using the homogeneous disease process-hierarchical model.

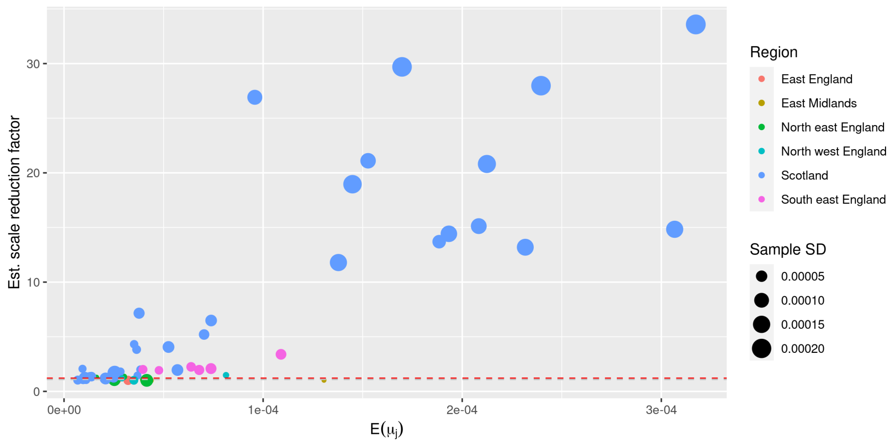
Parameter inference summary

Mainly for reasons of brevity, we have included a visual summary of inference results for only a selection of parameters and herd groups from the second analysis. Unfortunately this effort was less successful than the last, at least in terms of algorithm convergence, even though the range of herd group sizes (see Table 7.5) was not excessively large compared to those analysed using using the first model. It has been our experience throughout that computational problems can arise when the model in contention is a poor fit for the data. However the fact that the algorithm converged for the initial model (including simulated versions with two thousand herds) led to the suspicion that the algorithm was merely inadequate with respect to the increased dimensionality of model.

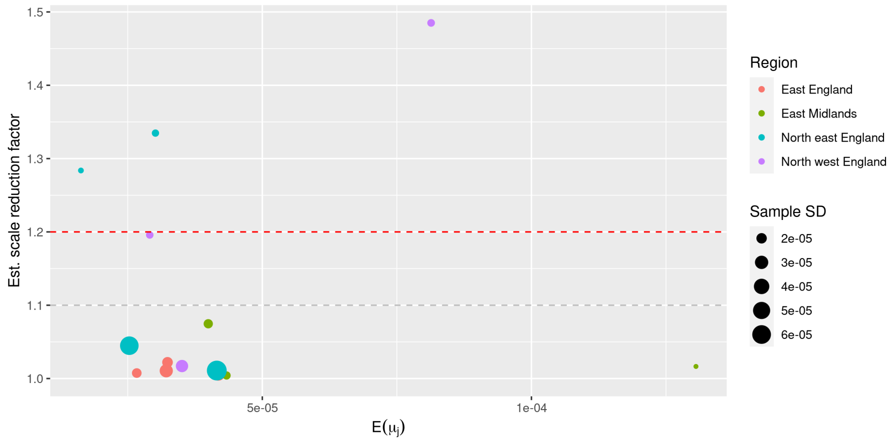
The estimates reported here (i.e. in the main text of the chapter) relate exclusively to force-of-infection risk, trade related and regional background. They are denoted by μ_T and μ_i respectively, where i indexes the i^{th} regional cohort. Unlike the other disease-process related parameters, μ samples did converge across

Markov chains. At least, for most herd groups as illustrated in Figure 7.10. Those plots highlight that algorithm conspicuously failed to converge for the largest herd group, Scotland with c.4,500 herds. However that may also be due in part to very low incidence of BTB in Scotland generally, leading to paucity of information in the data. Excluding that herd group (Figure 7.10b) we see that the estimates for μ_i are bunched across both regions and herd group with the exception of one outlier group in the East Midlands (and discounting one other partial outlier in NW England that may have simply failed to converge).

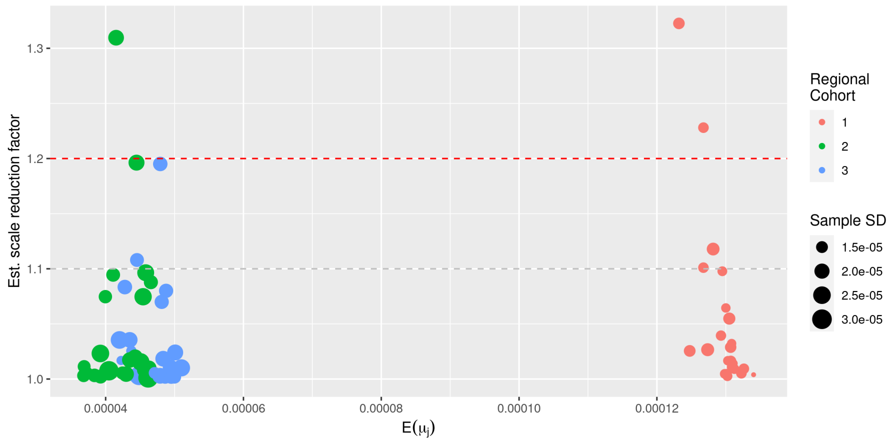
As noted before, the results reported here (including the meta-analyses and in the appendices) represent only a fraction of the analyses that were actually conducted for this work. However all results were recorded in a SQL database and analysed to ascertain the veracity of the outlying data point in the East Midlands group. That analysis is shown in Figure 7.10c. The results suggest that the data point is indeed valid. The estimates for μ_T are given in Figure 7.11. They seem to indicate that trade-related risk varies significantly by herd group (i.e. broad geographical region). It is also interesting to note that the high-reactor regions had estimates that were close to opposite ends of the spectrum, since it would seem to suggest the result is not merely an artefact of correlation with the number of reactors or general prevalence of BTB in a given region. All other parameter summaries for this analysis are provided in Appendix E.2.



(a) All herds



(b) As above, less Scotland and SE England.



(c) East Midlands only

Figure 7.10: Constant FOI parameter estimates.

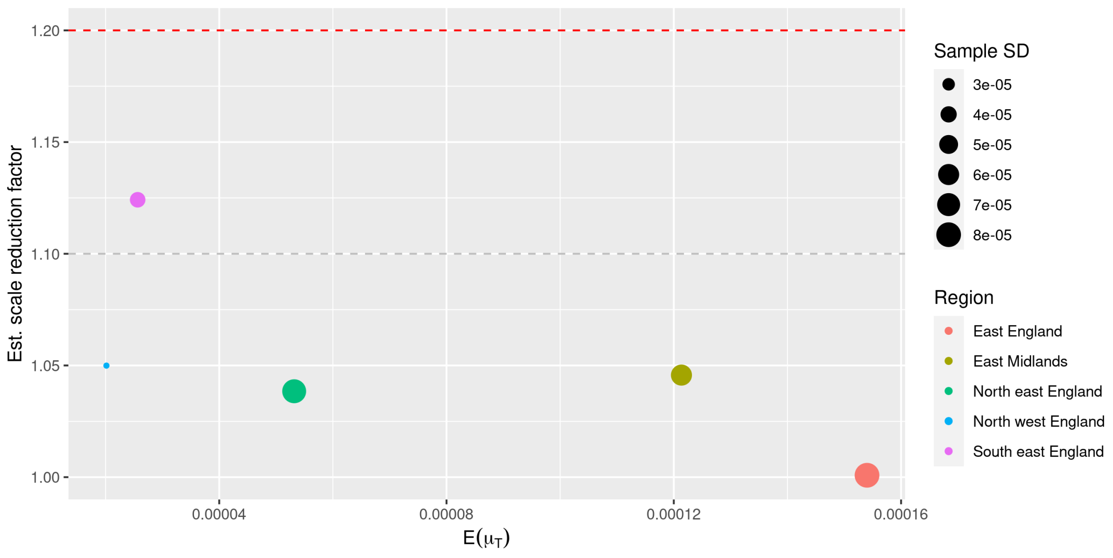


Figure 7.11: Trade-related FOI parameter estimates.

7.5.4 Heterogeneous disease process-hierarchical model

Here we report results from analysis of the third and final model. This model expands on the hierarchical nature of the last by introducing regional stratification for $\{\alpha_j, \beta_j\}$ in addition to μ_j. As per Table 7.6, only three herd groups have been included. These were among the smaller regional groups, and selected mainly as computationally efficient test cases.

Herd group	No. of Herds	Reactor herds	Reactors
East England	513	13	75
East Midlands	1966	231	1217
North east England	1015	22	103

Table 7.6: Herd groups analysed using the heterogeneous disease process-hierarchical model.

Parameter inference summary

The estimates obtained for the third model seemed to indicate that the hierarchical model was scalable at least to this degree. That is, convergence appeared to be mostly reasonable for all parameters in contrast to the second set of results. As per Figure 7.12, the sample mean of estimates for μ_i for the East Midlands group were also more consistent than before (recall the outlier from those results in the previous set of results) and the sample mean for μ_T is very similar to the one obtained using the previous model.

A final thing to note about the results is an interesting distinction between the estimates for the disease process parameters $\{\alpha_j, \beta_j\}$. The sample means for the first are evenly spread, where was the first exhibit strong regional correlation. This is difficult to explain but may indicate hidden correlations within the model, else it could suggest that β would be more appropriately stratified at the *broad* regional level (rather than county as in this case), As before, the parameter estimate summaries not included here are given in the appendices, in E.3. We now move on to a cursory simulated inference analysis for the purposes of validation, before concluding with a discussion concerning these results in their entirety and possibilities for future work.

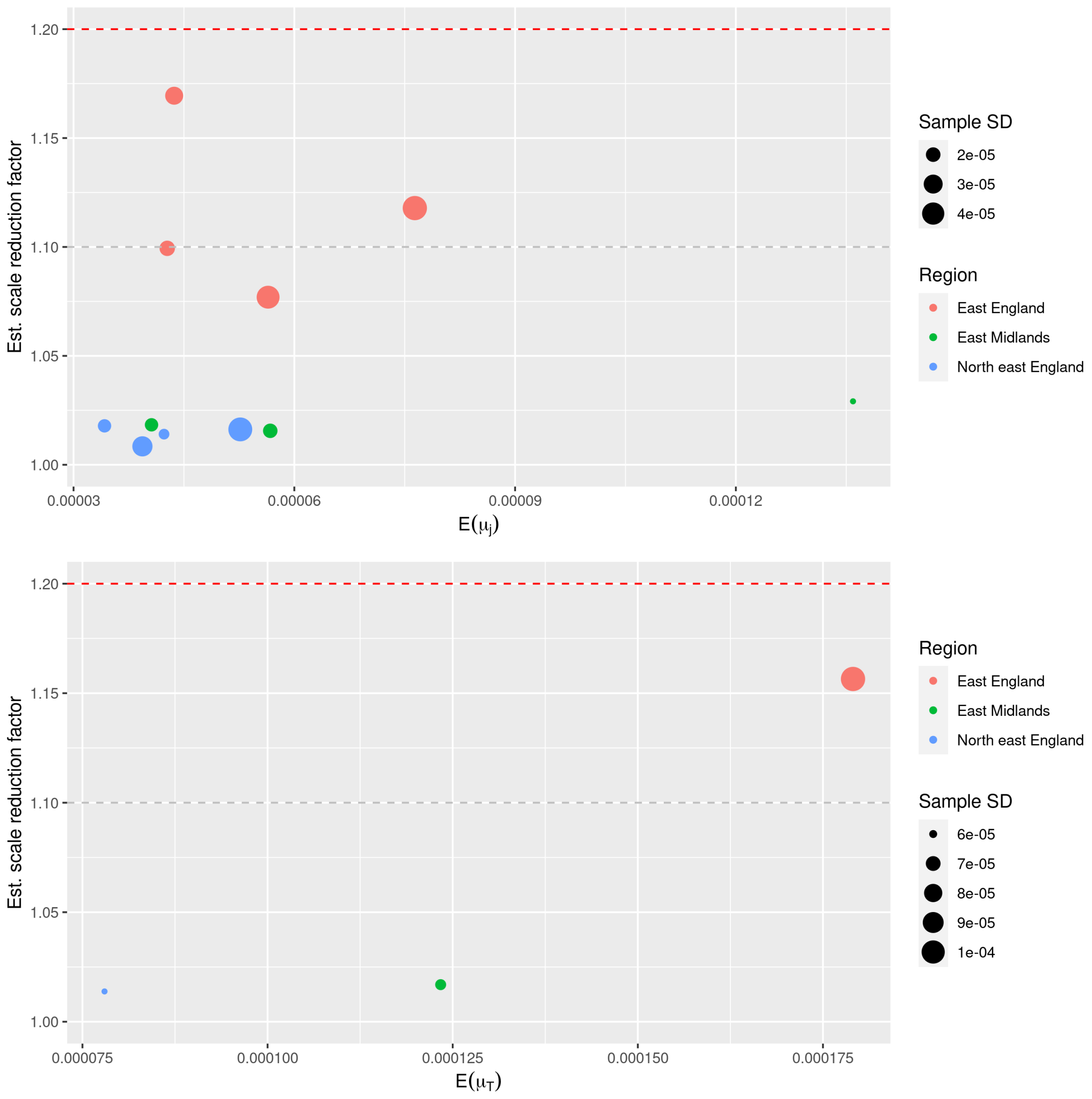


Figure 7.12: Constant FOI parameter estimates.

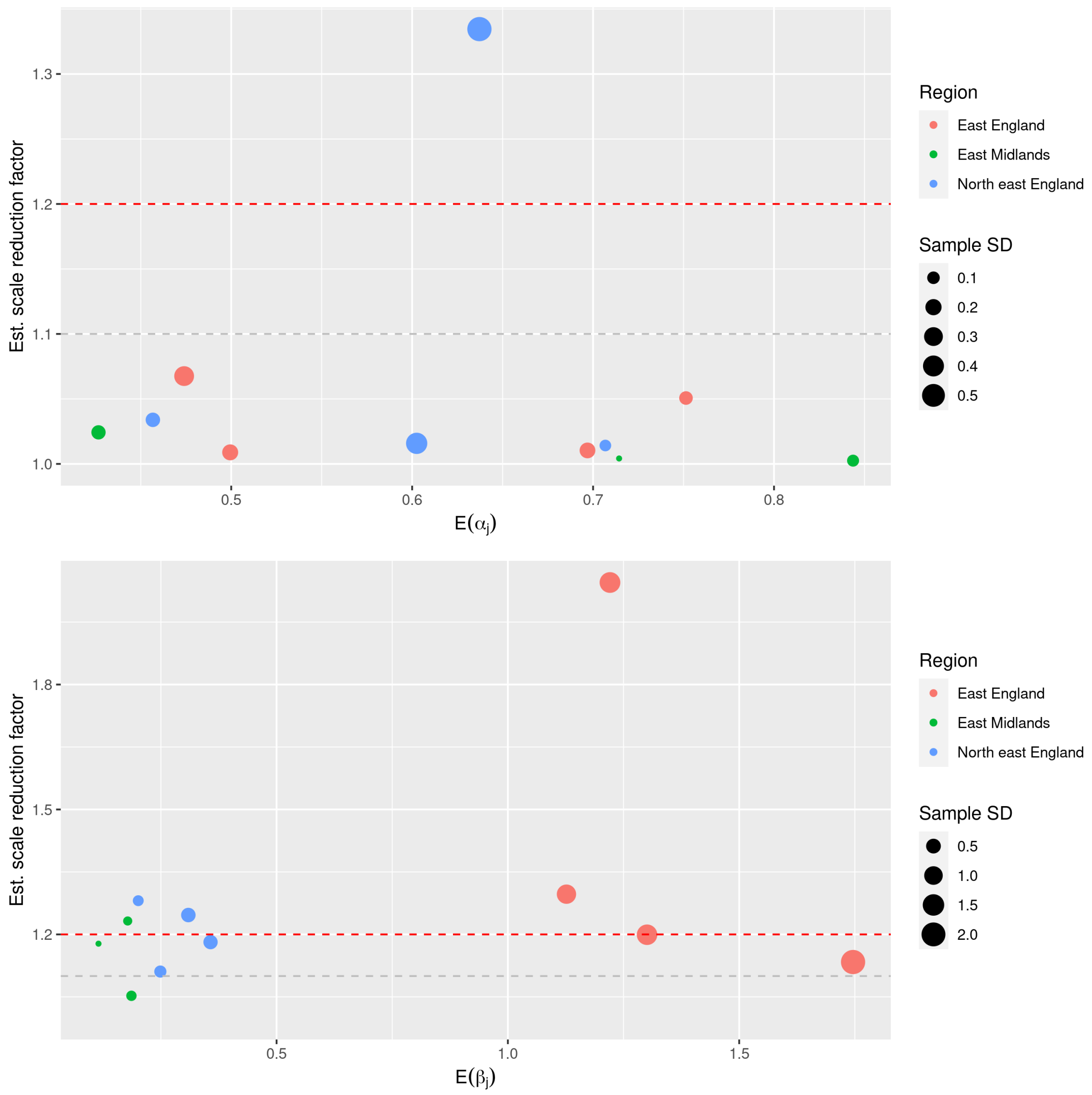


Figure 7.13: Disease process parameter estimates $\{\alpha_j, \beta_j\}$.

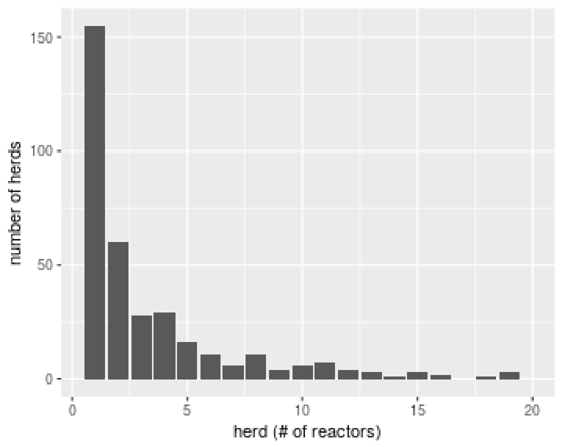
7.6 Validation: simulated inference

Here we report the results of an analysis based on simulated observations data that was carried out as a validation exercise. In particular, to help elucidate the efficacy and effective scale of the algorithm during its development.

7.6.1 Simulated surveillance data

The data for inference were simulated based on a system of two thousand farms, with 3168 reactors 'observed' in total during the simulated scenario. The parameters used to produce the simulation have been marked on the appropriate graphs for reference, e.g. see Figure 7.15.

As per Figure 7.14, the simulated data (this one and other simulations that were run) are comparably *overdispersed* in relation to real data, indicating that the model is an approximately good fit for the data, or at least isn't obviously a bad fit for the data.



σ_n	tests	+ve
1	2963	565
2	2975	890
3	3003	909
4	3059	804

Figure 7.14: Simulated disease surveillance data intended to (loosely) represent a subset of the VetNet data set. The model parameters are denoted $\theta = \{\mu, \alpha, \beta, \sigma_{1:4}\}$. Diagnostic test types are labelled σ_n and organised into four distinct categories within the model corresponding to: *routine*; *trade*-related; *risk*-induced, e.g. short interval testing, six-month 'follow-up' visits; and the *IFN* test trial in the real data set.

7.6.2 Parameter inference results

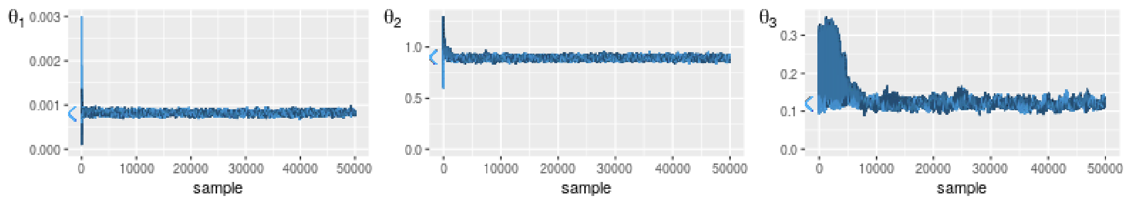
The analysis for the simulated study were based on the first iteration of the model. The parameter estimates obtained closely matched the simulation values, as can be seen in the marginal densities illustrated in Figure 7.16 in particular.

The algorithm also performed well, with $\hat{R} < 1.02$ indicating a strong degree of consistency across Markov chains (see Table 7.7).

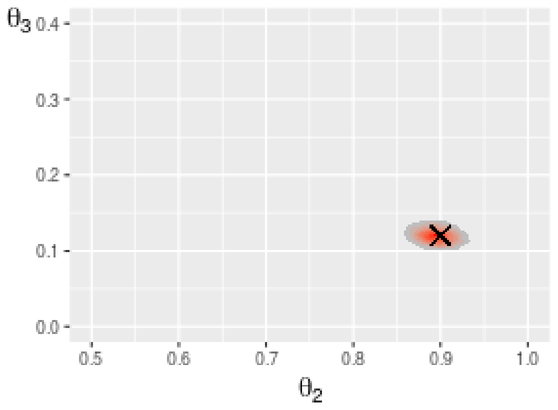
Parameter	**Simulation**	**Mean**	**SD**	$\hat{R}$	**97.5**
1: μ	0.0008	0.000822	4.27E-05	1.002	1.003
2: α	0.9	0.896	2.09E-02	1	1.001
3: β	0.12	0.127	3.40E-02	1.018	1.043
4: σ_1	0.5	0.476	3.65E-02	1.003	1.005
5: σ_2	0.7	0.685	3.47E-02	1.003	1.009
6: σ_3	0.8	0.764	2.99E-02	1	1.001
7: σ_4	0.85	0.833	2.67E-02	1.011	1.03

Table 7.7: DA-MCMC parameter inference summary for the simulated surveillance data analysis.

Additional simulation studies are necessary to both advance the design and further validate the efficacy of the algorithm, especially since the results presented here concern only the most basic version of the model. That topic and other further work are discussed in the concluding part of the chapter.

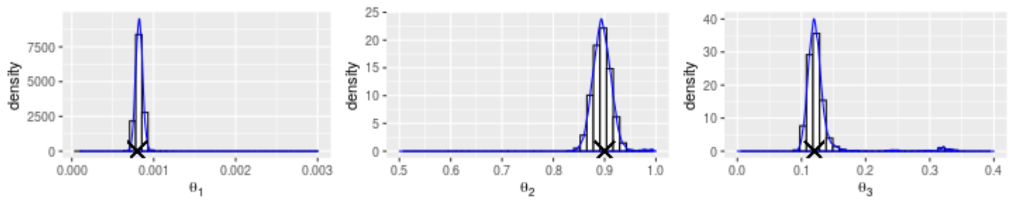


(a) Trace plots

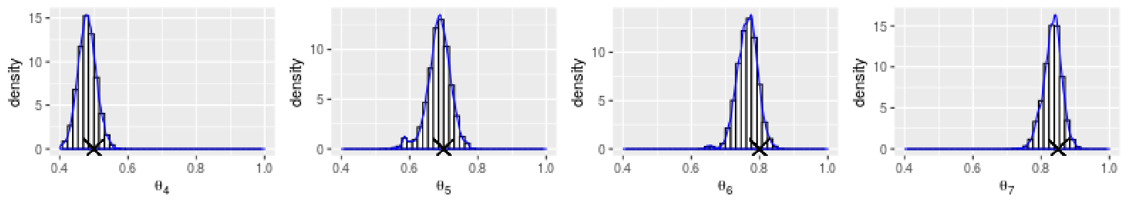


(b) $\theta_{2:3} = \{\alpha,\ \beta\}$

Figure 7.15: Parameter inference results for $\theta = \{\mu, \alpha, \beta, \sigma_{1:n}\}$. Tests are grouped by four categories in the model: *routine*; *trade*-related; *risk*-induced, e.g. six-month 'follow-up' visits; and the *IFN* test trial.



(a) $\theta_{1:3} = \{\mu, \alpha, \beta\}$



(b) $\theta_{4:7} = \sigma_{1:4}$

Figure 7.16: Parameter inference results for $\theta = \{\mu, \alpha, \beta, \sigma_{1:n}\}$. Tests are grouped by four categories in the model: *routine*; *trade*-related; *risk*-induced, e.g. six-month 'follow-up' visits; and the *IFN* test trial.

7.7 Summary

A key objective of this work was to develop models and methods scalable to systems comprised of many thousands of herds, in contrast to the results accomplished in the previous chapter which ran to less than twenty individual herds. Substantial progress has been made towards this end. We have presented the results of three separate analyses, each consisting of three or more herd groups which together comprise thousands of herds.

This was partly due to the construction of the model. Using only a single stochastic process rather than a coupled set of them to represent each individual herd led to a model that was more computationally tractable, i.e. easier to solve.

We have also reported tentative estimates for a range of interesting parameters related to BTB, notably external force-of-infection ('background') risk; trade-related risk; and detection probabilities associated with key diagnostic tests. Unfortunately those results can only be described as 'tentative' because the work was constrained by the time available and there is much still to do.

Further work

We conclude the chapter by laying out five key areas for further research and development of this work:

1. Further simulation study
2. Investigating and further automating the curation of data
3. Algorithm development
4. Model development
5. Tools for formal model assessment and comparison

7.7.1 Simulation studies

The models and algorithm were validated throughout using simulated inference studies as a natural part of the development process, although only the results for the basic model have been formally analysed and included in the chapter. This was mostly due to constrains on the time available, and also because the models presented here were considered as works-in-progress rather than models fit for publication.

In any case, further simulation studies including predictive checks of the kind described in Chapter 4 are required to advance both model and algorithm development.

7.7.2 Automated curation of data

Data quality proved to be a challenge throughout the project, in particular with regard to the *VetNet* data. This issue was covered extensively in Chapter 6, and some additional work was done to automate the curation of data for the work done in this chapter, in particular w.r.t. the classification of test codes.

However further investigation is required to ascertain whether anomalous (but worryingly consistent) estimates for the detection probability parameters were the result of incorrectly classified test codes, or else some other unrecognised pattern of usage in VetNet that hasn't yet been accounted for in the processing of records.

7.7.3 Algorithm development

The algorithm performed reasonably well for the first analysis but convergence became noticeably harder to achieve for certain parameters as stratification (and therefore dimensionality) in the model increased.

Overall, counties and regions with high incidence of BTB were found to be more tractable for analysis. That is not particularly surprising, since it is consistent with the same findings and challenges that led to the necessarily stringent data selection criteria adopted for the analysis presented in Chapter 6.

However the technique would likely benefit from additional simulation study and design tweaks oriented around the increasingly hierarchical structure of the model. For example, w.r.t. parameter proposals, since they are relatively easy to manipulate and optimise compared to the process variable that encodes the system trajectory.

7.7.4 Model development

As noted above the models presented in this chapter were regarded as still in a state of development at the time this document was being drafted. Recall that a key avenue of investigation w.r.t. the models presented in this chapter (and the previous one) relates to the question of *reinfection* and the role it plays in within-herd BTB dynamics.

This led to conceptual designs for three further models, one of which was incorporated within the analyses carried out for Chapter 6 (the work did not proceed in the precise order that it is presented here). Each of the models are intended to account for the possibility of reinfection, and but differ mainly in their precise treatment of the infection and reinfection processes. For example, there is no direct relation between the onset of infectiousness and disease state in the last model (7.7.4) which instead accounts for both infection and reinfection as mutually exciting processes. For that reason it is labelled SET (for test-sensitive) rather than SEI.

Reinfection as a Poisson process

The first model (depicted in Figure 7.17) is the one that was adopted for the analysis presented in Chapter 6 and referred to as the 'simple reinfection model'. It is formulated using coupled Poisson [point] processes in the standard manner for DPOMP models. However unlike the standard formulation of an SEI model, the presence of individuals in the infectious state is also assumed to 'excite' (i.e. increase) the $E \to I$ event rate. Hence, it is still a Poisson process model and accounts for [the possibility of] reinfection of latently infected individuals in a still Markovian way.

$$\begin{tikzcd}
\boxed{\mathrm{S}} \arrow[r, "\beta\mathrm{SI}"] & \boxed{\mathrm{E}} \arrow[r, "\gamma\mathrm{E}+(\beta)\theta\mathrm{EI}"] & \boxed{\mathrm{I}}
\end{tikzcd}$$

Figure 7.17: Poisson [point] process reinfection model. The compound rate for $E \to I$ incorporates the possibility of reinfection.

Reinfection as a Hawkes process

The second proposed model introduces a time-decaying self-excitation component for the $E \to I$ event rate, as an alternative means of accounting for reinfection (and is thus non-Markovian). It is a finite population model based on the *HawkesN* framework of [24].

$$\begin{tikzcd}[column sep=huge] \boxed{\text{S}} \arrow[r, "\beta\text{SI}"] & \boxed{\text{E}} \arrow[r, "\nu_E[\mu + \sum_\gamma \phi(t-t_\gamma)]"] & \boxed{\text{I}} \end{tikzcd}$$

Figure 7.18: Hawkes process SEI model self-exciting reinfection process $E \to I$. The model improves (perhaps!) on the models presented in this chapter by directly accounting for the presence of infectious individuals.

The 'γ' summation shown in Figure 7.18 is over all $E \to I$ events up to time t, and the 'self-exciting' kernel function ϕ is of a form like the one given by (2.11). The first term in the $E \to I$ event rate equation is given by:

$$\nu_E = 1 - \frac{N - E}{N} \tag{7.14}$$

Coupled Hawkes processes

The third model conceived for possible further investigation is the [Hawkes process] susceptible-exposed-test sensitive (SET) model. The bivariate event rate is mutually- (and self-) exciting. The 'test-sensitive' state is intended only to represent immune response and not the (permanent) infectiousness of individuals in that state. This is indirectly captured by excitation of the $S \to E$ event rate by events up to time t, including those that represent the $E \to T$ transition.

$$\begin{tikzcd}[column sep=huge] \boxed{\text{S}} \arrow[r, "\nu_S[\mu_\beta + \sum_\xi \phi_\beta(t-t_\xi)]"] & \boxed{\text{E}} \arrow[r, "\nu_E[\mu_\gamma + \sum_\xi \phi_\gamma(t-t_\xi)]"] & \boxed{\text{T}} \end{tikzcd}$$

Figure 7.19: Hawkes process SET model. The model is designated so because disease state does not have a direct bearing on transmission. In other words the infectiousness of individuals is not assumed to be dependent on the, e.g. immune response, which informs diagnostic testing. New infections are instead modelled as a finite population [of size S] Hawkes (i.e. HawkesN) process. Likewise for the 'immune response' transition event $E \to T$.

In this case the summations given in Figure 7.19 are over the entire (bivariate) event history and the kernel functions ϕ_β and ϕ_γ are (or can be) defined in the form:

$$\phi_{\beta}(\tau) = \beta\theta e^{-\theta\tau} \tag{7.15}$$

where β and γ are scalars, and θ parametrises exponential decay in the intensity of both event rates.

The added computational complexity of each of the proposed models compared to the ones that were applied in this chapter, would perhaps have constrained their usefulness in that analysis. At least, without further efforts to refine the algorithm. However the main reason why these three models and other similar ideas were not pursued was that the effort was deemed to be without merit in the absence of a formal means for comparing them. This is the final subtopic of further work that we address here.

7.7.5 Tools for model assessment

Lastly, an important limitation of these methods and results stems from the lack of tools developed for formal model assessment and comparison. In particular, reliable (and computationally feasible) methods for computing the marginal likelihood of the model. While that is a nontrivial task in complicated DA-MCMC schemes of this kind, the nature of the study and the focus on competing models underscores the need for them regardless.

Failing that, alternative measures such as BIC or even a distance-based approximation would be easier to implement and at least give a quantitative indication of model fit. This should therefore probably be considered among the highest priorities for further development.

To conclude these remarks, the work conducted for this chapter was undertaken in addition to the main work stream of the project, which revolved around small population models of a different nature as focused on in Chapter 6. The number of avenues available for further development and investigation is partly the result of time constraints but also signifies at least some preliminary success. That is, the approach seems both scalable, and potentially useful for elucidating the as yet poorly defined nature of within-herd BTB dynamics and transmission.

Chapter 8

Conclusion

"Prediction is difficult, especially about the future."

- Professor Glenn Marion (and many others!)

Recap: keys aims and objectives

The key objectives of this work were laid out in the introductory chapter as follows:

1. To characterise within-herd BTB dynamics in UK cattle herds using the data set described in §6.2.3. In the first instance, to carry out Bayesian parameter inference based on conventional epidemiological models.

2. To extend (1) to the problem of multi-model inference, and investigate which models best fit the data, including formal evaluation using Bayesian methods.

3. Ultimately, to carry the findings of (2) forward to a large-scale system-of-herds model, up to and including national level (i.e. many thousands of herds).

4. In synergy with all of the above, and in concert with the aims and ethos of Biomathematics and Statistics Scotland (BioSS) where the vast majority of this work was carried out: develop [Bayesian] tools and methods that are of potential value to the wider scientific community.

The thesis concludes with a review of the extent to which these overarching objectives were met, ending with a short summary of opportunities for further research noted throughout and final comments.

8.1 Epidemiological parameter estimates for BTB in UK cattle herds

Novel estimates for epidemiological parameters were produced for a range of models, not withstanding the difficulties presented by the variation of test records, and the inconsistent reliability of herd size data in *VetNet*, which appears to have been used to simply record different (rather than inaccurate) information in many cases.

The estimates include the results presented in Chapter 6 for a modest number of herds – some one hundred were analysed but only a narrow selection (less than 20) were carried forward to the results presented in that chapter. That was due in part to computational difficulties that were frequently encountered (i.e. the algorithms failed to adequately converge). In hindsight those difficulties might have been alleviated by paying more careful attention to prior selection [11]. In particular, the selection of prior distributions that were more informative and realistic w.r.t. actual BTB scenarios recorded in the data. The reliance on weakly-informative priors was borne of a desire to take an approach to scientific study that was cautious and conservative. In hindsight this may have reflected a failure to 'think like a Bayesian'. This was an important lesson and key takeaway in terms of personal development.

The [SMC] methods applied did prove effective in *some* cases (and convenient to implement) but the above considerations and certain other constraints mitigated against extending that work to a larger number of herds. Namely the cost, in terms of both computation and the amount of manual curation of data required (as discussed in the conclusion of that chapter). That goal was advanced instead by the hierarchical 'system-of-herds' model developed in Chapter 7. The latter work was in a sense less complete than the results accomplished in the previous chapter, with respect to the consideration of different models and in particular formal methods for assessing them. However the BTB parameter estimates obtained went beyond the previous results by characterising extraneous [regional] background risk and trade-related risk, *in addition* to the within-herd disease process. This was possible due to the scale of the analysis (up to thousands of herds) which also added a degree of statistical strength compared to the results based on individual herd records presented in Chapter 6. Extending the scale of the analysis was nominated as an objective in its own right for this very reason (see §8.3).

8.2 Model inference

The second objective was also addressed in Chapter 6: the evaluation of different models for characterising within-herd BTB dynamics. This aspect of the work raised many interesting and nuanced considerations that were unforeseen, such as the importance of *reinfection* and the fluid definition of *latency* (similar but not quite the same as clinical latency, or without symptoms). By extension, the latter implies shifting definitions of concepts such as a 'false positive', when considering the 'true' underlying [discrete] disease status of an individual within a modelling paradigm. Ultimately though, no single model was singled out as being a uniquely good fit, and that is perhaps unsurprising given the limited scalability of the method. Simply put, the number of herds analysed was too low to reveal any useful findings, and the stringent selection criteria necessarily applied to the data very likely introduced a degree of systematic bias. Nevertheless a number of distinctive models and individual [herd] data sets were successfully evaluated using formal Bayesian methods. In summary, this goal was accomplished in a technical sense, even if the results were somewhat underwhelming in terms of scientific findings.

8.3 Large scale 'system-of-herds' model

The [hierarchical] model and methods used in Chapter 7 allowed for a more extensive analysis, in terms of the number herds but also the hitherto (in this project) intractable problems of interest, in particular the role of farmer behaviour (i.e. cattle trading) in BTB incidence. Parameter estimates were provided relating to that and other quantities of interest, ranging from localised ongoing forces-of-infection to those that can be loosely interpreted as the sensitivity of several diagnostic tests (notwithstanding the equally fluid definition of a 'true positive' within a discrete state-space modelling paradigm).

The geographically stratified parameters in particular provide a useful example of 'transformed data' as envisaged by the originators of the project: a way to indirectly compare localised BTB risk for different geographical regions. In theory that is a potentially more useful indicator than the raw data alone (e.g. case numbers). That is because the latter does not distinguish (and account for) within-herd dynamics relative to other extraneous factors.

Unfortunately, the model and method were conceived too late in the project to fully explore their potential. Only three models were given consideration, all of which use the same essential parametrisation of the Hawkes process, though others are available. Furthermore the methods learned and developed during the first phase of the project for computing the marginal likelihood (i.e. for model comparison) were not well suited to the [DA-MCMC] algorithmic approach. That rendered the model-inference aspect of the analysis beyond reach, if only for lack of time. That is somewhat disappointing given that the other aspects of the approach seemed to work well, indicating that the model may be a reasonably good fit for this data. It would have been interesting to further explore to extent to which that is true (i.e. compared to others models) using other Bayesian methods too, such as simulation study (i.e. predictive checks).

8.4 The development of Bayesian tools and methods

Finally, the fourth objective was addressed by the work presented in Chapters 3 through 5. The algorithms presented in Chapter 3 demonstrate how the principles of modular construction can be used to understand, codify and identify opportunities for the further development of algorithms. The 'modules' in this case were the essential methods and concepts introduced as background material in Chapter 2.

The same broad approach was taken with the Bayesian workflow implemented in Chapter 4, which situated the algorithms developed in the previous chapter, as working modules in a complete *Bayesian data analysis*. All of this work is implemented in a practical sense by the software package introduced in Chapter 5. The primary motivation for the latter was application to the analysis presented in Chapter 6. Publication of the software as an open-source package that is available for others to use was judged to be worthy of the time and effort that was spent doing it. However, as Easterbrook intones in the quotation cited at the top of that chapter, making *truly* useful open-source software requires effort on the part of authors in order to build a user community around it (and also developing for a wide variety of platforms). Whilst that kind of commitment is beyond the means of a lone student, the knowledge and experience gained during development and documentation of the package proved useful in the immediate aftermath of the COVID-19 pandemic (which occurred towards the end of the project). That was when an opportunity arose to contribute software (in the form of another Julia package [1]) to a project organised as part of the Scottish COVID-19 Response Consortium (SCRC) – which certainly would come close to fulfilling Easterbrook's stringent criteria.

Furthermore the decision to publish the methods developed as a software package (rather than merely sharing code) enhances the reproducability of the applied methods and results presented in this thesis. There is therefore at least a chance that the package could be of use to others investigating this particular problem who wish to do so. It was certainly deemed worthy of the effort in that regard.

Future research opportunities

The methods presented in Chapter 3, in particular the ARQ-MCMC algorithm, could be extended in several ways discussed in that chapter. That algorithm in particular proved to be a useful and somewhat generalised 'workhorse' for solving problems in cases where other algorithms struggled to converge. For example, particle filters with excessively high variance. Essentially, only one key aspect challenges it more general usefulness: the need to select appropriate (i.e. problem specific) sampling intervals.

As discussed in 3.6.2, that problem is notionally easy to solve: adaptively (i.e. automatically) adjusted sampling intervals. Intuitively, the ideal interval has much to do with the variance of the target distribution itself, such that it can be related 'adaptively' to the algorithm and samples as they are obtained. However this elementary strategy must be balanced against the overarching motivation and 'waste recycling' paradigm of the algorithm. Arbitrarily adapting sampling intervals would be wasteful. The obvious solution noted in that chapter are adaptive harmonic [sampling] intervals. In plain terms, choosing (or 'adapting') new intervals in a way that 'overlaps', so as to deliberately reuse old sampling coordinates on a systematic basis.

Making the algorithm more generally useful would also greatly enhance the case for further developing the software implementation presented in Chapter 5, alongside other features that have been noted but not yet implemented, such as *Bayesian model averaging* (see the conclusion of Chapter 4).

Finally the work presented in Chapter 7 on Hawkes processes stands some-

[1] 'Data pipeline API' [Julia] docs: https://fairdatapipeline.github.io/DataPipeline.jl/stable/

what alone and unfinished, at least compared to earlier chapters. For example the *Bayesian model averaging* technique could be quite easily implemented for the software and methods presented in the earlier, originally planned chapters. By contrast methods for estimating the marginal likelihood on which that and other model validation techniques depend have not yet been developed for the work presented in Chapter 7.

A key aim of that chapter was to assess the *scalability* of the approach. Given the promising results (easily scalable to thousands, rather than dozens, of herds on commodity hardware) potentially suitable methods for estimating the marginal likelihood (such as [93]) warrant further investigation for purposes of model validation in that chapter. That is because doing so would permit more robust (and therefore more interesting) investigations of within-herd BTB dynamics directly, since competing models could be formally compared (or else estimates obtained from a range of models, in the case of Bayesian model averaging).

Summary and final comments

The model and parameter estimates for BTB presented in this thesis represent novel findings in terms of the methods applied to obtain them, even if interesting questions such as the role of reinfection and long-term latent infection in within-herd dynamics remain unresolved. Advancing the overarching model and methods developed in Chapter 7 (in particular towards formal model assessment) is one way in which those matters could be immediately further investigated.

The project did lead to a more developed understanding of Bayesian data analysis and its practical applications, including emergent (or at least, recently renewed) concepts such as modular construction and Bayesian workflows. This informed the development of new algorithms and software tools that have been published and made available to others. Unfortunately, and despite the order of presentation, Chapters 4 and 5 were essentially the last to be completed. This meant that the impact of that learning (and associated outputs) on 'subsequent' chapters was mostly retrospective. This accounts for certain continuity gaps across those chapters. It also reflects the iterative and cyclical flow of execution in the project.

The last is a characteristic of Bayesian data analysis generally that was noted in the main text, but also driven in this case by a constantly renewed effort to focus on the motivating problem (and further develop the means to do so). That is, the as yet still poorly defined (but vitally important) nature of within-herd BTB dynamics in cattle herds. It is sometimes said that PhD projects are never truly finished, only abandoned. The final conclusion proffered is that truer words have never been spoken.

Bibliography

[1] Robert E Kass and Adrian E Raftery. Bayes factors. *Journal of the American Statistical Association*, 90(430):773–795, 1995. Publisher: Taylor & Francis.

[2] Andrew J. K. Conlan, Trevelyan J. McKinley, Katerina Karolemeas, Ellen Brooks Pollock, Anthony V. Goodchild, Andrew P. Mitchell, Colin P. D. Birch, Richard S. Clifton-Hadley, and James L. N. Wood. Estimating the Hidden Burden of Bovine Tuberculosis in Great Britain. *PLoS Computational Biology*, 8(10):e1002730, October 2012.

[3] William O. Kermack and Anderson G. McKendrick. Contributions to the mathematical theory of epidemics—I. *Bulletin of mathematical biology*, 53(1-2):33–55, 1991.

[4] C. M. Pooley, S. C. Bishop, and G. Marion. Using model-based proposals for fast parameter inference on discrete state space, continuous-time Markov processes. *Journal of The Royal Society Interface*, 12(107):20150225–20150225, May 2015.

[5] Léo Grinsztajn, Elizaveta Semenova, Charles C. Margossian, and Julien Riou. Bayesian workflow for disease transmission modeling in Stan. *arXiv:2006.02985 [q-bio, stat]*, February 2021. arXiv: 2006.02985.

[6] Hyun M Yang and Silvia M Raimundo. Assessing the effects of multiple infections and long latency in the dynamics of tuberculosis. *Theoretical Biology and Medical Modelling*, 7(1):41, December 2010.

[7] Hadley Wickham. *ggplot2: Elegant Graphics for Data Analysis.* Springer-Verlag New York, 2016.

[8] Philip D. O'Neill and G. O. Roberts. *Bayesian inference for partially observed stochastic epidemics.* JSTOR, 1997.

[9] N. Chopin, P. E. Jacob, and O. Papaspiliopoulos. SMC ^{2} : an efficient algorithm for sequential analysis of state space models: Sequential Analysis of State Space Models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 75(3):397–426, June 2013.

[10] Arnaud Doucet and Adam M Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of nonlinear filtering*, 12(656-704):3, 2009.

[11] Andrew Gelman, Aki Vehtari, Daniel Simpson, Charles C. Margossian, Bob Carpenter, Yuling Yao, Lauren Kennedy, Jonah Gabry, Paul-Christian Bürkner, and Martin Modrák. Bayesian Workflow. *arXiv:2011.01808 [stat]*, November 2020. arXiv: 2011.01808.

[12] Art B Owen. Quasi-monte carlo sampling. *Monte Carlo Ray Tracing: Siggraph*, 1:69–88, 2003.

[13] JF Bourne. Bovine TB: The Scientific Evidence. A science base for a sustainable policy to control TB in cattle. Technical report, ISG, DEFRA, London, 2007.

[14] Charles O. Thoen, James H. Steele, and Michael J. Gilsdorf, editors. *Mycobacterium bovis infection in animals and humans.* Blackwell Pub, Ames, Iowa, 2nd ed edition, 2006. OCLC: ocm60767070.

[15] Ellen Brooks-Pollock, Gareth O. Roberts, and Matt J. Keeling. A dynamic model of bovine tuberculosis spread and control in Great Britain. *Nature*, 511(7508):228–231, July 2014.

[16] M. V. Thrusfield. *Veterinary epidemiology.* Butterworths, London ; Boston, 1986.

[17] A. O'Hare, R. J. Orton, P. R. Bessell, and R. R. Kao. Estimating epidemiological parameters for bovine tuberculosis in British cattle using a Bayesian partial-likelihood approach. *Proceedings of the Royal Society B: Biological Sciences*, 281(1783):20140248–20140248, April 2014.

[18] T. W. A. Little, C. Swan, H. V. Thompson, and J. W. Wilesmith. Bovine tuberculosis in domestic and wild mammals in an area of Dorset. II. The badger population, its ecology and tuberculosis status. *Journal of Hygiene*, 89(02):211–224, 1982.

[19] Andrew Gelman, John B Carlin, Hal Steven Stern, David B Dunson, Aki Vehtari, and Donald B Rubin. *Bayesian data analysis*. 2014. OCLC: 909477393.

[20] Sharon Bertsch McGrayne. *The theory that would not die*. Yale University Press, 2011.

[21] Chang-Jin Kim, Charles R Nelson, and others. State-space models with regime switching: classical and Gibbs-sampling approaches with applications. *MIT Press Books*, 1, 1999.

[22] R. J. Boys, D. J. Wilkinson, and T. B. L. Kirkwood. Bayesian inference for a discretely observed stochastic kinetic model. *Statistics and Computing*, 18(2):125–135, June 2008.

[23] Ken B. Newman, Stephen T. Buckland, and Byron J. T. Morgan, editors. *Modelling population dynamics: model formulation, fitting and assessment using state-space methods*. Methods in statistical ecology. Springer, New York, NY, 2014. OCLC: 891029166.

[24] Marian-Andrei Rizoiu, Swapnil Mishra, Quyu Kong, Mark Carman, and Lexing Xie. SIR-Hawkes: Linking Epidemic Models and Hawkes Processes to Model Diffusions in Finite Populations. *Proceedings of the 2018 World Wide Web Conference on World Wide Web - WWW '18*, pages 419–428, 2018. arXiv: 1711.01679.

[25] Penelope L Peterson. *International encyclopedia of education*. 2010. OCLC: 1040153950.

[26] Shoba Ranganathan, Kenta Nakai, and Christian Schönbach. *Encyclopedia of Bioinformatics and Computational Biology*. 2019. OCLC: 1061555297.

[27] Alan G Hawkes. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1):83–90, 1971.

[28] Marian-Andrei Rizoiu, Young Lee, Swapnil Mishra, and Lexing Xie. A Tutorial on Hawkes Processes for Events in Social Media. *arXiv:1708.06401 [cs, stat]*, August 2017. arXiv: 1708.06401.

[29] Angelos Dassios and Hongbiao Zhao. A dynamic contagion process. *Advances in Applied Probability*, 43(3):814–846, September 2011.

[30] Daniel T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, 81(25):2340–2361, December 1977.

[31] Jesper Møller and Jakob G Rasmussen. Perfect simulation of Hawkes processes. *Advances in applied probability*, 37(3):629–646, 2005.

[32] Angelos Dassios and Hongbiao Zhao. Exact simulation of Hawkes process with exponentially decaying intensity. *Electronic Communications in Probability*, 18(0), 2013.

[33] Y. Ogata. On Lewis' simulation method for point processes. *IEEE Transactions on Information Theory*, 27(1):23–31, January 1981.

[34] Christophe Andrieu, Arnaud Doucet, and Roman Holenstein. Particle markov chain monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342, 2010.

[35] Jonathan Fintzi, Xiang Cui, Jon Wakefield, and Vladimir N. Minin. Efficient Data Augmentation for Fitting Stochastic Epidemic Models to Prevalence Data. *Journal of Computational and Graphical Statistics*, 26(4):918–929, October 2017.

[36] Gavin J. Gibson and Eric Renshaw. Estimating parameters in stochastic compartmental models using Markov chain methods. *Mathematical Medicine and Biology*, 15(1):19–40, 1998.

[37] Gareth O Roberts and Osnat Stramer. Langevin diffusions and Metropolis-Hastings algorithms. *Methodology and computing in applied probability*, 4(4):337–357, 2002.

[38] Ray Watson. An application of a martingale central limit theorem to the standard epidemic model. *Stochastic Processes and their Applications*, 11(1):79–89, March 1981.

[39] David Lindenstrand and Åke Svensson. Estimation of the Malthusian parameter in an stochastic epidemic model using martingale methods. *Mathematical Biosciences*, 246(2):272–279, December 2013.

[40] W. K. Hastings. Monte Carlo Sampling Methods Using Markov Chains and Their Applications. *Biometrika*, 57(1):97, April 1970.

[41] Randal Douc, Olivier Cappé, and Eric Moulines. Comparison of Resampling Schemes for Particle Filtering. *arXiv:cs/0507025*, July 2005. arXiv: cs/0507025.

[42] Mathieu Gerber, Nicolas Chopin, and Nick Whiteley. Negative association, ordering and convergence of resampling methods. *arXiv:1707.01845 [stat]*, July 2017. arXiv: 1707.01845.

[43] Genshiro Kitagawa. Monte Carlo Filter and Smoother for Non-Gaussian Nonlinear State Space Models. *Journal of Computational and Graphical Statistics*, 5(1):1–25, March 1996.

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

[45] Ronald L. Iman, Jon C. Helton, and James E. Campbell. An Approach to Sensitivity Analysis of Computer Models: Part I—Introduction, Input Variable Selection and Preliminary Variable Assessment. *Journal of Quality Technology*, 13(3):174–183, July 1981.

[46] Il'ya Meerovich Sobol'. On the distribution of points in a cube and the approximate evaluation of integrals. *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki*, 7(4):784–802, 1967.

[47] J. H. Halton. Algorithm 247: Radical-inverse quasi-random point sequence. *Communications of the ACM*, 7(12):701–702, December 1964.

[48] Robert L. Cook. Stochastic sampling in computer graphics. *ACM Transactions on Graphics*, 5(1):51–72, January 1986.

[49] Frances Y Kuo and Ian H Sloan. Lifting the Curse of Dimensionality. 52(11):9, 2005.

[50] Josef Dick, Frances Y. Kuo, and Ian H. Sloan. High-dimensional integration: The quasi-Monte Carlo way. *Acta Numerica*, 22:133–288, May 2013.

[51] A. B. Owen and S. D. Tribble. A quasi-Monte Carlo Metropolis algorithm. *Proceedings of the National Academy of Sciences*, 102(25):8844–8849, June 2005.

[52] JG Liao. Variance reduction in Gibbs sampler using quasi random numbers. *Journal of Computational and Graphical Statistics*, 7(3):253–266, 1998.

[53] Seth D Tribble. *MARKOV CHAIN MONTE CARLO ALGORITHMS USING COMPLETELY UNIFORMLY DISTRIBUTED DRIVING SEQUENCES*. PhD, Stanford University, June 2007.

[54] Tobias Schwedes and Ben Calderhead. Quasi Markov Chain Monte Carlo Methods. *arXiv:1807.00070 [math, stat]*, June 2018. arXiv: 1807.00070.

[55] D. Frenkel. Speed-up of Monte Carlo simulations by sampling of rejected states. *Proceedings of the National Academy of Sciences*, 101(51):17571–17575, December 2004.

[56] Axel Finke, Ruth King, Alexandros Beskos, and Petros Dellaportas. Efficient Sequential Monte Carlo Algorithms for Integrated Population Models. *Journal of Agricultural, Biological and Environmental Statistics*, January 2019.

[57] D. Ceperley, G. V. Chester, and M. H. Kalos. Monte Carlo simulation of a many-fermion study. *Phys. Rev. B*, 16(7):3081–3099, October 1977.

[58] H Tjelmeland. Using all Metropolis–Hastings proposals to estimate mean values. Technical Report NTNU-S-2004-4, Trondheim TU. Inst. Math., Trondheim, 2004.

[59] D. Frenkel. Waste-Recycling Monte Carlo. In Mauro Ferrario, Giovanni Ciccotti, and Kurt Binder, editors, *Computer Simulations in Condensed Matter Systems: From Materials to Chemical Biology Volume 1*, volume 703, pages 127–137. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[60] Jean-Françcois Delmas and Benjamin Jourdain. Does Waste Recycling Really Improve the Multi-Proposal Metropolis–Hastings algorithm? an Analysis

Based on Control Variates. *Journal of Applied Probability*, 46(04):938–959, December 2009.

[61] John Geweke. Evaluating the Accuracy of Sampling-Based Approaches to the Calculation of Posterior Moments. In *IN BAYESIAN STATISTICS*, pages 169–193. University Press, 1992.

[62] Mary Kathryn Cowles and Bradley P. Carlin. Markov Chain Monte Carlo Convergence Diagnostics: A Comparative Review. *Journal of the American Statistical Association*, 91(434):883, June 1996.

[63] Andrew Gelman and Donald B. Rubin. Inference from iterative simulation using multiple sequences. *Statistical science*, pages 457–472, 1992.

[64] C. M. Pooley and G. Marion. Bayesian model evidence as a practical alternative to deviance information criterion. *Royal Society Open Science*, 5(3):171519, March 2018.

[65] Christian P. Robert, Nicolas Chopin, and Judith Rousseau. Harold Jeffreys's Theory of Probability Revisited. *Statistical Science*, 24(2):141–172, May 2009. arXiv: 0804.3173.

[66] Siddhartha Chib. Marginal likelihood from the Gibbs output. *Journal of the American Statistical Association*, 90(432):1313–1321, 1995.

[67] Michael Betancourt. Towards A Principled Bayesian Workflow, April 2020.

[68] Jonah Gabry, Daniel Simpson, Aki Vehtari, Michael Betancourt, and Andrew Gelman. Visualization in Bayesian workflow. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 182(2):389–402, 2019. Publisher: Wiley Online Library.

[69] Jennifer A Hoeting, David Madigan, Adrian E Raftery, and Chris T Volinsky. Bayesian model averaging: a tutorial (with comments by M. Clyde, David Draper and EI George, and a rejoinder by the authors. *Statistical science*, 14(4):382–417, 1999. Publisher: Institute of Mathematical Statistics.

[70] Steve M. Easterbrook. Open code for open science? *Nature Geoscience*, 7(11):779–781, November 2014.

[71] Bob Carpenter, Andrew Gelman, Matthew D. Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. *Stan* : A Probabilistic Programming Language. *Journal of Statistical Software*, 76(1), 2017.

[72] Martyn Plummer. JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling. *Working Papers*, page 8, 2003.

[73] David J. Lunn, Andrew Thomas, Nicky Best, and David Spiegelhalter. WinBUGS; A Bayesian Modelling Framework: Concepts, Structure, and Extensibility. *Statistics and Computing*, 10(4):325–337, October 2000.

[74] Radford M. Neal. MCMC using Hamiltonian dynamics. *arXiv:1206.1901 [physics, stat]*, June 2012. arXiv: 1206.1901.

[75] Iñaki Ucar, Bart Smeets, and Arturo Azcorra. **simmer** : Discrete-Event Simulation for *R*. *Journal of Statistical Software*, 90(2), 2019.

[76] Aaron A. King, Dao Nguyen, and Edward L. Ionides. Statistical Inference for Partially Observed Markov Processes via the *R* Package **pomp**. *Journal of Statistical Software*, 69(12), 2016.

[77] Ronald Ross. Some Quantitative Studies in Epidemiology. *Nature*, 87(2188):466–467, October 1911.

[78] Robin A Skuce, Adrian R Allen, and Stanley W J McDowell. Bovine Tuberculosis (TB): A Review Of Cattle-To-Cattle Transmission, Risk Factors And Susceptibility. Technical report, http://www.dardni.gov.uk, 2011.

[79] Roman Biek, Anthony O'Hare, David Wright, Tom Mallon, Carl McCormick, Richard J. Orton, Stanley McDowell, Hannah Trewby, Robin A. Skuce, and Rowland R. Kao. Whole Genome Sequencing Reveals Local Transmission Patterns of Mycobacterium bovis in Sympatric Cattle and Badger Populations. *PLoS Pathogens*, 8(11):e1003008, November 2012.

[80] Rowland Kao, M. G. Roberts, and T. J. Ryan. A Model of Bovine Tuberculosis Control in Domesticated Cattle Herds. *Proceedings: Biological Sciences*, 264(1384):1069–1076, July 1997.

[81] F.D. Menzies and S.D. Neill. Cattle-to-Cattle Transmission of Bovine Tuberculosis. *The Veterinary Journal*, 160(2):92–106, September 2000.

[82] N. D. Barlow, J. M. Kean, G. Hickling, P. G. Livingstone, and A. B. Robson. A simulation model for the spread of bovine tuberculosis within New Zealand cattle herds. *Preventive veterinary medicine*, 32(1-2):57–75, 1997.

[83] A. Melegaro, N. J. Gay, and G. F. Medley. Estimating the transmission parameters of pneumococcal carriage in households. *Epidemiology and Infection*, 132(3):433–441, June 2004.

[84] M. J. Smith, S. Telfer, E. R. Kallio, S. Burthe, A. R. Cook, X. Lambin, and M. Begon. Host-pathogen time series data in wildlife support a transmission function between density and frequency dependence. *Proceedings of the National Academy of Sciences*, 106(19):7905–7909, May 2009.

[85] M. J. Betancourt. A Bayesian Approach To Histogram Comparison. *arXiv:1009.5604 [physics]*, September 2010. arXiv: 1009.5604.

[86] R. de la Rua-Domenech, A.T. Goodchild, H.M. Vordermeier, R.G. Hewinson, K.H. Christiansen, and R.S. Clifton-Hadley. Ante mortem diagnosis of tuberculosis in cattle: A review of the tuberculin tests, gamma-interferon assay and other ancillary diagnostic techniques. *Research in Veterinary Science*, 81(2):190–210, October 2006.

[87] S.H. Downs, J.M. Broughan, A.V. Goodchild, P.A. Upton, and P.A. Durr. Responses to diagnostic tests for bovine tuberculosis in dairy and non-dairy cattle naturally exposed to Mycobacterium bovis in Great Britain. *The Veterinary Journal*, 216:8–17, October 2016.

[88] Roderick JA Little. Regression with missing X's: a review. *Journal of the American Statistical Association*, 87(420):1227–1237, 1992. Publisher: Taylor & Francis Group.

[89] Fredrik Lindsten, Adam M. Johansen, Christian A. Naesseth, Bonnie Kirkpatrick, Thomas B. Schön, John Aston, and Alexandre Bouchard-Côté. Divide-and-Conquer with Sequential Monte Carlo. *Journal of Computational and Graphical Statistics*, 26(2):445–458, April 2017. arXiv: 1406.4993.

[90] Tingnan Gong, Yu Chen, and Weiping Zhang. An Environmentally-Adaptive Hawkes Process with An Application to COVID-19. *arXiv:2101.09942 [stat]*, January 2021. arXiv: 2101.09942.

[91] Michele Garetto, Emilio Leonardi, and Giovanni Luca Torrisi. A time-modulated Hawkes process to model the spread of COVID-19 and the impact of countermeasures. *Annual Reviews in Control*, page S1367578821000080, March 2021.

[92] Raiha Browning, Deborah Sulem, Kerrie Mengersen, Vincent Rivoirard, and Judith Rousseau. Simple discrete-time self-exciting models can describe complex dynamic processes: A case study of COVID-19. *PLOS ONE*, 16(4):e0250015, April 2021.

[93] Panayiota Touloupou, Naif Alzahrani, Peter Neal, Simon E. F. Spencer, and Trevelyan J. McKinley. Efficient Model Comparison Techniques for Models Requiring Large Scale Data Augmentation. *Bayesian Analysis*, 13(2), June 2018.

[94] Seth D. Tribble and Art B. Owen. Construction of weakly CUD sequences for MCMC sampling. *Electronic Journal of Statistics*, 2(0):634–660, 2008. arXiv: 0807.4858.

[95] Matt J. Keeling. Using individual-based simulations to test the Levins metapopulation paradigm. *Journal of Animal Ecology*, 71(2):270–279, March 2002.

[96] R. Levins. Some Demographic and Genetic Consequences of Environmental Heterogeneity for Biological Control. *Bulletin of the Entomological Society of America*, 15(3):237–240, September 1969.

[97] Alan A. Berryman. The Orgins and Evolution of Predator-Prey Theory. *Ecology*, 73(5):1530–1535, October 1992.

[98] G. Macdonald. The analysis of equilibrium in malaria. *Tropical Diseases Bulletin*, 49(9):813–829, September 1952.

[99] Ivo M. Foppa. *A Historical Introduction to Mathematical Modeling of Infectious Diseases: Seminal Papers in Epidemiology*. Academic Press, 2016.

[100] Michael Y. Li. Five Classic Epidemic Models and Their Analysis. In *An Introduction to Mathematical Modeling of Infectious Diseases*, pages 35–77. Springer International Publishing, Cham, 2018.

Appendix A

ARQ-MCMC supplementary material

Here we provide additional commentary on the ARQ-MCMC algorithm introduced in Chapter 3, specifically concerning estimation of the marginal likelihood.

Estimating the marginal likelihood

For the ARQ MCMC algorithm (Algorithm 10) we could (notionally) obtain a good approximation of the marginal likelihood by computing:

$$\pi(y) \approx L = \frac{\sum_{\theta \in \Theta^i} \hat{\pi}(\theta|y)}{{N_R}^d} \tag{A.1}$$

where $\hat{\pi}(\theta|y)$ is computed according to parameter function ϕ and the sum is over all ${N_R}^d$ elements of Θ^i. In the theory of quasi Monte Carlo [94, 51] if the sequence construction used to generate Θ^i is strongly completely uniformly distributed (CUD) the above should provide an unbiased estimate in the limit as $N_C, \Gamma_L \rightarrow \infty$, computed as it is from another unbiased estimator $\phi(\theta)$. However, algorithm 10 uses an ordinary (i.e. PRNG-based) Metropolis-Hastings procedure to sample and resample a primitive QMC construction, albeit one that is at least weakly CUD.

This is somewhat irrelevant as we would not expect to sample every tuple in Θ^i. However on the basis that the importance sample Γ is sufficiently enriched to have incorporated the elements of Θ_i that correspond to the densest regions of posterior mass, we can (crudely) approximate (2.40) by:

$$\hat{L} = \frac{\sum_{\theta \in \Gamma} \hat{\pi}(\theta|y) + \sum_{\theta \notin \Gamma} E(\hat{\pi}(\theta|y))}{{N_R}^d} \tag{A.2}$$

where $E(...)$ the expectation for the marginal likelihood of the *unsampled* elements of Θ^i, denoted by $\theta \notin \Gamma$. This can be estimated quite easily, by sampling uniformly from $\theta \notin \Gamma$ and taking the average of $\phi(\theta)$. However we have found that in practice, simply computing:

$$L \approx \frac{\sum_{\theta \in \Gamma} \hat{\pi}(\theta|y)}{{N_R}^d} \tag{A.3}$$

provides a reasonable though slightly conservative approximation of 2.41 to suffice for model selection purposes, since the additionally sampled tuples tend to contribute relatively little to the marginal likelihood. Of course this relies on the assumption that we have good (rather than 'patchy') coverage of the posterior mass. In cases of anything other than moderately low d it would be more appropriate to compute an interpolated value. However the computational benefits of *ARQ MCMC* recede exponentially as d is increased, converging on the performance of standard particle MCMC, so it is not well suited to such problems in any case.

This leads to the following definition:

Appendix B

BayesianWorkflows.jl supplementary material

This appendix provides supplementary information for Chapter 5 (which introduces the *BayesianWorkflows.jl* package for Julia). It begins with basic instructions for using the package. That is followed by a series of examples and case studies based on real and simulated data, accompanied by sample code and results.

Note: the package was developed throughout the course of the project and has undergone many changes and iterations. The content in this section is included mainly to give a flavour of the work done for Chapter 5 rather than as useful or current instructions for using the package. The most up to date version of those will always be the online docs [1].

B.1 Installation and usage

As a prerequisite, the package naturally requires a working installation of the Julia programming language. Instructions for installing the package itself can be found on the home page the source code repository[2].

[1]Package documentation: https://mjb3.github.io/BayesianWorkflows.jl/stable/
[2]BayesianWorkflows.jl repo: https://github.com/mjb3/DiscretePOMP.jl

Getting started

Once the package has been installed, the following code snippet can be used to reproduce the SIS example cited in this chapter.

Download the Pooley dataset from the source code repository[3] and run:

```
using DPOMPs
y = get_observations_from_file("path/to/data/pooley.csv")
model = generate_model("SIS", [100,1])
mcmc = run_met_hastings_mcmc(model, y, [0.0025, 0.12])
```

Listing B.1: Getting started – run a basic *MBP-MCMC* analysis.

Along with the code provided in B, the code used to produce the other examples in this chapter are available in the package documentation along with animated examples and other useful reference material.

B.2 A simple example – SIS model

Overview of this section

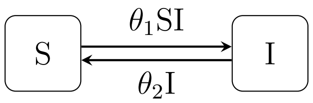
Here we demonstrate the main features of the package. In particular, we address:

- How to define a model, in §B.2.2 (including customised and predefined models).
- Model simulation using Algorithm 1, in §B.2.3.
- The main body of the section is given over to describing how to use functions for Bayesian parameter inference, with single-model inference covered in §B.2.4. That includes convergence diagnostics, as described in Chapter 2, along with and visualisation and analysis tools.
- Model comparison (i.e. multi-model inference) briefly covered in §B.2.7 with an applied example given in §B.3.3.

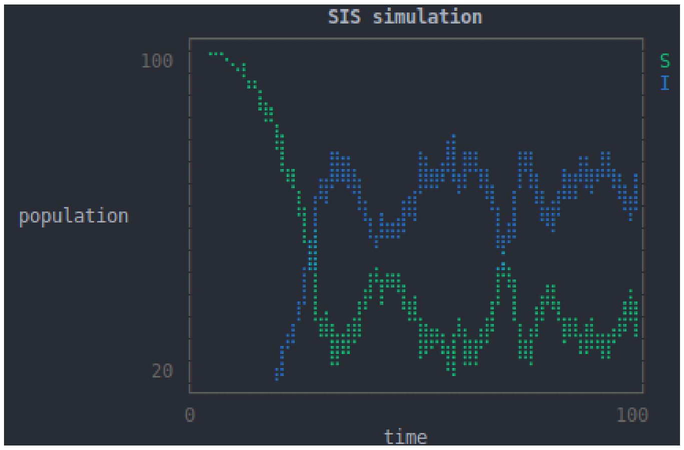
As such, this section provides instructions for using most of the package's core functions, and sample code for an exemplar. That problem is borrowed from the

[3]Pooley dataset: https://raw.githubusercontent.com/mjb3/BayesianWorkflows.jl/master/data/pooley.csv

paper by Pooley *et al.*,that introduced the MBP method [4], and is based on the susceptible-infectious-susceptible (SIS) model, as illustrated in Figure B.1. We also note that the model is a stochastic analogue of Levin's patch occupancy model in a meta-population [95, 96].



(a) SIS model



(b) Inline simulation plot.

Figure B.1: In contrast to the SIR model, the SIS model is intended to represent the dynamics of infectious diseases which do not confer long-lasting immunity (or patch occupancy in the aforementioned meta-population model of Levin). The example on the RHS was simulated (and visualised) using the package (Algorithm 1 – see §B.2.3.) The parameters used by [4] of $\theta = \{\beta := 0.003, \gamma := 0.1\}$ were also used here, and the results are approximately the same.

For the most part, the code samples given throughout this section are in sequence, such that they are not self-contained. Rather, the variables defined – in particular the model and simulated observations data – are typically required to run onward code samples.

More complete examples that are self-contained are provided in various appendices; with the online package documentation[4]; and also within the source code repository itself, in the 'examples' directory[5].

[4]Package documentation: https://mjb3.github.io/DiscretePOMP.jl/stable/

[5]Source code: https://github.com/mjb3/DiscretePOMP.jl

B.2.1 Installation

Instructions for downloading and installing the package can be found on the home page of the package's GitHub repository[6].

B.2.2 Defining models

In order for the main features of the package to be useful, a model must first be defined and instantiated. The package framework allows users to define (or customise) models according to their own requirements.

Also provided are several predefined models based on well known examples from epidemiology, and population-ecology more generally. They include the standard susceptible-infectious-recovered (SIR) model [3] depicted in Figure 2.2, and other well-known variants like the SEIR. A complete list is provided with the package documentation.

Analyses based on predefined models require the least amount of code, and are thus the fastest way to begin.

Example: a predefined SIS model

Creating a predefined model instance is straightforward; besides the model name (a String) the only parameter input required from the user is the initial population state variable, referred to in more general terms as the system's *initial condition*.

It is passed to the function as an Array of integers. For example:

```
using DPOMPs      # NB. install the package first
# generate model:
initial_condition = [100, 1]
model = generate_model("SIS", initial_condition)
```

Listing B.2: Generate a predefined model instance.

where both the model name and initial condition variables are mandatory arguments that are passed to the function used to create generate the model automatically.

[6]DiscretePOMP.jl repo: https://github.com/mjb3/DiscretePOMP.jl

Defining custom models

Models can also be specified manually. Among other things, this allows for an arbitrary number of user-defined event types. That aspect of the model can be determined by specifying a custom rate function and transition matrix.

Note that the instructions that follow are somewhat redundant, because the SIS model has already been predefined. However they are still useful for exposition.

We begin with the event rate function, which reflects the complete set of possible events. Thus for the SIS model we have:

$$r_1 = \theta_1 S I \tag{B.1}$$

$$r_2 = \theta_2 I \tag{B.2}$$

where the infection and recovery rates are labelled r_1 and r_2, respectively. Note that this will correspond to their positions in the output [Array] variable.

As an aside, note that the definition is identical to the SIR model, as per the rates denoted in the illustration in Figure 5.1a. It is actually the matrices that describe the *transition* between states that distinguishes this aspect of the model. For the SIS model it is written as:

$$T = \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix} \tag{B.3}$$

For illustrative comparison, the same is given for the SIR model by:

$$T = \begin{bmatrix} -1 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix} \tag{B.4}$$

Note that the row vectors are associated with each distinct type of event, and column vectors relate to the model state-space, e.g. S-I-R, from left-to-right in (B.4).

For the purpose of using the package, these are expressed in *Julia* as two variables; a function and the aforementioned [Array] variable, i.e. (B.3), as follows for the SIS model:

```
# rate function:
function sis_rf!(output, parameters::Array{Float64, 1}, ↩
    population::Array{Int64, 1})
     output[1] = parameters[1] * population[1] * population[2]
```

```
        output[2] = parameters[2] * population[2]
    end
    # transition matrix:
    tm = [-1 1; 1 -1]
```

Listing B.3: Rate function and transition matrix for the SIS model.

Note that the function's output variable is predefined[7]. In other words the user's only responsibility when defining the custom event rate function, is to populate it with the correct values.

The variables defined as above (including the function) can then either be used to instantiate a `DPOMPModel` directly, else supplied to the `generate_custom_model` helper function as input parameters when instantiating the model that way.

The helper function has a number of other optional input parameters, such as the as prior distribution, and the observation [log-likelihood] model. It is invoked like so:

```
# NB. requires the 'Distributions' package for prior
using Distributions
prior = Product(Uniform.(0, [0.01, 0.5])) # optional
model = generate_custom_model("SIS_custom", sis_rf!, ↩
    initial_condition, tm; prior = prior)
```

Listing B.4: Generating a custom model.

Here we have defined the prior distribution as a [multivariate] Uniform distribution using vector notation, simply for illustration. The lower bound for each component distribution is zero. The upper bound is 0.01 and 0.5 for θ_1 and θ_2 respectively.

A similar example is provided in Appendix B.5.1 for a different model. A more complete code sample for the SIS model definition, that incorporates a more expansive model definition, including both the prior distribution and an observation model, is provided in B.5.2.

[7]By *Julia* convention, the function name has an exclamation mark appended in use cases where the function operates on variables that are passed (first by convention) to it – as in 'function_name!'

Partial customisation

It is not necessary to fully (i.e. manually) specify a model or use the custom helper function in order to customise it. As described in the package manual, the `DPOMPModel` type is mutable.

That means that its properties can be replaced by user-defined values, including predefined models instantiated using the `generate_model` function. The package also provides predefined functions for use as, e.g. observation models, and other such mutable components that can be recombined.

As such, simulation and inference can be performed even on customised models using minimal amounts of code. Refer to the online package documentation for a complete list of predefined models and model components, and the default options for each.

Characterisation of the observations process

Bayesian methods can also be usefully applied to retrospectively characterise the the observation process itself, when it involves some degree of uncertainty. For example, the sensitivity and specificity of diagnostic assays, or the efficiency of capture campaigns.

This flexibility can be especially useful in situations where observation and data collection are costly and subject to inherent uncertainties. In *DiscretePOMP.jl*, the *observation likelihood model* is represented by a function variable, similar to the event rate function.

B.2.3 Model simulation

The package implements Algorithm 1 for simulating from models. Invoking the corresponding function is straightforward, requiring (at a minimum) only a model instance (i.e. the one that we just defined) and vector of model parameters θ.

```
# NB. first define the SIS 'model' variable, per above
theta = [0.003, 0.1]
x = gillespie_sim(model, theta)  # run simulation
p = plot_trajectory(x)           # plot (optional)
```

Listing B.5: Simulate a model realisation.

The [complete] system trajectory, denoted herein simply as x, is represented by the `x.trajectory` property of the output variable.

The other product of the simulation `x.observations` property is an array of Observation types which represent the set of simulated [partial] observations data, denoted as y, produced automatically and based on the observation function of the model. This provides a convenient way to assess the package's functionality for performing inference from such data, which we now demonstrate.

The final line of code in the sample above produces (but doesn't display) an inline plot of the realised (i.e. 'simulated') trajectory, as illustrated in Figure B.1b. Visualisation functionality generally is discussed further in §B.2.5.

B.2.4 Inference from data

This section is reserved primarily for demonstrating the package's functionality for single-model parameter inference. Methods for *multi-model* inference, including model comparison and selection, are essentially incorporated within a single function. This is briefly explained in §B.2.7 but the functionality itself is demonstrated using a dedicated example of model selection in practice, in §B.3.3.

In general, all these methods entail the use of [partial] data to learn about the posterior distribution of the model parameters θ. Note that this does not preclude the possibility of also learning about the posterior distribution of trajectory variable. However that is not directly addressed here.

There are three main [single-model] inference functions defined within the package, that together incorporate each of the specific methods described in Chapters 2 and 4. They are,

- `run_mcmc_analysis` – for performing data-augmented MCMC analyses (Algorithm 4).

- `run_ibis_analysis` – for iterative-batch-importance-sampling analyses (Algorithms 8 and 9).

- `run_arq_mcmc_analysis` – an implementation of the algorithm introduced in §3.3 as *ARQ-MCMC*, that in this case depends on a particle-filter (i.e. Algorithms 10 and 6 respectively).

One of the main distinguishing features of these functions is the way the results are formatted. MCMC is a form of *rejection sampling*; those results consist of a set of MCMC samples, and auxiliary information, such as the results of the Gelman-Rubin convergence diagnostic (as discussed in §2.4.6).

More specifically, they include estimates of the *scale-reduction-factors* for that test (labelled SRE within the package). By contrast, *importance sampling* results consist of a set of weighted samples, as one might expect. They lack a corresponding convergence diagnostic, but do include an estimate of the marginal likelihood $\pi(y)$, also referred to as the [Bayesian] model evidence.

The *ARQ-MCMC* is essentially an importance sampling algorithm, but the MCMC resamples used to weight the former are also included with the results.

As such the results of a call to this function includes all of the above, including further auxiliary information common to all, such as the algorithm runtime, and estimates for $\mathrm{E}(\theta)$ and corresponding variances.

Code samples for the summary and analysis of each type of results (or format) are given throughout this section. In general, functions that take them are overloaded for all results types. For example, the `tabulate_results` function is compatible with all formats. As such that the distinctions noted above are at least somewhat removed from the user's concern in practice.

We now describe the main *single-model inference* functionality in more detail, including code samples for the SIS model and corresponding output from the package, before moving on to the topic of *multi-model inference* in §B.2.7.

Running data-augmented MCMC analyses

Data-augmented methods are so-called in reference to latent variables – in this case, a single one that represents the complete system trajectory – used to 'augment' the expression that defines posterior density.

The general notion is then to sample the joint density of the model parameters including the latent variable. This is written throughout as $\pi(\theta, x|y)$, where x is the latent variable.

The primary method implemented for data-augmented MCMC (DA-MCMC) is the Metropolis-Hastings algorithm – i.e the one given in §2.4.2 as Algorithm 4.

The default *proposal* algorithm is Algorithm 5, or *MBP*, with standard (i.e. naive) MCMC moves available as the only other built-in alternative (customised proposals are also possible, as demonstrated in §B.4).

Thus the full [default] algorithm is *MBP-MCMC*, as first described in [4]. It is invoked by using the `run_mcmc_analysis` function.

The following code sample is sufficient to run a straightforward MCMC analysis – following on from the code samples given above for model definition and simulation:

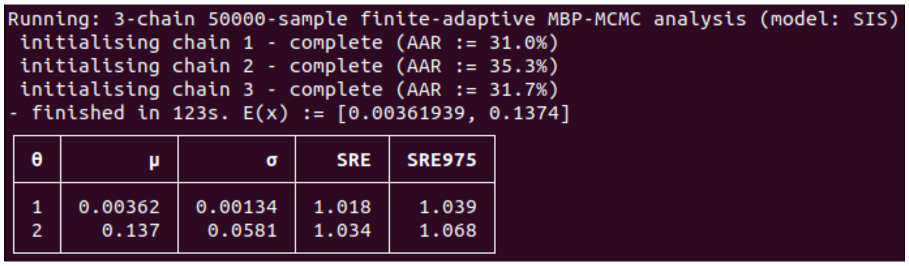
```
# NB. uses both the 'model' and 'x' variables, as per above
y = x.observations             # use simulated observations data
rs = run_mcmc_analysis(model, y; fin_adapt = true)
tabulate_results(rs)
```

Listing B.6: Run a finite-adaptive MBP-MCMC analysis.

At a minimum, the function is parametrised by two inputs: the model instance, and the observations data. Here we also specify `fin_adapt` = true for finite-adaptive MCMC[8] (as opposed to simply 'adaptive' – the default). Other optional parameters include: the number of iterations; the adaptation period; proposal algorithm (MBP or standard MCMC moves) and a vector of initial *model* parameter values.

The latter may be used to manually initialise the Markov chains. Here it is not specified, and so the initial model parameter values are sampled automatically from the prior distribution.

The final line of code in the sample above, tabulates the results of the analysis and displays it to the user. Sample output is provided in Figure B.2.



```
Running: 3-chain 50000-sample finite-adaptive MBP-MCMC analysis (model: SIS)
 initialising chain 1 - complete (AAR := 31.0%)
 initialising chain 2 - complete (AAR := 35.3%)
 initialising chain 3 - complete (AAR := 31.7%)
- finished in 123s. E(x) := [0.00361939, 0.1374]
┌───┬─────────┬─────────┬───────┬────────┐
│ θ │       μ │       σ │   SRE │ SRE975 │
├───┼─────────┼─────────┼───────┼────────┤
│ 1 │ 0.00362 │ 0.00134 │ 1.018 │  1.039 │
│ 2 │   0.137 │  0.0581 │ 1.034 │  1.068 │
└───┴─────────┴─────────┴───────┴────────┘
```

Figure B.2: Sample output from Julia for the code sample above. The tabulated results include the mean and standard deviation of posterior samples (μ and σ respectively) and the *potential scale reduction factors* $\hat{R}$.

[8]This means that the proposal distribution is only adapted during the adaptation (or 'burn-in') period.

This is only a summary of the results, which in their entirety include the MCMC samples; a covariance matrix of the same, and other information such as the computer runtime.

Iterative batch importance sampling

The second function covered here, `run_ibis_analysis`, incorporates two separate algorithms, both of which are based on the *iterative-batch-importance-sampling* technique of Chopin [44].

The default algorithm invoked is the one also proposed by Chopin, as *SMC*^{2} – described in Chapter 2 as Algorithm 8. It can be conceptually imagined as one particle filtering algorithm (e.g. Algorithm 6) embedded within another.

The second algorithm relies on the model-based-proposal (MBP) technique of Pooley (Algorithm 5). A description is given in §3.2 as Algorithm 9, or *MBP-IBIS*.

The latter is the option invoked in the code sample provided below. *SMC*^{2} can be used instead by either changing the 'algorithm' input parameter to 'SMC2', or else simply by not specifying it at all.

```
    rs = run_ibis_analysis(model, y; algorithm = 'MBPI')
    tabulate_results(rs)
```

Listing B.7: Run an MBP-IBIS analysis.

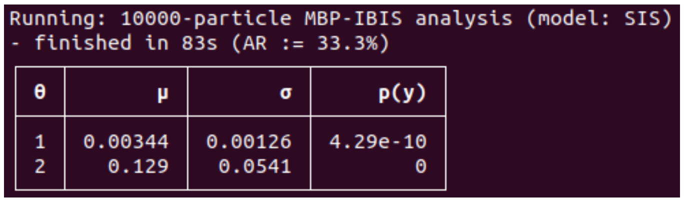
An estimate of the [log] model evidence, denoted $\ln p(y)$, is computed automatically and supplied with the tabulated results (the second line of code). Corresponding output is illustrated in Figure B.3.

Particle ARQ-MCMC

The third and last function described in this section for single-model inference corresponds to a single algorithm; an implementation of quasi-MCMC inspired method presented in §3.3 as *ARQ-MCMC*, combined with a standard particle filter.

The algorithm can be invoked by using the `run_arq_mcmc_analysis` function with a DPOMP model type, as defined within the package (although that is not the only option).

As implemented within the package, the latter is the same modular component as used within *SMC*^{2}, and the functions share certain input parameters accordingly.



```
Running: 10000-particle MBP-IBIS analysis (model: SIS)
- finished in 83s (AR := 33.3%)
┌───┬─────────┬─────────┬──────────┐
│ θ │       μ │       σ │     p(y) │
├───┼─────────┼─────────┼──────────┤
│ 1 │ 0.00344 │ 0.00126 │ 4.29e-10 │
│ 2 │   0.129 │  0.0541 │        0 │
└───┴─────────┴─────────┴──────────┘
```

(a) Marginal distribution of $\theta_1 := \beta$.

Figure B.3: Sample output and *MBP-IBIS* results, based on the SIS model problem. The tabulated summary consists of the weighted expectations value μ and the square root of the corresponding variance, σ. It also includes the natural log of the *model evidence* ($\ln p(y)$) estimated according to the method described in §2.5. 'AR' denotes the [MCMC] proposal Acceptance Rate (i.e. the 'mutation' step) – autotuned to achieve a rate of approximately $\sim \frac{1}{3}$ by default.

This is one specific application of *ARQ-MCMC*; the particle filter provides an unbiased estimate of the likelihood $\hat{\pi}(x|\theta, y)$, effectively allowing us to sample the posterior.

The function can be invoked using any [compatible] user-supplied function – not only probability densities, let alone `DPOMPModel` types. It is demonstrated here using the default [automatically generated] particle filtering algorithm provided with the package for the sake of brevity though.

That is because it requires only one additional input parameter at a minimum; the bounds of the parameter space to be mapped to the algorithm's internal 'unit cube'.

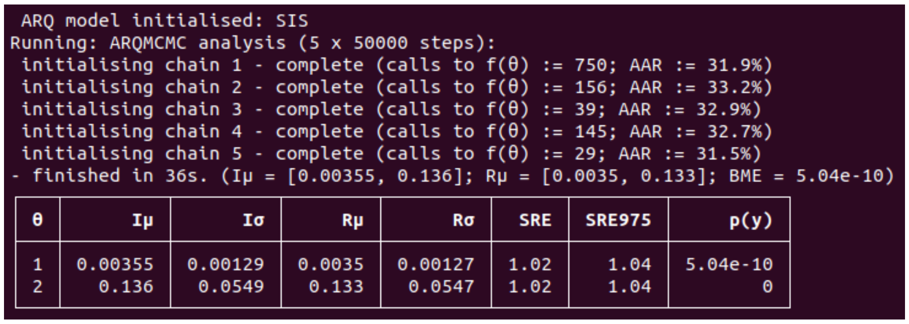
```
theta_lims = [0 0.01; 0 0.5] # NB. bounded parameter space
rs = run_arq_mcmc_analysis(model, y, theta_lims)
tabulate_results(rs)
```

Listing B.8: Run an ARQ-MCMC analysis.

The corresponding output for this code is given in Figure B.4. A crude approximation of the marginal likelihood is included with those results, but it is much less robust than the estimate provided by the IBIS algorithms. This accounts for the apparent[9] discrepancy with the Table 2.1, which records that the algorithm does

[9]Technically it is not a discrepancy; the table refers to [i.e. unbiased] *estimates* of the marginal likelihood and the *ARQ-MCMC* algorithm merely provides a biased approximation.

not compute this quantity.



```
 ARQ model initialised: SIS
Running: ARQMCMC analysis (5 x 50000 steps):
 initialising chain 1 - complete (calls to f(θ) := 750; AAR := 31.9%)
 initialising chain 2 - complete (calls to f(θ) := 156; AAR := 33.2%)
 initialising chain 3 - complete (calls to f(θ) := 39; AAR := 32.9%)
 initialising chain 4 - complete (calls to f(θ) := 145; AAR := 32.7%)
 initialising chain 5 - complete (calls to f(θ) := 29; AAR := 31.5%)
- finished in 36s. (Iμ = [0.00355, 0.136]; Rμ = [0.0035, 0.133]; BME = 5.04e-10)
```

θ	Iμ	Iσ	Rμ	Rσ	SRE	SRE975	p(y)
1	0.00355	0.00129	0.0035	0.00127	1.02	1.04	5.04e-10
2	0.136	0.0549	0.133	0.0547	1.02	1.04	0

Figure B.4: *ARQ-MCMC* results for the SIS model and data.

B.2.5 Assessing parameter inference

As has been shown, the `tabulate_results` function provides a straightforward way to view the results of the a single-model inference analysis. This approach is compatible with each of the three main functions described so far (even though the format of the results returned is slightly different in each case).

Other functionality common to all include the `print_results` function, for saving results to a directory of CSV files, and visualisation of the posterior samples (or resamples, in the case of the importance sampling scheme).

Sample code and output is provided below, with more information about each available in the package documentation.

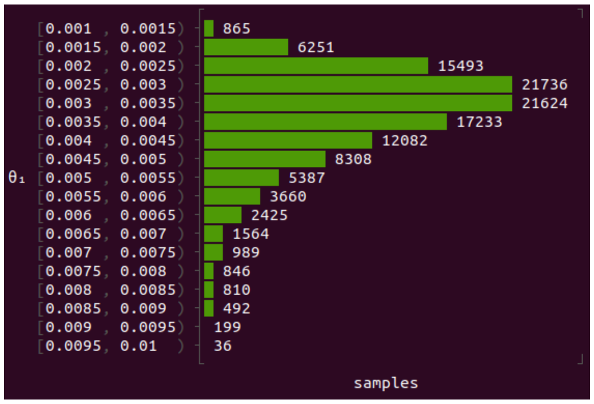
Visualisation using UnicodePlots.jl

```julia
println(plot_parameter_marginal(rs, 1))        # plot first marginal
println(plot_parameter_heatmap(rs, 1, 2))      # joint marginal
print_results(rs, 'path/to/dir')               # save to file
```

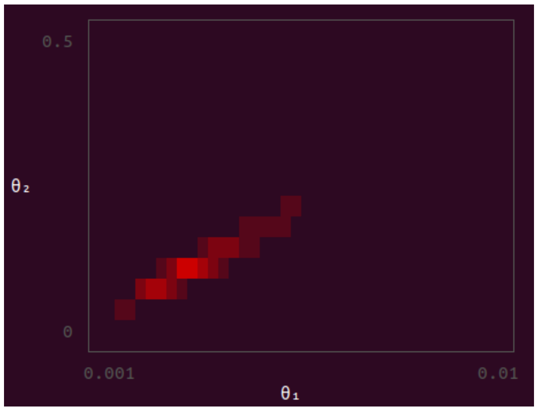
Listing B.9: Sample code for visualising the SIS model results (and saving them to file). The corresponding output is given in Figure B.5

Inline plots are implemented using the Julia package *UnicodePlots.jl*[10]. Thus,

[10]https://libraries.io/julia/UnicodePlots



(a) Marginal distribution of $\theta_1 := \beta$.



(b) Joint distribution of θ.

Figure B.5: Visualisations of posterior samples for $\{\theta_1, \theta_2\} := \{\beta\,\gamma\}$. The functions are compatible with all three inference approaches described in this section. The specific examples illustrated here were produced using results from the first method and code sample, i.e. *MBP-MCMC*.

they can be manipulated in the same way as other plots produced using that package. For example:

```
import UnicodePlots      # NB. install first, using Pkg.add
p = plot_parameter_heatmap(rs, 1, 2))
UnicodePlots.xlabel!(p, 'new x axis label')
UnicodePlots.ylabel!(p, 'new y axis label')
println(p)               # show plot
```

Listing B.10: Example: alter the axis labels of a UnicodePlot.

This is a simple example, merely intended to highlight that `p` in this case is a UnicodePlot like any other, thus the options available to users are extensive. See that package's documentation for further information and instructions.

Finally, the individual plotting functions implemented within *DiscretePOMP.jl* have been introduced throughout within context. However the complete list can be browsed in the manual section of the online package documentation.

Assessing convergence for MCMC

The main MCMC convergence diagnostic provided with the package are the *scale reduction factor* estimates provided automatically with single-model results – see the Gelman-Rubin diagnostic as discussed in §2.4.6.

'Mixing' in the Markov chains can also be visually assessed using the trace-plot function, `plot_parameter_trace(rs, 1)`, which may indicate problems in some cases (though not all). An example is illustrated in Figure B.6.

Naturally, the function is only applicable with results format produced using MCMC algorithms, including *ARQ-MCMC*. Attempting to invoke it for other types will result in an exception being thrown.

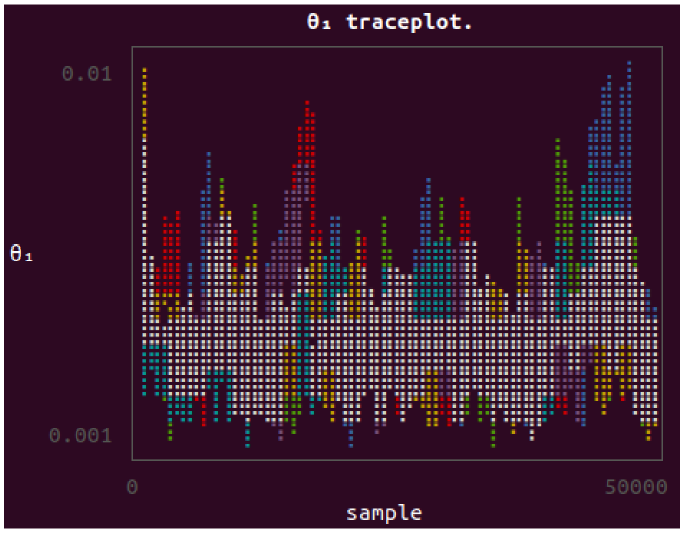


Figure B.6: Inline trace plot from DPOMPs in Julia, for the contact rate parameter in the SIS example.

B.2.6 Prediction

We have seen how parameter inference can be used to learn about the characteristics of a system. For example, we could infer the likely impact of control measures (e.g. a 'lockdown') on the contact rate parameter, by estimating it for two separate periods – with and without the control measure.

At some later time, we could use those results to 'predict' the likely impact of the same control measure (or the likely impact of removing it, depending on the circumstances). That is, by simulating the system as described above repeatedly, altering the simulation parameters as required. E.g. by choosing the expectations of the inferred distribution of θ; or resampling a set of MCMC samples.

Finally, note that the above is an everyday description of the exercise. Mathematically, we are performing numerical integration in the manner described by [30].

B.2.7 Inferring model structure

As described in §2.5, the marginal likelihood $p(y)$ can be interpreted as a probabilistic measure of model-fit; the *model evidence*. It can also be used to compute the Bayes factor, for directly comparing two models. An applied example of model selection using the Bayes factor is given in §B.3.3.

The methods described in that section for computing an estimate of the marginal likelihood have been implemented for the applicable algorithms. No additional effort is required on the part of the user, since those estimates are computed and included automatically with the results of [single model] analyses.

Model comparison using the Bayes factor

In the context of multiple ['candidate'] models, the model evidence gives rise to a natural (i.e. probabilistic and Bayesian) method for comparing different models; the Bayes factor. This is given by:

$$K_{i,j} = \frac{\pi(y|m_i)}{\pi(y|m_j)} \tag{B.5}$$

where $p(y|m_i)$ is the marginal likelihood computed [estimated] for the i^{th} model. According to the scale originally proposed by Jeffreys [65, 64], $K_{1,2} > 10$ can be considered to be strong evidence for favouring model m_1 over m_2.

$\log_{10}K$	K	Strength of evidence
0 to 1/2	1 to 3.2	Not worth more than a bare mention
1/2 to 1	3.2 to 10	Substantial
1 to 2	10 to 100	Strong
>2	>100	Decisive

Table B.1: Bayes factor interpretation, from Kass and Raftery[1].

This final stage of the analysis can be accomplished by the user quite easily based on the results of single-model analyses, since the calculation is simple; it is only the marginal likelihood itself that is difficult to compute. However a multi-model inference workflow has nonetheless been automated for convenience. It is invoked like so:

```
# NB. first define some models to compare, e.g. as m1, m2, etc
models = [m1, m2, m3]
results = run_model_comparison_analysis(models, y; n_runs = 10)
tabulate_results(results, null_index = 2)   # null model: m2
```

Listing B.11: Model comparison code. This is actual (rather than psuedo) code. However it is not executable without a valid vector of models (i.e. *m1*, etc must be defined). A full working example is provided with code and results in Appendix §B.3.3.

The Bayes factor is computed when the results are tabulated, with respect a baseline, or 'null' model, such that different choices of null model can be evaluated for the same inference results. Note that the Bayes factor of the 'null' model is therefore always = 1.0.

There is no intrinsic need to nominate a null model in this way (i.e. it is somewhat peculiar to this implementation) but doing so allows for the results to be given as a single vector of Bayes factors, for convenient and meaningful presentation to the user.

Models are defined in same way as described before, but users must take care that each is equipped with an observation model that is compatible with the same observations data y (even though the underlying parameter space and configuration of each model may be quite distinct).

Since a meaningful demonstration of this functionality requires more than one model, a completely self-contained example – also based on simulated data – is given in §B.3.3.

Per the sample code given here, tabulation of results works in the same way as before. If the null model is not specified, it is assumed to be the first model in the array used for the initial analysis, i.e. as passed to `run_model_comparison_analysis`.

The `plot_model_comparison` function provides options for visualisation, which are also explained and demonstrated in §B.3.3.

Finally, note that the entire analysis require MN independent calls to the underlying inference algorithm (Algorithm 8, or SMC^2, by default) where M is the number of models, and N is the value [optionally] passed as the `n_runs` named parameter to the `run_model_comparison_analysis` function. As such they may take a considerable amount of time to run, depending on the number and complexity of the underlying models, and data, that are being analysed.

It is therefore advisable to run single-model inference analyses for each model first, partly to ensure that the approximate total run time (i.e. of N runs each) is realistically feasible. Failing that, to at least begin with a low choice of N (the default is three).

B.3 Case studies

The key features of the package have been introduced using a single case study, borrowed from the epidemiological literature. We now provide a small selection of others to better represent the range of potential applications that such methods offer, as briefly spoken of in the introduction to this chapter. They include the models also illustrated in the introduction; predator-prey and the two-species malaria model in §B.3.1 and §B.3.2.

We also use this section to cover aspects of the functionality not yet fully explained; namely, multi-model inference (i.e. model comparison) in §B.3.3. Finally in §B.4 we demonstrate usage of custom features, by estimating the epidemiological parameters for a smallpox outbreak in Nigeria.

B.3.1 Predator-prey model

The first model we examine in this section is the Lotka-Volterra predator-prey model, loosely based on a [discrete-time] example published by Boys *et al.,* [97, 22].

Recall that individuals conceptually take one of n states; in this case predator or prey. The states are labelled in accordance with convention as $\{P, N\}$, respectively.

Three coupled processes (or 'event types') govern the internal dynamics of the model. They are, prey reproduction, labelled 'birth'; predation; and predator death. Events occur randomly, at rates labelled r_1, r_2 and r_3 respectively.

This definition (as shown in Figure B.7) is perhaps more directly relevant to interacting chemical species than to, say, a macroscopic ecosystem. In particular, predation (N $\rightarrow$ P) encompasses both prey death and predator reproduction in a way that is evocative of [bio]chemical reactions.

As demonstrated in the previous section, adaptations such as these can be accomplished quite straightforwardly, by defining a new event rate function and corresponding transition matrix, and overwriting those components for the predefined model (or by using them to define one from scratch).

As before, inter-event times are exponentially distributed, with the rates defined

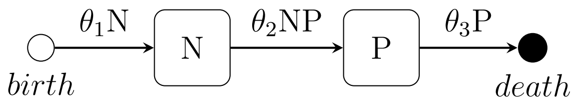


Figure B.7: The Lotka-Volterra predator-prey model. More accurately, it is a *discrete-state-space* conceptual realisation of the Lotka-Volterra predator-prey equations. In previous [epidemiological] examples, we rationalised model dynamics as the 'migration of individuals' between [discrete] states. For consistency we stay with the same symbolic representation, though it is perhaps not as fitting depending on the practical situation being considered.

as follows:

$$r_1 = \theta_1 N \tag{B.6}$$
$$r_2 = \theta_2 NP \tag{B.7}$$
$$r_3 = \theta_3 P \tag{B.8}$$

It is assumed that a prey reproduction event results in the addition of one individual to that population while the predator reproduction event adds one to that population and removes one prey. The predator death event removes one predator. Thus the transition matrix for the model is defined as:

$$T = \begin{bmatrix} 0 & 1 \\ 1 & -1 \\ -1 & 0 \end{bmatrix} \tag{B.9}$$

Specifying these aspects of the model manually is straightforward, the sample code is provided for illustration in B.5.1, but not necessary in this case because it is predefined. The sample code provided here uses the latter technique for brevity:

```
model = generate_model("LOTKA", [79, 71])
theta = [0.5, 0.0025, 0.3]
x = gillespie_sim(model, theta)  # run simulation
p = plot_trajectory(x)           # plot (optional)
```

Listing B.12: Simulate a Lotka-Volterra realisation.

The simulated realisation and observations dataset produced by this code are illustrated in Figure B.8.

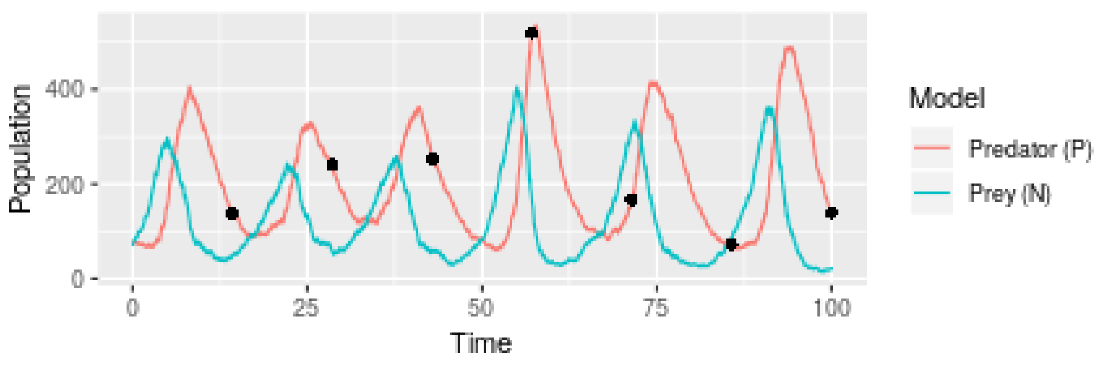


Figure B.8: Realisation of the Lotka-Volterra predator-prey model, produced using the code sample above. Note that this particular illustration was produced using the *ggplot2* package in *R* [7] – not the inline package function in Julia referred to above.

As noted in the previous section, the ability to simulate observations in particular is important for inference purposes.

B.3.2 Vector-host model

In the first case study, we saw how DPOMP-based models can be used to notionally represent different 'species', or other types of population process (e.g. reproduction, migration) beyond those considered in the previous epidemiological examples.

There are of course many conceivable situations where we might wish to combine such concepts, to yield a unified representation of both disease and more general population dynamics.

Here we consider such a model, based on the equations of George MacDonald [98] and Ronald Ross [77], and usually therefore referred to as the Ross-MacDonald model [99, 100].

It is a standard mathematical framework for modelling vector-borne disease, and was initially proposed for *malaria*, a potentially fatal infectious disease carried by mosquitoes which can affect humans and other animals – caused by a parasitic single-celled organism (or protozoa) of the *Plasmodium* genus.

As per the representation shown in Figure 5.1b, it is essentially an extension of the standard susceptible-infectious[-susceptible] (SIS) model [3] with individuals labelled S and I in accordance with convention, and two distinct species denoted by subscript.

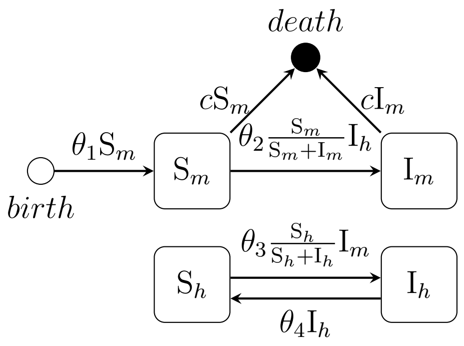


Figure B.9: A simplified DSS interpretation of the Ross-MacDonald malaria model; here it is assumed that the average lifespan of mosquitoes is known to be c^{-1}.

Note that within the package itself the use of subscripts in this way is not practical. Thus by default human individuals are labelled S and I, and susceptible and infectious mosquito populations are denoted by A and B respectively.

Code for simulating the Ross-MacDonald model

```
model = generate_model("ROSSMAC", initial_state)
theta = [0.5, 0.0025, 0.3]
x = gillespie_sim(model, theta)  # run simulation
p = plot_trajectory(x)           # plot (optional)
```

Listing B.13: Model simulation.

B.3.3 Model selection: simulated example

We now turn our attention to the task of statistical inference in the context of multiple models. That is, situations where there are multiple possible models that may fit the data in question, and our purpose is to select the one that is optimal.

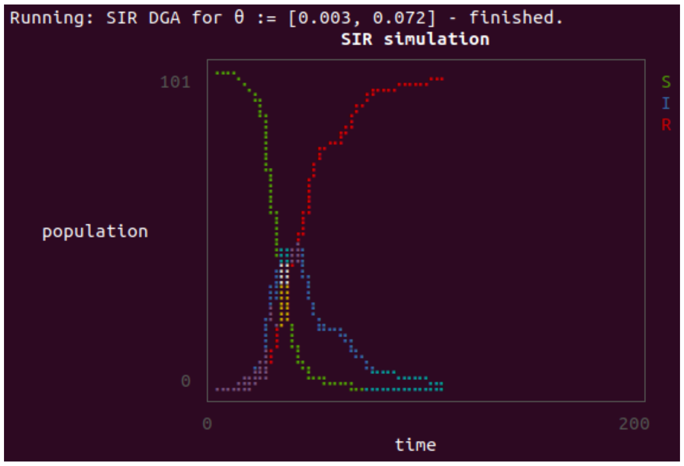
To demonstrate this in practice, we apply two models; the SIR, and the SEIR, and utilise observations data simulated from each (i.e. two separate scenarios). As with previous examples, we assume that the underlying system is partially observed; only the number of *infectious* individuals is recorded at each observation time. The same observation model is used for all – it is assumed that observation errors are normally distributed $\sim N(0, \sigma^2)$ and that $\sigma = 2$ (i.e. the default observation model in the package).

In plain terms then, the (imaginary) scientific question to be addressed, is whether or not an infectious disease includes a significant non-infectious latent period, which should be accounted for when attempting to explain its prevalence using a model and partial data.

We simulate and analyse both scenarios in order to better test the robustness of the approach under different conditions. Note that unlike other measures of model deviance, the model evidence $p(y)$ does not incorporate an explicit term penalty. In accordance with general scientific principles, we therefore assume that our preference (i.e. the 'null' model) in both cases is the simpler one, the SIR. This is the model used to compute the Bayes factor for each of the models being compared. It is computed 'on-the-fly' for the tabulation function itself, such that multiple candidates for the null can be considered for the same set of results.

We now present both scenarios in turn, along with code samples and instructions for reproducing the analyses; tabulating the results, and optionally specifying the null model; and visualising the log estimates of the model evidence obtained by the underlying [designated] algorithm.

Scenario one: simulated SIR data



```
Running: SIR DGA for θ := [0.003, 0.072] - finished.
```

Figure B.10: Package output: the simulated SIR epidemic. The parameters used were $\theta = \{0.003, 0.072\}$. Observations are recorded at intervals of slightly less than four time units. Since this only intended to be a simple illustrative example, it is assumed that the initial infection time is known to be $t_0 = 0$, and that the overall population size remains constant.

We begin with a preliminary assessment. That is, single-model inference for each of the models in consideration. That is to ensure that the algorithm and configuration chosen for the main analysis is adequate enough to ensure reliable estimates.

Only select output from the preliminary analyses have been included here, since that functionality has already been addressed at length.

In summary though, it was originally envisaged that three or four models would be assessed, including a susceptible-exposed-infectious-susceptible (SEIS) model. That model is very similar to the SEIR but allows that infection does *not* confer immunity to previously infected hosts.

The single-model inference analysis revealed that only the SIR and SEIR fit data simulated by each of those models sufficiently well to even provide reliable estimates of the model evidence, and so only those two were carried forward into the main analysis.

Single-model inference also provides a basis for cross-validation of the final [multi-model inference] results. For example, inference using any of the MCMC algorithms provides evidence for convergence via the *scale reduction factor estimates*.

In this case, parameter estimates obtained using the MBP-MCMC algorithm were used to validate those produced by the SMC^2 algorithm, which was then used in turn to compute independent estimates of the model evidence itself in the main analysis.

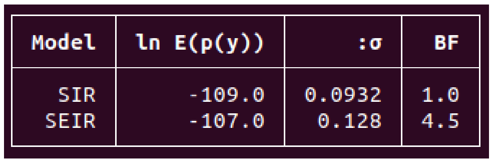
That analysis was run using:

```
# TBA: MODELS INCL OM
models = [seir, sir] # models must be defined first
mcomp = run_model_comparison_analysis(models[[1,2]], y)
tabulate_results(mcomp; null_index = 2)
p = plot_model_evidence(mcomp)
```

Listing B.14: Model comparison.

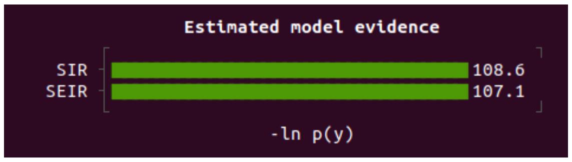
Here the 'null' model – the evidence for which is used to compute the Bayes factor for every model – is specified by passing the index of the null model, with respect to the array of models used to conduct the original analysis, as an additional named parameter to the tabulation function. Else it is assumed to be the first model that was specified in that array.

The output is given in Figure B.11.



Model	ln E(p(y))	:σ	BF
SIR	-109.0	0.0932	1.0
SEIR	-107.0	0.128	4.5

(a) Results table.



(b) Default plot.

Figure B.11: Model comparison output from DPOMPs in Julia. Both give the natural log of the [arithmetic] mean model evidence, estimated from (in this case) three independent runs of the SMC^2 algorithm per model. The Bayes factor (BF) is also given, based on the same.

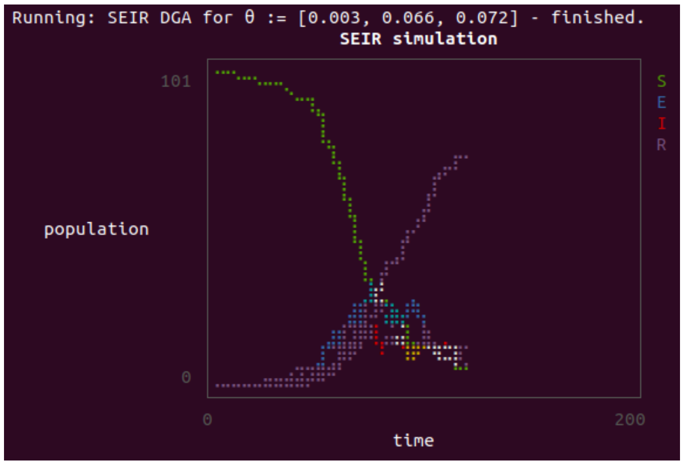
As per the results table on the LHS, the model evidence was actually better (i.e. higher, or equivalently, the *negative* log evidence was lower) for the 'wrong' model, the SEIR. However the Bayes factor is less than ten, suggesting that there is not [yet] sufficiently good evidence to justify rejecting the SIR model in favour of it.

Thus we have at least tentatively, and by Occam's Razor[11], arrived at the correct conclusion – we favour the SIR in this case.

The final line of code in the code sample above plots the [negative] geometric mean (i.e. the arithmetic mean of the evidence itself) by default. The same function can also be be used to produce a box-plot of the log estimates obtained, per the example given in Figure B.14 (which is based on the results of the second scenario).

Scenario two: simulated SEIR data

Here we provide the results of the second scenario for illustrative comparison. The code required to run it is virtually identical. However it is a different scenario, in that we now designate the null model as the one that did *not* produce the simulated results, in practical terms the 'wrong' one.



```
Running: SEIR DGA for θ := [0.003, 0.066, 0.072] - finished.
```

Figure B.12: The simulated SEIR epidemic. The parameters used were $\theta = \{0.003, 0.066, 0.072\}$. Observations are recorded intervals identical to that of the first scenario and it is again assumed that the initial infection time is known to be $t_0 = 0$, with the overall population size remaining constant.

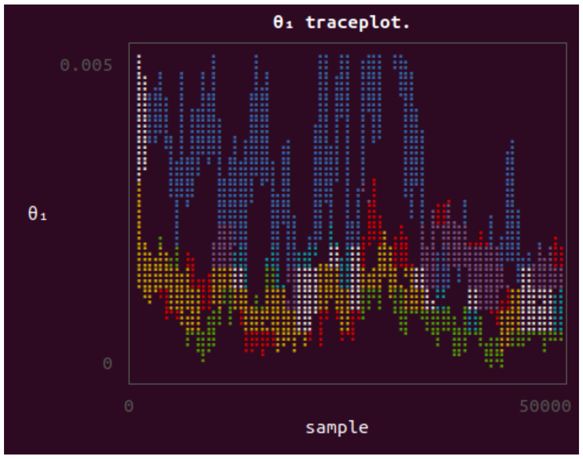
In practice another distinction, that is actually helpful in this case for illustrative purposes, is that a problem was encountered during the preliminary (i.e. single-model inference) validation analysis.

On that occasion, the DA-MCMC algorithm did not successfully converge. This was identified by the MCMC convergence diagnostic computed automatically with

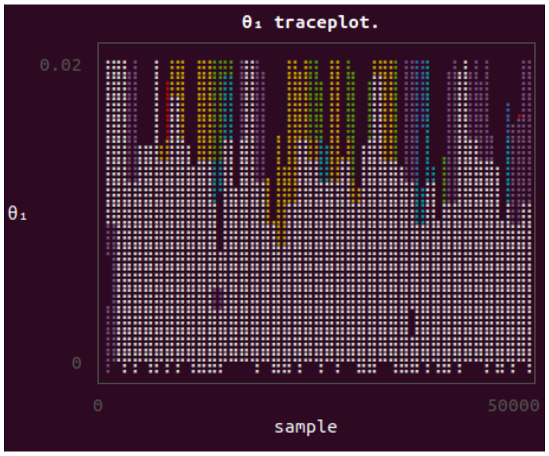
[11]Because our choice to designate the simplest model – the SIR – as the null model was guided by that principle at the outset.

the analysis, with the potential scale reduction factors denoted $\hat{R}$ estimated to be between 1.6 and 2, significantly surpassing the commonly used acceptable threshold of 1.1.

The issue was further confirmed by visual inspection of the trace-plots. An illustrative comparison with the ARQ-MCMC algorithm (which did successfully converge) is shown for the contact rate parameter in Figure B.13.



(a) MBP-MCMC

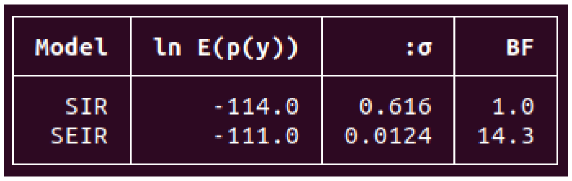


(b) ARQ-MCMC

Figure B.13: Trace-plots from the SEIR validation analyses conducted for the second scenario. θ_1 here is the contact rate parameter, more commonly denoted as β.

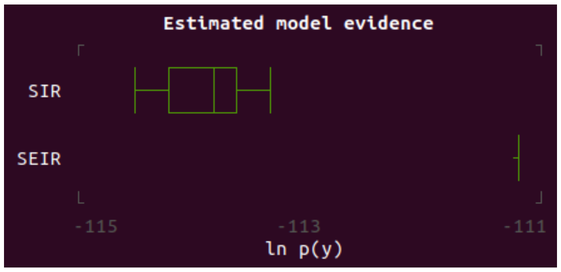
Secondary validation was carried out using the MBP-IBIS and ARQ-MCMC single-model algorithms, which both yielded parameter estimates consistent with SMC^2, allowing the main analysis to proceed as before.

That yielded the results given in Figure B.14. In this case the Bayes factor of the SEIR model is high enough that we can regard the results as strong evidence for favouring that model. Thus we have correctly identified the model used to simulate the results, based on only partial data - the number of infectious individuals recorded at intervals (and assuming that the distribution of observation errors is known).



```
┌───────┬────────────┬────────┬──────┐
│ Model │ ln E(p(y)) │     :σ │   BF │
├───────┼────────────┼────────┼──────┤
│   SIR │     -114.0 │  0.616 │  1.0 │
│  SEIR │     -111.0 │ 0.0124 │ 14.3 │
└───────┴────────────┴────────┴──────┘
```

(a) Results table.



(b) Box plot.

Figure B.14: Model comparison output from the second analysis, based on simulated SEIR data. The box plot shows the range of log estimates for the model evidence.

B.4 Another example application: Smallpox

Note that the code samples provided with this section are not compatible with the most recent version of the package.

This case study is based on data published by O'Neill and Roberts in their analysis of a smallpox outbreak within a closed community of one hundred and twenty individuals in Abakaliki, Nigeria [8].

B.4.1 Problem statement

The goal is to repeat the same kind of analysis as presented in the original paper, albeit for a *continuous-time* DPOMP model, and estimate epidemiological parameters for the SIR model.

Recovery times were recorded (i.e. assumed to be known) to within the closest day, and no information regarding the number of infected individuals is available, save that the epidemic is assumed to be complete following the final recovery – that is, the number of infected individuals is assumed to be zero at that time.

These assumptions lead to a fixed trajectory length, and specific requirements for the proposal algorithm. For example, since it is assumed that the total number of events is known, we can construct an algorithm that simply alters the time of events – there is no need to delete events or to insert new ones.

In summary:

- Events and the new times can easily be sampled on a uniform basis, ensuring that $g(X_{f \to i}) = g(X_{i \to f})$, such that the terms cancel in equation 2.21. This is the same as the standard MCMC proposal algorithm available the package.

- Some additional information is available regarding the times of recoveries; they are known to within a day. We can therefore propose new recovery event times such that they remain within that time frame (a day is regarded as a single time unit in this model).

- The initial sample, including the latent variable (i.e. a complete realisation of the model) can optionally be passed to the algorithm as a function (in place of one sampled automatically via Algorithm 1). That is preferable in this case, due to the information already known about the possible values of x.

As to the last, a constructor function is provided for manually initialising values of x – the `generate_custom_x0` function, with details given in the package documentation.

B.4.2 Model

The model is based on the standard SIR, so it makes sense to begin with that predefined option and make modifications as necessary. Thus:

```
# define model
model = DPOMPs.generate_model("SIR", [119, 1, 0]);
model.t0_index = 3
# add "medium" prior per the original paper
p1 = Distributions.Gamma(10, 0.0001)
p2 = Distributions.Gamma(10, 0.01)
p3 = Distributions.Uniform(-360, 0)
model.prior = Distributions.Product([p1,p2,p3])
```

Listing B.15: Model definition.

We also define the observations data itself:

```
# removal times and 'observation' (final time)
t = [0.0, 13.0, 20.0, 22.0, 25.0, 25.0, 25.0, 26.0, 30.0, 35.0, ↩
    38.0, 40.0, 40.0, 42.0, 42.0, 47.0, 50.0, 51.0, 55.0, 55.0, ↩
    56.0, 57.0, 58.0, 60.0, 60.0, 61.0, 66.0];
y = [DPOMPs.Observation(67.0, 1, 1.0, zeros(Int64,1))]
```

Listing B.16: Observations data.

In this case, we are dealing with a [partial] observation process, rather than a separate observation likelihood model as before, but we still define a 'dummy' observation array. The single observation time is be set to the maximum possible event time.

The observation model property must also be overwritten:

```
# dummy observation model
observation_model(y::DPOMPs.Observation, population::Array{Int64↩
    , 1}, theta::Array{Float64, 1}) = 0.0
model.obs_model = observation_model
```

Listing B.17: Observation model.

Custom proposal function

With the model defined, we move on to the proposal function itself. This generates new trajectories according to the requirements laid out in the problem statement. In this example, the function signature is specified for the correct data type. This is recommended, but not necessary.

```julia
## custom proposal algorithm
function custom_proposal(xi::DPOMPs.Particle)
    seq_f = deepcopy(xi.trajectory)                    # new trajectory
    t0 = xi.theta[model.t0_index]                      # NB. initial infection time
    evt_i = rand(1:length(xi.trajectory))              # choose event and new time
    evt_tm = xi.trajectory[evt_i].event_type == 1 ? (rand() * (y[end].time - t0)) + t0 : floor(xi.trajectory[evt_i].time) + rand()
    evt = DPOMPs.Event(evt_tm, xi.trajectory[evt_i].event_type)
    seq_f[evt_i] = evt                                 # replace with new event
    sort!(seq_f)                                       # sort
    return DPOMPs.Particle(xi.theta, xi.initial_condition, copy(xi.initial_condition), seq_f, xi.prior, zeros(2))
end # end of std proposal function
```

Listing B.18: Custom proposal function.

It is however important to return the correct data type (which is the same as the input parameter).

Initial trajectory function

We also specify a function sampling the initial trajectory variable, this time making use of the `generate_custom_particle` function to define the output. This instantiates the particle and computes its likelihood, allowing us to ensure that it is a valid trajectory:

```
## initial trajectory function
function gen_x0(theta::Vector{Float64})
    trajectory = DPOMPs.Event[]
    for i in 1:(length(t) - 1)      # infections: arbitrary t (must be > t0)
        push!(trajectory, DPOMPs.Event(i == 1 ? theta[model.t0_index] * rand() : t[i] * rand(), 1))
    end
    for i in eachindex(t)           # recoveries
        push!(trajectory, DPOMPs.Event(t[i],2))
    end
    x0 = DPOMPs.generate_custom_particle(model, y, trajectory; theta = theta)
    println("x0 log like: ", x0.log_like)
    return x0
end
```

Listing B.19: Initial trajectory function.

B.4.3 MCMC analysis

Finally, we can run the analysis itself. The package documentation has complete information about using `run_custom_mcmc_analysis` function. However it essentially works in the same manner as for the standard function.

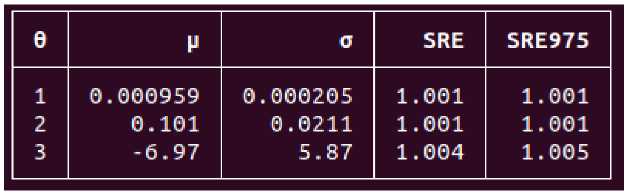
The custom proposal function is a required parameter, but the initial trajectory function is optional. The remaining optional parameters are named, and most correspond to the standard function.

```
## run MCMC
rs = run_custom_mcmc_analysis(model, y, custom_proposal, gen_x0;↩
    steps = 100000)
tabulate_results(rs)
println(plot_parameter_trace(rs, 1))
println(plot_parameter_marginal(rs, 1))
println(plot_parameter_heatmap(rs, 1, 2))
```

Listing B.20: Run the analysis.

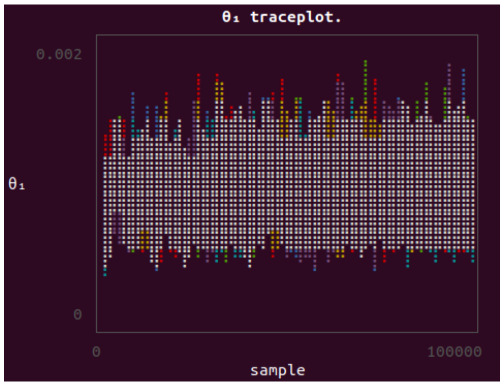
The results and other sample output are given in Figure B.15. Due to the nature of the underlying models and approach, they are not directly comparable with those of the original authors. In this case we also used a multi-chain analysis, so as to make better use of the convergence diagnostics within the package.

However a single-chain validation analysis was carried out using a separate software implementation, and compared to the results of the original work, notwithstanding the remaining differences in approach. The results for that analysis are given in B.6.

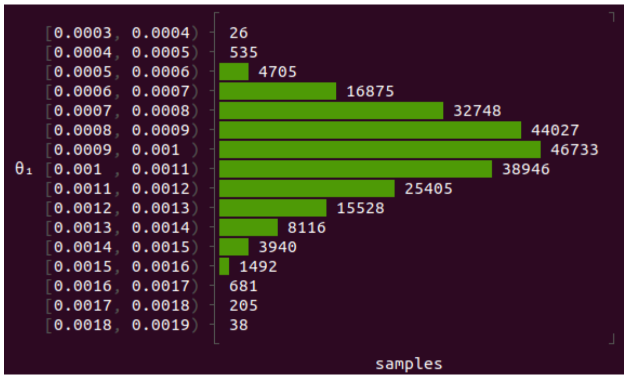


θ	μ	σ	SRE	SRE975
1	0.000959	0.000205	1.001	1.001
2	0.101	0.0211	1.001	1.001
3	-6.97	5.87	1.004	1.005

(a) Results table.

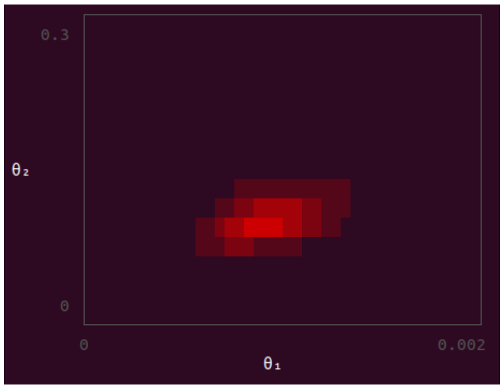


(b) Trace plot.



```
   [0.0003, 0.0004) ┤ 26
   [0.0004, 0.0005) ┤ 535
   [0.0005, 0.0006) ┤ 4705
   [0.0006, 0.0007) ┤ 16875
   [0.0007, 0.0008) ┤ 32748
   [0.0008, 0.0009) ┤ 44027
   [0.0009, 0.001 ) ┤ 46733
θ₁ [0.001 , 0.0011) ┤ 38946
   [0.0011, 0.0012) ┤ 25405
   [0.0012, 0.0013) ┤ 15528
   [0.0013, 0.0014) ┤ 8116
   [0.0014, 0.0015) ┤ 3940
   [0.0015, 0.0016) ┤ 1492
   [0.0016, 0.0017) ┤ 681
   [0.0017, 0.0018) ┤ 205
   [0.0018, 0.0019) ┤ 38
                          samples
```

(c) θ_1 marginal.



(d) θ_1, θ_2 joint marginal.

Figure B.15: Inference results for the analysis of smallpox data, loosely based on that of O'Neill and Roberts [8].

B.5 Sample code

Here we provide various code snippets. Please note

B.5.1 Rate function sample code

The following sample code defines the event rate function and transition matrix for the Lotka-Volterra predator-prey model [97].

```
# rate function
function lotka_rf(output, parameters::Array{Float64, 1}, population::Array{Int32, 1})
    # prey; predator reproduction; predator death
    output[1] = parameters[1] * population[2]
```

```
        output[2] = parameters[2] * population[1] * population[2]
        output[3] = parameters[3] * population[1]
    end
    # transition matrix
    m_t = [0 1; 1 -1; -1 0]
```

Listing B.21: Lotka-Volterra rate function

Note that DPOMPs automatically defines the output array as the correct size, and passes a reference to it when calling the function is [internally] called within the package.

As such, the user simply has to populate it for each event type, given the model parameters; the population vector; and any user-defined constants, or other valid Julia expression that the user wishes to introduce.

B.5.2 Custom SIS model code

The following snippet of code manually defines the SIS model example TBC**

```julia
## define model
# rate function
function sis_rf!(output, parameters::Array{Float64, 1}, population::Array{Int64, 1})
    output[1] = parameters[1] * population[1] * population[2]
    output[2] = parameters[2] * population[2]
end
# define obs function (no error)
obs_fn(population::Array{Int64, 1}) = population
# prior
function prior_density(parameters::Array{Float64, 1})
    parameters[1] > 0.0 || return 0.0
    parameters[2] > 0.0 || return 0.0
    return 1.0
end
# obs model
function si_gaussian(y::Array{Int64, 1}, population::Array{Int64, 1})
    obs_err = 2
    tmp1 = log(1 / (sqrt(2 * pi) * obs_err))
    tmp2 = 2 * obs_err * obs_err
    obs_diff = y[2] - population[2]
    return tmp1 - ((obs_diff * obs_diff) / tmp2)
end
# define model
model = DPOMPs.DPOMPsModel("SIS", sis_rf!, [-1 1; 1 -1], [100, 1], obs_fn, prior_density, si_gaussian, 0)
```

Listing B.22: Generate a ‘custom’ SIS model instance

B.6 Smallpox validation analysis

An analysis for the Smallpox case study example given in the main body of this appendix was carried out using a separate software implementation for the purposes of validation. Those results are reported here in brief, since the model and problem have already been described.

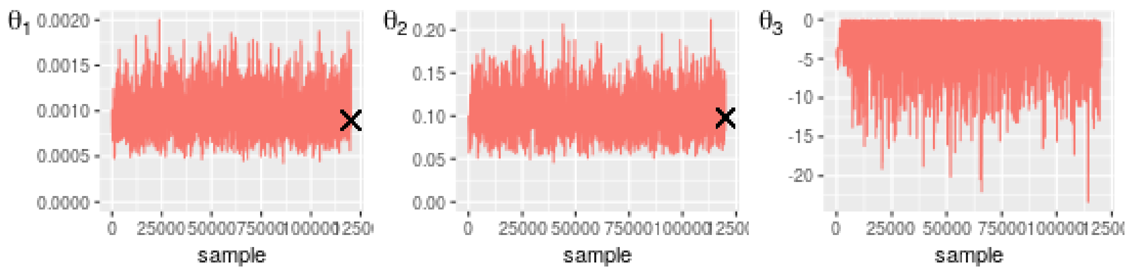


Figure B.16: θ trace plots from the custom MCMC analysis of the smallpox data with 'medium' prior distributions for transmission, removal coefficients θ_1; θ_2, initial infection time θ_3. The results reported by O'Neill and Roberts for their discrete time model are not directly comparable, but are nonetheless marked for reference.

θ	E(X)	SD	Mode	95% CI	z
1	0.00099	0.000211	0.00094	(0.00063 - 0.0015)	0.05
2	0.1	0.0221	0.095	(0.066 - 0.15)	0.15
3	-3.2	3.11	-0.085	(-12 - -0.077)	0.06

Table B.2: SIR model MCMC analysis: results from a single-chain 'custom' MCMC analysis, based on the dataset and proposal function described. The prior density function used in this example matches that of O'Neill and Roberts (described in that paper as the 'medium' prior distribution). The estimates obtained by the authors (albeit using a different model) were reported as $\theta_1 = 0.0011$ and $\theta_2 = 0.107$ for $[\theta] = \{\beta, \gamma\}$. The initial infection time $\theta_3 = t_0$ was not reported in that paper.

Chain	Proposed	Accepted	Rate
Adapted	100000	61253	0.613
Burn in	20000	12975	0.649
Total	120000	74228	0.619

Table B.3: SIR model MCMC analysis: MCMC proposal summary

Appendix C

Detailed results for within-herd BTB scenarios

Here we provide additional information concerning the initial inference analyses that were carried out as preliminary work towards the results reported in Chapter 6. They were produced more as a proof of concept, with heterogeneity of diagnostic test types ignored, for example. They consist of three independent 'herd scenarios' that are somewhat similar to simulated scenario results also reported in that chapter.

The are included here because they represent the first results obtained throughout the course of the project that were based on actual BTB surveillance data. Perhaps more importantly, they provide a flavour as to how the main results reported in the corresponding chapter were compiled.

Each scenario includes a summary of the underlying observational data in the first Figure. Also included are visual and tabulated summaries of the parameter inference results, and lastly estimates obtained for the marginal likelihood.

C.1 Simulated scenarios

Here we report the full inference results for the simulation scenarios discussed in Chapter 6.

SEI-tdR validation analysis

C.1.1 Scenario one: one diagnostic test

```
## SMC2 analysis:
##   Theta      iMu     iSD
## 1     1  1.1e-04 7.0e-05
## 2     2  1.9e-02 2.4e-02
## 3     3 -1.5e+02 8.7e+01
## 4     4  6.5e-01 1.4e-01
```

Figure C.1: Inference summaries for the primary (i.e. SMC^2) $SEIR$ analysis: scenario one. The model parameters labeled from one to four are as follows: contact parameter $S \rightarrow E$; progression rate $E \rightarrow I$; SICCT test sensitivity; and onset of infectiousness in the first affected individual, also denoted in the main text as t_0.

```
## MBP IBIS analysis:
##   Theta      iMu     iSD
## 1     1  1.1e-04 6.2e-05
## 2     2  1.4e-02 2.3e-02
## 3     3 -1.7e+02 9.1e+01
## 4     4  6.9e-01 1.6e-01
```

```
## ARQMCMC analysis:
##   Theta      iMu     iSD      rMu     rSD SRE (95%)
## 1     1  1.1e-04 5.1e-05  1.1e-04 5.5e-05   1     1
## 2     2  1.3e-02 2.2e-02  1.6e-02 2.5e-02   1     1
## 3     3 -1.6e+02 9.9e+01 -1.6e+02 1.0e+02   1     1
## 4     4  6.9e-01 1.6e-01  6.7e-01 1.6e-01   1     1
```

Figure C.2: Inference summaries for the two validation analyses corresponding to the results given in Figure C.1 (incident one).

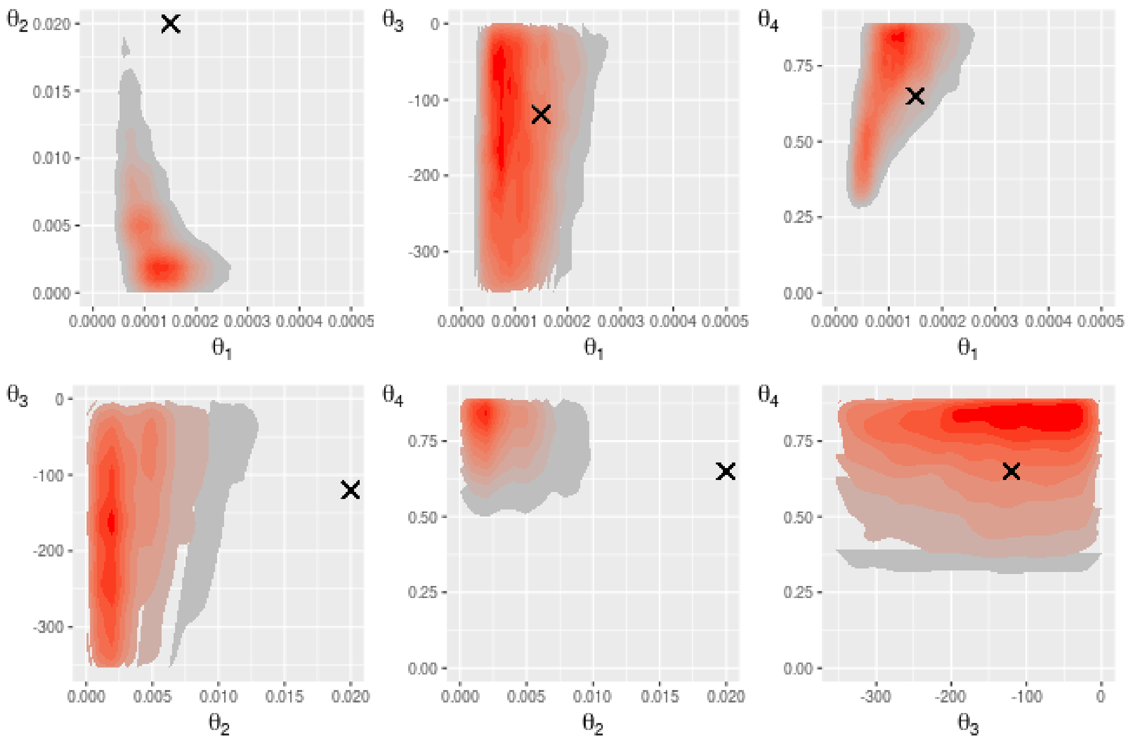


Figure C.3: Twin parameter marginal densities for the *SEIR* model analysis (Scenario one; ARQ-MCMC).

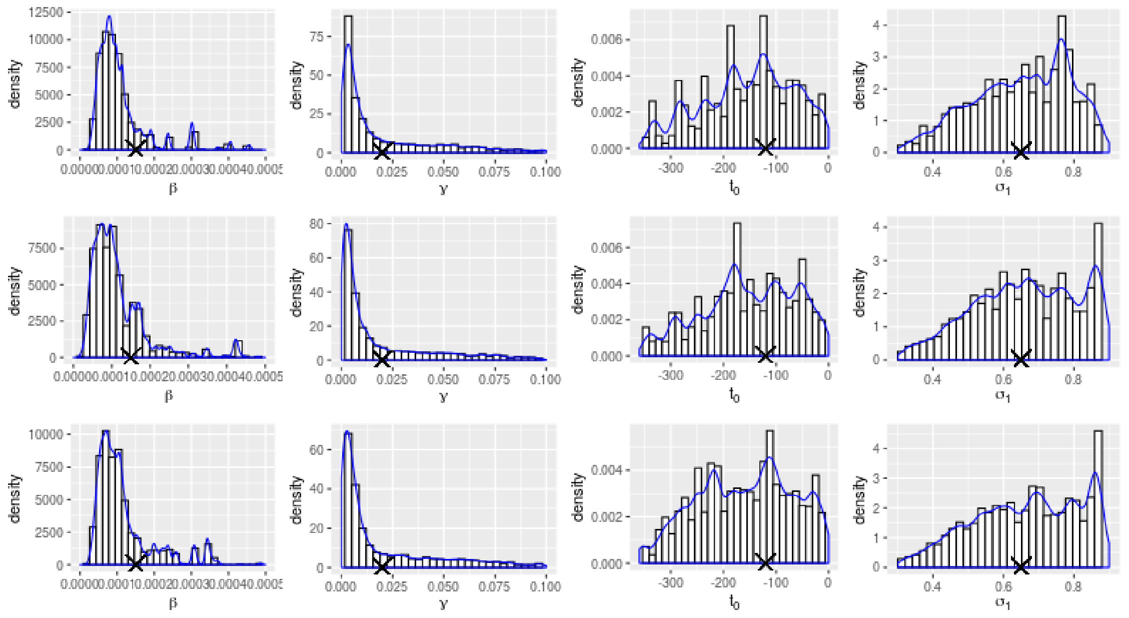


Figure C.4: Marginal model parameter distributions for the first *SEIR* model scenario. Each row represents a separate run of the SMC^2 algorithm with $N = \{6000, 8000, 10000\}$ (independent proposals). Left to right: contact rate $S \rightarrow E$; progression rate $E \rightarrow I$; onset of infectiousness in the first affected individual; and diagnostic test sensitivity. The simulation values have been marked for reference.

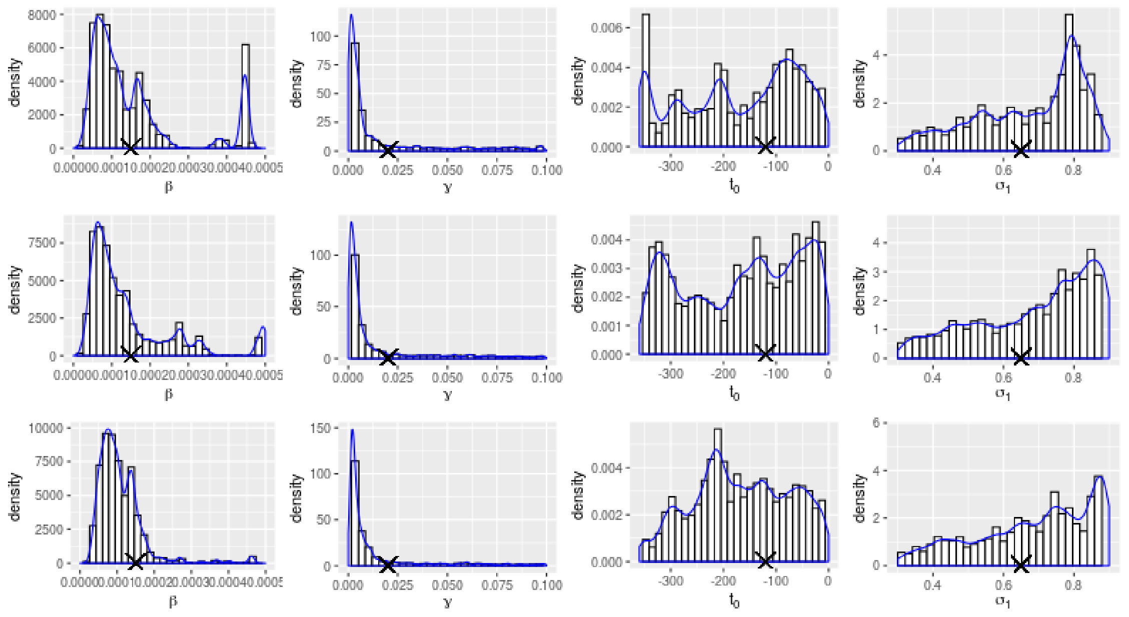


Figure C.5: As per Figure C.4, but for the MBP-IBIS algorithm. Each row represents a separate run of the algorithm with ...

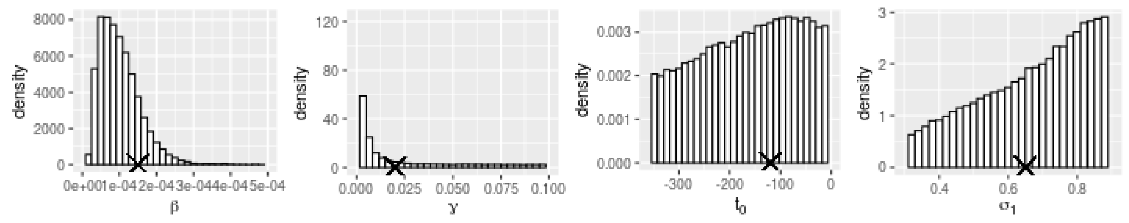


Figure C.6: As per Figure C.4, but for the ARQ-MCMC algorithm. Each row represents a separate run of the algorithm with ...

C.1.2 Scenario two: two diagnostic tests

```
## SMC2 analysis:
##   Theta      iMu     iSD
## 1     1  1.3e-04 2.4e-05
## 2     2  5.4e-02 2.2e-02
## 3     3 -1.1e+02 7.5e+01
## 4     4  5.3e-01 6.5e-02
## 5     5  7.2e-01 1.4e-01
```

```
## MBP IBIS analysis:
##   Theta      iMu     iSD
## 1     1  1.3e-04 2.6e-05
## 2     2  5.0e-02 2.6e-02
## 3     3 -1.3e+02 9.2e+01
## 4     4  5.2e-01 7.8e-02
## 5     5  7.2e-01 1.4e-01
```

```
## ARQMCMC analysis:
##   Theta      iMu     iSD      rMu     rSD SRE (95%)
## 1     1  1.2e-04 2.1e-05  1.3e-04 2.6e-05   1      1
## 2     2  5.2e-02 2.4e-02  5.2e-02 2.5e-02   1      1
## 3     3 -1.0e+02 7.4e+01 -1.2e+02 9.0e+01   1      1
## 4     4  5.3e-01 6.0e-02  5.3e-01 7.6e-02   1      1
## 5     5  7.4e-01 1.4e-01  7.3e-01 1.5e-01   1      1
```

Figure C.7: Inference summaries for all three *SEIR* analyses (scenario two. The model parameters labeled from one to five are as follows: contact parameter $S \rightarrow E$; progression rate $E \rightarrow I$; SICCT test sensitivity; and onset of infectiousness in the first affected individual, also denoted in the main text as t_0.

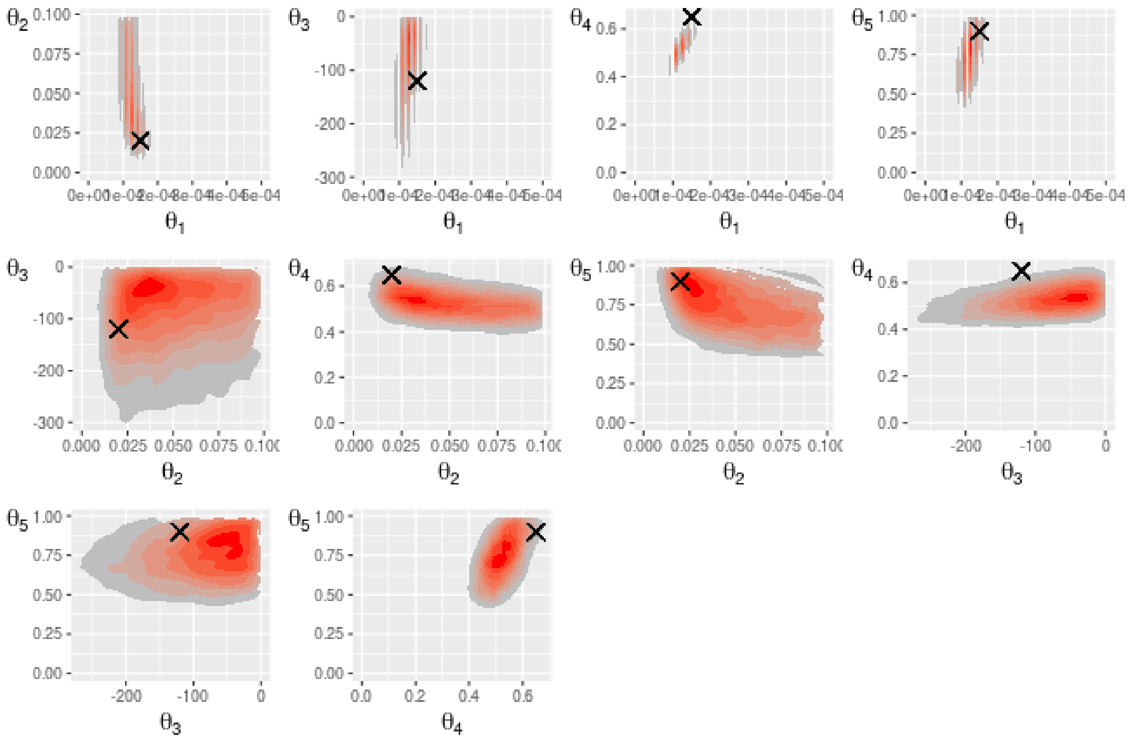


Figure C.8: Twin parameter marginal densities for the *SEIR* model analysis (Scenario two; ARQ-MCMC).

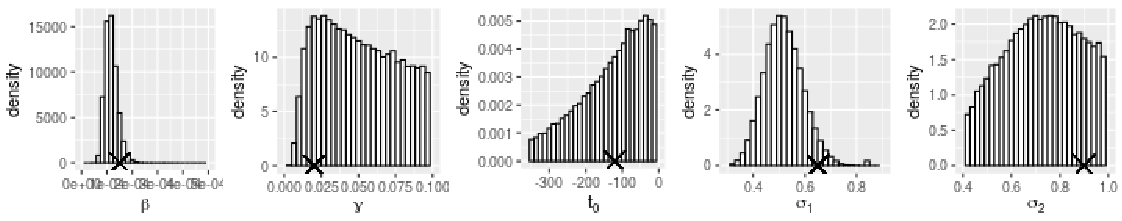


Figure C.9: As per Figure C.10, but for the ARQ-MCMC algorithm. Each row represents a separate run of the algorithm with ...

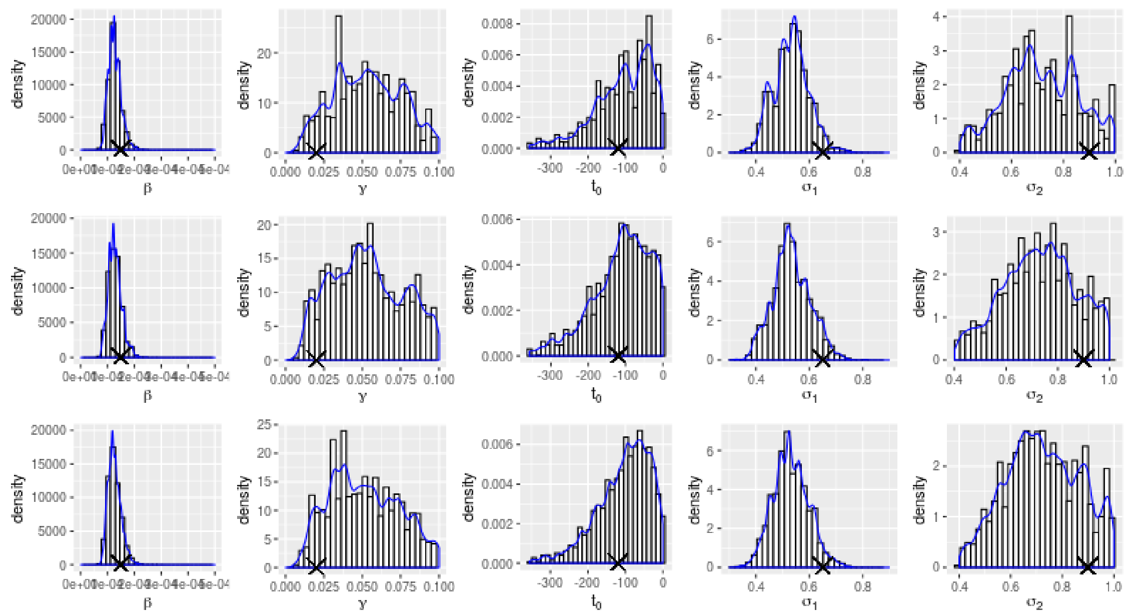


Figure C.10: Marginal model parameter distributions for the first *SEIR* model scenario. Each row represents a separate run of the SMC^2 algorithm with $N = \{6000, 8000, 10000\}$ (independent proposals). Left to right: contact rate $S \rightarrow E$; progression rate $E \rightarrow I$; onset of infectiousness in the first affected individual; and diagnostic test sensitivity. The simulation values have been marked for reference.

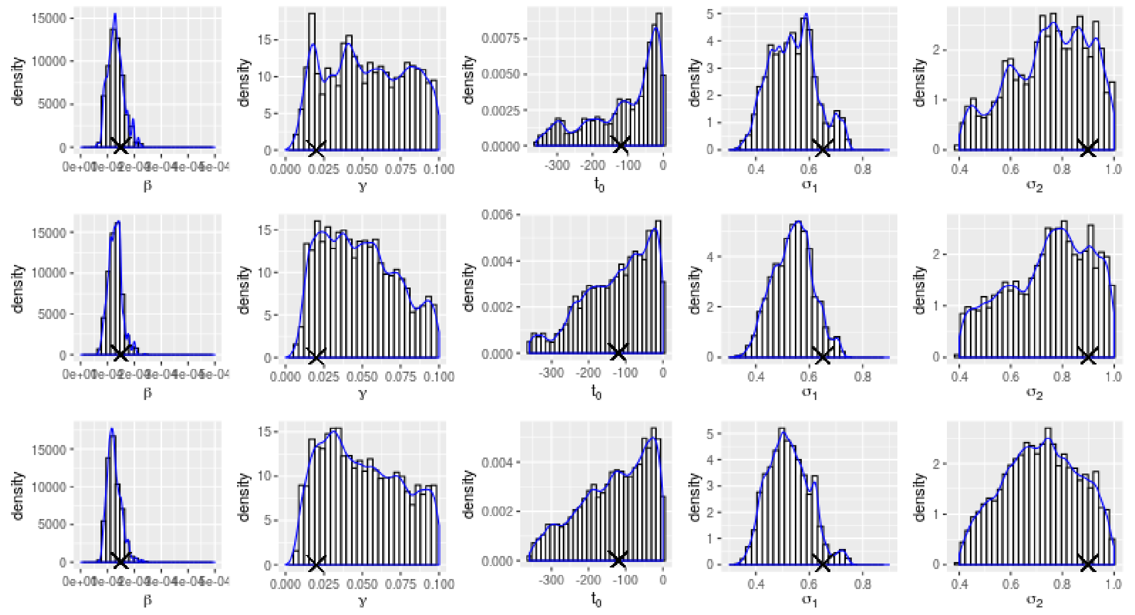
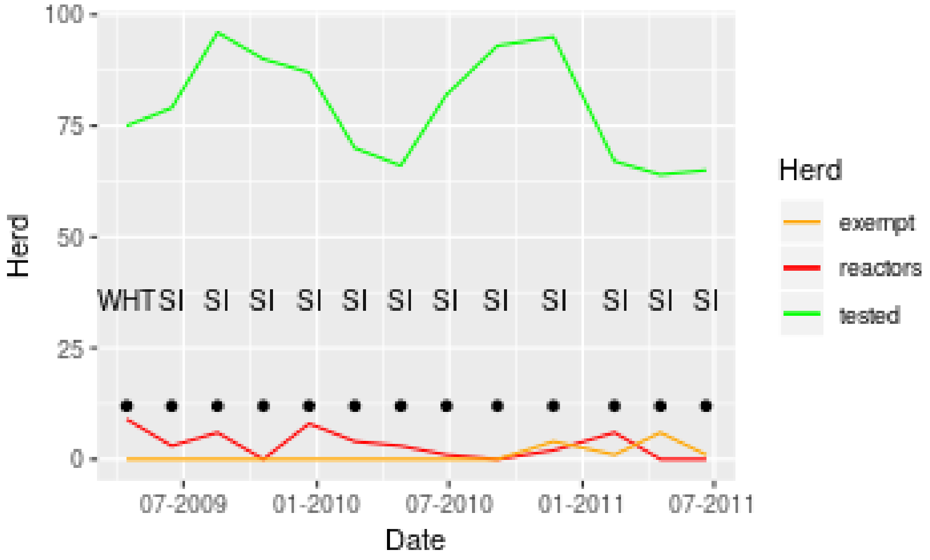


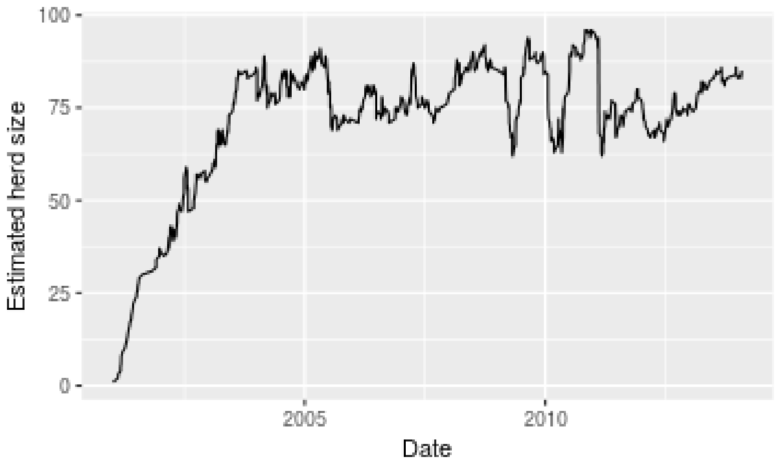
Figure C.11: As per Figure C.10, but for the MBP-IBIS algorithm. Each row represents a separate run of the algorithm.

C.2 Incident one

The first (real) herd analysis is from a farm located in Staffordshire with a head count fluctuating between about seventy and one hundred cattle.



(a) BTB surveillance data from *VetNet*.



(b) herd size estimated from *CTS* movement data.

Figure C.12: The first epidemic is based on an incident recorded in the *VetNet* database at a farm in Staffordshire which lasted about two years following a breakdown triggered by a routine whole herd test in 2009. Migration data is based on corresponding data as recorded in *CTS*. The latter is also used to corroborate the approximate herd size at the beginning of each modelled scenario.

SEIR model parameter estimates

The first set of results (for each of three scenarios) are based on the SEIR variant of the test-determined removal model.

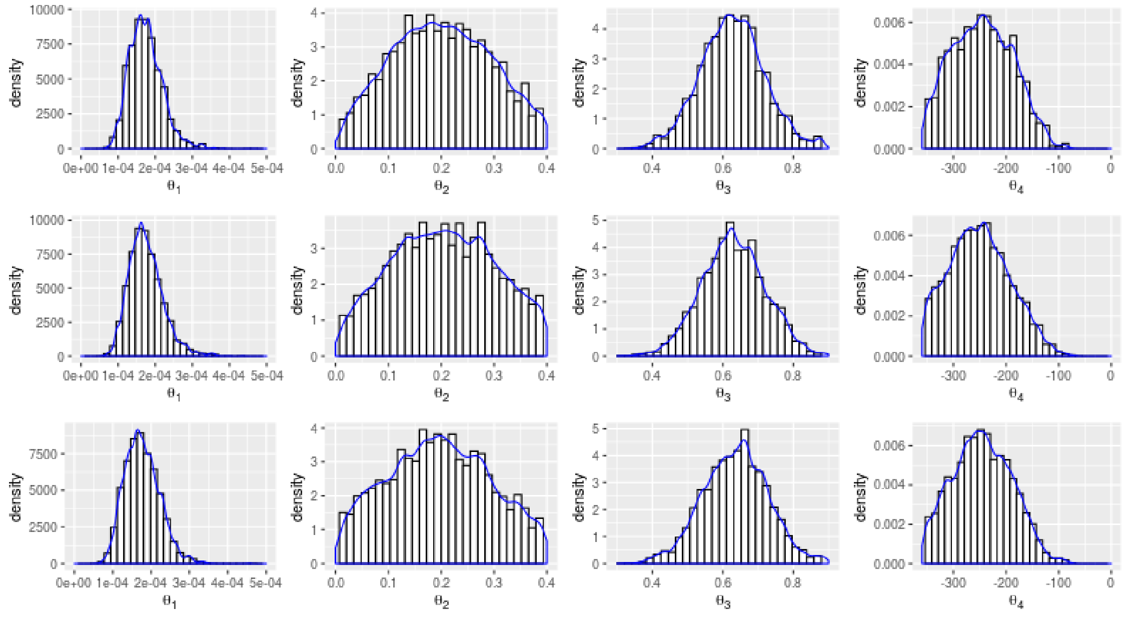


Figure C.13: Marginal model parameter distributions for the first VetNet data set fitted to the *SEIR* model using the SMC^2 algorithm with $N = \{6000, 8000, 10000\}$ (independent proposals). Left to right: contact rate $S \rightarrow E$; progression rate $E \rightarrow I$; SICCT test sensitivity; and onset of infectiousness in the first affected individual.

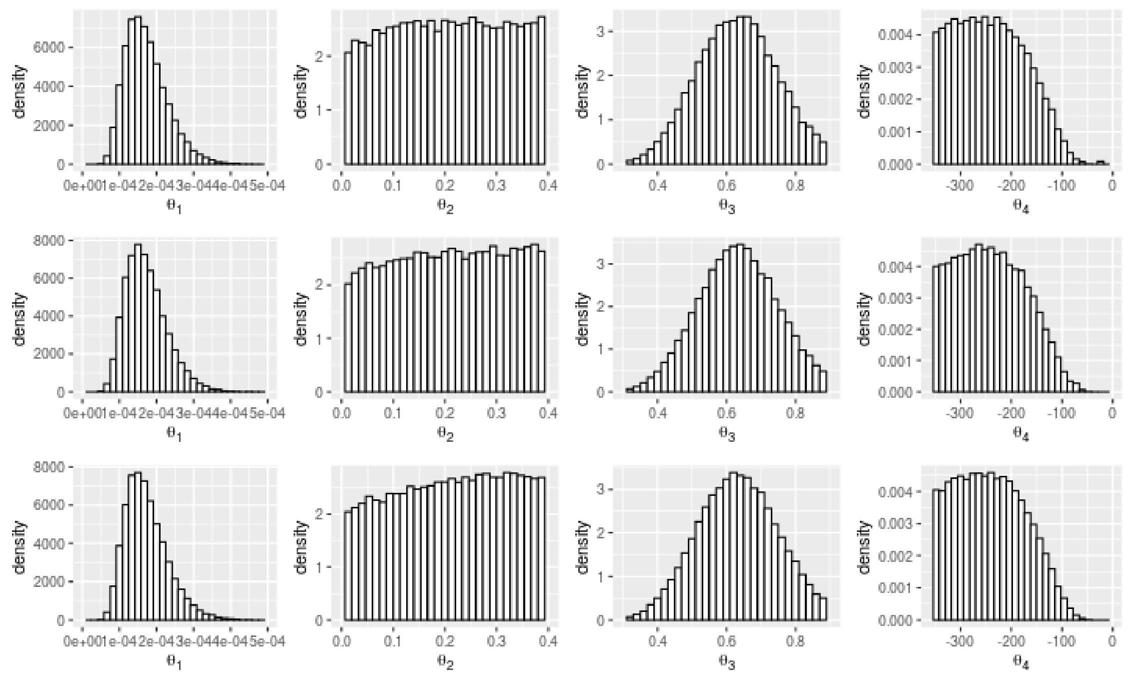
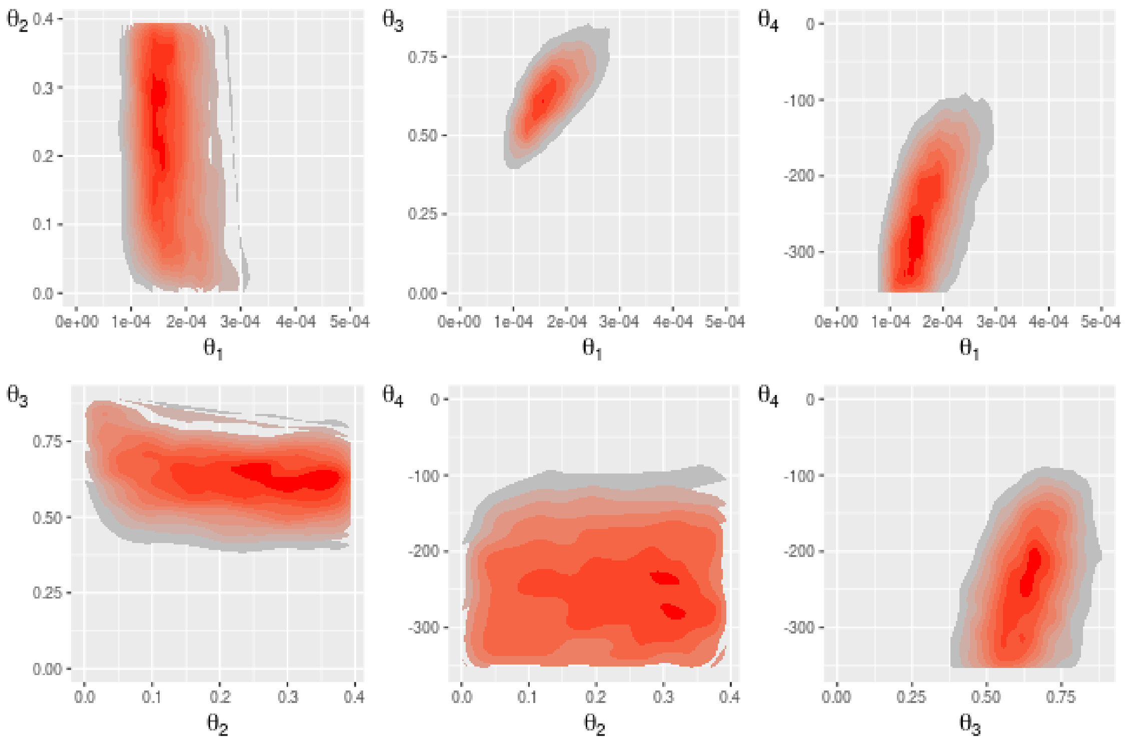


Figure C.14: Marginal model parameter distributions for the first VetNet data set fitted to the *SEIR* model using the ARQ MCMC algorithm with $\Gamma_L = \{1, 3, 7\}$ ($\Gamma_R = 30$). The algorithm is much less efficient than SMC^2 but tends to produce consistent results.



(a) Twin parameter marginal densities for the *SEIR* model analysis (ARQ MCMC).

```
## SMC2 analysis:
##   Theta      iMu     iSD
## 1     1  1.8e-04 4.5e-05
## 2     2  2.0e-01 9.7e-02
## 3     3  6.4e-01 9.2e-02
## 4     4 -2.5e+02 5.6e+01
```

```
## ARQMCMC analysis:
##   Theta      iMu     iSD      rMu     rSD SRE (95%)
## 1     1  1.8e-04 5.0e-05  1.8e-04 5.7e-05   1     1
## 2     2  2.1e-01 1.1e-01  2.1e-01 1.1e-01   1     1
## 3     3  6.3e-01 1.1e-01  6.4e-01 1.1e-01   1     1
## 4     4 -2.4e+02 6.7e+01 -2.4e+02 7.0e+01   1     1
```

Figure C.16: Inference summaries for the two *SEIR* analyses (incident one). The model parameters labeled from one to four are as follows: contact parameter $S \rightarrow E$; progression rate $E \rightarrow I$; SICCT test sensitivity; and onset of infectiousness in the first affected individual, also denoted in the main text as t_0.

SETIR model parameter estimates

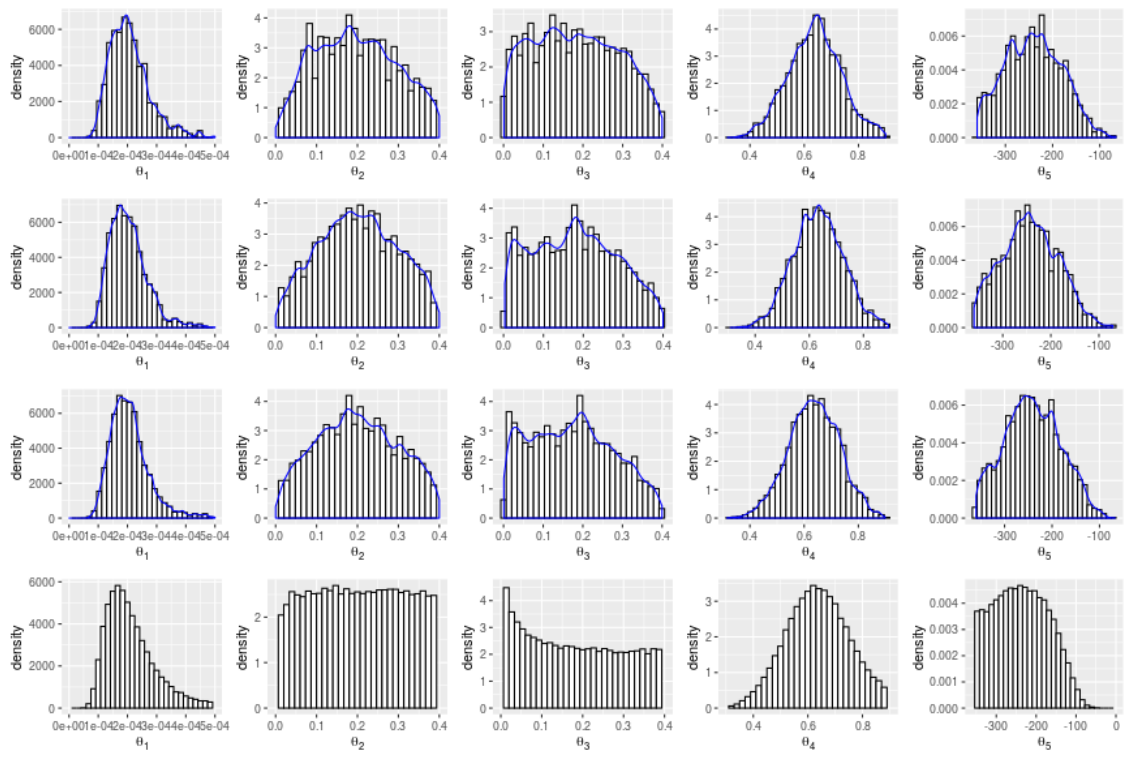


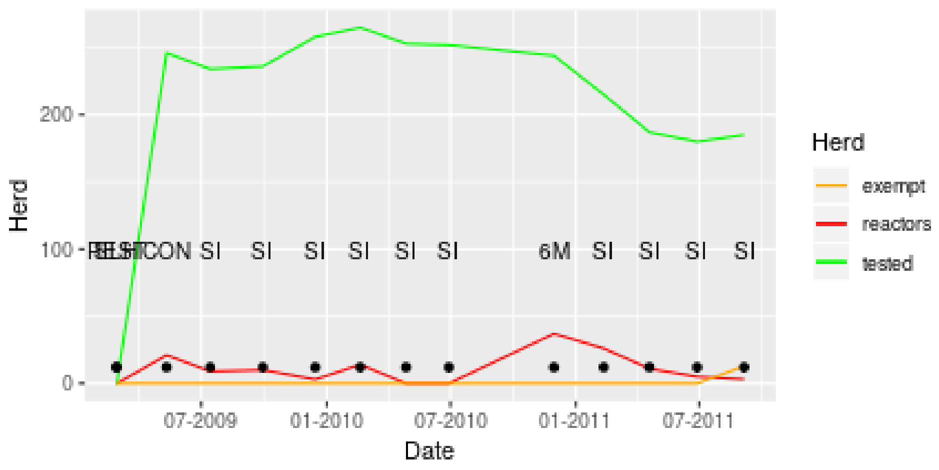
Figure C.17: Marginal model parameter distributions for the first VetNet data set fitted to the *SETIR* model using the SMC^2 algorithm with $N = \{6000, 8000, 10000\}$ (independent proposals) and ARQ MCMC.

```
## ARQMCMC analysis:
##   Theta      iMu     iSD      rMu     rSD SRE (95%)
## 1     1  2.0e-04 6.4e-05  2.2e-04 8.3e-05   1     1
## 2     2  2.1e-01 1.1e-01  2.0e-01 1.1e-01   1     1
## 3     3  1.8e-01 1.1e-01  1.8e-01 1.2e-01   1     1
## 4     4  6.3e-01 9.1e-02  6.4e-01 1.1e-01   1     1
## 5     5 -2.4e+02 6.2e+01 -2.4e+02 7.0e+01   1     1
```

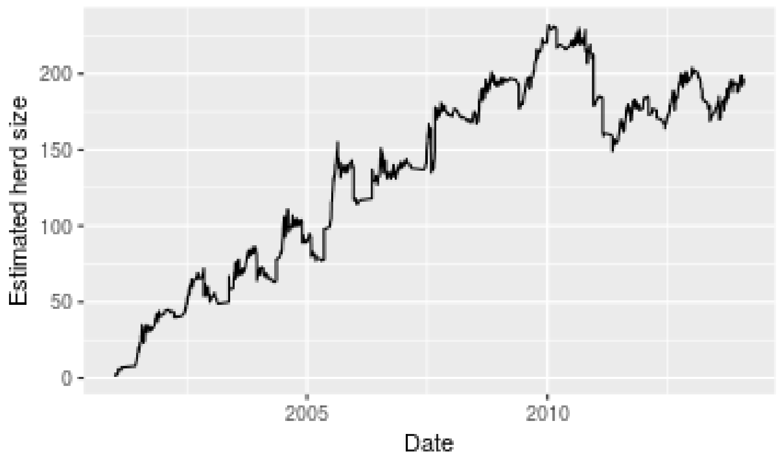
```
## SMC2 analysis:
##   Theta      iMu     iSD
## 1     1  2.1e-04 6.6e-05
## 2     2  2.0e-01 9.9e-02
## 3     3  1.7e-01 1.0e-01
## 4     4  6.4e-01 9.4e-02
## 5     5 -2.4e+02 5.9e+01
```

Figure C.18: Inference summaries for the two *SETIR* analyses (incident one). The model parameters, labeled from one to five, are as follows: contact parameter $S \to E$; progression rate $E \to T$; progression rate $T \to I$; SICCT test sensitivity; and onset of infectiousness in the first affected individual.

C.3 Incident two



(a) BTB surveillance data from *VetNet*.



(b) herd size estimated from *CTS* movement data.

Figure C.19: The second data set is based on two incidents recorded in the VetNet database which occured about eighteen months apart at another farm in Staffordshire. The first was triggered by 'contiguous' herd testing following a slaughterhouse case and the latter by the six month follow up test following the first incident.

```
## ARQMCMC analysis:
##   Theta      iMu     iSD      rMu     rSD SRE (95%)
## 1     1  8.1e-05 1.3e-05  8.3e-05 2.5e-05   1     1
## 2     2  2.7e-01 8.6e-02  2.6e-01 8.9e-02   1     1
## 3     3  8.1e-01 5.2e-02  8.1e-01 6.6e-02   1     1
## 4     4 -1.7e+02 6.3e+01 -1.8e+02 6.5e+01   1     1
```

```
## SMC2 analysis:
##   Theta      iMu     iSD
## 1     1  8.1e-05 1.2e-05
## 2     2  2.6e-01 7.2e-02
## 3     3  8.1e-01 4.7e-02
## 4     4 -1.7e+02 4.6e+01
```

Figure C.20: Inference summaries for the two *SEIR* analyses (incident two).

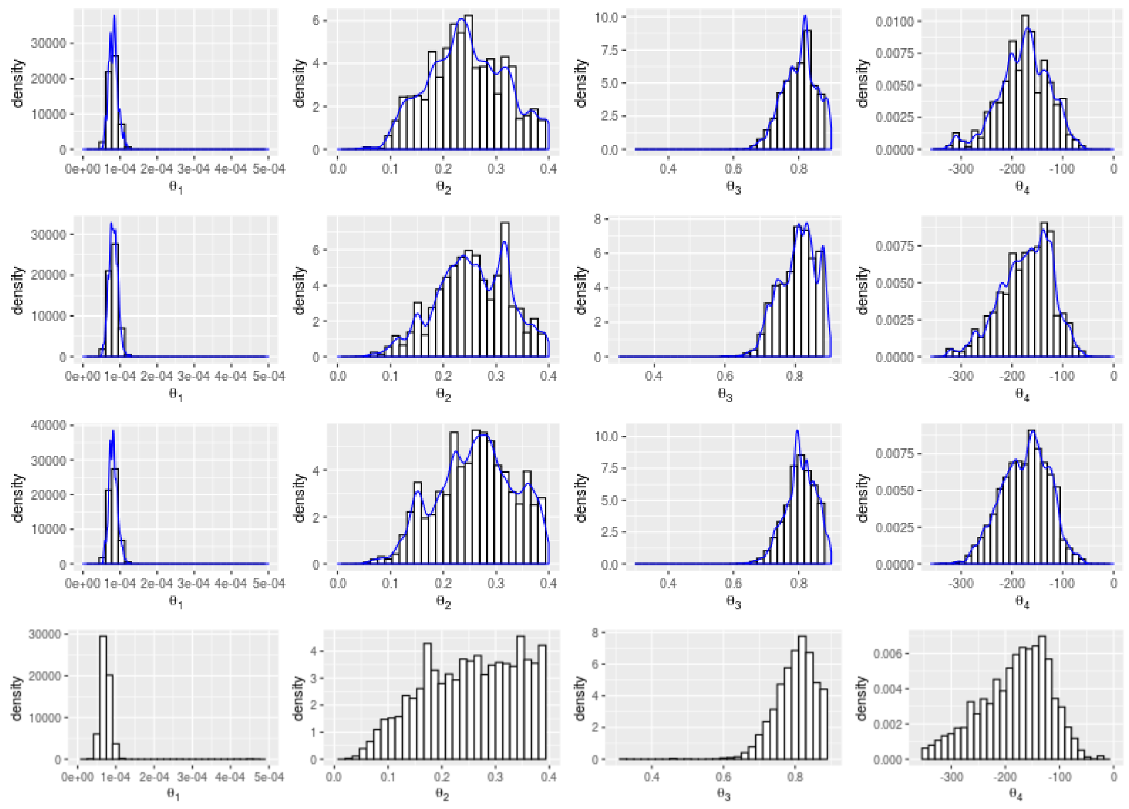


Figure C.21: Marginal model parameter distributions for the second VetNet data set fitted to the *SEIR* model using the SMC^2 algorithm with $N = \{6000, 8000, 10000\}$ (independent proposals) and ARQ MCMC ($\Gamma_L = 7$). Left to right: contact rate $S \to E$; progression rate $E \to I$; SICCT test sensitivity; onset of infectiousness in the first affected individual.

```
## ARQMCMC analysis:
##   Theta      iMu     iSD      rMu     rSD SRE (95%)
## 1     1  1.8e-04 1.1e-04  1.8e-04 1.1e-04 1.1   1.3
## 2     2  2.7e-01 8.3e-02  2.7e-01 8.9e-02 1.0   1.0
## 3     3  1.1e-01 1.2e-01  1.1e-01 1.2e-01 1.1   1.4
## 4     4  8.2e-01 4.3e-02  8.0e-01 7.6e-02 1.0   1.0
## 5     5 -1.3e+02 4.5e+01 -1.5e+02 6.2e+01 1.0   1.1
```

```
## SMC2 analysis:
##   Theta      iMu     iSD
## 1     1  1.3e-04 7.8e-05
## 2     2  2.5e-01 8.3e-02
## 3     3  1.5e-01 1.2e-01
## 4     4  8.1e-01 5.2e-02
## 5     5 -1.6e+02 4.8e+01
```

Figure C.22: Inference summaries for the two *SETIR* analyses (incident two). The model parameters, labeled from one to five, are as follows: contact parameter $S \rightarrow E$; progression rate $E \rightarrow T$; progression rate $T \rightarrow I$; SICCT test sensitivity; and onset of infectiousness in the first affected individual.

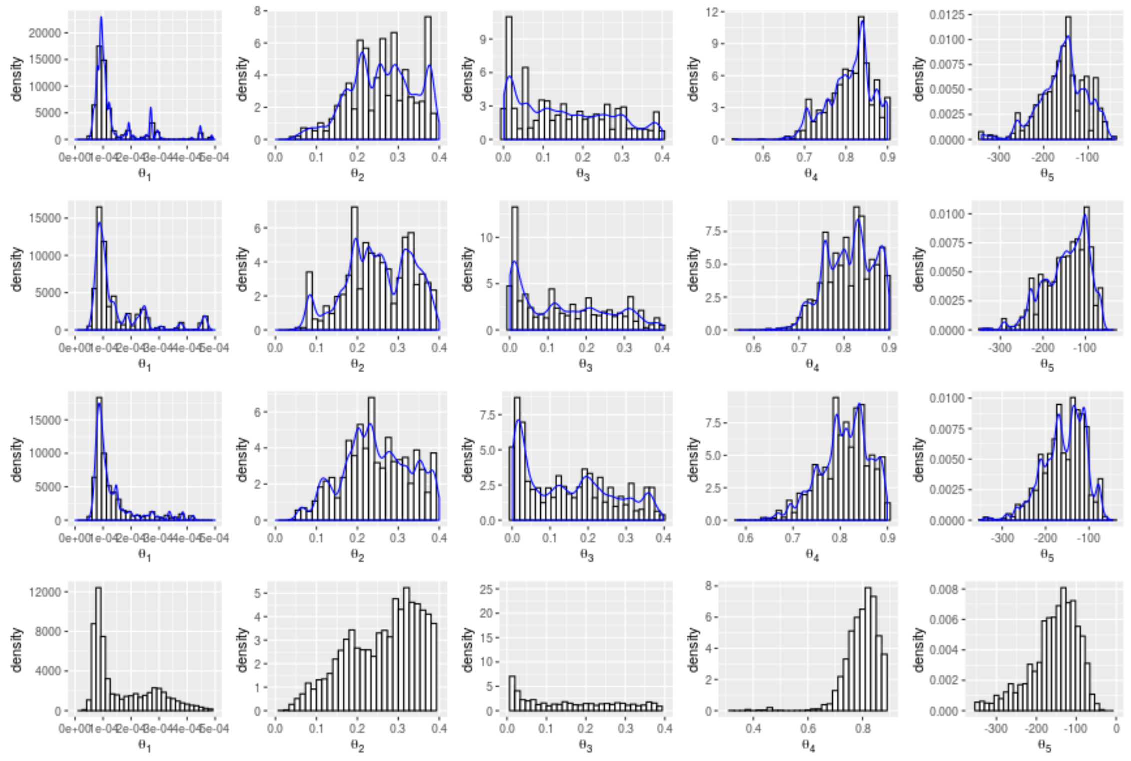
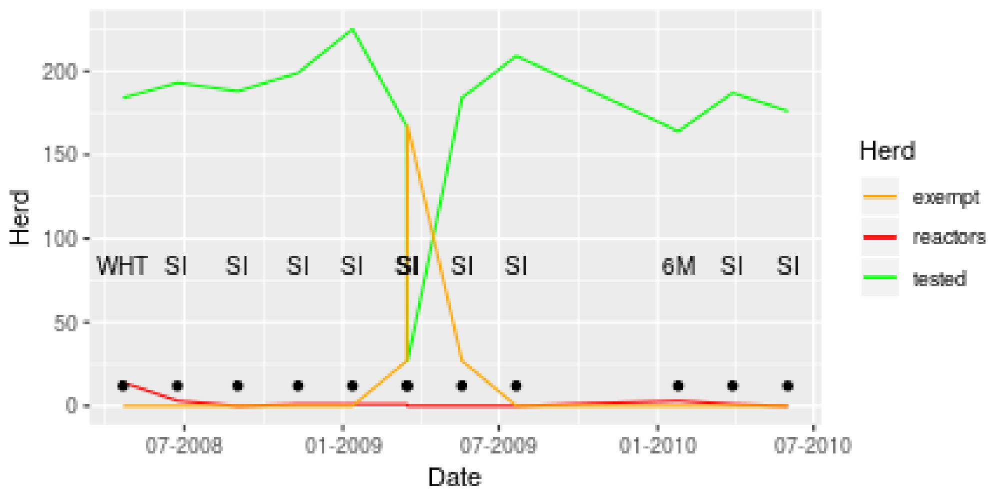
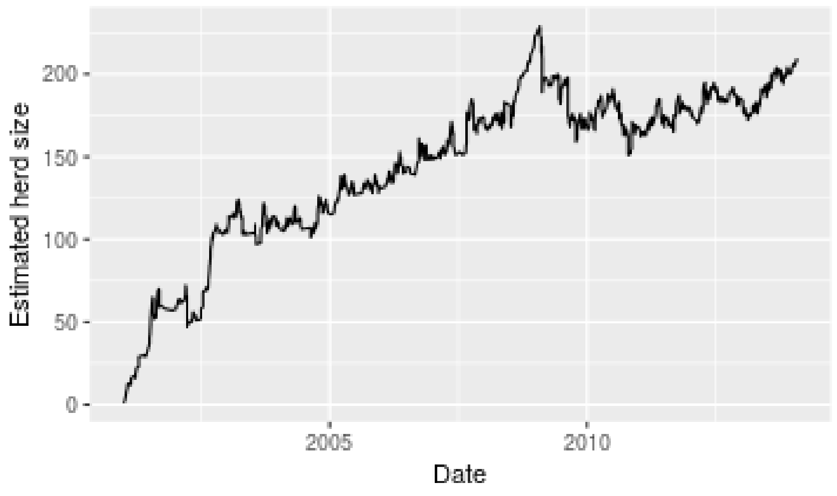


Figure C.23: Marginal model parameter distributions for the second VetNet data set fitted to the *SETIR* model using the SMC^2 algorithm with $N = \{6000, 8000, 10000\}$ (independent proposals) and ARQ MCMC ($\Gamma_L = 1$). Left to right: contact $S \rightarrow E$; progression $E \rightarrow T$; progression $T \rightarrow I$; SICCT test sensitivity; and onset of infectiousness in the first affected individual.

C.4 Incident three



(a) BTB surveillance data from *VetNet*.



(b) herd size estimated from *CTS* movement data.

Figure C.24: The final data set is based on two incidents that occurred at a farm in Shropshire. The first was triggered by a routine whole herd test and the second by the six monthly follow up test related to the first incident.

```
## ARQMCMC analysis:
##   Theta      iMu     iSD      rMu     rSD SRE (95%)
## 1     1  5.2e-05 1.7e-05  5.2e-05 2.0e-05   1     1
## 2     2  1.3e-01 1.2e-01  1.3e-01 1.2e-01   1     1
## 3     3  7.2e-01 1.2e-01  7.2e-01 1.2e-01   1     1
## 4     4 -3.0e+02 4.5e+01 -3.0e+02 4.7e+01   1     1
```

```
## SMC2 analysis:
##   Theta      iMu     iSD
## 1     1  5.0e-05 1.4e-05
## 2     2  1.6e-01 1.1e-01
## 3     3  7.0e-01 1.0e-01
## 4     4 -3.0e+02 4.0e+01
```

Figure C.25: Inference summaries for the two *SEIR* analyses (incident three).

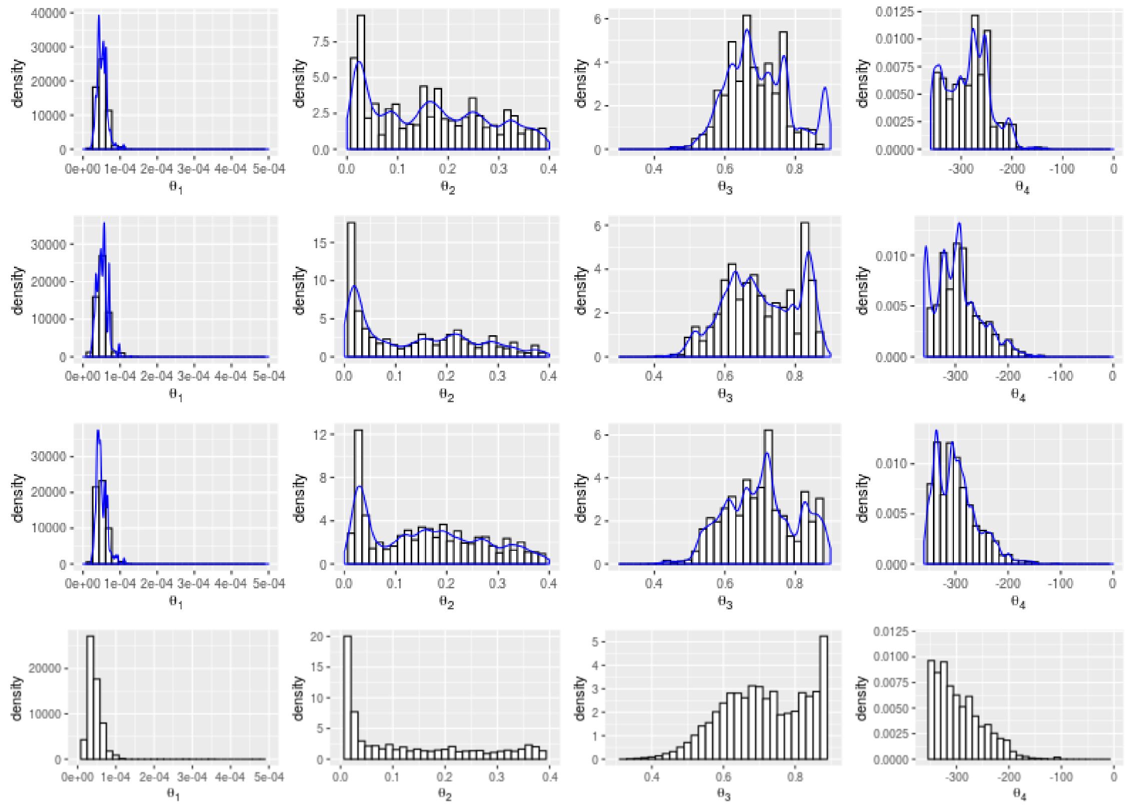


Figure C.26: Marginal model parameter distributions for the third VetNet data set fitted to the *SEIR* model using the SMC^2 algorithm with $N = \{6000, 8000, 10000\}$ (independent proposals) and ARQ MCMC ($\Gamma_L = 3$). Left to right: contact rate $S \to E$; progression rate $E \to I$; SICCT test sensitivity; onset of infectiousness in the first affected individual.

```
## ARQMCMC analysis:                             ## SMC2 analysis:
##   Theta      iMu     iSD      rMu     rSD ##   Theta      iMu     iSD
## 1     1  6.7e-05 3.0e-05  6.9e-05 4.1e-05 ## 1     1  6.6e-05 4.9e-05
## 2     2  8.0e-02 1.0e-01  1.3e-01 1.2e-01 ## 2     2  1.6e-01 1.1e-01
## 3     3  1.6e-01 1.2e-01  1.5e-01 1.2e-01 ## 3     3  1.8e-01 1.1e-01
## 4     4  7.7e-01 1.1e-01  7.2e-01 1.2e-01 ## 4     4  7.0e-01 9.5e-02
## 5     5 -3.1e+02 3.8e+01 -2.9e+02 5.3e+01 ## 5     5 -2.9e+02 4.6e+01
```

Figure C.27: Inference summaries for the two *SETIR* analyses (incident two). The model parameters, labeled from one to five, are as follows: contact parameter $S \rightarrow E$; progression rate $E \rightarrow T$; progression rate $T \rightarrow I$; SICCT test sensitivity; and onset of infectiousness in the first affected individual.

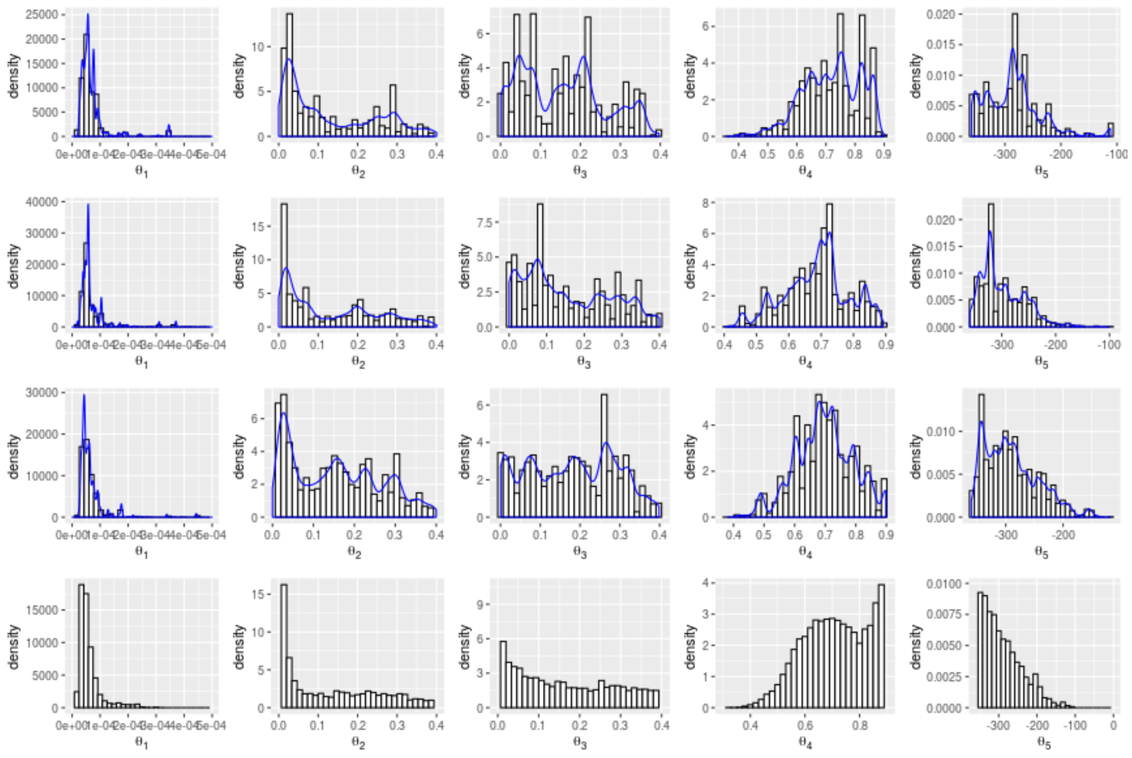
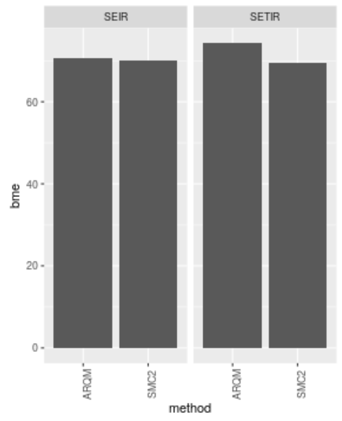
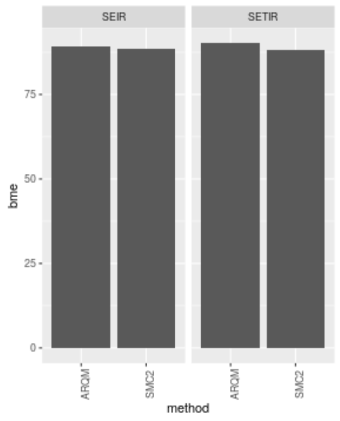


Figure C.28: Marginal model parameter distributions for the second VetNet data set fitted to the *SETIR* model using the SMC^2 algorithm with $N = \{6000, 8000, 10000\}$ (independent proposals) and ARQ MCMC ($\Gamma_L = 1$). Left to right: contact $S \to E$; progression $E \to T$; progression $T \to I$; SICCT test sensitivity; and onset of infectiousness in the first affected individual.

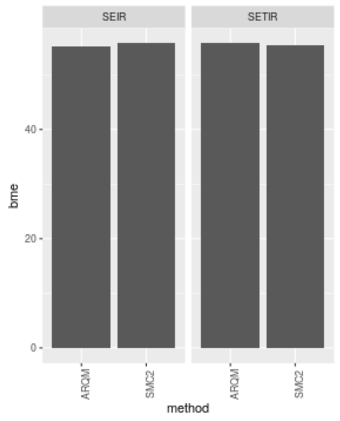
C.5 Model evidence



(a) Incident one.



(b) Incident two.



(c) Incident three.

Figure C.29: Bayesian model evidence for the individual herd scenarios presented in this chapter. Independent estimates were obtained using the SMC^2 and *ARQ MCMC* algorithms, as described in Chapters 2 and 3 respectively.

Appendix D

Detailed results for meta-analysis of herd outbreaks for BTB

D.1 Model comparison

Yorkshire herds

The first subset is comprised of half a dozen herds drawn from the Yorkshire region. Frequency-dependent models performed reasonably well for this subset. The exception was a case that polarised the reinfection models, with the super reinfection model [marginally] outperforming all others. Those results are given in Tables D.1 and D.2.

East of England herds

The simple reinfection model bucked its own trend by generally outperforming other models for the East of England cohort. These cases provide among the strongest (but still inconclusive) evidence for carrying the reinfection concept forward in future attempts to resolve longstanding scientific questions concerning the nature and pathology of BTB. Those results are given in Table D.3.

Scottish herds

The final and largest subgroup was drawn from Scotland. Those results are given in Tables D.4 and D.5. This was perhaps the most 'mixed' group of all, including with respect to frequency- versus density-dependent models.

Herd	**Model**	**E(BME)**	**Var(BME)**	**BF**
7397	TDR-SETIR (Freq. Dep)	122.8	29.8	17.1
7397	TDR-SEIR (Freq. Dep)	128.5	13.7	1.0
7397	SRI-SEIR	130.4	36.0	0.4
7397	TDR-SETIR (Dens. Dep)	133.2	86.0	0.1
7397	TDR-SEIR (Dens. Dep)	138.5	594.4	0.0
7397	TDR-SEYZ	179.7	4.6	0.0
7454	TDR-SETIR (Freq. Dep)	64.8	11.1	1.3
7454	TDR-SEIR (Freq. Dep)	65.3	14.8	1.0
7454	TDR-SETIR (Dens. Dep)	65.4	7.0	1.0
7454	TDR-SEIR (Dens. Dep)	65.6	9.9	0.9
7454	SRI-SEIR	66.5	0.3	0.6
7454	TDR-SEYZ	66.9	0.0	0.5
7456	TDR-SEYZ	109.4	83.1	26.6
7456	TDR-SETIR (Dens. Dep)	113.3	5.2	3.8
7456	TDR-SETIR (Freq. Dep)	114.9	1.9	1.7
7456	TDR-SEIR (Freq. Dep)	115.9	3.3	1.0
7456	TDR-SEIR (Dens. Dep)	118.1	7.8	0.3
7456	SRI-SEIR	118.4	7.6	0.3

Table D.1: First three (of six) herd model comparison scenarios drawn from the Yorkshire group.

It was hoped that the relatively low incidence of BTB in Scotland generally might isolate any extraneous factors that were unaccounted for in the dynamics of the model, from the within-herd dynamics they were intended to evaluate. However the approach still yielded no clear findings. These matters are addressed further in the discussion at the end of the chapter.

For now we turn out attention (in the next sub-section) to the parameter inference results. They are perhaps not as interesting due to the fact that no single model stands out as especially worthy. However they are included for completeness, and also serve a role in evaluating both the consistency of algorithms, and correlation between important parameters within the model.

Herd	**Model**	**E(BME)**	**Var(BME)**	**BF**
7457	TDR-SETIR (Freq. Dep)	68.5	0.2	1.3
7457	TDR-SETIR (Dens. Dep)	68.7	2.8	1.3
7457	TDR-SEIR (Freq. Dep)	69.1	0.1	1.0
7457	TDR-SEIR (Dens. Dep)	70.7	12.3	0.5
7457	SRI-SEIR	72.6	6.2	0.2
7457	TDR-SEYZ	110.4	462.3	0.0
7485	TDR-SEYZ	31.1	0.0	2.9
7485	TDR-SETIR (Dens. Dep)	32.2	0.2	1.7
7485	TDR-SETIR (Freq. Dep)	32.6	0.0	1.4
7485	TDR-SEIR (Dens. Dep)	33.2	0.0	1.0
7485	TDR-SEIR (Freq. Dep)	33.2	0.0	1.0
7485	SRI-SEIR	34.9	0.0	0.4
7675	TDR-SETIR (Freq. Dep)	50.9	0.9	1.3
7675	TDR-SEIR (Freq. Dep)	51.5	0.3	1.0
7675	TDR-SETIR (Dens. Dep)	51.6	1.9	1.0
7675	TDR-SEIR (Dens. Dep)	53.6	8.8	0.4
7675	SRI-SEIR	55.2	1.1	0.2
7675	TDR-SEYZ	80.0	12.3	0.0

Table D.2: Second three (of six) herd model comparison scenarios drawn from the Yorkshire group.

Herd	Model	E(BME)	Var(BME)	BF
7772	SRI-SEIR	109.4	5.2	265,155.4
7772	TDR-SETIR (Freq. Dep)	121.9	456.3	516.8
7772	TDR-SETIR (Dens. Dep)	128.1	981.3	24.0
7772	TDR-SEIR (Freq. Dep)	134.4	1,271.0	1.0
7772	TDR-SEYZ	136.7	14.9	0.3
7772	TDR-SEIR (Dens. Dep)	150.0	3,154.4	0.0
7775	SRI-SEIR	37.0	0.0	3.8
7775	TDR-SETIR (Freq. Dep)	39.4	9.1	1.2
7775	TDR-SEIR (Freq. Dep)	39.7	8.5	1.0
7775	TDR-SEIR (Dens. Dep)	46.6	29.2	0.0
7775	TDR-SETIR (Dens. Dep)	48.2	27.7	0.0
7775	TDR-SEYZ	91.7	7.1	0.0
7777	TDR-SEYZ	69.3	57.5	7,221.6
7777	SRI-SEIR	84.3	4.8	4.0
7777	TDR-SEIR (Freq. Dep)	87.1	48.3	1.0
7777	TDR-SETIR (Freq. Dep)	87.2	51.0	0.9
7777	TDR-SETIR (Dens. Dep)	90.1	58.7	0.2
7777	TDR-SEIR (Dens. Dep)	91.1	58.6	0.1
7805	SRI-SEIR	73.5	0.1	4.7
7805	TDR-SEYZ	73.9	7.2	3.9
7805	TDR-SETIR (Freq. Dep)	76.2	18.4	1.3
7805	TDR-SETIR (Dens. Dep)	76.3	20.1	1.2
7805	TDR-SEIR (Freq. Dep)	76.6	16.6	1.0
7805	TDR-SEIR (Dens. Dep)	76.9	16.7	0.9

Table D.3: Model comparison scenarios for the East of England group.

Herd	**Model**	**E(BME)**	**Var(BME)**	**BF**
31336	TDR-SETIR (Dens. Dep)	55.9	0.7	2.8
31336	TDR-SETIR (Freq. Dep)	56.1	0.0	2.5
31336	TDR-SEIR (Dens. Dep)	57.0	0.5	1.6
31336	TDR-SEIR (Freq. Dep)	58.0	3.6	1.0
31336	SRI-SEIR	79.9	1,198.8	0.0
31336	TDR-SEYZ	82.2	11.6	0.0
31376	TDR-SETIR (Freq. Dep)	56.2	0.0	1.1
31376	TDR-SEIR (Freq. Dep)	56.4	0.0	1.0
31376	TDR-SETIR (Dens. Dep)	58.1	0.1	0.4
31376	TDR-SEIR (Dens. Dep)	58.7	0.2	0.3
31376	SRI-SEIR	60.2	0.1	0.2
31376	TDR-SEYZ	69.7	50.1	0.0
31472	TDR-SEIR (Freq. Dep)	75.3	47.5	1.0
31472	TDR-SETIR (Freq. Dep)	75.6	49.2	0.9
31472	TDR-SEIR (Dens. Dep)	80.0	57.7	0.1
31472	TDR-SETIR (Dens. Dep)	80.2	55.1	0.1
31472	SRI-SEIR	87.6	0.0	0.0
31472	TDR-SEYZ	108.7	12.2	0.0
31495	SRI-SEIR	86.4	0.6	1.8
31495	TDR-SEIR (Freq. Dep)	87.5	4.5	1.0
31495	TDR-SEIR (Dens. Dep)	96.5	36.9	0.0
31495	TDR-SETIR (Freq. Dep)	96.8	12.4	0.0
31495	TDR-SETIR (Dens. Dep)	125.1	297.6	0.0

Table D.4: Model comparison scenarios for the first four (of eight) herds comprising the Scottish group.

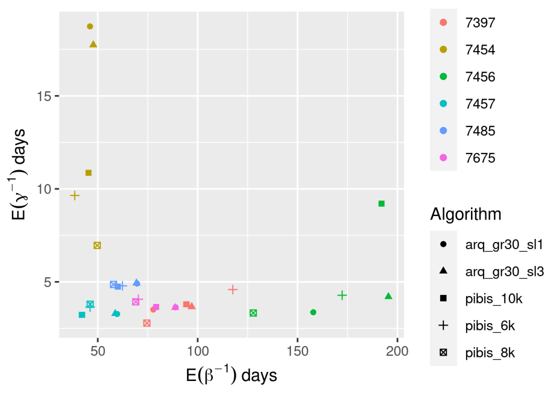
Herd	**Model**	**E(BME)**	**Var(BME)**	**BF**
31834	TDR-SETIR (Freq. Dep)	42.2	15.6	1.7
31834	TDR-SETIR (Dens. Dep)	42.8	8.2	1.3
31834	TDR-SEIR (Freq. Dep)	43.3	11.7	1.0
31834	TDR-SEIR (Dens. Dep)	44.8	10.0	0.5
31834	SRI-SEIR	45.5	0.7	0.3
31834	TDR-SEYZ	78.8	2.9	0.0
31906	TDR-SEYZ	340.3	13,403.3	4.75E+159
31906	TDR-SETIR (Dens. Dep)	908.3	301,648.6	2.18E+036
31906	TDR-SEIR (Freq. Dep)	1,075.6	342,619.7	1.0
31906	SRI-SEIR	1,233.2	2,293.8	0.0
31906	TDR-SEIR (Dens. Dep)	1,380.9	71,885.5	0.0
31906	TDR-SETIR (Freq. Dep)	1,742.0	220,867.8	0.0
32000	TDR-SEYZ	42.0	7.0	29.0
32000	TDR-SEIR (Dens. Dep)	46.6	2.5	2.8
32000	TDR-SETIR (Dens. Dep)	46.8	1.2	2.6
32000	SRI-SEIR	47.7	0.1	1.7
32000	TDR-SEIR (Freq. Dep)	48.7	0.1	1.0
32000	TDR-SETIR (Freq. Dep)	48.8	0.0	0.9
32012	TDR-SEYZ	96.2	287.1	274,519.1
32012	TDR-SETIR (Dens. Dep)	119.1	1.8	3.0
32012	TDR-SEIR (Dens. Dep)	120.1	2.1	1.8
32012	SRI-SEIR	120.1	3.6	1.8
32012	TDR-SETIR (Freq. Dep)	121.1	0.1	1.1
32012	TDR-SEIR (Freq. Dep)	121.3	0.4	1.0

Table D.5: Model comparison scenarios for the second four (of eight) herds comprising the Scottish group.

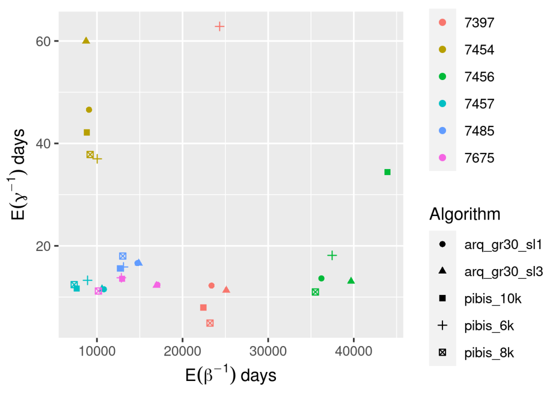
D.2 Selected parameter estimates

Here we report selected parameter estimates grouped by type across regions. The complete model parameter sets are given in the following section (and also reported in tables.) As with the results already reported, several algorithm configurations (and runs) were used to compute parameter estimates for each of the three herd groups.

Estimates for the latent period

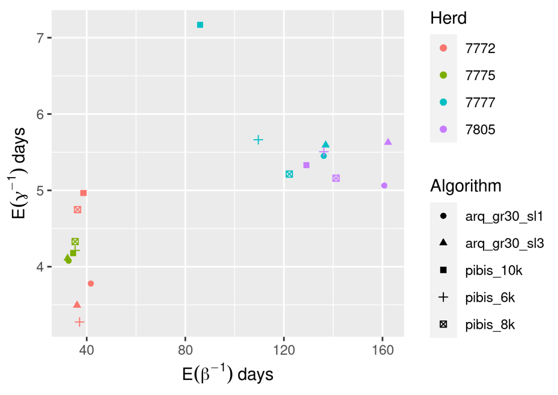


(a) Frequency-dependent

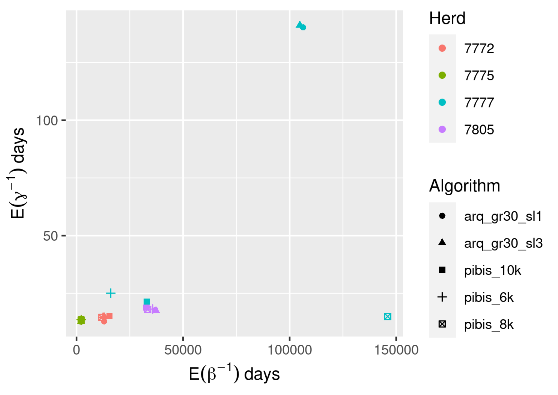


(b) Density-dependent

Figure D.1: SEIR model estimates for Yorkshire

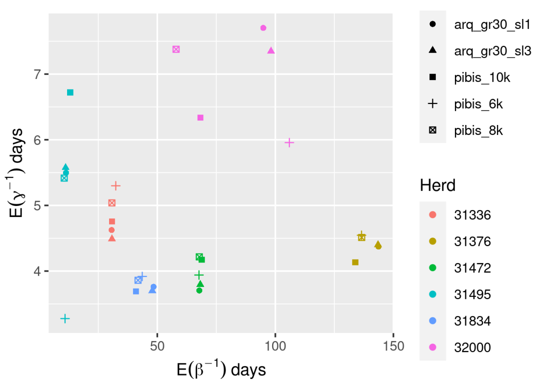


(a) Frequency-dependent

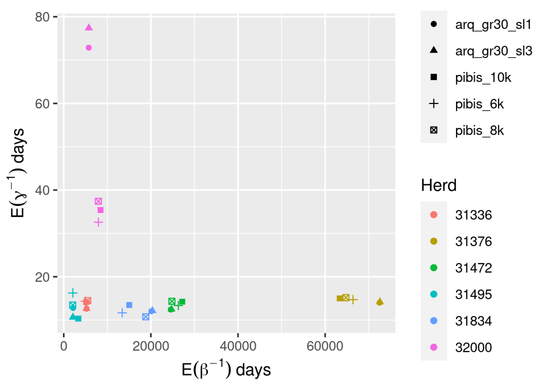


(b) Density-dependent

Figure D.2: SEIR model estimates for East of England

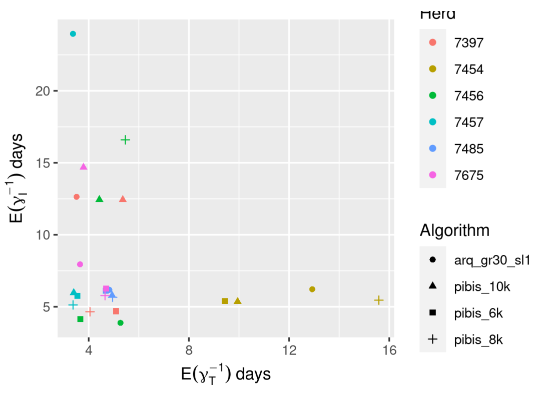


(a) Frequency-dependent

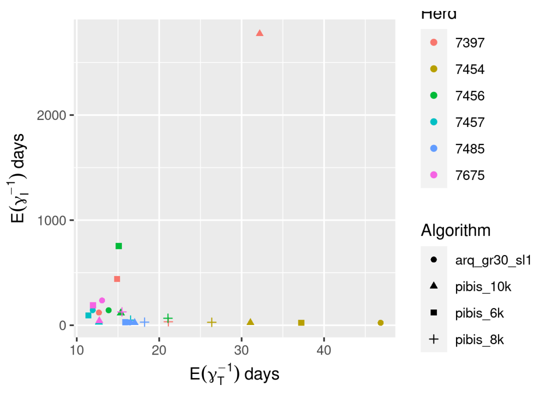


(b) Density-dependent

Figure D.3: SEIR model estimates for Scotland

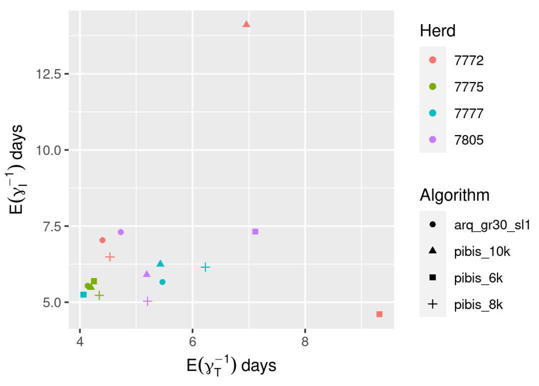


(a) Frequency-dependent

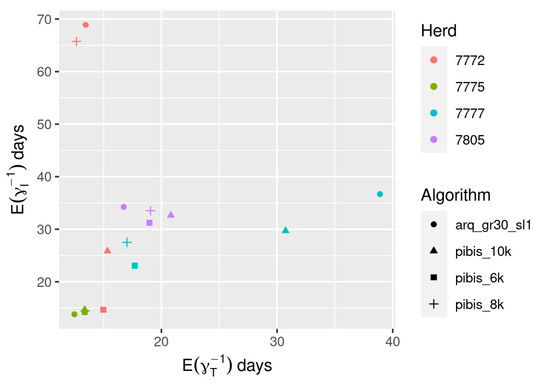


(b) Density-dependent

Figure D.4: SETIR model estimates for Yorkshire

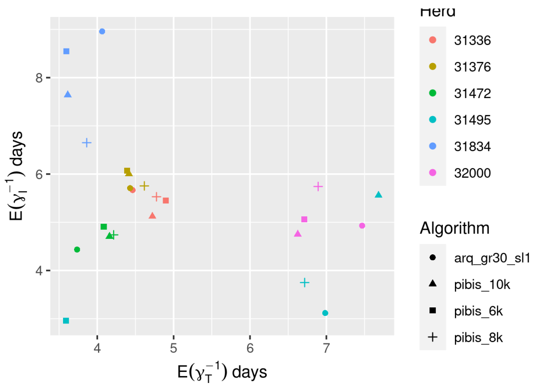


(a) Frequency-dependent

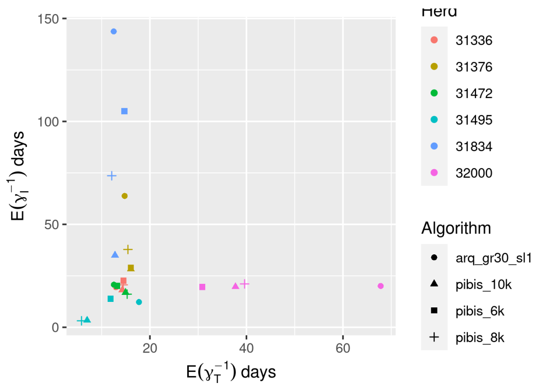


(b) Density-dependent

Figure D.5: SETIR model estimates for East of England

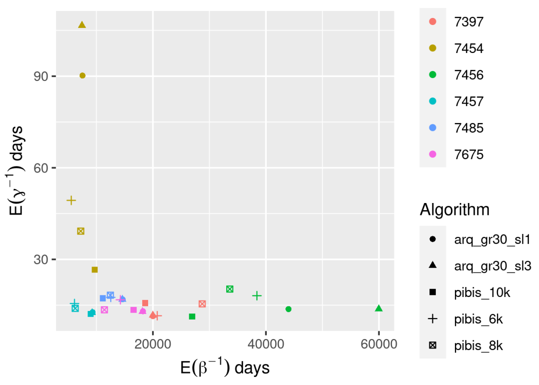


(a) Frequency-dependent

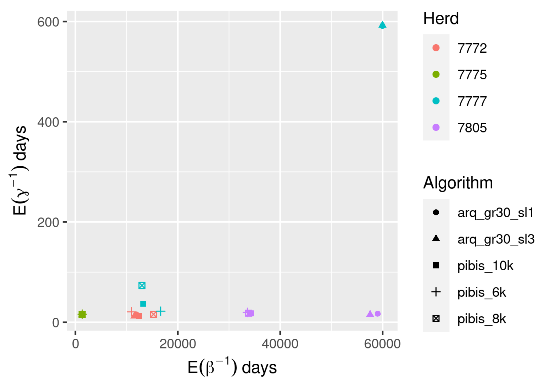


(b) Density-dependent

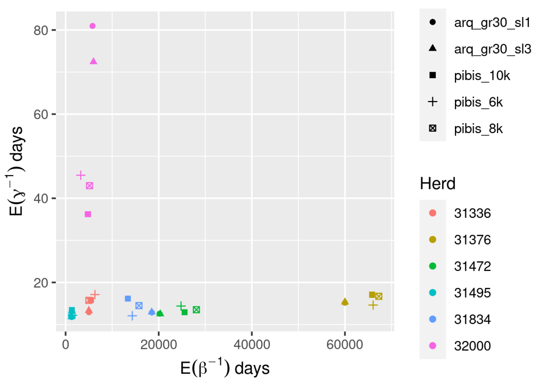
Figure D.6: SETIR model estimates for Scotland



(a) Yorkshire



(b) East of England



(c) Scotland

Figure D.7: Simple reinfection model estimates.

Frequency dependent TDR-SEIR model

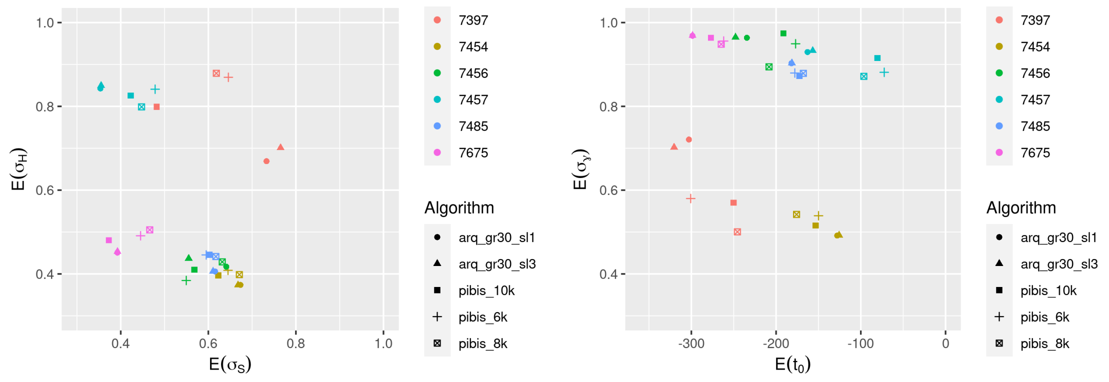


Figure D.8: Yorkshire

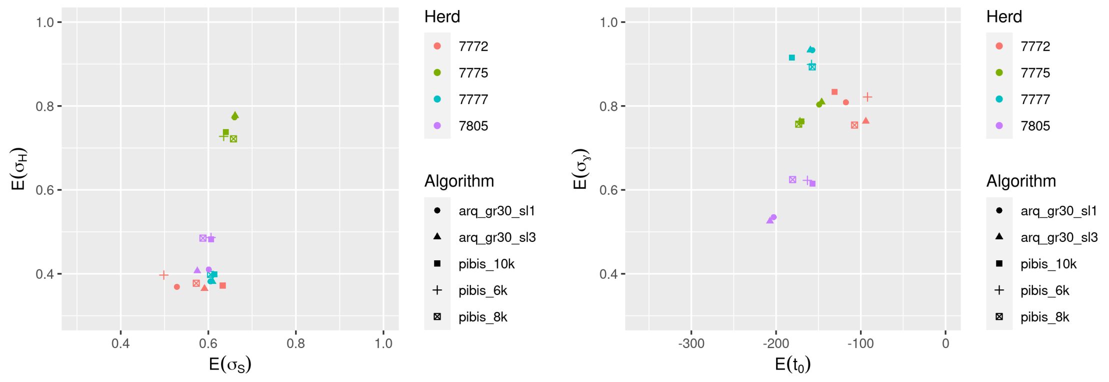


Figure D.9: East of England

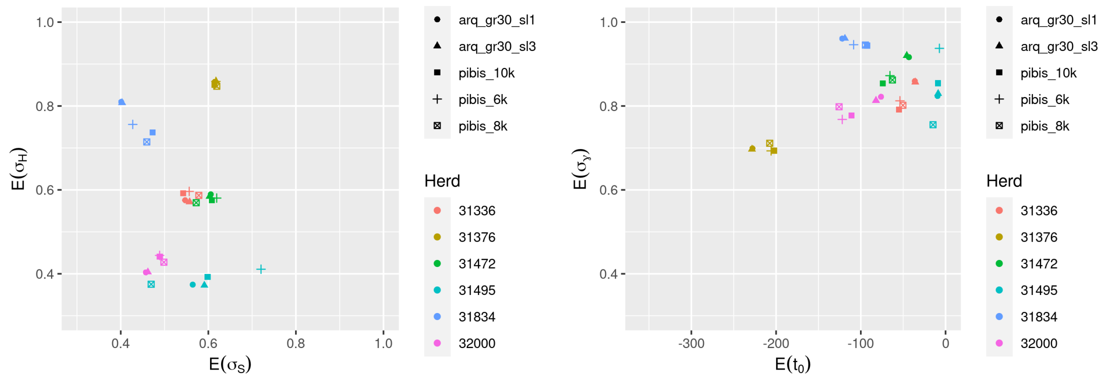


Figure D.10: Scotland

Density dependent TDR-SEIR model

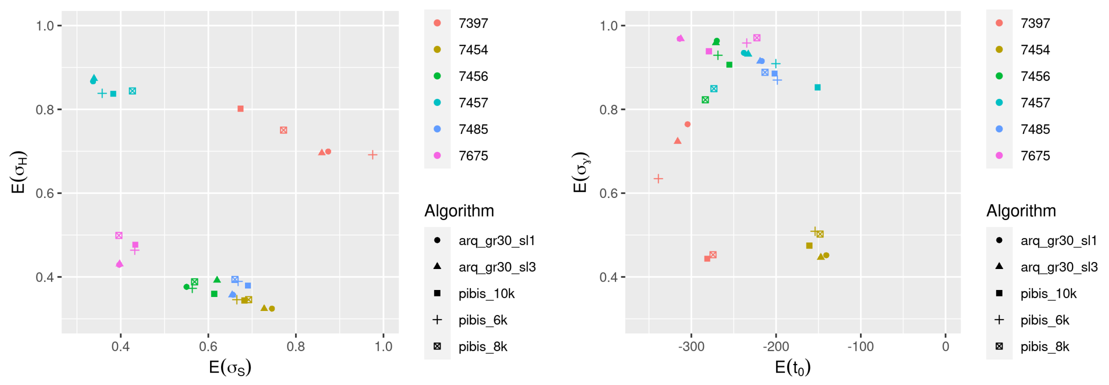


Figure D.11: Yorkshire

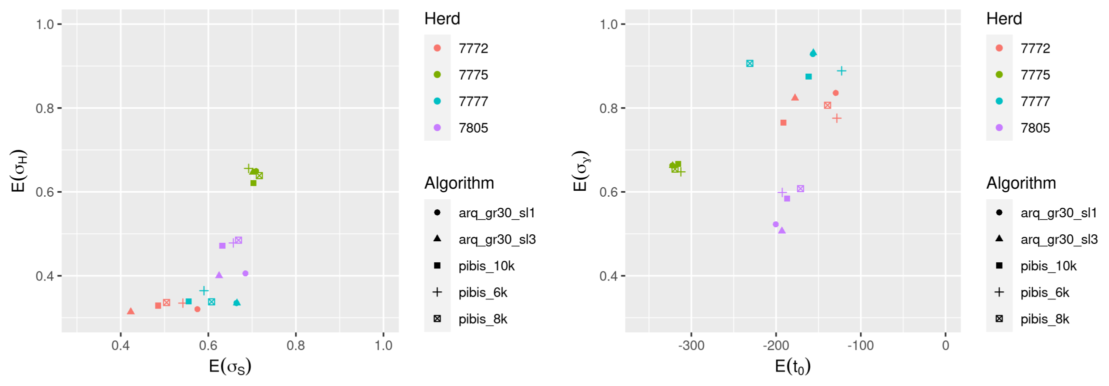


Figure D.12: East of England

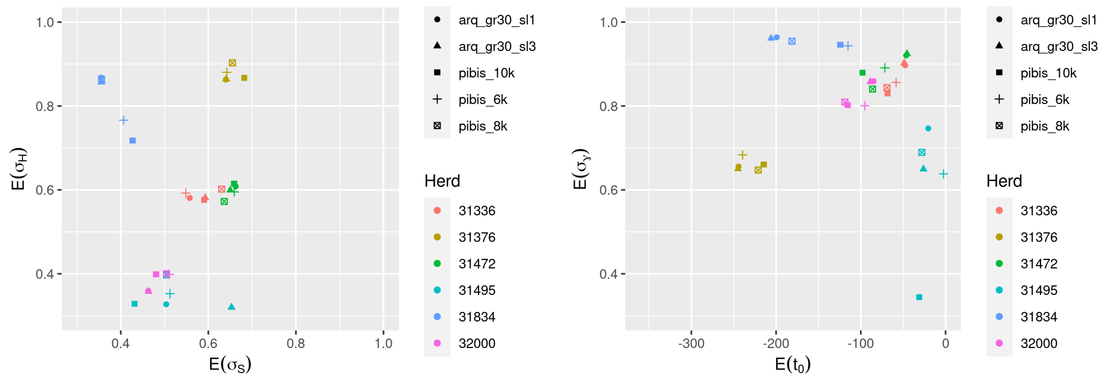


Figure D.13: Scotland

Frequency dependent TDR-SETIR model

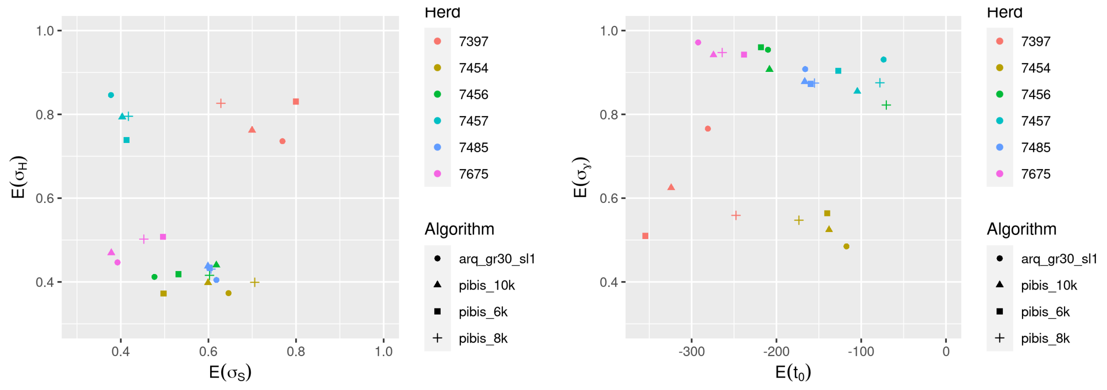


Figure D.14: Yorkshire

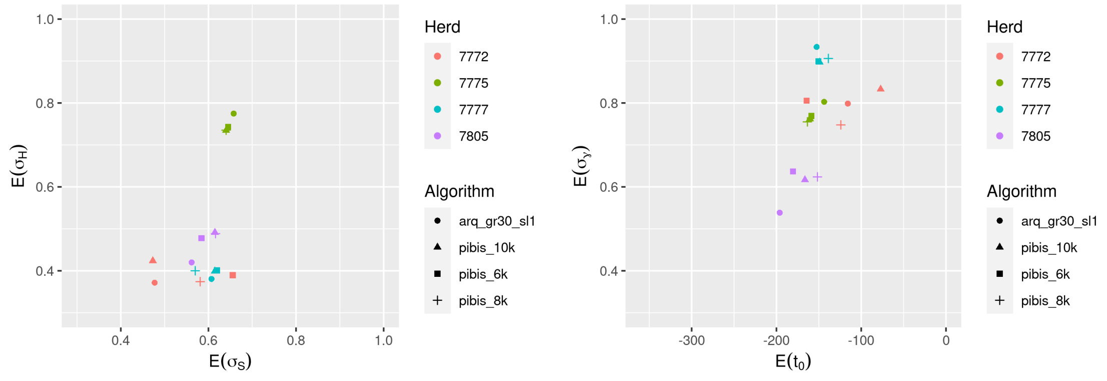


Figure D.15: East of England

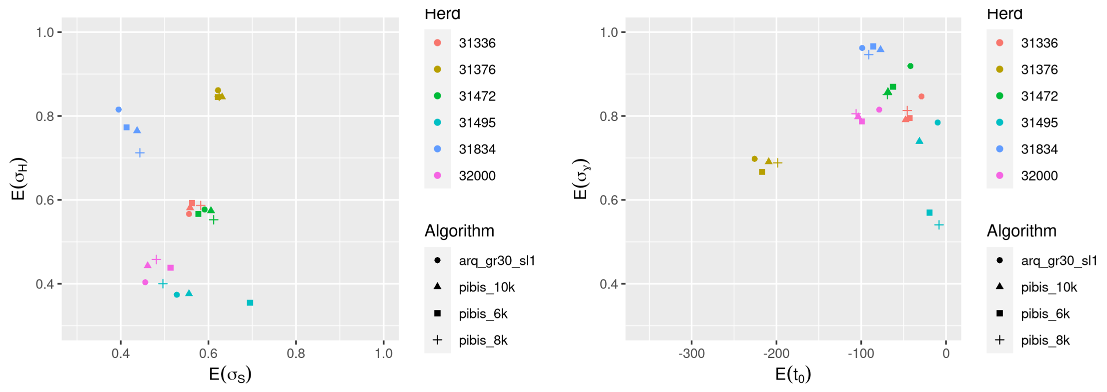


Figure D.16: Scotland

Density dependent TDR-SETIR model

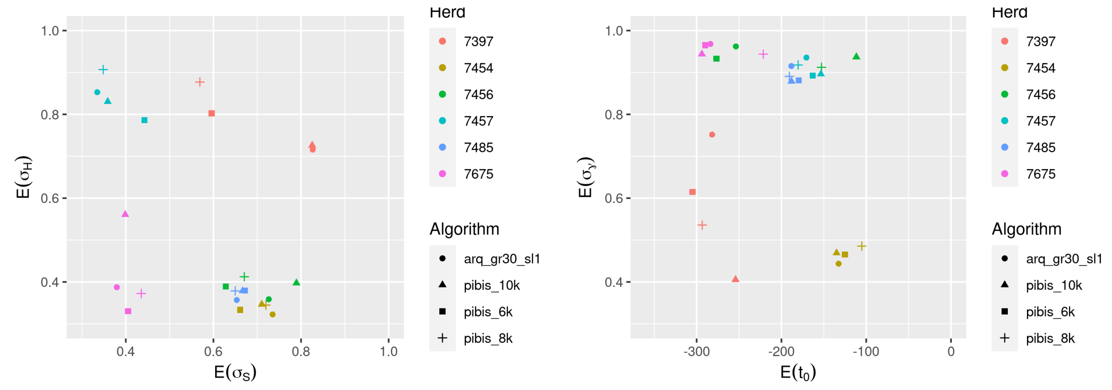


Figure D.17: Yorkshire

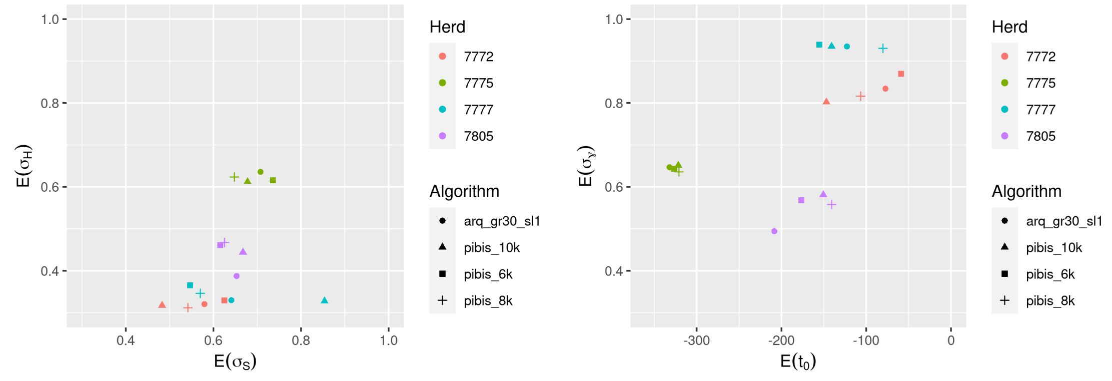


Figure D.18: East of England

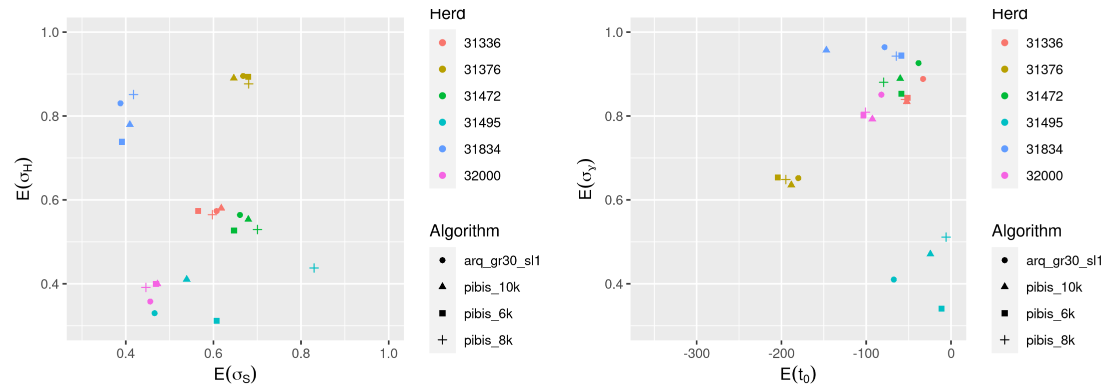


Figure D.19: Scotland

Reinfection (SEYZ) model

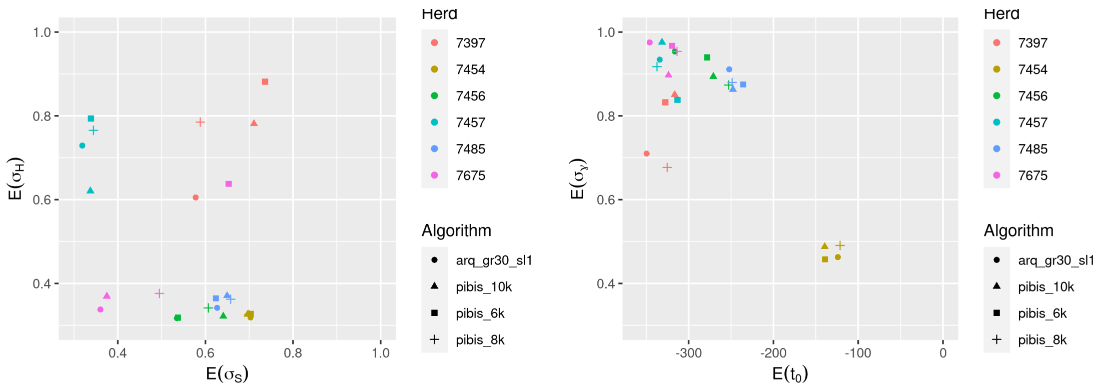


Figure D.20: Yorkshire

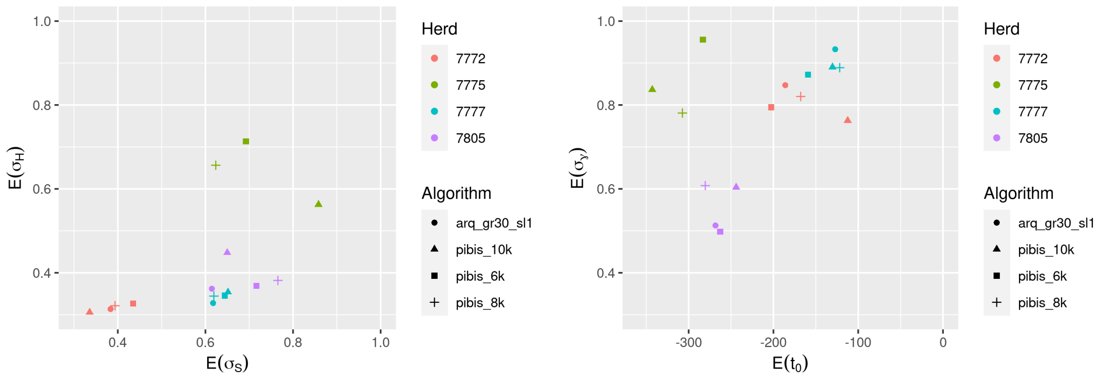


Figure D.21: East of England

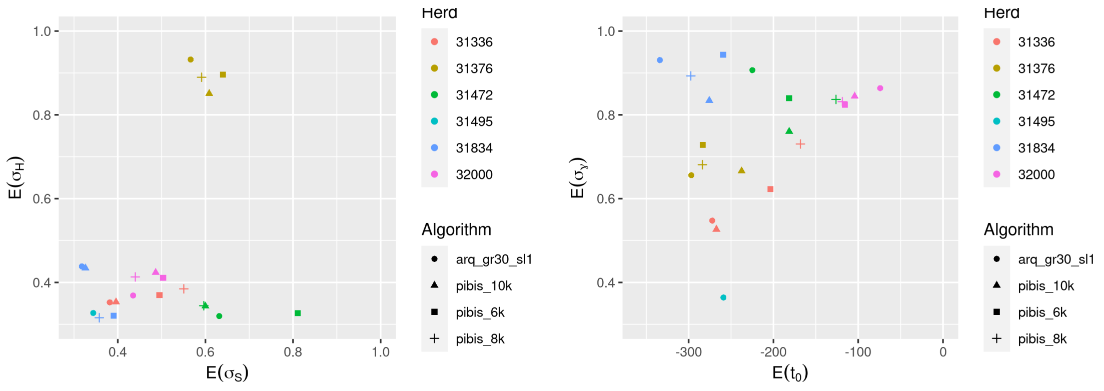


Figure D.22: Scotland

Alternative reinfection model

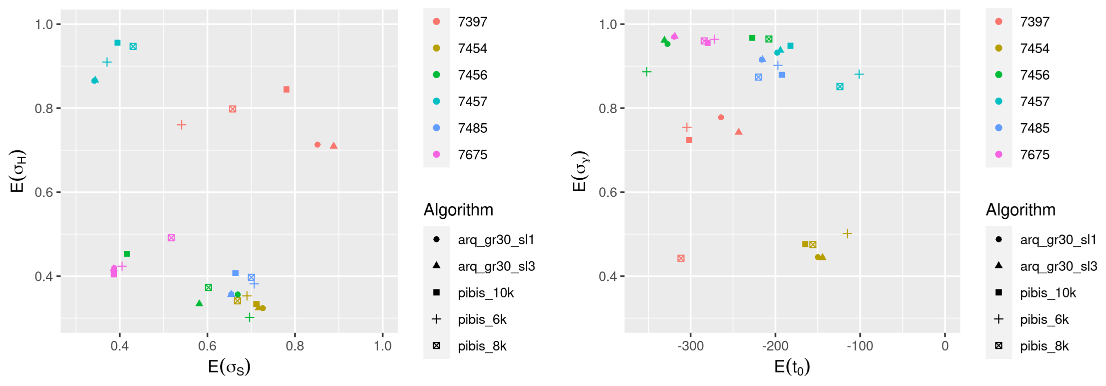


Figure D.23: Yorkshire

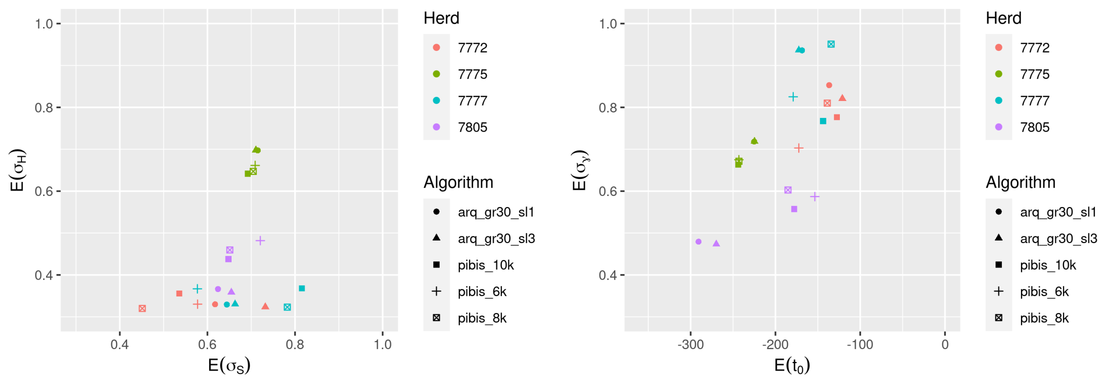


Figure D.24: East of England

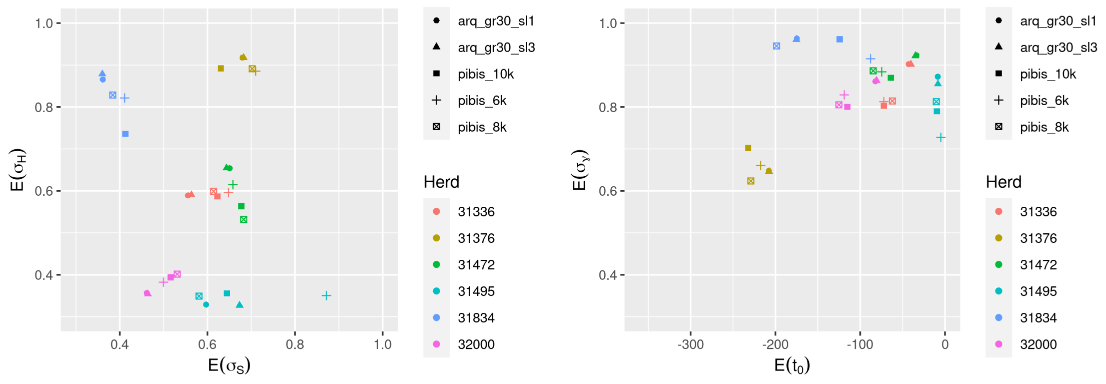


Figure D.25: Scotland

D.3 Within-herd estimates by model

Here we provide a visual summary of the parameter estimates that comprise part of the main results presented in chapter 6 (see §6.4) organised by model.

θ_i	Symbol	Description	Prior: U_a	U_b
1	β	Infection (density dependent)	0.000	0.001
1	β	Infection (frequency dependent)	0.0	0.1
2	γ_T	Progression to T	0.0	0.1
3	γ_I	Progression to I	0.0	0.1
4	t_0	Time of initial onset	-360.0	0.0
5	σ_S	SICCT test sensitivity – standard	0.3	1.0
6	σ_H	SICCT test sensitivity – high	0.3	1.0
7	σ_γ	IFN$_\gamma$ test sensitivity	0.3	1.0

Table D.6: Test-determined removal model parameters.

TDR-SEIR model

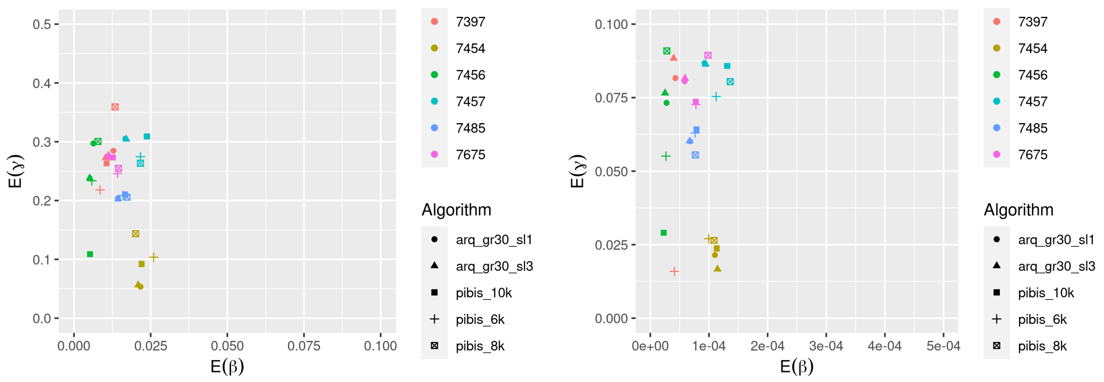


Figure D.26: Yorkshire

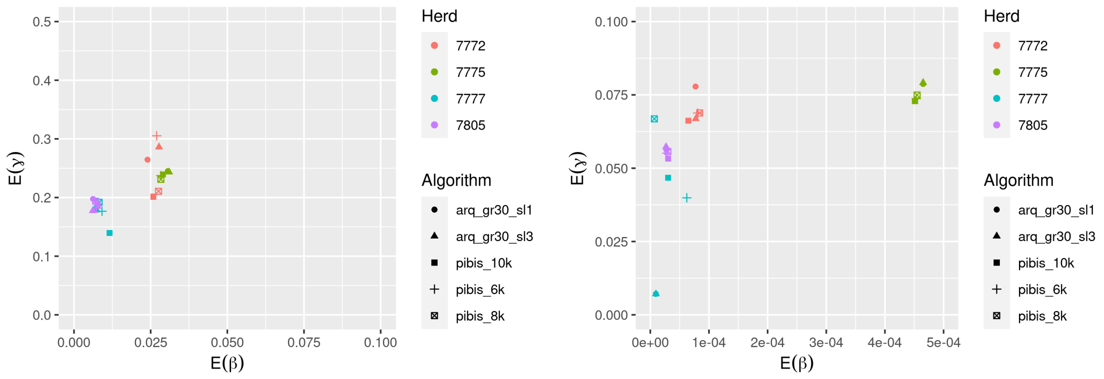
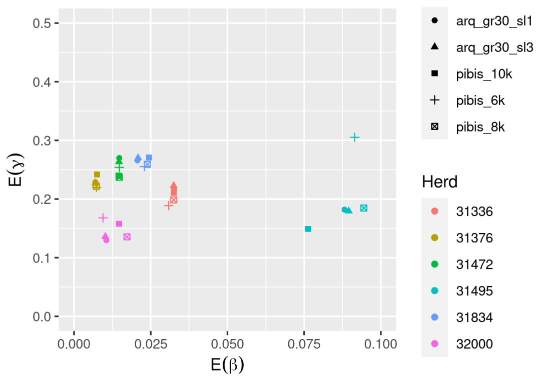
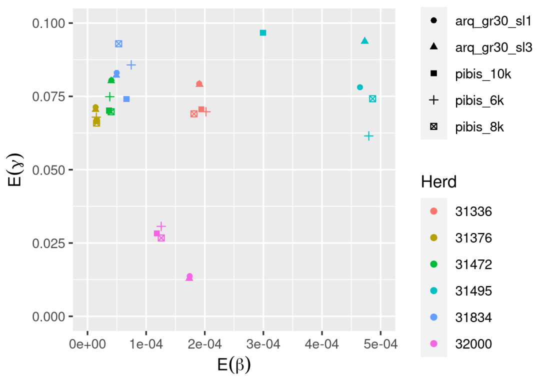


Figure D.27: East of England



(a) Frequency-dependent TDR-SEIR



(b) Density-dependent TDR-SEIR

Figure D.28: Scotland

Frequency dependent TDR-SETIR model

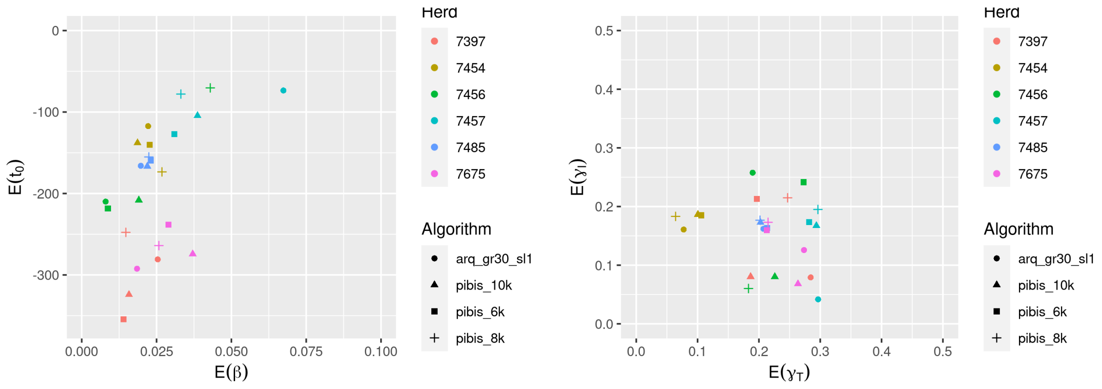


Figure D.29: Yorkshire

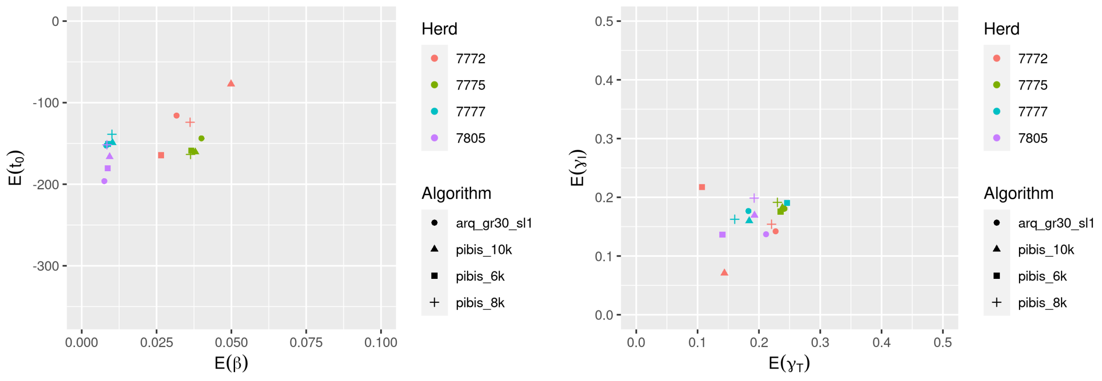


Figure D.30: East of England

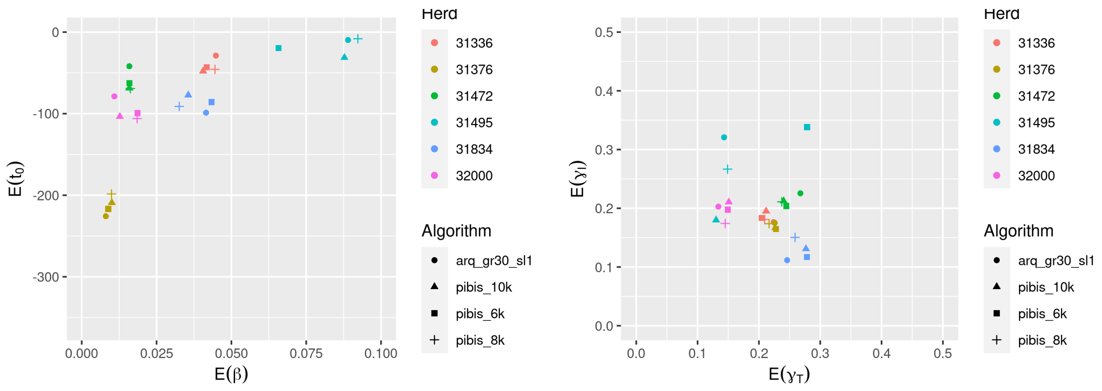


Figure D.31: Scotland

Density dependent TDR-SETIR model

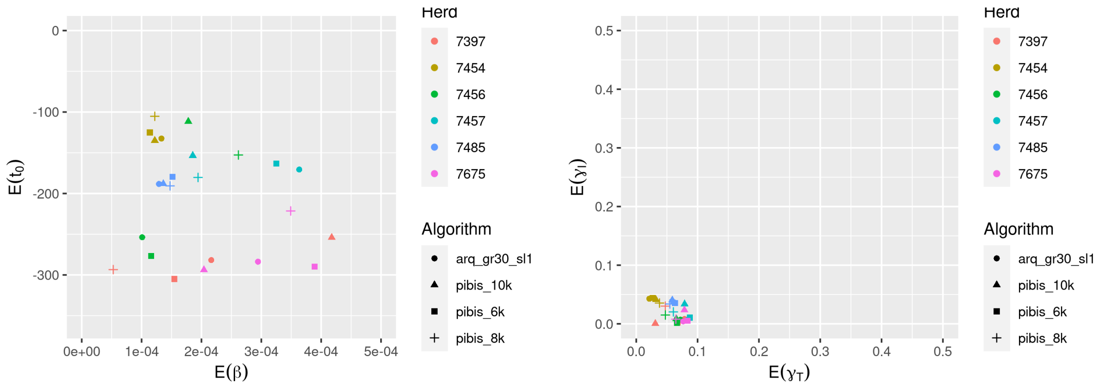


Figure D.32: Yorkshire

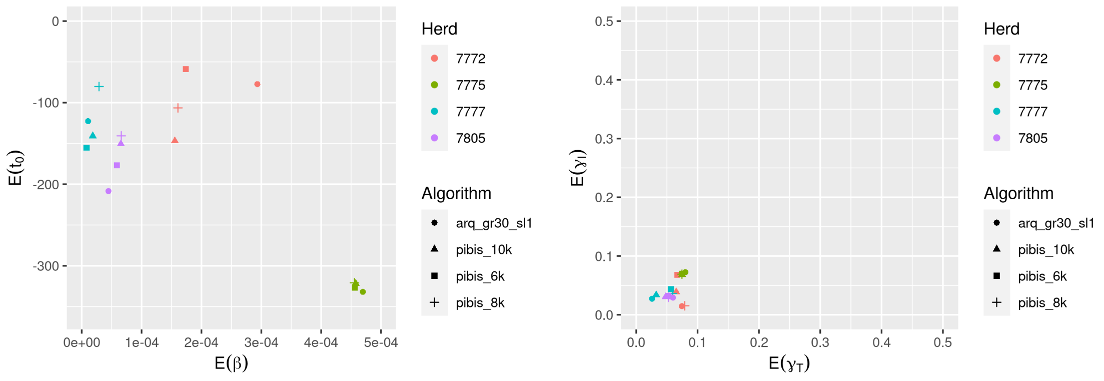


Figure D.33: East of England

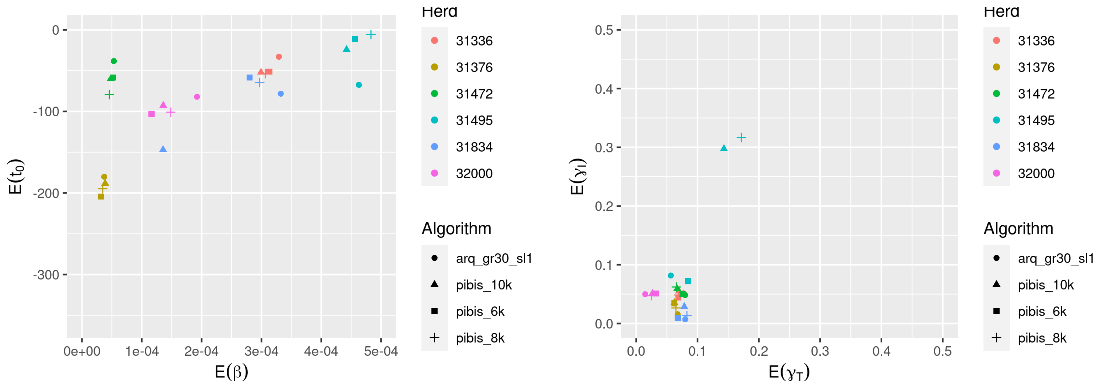


Figure D.34: Scotland

Reinfection (SEYZ) model

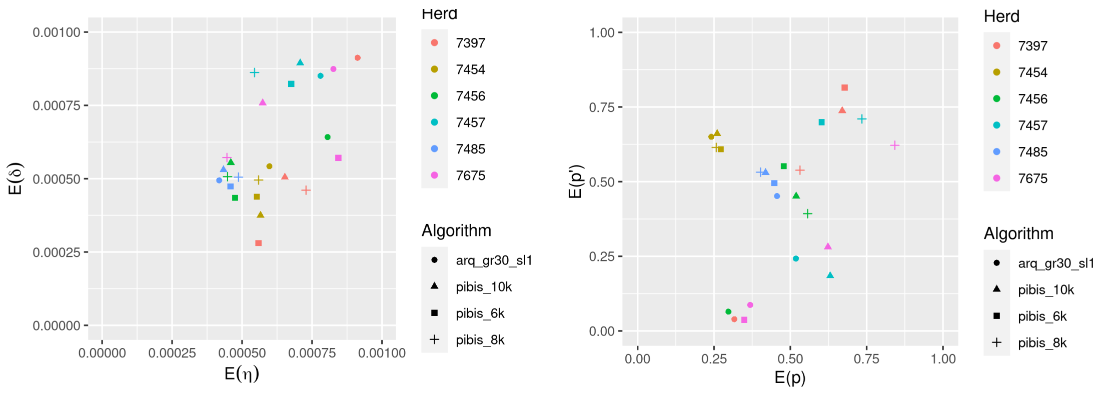


Figure D.35: Yorkshire

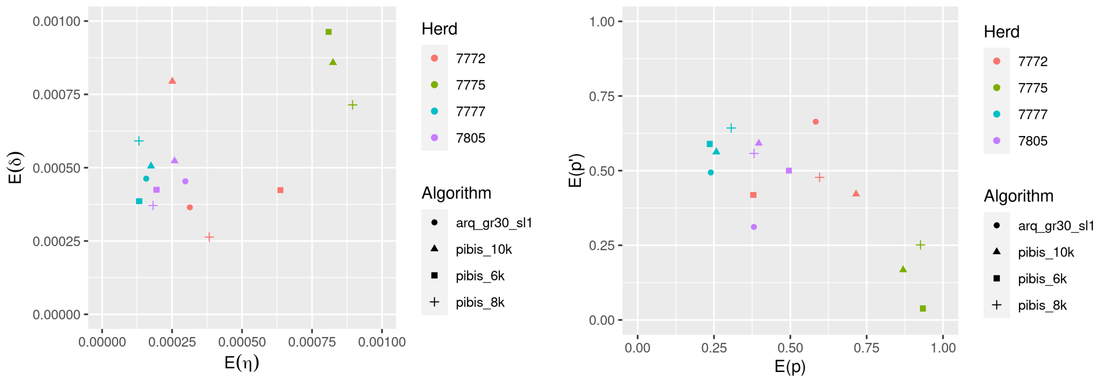


Figure D.36: East of England

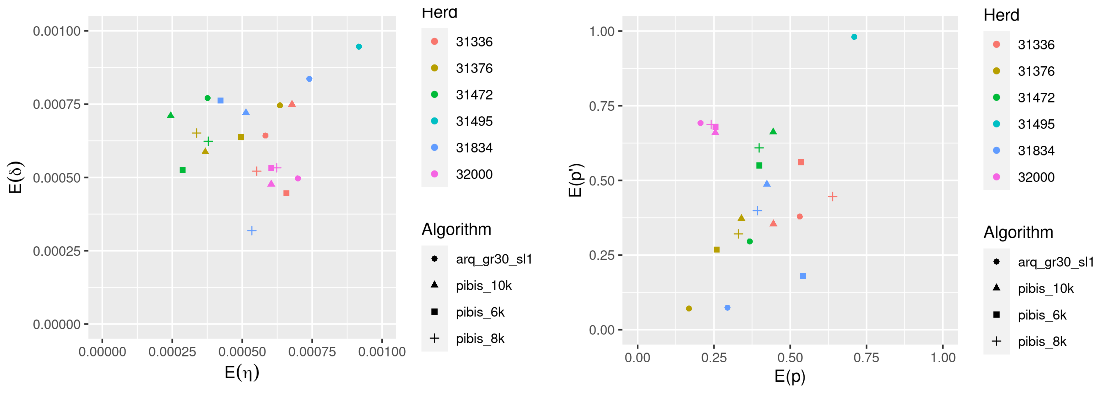
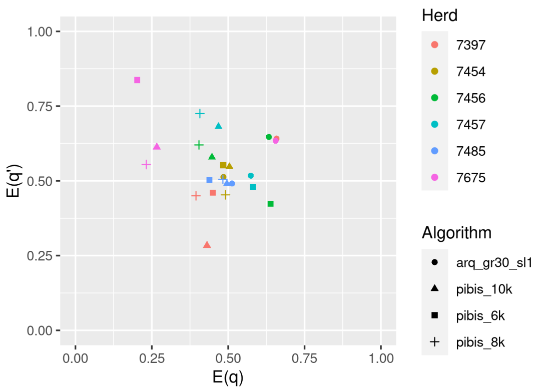
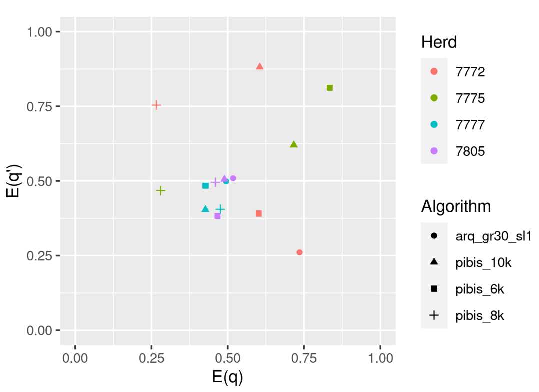


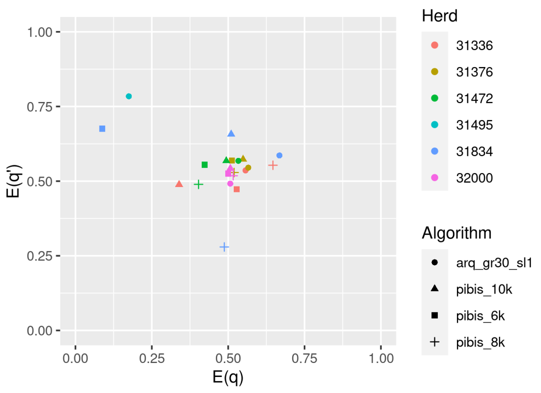
Figure D.37: Scotland



(a) Yorkshire



(b) East of England



(c) Scotland

Figure D.38: Joint marginal model parameter distributions .

θ_i	Symbol	Description	Prior: U_a	U_b
1	η	Infection	0.000	0.001
2	δ	Recovery	0.000	0.001
3	p'	Super infection	0.0	1.0
4	p	Progression	0.0	1.0
5	q	Loss of temporary immunity	0.0	1.0
6	q'	Remission	0.0	1.0
7	t_0	Time of initial onset	-360.0	0.0
8	σ_S	SICCT test sensitivity – standard	0.3	1.0
9	σ_H	SICCT test sensitivity – high	0.3	1.0
10	σ_γ	IFN$_\gamma$ test sensitivity	0.3	1.0

Table D.7: Reinfection model parameters.

θ_i	Symbol	Description	Prior: U_a	U_b
1	t_0	Time of initial onset	-360.0	0.0
2	σ_S	SICCT test sensitivity – standard	0.3	1.0
3	σ_H	SICCT test sensitivity – high	0.3	1.0
4	σ_γ	IFN$_\gamma$ test sensitivity	0.3	1.0
5	β	Infection	0.0	0.01
6	γ	Progression	0.0	0.1
7	κ	Reinfection scalar	0.0	10.0

Table D.8: Alternative reinfection model parameters.

Alternative reinfection model

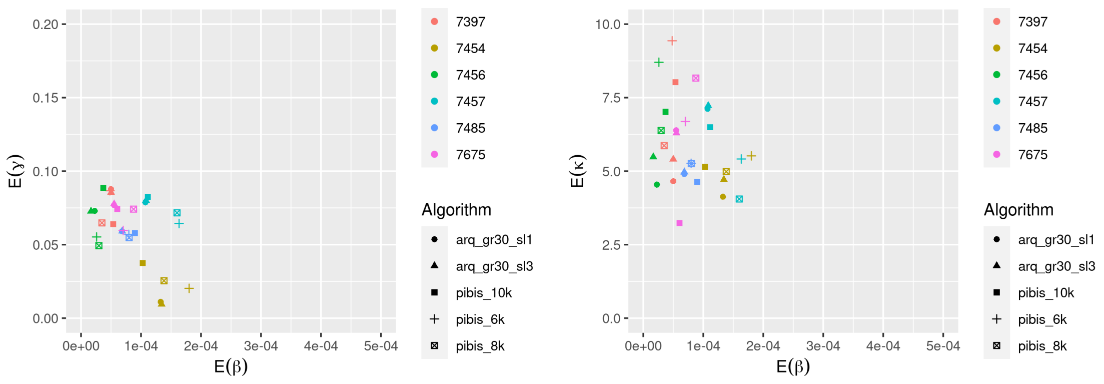


Figure D.39: Yorkshire

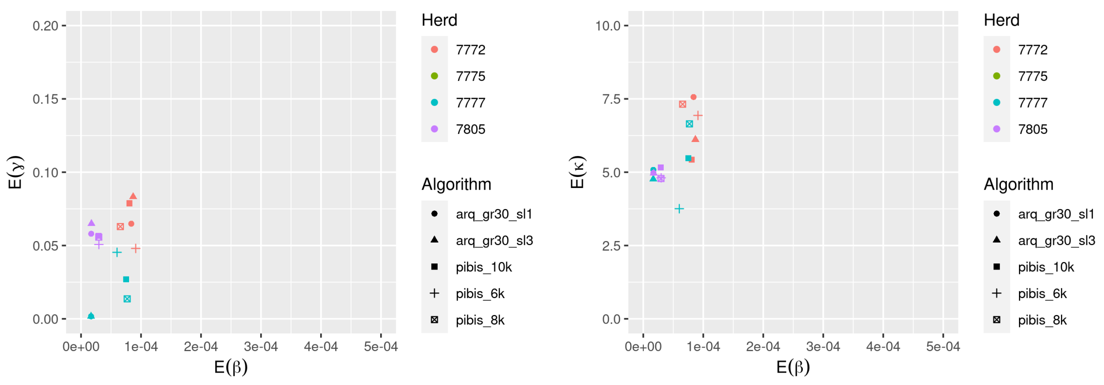


Figure D.40: East of England

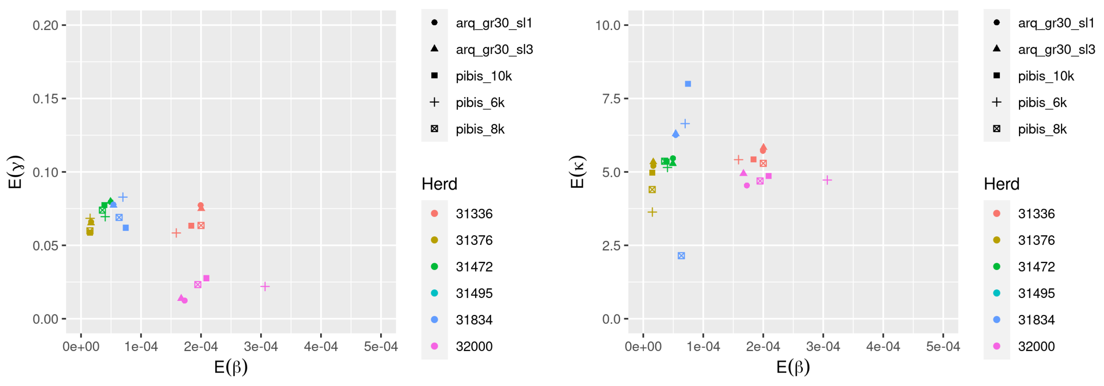


Figure D.41: Scotland

Appendix E

Hawkes model results

Here we provide more detailed inference results corresponding to those reported in Chapter 7.

E.1 Homogeneous process model results

E.1.1 North east England

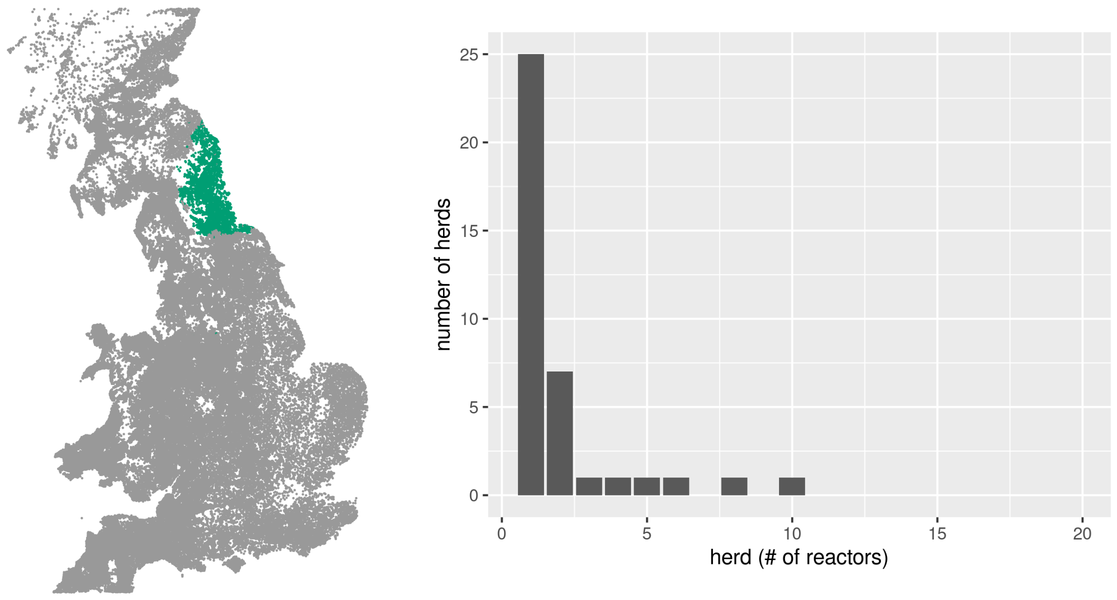


Figure E.1: Herd and reactor distribution. The latter is given for the total number reactors by herd for the entire observation period.

Number of herds: 1545
Reactor herds: 40
Number of reactors, i.e. animals: 135

σ_i	animals	+ve
1	4725	18
2	733	2
3	2364	49
4	10	66
5	1	0

Figure E.2: VetNet surveillance data for selected North east England herds. The diagnostic test data are categorised as follows: standard SICCT herd tests; high-sensitivity SICCT herd tests; risk-based surveillance testing; and the IFN$_\gamma$ blood test trial, denoted σ_i for 1:4 respectively.

Marginal sample distributions

Here we provide a visual summary of the parameter estimates for the basic seven-parameter model (for the North east England group.) A tabulated summary is provided in Table E.1.

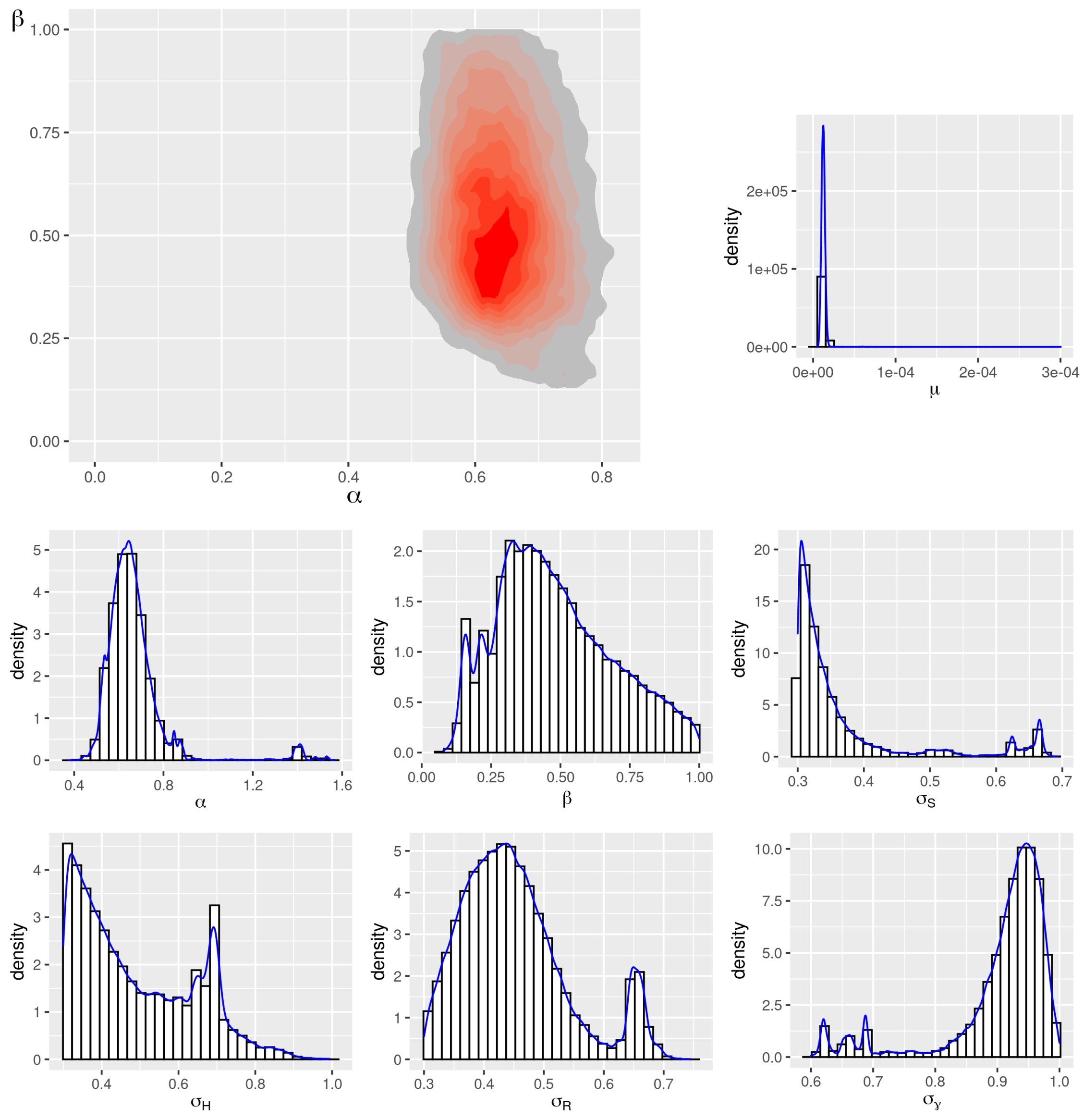


Figure E.3: Marginal sample densities for North east England.

Trace plots

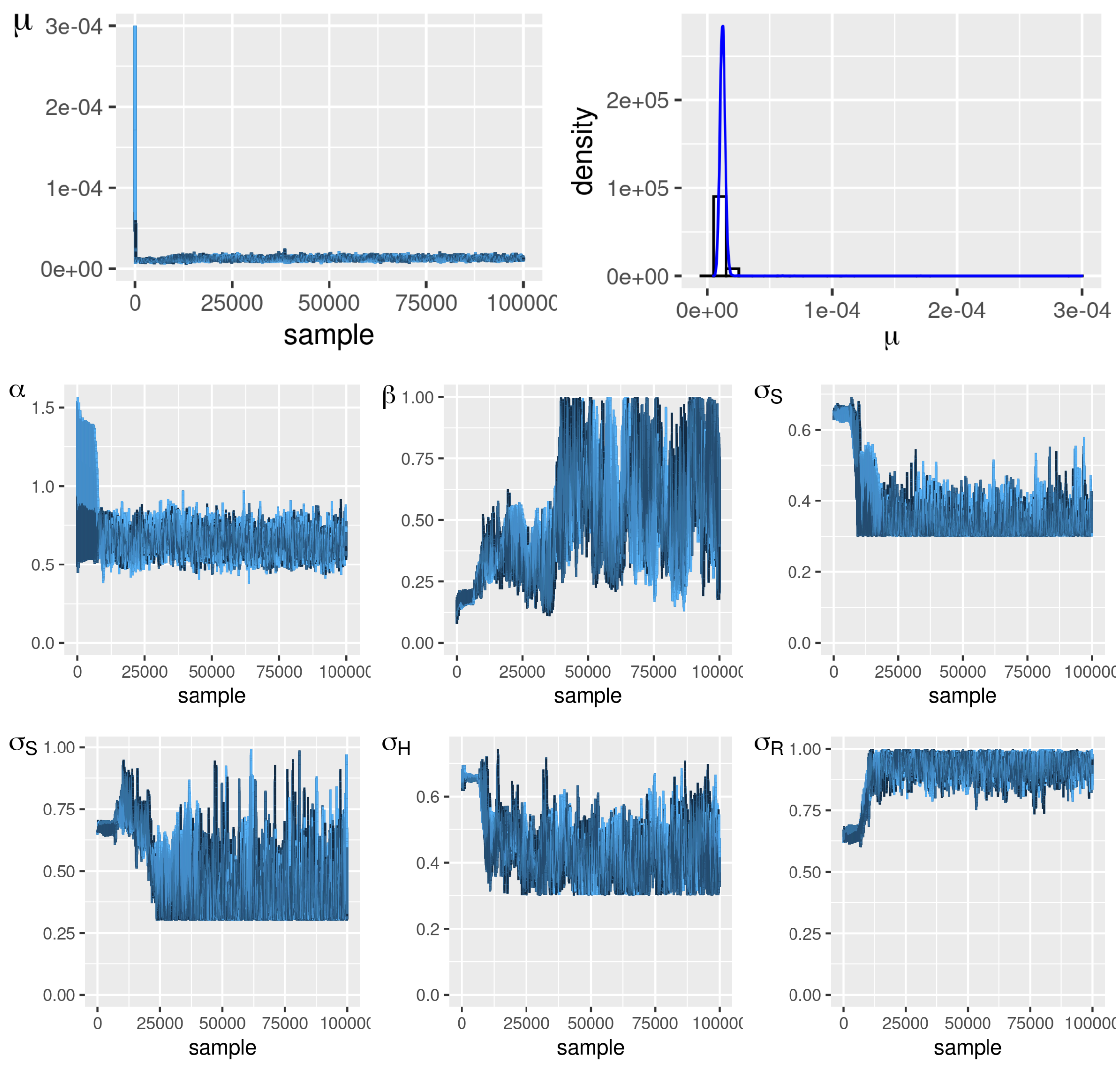
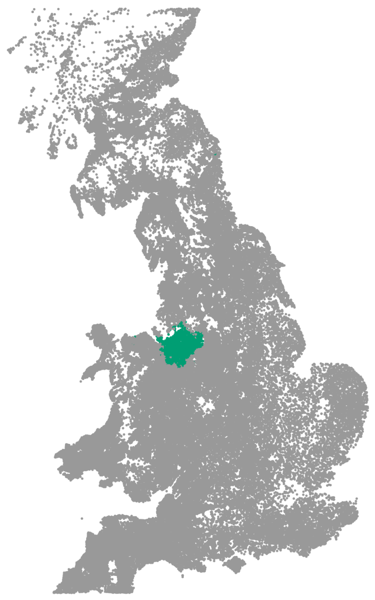


Figure E.4: Trace plots for the North east England analysis.

E.1.2 Cheshire



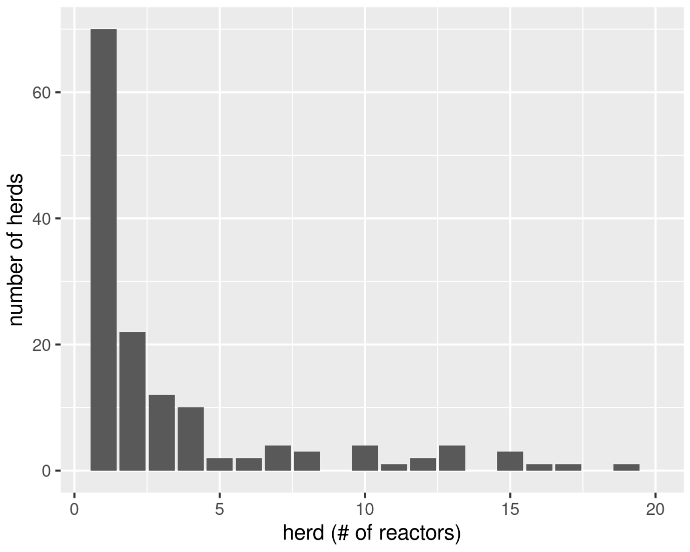


Figure E.5: Herd and reactor distribution. The latter is given for the total number reactors by herd for the entire observation period.

Number of herds: 1364
Reactor herds: 152
Number of reactors, i.e. animals: 955

σ_i	animals	+ve
1	4278	101
2	7582	23
3	4833	744
4	29	87

Figure E.6: VetNet surveillance data for selected Cheshire herds. The diagnostic test data are categorised as follows: standard SICCT herd tests; high-sensitivity SICCT herd tests; risk-based surveillance testing; and the IFN$_\gamma$ blood test trial, denoted σ_i for 1:4 respectively.

Marginal sample distributions

Here we provide a visual summary of the parameter estimates for the basic seven-parameter model (for the Cheshire group.) A tabulated summary is provided in Table E.2.

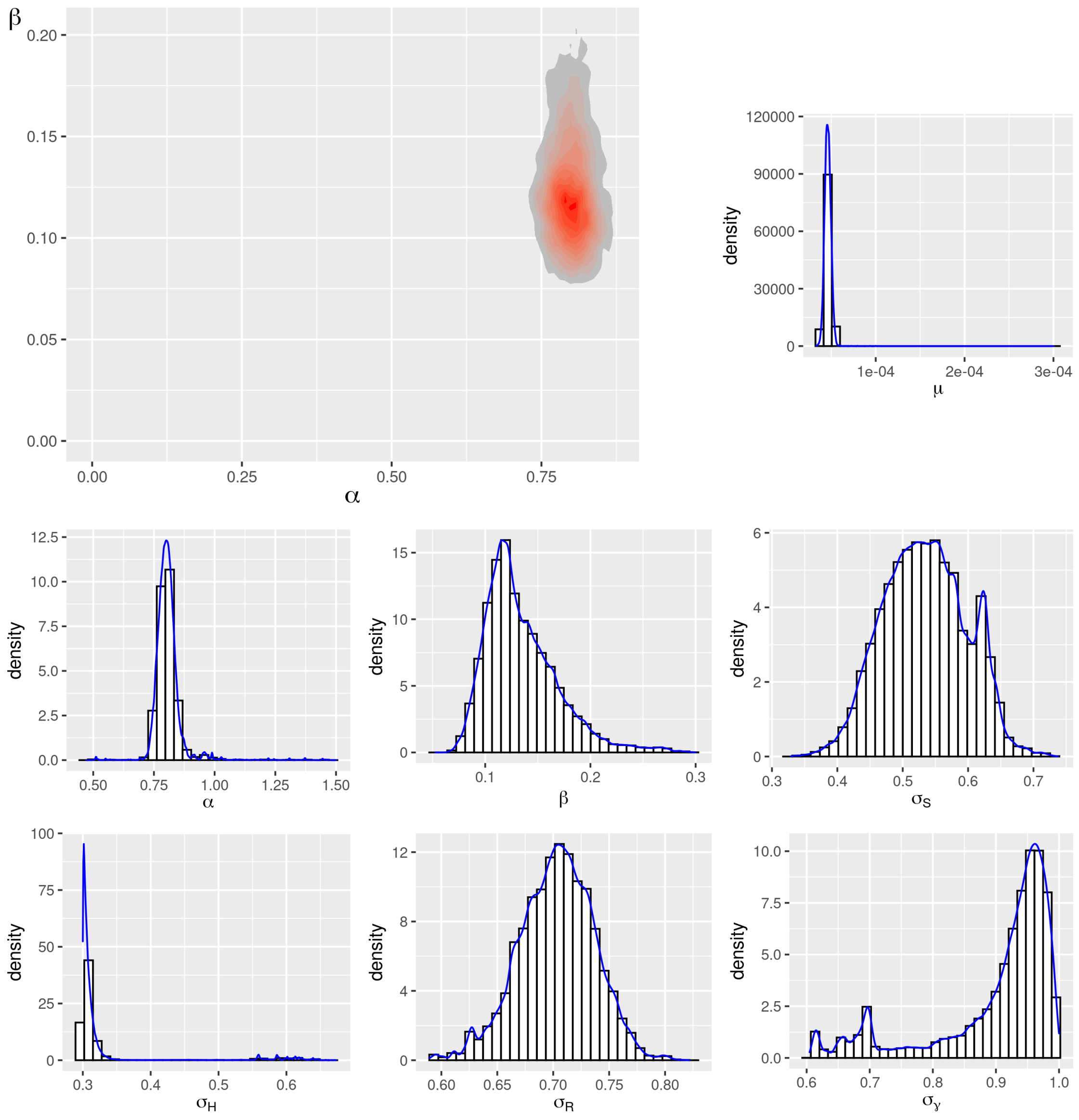


Figure E.7: Marginal sample densities for Cheshire.

Trace plots

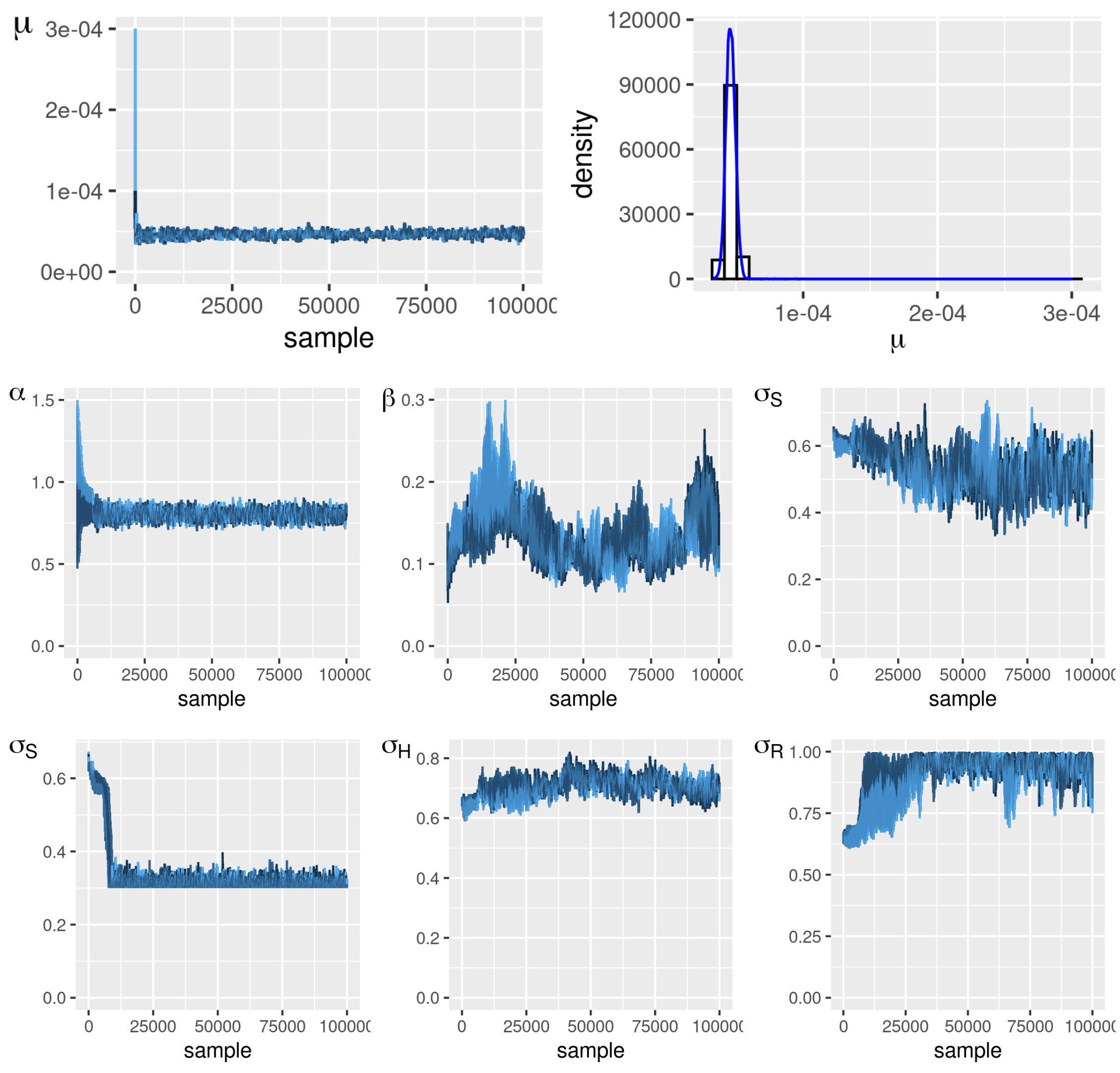


Figure E.8: Trace plots for the Cheshire analysis.

E.1.3 Oxfordshire

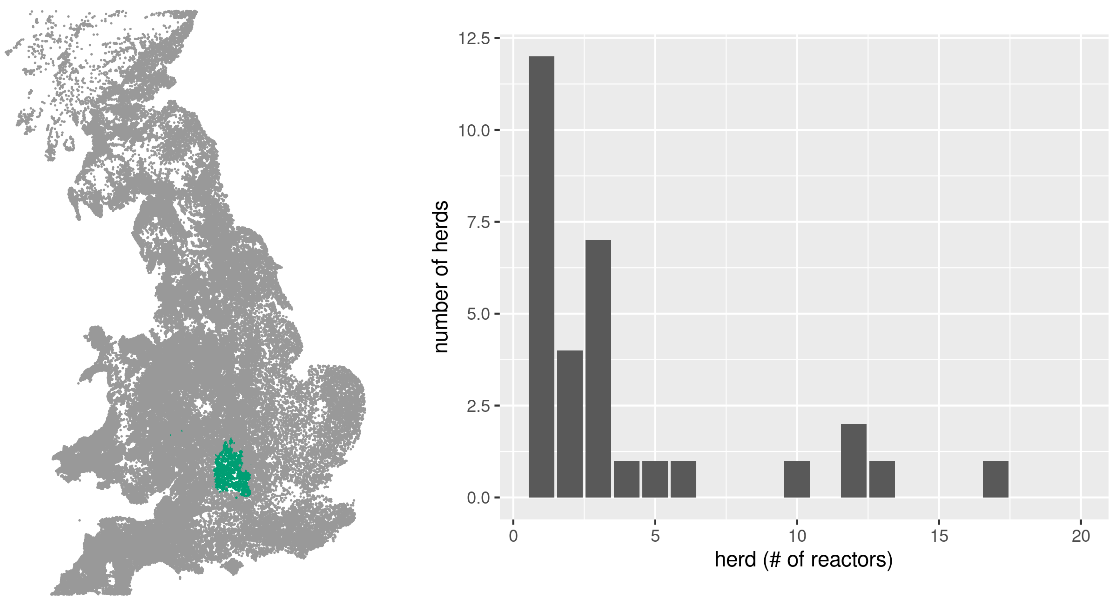


Figure E.9: Herd and reactor distribution. The latter is given for the total number reactors by herd for the entire observation period.

Number of herds: 502
Reactor herds: 35
Number of reactors, i.e. animals: 242

σ_i	animals	+ve
1	1284	34
2	597	1
3	1466	164
4	12	43

Figure E.10: VetNet surveillance data for selected Oxfordshire herds. The diagnostic test data are categorised as follows: standard SICCT herd tests; high-sensitivity SICCT herd tests; risk-based surveillance testing; and the IFN$_\gamma$ blood test trial, denoted σ_i for 1:4 respectively.

Marginal sample distributions

Here we provide a visual summary of the parameter estimates for the basic seven-parameter model (for the Oxfordshire group.) A tabulated summary is provided in Table E.3.

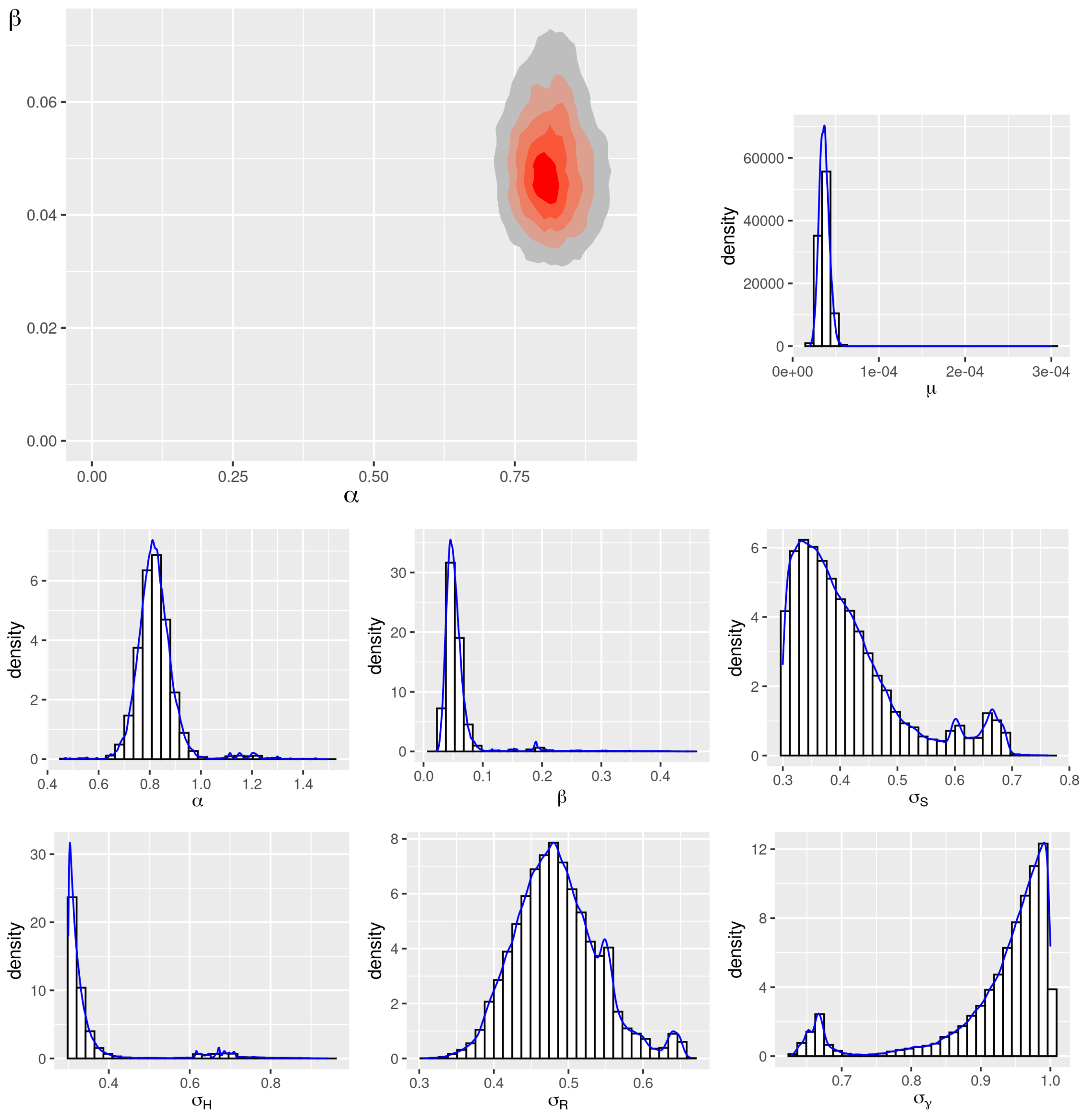


Figure E.11: Marginal sample densities for Oxfordshire.

Trace plots

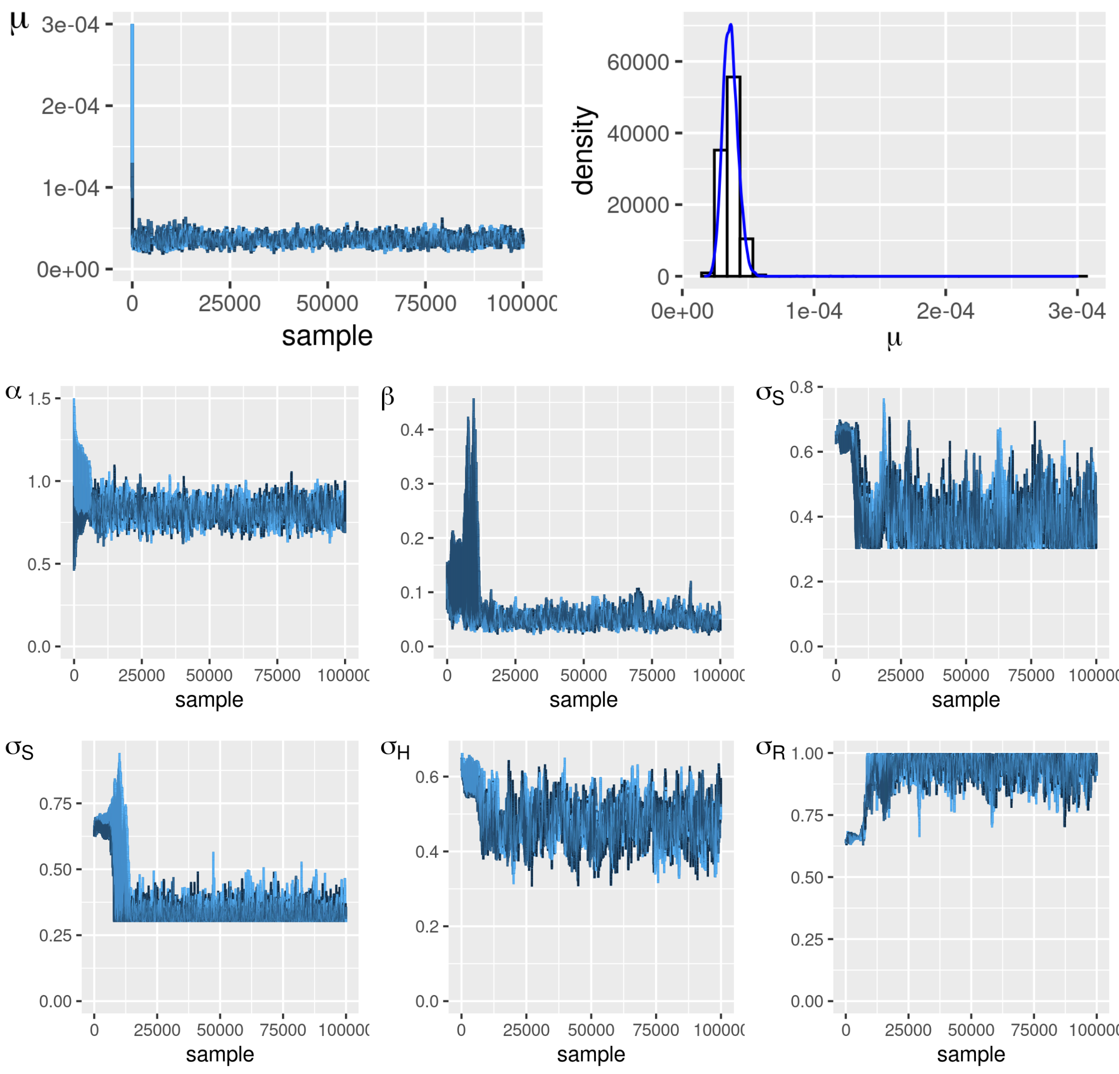
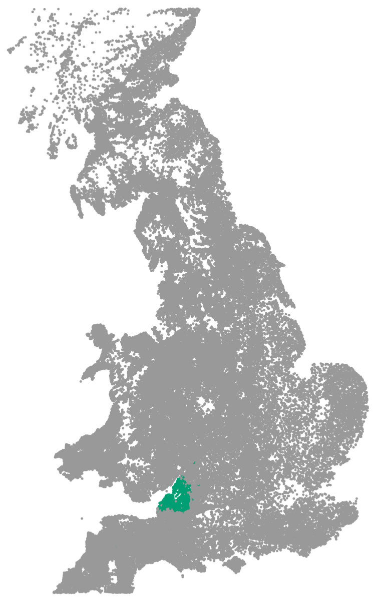


Figure E.12: Trace plots for the Oxfordshire analysis.

E.1.4 Avon



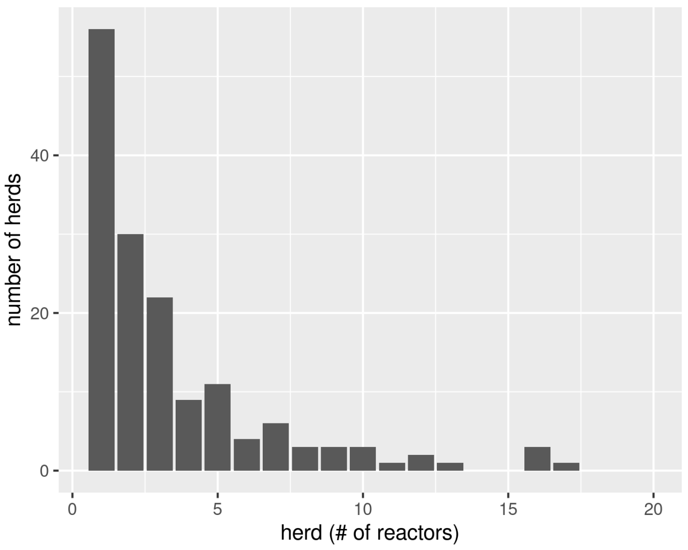


Figure E.13: Herd and reactor distribution. The latter is given for the total number reactors by herd for the entire observation period.

Number of herds: 632
Reactor herds: 161
Number of reactors, i.e. animals: 737

σ_i	animals	+ve
1	3971	205
2	1647	22
3	2402	486
4	30	24

Figure E.14: VetNet surveillance data for selected Avon herds. The diagnostic test data are categorised as follows: standard SICCT herd tests; high-sensitivity SICCT herd tests; risk-based surveillance testing; and the IFN$_\gamma$ blood test trial, denoted σ_i for 1:4 respectively.

Marginal sample distributions

Here we provide a visual summary of the parameter estimates for the basic seven-parameter model (for the Avon group.) A tabulated summary is provided in Table E.4.

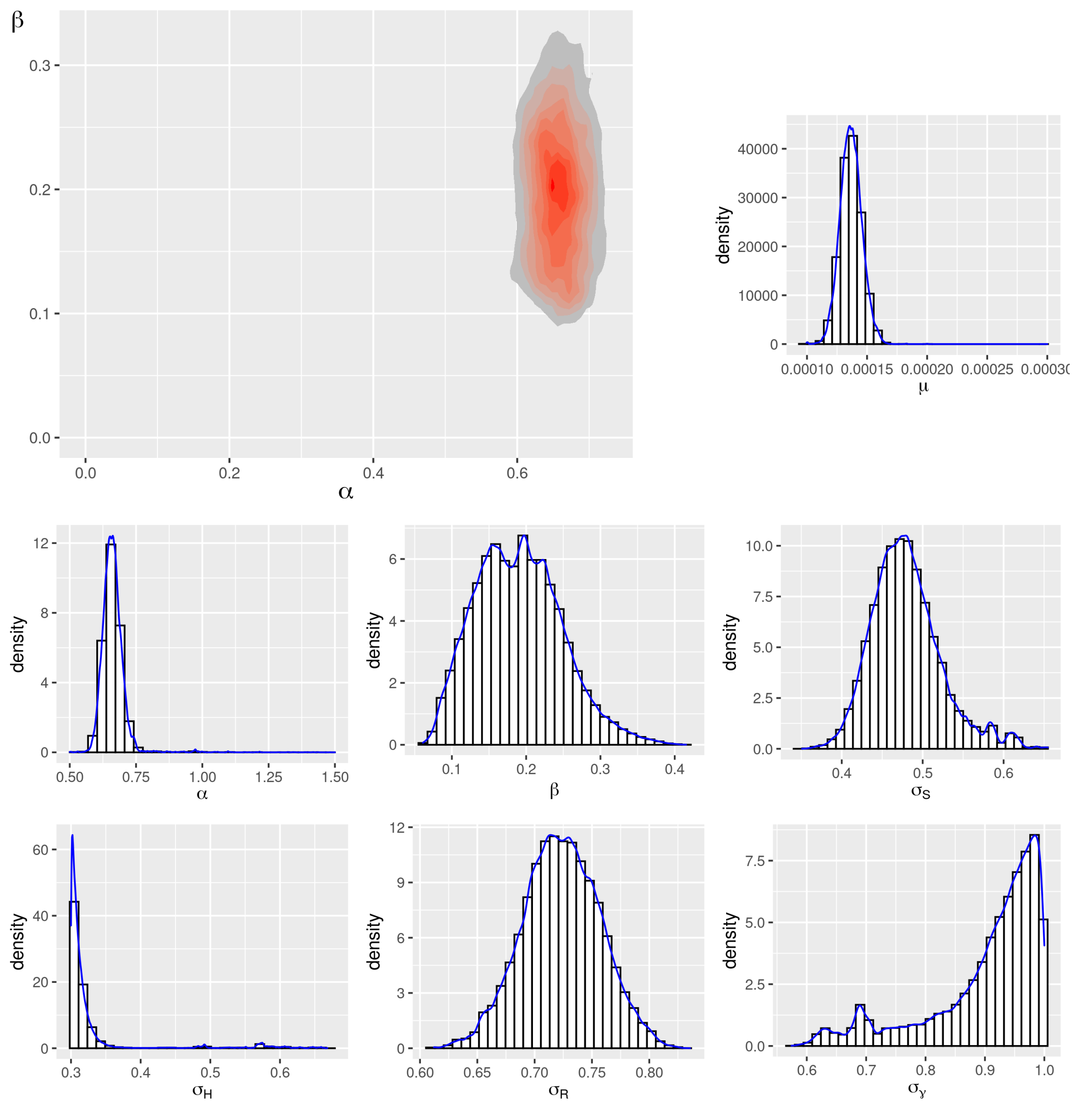


Figure E.15: Marginal sample densities for Avon.

Trace plots

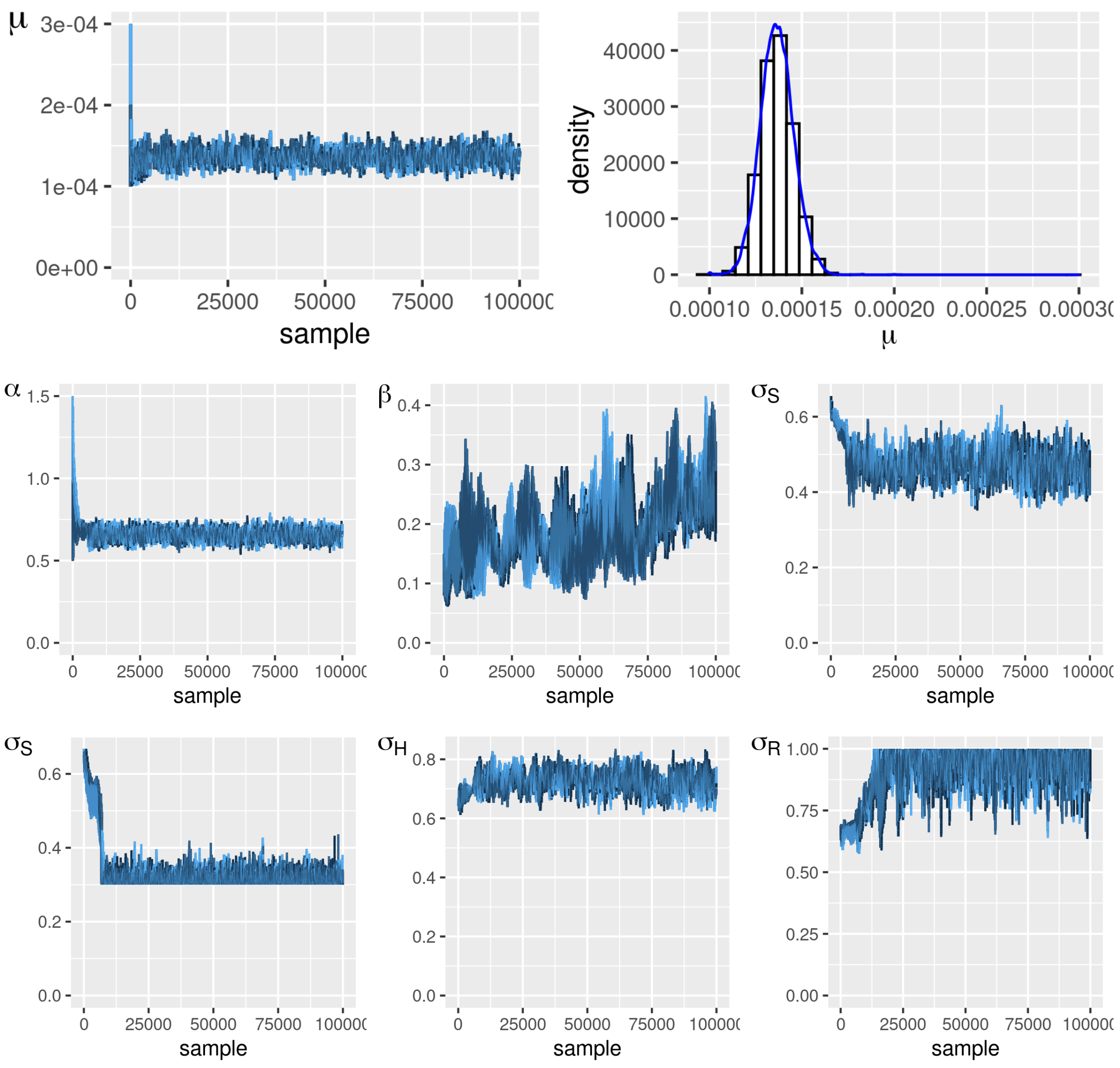
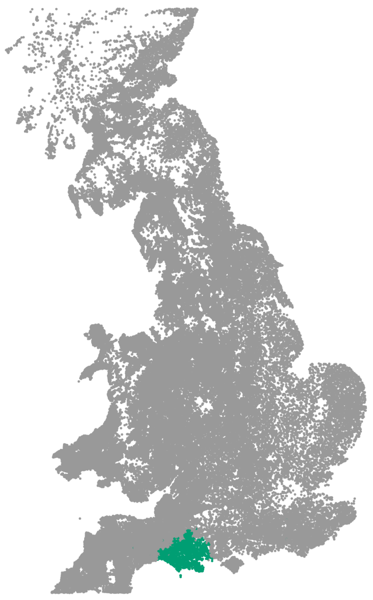


Figure E.16: Trace plots for the Avon analysis.

E.1.5 Dorset



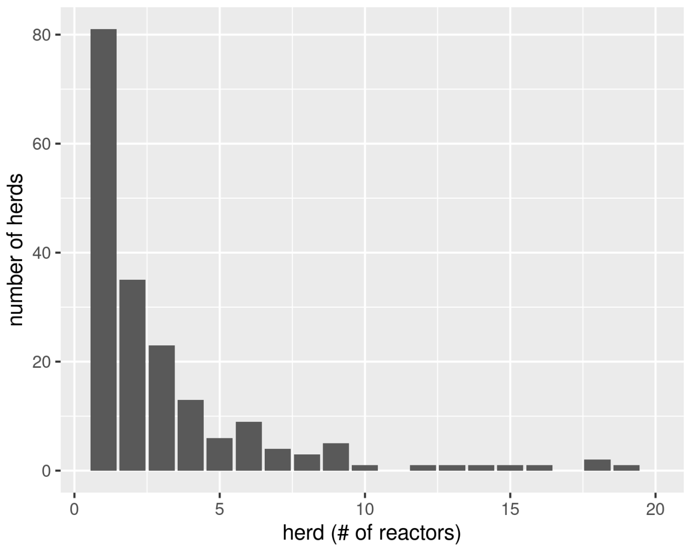


Figure E.17: Herd and reactor distribution. The latter is given for the total number reactors by herd for the entire observation period.

Number of herds: 971
Reactor herds: 196
Number of reactors, i.e. animals: 849

σ_i	animals	+ve
1	3935	186
2	3321	30
3	3649	496
4	54	137

Figure E.18: VetNet surveillance data for selected Dorset herds. The diagnostic test data are categorised as follows: standard SICCT herd tests; high-sensitivity SICCT herd tests; risk-based surveillance testing; and the IFN$_\gamma$ blood test trial, denoted σ_i for 1:4 respectively.

Marginal sample distributions

Here we provide a visual summary of the parameter estimates for the basic seven-parameter model (for the Dorset group.) A tabulated summary is provided in Table E.5.

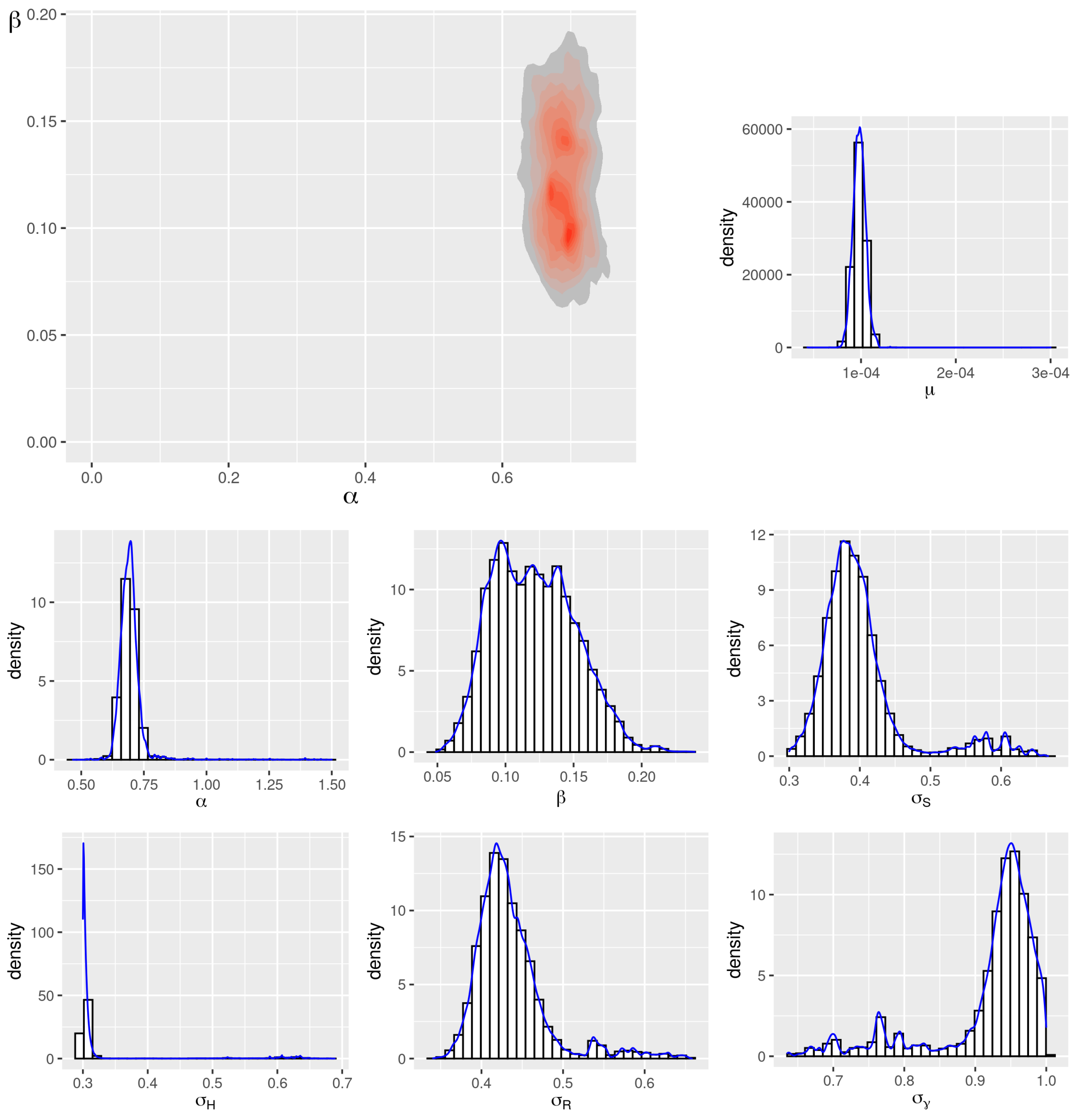


Figure E.19: Marginal sample densities for Dorset.

Trace plots

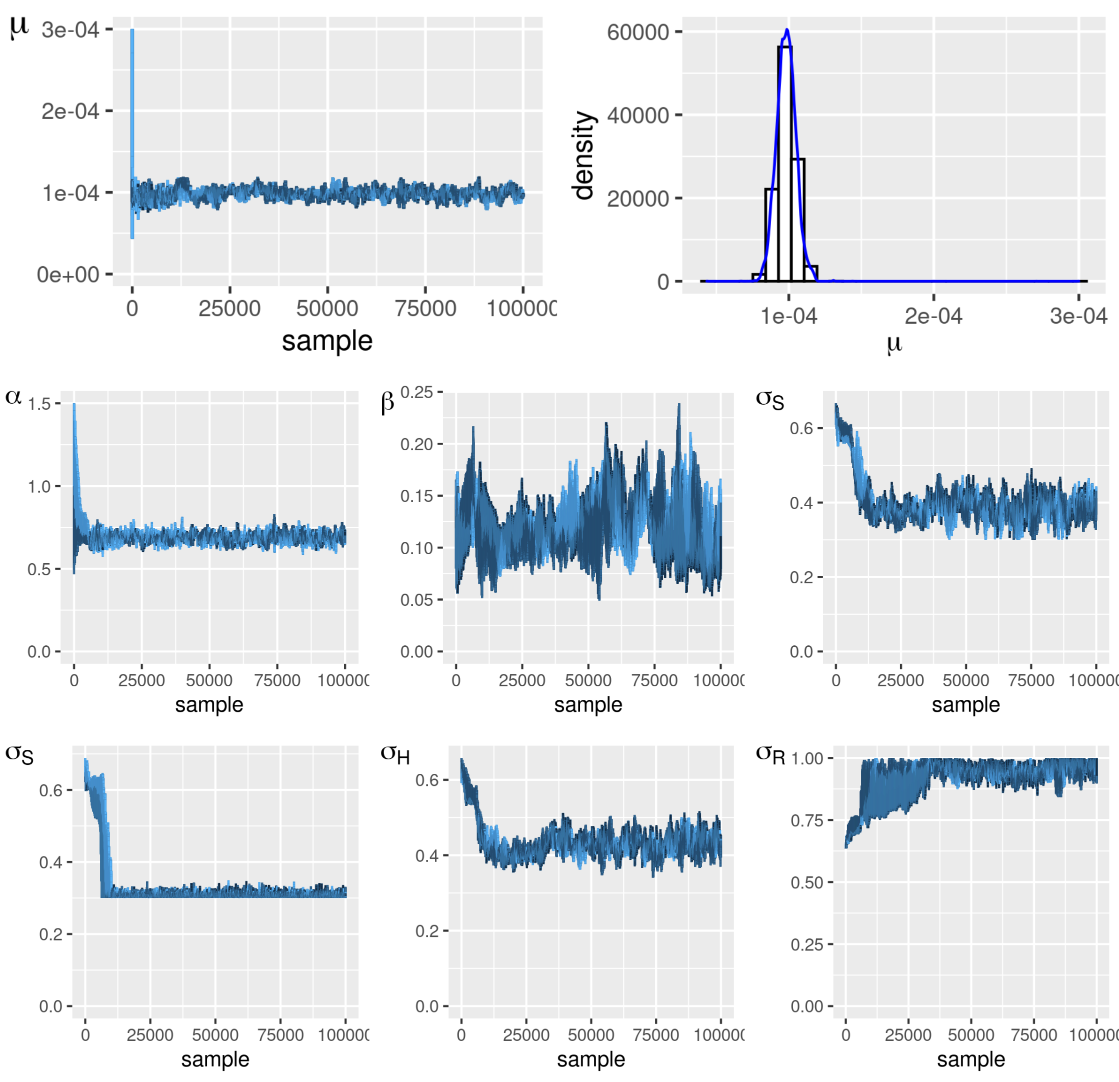
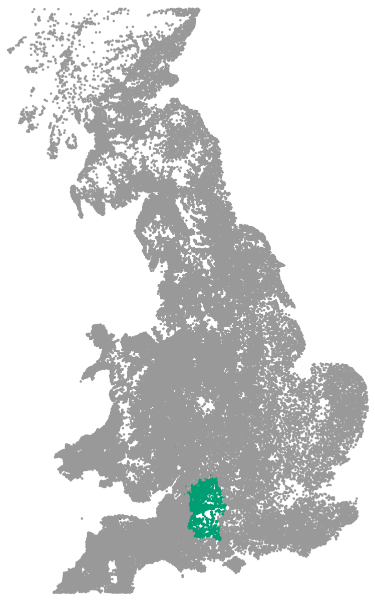


Figure E.20: Trace plots for the Dorset analysis.

E.1.6 Wiltshire



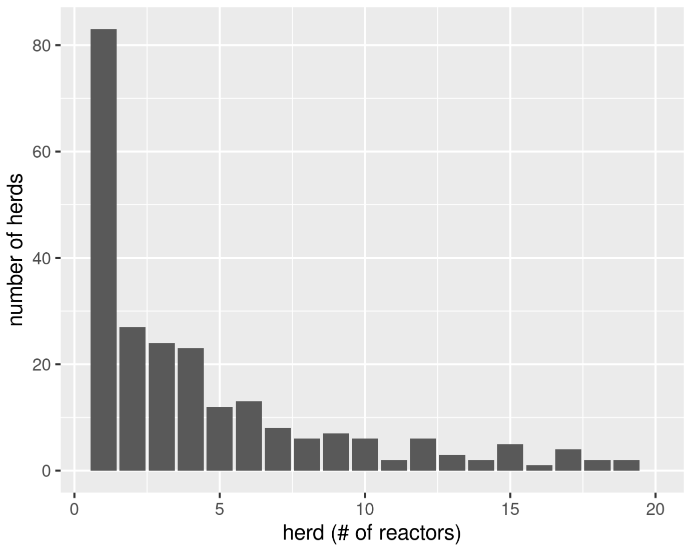


Figure E.21: Herd and reactor distribution. The latter is given for the total number reactors by herd for the entire observation period.

Number of herds: 851
Reactor herds: 258
Number of reactors, i.e. animals: 1801

σ_i	animals	+ve
1	4371	478
2	1800	35
3	3901	1225
4	36	63

Figure E.22: VetNet surveillance data for selected Wiltshire herds. The diagnostic test data are categorised as follows: standard SICCT herd tests; high-sensitivity SICCT herd tests; risk-based surveillance testing; and the IFN$_\gamma$ blood test trial, denoted σ_i for 1:4 respectively.

Marginal sample distributions

Here we provide a visual summary of the parameter estimates for the basic seven-parameter model (for the Wiltshire group.) A tabulated summary is provided in Table E.6.

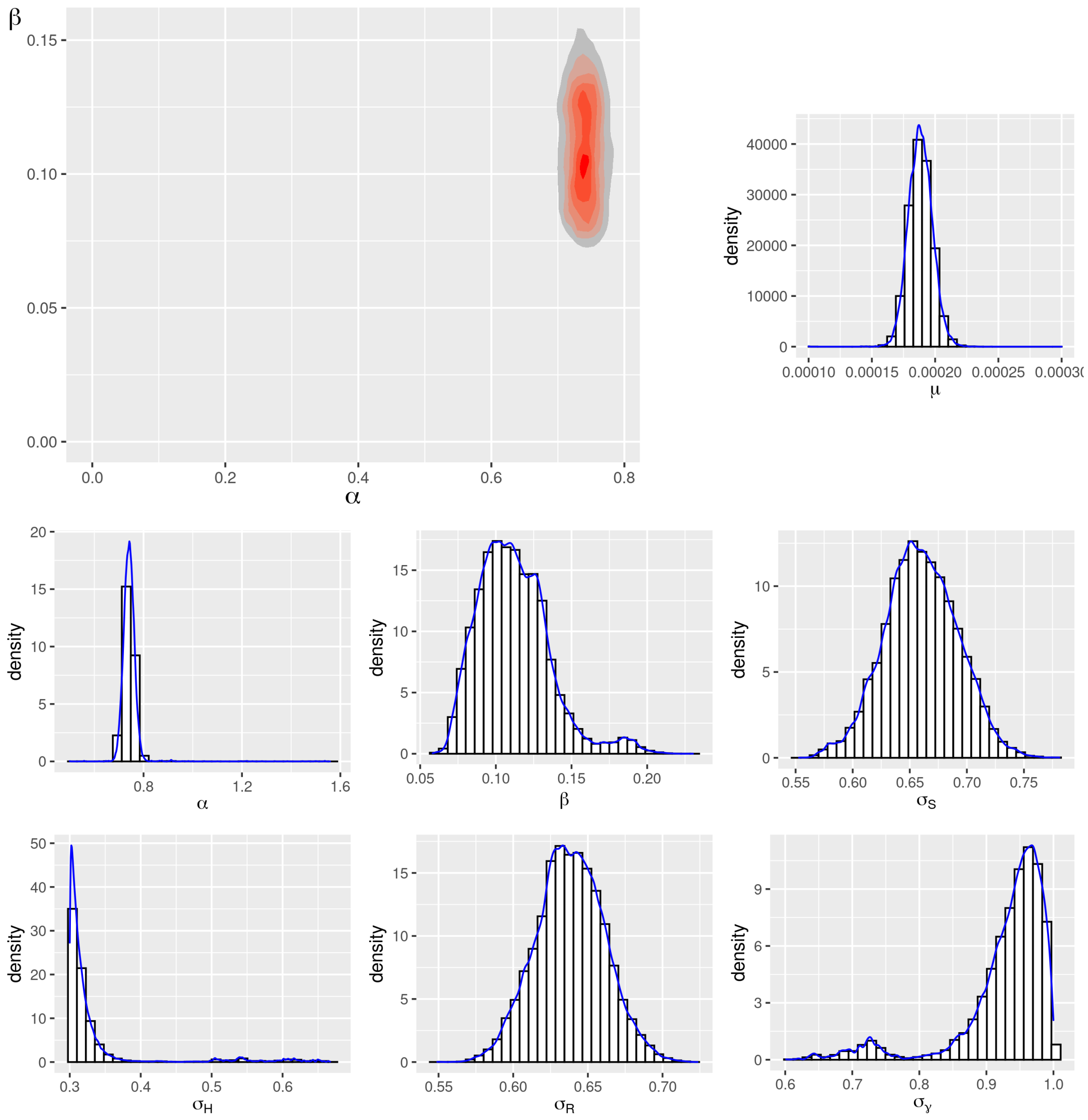


Figure E.23: Marginal sample densities for Wiltshire.

Trace plots

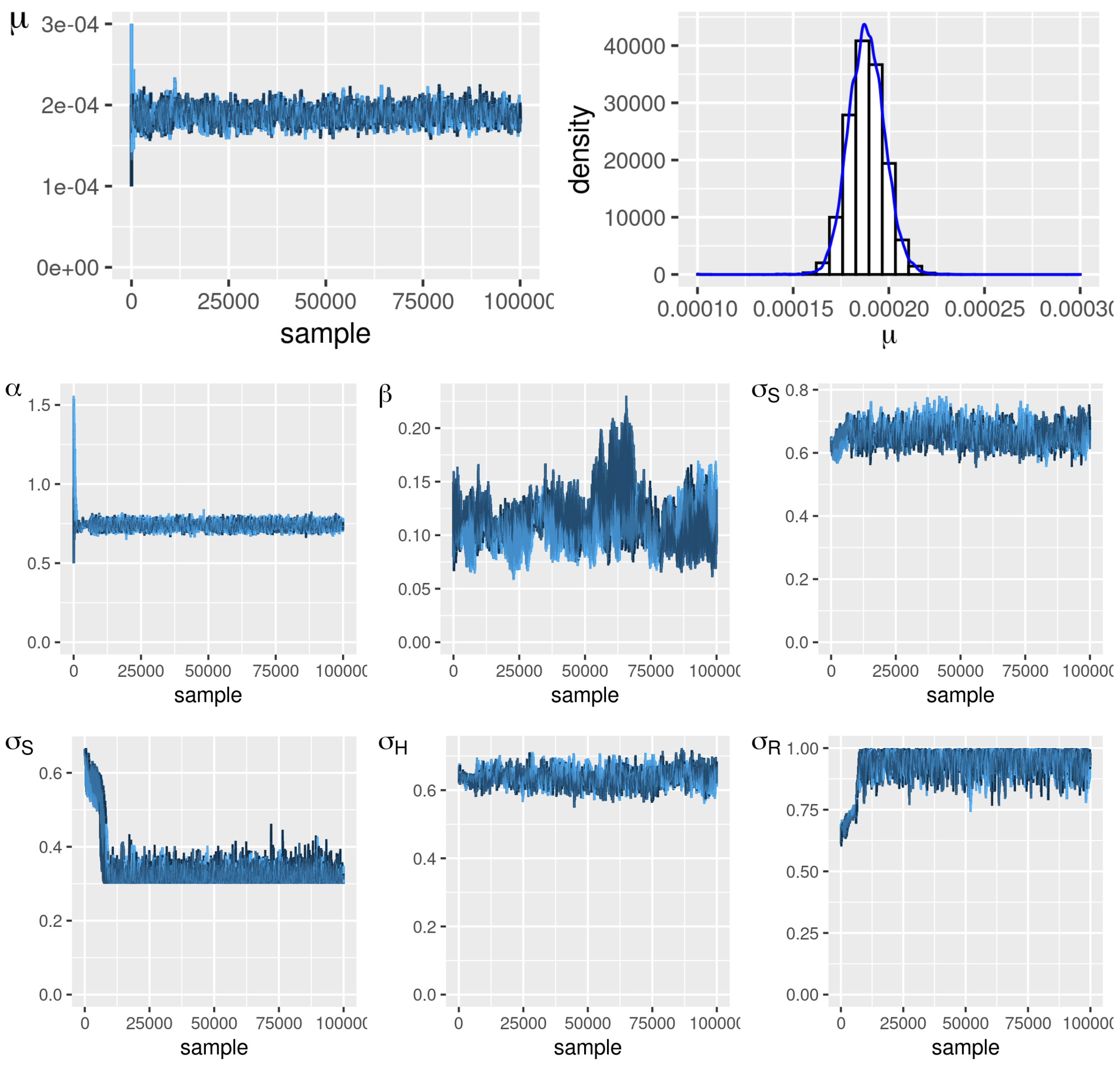
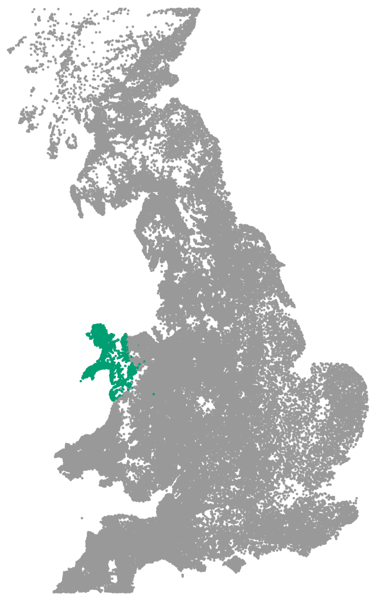


Figure E.24: Trace plots for the Wiltshire analysis.

E.1.7 Gwynedd



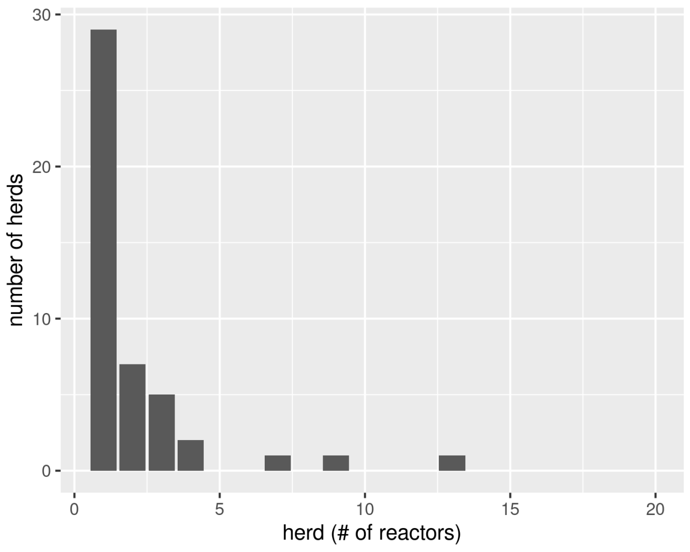


Figure E.25: Herd and reactor distribution. The latter is given for the total number reactors by herd for the entire observation period.

Number of herds: 1585
Reactor herds: 46
Number of reactors, i.e. animals: 95

σ_i	animals	+ve
1	3514	8
2	3303	1
3	5121	64
4	11	22

Figure E.26: VetNet surveillance data for selected Gwynedd herds. The diagnostic test data are categorised as follows: standard SICCT herd tests; high-sensitivity SICCT herd tests; risk-based surveillance testing; and the IFN$_\gamma$ blood test trial, denoted σ_i for 1:4 respectively.

Marginal sample distributions

Here we provide a visual summary of the parameter estimates for the basic seven-parameter model (for the Gwynedd group.) A tabulated summary is provided in Table E.7.

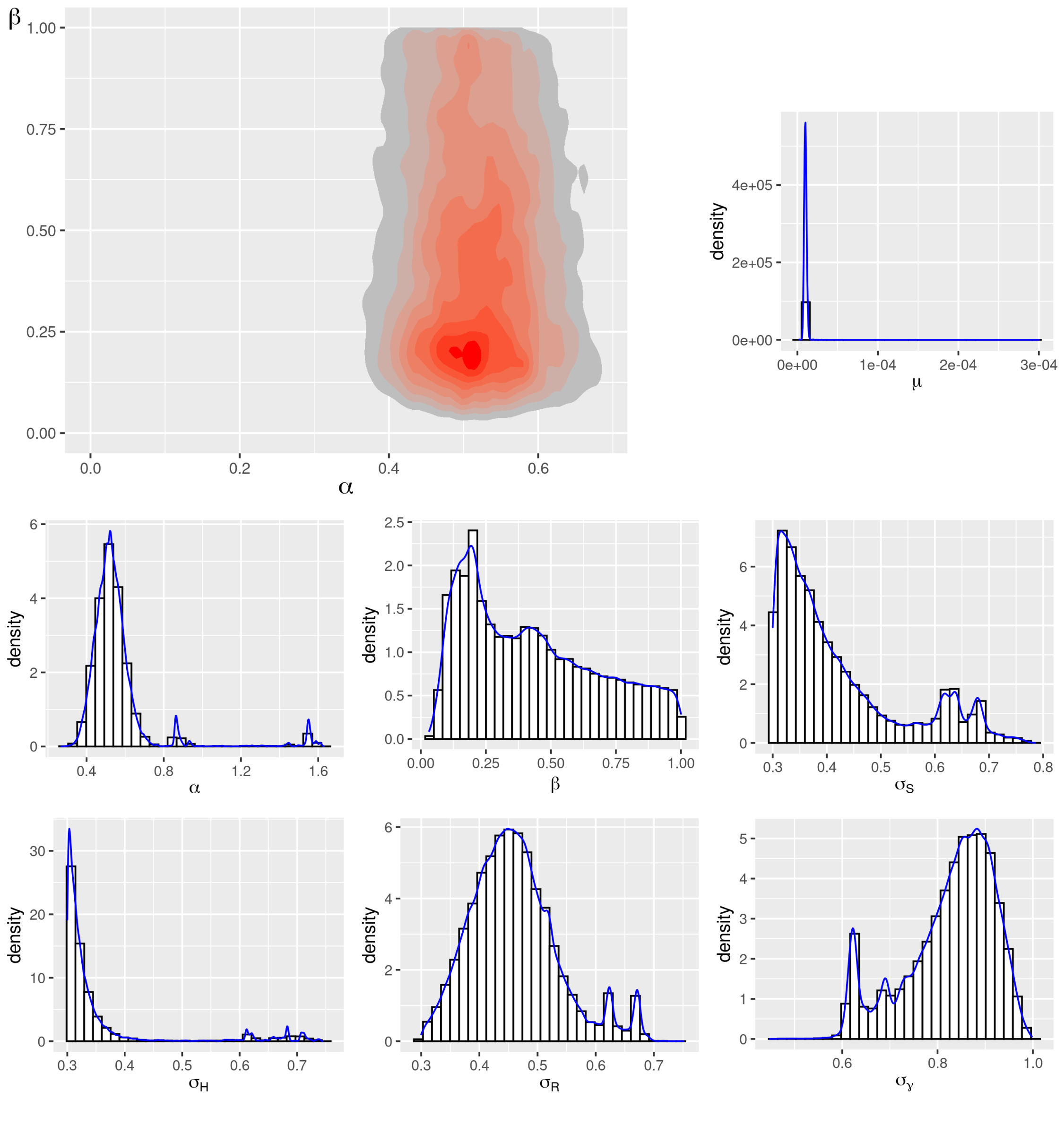


Figure E.27: Marginal sample densities for Gwynedd.

Trace plots

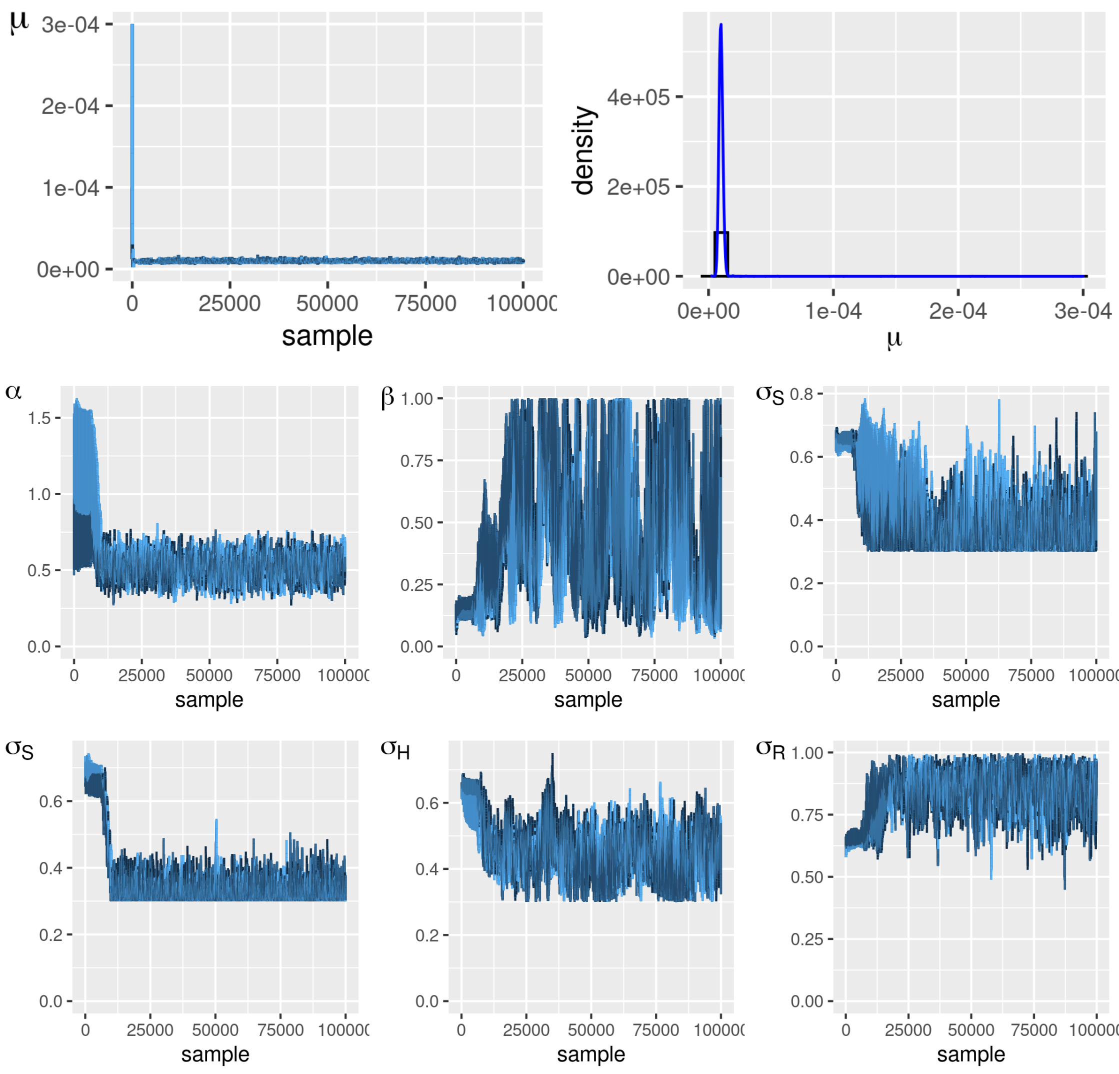
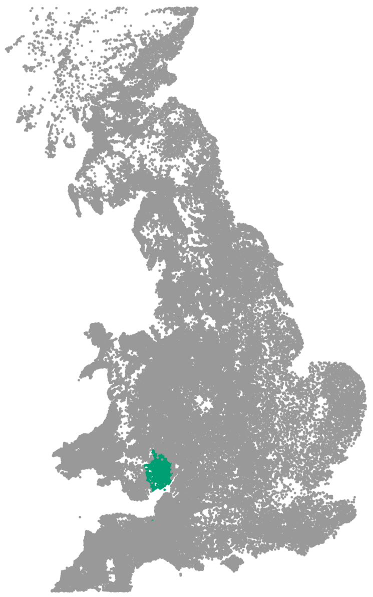


Figure E.28: Trace plots for the Gwynedd analysis.

E.1.8 Gwent



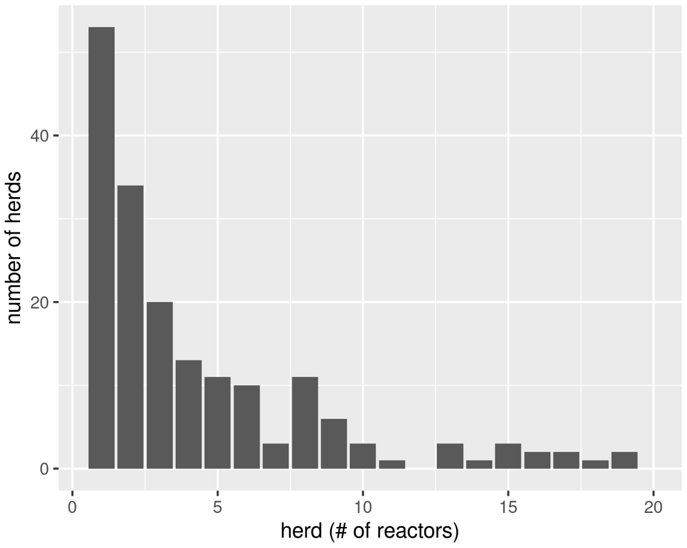


Figure E.29: Herd and reactor distribution. The latter is given for the total number reactors by herd for the entire observation period.

Number of herds: 612
Reactor herds: 204
Number of reactors, i.e. animals: 1816

σ_i	animals	+ve
1	3077	385
2	1184	23
3	3399	1336
4	63	72

Figure E.30: VetNet surveillance data for selected Gwent herds. The diagnostic test data are categorised as follows: standard SICCT herd tests; high-sensitivity SICCT herd tests; risk-based surveillance testing; and the IFN$_\gamma$ blood test trial, denoted σ_i for 1:4 respectively.

Marginal sample distributions

Here we provide a visual summary of the parameter estimates for the basic seven-parameter model (for the Gwent group.) A tabulated summary is provided in Table E.8.

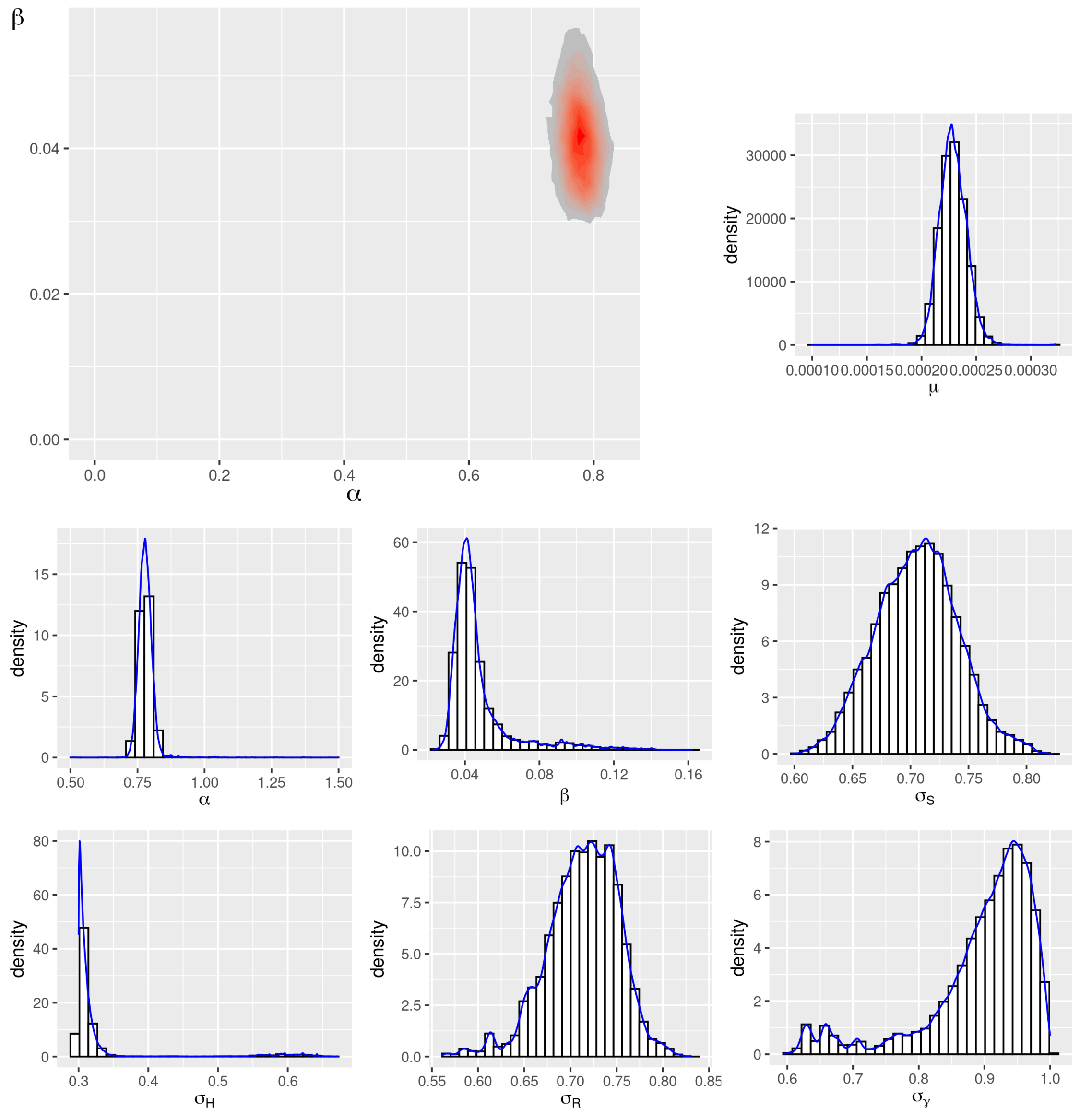


Figure E.31: Marginal sample densities for Gwent.

Trace plots

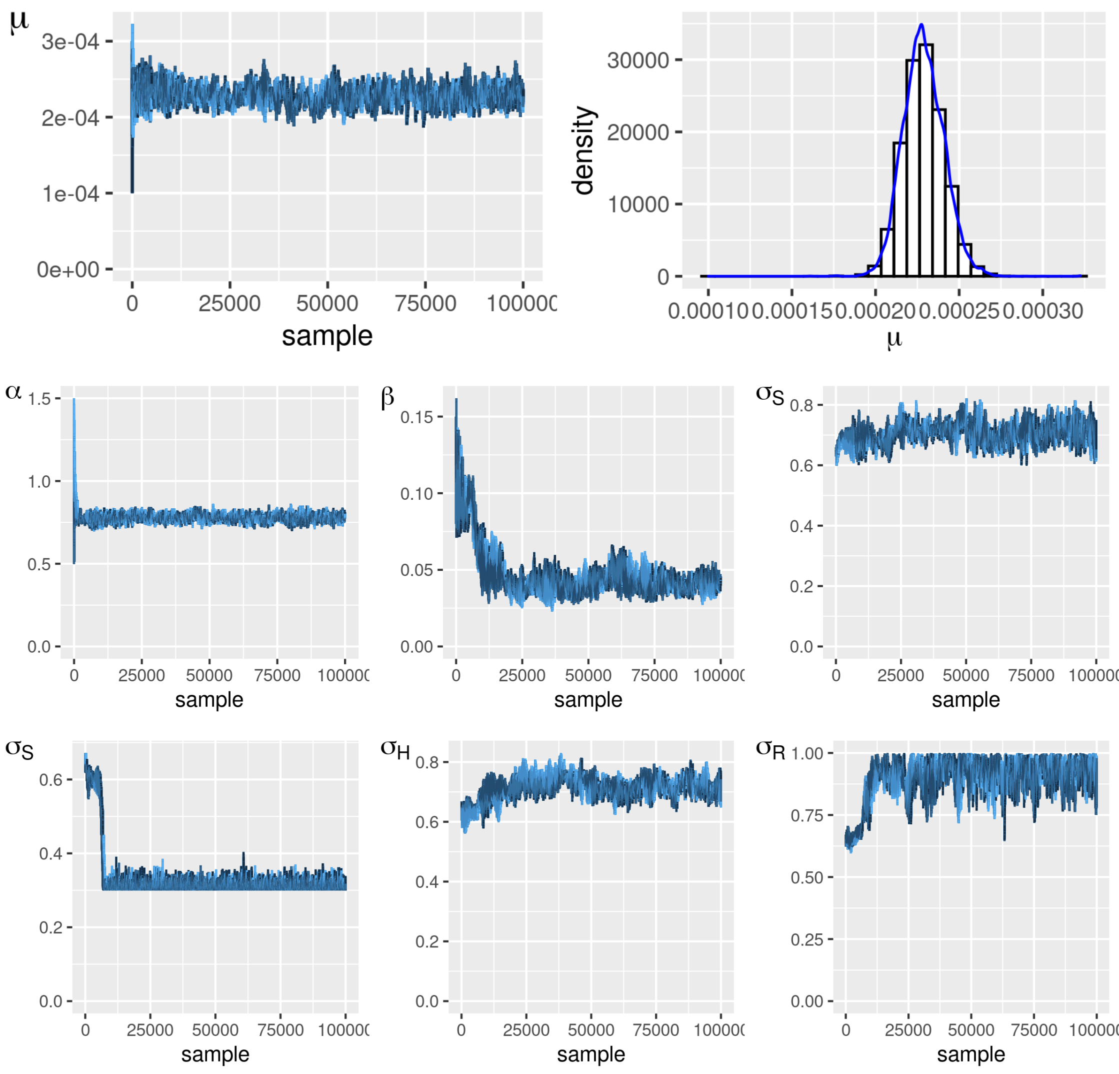
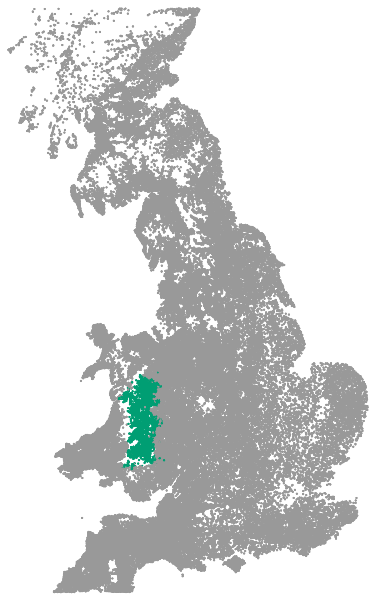


Figure E.32: Trace plots for the Gwent analysis.

E.1.9 Powys



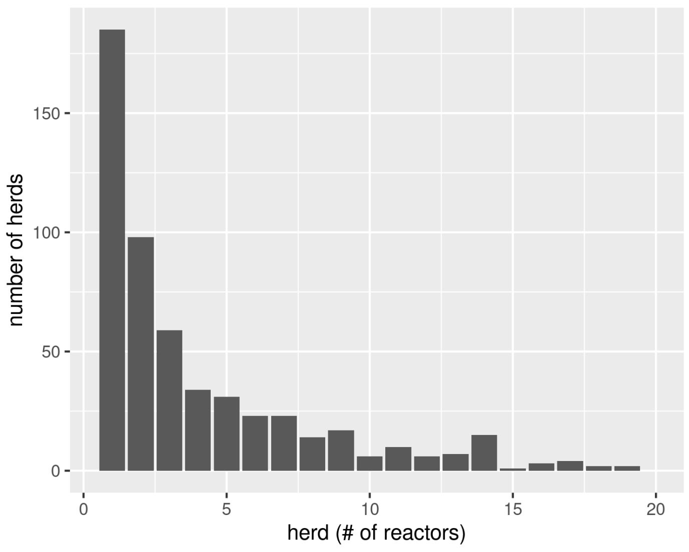


Figure E.33: Herd and reactor distribution. The latter is given for the total number reactors by herd for the entire observation period.

Number of herds: 1763
Reactor herds: 571
Number of reactors, i.e. animals: 3298

σ_i	animals	+ve
1	8612	665
2	7877	105
3	8654	2389
4	107	139

Figure E.34: VetNet surveillance data for selected Powys herds. The diagnostic test data are categorised as follows: standard SICCT herd tests; high-sensitivity SICCT herd tests; risk-based surveillance testing; and the IFN$_\gamma$ blood test trial, denoted σ_i for 1:4 respectively.

Marginal sample distributions

Here we provide a visual summary of the parameter estimates for the basic seven-parameter model (for the Powys group.) A tabulated summary is provided in Table E.9.

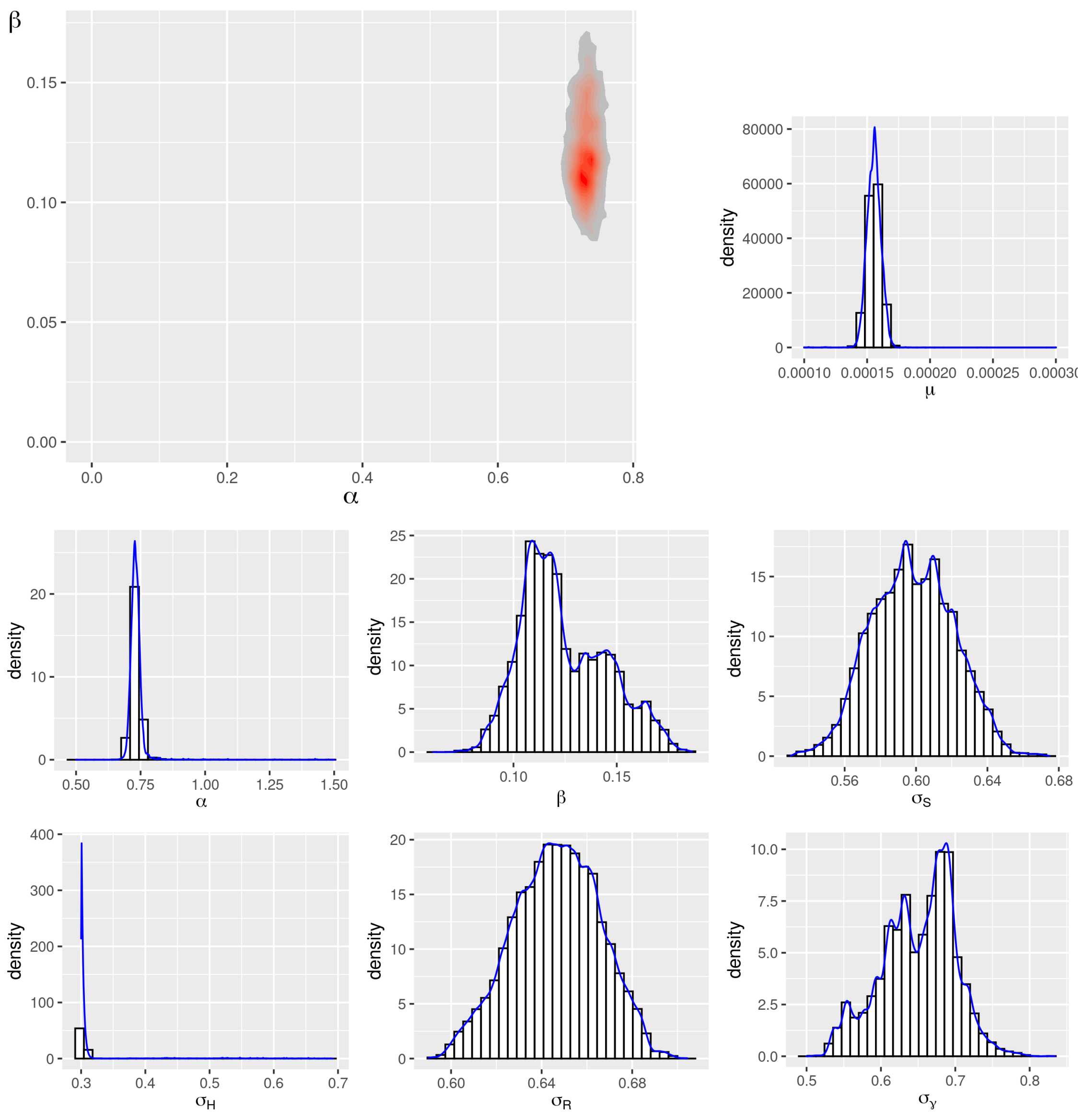


Figure E.35: Marginal sample densities for Powys.

Trace plots

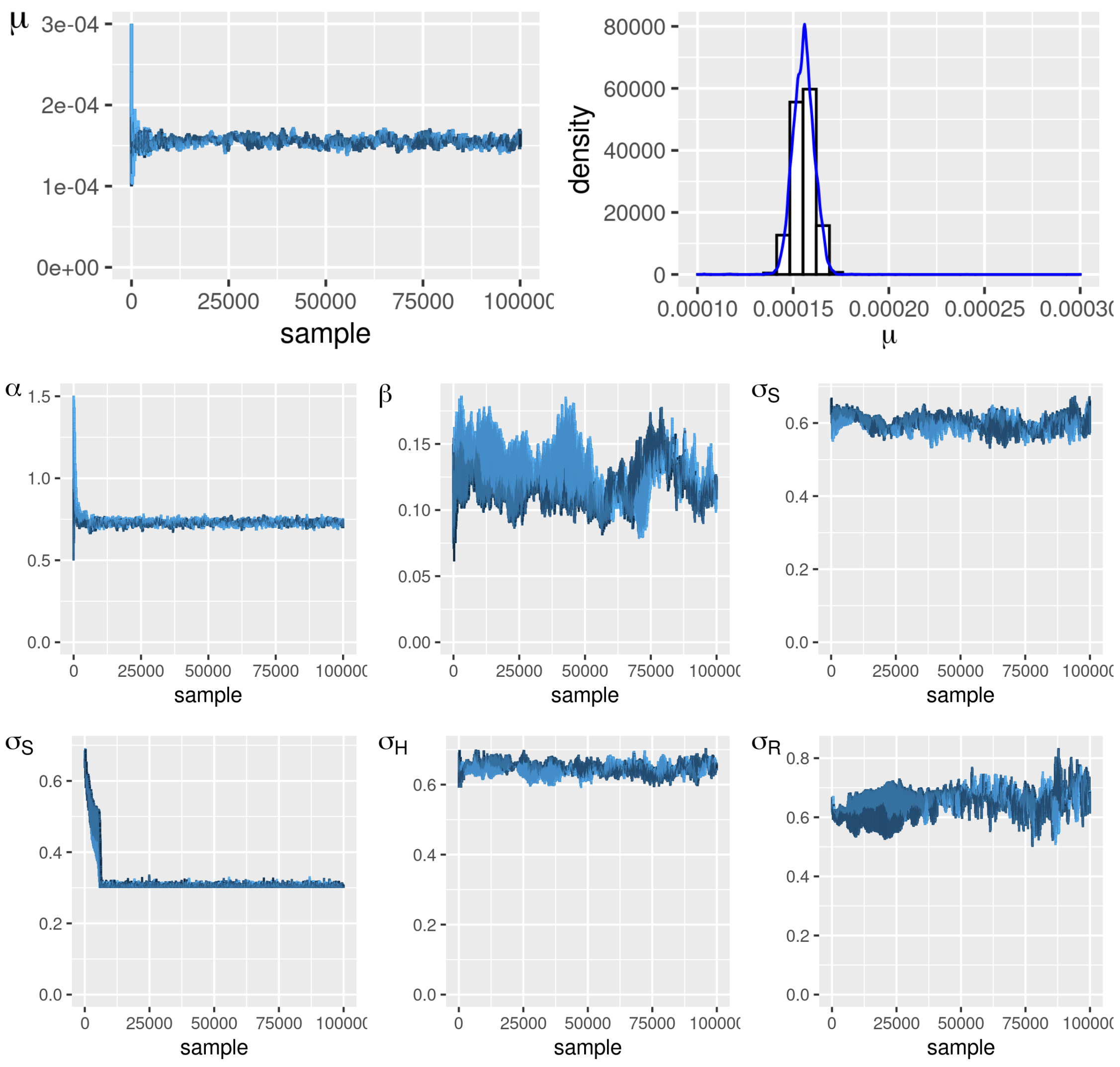


Figure E.36: Trace plots for the Powys analysis.

E.1.10 Tabulated results

θ_i	Mean	SD	$\hat{R}$	97.5
1	0.00001	0.00000	1.0	1.1
2	0.66500	0.14100	1.0	1.0
3	0.48100	0.20600	1.0	1.0
4	0.36800	0.09740	1.0	1.1
5	0.49000	0.15000	1.1	1.2
6	0.45200	0.08930	1.0	1.1
7	0.90600	0.08450	1.0	1.1
8	0.63100	0.19100	1.0	1.0

Table E.1: Standard Hawkes model parameter inference results for North east England. Number of herds (reactors) := 1545 (135.)

θ_i	Mean	SD	$\hat{R}$	97.5
1	0.00005	0.00000	1.0	1.0
2	0.80600	0.05690	1.0	1.0
3	0.13500	0.03470	1.0	1.1
4	0.53500	0.06290	1.0	1.0
5	0.33000	0.07500	1.0	1.0
6	0.70300	0.03340	1.1	1.2
7	0.90100	0.09700	1.1	1.1

Table E.2: Standard Hawkes model parameter inference results for Cheshire. Number of herds (reactors) := 1364 (955.)

θ_i	Mean	SD	$\hat{R}$	97.5
1	0.00004	0.00001	1.0	1.0
2	0.82300	0.07940	1.0	1.0
3	0.05820	0.03960	1.0	1.0
4	0.41200	0.09420	1.0	1.0
5	0.35900	0.11100	1.0	1.0
6	0.48500	0.05590	1.0	1.0
7	0.92400	0.08760	1.0	1.0

Table E.3: Standard Hawkes model parameter inference results for Oxfordshire. Number of herds (reactors) := 502 (242.)

θ_i	Mean	SD	$\hat{R}$	97.5
1	0.00014	0.00001	1.0	1.0
2	0.66200	0.04730	1.0	1.0
3	0.19000	0.05740	1.1	1.1
4	0.48000	0.04300	1.0	1.0
5	0.32900	0.06450	1.0	1.0
6	0.72300	0.03360	1.1	1.2
7	0.90100	0.09520	1.0	1.0

Table E.4: Standard Hawkes model parameter inference results for Avon. Number of herds (reactors) := 632 (737.)

θ_i	Mean	SD	$\hat{R}$	97.5
1	0.00010	0.00001	1.0	1.0
2	0.69500	0.05520	1.0	1.0
3	0.12100	0.02990	1.0	1.1
4	0.40100	0.06110	1.1	1.2
5	0.32600	0.07680	1.0	1.0
6	0.43600	0.04460	1.1	1.3
7	0.92000	0.07580	1.1	1.3

Table E.5: Standard Hawkes model parameter inference results for Dorset. Number of herds (reactors) := 971 (849.)

θ_i	Mean	SD	$\hat{R}$	97.5
1	0.00019	0.00001	1.0	1.0
2	0.74200	0.03480	1.0	1.0
3	0.11200	0.02380	1.2	1.4
4	0.66000	0.03190	1.0	1.1
5	0.33400	0.06770	1.0	1.1
6	0.63800	0.02290	1.0	1.0
7	0.92800	0.06780	1.0	1.0

Table E.6: Standard Hawkes model parameter inference results for Wiltshire. Number of herds (reactors) := 851 (1801.)

θ_i	Mean	SD	$\hat{R}$	97.5
1	0.00001	0.00000	1.0	1.0
2	0.55900	0.18800	1.0	1.0
3	0.42900	0.25800	1.1	1.3
4	0.42100	0.11500	1.0	1.1
5	0.35300	0.09710	1.0	1.0
6	0.46000	0.07500	1.2	1.5
7	0.82600	0.09310	1.0	1.0

Table E.7: Standard Hawkes model parameter inference results for Gwynedd. Number of herds (reactors) := 1585 (95.)

θ_i	Mean	SD	$\hat{R}$	97.5
1	0.00023	0.00001	1.0	1.0
2	0.77900	0.02960	1.0	1.0
3	0.04700	0.01690	1.1	1.4
4	0.70600	0.03500	1.0	1.1
5	0.32800	0.07150	1.0	1.0
6	0.71400	0.03840	1.1	1.2
7	0.89600	0.08340	1.0	1.0

Table E.8: Standard Hawkes model parameter inference results for Gwent. Number of herds (reactors) := 612 (1816.)

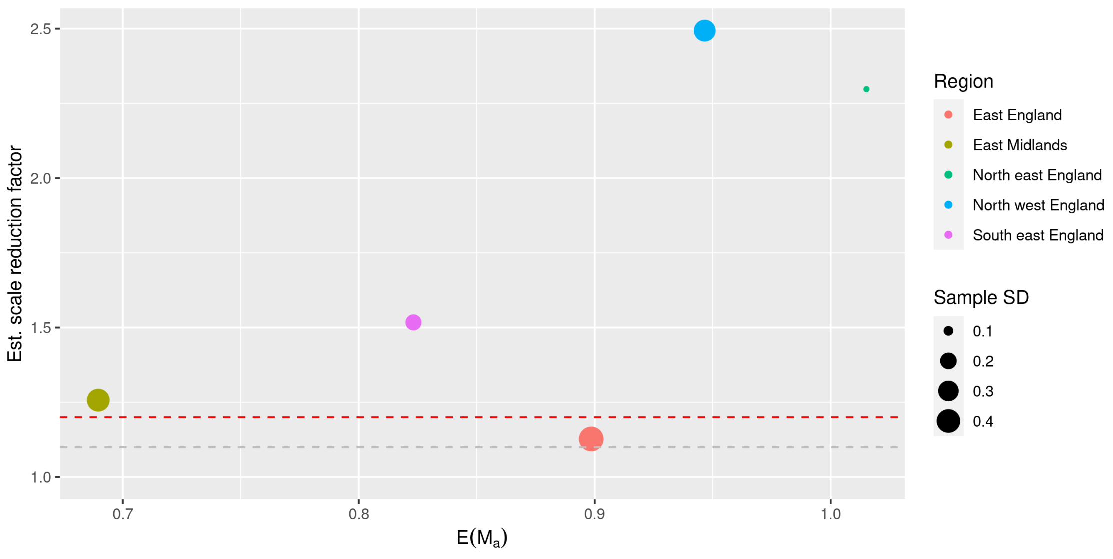
θ_i	Mean	SD	$\hat{R}$	97.5
1	0.00015	0.00001	1.0	1.1
2	0.73200	0.03530	1.0	1.0
3	0.12400	0.02070	1.1	1.4
4	0.59800	0.02270	1.2	1.5
5	0.31500	0.05220	1.0	1.0
6	0.64600	0.01900	1.0	1.0
7	0.65000	0.04900	1.1	1.4

Table E.9: Standard Hawkes model parameter inference results for Powys. Number of herds (reactors) := 1763 (3298.)

E.2 Homogeneous disease process-hierarchical model results

E.2.1 Summaries

Here we report the summaries of posterior samples for parameters from the second analysis that not were not included in the main text.



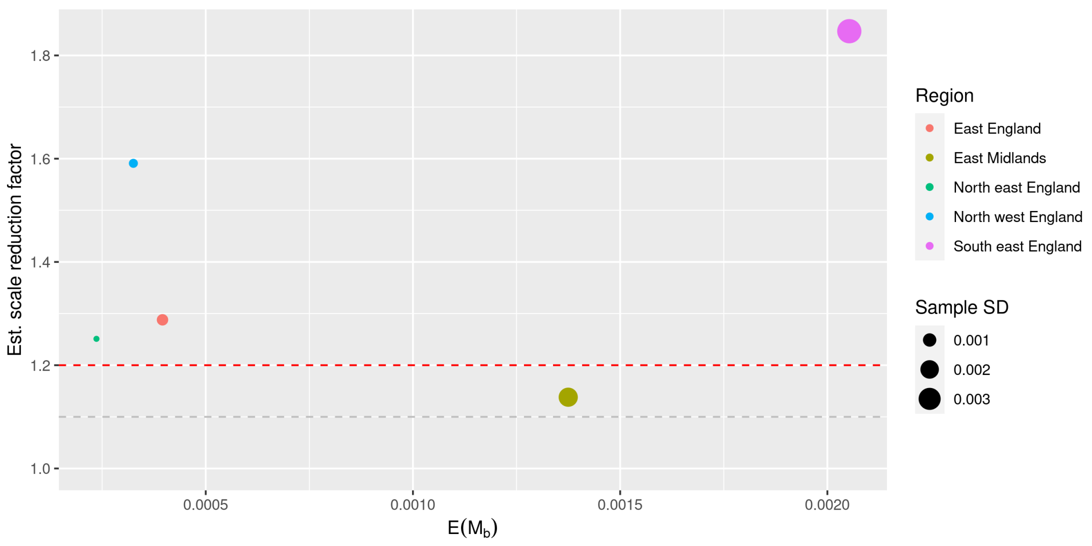
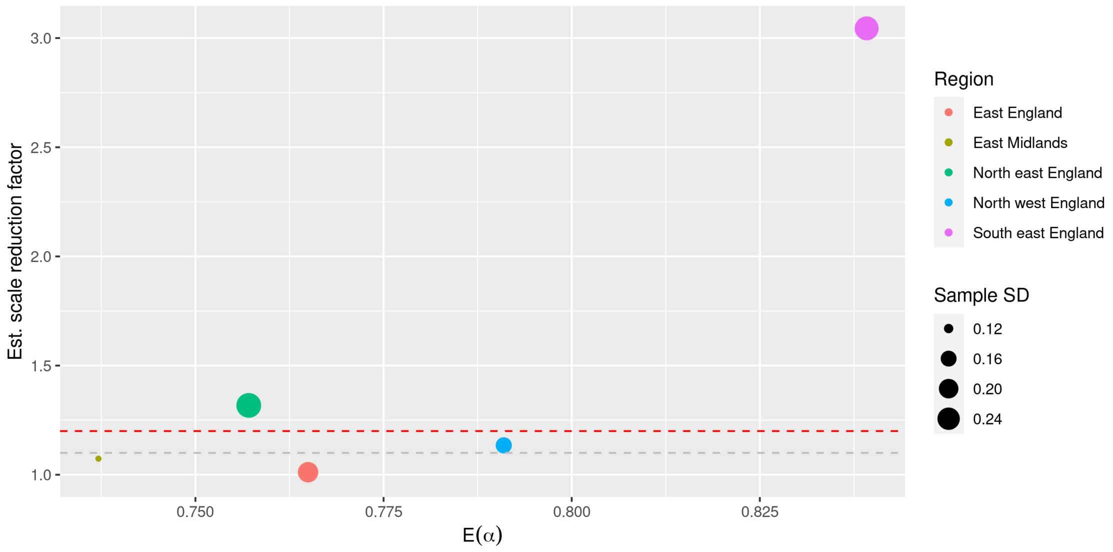


Figure E.37: Constant FOI hierarchical parameter estimates.



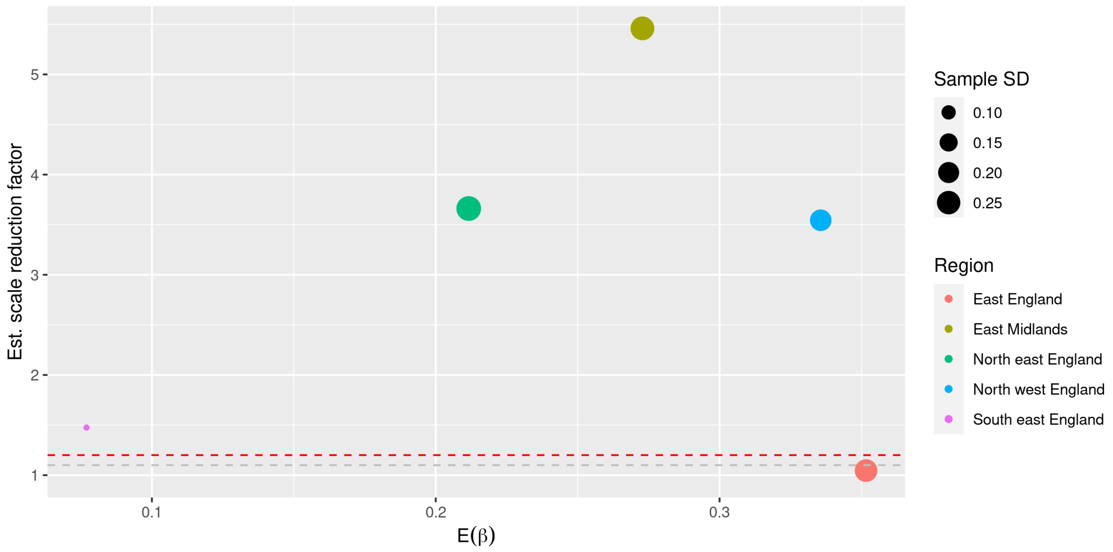
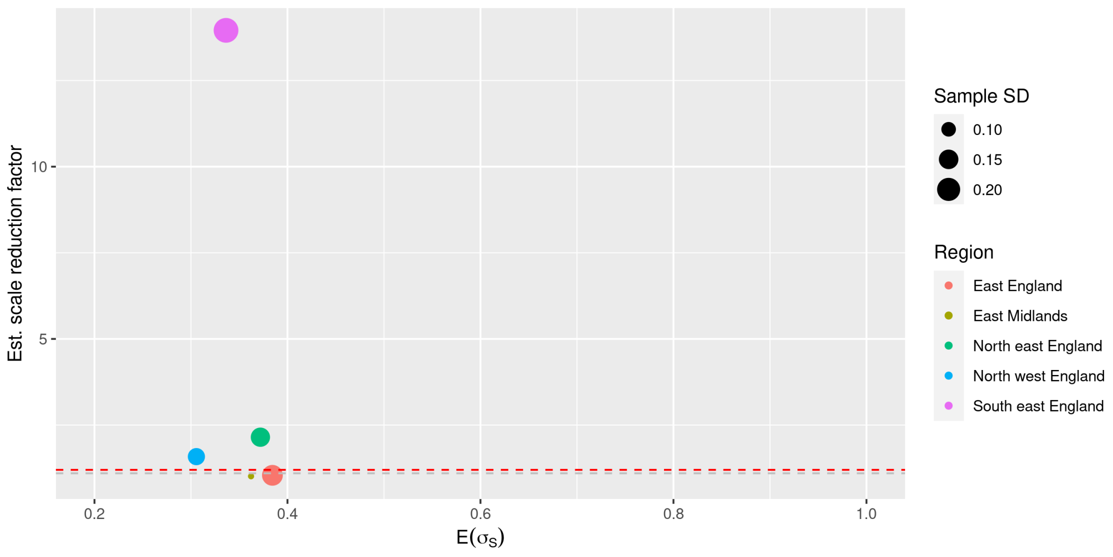
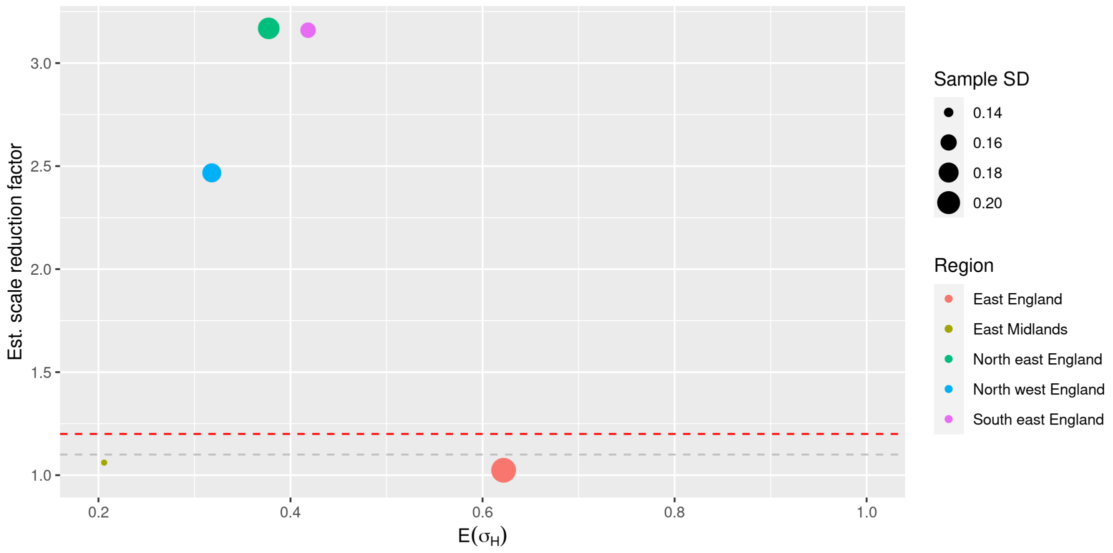


Figure E.38: Disease process parameter estimates $\{\alpha, \beta\}$.

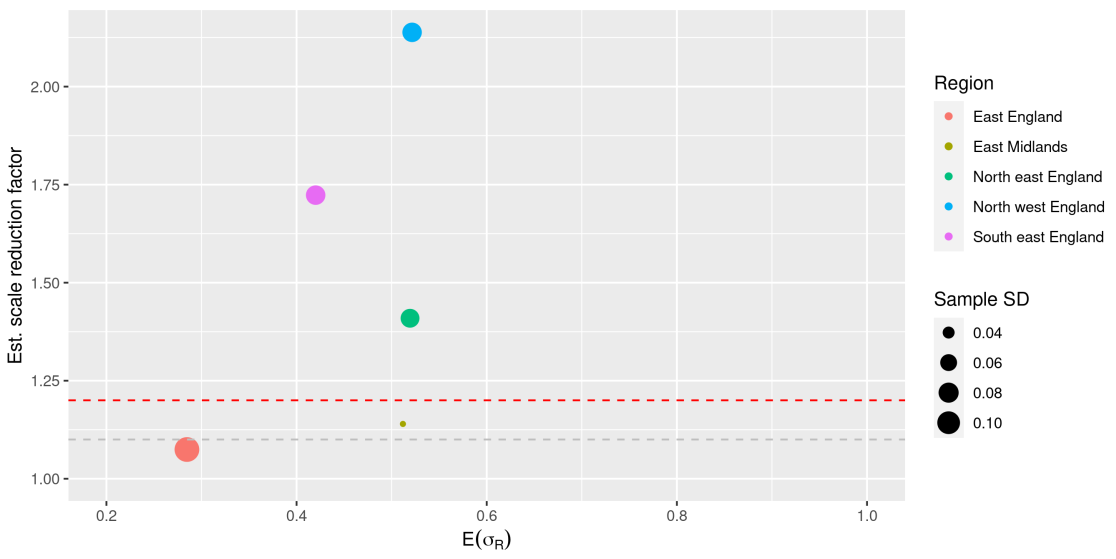


(a) Standard sensitivity.

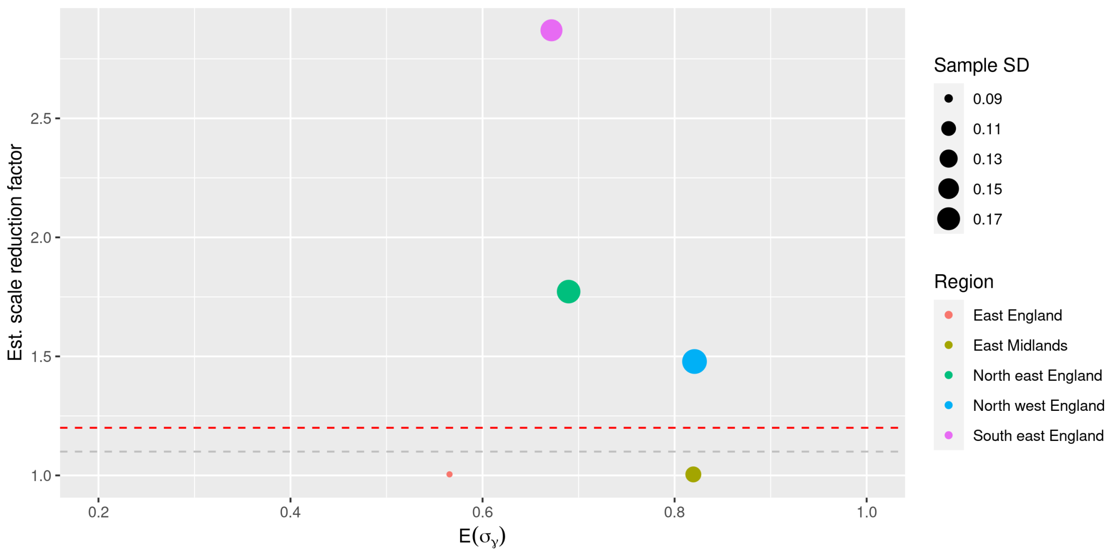


(b) High sensitivity.

Figure E.39: Diagnostic test sensitivity parameter estimates: SICCT test.



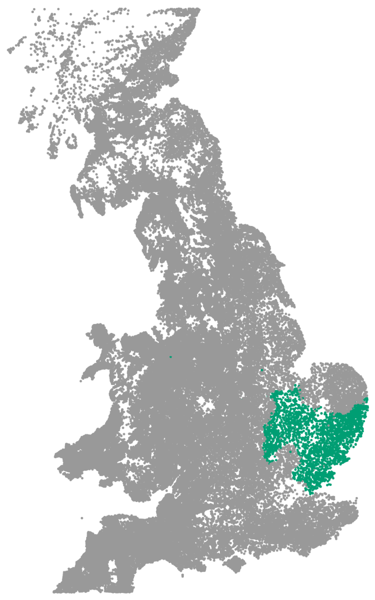
(a) Trade-related testing.



(b) IFN$_\gamma$ blood test.

Figure E.40: Diagnostic test sensitivity parameter estimates: other.

E.2.2 East England



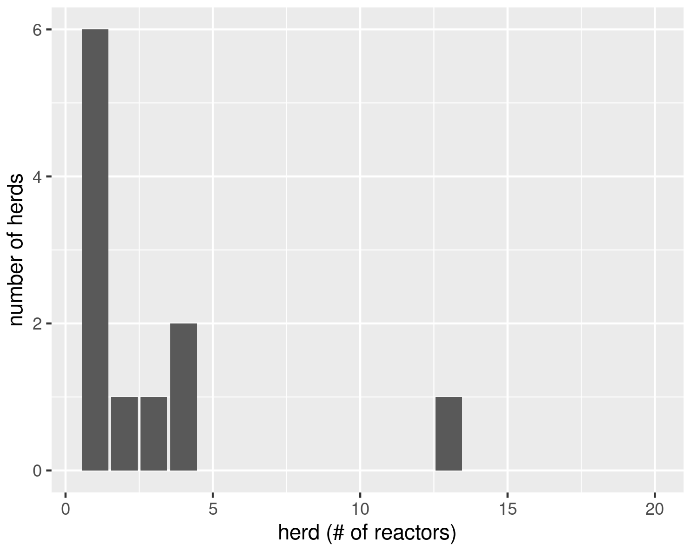


Figure E.41: Herd and reactor distribution. The latter is given for the total number reactors by herd for the entire observation period.

Number of herds: 513
Reactor herds: 13
Number of reactors, i.e. animals: 75

σ_i	animals	+ve
1	981	11
2	93	2
3	1619	29
4	20	33

Figure E.42: VetNet surveillance data for selected East England herds. The diagnostic test data are categorised as follows: standard SICCT herd tests; high-sensitivity SICCT herd tests; risk-based surveillance testing; and the IFN$_\gamma$ blood test trial, denoted σ_i for 1:4 respectively.

Marginal sample distributions

Here we provide a visual summary of the parameter estimates for the hierarchical model with homogeneous disease processes (for the East England group.) A tabulated summary is provided in Table E.10.

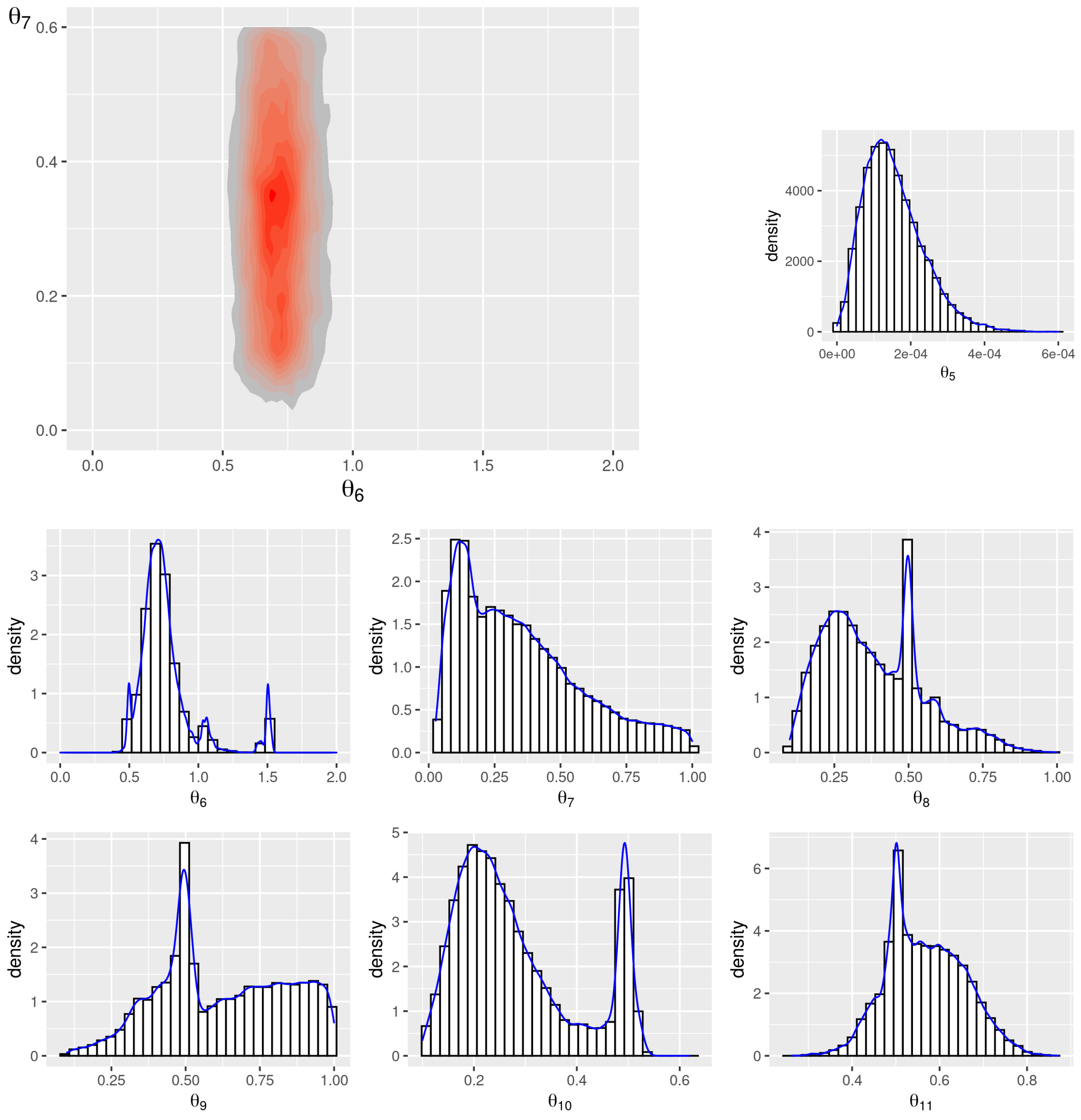


Figure E.43: Marginal sample densities for East England.

Trace plots

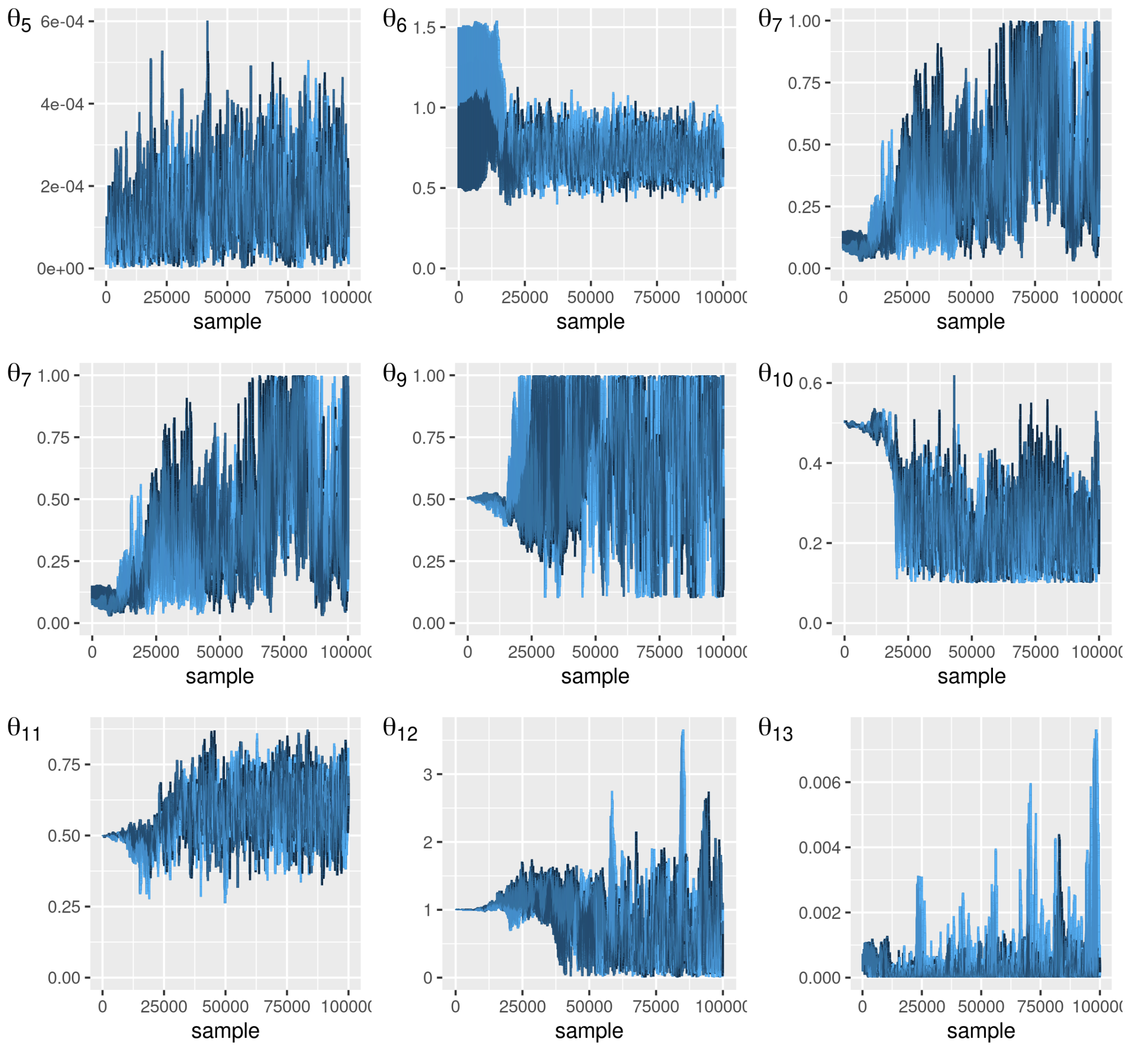
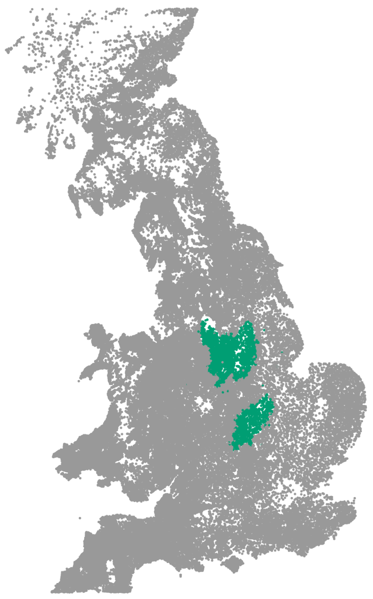


Figure E.44: Trace plots for the East England analysis.

E.2.3 East Midlands



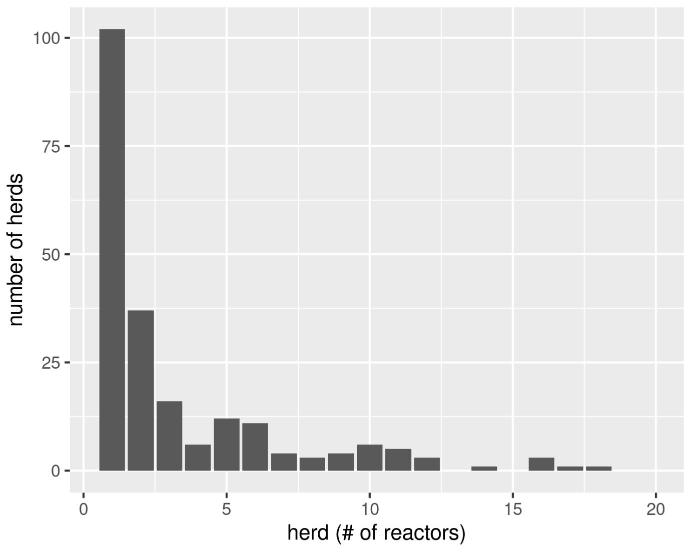


Figure E.45: Herd and reactor distribution. The latter is given for the total number reactors by herd for the entire observation period.

Number of herds: 1966
Reactor herds: 231
Number of reactors, i.e. animals: 1217

σ_i	animals	+ve
1	5100	181
2	6142	17
3	7475	890
4	40	129

Figure E.46: VetNet surveillance data for selected East Midlands herds. The diagnostic test data are categorised as follows: standard SICCT herd tests; high-sensitivity SICCT herd tests; risk-based surveillance testing; and the IFN$_\gamma$ blood test trial, denoted σ_i for 1:4 respectively.

Marginal sample distributions

Here we provide a visual summary of the parameter estimates for the hierarchical model with homogeneous disease processes (for the East Midlands group.) A tabulated summary is provided in Table E.11.

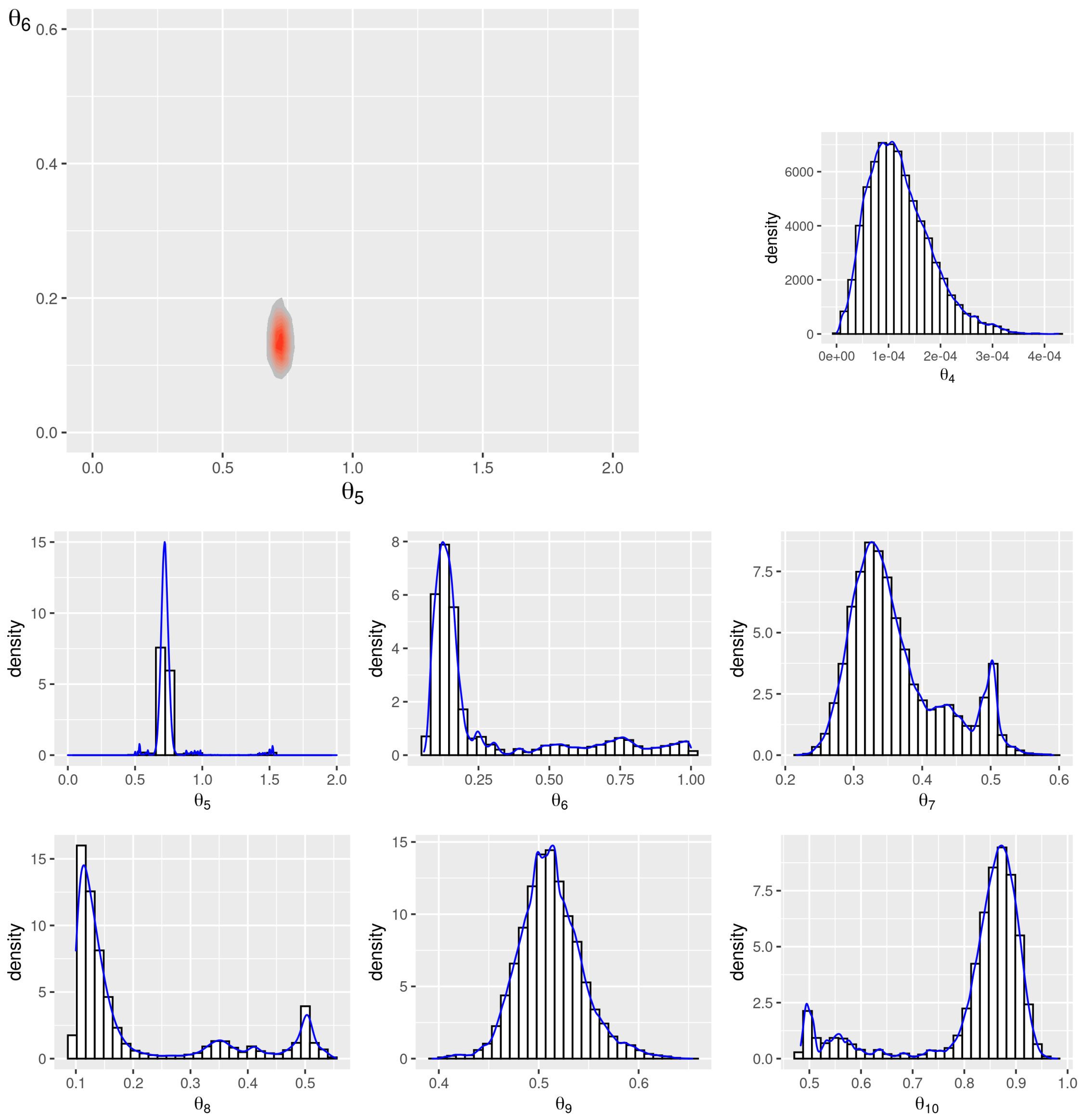


Figure E.47: Marginal sample densities for East Midlands.

Trace plots

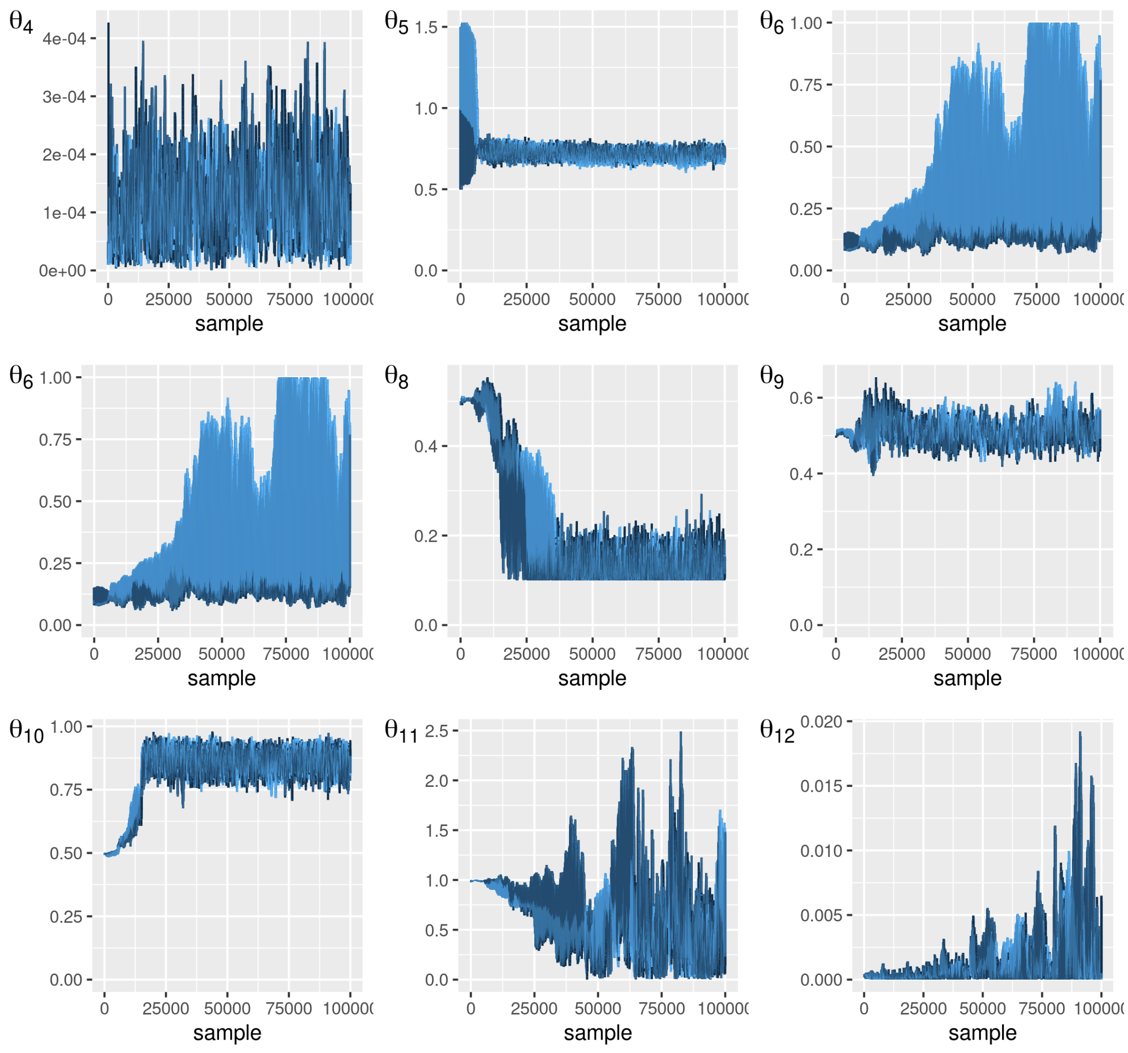
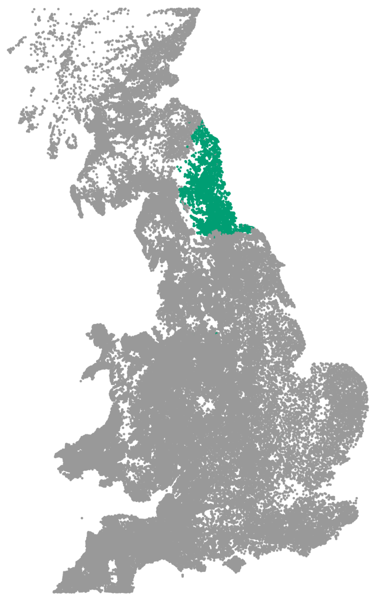


Figure E.48: Trace plots for the East Midlands analysis.

E.2.4 North east England



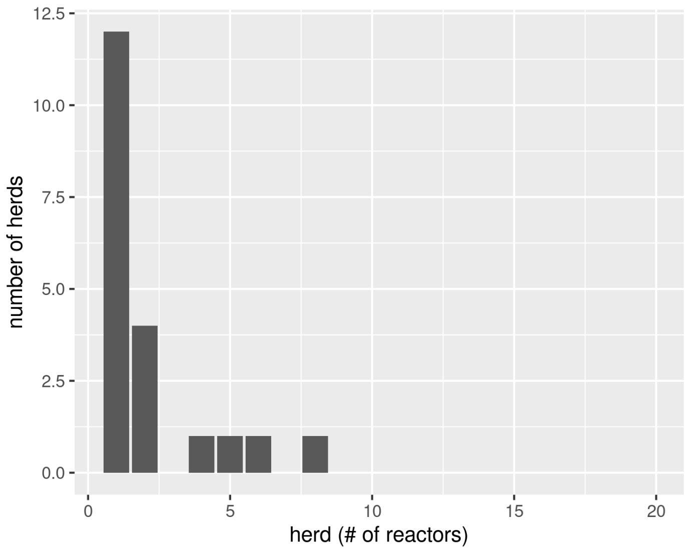


Figure E.49: Herd and reactor distribution. The latter is given for the total number reactors by herd for the entire observation period.

Number of herds: 1015
Reactor herds: 22
Number of reactors, i.e. animals: 103

σ_i	animals	+ve
1	2383	14
2	691	1
3	2017	30
4	8	58

Figure E.50: VetNet surveillance data for selected North east England herds. The diagnostic test data are categorised as follows: standard SICCT herd tests; high-sensitivity SICCT herd tests; risk-based surveillance testing; and the IFN$_\gamma$ blood test trial, denoted σ_i for 1:4 respectively.

Marginal sample distributions

Here we provide a visual summary of the parameter estimates for the hierarchical model with homogeneous disease processes (for the North east England group.) A tabulated summary is provided in Table E.12.

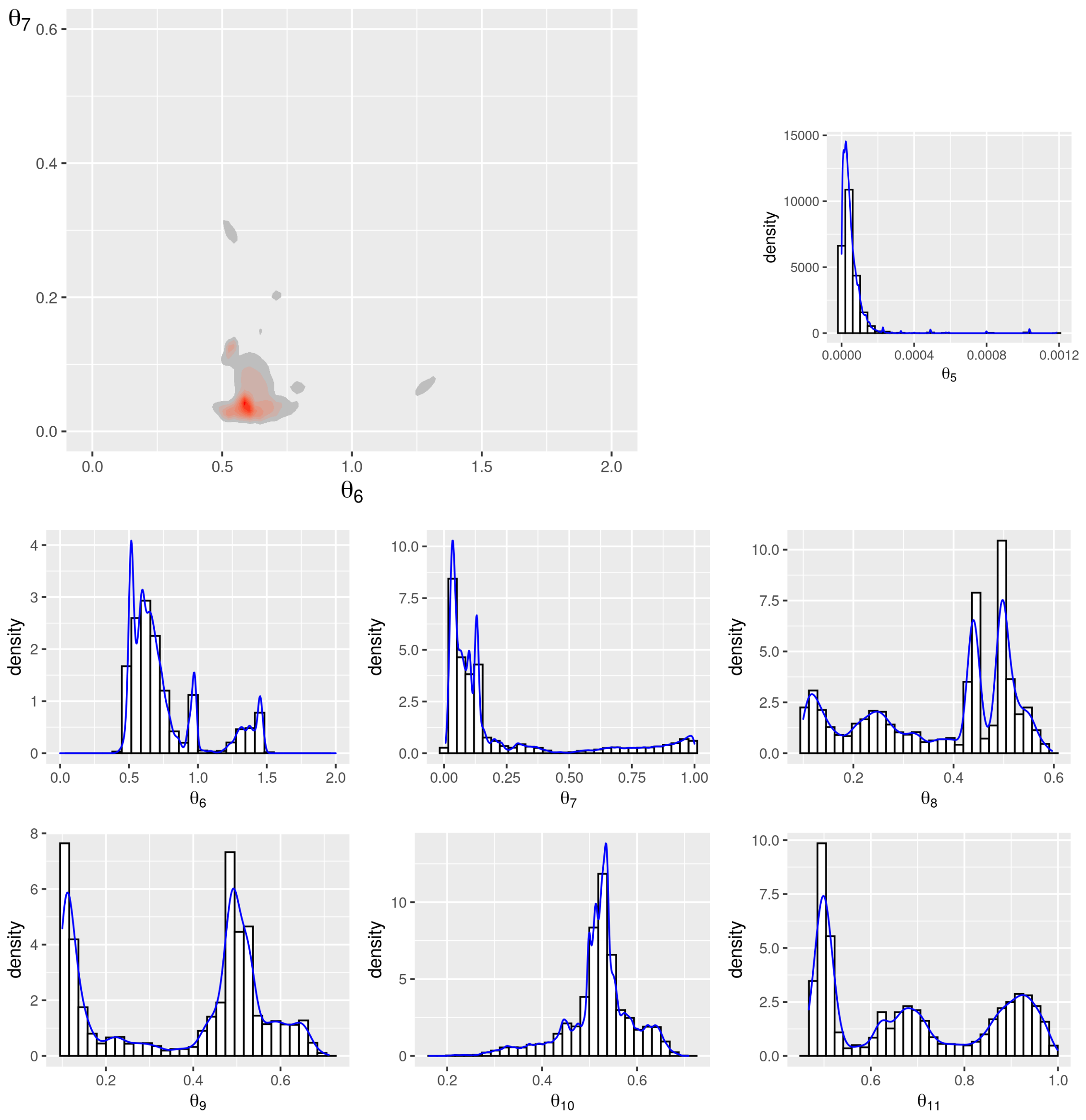


Figure E.51: Marginal sample densities for North east England.

Trace plots

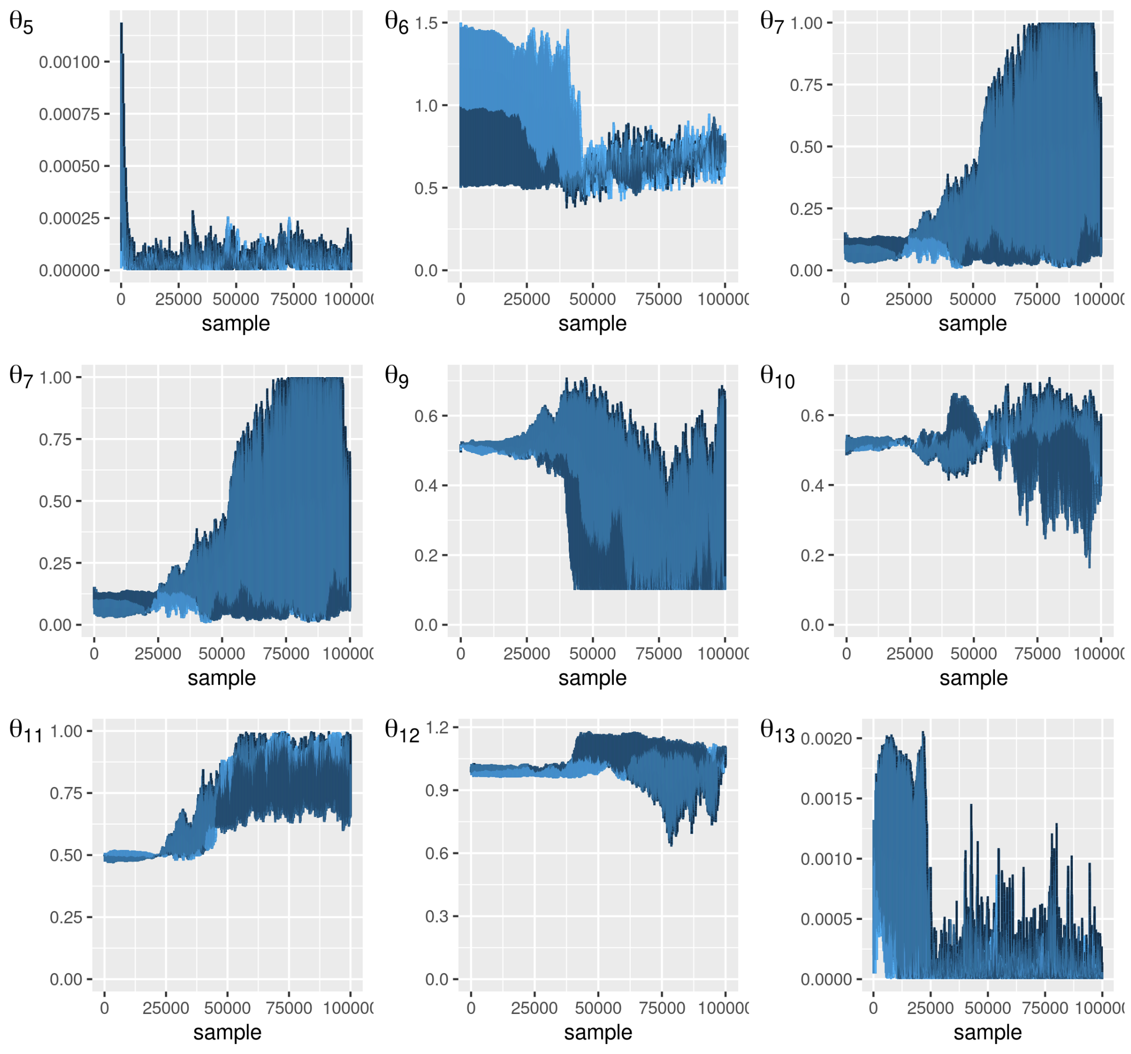


Figure E.52: Trace plots for the North east England analysis.

E.2.5 North west England

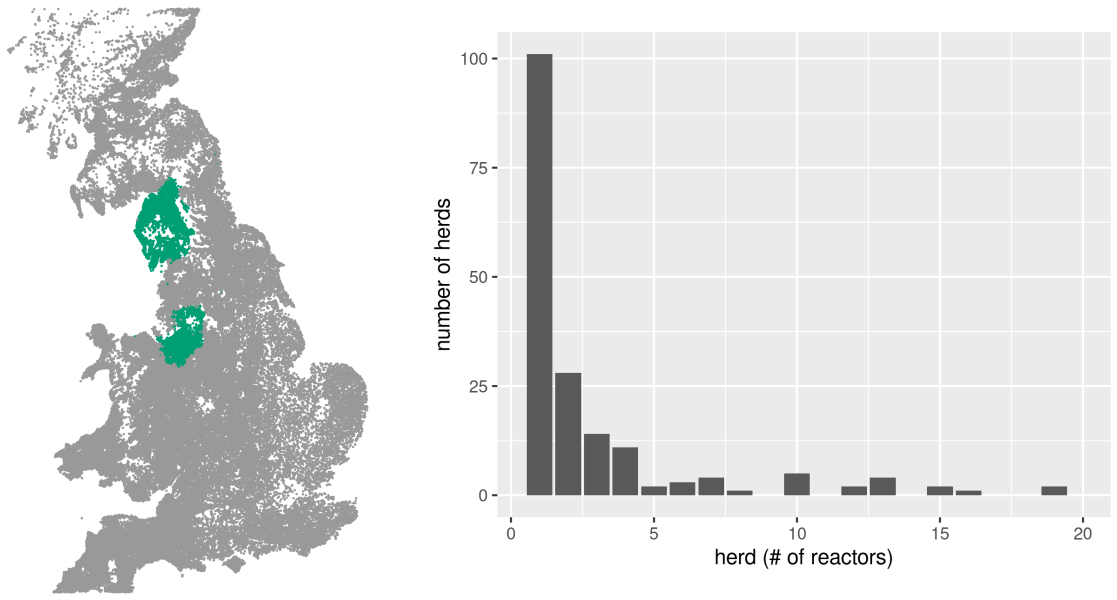


Figure E.53: Herd and reactor distribution. The latter is given for the total number reactors by herd for the entire observation period.

Number of herds: 3432
Reactor herds: 190
Number of reactors, i.e. animals: 989

σ_i	animals	+ve
1	7953	97
2	12540	27
3	8972	694
4	45	171

Figure E.54: VetNet surveillance data for selected North west England herds. The diagnostic test data are categorised as follows: standard SICCT herd tests; high-sensitivity SICCT herd tests; risk-based surveillance testing; and the IFN$_\gamma$ blood test trial, denoted σ_i for 1:4 respectively.

Marginal sample distributions

Here we provide a visual summary of the parameter estimates for the hierarchical model with homogeneous disease processes (for the North west England group.) A tabulated summary is provided in Table E.13.

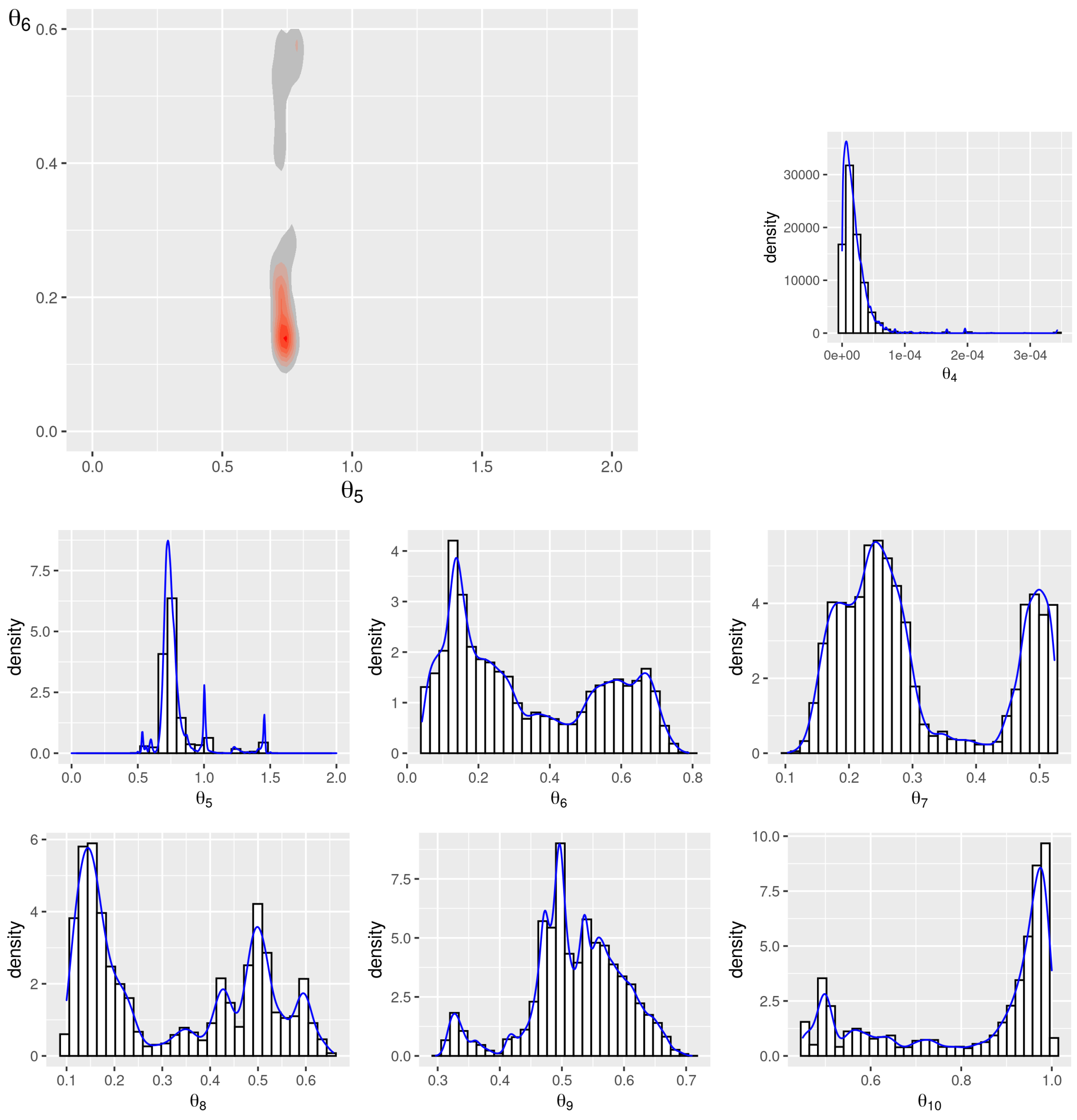


Figure E.55: Marginal sample densities for North west England.

Trace plots

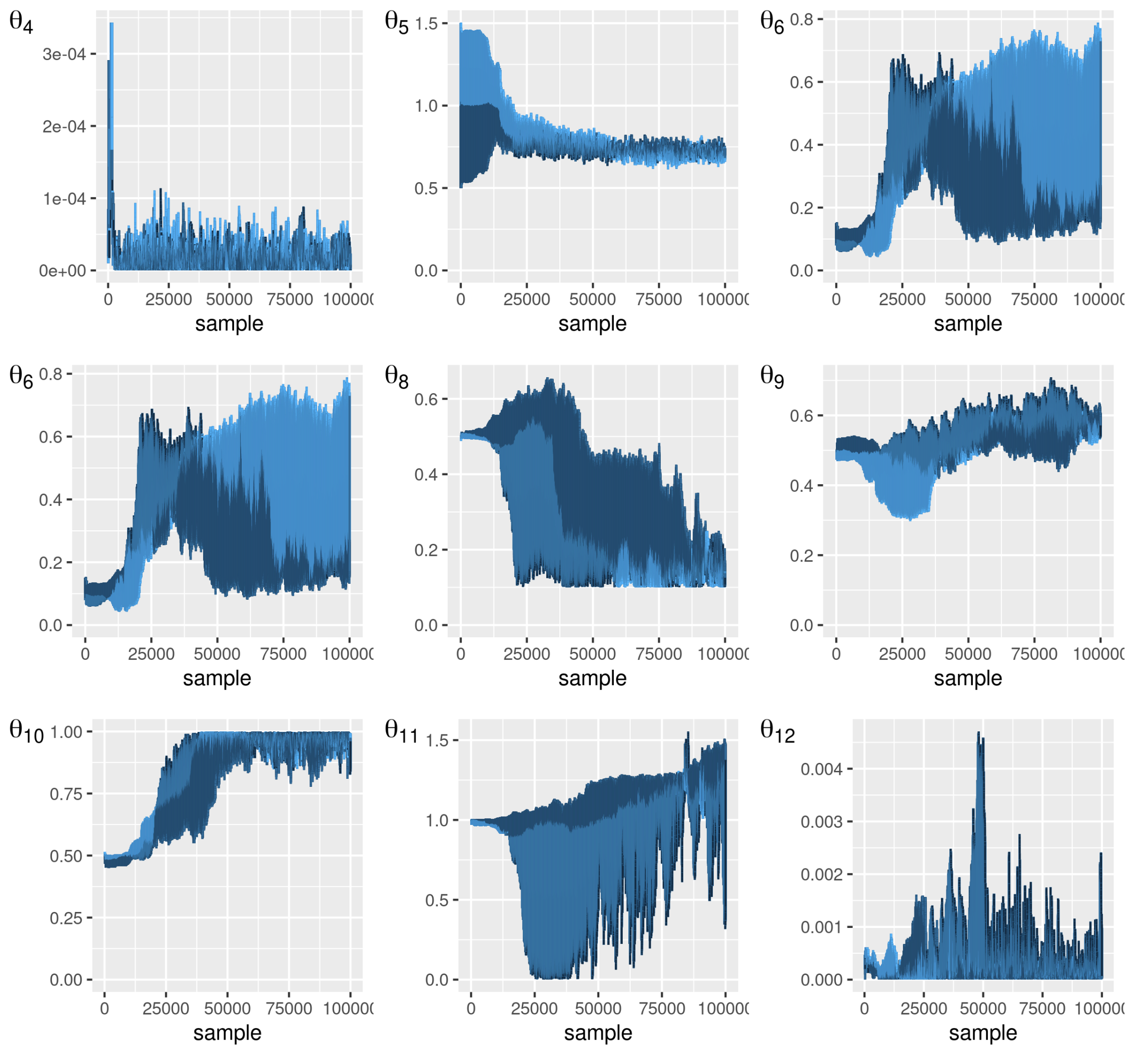


Figure E.56: Trace plots for the North west England analysis.

E.2.6 South east England

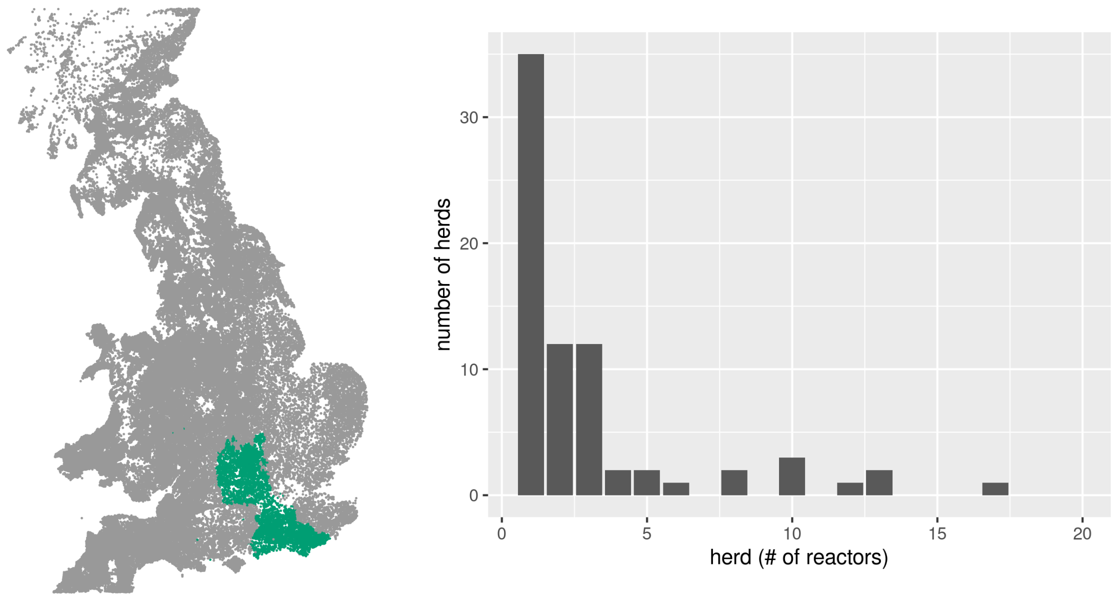


Figure E.57: Herd and reactor distribution. The latter is given for the total number reactors by herd for the entire observation period.

Number of herds: 1393
Reactor herds: 78
Number of reactors, i.e. animals: 357

σ_i	animals	+ve
1	3102	36
2	1664	9
3	3786	246
4	37	66

Figure E.58: VetNet surveillance data for selected South east England herds. The diagnostic test data are categorised as follows: standard SICCT herd tests; high-sensitivity SICCT herd tests; risk-based surveillance testing; and the IFN$_\gamma$ blood test trial, denoted σ_i for 1:4 respectively.

Marginal sample distributions

Here we provide a visual summary of the parameter estimates for the hierarchical model with homogeneous disease processes (for the South east England group.) A tabulated summary is provided in Table E.14.

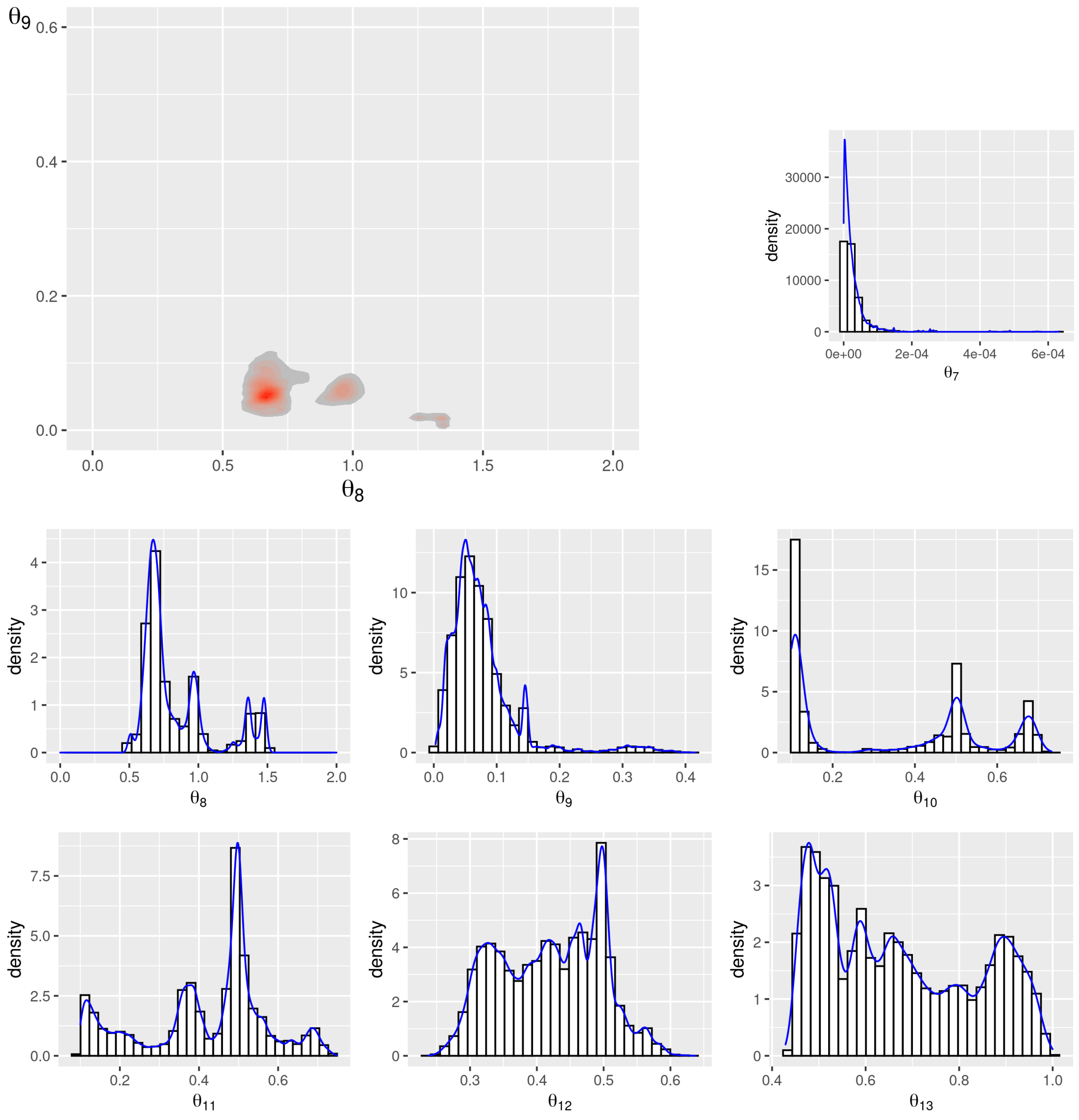


Figure E.59: Marginal sample densities for South east England.

Trace plots

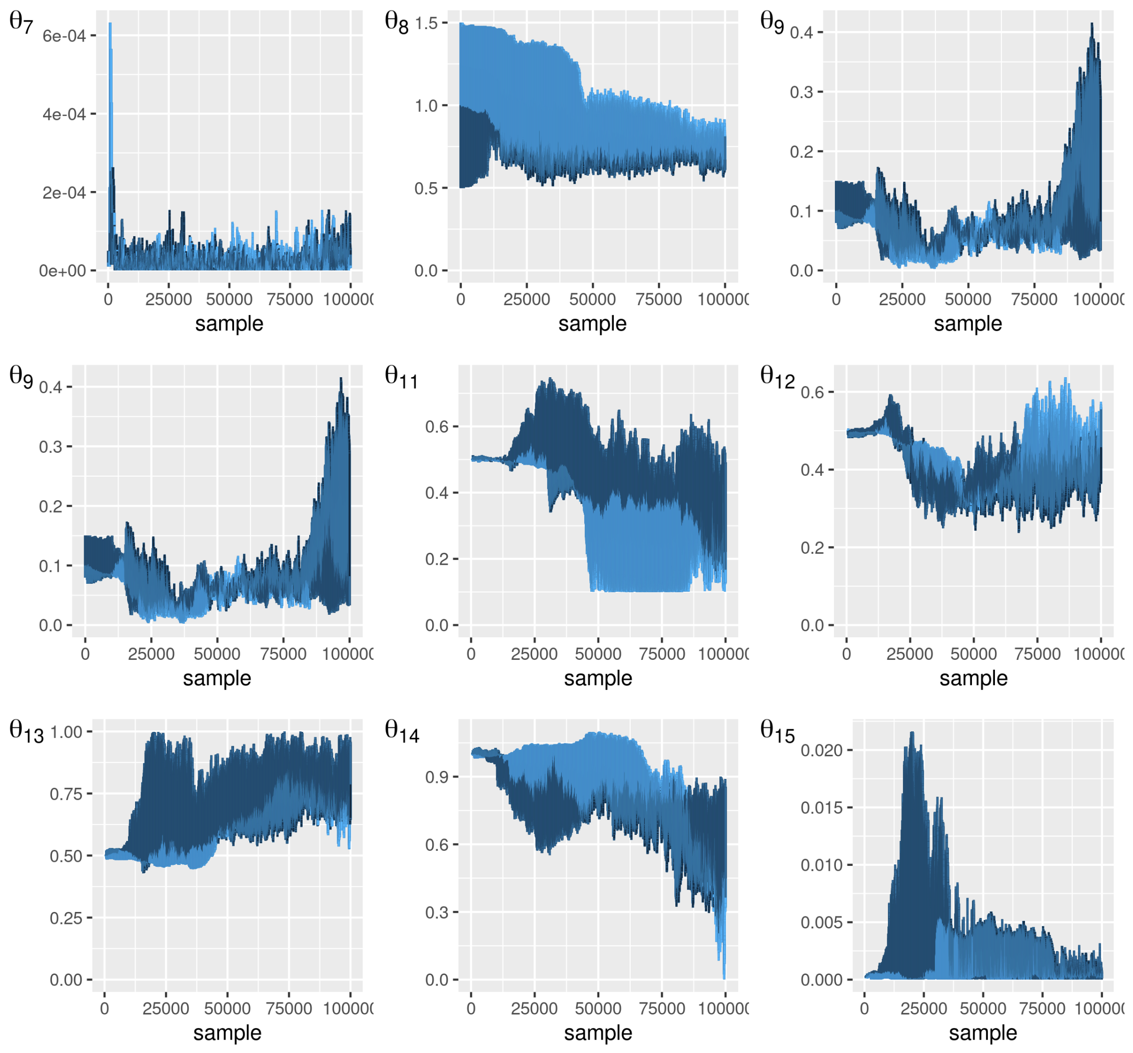


Figure E.60: Trace plots for the South east England analysis.

E.2.7 Scotland

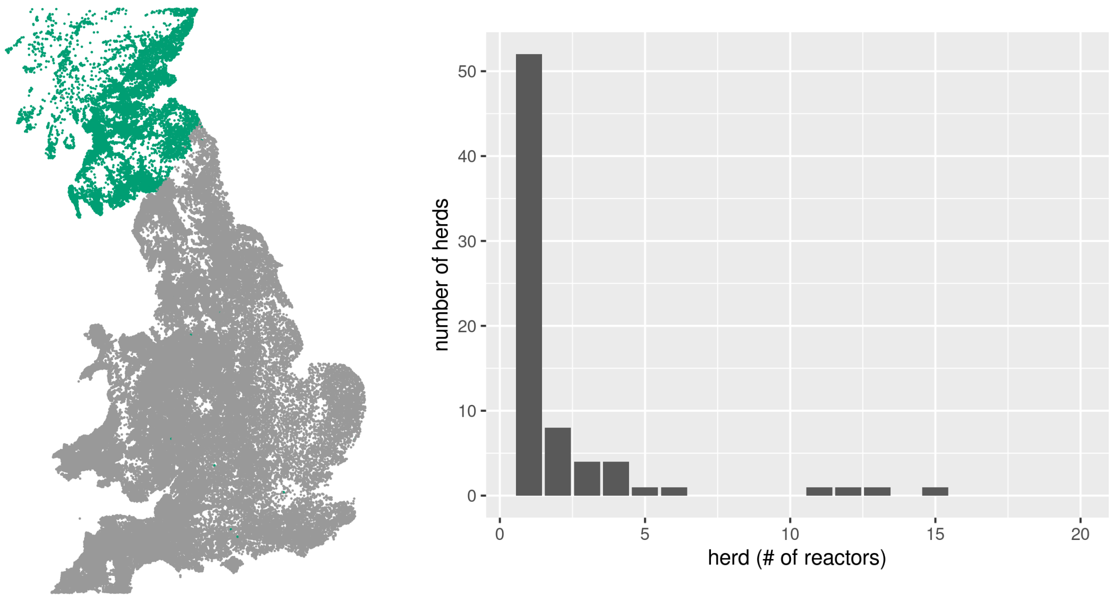


Figure E.61: Herd and reactor distribution. The latter is given for the total number reactors by herd for the entire observation period.

Number of herds: 4510

Reactor herds: 77

Number of reactors, i.e. animals: 280

σ_i	animals	+ve
1	11053	19
2	3450	3
3	4884	176
4	24	82

Figure E.62: VetNet surveillance data for selected Scotland herds. The diagnostic test data are categorised as follows: standard SICCT herd tests; high-sensitivity SICCT herd tests; risk-based surveillance testing; and the IFN$_\gamma$ blood test trial, denoted σ_i for 1:4 respectively.

Marginal sample distributions

Here we provide a visual summary of the parameter estimates for the hierarchical model with homogeneous disease processes (for the Scotland group.) A tabulated summary is provided in Table E.15. Note that the algorithm achieved a poor degree of convergence for many parameters, as indicated by $\hat{R}$.

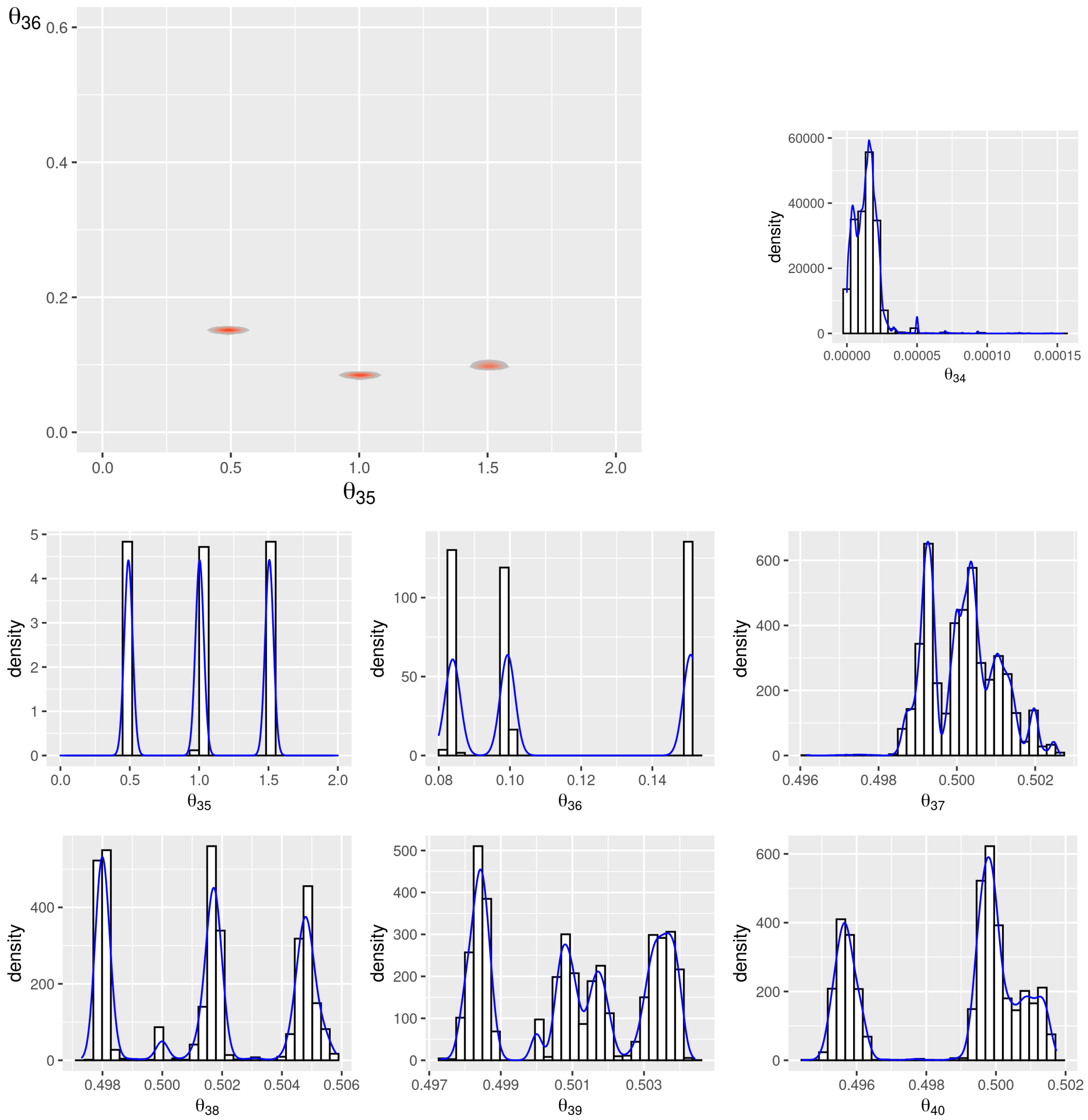


Figure E.63: Marginal sample densities for Scotland.

Trace plots

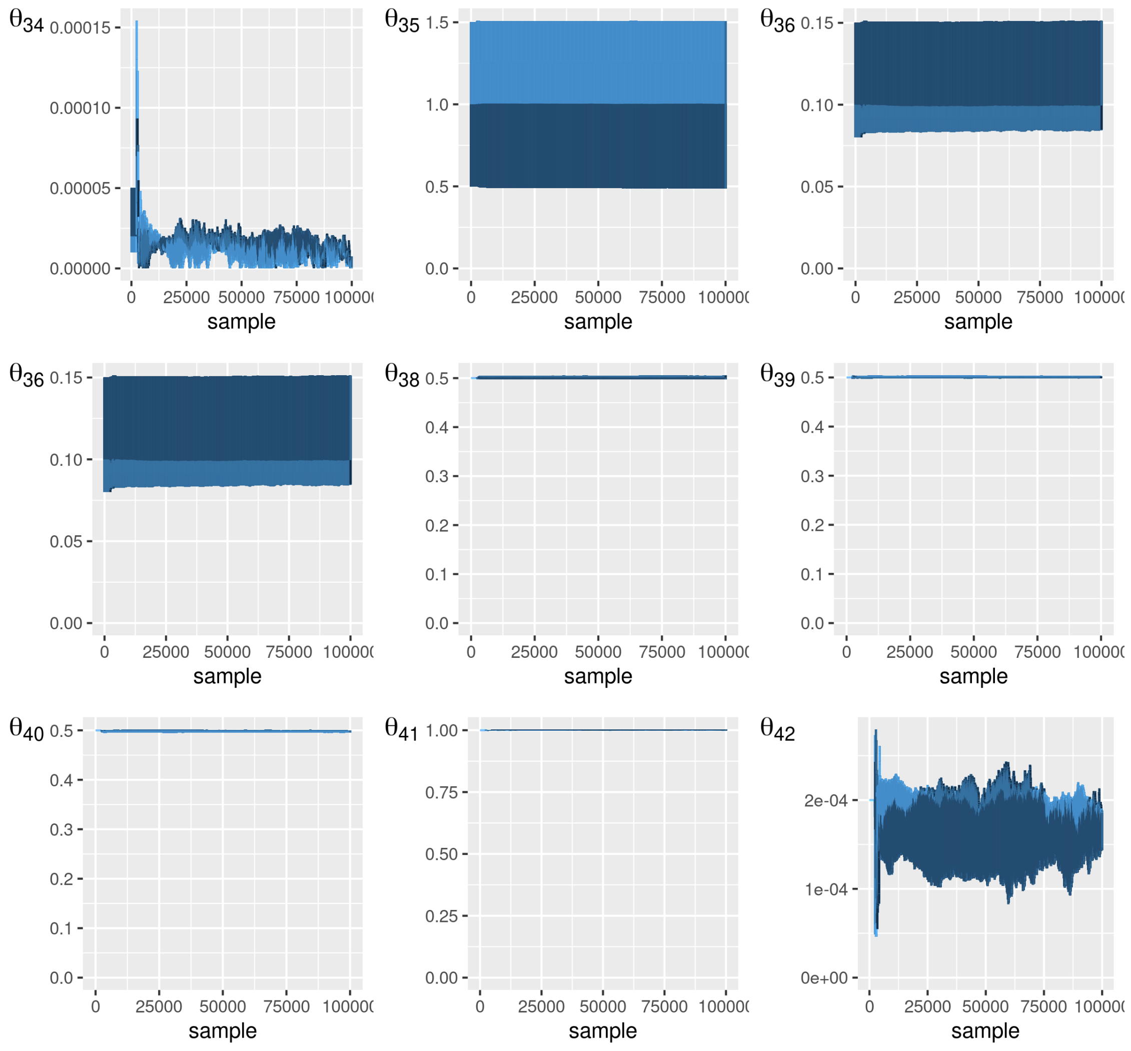


Figure E.64: Trace plots for the Scotland analysis.

E.2.8 Tabulated results

θ_i	Type	Mean	SD	$\hat{R}$	97.5
1	1	0.00003	0.00002	1.0	1.0
2	1	0.00003	0.00003	1.0	1.0
3	1	0.00004	0.00003	1.0	1.0
4	1	0.00003	0.00002	1.0	1.1
5	2	0.00015	0.00008	1.0	1.0
6	3	0.76497	0.21132	1.0	1.0
7	4	0.35162	0.23306	1.0	1.1
8	5	0.38440	0.16628	1.0	1.1
9	5	0.62194	0.21437	1.0	1.1
10	5	0.28467	0.11730	1.1	1.2
11	5	0.56560	0.08843	1.0	1.0
12	6	0.89852	0.44220	1.1	1.4
13	6	0.00040	0.00076	1.3	2.1

Table E.10: Basic hierarchical Hawkes model parameter inference results for East England. Number of herds (reactors) := 513 (75.)

θ_i	Type	Mean	SD	$\hat{R}$	97.5
1	1	0.00013	0.00001	1.0	1.1
2	1	0.00004	0.00002	1.1	1.2
3	1	0.00004	0.00002	1.0	1.0
4	2	0.00012	0.00006	1.0	1.1
5	3	0.73712	0.11517	1.1	1.2
6	4	0.27282	0.25733	5.5	27.3
7	5	0.36225	0.06705	1.0	1.0
8	5	0.20594	0.13754	1.1	1.1
9	5	0.51186	0.03151	1.1	1.4
10	5	0.81950	0.11625	1.0	1.0
11	6	0.68962	0.37199	1.3	1.8
12	6	0.00137	0.00217	1.1	1.3

Table E.11: Basic hierarchical Hawkes model parameter inference results for East Midlands. Number of herds (reactors) := 1966 (1217.)

θ_i	Type	Mean	SD	$\hat{R}$	97.5
1	1	0.00003	0.00001	1.3	2.2
2	1	0.00004	0.00007	1.0	1.0
3	1	0.00003	0.00006	1.0	1.1
4	1	0.00002	0.00001	1.3	2.0
5	2	0.00005	0.00008	1.0	1.1
6	3	0.75709	0.27895	1.3	2.2
7	4	0.21163	0.28057	3.7	18.3
8	5	0.37195	0.14676	2.1	7.6
9	5	0.37732	0.19076	3.2	6.0
10	5	0.51934	0.07150	1.4	2.4
11	5	0.68964	0.17708	1.8	3.5
12	6	1.01515	0.08721	2.3	5.3
13	6	0.00024	0.00047	1.3	1.8

Table E.12: Basic hierarchical Hawkes model parameter inference results for North east England. Number of herds (reactors) := 1015 (103.)

θ_i	Type	Mean	SD	$\hat{R}$	97.5
1	1	0.00003	0.00001	1.2	1.6
2	1	0.00008	0.00001	1.5	2.2
3	1	0.00004	0.00003	1.0	1.1
4	2	0.00002	0.00002	1.0	1.1
5	3	0.79097	0.16251	1.1	1.4
6	4	0.33565	0.21105	3.5	8.4
7	5	0.30561	0.12306	1.6	3.2
8	5	0.31803	0.17433	2.5	6.0
9	5	0.52150	0.07563	2.1	4.0
10	5	0.82074	0.18955	1.5	3.5
11	6	0.94661	0.34063	2.5	8.5
12	6	0.00033	0.00056	1.6	5.6

Table E.13: Basic hierarchical Hawkes model parameter inference results for North west England. Number of herds (reactors) := 3432 (989.)

θ_i	Type	Mean	SD	$\hat{R}$	97.5
1	1	0.00006	0.00003	2.2	4.2
2	1	0.00007	0.00005	2.1	4.1
3	1	0.00004	0.00003	2.0	3.9
4	1	0.00007	0.00004	2.0	3.6
5	1	0.00005	0.00002	1.9	3.5
6	1	0.00011	0.00004	3.4	6.9
7	2	0.00003	0.00004	1.1	1.3
8	3	0.83919	0.26441	3.0	11.7
9	4	0.07696	0.05712	1.5	3.6
10	5	0.33633	0.22746	14.0	69.9
11	5	0.41831	0.15700	3.2	6.7
12	5	0.42008	0.07587	1.7	3.1
13	5	0.67178	0.16185	2.9	6.2
14	6	0.82319	0.18789	1.5	2.7
15	6	0.00205	0.00374	1.8	3.8

Table E.14: Basic hierarchical Hawkes model parameter inference results for South east England. Number of herds (reactors) := 1393 (357.)

θ_i	Type	Mean	SD	$\hat{R}$	97.5
1	1	0.00015	0.00011	21.1	40.9
2	1	0.00001	0.00002	1.4	2.1
3	1	0.00001	0.00002	2.1	5.5
4	1	0.00002	0.00004	1.3	1.8
5	1	0.00001	0.00003	1.4	2.0
6	1	0.00010	0.00010	26.9	57.8
7	1	0.00004	0.00005	7.2	18.1
8	1	0.00003	0.00009	1.7	2.8
9	1	0.00024	0.00021	28.0	63.0
10	1	0.00001	0.00006	1.2	1.8
11	1	0.00021	0.00017	20.8	52.4
12	1	0.00002	0.00006	1.2	1.5
13	1	0.00021	0.00012	15.1	31.0
14	1	0.00019	0.00008	13.7	27.0
15	1	0.00004	0.00002	1.4	2.7
16	1	0.00017	0.00021	29.7	64.6
17	1	0.00004	0.00003	3.8	8.6
18	1	0.00001	0.00003	1.0	1.1
19	1	0.00014	0.00015	11.8	33.5
20	1	0.00007	0.00005	6.5	17.7
21	1	0.00001	0.00002	1.1	1.1
22	1	0.00005	0.00005	4.1	14.2
23	1	0.00004	0.00002	2.0	3.3
24	1	0.00001	0.00004	1.1	1.4
25	1	0.00004	0.00002	4.3	8.8
26	1	0.00006	0.00005	2.0	5.3
27	1	0.00031	0.00015	14.8	31.2
28	1	0.00019	0.00013	14.4	31.8

Table E.15: Basic hierarchical Hawkes model parameter inference results for Scotland – continued on next page. Number of herds (reactors) := 4510 (280.

θ_i	Type	Mean	SD	$\hat{R}$	97.5
29	1	0.00023	0.00014	13.2	27.9
30	1	0.00014	0.00018	19.0	49.9
31	1	0.00003	0.00002	1.8	3.5
32	1	0.00032	0.00021	33.6	66.6
33	1	0.00007	0.00004	5.2	10.1
34	2	0.00001	0.00001	1.9	3.1
35	3	1.00009	0.41471	1116.5	2394.9
36	4	0.11131	0.02861	208.5	539.5
37	5	0.50017	0.00088	5.0	10.1
38	5	0.50147	0.00277	23.4	55.6
39	5	0.50100	0.00211	11.9	24.8
40	5	0.49871	0.00217	11.9	31.8
41	6	1.00042	0.00142	13.0	26.3
42	6	0.00018	0.00004	6.4	14.2

Table E.16: Basic hierarchical Hawkes model parameter inference results for Scotland – carried over from previous page. Number of herds (reactors) := 4510 (280.)

E.3 Heterogeneous disease process-hierarchical model results

E.3.1 Summaries

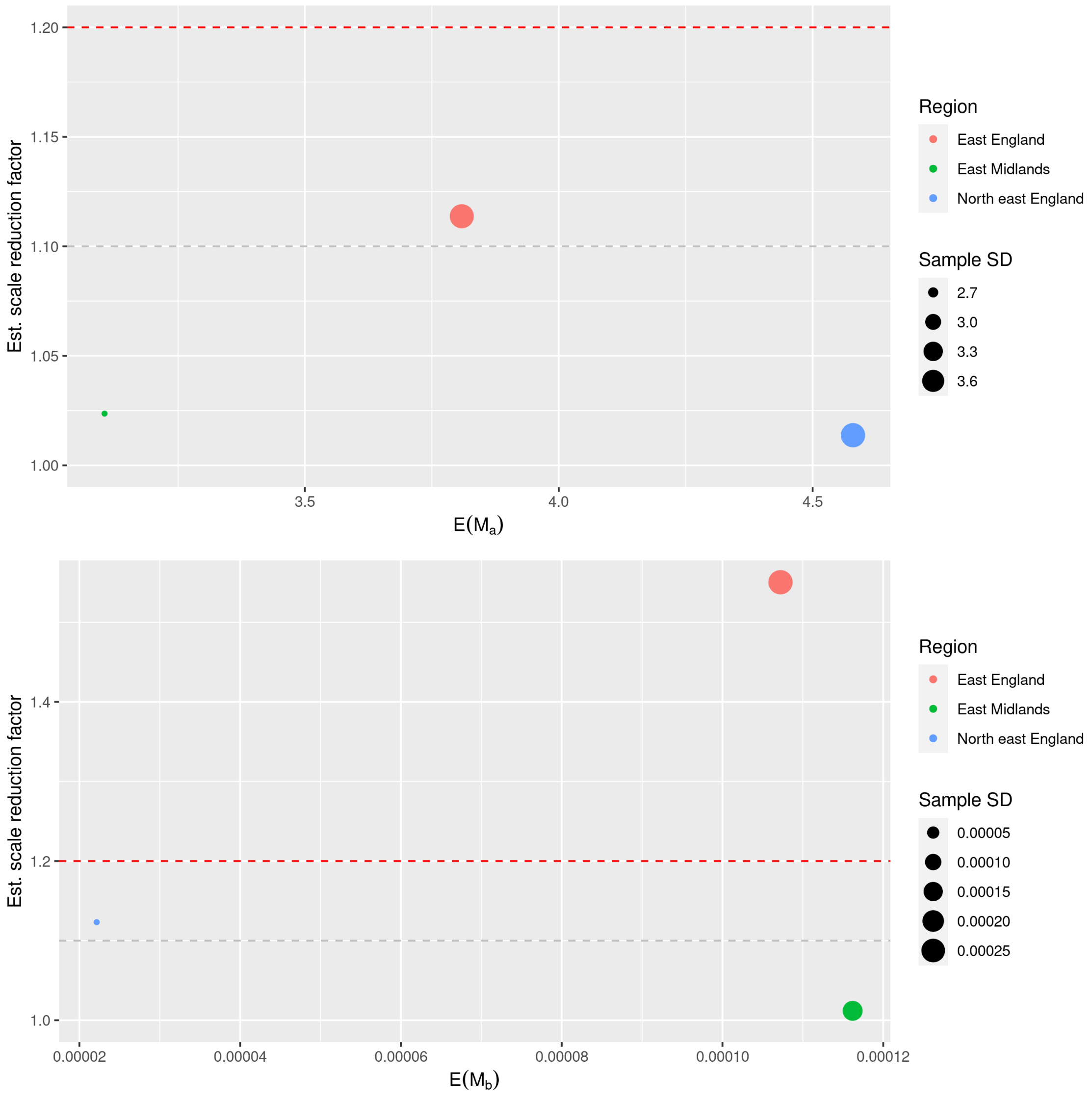
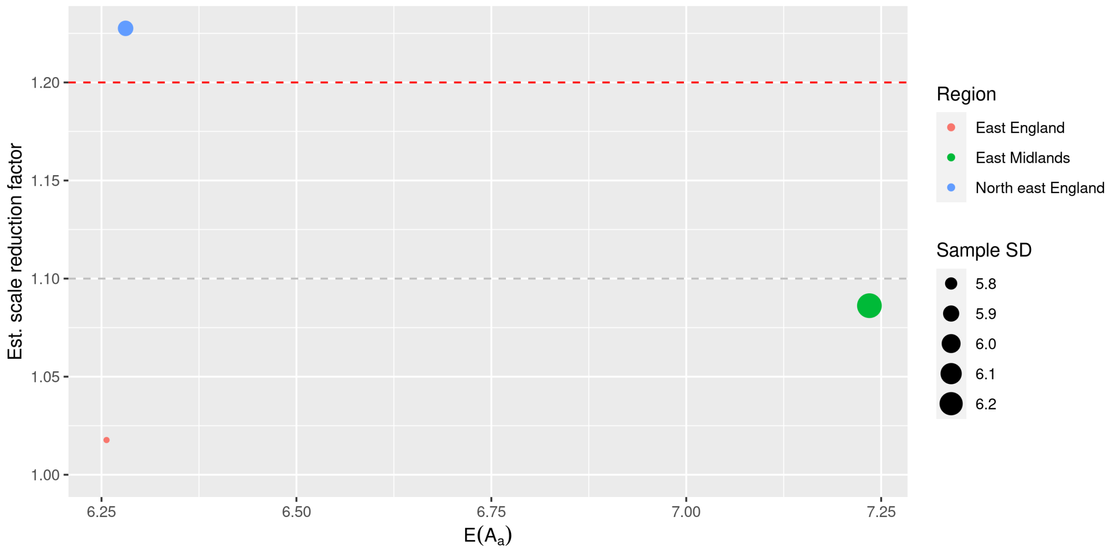


Figure E.65: Constant FOI hierarchical parameter estimates.



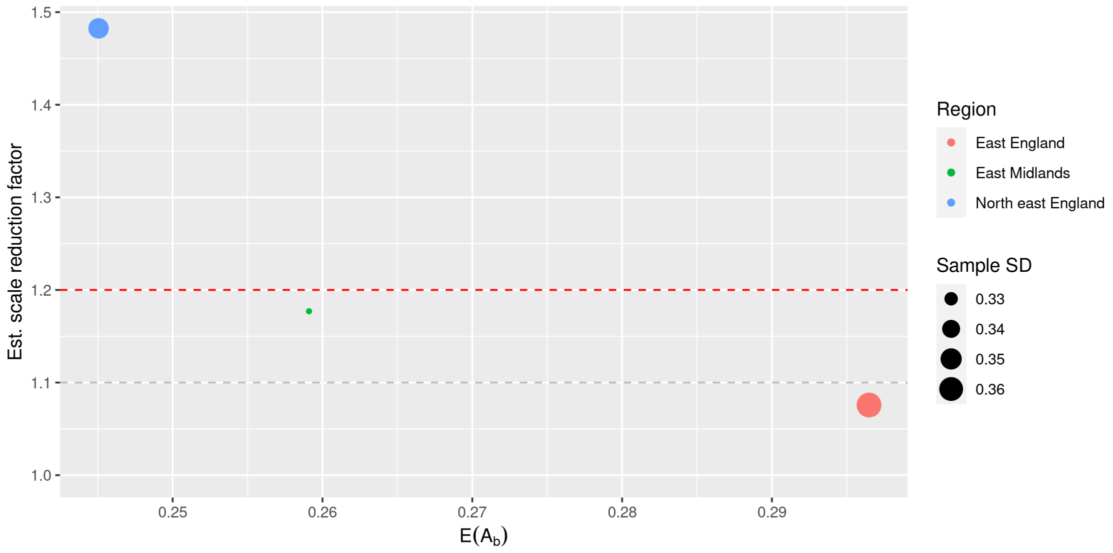
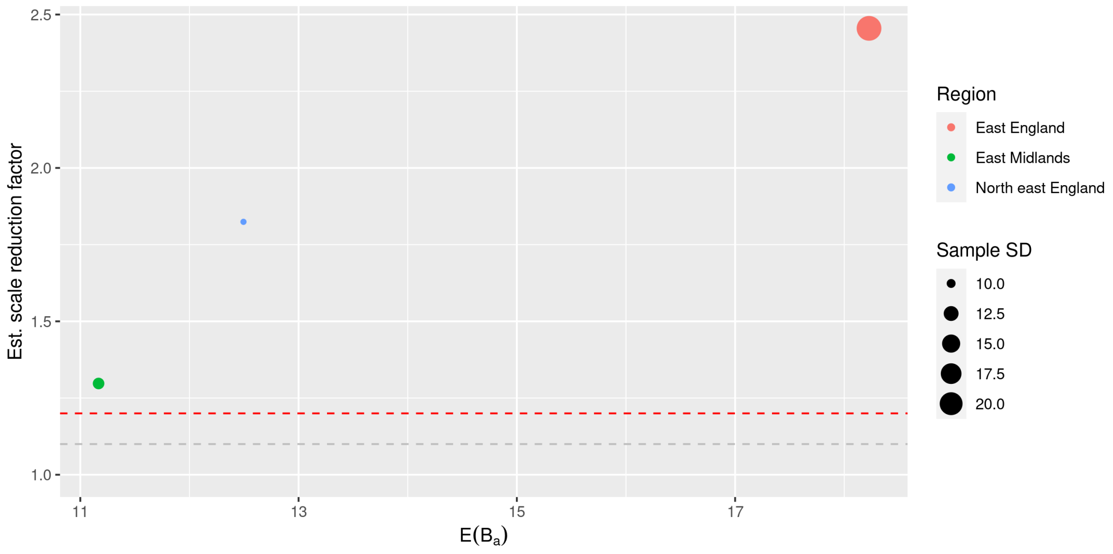


Figure E.66: Disease process parameter estimates $\alpha \sim \Gamma(A_a, A_b)$.



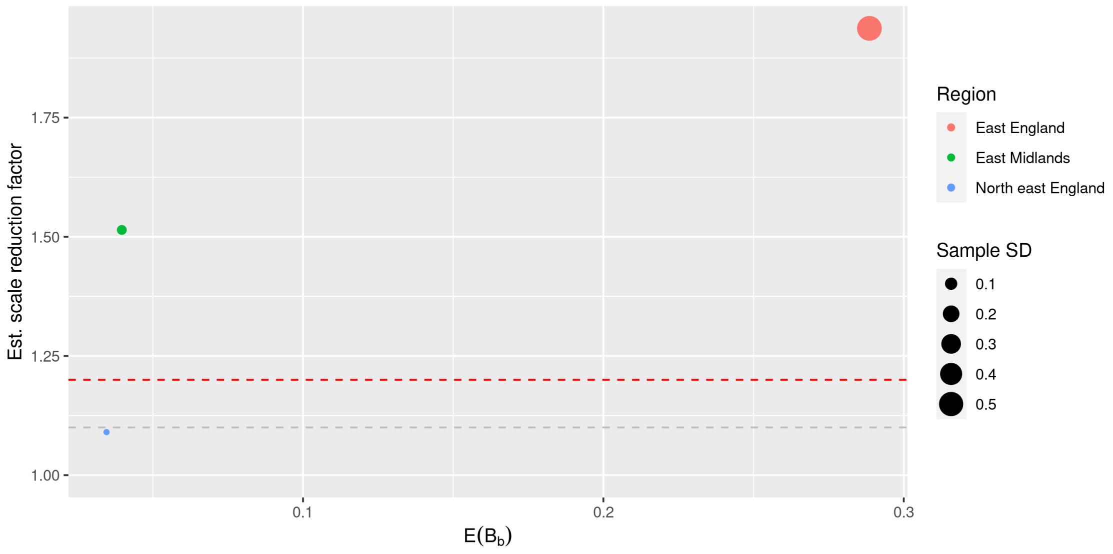
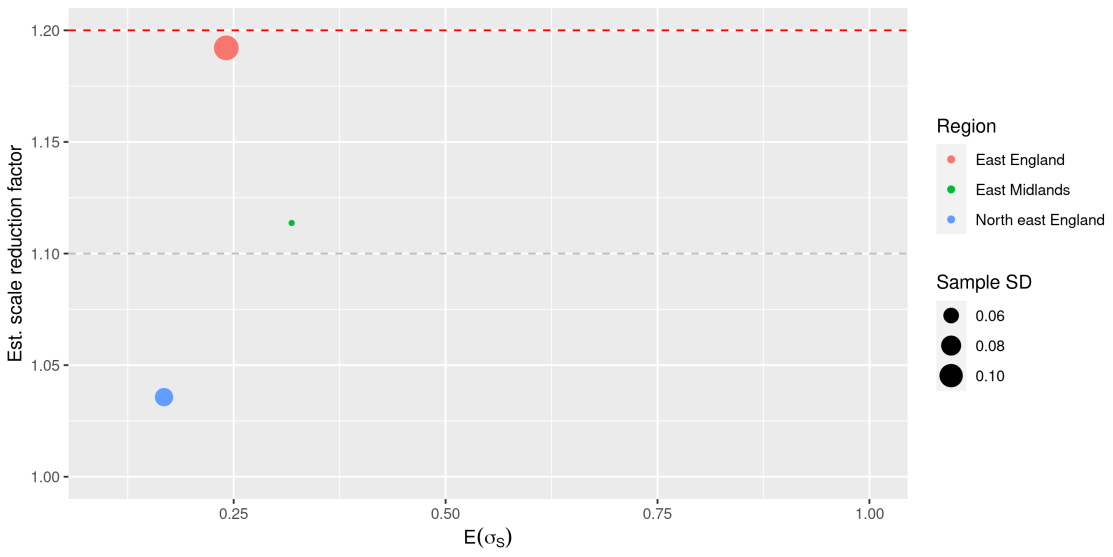
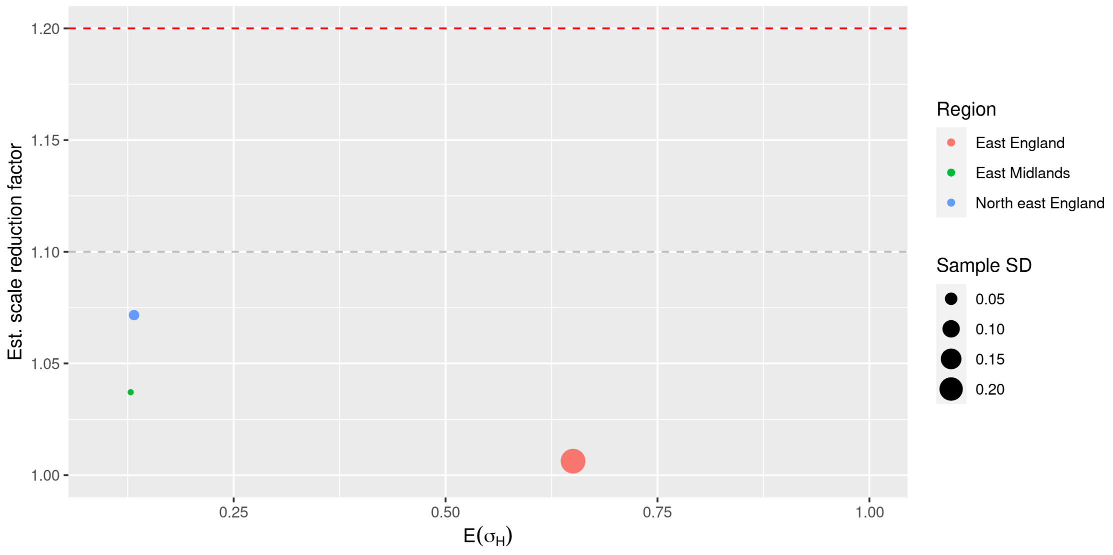


Figure E.67: Disease process parameter estimates $\beta \sim \Gamma(B_a, B_b)$.

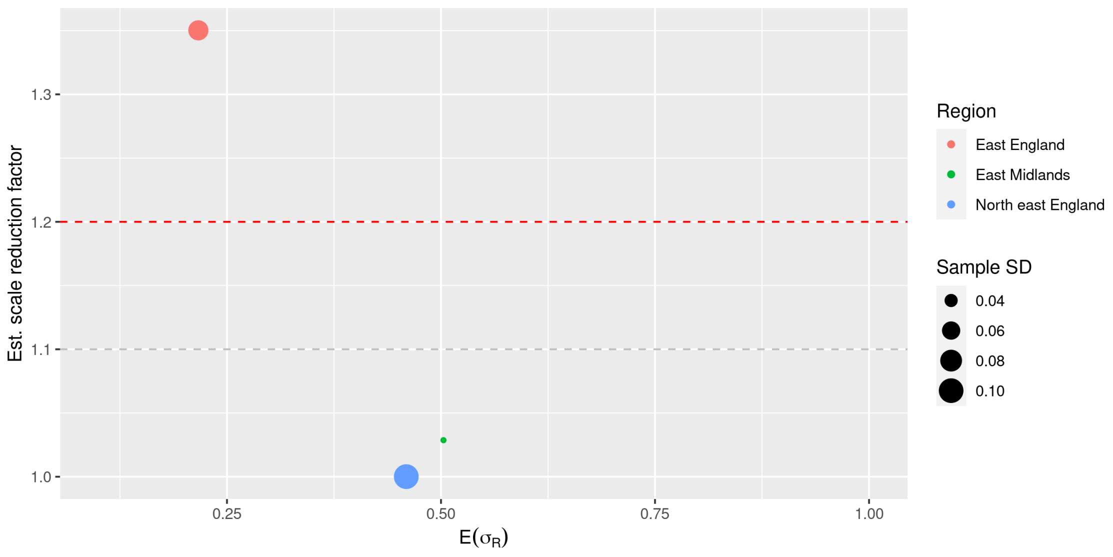


(a) Standard sensitivity.

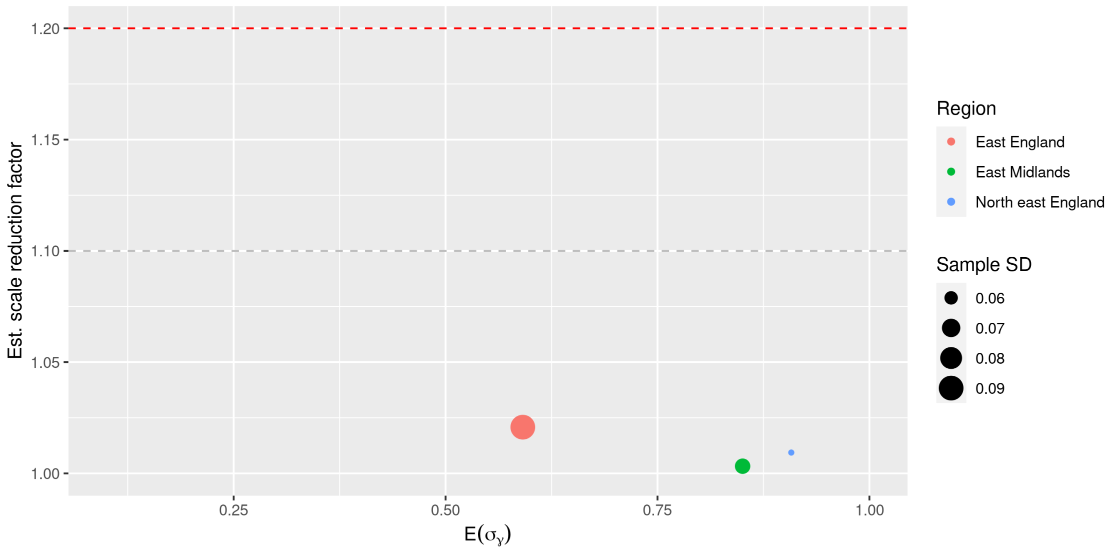


(b) High sensitivity.

Figure E.68: Diagnostic test sensitivity parameter estimates: SICCT test.



(a) Trade-related testing.



(b) IFN$_\gamma$ blood test.

Figure E.69: Diagnostic test sensitivity parameter estimates: other.

E.3.2 East England

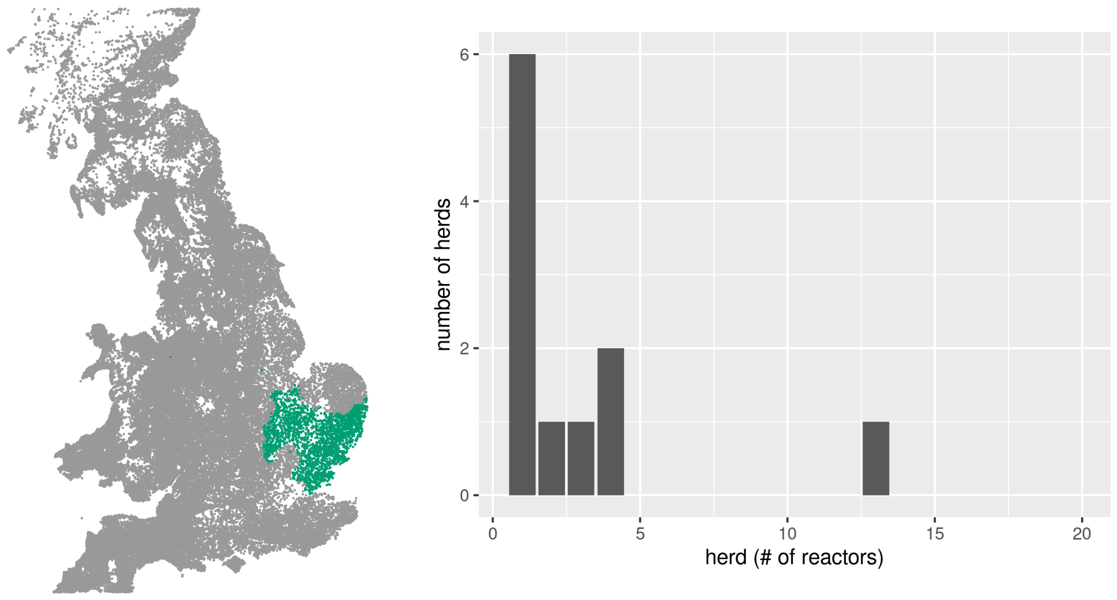


Figure E.70: Herd and reactor distribution. The latter is given for the total number reactors by herd for the entire observation period.

Number of herds: 513
Reactor herds: 13
Number of reactors, i.e. animals: 75

σ_i	animals	+ve
1	981	11
2	93	2
3	1619	29
4	20	33

Figure E.71: VetNet surveillance data for selected East England herds. The diagnostic test data are categorised as follows: standard SICCT herd tests; high-sensitivity SICCT herd tests; risk-based surveillance testing; and the IFN$_\gamma$ blood test trial, denoted σ_i for 1:4 respectively.

Marginal sample distributions

Here we provide a visual summary of the parameter estimates for the hierarchical model with heterogeneous disease processes (for the East England group.) A tabulated summary is also provided, in Table E.17.

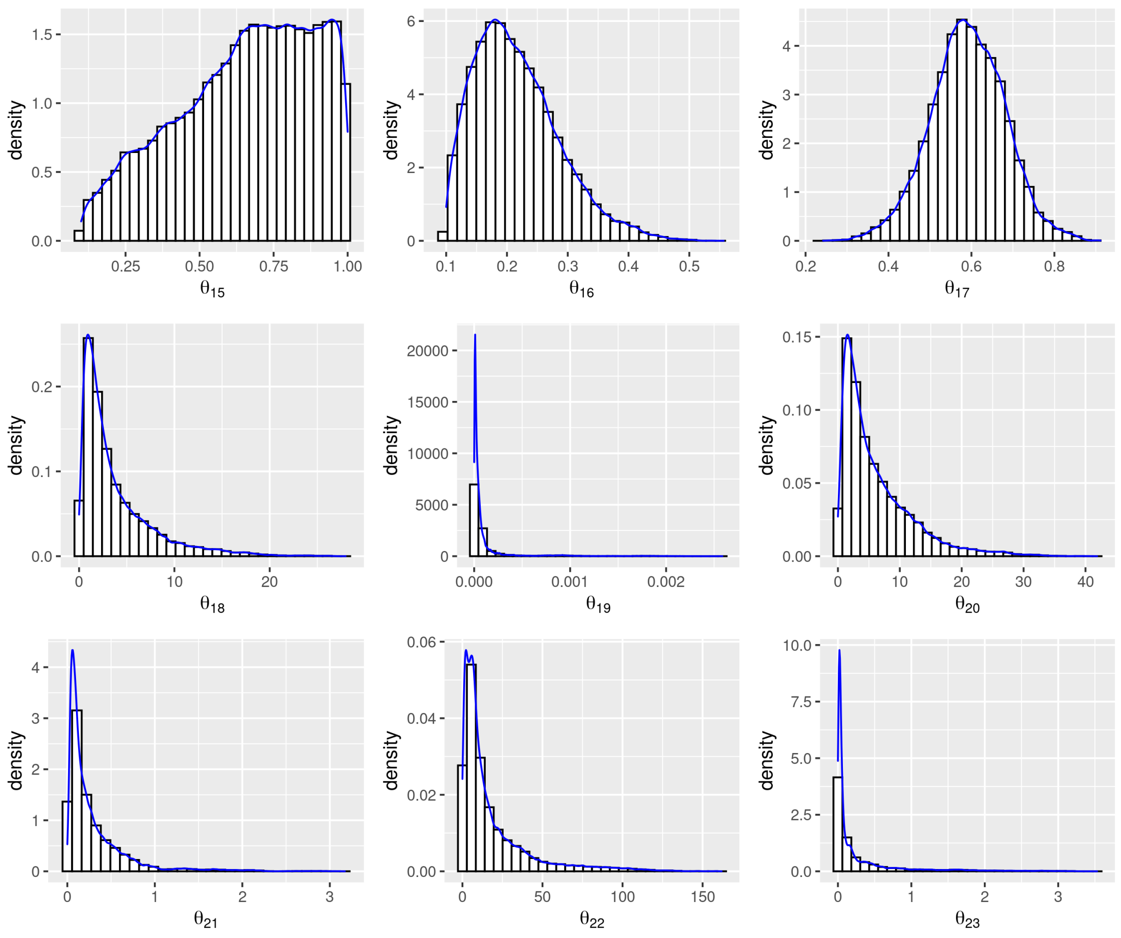
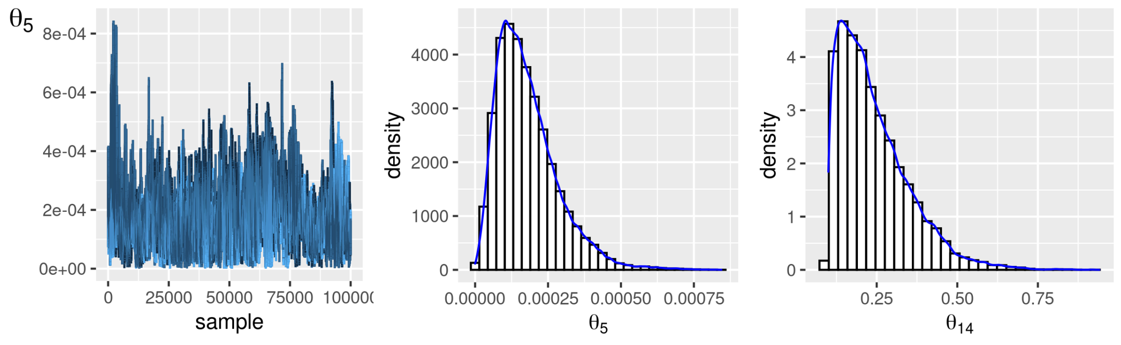


Figure E.72: Marginal densites for the East England analysis.

Traceplots



(a) Marginal density and trace plot for the trade: μ_T parameter.

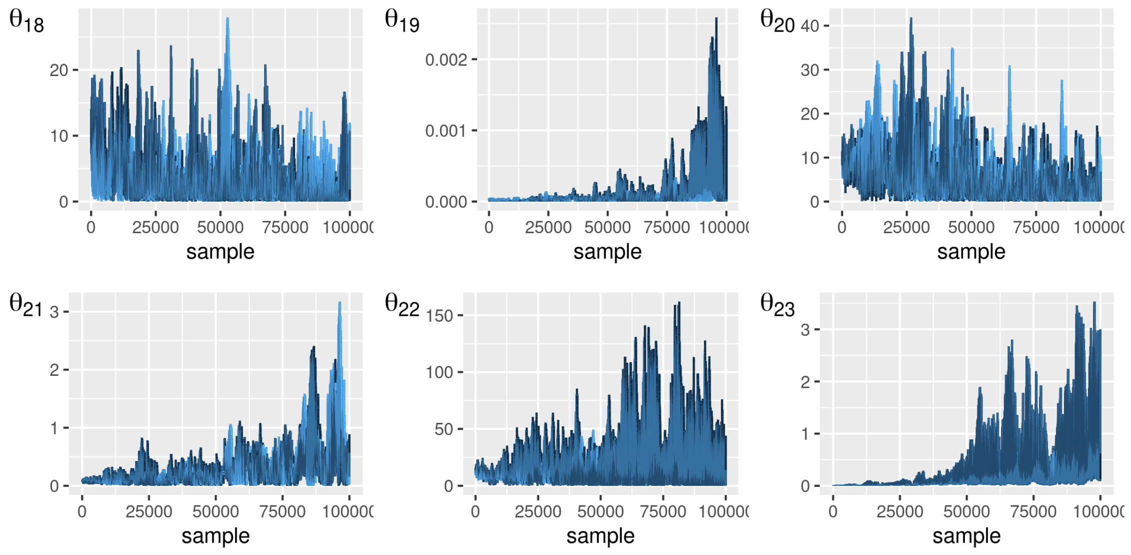


Figure E.73: Trace plots for the East England analysis.

E.3.3 East Midlands

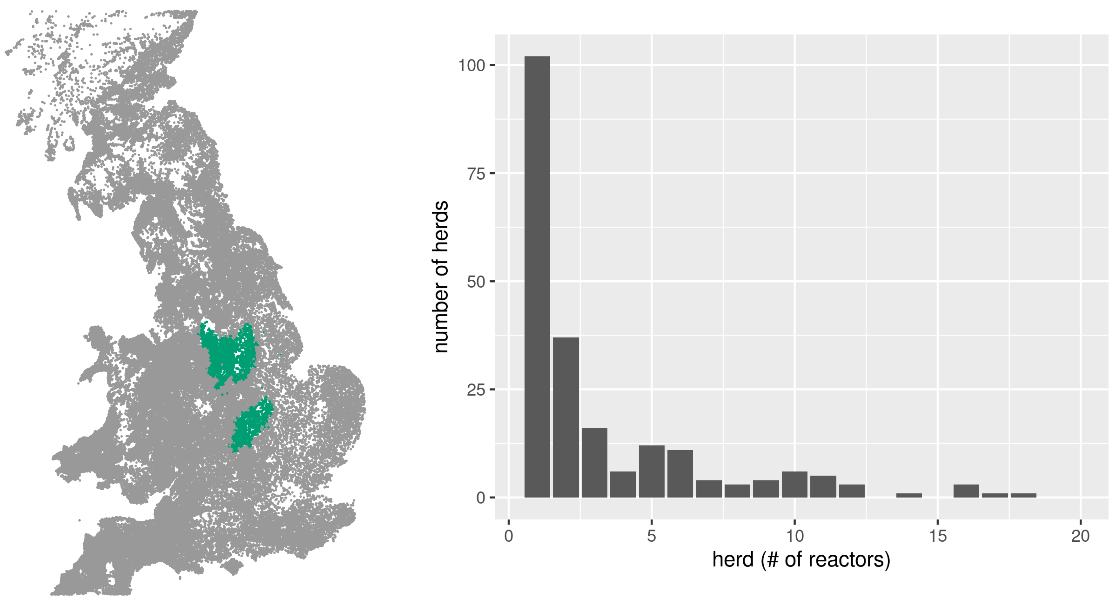


Figure E.74: Herd and reactor distribution. The latter is given for the total number reactors by herd for the entire observation period.

Number of herds: 1966
Reactor herds: 231
Number of reactors, i.e. animals: 1217

σ_i	animals	+ve
1	5100	181
2	6142	17
3	7475	890
4	40	129

Figure E.75: VetNet surveillance data for selected East Midlands herds. The diagnostic test data are categorised as follows: standard SICCT herd tests; high-sensitivity SICCT herd tests; risk-based surveillance testing; and the IFN$_\gamma$ blood test trial, denoted σ_i for 1:4 respectively.

Marginal sample distributions

Here we provide a visual summary of the parameter estimates for the hierarchical model with heterogeneous disease processes (for the East Midlands group.) A tabulated summary is also provided, in Table E.18.

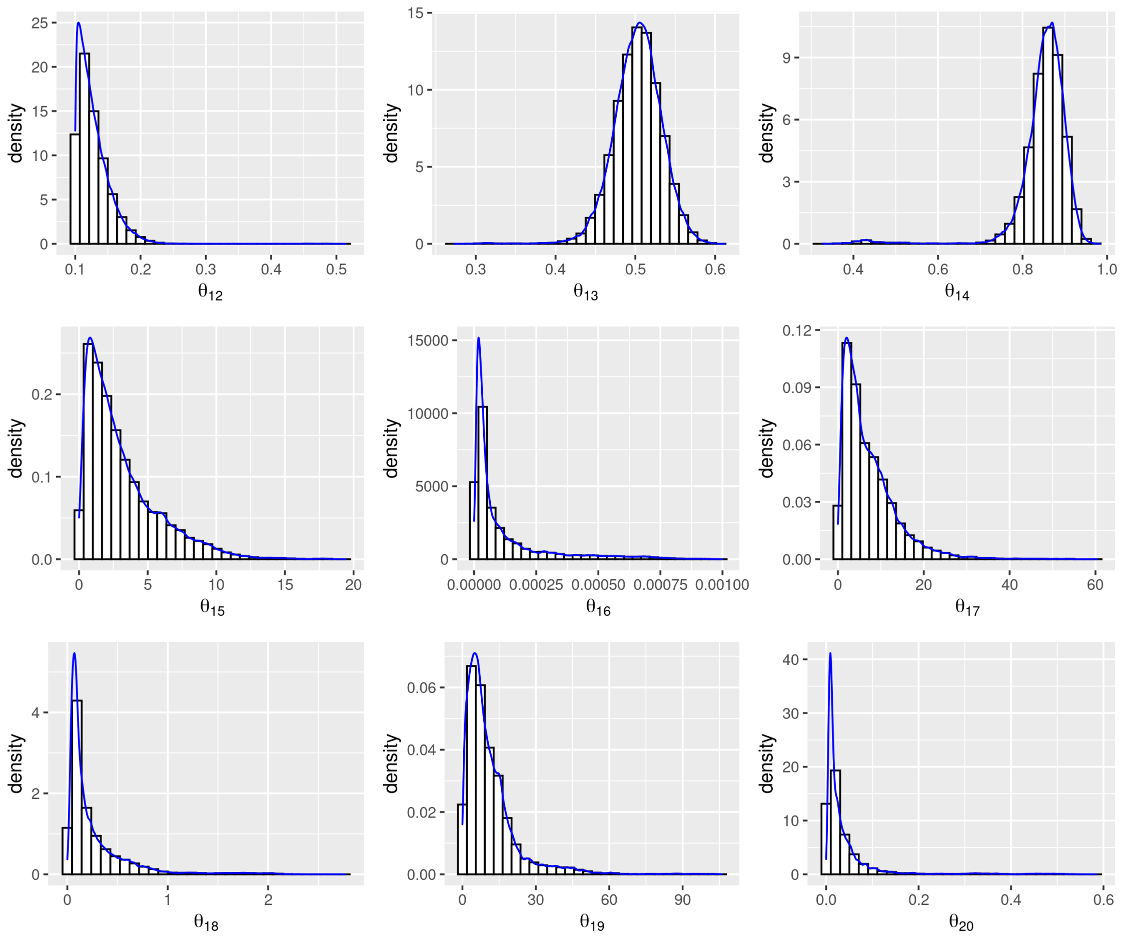
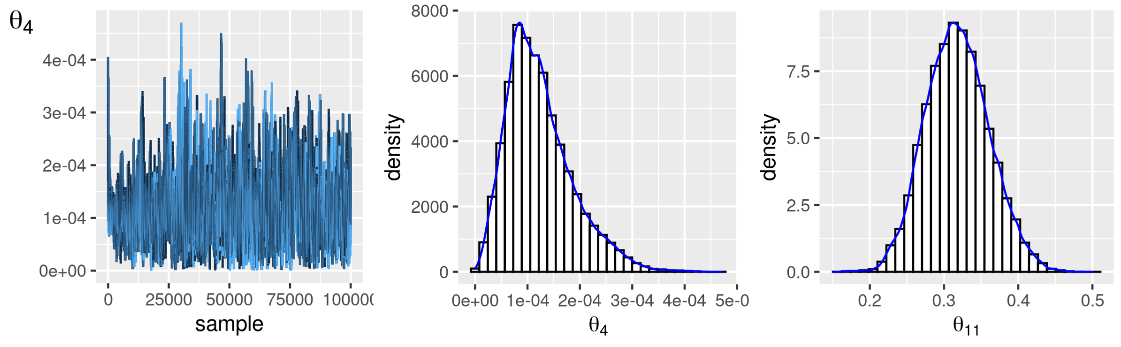


Figure E.76: Marginal densites for the East Midlands analysis.

Traceplots



(a) Marginal density and trace plot for the trade: μ_T parameter.

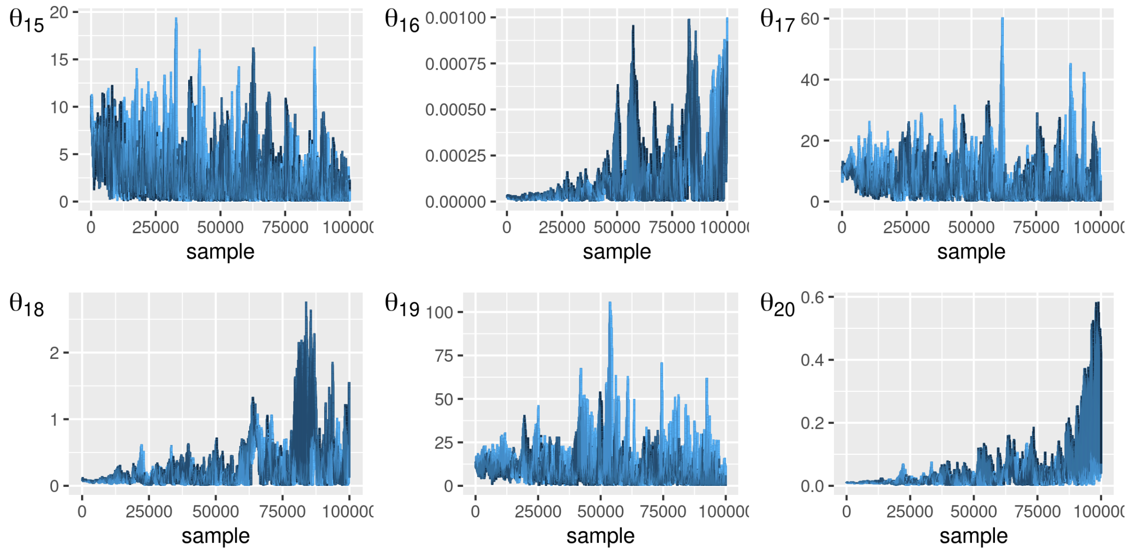
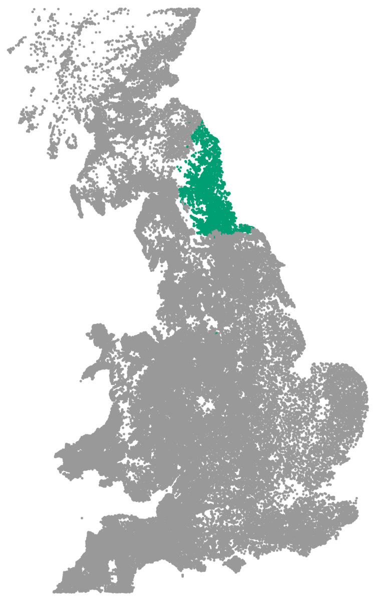


Figure E.77: Trace plots for the East Midlands analysis.

E.3.4 North east England



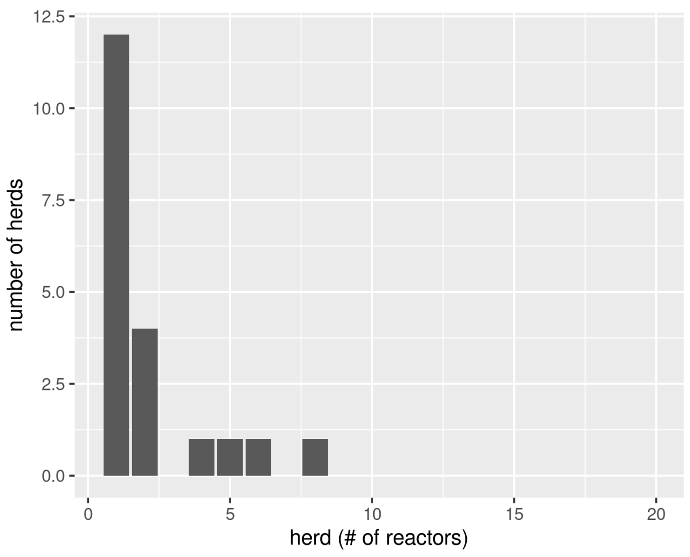


Figure E.78: Herd and reactor distribution. The latter is given for the total number reactors by herd for the entire observation period.

Number of herds: 1015
Reactor herds: 22
Number of reactors, i.e. animals: 103

σ_i	animals	+ve
1	2383	14
2	691	1
3	2017	30
4	8	58

Figure E.79: VetNet surveillance data for selected North east England herds. The diagnostic test data are categorised as follows: standard SICCT herd tests; high-sensitivity SICCT herd tests; risk-based surveillance testing; and the IFN$_\gamma$ blood test trial, denoted σ_i for 1:4 respectively.

Marginal sample distributions

Here we provide a visual summary of the parameter estimates for the hierarchical model with heterogeneous disease processes (for the North east England group.) A tabulated summary is also provided, in Table E.19.

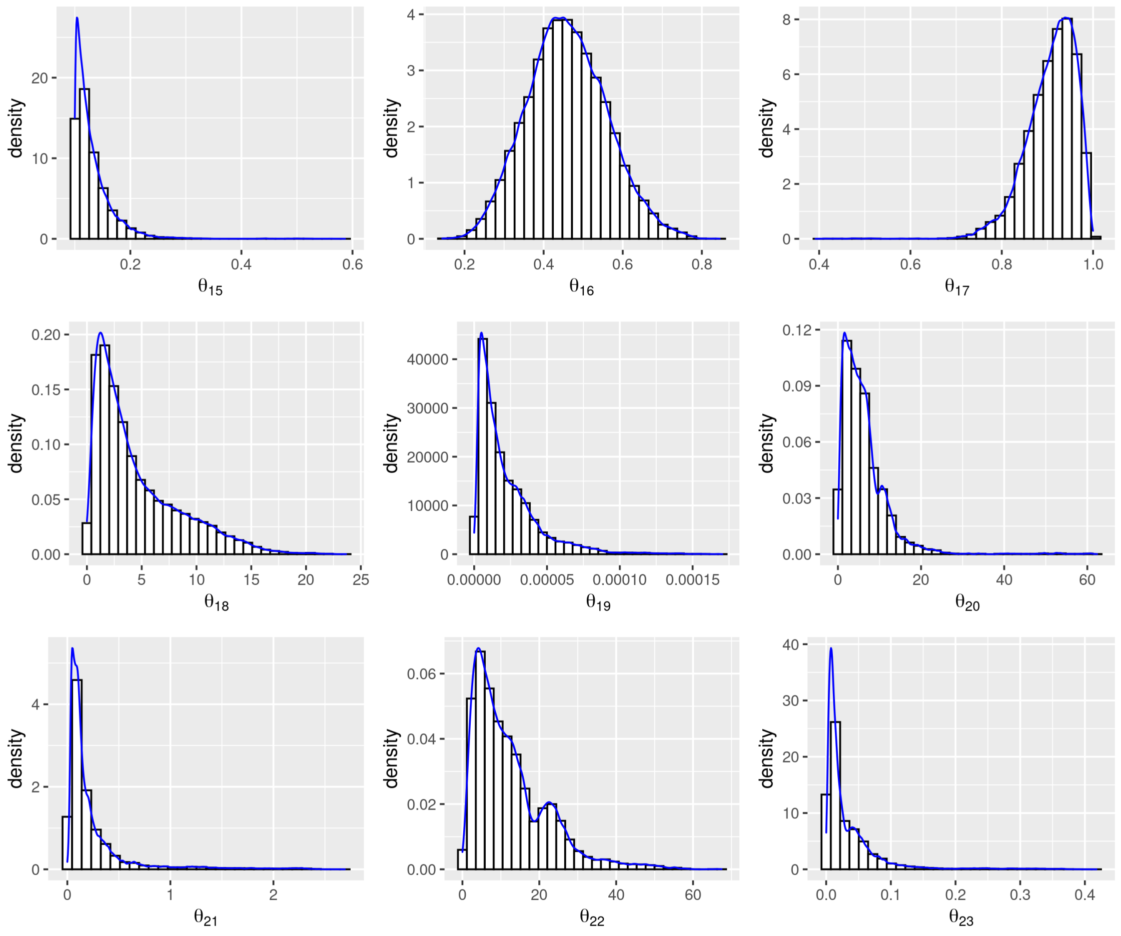
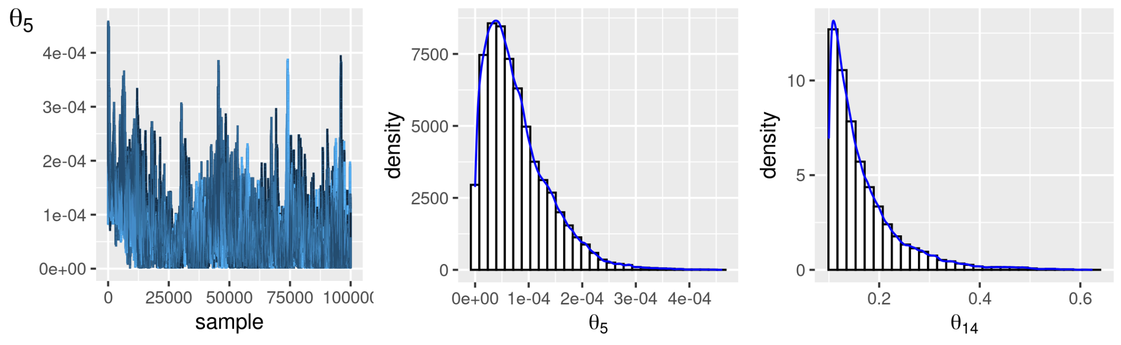


Figure E.80: Marginal densites for the North east England analysis.

Traceplots



(a) Marginal density and trace plot for the trade: μ_T parameter.

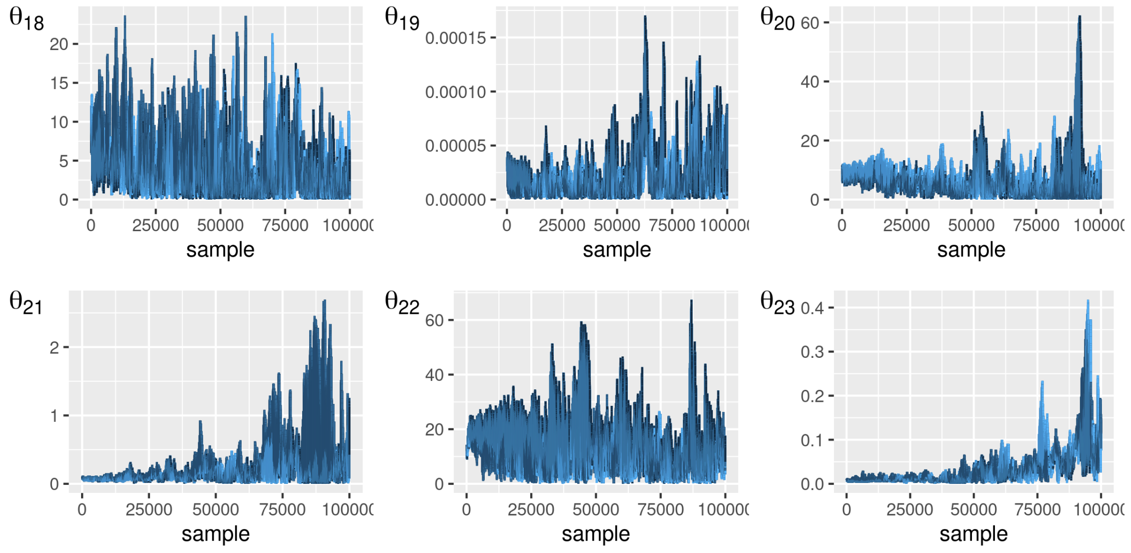


Figure E.81: Trace plots for the North east England analysis.

E.3.5 Tabulated results

θ_i	Type	Mean	SD	$\hat{R}$	97.5
1	1	0.00004	0.00003	1.2	1.5
2	1	0.00006	0.00004	1.1	1.2
3	1	0.00008	0.00005	1.1	1.4
4	1	0.00004	0.00002	1.1	1.3
5	2	0.00018	0.00011	1.2	1.5
6	3	0.69690	0.18566	1.0	1.0
7	3	0.47399	0.34401	1.1	1.2
8	3	0.49943	0.19281	1.0	1.0
9	3	0.75135	0.12380	1.1	1.2
10	4	1.12665	1.11948	1.3	1.9
11	4	1.74615	2.07334	1.1	1.3
12	4	1.30081	1.32197	1.2	1.6
13	4	1.22070	1.36326	2.0	4.3
14	5	0.24125	0.10965	1.2	1.5
15	5	0.65035	0.22885	1.0	1.0
16	5	0.21656	0.07008	1.4	2.0
17	5	0.59108	0.09001	1.0	1.1
18	6	3.80898	3.84957	1.1	1.3
19	6	0.00011	0.00027	1.6	5.5
20	7	6.25645	5.74147	1.0	1.0
21	7	0.29648	0.36387	1.1	1.1
22	8	18.22702	22.46303	2.5	10.0
23	8	0.28857	0.52707	1.9	7.5

Table E.17: Heterogenous disease process-hierarchical Hawkes model parameter inference results for East England. Number of herds (reactors) := 513 (75.)

θ_i	Type	Mean	SD	$\hat{R}$	97.5
1	1	0.00014	0.00001	1.0	1.1
2	1	0.00004	0.00002	1.0	1.0
3	1	0.00006	0.00002	1.0	1.0
4	2	0.00012	0.00006	1.0	1.1
5	3	0.71438	0.02869	1.0	1.0
6	3	0.84369	0.08788	1.0	1.0
7	3	0.42665	0.14164	1.0	1.1
8	4	0.17822	0.09140	1.2	1.7
9	4	0.11511	0.03255	1.2	1.5
10	4	0.18628	0.14867	1.1	1.1
11	5	0.31841	0.04248	1.1	1.4
12	5	0.12852	0.02620	1.0	1.1
13	5	0.50289	0.02987	1.0	1.1
14	5	0.85040	0.06335	1.0	1.0
15	6	3.10529	2.63685	1.0	1.1
16	6	0.00012	0.00016	1.0	1.0
17	7	7.23493	6.28737	1.1	1.2
18	7	0.25910	0.32388	1.2	1.5
19	8	11.16747	10.80835	1.3	2.1
20	8	0.03966	0.06644	1.5	4.4

Table E.18: Heterogenous disease process-hierarchical Hawkes model parameter inference results for East Midlands. Number of herds (reactors) := 1966 (1217.)

θ_i	Type	Mean	SD	$\hat{R}$	97.5
1	1	0.00004	0.00001	1.0	1.0
2	1	0.00005	0.00005	1.0	1.0
3	1	0.00004	0.00003	1.0	1.0
4	1	0.00003	0.00002	1.0	1.1
5	2	0.00008	0.00006	1.0	1.0
6	3	0.70678	0.08110	1.0	1.0
7	3	0.63718	0.58033	1.3	2.4
8	3	0.60250	0.41467	1.0	1.1
9	3	0.45670	0.14630	1.0	1.1
10	4	0.20104	0.17795	1.3	1.8
11	4	0.35729	0.45170	1.2	1.5
12	4	0.30932	0.47243	1.2	1.8
13	4	0.24878	0.26431	1.1	1.3
14	5	0.16785	0.07128	1.0	1.1
15	5	0.13243	0.03677	1.1	1.2
16	5	0.45943	0.10063	1.0	1.0
17	5	0.90782	0.05455	1.0	1.0
18	6	4.57949	3.88165	1.0	1.0
19	6	0.00002	0.00002	1.1	1.3
20	7	6.28088	5.87895	1.2	1.7
21	7	0.24505	0.34751	1.5	3.6
22	8	12.49514	9.72486	1.8	4.0
23	8	0.03458	0.04755	1.1	1.2

Table E.19: Heterogenous disease process-hierarchical Hawkes model parameter inference results for North east England. Number of herds (reactors) := 1015 (103.)