Markov chain Monte Carlo for continuous-time discrete-state systems

Vinayak A. P. Rao

Gatsby Computational Neuroscience Unit
University College London
17 Queen Square
London WC1N 3AR, United Kingdom

THESIS

Submitted for the degree of
Doctor of Philosophy, University College London

2012

I, Vinayak A. P. Rao, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Abstract

A variety of phenomena are best described using dynamical models which operate on a discrete state space and in continuous time. Examples include Markov (and semi-Markov) jump processes, continuous-time Bayesian networks, renewal processes and other point processes. These continuous-time, discrete-state models are ideal building blocks for Bayesian models in fields such as systems biology, genetics, chemistry, computing networks, human-computer interactions etc. However, a challenge towards their more widespread use is the computational burden of posterior inference; this typically involves approximations like time discretization and can be computationally intensive.

In this thesis, we describe a new class of Markov chain Monte Carlo methods that allow efficient computation while still being exact. The core idea is an auxiliary variable Gibbs sampler that alternately resamples a random discretization of time given the state-trajectory of the system, and then samples a new trajectory given this discretization. We introduce this idea by relating it to a classical idea called *uniformization*, and use it to develop algorithms that outperform the state-of-the-art for models based on the Markov jump process. We then extend the scope of these samplers to a wider class of models such as nonstationary renewal processes, and semi-Markov jump processes. By developing a more general framework beyond uniformization, we remedy various limitations of the original algorithms, allowing us to develop MCMC samplers for systems with infinite state spaces, unbounded rates, as well as systems indexed by more general continuous spaces than time.

Acknowledgments

I have been fortunate to spend more than four years working with Yee Whye Teh, and would like to thank him for his help and patience. Yee Whye is an inspiration, both in how much he knows and how much he still wants to learn, and I have benefitted immensely from his ideas and his feedback, and from the general confidence that I was in very good hands.

I will greatly miss the camaraderie and the intellectual stimulation of the Gatsby Unit (typified by tea and talks every day). Peter Dayan, the director deserves credit for keeping it the way it is, and for somehow making time in his busy schedule to attend every talk that everyone gives. I would like to thank him, and the other faculty, Arthur, Maneesh and Peter L., for help, comments and advice. Many thanks also to David Dunson and Erik Sudderth, both of whom made time for me when I visited towards the end of my PhD, as well as to my examiners, Arnaud Doucet and Mark Girolami, for finding time to read and comment on this thesis.

I'd like to thank Chris Sherlock, Guido Sanguinetti, Iain Murray and Ryan Adams for making their code available to me (the LaTeX template of this thesis can be traced back to Iain).

In my nearly-five-but-still-not-enough years in London, I have made many dear friends whom I hope to stay in touch with. There are many others from before as well; I owe them all thanks. I won't try to list everyone, and apologize to those I've forgotten; at the Gatsby, I'd especially like to thank Amit, Andriy, Biljana, David B, David S, Gabi, Jan, Kai, Lars, Mani, Maria, Phillipp, Ricardo and Ross. Charles and Lloyd, my officemates over the last two years deserve special credit for putting up with me.

My PhD was funded by the Gatsby charitable foundation, and I am grateful for their support. I also thank the Bogue foundation for sponsoring a research visit to the US.

Finally, and most importantly, I'd like to thank my parents Pramesh and Parvathy Rao for their love and support throughout the years.

Contents

List of figures

List of tables

List of algorithms

Chapter 1

Introduction

1.1 Modelling in continuous time

Many applications require modelling the time evolution of a dynamical system. A simple and popular approach is to discretize time and work with the resulting discrete-time model. Such systems have been well studied in the time series modelling literature (Rabiner, 1989; Murphy, 2002) and find wide application in fields like statistics, machine learning, signal processing, computational biology etc. A particular driving force for the growing sophistication of these models has been the parallel development of techniques for efficient inference. Some of the most popular and flexible approaches to inference are sampling-based Monte Carlo approaches, see for example (Robert and Casella, 2005; Gilks et al., 1996; Gelman et al., 2010; Doucet et al., 2001). These have been particularly important in the Bayesian community, where they form a natural approach to posterior inference in probabilistic models of various phenomena. The modularity of these techniques has allowed the straightforward development of complex, hierarchical models based on simple building blocks.

Often, one is interested in modelling a system whose evolution is asynchronous with a number of different time scales. In such a situation, the behaviour of the resulting time-discretized model can be sensitive to the chosen time scale. To achieve reasonable approximations, a sufficiently fine time-resolution may be needed that can make these approaches impractical for large problems. A more natural approach is to work directly in continuous time; in fact, it is often convenient make continuous approximations to inherently discrete systems (for example in genetics, where base-position along a strand of DNA is sometimes treated as a real number). Moreover, continuous-time systems often lend themselves to easier theoretical analysis, and often arise naturally out of the physical and statistical laws characterizing the systems' evolution.

A major impediment towards the more widespread use of these models is the problem of inference. While the system itself might be easy to characterize, introducing partial

and noisy observations of its state introduces interactions which break down the simplicity of these models. In the language of Bayesian statistics, a continuous-time model specifies a prior distribution over continuous-time trajectories, and the posterior distribution resulting from observations via some likelihood function is often intractable. Additionally, the parameters governing the system dynamics are often not known, and must also be inferred from data. The focus of this thesis is on sampling algorithms that can be used to explore these intractable distributions.

A typical approach to posterior sampling for continuous-time models involves discretizing time and then running an appropriate discrete-time sampling algorithm on the resulting system. This has a number of drawbacks, not least of which is that we lose the advantages that motivated the use of a continuous-time model in the first place. Time-discretization introduces biases into our inferences, and to control this, one might have to discretize time at a resolution that results in a very large number of discrete time steps. This can be computationally expensive.

In this thesis, we bring a workhorse of the discrete-time domain, the forward-filtering backward-sampling algorithm (Früwirth-Schnatter, 1994; Carter and Kohn, 1996), to continuous-time. The forward-backward algorithm is a dynamic programming algorithm that recursively accounts for successive observations during a forward pass though time. Having filtered in all observations, it then instantiates a trajectory of the system via a backward pass. While developed originally for finite state hidden Markov models and linear Gaussian systems, this algorithm also forms the core of samplers for more complicated systems like nonlinear and non-Gaussian time series (Neal et al., 2004), infinite state time series (Van Gael et al., 2008), non-Markovian systems (Dewar et al., 2012) etc. We show in this thesis how to extend these ideas to the continuous-time domain.

The core of our approach is an auxiliary variable Gibbs sampler that proceeds by repeating two steps. The first uses a *random* discretization of time to sample a new trajectory using the forward-backward algorithm. The second then samples a new time discretization given this new trajectory. A random discretization allows a relatively coarse grid, while still keeping inferences unbiased. The forward-backward algorithm can thus be run on relatively short chains, resulting in computational savings. We show how resampling the random time-discretization can be performed efficiently by exploiting properties of the Poisson process.

1.2 Thesis contributions and organization

In this thesis, we focus on discrete-state pure-jump processes in continuous-time where any finite time interval has only a finite number of state transitions. Generalizing a classical idea called *uniformization* (Jensen, 1953), we show how it is often possible to

characterize such systems as discrete-time systems resulting from a *random* discretization of time. In constrast to methods that discretize time with a regular grid, randomization allows us to eliminate bias using a much coarser grid; resulting a discrete-time systems with fewer time steps. Given such a characterization, we proceed to develop auxiliary variable Gibbs samplers that alternately resample the system trajectory given the random discretization, and then a new time-discretization given the system trajectory. Fot the first step, we can leverage the large literature on MCMC for discrete-time systems, bringing their power to continuous-time problems. We exploit properties of the Poisson process to carry out the second step efficiently.

We briefly outline the contents of the various chapters.

- Chapter 2 provides a review of properties of the Poisson process. The Poisson process is fundamental to the ideas developed in this thesis. Firstly, it forms the basis of different randomized characterizations of the systems we will study. Additionally, the efficiency of our algorithms stem, in large part, from the independence or memoryless properties of the Poisson process. We also introduce the notion of the probability density of the Poisson process. This will prove useful in representing probabilities of the more complicated systems we study, and will allow us to prove results more easily.

- Chapter 3 describes our first contribution, an MCMC sampler for Markov jump processes. Our sampler is based on the idea of uniformization and was described in (Rao and Teh, 2011a). In additional to work from that paper, we also apply our sampler to the Markov-modulated Poisson process, where we show it to be significantly more efficient than a state-of-the-art sampler designed specifically for this model.

- Chapter 4 is also based on work in (Rao and Teh, 2011a) and applies our ideas to a class of structured MJPs called continuous-time Bayesian networks. We elaborate on the material from that paper, showing on how to extend our sampler from chapter 3 for greater efficiency and why the resulting algorithm is correct. In doing so, we lay the seeds for the more general thinning schemes described in chapter 6.

- In Chapter 5 we move beyond memoryless systems to renewal processes. These are generalizations of the Poisson process with arbitrary waiting times. To demonstrate the usefulness of our sampler, we describe a non-stationary renewal process that generalizes work by Adams et al. (2009) for doubly stochastic Poisson processes. We should how uniformization allows us to draw exact samples from the model and allows us to perform efficient inference. This work was published in (Rao and Teh, 2011b).

- In Chapters 6 and 7, we move beyond the framework of uniformizing to a more general scheme of dependent thinning. Working in such a framework allows us to

extend our methods to a wider class of models with unbounded event rates. This framework also affords us more flexibility to trade-off the computational cost per MCMC iteration and the independence across MCMC samples. Work from these chapters is described in a paper that is under submission.

- Chapter 8 shows how our ideas can be adapted to processes on more general spaces. In particular, we look at a class of 'repulsive processes' on a 2-dimensional Euclidean space. These processes also have a thinning construction starting from an underlying Poisson process. We show how our sampler can allow us to perform efficient MCMC inference on this class of models. Part of the work in this chapter was done with David Dunson at Duke University.

- Finally, we end with a summary, and a discussion of possible avenues for future research in Chapter 9.

Chapter 2

The Poisson process

2.1 Introduction

Most of the systems studied in this thesis are built upon an underlying Poisson process. Even if it may not be traditional to view these systems this way, the MCMC sampling algorithms that we will describe exploit this construction, and involve reconstructing this latent Poisson process at some stage. With this in mind, we begin with an introduction to the Poisson process. Though most of this thesis is devoted to processes on the real line, we do consider point processes on more general Euclidean spaces in chapter 8, with further generalizations possible. Moreover, it is useful to distinguish between properties intrinsic to the Poisson process, and those imposed by the ordering of the real line. Thus, our exposition will follow Kingman (1993) by considering Poisson processes on general spaces. We will review the Poisson process on the real line towards the end of this chapter, where we establish some conventions that we will follow for all stochastic processes on the real line. In our study of the Poisson process, we will also introduce the notion of the probability density of a Poisson process (Daley and Vere-Jones, 2008), something that will simplify calculations in later chapters.

2.2 The Poisson process

Informally, the Poisson process is a probability distribution over countable subsets of some space. A sample from this process is thus a collection of points in the space, so that the Poisson process is an example of a *stochastic point process*. The space in which the points lie is commonly the real line or a d-dimensional Euclidean space (often subsets thereof), but more generally can be some complete, separable, metric space $\mathcal{T}$. Let Σ be the Borel σ-algebra on $\mathcal{T}$, generated by the open sets in the topology on $\mathcal{T}$. Let Π be the random set drawn from the Poisson process, and for any measurable* set

*We shall henceforth deal only with measurable sets, and occasionally drop this qualifier.

$A \in \Sigma$, let $\Pi(A)$ denote the number of points lying in A:

$$\Pi(A) = \#\{\Pi \cap A\} \tag{2.1}$$

Then for some measure $\Lambda(\cdot)$ on $(\mathcal{T}, \Sigma)$, we have the following definition:

Definition 2.1. *A Poisson process with mean $\Lambda(\cdot)$ is a stochastic point process satisfying the following properties:*

1. *The number of points in disjoint subsets of $\mathcal{T}$ are independent random variables.*
2. *The number of points in any set A, $\Pi(A)$, is Poisson distributed with mean $\Lambda(A)$.*

Recalling the definition of the Poisson distribution, we have that

$$P(\Pi(A) = n) = \frac{\Lambda(A)^n \exp(-\Lambda(A))}{n!} \tag{2.2}$$

Observe that to distinguish individual points, we need Σ to include the singleton sets $\{t\}$ $\forall t \in \mathcal{T}$. Moreover, since the random set Π either includes or does not include a point t, $\Pi(\{t\})$ can either equal 0 or 1. It follows from equation (2.2) that for the Poisson process to be well defined, the measure $\Lambda(\cdot)$ must be non-atomic, so that

$$\Lambda(\{t\}) = 0 \quad \forall t \in \mathcal{T} \tag{2.3}$$

It turns out that the non-atomicity of the mean measure is almost sufficient to guarantee the existence of the Poisson process; all that is additionally required is a mild σ-finiteness condition that Λ be expressible as a countable sum of finite measures:

Theorem 2.1. (Existence theorem, Kingman (1993)) *Let Λ be a non-atomic measure on $\mathcal{T}$ which can be expressed in the form*

$$\Lambda = \sum_{n=1}^{\infty} \Lambda_n, \qquad \Lambda_n(\mathcal{T}) < \infty \tag{2.4}$$

Then there exists a Poisson process on $\mathcal{T}$ having Λ as its mean measure.

While it is common to think of realizations Π of a Poisson process on $(\mathcal{T}, \Sigma)$ as 'collections of points in $\mathcal{T}$', the standard approach to constructing a probability space for this stochastic process involves viewing Π as a *random measure* on $(\mathcal{T}, \Sigma)$ (Kingman, 1993). In particular, a sample Π from a Poisson process is an element of $\mathcal{N}$, the space of integer-valued measures on $\mathcal{T}$. The measure $\Pi(A)$ of any subset $A \in \Sigma$ is the number of points in that subset. This defines a map $\pi_A(\Pi) : \mathcal{N} \to \mathbb{Z}^{+\dagger}$; the σ-algebra on $\mathcal{N}$ is then defined as the σ-algebra generated by the set of maps π_A for all $A \in \Sigma$. Call

[†] $\mathbb{Z}^+$ being the space of nonnegative integers

this $\Sigma_{\mathcal{N}}$; thus $\Sigma_{\mathcal{N}}$ is the smallest σ-algebra containing the sets $\pi_A^{-1}(B)$ for all $A \in \Sigma$ and all sets of nonnegative integers B. Completing the specification of the probability triplet, one requires that for some measure Λ (the mean measure) and any disjoint sets $A_1, A_2, ..., A_n$ in Σ, the random variables $\Pi(A_i)$ are Poisson distributed with mean $\Lambda(A_i)$ and are jointly independent. The existence theorem tells us that there does exist a probability measure P_Λ that satisfies this, and the triplet $(\mathcal{N}, \Sigma_{\mathcal{N}}, P_\Lambda)$ constitutes the Poisson process. In this context, the independence property of the Poisson process is sometimes called *complete randomness*, so that the Poisson process is an example of a completely random measure (Kingman, 1993).

2.3 Properties of the Poisson process

We now list a number of useful properties of the Poisson process; we refer the interested reader to Kingman (1993) for proofs. In section 2.4, we provide some elementary proofs for the special case when the Poisson mean measure is finite (i.e. $\Lambda(\mathcal{T}) < \infty$).

Theorem 2.2. (Disjointness theorem, Kingman (1993)) *Let Π_1 and Π_2 be independent Poisson processes on $\mathcal{T}$, and let A be a measurable set with $\Lambda_1(A)$ and $\Lambda_2(A)$ finite. Then Π_1 and Π_2 are disjoint with probability 1 on A:*

$$P\{\Pi_1 \cap \Pi_2 \cap A = \varnothing\} = 1 \tag{2.5}$$

The next theorem states the obvious but useful result that a Poisson process restricted to some measurable set is still a Poisson process, whose mean measure is now the restriction of the original mean measure to that set.

Theorem 2.3. (Restriction theorem, Kingman (1993)) *Let Π be a Poisson process with mean measure Λ on $\mathcal{T}$ and let T_1 be a measurable subset of $\mathcal{T}$. Then the random set $\Pi_1 = \Pi \cap T_1$ can be regarded either as a Poisson process on $\mathcal{T}$ with mean measure $\Lambda_1(A) = \Lambda(A \cap T_1)$ or as a Poisson process on T_1 whose mean measure is the restriction of Λ to T_1.*

This theorem is a straightforward consequence of the definition of the Poisson process. Besides allowing us to easily study restrictions of a Poisson process, it (along with the independence property) also allows us to define a 'global' Poisson process by combining independent Poisson processes defined on disjoint spaces. The next theorem tells us that we can also 'combine' independent Poisson processes defined on the *same* space $\mathcal{T}$ to get a Poisson process.

Theorem 2.4. (Superposition theorem, Kingman (1993)) *Let $\Pi_1, \Pi_2, \ldots$ be a countable collection of independent Poisson processes on $\mathcal{T}$, and let Π_n have mean measure Λ_n*

for each n. Then, their superposition

$$\Pi = \bigcup_{n=1}^{\infty} \Pi_n \tag{2.6}$$

is a Poisson process with mean measure

$$\Lambda = \sum_{n=1}^{\infty} \Lambda_n \tag{2.7}$$

The following theorem characterizes the result of applying a transformation to the points sampled from a Poisson process; under appropriate conditions, this is still a Poisson process (with a different mean measure).

Theorem 2.5. (Mapping theorem, Kingman (1993)) *Let Π be a Poisson process with mean measure Λ on the space $\mathcal{T}_1$, and let $f : \mathcal{T}_1 \to \mathcal{T}_2$ be a measurable function such that the induced measure Λ^* has no atoms. Then $f(\Pi)$ is a Poisson process on $\mathcal{T}_2$ having Λ^* as its mean measure.*

A common mapping function is the projection operator $\pi : \mathcal{T}_1 \times \mathcal{T}_2 \to \mathcal{T}_2$, which maps a Poisson process on a product space down to one of the component spaces. The next theorem describes the 'inverse' of this projection, where a Poisson process is 'lifted' into an ambient product space. Towards this, let Π be a Poisson process on the space $\mathcal{T}$ with mean measure Λ. Consider a probability measure $P(t, \cdot)$ on some other space M (note that P is allowed to depend on t). Assign each point t of Π a random 'mark' drawn independently from $P(t, \mathrm{d}m)$, thus transforming it to a point in the product space $\mathcal{T} \times M$. Then the set

$$\Pi^* = \{(t, m_s) : t \in \Pi\} \tag{2.8}$$

forms a random subset of $\mathcal{T} \times M$. Not surprisingly, this is also a Poisson process:

Theorem 2.6. (Marking theorem, Kingman (1993)) *The random subset Π^* is a Poisson process on $\mathcal{T} \times M$ with mean measure Λ^* given by*

$$\Lambda^*(A) = \iint_{(t,m)\in A} \Lambda(\mathrm{d}t)\, P(t, \mathrm{d}m) \tag{2.9}$$

It is possible to generalize the mapping theorem from deterministic transformations to stochastic ones. Most of the time, these can be viewed as the result of sequentially applying the mapping and the marking theorems.

Finally, for completeness, we include two important properties of the Poisson process. The first is Campbell's theorem:

Theorem 2.7. (Campbell's theorem, Kingman (1993)) *Let* Π *be a Poisson process with mean measure* Λ *on the space* $\mathcal{T}$*. For a measurable function* $f : \mathcal{T} \to \mathbb{R}$*, the sum*

$$\Sigma = \langle \Pi, f \rangle = \sum_{t \in \Pi} f(t)$$

is absolutely convergent if and only if

$$\int_{\mathcal{T}} \min(|f(t)|, 1)\Lambda(\mathrm{d}t) < \infty$$

When this is true,

$$\mathbb{E}(e^{\theta\Sigma}) = \exp\left\{\int_{\mathcal{T}} \left(e^{\theta f(t)} - 1\right) \Lambda(\mathrm{d}t)\right\} \tag{2.10}$$

for any complex θ *for which the integral on the right converges (and thus whenever* θ *is purely imaginary). Moreover,*

$$\mathbb{E}(\Sigma) = \int_{\mathcal{T}} f(t)\Lambda(\mathrm{d}t) \tag{2.11}$$

if and only if the integral converges. In that case,

$$\mathrm{var}(\Sigma) = \int_{\mathcal{T}} f(t)^2\Lambda(\mathrm{d}t) \tag{2.12}$$

whether finite or infinite.

Equation (2.10) in Campbell's theorem gives the characteristic functional of the Poisson process; if, for some stochastic point process, this holds for a sufficiently large class of functions, we can establish that point process to be Poisson. Bertoin (2006) calls equation (2.10) *Campbell's formula*, though some authors (eg. Daley and Vere-Jones (2008)) use that name to refer to equation (2.11).

Now, for an event, $B \in \Sigma_{\mathcal{N}}$ and any event $A \in \Sigma$, define *Campbell's measure*, $\mathcal{C}(B, A)$ as:

$$\mathcal{C}(B, A) = \mathbb{E}\left[1_B(\Pi)\Pi(A)\right] \tag{2.13}$$

Here 1_B is the indicator function for the set B, so that the expression above gives (upto a normalization constant) the expected number of Poisson events in A conditioned on B being true. Clearly, for any B, $\mathcal{C}(B, \cdot)$ is absolutely continuous w.r.t. $\Lambda(\cdot)$, allowing us to define the Radon-Nikodym derivative:

$$\frac{\mathrm{d}\mathcal{C}(B, \cdot)}{\mathrm{d}\Lambda}(t) = P^t_\Lambda(B) \tag{2.14}$$

The quantity $P^t_\Lambda(\cdot)$ corresponds (again, upto a normalization factor) to a probability

measure on $(\mathcal{N}, \Sigma_{\mathcal{N}})$, and is called the *Palm distribution*. The Palm distribution of a point process can be thought as the posterior distribution given that there is an event at location t. The independence property of the Poisson property suggests that the distribution of Π under the posterior P_Λ^t is identical the distribution of $\Pi + \delta_t$, with Π distributed according to the Poisson process prior. Not only is this true, but this also characterizes the Poisson process (Zuyev, 2006):

Theorem 2.8. (Slivnyak's theorem, (Slivnyak, 1962), or the Palm formula, (Bertoin, 2006)) *Let Π be a Poisson process with mean measure Λ on the space $\mathcal{T}$. For a non-negative measurable functional $G : \mathcal{T} \times \mathcal{N} \to \mathbb{R}^+$,*

$$\mathbb{E}\left(\langle \Pi, G(\cdot, \Pi)\rangle\right) \equiv \mathbb{E}\left(\sum_{t\in\Pi} G(t, \Pi)\right) = \int_{\mathcal{T}} \mathbb{E}\left(G(t, \delta_t + \Pi)\Lambda(\mathrm{d}t)\right) \tag{2.15}$$

Letting $\mathbb{E}^t$ denote expectations with respect to P_Λ^t, we then also have the *refined Campbell formula*:

$$\mathbb{E}\left(\langle \Pi, G(\cdot, \Pi)\rangle\right) = \int_{\mathcal{T}} \mathbb{E}^t\left(G(t, \Pi)\right)\Lambda(\mathrm{d}t) \tag{2.16}$$

Compare this with equation (2.11), where the functional $G(t, \Pi)$ was restricted to be independent of Π.

The results above apply to general Poisson processes, and for a demonstation of their usefulness in this context, see for example Rao and Teh (2009). Instead, from now on, we shall limit ourselves to the special case of Poisson processes with finite realizations. In this case, the Poisson process has an intuitive constructive definition that will obviate the need for any existence proofs. Moreover, many of the Poisson properties listed above will follow intuitively from this definition. Following Daley and Vere-Jones (2008), we shall also introduce the notion of the *probability density* of a Poisson process, and simple manipulations of this density will allow us to establish the remaining Poisson properties.

2.4 Finite Poisson processes

Suppose that the total measure of the space $\mathcal{T}$ is finite, i.e. $\Lambda(\mathcal{T}) < \infty$. By definition, the total number of Poisson events, $\#\Pi = \Pi(\mathcal{T})$ is Poisson distributed with mean $\Lambda(\mathcal{T})$, and is thus finite almost surely. Let $\{A_1, A_2, \cdots, A_p\}$ be a partition of the space $\mathcal{T}$ (i.e. these are disjoint sets whose union is $\mathcal{T}$). Then, from the definition of the Poisson

process,

$$P(\Pi(A_1) = n_1, \Pi(A_2) = n_2, \cdots, \Pi(A_p) = n_p) = \prod_{i=1}^{p} \frac{\Lambda(A_i)^{n_i} \exp(-\Lambda(A_i))}{n_i!} \tag{2.17}$$

Conditioning on $\Pi(\mathcal{T}) = \sum_{i=1}^{p} n_i = n$, it follows from Bayes' rule that

$$P(\Pi(A_1) = n_1, \cdots, \Pi(A_p) = n_p | \Pi(\mathcal{T}) = n) = \frac{n!}{\Lambda(\mathcal{T})^n} \prod_{i=1}^{p} \left(\frac{\Lambda(A_i)^{n_i}}{n_i!} \right) \tag{2.18}$$

$$= \frac{n!}{n_1! \cdots n_p!} \prod_{i=1}^{p} \left(\frac{\Lambda(A_i)}{\Lambda(\mathcal{T})} \right)^{n_i} \tag{2.19}$$

Equation (2.19) is just the multinomial distribution; since this is true for any measurable partition of $\mathcal{T}$, it follows that conditioned on the total number of Poisson events, the locations of the points are i.i.d. variables drawn from the probability measure $\frac{\Lambda(\cdot)}{\Lambda(\mathcal{T})}$.

This last insight provides a very intuitive way to think about the finite mean Poisson process: first sample the number of events from a Poisson distribution with mean $\Lambda(\mathcal{T})$, and then sample their locations i.i.d. from the normalized mean measure. We shall use this as the definition of the Poisson process. Such a constructive definition has the advantage of allowing us to bring geometric intuition into our handling of the Poisson process. Furthermore, by limiting ourselves to the finite mean Poisson process, we can introduce the notion of the probability density of a Poisson process, something that once again is more intuitive than the traditional approach of dealing with characteristic functionals of Poisson processes. This will prove useful in later sections when we study other stochastic processes based on an underlying Poisson process. Working with densities will, for instance, allow the straightforward application of Bayes' rule, allowing us to make inferences about the latent Poisson process.

We emphasize again that these ideas apply only to Poisson processes whose realizations are finite almost surely. We shall consequently consider a Poisson process as a stochastic process taking values in the space $\mathcal{T}^{\cup}$ of all finite sequences in $\mathcal{T}$. We shall refer to elements of this space (and thus realizations of the Poisson process) as T, rather than Π or Π^* (though sometimes, we use other capital letters like S, E or G). For each finite n, let $\mathcal{T}^n$ be the n-fold product space of $\mathcal{T}$, equipped with the usual product σ-algebra, Σ^n. We shall refer to elements of $\mathcal{T}^n$ as T^n. Note that T^n is a random sequence of length n, i.e. $T^n \equiv (t_1, \cdots, t_n)$. We define $\mathcal{T}^0$ as a single point satisfying $\mathcal{T}^0 \times \mathcal{T} = \mathcal{T} \times \mathcal{T}^0 = \mathcal{T}$ and equip it with the trivial σ-algebra $\Sigma^0 = \{\emptyset, \mathcal{T}^0\}$. Then, define $\mathcal{T}^{\cup} \equiv \bigcup_{i=0}^{\infty} \mathcal{T}^i$ as the resulting union space, which we equip with the σ-algebra $\Sigma^{\cup} \equiv \{\bigcup_{i=0}^{\infty} A^i \ \forall A^i \in \Sigma^i\}$. Thus,

$$B \cap \mathcal{T}^n \in \Sigma^n \quad \forall B \in \Sigma^{\cup} \tag{2.20}$$

Next, assume a measure μ on $(\mathcal{T}, \Sigma)$ (for Euclidean spaces, μ is typically the Lebesgue measure). Letting μ^n be the n-fold product measure on the product space $(\mathcal{T}^n, \Sigma^n)$, assign any set $B \in \Sigma^\cup$ the measure

$$\mu^\cup(B) = \sum_{i=1}^{\infty} \frac{1}{i!} \mu^i(B \cap \mathcal{T}^i) \tag{2.21}$$

$$= \sum_{i=1}^{\infty} \int_{B \cap \mathcal{T}^i} \frac{1}{i!} \mu^i(\mathrm{d}T^i) \tag{2.22}$$

Now assume the Poisson mean measure Λ admits a density λ with respect to μ, so that

$$\Lambda(A) = \int_A \lambda(t) \mu(\mathrm{d}t), \quad A \in \Sigma \tag{2.23}$$

Following our constructive definition of the Poisson process, we have that for a sequence T in $\mathcal{T}^\cup$ of length $|T| = n$,

$$P(T \in \mathrm{d}T) = \frac{\exp(-\Lambda(\mathcal{T}))}{n!} \prod_{j=1}^{n} \Lambda(\mathrm{d}t_j) \tag{2.24}$$

$$= \exp(-\Lambda(\mathcal{T})) \left(\prod_{j=1}^{n} \lambda(t_j) \right) \frac{\mu^n(\mathrm{d}T)}{n!} \tag{2.25}$$

$$= \exp(-\Lambda(\mathcal{T})) \left(\prod_{j=1}^{|T|} \lambda(t_j) \right) \mu^\cup(\mathrm{d}T) \tag{2.26}$$

Thus, the distribution of the random sequence T has density $\exp(-\Lambda(\mathcal{T})) \prod_{j=1}^{|T|} \lambda(t_j)$ w.r.t. the measure $\mu^\cup$. We shall call this density λ_s, i.e.

$$\lambda_s(T) = \exp(-\Lambda(\mathcal{T})) \prod_{j=1}^{|T|} \lambda(t_j) \tag{2.27}$$

The subscript s is a reminder that this density on sequences in $(\mathcal{T}^\cup, \Sigma^\cup)$ is symmetric in the coordinates. In particular, the sequence $T \equiv (t_1, \cdots, t_n)$ is an *ordered* sequence, whereas a random set sampled from a Poisson process is unordered. Under our representation, each of the $n!$ permutations of T corresponds to the same Poisson process realization and has the same density. The factorial term in the base measure (equation (2.21)) corrects for this many-to-one mapping. The density λ_s is sometimes called the *Janossy density* of the finite point process (Daley and Vere-Jones, 2008).

Theorem 2.9. (Density of a Poisson process) *The density $\lambda_s(T)$ on $(\mathcal{T}^\cup, \Sigma^\cup)$ is a probability density w.r.t. $\mu^\cup$, and the resulting stochastic process corresponds to a Poisson process on $(\mathcal{T}, \Sigma)$ with intensity $\lambda(t)$.*

Proof. That $\lambda_s(T)$ is a probability density can be easily seen:

$$\int_{\mathcal{T}^\cup} \lambda_s(T)\mu^\cup(\mathrm{d}T) = \exp(-\Lambda(\mathcal{T}))\sum_{n=0}^{\infty}\int_{\mathcal{T}^n}\lambda_s(T)\mu^\cup(\mathrm{d}T) \tag{2.28}$$

$$= \exp(-\Lambda(\mathcal{T}))\sum_{n=0}^{\infty}\frac{1}{n!}\int_{\mathcal{T}^n}\prod_{j=1}^{n}\lambda(t_j)\mu^n(\mathrm{d}T^n) \tag{2.29}$$

$$= \exp(-\Lambda(\mathcal{T}))\sum_{n=0}^{\infty}\frac{1}{n!}\Lambda(\mathcal{T})^n \tag{2.30}$$

$$= 1 \tag{2.31}$$

Next, consider the event that k out of n events lie in a set A. Call this event B_n. Since there are $\binom{n}{k}$ ways of choosing k out of n components, it follows that the probability of B_n is given by

$$P(B_n) = \frac{n!}{k!(n-k)!}\exp(-\Lambda(\mathcal{T}))\int_{B_n}\prod_{j=1}^{|T|}\lambda(t_j)\mu^\cup(\mathrm{d}T) \tag{2.32}$$

$$= \frac{1}{k!(n-k)!}\exp(-\Lambda(\mathcal{T}))\int_{B_n}\prod_{j=1}^{|T|}\lambda(t_j)\mu^n(\mathrm{d}T^n) \tag{2.33}$$

$$= \frac{\exp(-\Lambda(\mathcal{T}))}{k!(n-k)!}\Lambda(A)^k\left(\Lambda(\mathcal{T})-\Lambda(A)\right)^{n-k} \tag{2.34}$$

Define B as the event that k events occur in the set A. It follows that

$$P(B) = \sum_{n=k}^{\infty}P(B_n) \tag{2.35}$$

$$= \exp\left(-\Lambda(\mathcal{T})\right)\sum_{n=k}^{\infty}\frac{\Lambda(A)^k(\Lambda(\mathcal{T})-\Lambda(A))^{n-k}}{k!(n-k)!} \tag{2.36}$$

$$= \frac{\Lambda(A)^k\exp\left(-\Lambda(\mathcal{T})\right)}{k!}\sum_{i=0}^{\infty}\frac{(\Lambda(\mathcal{T})-\Lambda(A))^i}{i!} \tag{2.37}$$

$$= \frac{\Lambda(A)^k\exp(-\Lambda(A))}{k!} \tag{2.38}$$

Thus, the number of events in a set A is Poisson distributed with mean $\Lambda(A)$ for any $A \in \Sigma$. It remains to show that for disjoint sets $\{A_1, \cdots, A_p\}$, the random variables $\{\Pi(A_1), \cdots, \Pi(A_p)\}$ are independent. This follows from Rényi's theorem (corollary 12.9, (Kallenberg, 2002)), so that the density λ_s defines a Poisson process with mean Λ (and intensity λ). □

2.5 Properties of the finite Poisson process

Clearly, all the properties listed in section 2.3 continue to hold in the finite case. Nevertheless, it is worth reviewing them in light of the intuitive properties of the finite Poisson process. For instance, both the disjointness theorem and the restriction theorem are obvious consequences of the constructive definition of the Poisson process. Similarly, recall the construction of the point process Π^* (equation (2.8)) outlined in the marking theorem; this involved assigning every element $t_i \in \Pi$ a random mark $m_i \sim P(t_i, \cdot)$. In the finite case, this corresponds to sampling a Poisson($\Lambda(\mathcal{T})$) number of points, and then drawing their locations in $\mathcal{T} \times M$ i.i.d. from $\frac{\Lambda(\mathrm{d}t)P(t,\mathrm{d}m)}{\Lambda(\mathcal{T})}$ by first drawing t from $\Lambda(\cdot)/\Lambda(\mathcal{T})$ and then drawing m conditionally from $P(t, \mathrm{d}m)$. It follows by definition that Π^* is a Poisson process on $\mathcal{T} \times M$ with mean measure $\Lambda(\mathrm{d}t)\, P(t, \mathrm{d}m)$.

Now, consider an application of the marking theorem where the marks only take k discrete values (so that $P(t, \cdot)$ is a discrete distribution for any t). Let T be the original Poisson process and T^* the marked Poisson process. The restriction theorem implies that T_i^*, the subset of points in T^* with mark i, is also a Poisson process with mean measure $\Lambda(\mathrm{d}t)P(t, i)$ (and intensity $\lambda(t)P(t, i)$).

As an important and well known consequence, consider two Poisson processes with intensities $\lambda^*(t)$ and $\lambda(t)$, with $\lambda^*(t) \geq \lambda(t)\, \forall t \in \mathcal{T}$. Let T^* be the random set sampled from the first Poisson process, and to each point $t \in T^*$ assign one of two marks, 'O' (or 'keep') with probability $\frac{\lambda(t)}{\lambda^*(t)}$ and 'X' (or 'thin') with probability $\left(1 - \frac{\lambda(t)}{\lambda^*(t)}\right)$. See figure 2.1, and define T as the set of points with label 'O' (the 'kept' points). Then:

Theorem 2.10 (Thinning theorem, Lewis and Shedler (1979))**.** *The random set T is a draw from a Poisson process with intensity $\lambda(t)$.*

Proof. We saw in the previous section that this is true from the marking and restriction theorems; however, we provide an alternate proof to show the utility of the Poisson process density introduced in section 2.4.

Recall that the sample T^* has density

$$P(T^*) = \exp\left(-\Lambda^*(\mathcal{T})\right) \prod_{t \in T^*} \lambda^*(t) \tag{2.39}$$

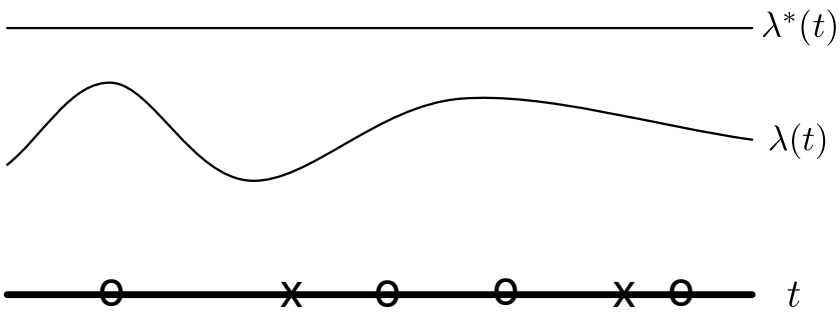

Figure 2.1: Thinning a sample from a homogeneous Poisson process

Let $T = \{t_1, \cdots, t_n\}$ denote the subset of T^* that was assigned the mark 'keep'. Let $T^c = T^* \setminus T$ be its complement. Let the size of T^c be k, so that $|T|^* = (n+k)$. Since both T^* and T are unordered, there are $\binom{n+k}{n}$ was of choosing T from T^*. Then,

$$P(T^*, T) = \frac{(n+k)!}{n!k!} \prod_{t \in T^c} \left(1 - \frac{\lambda(t)}{\lambda^*(t)}\right) \prod_{t \in T} \frac{\lambda(t)}{\lambda^*(t)} \left(\exp\left(-\Lambda^*(\mathcal{T})\right) \left(\prod_{t \in T^*} \lambda^*(t)\right)\right) \tag{2.40}$$

$$= \frac{(n+k)!}{n!k!} \exp\left(-\Lambda^*(\mathcal{T})\right) \prod_{t \in T^c} \left(\lambda^*(t) - \lambda(t)\right) \prod_{t \in T} \lambda(t) \tag{2.41}$$

Integrating out the locations in T^c (and recalling that $|T^c| = k$), we have

$$P(T, k) = \frac{(n+k)!}{n!k!} \exp\left(-\Lambda^*(\mathcal{T})\right) \left(\frac{n!}{(n+k)!} \prod_{t \in T^c} \int_{\mathcal{T}} \left(\lambda^*(t) - \lambda(t)\right) \mu(\mathrm{d}t)\right) \prod_{t \in T} \lambda(t) \tag{2.42}$$

$$= \exp\left(-\Lambda^*(\mathcal{T})\right) \frac{1}{k!} \left(\Lambda^*(\mathcal{T}) - \Lambda(\mathcal{T})\right)^k \prod_{t \in T} \lambda(t) \tag{2.43}$$

The $\frac{k!}{(n+k)!} = \frac{1}{(n+k)(n+k-1)\cdots}$ term arises since $\mu^{\cup}(A \times B^{n-1}) = \frac{1}{n}\mu(A)\mu^{\cup}(B^{n-1})$ for $A \in \Sigma,\ B^{n-1} \in \Sigma^{n-1}$. Summing out k, we have

$$P(T) = \exp\left(-\Lambda^*(\mathcal{T})\right) \left(\prod_{t \in T} \lambda(t)\right) \sum_{k=0}^{\infty} \frac{1}{k!} \left(\Lambda^*(\mathcal{T}) - \Lambda(\mathcal{T})\right)^k \tag{2.44}$$

$$= \exp\left(-\Lambda^*(\mathcal{T})\right) \left(\prod_{t \in T} \lambda(t)\right) \exp\left(\Lambda^*(\mathcal{T}) - \Lambda(\mathcal{T})\right) \tag{2.45}$$

$$= \exp\left(-\Lambda(\mathcal{T})\right) \prod_{t \in T} \lambda(t) \tag{2.46}$$

This is the density of a Poisson process with intensity $\lambda(\cdot)$, completing the proof. □

The thinning theorem has important practical consequences, allowing one to sample from a Poisson process with a complicated intensity $\lambda(t)$ by thinning a sample from a simpler Poisson process whose intensity function $\lambda^*(t)$ dominates $\lambda(t)$ everywhere. Ob-

serve that our original construction of the Poisson process requires solving the possibly intractable integral

$$\Lambda(\mathcal{T}) = \int_{\mathcal{T}} \lambda(t)\, \mu(\mathrm{d}t) \tag{2.47}$$

$\Lambda(\mathcal{T})$ is both the mean of the Poisson distribution governing the number of events as well as the normalization constant of the distribution governing their locations. Sampling by the thinning construction only requires evaluating $\lambda(t)$ pointwise. The bounding intensity is usually taken to be a constant, see figure 2.1 for an illustration. The thinning theorem was exploited by Adams et al. (2009) to define an elegant non-parametric approach to modelling inhomogeneous Poisson processes. We review this work in chapter 5, where (among other things) we develop a more efficient MCMC sampler by exploiting a corollary to this theorem. The notion of independent thinning can also be relaxed to allow the marks depend on each other, allowing the construction of non-Poissonian processes from the original Poisson processes. The bulk of this thesis is devoted to developing and studying efficient MCMC algorithms for such models.

Note that by symmetry, the set of thinned points with label 'X', $T^c \equiv T^* \setminus T$ is also a Poisson process, now with intensity $\lambda^*(t) - \lambda(t)$. Moreover, the sets $\mathcal{T} \times \{\text{'O'}\}$ and $\mathcal{T} \times \{\text{'X'}\}$ are disjoint, so that by the independence property, these two Poisson processes are independent. Combined with the superposition theorem, we have the following simple corollary:

Corollary 2.1. *Let T be a Poisson process with intensity $\lambda(s)$, constructed by thinning a Poisson process T^* with a larger intensity $\lambda^*(t)$. Then, conditioned on T, the thinned points $T^c \equiv T^* \setminus T$ are distributed as an independent Poisson process with intensity $\lambda^*(t) - \lambda(t)$. After sampling T^c independently of T, the union $T \cup T^c$ is a Poisson process with intensity $\lambda^*(t)$.*

This result will prove very useful in subsequent chapters.

2.6 The Poisson process on the real line

We now address the important special case of the Poisson process on the real line. In this case, the space $\mathcal{T}$ is often referred to as 'time' and its elements are indexed by t. In most modelling applications, the process is defined on the nonnegative real line $\mathbb{R}^+$; in fact, in this thesis, we shall only consider finite intervals $[t_{start}, t_{end}]$ for some finite time t_{end} (resulting, when μ is the Lebesgue measure, in a finite Poisson process). The ordering of the real line means that Poisson events can be thought to occur sequentially, and in many contexts, it is convenient to view the Poisson process on the real line as a stochastic process with right-continuous piecewise-constant paths[‡]. The time between

[‡]We leave this development for later chapters.

successive events is referred to as the waiting time or the holding time, and it is well known that for a homogeneous Poisson process with intensity λ, the waiting time is exponentially distributed with rate λ. For the inhomogeneous case, given that the i^{th} event occurs at time t_i, the probability that the waiting time until the event $i+1$ is larger that τ is

$$P\left(t_{i+1} > t_i + \tau\right) = \exp\left(-\int_{t_i}^{t_i+\tau} \lambda(t)\mu(\mathrm{d}t)\right) \tag{2.48}$$

Note that this is a simple consequence of equation (2.2), since $P\left(t_{i+1} > t_i + \tau\right)$ is just the probability that no events occur in the interval $(t_i, t_i + \tau]$. The latter interpretation reveals that equation (2.48) is true whether or not an event occured at time t_i; thus, the memorylessness of the exponential distribution is identical to the independence property or the complete randomness of the Poisson process.

Now, consider a sample $T \equiv \{t_1, \cdots, t_n\}$ from a Poisson process on the interval $[t_{start}, t_{end}]$ whose intensity function is $\lambda(t)$ w.r.t. Lebesgue measure μ. Following section 2.4, the density of this sample w.r.t. $\mu^{\cup}$ is given by

$$P\left(T\right) = \exp\left(-\int_{t_{start}}^{t_{end}} \lambda(t)\mu(\mathrm{d}t)\right) \prod_{i=1}^{n} \lambda(t_i) \tag{2.49}$$

$$= \left(\prod_{i=1}^{n} \lambda(t_i) \exp\left(-\int_{t_{i-1}}^{t_i} \lambda(t)\mu(\mathrm{d}t)\right)\right) \exp\left(-\int_{t_n}^{t_{end}} \lambda(t)\mu(\mathrm{d}t)\right) \tag{2.50}$$

where $t_0 = t_{start}$.

For the homogeneous case, equation (2.50) has the simple interpretation as the density of n independent draws $(t_1, t_2 - t_1, \cdots, t_n - t_{n-1})$ from a rate λ exponential multiplied by the probability of a final draw larger than $(T - t_n)$. Using equation (2.48), this generalizes to the inhomogeneous case, so that the probability density of the next Poisson event after an event at time t_i is given by

$$P\left(t_{i+1} = t_i + \tau\right) = \lambda(t_i + \tau) \exp\left(-\int_{t_i}^{t_i+\tau} \lambda(t)\mu(\mathrm{d}t)\right) \tag{2.51}$$

The ordered structure of the real line makes it possible to associate a sample from the Poisson process on the real line with a unique *ordered* sequence. In particular, we can associate the random set $\{t_1, \cdots, t_n\}$ with the random sequence $(t_1, \cdots, t_n)$, where $t_i < t_{i+1}$. This is more intuitive that the many-to-one mapping outlined in section 2.4 for Poisson processes on general spaces, and in this thesis, we shall follow this convention for all stochastic processes on the real line. Thus, we shall define a Poisson process on the interval $[t_{start}, t_{end}]$ as a random sequence of random length, obtained by sequentially sampling times $t_1, t_2, \cdots$ etc from equation (2.51) until we exit the interval. Implicitly, this sequential construction defines a distribution over the

sequence length n as well as a distribution over event times (note that latter is an *asymmetric* distribution, with t_i always less than t_{i+1}). Define the resulting density over length n sequences (w.r.t. μ^n) as

$$\lambda_a(T) = \exp\left(-\int_{t_{start}}^{t_{end}} \lambda(t)\,\mu(\mathrm{d}t)\right) \prod_{i=1}^{N} \lambda(t_i) \qquad t_{i+1} > t_i \ \forall i \tag{2.52}$$

$$= 0 \qquad \text{otherwise} \tag{2.53}$$

The subscript 'a' in λ_a is a reminder that the density is asymmetric in the components.

Again, as in section 2.4, we shall consider a Poisson process as an element of $(\mathcal{T}^{\cup}, \Sigma^{\cup})$, the union of the product spaces $(\mathcal{T}^n, \Sigma^n)$. We define a measure $\mu^{<}$ on this space as

$$\mu^{<}(B) = \sum_{i=0}^{\infty} \mu^i(B \cap \mathcal{T}^i) \tag{2.54}$$

Note that unlike the measure $\mu^{\cup}$ from equation (2.21), we do not need a factorial correction, since we associate each Poisson sample with a unique sequence in $\mathcal{T}^{\cup}$. It is clear from theorem 2.9 that for any $T \in \mathcal{T}^{\cup}$, the density $\lambda_a(T)$ w.r.t. the measure $\mu^{<}$ defines a Poisson process on $\mathcal{T}$.

We conclude by introducing the (nonstationary) *hazard rate function*, $h(t, \tau)$ (often just called the hazard function). We shall define this more carefully in chapter 5; for now, it suffices to say that this gives the instantaneous event rate at time t, τ time units after the last event. In other words, this is just the conditional density $P(t_{i+1} = t | t_i = t - \tau)$, and from equations (2.48) and (2.51), is given by

$$h(t, \tau) = \lambda(t) \tag{2.55}$$

This is yet another manifestation of the independence or memorylessness of the Poisson process: the rate at which events occur is independent of the past, depending only on the current value of the intensity function.

Equation (2.55) provides an intuitive way of thinking about the thinning theorem (theorem 2.10). Recall that the latter allows one to sample from a Poisson process with intensity $\lambda(\cdot)$ by first sampling from a Poisson process with intensity $\lambda^*(\cdot)$ that pointwise dominates $\lambda(\cdot)$, and then keeping a point located at t with probability $\lambda(t)/\lambda^*(t)$. Clearly, sampling from a higher rate Poisson process produces more events on average than the Poisson process of interest. The second step tells us that whether to keep or reject a point at t depends only on the values of the two intensity functions evaluated at t (as equation (2.55) suggests). In particular, if $\lambda^*(t)$ is c times larger than $\lambda(t)$, then we keep the point at t with probability $1/c$.

In later chapters, we shall look at a generalization of thinning called *uniformization*.

This will allow the construction of non-Poissonian systems with memory from an underlying Poisson process. Again, the idea is similar to thinning: we first sample a set of points from a Poisson process whose intensity at any time dominates the event rates of the process of interest. Now, rather than deleting points independently, we do so via a forward pass through the sample. The probability of keeping a point will be given by the ratio of the event rate of the stochastic process *conditioned on its instantiated past* divided by the instantaneous rate of the Poisson process. We will finally generalize this even further, by allowing the underlying point process to also be non-Poisson.

Chapter 3

Markov jump processes

3.1 Introduction

In this chapter, we introduce the Markov jump process (MJP), while the next chapter (chapter 4) focuses on a class of structured MJPs called continuous-time Bayesian networks (CTBNs). MJPs are one of the simplest continuous-time stochastic processes (see for example Çinlar (1975), or for a more rigorous introduction, Gikhman and Skorokhod (2004)), and find wide application in fields such as chemistry (Gillespie, 1977), genetics (Fearnhead and Sherlock, 2006), social network dynamics, (Fan and Shelton, 2009), human-computer interaction (Nodelman and Horvitz, 2003) etc. In all these applications, the MJP serves as a prior distribution over the trajectory of the state of a system which evolves in a piecewise-constant manner. Typically, this trajectory is only partially observed, and the challenge in the Bayesian framework is then to characterize the posterior distribution over trajectories (and any related parameters) given observations.

Our contribution in this chapter is to develop a novel and flexible Markov chain Monte Carlo (MCMC) sampler for posterior inference in MJPs. Our sampler is based on a generalization of the thinning theorem (theorem 2.10) called *uniformization*, that constructs an MJP sample from an underlying Poisson process. In our experiments, we demonstrate state-of-the-art performance with our sampler, its efficiency stemming from the independence property of the Poisson process and the Markov property of MJP.

We start with a review of Markov jump processes in section 3.2. In section 3.3 we introduce the idea of uniformization, and in section 3.5, we describe our MCMC sampler for the simple case of a homogeneous Markov jump process with discrete, noisy observations. In section 3.6, we apply our sampler to the more complicated Markov modulated Poisson process. In our experiments, we study properties of our sampler and compare it to a state-of-the-art sampler for Markov modulated Poisson processes.

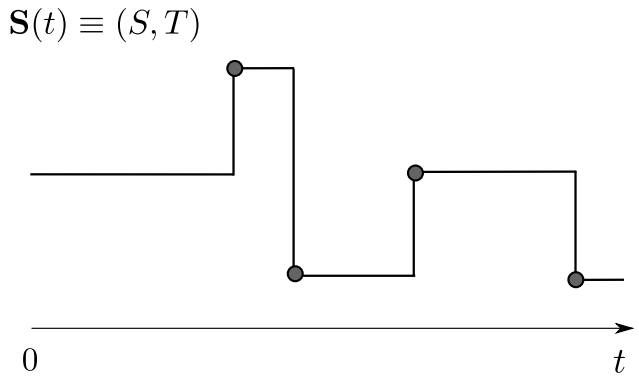

Figure 3.1: Sample path from a Markov jump process

We end with a discussion in section 3.7.

3.2 Markov Jump Processes

A Markov jump process $\{\mathbf{S}(t), t \in [0, \infty)\}$ is a stochastic process on the nonnegative real line $\mathbb{R}^+$ with right-continuous, piecewise-constant paths (see figure 3.1). The state space of the paths is some measurable space $(\mathcal{S}, \Sigma)$ which is typically assumed to be countable (and often, finite), though neither need be the case (see for instance, Breuer (2003)). We require that the singletons sets are measurable, i.e. $\{s\} \in \Sigma,\ \forall s \in \mathcal{S}$.

Let $\mathcal{F}$ be the space of right-continuous piecewise-constant functions $f : \mathbb{R}^+ \to \mathcal{S}$ from the nonnegative real line to $\mathcal{S}$. Thus, for every $f \in \mathcal{F}$ and $t \in \mathbb{R}^+$, there exists a δ such that $f(t + h) = f(t)\ \forall h \in [0, \delta)$. For each t, define a map $\pi_t : \mathcal{F} \to \mathcal{S}$ projecting an element of $\mathcal{F}$ to its value at the time instant t. We endow $\mathcal{F}$ with the σ-algebra generated by the sets $\{\pi_t^{-1}(A) : t \in [u, v]\},\ \forall u < v \in \mathbb{R}^+, A \in \Sigma$; call this σ-algebra $\Sigma_{\mathcal{F}}$. The triplet (u, v, A) corresponds to the set of all functions in $\mathcal{F}$ taking values in A at all times in the interval $[u, v]$. Let P be a probability measure on $(\mathcal{F}, \Sigma_{\mathcal{F}})$, and let $\mathbf{S}(t)$ denote the induced random variable on $\mathcal{S}$ at any time t. Then the stochastic process $\mathbf{S} = \{\mathbf{S}(t)\}_{t \geq 0}$ associated with the triplet $(\mathcal{F}, \Sigma_{\mathcal{F}}, P)$ is a *pure jump process*.

The stochastic process $\mathbf{S}$ is called a *Markov* process if for all $u < t$ and $A \in \Sigma$,

$$P\left(\mathbf{S}(t) \in A | \{\mathbf{S}(t') : 0 \leq t' \leq u\}\right) = P\left(\mathbf{S}(t) \in A | \mathbf{S}(u)\right) \tag{3.1}$$

This is the familiar notion that given the present, the future is independent of the past.

Finally, we introduce the notion of stochastic continuity: this means that the probability of a jump at any time is 0. Thus, a pure jump process is *stochastically continuous* when

$$\lim_{h \to 0} P\left(\mathbf{S}(t + h) \in A | \mathbf{S}(t) = s\right) = 1_A(s) \tag{3.2}$$

for all $t \in \mathbb{R}^+, s \in \mathcal{S}$ and $A \in \Sigma$ (here $1_A(\cdot)$ is the indicator function for the set A).

Definition 3.1. *A Markov jump process* $\mathbf{S}$ *is a stochastically continuous, pure jump Markovian process on* $\mathbb{R}^+$.

Now, consider the event $Q^s_{t_1 t_2} = \{\mathbf{S}(t) = s \ \forall t \in (t_1, t_2] \,|\, \mathbf{S}(t_1) = s\}$. This is the (measurable) event that the MJP remains in state s over the interval $(t_1, t_2]$ given that it started in state s at time t_1. We denote its probability by $D(s, t_1, t_2)$, so that

$$D(s, t_1, t_2) = P\left(\mathbf{S}(t) = s \ \forall t \in (t_1, t_2] \,|\, \mathbf{S}(t_1) = s\right) \tag{3.3}$$

Denoting by $\tau(t_1)$ the time of the next jump after t_1, we also have that

$$D(s, t_1, t_2) = P\left(\tau(t_1) > t_2 \,|\, \mathbf{S}(t_1) = s\right), \quad t_2 > t_1 \tag{3.4}$$

Following Gikhman and Skorokhod (2004), it is not hard to see that $D(s, t_1, t_2)$ is monotonically nonincreasing in t_2, satisfies $\lim_{h \to 0^+} D(s, t, t + h) = 1$ as well as

$$D(s, t_1, t_2) = D(s, t_1, \tilde{t}\,) D(s, \tilde{t}, t_2) \quad \text{for } t_1 < \tilde{t} < t_2 \tag{3.5}$$

Consequently, we can show that D has the representation

$$D(s, t_1, t_2) = \exp\left(-\int_{t_1}^{t_2} q(s, u) \mathrm{d}u\right) \tag{3.6}$$

for a nonnegative function $q(s, u)$. We can view the state of the MJP as evolving with time, with $q(s, u)$ giving the rate of leaving state s at time u. It is worth comparing equation (3.6) with equation (2.48), the waiting time until the next event in a Poisson process. We see that if s is the current state of the MJP, then the waiting time until the next jump is identical to the waiting time until the next event of a Poisson process with intensity $q(s, u)$. Thus, from the independence property of the Poisson process, given the current state of the MJP, the time until the next jump is independent of the past.

We next look at characterizing the new state the process enters after leaving the current state. Gikhman and Skorokhod (2004) show that there are probability distributions $\Pi_t(s, \cdot)$ on $(\mathcal{S}, \Sigma)$ such that

$$\begin{aligned} &P\left(u < \tau(t_1) \leq v, \mathbf{S}(\tau(t_1)) \in B \,|\, \mathbf{S}(t_1) = s\right) \\ &\qquad = e^{-\int_{t_1}^{u} q(s,w) \mathrm{d}w} \left(\int_u^v e^{-\int_u^t q(s,w) \mathrm{d}w} q(s, t) \Pi_t(s, B) \mathrm{d}t\right) \end{aligned} \tag{3.7}$$

The interpretation of the terms in the equation above is clear. The first is the probability of staying in state s over the interval $(t_1, u]$, while $e^{-\int_u^t q(s,w) \mathrm{d}w} q(s, t)$ is the probability density of the time of the first jump after u occuring at time t (i.e. $\tau(u) = t$). The distributions $\Pi_t(s, \cdot)$ represent the probability of jumping from state s into some measurable set B, conditioned on the jump time being t. Finally, we integrate over all

possible values of t in $(u, v]$.

We can now define the infinitesimal generator A_t of a Markov jump process:

$$A_t(s, B) \equiv q(s, t) \left(\Pi_t(s, B) - 1_B(s)\right) \tag{3.8}$$

The generator can be viewed as the *net* rate at which the MJP in state s moves *into* the set B; observe that this quantity can be negative. The generator function (along with an initial distribution over states at time 0) specifies all marginal distributions of the MJP and thus, via Kolmogorov's extension theorem (Kallenberg, 2002), completely characterizes the Markov jump process.

For a *homogeneous* MJP, the generator function A_t as well as the leaving rates $q(s, t)$ and the distributions Π_t are independent of time, and we write them as $A(s, \cdot), q(s)$ and $\Pi(s, \cdot)$. It then follows from equation (3.6) that when the system is in state s, the time until the next jump is exponentially distributed with rate $q(s)$, after which the system samples a new state from the distribution $\Pi(s, \cdot)$. This is the essence of *Gillespie's algorithm*, whose description we leave until the next section.

3.2.1 Finite state MJPs

When the state space $\mathcal{S}$ of a Markov jump process is countable, it is common to map it to the space of natural numbers, i.e. $\mathcal{S} = \{1, 2, \cdots\}$. For a countable state-space, the infinitesimal generator is completely specified by the set of rates

$$A_t(s, \{s'\}) = q(s, t) \left(\Pi_t(s, s') - 1_{s'}(s)\right) \quad \forall s, s' \in \mathcal{S} \tag{3.9}$$

For an N-state MJP, A_t is just an N-by-N matrix. For the homogeneous case, we drop the subscript and call the generator A. This matrix A is called the *generator* or *rate* matrix of the Markov jump process. Its off-diagonal elements are non-negative, with $A_{ss'}$ representing the rate of transiting from state s to state s'. The diagonal entries are $A_s \equiv A_{ss} = -\sum_{s' \neq s} A_{ss'}$ for each s so that its rows sum to 0. $|A_s|$ characterises the total rate of leaving state s.

As mentioned earlier, the marginal distributions of the MJP, and thus the MJP itself is completely charaterized by the rate matrix A along with the initial distribution over states (we call the latter π_0). For a finite state MJP, we can calculate the marginal π_t at any time t analytically using the *matrix exponential*. In particular,

$$\pi_t = \exp(At)^\intercal \pi_0 \tag{3.10}$$

Recall that $\exp(At)$ is given by the (always converging) power series

$$\exp(At) = \sum_{i=0}^{\infty} \frac{1}{i!} (At)^i \tag{3.11}$$

The equation above hints at a connection with the Poisson process (see for example equation (2.2)); we shall reveal the exact nature of this relation in section 3.3.

We shall now restrict ourselves to MJPs on a finite time interval $\mathcal{T} \equiv [t_{start}, t_{end}]$. In this case, if all event rates are finite, a trajectory of the MJP will almost surely have only a finite number of jumps. Let these occur at the ordered times $(t_1, \cdots, t_n)$ (so that there are n state transitions). Define $T \equiv (t_0, t_1, \cdots, t_n, t_{n+1})$, where $t_0 = t_{start}$ and $t_{n+1} = t_{end}$. Let S be the corresponding sequence of states, i.e. $S = (s_0, s_1, \cdots, s_n, s_{n+1})$ where $s_i = \mathbf{S}(t_i)$. Observe that this pair (S, T) completely characterizes the MJP trajectory over $\mathcal{T}$; in fact, t_0 as well as the last pair (s_{n+1}, t_{n+1}) are redundant (we still include these for notational convenience). The filled circles in figure 3.1 represent the pair (S, T). We can sample (S, T) by the following generative process (called Gillespie's algorithm (Gillespie, 1977)):

Algorithm 3.1 Gillespie's algorithm to sample an MJP path on the interval $[t_{start}, t_{end}]$

Input: The rate matrix A and the initial distribution over states π_0.
Output: An MJP trajectory $\mathbf{S}(t) \equiv (S, T)$.

1: Assign the MJP a state $s_0 \sim \pi_0$. Set $t_0 = t_{start}$ and $i = 0$.
2: **while** $t_i < t_{end}$ **do**
3: Draw $z \sim \exp(|A_{s_i}|)$ and increment i.
4: Let $t_i = t_{i-1} + z$.
5: The MJP jumps to a new state s_i at time t_i, with
6: $$p(s_i = l | s_{i-1}) \propto A_{s_{i-1} l} \quad \forall l \neq s_{i-1}$$
7: **end while**
8: Set $t_i = t_{end}$, and $s_i = s_{i-1}$.

It is clear that so long as the leaving rates A_i are finite, the procedure above will terminate, returning a trajectory with a finite number of jumps.

3.3 Uniformization for MJPs

Gillespie's algorithm is the most direct way to sample a trajectory from an MJP. In this section, we introduce an alternate scheme called of *uniformization* (Jensen, 1953; Çinlar, 1975; Hobolth and Stone, 2009). While less intuitive than Gillespie's algorithm, uniformization establishes a direct connection between the Markov jump process and the Poisson process. In later sections, we shall exploit this construction (and the independence property of the Poisson process) to develop an efficient MCMC sampling algorithm for posterior inference in MJPs.

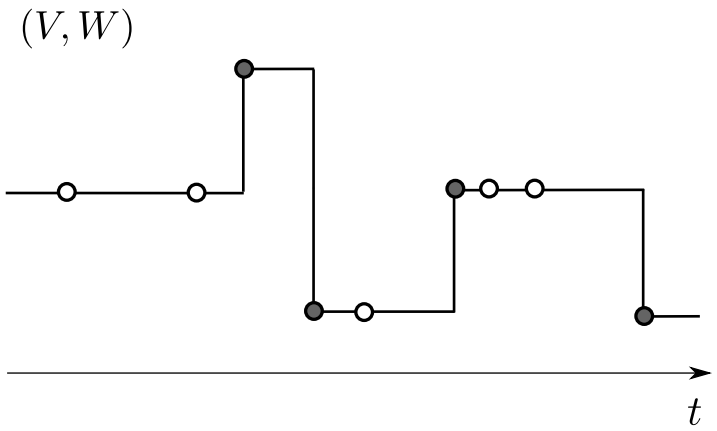

Figure 3.2: Uniformization: thin events from a *subordinating* Poisson process by running a discrete-time Markov chain on this set. The empty circles are the thinned events.

Recall that for an MJP with rate matrix A, element $A_{ss'}$, $s \neq s'$, is rate of leaving state s for s', while $|A_s|$ is the total rate of leaving state s. Choose some $\Omega \geq \max_s(|A_s|)$ and let $W = (w_1, \cdots, w_{|W|})$ be an ordered set of times on the interval $[t_{start}, t_{end}]$ sampled from a homogeneous Poisson process with intensity Ω. We will treat W as a set of candidate jump times for the MJP. Because the Poisson rate Ω dominates the leaving rates of all states of the MJP, W will on average contain more events than an MJP trajectory on the same interval will have jumps. Thus, we will construct an MJP sample from W by thinning events from W (as well as assigning states to the remaining times).

The thinning theorem (theorem 2.10) from the previous chapter constructs a sample from a Poisson process by independently deleting events from a Poisson process with a higher rate. The Markov structure of the MJP means that such independent thinning no longer applies. Instead, the Markov property implies that whether a point w_i should be kept or not depends only on the rate of exiting the state the MJP is currently in. This in turn is determined by the MJP state at the previous Poisson event w_{i-1}, since by construction, no intervening events could have occured. This suggests that deciding to thin or keep points in W amounts to running a discrete-time Markov chain on the set of times W. This is exactly what uniformization does.

Let I be the identity matrix, and define

$$B = \left(I + \frac{1}{\Omega}A\right) \tag{3.12}$$

Observe that B is a stochastic matrix with nonnegative elements and rows adding up to 1. Now, consider a discrete-time Markov chain with initial distribution π_0 and transition matrix B. Assign t_{start} a state drawn from π_0, and sequentially assign each time w_i in W a state v_i conditioning on the state v_{i-1} at w_{i-1}, with B as the

Markov transition operator. This discrete-time Markov chain (sometimes called the embedded Markov chain) is said to be *subordinated* to the original Poisson process. It will assign a sequence of states V to the times W; just as (S, T) characterizes an MJP path, (V, W) also characterizes a sample path of some piecewise-constant, right-continuous stochastic process on $[t_{start}, t_{end}]$. Importantly, note that unlike (S, T), the discrete-time trajectory represented by (V, W) can have self-transitions. We will call these *virtual jumps*, and treat (V, W) as a redundant representation of a continuous-time jump process without self-transitions. As the parameter Ω increases, the number of events in W increases; at the same time the diagonal entries of B start to dominate, so that the number of self-transitions also increases. The following theorem shows that these two effects exactly compensate each other, so that the process represented by (V, W) is precisely the desired MJP:

Theorem 3.1 (Uniformization theorem, Jensen (1953))**.** *For any* $\Omega \geq \max_s(|A_s|)$*,* (S, T) *and* (V, W) *define the same Markov jump process* $\mathbf{S}(t)$*.*

Proof. We follow Hobolth and Stone (2009). From equations (3.12) and (3.10),

$$\begin{aligned}
\pi_t &= \exp(At)^\intercal \pi_0 \\
&= \exp\left(\Omega(B - I)t\right)^\intercal \pi_0 \\
&= \exp\left(-\Omega t\right) \exp\left(\Omega t B\right)^\intercal \pi_0 \\
&= \sum_{i=0}^{\infty} \left(\left(\exp\left(-\Omega t\right) \frac{(\Omega t)^i}{i!} \right) \left(\left(B^\intercal\right)^i \pi_0 \right) \right)
\end{aligned} \tag{3.13}$$

The first term in the summation is the probability of a rate Ω Poisson producing i events in an interval of length t, i.e. $|W| = i$. The second term gives the marginal distribution over states for a discrete-time Markov chain after i steps, given that the initial state is drawn from π_0, and subsequent states are assigned according a transition matrix B. Summing over i, we obtain the marginal distribution over states at the end time t. Thus, the transition kernels induced by the uniformization procedure agree with those of the Markov jump process $(\exp(At))$ for all t. The two processes also share the same initial distribution of states, π_0. Consequently, all finite dimensional distributions also agree, so that following Kolmogorov's extension theorem (Kallenberg, 2002), both define the same stochastic process.

□

3.3.1 Probability densities for MJPs

It is clear from Gillespie's algorithm that if the transition rates of all states are bounded, then an MJP will make only a finite number of state transitions over a finite interval $\mathcal{T}$. As described earlier, the set of times T, and the corresponding states S completely

specify the MJP trajectory over $\mathcal{T}$. Recalling that the state space of the MJP is $\mathcal{S}$, it follows that a sample from an MJP can be viewed as a sequence of elements in the product space $\mathcal{M} \equiv \mathcal{S} \times \mathcal{T}$. Just as in section 2.4, we define $\mathcal{M}^i$ as the i-fold product space and then construct a union space $\mathcal{M}^\cup \equiv \bigcup_{i=1}^\infty \mathcal{M}^i$, elements of which represent finite length pure-jump paths. Recall from section 2.6 that $\mathcal{T}^\cup$ is the space of finite sequences in $\mathcal{T}$, and $\mu^<$ is measure defined on this space as:

$$\mu^<(B) = \sum_{i=1}^{\infty} \mu^i(B \cap \mathcal{T}^i) \tag{3.14}$$

$$= \sum_{i=1}^{\infty} \int_{B \cap \mathcal{T}^i} \mu^i(\mathrm{d}\mathcal{T}^i) \tag{3.15}$$

From Gillespie's algorithm, we see that sampling an MJP trajectory involves sequentially sampling waiting times from an exponential density and new states from a discrete distribution, both of which depend on the current state. This suggests that under an MJP, a random element (S, T) in $\mathcal{M}^\cup$ of length $|T| = |S|$ has density

$$P(S,T) = \pi_0(s_0) \left(\prod_{i=1}^{|T|-1} |A_{s_{i-1}}| e^{-|A_{s_{i-1}}|(t_i - t_{i-1})} \frac{A_{s_{i-1}s_i}}{|A_{s_{i-1}}|} \right) \cdot e^{-|A_{s_{|T|-1}}|(t_{|T|} - t_{|T|-1})} \tag{3.16}$$

$$= \pi_0(s_0) \left(\prod_{i=1}^{|T|-1} A_{s_{i-1}s_i} \right) \exp\left(- \int_{t_{start}}^{t_{end}} |A_{\mathbf{S}(t)}| \mathrm{d}t \right) \tag{3.17}$$

w.r.t. the measure $\mu^<$. The ith term in the product in equation (3.16) is the probability density of waiting for time $t_i - t_{i-1}$ in state s_{i-1} (with rate $A_{s_{i-1}}$), and then transitioning to state s_i. The last term is the probability of waiting for longer than $t_{|T|} - t_{|T|-1}$ in state $s_{|T|-1}$; this is because the last time $t_{|T|}$ does not correspond to an MJP transition (rather it is the end of the observation interval).

Theorem 3.2. (Density of a Markov jump process) *The Markov jump process is a stochastic process on* $(\mathcal{M}^\cup, \Sigma^\cup)$ *with a density w.r.t. the measure* $\mu^<$ *given by*

$$P(S,T) = \pi_0(s_0) \left(\prod_{i=1}^{|T|-1} A_{s_{i-1}s_i} \right) \exp\left(- \int_{t_{start}}^{t_{end}} |A_{\mathbf{S}(t)}| \mathrm{d}t \right) \tag{3.18}$$

Proof. We know the density of a Poisson process w.r.t. the measure $\mu^<$ (theorem 2.9). Also, the uniformization theorem tells us how to contruct an MJP trajectory from a sample from a Poisson process. Thus we can prove the result above by a simple application of the rules of probability. Accordingly, consider a sample W from a Poisson process with intensity Ω. Its density w.r.t. $\mu^<$ is

$$P(W) = \exp\left(-\Omega(t_{end} - t_{start})\right) \Omega^{|W|} \tag{3.19}$$

For convenience, in this section, we define W to not include the endpoints (though T does). Let the subordinated Markov chain change state n times. As defined previously, let $T = (t_0, \cdots t_{n+1})$ be this set of times (including the end times), so that $|T| = n + 2$, and let S be the corresponding state values. Let there be m_i virtual jumps on the interval (t_i, t_{i+1}), so that the Markov chain rejected m_i opportunities to change state before moving from s_i to s_{i+1}. Thus,

$$\begin{aligned} P(V, W) &= \exp\left(-\Omega(t_{end} - t_{start})\right) \Omega^{|W|} \\ &\left(\pi_0(s_0) \prod_{i=1}^{|T|-1} \left(\prod_{j=1}^{m_{i-1}} \left(1 - \frac{|A_{s_{i-1}}|}{\Omega}\right)\right) \frac{A_{s_{i-1}s_i}}{\Omega}\right) \left(\prod_{j=1}^{m_{|T|-1}} \left(1 - \frac{|A_{s_{|T|-1}}|}{\Omega}\right)\right) \qquad (3.20) \\ &= \pi_0(s_0) \exp\left(-\Omega(t_{end} - t_{start})\right) \\ &\left(\prod_{i=1}^{|T|-1} \left(\Omega - |A_{s_{i-1}}|\right)^{m_{i-1}} A_{s_{i-1}s_i}\right) \left(\Omega - |A_{s_{|T|-1}}|\right)^{m_{|T|-1}} \qquad (3.21) \end{aligned}$$

Next, integrate out the locations of the virtual jumps, so that

$$\begin{aligned} P(S, T, \{m_i\}) &= \pi_0(s_0) \exp\left(-\Omega(t_{end} - t_{start})\right) \prod_{i=1}^{|T|-1} A_{s_{i-1}s_i} \qquad (3.22) \\ &\prod_{i=1}^{|T|} \left(\left(\Omega - |A_{s_{i-1}}|\right)^{m_{i-1}} \int_{t_{i-1}}^{t_i} \cdots \int_{u_{m_{i-1}}}^{t_i} \mu(\mathrm{d}u_1) \cdots \mu(\mathrm{d}u_{m_{i-1}})\right) \\ &= \pi_0(s_0) \exp\left(-\Omega(t_{end} - t_{start})\right) \prod_{i=1}^{|T|-1} A_{s_{i-1}s_i} \\ &\prod_{i=1}^{|T|} \left(\left(\Omega - |A_{s_{i-1}}|\right)^{m_{i-1}} \frac{(t_i - t_{i-1})^{m_{i-1}}}{m_{i-1}!}\right) \end{aligned}$$

Here we have recognized that the integration above just calculates the volume of a

simplex of side $(t_i - t_{i-1})$. Summing out the m_i's, we get

$$P(S,T) = \pi_{s_0} \exp\left(-\Omega(t_{end} - t_{start})\right) \prod_{i=1}^{|T|-1} A_{s_{i-1}s_i} \tag{3.23}$$

$$\prod_{i=0}^{|T|-1} \left(\sum_{m_i=0}^{\infty} \frac{\left((\Omega - |A_{s_i}|)\,(t_{i+1} - t_i)\right)^{m_i}}{m_i!} \right)$$

$$= \pi_{s_0} \exp\left(-\Omega(t_{end} - t_{start})\right) \prod_{i=1}^{|T|-1} A_{s_{i-1}s_i} \prod_{i=0}^{|T|-1} \exp\left((\Omega - |A_{s_i}|)\,(t_{i+1} - t_i)\right)$$

$$= \pi_{s_0} \prod_{i=1}^{|T|-1} A_{s_{i-1}s_i} \prod_{i=0}^{|T|-1} \exp\left(-|A_{s_i}|\,(t_{i+1} - t_i)\right)$$

$$= \pi_{s_0} \left(\prod_{i=1}^{|T|-1} A_{s_{i-1}s_i} \right) \exp\left(- \int_{t_{start}}^{t_{end}} |A_{\mathbf{S}(t)}| \mathrm{d}t \right) \tag{3.24}$$

□

3.4 MJPs in Bayesian modelling applications

The Markov property of the MJP makes it both a realistic model for various physical and chemical phenomena, as well as a convenient approximation for more complex phenomena in biology, finance, queuing systems etc. In Bayesian modelling applications, the MJP plays the role of a prior over the state of some system that evolves in a piecewise-constant manner. Examples of such applications include chemical and biological systems, where the state of the MJP represents the sizes of various interacting *species* (eg. Gillespie (1977); Golightly and Wilkinson (2011)). In queuing applications, the state may represent the number of pending jobs in a queue (Asmussen, 2003; Breuer, 2003; Tijms, 1986), with the arrival and processing of jobs treated as independent events. Another application is genetics, where 'time' actually represents position along a strand of genetic matter. Here, the MJP trajectory represents a segmentation of, say, a strand of DNA, with different regions corresponding to, say, different mutation rates (Fearnhead and Sherlock, 2006; Rodrigue et al., 2008). MJPs also find wide application in finance, for example, Elliott and Osakwe (2006) use an MJP to model switches in the parameters that govern the dynamics of stock prices (the latter being modelled as a Lévy process).

In the applications listed above, the MJP trajectory is usually not observed completely. Instead, one is provided with observations at a discrete set of times, and often, these observations are noisy. In a Bayesian setting, one then attempts to characterize the posterior distribution over state-trajectories given these observations. This distribution is almost always *not* an MJP, and obtaining analytic characterizations of this is

impossible in all but the simplest situations. The situation is further complicated by the fact that one usually does not know the MJP parameters, and has to work with a prior distribution over these as well.

One approach is to deterministically approximate the intractable posterior with a simpler class of distributions. Examples of such an approach include mean-field approximations, expectation propagation and other variational approximations (Nodelman et al., 2002, 2005; Opper and Sanguinetti, 2007; Cohn et al., 2010). These methods have the disadvantage of being biased; moreover they are not easily adapted to situations where the MJP is part of a larger hierarchical model, for example, when one requires the posterior distribution over some unknown parameters as well. Consequently, Monte Carlo based inference methods are by far the most popular approach to approximating the MJP posterior. Various sampling based approximations have been proposed in the literature (Fearnhead and Sherlock, 2006; Boys et al., 2008; El-Hay et al., 2008; Fan and Shelton, 2008; Hobolth and Stone, 2009), but these also have disadvantages: usually they involve expensive computations like matrix exponentiation, matrix diagonalization or root-finding, or are biased, involving some form of time discretization. Additionally, many of these methods do not extend easily to complicated likelihood functions, which require specialized sampling algorithms. For instance, the contribution of Fearnhead and Sherlock (2006) is to develop an exact sampler for Markov modulated Poisson processes (MMPPs), where an MJP modulates the rate of a Poisson process. In next section, we describe a novel sampler that addresses these limitations of existing methods.

3.5 MCMC inference via Uniformization

Consider the problem of sampling an MJP path $\mathbf{S}(t)$ over the interval $\mathcal{T} = [t_{start}, t_{end}]$, given a set of noisy observations of its state. In the simplest case, we observe the state of the process at the boundaries t_{start} and t_{end}. More generally, we are given the initial distribution over states π_0 as well as a set of O noisy observations $X = \{X_{t_1^o}, ...X_{t_O^o}\}$ at a discrete set of times $T^o = \{t_1^o, \ldots, t_O^o\}$. The observations have likelihoods $P(X_{t_i^o}|\mathbf{S}(t_i^o))$ and we wish to sample from the posterior $P(\mathbf{S}|X)$, or equivalently $P(S, T|X)$. In section 3.6 on Markov modulated Poisson processes, as well as in chapter 4 on continuous-time Bayesian networks, we consider observations that are effectively *continuous-time*. As we will show, our method handles these cases quite naturally as well.

To understand our approach, first recall that the MJP is the continuous-time limit of a discrete-time Markov chain. Sampling a trajectory of a finite state discrete-time Markov chain given noisy observations can be easily and efficiently done using the forward filtering backward sampling dynamic programming algorithm. During the forward pass, this algorithm iteratively calculates the marginal distribution over states at step i given the observations until this time. At the end of the forward pass, we have the

marginal distribution over states at the end time given all observations, and we sample a value from this distribution. We then make a backward pass through the chain, successively sampling the state at step i given the associated marginal distribution calculated during the forward pass, and the sampled value at step $(i+1)$. The Markov dependencies allow all calculations to be performed very efficiently, and at the end we have a sample of the Markov chain from the posterior distribution.

We can try to adapt this approach to the MJP by discretizing time, and running the forward-backward algorithm on this system. This will return a trajectory that can change state only at a finite set of times, chosen *a priori*. Since an MJP can change state at *any* time, such an approach is clearly biased. To keep this bias small, we will need to discretize time at a fine resolution. This now requires us to run the forward-backward algorithm on a long Markov chain, and can be inefficient.

Instead of trying to reduce the bias by choosing a fine time-discretization, our approach will be to eliminate it altogether by choosing a *random* discretization. Uniformization provides us with exactly such a randomized discretization.

Recall that uniformization proceeds by sampling a set of Poisson events W and then assigning them state-labels V via a discrete-time Markov chain. We will define $W = \left(w_0, w_1, \cdots, w_{|W|}\right)$ to include the endpoints of the interval (i.e. $w_0 = t_{start}$ and $w_{|W|} = t_{end}$). W forms a random discretization of the interval $[t_{start}, t_{end}]$, and under the MJP, we assign W labels V by running a Markov chain with transition matrix B from equation (3.12). Resampling the labels V now involves running the forward-backward algorithm with transition matrix B. Each transition from w_i to w_{i+1} must incorporate evidence from all observations in the interval $(w_i, w_{i+1}]$. At the end, we obtain a new state sequence $\tilde{V}$, and the pair $(\tilde{V}, W)$ maps to a new MJP path $(\tilde{S}, \tilde{T})$ (see the bottom two panels in figure 3.3).

The question now remains: how do we obtain the set W? In subsection 3.5.1, we review some methods that try to sample W from $P(W|X)$, its posterior distribution given the observations (Hobolth and Stone, 2009; Fearnhead and Sherlock, 2006). In general, this is not straightforward, depending critically on the likelihood model. Instead, we will show that sampling from $P(W|\mathbf{S}(\cdot), X)$, *conditioned* on the current MJP trajectory, is easy. In fact, we will see that $P(W|\mathbf{S}(\cdot), X) = P(W|\mathbf{S}(\cdot))$, so that unlike methods that involve sampling $P(W|X)$, our sampler applies to a wide range of observation processes (the observations enter only via the likelihood terms in the forward-backward algorithm). At a high level, our sampler is an auxiliary variable Gibbs sampler that proceeds by alternately sampling the Poisson events W given the MJP trajectory, and then a new trajectory given W.

Look at figure 3.3 and remember that the pairs (S, T) and (V, W) map to the same MJP trajectory. The only difference is the existence of a set of virtual jumps in (V, W). Call this U; recovering the uniformized representation (V, W) given (S, T) involves sampling

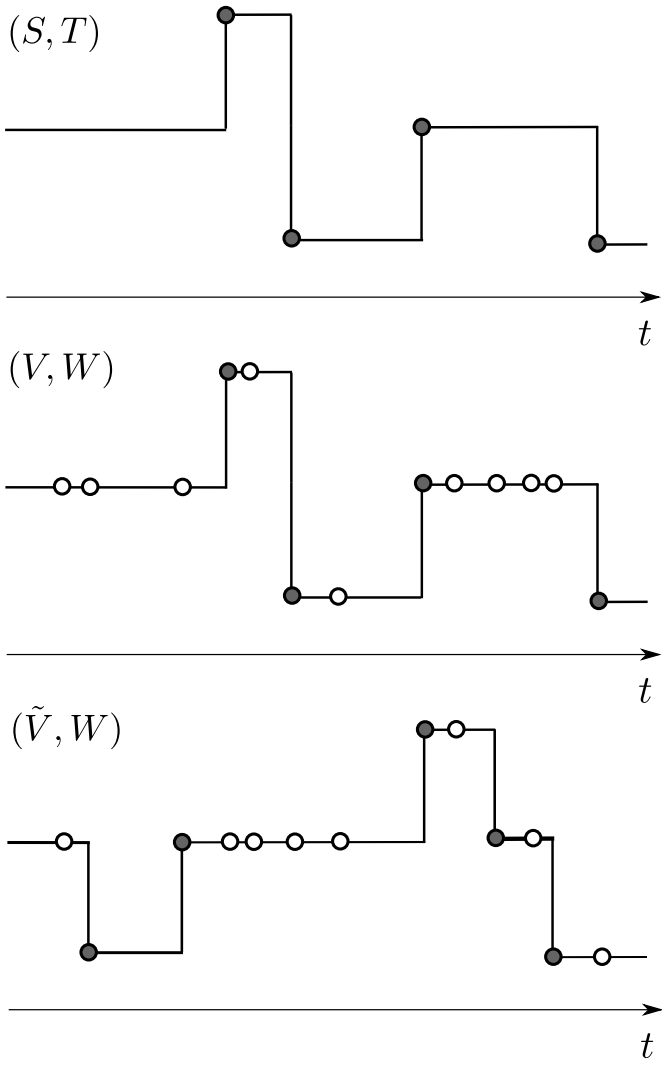

Figure 3.3: Uniformization-based Gibbs sampler: starting with an MJP trajectory, resample the thinned events and then resample the trajectory giving all Poisson events

U given (S, T). Now, W is drawn from a rate Ω Poisson process, while the probability that an element of W also belongs to U (i.e. the probability of it being thinned) when the MJP is in state s is $(1 - \frac{|A_s|}{\Omega})$. The last expression is the probability that the embedded Markov chain makes a self-transition. Corollary 2.1 of the thinning theorem then suggests that we can reconstruct such thinned events by sampling from a Poisson process with rate $(\Omega - |A_s|)$. Of course, the state of the MJP and thus the probability of thinning varies with time; so consider sampling U from an *inhomogeneous* Poisson process with intensity $(\Omega - |A_{\mathbf{S}(t)}|)$. This intensity is piecewise-constant, taking the value $(\Omega - |A_{s_i}|)$ on the interval $[t_i, t_{i+1})$. Define $|U_i|$ as the number of auxiliary times in this interval, so that $|U| = \sum_{i=0}^{|T|-1} |U_i|$. The probability density of U is then

$$P(U|S,T) = \left(\prod_{i=0}^{|T|-1} (\Omega - |A_{s_i}|)^{|U_i|} \right) \exp\left(- \int_{t_{start}}^{t_{end}} (\Omega - |A_{\mathbf{S}(t)}|) \mathrm{d}t \right) \tag{3.25}$$

Proposition 3.1. *For any $\Omega \geq \max_s(|A_s|)$, the Markov jump process (S,T) with auxiliary times U sampled from equation (3.25) is equivalent to the times W sampled from the subordinating Poisson process along with the states V assigned via the subordinated Markov chain. In other words, $P(S,T,U) = P(V,W)$.*

Proof. Multiplying equation (3.17) with equation (3.25), we see that

$$P(S,T,U) = \frac{\Omega^{|U|+|T|-2}}{e^{\Omega(t_{end}-t_{start})}} \cdot \pi_0(s_0) \prod_{i=0}^{|T|-1} \left(1 - \frac{|A_{s_i}|}{\Omega}\right)^{|U_i|} \prod_{i=1}^{|T|-1} \frac{A_{s_i s_{i-1}}}{\Omega} \tag{3.26}$$

The first term on the right is the probability of an ordered set of times $T \cup U$ under a homogeneous Poisson process with rate Ω (recall that the endpoints of T are fixed). The second term is the probability of a sequence of states under a Markov chain with initial distribution π_0 and transition matrix $B = (I + \frac{1}{\Omega}A)$. These are just W and V. □

Having resampled U (and thus W), we assign W a new set of labels $\tilde{V}$ by running the forward-backward algorithm as described earlier. To incorporate the likelihoods of observations X into this process, let $X_{[w_i,w_{i+1})}$ represent the observations in the interval $[w_i, w_{i+1})$. Throughout this interval, the MJP is in state v_i, giving a likelihood term:

$$L_i(v_i) = p\left(X_{[w_i,w_{i+1})} | \mathbf{S}(t) = v_i\right), \quad t \in [w_i, w_{i+1}) \tag{3.27}$$

For the case of noisy observations of the MJP state at a discrete set of times T^o, this simplifies to

$$L_i(v_i) = \prod_{j:t_j^o \in [w_i,w_{i+1})} p(X_{t_j^o} | \mathbf{S}(t_j^o) = v_i) \tag{3.28}$$

Conditioned on the times W, V is a Markov chain with likelihoods given by equation (3.27), so we can efficiently resample V using the standard forward filtering-backward sampling algorithm. The cost of this is $O(N^2|V|)$, quadratic in the number of states and linear in the length of the chain. Further any structure in A (e.g. sparsity) is inherited by B and can be exploited easily.

Let $\tilde{V}$ be the new state sequence. Then $(\tilde{V}, W)$ will correspond to a new MJP path $\tilde{\mathbf{S}}(t) \equiv (\tilde{S}, \tilde{T})$, obtained by discarding virtual jumps from $(\tilde{V}, W)$.

Proposition 3.2. *The auxillary variable Gibbs sampler described above has the posterior distribution $p(\mathbf{S}(t)|X)$ as its stationary distribution. Moreover, if $\Omega > \max_i(|A_i|)$, the resulting Markov chain is ergodic.*

Proof. The first statement follows from the fact that the algorithm simply introduces auxiliary variables U followed by conditionally sampling V given X and W. To show ergodicity, note that if $\Omega > \max_s(|A_s|)$, then the intensity of the subordinating Poisson process is strictly positive. Consequently, there is positive probability density of sampling appropriate auxiliary jump times U and moving from any MJP path to any other. □

Algorithm 3.2 Block Gibbs sampler for a Markov jump process on the interval $[t_{start}, t_{end}]$

Input: A set of observations X and parameters A (the rate matrix), π_0 (the initial distribution over states) and $\Omega > \max_s(|A_s|)$.
The previous MJP path, $\mathbf{S}(t) \equiv (S, T)$, where T is the set of transition times (including end times), and S is the set of corresponding state values.

Output: A new MJP trajectory $\tilde{\mathbf{S}}(t) \equiv (\tilde{S}, \tilde{T})$.

1: Sample $U \subset [t_{start}, t_{end}]$ from a Poisson process with piecewise-constant rate $(\Omega - |A_{\mathbf{S}(t)}|)$. Define $W = T \cup U$.
2: Sample a path from a discrete-time Markov chain with $|W|$ steps using the forward-backward algorithm. The transition matrix of the Markov chain is $B = (I + \frac{1}{\Omega}A)$, while the initial distribution over states is π_0. The likelihood of state s at step i is $L_i(s) = P\left(X_{[w_i, w_{i+1})} | \mathbf{S}(t) = s\right)$, $t \in [w_i, w_{i+1})$ with $X_{[w_i, w_{i+1})}$ representing the observations in the interval $[w_i, w_{i+1})$.
3: Let $\tilde{T}$ be the set of times in W when the Markov chain changes state (as well as the end times). Define $\tilde{S}$ as the corresponding set of state values.

Note that it is essential for $\Omega > \max_s(|A_s|)$; equality is not sufficient for ergodicity. For example, if all diagonal elements of A are equal to $-\Omega$, then the subordinating Poisson process will have intensity 0, and consequently the set of jump times T will never be changed by the sampler above. In fact, the only dependence between successive samples of the Gibbs sampler is through the shared jump times T, since the state sequence $\tilde{V}$ is independent of V given W. By increasing Ω, more auxiliary virtual jumps are introduced, increasing the probability of different jump times, leading to faster mixing. Of course, as a consequence, the HMM chain grows longer, leading to a linear increase in the computational cost per Gibbs iteration. Thus the parameter Ω allows a trade-off between mixing rate and computational cost. We look at the effect of this parameter in subsection 3.5.3; in all other experiments, we set $\Omega = \max_s(2|A_s|)$. We find this works quite well, with the samplers typically converging after less than 5 iterations.

3.5.1 Comparison with existing sampling algorithms

A simple Monte Carlo approach to obtaining posterior samples from an endpoint-conditioned MJP (i.e. an MJP with noiseless observations at the endpoints of an observation interval) is rejection sampling: sample paths from the prior given the observed

start-state and reject those that do not end in the observed end-state (Nielsen, 2002). For multiple noiseless observations over an interval, one uses the Markov property of the MJP to break the problem into a number of independent endpoint conditioned inference problems.

Rejection sampling can be extended to the case of noisy observations by importance sampling or, more practically, by sequential Monte Carlo methods like particle filtering (Fan and Shelton, 2008). Recently, Golightly and Wilkinson (2011) have applied particle MCMC methods to correct the bias introduced by standard particle filtering methods. However, these methods are efficient only in situations where the data exerts a relatively weak influence on the trajectory (compared to the prior): a large state-space or an unlikely end state can result in large numbers of rejections or small effective sample sizes. Though these algorithms are simple and general purpose, their flexibility means they do not fully exploit the structure of the MJP, and often require complicated modifications to make proposals that 'hit the data'.

A second approach, more specific to the MJP, uses matrix exponentiation (equation (3.10)) to integrate out the infinitely many paths leading from the state at the time of one observation to state at the next. In particular, let t_i be the time of the ith observation, and $P(\mathbf{S}(t_i)|X_{[0,t_i]})$ be a vector of the probability over states at time t_i, given all observations upto (and including) the ith observation. Then,

$$P(\mathbf{S}(t_{i+1}) = s|X_{[0,t_{i+1}]}) \propto P(X_{t_{i+1}}|s) \left[\exp\left(A(t_{i+1} - t_i)\right)^\intercal P(\mathbf{S}(t_i)|X_{[0,t_i]})\right]_s \tag{3.29}$$

This suggests a dynamic programming algorithm to sample the MJP state at a finite set of times $\tilde{T} \equiv (\tilde{t}_1, \cdots \tilde{t}_m)$: make a forward pass through this set, successively calculating the marginal distribution over states using equation (3.29) (starting with the initial distribution over states, π_0). Having calculated the distribution at the end time $\tilde{t}_m$, sample the MJP state at this time. Now, make a backward pass through the times, conditionally sampling a new state at $\tilde{t}_i$ given the state at time $\tilde{t}_{i+1}$. For more details, see (Hobolth and Stone, 2009) and the references therein.

This method has the advantage of exploiting the properties of the MJP to make 'optimal' proposals, which unlike with the methods of the previous paragraph are always accepted. One might view this difference as similar to that between running the standard forward-filtering backward-sampling algorithm and say, a particle filter on the discrete-time Markov chain. However, this analogy breaks down computationally, since matrix exponentiation is an expensive operation that scales as $O(N^3)$, N being the number of states. Thus, this method does not scale well when the dimensionality of the MJP state space is large, and in particular, this does not extend to MJPs with infinite state spaces. Also, the matrix resulting from matrix exponentiation is dense and any structure, e.g. sparsity, in the rate matrix A cannot be exploited. Note also that the set of times $\tilde{T}$ must include the set of observation times, and we therefore need at least as many matrix-exponentiations as there are observations. As we will see in

the section on Markov modulated Poisson processes, there are many situations where the frequency of observations is much higher than the frequency of state changes in the MJP, and ideally, we would like the number of expensive matrix exponentials to scale with the latter quantity. Our MCMC sampler does not require any expensive matrix exponentiations; moreover, the length of the discrete-time Markov chain scales with the number of Poisson events (and thus, on the uniformization rate Ω). This is a property of the dynamics of the MJP, rather than, say, the observation process. We elaborate on this point in section 3.6.

Another limitation of the previous scheme is that we recover the MJP state only at a finite set of times. Having marginalized out the states at all remaining times, we need an additional step to fill in the rest of the trajectory. Sampling the entire trajectory is important in situations where one is performing inference on the MJP parameters (subsection 3.5.2), here one needs statistics like the total time spent in each state and the number of transitions between each pair of MJP states. One option to fill in the MJP trajectory is to use rejection sampling. A more popular approach is to use uniformization as outlined in Hobolth and Stone (2009). Like our sampler, these methods proceed by sampling the Poisson events W in the interval between observations, and then running a discrete-time Markov chain on this set of times to sample a new trajectory. However, sampling from the posterior distribution over the number of Poisson events can be tricky (depending crucially on the observation process), and usually requires a random number of $O(N^3)$ matrix multiplications (as the sampler iterates over the possible number of Poisson events). Our sampler is also based on unformization, but unlike existing work which produce *independent* samples, ours is an *MCMC* algorithm. By sampling the Poisson process *conditioned* on the current trajectory, the details of the observation process become irrelevant. The latter only enter when running the HMM forward-backward algorithm. In this sense, our sampler is a convenient general purpose sampler for MJP-based models, with the user only having to provide a function that calculates the probability of observations in any segment of time where the MJP remains in a fixed state. At the price of producing correlated samples, our method extends naturally to various extensions of MJPs, scales as $O(n^2)$, does not require matrix exponentiation, and easily exploits structure in the rate matrix. Moreover, we demonstrate that our sampler mixes very rapidly.

3.5.2 Bayesian inference on the MJP parameters

Having described an MCMC algorithm to sample a new MJP trajectory given an old trajectory and a set of parameters A and π_0, we can perform a fully Bayesian analysis by placing priors on the MJP parameters as well. We can then embed our MCMC sampler within an outer Gibbs sampler that alternately resamples the trajectory given the parameters, and then the parameters given the trajectory. Working in this framework further amplifies the benefits of our method. As we shall see, the computational cost

other samplers incur to produce independent samples of the MJP trajectory is now even more wasteful, since the mixing of the overall Gibbs sampler is fairly insensitive to whether the conditional updates are independent or correlated.

Like Fearnhead and Sherlock (2006), we place independent gamma priors on the diagonal elements $|A_s|$ (i.e. on the leaving rate of each state s) and independent Dirichlet priors on the probabilities of transitioning from each state to all the other states. In particular, defining $p_{s,s'} = \frac{A_{s,s'}}{|A_s|}$, we let:

$$|A_s| \sim Gamma(\alpha_1, \alpha_2) \tag{3.30}$$

$$(p_{s,1}, \ldots, p_{s,s-1}, p_{s,s+1}, \ldots p_{s,N}) \sim Dirichlet(\beta) \tag{3.31}$$

This prior over the rate matrix A is conjugate; the sufficient statistics required to calculate the posterior (given a trajectory $\mathbf{S}(t)$) are the total number of state transitions, the total amount of time spent in each state and the number of transitions between each pair of states. Thus, let $n_{s,s'}$ be the number of transitions from state s to s', and n_s be the number of times the MJP leaves state s (so that $n_s = \sum_{s' \in \mathcal{S}} n_{s,s'}$). Let T_s be the total amount of time spent in state s. Then,

$$|A_s| \,|\, (S,T) \sim Gamma(\hat{\alpha}_{1,s},\ \hat{\alpha}_{2,s}) \tag{3.32}$$

$$(p_{s,1}, \ldots, p_{s,s-1}, p_{s,s+1}, \ldots p_{s,N}) | (S,T) \sim Dirichlet\left(\beta + (n_{s,1}, \ldots, n_{s,s-1}, n_{s,s+1}, \ldots n_{s,N})\right) \tag{3.33}$$

Here $\hat{\alpha}_{1,s} = \alpha_1 + n_s$ and $1/\hat{\alpha}_{2,s} = 1/\alpha_2 + 1/T_s$.

We either fix π_0, the initial distribution over states, to the discrete uniform distribution or set it equal to the equilibrium distribution of the rate matrix A. In the latter case, the distribution described previously becomes a Metropolis-Hastings proposal distribution, and we accept an A^{new} sampled from this distribution with probability proportional to the probability of initial state under the equilibrium distribution of A^{new} divided by that of the initial state under the equilibrium distribution of A^{old}. Note that computing the equilibrium distribution requires solving an $O(N^3)$ eigenvector problem, so that in this case, the overall Gibbs sampler is cubic (even though our MCMC sampler scales as $O(N^2)$).

3.5.3 Experiments

In this set of experiments, we look at the effect of the subordinating Poisson rate Ω on the mixing of our MCMC sampler. We generated a random 5-by-5 matrix A (with hyperparameters $\alpha_1 = \alpha_2 = \beta = 1$, see equations (3.30) and (3.31)). The state of this MJP trajectory was observed via a Poisson likelihood model (see section 3.6), and

samples produced by a C++ implementation of our algorithm were used to characterize the resulting posterior. Each run consisted of 10000 iterations with a burn-in of 1000 samples. The left plot in figure 3.4 shows effective sample sizes (ESS) against computation times for different values of the ratio $(\Omega / \max_s(|A_s|))$. For each MCMC sample, we calculated the the number of transitions as well as the time the MJP trajectory spent in each state, and for all MCMC runs, the effective sizes of these statistics were calculated using R-CODA (Plummer et al., 2006). The 'overall' ESS of an MCMC run was the median ESS across all these statistics. Figure 3.4 (left) shows this median averaged across a 1000 runs, keeping A fixed. We see that increasing Ω does increase the mixing rate, however the added computational cost quickly swamps out any benefit this might afford. Figure 3.4 (right) is a similar plot for the case of Bayesian inference on the MJP parameters: here rather than keeping the parameters fixed, these were resampled as described in subsection 3.5.2. Now, the effective sample size of an MCMC run was the median ESS of all MJP parameters; the figure shows this number averaged across a 1000 runs. Interestingly, for this problem, ESSs are fairly insensitive to Ω, suggesting an dependent Gibbs update is as effective as a conditionally independent Gibbs update. We found this to be true in general; when embedded within an outer Gibbs sampler, our sampler (with $\Omega = 2 \max_s(|A_s|)$) produced similar effective parameter sizes as an MJP sampler that produces independent samples. In any case, we shall see that the computational savings provided by our sampler far outweigh the cost of dependent samples.

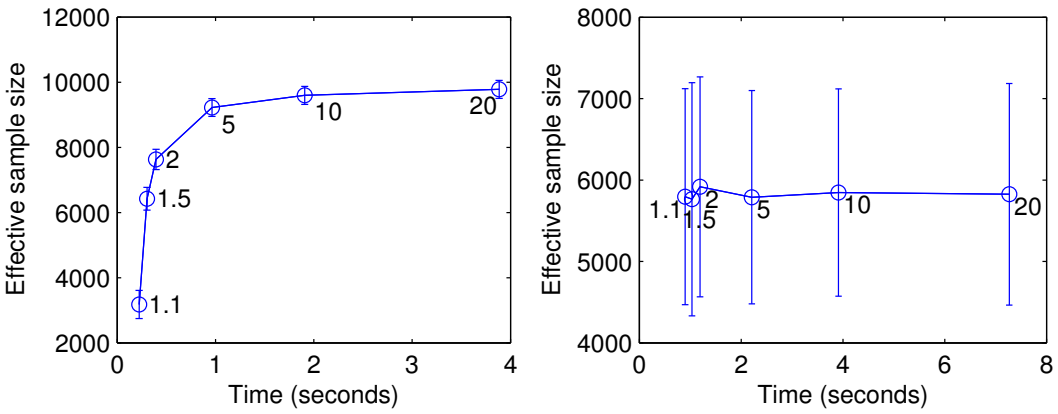

Figure 3.4: Effective sample sizes vs computation times for different settings of Ω for (left) a fixed rate matrix A and (right) Bayesian inference on the rate matrix

In light of these results, for all subsequent experiments, we set $\Omega = 2 \max_s(|A_s|)$. Figure 3.5 shows the initial burn-in of a sampler with this setting for different initializations (the vertical axis shows the number of state transitions in the current MJP sample). This quantity quickly reaches its equilibrium value within a few samples.

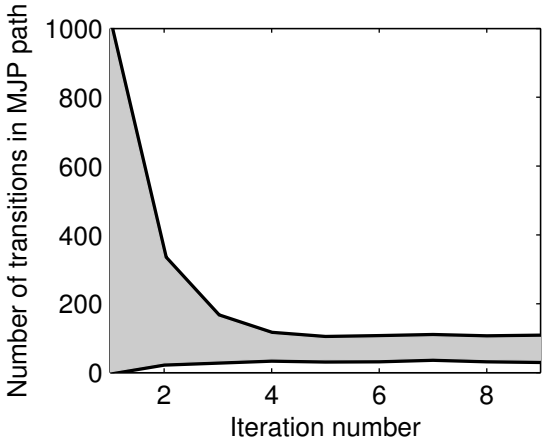

Figure 3.5: Traceplot of the number of MJP jumps for different initializations

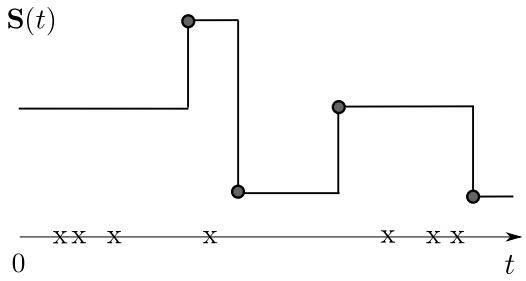

Figure 3.6: A realization of a Markov modulated Poisson process. Note that that Poisson events 'x' are unrelated to the uniformization-based construction, and need not be aligned with eg. the MJP jumps

3.6 Markov modulated Poisson processes (MMPPs)

A Markov modulated Poisson processes (figure 3.6) is an doubly-stochastic Poisson process (also called a Cox process (Cox, 1955)) whose intensity function is piecewise-constant and distributed according to a Markov jump process. Let the MJP have N states, and supposed it is parametrized by an initial distribution over states π_0 and a rate matrix A. Associate with each state s a nonnegative constant λ_s, call this the output or emission rate of state s. Then a set of points O is distributed as an MMPP when

$$\mathbf{S}(t) \sim \text{MJP}(\pi_0, \text{A}) \tag{3.34}$$

$$O \sim Poiss(\lambda_{\mathbf{S}(t)}) \tag{3.35}$$

Note that the Poisson process O is different from the subordinating Poisson process used in the uniformization-based construction of the MJP, and we shall refer to it as the output Poisson process. MMPPs have been used to model phenomenon like the distribution of rare DNA motifs along a gene (Fearnhead and Sherlock (2006)), photon arrival in single molecule fluorescence experiments (Burzykowski et al., 2003), web page

requests (Scott and Smyth, 2003) etc.

The Poisson observations effectively form continuous-time observations of the latent MJP, with the *absence* of Poisson events also providing information about the MJP state. For example, the absence of observed Poisson events over any long interval would suggest that it is unlikely for the latent MJP to have spent that interval in a state with a high emission rate.

Fearnhead and Sherlock (2006) developed an exact sampler for MMPPs, exploiting the fact that over an interval where the MJP remains in a fixed state, the probability of no Poisson events is exponential in the length of the interval (equation (2.48)). Observing that the MJP waiting times are also exponentially distributed, they define an extended MJP where the rate of exiting any state is the sum of rate of leaving that state and the sum of the emission rate corresponding to that state. Upon exiting the state, the extended system can either move to another MJP state or (on emitting a output Poisson event), move to an absorbing state. Given a distribution over states at any time, this allowed them to calculate the distribution over states at any subsequent time: for times after the next Poisson event, the system will be in the absorbing state; for times before the next Poisson event, the distribution over states can by calculated by matrix-exponentiating the extended rate matrix (equation (3.10)). Thus, they start with an initial distribution over states, and sequentially calculate the distribution over states just before a Poisson observation, given the distribution over states just before the previous observation. They follow this forward-filtering stage with a backward-sampling stage where they instantiate the state of the MJP at all Poisson events (as well as at the start and end times). Having sampled the state of the MJP at this set of discrete times, they finally use a uniformization-based endpoint conditioned MJP sampler to fill in the MJP trajectory between every pair of adjacent times.

The main advantage of this method is that it produces independent samples from the MMPP posterior. However, it does so at the price of being fairly complicated and computationally intensive. Moreover, it has the disadvantage of operating at the timescale of the Poisson observations rather than the dynamics of the latent MJP: for large Poisson rates, this can be quite inefficient as we shall demonstrate.

Our MCMC sampler outlined in the previous section can be straightforwardly extended to the MMPP without any of these disadvantages. Resampling the subordinating Poisson events (step 1 in algorithm 3.2) remains unaffected, since conditioned on the current MJP trajectory, their distribution is independent of the observations. Step 2 requires calculating the emission likelihoods $L_i(s)$; over any interval $[w_i, w_{i+1})$, this is given by

$$L_i(s) = (\lambda_s)^{|O_i|} \exp\left(-\lambda_s(w_{i+1} - w_i)\right), \tag{3.36}$$

Here, $|O_i|$ is the number of output Poisson events in the interval $[w_i, w_{i+1})$ and λ_s is

the Poisson rate of state s. We place conjugate Gamma priors on the λ_s's. Note that evaluating equation (3.36) requires counting the number of observed Poisson events between every successive pair of subordinating Poisson process events. The flexibility of our approach can be seen by comparing this modification of our original algorithm with the complicated approach Fearnhead and Sherlock (2006) had to take.

3.6.1 Experiments

In the following, we compare a C++ implementation of our algorithm with an implementation* of the algorithm of Fearnhead and Sherlock (2006), coded in C. We performed fully Bayesian inference, sampling both the MJP parameters (as described in subsection 3.5.2) and the Poisson likelihood rates λ_s. In all instances, our algorithm did significantly better, the performance improvement increasing with the complexity of the problem.

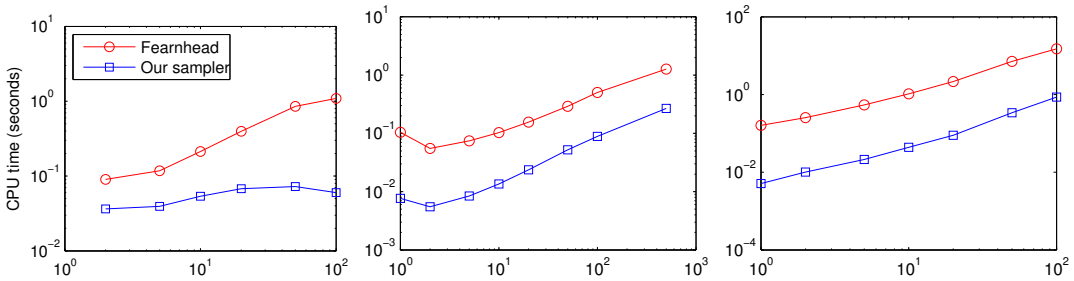

Figure 3.7: CPU time to produce a 100 effective samples as we observe (left) increasing number of Poisson events in an interval of length 10 (centre), 10 Poisson events over increasing time intervals and (right) increasing intervals with the number of events increasing on average.

Figure 3.7 shows the first set of experiments, where the number of states of the latent MJP was fixed to 5. Like in the previous experiment, the prior on the rate matrix A had parameters $\alpha_1 = \alpha_2 = \beta = 1$. The shape parameter of the Gamma prior on the output Poisson rate of state s, λ_s was set to s (thereby breaking symmetry across states), the scale parameter was fixed at 1. We used the same hyperpriors over the MMPP parameters for all runs. In all cases, we estimated the time required to produce 100 effective samples from a run of 10000 samples (with a burn-in of 1000). For each MCMC iteration, we alternately sampled the MJP trajectory and the MMPP parameters, and at the end of the MCMC run, the effective sample size (ESS) of each parameter was estimated. The 'overall' effective sample size was the median ESS of all parameters; this and the overall simulation time was used to estimate the time required to produce 100 effective samples. Each point in the figures is this time, averaged over 10 random datasets.

In the leftmost plot, we plot this time as an increasing number of Poisson events were observed, randomly distributed on an interval of fixed length 10. For our sampler,

*Downloaded from Chris Sherlock's webpage

we see that increasing the number of observations leaves the computation time largely unaffected, while for the sampler of Fearnhead and Sherlock (2006), this increases quite significantly. This reiterates the point that our sampler works at the timescale of the latent MJP, while Fearnhead and Sherlock (2006) work at the timescale of the observed Poisson process.

In the middle plot, we fix the number of observations to 10, increasing the length of the observation interval instead, while in the final plot, we increase both the interval length and the average number of observations in that interval. In both these cases, once again our sampler offers improvements of more than an order of magnitude. In fact, the only problems where we observed the sampler of Fearnhead and Sherlock (2006) to outperform ours were low-dimensional problems with only a few Poisson observations in a long interval, and with a rate matrix with a single, very unstable state. The latter condition results in a high uniformization rate Ω but only a few state transitions (since the system typically spends most of the time in the stable states). The resulting large number of virtual jumps can make our sampler inefficient. We return to this problem in chapter 7.

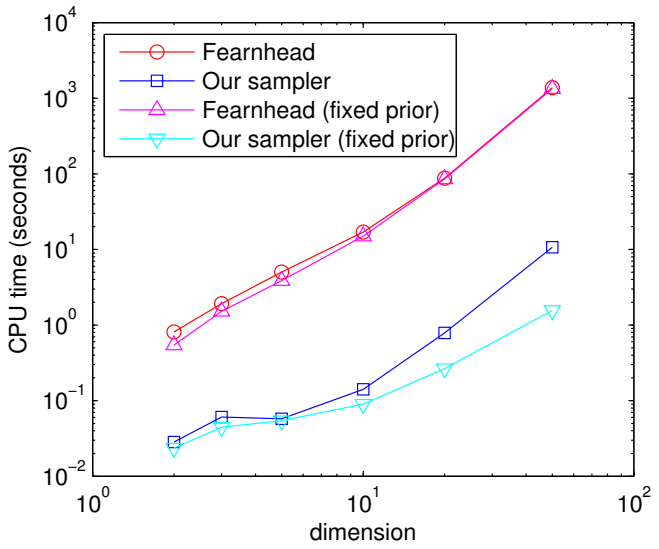

Figure 3.8: CPU time required to produce 100 effective samples as the state space of the MJP is increased

In figure 3.8, we plot the time to produce 100 effective samples as the size of the MJP state space increases. In this case, we fixed the number of Poisson observations to 10, over an interval of length 10. We see that our sampler (plotted with squares) offers substantial speed-up over the sampler of Fearnhead and Sherlock (2006) (plotted with circles); however we see that for both samplers, computation time scales cubically with the latent dimension. However, recall that this cubic scaling is not a property of our MJP-path sampler; rather it is a consequence of using the equilibrium distribution of a sampled rate matrix as the initial distribution over states (this requires calculating an eigenvector of a proposed rate matrix). If we fix the initial distribution over states to

the discrete uniform distribution, we observe that our sampler now scales quadratically (the cyan curve with downward triangles).

3.7 Discussion

In this chapter, we proposed a novel Markov chain Monte Carlo sampling method for Markov jump processes. Our method exploits the simplification of the structure of the MJP resulting from the introduction of auxiliary variables via uniformization. This constructs a Markov jump process by subordinating a discrete-time Markov chain to a Poisson process. Our sampler is a blocked Gibbs sampler in this augmented space and proceeds by alternately resampling the Poisson process given the Markov chain and vice versa. This auxiliary variable Gibbs sampler is computationally very efficient as it does not require time discretization, matrix exponentiation or diagonalization, and can exploit structure in the rate matrix. Importantly, our sampler easily generalizes to MJP-based models like Markov modulated Poisson processes. In our experiments, we studied our sampler empirically, demonstrating a significant speed-up compared to a state-of-the-art sampler for MMPPs.

Our sampler will form the basis of the more complicated algorithms studied in later chapters. In all cases, we will first provide an alternate approach to sampling from a system of interest; this will involve thinning a sample from a system with higher event rates. The alternate system as well as the thinning procedure will usually be more complicated than a Poisson process and a discrete-time Markov chain. However, inference will still proceed as we did here: reconstruct the thinned events using properties of the Poisson process and then resample the trajectory using sampling ideas from discrete-time systems.

Our method opens a number of avenues that we do not explore in this thesis. One concerns the subordinating Poisson rate Ω which acts as a free-parameter of the sampler. While our heuristic of setting this to $2 \max_s |A_s|$ worked well in our experiments, this may not be the case for rate matrices with states of widely varying stability. One approach is to 'learn' a good setting of this parameter via adaptive MCMC methods (Andrieu and Thoms, 2008). More fundamentally, it would be interesting to investigate if theoretical claims can be made about the 'best' setting of this parameter under some measures of mixing speed and computational cost. In chapter 7, we describe an alternate approach, where rather that using a single rate Ω, we associate different rates with different states.

Next, there are a number of immediate generalizations of our sampler. First, our algorithm is easily applicable to inhomogeneous Markov jump processes where techniques based on matrix exponentiation cannot be applied. Chapter 5 will reveal how this can be done in a straightforward manner. In chapter 6 we will also look at generalizing our

sampler to semi-Markov processes where the holding times of the states follow non-exponential distributions; these models find applications in fields like biostatistics and queuing theory (Mode and Pickens, 1988), neuroscience (McFarland et al., 2011) etc.

By combining our technique with slice sampling ideas (Neal, 2003a), we can explore Markov jump processes with a countably infinite state spaces. In our case, the complication with these models is not so much the infinite state spaces involved; rather it is the fact that the maximum event rate in these models can be infinite. This means that we cannot directly choose a Poisson rate Ω that dominates all event rates in the system; so that a straightfoward application of our ideas impossible. We shall see such problems in the next two chapters, though we will leave it until chapter 6 before we fully resolve this issue.

Chapter 4

Continuous-time Bayesian networks (CTBNs)

4.1 Introduction

Continuous-time Bayesian networks (CTBNs) are compact, multi-component representations of Markov jump processes that have structured rate matrices (Nodelman et al., 2002). Special instances of these models have long existed in the literature, particularly stochastic kinetic models like the Lotka-Volterra equations (these describe interacting populations of animal species, chemical reactants, gene regulation networks, etc (Wilkinson, 2009)). There have also been many related developments (see for example Bolch et al. (1998); Didelez (2008)). We shall however deal with CTBNs, a framework introduced in Nodelman et al. (2002) that harnesses the representational power of Bayesian networks to characterize structured MJPs.

Just as the familiar Bayesian network uses a collection of smaller conditional probability tables to represent a probability table whose size is exponential in the number of variables, so too a CTBN represents a structured rate matrix with smaller conditional rate matrices. An m-component CTBN represents the state of an MJP with the states of m nodes $\mathbf{S}^1(t), \ldots, \mathbf{S}^m(t)$ in a directed (and possibly cyclic) graph $\mathcal{G}$. Figure 4.1 shows two CTBNs, the 'predator-prey network' and the 'drug-effect network'. The former is a CTBN governed by the Lotka-Volterra equations; it describes the dynamics of the population sizes of two interacting species, a 'predator' and a 'prey'. We shall look at it more closely in subsection 4.3.1. The latter is a popular CTBN used to model the dependencies in events leading to and following a patient taking a drug.

Loosely speaking, each node of the CTBN acts as an MJP with a particular rate matrix that depends on the instantaneous configuration of its parents (and not its children, although the presence of cycles means a child can be a parent as well). The trajectories of all nodes are piecewise constant, and when a node changes state, the event rates of

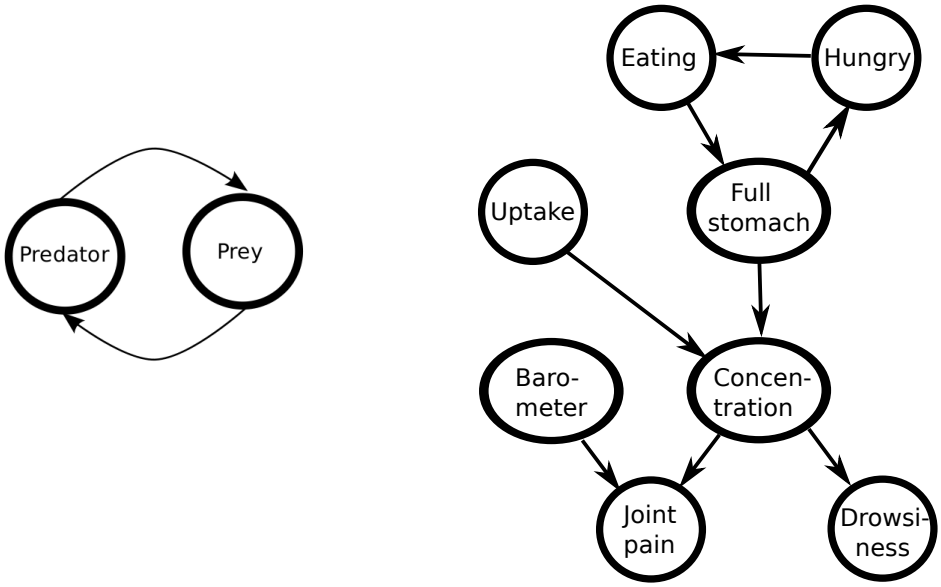

Figure 4.1: The predator-prey network (left) and the drug-effect CTBN (right)

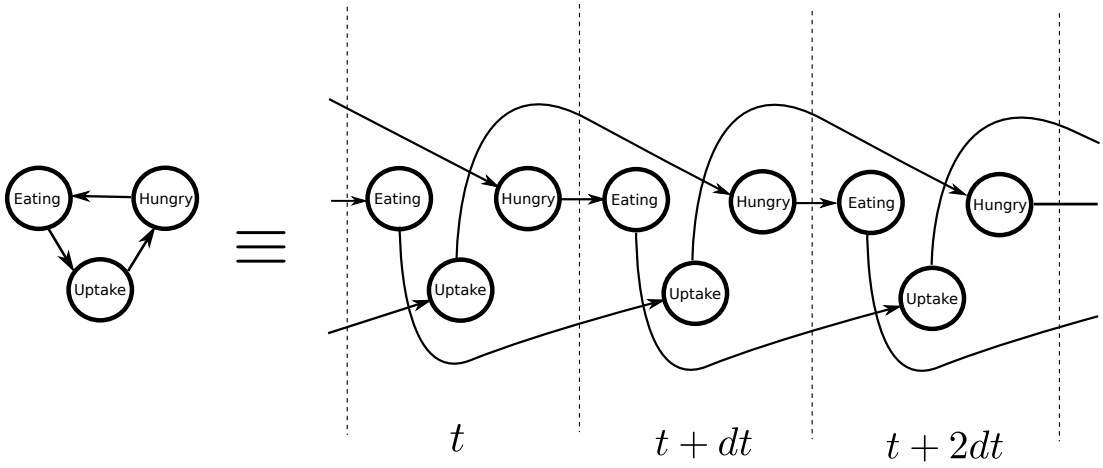

Figure 4.2: Expanded CTBN

all its children change. The graph $\mathcal{G}$ and the set of rate matrices (one for each node and for each configuration of its parents) characterize the dynamics of the CTBN. The former describes the structure of the dependencies between the various components of the CTBN, and the later quantifies these dependencies. Completing the specification of the CTBN is an initial distribution π_0 over the nodes, often specified via some directed acyclic Bayesian network $\mathcal{B}$.

It is convenient to think of a CTBN as a compact representation of an expanded (and now acyclic) graph, consisting of the nodes of $\mathcal{G}$ repeated infinitely along a continuum (viz. time). In this graph, arrows lead from a node at a time t to instances of its children at time $t + \mathrm{d}t$. Figure 4.2 displays this for a section of the drug-effect CTBN. The rates associated with a particular node at time $t + \mathrm{d}t$ are determined by the configuration of its parents at time t. Figure 4.2 is the continuous-time limit of a class of discrete-time models called dynamic Bayesian networks or DBNs (Murphy, 2002). In a DBN, the

state of a node at stage $i+1$ is sampled from a probability distribution determined by the configuration of its parents at stage i. Just as MJPs are continuous-time limits of discrete-time Markov chains, CTBNs are also continuous-time limits of DBNs.

If we collapse the nodes at each stage of a DBN into a single node, we recover a discrete-time Markov chain whose transition matrix is easily computed from the various transition matrices of the nodes of the DBN. In exactly the same sense, it is possible to combine all local rate matrices of a CTBN into one global rate matrix; see Nodelman et al. (2002) for a description of this operation which they call *amalgamation*. In this sense, a CTBN is just an MJP whose state-space is the product state-space of all component nodes and whose rate matrix lends itself to local decomposition.

The fact that we can write down an overall rate matrix on the joint state-space of the CTBN makes it possible, conceptually at least, to sample a trajectory over an interval $[t_{start}, t_{end}]$ using Gillespie's algorithm. However, with an eye towards inference, we would like to do this by exploiting the structure in the graph $\mathcal{G}$. If $\mathcal{G}$ had no loops (i.e. it were a chain or a tree), we could view the CTBN as a hierarchy of Markov jump processes modulated by Markov jump processes. We could then sample an overall CTBN trajectory by sampling trajectories of the root nodes, and then moving down the graph sampling trajectories of those nodes whose parents have instantiated trajectories. For the last step, we run Gillespie's algorithm on each interval over which a node's parents remain unchanged, using the rate matrix dictated by the parents' states, and using the terminating state in the previous interval as the initial state for the current interval.

To sample from a CTBN with a general graph $\mathcal{G}$, we return to its representation in figure 4.2. As mentioned earlier, the configuration of the CTBN nodes at time t determines the rates of all possible state transitions, and these rates remain unchanged until the next event (when the CTBN configuration changes). Recall from section 2.6 that a constant event rate λ implies that the waiting time till the next event is exponentially distributed with parameter λ. Thus, given the CTBN configuration at time t, we read off the rates of all possible events (each event corresponding to a node of the CTBN leaving its current state for some new state), sample all waiting times, and choose the first to occur. We move the clock forward by this time and update the configuration of the CTBN accordingly. Then, exploiting the memoryless property of the exponential distribution, we once again resample future waiting times of all events (discarding their previous values), and repeat this procedure until we exit the interval. Of course, as with Gillespie's algorithm, it is possible to exploit properties of the exponential to directly sample the time of the earliest event (this has a rate equal to the sum of all event rates) and then sample the event identity (now each event has a probability proportional to its rate). In other words, we proceed by repeatedly sampling the time of the next state change and then the identity of this state transition.

Algorithm 4.1 summarizes the generative process for the CTBN. Like chapter 3, we will

represent the trajectory of the CTBN, $\mathbf{S}(t)$, with the pair of sequences (S, T). Assume that the CTBN has m nodes. Then s_i, the ith element of S, is an m-component vector representing the states of all nodes at t_i, the time of the ith jump. We write this as $s_i = (s_i^1, \cdots, s_i^m)$. The rate matrix of an node n will vary over time as the configuration of its parents changes, and we will write $A^{n,t}$ for the relevant matrix at time t.

Algorithm 4.1 Algorithm to sample a CTBN trajectory on the interval $[t_{start}, t_{end}]$

Input: The CTBN graph $\mathcal{G}$, a set of rate matrices $\{A\}$ for all nodes and for all parent configurations and an initial distribution over states π_0.

Output: A CTBN trajectory $\mathbf{S}(t) \equiv (S, T)$.

1: Assign the CTBN a configuration $s_0 \equiv (s_0^1, s_0^2, ...) \sim \pi_0$. Set $t_0 = t_{start}$ and $i = 0$.
2: **while** $t_i < t_{end}$ **do**
3: For each node k, draw $z^k \sim \exp(|A^{k,t_i}_{s_i^k}|)$. Increment i.
4: Let $K = \text{argmin}_k\, z^k$ be the first node to jump.
5: Let $t_i = t_{i-1} + z^K$ be the next jump time.
6: Suppose $s_{i-1}^K = s'$. Set $s_i^K = s$ with $P(s_i^K = s | s_{i-1}^K = s') \propto A^{K,t_i}_{s's}$ $\forall s \neq s'$.
7: Set $s_i^k = s_{i-1}^k$ $\forall k \neq K$.
8: **end while**
9: Set $t_i = t_{end}$, and $s_i = s_{i-1}$.

From algorithm 4.1, and following subsection 3.3.1, we can write down the probability density of (S, T) as

$$P(S, T) = \pi_0(s_0) \prod_{i=1}^{|T|-1} P(s_i, t_i | s_{i-1}, t_{i-1}), \quad \text{where} \tag{4.1}$$

$$P(s_i, t_i | s_{i-1}, t_{i-1}) = \prod_{n=1}^{m} \left(\exp\left(-|A^{n,t_{i-1}}_{s_{i-1}^n}|(t_i - t_{i-1}) \right) \left(A^{n,t_{i-1}}_{s_{i-1}^n s_i^n} \right)^{1(s_i^n \neq s_{i-1}^n)} \right) \tag{4.2}$$

We recall that $A^{n,t_{i-1}}$ is the rate matrix of node n determined by the configuration of its parents at time t_{i-1}. The first part of the term in the product in equation (4.2) is the probability of node remaining unchanged from t_{i-1} to t_i, while the second term is the probability of node n changing from s_{i-1}^n to s_i^n (if it does).

4.2 Inference in CTBNs

Have described the prior distribution over trajectories that a CTBN encodes, we now consider the problem of posterior inference over trajectories given (possibly noisy) observations at a discrete set of times*. Even though a CTBN can be interpreted as a simple MJP over an expanded state space, this state space is exponentially large in the number of nodes, so that sampling algorithms (even our algorithm from section 3.2)

*Extensions to more complicated observations like a Poisson process modulated by the CTBN trajectory are easily handled.

cannot be applied directly. To develop a tractable MCMC sampler that exploits the structure represented by the graph $\mathcal{G}$, we instead consider a Gibbs sampler. This will proceed by iteratively resampling the trajectory of each node, conditioned on the trajectories of the other nodes in the CTBN.

Write the parents and children of a node n as $\mathcal{P}(n)$ and $\mathcal{C}(n)$ respectively. Let $\mathcal{MB}(n)$ be the Markov blanket of node n, so that

$$\mathcal{MB}(n) = \mathcal{P}(n) \cup \mathcal{C}(n) \cup \{\mathcal{P}(c) \ \forall c \in \mathcal{C}(n)\} \tag{4.3}$$

Given the entire trajectories of all nodes in $\mathcal{MB}(n)$, node n is independent of all other nodes in the network (Nodelman et al., 2002). Note however that we have to condition on the *entire* trajectory of the Markov blanket, otherwise the temporal dynamics of the network cause all nodes in the graph to become entangled (Nodelman et al., 2002). That is, the present state of some node outside the Markov blanket tells us something about that node's previous states, which in turn tells us something about previous configurations of the Markov blanket $\mathcal{MB}(n)$, thus resulting in dependence with the current state of n.

The Markov property of the CTBN suggests a Gibbs sampling scheme where the trajectory of each node is resampled given that of its Markov blanket. This was the approach followed by El-Hay et al. (2008). However, even without any observations, sampling a node trajectory conditioned on the complete trajectory of its Markov blanket is not straightforward. To see this, plug equation (4.2) into equation (4.1), and interchange the order of multiplication; we get

$$P(S,T) = \pi_0(s_0) \prod_{n=1}^{m} \prod_{i=1}^{|T|} \left(\exp\left(-|A^{n,t_{i-1}}_{s^n_{i-1}}|(t_i - t_{i-1})\right) \left(A^{n,t_{i-1}}_{s^n_{i-1}s^n_i}\right)^{1(s^n_i \neq s^n_{i-1})} \right) \tag{4.4}$$

$$= \pi_0(s_0) \prod_{n=1}^{m} \phi(S^n, T^n | S^{\mathcal{P}(n)}, T^{\mathcal{P}(n)}) \tag{4.5}$$

In equation (4.5), we represent the trajectory of a set of nodes $\mathcal{N}$ as $(S^{\mathcal{N}}, T^{\mathcal{N}})$. In particular we write the trajectory of node n as (S^n, T^n), and of its parents as $(S^{\mathcal{P}(n)}, T^{\mathcal{P}(n)})$. Note that $(S,T) \equiv (S^{\mathcal{G}}, T^{\mathcal{G}})$. Also,

$$\phi(S^n, T^n | S^{\mathcal{P}(n)}, T^{\mathcal{P}(n)}) = \prod_{i=1}^{|T|} \exp\left(-|A^{n,t_{i-1}}_{s^n_{i-1}}|(t_i - t_{i-1})\right) \left(A^{n,t_{i-1}}_{s^n_{i-1}s^n_i}\right)^{1(s^n_i \neq s^n_{i-1})} \tag{4.6}$$

Now, if $A^{n,t_{i-1}}$ is constant, the factor $\phi(\cdot)$ in equation (4.6) is just the density of an MJP with initial state s_0 (see subsection 3.3.1). Since $A^{n,t_{i-1}}$ varies in a piecewise-constant manner, $\phi(\cdot)$ is actually the density of a piecewise-inhomogeneous MJP (or of a sequence of MJPs). In any event, sampling such a trajectory is straightforward. The complication however is that the trajectory of node n also affects the densities of its

children $\mathcal{C}(n)$. In particular, the conditional distribution over (S^n, T^n) has the form

$$P(S^n, T^n) \propto \pi_0(s_0)\phi(S^n, T^n|S^{\mathcal{P}(n)}, T^{\mathcal{P}(n)}) \prod_{c\in\mathcal{C}(n)} \phi(S^c, T^c|S^{\mathcal{P}(c)}, T^{\mathcal{P}(c)}) \tag{4.7}$$

Thus, even over an interval of time where the parent configuration remains constant, the conditional distribution of the path is not a homogeneous MJP because of the effect of the node's children; these act as 'observations' that are continuously observed. Effectively, we have a 'Markov jump process-modulated Markov jump process', and we need to sample the latent MJP having observed the modulated child MJPs. Observe that this problem is a generalization of the inference problem for the Markov modulated Poisson process (section 3.6). As we mentioned earlier, an additional (though minor) complication is the piecewise-constant inhomogeneity introduced by transitions of parent nodes of n. Additionally, the parameters governing the likelihood of the children also vary in a piecewise-constant manner, due to changes in the state of the childrens other parents. Finally, we also need to account for actual observations of the state of the CTBN node.

El-Hay et al. (2008) described a matrix-exponentiation-based Gibbs sampler that repeatedly samples the time of the next transition of node n and assigns the node a new state. At a high-level, each Gibbs step of their sampler is similar to that of Fearnhead and Sherlock (2006), with the Poisson observations of the MMPP generalized to transitions in the trajectories of child nodes. Consequently, it involves an expensive forward-backward algorithm involving matrix exponentials. In addition El-Hay et al. (2008) resort to discretizing time: to obtain the time of the next transition of the node, they perform a binary search on the time interval up to a specified accuracy (they argue that this approximation allows the user to specify a desired 'precision'). We next show how our uniformization-based sampler from chapter 3 can easily be adapted to produce samples efficiently without having to resort to any approximations: all this essentially requires is a new likelihood function $L_i(s)$ that depends on the number of state transitions the children make as well as how much time they spend in each state for each parent configuration. In our experiments, we show that besides being exact, our sampler produces significant computational gains.

4.2.1 Auxiliary Variable Gibbs sampling for CTBNs

In this section, we describe a Gibbs sampling algorithm to simulate the CTBN posterior over an interval $[t_{start}, t_{end}]$, given a set of observations X at times $\{t_1^o, ...t_O^o\}$. An iteration of the overall algorithm proceeds by performing Gibbs updates on all nodes in the CTBN; in the following we describe the update step for a single node n. Thus, we are given the complete sample paths of all nodes in node n's Markov blanket $\mathcal{MB}(n)$ and a starting distribution π_0 over states at time t_{start}. Importantly, our algorithm

produces a dependent Gibbs update, so that we also need the old trajectory of node n. To avoid notational clutter, we suppress all references to the node index n. Thus, call the old trajectory of node n, $\mathbf{S}(t) \equiv (S, T)$, and the new trajectory $\tilde{\mathbf{S}}(t) \equiv (\tilde{S}, \tilde{T})$. Recall also that over the time interval $[t_{start}, t_{end}]$, the parents of node n can change state; consequently the rate matrix governing the dynamics of node n changes in a piecewise constant manner. We do not indicate the dependence of rate matrices on the configuration of the parents, and instead just call the relevant rate matrix at time t, A^t.

The Gibbs update for node n begins as depicted in subplot a of figure 4.3, with the current trajectory of node n and that of its Markov blanket. Like chapter 3, we first reconstruct the thinned Poisson events, and then update the trajectory. In principle, we could imagine (S, T) had a uniformized construction from a subordinating Poisson process with rate Ω, so that we resample the thinned events from an piecewise-inhomogeneous Poisson process with rate $(\Omega - |A^t_{\mathbf{S}(t)}|)$. However, such an Ω would have to dominate the event rates corresponding to *all* configurations of the parents of node n. Abusing notation, let A^p be the rate matrix when $\mathcal{P}(n)$ takes on configuration p (it will always be clear from the context whether the superscript refers to time or parent configuration). Then we need

$$\Omega \geq |A^p_s| \qquad \forall p, s \tag{4.8}$$

This can be inefficient, particularly in large CTBNs with a few unstable states. In such a situation, the subordinating Poisson rate Ω can be determined by a possibly atypical configuration p of $\mathcal{P}(n)$ that leads to instability in node n (and thus large values of $|A^p_s|$ for some s). This can leads to a very large number of thinned events, and a consequent inefficiency in the forward-backward algorithm.

Instead, since the rate matrix A^t varies in a piecewise-constant manner, we might consider subordinating it to a piecewise inhomogeneous Poisson process. For a rate matrix A^p, define a corresponding Poisson rate $\Omega^p \geq \max_s(|A^p_s|)$, and (abusing notation again) define Ω^t as the Poisson rate at time t. We then resample the thinned events, now from a Poisson process with rate $\left(\Omega^t - |A^t_{\mathbf{S}(t)}|\right)$. Now, the Poisson rate at any time time t is dictated by the relevant configuration of the Markov blanket of the node. Like chapter 3, the posterior Poisson intensity of the thinned events is still piecewise constant, changing only when either $\mathbf{S}(t)$ changes state (the times in T) or when one of the parents changes state (we call this set of times P).

The correctness of such an approach is obvious for a piecewise-inhomogeneous MJP; we can just view this as a sequence of MJPs with different parameters. Our situation is a bit more subtle (though still straightforward). In particular, the rate matrix A^t at any time t is *not* fixed, but varies from Gibbs iteration to iteration as the configuration of $\mathcal{MB}(n)$ changes. One way to see why our scheme is still valid is by viewing the overall Gibbs update from $\mathbf{S}(t)$ to $\tilde{\mathbf{S}}(t)$ as a transition operator parametrized by the

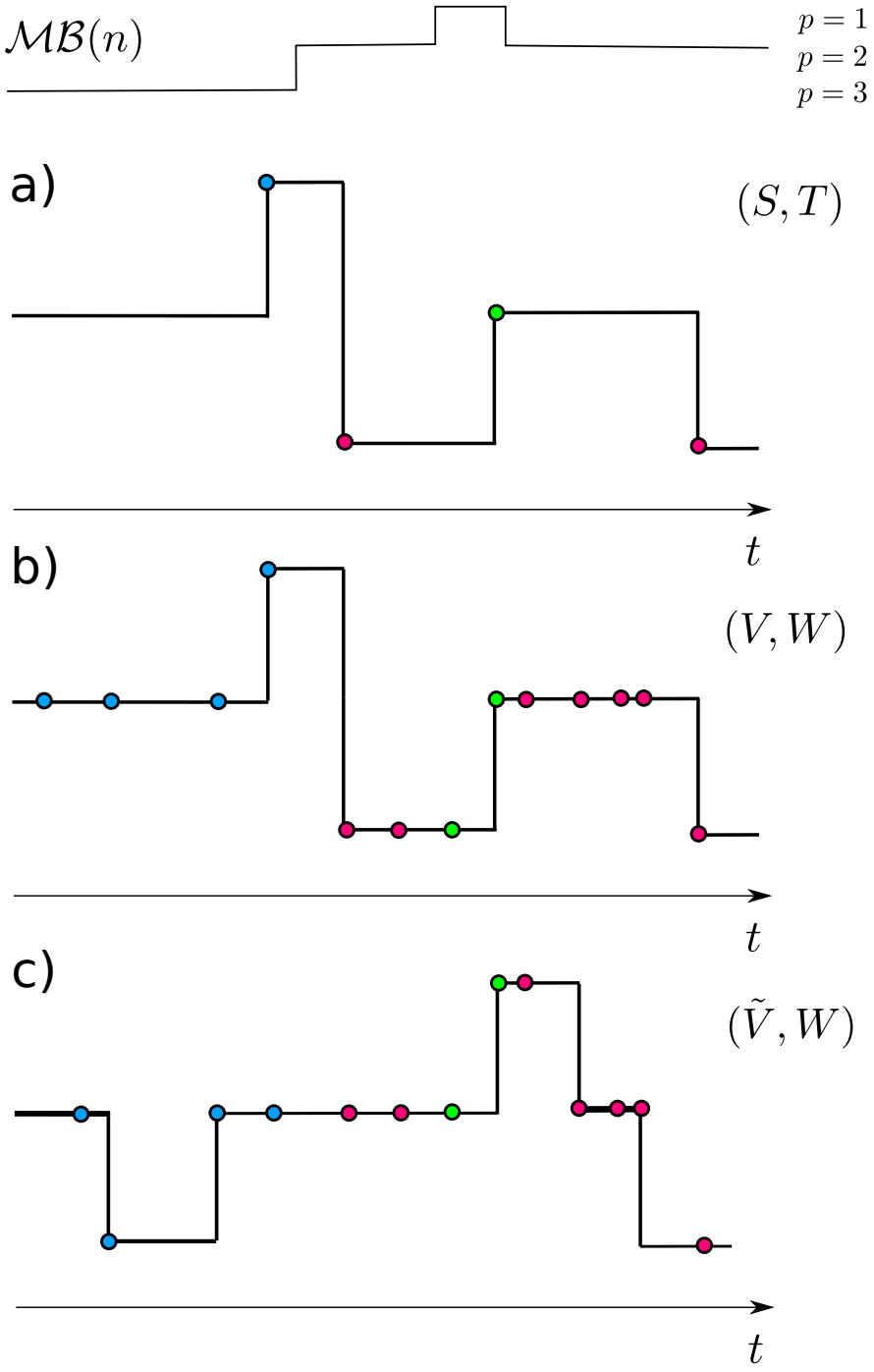

Figure 4.3: Gibbs update for a node of a CTBN. The colours refer to the associated Markov blanket configuration.

Poisson rate Ω. We saw in section 3.5 that any operator with $\Omega > \max_s |A_s^t|$ has the correct stationary distribution. Now, under our scheme, we choose a particular Ω (and therefore a particular transition operator) depending on the configuration of the node's Markov blanket. This is valid.

Figure 4.3(b) shows the result of resampling the thinned events U from the rate $(\Omega^t - |A^i_{\mathbf{S}(t)}|)$ Poisson process. We have coloured the Poisson events to correspond to the

associated Markov blanket configuration.

Figure 4.3(c) shows the final step in the Gibbs update, where we thin the set $W \equiv (T \cup U)$ by constructing a subordinated Markov chain on the set of times $\tilde{W} \equiv T \cup U \cup P$. For this step, we include P only to emphasize that the parameters of the Markov chain (its transition and emission matrices) change after events in P; it is important to realize that the MJP path for node n will not change state at the times in P (so that when $t \in P$ the transition matrix is simply $B^t = I$). At times $t \in T \cup U$, the transition matrix is

$$B^t = I + \frac{1}{\Omega^t} A^t \tag{4.9}$$

Since the Poisson rate Ω^t varies with time, the transition operator B^t must do so too.

Characterizing the emission matrix of the Markov chain is easy; observe that if node n had no children, we could proceed by resampling the states of the subordinated hidden Markov model using the likelihood function $L_i(s)$ in equation (3.28). To account for the presence of children $\mathcal{C}$, we must weigh the probability of a complete trajectory $\mathbf{S}(t)$ with the probabilities of the child trajectories under that path, see equation (4.7). Each child factor $\phi(S^c, T^c | S^{\mathcal{P}(c)}, T^{\mathcal{P}(c)})$ is the density an MJP, and from the Markov property, this factorizes as

$$\phi(S^c, T^c | S^{\mathcal{P}(c)}, T^{\mathcal{P}(c)}) = \prod_{i=0}^{|\tilde{W}|-1} \phi^i(S^c, T^c | S^{\mathcal{P}(c)}, T^{\mathcal{P}(c)}) \tag{4.10}$$

Here, $\phi^i(S^c, T^c | S^{\mathcal{P}(c)}, T^{\mathcal{P}(c)})$ is the density of a segment of the child trajectory over $(\tilde{w}_i, \tilde{w}_{i+1})$ for successive elements in $\tilde{W}$. As before, we define $\tilde{w}_0 = t_{start}$, and $\tilde{w}_{|\tilde{W}|} = t_{end}$. Evaluating ϕ^i under any configuration of s^n is now a simple matter of counting how much time the child node spent in each state, and well as the number of transitions between each pair of states, under each setting of the other parents of c.

The total likelihood function for the state of node n at step i of the hidden Markov model (i.e. over the interval $[w_i, w_{i+1})$) must include all children as well as the observations. This is just the product of the individual terms:

$$\tilde{L}_i(s) = L_i(s) \prod_{c \in \mathcal{C}} \phi^i(S^c, T^c | S^{\mathcal{P}(c)}, T^{\mathcal{P}(c)}) \tag{4.11}$$

Calculating equation (4.11) is straightforward as we make a forward pass through the event times. Given the transition probability (equation (4.9)) and the likelihood (equation (4.11)) of the Markov chain at step i, we use the forward filtering-backward sampling algorithm to obtain a trajectory of node n (subplot c in figure 4.3).

Since the new trajectory $\tilde{\mathbf{S}}(t)$ is obtained via introducing auxiliary variables and conditionally sampling a new path in the extended space, the MCMC sampler retains the

conditional distribution as its stationary distribution. Ergodicity of the conditional update, and thus the overall Gibbs sampler is straightforward to see, so that we have the result:

Proposition 4.1. *The auxillary variable Gibbs sampler described above converges to the posterior distribution over the CTBN sample paths.*

Note that unlike the Gibbs sampler of El-Hay et al. (2008) which produces independent samples from the conditional distribution, ours produces dependent Gibbs updates. With the trajectory updates part of an overall Gibbs cycle, we find that a conditionally independent sample has a negligible benefit towards mixing, and is significantly wasteful, once the computational cost is factored in.

4.3 Experiments

In the following, we evaluate a C++ implementation of our algorithm on a number of CTBNs. As in chapter 3, for a rate-matrix A^p, the parameter Ω^p was set to $2\max_s(|A^p_s|)$, so that for any node, the rate of the subordinating Poisson process varies with the configuration of its parents.

4.3.1 The Lotka-Volterra process

We first apply our sampler to the Lotka-Volterra process (Wilkinson, 2009; Opper and Sanguinetti, 2007). Commonly referred to as the predator-prey model, this describes the evolution of two interacting populations of 'prey' and 'predator' species. The two species form the two nodes of a cyclic CTBN (figure 4.1), whose states x and y represent the sizes of the prey and predator populations. The process rates are given by

$$A(\{x,y\} \to \{x+1,y\}) = \alpha x \qquad A(\{x,y\} \to \{x-1,y\}) = \beta xy$$
$$A(\{x,y\} \to \{x,y+1\}) = \delta xy \qquad A(\{x,y\} \to \{x,y-1\}) = \gamma y$$

where the parameters were set as follows: $\alpha = 5\times10^{-4}, \beta = 1\times10^{-4}, \gamma = 5\times10^{-4}, \delta = 1\times10^{-4}$. All other rates are 0. This defines two infinite sets of infinite-dimensional conditional rate matrices. In its present form, our sampler cannot handle this infinite state-space; observe that for any of the two rate matrices, $\max(|A_s|) = \infty$, so that uniformization is impossible. We describe how to overcome this limitation in chapter 7. Here, like Opper and Sanguinetti (2007), we limit the maximum number of individuals of each species to 200, leaving us with 400 200-dimensional matrices. Note that these matrices are tridiagonal and very sparse; at any time the size of each population can change by at most one. Consequently, the complexity of our algorithm scales *linearly* with the number of states (we did not modify our code to exploit this structure,

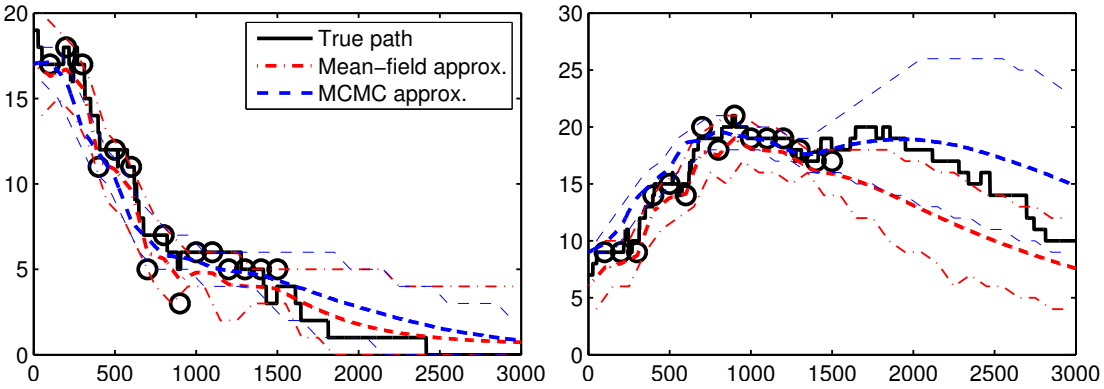

Figure 4.4: Posterior (mean and 90% confidence intervals) over (left) prey and (right) predator paths (observations (circles) only until 1500).

though this is fairly straightforward). A 'true' path of predator-prey population sizes was sampled from this process, and its state at time $t = 0$ was observed noiselessly. Additionally 15 noisy observations were generated, and spaced uniformly at intervals of 100 from $t = 100$ onwards. The noise process was:

$$p(X(t)|\mathbf{S}(t)) \propto \frac{1}{2^{|X(t)-\mathbf{S}(t)|} + 10^{-6}} \tag{4.12}$$

Given these observations (as well as the true parameter values), we approximated the posterior distribution over paths by two methods: using 1000 samples from our uniformization-based MCMC sampler (with a burn-in period of 100) and using the mean-field (MF) approximation of Opper and Sanguinetti (2007)[†]. Figure 4.4 shows the true paths (in black), the observations (as circles) as well as the posterior means and 90% confidence intervals produced by the two algorithms for the prey (left) and predator (right) populations. As can be seen, both algorithms do well over the first half of the interval where data is present. In the second half, the MF algorithm appears to underestimate the predicted size of the predator population; on the other hand, the MCMC posterior reflects the truth better. In general, we found the MF algorithm to underestimate the posterior variance in the MJP trajectories, especially over regions with few observations.

4.3.2 Average relative error vs number samples

For the remaining experiments, we compared our sampler with the Gibbs sampler of El-Hay et al. (2008); for this comparison, we used the CTBN-RLE package of Shelton et al. (2010) (also implemented in C++). In all our experiments, as with the MMPP, we found our algorithm to be significantly faster, especially for large inference problems. To prevent details of the two implementations from clouding the picture and to reiterate the benefit afforded by avoiding complex computations, we also measured the amount

[†]We thank Guido Sanguinetti for providing us with his code

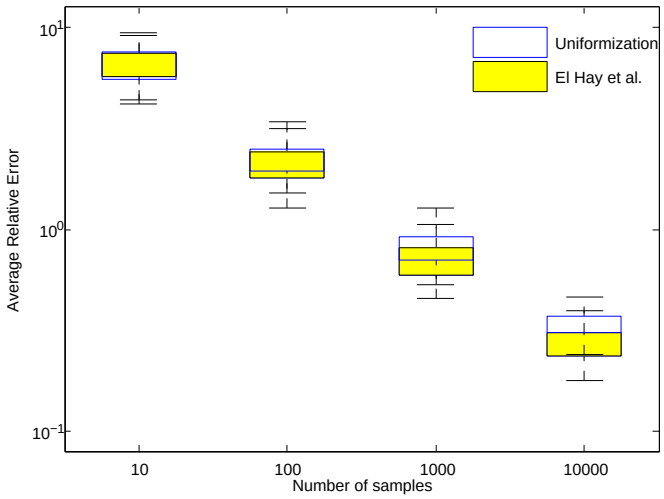

Figure 4.5: Average relative error vs number of samples for 1000 independent runs; burn-in = 200. Note that in this scenario Uniformization was about 12 times faster, so that for the same computational effort it produces significantly lower errors.

of time CTBN-RLE spent calculating matrix exponentials. This constituted between 10% to 70% of the total running time of the algorithm. In the plots we refer to this as 'El Hay et al. (Matrix Exp.)'. We found that our algorithm took less time than even this.

In our next experiment, we followed El-Hay et al. (2008) in studying how *average relative error* varies with the number of samples from the Markov chain. Average relative error is defined by $\sum_j \frac{|\hat{\theta_j}-\theta_j|}{\theta_j}$, and measures the total normalized difference between empirical $(\hat{\theta_j})$ and true (θ_j) averages of sufficient statistics of the posterior. The statistics in question are the time spent by each node in different states as well as the number of transitions from each state to the others. The exact values were calculated by numerical integration when possible, otherwise from a very long run of CTBN-RLE.

As in El-Hay et al. (2008), we consider a CTBN with the topology of a chain, consisting of 5-nodes, each with 5 states. The states of the nodes was observed at times 0 and 20 and we produced posterior samples of paths over the time interval $[0, 20]$. We calculate the average relative error as a function of the number of samples, with a burn-in of 200 samples. Figure 4.5 shows the results from running 1000 independent chains for both samplers. Not surprisingly, the sampler of El-Hay et al. (2008), which produces conditionally independent samples, has slightly lower errors. However the difference in relative errors is minor, and is negligible when considering the dramatic (sometimes up to two orders of magnitude; see below) speed improvements of our algorithm. For instance, to produce the 10000 samples, the El-Hay et al. (2008) sampler took about 6 minutes, while our sampler ran in about 30 seconds.

4.3.3 Time requirements

In the next three experiments, we compare the times required by CTBN-RLE and our uniformization-based sampler to produce 100 effective samples for CTBNs of different configurations. These times were estimated from runs of 10000 samples after a burn-in of a 1000 samples. Since CTBN-RLE does not support Bayesian inference for CTBN parameters, we kept these fixed and produced ESS estimates from the number of transitions of each node and the amount of time spent in each state (see subsection 3.6.1 for details). Each MCMC run produced samples from an endpoint-conditioned CTBN with random parameters and each point in the figures is an average over 10 simulations.

In the first of this set of experiments, we measured the times to produce these samples for the chain-shaped CTBN described above, as the number of nodes in the chain increases. The topmost plot in figure 4.6 shows the results. As might be expected, the time required by our algorithm grows linearly with the number of nodes. For El-Hay et al. (2008), the complete algorithm has a cost that grows faster than linear (quickly becoming unmanageable). The time spent calculating matrix exponentials *does* grow linearly, however our uniformization-based sampler always takes less time than even this.

Next, we kept the length of the chain fixed at 5, instead increasing the number of states per node. Once again, our sampler is always faster. Asymptotically, one would expect our sampler to scale as $O(n^2)$ and El-Hay et al. (2008) as $O(n^3)$, and while we haven't hit that regime yet, we can see that the cost of our sampler grows more slowly with the number of states.

Our final experiment (to the bottom) measures the time required as the length of the time interval over which the CTBN paths take values increases. For this experiment, we used the drug-effect network shown in figure 4.1: here the parameters were set to standard values (obtained from CTBN-RLE) and the state of the network was fully observed at the beginning and end times. Again, our algorithm is the faster of the two showing a linear increases in computational costs with the length of the interval. It is worth pointing out here that the algorithm of El-Hay et al. (2008) has a 'precision' parameter, and that by reducing the desired temporal precision, faster performance can be obtained. However, since our sampler produces *exact* samples (up to numerical precision), we feel our comparison is fair. In the experiments, we left this parameter at its default value.

4.4 Discussion

In this chapter, we extended our uniformization-based sampler from chapter 3 to CTBNs. In our experiments, we showed a significant performance improvement over a

state-of-the-art Gibbs sampler for CTBNs. More broadly, this chapter served to demonstrate the flexibility of our approach of alternately resampling thinned events given a trajectory, and then a new trajectory with the forward-backward algorithm. In later chapters, we shall extend this approach to more general continuous-time systems. A novelty of our approach in this chapter is that we allowed the uniformization parameter to vary with the configuration of the Markov blanket. We will later extend this idea by allowing the uniformization rate to depend on the current state of the node of interest. Among other things, this will allow us to avoid truncating the state-space of the MJP to bound maximum event rates. We shall revisit the Lotka-Volterra model in chapter 7 in the light of these ideas.

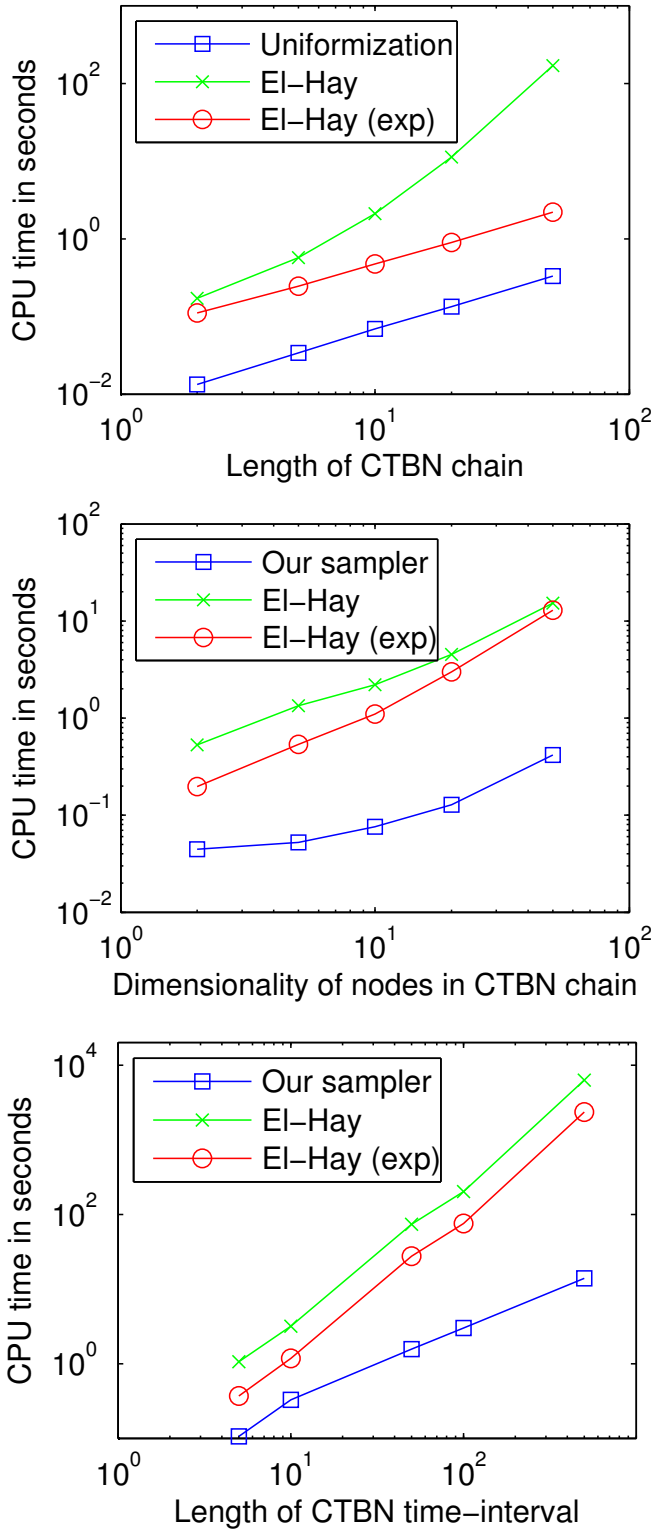

Figure 4.6: (top) CPU time vs length of CTBN chain. (middle) CPU time vs number of states of CTBN nodes. (bottom) CPU time vs time interval of CTBN paths.

Chapter 5

Modulated renewal processes

5.1 Introduction

In this chapter, we study generalizations of the Poisson process on the real line called renewal processes; these will allow us to relax the sometimes unrealistic independence (or memorylessness) assumptions of the Poisson process. We introduce a uniformization-based construction for these stochastic point processes, and demonstrate its utility by proposing a nonparametric Bayesian model where a renewal process is modulated by a Gaussian process intensity function. Without uniformization, drawing exact samples from this model (and thus posterior inference) is intractable. Our model extends work by Adams et al. (2009) on the Poisson process, using uniformization instead of Poisson thinning. In the style of previous chapters, we exploit properties of the Poisson process and develop a more natural and efficient blocked Gibbs sampler than the incremental Metropolis-Hastings algorithm used in Adams et al. (2009). In our experiments we demonstrate the usefulness of our model and sampler on a number of synthetic and real datasets.

5.2 Renewal processes

Renewal processes are stochastic point processes on the real line where waiting times between successive events are drawn i.i.d. from some distribution. The simplest example of a renewal process is the homogeneous Poisson process; as we saw in section 2.6, this has inter-event times that are exponentially distributed. While the independence/memoryless property of the Poisson process is convenient from the point of analysis and simulation, it is often not appropriate for modelling real-world phenomena. To borrow terminology from the reliability engineering literature, the independence property encodes an 'as bad as old after a repair' property (Lawless and Thiagarajah, 1996) that is often not realistic. Thus, suppose that events correpond to failures of

a component (which is then repaired instantaneously): one might expect the rate of failure immediately after a repair to be lower than before the breakdown. Similarly, in neuroscience, immediately after firing, a neuron is depleted of its resources and incapable of firing again, and the gamma distribution is used to model interspike intervals that have 'recovery times' (Cunningham et al., 2008; Kass and Ventura, 2001). Similarly, because of the phenomenon of elastic rebound, some time is required to recharge stresses released after an earthquake, and an inverse Gaussian distribution is used to model intervals between major earthquakes (Parsons, 2008). Other examples include using the Pareto distribution to better capture the burstiness and self-similarity of network traffic arrival times (Paxson and Floyd, 1995), and the Erlang distribution to model the fact that buying incidence of frequently purchased goods is less variable than Poisson (Wu, 2000).

In this chapter, we shall deal with finite renewal processes defined on some finite interval of the real line, $\mathcal{T} \equiv [t_{start}, t_{end}]$. A renewal process is characterized by a *renewal density* g, with waiting times between successive events drawn independently from g. Thus, one samples a realization of a renewal process by sampling an ordered sequence of times $(t_1, t_2, \cdots, t_n)$ with $(t_i - t_{i-1}) \sim g$ until one exits the interval. Here, we define $t_0 = t_{start}$; since $t_1 - t_{start} \sim g$, we are assuming that there is an event at t_{start}. For simplicity, we shall work with this assumption; often, the waiting time until the first event is taken to be exponentially distributed. It is easy to adapt our ideas to handle this case as well. As in earlier sections, we define $t_{n+1} = t_{end}$, and view the random sequence $T \equiv (t_0, t_1, \cdots, t_{n+1})$ as a point in the space $\mathcal{T}^{\cup}$ of finite random sequences in $\mathcal{T}$. It is clear that any element T of $\mathcal{T}^{\cup}$ has density w.r.t. the measure $\mu^{<}$ given by

$$p(T) = \begin{cases} \left(\prod_{i=1}^{|T|-1} g(t_i - t_{i-1})\right)\left(1 - G(t_{|T|} - t_{|T|-1})\right) & t_{i+1} > t_i \ \forall i \\ 0 & \text{otherwise} \end{cases} \tag{5.1}$$

Here, $|T| = n+2$ and $G(\cdot)$ is the cumulative distribution function of the renewal density $g(\cdot)$. The ith term in equation (5.1) is the probability density that the ith event occurs after a delay of $t_i - t_{i-1}$, while the last term is the probability that no event occurs in the interval $(t_n, t_{end}]$. When $g(\cdot)$ is the exponential distribution, this reduces to the Poisson density in equation (2.52).

5.2.1 Hazard functions

Associated with the renewal density $g(\cdot)$ is a *hazard function* $h(\cdot)$. The (stationary) hazard function $h(\tau)$ is defined as the event rate after τ time units have elapsed since the previous event, conditional on the waiting time being at least τ. Thus, for an infinitesimal $\Delta > 0$, $h(\tau)\Delta$ is the probability of the inter-event interval being in $[\tau, \tau+\Delta]$

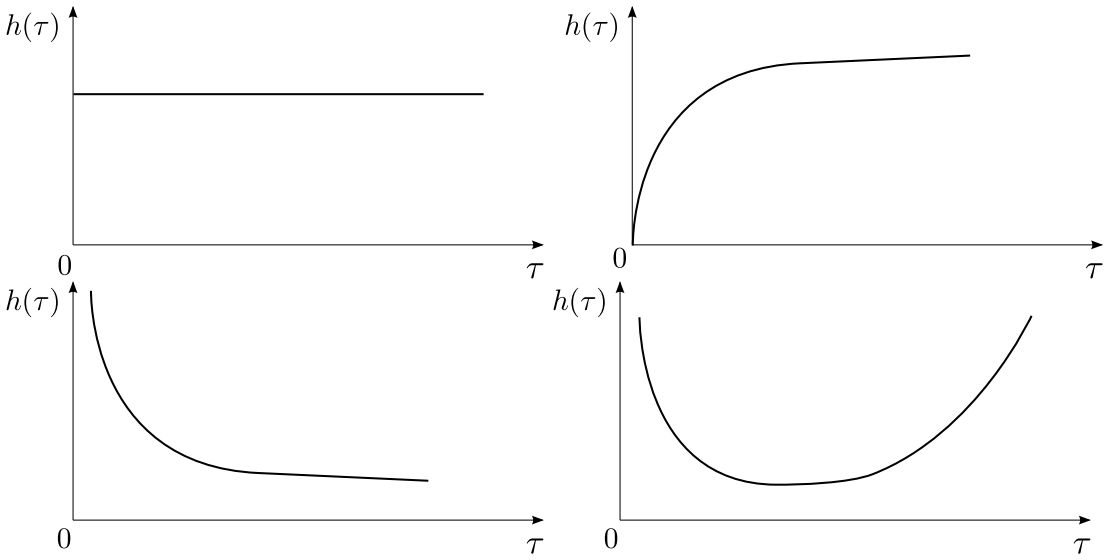

Figure 5.1: Left to right: hazard functions $h(\tau)$ for memoryless (Poisson), refractory and bursty renewal processes, and the bathtub hazard function. τ is the time since the last event.

conditioned on it being at least τ, so that:

$$h(\tau) = \frac{g(\tau)}{1 - \int_0^\tau g(u)\mathrm{d}u} \tag{5.2}$$

Given $h(\tau)$, one can also calculate the renewal density (we derive this in proposition 5.1 for more general case of a nonstationary hazard function), so that there is a one-to-one correspondence between $g(\tau)$ and $h(\tau)$.

The hazard function provides a clear illustration of the nature of the deviation from memorylessness of a particular renewal density g. The upper left plot in figure 5.1 shows the hazard function of a homogeneous Poisson process; this is a constant (equal to the Poisson process intensity), and is independent of the time since the last event. The top right plot shows the hazard function of a refractory renewal process; here the event rate drops immediately after an event, as the system 'recovers' from the last event. Such hazard functions are commonly produced by the gamma distribution or the Weibull distribution, both with shape parameter greater than 1. Also common are distributions like the inverse-Gaussian or the Levy distribution. To the bottom left, we have a hazard function used to model bursty and heavy-tailed activity. Hazard functions of this kind are produced by gamma or Weibull distribution, now with shape parameter less than 1. Both these distributions reduce to the exponential (and thus, the corresponding renewal processes to the Poisson) when this parameter is set to 1.

Of course, more complex hazard functions are possible and widely used. A common example is the 'bathtub curve' from reliability engineering (this is the bottom right plot in figure 5.1); unlike the other functions, this is not a monotone function of the time since the last event. For simplicity however, we shall restrict ourselves to the gamma

and the Weibull distributions in this thesis.

5.2.2 Modulated renewal processes

Modelling inter-event times as i.i.d. draws from a general renewal density can allow larger or smaller variances than an exponential with the same mean (called, respectively, underdispersion and overdispersion), but this now encodes an 'as good as new after a repair' property: after an event, the system is reset back to its 'initial state'. This is often only an approximation: because of age or other time-varying factors, the inter-event distribution of the point process can vary with time. For instance, internet traffic can vary with time of the day, day of the week and in response to advertising and seasonal trends. Similarly, an external stimulus can modulate the firing rate of a neuron, economic trends can modulate financial transactions etc. The most popular approach to modelling such nonstationarity is via the time-varying intensity of an inhomogeneous Poisson process. In this case, rather than being some constant (like the leftmost plot in figure 5.1), the instantaneous hazard function varies with time and we write it as $h(t)$*. As we saw in section 2.6, this means that the rate at which events occur now varies with time. Because renewal processes are not memoryless, in their case, the event rate will depend not just on the current time t, but also the time since the last event τ. We write this as $h(\tau, t)$. There has been work extending nonstationarity to renewal processes, and different approaches differ in how they couple the effects of the memory of the renewal process with the nonstationarity of the process. We review these in subsection 5.2.4, first however, we describe a simple approach that we will follow.

Let $\lambda(t)$ be some time-varying intensity function; this is the external signal that characterizes the nonstationarity of the renewal process. We let this signal modulate the instantaneous value of the hazard function $h(\tau)$ so that the event rate now depends on both the time τ since the last event, as well as on the absolute time t (Cox, 1972; Kass and Ventura, 2001). In other words, we define the inhomogeneous hazard function $h(\tau, t)$ as:

$$h(\tau, t) = m(h(\tau), \lambda(t)) \tag{5.3}$$

Here $m(\cdot, \cdot) : \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R}_+$ is some *interaction function* the governs how the rates $\lambda(\cdot)$ and $h(\cdot)$ are coupled together. Simple possibilities include include additive interactions $(h(\tau) + \lambda(t))$ or multiplicative interactions $(h(\tau)\lambda(t))$. For concreteness, we assume multiplicative interactions in what follows, however our results extend easily to general interaction functions. Figure 5.2 illustrates how such a function $\lambda(t)$ modulates the hazard rate (the grey curve) in the time following an event.

With a modulated hazard rate, the distribution of inter-event times is no longer stationary, and the deviation from $g(\cdot)$ is determined by how much λ deviates from

*In chapter 2 we called this $\lambda(t)$.

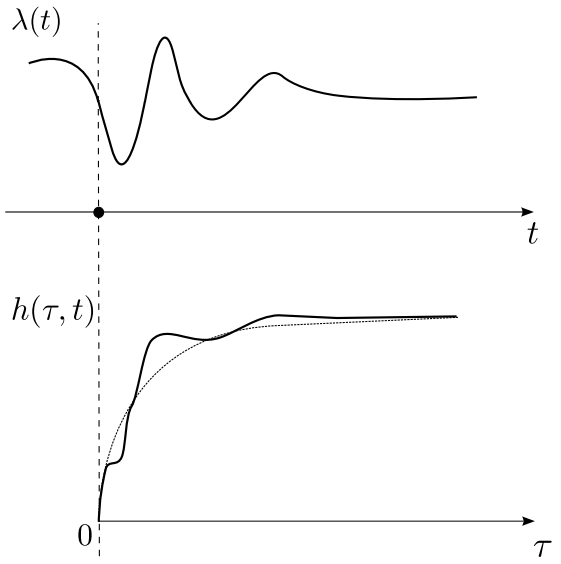

Figure 5.2: A modulated hazard function $h(\tau, t)$, produced by a multiplicative interaction between $h(\tau)$ (grey) and $\lambda(t)$.

some baseline. It is however possible to solve for the distribution of inter-event times, as we show in the next proposition:

Proposition 5.1. *For a renewal process with a nonstationary hazard function $h(\tau, t)$, the density of τ, the waiting time until the next event, given that the last event occured at time t_{prev} is given by*

$$g(\tau|t_{prev}) = h(\tau, t_{prev} + \tau) \exp\left(-\int_0^\tau h(u, t_{prev} + u)\mathrm{d}u\right) \tag{5.4}$$

Proof. By definition of the hazard function,

$$h(\tau, t_{prev} + \tau) = \frac{g(\tau|t_{prev})}{1 - \int_0^\tau g(u|t_{prev})\mathrm{d}u} \tag{5.5}$$

Let $y = 1 - \int_0^\tau g(u|t_{prev})\mathrm{d}u$. It follows that

$$h(\tau, t_{prev} + \tau) = \frac{-\mathrm{d}y/\mathrm{d}\tau}{y}, \text{ so that} \tag{5.6}$$

$$y = \exp\left(-\int_0^\tau h(u, t_{prev} + u)\mathrm{d}u\right) \tag{5.7}$$

Substituting back for y, we get

$$1 - \int_0^\tau g(u|t_{prev})\mathrm{d}u = \exp\left(-\int_0^\tau h(u, t_{prev} + u)\mathrm{d}u\right) \tag{5.8}$$

Differentiating w.r.t. τ, we get equation (5.4). □

Observe that with a constant hazard function and multiplicative modulation, equation (5.4) recovers the inter-event time for the inhomogeneous Poisson process.

5.2.3 Gaussian process intensity functions

Our goal in this chapter is to study a doubly stochastic renewal process with a random intensity function. Having described the mechanism by which a given nonstationarity modulates a renewal process, we now specify a prior on the intensity function $\lambda(t)$. Coupled with priors on the parameters of the hazard function, such a doubly stochastic model is useful in applications where one is interested in estimating both hazard parameters as well as the intensity function $\lambda(t)$. Rather than limiting $\lambda(t)$ to some parametric class of functions, we take a Bayesian nonparametric approach, modelling $\lambda(t)$ with a Gaussian process (GP) prior. Such a nonparametric prior has support over a rich class of functions and avoids the need for any 'hard' decisions about which function class we wish to limit ourselves to. We refer the reader to (Rasmussen and Williams, 2006) for details about Gaussian processes. For our purposes, it suffices to state that a GP prior is characterized by a mean function $\mu(\cdot)$ and a covariance kernel $K(\cdot,\cdot)$. The random function f evaluated on any finite set of points X is distributed as a Gaussian, with a mean and covariance matrix corresponding to μ and K evaluated on this set of points. We call the resulting model a *Gaussian process modulated renewal process*.

An issue with the model specified above is that samples from a GP can take negative values. Following (Adams et al., 2009), we address this by transforming the GP with a sigmoidal link function. Besides ensuring that the intensity function is nonnegative, this provides us with a bound on the modulating function. This is important for a uniformization-based construction of a renewal process, where we will need to sample from a Poisson process whose rate dominates all event rates in the system of interest.

Finally, we use the gamma family for the hazard function (figure 5.3):

$$h(\tau) = \frac{\tau^{\gamma-1}e^{-\gamma\tau}}{\int_\tau^\infty u^{\gamma-1}e^{-\gamma u}\mathrm{d}u} \tag{5.9}$$

Here γ is the gamma shape parameter. Note that in order to ensure identifiability, we parametrize the hazard function to produce 1 event per unit time; any deviation from this rate can then be attributed to the modulating function. This allows us to decouple the rate of observed events from how they are spread out or clustered in time. Other, more flexible parametrizations may be used as well. Our complete model is thus

$$l(\cdot) \sim \mathcal{GP}(\mu, K), \tag{5.10}$$

$$\lambda(\cdot) = \hat{\lambda}\sigma(l(\cdot)), \tag{5.11}$$

$$G \sim \mathcal{R}(\lambda(\cdot), h(\cdot)) \tag{5.12}$$

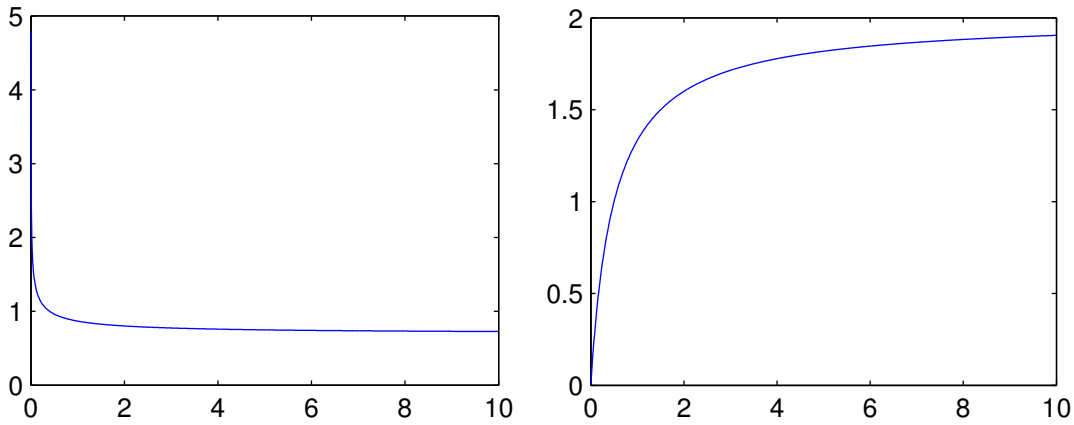

Figure 5.3: Hazard functions for the Gamma distribution with $\gamma = 0.7$ (left) and $\gamma = 2$ (right).

Here, $\mathcal{R}(\lambda(\cdot), h(\cdot))$ refers to the modulated renewal process described earlier with a base hazard function $h(\cdot)$, and a modulating function $\lambda(\cdot)$. $\hat{\lambda}$ is a positive scale parameter, and $\sigma(x) = (1 + \exp(-x))^{-1}$ is the sigmoidal link function. We place a gamma hyperprior on $\hat{\lambda}$ as well as hyperpriors on the GP hyperparameters.

5.2.4 Related work

The idea of defining a nonstationary renewal process by modulating the hazard function dates back to Cox (1972). Early work (Berman, 1981) focussed on hypothesis testing for the stationarity assumption. (Ogata, 1981; Berman and Turner, 1992; Lawless and Thiagarajah, 1996) proposed parametric (generalized linear) models where the intensity function was a linear combination of some known functions; these regression coefficients were estimated via maximum likelihood. Sahin (1993) consider general modulated hazard functions as well; however they assume it has known form and are concerned with calculating statistical properties of the resulting process. Finally, Kass and Ventura (2001) describe a model that is a generalization of ours, but again have to resort to maximum likelihood estimation of the relevant parameters. Our ideas can easily be extended to their more general model as well.

A different approach to producing inhomogeneity is by first sampling from a homogeneous renewal process and then rescaling time (Brown et al., 2002; Gerhardt and Nelson, 2009). The trend renewal process (Lindqvist, 2011) uses such an approach, and the authors propose an iterative kernel smoothing scheme to approximate a maximum likelihood estimate of the intensity function. Cunningham et al. (2008) also use time-rescaling to introduce inhomogeneity and, like us, place a Gaussian process prior on the intensity function. Unlike us however, they had to discretize time and used a variational approach to inference.

Finally, we note that our approach generalizes Adams et al. (2009), who describe a

model which is a special case of ours, using exponential waiting times instead of a more general renewal distribution. Thus, they define a doubly stochastic *Poisson* process, and are able to sample from it without any time discretization by exploiting the thinning theorem. In the next sections we describe a generalization of their sampling scheme to more general renewal processes using a construction based on uniformization. We also describe how to perform inference more efficiently than they did.

5.3 Sampling via Uniformization

Before we consider Markov chain Monte Carlo (MCMC) inference for our model, observe that even naïvely generating samples from the prior is expensive. This requires evaluating integrals of a continuous-time function drawn from a GP (see equation (5.4)). One approach is to discretize time, instantiate the GP on this grid, and then approximate any integrals by the corresponding summations (Cunningham et al., 2008). This, however, can be time consuming, and introduces biases into our inferences. In this section, we show how uniformization allows us to efficiently draw *exact* samples from the model without any approximations such as time-discretization. In this sense, uniformization in fundamental to our problem, which is otherwise intractable. Contrast this with uniformization for MJPs, where it served as an alternate sampling scheme that we exploited to develop an efficient algorithm for MCMC inference.

Ideas relating a renewal process to a thinned latent Poisson process exist in the literature (Ogata, 1981; Shanthikumar, 1986), but these are usually for homogeneous renewal processes, and were not developed with an eye towards inference and sampling. Recall that uniformization for MJPs generalized thinning for Poisson processes by accounting for the Markov dependencies of the MJP. Rather than thinning points independently, we did so by running a discrete-time Markov chain over a set of events sampled from a dominating Poisson process. A similar idea extends to the case of renewal processes. We will assume that both the intensity function $\lambda(t)$ and the hazard function $h(\tau)$ are bounded, so that there exists a constant Ω such that

$$\Omega \geq \max_{t,\tau} h(\tau)\lambda(t) \tag{5.13}$$

Note that because of the sigmoidal link function, our model has $\lambda(t) \leq \lambda^*$, while the gamma hazard $h(\tau)$ is bounded by the shape parameter γ if $\gamma \geq 1$ (see figure 5.3). We now sample a set of times $E = \{E_0 = t_{start}, E_1, E_2, \ldots\}$ from a homogeneous Poisson process with rate Ω, and thin this set by running a discrete time Markov chain on the times in E. Let $Y_0 = 0, Y_1, Y_2, \ldots$ be an integer-valued Markov chain, where each Y_i either equals Y_{i-1} or i. We interpret Y_i as indexing of the last *unthinned* event prior or equal to E_i. That is, $Y_i = Y_{i-1}$ means that E_i is thinned, and $Y_i = i$ means E_i is not thinned. Note that $E_i - E_{Y_{i-1}}$ gives the time since the last unthinned event, so

that $h(E_i - E_{Y_{i-1}})\lambda(E_i)$ gives the hazard rate at time E_i given the state of the renewal process instantiated so far. Since we have generated events from a Poisson process with rate Ω, it follows that we keep event E_i with probability $\frac{h(E_i - E_{Y_{i-1}})\lambda(E_i)}{\Omega}$. Thus, for $i > j \geq 0$, define the transition probabilities of the Markov chain (conditioned on E) as follows,

$$p(Y_i = i | Y_{i-1} = j) = \frac{h(E_i - E_j)\lambda(E_i)}{\Omega}, \qquad \text{(accept } E_i\text{)} \tag{5.14}$$

$$p(Y_i = j | Y_{i-1} = j) = 1 - \frac{h(E_i - E_j)\lambda(E_i)}{\Omega} \qquad \text{(thin } E_i\text{)} \tag{5.15}$$

All other probabilities are 0. After drawing a sample from Y, we define $F = \{E_i \in E \text{ s.t. } Y_i = i\}$.

Proposition 5.2. *For any $\Omega \geq \max_{t,\tau} h(\tau)\lambda(t)$, F is a sample from a modulated renewal process with hazard $h(\cdot)$ and modulating intensity $\lambda(\cdot)$.*

Proof. For a proof of a similar result, see Ogata (1981). Below, we include a different proof based on densities.

We need to show that $F_i - F_{i-1} \sim g(\cdot)$, defined in equation (5.4).
Denote by $E^* = (e_1^*, \cdots, e_n^*)$ the restriction of E to the interval (F_{i-1}, F_i), not including boundaries, so that $|E^*| = n$. Then,

$$\begin{aligned} &P(F_i, E^* | F_{i-1}) \\ &\quad = \Omega^{n+1} \exp(-\Omega(F_i - F_{i-1})) \prod_{j=1}^{n} \left(1 - \frac{\lambda(e_j^*)h(e_j^* - F_{i-1})}{\Omega}\right) \frac{\lambda(F_i)h(F_i - F_{i-1})}{\Omega} \\ &\quad = \exp(-\Omega(F_i - F_{i-1})) \prod_{j=1}^{n} \left(\Omega - \lambda(e_j^*)h(e_j^* - F_{i-1})\right) \lambda(F_i)h(F_i - F_{i-1}) \end{aligned} \tag{5.16}$$

The first term in the first expression above is the probability density of an $(n+1)$-event sample (E^* followed by F_i) under a rate Ω Poisson process, while the second term given the probability of thinning all events except the last. Integrating out the locations of the elements in E^*, we have

$$\begin{aligned} P(F_i, n | F_{i-1}) &= \lambda(F_i)h(F_i - F_{i-1}) \exp\left(-\Omega(F_i - F_{i-1})\right) \\ &\qquad \int_{F_{i-1}}^{F_i} \int_{t_1}^{F_i} \cdots \int_{t_{n-1}}^{F_i} \mathrm{d}t_1 \mathrm{d}t_2 \ldots \mathrm{d}t_n \prod_{j=1}^{n} \left(\Omega - \lambda(t_j)h(t_j - F_{i-1})\right) \end{aligned} \tag{5.17}$$

$$\begin{aligned} &= \lambda(F_i)h(F_i - F_{i-1}) \exp\left(-\Omega(F_i - F_{i-1})\right) \\ &\qquad \frac{1}{n!} \left(\int_{F_{i-1}}^{F_i} \mathrm{d}t \left(\Omega - \lambda(t)h(t - F_{i-1})\right) \right)^n \end{aligned} \tag{5.18}$$

Summing over n, we then have

$$P(F_i|F_{i-1}) = \lambda(t)h(F_i - F_{i-1})\exp\left(-\Omega(F_i - F_{i-1})\right)\exp\left(\int_{F_{i-1}}^{F_i} \mathrm{d}t\,(\Omega - \lambda(t)h(t - F_{i-1}))\right)$$
$$= \lambda(F_i)h(F_i - F_{i-1})\exp\left(-\int_{F_{i-1}}^{F_i} \mathrm{d}t\lambda(t)h(t - F_{i-1})\right) \tag{5.19}$$

Comparing with equation (5.4), we see we have the desired result.

□

Now recall that $l(\cdot)$ is distributed according to a Gaussian process. The uniformization procedure above only requires values of the intensity function at the times in E (which is a finite set on a finite interval), and this is easily obtained by sampling from a finite dimensional Gaussian $\mathcal{N}(\mu_E, K_E)$, with mean and covariance corresponding to the GP parameters μ and K evaluated at E. Our procedure to sample from a GP-modulated renewal process now follows: sample E from a homogeneous Poisson process on $[t_{start}, t_{end}]$, instantiate the GP on this finite set of points and then thin the set by running the Markov chain described previously. Defining l_E as $l(t)$ evaluated on the set E, E_i^* as the restriction of E to the interval (F_{i-1}, F_i), and $T = t_{end} - t_{start}$, we can write the joint distribution:

$$P(F, l, E) = \Omega^{|E|-2}e^{-\Omega T}\mathcal{N}(l_E|\mu_E, K_E)\prod_{i=1}^{|F|-1}\left(\frac{\lambda(F_i)h(F_i - F_{i-1})}{\Omega}\right)\prod_{i=1}^{|F|}\prod_{e\in E_i^*}\left(1 - \frac{\lambda(e)h(e - F_{i-1})}{\Omega}\right) \tag{5.20}$$

5.3.1 Inference

We now consider posterior inference over the modulating function $\lambda(t)$ (and any unknown hyperparameters) given an observed set of event times G. Our sampling algorithm is similar to that from chapter 3. We imagine G was generated via uniformization, so that there exists an unobserved set of thinned events $\tilde{G}$. We then proceed by Markov chain Monte Carlo, setting up a Markov chain whose state consists of the number and locations of $\tilde{G}$, the values of the GP on the set $G \cup \tilde{G} = E$ as well as the current sampled hyperparameters. Note from equation (5.20) that given these values, the value of the modulating function at any other location is independent of the observations and can be sampled from the conditional distribution of a multivariate Gaussian.

The challenge now is to construct a transition operator that results in this Markov chain having the desired posterior distribution as its equilibrium distribution. In their work on doubly stochastic Poisson processes, Adams et al. (2009) defined a birth-death transition operator that proposed insertions and deletions of thinned events. They also defined an operator that randomly perturbed the locations of the thinned

events. The proposals generated by these operators were accepted or rejected using a Metropolis-Hastings correction. The remaining variables were updated using standard Gaussian process techniques. Following ideas from chapter 3, we show that instead of incrementally updating $\tilde{G}$, it is actually possible to produce a new *independent* sample of the entire set $\tilde{G}$ (conditioned on all other variables). This leads to a more natural sampler that does not require any external tuning and that mixes more rapidly because of the global nature of the transitions. Algorithm 5.1 lists the steps involved, while figure 5.4 illustrates the key ideas.

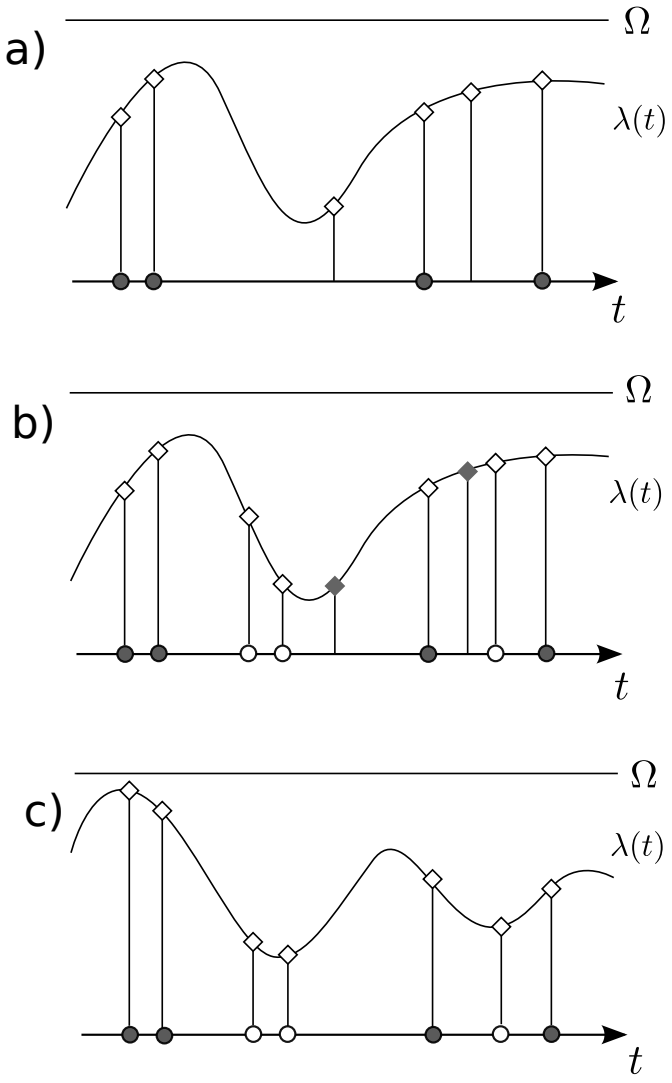

Figure 5.4: a) Renewal events (filled circles) and the GP values (evaluated on two discarded thinned events as well) b) New thinned events (empty circles) and their GP values. The GP values at the old thinned locations, depicted with filled squares are discarded at the end of this step c) Resample GP values

To understand our algorithm, suppose first that the modulating function $\lambda(t)$ is known for all t. Then, from equation (5.4), the probability density of the set of events G on

the interval $[t_{start}, t_{end}]$ is[†]:

$$P(G|\lambda(t)) = \prod_{i=1}^{|G|-1} \lambda(G_i)h(G_i - G_{i-1}) \prod_{i=1}^{|G|} \exp\left(-\int_{G_{i-1}}^{G_i} \lambda(t)h(t - G_{i-1})\mathrm{d}t\right) \tag{5.21}$$

Now, suppose that in each consecutive interval (G_{i-1}, G_i) we independently sample a set of events $\tilde{G}_i^*$ from an inhomogeneous Poisson process with rate $(\Omega - \lambda(t)h(t - G_{i-1}))$, and let $\tilde{G} = \cup_i \tilde{G}_i^*$. A little algebra shows that:

$$\begin{aligned} P(\tilde{G}, G|\lambda(t)) =& \prod_{i=1}^{|G|} \left(\exp\left(-\int_{G_{i-1}}^{G_i} \mathrm{d}t\, (\Omega - \lambda(t)h(t - G_{i-1})\right) \prod_{\tilde{g} \in \tilde{G}_i^*} (\Omega - \lambda(\tilde{g})h(\tilde{g} - G_{i-1})) \right) \\ & \times \prod_{i=1}^{|G|-1} \lambda(G_i)h(G_i - G_{i-1}) \prod_{i=1}^{|G|} \exp\left(-\int_{G_{i-1}}^{G_i} \lambda(t)h(t - G_{i-1})\mathrm{d}t\right) \end{aligned} \tag{5.22}$$

$$= \Omega^{|G|+|\tilde{G}|-2} \exp\left(-\Omega T\right) \prod_{i=1}^{|G|-1} \left(\frac{\lambda(G_i)h(G_i - G_{i-1})}{\Omega}\right) \prod_{i=1}^{|G|} \prod_{\tilde{g} \in \tilde{G}_i^*} \left(1 - \frac{\lambda(\tilde{g})h(\tilde{g} - G_{i-1})}{\Omega}\right) \tag{5.23}$$

Comparing with equation (5.20), we have the following proposition:

Proposition 5.3. *The sets (E, F) and $(G \cup \tilde{G}, G)$ are equivalent i.e. they have the same distribution.*

In other words, given a set of event times G, sampling from the inhomogeneous Poisson process $\tilde{G}$ reconstructs the events thinned in the procedure of section 5.3. Let $G_{last}(t)$ represent the renewal event before time t, so that the thinned events are sampled from a Poisson process with intensity $(\Omega - \lambda(t)h(t - G_{last}(t)))$. Compare this with chapter 3, where the thinned events were distributed as a Poisson process with rate $(\Omega - |A_{\mathbf{S}(t)}|)$; in both cases, this is just Ω minus the hazard rate at time t.

The only complication left is that we do not know the function $\lambda(t)$ everywhere. This is easily overcome by uniformization (in fact, just by thinning, since $\tilde{G}$ is a Poisson process). Specifically, let G be the set of observed events and $\tilde{G}_{prev}$ the previous set of thinned events. To sample the new set $\tilde{G}_i^*$ from the Poisson process on $[G_{i-1}, G_i]$ with rate $(\Omega - \lambda(t)h(t - G_{i-1}))$, we first sample a set of points A from a homogeneous Poisson process on $[G_{i-1}, G_i]$ with rate Ω and instantiate the Gaussian process on those points, *conditioned* on $G \cup \tilde{G}_{prev}$ and $l_{G \cup \tilde{G}_{prev}}$. All this step involves is conditionally sampling from a multivariate Gaussian; in particular, it does not require any complicated GP-sampling algorithm. Finally, we keep an element $a \in A$ with probability $1 - \frac{\lambda(a)h(a - G_{i-1})}{\Omega}$. By the thinning theorem, this gives us a sample from the Poisson process with intensity $(\Omega - \lambda(t)h(t - G_{i-1}))$.

[†]Recall that $G_0 = t_{start}$ and $G_{|G|} = t_{end}$ are not renewal events.

Having resampled $\tilde{G}$ (and the associated set of GP values), we next must resample the value of the GP at G. Observe from equation (5.20) that this step is identical to GP inference for a classification problem: the thinned events are assigned to class '0' and the renewal events to class '1', with the likelihood of class '1' at time E_i given by $\frac{h(E_i - E_{Y_i-1})\lambda(E_i)}{\Omega}$. There are a number of standard MCMC approaches to this problem (eg. hybrid Monte Carlo (Neal, 1993)); we proceed by elliptical slice sampling (Murray et al., 2010) using code available on Iain Murray's website[‡].

Algorithm 5.1 Blocked Gibbs sampler for GP-modulated renewal process on the interval $[t_{start}, t_{end}]$

Input: Sets G and $\tilde{G}_{prev}$ of event and thinned times, and values of l on $G \cup \tilde{G}_{prev}$.
Output: A new set of thinned times $\tilde{G}_{new}$ and a new instantiation $l_{G \cup \tilde{G}_{new}}$ of the GP on $G \cup \tilde{G}_{new}$.

1: Sample $A \subset [t_{start}, t_{end}]$ from a Poisson process with rate Ω.
2: Sample $l_A | l_{G \cup \tilde{G}_{prev}}$ from a Gaussian conditional.
3: Thin A, keeping element $a \in A \cap [G_{i-1}, G_i]$ with probability $\left(1 - \frac{\hat{\lambda}\sigma(l(a))h(a - G_{i-1})}{\Omega}\right)$.
4: Let $\tilde{G}_{new}$ be the resulting set and $l_{\tilde{G}_{new}}$ be the restriction of l_A to this set. Discard $\tilde{G}_{prev}$ and $l_{\tilde{G}_{prev}}$.
5: Resample $l_{G \cup \tilde{G}_{new}}$ using, for example, elliptical slice sampling.

Additionally, we sample the relevant hyperparameters. The gamma prior on $\hat{\lambda}$ is conjugate to the density of the homogeneous Poisson process (see equation (2.27)), resulting in a gamma posterior:

$$\hat{\lambda} | E \sim Gamma(\hat{\alpha}, \hat{\beta}) \tag{5.24}$$

Here, $\hat{\alpha} = \alpha + |E|$, while $1/\hat{\beta} = 1/\beta + 1/(t_{end} - t_{start})$. We resampled the GP hyperparameters using a slice sampler (Murray and Adams, 2010) (once again, using code from Iain Murray's webpage).

We also placed a prior on the parameter γ of the hazard function (see equation (5.9)). Recall that for our approach to apply, we require that the hazard function $h(\tau)$ be bounded. For values of γ less than 1 (corresponding to 'bursty' renewal processes), the gamma hazard function becomes unbounded, and our uniformization-based approach no longer applies. We discuss how to deal with such unbounded hazard function in chapter 6. For now, we restrict ourselves to values larger than 1 by placing an exponential prior shifted to have a minimum value of 1:

$$(\gamma - 1) \sim \exp(\chi) \tag{5.25}$$

The parameter γ of the hazard function was then updated using random walk Metropolis-Hastings moves, with the acceptance ratios calculated using equations (5.20) and (5.25).

[‡] http://homepages.inf.ed.ac.uk/imurray2/

5.3.2 Computational considerations

The computational bottleneck for inference in our model involves the Gaussian process: sampling a GP on a set of points is, in the worst case, cubic in the size of that set. In our model, each iteration sees on average $|G| + 2|E|$ values of the GP, where $|G|$ is the number of observations and $|E|$ is the average number of points sampled from the subordinating Poisson process. Note that $|E|$ varies from iteration to iteration (being proportional to the scaling factor λ^*). Since we perform posterior inference on λ^*, the complexity of our model can be thought to adapt to that of the problem. This is in contrast with time-discretization approaches, where a resolution is picked beforehand, fixing the complexity of the inference problem accordingly. For instance, Cunningham et al. (2008) use a resolution of 1ms to model neural spiking, making it impossible to naïvely deal with spike trains extending over more than a second. However as they demonstrate in their work, instantiating a GP on a regular lattice allows the development of fast *biased* inference algorithms that scale linearly with the number of grid-points. In our case, the Gaussian processes is sampled at random locations. Moreover, these locations change each iteration, requiring the inversion of a new covariance matrix; this is the price we have to pay for an exact sampler.

In chapter 6, we discuss an approach that will allow us to reduce the computational burden by reducing the number of thinned events $|E|$. Essentially, instead of subordinating the renewal process to a homogeneous Poisson process, we construct another point process whose hazard rate more tightly bounds $h(\tau, t)$. This scheme will also allow us to overcome the requirement that the hazard function be bounded, allowing us to deal with bursty renewal processes (such as a Gamma renewal process with shape parameter less that 1).

Such an approach can only reduce the number of thinned events, and will not be of much use if $|G|$, the number of observations itself is large. In such a situation, rather that working with a general covariance kernel (such as a squared-exponential kernel), one can limit oneself to covariance kernels giving precision matrices with compact support. The resulting banded diagonal precision matrix, combined with the fact that we are working in 1-dimensional time, will allow us to use the forward-backward algorithm to carry our inference efficiently. Essentially such an algorithm will exploit a Gaussian process with a Markov structure. Inference for such models will scale cubically with the number of points that lie within an interval of width equal to the support of the kernel, and only *linearly* with the total observation interval.

An alternate approach is to call upon the vast literature concerning approximate inference for Gaussian processes (Rasmussen and Williams, 2006). A question then is how these approximations compare with discrete-time observations like in Cunningham et al. (2008). This is an interesting question in its own right, and raises the possibility of approximate inference algorithms that combine ideas from Cunningham

et al. (2008) with the adaptive nature of our approach.

5.4 Experiments

In this section we evaluate our model and sampler on a number of datasets. All experiments were run using implementations in Matlab. In all experiments, the baseline event-rate of the modulated renewal process (i.e. the rate when the intensity function is fixed at 1) was set to the empirical rate of the observed point process. As a result, any inferences about the shape parameter are a consequence of the dispersion of the point process rather than of some sort of rate matching. In all experiments, we set the exponential parameter χ to 0.1 (see equation (5.25)), resulting in a fairly noninformative prior on γ.

5.4.1 Synthetic data

Our first set of experiments used three synthetic datasets generated by modulating a gamma renewal process (with the shape parameter γ set to 3) with three different functions (see figure 5.5):

- $\lambda_1(t) = 2\exp(t/5) + \exp(-((t-25)/10)^2, \quad t \in [0, 50]$: 44 events
- $\lambda_2(t) = 5\ \sin(t^2) + 6, \quad t \in [0, 5]$: 12 events
- $\lambda_3(t)$: a piecewise linear function , $\quad t \in [0, 100]$: 153 events

For each function, we also generated 10 test sets. We then ran three settings of our model: with the shape parameter fixed to 1, with the shape parameter fixed to the truth, and with a shifted-exponential hyperprior on the shape parameter. We call these setting (MRP Exp), (MRP Gam3) and (MRP Full) respectively. For comparison, we also ran an approximate discrete-time sampler where the Gaussian process was instantiated on a regular grid covering the interval of interest. In this case, all intractable integrals were approximated numerically and we used elliptical slice sampling to run MCMC on this Gaussian vector. In all cases, we used a GP with a squared exponential kernel, and placed log-normal priors on the GP hyperparameters.

Figure 5.5 shows the results from 5000 MCMC samples after a burn-in of 1000 samples. We quantify these in Table 5.1 by calculating the l_2 distance of the posterior means (evaluated on a fixed grid) from the truth. We also calculated the mean predictive probabilities of the 10 test sequences. Not surprisingly, the inhomogeneous Poisson process forms a poor approximation to the gamma renewal process; in particular, it underestimates the intensity function required to produce a clustered sequence of events. Fixing the shape parameter to the truth significantly reduces the l_2 error and increases the predictive probabilities, but interestingly, for these datasets, the model

	MRP Exp	MRP Gam3	MRP Full	Disc25	Disc100
l_2 error	7.8458	3.19	2.548	4.089003	**2.426973**
log pred. prob.	-47.5469	-38.0703	**-37.3712**	-41.646350	-41.016425
l_2 error	141.0067	**56.2183**	58.4361	91.321069	57.896300
log pred. prob.	-3.704396	**-2.945298**	-3.280871	-5.245478	-3.848443
l_2 error	82.0289	**11.4167**	13.4441	122.335151	38.047332
log pred. prob.	-89.8787	**-48.2777**	-48.57	87.170034	-55.802997

Table 5.1: l_2 distance from the truth, and mean log-predictive probabilities of the held-out datasets for synthetic datasets 1(top) to 3(bottom).

with a prior on the shape parameter performs comparably with the 'oracle' model. We have also included plots of the posterior distribution over the gamma parameter; these are peaked around 3. Discretizing time into a 100 bins (Disc100) results in comparable performance for the first two datasets on the l_2 error. For the third, (which spans a longer interval and has a larger event count), we had to increase the resolution to 500 bins to improve accuracy. Discretizing to 25 bins was never sufficient. A conclusion is that with time discretization, to keep the bias small, one must be conservative in choosing the time-resolution; however, evaluating a GP on a fine grid can result in slow mixing. Our sampler has the advantage of automatically picking the 'right' resolution. However as we discussed in the section on computation, time discretization has its own advantages that make it a viable approach (Cunningham et al., 2008).

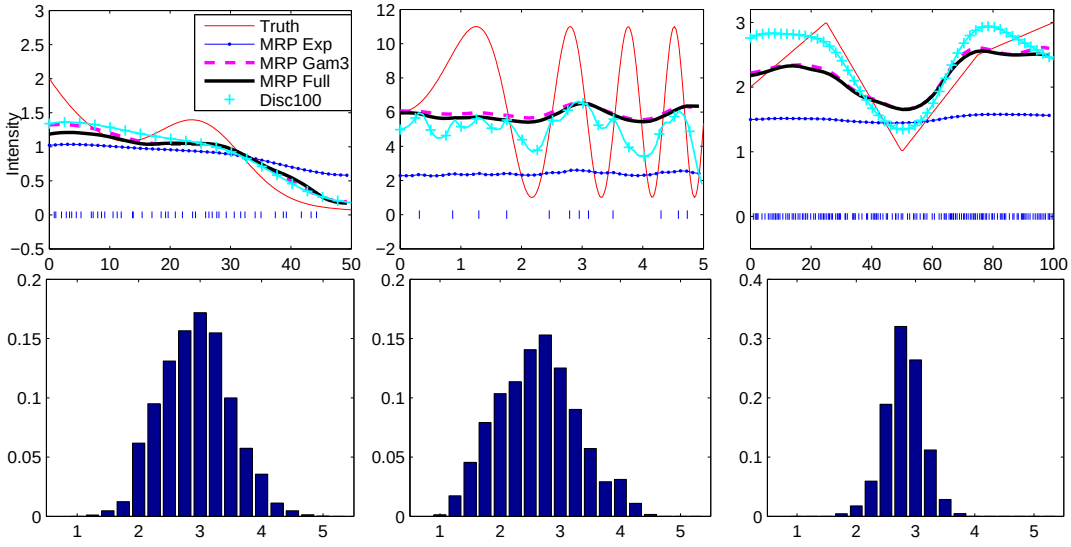

Figure 5.5: Synthetic Datasets 1-3: Posterior mean intensities plotted against time (top) and gamma shape posteriors (bottom)

5.4.2 Identifiability of the Gamma shape parameter

In this experiment, we looked more carefully at the issue of the identifiability of the Gamma shape parameter under our model. We generated a sequence of renewal events

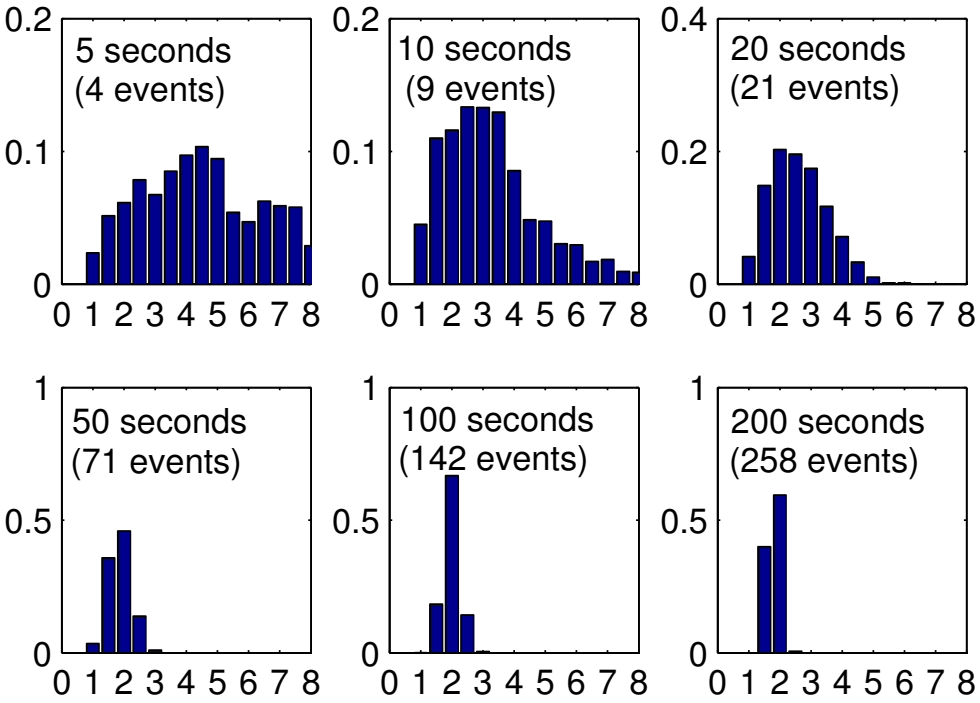

Figure 5.6: Posterior over the gamma shape parameter having observed the modulated renewal process over intervals of increasing length

from our GP-modulated renewal process over the interval $[0, 200]$. The modulating function was drawn from a Gaussian process with a squared exponential covariance kernel. The GP mean was set to 2, and both the log-length scale and the log-variance of the kernel were set to 1. The GP was transformed by a sigmoid function, scaled by a factor of 5, and then used to modulate a Gamma(2,1/2) renewal process to produce a sequence of 258 events over the interval $[0, 200]$.

We then looked at the posterior inferences produced by our sampler as we observed longer and longer sequences of renewal events. In other words, we restricted the modulated renewal events to the interval $[0, T]$, and for increasing values of T, looked at the sequence of posterior distributions over γ. We placed an exponential prior on $(\gamma - 1)$, the mean of this distribution was set to 10, resulting in an uninformative prior over γ. We placed a Gamma$(5, 1)$ prior on $\hat{\lambda}$, the scaling constant of the sigmoid. The cubic cost of inference with the squared exponential kernel meant that we could not handle interval lengths much longer than 200. Moreover, the most expensive operation of our sampler is the slice-sampler that resamples the GP hyperparmeters, and we therefore fixed the GP hyperparameters to their true values. Thus the parameters we performed inference over were the latent GP, the scale parameter $\hat{\lambda}$ and the Gamma shape parameter γ.

Figure 5.6 shows the posterior distribution over γ as T took values $1, 5, 10, 50, 100$ and 200. The posterior over γ starts from very uninformative to a distribution more and more concentrated around the truth. This suggests that we are able to average out

	5 seconds	10 seconds	20 seconds	50 seconds	100 seconds	200 seconds
Mean	4.8312	3.7600	2.8498	2.1137	2.2360	2.0484
Median	4.7001	3.3738	2.7561	2.0827	2.2368	2.0386
.1 quartile	2.2144	1.7721	1.7149	1.6683	1.8989	1.8305
.9 quartile	7.6652	6.3029	4.0993	2.6126	2.5602	2.2693

Table 5.2: Convergence of the posterior on the Gamma shape paremeter

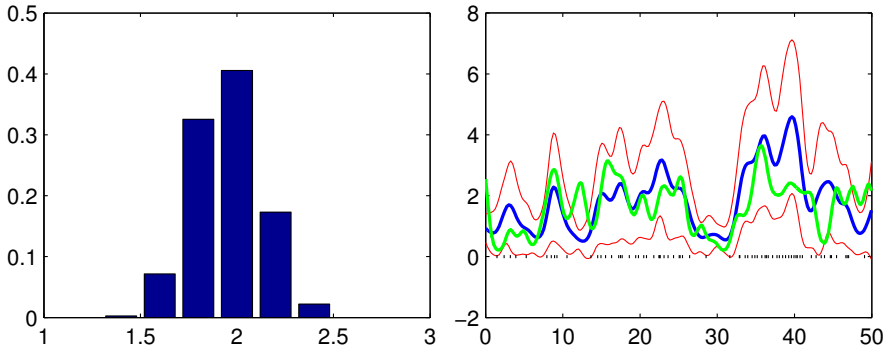

Figure 5.7: (left) Posterior over the Gamma shape parameter having observed 258 events over 200 seconds (right) The true modulating function (green), as well as the posterior mean ± 1.5 standard deviations

the fluctuations introduced by the nuisance parameter (the GP) and identify the true refractoriness of the model. The posterior was constructed from 2000 MCMC iterations, after a burnin period of 500.

Figure 5.7 shows more clearly the posterior distribution given the entire set of 258 renewal events. It also includes the true modulating function (in green), and the posterior distribution over the modulating function (along with 1.5 standard deviations). For clarity we plot only the restriction to the interval $[0, 50]$. The modulating function is clearly not identifiable due to statistical noise (equivalently, we have only a fixed amount of information about the GP value at any time, independent of the length of the renewal sequence). However, if we observe more and more renewal sequences modulated by the same function, we can obtain enough information to recover this quantity as well.

5.4.3 Coal mine disaster data

For our next experiment, we ran our model on the coal mine disaster dataset commonly used in the point process literature. This dataset records the dates of a series of 191 coal mining disasters, each of which killed ten or more men (Jarrett, 1979). Figure 5.8(left) shows the posterior mean of the intensity function (surrounded by 1 standard deviation) returned by our model. Not included is the posterior distribution over the shape parameter; this concentrated in the interval from 1 to 1.1, suggesting that the data is well modelled as an inhomogeneous Poisson process, and is in agreement with

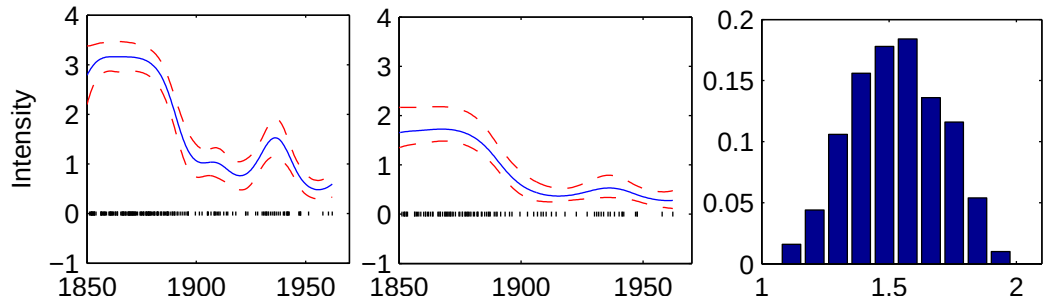

Figure 5.8: Left: Posterior mean intensity for coal mine data with 1 standard deviation error bars (plotted against time in years). Centre: Posterior mean intensity for 'thinned' coalmine data with 1 standard deviation error bars. Right: Gamma shape posterior for 'thinned' coal mine data.

(Jarrett, 1979). As sanity check, and to shed further light on the issue of identifiability, we processed the dataset by deleting every alternate event. With such a transformation, a homogeneous Poisson would reduce to a gamma renewal process with shape 2. This need not be the case for an inhomogeneous Poisson process, with the result depending on the nature of the inhomogeneity. Our model returns a posterior peaked around 1.5 (in agreement with the shape of the inhomogeneity). Note that the posteriors over intensity functions are similar (except for the obvious scaling factor of about 2).

5.4.4 Spike timing data

We next ran our model on neural spike train data recorded from grasshopper auditory receptor cells (Rokem et al., 2006). [§] Rokem et al. (2006). This dataset is characterized by a relatively high firing rate ($\sim$ 150 Hz), making refractory effects more prominent. We plot the posterior distribution over the intensity function given a sequence of 200 spikes in a 1.6 second interval. We also included the posterior distribution over gamma shape parameters in figure 5.9; this concentrates around 1.5, agreeing with the refractory nature of neuronal firing. The results above follow from using noninformative hyperpriors; we have also plotted the log-transformed stimulus, an amplitude-modulated signal. In practice, other available knowledge (viz. the shape parameter, the stimulus length-scale, the transformation from the stimulus to the input of the neuron etc) can be used to make more accurate inferences.

5.4.5 Computational efficiency and mixing

For our final experiment, we compare our proposed blocked Gibbs sampler with the Metropolis-Hastings sampler of Adams et al. (2009)[¶]. We ran both algorithms on two datasets, synthetic dataset 1 from section 5.4.1 and the coal mine disaster dataset.

[§]Collected by Ariel Rokem at Andreas Herz's lab; provided through the CRCNS program (`http://crcns.org`)

[¶]We thank Ryan Adams for providing us with his code.

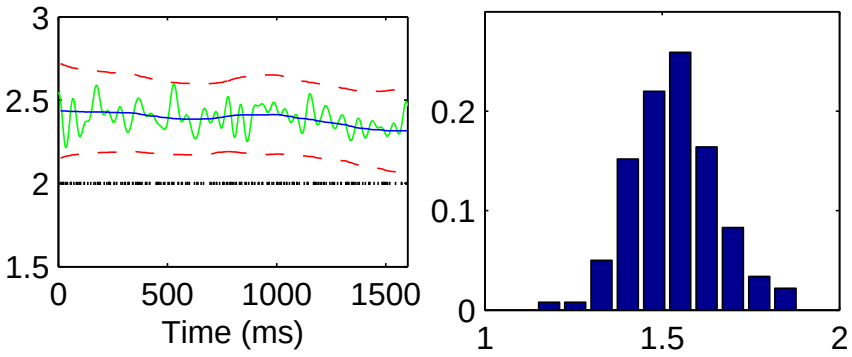

Figure 5.9: Left: Posterior mean intensity for neural data with 1 standard deviation error bars. Superimposed is the log stimulus (scaled and shifted). Right: Posterior over the gamma shape parameter.

	Synthetic dataset 1			Coalmine dataset		
	Mean ESS	Minimum ESS	Time(sec)	Mean ESS	Minimum ESS	Time(sec)
Gibbs	93.45 ± 6.91	50.94 ± 5.21	77.85	53.54 ± 8.15	24.87 ± 7.38	282.72
MH	56.37 ± 10.30	19.34 ± 11.55	345.44	47.83 ± 9.18	18.91 ± 6.45	1703

Table 5.3: Sampler comparisons. Numbers are per 1000 samples.

All involved 20 MCMC runs of 5000 iterations each (following a burn-in period of a 1000 iterations). For both datasets, we evaluated the latent GP on a uniform grid of 200 points, calculating the effective sample size (ESS) of each component of the Gaussian vectors (using R-CODA (Plummer et al., 2006)). For each run, we return the mean and the minimum ESS across all 200 components. In Table 5.3, we report these numbers: not only does our sampler mix faster (resulting in larger ESSs), but also takes less computation time. Additionally, our sampler is simpler and more natural to the problem, and does not require any external tuning parameters.

5.5 Discussion

In this chapter, we have described how uniformization allows us to produce exact samples from a nonstationary renewal process whose hazard function is modulated by a Gaussian process. Like the previous chapters, we exploited our uniformization-based construction to develop a novel and efficient MCMC sampler for posterior inference.

There are a number of interesting avenues worth following. First is the restriction that the hazard function be bounded: while this covers a large and useful class of renewal processes, it would be useful to extend our approach to produce exact samples for renewal processes with unbounded hazard functions. We leave the description of such a scheme for the next chapter. In any case, following Ogata (1981), it is easy to extend our ideas to Bayesian inference for more general point processes with bounded hazard rates. For instance, the firing rate at any instant can depend not just on the time since the last event, but on the entire pattern of the previous event history. Such models are

often more realistic descriptions of various phenomena (Paninski et al., 2007), and for the case of a completely observed point process, we can handle such extensions without incurring any additional computational burden.

Observe that while our generative process involved running a Markov chain on the set of Poisson events, unlike chapter 3, posterior inference did not involve running the forward-backward algorithm. The reason is that we are working in a framework where the renewal events are completely and perfectly observed. We can easily relax this restriction, allowing the event times to be observed noisily, and even allow missing times. Such a situation *will* require a forward-backward sampling scheme, and can have a complexity that scales quadratically with the number of Poisson events (this is not linear like the MJP because our system is no longer Markov). The next chapter which describes inference for continuous-time semi-Markov jump processes will make clear how one might deal with issues like noisy observation times, missing events etc.

A limitation with our proposal of using a Gaussian process prior on the modulating functions is that inference scales cubically with the total number of Poisson points (thinned or otherwise). Thus our approach will not scale well to large problems. As we suggested, because we are working with point processes (and therefore GPs) on the real line, it is possible to choose covariance kernels (other than the squared-exponential) that allow efficient, *linear* inference. The idea is essentially to use a kernel whose precision matrix has a finite support, and then use efficient forward-backward sampling techniques. We also show in the next chapter how we can reduce the number of thinned events, making GP inference easier. There is also a vast literature concerning approximate sampling for Gaussian processes. An important question is how these approximations compare to approximations introduced via time-discretization. Additionally, even though we considered GP modulating functions, our uniformization-based sampler will also be useful for Bayesian inference involving simpler priors on modulating functions, eg. splines or Markov jump processes.

Chapter 6

Beyond uniformization: subordinating to general continuous-time processes

6.1 Introduction

In the last three chapters, we studied a framework for efficient posterior inference in continuous-time discrete-state systems based on the idea of uniformization. We started with the Markov jump process, and after studying two extensions of this model (viz. the MMPP and the CTBN), we moved on to renewal processes. In this chapter, we extend our ideas to semi-Markov processes. Semi-Markov processes are essentially generalizations of the MJP where the waiting times of each state follow some general density on $\mathbb{R}_+$ beyond the memoryless exponential. Equivalently, these are generalizations of renewal processes that allow for multiple states with different dynamics. Working with these processes, we shall see that the uniformization framework of the previous chapters can prove restrictive, and we will develop methods beyond uniformization to carry out MCMC inference.

Recall that uniformization involves first sampling candidate event times from a Poisson process whose rate dominates all event rates in the system of interest. This restricts us to systems with bounded event rates; we saw for example that our methods in the last chapter do not extend to bursty renewal densities with unbounded hazard rates. Similarly, recall that in the Lotka-Volterra model (subsection 4.3.1), birth and death rates are proportional to the sizes of the relevant populations. Since we cannot *a priori* bound the maximum size of a population over any finite interval, we cannot construct a constant bound on all event rates in the system. In chapter 4, we got around this issue by approximating the original system with a truncated one that *does* have a bounded population size. This however introduces a bias into our inferences, and to

keep this small, we needed a conservative bound on the population size (and thus on the maximum event rate in the system). This in turn can lead to bounding rates that are significantly larger than typical rates witnessed in the system, introducing a large number of thinned events that have to be resampled at the time of inference. A related source of inefficiency is the presence of states in an MJP with widely differing event rates. Again, by picking a single rate Ω that dominates all event rates in the system, the average number of Poisson events (and thus the computational cost of our algorithm) scales with the leaving rate of the most unstable state. At the same time, this state is often the one that the system will spend the least amount of time in.

A hint of the ideas that follow was provided when we studied inference for CTBNs (section 4.2). There, rather than picking a single dominating Poisson rate for a node of a CTBN, we allowed the dominating rate to depend on the current configuration of the parents of the node. In this chapter, we extend this idea, allowing the dominating Poisson rate to vary not just with configuration of a node's Markov blanket (if any) but also with the state of the node itself. This will allow us to develop a general framework for MCMC inference for a much wider variety of continuous-time discrete-state systems. First however, to provide ourselves with a concrete problem to address, we introduce semi-Markov processes.

6.2 Semi-Markov processes

Like the Markov jump process, a semi-Markov process is a right-continuous, piecewise-constant stochastic process on the nonnegative real-line taking values in some state space $\mathcal{S}$ (Feller, 1964; Sonderman, 1980). We abbreviate this process as sMJP, the 'J' emphasizing that it always is a jump process. For simplicity, we assume the state space $\mathcal{S}$ is finite, labelling its elements from 1 to N. We also assume the process is stationary. Then, the sMJP is parametrized by π_0, a probability measure on $\mathcal{S}$ giving the initial distribution over states, as well as an $N \times N$ matrix of *subdistribution functions* $D = [D_{ss'}(\cdot)]$. The value $D_{ss'}(\tau)$ gives the cumulative probability distribution of the system transitioning to state s', τ time units after entering state s. Thus, $D_{ss'}(\tau)$ is a positive, monotonic function of τ, with

$$\sum_{s' \in \mathcal{S}} D_{ss'}(\infty) = 1 \quad \forall s \tag{6.1}$$

We point out that unlike an MJP, sMJPs are usually defined to allow self-transitions; this amounts to resetting the current waiting time τ of a state. Assume that the distributions $D_{ss'}(\tau)$ admit densities with respect to Lebesgue measure, and call these $r_{ss'}(\tau)$. From equation (6.1), we see that $D_{ss'}(\infty)$ is usually less than 1, so that $r_{ss'}(\tau)$ is not normalized. Rather, it is related the joint probability of waiting in state s for a time τ and then jumping to state s' (however, when we talk of sampling a time τ from

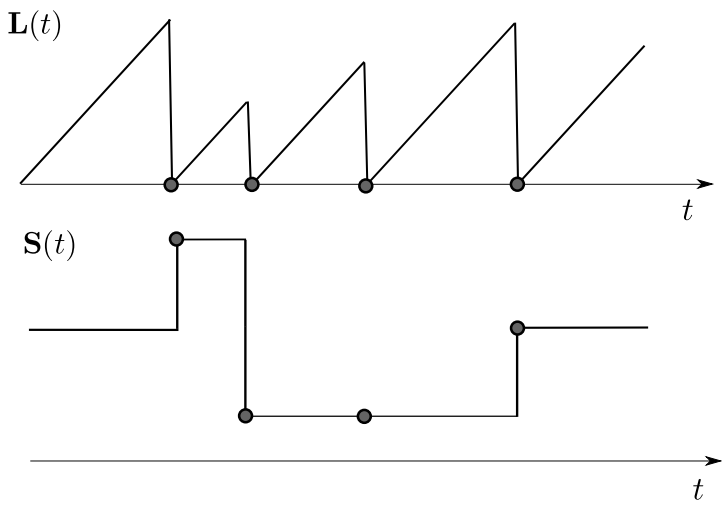

Figure 6.1: A Sample sMJP trajectory. The times of the filled dots correspond to T, while the corresponding values of $\mathbf{L}(t)$ and $\mathbf{S}(t)$ correspond to L and S respectively.

$r_{ss'}(\cdot)$, we will mean from the normalized density). Let $A_{ss'}(\tau)$ be the hazard function associated with the density $r_{ss'}(\tau)$, so that following chapter 5, we can easily show that:

$$r_{ss'}(\tau) = A_{ss'}(\tau) \exp\left(-\int_0^\tau A_{ss'}(u)\mathrm{d}u\right), \tag{6.2}$$

$$A_{ss'}(\tau) = \frac{r_{ss'}(\tau)}{\int_0^\infty r_{ss'}(u)\mathrm{d}u - \int_0^\tau r_{ss'}(u)\mathrm{d}u}, \quad \text{and} \tag{6.3}$$

$$D_{ss'}(\tau) = 1 - \exp\left(-\int_0^\tau A_{ss'}(u)\mathrm{d}u\right) \tag{6.4}$$

$A_{ss'}(\tau)$ gives the rate of transitioning to state s', τ time units after having entered state s. Entering state s initiates a 'race' between the hazard functions $A_{ss'}(\cdot)\ \forall s' \in \mathcal{S}$ to produce the first event. Accordingly, we sample waiting times $\tau_{s'} \sim A_{ss'}(\cdot)\ \forall s' \in \mathcal{S}$ (we shall interchangeably talk about sampling a waiting time from the density $r_{ss'}(\cdot)$ or from its associated hazard function $A_{ss'}(\cdot)$). The sMJP then enters a new state corresponding to the smallest of these waiting times. Let this state be s_{new} (we allow self-transitions, so s_{new} can equal s), and let the waiting time be τ_{hold} (so that $\tau_{hold} = \tau_{s_{new}} = \min_{s'} \tau_{s'}$). Then, advance the current time by τ_{hold}, set the sMJP state to s_{new}, and repeat this procedure, now with the rate functions $A_{s_{new}s'}(\cdot)\ \forall s' \in \mathcal{S}$. This direct approach to sampling an sMJP trajectory corresponds to Gillespie's algorithm for Markov jump processes. The lower plot in figure 6.1, shows a sample sMJP trajectory, with the filled dots representing the state transitions times T and the corresponding state values S.

From equation (6.4), we have that

$$P(\tau_{hold} > \tau) = \prod_{s' \in \mathcal{S}} P(\tau_{s'} > \tau) \tag{6.5}$$

$$= \prod_{s' \in \mathcal{S}} \exp\left(- \int_0^\tau A_{ss'}(u) \mathrm{d}u \right) \tag{6.6}$$

$$= \exp\left(- \int_0^\tau A_s(u) \mathrm{d}u \right) \tag{6.7}$$

where

$$A_s(\tau) = \sum_{s' \in \mathcal{S}} A_{ss'}(\tau) \tag{6.8}$$

Similarly, it follows that

$$r_s(\tau_{hold}) \equiv P(\tau_{hold} = \tau) \tag{6.9}$$

$$= A_s(\tau) \exp\left(- \int_0^\tau A_s(u) \mathrm{d}u \right) \tag{6.10}$$

Comparing with equation (6.2), we see (as we might expect) that $A_s(\tau)$, gives the rate of *any* transition out of state s occurring (including self-transitions), after τ time units have elapsed since entering s. Thus $A_s(\cdot)$ is the hazard function of the minimum waiting time τ_{hold}. Let s_{next} be the new state that the sMJP now jumps into, and s_{curr} be the current state. Then from equations (6.2) and (6.10) we see that

$$P(s_{next} = s' | s_{curr} = s, \tau_{hold} = \tau) \propto A_{ss'}(\tau) \tag{6.11}$$

so that

$$P(s_{next} = s', t_{next} = t_{curr} + \tau_{hold} | s_{curr} = s) = A_{ss'}(\tau_{hold}) \exp\left(- \int_0^{\tau_{hold}} A_s(u) \mathrm{d}u \right) \tag{6.12}$$

Thus, an equivalent way to sample an sMJP trajectory is by first sampling the time until the next jump, t_{hold}, from equation (6.10), and then sampling the new state s' from equation (6.11). When the hazard functions $A_{ss'}(\cdot)$ are all constant (so that the distributions $r_{ss'}(\cdot)$ are exponential), and when $A_{ss} = 0$ for all $s \in \mathcal{S}$, we recover the Markov jump process. Gillespie's algorithm corresponds to directly sampling the waiting time t_{hold} from equation (6.10), and then picking a new state using equation (6.11). Note that the rate matrix A defined here for the MJP is slightly different from the generator matrix A defined in chapter 3. For the latter, the diagonal element A_{ss} equalled $-A_s$, the negative of the total rate of leaving state s. Here, on the other hand, the diagonal

element A_{ss} gives the rate of making a self-transition from state s back to itself. We shall work with this definition of A for this and the next chapter.

In chapter 3, we represented a sample trajectory as $\mathbf{S}(t) \equiv (S, T)$, where T is is the set of jump times (including the endpoints) and S is the corresponding set of state values. For the sMJP, $\mathbf{S}(t)$ alone is not sufficient to identify when self-transitions occurred. There are a number of ways to augment $\mathbf{S}(t)$ to identify self-transitions as well. For later purposes, we do this by including another function $\mathbf{L}(t)$ (shown in figure 6.1). At any time t, this gives the time since the last state transition (inclusive of the current time t); i.e.

$$\mathbf{L}(t) = \min_{t^* \in T, t^* \leq t} (t - t^*) \tag{6.13}$$

Now, the pair $(\mathbf{S}(t), \mathbf{L}(t))$ is an equivalent representation of (S, T). Observe that $\mathbf{L}(t)$ evaluated on the set of times in T equals 0.

Figure 6.1 shows all the relevant quantities, while algorithm 6.1 describes the steps involved in sampling an sMJP trajectory over a finite interval $[t_{start}, t_{end}]$.

Algorithm 6.1 Algorithm to sample an sMJP path on the interval $[t_{start}, t_{end}]$

Input: The initial distribution over states π_0 and the collection of subdistribution functions $D_{ss'}(\cdot)\ \forall s, s' \in \mathcal{S}$. The latter specify the transition hazard rates $A_{ss'}(\cdot)$ and the waiting time densities $r_{ss'}(\cdot)$.

Output: An sMJP trajectory (S, T).

1: Assign the MJP a state $s_0 \sim \pi_0$. Set $t_0 = t_{start}$ and $i = 0$.
2: **while** $t_i < t_{end}$ **do**
3: Increment i.
4: For all $s \in \mathcal{S}$, draw $\tau_s \sim r_{s_{i-1}s}(\cdot)$. Let $z = \operatorname{argmin} \tau_s$ and $\tau_{hold} = \tau_z$.
5: Set $t_i = t_{i-1} + \tau_{hold}$.
6: Set $s_i = z$. Note that s_i can equal s_{i-1}.
7: **end while**
8: Set $t_i = t_{end}$, and $s_i = s_{i-1}$.

By allowing the waiting times to follow a general density, the process $\mathbf{S}(t)$ is no longer Markov. We can thus include memory effects like burstiness and refractoriness in the system dynamics. However, given the state of the system at any time, knowing *when* the process entered that state helps predict the time and destination of the next transition. Thus, given the pair $(\mathbf{S}(t), \mathbf{L}(t))$, the future *is* independent of the past, and this system is Markov. Later, when we describe an MCMC sampler for sMJPs, we will construct a forward-backward sampling algorithm on a Markov chain whose state at time t is $(\mathbf{S}(t), \mathbf{L}(t))$. For later use, we note down the transition densities for this augmented Markov chain. Let $t_3 > t_2 \geq t_1$, and let t_3^- be the time infinitesimally before t_3. Then,

from equations (6.10) and (6.7), we have

$$P(\mathbf{S}(t_3) = s', \mathbf{L}(t_3) = 0, \mathbf{L}(t_3^-) = t_3 - t_1 | \mathbf{S}(t_2) = s, \mathbf{L}(t_2) = t_2 - t_1)$$
$$= A_s(t_3 - t_1) \exp\left(- \int_{(t_2 - t_1)}^{(t_3 - t_1)} A_s(\tau) \mathrm{d}\tau \right) \frac{A_{ss'}(t_3 - t_1)}{A_s(t_3 - t_1)} \quad (6.14)$$
$$= A_{ss'}(t_3 - t_1) \exp\left(- \int_{(t_2 - t_1)}^{(t_3 - t_1)} A_s(\tau) \mathrm{d}\tau \right) \quad (6.15)$$

The first term on the right-hand side of equation (6.14) is the probability of t_{hold} equalling $(t_3 - t_1)$ conditioned on it being greater that $(t_2 - t_1)$. The second term is the probability of then jumping to state s' (we repeat that s' can equal s). We also have that

$$P\left(\mathbf{S}(t_3) = s, \ \mathbf{L}(t_3) = t_3 - t_1 | \mathbf{S}(t_2) = s, \mathbf{L}(t_2) = t_2 - t_1\right)$$
$$= P(\mathbf{L}(t_3) = t_3 - t_1 | \mathbf{S}(t_2) = s, \mathbf{L}(t_2) = t_2 - t_1) \quad (6.16)$$
$$= \exp\left(- \int_{(t_2 - t_1)}^{(t_3 - t_1)} A_s(\tau) \mathrm{d}\tau \right) \quad (6.17)$$

6.3 Dependent thinning for semi-Markov processes

Chapter 5 suggests a uniformization-based approach to sampling from a semi-Markov process by subordinating it to a Poisson process with rate $\Omega \geq \sup_{s,s',\tau} A_{ss'}(\tau)$. Such an approach was described in Sonderman (1980). Assuming such a finite Ω exists, this procedure is a straightforward combination of the uniformization schemes for MJPs and for renewal processes. We do not specify it here; as we shall see later, this will turn out to be a special case of the algorithm we propose. Instead, we reiterate its limitations. First, if Ω is much larger than typical events rates in the system, then the large number of thinned events can lead to significant inefficiency. More importantly, this scheme requires $\sup_{s,s',\tau} A_{ss'}(\tau)$ to be finite. As we saw in the introduction, this is not always the case (a simple example is when the density of a waiting time $r(\tau)$ is gamma distributed with shape parameter less than 1).

In this section, we will describe an alternate approach to sampling an sMJP trajectory, based on a procedure of dependent thinning. We saw in the previous section that we could sample an sMJP trajectory (S, T) by successively sampling τ_{hold}, the waiting time until the next event, and then the event identity. Our approach here is to sample a *candidate* event time from a distribution corresponding to a rate $U_s(\tau)$ that dominates $A_s(\tau)$. Thus, for every hazard function $A_s(\tau)$, pick some other dominating hazard function $U_s(\tau)$, so that

$$U_s(\tau) \geq A_s(\tau) \quad \forall s, \tau \quad (6.18)$$

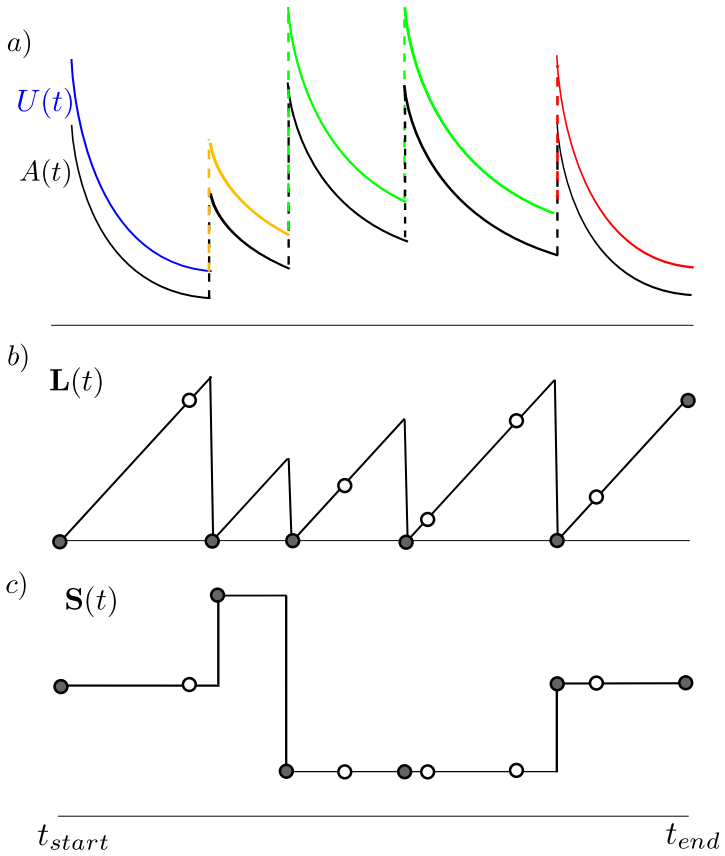

Figure 6.2: A Sample sMJP trajectory. The filled dots correspond to actual events, while the empty dots are thinned events. Note that we can distinguish self-transitions from thinned events from the values of $\mathbf{L}(\cdot)$ at these times.

To correct for the fact that candidate events are being produced at a rate higher than the actual event rate in the system, we probabilistically thin these events. Define the sequence W as the union of actual event times, T (the filled dots in figure 6.2), together with the thinned event times (which we call $\tilde{W}$, these are the empty circles in figure 6.2). We define $V = (v_0, \cdots, v_{|W|})$ as the sequence of state assignments to the times W, and $L = (l_1, \cdots, l_{|W|})$ as the corresponding values of l (so that $l_i = \mathbf{L}(w_i)$). These quantities are also shows in figure 6.2. Observe now that $v_i = v_{i+1}$ does not necessarily imply a self-transition at time w_i, since v_i could correspond to a thinned candidate event. For actual events (including self-transitions), $l_i = 0$, otherwise l_i tells us that the last actual transition occured at time $w_i - l_i$.

Our scheme is now a straightforward application of the fact that given (v_i, l_i), the future is completely independent of events before w_i. Suppose the system just entered state v_i at time w_i (so that $l_i = 0$). We sample the next candidate event time w_{i+1}, with $\Delta w_i = (w_{i+1} - w_i)$ drawn from the hazard function $U_{v_i}(\cdot)$. Recall that $U_{v_i}(\cdot)$ dominates $A_{v_i}(\cdot)$, so that Δw_i will on average be smaller than a sample from $A_{v_i}(\cdot)$. We correct for this by treating w_{i+1} as an actual event with probability $\frac{A_{v_i}(\Delta w_i + l_i)}{U_{v_i}(\Delta w_i + l_i)}$. If this is the case, we sample a new state v_{i+1} with probability proportional to $U_{v_i v_{i+1}}(\Delta w_i + l_i)$, and set $l_{i+1} = 0$. On the other hand, if the event is rejected, we keep v_{i+1} equal

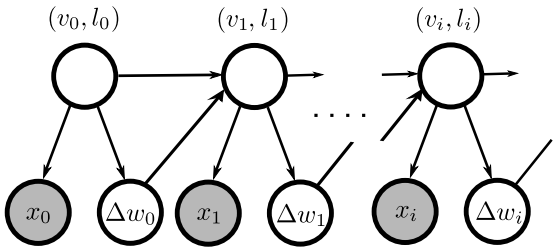

Figure 6.3: Discrete-time Markov chain for the forward-backward algorithm. $\Delta w_i = w_{i+1} - w_i$, and x_i represents actual observations in the interval (w_i, w_{i+1}).

to v_i, and set $l_{i+1} = (\Delta w_i + l_i)$. We now sample Δw_{i+1} (and thus w_{i+2}), such that $(\Delta w_{i+1} + l_{i+1}) \sim U_{v_{i+1}}(\cdot)$. More simply, we sample a waiting time from $U_{v_{i+1}}(\cdot)$, *conditioned on it being greater than* l_{i+1}. Again, accept this point with probability $\frac{A_{v_{i+1}}(\Delta w_{i+1}+l_{i+1})}{U_{v_{i+1}}(\Delta w_{i+1} l_{i+1})}$, and repeat this process.

Figure 6.2 and algorithm 6.2 describes our dependent thinning scheme; this amounts to successively sampling a new candidate time w_i, and assigning it label (v_i, l_i). Figure 6.3 summarizes this procedure with a graphical model. The latter also depicts observations X of the sMJP trajectory; we explain this later.

Algorithm 6.2 State-dependent thinning for sMJPs

Input: A matrix of hazard functions $A_{ss'}(\tau), \forall s, s' \in \mathcal{S}$ and an initial distribution over states π_0.
For each state s, a dominating hazard function $U_s(\tau) \geq A_s(\tau)\ \forall \tau$, where $A_s(\tau) = \sum_j A_{sj}(\tau)$.

Output: An sMJP path $(V, L, W) \equiv ((v_i, l_i, w_i))$ on the interval $[t_{start}, t_{end}]$.

1: Draw $v_0 \sim \pi_0$ and set $w_0 = t_{start}$. Set $l_0 = 0$ and $i = 0$.
2: **while** $w_i < t_{end}$ **do**
3: Sample $\tau_{hold} \sim U_{v_i}(\cdot)$, conditioning on $\tau_{hold} > l_i$.
4: Let $\Delta w_i = \tau_{hold} - l_i$, and $w_{i+1} = w_i + \Delta w_i$.
5: **with probability** $\frac{A_{v_i}(\tau_{hold})}{U_{v_i}(\tau_{hold})}$
6: Set $l_{i+1} = 0$, and sample v_{i+1}, with $P(v_{i+1} = s' | v_i) \propto A_{v_i s'}(\tau_{hold}),\ s' \in \mathcal{S}$.
7: **else**
8: Set $l_{i+1} = l_i + \Delta w_i$, and $v_{i+1} = v_i$.
9: **end**
10: Increment i.
11: **end while**
12: Set $w_{|W|} = t_{end}$, $v_{|W|} = v_{|W|-1}$, $l_{|W|} = l_{|W|} + w_{|W|} - w_{|W|-1}$.

Conceptually, our algorithm can be thought to bridge a gap between Gillespie's algorithm and uniformization. When the dominating hazard functions $U_i(\cdot)$ are identical to the corresponding hazard functions $A_i(\cdot)$, we recover Gillespie's algorithm (where we have no thinned events). Uniformization corresponds to a single, constant dominating hazard function; since this is constant across all states of the system, we can first instantiate the subordinating events W and then assign them labels. Our strategy here attempts to contruct a dominating hazard function that approximates the actual

hazard function of interest more closely, while still introducing latent thinned events. This offers us the necessary wiggle room to construct an MCMC algorithm for posterior inference; simultaneously, it avoids the inefficiency that results from too many thinned events. It is important to realize that while generating fewer thinned events can 'increase efficiency' by requiring fewer computations per MCMC iteration, it also increases the dependence between the thinned events W and the sMJP trajectory, resulting in slower mixing. We explain this more carefully in the section on inference, but the trade-off here is similar to the choice of the bounding rate Ω for uniformization from chapter 3. Our scheme offers a more refined control over how we can trade off mixing rate and computational cost of each iteration.

It is worth emphasizing that the coupled construction of the point process and the label-assignment process means that the former is not a Poisson process. However, conditioned on the sMJP trajectory, this point process has a simple structure that allows efficient inference. Again, we shall look at this in section 6.4 on inference; first however, we show that algorithm 6.2 is correct.

Proposition 6.1. *The path (V, L, W) returned by algorithm 6.2 corresponds to a sample from the semi-Markov process parametrized by (π_0, D).*

Proof. Without any loss of generality, assume that the system has just entered state $s \in \mathcal{S}$ at time 0. We need to show that the probability density of the system next transitioning to a state s' at time t follows equation (6.12).

Suppose that t is the time of nth candidate jump, so that there were $n-1$ rejected transitions on the interval $[0, t]$. Let these occur at times $(w_1, w_2, \ldots, w_{n-1})$, with $t = w_n$. Recalling that these were generated from the hazard function $U_s(t)$, and letting $[n-1]$ represent the set of integers $\{1, \cdots, n-1\}$, we have:

$$
\begin{aligned}
&P((w_1, \ \ldots w_n), \{v_i = s, l_i = (w_i - w_0)\ \forall i \in [n-1]\}, v_n = s', l_n = 0 | w_0, v_0 = s) \\
&\quad = \left(\prod_{k=1}^{n-1} U_s(l_k) \exp\left(- \int_{l_{k-1}}^{l_k} U_s(\tau) \mathrm{d}\tau \right) \left(1 - \frac{A_s(l_k)}{U_s(l_k)} \right) \right) \qquad (6.19) \\
&\qquad \left(U_s(l_{n-1} + \Delta w_{n-1}) \exp\left(- \int_{l_{n-1}}^{l_{n-1}+\Delta w_{n-1}} U_s(\tau) \mathrm{d}\tau \right) \left(\frac{A_{ss'}(l_{n-1} + \Delta w_{n-1})}{U_s(l_{n-1} + \Delta w_{n-1})} \right) \right) \\
&\quad = \exp\left(- \int_0^{l_{n-1}+\Delta w_{n-1}} U_s(\tau) \mathrm{d}\tau \right) \left(\prod_{k=1}^{n-1} (U_s(l_k) - A_s(l_k)) \right) A_{ss'}(l_{n-1} + \Delta w_{n-1}) \\
&\qquad (6.20)
\end{aligned}
$$

Integrating out w_1 to w_{n-1} (and thus l_1 to l_{n-1}), we have

$$P(w_n = t,\ \{v_i = s\ \forall i \in [n-1]\}, v_n = s', l_n = 0 | w_0 = 0, v_0 = s) \tag{6.21}$$
$$= \exp\left(-\int_0^t U_s(\tau)\mathrm{d}\tau\right) A_{ss'}(w_n) \left(\int_0^t \int_{l_1}^t \cdots \int_{l_{n-2}}^t \prod_{k=1}^{n-1} (U_s(l_k) - A_s(l_k)\mathrm{d}l_k)\right)$$
$$= A_{ss'}(t) \exp\left(-\int_0^t U_s(\tau)\mathrm{d}\tau\right) \frac{1}{(n-1)!} \left(\int_0^t \mathrm{d}\tau\,(U_s(\tau) - A_s(\tau))\right)^{n-1} \tag{6.22}$$

The expression above gives the probability of transitioning from state s to s' after a wait of t time units, with $n-1$ rejected candidate jumps. Summing out $n-1$, we get the transition probability. Thus,

$$P(s_{next} = s', t_{next} = t | s_{curr} = s, t_{curr} = 0)$$
$$= A_{ss'}(t) \exp\left(-\int_0^t U_s(\tau)\mathrm{d}\tau\right) \sum_{n-1=0}^{\infty} \frac{1}{(n-1)!} \left(\int_0^t \mathrm{d}\tau\,(U_s(\tau) - A_s(\tau))\right)^{n-1}$$
$$= A_{ss'}(t) \exp\left(-\int_0^t A_s(\tau)\mathrm{d}\tau\right) \tag{6.23}$$

From equation (6.12), we see that this is the desired result. □

6.4 Posterior inference via MCMC

Our thinning-based construction outlined in the previous section simplifies the structure of the sMJP posterior, and allows us to now define an auxiliary variable Gibbs sampler on the augmented space (V, L, W). As in previous chapters, we alternately resample the thinned events given the current trajectory of the sMJP, and then the sMJP trajectory given the union of the thinned events with transition times of the previous trajectory, setting up a Markov chain over the thinned representation (V, L, W) of the sMJP. We describe both operations in detail below.

6.4.1 Resampling the thinned events given the sMJP trajectory

Let (S, T) be the current sMJP trajectory. We need to resample the thinned events (we called this set $\tilde{W}$) to recover the thinned representation (V, L, W). Note that each thinned event $\tilde{w}_i \in \tilde{W}$ in the interval (t_i, t_{i+1}) has a corresponding label $(\tilde{v}_i, \tilde{l}_i)$ equal to $(s_i, \tilde{w}_i - t_i)$.

To simplify notation, we define the *instantaneous* hazard function $A(t)$, and the instan-

taneous dominating hazard function $U(t)$ as

$$A(t) = A_{\mathbf{S}(t)}(\mathbf{L}(t)) \tag{6.24}$$

$$U(t) = U_{\mathbf{S}(t)}(\mathbf{L}(t)) \tag{6.25}$$

The black and coloured curves in figure 6.2 show these quantities. Observe that the sMJP trajectory completely determines these hazard rates. Loosely speaking, we can view the set of events W as a sample from a Poisson process with intensity $U(t)$, and the actual set of transition times T as a subset of W sampled from a Poisson process with rate $A(t)$ (see figure 6.2). Corollary 2.1 for the thinning theorem then suggests that we can recover the thinned events $\tilde{W}$ by sampling from a Poisson process with intensity $(U(t) - A(t))$. The following proposition shows that this is indeed the case.

Proposition 6.2. *Conditioned on a trajectory (S, T) of the sMJP, the thinned events $\tilde{W}$ are distributed as a Poisson process with intensity $U(t) - A(t)$.*

Proof. We will consider the interval of time $[t_i, t_{i+1}]$, so that the sMJP entered state s_i at time t_i, and remained there until time t_{i+1}, when it transitioned to state s_{i+1}. Exploiting the independence properties of the sMJP and the Poisson process, we only need to consider resampling thinned events on this interval. Call this set of thinned events $\tilde{W} \equiv \{\tilde{w}_1, \cdots, \tilde{w}_{n-1}\} \in [t_i, t_{i+1}]$, and call the corresponding set of labels $\tilde{V} \equiv \{\tilde{v}_1, \cdots, \tilde{v}_{n-1}\}$ and $\tilde{L} \equiv \{\tilde{l}_1, \cdots, \tilde{l}_{n-1}\}$ (to avoid notational clutter, we do not indicate that $\tilde{W}$ and $\tilde{L}$ are actually restrictions to $[t_i, t_{i+1}]$). Observe that each element of $\tilde{v}_j \in \tilde{V}$ equals s_i, while each element $\tilde{l}_j \in \tilde{L}$ equals $\tilde{w}_j - t_i$. We write this as $\tilde{V} = s_i$ and $\tilde{L} = \tilde{W} - t_i$. Then, by Bayes rule, with equations (6.19) and (6.15) as the joint and marginal, we have

$$\begin{aligned}
&P(\tilde{W}, \tilde{V} = s_i, \tilde{L} = \tilde{W} - t_i | s_i, t_i, s_{i+1}, t_{i+1}) \qquad (6.26)\\
&= \frac{P(\tilde{W}, \tilde{V} = s_i, \tilde{L} = \tilde{W} - t_i, v_n = s_{i+1}, w_n = t_{i+1}, l_n = 0 | v_0 = s_i, w_0 = t_i, l_0 = 0)}{P(s_{i+1}, t_{i+1} | s_i, t_i)} \\
&= \frac{\exp\left(-\int_{t_i}^{t_{i+1}} U(\tau) \mathrm{d}\tau\right) \left(\prod_{k=1}^{n-1} (U(\tilde{w}_k) - A(\tilde{w}_k))\right) A_{s_i s_{i+1}}(t_{i+1} - t_i)}{A_{s_i s_{i+1}}(t_{i+1} - t_i) \exp\left(-\int_{t_i}^{t_{i+1}} A(\tau) \mathrm{d}\tau\right)} \\
&= \exp\left(-\int_{t_i}^{t_{i+1}} U(\tau) - A(\tau) \mathrm{d}\tau\right) \left(\prod_{k=1}^{n-1} (U(v_k) - A(v_k))\right)
\end{aligned}$$

This is just the density of a Poisson process on (t_i, t_{i+1}) with intensity $(U(t) - A(t))$, which is what we set out to prove. □

Observe that this step is independent of any observations. Sampling from the Poisson process is relatively straightforward by choosing the bounding rates U_i appropriately; we provide a concrete example in section 6.5. The hazard functions $A(t)$ and $U(t)$ remain unchanged at the end of this step.

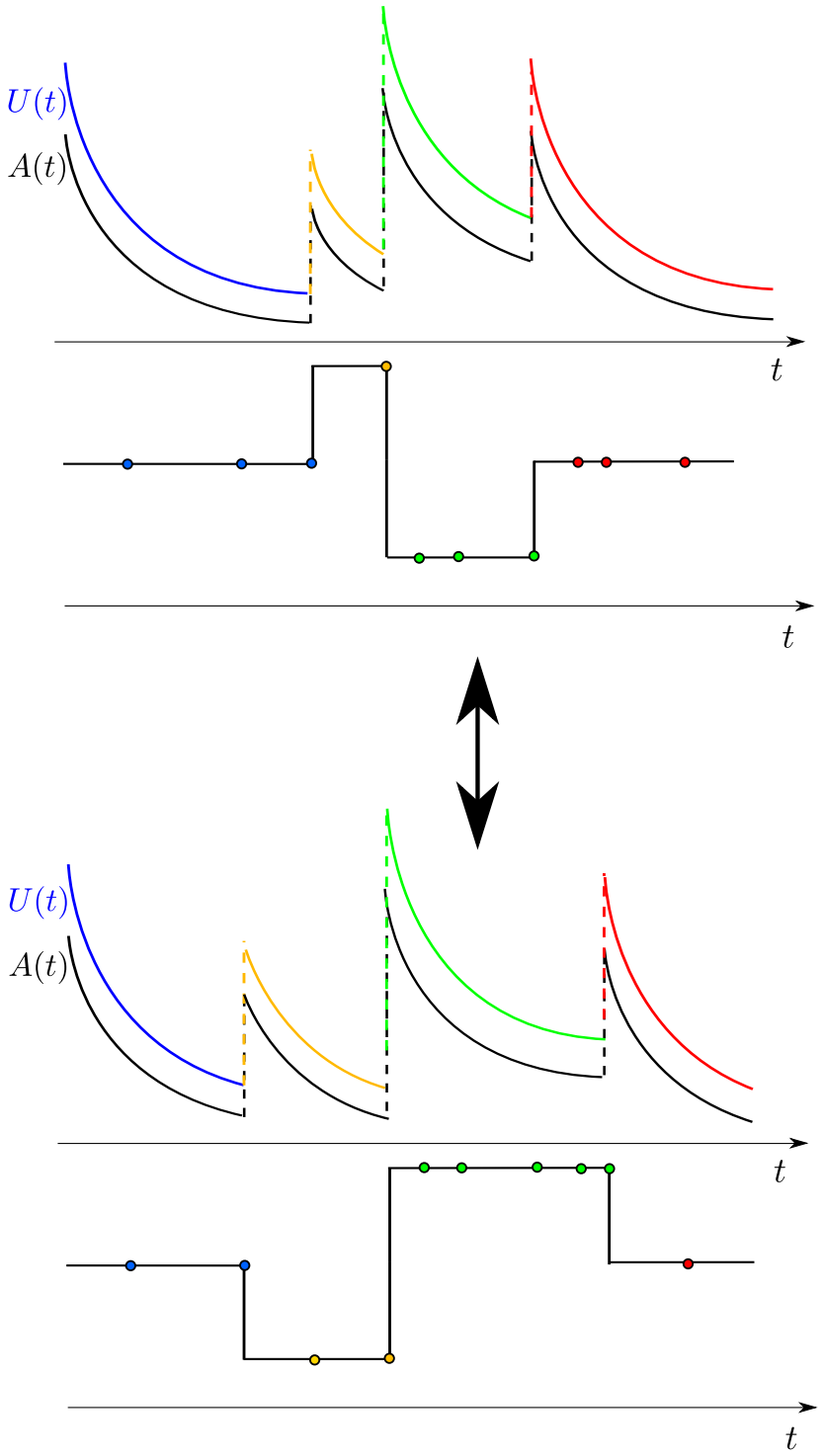

Figure 6.4: Resampling the sMJP trajectory: observe that a new trajectory results in a new bounding rate $U(t)$, and we need to account for the probability of the Poisson events W under this rate function.

6.4.2 Resampling the sMJP trajectory given the set of times W

This step is a bit more subtle than with the MJP. Like chapter 3, we want to assign each element $w_i \in W$ a label (v_i, l_i), by running the forward filtering backward sampling

algorithm over the set of times in W. Observe however, that l_i can take values in the set $\{0, w_i - w_{i-1}, \cdots, w_i - w_0\}$, so that the dimensionality of the state space at step i is i (and thus increases with $|W|$). Consequently, the forward-backward algorithm for the sMJP is more expensive than for the MJP. The usual N^2C scaling of the forward-backward algorithm (N being the number of states and C being the length of the chain) would suggest a computational cost that scales cubically with $|W|$. Note, however, that l_i can only equal 0 or $l_{i-1} + \Delta w_{i-1}$. This sparsity in the possible state transitions results only in a quadratic scaling (at worst).

Next, observe from figure 6.4 that changing the sMJP trajectory results in a change in the instantaneous hazard functions $A(t)$ and $U(t)$. This is a consequence of the fact that unlike uniformization, candidate jump times are now drawn from a point process whose intensity depends on the sMJP trajectory. A new trajectory results in a new hazard function, and we need to account for the probability of the events in W under this new hazard function. It is however straightforward to adapt the forward-backward sampling algorithm to make this correction; we effectively treat the elements of W as additional 'observations' of the state of the Markov chain. During the forward filtering pass, as we calculate the probability of being in a particular state (v_i, l_i) over an interval (w_i, w_{i+1}), we also include the probability of waiting for a time $\Delta w_i = (w_{i+1} - w_i)$ until the next event under the resulting hazard function $U_{v_i}(\tau + l_i)$. Write this probability as $P(w_{i+1}|w_i, v_i, l_i)$, it is given by

$$P(w_{i+1}|w_i, v_i, l_i) = U_{v_i}(l_i + \Delta w_i) \exp\left(-\int_{l_i}^{(l_i+\Delta w_i)} U_{v_i}(\tau)\mathrm{d}\tau\right) \tag{6.27}$$

When running the forward-backward algorithm, we must also include this term in our calculations.

Figure 6.3 provides a graphical demonstration of the overall discrete-time system we have to solve. It includes observations X of the sMJP state, with x_i representing all observations in the interval $(w_i,\ w_{i+1})$. Let $P(x_i|v_i)$ be the corresponding likelihood function. Then, the joint distribution over the entire set of variables factorizes as:

$$P(V, L, W, X) = P(v_0, l_0, w_0) \prod_{i=0}^{|W|-1} P(x_i|v_i)P(w_{i+1}|v_i, l_i)P(v_{i+1}, l_{i+1}|v_i, l_i, \Delta w_i) \tag{6.28}$$

Observe also that $w_{|W|} = t_{end}$ does not correspond to a real event, rather it is the end of the observation interval. Consequently, while for $i < (|W| - 1)$, $P(w_{i+1}|w_l, v_i, l_i)$ is given by equation (6.28), we also have that

$$P(w_{|W|}|w_{|W|-1}, v_{|W|-1}, l_{|W|-1}) = \exp\left(-\int_{(l_{|W|-1})}^{(l_{|W|-1}+\Delta w_{|W|-1})} U_{v_{|W|-1}}(\tau)\mathrm{d}\tau\right) \tag{6.29}$$

The forward-filtering stage moves sequentially through the times in W, successively

calculating the probabilities $P(v_i, l_i, w_{1:i+1}, x_{1:i})$ using the recursion:

$$P(v_i, l_i, w_{1:i+1}, x_{1:i}) = P(x_i|v_i)P(w_{i+1}|v_i, l_i) \sum_{v_{i-1}, l_{i-1}} P(v_i, l_i|v_{i-1}, l_{i-1}, \Delta w_i)P(v_{i-1}, l_{i-1}, w_{1:i}, x_{1:i-1}) \tag{6.30}$$

The transition probabilities $P(v_i, l_i|v_{i-1}, l_{i-1})$ are given in equations (6.15) and (6.17), with the probabilities of all other state transitions equal to 0. In the summation above, v_i and v_{i-1} take values in $\mathcal{S}$. Additionally, l_i either equals 0 or $l_{i-1} + \Delta w_{i-1}$, while l_{i-1} takes values in $\{0, w_{i-1} - w_{i-2}, \cdots, w_{i-1} - w_0\}$. Thus, the ith step of the forward filtering stage scales as $O(N^2 i)$. Since there are $|W|$ such updates, the overall iteration of the MCMC sampler scales as $O(N^2|W|^2)$.

6.5 Calculations for an sMJP with Weibull hazards

In this section, we work through the details of a particular sMJP that we will later use in our experiments. Consider the Weibull density with shape parameter α and scale parameter λ; this has the form

$$r(\tau|\alpha, \lambda) = \begin{cases} \frac{\alpha}{\lambda}\left(\frac{\tau}{\lambda}\right)^{\alpha-1} \exp\left(-(\tau/\lambda)^\alpha\right) & \tau \geq 0 \\ 0 & \tau < 0 \end{cases} \tag{6.31}$$

Straightforward calculation shows that the cumulative distribution function is given by:

$$D(\tau|\alpha, \lambda) = 1 - \exp\left(-(\tau/\lambda)^\alpha\right) \tag{6.32}$$

and the hazard function $A(\tau|\alpha)$ is given by:

$$A(\tau|\alpha, \lambda) = \frac{\alpha}{\lambda}\left(\frac{\tau}{\lambda}\right)^{\alpha-1} \tag{6.33}$$

The shape parameter α controls the 'burstiness', with $\alpha > 1$ giving a refractory distribution, and values less than 1 giving burstiness or underdispersion ($\alpha = 1$ recovers the exponential distribution). Figure 6.5 plots the logarithm of these hazard functions for $\alpha = 2$ and 0.7. Note that for $\alpha < 1$, the hazard function is unbounded, tending to infinity as $\tau \to 0$. Additionally, unlike the gamma hazard which plateaus as $\tau \to \infty$ when $\alpha > 1$, the Weibull hazard is also unbounded for the refractory case, tending to infinity as τ tends to infinity. This is less of an issue since we always deal with sMJP trajectories over finite intervals. Given the closed form of the distribution function (equation (6.32)), we can easily draw various conditional samples from the Weibull distribution. This is particularly useful when we must sample from $r(\tau|\alpha, \lambda)$ conditioned

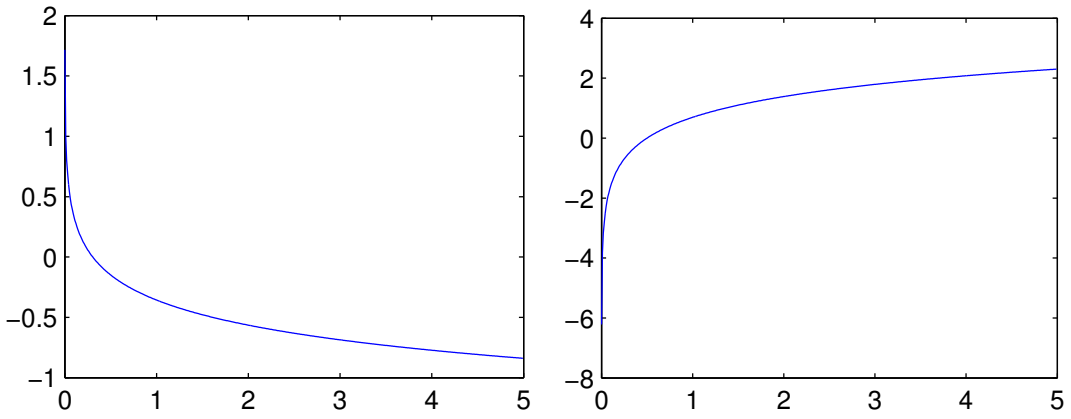

Figure 6.5: Log-hazard functions for the Weibull distribution with $\alpha = 0.7$ (left) and $\alpha = 2$ (right). In both cases, $\lambda = 1$.

on τ being larger than some minimum value τ_{min}.

Now, consider a semi-Markov process with Weibull hazard functions. For each pair of states s and s', the corresponding rate funtion $A_{ss'}(\cdot)$ takes the form given in equation (6.33) with its own pair of parameters $(\alpha_{ss'}, \lambda_{ss'})$.

For the Weibull hazard $A(\cdot)$, for some $\Omega > 1$, we use the following simple upper bound $U(\tau)$:

$$U(\tau) = \Omega A(\tau|\alpha, \lambda) = \frac{\Omega\alpha}{\lambda}\left(\frac{\tau}{\lambda}\right)^{\alpha-1} = \frac{\alpha}{\tilde{\lambda}}\left(\frac{\tau}{\tilde{\lambda}}\right)^{\alpha-1} \tag{6.34}$$

Here, $\tilde{\lambda} = \lambda/\sqrt[\alpha]{\Omega}$. Thus sampling from the distribution corresponding to this dominating hazard function reduces to straightforward sampling from a Weibell with a smaller scale parameter $\tilde{\lambda}$. Note that with this choice of $U(\cdot)$, each candidate event is rejected with probability $1 - \frac{1}{\Omega}$; this can be a guide to choosing Ω. In our experiments, we set Ω equal to 2. More generally, we can assign a different Ω_s to each state s. This allows the dominating point processes across different states to resemble each other, and makes resampling a new trajectory less sensitive to the current set of events W (recall that we sample the new trajectory conditioned on W).

We now provide some details of the inference algorithm for the sMJP with Weibull holding times.

1. **Resampling the rejected events $\tilde{W}$ given the semi-MJP trajectory**: Consider the interval (t_i, t_{i+1}), with the sMJP entering state s_i at t_i, and remaining there until t_{i+1}. Resampling the thinned events $\tilde{W}$ over this interval involves sampling from a Poisson process with intensity $(U(t) - A(t)) = (\Omega - 1)A(t) = (\Omega - 1)\sum_{s'} A_{s_i s'}(t - t_i)$. By the superposition theorem (theorem 2.4), we can instantiate this by sampling N independent Poisson processes on the interval $(0, t_{i+1} - t_i)$. The nth has intensity $(\Omega - 1)A_{s_i n}(\cdot) \equiv \hat{A}_{s_i n}(\cdot)$, where, like before, $\hat{A}_{s_i n}(\cdot)$ is a Weibull hazard function obtained by an $(\Omega - 1)^{1/\alpha_{s_i n}}$ correction to

the scale factor of $A_{s_i n}(\cdot)$. A simple way to sample such a Poisson process is by first drawing the number of events from a Poisson distribution with mean $\int_0^{(t_{i+1}-t_i)} \hat{A}_{s_i n}(u)\mathrm{d}u$, and then drawing that many events i.i.d. from the Weibull $\hat{A}_{s_i n}$, conditioning on them being less than $t_{i+1} - t_i$. Call this sequence $\tilde{T}_n$, and define $\tilde{T} = \cup_{n \in \mathcal{S}} \tilde{T}_n$. Then $W \equiv \tilde{T} + t_i$ is the set of resampled thinned events on the interval (t_i, t_{i+1}). From the independence property of the Poisson process, we can repeat this over each segment of the sMJP path.

2. **Resampling the semi-MJP trajectory given the set of times W:**
This just involves running the forward-backward algorithm on a discrete-time Markov chain as outlined in the previous section. For the Weibull distribution, we have that

$$\exp\left(-\int_{t_i}^{t_{i+1}} U_{s_i}(\tau)d\tau\right) = \prod_{s'} \exp\left(-\int_{t_i}^{t_{i+1}} U_{s_i s'}(\tau)d\tau\right) \tag{6.35}$$

$$= \prod_{s'} \exp\left(-\Omega \int_{t_i}^{t_{i+1}} \frac{\alpha_{s_i s'}}{\lambda_{s_i s'}} \left(\frac{\tau}{\lambda_{s_i s'}}\right)^{\alpha_{s_i s'}-1} d\tau\right) \tag{6.36}$$

$$= \prod_{s'} \exp\left(\Omega\left(\frac{t_i}{\lambda_{s_i s'}}\right)^{\alpha_{s_i s'}} - \Omega\left(\frac{t_{i+1}}{\lambda_{s_i s'}}\right)^{\alpha_{s_i s'}}\right) \tag{6.37}$$

Using this (as well as the equations provided at the beginning of this section), we can easily compute all probabilities for the forward filtering (equation (6.30)) as well as the backward sampling stages of the forward-backward algorithm.

6.6 Experiments

In this section, we evaluate our sampler on the Weibull sMJP described in the previous section. In all experiments, the number of states was set to 3. The shape parameters for each hazard function ($\alpha_{ss'}$) were uniformly distributed on the interval $[0, 3]$, so that a transition from a state s to a state s' was bursty with probability $1/3$ (recall that uniformization cannot handle this situation). The scale parameter was always set to 1.

We compared the performance of our sampler with a particle MCMC sampler (Andrieu et al., 2010); in particular, we implemented the particle independent Metropolis-Hastings sampler described in that paper. Let P be the number of particles; in any iteration, at any time t, each of these P particles represents a trajectory from the beginning of the observation interval until t. Let $\mathbf{S}^p(t)$ represent the state of particle p at time t. We then propagate each of these trajectories via algorithm 6.1 to the time $t_o > t$ of the next observation x_o. At this time, we assign particle p a weight w_o^p equal to $P(x_o|S^p(t_o))$, the likelihood of its current state under the observation x_o. Define Z_o as the sum of the particle weights, i.e. $Z_o = \sum_{p=1}^{P} w_o^p$, and resample P particles (with replacement) with probability equal to the weights normalized by Z_o. We repeat

this procedure, traversing all observations until we reach the end of the observation interval. At this time, we once again assign each particle a weight and now pick 1 of the P particles proportional to its weight (if there are no observations at the endtime, we pick one of the particles uniformly at random). The trajectory of this particle serves as our Metropolis-Hastings proposal, call this $\mathbf{S}^{new}(\cdot)$. Associated with this trajectory is a weight Z^{new} equal to the product of all the weights it encountered as it traversed the observations i.e. $Z^{new} = \prod_{o=1}^{|O|} Z_o$ ($|O|$ being the number of observations). Let $\mathbf{S}^{old}(\cdot)$ be the MCMC trajectory at the current iteration of the Markov chain, and let Z^{old} be its associated weight. Then, we set the trajectory for the next iteration of the Markov chain equal to $\mathbf{S}^{new}(\cdot)$ with probability $\min(1, Z^{new}/Z^{old})$, otherwise leaving it at $\mathbf{S}^{old}(\cdot)$. We refer the reader to Andrieu et al. (2010) for more details. We tried various values for the number of particles P; for our problems, 10 seemed to produce the most effective samples per unit time.

We implemented both our thinning-based sampler and the particle MCMC sampler in Matlab. We observed that low settings of α resulted in numerical errors with our implementations of both samplers, and to avoid such issues, any α less that 0.6 was thresholded to 0.6. All experiments averaged results across multiple runs with different random parameter settings.

6.6.1 Effect of the observations

Our first experiment compared the performance of both samplers as the effect of the observations became stronger and stronger. We set the number of observations to 10, distributing these over an interval of length 25. Each observation had an associated likelihood term that favoured a particular, random state (the 'true' state) over the other two states. We set the ratio of the likelihood of the true state to the likelihood of any other state, $p(x_i|s^{true})/p(x_i|s^{other})$, to be 100. We then associated an 'inverse-temperature' parameter inv with the likelihood term $P(x_i|s)$, so that the effective likelihood at the ith observation was $(P(x_i|s))^{inv}$. As this parameter varied from 0 to 1, the problem moved from sampling from the prior (where the observations were irrelevant) to a situation where the trajectory was observed (almost) perfectly at 10 random times.

As described in the previous section, our MCMC sampler was set up with the dominating hazard rate at any instant equal to twice the true hazard rate (i.e. $\Omega = 2$ and $U_{ij}(t) = 2A_{ij}(t)$), giving a probability of thinning equal to 0.5. After each simulation, we calculated the empirical distribution of the time spent in each state as well as the number of state transitions, and then (as in chapter 3), used R-coda (Plummer et al., 2006) to estimate effective sample sizes for these quantities. The effective sample size of the simulation was then set as the median of the effective sample sizes of all these statistcs.

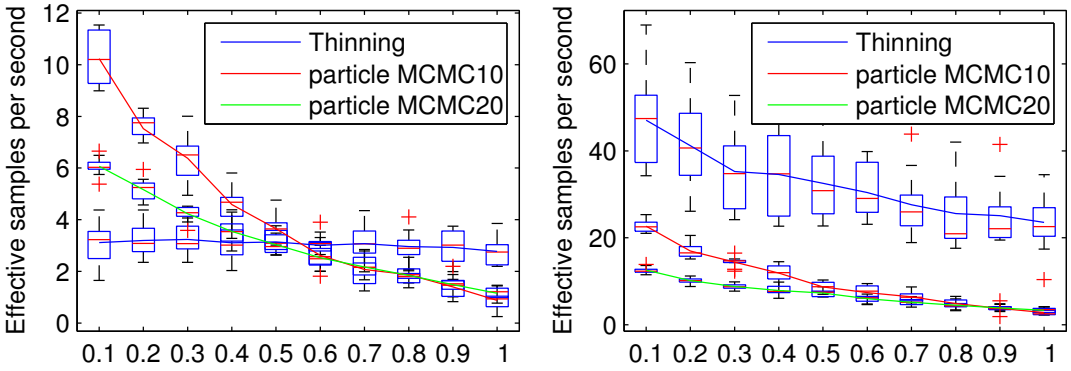

Figure 6.6: The number of effective samples produced per unit time vs the inverse-temperature of the likelihood, when the trajectories are over an interval of length 25 (left) and 2 (right). We compare our sampler with a 10 and 20 particle MCMC sampler.

The left plot in figure 6.6 shows the effective number of samples produced per unit time by both samplers as the inverse-temperature is increased, averaging results from 10 random parametrizations of the sMJP. We see (as one might expect), that when the effect of the observations is weak, particle MCMC (which uses the prior distribution to make local proposals), outperforms our thinning-based sampler. Particle MCMC also has the benefit of being simpler implementation-wise, and is about 2-3 times faster (in terms of raw computation time) for a Weibull sMJP, than our sampler. However as the effect of the likelihood increases, the number accepted proposals starts to decreases, and particle MCMC started to have more and more difficulty hitting the data. In contrast, we see that our sampler is fairly insensitive to the effect of the likelihood, eventually outperforming the particle MCMC sampler. While there exist techniques to generate more data-driven proposals for the particle MCMC (Andrieu et al., 2010; Golightly and Wilkinson, 2011), these compromise the appealing simplicity of the original particle MCMC sampler. Moreover, none of these really have the ability to propagate information back from the future (unlike our forward-backward algorithm), rather they make more and more local moves (for instance, by updating the sMJP trajectory on smaller and smaller subsets of the observation interval).

Varying the strength of the observations is one way to study how the two samplers handle problems where the posterior deviates from the prior. A second approach is to distribute the same number of observations over intervals of decreasing length. We set each observation to favour a random state of the sMJP, so that distributing these observations over shorter and shorter intervals demands an sMJP trajectory that switches states more and more rapidly. The right plot in figure 6.6 plots the same quantities described in the previous paragraph, now with the observation interval set to a smaller length of 2. Here, the benefit of our sampler is even more pronounced, with the forward-backward step in our algorithm handling the effect of the observations without any difficulty. An additional benefit is that over short intervals, the quadratic cost of

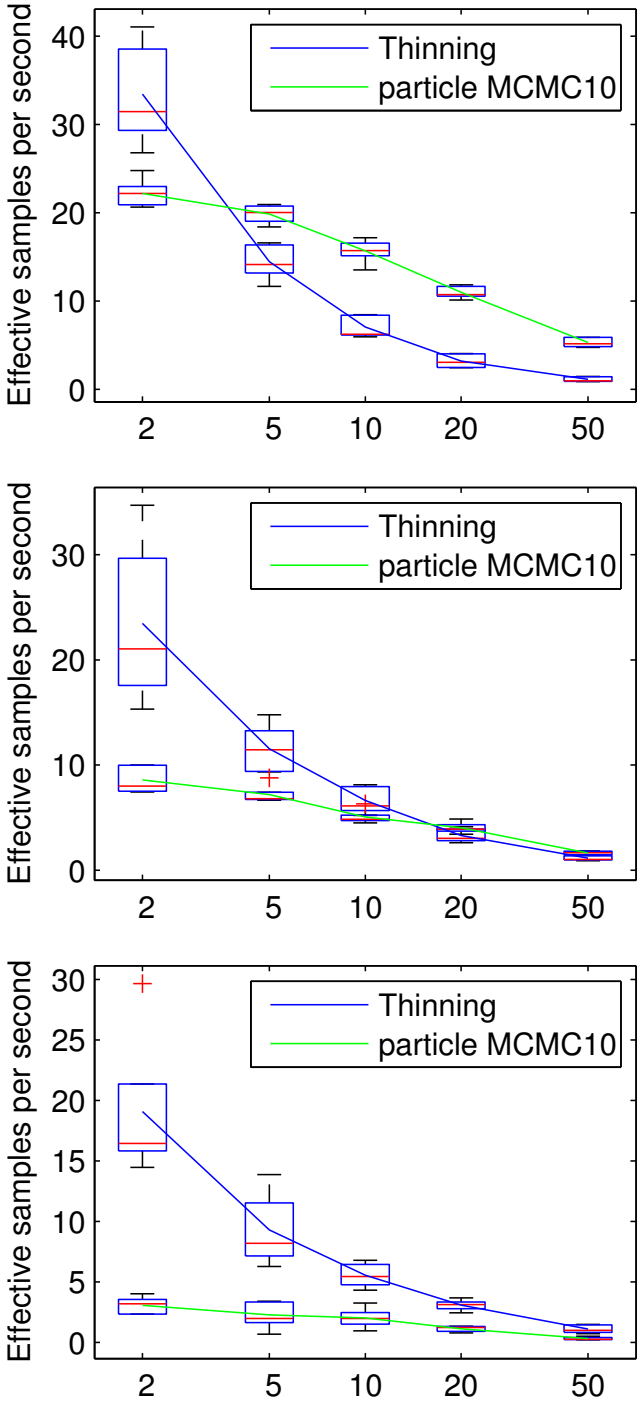

Figure 6.7: The number of effective samples produced per unit time vs interval length, for low, medium and high inverse-temperature settings.

the forward-backward step of our algorithm is much less pronounced.

6.6.2 Effect of the observation interval length

The next experiment studies more carefully the performance of the two samplers as the timescales required to explain the observations deviate from the prior. For three setting of the inverse temperature parameter (0.1, 0.5 and 0.9), we calculated the number of effective samples produced per unit time over intervals whose lengths increased from 2 to 50. Once again, we averaged results from 10 random settings of the sMJP parameters. Figure 6.7 shows the results for the low, medium and high settings of the the inverse temperature. As expected, our sampler performs best in the low temperature and short interval regimes where the posterior deviates from the prior. Additionally, for short observation intervals, both samplers took similar times to run the same number of MCMC iterations.

We point out here that a combination of shorter computation time per iteration along with the ability to propagate back information from future observations does not necessarily imply larger effective sample sizes per unit time. Even in the absence of observations, samples produced by our algorithm are correlated because of the dependence between the sMJP trajectory and the candidate jump times W. Consider for instance two states s and s' with widely different leaving rates A_s and $A_{s'}$. Under our construction here, the bounding rates U_s and $U_{s'}$ are multiples of A_s and $A_{s'}$, and thus will also differ significantly. Recall now that when we resample a trajectory, we need to account for the probability of the events W under the resulting hazard function (see figure 6.4). Thus, if the sMJP were in state s over an interval of time, it is unlikely to be in state s' over the same interval in the new sample. In general, our scheme is unlikely to produce a new trajectory whose hazard function $U^{new}(\cdot)$ differs too much from that of the old trajectory, $U^{old}(\cdot)$. Such a dependence did not arise in the case of uniformization, where U^{new} and U^{old} were constant and equal. In that case, the only dependence between two successive samples was a shared subset of candidate transition times. By contrast, in the absence of observations, successive samples produced by the particle MCMC sampler are independent.

6.7 Linearizing inference in the length of W

In this section, we return to the issue of the cost of inference scaling quadratically with the number of candidate jump times $|W|$. Recall that this is a consequence of the fact that for a general sMJP, the future behaviour depends not just on the current state s_i, but also on the duration for which the sMJP has been in that state. Thus, when running the forward-backward algorithm, at stage i, we need to explicitly represent probabilities corresponding to all possible values of l_i, the time since the sMJP entered state s_i. For sMJPs observed over long intervals, the set W can be large, making quadratic scaling intolerably expensive.

There are a number of approaches to addressing this issue for discrete-time sMJPs, and we can adapt these to the forward-backward sampling stage of our MCMC algorithm. One simple approach is break the observation interval into a number of smaller segments, and sequentially update the trajectory in each segment conditioned on the rest of the trajectory. An alternative is to follow a slice-sampling approach (Dewar et al., 2012). Here, at the ith stage of the forward-backward algorithm, rather than allowing l_i to take all values $\{0, w_i - w_{i-1}, \cdots, w_i - w_0\}$, we restrict its range from $\{0, \cdots, w_i - w_{c_i}\}$ for some random slice variable $c_i \in \{0, \cdots, i\}$. By allowing s_i to vary from one iteration to the next, we can construct an exact MCMC chain, while taking advantage of the fact that very long holding times are atypical.

The quadratic scaling of inference for sMJPs is a worst case result, and there exist hazard rates which still allow efficient inference over long observation intervals. Clearly, the MJP, with constant transition rates is one such example. In general, if the hazard function has a constant tail (so that the corresponding waiting-time density has an exponential tail), then the system has only a finite window of memory. Thus, suppose that for some time τ_{win}, the hazard function has the form:

$$\begin{aligned} A(\tau) &= \tilde{A}(\tau) \quad \tau \leq \tau_{win} \\ &= A_{tail} \quad \tau > \tau_{win} \end{aligned} \tag{6.38}$$

Then at time t during the forward filtering state, we need to account only for those values of l_n that range from 0 to τ_{win}; all larger values of l_n can be summarized by a single state. Consequently, the cost of the resulting forward-backward dynamic programming algorithm now scales quadratically with the length of this memory window τ, and only linearly with the total observation interval.

Even if the actual hazard function is not of this form, we can approximate it with such a hazard function, and use the sampled paths as Metropolis-Hastings proposals for samples from the original system. Such an approximation would be suitable for the gamma hazard functions from chapter 5 for example; recall that these plateau out to a constant value as $\tau \to \infty$.

6.8 Discussion

In this chapter, we described a general framework for MCMC inference in semi-Markov processes. Our scheme is based on a procedure of dependent thinning that generalizes uniformization. Given the state of the system at any instant, we define a hazard function that dominates the true hazard function. Our scheme then proceeds by sequentially sampling the time of the next candidate event given this function, and then updating the state of the system at this time. Our scheme now allows us to perform MCMC inference by alternately sampling thinned events given the current trajectory,

and then a new trajectory given all candidate event times. At a high level, the first step can be viewed as sampling a random discretization of time. We showed how this can be done relatively easily exploiting properties of the Poisson process. Given this discretization of time, we can leverage available discrete-time MCMC schemes to update the sMJP trajectory. In general, it is straightforward to extend our approach here to piecewise-constant stochastic processes with a more complicated dependence on the past.

There are a number of possible avenues for further study. In our experiments in this chapter, we set the dominating rates to be twice the true event rates at any time. While this is convenient, it can result in poor mixing because of the requirement that the new and old instantaneous hazard functions resemble each other. Recall that for uniformization, the dominating hazard function was a constant Ω independent of the state of the system. By constructing more complicated dominating functions, it is possible to approximate such a constant rate, while avoiding the need for a very high dominating rate that results from say, rare events with very high leaving rates. Such an approach allows us to trade off computational costs and mixing rates more carefully. The reason we needed the old and new hazard functions to resemble each other was because they both had to explain the same set of candidate transition times W. We can thus also consider schemes that propose a new set of candidate times W_{new}, allowing more global moves.

We saw that for general sMJPs, inference scales quadratically with W, the number of candidate jump times. We discussed a number of possible approaches to dealing with this problem in section 6.7, it is worth studying them in further detail. In our experiments, we studied sMJPs with fixed parameters. Like subsection 3.5.2, it is possible to take a fully Bayesian approach, placing priors on these parameters as well. For instance, Berger and Sun (1993) discuss parameter inference for the Weibull distribution. With such a Bayesian approach, care needs to be taken with state-dependent bounding functions that attempt to approximate uniformization, since this will have to adapt to the varying parameter values.

Chapter 7

MJPs with unbounded rates

7.1 Introduction

Armed with ideas from the previous chapter, we return back to the problem of MCMC inference for Markov jump processes. We consider two limitations of the uniformization-based approach described in chapter 3: the need to truncate the state-space of systems with unbounded event rates, and the inefficiency resulting from using a single bounding rate for systems with combinations of very stable and very unstable states. Our approach based on dependent thinning from chapter 6 is directly applicable to such systems, and allows us to construct MCMC algorithms with fewer thinned events than the uniformization-based sampler. In this chapter, we provide an alternate (but equivalent) description of the approach outlined in chapter 6; this is based on a construction of the MJP from a family of Poisson processes. Besides helping us understand our algorithm better, this can also lead to extensions which can improve the performance of the sampler. As an application, we will consider a model from queuing theory, the M/M/c/c queue.

7.2 Dependent thinning for MJPs

Consider a Markov jump process with state-space $\mathcal{S}$. We allow the cardinality of $\mathcal{S}$ to be countably infinite. We will follow the notation of the previous chapter (rather than chapter 3), so that $A_{ss'}$ gives the rate of transitioning from state s to s' for all s and s'. In particular, since we are dealing with MJPs, $A_{ss} = 0\ \forall s$. For any state $s \in \mathcal{S}$, the leaving rate is a constant $A_s = \sum_{s' \in \mathcal{S}} A_{ss'}$. We require A_s to be finite for each s, however unlike chapter 3, we will not require a finite constant $\Omega > A_s, \forall s$. Upon leaving state s, the probability of transitioning to state s' is $p_s(s') \propto A_{ss'}$, with $p_s(s) = 0$. Given an initial distribution over states π_0, Gillespie's algorithm (algorithm 3.1) provides a simple and direct way to sample a trajectory of this system over an interval $[t_{start}, t_{end}]$.

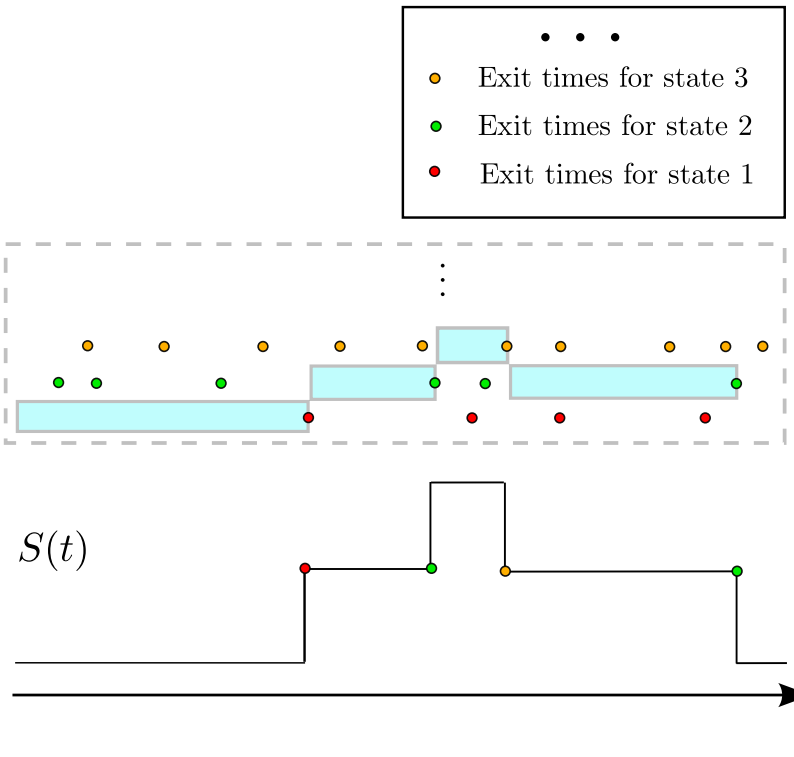

Figure 7.1: Gillespie's algorithm for MJPs (with auxiliary Poisson events)

In chapter 6, we described a thinning-based alternative to Gillespie's algorithm. For each A_s, define a constant $U_s > A_s$. U_s gives the rate at which *candidate* leaving times are generated in state s, so that the time until the next candidate time is sampled from an exponential with rate U_s. We reject each such event with probability $\left(1 - \frac{A_s}{U_s}\right)$, otherwise we transition to state s' with probability $p_s(s')$. Note that since we are dealing with a Markov system, we do not need to represent the duration for which the system has been in its current state. This makes the entire procedure considerably simpler that that for general semi-Markov processes (chapter 6).

The Markov structure of the problem allows us to define an equivalent construction in terms of a family of Poisson processes, one for each state. This is demonstrated in Figure 7.1. For each state s, sample a realization of a rate A_s Poisson process on $[t_{start}, t_{end}]$. Assign all events of this process the label s. Now, to sample a trajectory, assign the MJP an initial state drawn from the prior π_0. Suppose we pick state s_0, then the MJP remains in this state until the first event labelled s_0. By the memoryless property of the Poisson process, this waiting time is exponentially distributed with rate A_{s_0}, as required by the definition of the MJP. At this time, the MJP moves to a random new state s_1, with $p_{s_0}(s_1) \propto A_{s_0 s_1}$. Repeat the procedure until the end of the interval. Clearly, this procedure is equivalent to to Gillespie's algorithm. The cyan-shaded region of figure 7.1 then defines the MJP trajectory (S, T). The times of all events in this regions define T, while their corresponding labels define S. By the independence property of the Poisson process, everything outside this region is

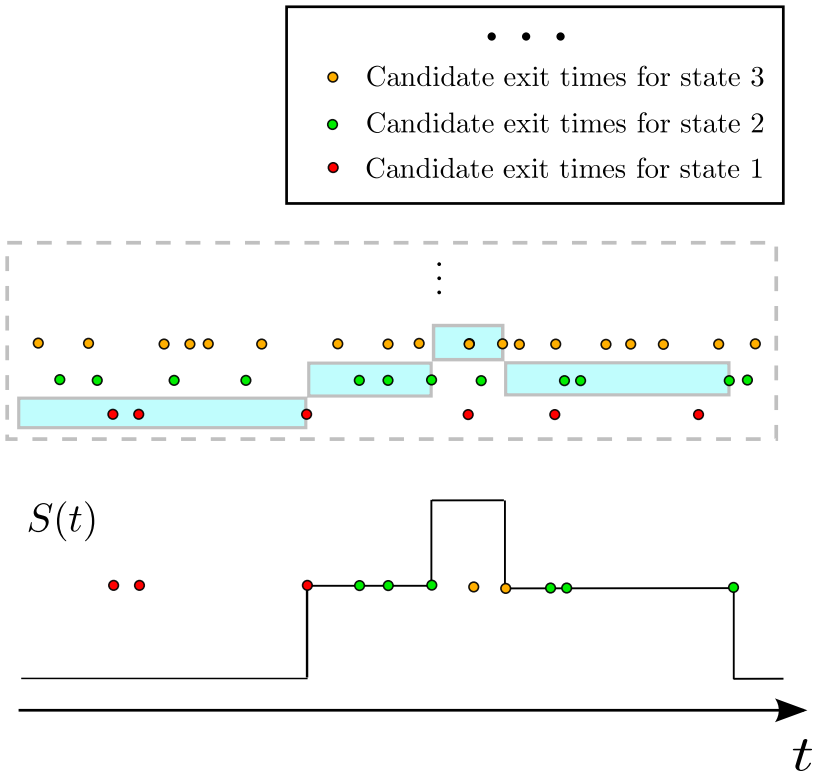

Figure 7.2: Thinning based construction for MJPs (with auxiliary Poisson events)

irrelevant.

We can now introduce auxiliary thinned variables by an obvious application of the thinning theorem. For each state s, sample events from a Poisson process, now with the rate $U_s > A_s$ on $[t_{start}, t_{end}]$. To sample a trajectory, once again assign the MJP an initial state s_0 drawn from π_0. Once again, the MJP remains in this state until the first event labelled s_0, however now it changes state only with probability A_{s_0}/U_{s_0}. If it does decide to change state, it picks a new state s_1 with probability proportional to $A_{s_0 s_1}$. Again, repeat the procedure until the end of the interval. It is clear that this procedure is equivalent to the dependent thinning scheme outlined earlier (and in chapter 6). Figure 7.2 demonstrates this graphically; once again, the events inside the cyan region define (V, W) the thinning representation of the MJP. Like chapter 3, we no longer need the set of waiting times L, since the original system is now Markov without self-transitions.

It is now easy to understand the MCMC sampler of the previous chapter. Recall that this proceeded by alternately resampling the thinned events given the MJP trajectory, and then the trajectory given the set of candidate transition times. Knowing the MJP trajectory amounts to knowing the the cyan region in figure 7.2 (as well as the events at the right edges corresponding to actual transitions). Resampling the thinned events

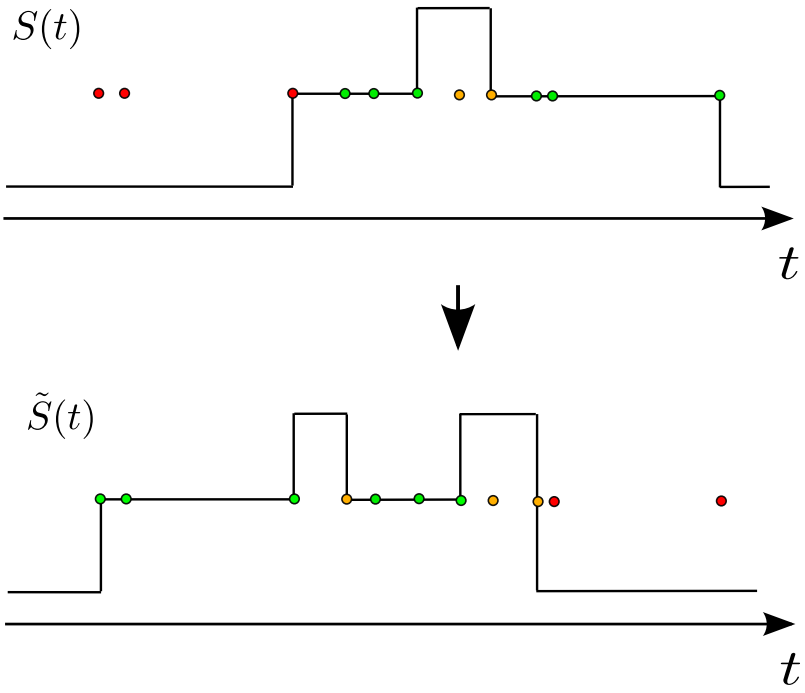

Figure 7.3: Resampling the MJP trajectory

in the interior of the cyan region is now a simple application of the corollary to the thinning theorem (corollary 2.1): when in a region corresponding to state s, sample from a Poisson process with constant intensity $(U_s - A_s)$. Proposition 6.2 shows that this is correct.

Resampling a new trajectory given the set of candidate times involves discarding the old labels of the Poisson events in the cyan region, and relabelling the events using the forward-backward algorithm. This is shown in figure 7.3. Note that we need to account for the probability of the new labels assigned to the Poisson events; we saw in the previous chapter how we to adapt the forward-backward algorithm to do so.

By assigning labels to candidate jump times, we associate each segment of the MJP trajectory with a window of events from the Poisson process labelled with the corresponding state. Since each of these Poisson processes has a finite rate, we will have only a finite number of candidate state transitions over any finite interval. Even if the maximum event rate in the system is unbounded, any realization of the system trajectory will have a finite maximum rate.

By contrast, uniformization involves constructing a MJP from a single subordinating Poisson process. In order to avoid assigning labels to these Poisson events, we need its rate to dominate all event rates in the system, something which is not always possible. However, since the Poisson process is independent of the system trajectory, there is a smaller dependence across samples. Note though, that our new algorithm requires only a slight modification of the uniformization-based sampler. It samples the thinned events

from a slightly different Poisson process, and has a single additional term $P(\Delta w_i|v_i)$ in the forward-backward algorithm.

7.3 The M/M/c/c queue

In this section, we apply our ideas to a simple MJP, the M/M/c/c queue. In queuing theory (Kendall, 1953), an M/M/c/k queue is a system consisting of c 'servers' and a queue of size $k-c$. A much studied instance of these systems is the M/M/c/∞ queue (abbreviated as M/M/c); here the queue is infinitely large. In this section, we focus on the M/M/c/c queue, which, despite its name, does not possess any queue. The 'M' terms indicate that the arrival process is Poisson and service times of each server is exponentially distributed (so that both processes are memoryless). For an M/M/c/c queue, individuals (customers, messages, packets, manufacturing jobs etc) arrive via a homogeneous Poisson process and are instantly handed to one of the servers; when no servers are free, they are discarded. The M/M/c/c queue is sometimes called the Erlang loss model (Medhi, 2002) and has been used to model a variety of phenomena such as traffic in telephone networks, computer networks etc (Asmussen, 2003).

Let α be the rate of the arrival process, and let the average service time of the servers be $1/\beta$ (remember that this quantity is exponentially distributed). Let $\mathbf{S}(t)$ represent the number of busy servers at time t. Then, under the M/M/c/c queue, the stochastic process $\mathbf{S}(t)$ evolves according to a simple Markov jump process on the space $\mathcal{S} = \{1, \cdots, c\}$. This MJP is a birth-death process whose state can change only by 1. When $\mathbf{S}(t) = s < c$, a transition from s to $s+1$ occurs with a rate α. On the other hand, a transition from s to $s-1$ occurs with rate $s\beta$. In our previous notation, the various transition rates $A_{ss'}$ are:

$$A_{ss'} = \begin{cases} \alpha & s' = s+1, s < c \\ s\beta & s' = s-1 \\ 0 & \text{otherwise} \end{cases} \tag{7.1}$$

A special instance of the M/M/c/c is the M/M/∞ queue, where the number of servers is infinite. This is sometimes called an *immigration-death* process (Asmussen, 2003). Individuals enter the population according to a homogeneous Poisson process whose rate is independent of the population size, while each individual has a fixed rate of dying (so that the rate at which the population decreases by 1 is proportional to the population size). This is often used as an approximation of the M/M/c/c queue with large numbers of servers, although it is an interesting model in its own right. Observe that since the number of active jobs in the M/M/∞ queue is unbounded, we cannot upper bound the event rates in the system (see equation (7.1)). Thus, our uniformization-based MCMC sampler is not applicable to this system. Instead, one has to approximate the system

with an M/M/c/c queue; this is how we proceeded when we faced a similar problem with the Lotka-Volterra model in subsection 4.3.1. By contrast, the leaving rate of any state s is $s\alpha + \beta$. Since this is finite, we can apply our thinning based sampler.

In the following, we consider the evolution of an M/M/∞ queue over an interval $[0, t_{end}]$. Our dependent thinning scheme ensures only a finite number of candidate state transitions over this interval. Suppose that the state of the system was perfectly observed at time 0 to be s_0. The birth-death nature of the process means that at the ith candidate jump time w_i, $\mathbf{S}(w_i)$ can take a maximum value of $(s_0 + i)$. Thus in this case, the dimensionality of all messages is finite, allowing a straightforward application of the forward-backward algorithm. A complication arises when the initial state is noisily observed. If we allow the range of s_0 to be infinite, then even if the number of steps in the forward-backward algorithm is finite, the dimensionality of each message is infinite. A simple way around this problem is to take a slice sampling approach (Neal, 2003a; Walker, 2007), instantiating only a finite number of states at any iteration.

Accordingly, associate a slice variable l with the initial distribution over states, and let it be uniformly distributed on the interval $[0, 1]$. Observe that

$$\pi_0(s_0) = P(l < \pi_0(s_0)) = \int_0^1 1(l < \pi_0(s_0))\mathrm{d}l \tag{7.2}$$

Thus, the joint probability of initial state s_0 and l is given by

$$P(s_0, l) = 1(l < \pi_0(s_0)) \tag{7.3}$$

Given the state s_0, we resample l uniformly on the interval $[0, \pi_0(s)]$. Conversely, for a given value of the slice variable l, s_0 is uniformly distribution over all states s such that $\pi_0(s) > l$.

Let s_0^{max} be the largest state satisfying this condition:

$$s_0^{max}(l) = \max s \text{ s.t. } \pi_0(s) > l \tag{7.4}$$

Let l be the current value of the slice variable, so that $s_0^{max}(l)$ is the maximum value of state at time 0. Then, the maximum value of the s_i, the state at step i of the forward-backward algorithm, is $s_0^{max}(l) + i$. We now can easily run the forward-backward algorithm to sample a new trajectory. At the end of this step, let $\tilde{s}_0$ be the new state at time 0; we then resample a new value of the slice variable $\tilde{l}$ as follows:

$$\tilde{l} \sim \mathcal{U}(0, \pi_0(\tilde{s})) \tag{7.5}$$

It it possible to introduce more slice variables for more control over the dimensionality of the MJP state space; however, we will not discuss such schemes here.

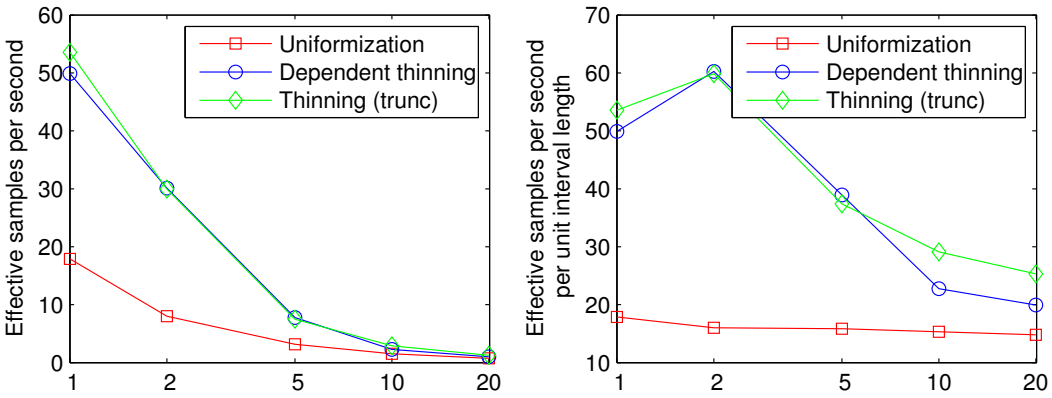

Figure 7.4: The M/M/∞ queue: (left) ESS per unit time, (right) ESS per unit time scaled by interval length.

7.3.1 Experiments

In the following, we considered an M/M/∞ queue, with parameters α and β set to 10 and 1 respectively. For some t_{end}, the state of the system was observed perfectly at three times 0, $t_{end}/10$ and t_{end}, with values 10, 2 and 15 respectively. Conditioned on these, we sought the posterior distribution of the state trajectory on the interval $[0, t_{end}]$. We compared our uniformization-based sampler from chapter 3 with the the generalized thinning-based sampler we outlined in this chapter. To run uniformization, we approximated the M/M/∞ system with an M/M/50/50 system. We also applied the thinning-based sampler to this truncated approximation, labelling it as 'Thinning (trunc)'. All samplers were implemented in Matlab. For the uniformization-based sampler, we set $\Omega = 2$, so that the subordinating Poisson process had a rate of 120. For the other two samplers, we set the thinning probability equal to a half, so that for any state s with leaving rate A_s, candidate leaving events were generated from a rate $2A_s$ process. The large state spaces involved makes particle MCMC very inefficient, and we did not include it in our results.

For all three samplers, we calculated effective sample sizes produced per unit time as we varied the interval length t_{end} from 1 to 20. In all cases, we ran 10000 MCMC iterations with a burn-in period of 1000. At the end of any MCMC run, we calculated the number of state transitions, as well the amount of time spent in states 0 to 20. We estimated effective sample sizes for all these statistics, and summarized them with their median. The left plot in figure 7.4 plots this for the three samplers as we varied t_{end}. Sampling a trajectory on a long interval will take more time than on a short one, and to more clearly distinguish performance for large values of t_{end}, the right plot in figure 7.4 scales the each result from the left plot with the length of the interval t_{end}.

We see from figure 7.4 that for short intervals, uniformization is significantly more inefficient that our other two samplers. This is because the subordinating Poisson rate of 120 is much larger than the observed rates in typical trajectories sampled from the pos-

terior. Thus, a large number of the candidate transition times had to be thinned and the long Markov chains for the forward-backward algorithm resulted in long computation times. By contrast, the other two samplers produce much fewer candidate transition times, and therefore require much less computation time per iteration. Slower mixing notwithstanding, they perform much better. Interestingly, running the thinning-based sampler on the truncated M/M/50 queue offers no significant computational benefit over running it on the full model.

As the observation interval becomes longer and longer, the MJP can make larger and larger excursions (especially over the interval $[t_{end}/10, t_{end}]$). Thus as t_{end} increases, the number of thinned events in all three samplers starts to become comparable. This, coupled with its faster mixing, causes the uniformization-based sampler to approach the performance of the other two samplers. At the same time, we see that the difference between the truncated and the untruncated samplers starts to widen. Of course, we should also remember that over long intervals, the effect of truncating the system size to 50 becomes more and more likely to introduce biases into our inferences.

7.4 The effect of an unstable state

In our next experiment, we compared Matlab implementations of the uniformization-based sampler (which we call 'uniformization'), its thinning-based generalization (we just call this 'thinning') and a particle MCMC sampler. The first two samplers were set up as in the last section, while particle MCMC was run with 20 particles.

All three samplers were applied to a simple 3 state MJP. Two of the states of this system had leaving rates equal to 1, while the leaving rate of the third state was varied from 1 to 20 (call this rate γ). On leaving state i, the probability of transitioning to state $j = (i+1) \bmod 3$ was uniformly drawn between 0 and 1:

$$p_i(j) \sim \mathcal{U}(0,1) \quad \forall i, \text{ with } j = (i+1) \bmod 3 \tag{7.6}$$

Thus, the transition probability to $(i+2) \bmod 3$ was just $1 - p_i(j)$.

Following this procedure, we constructed 10 random parameterizations of the MJP for each setting of γ. We then distributed 5 random observation times over the interval $[0, 10]$; these observations were set to randomly favour one state over the others by a factor of 100. As in subsection 6.6.1, the likelihood functions were raised to an inverse temperature inv, and we considered 3 settings of this parameter: 'low' (0.1), 'mid' (0.5) and 'high' (0.9).

For each inverse temperature setting, and for each setting of γ, we evaluated the performance of the three samplers on the 10 random MJPs. Figure 7.5 shows the results for the low, medium and high settings of inv. Each plot shows the number of effective

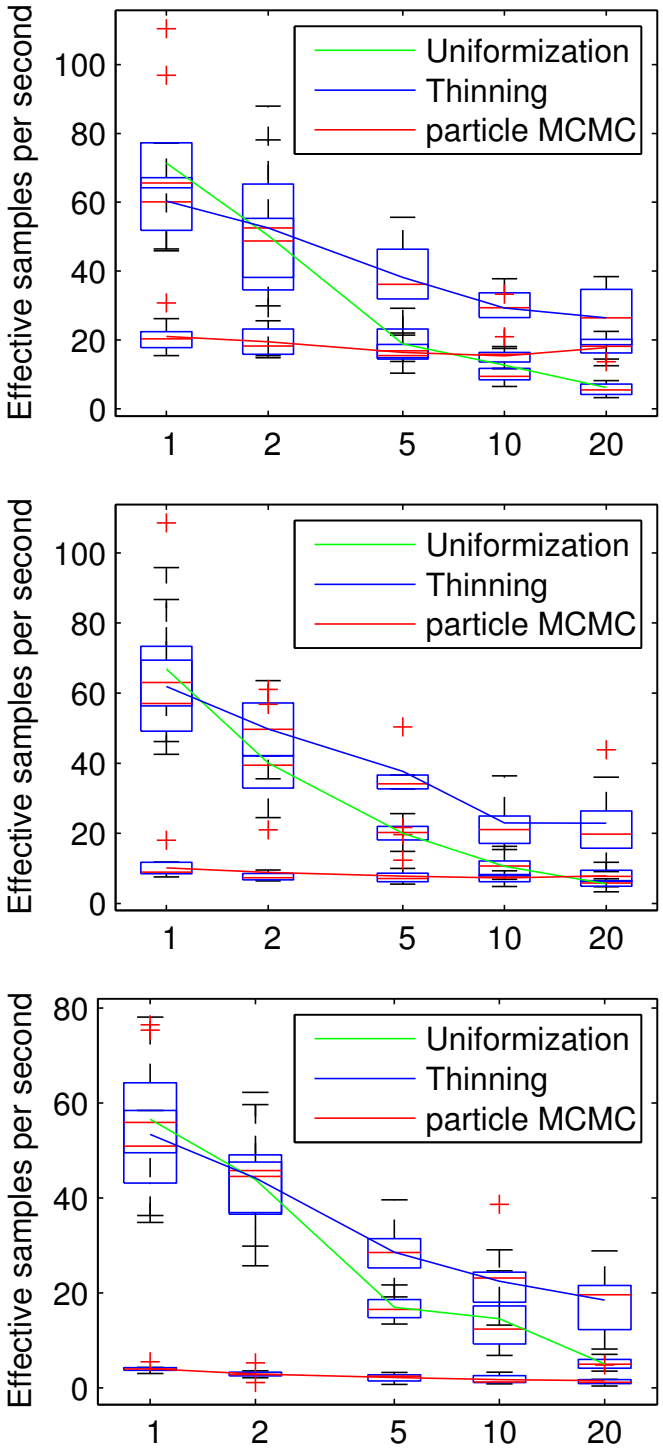

Figure 7.5: Comparison of samplers as the leaving rate γ of a state increases. Temperature decreases from top to botton

samples (estimated as in chapter 6) produced per unit time, for increasing values of γ.

Firstly, we see that the two samplers significantly outperform the particle MCMC sampler. The Markov structure of the MJP makes our Poisson samplers very natural and efficient (in particular, they are much simpler than the samplers for the semi-

Markov process). The dependencies introduced by the shared Poisson processes is also much weaker than with the sMJP. Additionally, running a particle filter with 20 particles took about twice as long as the other two samplers.

Next, we find that while both the uniformization and the thinning samplers perform comparably for low values of γ, the thinning sampler starts to outperform uniformization for γ's greater than 2. In fact, for weak observations and large γs, even particle MCMC outperforms uniformization.

7.5 Discussion

In this chapter, we applied our dependent-thinning based sampler to MJPs, showing under what circumstances it offers a more efficient approach to inference than uniformization. In our experiments, we compared the two samplers on a queuing model as well as an MJP with one very unstable state.

In our experiments we looked at M/M/c/c queuing systems, though it is possible to generalize to other systems. Examples include the M/M/c queue (this has c servers, but unlike the lossy M/M/c/c systems, jobs are buffered in an infinite queue). Now, the state of the system must include the state of the queue. In general, we can look at M/M/c/k systems with c servers, and a queue of size k. Other generalizations include G/G/c/k queues, where the arrival and service times are generalized beyond the memoryless exponential. Similarly, we can generalize the birth-death nature of these processes to allow global state changes (such as catastrophies where the system is reset, or the parameters are modulated by some external factor). We can also look at networks of interacting queues, the Lotka-Volterra model of chapter 4 being an example of such a system.

Chapter 8

Spatial repulsive point processes

8.1 Introduction

In this chapter, we show how ideas from previous chapters can be extended from the real line to more general spaces. We shall restrict outselves to *spatial point processes* defined on two-dimensional Euclidean spaces, although it should be clear how these ideas generalize to more complex spaces. Spatial point processes find wide use in fields such as ecology (Hill, 1973), geography (Kendall, 1939), epidemiology (Knox, 2004), sociology (Hansford-Miller, 1968), astronomy (Peebles, 1974) etc. As one might imagine, the simplest and most popular model for such processes is the Poisson process. However, the independence property of the Poisson process is a simplification that is often unsuitable for modelling applications, and one might also wish to capture interactions between nearby events. For example, when a point process is used to model the distribution of trees in a geographical area, competition for light and other resources would suggest an inter-event distance that is more spread out than the Poisson process (Strand, 1972). Other applications where modelling such interactions is important include the distribution of cities (Glass and Tobler, 1971), galaxies (Peebles, 1974), infected agents in epidemiological studies (Jewell et al., 2009) etc.

In chapter 5, we saw that point processes on the real line could deviate from the Poisson by either being more 'bursty' or more 'refractory'. In higher dimensions, these are called *clustered* and *repulsive* point processes respectively. Our focus in this chapter will be the latter, characterized by being more regular (underdispersed) than the Poisson process. The physical reasons for such repulsion could be competition for finite resources (in the case of cities or trees, for example), interaction between rigid objects (such as cells) or repulsive forces between particles. However, developing a flexible and tractable statistical framework to study such repulsiveness is not straightforward on spaces more complicated than the real line. For the latter, we saw a powerful and convenient framework, viz. that of renewal processes. In essence, renewal processes exploit the ordering of the real line to define a Markov process where the time of an event depends only on

the time since the last event*. By allowing these waiting times to have distributions other than the exponential, one can define more flexible classes of point processes. Such an approach does not generalize easily to higher dimensions, however. For instance, consider the *avoidance function*, $a(A)$; this is the probability that no events occur in a set A (Daley and Vere-Jones, 2008). For the homogeneous Poisson with intensity λ, letting $\mu(A)$ represent the area of this set, we have from equation (2.2) that this probability decays exponentially with the area of the set (i.e. $P(N(A) = 0) = \exp(-\lambda\mu(A))$). As with renewal processes, one might try to generalize this, noting from Daley and Vere-Jones (2008) that a simple† point process is completely specified by its avoidance function evaluated on all measurable sets. However, care needs to be taken to ensure that such a process is well defined, since unlike the renewal process, such a generalization does not provide us with a direct constructive definition of the point process. Additionally, the specification above does not intuitively describe the nature of, say, pairwise interactions between events. Inference over any parameters of the avoidance function is also not straightforward.

A more direct framework for modelling interactions in point processes is that of *Gibbs processes* (Daley and Vere-Jones, 2008). Such processes arose from the statistical physics literature to describe systems of interacting particles. A Gibbs process assigns a potential energy $U(S)$ to any configuration of events $S = \{s_1, \cdots, s_n\}$, defined most generally as:

$$U(\{s_1, \cdots, s_n\}) = \sum_{i=1}^{n} \sum_{1 \le j_1 < \cdots < j_i \le n} \psi_i(s_{j_1}, \cdots, s_{j_i}) \tag{8.1}$$

where ψ_i is an ith order potential term. Usually, interactions are limited to pairwise interactions, so that the energy is given by

$$U(\{s_1, \cdots, s_n\}) = \sum_{i=1}^{n} \psi_1(s_i) + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \psi_2(s_i, s_j) \tag{8.2}$$

By choosing the pairwise potentials appropriately, we can flexibly model different kinds of interactions. Usually, the interaction kernels are chosen to be stationary, depending only on the distance r between the two points. Typical choices include

$$\psi_2(r) = -\log(1 - e^{-(r/\sigma)^2}) \tag{8.3}$$

$$\psi_2(r) = -(\sigma/r)^n \tag{8.4}$$

*One can easily generalize to more complicated dependencies on the past.

†A point process is simple if no more than one event can occur at any location.

Alternately, we can have 'hardcore' processes with interaction potentials defined as

$$\psi_2(r) = \begin{cases} \infty & r \leq R \\ 0 & r > R \end{cases} \tag{8.5}$$

Recalling the notion of the Janossy density of a point processes (section 2.4), the probability density of any configuration is then proportional to its exponentiated negative energy. Letting θ represent the parameters that characterize the potential energy, we have

$$p(S|\theta) = \frac{\exp(-U(S;\theta))}{Z(\theta)} \tag{8.6}$$

Here, we see the price we have to pay for the flexibility afforded by this modelling framework. The normalization constant $Z(\theta)$ is usually intractable, making even sampling from the prior difficult (typically, this requires a coupling from the past approach (Møller and Waagepetersen, 2007)). Inference over the parameters usually proceeds by maximum likelihood or pseudolikelihood methods (Møller and Waagepetersen, 2007; Mateu and Montes, 2001).

8.2 Matérn repulsive point processes

A simple and direct approach to constructing repulsive point processes was proposed in Matérn (1986) and is based on the idea of thinning a Poisson process. Matérn actually proposed three related schemes, now called (in order of increasing complexity) the Matérn type-I, type-II and type-III hardcore point processes. The type-I process has the following generative process: sample a *primary point process* from a homogeneous Poisson process with some intensity (say λ), and then delete all points separated by a distance less than R. While the simplicity of this scheme makes it amenable to theoretical analysis, the thinning strategy here is often too aggressive. In particular, one can show that the average number of events in any area is not monotonic in the intensity λ; rather it first increases to a maximum value, before then decreasing with λ. The reason for this is that as the number of primary Poisson events increase, the probability of a point falling within a radius R of some other point also increases, thereby increasing the probability of it being thinned. As we increase λ, this latter effects begins to dominate, so that eventually the density of points begins to decrease with λ.

The Matérn type-II process tries to rectify this. Rather than deleting *both* interacting points, we break symmetry by assigning each point an 'age'. When there is a conflict between two points, the older point always wins. Observe that this construction implies that an event can be thinned because of the influence of an earlier point that was also thinned. This makes this procedure slightly unnatural; one might expect only surviving

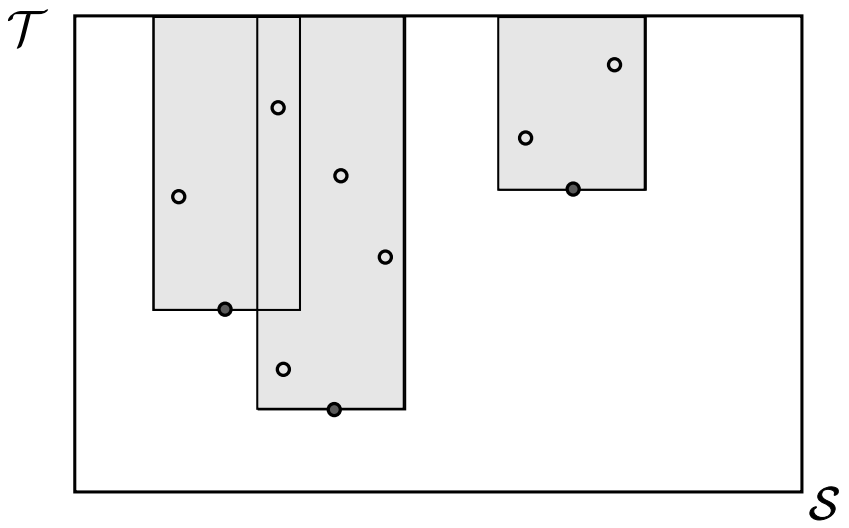

Figure 8.1: The Matérn type-III hardcore point process on a one dimensional space $\mathcal{S}$: The filled dots (projected onto $\mathcal{S}$) represent the Matérn events, the empty dots being the thinned events. The shaded region is the Matérn shadow.

points to influence future events. Additionally, while this process supports higher event densities than the type-I process, this density is still not monotonic with λ, making this parameter hard to interpret.

The final process, the Matérn type-III process does not have these limitations; as we describe more carefully in the next section, a newer event is thinned only if it falls within a radius R of an older event that was not thinned before. We shall focus on variations of this process for the rest of this chapter.

8.2.1 The Matérn type-III repulsive point process

A Matérn type-III hardcore point process on a space $(\mathcal{S}, \Sigma)$ is a repulsive point process parametrized by an intensity λ and an interaction radius R. The process is obtained by thinning events of a homogeneous *primary* Poisson process F with intensity λ. Each event $f \in F$ of the primary process is independently assigned a random mark t, the 'time' of its birth. Without any loss of generality, we assume this takes values in the interval $[0, 1]$, which we call $\mathcal{T}$. From the marking theorem (theorem 2.6), the set of pairs (f_i, t_i) forms a Poisson process F^+ on the space $\mathcal{S} \times \mathcal{T}$ (whose intensity is still λ). We shall call the set of birth times T^F, and write F^+ as (F, T^F). The set T^F induces an ordering on the events in F^+ (with probability 1, all events will have different times of birth), and thus on F. The secondary process $G^+ \equiv (G, T^G)$ is then obtained by traversing the elements of F^+ in this order and deleting all points within a distance R of any *earlier* and *undeleted* point. We obtain the Matérn process G by projecting G^+ onto $\mathcal{S}$.

Figure 8.1 shows the relevant events, with the filled dots forming the Matérn process G and the empty dots representing thinned events. Both together form the primary process. Define the 'shadow' of a point $u \equiv (s,t)$ as the subset of $\mathcal{S} \times \mathcal{T}$ consisting of all locations whose $\mathcal{S}$-coordinate differs from s by less than R, and whose $\mathcal{T}$-coordinate is greater than t:

$$\mathscr{H}(u,R) = \left\{(\tilde{s},\tilde{t}) \in \mathcal{S} \times \mathcal{T} : \|s - \tilde{s}\| < R \text{ and } t < \tilde{t}\right\}$$

The shadow of a set is the union of the shadows of its elements. The shaded area in figure 8.1 shows the shadow of all Matérn events, G. Note that all thinned events must lie in the shadow of the Matérn events, otherwise they couldn't have been thinned. Similarly, Matérn points cannot lie in each others shadows; however, they *can* fall within the shadow of some thinned event.

The Matérn type-III process has a number of desirable properties. Firstly, in many applications, its thinning mechanism is more natural than that of the type-I and II processes. In particular, it is a realistic model for various spatio-temporal phenomena, where the latent birth times are not observed, and must be inferred. Another advantage is that the average number of points in any area increases monotonically with the intensity λ of the primary process; in fact as λ tends to infinity, this average reaches the 'jamming limit', viz. the maximum density at which spheres of radius R can be packed in a bounded area (Møller et al., 2010). This implies that this process can support higher packing densities than the type-I and type-II processes with the same parameters. The monotonicity property is also important in applications where we model inhomogeneity by allowing λ to vary with location (see section 8.4), since λ now has an easy interpretibility.

In spite of these properties, the Matérn type-III repulsive process has not found widespread use in the spatial point process modelling community. Theoretically, it is the least well understood of the three Matérn processes; for instance, there does not exist a closed form expression for the average number of points in any region. This complication arises because of the dependent nature of the thinning. For the type-I and II processes, to decide whether a point is thinned or not, one only has to look at the primary Poisson process within a neighbourhood of radius R. For the type-III process, one also needs to know whether each of these neighbouring points was thinned or not, thereby requiring knowledge of primary points within neighbourhoods of those points as well. This expanding influence means that any primary event can potentially affect *all* points that were born afterwards. This can also result in tricky edge effects if one regards the area of interest, $\mathcal{S}$, as a subset of a larger ambient space (Møller et al., 2010). For the type-I and II processes, this would require us to instantiate the primary process on the union of $\mathcal{S}$ with a surrounding boundary of width R. For the type-III process, because of the effect described previously, this boundary can be arbitrarily wide. We bypass this latter issue by defining the Matérn process as generated by a

Poisson process on $\mathcal{S}$ (and not as an observed subset of some larger process).

A more practical impediment to the use of this process (and this is true for all Matérn processes) is that there do not exist efficient techniques for inference over parameters like λ or the radius of interaction R. Typical inference schemes involve incremental birth-death samplers which proceed by randomly inserting or deleting thinned events, setting up a Markov chain which converges to the true posterior over thinned events (Møller et al., 2010; Huber and Wolpert, 2009; Adams, 2009). Given the entire set of thinned events, one can perform inference over the parameters λ and R. However, the incremental nature of these updates can make the sampler mix quite slowly. The birth-death sampler can be adapted to a coupling from the past scheme to draw perfect samples of the thinned events (Huber and Wolpert, 2009). This can then be used to approximate the likelihood $p(G|\lambda, R)$ or perhaps, to drive a Markov chain following ideas from Andrieu and Roberts (2009). However, this too can be quite inefficient, with long waiting times until the sampler returns a perfect sample.

In the following sections, we show how to perform efficient inference for the Matérn type-III sampler using ideas similar to those in previous chapters. First, however, we specify the probability density of this point process; this will enable us to verify the correctness of our sampler.

8.2.2 Probability density of the Matérn type-III point process

Since the Matérn process is derived by thinning a primary Poisson process, it will be a finite point process if the primary process is finite. We restrict ourselves to this situation, thus associating (unordered) samples from the Matérn process with random sequences in the space $(\mathcal{S}^\cup, \Sigma^\cup)$ of finite sequences in $\mathcal{S}$ (see section 2.4). On the other hand, events of the augmented point process F^+ and G^+ lie in the product space $(\mathcal{S} \times \mathcal{T})$, where $\mathcal{T}$ is just the unit interval with the usual Borel σ-algebra. Let μ be a base measure on this product space; for the case when $\mathcal{S}$ is a subset of the two-dimensional Euclidean space, μ is just the Lebesgue measure on $\mathbb{R}^3$. We will then derive the density of the augmented Matérn type-III process $G^+ = (G, T^G)$ with respect to the measure $\mu^\cup$. Recall from equation (2.21) that this base measure has a factorial term correcting for the many-to-one mapping from the space of ordered sequences to an unordered sample. We will write down the density of the augmented primary Poisson process F^+ w.r.t. the measure $\mu^\cup$, and then use the thinning construction of the Matérn type-III process to calculate the probability density of G^+.

Theorem 8.1. *Let $G^+ = (G, T^G)$ be a sample from a Matérn type-III process augmented with the set of birth times. Let the process have intensity λ and interaction radius R. Then, letting I denote the indicator function, its density w.r.t. the measure*

$\mu^{\cup}$ is given by

$$P(G^+|\lambda, R) = \exp\left(-\lambda\mu\left((\mathcal{S}\times\mathcal{T})\setminus\mathscr{H}(G^+,R)\right)\right)\lambda^{|G^+|}\left(\prod_{i=1}^{|G^+|} I(G^+\cap\mathscr{H}(g_i^+,r_i)=\emptyset)\right) \tag{8.7}$$

Proof. Recall the definition of $\mathscr{H}(G^+,R)$, the shadow of G^+:

$$\mathscr{H}(G^+,R) = \{(s,t) : \exists\ i \text{ such that } \|s-g_i\| < R \text{ and } t > t_i^G\}$$

In equation (8.7) above, $(\mathcal{S}\times\mathcal{T})\setminus\mathscr{H}(G^+,R)$ represents the complement of the shadow $\mathscr{H}(G^+,R)$. Also, I is the indicator function, with the product term requiring that no Matérn event lie within the Matérn shadow. Now, let $|G|$, the size of the set G be k. G^+ is obtained by thinning F^+, a sample from a homogeneous Poisson process with intensity λ. Let the size of F^+ be $n > k$; its density w.r.t. the measure $\mu^{\cup}$ is then

$$P(F^+) = \exp\left(-\lambda\mu(\mathcal{S}\times\mathcal{T})\right)\lambda^n \tag{8.8}$$

Now, there are $\binom{n}{k}$ ordered versions of F^+ mapping deterministically to the Matérn sequence G^+, so that the conditional density of G^+ is given by

$$P(G^+|F^+) = \frac{n!}{k!(n-k)!} I(F^+\in\mathscr{H}(G^+,R))\prod_{i=1}^{|G|} I\left(G^+\cap\mathscr{H}(g_i^+,r_i)=\emptyset\right) \tag{8.9}$$

The first indicator term in the equation above requires all primary points F^+ to fall within the shadow $\mathscr{H}(G^+,R)$, and the second term requires that no Matérn event lies in the shadow. Following Huber and Wolpert (2009), we write this term more compactly as $I_{\rho(G)>R}$, where $\rho(G)$ is the minimum distance between the elements of the set G. Thus,

$$P(G^+|F^+) = \frac{n!}{k!(n-k)!} I(F\in\mathscr{H}(G^+,R))I_{\rho(G)>R} \tag{8.10}$$

Then we have

$$P(F^+,G^+) = \frac{n!}{k!(n-k)!}\exp(-\lambda\mu(\mathcal{S}\times\mathcal{T}))\lambda^n I(F\in\mathscr{H}(G^+,R))I_{\rho(G)>R} \tag{8.11}$$

Integrating out the locations of $n-k$ thinned events, we have

$$P\left(G^+,|F^+|=n\right) = \frac{n!}{k!(n-k)!}\exp(-\lambda\mu(\mathcal{S}\times\mathcal{T}))\lambda^k\frac{\left(\lambda\mu(\mathscr{H}(G^+,R))\right)^{n-k}}{n!/k!}I_{\rho(G)>R} \tag{8.12}$$

$$= \exp(-\lambda\mu(\mathcal{S}\times\mathcal{T}))\lambda^k\frac{\left(\lambda\mu(\mathscr{H}(G^+,R))\right)^{n-k}}{(n-k)!}I_{\rho(G)>R} \tag{8.13}$$

The $n!/k!$ term arises because of the factorial term in the base measure $\mu^{\cup}$, see equation (2.42). Finally, summing out n, we have

$$P\left(G^{+}\right) = \exp(-\lambda\mu(\mathcal{S}\times\mathcal{T}))\lambda^{k} I_{\rho(G)>R} \sum_{n=k}^{\infty} \frac{(\lambda\mu(\mathscr{H}(G^{+},R)))^{n-k}}{(n-k)!} \tag{8.14}$$

$$= \exp\left(-\lambda\left(\mu(\mathcal{S}\times\mathcal{T}) - \mu(\mathscr{H}(G^{+},R))\right)\right)\lambda^{k} I_{\rho(G)>R} \tag{8.15}$$

This is what we set out to prove. A similar result was derived in Huber and Wolpert (2009); they express the Matérn type-III density with respect to a homogeneous Poisson process with intensity 1. However, their proof technique is less direct than ours, proceeding via a coupling from the past construction. □

8.2.3 Inference for Matérn type-III processes

We now show how the apparently complicated dependent thinning in the generative procedure of the Matérn type-III process actually allows for efficient inference. The insight here is that a point of the primary Poisson process F can be thinned only by an element of the secondary process. Consequently, given the secondary process, there are no interactions between the thinned events themselves. It turns out then, that given the secondary process, the thinned events are just Poisson distributed, making it possible to sample them directly. This can be much more efficient than an incremental scheme that updates the thinned Poisson set one event at a time. Note that such a strategy does not extend to Matérn type-I and II processes, where the fact that thinned events *can* delete each other means that the posterior is no longer Poisson. For instance, for any of these processes, it is not possible for a thinned event to occur by itself within any neighbourhood of radius R (else it couldn't have been thinned in the first place). However, two or more events *can* occur together. Clearly such a process is not Poisson, rather it possesses a clustered structure.

Now consider a sample G from a Matérn type-III process. Assume the birth times of these events have been instantiated, so that we have a realization of G^{+}. The thinned events (call them $\tilde{G}^{+}$, so that $F^{+} = G^{+} \cup \tilde{G}^{+}$) can only lie within the shadow $\mathscr{H}(G^{+},R)$ (see figure 8.1). Recalling that the primary process is a sample from a homogeneous Poisson process with intensity λ, it might appear that the thinned events are distributed as the Poisson process restricted to the shadow $\mathscr{H}(G^{+},R)$. The next proposition verifies that this intuition is correct.

Proposition 8.1. *Let $G^{+} = (G, T^{G})$ be a sample from an Matérn type-III hardcore process on the space $(\mathcal{S},\Sigma)$, augmented with its birth times. Let the primary intensity be λ and the radius of interaction be R. Then, the posterior distribution of the locations and birth times of the thinned set, $\tilde{F}^{+} = (\tilde{F}, T^{\tilde{F}})$ given G^{+} is a Poisson process with intensity λ, restricted to the shadow of the observations $\mathscr{H}(G^{+},R)$.*

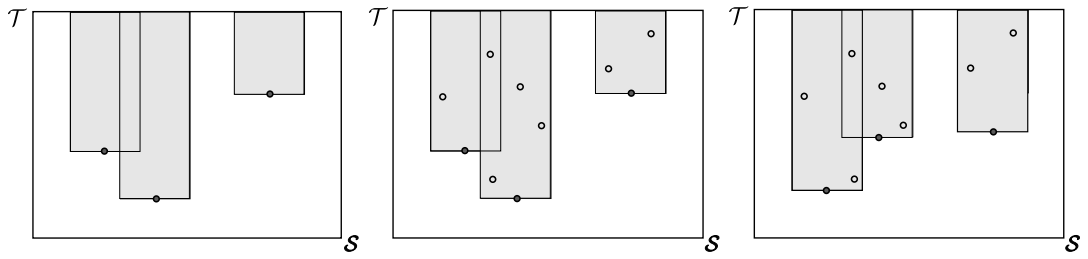

Figure 8.2: Updating latent variable in the Matérn type-III process. (left) current value of G^+, (centre) after resampling $\tilde{F}^+$, (right) after resampling T^G

Proof. Plugging equation (8.10) and equation (8.9) into Bayes' rule, we have

$$p(\tilde{G}^+|G^+) = \frac{p(\tilde{G}^+, G^+)}{p(G^+)} = \frac{p(F^+)}{p(G^+)} \tag{8.16}$$

$$= \exp\left(-\lambda\left(\mathscr{H}(G^+, R)\right)\right) \lambda^{|\tilde{G}^+|} I(\tilde{G}^+ \in \mathscr{H}(G^+, R)) \tag{8.17}$$

This is just the density of a Poisson process whose intensity is $\lambda(x)$ for $x \in S(G)$ and 0 everywhere else. □

Having reconstructed the primary Poisson process F^+, we next want to resample the birth times T^G of the Matérn events G. We do this iteratively, conditionally resampling these one at a time for all elements of G. Observe that for any primary event, the birth time is *a priori* distributed uniformly on the interval $[0, 1]$. For a Matérn event, we have the additional constraint that its resampled birth time does not expose any thinned event to lie outside the resulting shadow $\mathscr{H}$ (see figure 8.1). In other words, for element $g \in G$, we resample t_g^G uniformly on the interval $[0, t_{min}]$, where t_{min} is the time of the oldest thinned event that does not lie in $\mathscr{H}((G \setminus g)^+, R)$. We can easily prove this the way we proved proposition 8.1.

We note that unlike, say, the MJP, where we used the forward-backward algorithm to jointly resample the labels of the underlying Poisson process, we can no longer easily produce a joint update of the birth times T^G that is conditionally independent of the previous values. While it is possible to develop more global moves, we found it sufficient to sweep through the Matérn events, updating their birth times one at a time. This, together with jointly updating all the thinned event locations and birth times was sufficient to ensure that the Markov chain mixed rapidly. Figure 8.2 shows the steps in one such cycle.

In our experiments, we naïvely resampled the thinned events $\tilde{G}^+$ applying the restriction theorem (theorem 2.3). In particular, we sampled a Poisson process on $(\mathcal{S} \times \mathcal{T})$ and thinned out all those points that did not lie within $\mathscr{H}(G^+, R)$. Again, it is possible to be more clever about this (eg. by covering $\mathscr{H}(G, R)$ with a disjoint set of rectangles). We found this additional complexity to be unnecessary for the values of λ and $\mu(\mathcal{S} \times \mathcal{T})$ that we considered.

Having reconstructed the thinned events, it is straightforward to resample the Matérn parameters λ and R. Like the birth times T^G, for any prior on R, the posterior is just this prior distribution truncated so that the resulting shadow $\mathscr{H}(G^+, R^{new})$ is consistent with all the events and their labels (see equation (8.11)). The lower bound for this truncation requires that no thinned event lie outside the new shadow, while the upper bound requires that no Matérn event lie inside the shadow. We place a uniform, noninformative prior on R. We also place a $Gamma(a, b)$ prior on the primary Poisson intensity λ; as we saw in chapter 5, this is conjugate to the primary Poisson process and results in a $Gamma(a_{post}, b_{post})$ posterior where $a_{post} = a + |F|$, $\frac{1}{b_{post}} = \frac{1}{b} + \frac{1}{\mu(X)}$. Algorithm 8.1 describes one iteration of our sampler.

Algorithm 8.1 MCMC sampler for posterior inference in a Matérn type-III hardcore process on the space $\mathcal{S}$

Input:	The set of Matérn observations G, and their birth times t_G. The values of the parameters λ and R.
Output:	A new set of thinned events $\tilde{G}^+ \equiv (\tilde{G}, T^{\tilde{G}})$ and Matérn birth times T^G. New values of the parameters λ and R.

1: Construct the shadow $\mathscr{H}(G^+, R)$ of G^+.
2: Sample $\tilde{G}^+$ from a Poisson process with intensity λ restricted to $\mathscr{H}$.
3: Proceed iteratively through the Matérn events G. For each event g, resample its birth time, ensuring that the new shadow still supports all thinned events $\tilde{F}^+$.
4: Resample the interaction radius R ensuring that the new shadow a) supports all thinned events $\tilde{F}^+$, and b) does not cover any Matérn event $g^+ \in G^+$.
5: Resample the Poisson intensity λ from its gamma posterior.

8.3 Experiments

In this section, we evaluate our sampler on two datasets, the classic redwood dataset (Ripley, 1977) and the Swedish pine tree dataset (Ripley, 1988) (see figure 8.3). The former records the locations of trees belonging to the Californian giant redwood family, and consists of 62 trees located within an area normalized to a square whose sides have length 5. The Swedish pine tree dataset consists of 71 trees, again located in a 5-by-5 square. Both datasets are available as part of R package spatstat (Baddeley and Turner, 2005). We model the locations of the trees in both datasets as distributed according to a Matérn type-III hardcore point process, placing a $Gamma(1, 1)$ prior on the intensity λ and a noninformative prior on the radius R (or equivalently a uniform prior in the interval $[0, 5]$). Results were obtained from MCMC runs with 10000 iterations, with a burn-in of 1000 samples (a Matlab implementation of our sampler takes about 10 seconds to produce 10000 MCMC samples).

The left and centre plots in figure 8.4 shows the posterior distributions over λ and R produced by our sampler for the redwood dataset, while figure 8.5 shows these

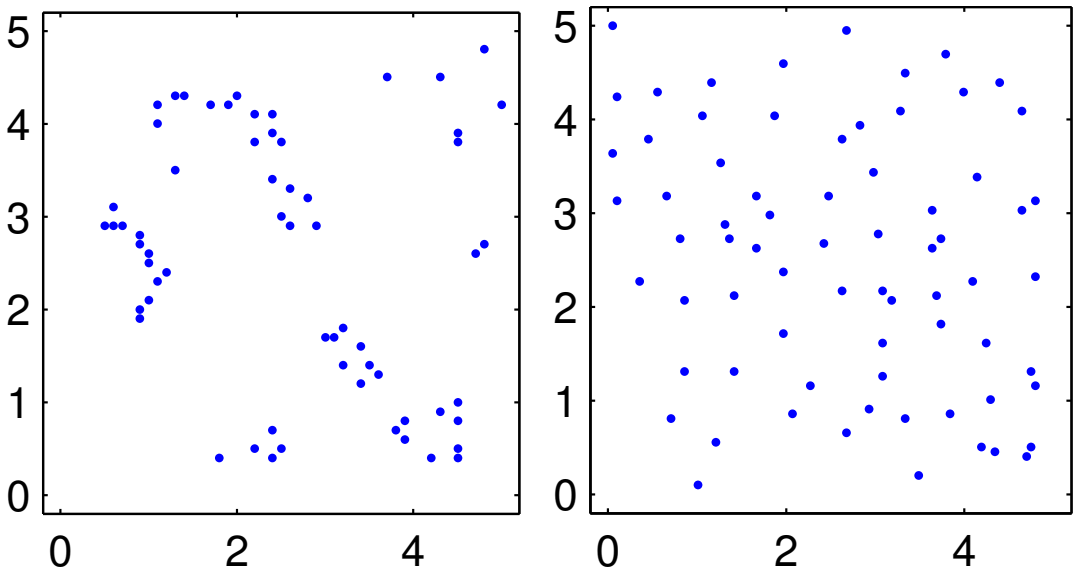

Figure 8.3: The redwood tree dataset (left) and the Swedish pine tree dataset (right)

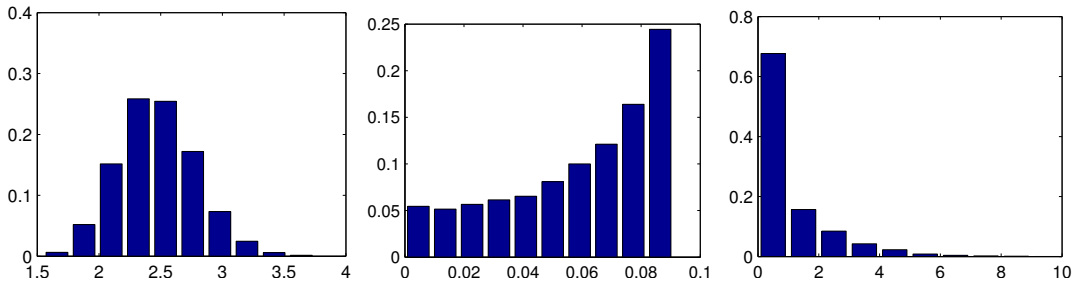

Figure 8.4: Redwood dataset: posterior distributions of Matérn intensity (left), interaction radius (centre) as well as the number of thinned events (right).

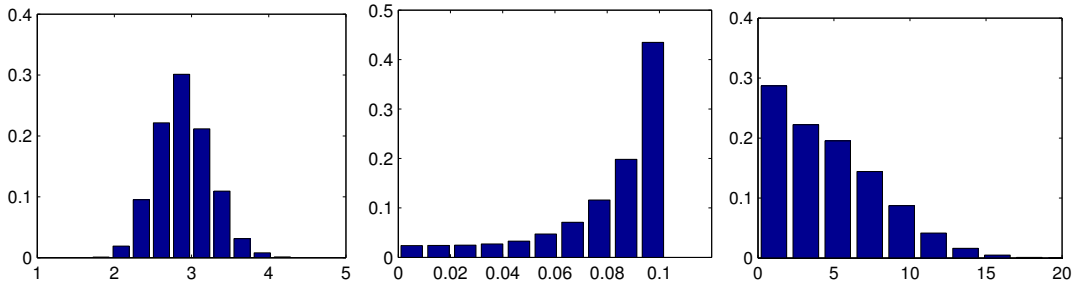

Figure 8.5: Swedish pine tree dataset: posterior distributions of Matérn intensity (left), interaction radius (centre) as well as the number of thinned events (right).

	Redwood dataset	Swedish tree dataset
Matérn interaction radius	473.64	344.51
Latent times (averaged across observations)	1000	989.47
Primary Poisson intensity	988.93	954.7

Table 8.1: Effective sample sizes (per 1000 samples) for the Matérn type-III hardcore model

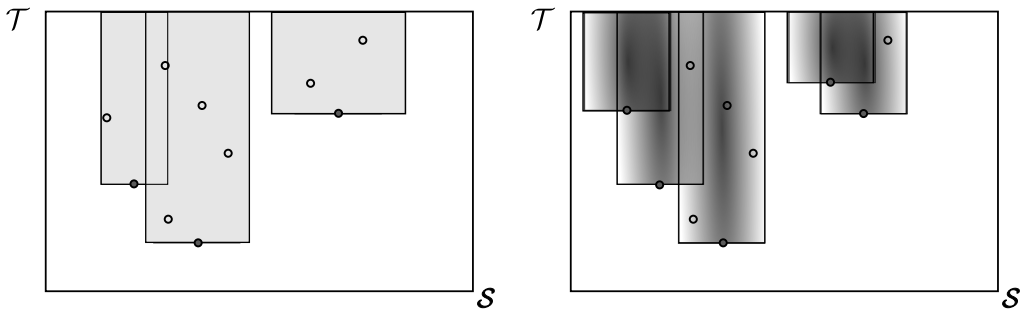

Figure 8.6: Matérn type-III softcore point processes with: a) varying interaction radii b) probabilistic deletion

quantities for the Swedish dataset. Recall that the area of the square is 25, while the number of Matérn events is of the order 70 in both datasets. The fact that the posterior distributions over the intensities λ concentrate around 2.5 to 3 suggests that the number of thinned events is small. That this is indeed the case can be seen from the rightmost plots in the two figures showing the posterior distributions over the number of points deleted due to Matérn thinning. A small number of thinned events suggests a weak repulsive effect.

The reason for this weak repulsion is because the Matérn type-III hardcore model specified above is too inflexible for this data. Observe that the posterior distribution of the interaction radius is bounded above by the minimum separating distance between all pairs of observations (otherwise one of these events would have deleted the other). In both datasets, we have at least one pair of events that has a very small separation. A small interaction radius results in a small shadow, and since thinned events are restricted to lie within this shadow, we have a small number of thinned events. This suggests that a hardcore model with a single interaction radius is inappropriate for these datasets. Another limitation can be seen for the redwood dataset. Though our model is homogeneous, this process clearly is not. We address these limitations in the next section. For completeness we also include effective sample sizes produced by our sampler; these are shown in table 8.1 and demonstrate that our sampler mixes rapidly.

8.4 Generalized Matérn type-III processes

For more applications of the Matérn type-III process, we refer the reader to Møller et al. (2010) or Huber and Wolpert (2009). However, the real advantage of our MCMC sampler is that it easily and efficiently extends to more complicated variations of the Matérn type-III process. For instance, instead of requiring all Matérn events to have identical interaction radii, we can assign each one an independent radius drawn from some prior distribution $p(R)$. Such Matérn processes are called 'softcore' (Huber and Wolpert, 2009). In this case, the primary process can be viewed as a Poisson process on a space whose coordinates are location $\mathcal{S}$, birth time $\mathcal{T}$ and interaction radius $\mathcal{R}$. Given a realization of this process $F^+ \equiv (F, T^F, R^F)$, we define a secondary point process $G^+ \equiv (G, T^G, R^G)$ by thinning deleting all points that fall within the radius associated with an older, undeleted primary event. The set of locations G constitue a sample from the softcore Matérn type-III process. Given the triplet (G, T^G, R^G), we can once again calculate the shadow $\mathscr{H}$, now defined as:

$$\mathscr{H}(G^+) = \{(s,t) : \exists\ i \text{ such that } \|s - g_i\| < r_i^G \text{ and } t > t_i^G\}$$

Figure 8.6(a) illustrates such a shadow; note that the interaction radii of the thinned events are irrelevant. Resampling the thinned events is now identical to the previous section, viz. resample a Poisson process with intensity restricted to the shadow $\mathscr{H}(G^+)$. We now need to resample the interaction radius r_g^G of each Matérn event g as well, and we do this by sequentially updating these one at a time. Once again, the posterior is the prior $P(R)$ truncated so that the resulting shadow is consistent with the instantiated primary and secondary events. A change in a radius r_g^G now produces only a local change in the shadow $\mathscr{H}$, so that most of these updates are uncoupled.

Another approach to soft repulsion is to *probabilistically* thin events of the primary Poisson process; such an approach was suggested in (Adams, 2009). The probability of deletion can depend on the distance of a point to a previous unthinned point, and a primary event is retained only if it is left unthinned by all surviving points with earlier birth times. To keep this process efficient, (Adams, 2009) suggests using a deletion kernel with compact support; figure 8.6(b) illustrates the resulting shadow. Where previously the Matérn events defined a black-or-white shadow, now the shadow can have intermediate grey values corresponding to the probability of deletion. Our sampler handles this case without any difficulty; following corollary 2.1, it is easy to see that the thinned events $\tilde{F}$ are now drawn from a Poisson process with intensity $\lambda\mathscr{H}(G^+)$. Sampling from this Poisson process is a simple application of the thinning theorem: instantiate a Poisson process with intensity λ on $\mathcal{S} \times \mathcal{T}$, and keep each point with probability $\mathscr{H}(G^+)$. Notice that for the hardcore model, this reduces to the sampling scheme of proposition 8.1.

Another extension, suggested in Adams (2009), and which follows naturally from ideas

in chapter 5, is to allow nonstationarity in the intensity function λ. Instead of requiring the primary process to be homogeneous with a constant intensity λ, we allow $\lambda(s)$ to vary over $\mathcal{S}$. Such a model would be suitable for the redwood dataset, for instance. Like chapter 5, we place a (transformed) Gaussian process prior on the intensity function $\lambda(t)$ as follows:

$$l(\cdot) \sim \mathcal{GP}(\mu, K), \tag{8.18}$$

$$\lambda(\cdot) = \hat{\lambda}\sigma(l(\cdot)) \tag{8.19}$$

where $\mu(\cdot)$ and $K(\cdot,\cdot)$ are the GP mean and covariance kernel, $\hat{\lambda}$ is a positive scale parameter, and $\sigma(x) = (1+\exp(-x))^{-1}$ is the sigmoid transformation. Like chapter 5, this sigmoid transformation serves two purposes: to ensure that the intensity $\lambda(s)$ is nonnegative, and to provide a bound $\hat{\lambda}$ on the Poisson intensity. As we saw in chapter 5, such a bound makes sampling from the inhomogeneous primary process a straightforward application of the thinning theorem: sample a random set E from a homogeneous Poisson process with intensity $\hat{\lambda}$, instantiate the Gaussian process $l(\cdot)$ on this set and keep an element e with probability $\sigma(l(e))$. Observe that there are now two stages of thinning, the first is an application of the Poisson thinning theorem to obtain the inhomogeneous primary process F from the homogeneous Poisson process E, and the second, the Matérn thinning to obtain G from F. In algorithm 8.2, we outline the generative process for an inhomogeneous Matérn type-III softcore process.

Algorithm 8.2 Algorithm to sample an inhomogeneous Matérn type-III softcore point process on a space $\mathcal{S}$

Input: A Gaussian process prior $\mathcal{GP}(\mu, K)$ on the space $\mathcal{S}$, a constant $\hat{\lambda}$ and a distribution $p(R)$ over interaction radii.
Output: A sample G from the Matérn type-III process.

1: Sample E from a homogeneous Poisson process with intensity $\hat{\lambda}$.
2: Sample from the Gussian process $l(\cdot)$ on this set of points. Call this l_E.
3: Let $\lambda_E = \sigma(l_E)$. Keep a point $e \in E$ with probability $\lambda(e)$, otherwise thin it. The surviving set of points form the primary process F.
4: Assign F a set of random birth times T^F uniformly on the interval $[0, 1]$.
5: Assign F a set of random interaction radii R^F i.i.d. from $P(R)$.
6: Proceed through the elements of F in the order of their birth, deleting any event that lies within a radius r_i^F of an earlier, undeleted event i.
7: The surviving set of points G form the secondary Matérn type-III point process.

Like subsection 8.2.3, we place a gamma hyperprior on $\hat{\lambda}$. We can also place hyperpriors on the GP hyperparameters.

8.4.1 Inference for the inhomogeneous Matérn type-III process

(Adams, 2009) defined an inhomogeneous softcore Matérn process similar to the model

in the algorithm 8.2 (their model introduced softcore repulsions by probabilistic thinning, instead of the independent interaction radius approach we took). To perform posterior inference given the Matérn events, they defined an MCMC algorithm that proceeded via an incremental birth-death process. More specifically, conditioned on all other variables (including the GP values l_E), they defined Metropolis-Hastings proposals by randomly adding to or deleting elements from the thinned sets $\tilde{G}$ and $\tilde{F}$ (here, $G \cup \tilde{G} = F$ and $F \cup \tilde{F} = E$). They also defined proposals that perturbed the locations of the elements of these sets. After updating the set E, they could perform inference on the latent GP values, and $\hat{\lambda}$. Additionally, rather than inducing an ordering on F indirectly via a set of birth times, they directly defined a random permutation on the elements of F. Inference on this permutation proceeded via moves that select and attempt to swap pairs of elements in the permutation. All these moves listed above are local and incremental, and can result in very poor mixing.

We saw in subsection 5.3.1 how we could jointly resample $\tilde{F}$, the Poisson events deleted in the first thinning stage that produced the inhomogeneous Poisson process F from E. This is a straightforward application of the corollary of the thinning theorem (corollary 2.1): $\tilde{F}$ is a draw from an inhomogeneous Poisson process with intensity $(\hat{\lambda} - \lambda(\cdot))$. Similarly, following subsection 8.2.3, we can easily see how to resample the Matérn -thinned events $\tilde{G}$: these can be sampled from a Poisson process with intensity $\lambda(\cdot)$ restricted to the shadow $\mathscr{H}(G^+)$. Observe that this step jointly produces a conditionally independent sample of the number of Matérn -thinned events, their locations and their relative ordering given the ordering of the Matérn events. Each of these steps was performed on an element by element basis in Adams (2009).

Resampling the GP evaluated on the set $E = F \cup \tilde{F}$ reduces to a GP classification problem, with elements of F belonging to class 1 and those of $\tilde{F}$ belonging to class 0 (see chapter 5). We used elliptical slice sampling (Murray et al., 2010) to perform this step. Similarly, following chapter 5 we resampled the GP bound $\hat{\lambda}$ from its gamma posterior.

Finally, we resampled the marks of the Matérn events (viz. their birth times and interaction radii) incrementally as described in subsection 8.2.3. We describe our overall sampler in algorithm 8.3.

Algorithm 8.3 MCMC sampler for an inhomogeneous Matérn type-III softcore point process on a space $\mathcal{S}$

Input: A set of Matérn events $G^+ \equiv (G, T^G, R^G)$, a set of thinned primary events $\tilde{G}^+ \equiv (\tilde{G}, T^{\tilde{G}})$ and a set of thinned Poisson events $\tilde{F}$. A GP sample l_E on $E \equiv G \cup \tilde{G} \cup \tilde{F}$

Output: New sets T^G_{new}, R^G_{new}, $\tilde{G}^+_{new}$, $\tilde{E}_{new}$ and a new instantiation of the GP on $G \cup \tilde{G}_{new} \cup \tilde{F}_{new}$.

1: Calculate the shadow $\mathscr{H}(G^+)$ of G^+.
2: **Resample the Matérn thinned events $\tilde{G}^+$:**
3: Sample a set of events $A^+ \equiv (A, T^A)$ from a homogeneous Poisson process with intensity $\hat{\lambda}$ on $\mathcal{S} \times \mathcal{T}$. Keep only those in the shadow $\mathscr{H}(G^+)$.
4: Sample $l_A | l_E$ (conditionally from a multivariate normal). Let $\lambda_A = \sigma(l_A)$.
5: Keep a point $a \in A$ with probability $\lambda(a)$, otherwise thin it.
6: The surviving set of points form the new set of Matérn thinned events $\tilde{G}^+_{new}$.
7:
8: **Resample the Poisson thinned events $\tilde{F}$:**
9: Define $E_{new} \equiv G \cup \tilde{G}_{new} \cup \tilde{F}$
10: Sample a set of events $B^+ \equiv (B, T^B)$ from a homogeneous Poisson process with intensity $\hat{\lambda}$ on $\mathcal{S} \times \mathcal{T}$.
11: Sample $l_B | l_{E_{new}}$ (conditionally from a multivariate normal). Let $\lambda_B = \sigma(l_B)$.
12: Keep a point $b \in B$ with probability $1 - \lambda(b)$, otherwise thin it. The surviving set of points form the new set of Poisson thinned events $\tilde{F}_{new}$.
13: Define $E_{new} \equiv G \cup \tilde{G}_{new} \cup \tilde{F}_{new}$
14:
15: **Resample the Matérn birth-times and interaction radii:**
16: For each Matérn observation g, resample its birth time T^G_g uniformly between 0 and the time of the earliest thinned event that lies *only* in its shadow.
17: For each Matérn observation g, resample its interaction radius R^G_g. This is just the prior on R truncated so that no thinned event lies outside the resulting shadow and no Matérn event lies in it.
18:
19: **Resample the GP values $l_{E_{new}}$:**
20: We used elliptical slice sampling (Murray et al., 2010).

8.4.2 Experiments

We return to the redwood and the Swedish tree datasets, now modelling them with the nonstationary softcore Matérn type-III process described in the previous section. We assigned each Matérn observation an interaction radius drawn uniformly on the interval $(0, 5)$[‡], and placed a $Gamma(1, 1)$ prior on the scaling parameter $\hat{\lambda}$. We modelled the inhomogeneous intensity function $\lambda(\cdot)$ using a Gaussian process with a squared-exponential kernel, and placed lognormal hyperpriors on the GP hyperparameters. Like chapter 5, we used elliptical slice sampling (Murray et al., 2010) to resample the GP values given all thinned events (step 19 in algorithm 8.3). The GP hyperparameters were resampled by slice-sampling (Murray and Adams, 2010). Again, all results were from 10000 iteration MCMC runs, with a burn-in period of 1000.

[‡]We could also place hyperpriors on the limits of this distribution, though we did not do this.

Figure 8.7 shows the posterior mean and standard deviation of the intensity of the modulating function $\lambda(\cdot)$ for the redwood dataset. The intensity function deviates strongly from a constant, and the nonstationary process is clearly far more suitable for this dataset than the homogeneous Matérn process. The left plot in figure 8.8 displays the strength of the repulsion between events. Here, we divided $\mathcal{S}$ into a 30 by 30 grid, and for each element of this grid, we plot the average number of primary Poisson events that were deleted due to Matérn thinning ($|\tilde{F}|$) divided by the area of the grid element. The plot thus shows the rate at which events are deleted due the Matérn repulsion. A high rate of Matérn thinning suggests that the process deviates strongly from the primary Poisson process. We see that while this rate of thinning is indeed high in the regions with clusters of trees, it is restricted to a relatively small portion of $\mathcal{S}$. Because the trees are clustered together in regions where the intensity $\lambda(\cdot)$ is high, the interaction radii are small, as are the number of events deleted due to Matérn thinning. This is confirmed by figure 8.9, showing posterior distributions of the scale parameter $\hat{\lambda}$, the interaction radii R^G (pooled across all events) as well as the number of Matérn -thinned events. We see that the Matérn radii are fairly small, suggesting this dataset might be modelled quite well by an inhomogeneous Poisson process.

We have also estimated and plotted of the rate at which events $\tilde{F}$ are thinned during the construction of the inhomogeneous Poisson process F from E. The right plot in figure 8.8 shows this rate of deletion; by definition it is given by $(\hat{\lambda} - \lambda(\cdot))$. In regions where this quantity is high, a large number of the primary Poisson events were thinned. Recall that this thinning required evaluating the GP on this set of points, so that a large set will affect the efficiency of our algorithm. We can try to improve efficiency by modulating a more complicated bounding function than a constant, thereby reducing the number of GP evaluations. However, now we will have to perform inference over this function as well (instead of just the constant $\hat{\lambda}$), and this can slow down mixing.

Figure 8.10 and figure 8.11 shows the corresponding plots for the Swedish dataset. In this case, the intensity function is fairly constant, agreeing with the fact the the distribution of the trees is fairly uniform. Thus, a homogeneous Matérn type-III process might be suitable for this dataset. However, the plot showing the rates of Matérn thinning suggests that this process is significantly more dispersed that a Poisson process. This is confirmed by plots for the number of thinned events and the posterior over the interaction radii (figure 8.12). Thus, this dataset could be viewed as a sample from a homogeneous Matérn type-III softcore process.

Finally, we include plots showing the effective number of samples of various statistics produced per 1000 MCMC iterations (table 8.2). For the last row of the table, we evaluated the GP intensity on a regular 10 by 10 grid. For each component of the resulting vector, we calculated the effective sample size and reported the median value. Once again, the table demonstrates that our sampler mixes the relatively rapidly. The 1 in 20 effective sample size for the GP intensity for the redwood dataset is typical

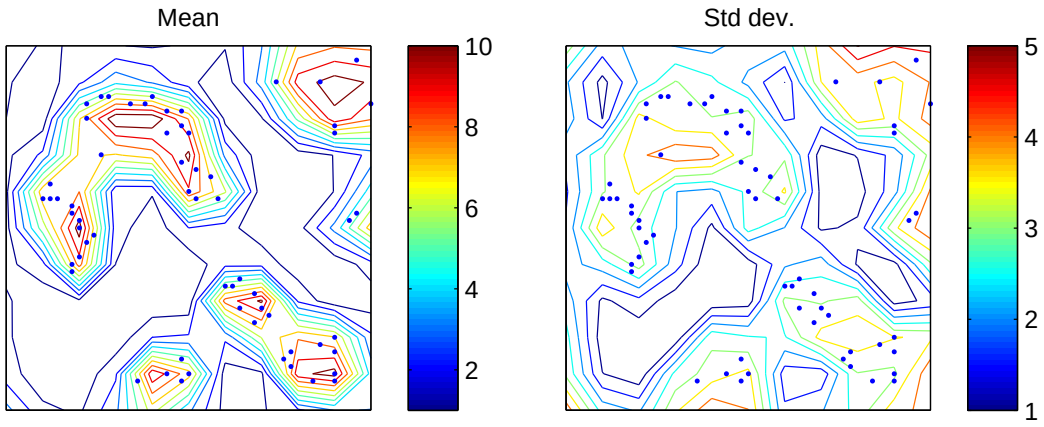

Figure 8.7: Posterior mean (left) and standard deviation (right) of the intensity of the primary process for the redwood tree dataset

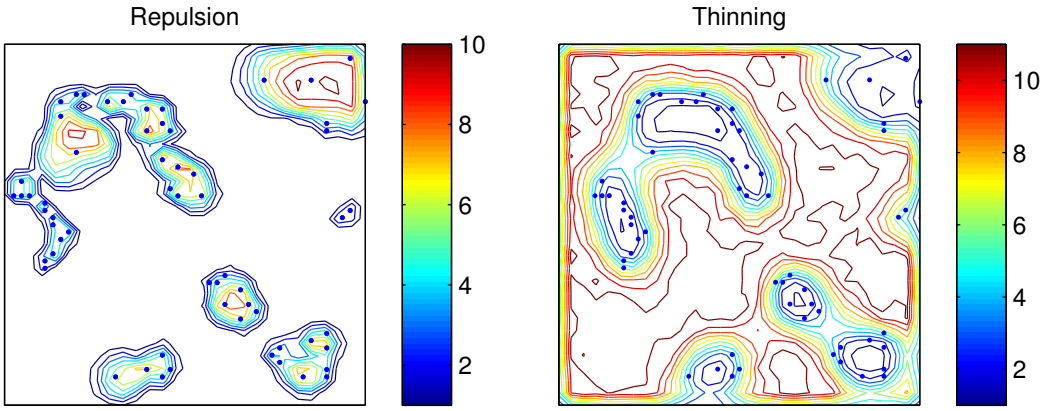

Figure 8.8: Left: Posterior rate of deletions due to repulsion, and right: Posterior rate of deletions in sampling the primary process of the redwood tree dataset

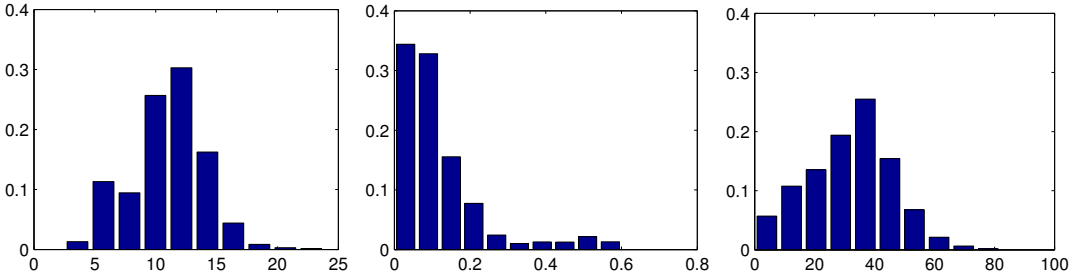

Figure 8.9: Redwood dataset: Posterior distribution of Matérn intensity (left), radius (centre) as well as the number of thinned events (right).

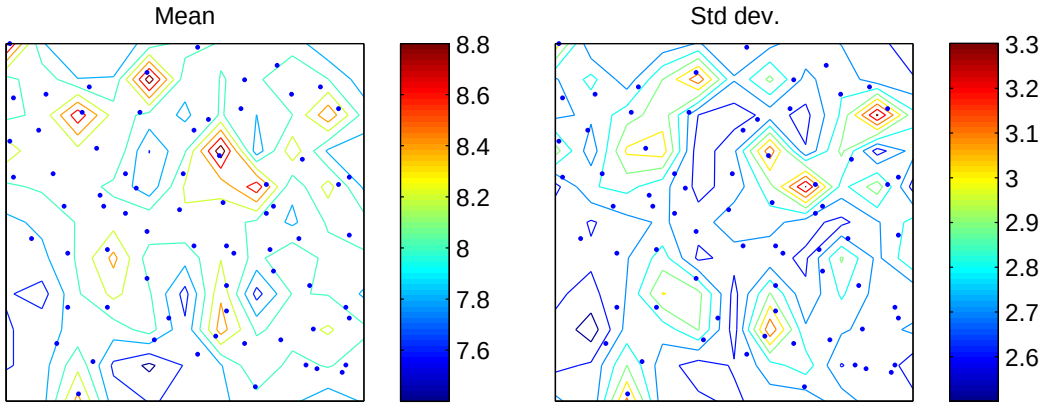

Figure 8.10: Posterior mean (left) and standard deviation (right) of the rate of the primary process of the Swedish pine tree dataset

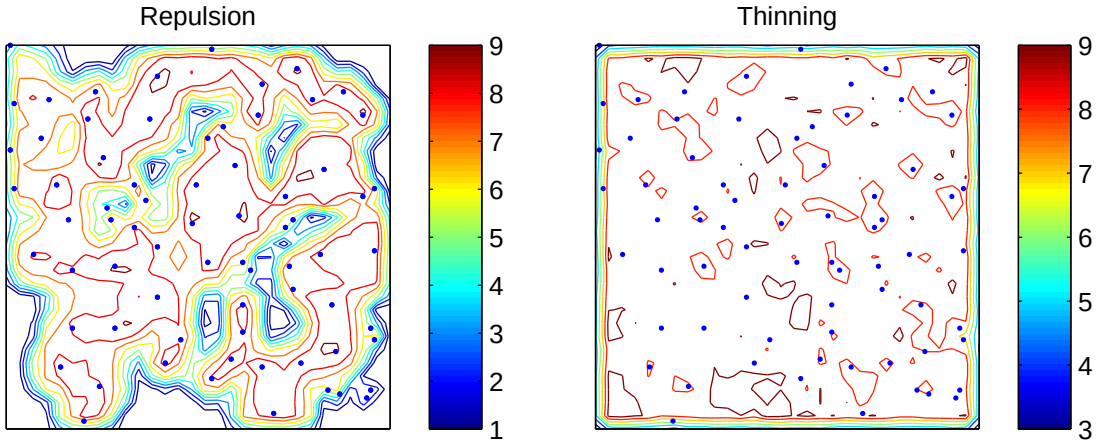

Figure 8.11: Left: Posterior rate of deletions due to repulsion, and right: Posterior rate of deletions in sampling the primary process of the Swedish pine tree datasets

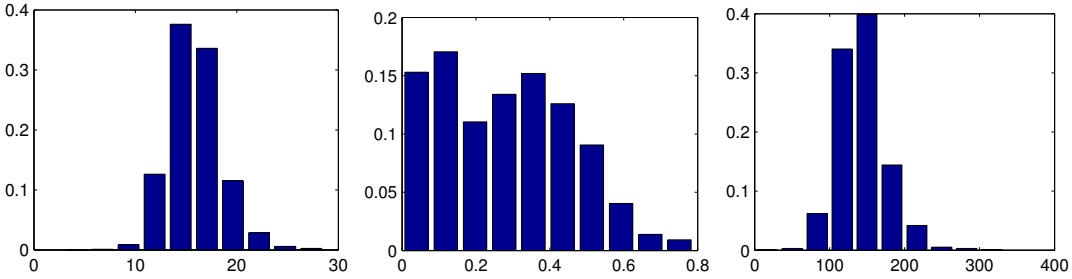

Figure 8.12: Swedish pine tree dataset: Posterior distribution of Matérn intensity (left), radius (centre) as well as the number of thinned events (right).

	Redwood dataset	Swedish tree dataset
Average Matérn interaction radius	314.13	370.01
Latent times (averaged across observations)	994.86	988.18
Primary Poisson GP intensity (evaluated on a grid)	52.6	627.19

Table 8.2: Effective sample sizes (per 1000 samples) for the inhomogeneous Matérn type-III softcore model

for Gaussian process (Murray et al., 2010); the larger number for the Swedish dataset reflects the fact that it is relatively homogeneous.

8.5 Discussion

In this chapter, we described how to perform MCMC inference for the Matérn type-III point process, a class of repulsive point processes generated by thinning a primary Poisson process. Continuing with the theme of this thesis, our scheme exploits the independence properties of the the Poisson process to jointly resample thinned events given the Matérn events. Having reconstructed the latent Poisson process, we then resample the remaining variables. We showed how our sampler easily generalizes to more complicated extensions of the Matérn type-III process, and as an example, studied an inhomogeneous softcore Matérn type-III process. The efficiency of our sampler stems from its ability to jointly resample the various sets of thinned events. The variables associated with the Matérn events (such as the birth times T^G and the interaction radii R^G of the Matérn events) were updated sequentially. Though we found this to perform adequately, it is worth attempting to devise more global moves for these quantities. Similarly, we saw that the posterior distributions of the thinned events in Matérn type-I and II processes is not Poisson. Using our approach of characterizing the posteriors with probability densities, it is interesting to see when we can efficiently sample the thinned events in these models (eg. by transforming a sample from a Poisson process).

In our experiments, we applied our sampler to two datasets, the Swedish and the redwood tree datasets. These modelled the distributions of events in a 2-dimensional space, though it is easy to generalize to higher dimensions. This could be useful to model, say, the distribution of galaxies in space, features in some feature space etc.

Our experiments assumed the Matérn events G were observed perfectly, though we can also introduce noise into this observation process. In this case, given the observed point process G_{obs}, we have to instantiate the latent Matérn process G. Given this, we can resample all other variables as outlined previously. However, resampling the locations of the events in G would require incremental updates, and if we allow for missing or extra events, we would need a birth-death sampler as well. A direction for future study to see how these moves can be made efficiently.

The use of a GP prior on the intensity function means that like chapter 5, we will not be able to scale our inhomogeneous model to problems with a very large number of Matérn events. In such situations, we must look at approximate inference methods for GP inference, or using alternate, more tractable priors to model nonstationarity.

Chapter 9

Summary and future work

In this thesis, we described a framework for exact MCMC inference in continuous-time discrete-state systems. At a high level, our approach builds upon a thinning construction for continuous-time systems where candidate events are generated at rates higher than those in the system. Given such a construction, we proceed by first resampling the thinned events given the current trajectory of the system. The new set of thinned events, along with the actual events of the old trajectory, results in a random discretization of time, and given this, we sample a new trajectory of the system. This results in an auxiliary variable Gibbs sampler with the correct stationary distribution. We showed how one can expoit independence properties of the Poisson process to perform the first step efficiently. We can leverage MCMC sampling techniques from the discrete-time literature like the forward-backward algorithm to perform the second step efficiently.

The first application of our sampler that we considered was inference in Markov jump processes, where we achieved state-of-the-art performance on a number of models based on the MJP. We then showed how our sampler can be extended to related models like renewal and semi-Markov processes, as well as spatial processes like Matérn type-III processes (which have a latent, unobserved temporal ordering).

Conceptually, the idea of randomly discretizing time is related to ideas from Andrieu and Roberts (2009). There, the authors describe and analyze a general framework for exact Metropolis-Hastings sampling, where an intractable acceptance ratio is replaced by an unbiased estimate. Particle MCMC is a particular construction of such a 'pseudo-marginal', applicable to time-series data. Our idea of a random time-discretization is an other construction, now relevant to continuous-time models. Of course, our sampler is an auxiliary variable Gibbs sampler; however, it can be adapted to produce an exact Metropolis-Hastings sampler as well. It is then worth investigating how our ideas can be combined with ideas from particle MCMC and Andrieu and Roberts (2009) to construct samplers with improved mixing properties. Similarly, such an approach would help understand our sampler better theoretically; all our evaluations in this thesis have been empirical. A concrete example of such theoretical problem is the choice of the

subordinating Poisson rate Ω for uniformization sampler of chapter 3. This parameter trades of mixing and computational cost, and it is of interest to be able to quantify its effect in a precise way. Similarly, we saw in chapter 7, how to reduce the number of thinned events by allowing state-dependent dominating event rates. This too comes at the price of slower mixing, and it would be interesting to study this theoretically.

The idea of a Poisson discretization of time is closely connected to work on the exact sampling for diffusions; for instance, Beskos and Roberts (2005) describe a rejection sampling scheme to sample from stochastic differential equations based on this idea. It is thus worth seeing if our ideas can be extended to stochastic processes with continuous state spaces. Similarly, we can look at whether our ideas are applicable to jump diffusion processes (Casella and Roberts, 2011).

An application of our sampler that we have not discussed in this thesis is from (Teh et al., 2011), Here, our uniformization-based sampler from chapter 3 was used for inference in a class of Bayesian nonparametric models called Fragmentation Coagulation processes; these were applied to problems in population genetics. Our sampler is also applicable to other nonparametric models such models on trees (viz. the Dirichlet diffusion tree (Neal, 2003b), and Kingman's coalescent (Kingman, 1982; Teh et al., 2008),these are respectively fragmentation and coagulation processes in continuous time), as well as infinite-state Markov jump processes (Saeedi and Bouchard-Côté, 2011). Closely related to ideas from chapter 5 is the problem of survival analysis, and we can apply our ideas to the problem of nonparametric hazard function estimation (Berliner and Hill, 1988; Nieto-Barajas and Walker, 2010). Another application is to continuous-time Markov decision processes (Guo and Hernández-Lerma, 2009) (or controlled MJPs).

Bibliography

R. P. Adams. *Kernel methods for nonparametric Bayesian inference of probability densities and point processes*. PhD thesis, University of Cambridge, 2009. (pages 129, 136, 137, and 138)

R. P. Adams, I. Murray, and D. J. C. MacKay. Tractable nonparametric Bayesian inference in Poisson processes with Gaussian process intensities. In *Proceedings of the 26th International Conference on Machine Learning (ICML)*, 2009. (pages 14, 27, 71, 76, 77, 80, and 89)

C. Andrieu and G. O. Roberts. The pseudo-marginal approach for efficient Monte Carlo computations. *The Annals of Statistics*, 37(2):697–725, Apr. 2009. ISSN 0090-5364. (pages 129 and 145)

C. Andrieu and J. Thoms. A tutorial on adaptive MCMC. *Statistics and Computing*, 18(4):343–373, Dec. 2008. ISSN 0960-3174. (page 54)

C. Andrieu, A. Doucet, and R. Holenstein. Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society Series B*, 72(3):269–342, 2010. (pages 107, 108, and 109)

S. Asmussen. *Applied Probability and Queues*. Applications of Mathematics. Springer, 2003. ISBN 9780387002118. (pages 40 and 118)

A. Baddeley and R. Turner. Spatstat: an R package for analyzing spatial point patterns. *Journal of Statistical Software*, 12(6):1–42, 2005. ISSN 1548-7660. (page 133)

J. Berger and D. Sun. Bayesian analysis for the poly-Weibull distribution. *J. Amer. Statist. Assoc.*, 88:1412–1418, 1993. (page 113)

L. M. Berliner and B. M. Hill. Bayesian nonparametric survival analysis. *Journal of the American Statistical Association*, 83(403):pp. 772–779, 1988. ISSN 01621459. URL http://www.jstor.org/stable/2289303. (page 146)

M. Berman. Inhomogeneous and modulated gamma processes. *Biometrika*, 68(1):143, 1981. (page 77)

M. Berman and T. R. Turner. Approximating point process likelihoods with GLIM. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 41(1):pp. 31–38, 1992. ISSN 00359254. (page 77)

J. Bertoin. *Random fragmentation and coagulation processes*, volume 102 of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, Cambridge, 2006. (pages 20 and 21)

A. Beskos and G. Roberts. Exact simulation of diffusions. *Annals of applied probability*, 15(4):2422 – 2444, November 2005. (page 146)

G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi. *Queueing networks and Markov chains: modeling and performance evaluation with computer science applications*. Wiley-Interscience, New York, NY, USA, 1998. ISBN 0-471-19366-6. (page 56)

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, 2008. (page 41)

L. Breuer. *From Markov jump processes to spatial queues*. Springer, 2003. ISBN 978-1-4020-1104-7. (pages 32 and 40)

E. N. Brown, R. Barbieri, V. Ventura, R. E. Kass, and L. M. Frank. The time-rescaling theorem and its application to neural spike train data analysis. *Neural computation*, 14(2):325–46, Feb. 2002. ISSN 0899-7667. (page 77)

T. Burzykowski, J. Szubiakowski, and T. Rydén. Analysis of photon count data from single-molecule fluorescence experiments. *Chemical Physics*, 288(2-3):291–307, 2003. (page 50)

C. K. Carter and R. Kohn. Markov chain Monte Carlo in conditionally Gaussian state space models. *Biometrika*, 83:589–601, 1996. (page 13)

B. Casella and G. O. Roberts. Exact simulation of jump-diffusion processes with Monte Carlo applications. *Methodology and Computing in Applied Probability*, 13(3):449–473, 2011. (page 146)

E. Çinlar. *Introduction to Stochastic Processes*. Prentice Hall, 1975. (pages 31 and 35)

I. Cohn, T. El-Hay, N. Friedman, and R. Kupferman. Mean field variational approximation for continuous-time Bayesian networks. *J. Mach. Learn. Res.*, 11:2745–2783, December 2010. ISSN 1532-4435. (page 41)

D. Cox. The statistical analysis of dependencies in point processes. In P. Lewis, editor, *Stochastic point processes*, pages 55–56. New York: Wiley 1972, 1972. (pages 74 and 77)

D. R. Cox. Some Statistical Methods Connected with Series of Events. *Journal of the Royal Statistical Society. Series B (Methodological)*, 17(2):129–164, 1955. (page 50)

J. P. Cunningham, B. M. Yu, K. V. Shenoy, and M. Sahani. Inferring neural firing rates from spike trains using Gaussian processes. In *Advances in Neural Information Processing Systems 20*, 2008. (pages 72, 77, 78, 84, and 86)

D. J. Daley and D. Vere-Jones. *An Introduction to the Theory of Point Processes*. Springer, 2008. (pages 16, 20, 21, 23, and 125)

M. Dewar, C. Wiggins, and F. Wood. Inference in hidden Markov models with explicit state duration distributions. *IEEE Signal Processing Letters*, page To Appear, 2012. (pages 13 and 112)

V. Didelez. Graphical models for marked point processes based on local independence. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(1): 245–264, 2008. ISSN 1467-9868. (page 56)

A. Doucet, N. de Freitas, and N. J. Gordon. *Sequential Monte Carlo Methods in Practice*. Statistics for Engineering and Information Science. New York: Springer-Verlag, May 2001. (page 12)

T. El-Hay, N. Friedman, and R. Kupferman. Gibbs sampling in factorized continuous-time Markov processes. In *UAI*, pages 169–178, 2008. (pages 41, 60, 61, 65, 66, 67, and 68)

R. Elliott and C.-J. Osakwe. Option pricing for pure jump processes with Markov switching compensators. *Finance and Stochastics*, 10:250–275, 2006. (page 40)

Y. Fan and C. Shelton. Learning continuous-time social network dynamics. In *Proceedings of the Twenty-Fifth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-09)*, pages 161–168, Corvallis, Oregon, 2009. AUAI Press. (page 31)

Y. Fan and C. R. Shelton. Sampling for approximate inference in continuous time Bayesian networks. In *Tenth International Symposium on Artificial Intelligence and Mathematics*, 2008. (pages 41 and 46)

P. Fearnhead and C. Sherlock. An exact Gibbs sampler for the Markov-modulated Poisson process. *Journal Of The Royal Statistical Society Series B*, 68(5):767–784, 2006. (pages 31, 40, 41, 42, 48, 50, 51, 52, 53, and 61)

W. Feller. On semi-Markov processes. *Proceedings of the National Academy of Sciences of the United States of America*, 51(4):pp. 653–659, 1964. ISSN 00278424. (page 93)

Früwirth-Schnatter. Data augmentation and dynamic linear models. *J. Time Ser. Anal.*, 15:183–202, 1994. (page 13)

A. Gelman, S. Brooks, G. Jones, and X. Meng. *Handbook of Markov Chain Monte Carlo*. Chapman & Hall/CRC Handbooks of Modern Statistical Methods. CRC Press, 2010. ISBN 9781420079418. (page 12)

I. Gerhardt and B. L. Nelson. Transforming renewal processes for simulation of non-stationary arrival processes. *INFORMS Journal on Computing*, 21(4):630–640, Apr. 2009. ISSN 1091-9856. (page 77)

I. I. Gikhman and A. V. Skorokhod. *The theory of stochastic processes. II. Translated from the Russian by S. Kotz. Corrected printing of the first edition.* Classics in Mathematics. Berlin: Springer. viii, 2004. (pages 31 and 33)

W. R. Gilks, S. Richardson, and D. J. Spiegelhalter. *Markov Chain Monte Carlo in Practice*. Chapman and Hall, 1996. (page 12)

D. T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *J. Phys. Chem.*, 81(25):2340–2361, 1977. ISSN 0022-3654. (pages 31, 35, and 40)

L. Glass and W. R. Tobler. Uniform distribution of objects in a homogeneous field: Cities on a plain. *Nature*, 233(5314):67–68, June 1971. (page 124)

A. Golightly and D. J. Wilkinson. Bayesian parameter inference for stochastic biochemical network models using particle Markov chain Monte Carlo. *Interface Focus*, 1(6):807–820, December 2011. (pages 40, 46, and 109)

X. Guo and O. Hernández-Lerma. *Continuous-Time Markov Decision Processes: Theory and Applications*. Stochastic Modelling and Applied Probability. Springer, 2009. ISBN 9783642025464. URL http://books.google.co.uk/books?id=tgi-opMlLTwC. (page 146)

F. H. Hansford-Miller. *A geographical and statistical analysis of religious nonconformity in England and Wales*. PhD thesis, University College London, 1968. (page 124)

M. O. Hill. The intensity of spatial pattern in plant communities. *Journal of Ecology*, 61(1):pp. 225–235, 1973. ISSN 00220477. (page 124)

A. Hobolth and E. A. Stone. Simulation from endpoint-conditioned, continuous-time Markov chains on a finite state space, with applications to molecular evolution. *Ann Appl Stat*, 3(3):1204, 2009. ISSN 1941-7330. (pages 35, 37, 41, 42, 46, and 47)

M. L. Huber and R. L. Wolpert. Likelihood-based inference for Matérn type III repulsive point processes. *Advances in Applied Probability*, 41(4), 2009. 958–977. (pages 129, 130, 131, and 136)

B. Y. R. G. Jarrett. A note on the intervals between coal-mining disasters. *Biometrika*, 66(1):191–193, 1979. (pages 88 and 89)

A. Jensen. Markoff chains as an aid in the study of Markoff processes. *Skand. Aktuarietiedskr.*, 36:87–91, 1953. (pages 13, 35, and 37)

C. P. Jewell, T. Kypraios, P. Neal, and G. O. Roberts. Bayesian Analysis for Emerging Infectious Diseases. *Bayesian Analysis*, 4(4):465–496, 2009. (page 124)

O. Kallenberg. *Foundations of Modern Probability*. Probability and its Applications. Springer-Verlag, New York, Second edition, 2002. ISBN 0-387-95313-2. (pages 24, 34, and 37)

R. E. Kass and V. Ventura. A spike-train probability model. *Neural Computation*, 13 (8):1713–1720, 2001. (pages 72, 74, and 77)

D. G. Kendall. Stochastic Processes Occurring in the Theory of Queues and their Analysis by the Method of the Imbedded Markov Chain. *The Annals of Mathematical Statistics*, 24(3):338–354, 1953. ISSN 00034851. (page 118)

M. G. Kendall. The geographical distribution of crop productivity in england. *Journal of the Royal Statistical Society*, 102(1):21–62, 1939. ISSN 09528385. (page 124)

J. F. C. Kingman. The coalescent. *Stochastic Processes and their Applications*, 13: 235–248, 1982. (page 146)

J. F. C. Kingman. *Poisson processes*, volume 3 of *Oxford Studies in Probability*. The Clarendon Press Oxford University Press, New York, 1993. ISBN 0-19-853693-3. Oxford Science Publications. (pages 16, 17, 18, 19, and 20)

G. Knox. Epidemiology of childhood leukemia in northumberland and durham. *The Challenge of Epidemiology: Issues and Selected Readings*, 1(1):384–392, Jan 2004. (page 124)

J. F. Lawless and K. Thiagarajah. A point-process model incorporating renewals and time trends, with application to repairable systems. *Technometrics*, 38(2):131–138, 1996. (pages 71 and 77)

P. A. W. Lewis and G. S. Shedler. Simulation of nonhomogeneous Poisson processes with degree-two exponential polynomial rate function. *Operations Research*, 27(5): 1026–1040, Sept. 1979. ISSN 0030-364X. (page 25)

B. H. Lindqvist. Nonparametric estimation of time trend for repairable systems data. In V. Rykov, N. Balakrishnan, and M. Nikulin, editors, *Mathematical and Statistical Models and Methods in Reliability*, Statistics for Industry and Technology, pages 277–288. Birkhuser Boston, 2011. ISBN 978-0-8176-4971-5. (page 77)

B. Matérn. *Spatial Variation*. Springer-Verlag Berlin and Heidelberg GmbH & Co. K, second edition, 1986. ISBN 3540963650. (page 126)

J. Mateu and F. Montes. Likelihood inference for Gibbs processes in the analysis of spatial point patterns. *International Statistical Review / Revue Internationale de Statistique*, 69(1):pp. 81–104, 2001. ISSN 03067734. (page 126)

J. M. McFarland, T. T. G. Hahn, and M. R. Mehta. Explicit-Duration Hidden Markov Model Inference of UP-DOWN States from Continuous Signals. *PLoS ONE*, 6(6), June 2011. (page 55)

J. Medhi. *Stochastic Models in Queueing Theory, Second Edition*. Academic Press, 2 edition, Nov. 2002. ISBN 0124874622. (page 118)

C. J. Mode and G. T. Pickens. Computational methods for renewal theory and semi-Markov processes with illustrative examples. *The American Statistician*, 42(2):pp. 143–152, 1988. ISSN 00031305. (page 55)

J. Møller and R. P. Waagepetersen. Modern Statistics for Spatial Point Processes. *Scandinavian Journal of Statistics*, 34(4):643–684, Dec. 2007. ISSN 0303-6898. (page 126)

J. Møller, M. L. Huber, and R. L. Wolpert. Perfect simulation and moment properties for the Matérn type III process. *Stochastic Processes and their Applications*, 120(11): 2142–2158, 2010. (pages 128, 129, and 136)

K. P. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, University of California, Berkeley, 2002. (pages 12 and 57)

I. Murray and R. P. Adams. Slice sampling covariance hyperparameters of latent Gaussian models. In *Advances in Neural Information Processing Systems 23*, 2010. (pages 83 and 139)

I. Murray, R. P. Adams, and D. J. MacKay. Elliptical slice sampling. *JMLR: W&CP*, 9, 2010. (pages 83, 138, 139, and 143)

R. M. Neal. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, Department of Computer Science, U of Toronto, 1993. (page 83)

R. M. Neal. Slice sampling. *Annals of Statistics*, 31:705–767, 2003a. (pages 55 and 119)

R. M. Neal. Density modeling and clustering using Dirichlet diffusion trees. In *Bayesian Statistics*, volume 7, pages 619–629, 2003b. (page 146)

R. M. Neal, M. J. Beal, and S. T. Roweis. Inferring state sequences for non-linear systems with embedded hidden Markov models. In *Advances in Neural Information Processing Systems 16 (NIPS)*, volume 16, pages 401–408. MIT Press, 2004. (page 13)

R. Nielsen. Mapping mutations on phylogenies. *Syst Biol*, 51(5):729–739, 2002. (page 46)

L. E. Nieto-Barajas and S. G. Walker. Bayesian nonparametric survival analysis via levy driven markov processes. *Statistica Sinica*, 2010. URL http://kar.kent.ac.uk/10538/. (page 146)

U. Nodelman and E. Horvitz. Continuous time Bayesian networks for inferring users presence and activities with extensions for modeling and evaluation. Technical Report MSR-TR-2003-97, Microsoft Research, 2003. (page 31)

U. Nodelman, C. Shelton, and D. Koller. Continuous time Bayesian networks. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 378–387, 2002. (pages 41, 56, 58, and 60)

U. Nodelman, D. Koller, and C. Shelton. Expectation propagation for continuous time Bayesian networks. In *Proceedings of the Twenty-first Conference on Uncertainty in AI (UAI)*, pages 431–440, July 2005. (page 41)

Y. Ogata. On Lewis' simulation method for point processes. *IEEE Transactions on Information Theory*, 27(1):23–31, 1981. (pages 77, 78, 79, and 90)

M. Opper and G. Sanguinetti. Variational inference for Markov jump processes. In *Advances in Neural Information Processing Systems 20*, 2007. (pages 41, 65, and 66)

L. Paninski, J. Pillow, and J. Lewi. Statistical models for neural encoding, decoding, and optimal stimulus design. *Progress in brain research*, 165:493, 2007. (page 91)

T. Parsons. Earthquake recurrence on the south Hayward fault is most consistent with a time dependent, renewal process. *Geophysical Research Letters*, 35, 2008. (page 72)

V. Paxson and S. Floyd. Wide area traffic: the failure of Poisson modeling. *IEEE/ACM Transactions on Networking*, 3(3):226–244, June 1995. ISSN 10636692. (page 72)

P. J. E. Peebles. The nature of the distribution of galaxies. *Astronomy and Astrophysics*, 32:197, May 1974. (page 124)

M. Plummer, N. Best, K. Cowles, and K. Vines. CODA: Convergence diagnosis and output analysis for MCMC. *R News*, 6(1):7–11, March 2006. (pages 49, 90, and 108)

L. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77:257–285, 1989. (page 12)

V. Rao and Y. W. Teh. Spatial normalized gamma processes. In *Advances in Neural Information Processing Systems*, 2009. (page 21)

V. Rao and Y. W. Teh. Fast MCMC sampling for Markov jump processes and continuous time Bayesian networks. In *Proceedings of the International Conference on Uncertainty in Artificial Intelligence*, 2011a. (page 14)

V. Rao and Y. W. Teh. Gaussian process modulated renewal processes. In *Advances in Neural Information Processing Systems 23*, 2011b. (page 14)

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006. (pages 76 and 84)

B. D. Ripley. Modelling spatial patterns. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(2):pp. 172–212, 1977. ISSN 00359246. (page 133)

B. D. Ripley. *Statistical inference for spatial processes / B.D. Ripley*. Cambridge University Press, Cambridge [England] ; New York, 1988. (page 133)

C. P. Robert and G. Casella. *Monte Carlo Statistical Methods (Springer Texts in Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005. ISBN 0387212396. (page 12)

N. Rodrigue, H. Philippe, and N. Lartillot. Uniformization for sampling realizations of Markov processes: applications to Bayesian implementations of codon substitution models. *Bioinformatics*, 24(1):56–62, 2008. (page 40)

A. Rokem, S. Watzl, T. Gollisch, M. Stemmler, and A. V. Herz. Spike-Timing Precision Underlies the Coding Efficiency of Auditory Receptor Neurons. *Journal of Neurophysiology*, pages 2541–2552, 2006. (page 89)

A. Saeedi and A. Bouchard-Côté. Priors over Recurrent Continuous Time Processes. In *Advances in Neural Information Processing Systems 24 (NIPS)*, volume 24, 2011. (page 146)

I. Sahin. A generalization of renewal processes. *Operations Research Letters*, 13(4): 259–263, May 1993. ISSN 01676377. (page 77)

S. L. Scott and P. Smyth. The Markov modulated Poisson process and Markov Poisson cascade with applications to web traffic modeling. *Bayesian Statistics*, 7:1–10, 2003. (page 51)

J. G. Shanthikumar. Uniformization and hybrid simulation/analytic models of renewal processes. *Oper. Res.*, 34:573–580, July 1986. ISSN 0030-364X. (page 78)

C. Shelton, Y. Fan, W. Lam, J. Lee, and J. Xu. Continuous time Bayesian network reasoning and learning engine, 2010. (page 66)

I. Slivnyak. Some properties of stationary flows of homogeneous random events. *Theory Probab. Appl.*, 7:336–341, 1962. (page 21)

D. Sonderman. Comparing semi-Markov processes. *Mathematics of Operations Research*, 5(1):110–119, 1980. (pages 93 and 97)

L. Strand. A model for stand growth. In *IUFRO Third Conference Advisory Group of Forest Statisticians, INRA, Institut National de la Recherche Agronomique, Paris.*, pages 207–216, 1972. (page 124)

Y. W. Teh, H. Daume III, and D. M. Roy. Bayesian agglomerative clustering with coalescents. In *Advances in Neural Information Processing Systems*, volume 20, 2008. (page 146)

Y. W. Teh, C. Blundell, and L. T. Elliott. Modelling genetic variations with fragmentation-coagulation processes. In *Advances In Neural Information Processing Systems*, 2011. (page 146)

H. Tijms. *Stochastic modelling and analysis: a computational approach*. Wiley series in probability and mathematical statistics: Applied probability and statistics. Wiley, 1986. ISBN 9780471909118. (page 40)

J. Van Gael, Y. Saatci, Y. W. Teh, and Z. Ghahramani. Beam sampling for the infinite hidden Markov model. In *Proceedings of the International Conference on Machine Learning*, volume 25, 2008. (page 13)

S. G. Walker. Sampling the Dirichlet mixture model with slices. *Communications in Statistics - Simulation and Computation*, 36:45, 2007. ISSN 0361-0918. (page 119)

D. J. Wilkinson. Stochastic modelling for quantitative description of heterogeneous biological systems. *Nature Reviews Genetics*, 10:122–133, 2009. (pages 56 and 65)

C. Wu. Counting your customers: Compounding customer's in-store decisions, interpurchase time and repurchasing behavior. *European Journal of Operational Research*, 127(1):109–119, Nov. 2000. ISSN 03772217. (page 72)

S. Zuyev. Strong Markov property of Poisson processes and Slivnyak formula. In *Lecture Notes in Statistics*, pages 77–84. Springer, 2006. (page 21)