Kernel Methods for Nonparametric Bayesian Inference of Probability Densities and Point Processes

Ryan Prescott Adams
S.B., Massachusetts Institute of Technology (2004)

St. John's College
University of Cambridge

This dissertation is submitted for the degree of
Doctor of Philosophy

Inference Group
Cavendish Laboratory
University of Cambridge

2009

Declaration

I hereby declare that my dissertation entitled "Kernel Methods for Nonparametric Bayesian Inference of Probability Densities and Point Processes" is not substantially the same as any that I have submitted for a degree or diploma or other qualification at any other University.

I further state that no part of my dissertation has already been or is being concurrently submitted for any such degree or diploma or other qualification.

This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration, except where specifically indicated in the text. This dissertation does not exceed sixty thousand words in length.

Abstract

Nonparametric kernel methods for estimation of probability densities and point process intensities have long been of interest to researchers in statistics and machine learning. Frequentist kernel methods are widely used, but provide only a point estimate of the unknown density. Additionally, in frequentist kernel density methods, it can be difficult to select appropriate kernel parameters. The Bayesian approach to inference potentially resolves both of these deficiencies, by providing a distribution over the unknowns and enabling a principled approach to kernel selection. Constructing a Bayesian nonparametric kernel density method has proven to be difficult, however, due to the need to integrate over an infinite-dimensional random function in order to evaluate the likelihood. To avoid this intractability, all Bayesian kernel density methods to date have either used a crippled model or a finite-dimensional approximation.

Recent advances in Markov chain Monte Carlo methods have improved the situation for these *doubly-intractable* posterior distributions, however. If data can be generated exactly from the model, then it is possible to perform inference without computing the intractable likelihood. I propose two new kernel-based models that enable an exact generative procedure: the Gaussian process density sampler (GPDS) for probability density functions, and the sigmoidal Gaussian Cox process (SGCP) for the Poisson process. With generative priors, I show how it is now possible to construct two different kinds of Markov chains for inference in these models. These Markov chains have the desired posterior distribution as their equilibrium distributions, and, despite a parameter space with uncountably many dimensions, require only a finite amount of computation to simulate. The GPDS and SGCP, and the associated inference procedures, are the first kernel-based nonparametric Bayesian methods that allow inference without a finite-dimensional approximation.

I also present several additional kernel-based models for data that extend the Gaussian process density sampler and sigmoidal Gaussian Cox process to other situations. The *Archipelago* model extends the GPDS to address the task of semi-supervised learning, where a flexible density estimate can improve the performance of a classifier when unlabeled data are available. I also generalise the SGCP to enable a nonparametric inhomogeneous Neyman–Scott process, and present a soft-core generalisation of the Matérn repulsive process that similarly allows non-approximate inference via Markov chain Monte Carlo.

To my grandmothers, Lois and LaDel.

Acknowledgements

I am extremely fortunate to have David MacKay as my supervisor and friend. It would be no understatement to say that he has taught me how to think. More than that, though, he has taught me how to communicate ideas and reason clearly. There is still a long way to go, however, and I struggle every day to live up to the intellectual standard he sets.

Zoubin Ghahramani has always made time for me and has been an invaluable source of ideas and advice on all aspects of research and academia. I enjoyed immensely my visits with him at Carnegie Mellon when we were both in Pennsylvania.

I also wish to thank Steve Gull, Leslie Pack Kaelbling, and Christopher Sawyer-Lauçanno, mentors who have given me their time and advice, without which I could not have reached this point.

I am grateful to my contemporaries in the Inference Group — Chris Ball, Phil Cowans, Oliver Stegle, David Stern, Philip Sterne, Keith Vertanen, Hanna Wallach, Seb Wills — for all of their help over the years. I appreciate their patience as I bombarded them with ridiculous ideas, and I benefited from the ideas I got in return. Among this group I would especially like to thank Iain Murray, whose keen insight provided the foundation upon which much of this thesis was built.

I have benefited greatly from the generosity of the Gates Cambridge Trust and St. John's College. I also wish to thank the Canadian Institute for Advanced Research, who funded my travels to Toronto and made possible some of the work that led to this thesis.

I wish to thank my wonderful friends in Cambridge who have given me fond memories for a lifetime.

My parents, John and Cathy, my brother Vincent and my sister Katherine have enthusiastically supported every one of my ambitions, and this PhD was no different. My family has given me opportunities that few people ever have, and I will always be grateful.

Finally, there is nothing I could do to sufficiently express my gratitude to Brenda. She has been supportive at every turn of a very twisty road. She's proofread my papers, listened to my practice talks, and raided the library on my behalf. Not a single word of this thesis would exist without her.

Contents

List of Figures

List of Tables

List of Algorithms

Mathematical Conventions

Notation

In this thesis, I will generally use lowercase letters, such as x, to denote scalar values, bold lowercase, such as $\boldsymbol{x}$, to denote vectors, and uppercase bold, such as $\boldsymbol{X}$, to denote matrices. When referring to specific elements within vectors and matrices I will use subscripts paired with unbolded notation, such as x_j for the jth component of $\boldsymbol{x}$ or X_{ij} for the entry of $\boldsymbol{X}$ in row i and column j. When referring to a *specific* matrix or vector, I will often use bold lettering with a subscript, for example $\boldsymbol{K}_N$ or $\boldsymbol{k}_M$. I will denote the transpose of $\boldsymbol{X}$ as $\boldsymbol{X}^\mathsf{T}$. I will generally use uppercase calligraphic characters, such as $\mathcal{X}$ to refer to spaces. When referring to functions in general, I will use the normal notation $f(\cdot)$, however, much of this thesis is concerned with functions as infinite-dimensional vectors on which it is possible to construct distributions. In these cases, I will use the vector notation, such as $\boldsymbol{f}$.

In algorithms I will use the leftarrow $\leftarrow$ to indicate assignment and the tilde $\sim$ to indicate that a quantity is drawn from a particular distribution. Within functions, I will use $f(a\,;\,b)$ to indicate a function with input a parameterised by b. If I use the notation $p(a \mid b)$, I am indicating a probability distribution on a conditioned on b. I will denote a Markov chain Monte Carlo proposal to transition from a to $\hat{a}$ with $q(\hat{a} \leftarrow a)$. In general, I will use "hatted" variables such as $\hat{a}$ to indicate Markov chain Monte Carlo proposals, and "tilded" variables such as $\tilde{a}$ to indicate rejection sampling proposals.

The notation $\mathbb{R}$ refers to the extended real numbers; $\mathbb{R}^+$ denotes the nonnegative extended real numbers. The notation $\mathbb{N}$ refers to the natural numbers, including zero.

For a comprehensive list of symbols used in this thesis, see the Symbol Glossary on page 126.

Common Distributions

There are several distributions to which I will frequently refer.

Uniform:

$$\mathcal{U}(x \in \mathbb{R} \,|\, a, b) = \begin{cases} \frac{1}{b-a} & \text{if } a < x < b \\ 0 & \text{otherwise} \end{cases}$$

$$\mathcal{U}(\boldsymbol{x} \in \mathbb{R}^D \,|\, \mathcal{V}) = \begin{cases} \frac{1}{\mu(\mathcal{V})} & \text{if } \boldsymbol{x} \in \mathcal{V} \\ 0 & \text{otherwise} \end{cases}$$

Gaussian (Normal):

$$\mathcal{N}(\boldsymbol{x} \in \mathbb{R}^D \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = (2\pi)^{-\frac{d}{2}} |\boldsymbol{\Sigma}|^{-\frac{1}{2}} \exp\left\{ -\frac{1}{2} (\boldsymbol{x} - \boldsymbol{\mu})^\mathsf{T} \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu}) \right\}$$

Gamma:

$$\mathcal{GA}(x \in \mathbb{R}^+ \,|\, \alpha, \beta) = \frac{1}{\beta^\alpha \, \Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}$$

Poisson:

$$\mathcal{PO}(k \in \mathbb{N} \,|\, \lambda) = \frac{1}{k!} e^{-\lambda} \lambda^k$$

Exponential:

$$\mathcal{EX}(x \in \mathbb{R}^+ \,|\, \lambda) = \lambda \, e^{-\lambda x}$$

Chapter 1

Introduction

Nonparametric kernel methods for estimation of probability densities and point process intensities have long been of interest to researchers in statistics and machine learning. Frequentist approaches to kernel estimation of probability densities were introduced by Rosenblatt (1956) and Parzen (1962), and extended to point processes by Diggle (1985). These methods are widely used, but provide only a point estimate of the unknown density. Additionally, in frequentist kernel density methods, it can be difficult to select appropriate kernel parameters. The Bayesian approach to inference potentially resolves both of these deficiencies, by providing a distribution over the unknowns and enabling a principled approach to kernel selection. Constructing a Bayesian nonparametric kernel density method has proven to be difficult, however, due to the need to integrate over an infinite-dimensional random function. To avoid this intractability, all Bayesian kernel density methods to date have either used a crippled model or a finite-dimensional approximation. In this thesis, I take advantage of recent developments in Markov chain Monte Carlo methods to develop the first fully-nonparametric kernel-based Bayesian methods for inference of probability density functions and point process intensity functions.

1.1 Bayesian Nonparametric Modeling

The Bayesian philosophy of inference brings many advantages to data analysis. It specifies a way to incorporate new data consistently. It allows for the representation of the uncertainty of unobserved quantities. It enables nuisance parameters to be marginalised out. Comparing models of different dimensionality, however, requires computations that can be difficult to perform. Various methods have been proposed to ease the computational burden of model selection: reversible-jump Markov chain Monte Carlo (RJMC) (Green, 1995), the Laplace approximation (Lindley, 1980; MacKay, 1992b, 1998b), the Bayesian information criterion (BIC) (Schwarz, 1978), and variational methods (e.g. Minka (2001)). In contrast to these methods, which infer di-

mensionality after seeing the data, the nonparametric Bayesian approach constructs models that have an infinite number of dimensions *a priori*. The word "nonparametric" is misleading: it is not that there are *no* parameters, it is that there are an *infinite* number of parameters.

The most popular tool for nonparametric Bayesian modeling of probability distributions is the Dirichlet process (DP) (Ferguson, 1973). Samples from the Dirichlet process and its cousins, the Pitman–Yor process (Pitman and Yor, 1997; Ishwaran and James, 2001) and the Indian buffet process (Griffiths and Ghahramani, 2006), are discrete with probability one, however. This makes the vanilla Dirichlet process unsuitable for modeling continuous random variables that require probability density functions.

To fill the gap between nonparametric priors on discrete distributions and nonparametric priors on continuous densities, the Dirichlet process is frequently used to add a countably-infinite number of parameters into a continuous model. The most popular example is the infinite mixture of parametric distributions (Escobar and West, 1995; Rasmussen, 2000). Dirichlet process mixture models (DPMMs) have also been used for point processes in time and space (Kottas and Sansó, 2007), using beta distributions for the components. Other methods for using a Dirichlet process to construct a prior that assigns mass to continuous densities include kernel convolution (Lo, 1984) and the Dirichlet diffusion tree (DFT) (Neal, 2001, 2003a). The Dirichlet process is also closely related to the Pólya tree model (Ferguson, 1974; Mauldin et al., 1992; Lavine, 1992, 1994), which can be configured to produce densities. Pólya trees are not tractable as fully-infinite models, however, and require finite-depth approximations for practical computation (Walker et al., 1999). For extensive discussion of topics related to nonparametric Bayesian models for random distributions, see Walker et al. (1999) or Ghosh and Ramamoorthi (2003).

Prior beliefs about a distribution over data are often about the probability density function — its continuity, support and smoothness properties, for example. Similarly, when constructing point process models we would like to specify directly our beliefs about the associated intensity function. There is a rich literature on incorporating prior beliefs about functions into nonparametric Bayesian regression models, using splines, neural networks and stochastic processes (e.g. DiMatteo et al. (2001), MacKay (1992a), and O'Hagan (1978)). However, as will be discussed in Section 1.3 and Section 1.4, priors on general functions have largely resisted application to modeling of densities and point process intensities, as it is difficult to construct a tractable transformation such that realisations from the prior are nonnegative and integrate to a known value.

The rest of this chapter is organised as follows: Section 1.2 gives a brief review of Gaussian processes, Section 1.3 discusses the standard approach to modeling probability density functions with Gaussian processes, and Section 1.4 examines the log Gaussian Cox process. In Section 1.5 I review a handful of other models that use Gaussian processes for modeling data, and in Section 1.6 I give an outline of the

overall thesis structure.

1.2 The Gaussian Process

The main tool I will use to specify prior beliefs about probability densities and point process intensity functions is the Gaussian process (GP). The Gaussian process provides a nonparametric distribution over functions of the form $g(\cdot) : \mathcal{X} \to \mathbb{R}$. The domain $\mathcal{X}$ is chosen based on what sort of functions the Gaussian process should support, but the range is always the real line. In this thesis, it can be assumed that $\mathcal{X}$ is the D-dimensional Euclidean space $\mathbb{R}^D$.

Use of the Gaussian process as a tool for smoothing and interpolation in the time domain has a long history, going back at least to Wiener (1949). The modern and more general approach to Gaussian processes for regression was introduced by O'Hagan (1978). Gaussian processes have also been regularly applied to problems in spatial statistics, under the name *kriging* (e.g. Cressie (1991)). The Gaussian process was made popular in the machine learning community by Neal (1994) and Williams and Rasmussen (1996). The approach of Rasmussen and Williams (2006) is the one I will largely follow.

The Gaussian process is a distribution on functions. The central idea is that if one conditions on a finite set of (input or *covariate*) values in the domain, one gets a joint Gaussian distribution over the corresponding (output or *response*) values in the range. More precisely, given an indexed subset of $\mathcal{X}$ with size N, denoted $\{\boldsymbol{x}_n \in \mathcal{X}\}_{n=1}^N$, there is an associated Gaussian distribution on $\mathbb{R}^N$. The dimensions of this Gaussian are interpreted as corresponding to the outputs $\{y_n = g(\boldsymbol{x}_n)\}_{n=1}^N$ of a random function $g(\boldsymbol{x})$. The N-dimensional Gaussian distribution is parameterised by an N-dimensional mean vector $\boldsymbol{m}_N$ and an $N \times N$ covariance matrix $\boldsymbol{C}_N$.

For the Gaussian process to make sense as a distribution on functions, there must be a consistent way to arrive at $\boldsymbol{m}_N$ and $\boldsymbol{C}_N$ for any finite N and any set $\{\boldsymbol{x}_n\}_{n=1}^N$. These parameters are generated via a *mean function* $m(\cdot) : \mathcal{X} \to \mathbb{R}$ and a *covariance function* $C(\cdot, \cdot) : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ as

$$[\boldsymbol{m}_N]_n = m(\boldsymbol{x}_n) \qquad [\boldsymbol{C}_N]_{n,n'} = C(\boldsymbol{x}_n, \boldsymbol{x}_{n'}). \tag{1.1}$$

In this thesis I will take the mean function to be zero, i.e. $m(\boldsymbol{x}) = 0, \forall \boldsymbol{x} \in \mathcal{X}$. I will denote any parameters that control $C(\cdot, \cdot)$ as $\boldsymbol{\theta}$ and refer to them as *hyperparameters*.

1.2.1 Covariance Functions

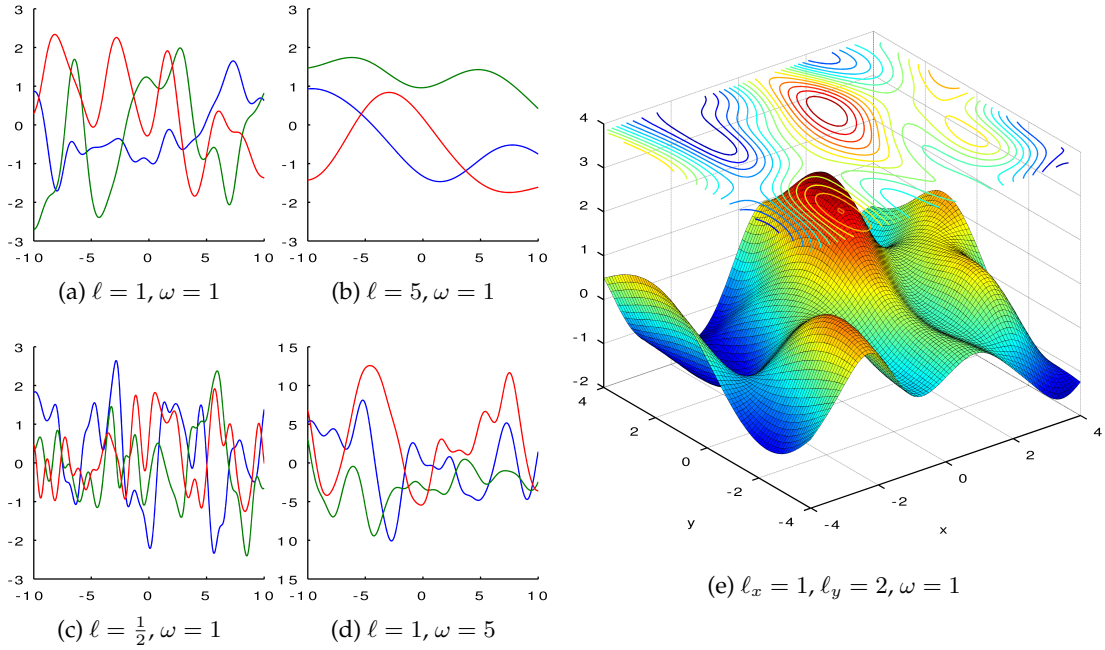
The covariance function determines what functions are typical samples from the Gaussian process. The choice of an appropriate $C(\cdot, \cdot)$ is central to specifying a prior that reflects our beliefs. There are many valid covariance functions; for reviews see Abrahamsen (1997) or Rasmussen and Williams (2006, Chapter 4). The main requirement for a covariance function is that it be a positive semidefinite kernel. Positive semidefinite kernel functions are those for which the matrix $\boldsymbol{C}_N$, as constructed in Equation 1.1, is positive semidefinite for any finite subset of $\mathcal{X}$. An $N \times N$ matrix $\boldsymbol{C}$ is positive semidefinite if the quadratic product $\boldsymbol{z}^\mathsf{T}\boldsymbol{C}\boldsymbol{z} \geq 0$ for all $\boldsymbol{z} \in \mathbb{R}^N$. Covariance matrices for Gaussian distributions must be positive definite in order for the density function to integrate to one. Positive semidefinite covariance functions, guarantee that the Gaussian distribution on the output space of the GP is always valid, regardless of what inputs are selected.

Frequently, prior beliefs about functions are very general. It might be desirable for typical functions to exhibit a particular amount of "bumpiness" and for this bumpiness to be constant across the input space. In such cases, it may be appropriate to choose a *stationary* covariance function, which depends on the input space only through a distance between the two inputs. Not only are stationary covariance functions more intuitive, but they can be constructed in Fourier space via Bochner's theorem (Gibbs, 1997). In Fourier space, choice of covariance can be guided by spectral density, rather than the interaction of distance and correlation.

Choosing an appropriate covariance function for a particular problem, whether stationary or nonstationary, can be a difficult task. Although it is an important topic, I will not address covariance function choice in this thesis. For a detailed treatment of covariance function selection see, for example, Paciorek (2003). Unless stated otherwise, I will assume the "squared exponential" covariance function (Rasmussen and Williams, 2006):

$$C(\boldsymbol{x}, \boldsymbol{x}') = \omega^2 \exp\left\{ -\frac{1}{2} \sum_{d=1}^{D} \frac{(x_d - x'_d)^2}{\ell_d^2} \right\}. \tag{1.2}$$

There are $D + 1$ hyperparameters comprising $\boldsymbol{\theta}$ in this covariance function: ω is an "amplitude" that specifies how far from zero the function outputs are likely to extend, and $\boldsymbol{\ell}$ specifies a length scale for each dimension. Figure 1.1 shows typical samples from a Gaussian process with the squared exponential covariance function in one and two dimensions.



(a) $\ell = 1, \omega = 1$ (b) $\ell = 5, \omega = 1$

(c) $\ell = \frac{1}{2}, \omega = 1$ (d) $\ell = 1, \omega = 5$

(e) $\ell_x = 1, \ell_y = 2, \omega = 1$

Figure 1.1: Typical samples from a Gaussian process with the squared exponential covariance function of Equation 1.2. (a)–(d) Three typical functions for a particular setting of the hyperparameters. (e) Two input dimensions with different length scales.

1.2.2 Interpretations and Computation

There are several different interpretations of the Gaussian process construction. One useful interpretation is to consider the Gaussian process to be a multivariate Gaussian distribution on a vector with index set $\mathcal{X}$, which may be uncountably infinite. Alternatively, one may interpret the Gaussian process as a Bayesian approach to learning with the reproducing kernel Hilbert space (RKHS) construction of Aronszajn (1950). A third interpretation of the Gaussian process is as a Bayesian regression model with a possibly-infinite number of basis functions. Similarly, Neal (1996) equates the Gaussian process to a neural network in an infinite limit. I recommend Rasmussen and Williams (2006) or MacKay (1998a) for further detail on the relationship between the Gaussian process framework and other models.

In this thesis I will generally take the view of the GP as an infinite-dimensional Gaussian distribution. This interpretation makes it easy to see how many of the computations performed with the Gaussian process can be done via simple linear algebra. For example, given some data $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$ that are to be modeled with a Gaussian process, it may be desirable to know the *marginal likelihood* to guide the choice of covariance function and hyperparameters. The marginal likelihood is the probability of the data, given the model, but integrating out the parameters. In this case the parameters are the functions, which are infinite-dimensional objects. Remarkably, in the Gaussian process it is possible to integrate over this infinite-dimensional space by simply discarding the irrelevant entries of the mean vector and covariance matrix. The log

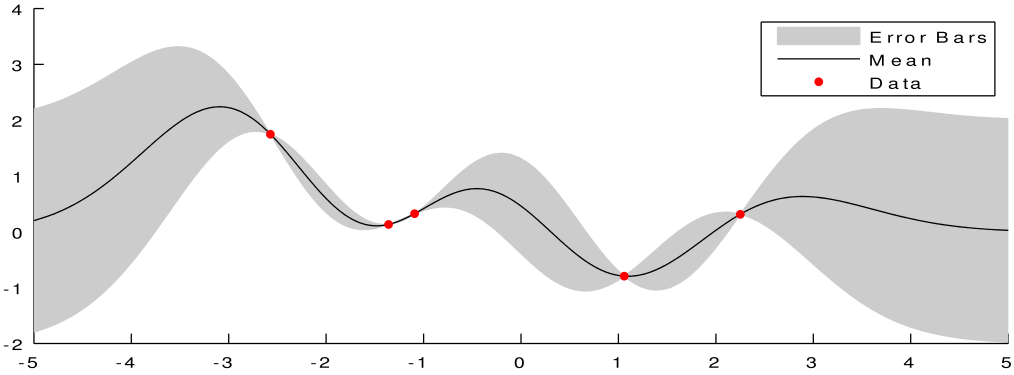


Figure 1.2: An example of a "sausage-link plot" showing the marginal predictive distribution from a Gaussian process. There are five data shown with red circles. The posterior mean function is shown in black, and the grey areas show two standard deviations of the marginal predictive distributions. This plot was generated with a squared exponential covariance function with $\ell = 1$ and $\omega = 1$.

marginal likelihood then has the simple form

$$\ln p(\{y_n\}_{n=1}^N \mid \{\boldsymbol{x}_n\}_{n=1}^N, \boldsymbol{\theta}) = -\frac{N}{2}\ln 2\pi - \frac{1}{2}\ln|\boldsymbol{C}_N| - \frac{1}{2}\boldsymbol{y}^\mathsf{T}\boldsymbol{C}_N^{-1}\boldsymbol{y} \tag{1.3}$$

where $\boldsymbol{y} = [y_1, y_2, \ldots, y_N]^\mathsf{T}$.

Having seen N data $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$, we might want to make predictions of the function values at a new set of M input locations $\{\boldsymbol{x}_m\}_{m=1}^M$. This M-dimensional conditional (or *predictive*) distribution integrates out the value of the function at all unseen locations and is Gaussian. I denote the *cross-covariance* matrix between the N data already seen and the M data at which I make predictions, as $\boldsymbol{C}_{N,M}$. This matrix is computed by applying the covariance function to the two sets via

$$[\boldsymbol{C}_{N,M}]_{n,m} = C(\boldsymbol{x}_n, \boldsymbol{x}_m). \tag{1.4}$$

Having computed this matrix, it is now possible to find the parameters of the Gaussian predictive distribution for the function values at the $\{\boldsymbol{x}_m\}_{m=1}^M$. I denote the predictive mean and covariance as $\boldsymbol{\mu}_\star$ and $\boldsymbol{\Sigma}_\star$, respectively, and they are computed via

$$\boldsymbol{\mu}_\star = \boldsymbol{C}_{N,M}^\mathsf{T}\,\boldsymbol{C}_N^{-1}\,\boldsymbol{y} \tag{1.5}$$

$$\boldsymbol{\Sigma}_\star = \boldsymbol{C}_M - \boldsymbol{C}_{N,M}^\mathsf{T}\,\boldsymbol{C}_N^{-1}\,\boldsymbol{C}_{N,M} \tag{1.6}$$

where $\boldsymbol{C}_M$ is calculated for the prediction locations as in Equation 1.1. For one-dimensional inputs, it is frequently useful to view the marginal conditional distributions across an entire region via a "sausage-link plot" — Figure 1.2 provides an example.

1.3 The Logistic Gaussian Process

Having established that the Gaussian process is a useful nonparametric prior on functions, I now examine how a GP could be used to build a distribution on probability density functions. Such a construction would result in a nonparametric Bayesian kernel density model.

If $\mathcal{V}$ is a bounded subset of $\mathcal{X}$, and $g(\boldsymbol{x})$ is a function drawn from a Gaussian process, it is possible to arrive at a probability density function $f(\boldsymbol{x})$ with support $\mathcal{V}$ by transforming $g(\boldsymbol{x})$:

$$f(\boldsymbol{x}) = \frac{1}{\mathcal{Z}[\boldsymbol{g}]} \exp\{g(\boldsymbol{x})\} \tag{1.7}$$

$$\mathcal{Z}[\boldsymbol{g}] = \int_{\mathcal{V}} \mathrm{d}\boldsymbol{x}' \ \exp\{g(\boldsymbol{x}')\}. \tag{1.8}$$

This transformation is called the *logistic Gaussian process* (LGP). It was introduced by Leonard (1978) and further developed by Lenk (1988, 1991) and by Thorburn (1986). If the mean function and covariance kernel of the Gaussian process are continuous and bounded, then the integral in Equation 1.8 is well-defined (Lenk, 1988; Tokdar, 2006). Under these constraints, the distribution on $f(\boldsymbol{x})$ that the GP implies meets the requirements for a prior on probability density functions: the exponential function ensures that it is everywhere nonnegative and the normalisation constant $\mathcal{Z}[\boldsymbol{g}]$ ensures that it integrates to one. Tokdar and Ghosh (2007) showed that this prior is consistent.

Unfortunately, the logistic Gaussian process is not tractable for inference. First, $g(\boldsymbol{x})$ is an infinite object which cannot be represented with a finite amount of memory. Second, the normalisation constant $\mathcal{Z}[\boldsymbol{g}]$ cannot be evaluated, as it requires integrating over a random infinite-dimensional function. To illustrate the difficulty, I will write $\boldsymbol{g}$ to denote the function $g(\boldsymbol{x})$ as an object on which it is possible to perform inference, ignoring for now the fact that it is infinite. Given N data $\{\boldsymbol{x}_n\}_{n=1}^N$, Bayes' theorem says

$$p(\boldsymbol{g} \mid \{\boldsymbol{x}_n\}_{n=1}^N) = \frac{\mathcal{GP}(\boldsymbol{g}) \ \mathcal{Z}[\boldsymbol{g}]^{-N} \ \exp\left\{\sum_{n=1}^N g(\boldsymbol{x}_n)\right\}}{\int \mathrm{d}\boldsymbol{g}' \ \mathcal{GP}(\boldsymbol{g}') \ \mathcal{Z}[\boldsymbol{g}']^{-N} \ \exp\left\{\sum_{n=1}^N g'(\boldsymbol{x}_n)\right\}}, \tag{1.9}$$

where I am using $\mathcal{GP}(\boldsymbol{g})$ to indicate a Gaussian process prior on $\boldsymbol{g}$. It is common for complex probabilistic models to have the intractable integral that appears in the denominator of Equation 1.9. This quantity is the marginal likelihood, and it does not depend on $\boldsymbol{g}$. Bayesians have developed many ways to deal with posterior computation when the marginal likelihood is unknown. Markov chain Monte Carlo (MCMC), for example, typically only requires that the posterior distribution on parameters be known to within a constant.

The difficulty with the logistic Gaussian process is that Equation 1.9 is a *doubly-*

intractable posterior distribution (Murray et al., 2006). Not only is there an intractable quantity in the denominator, but even the likelihood function

$$p(\{\boldsymbol{x}_n\}_{n=1}^N \,|\, \boldsymbol{g}) = \mathcal{Z}[\boldsymbol{g}]^{-N} \, \exp\left\{\sum_{n=1}^N g(\boldsymbol{x}_n)\right\} \tag{1.10}$$

requires knowledge of $\mathcal{Z}[\boldsymbol{g}]$. As $\mathcal{Z}[\boldsymbol{g}]$ depends on $\boldsymbol{g}$, it cannot be ignored, even when using MCMC. This constant gets its symbol from the German *Zustandssumme* ("sum of states") and in statistical physics this quantity is commonly called the *partition function*. Within machine learning, this quantity arises frequently in the distributions defining undirected graphical models, such as Boltzmann machines (Ackley et al., 1985; Smolensky, 1986). In these finite models, $\mathcal{Z}[\boldsymbol{g}]$ involves a sum over a very large, but finite, number of states. In the logistic Gaussian process, however, calculating $\mathcal{Z}[\boldsymbol{g}]$ requires an integral over an uncountably-infinite number of states: the values of $g(\boldsymbol{x})$ for all $\boldsymbol{x}$ in $\mathcal{V}$.

To make the logistic Gaussian process practical for inference, two finite-dimensional surrogate models have been proposed. The literature does not provide examples of these approximations being used to model data with more than two dimensions. Lenk (1991, 2003) approximates the Gaussian process with a finite number of basis functions, chosen via a truncated Karhunen–Loève expansion. As realisations of $g(\boldsymbol{x})$ are now finite-dimensional, it is possible to determine the normalisation constant $\mathcal{Z}[\boldsymbol{g}]$. As noted by Lenk (2003), however, the required number of basis functions grows exponentially with dimension. It is unclear in this approximation how to choose multi-dimensional basis functions that allow the logistic Gaussian process to be both expressive and efficient. Tokdar (2007) proposes approximating the Gaussian process with a finite-dimensional proxy distribution over a fixed grid of "knots." When it is necessary to evaluate the function at a new location, the value is imputed from the mean function of the Gaussian process, conditioned on the knots, as in Equation 1.5. Tokdar (2007) suggests calculating the normalisation constant $\mathcal{Z}[\boldsymbol{g}]$ numerically from the knots using, for example, the trapezoidal rule or Simpson's rule. It is suggested that reversible-jump Monte Carlo be used to determine the number of knots in higher dimensions, but it is unclear how the normalisation constant is to be estimated when the knots are irregularly spaced. Additionally, when implementing this approximation for the logistic Gaussian process, I have found there to be pathological interaction between the high-order interpolation done by the GP imputation and low-order numerical estimates of the normalisation constant.

In Chapter 2, I present a model called the *Gaussian process density sampler* (GPDS). Like the logistic Gaussian process, the GPDS is a prior on densities arising from a Gaussian process. Unlike the logistic Gaussian process, however, it is possible to simulate data exactly from a random density drawn from the GPDS prior. With a method available for exact generation of data, it is possible to perform inference via Markov chain Monte

Carlo even though the likelihood is intractable.

1.4 The Log Gaussian Cox Process

The Poisson process is a widely-used model for point data in temporal and spatial settings. It forms the foundation for several other types of point processes with more sophisticated properties (e.g. Stoyan and Stoyan (1994) or Møller and Waagepetersen (2004)). The inhomogeneous variant of the Poisson process allows the rate of arrivals to vary in time (or space), but typically we do not have a preconceived idea of the appropriate functional form for this variation. In this setting, it is often desirable to use another stochastic process to describe nonparametrically the variation in the Poisson intensity function. This construction is called a *doubly-stochastic* Poisson process, or a Cox process (Cox, 1955), and it has been applied in a variety of settings, such as neuroscience (Brown, 2005; Cunningham et al., 2008b), astronomy (Gregory and Loredo, 1992), and forestry (Heikkinen and Arjas, 1999).

One variant of the Cox process is the Gaussian Cox process, where the intensity function is a transformation of a random realisation from a Gaussian process (GP). As in the logistic Gaussian process of Section 1.3, this is a particularly convenient way to nonparametrically specify beliefs about the intensity function via a kernel. We are able to select a Gaussian process covariance function based on our prior conceptions of the intensity function, without having to choose a particular parameterisation or a finite set of basis functions.

The most straightforward way to build a Poisson process from a Gaussian process is to use an exponentiated draw from the GP as the intensity function. That is, if the intensity function is $\lambda(\boldsymbol{x})$ on the domain $\mathcal{X}$, then a random GP realisation is the log of $\lambda(\boldsymbol{x})$, i.e. $g(\boldsymbol{x}) = \ln \lambda(\boldsymbol{x})$. This construction is called the *log Gaussian Cox process* (LGCP) and it has been studied by Rathbun and Cressie (1994) and by Møller et al. (1998).

Unfortunately, likelihood-based inference in the LGCP is intractable, for very similar reasons to those that render the logistic Gaussian process intractable. Not only is there again an infinite-dimensional function $g(\boldsymbol{x})$ that must be represented in finite memory, but there is also an intractable likelihood. If the data are a sequence of K events $\{\boldsymbol{x}_k\}_{k=1}^K$ on a finite region $\mathcal{V}$ in $\mathcal{X}$, then the Poisson process likelihood of $\boldsymbol{g}$ is

$$p(\{\boldsymbol{x}_k\}_{k=1}^K \,|\, \boldsymbol{g}) = \exp\left\{ -\left(\int_{\mathcal{V}} \mathrm{d}\boldsymbol{x} \;\; \exp\{g(\boldsymbol{x})\} \right) + \sum_{k=1}^{K} g(\boldsymbol{x}_k) \right\}. \tag{1.11}$$

As with $\mathcal{Z}[\boldsymbol{g}]$ in Equation 1.8, the integral in Equation 1.11 involves an infinite-dimensional random function, and is unknowable for nontrivial Gaussian process priors on $g(\boldsymbol{x})$. When incorporated into Bayes' Theorem, this likelihood results in

another example of a doubly-intractable posterior distribution.

To address this intractability, Møller et al. (1998) propose discretising the region $\mathcal{V}$. In this finite-dimensional approximation, each cell in the grid is considered to have a constant rate. The logs of these constant rates are then tied together via the Gaussian process prior. Relative to the knot-based approximation of the logistic Gaussian process, this approximation seems well-behaved in practice. Nevertheless, it requires the practitioner to make an *a priori* choice of bin size.

Chapter 3 presents an alternative model to the log Gaussian Cox process, called the *sigmoidal Gaussian Cox process* (SGCP). This model transforms draws from a Gaussian process into random Poisson process intensity functions, but has the additional property that it is fully generative. That is, it is possible to generate exact Poisson data from a random intensity function drawn from the prior. As in the Gaussian process density sampler, this enables tractable fully-nonparametric Bayesian inference via Markov chain Monte Carlo.

1.5 Other Density-Related Gaussian Process Models

The logistic Gaussian process and the log Gaussian Cox process are not the only models to incorporate Gaussian processes into Bayesian density models. Here I briefly review several related models.

1.5.1 The Csató Density Model

Exponentiating the draw from a Gaussian process as in Equation 1.7 is only one way to arrive at a nonnegative random function. Csató (2002) proposes the transformation

$$f(\boldsymbol{x}) = \frac{g(\boldsymbol{x})^2}{\int_{\mathcal{V}} \mathrm{d}\boldsymbol{x}' \; g(\boldsymbol{x}')^2} \tag{1.12}$$

as a way to construct a probability density $f(\boldsymbol{x})$ from a random function $g(\boldsymbol{x})$. Posterior inference with this model is doubly-intractable, as in the logistic Gaussian process, but it also has the problem of having many possible posterior modes. These modes arise because various sign-changes of the latent function $g(\boldsymbol{x})$ can result in identical $f(\boldsymbol{x})$. While the LGP has redundancy — adding a constant function to $g(\boldsymbol{x})$ results in the same $f(\boldsymbol{x})$ — it does not have this sign pathology.

1.5.2 The Gaussian Process Latent Variable Model

The *Gaussian process latent variable model* (GPLVM) of Lawrence (2005) is another way to use a Gaussian process to model the distribution of complex data. In the GPLVM,

the Gaussian process prior is not placed on the probability density function itself, but the GP is used to provide a nonparametric distribution over maps from a low-dimensional latent space into a high-dimensional observed space. The objective of the GPLVM is to find low-dimensional explanations of high-dimensional data, and it can be viewed as a kernelised dual of probabilistic principal components analysis (Tipping and Bishop, 1999), or as a nonparametric generalisation of the density network (MacKay, 1995). If the observations are in a D-dimensional space, then the N observed data $\boldsymbol{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_N]$ can be modeled by a linear transformation of N data $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N]$ in a latent D'-dimensional space, plus zero-mean independent Gaussian noise with variance σ^2:

$$p(\boldsymbol{Y} \mid \boldsymbol{A}, \boldsymbol{X}) = \prod_{n=1}^{N} \mathcal{N}(\boldsymbol{y}_n \mid \boldsymbol{A}\boldsymbol{x}_n, \sigma^2 \boldsymbol{I}_D) \tag{1.13}$$

where $\boldsymbol{A}$ is a $D \times D'$ transformation matrix. A zero-mean spherical Gaussian prior is now placed on the components of $\boldsymbol{A}$:

$$p(\boldsymbol{A}) = \prod_{d=1}^{D'} \mathcal{N}(\boldsymbol{a}_d \mid \boldsymbol{0}_D, \boldsymbol{I}_D) \tag{1.14}$$

and $\boldsymbol{A}$ is marginalised out, giving

$$p(\boldsymbol{Y} \mid \boldsymbol{X}) = \prod_{d=1}^{D} \mathcal{N}(\boldsymbol{y}^{(d)} \mid \boldsymbol{0}_N, \boldsymbol{X}\boldsymbol{X}^\mathsf{T} + \sigma^2 \boldsymbol{I}_N), \tag{1.15}$$

where $\boldsymbol{y}^{(d)}$ is the length-N vector of the observed values in the dth dimension. Lawrence (2005) makes the observation that when $\boldsymbol{A}$ is marginalised out, Equation 1.15 gives the same probability distribution as would D independent Gaussian processes with linear kernels and Gaussian observation noise σ^2. This linear kernel can be replaced with an infinite-dimensional one, such as discussed in Section 1.2.1. This is equivalent to applying the "kernel trick" (e.g. Vapnik (1995)) to the inner product matrix $\boldsymbol{X}\boldsymbol{X}^\mathsf{T}$. The GPLVM then optimises the values in the latent space to find appropriate explanations for the observed data.

The GPLVM has a straightforward interpretation and has been successfully applied to several difficult real-world problems, but it is not tackling the problems addressed by this thesis. It does not place a distribution on the latent variables and it optimises their values. We cannot easily find the value of the density for some new observed datum.

1.5.3 The Cunningham Poisson Model

Cunningham et al. (2008a) propose a Cox process model for temporal data that is similar to the log Gaussian Cox process, but rather than transforming the random func-

tions into intensity functions, they use them directly, i.e. $\lambda(\boldsymbol{x}) = g(\boldsymbol{x})$. This approach is used to avoid the nonlinear effects of the exponential transform, but the result is an inconsistent model that assigns prior mass to negative intensities. The main focus of Cunningham et al. (2008a) is on finding an efficient computational scheme for maximum *a posteriori* (MAP) inference that is constrained to prevent the aforementioned negative intensities. Their approach is limited to the time domain, as they use a renewal-process formalism. They use a discretisation for their finite-dimensional approximation, in a fashion similar to Møller et al. (1998).

1.6 Overview

This rest of this thesis constructs new nonparametric probability density and point process models based on Gaussian processes and explains how inference can be performed in these models by extending recent developments in Markov chain Monte Carlo methods. In Chapter 2, I present a method for constructing priors on probability density functions that incorporates a Gaussian process. In Chapter 3, I develop a similar prior for the intensity function of a Poisson process. Chapter 4 and Chapter 5 review two recently-developed Markov chain Monte Carlo methods and apply them to the new priors on probability densities and point process intensity functions. In Chapter 6, I apply these models and inference algorithms to a few example problems. Chapter 7 presents some extensions to the models. In Chapter 8, I discuss the task of semi-supervised learning and show how the density model of Chapter 2 can be expanded to solve this problem in a nonparametric Bayesian manner. Chapter 9 reviews the thesis and discusses some potential future directions of research.

Several papers have been or will be published whose content overlaps significantly with the work presented in this thesis. Adams et al. (2009a) presents the model in Chapter 2 and some of the material in Chapter 5 and Chapter 6. Adams et al. (2009b) (and also Adams et al. (2008)) overlaps with material in Chapter 2, Chapter 4, Chapter 5, Chapter 6, and Chapter 7. Material from Chapter 3, Chapter 5 and Chapter 6 appear in Adams et al. (2009c). Adams and Ghahramani (2009) presents the model in Chapter 8.

Chapter 2

The Gaussian Process Density Sampler

This chapter constructs the *Gaussian process density sampler* (GPDS), which is a distribution on probability density functions, based on a Gaussian process. I discuss various properties of this prior and develop a procedure for generating data that have been drawn from a single density, where the density was drawn from this prior.

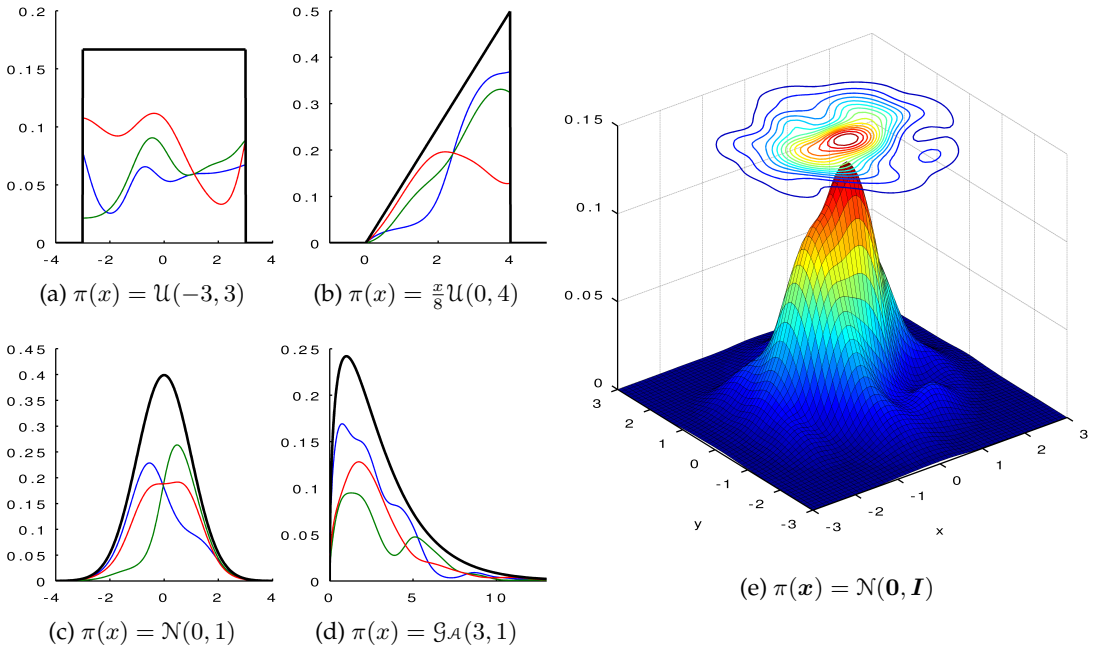
2.1 The Prior on Probability Density Functions

Recall from Section 1.2 that I am treating a Gaussian process as a prior on functions that have $\mathcal{X}$ as their domain and the real line as their range. I now address the topic of defining a prior on probability density functions (PDFs) on $\mathcal{X}$. As in Section 1.2 I continue to use $\mathbb{R}^D$ as the example for $\mathcal{X}$. Probability density functions have two requirements: they must be everywhere nonnegative and they must integrate to one. Sample realisations from a Gaussian process do not meet these constraints *a priori*, so the realisations must be transformed into the space of PDFs. If $g(\boldsymbol{x})$ is a sample function from a Gaussian process, I transform it into a PDF $f(\boldsymbol{x})$ via

$$f(\boldsymbol{x}) = \frac{1}{\mathcal{Z}_\pi[\boldsymbol{g}]}\, \Phi(g(\boldsymbol{x}))\, \pi(\boldsymbol{x}) \tag{2.1}$$

where $\pi(\boldsymbol{x})$ is an arbitrary *base density* on $\mathcal{X}$. The function $\Phi(\cdot) : \mathbb{R} \to (0, 1)$ is a positive function with upper bound 1. I use the bold notation $\boldsymbol{g}$ to refer to the function $g(\boldsymbol{x})$ compactly as a vector of (infinite) length on which it is possible to perform inference. The normalisation constant $\mathcal{Z}_\pi[\boldsymbol{g}]$ is a functional of $g(\boldsymbol{x})$:

$$\mathcal{Z}_\pi[\boldsymbol{g}] = \int_{\mathcal{X}} \mathrm{d}\boldsymbol{x}'\; \Phi(g(\boldsymbol{x}'))\, \pi(\boldsymbol{x}'). \tag{2.2}$$



(a) $\pi(x) = \mathcal{U}(-3, 3)$ (b) $\pi(x) = \frac{x}{8}\mathcal{U}(0, 4)$

(c) $\pi(x) = \mathcal{N}(0, 1)$ (d) $\pi(x) = \mathcal{G}_{\mathcal{A}}(3, 1)$

(e) $\pi(\boldsymbol{x}) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$

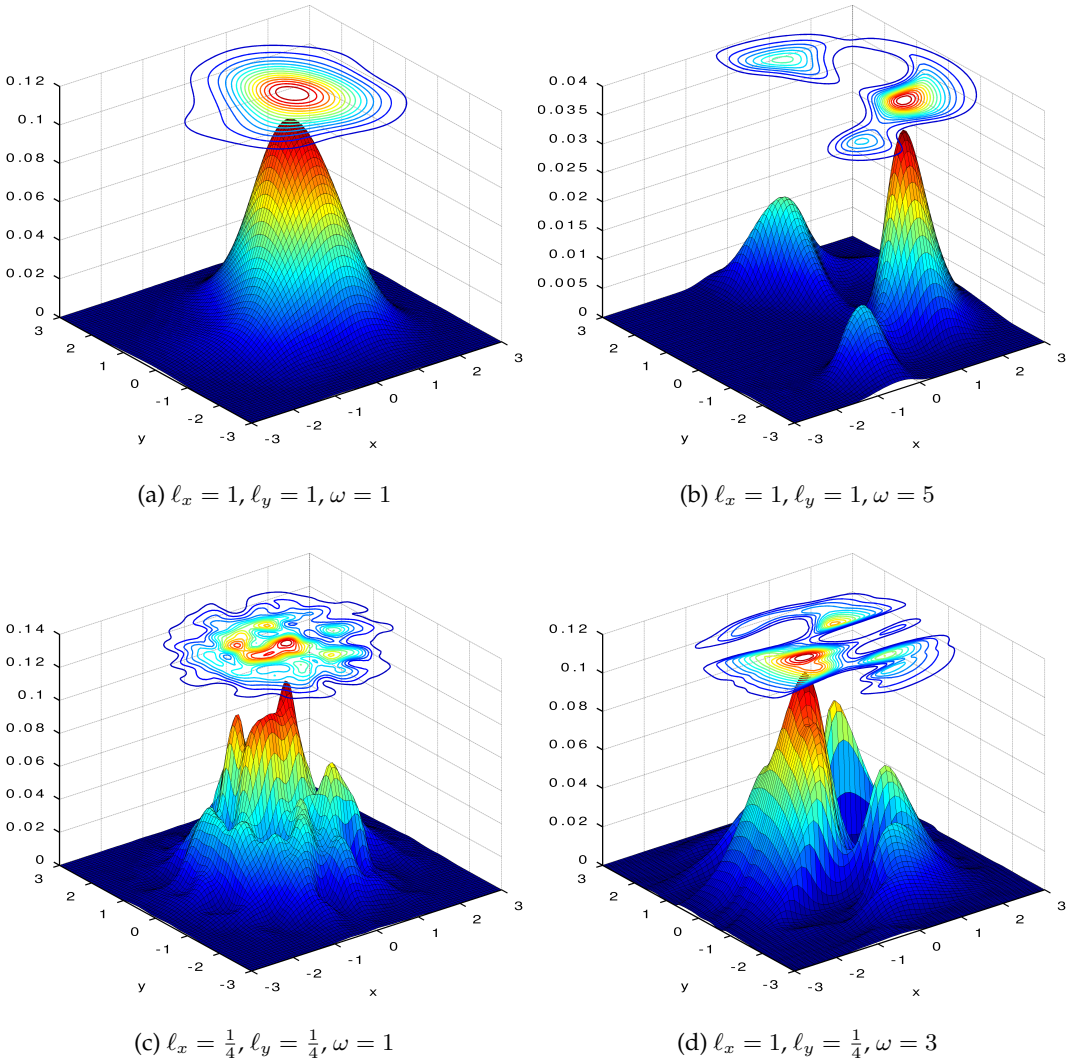
Figure 2.1: Unnormalised typical samples from the GPDS prior, i.e. samples from a Gaussian process transformed by Equation 2.1. In each case, the Gaussian process uses a squared-exponential covariance function with $\ell = 1$ and $\omega = 1$. (a)–(d) are unidimensional, using four different base densities $\pi(\boldsymbol{x})$. Random unnormalised densities $\Phi(g(\boldsymbol{x}))\,\pi(\boldsymbol{x})$ are shown in colours, with with $\pi(\boldsymbol{x})$ shown via a thicker black line. (a) The base density is uniform on $(-3, 3)$. (b) The base density is a "wedge" from 0 to 4. (c) The base density is a standard normal distribution. (d) The base density is a gamma distribution with shape $\alpha = 3$ and inverse scale $\beta = 1$. (e) This is a two-dimensional sample density where $\pi(\boldsymbol{x})$ is the spherical Gaussian with zero mean and unit variance. The length scales in both dimensions are $\frac{1}{2}$.

I include a subscript π to indicate implicit dependence on the density $\pi(\boldsymbol{x})$. As $\pi(\boldsymbol{x})$ is a normalised probability density function, the constant $\mathcal{Z}_\pi[\boldsymbol{g}]$ is restricted to the interval $(0, 1)$ for any $g(\boldsymbol{x})$. Through the map defined by Equation 2.1, a Gaussian process prior becomes a prior distribution over normalised probability density functions on $\mathcal{X}$. Figure 2.1 shows several samples from this prior.

2.1.1 The Response Function $\Phi(\cdot)$

The function $\Phi(\cdot)$ may be interpreted as a response function (Rasmussen and Williams, 2006), or as a warping as in Snelson et al. (2004). Although $\Phi(\cdot)$ is only required to be positive and bounded, it is convenient for inference if it is a bijective map between $\mathbb{R}$ and $(0, 1)$. If $\Phi(\cdot)$ is bijective then each realisation $g(\boldsymbol{x})$ from the Gaussian process corresponds to a unique function that maps $\mathcal{X}$ to $(0, 1)$. Sigmoids, such as the cumulative normal distribution function and the logistic function, are bijective functions with this domain and range. In this thesis, I take $\Phi(\cdot)$ to be the logistic function

$$\Phi(z) = \frac{1}{1 + e^{-z}} \tag{2.3}$$



(a) $\ell_x = 1, \ell_y = 1, \omega = 1$

(b) $\ell_x = 1, \ell_y = 1, \omega = 5$

(c) $\ell_x = \frac{1}{4}, \ell_y = \frac{1}{4}, \omega = 1$

(d) $\ell_x = 1, \ell_y = \frac{1}{4}, \omega = 3$

Figure 2.2: Unnormalised samples from the GPDS prior for four different settings of the squared exponential covariance function of Equation 1.2. In each case the base density is a spherical Gaussian with zero mean and unit variance.

as it is inexpensive to calculate.

As a side note, if the Gaussian process has a mean of zero and unit variance everywhere, i.e. $C(\boldsymbol{x}, \boldsymbol{x}) = 1, \forall \boldsymbol{x} \in \mathcal{X}$, and $\Phi(\cdot)$ is the standard normal cumulative distribution function

$$\Phi_{\mathsf{norm-cdf}}(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z'} \mathrm{d}z' \, e^{-\frac{1}{2}(z')^2} = \frac{1}{2}\left(1 + \mathrm{erf}\left\{\frac{z}{\sqrt{2}}\right\}\right), \tag{2.4}$$

where $\mathrm{erf}(\cdot)$ is the error function (Abramowitz and Stegun, 1964), then the realisations $\Phi(g(\boldsymbol{x}))$ are from an infinite generalisation of a Gaussian copula (Nelsen, 2007). The marginals of this "Gaussian copula process" are uniform on $(0, 1)$.

2.1.2 The Base Density $\pi(x)$

I use the term *base density* for $\pi(\boldsymbol{x})$ in analogy to the Dirichlet process base measure (Ferguson, 1973). The inclusion of the base density $\pi(\boldsymbol{x})$ makes the model semiparametric by some definitions (Tokdar and Ghosh, 2007). As pointed out by Müller et al. (2004), "nonparametric" Bayesian models would be better described as "massively parametric." Bayesian semiparametric models are also massively parametric, and so I view nonparametric versus semiparametric, in the Bayesian context, to be a false dichotomy.

I introduced hyperparameters $\boldsymbol{\theta}$ for the Gaussian process covariance function in Section 1.2, and here I introduce additional hyperparameters $\boldsymbol{\psi}_\pi$ that control the base density.

2.1.3 The Effect of Gaussian Process Hyperparameters

As in Gaussian process regression, the choice of covariance function determines the properties of typical sample probability density functions drawn from the Gaussian process density sampler prior. For a given covariance function, the hyperparameters provide additional knobs to turn. In this case, I am assuming the squared-exponential covariance function of Equation 1.2 and the hyperparameters available are the length scales $\boldsymbol{\ell} = \{\ell_d\}_{d=1}^D$ and the amplitude ω. The length scales specify how close in value the probabilities of two points in the data space should be, relative to how far apart the points are. The $\boldsymbol{\ell}$ determines to what extent "similar data should have similar probabilities," where similarity is determined by an appropriately-defined distance metric on $\mathcal{X}$. The amplitude parameter determines how much variation there is between the areas that have the largest densities versus those that have the smallest densities. Large values of ω cause typical samples of $g(\boldsymbol{x})$ to be more likely to saturate the sigmoid $\Phi(\cdot)$, leading to cliffs in the density and plateaus where the shape of the density is very close to zero or very close to $\pi(\boldsymbol{x})$. Figure 2.2 shows the effects of varying these hyperparameters in two dimensions.

2.2 Generating Data from the Prior

The reason for introducing the GPDS construction of Section 2.1 rather than using the logistic Gaussian process discussed in Section 1.3 is that the transformation in Equation 2.1 allows the simulation of samples on the data space $\mathcal{X}$, drawn from a random density $f(\boldsymbol{x})$ that has been drawn from from the prior. It is possible, in effect, to draw a random density $f(\boldsymbol{x})$ from the prior and then draw exact data on $\mathcal{X}$ that are i.i.d. from $f(\boldsymbol{x})$. Of course, $f(\boldsymbol{x})$ is unknown, but the data are still exact. I use a variant of rejection sampling to draw these data.

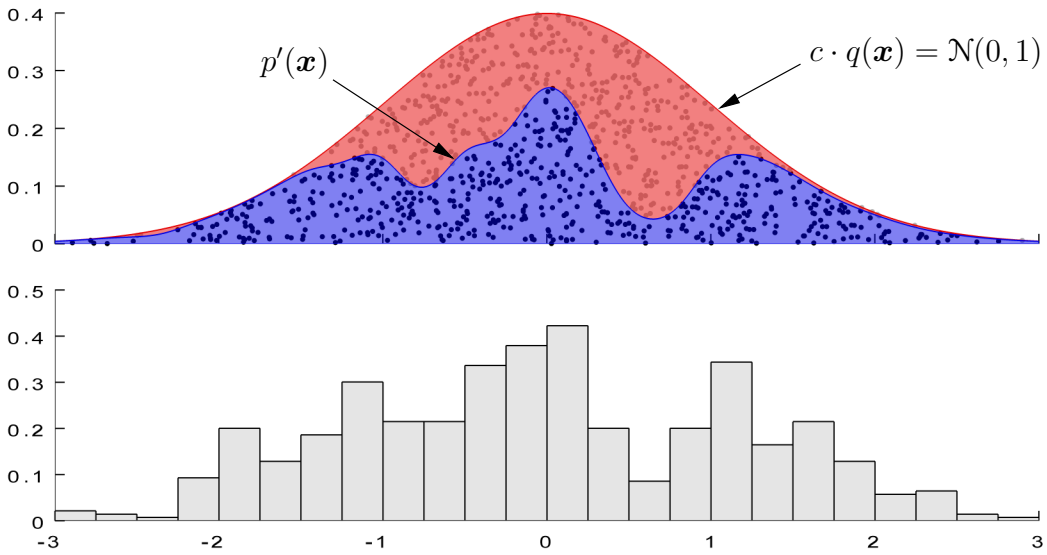


Figure 2.3: Schematic representation of rejection sampling. The red area is the dominating density $c \cdot q(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x} \,|\, 0, 1)$, a standard Gaussian distribution with zero mean and unit variance. In this example, $c = 1$. The blue area shows a more complicated unnormalised density $p'(\boldsymbol{x})$, from which samples are required. Both densities are truncated to the interval $[-3, 3]$. There are 1000 points uniformly distributed in the combined red and blue areas. Only points in the blue area are kept, and the histogram in the lower plot shows the resulting empirical density.

2.2.1 Rejection Sampling

Rejection sampling (e.g. Devroye (1986, Chapter 2), Robert and Casella (2004, Chapter 2), or MacKay (2003, Chapter 29)) is an algorithm for generating exact samples from a probability density function. By "exact" I mean that the samples are not biased, for example, by the starting state of a finite Markov chain. Here, I will denote the density from which I intend to draw samples as $p(\boldsymbol{x})$, and the known function proportional to $p(\boldsymbol{x})$ as $p'(\boldsymbol{x})$. Rejection sampling is applicable in situations where $p(\boldsymbol{x})$ is known within a constant factor but is too complicated for direct sampling. One chooses a *proposal density* $q(\boldsymbol{x})$ from which it is possible to generate exact samples, and a constant c such that the function $c \cdot q(\boldsymbol{x})$ provides an upper bound for $p'(\boldsymbol{x})$, i.e. $\forall \boldsymbol{x} \in \mathcal{X}, c \cdot q(\boldsymbol{x}) \geq p'(\boldsymbol{x})$.

To perform rejection sampling, in the rth iteration generate a random variate $\tilde{\boldsymbol{x}}_r$ from the proposal density $q(\boldsymbol{x})$. A random variate u_r is then drawn uniformly on the interval between zero and the upper-bounding function at $\tilde{\boldsymbol{x}}_r$, that is $u_r \sim \mathcal{U}(0, c \cdot q(\tilde{\boldsymbol{x}}_r))$. The proposal $\tilde{\boldsymbol{x}}_r$ is accepted if u_r is less than $p'(\tilde{\boldsymbol{x}}_r)$, and otherwise rejected. This idea is shown graphically in Figure 2.3, highlighting why this algorithm works: if the pairs $(\tilde{\boldsymbol{x}}_r, u_r)$ are drawn uniformly beneath $p'(\boldsymbol{x})$, then the marginal distribution of $\tilde{\boldsymbol{x}}_r$ is $p(\boldsymbol{x})$ (Robert and Casella, 2004). To generate these uniform pairs $(\tilde{\boldsymbol{x}}_r, u_r)$, generate uniformly from a larger envelope provided by $c \cdot q(\boldsymbol{x})$ and discard those that fall above $p'(\boldsymbol{x})$. Algorithm 2.1 shows the procedure in pseudocode.

Algorithm 2.1 Generate N exact samples from $p(\boldsymbol{x})$ via rejection sampling.

Inputs:
- Number of samples to draw N
- Unnormalised density $p'(\boldsymbol{x}) \propto p(\boldsymbol{x})$
- Proposal density $q(\boldsymbol{x})$
- Constant c such that $c \cdot q(\boldsymbol{x}) \geq p'(\boldsymbol{x}), \forall \boldsymbol{x}$

Outputs:
- N samples from $p(\boldsymbol{x})$, $\mathcal{D} = \{\tilde{\boldsymbol{x}}_n\}_{n=1}^N$

$r \leftarrow 0$ ▷ Count the number of proposals made.
repeat
  $\tilde{\boldsymbol{x}}_r \sim q(\boldsymbol{x})$ ▷ Draw a proposal.
  $u_r \sim \mathcal{U}(0, c \cdot q(\tilde{\boldsymbol{x}}_r))$ ▷ Draw uniformly beneath the proposal distribution.
  if $u_r < p'(\tilde{\boldsymbol{x}}_r)$ **then** ▷ Acceptance rule.
    $\mathcal{D} \leftarrow \mathcal{D} \cup \tilde{\boldsymbol{x}}_r$ ▷ Store the proposal, discard u_r.
  end if
  $r \leftarrow r + 1$
until $|\mathcal{D}| = N$ ▷ Loop until N samples are accepted.
return $\mathcal{D}$

Rejection sampling for probability densities is generally credited to von Neumann (1951), although the concept of the reject/accept procedure for calculation by Monte Carlo simulation is at least as old as 1733 when Georges-Louis Leclerc, Comte de Buffon, proposed repeated drops of a needle as a means of estimating the constant π. This was published years later in Leclerc de Buffon (1777), and has come to be known as "Buffon's Needle" (see Holgate (1981) for further discussion).

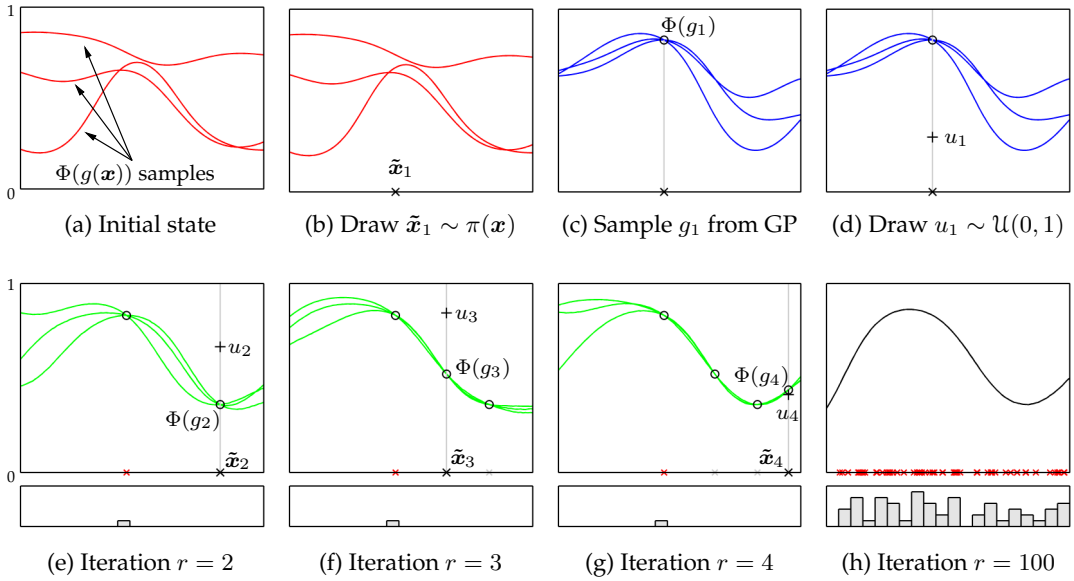
As discussed in MacKay (2003, Chapter 29), rejection sampling tends to become exponentially less efficient with increasing dimensionality. The function $c \cdot q(\boldsymbol{x})$ generally becomes a very loose bound on $p'(\boldsymbol{x})$ as the dimensionality increases, and so most proposals are rejected.

2.2.2 Sampling from the GPDS

Rejection sampling can be used to simulate a set of data from a common density drawn from the the prior described in Section 2.1. The proposal density in this case is $\pi(\boldsymbol{x})$ with $c = 1$. As the function $\Phi(g(\boldsymbol{x}))$ is less than one for all $g(\boldsymbol{x})$, $\pi(\boldsymbol{x})$ is guaranteed to be an upper bound for $\Phi(g(\boldsymbol{x}))\,\pi(\boldsymbol{x})$ regardless of the realisation from the Gaussian process. I assume that it is possible to draw samples directly from $\pi(\boldsymbol{x})$.

If $g(\boldsymbol{x})$ were known, it would be possible to generate samples from the associated density $f(\boldsymbol{x}) \propto \Phi(g(\boldsymbol{x}))\,\pi(\boldsymbol{x})$ as in Section 2.2.1. However, in the GPDS, $g(\boldsymbol{x})$ is not known: it is a random function drawn from a Gaussian process prior. Nevertheless, rejection sampling can be used by "discovering" $g(\boldsymbol{x})$ during the procedure, sampling the function from the GP only at the places where it is necessary. As it is required only to know $g(\boldsymbol{x})$ at the $\{\tilde{\boldsymbol{x}}_r\}$ to accept or reject these proposals, the samples are still exact.

It is possible to generate the samples sequentially, as shown graphically in Figure 2.4 and via pseudocode in Algorithm 2.2. In each loop, a proposal is drawn from the base



(a) Initial state (b) Draw $\tilde{\boldsymbol{x}}_1 \sim \pi(\boldsymbol{x})$ (c) Sample g_1 from GP (d) Draw $u_1 \sim \mathcal{U}(0, 1)$

(e) Iteration $r = 2$ (f) Iteration $r = 3$ (g) Iteration $r = 4$ (h) Iteration $r = 100$

Figure 2.4: Several iterations of the GPDS sampling procedure. In these figures, the base density $\pi(\boldsymbol{x})$ is uniform on the interval shown. In Figures (a)–(g), three posterior function samples are shown. Figure (h) only shows one function sample from the GP posterior. Figures (e)–(h) also show histograms of the accepted samples.

density $\pi(\boldsymbol{x})$ and the function $g(\boldsymbol{x})$ is sampled from the Gaussian process at this proposed coordinate, conditioning on all the function values already sampled. I will call these data the *conditioning set* for the function $g(\boldsymbol{x})$, and will denote the conditioning inputs as $\boldsymbol{X}$ and the conditioning function values as $\boldsymbol{G}$. After the function is sampled, a variate is drawn uniformly from $(0, 1)$ and compared to the Φ-squashed function at the proposal location. If the uniform variate falls below $\Phi(g(\boldsymbol{x}))$ then accept the proposal, otherwise reject. The proposals and their function values are added into the conditioning set regardless of whether that proposal was accepted or rejected. The loop repeats until there have been as many acceptances as are required.

2.2.3 Infinite Exchangeability

If $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots$ is a possibly-infinite sequence of data from a random process, the procedure to generate the data is said to be *infinitely exchangeable* if the joint probability of any finite-length subsequence is the same under reordering. In other words, the procedure is exchangeable if $p(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_R) = p(\boldsymbol{x}_{\Pi(1)}, \boldsymbol{x}_{\Pi(2)}, \ldots, \boldsymbol{x}_{\Pi(R)})$ for any R and all permutations Π (Bernardo, 1996).

Exchangeability is a desirable property for Bayesian analysis. By de Finetti's theorem (Hewitt and Savage, 1955), it implies that the data may be considered conditionally independent, given some possibly-infinite set of parameters γ. Moreover, as discussed by Bernardo (1996), the exchangeability of a sequence $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots \boldsymbol{x}_R\}$ implies

Algorithm 2.2 Generate N samples from a density drawn from the GPDS prior.

Inputs:
- Number of samples to draw N
- Gaussian process covariance function $C(\boldsymbol{x}, \boldsymbol{x}' ; \boldsymbol{\theta})$
- Base density $\pi(\boldsymbol{x} \mid \boldsymbol{\psi}_\pi)$

Outputs:
- N samples $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^N$ from a density drawn from the prior.

$\boldsymbol{X} \leftarrow \emptyset$, $\boldsymbol{G} \leftarrow \emptyset$ ▷ Initially the conditioning sets are empty.
$\mathcal{D} \leftarrow \emptyset$ ▷ Initialise the set to be returned.
$r \leftarrow 0$ ▷ Count the number of proposals.
repeat
  $\tilde{\boldsymbol{x}}_r \sim \pi(\boldsymbol{x} \mid \boldsymbol{\psi}_\pi)$ ▷ Draw a proposal.
  $g(\tilde{\boldsymbol{x}}_r) \sim \mathcal{GP}(g \mid \tilde{\boldsymbol{x}}_r, \boldsymbol{X}, \boldsymbol{G}, \boldsymbol{\theta})$ ▷ Sample from the GP at the proposal.
  $u_r \sim \mathcal{U}(0, 1)$ ▷ Draw uniformly on $(0, 1)$.
  if $u_r < \Phi(g(\tilde{\boldsymbol{x}}_r))$ **then** ▷ Acceptance rule.
    $\mathcal{D} \leftarrow \mathcal{D} \cup \tilde{\boldsymbol{x}}_r$ ▷ Store the proposal, discard u_r.
  end if
  $\boldsymbol{X} \leftarrow \boldsymbol{X} \cup \tilde{\boldsymbol{x}}_r$ ▷ Update the conditioning sets, even on rejections.
  $\boldsymbol{G} \leftarrow \boldsymbol{G} \cup g(\tilde{\boldsymbol{x}}_r)$
  $r \leftarrow r + 1$
until $|\mathcal{D}| = N$ ▷ Loop until N samples are accepted.
return $\mathcal{D}$

that there exists a probability distribution $p(\gamma)$ such that

$$p(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_R) = \int_\Gamma \mathrm{d}\gamma \; p(\gamma) \prod_{r=1}^R p(\boldsymbol{x}_r \mid \gamma). \tag{2.5}$$

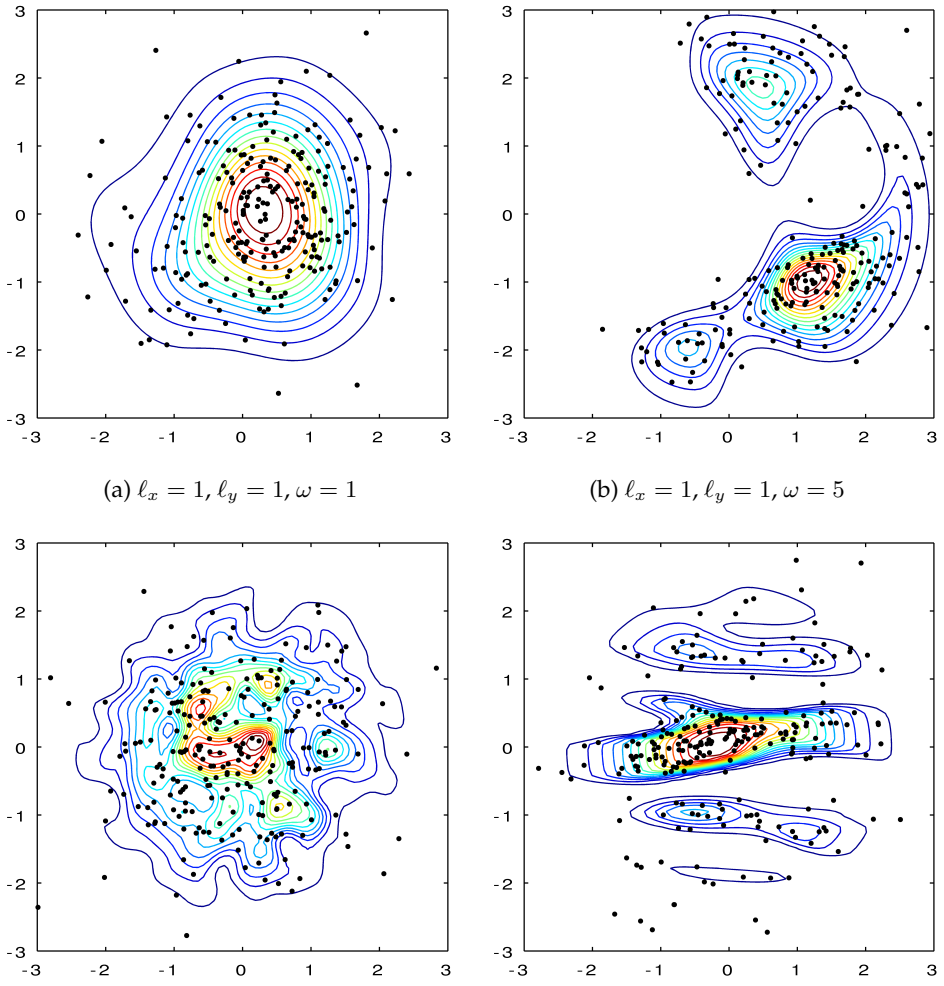
Additionally, exchangeability implies the existence of a predictive distribution on the data space given by

$$p(\boldsymbol{x} \mid \boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_R) = \int_\Gamma \mathrm{d}\gamma \; p(\boldsymbol{x} \mid \gamma) \, p(\gamma \mid \boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_R). \tag{2.6}$$

These properties are critical to the validity of the inference and prediction tasks that I will examine in later chapters.

The data generation procedure of Section 2.2.2 and Algorithm 2.2 is infinitely exchangeable; the joint probability of a set of accepted proposals is the same under reordering. Informally, one can see that the procedure is exchangeable by considering a different way of generating the data. We could draw R independent variates $\{\tilde{\boldsymbol{x}}_r\}_{r=1}^R$ from $\pi(\boldsymbol{x})$. We could then sample the function values $\{g(\tilde{\boldsymbol{x}}_r)\}_{r=1}^R$ from the R-dimensional Gaussian distribution implied by the $\{\tilde{\boldsymbol{x}}_r\}_{r=1}^R$. The multivariate Gaussian is exchangeable in its components. Finally, each accept/reject decision is an independent Bernoulli draw with probability of acceptance $\Phi(g(\tilde{\boldsymbol{x}}_r))$. We could perform each step simultaneously and draw all R at once, without introducing a concept of ordering. This procedure would work for any R.

More formally, I introduce $b_r \in \{0, 1\}$ to indicate whether the rth proposal $\tilde{\boldsymbol{x}}_r$ is accepted (1) or rejected (0). I use the shorthand τ_r to indicate the rth triplet $(\tilde{\boldsymbol{x}}_r, g(\tilde{\boldsymbol{x}}_r), b_r)$. Consider the joint probability of a length-R sequence of triplets from the generative



(a) $\ell_x = 1, \ell_y = 1, \omega = 1$ (b) $\ell_x = 1, \ell_y = 1, \omega = 5$

(c) $\ell_x = \frac{1}{4}, \ell_y = \frac{1}{4}, \omega = 1$ (d) $\ell_x = 1, \ell_y = \frac{1}{4}, \omega = 3$

Figure 2.5: Using the same densities drawn in Figure 2.2, 250 draws on data space using the algorithm of Section 2.2.2.

algorithm:

$$p(\tau_1, \ldots, \tau_r, \tau_{r+1}, \ldots, \tau_R) = \\ p(\tau_1) \cdots p(\tau_r \,|\, \{\tau_{r'}\}_{r'=1}^{r-1})\; p(\tau_{r+1} \,|\, \{\tau_{r'}\}_{r'=1}^{r}) \cdots p(\tau_R \,|\, \{\tau_{r'}\}_{r'=1}^{R-1}). \quad (2.7)$$

To prove exchangeability, it suffices to show that τ_r and τ_{r+1} can be exchanged for any $r < R$ without changing the probability, as it is possible to arrive at any permutation of $\{1, 2, \ldots, R\}$ by adjacent pairwise swaps (Cormen et al., 2001). By causality, the joint probability of exchanging τ_r and τ_{r+1} does not affect the joint probability $p(\tau_1, \ldots, \tau_{r-1})$, so it is only necessary to show that

$$p(\tau_r, \tau_{r+1}, \ldots, \tau_R \,|\, \{\tau_{r'}\}_{r'=1}^{r-1}) = p(\tau_{r+1}, \tau_r, \ldots, \tau_R \,|\, \{\tau_{r'}\}_{r'=1}^{r-1}). \quad (2.8)$$

I write the joint for the left side of Equation 2.8:

$$p(\tau_r, \tau_{r+1}, \ldots, \tau_R \,|\, \{\tau_{r'}\}_{r'=1}^{r-1}) = \prod_{r'=r}^{R} \pi(\tilde{\boldsymbol{x}}_{r'})\ \mathcal{GP}(g(\tilde{\boldsymbol{x}}_{r'}) \,|\, \{\tilde{\boldsymbol{x}}_{r''}, g(\tilde{\boldsymbol{x}}_{r''})\}_{r''=1}^{r'-1}, \tilde{\boldsymbol{x}}_{r'})\ \Phi(g(\tilde{\boldsymbol{x}}_{r'}))^{b_{r'}}\ [1 - \Phi(g(\tilde{\boldsymbol{x}}_{r'}))]^{1-b_{r'}}.$$

The $\pi(\tilde{\boldsymbol{x}}_{r'})$, $\Phi(g(\tilde{\boldsymbol{x}}_{r'}))^{b_{r'}}$ and $[1 - \Phi(g(\tilde{\boldsymbol{x}}_{r'}))]^{1-b_{r'}}$ terms inside the product are not conditioned on any previous parts of the sequence, so their probability does not change under reordering. After removing these invariant factors Equation 2.8 has only Gaussian process terms. I combine these to form the joint distribution

$$\mathcal{GP}(g(\tilde{\boldsymbol{x}}_r), g(\tilde{\boldsymbol{x}}_{r+1}) \,|\, \{\tilde{\boldsymbol{x}}_{r'}, g(\tilde{\boldsymbol{x}}_{r'})\}_{r'=1}^{r-1}, \tilde{\boldsymbol{x}}_r, \tilde{\boldsymbol{x}}_{r+1}) \tag{2.9}$$

in which $g(\tilde{\boldsymbol{x}}_r)$ and $g(\tilde{\boldsymbol{x}}_{r+1})$ can be exchanged as long as $\tilde{\boldsymbol{x}}_r$ and $\tilde{\boldsymbol{x}}_{r+1}$ are also exchanged.

2.3 Summary

This chapter introduced a prior on probability density functions that is based on transformation of a Gaussian process. This prior is unique in that one can draw data from a single density drawn from the prior. I call the prior and the associated data generation procedure the Gaussian process density sampler.

Later chapters will demonstrate two ways to perform inference using this prior. The ability to generate exact data will be crucial to the inference method of Chapter 4. In Chapter 5, I will actually model the GPDS rejection sampler itself to perform inference.

Chapter 3

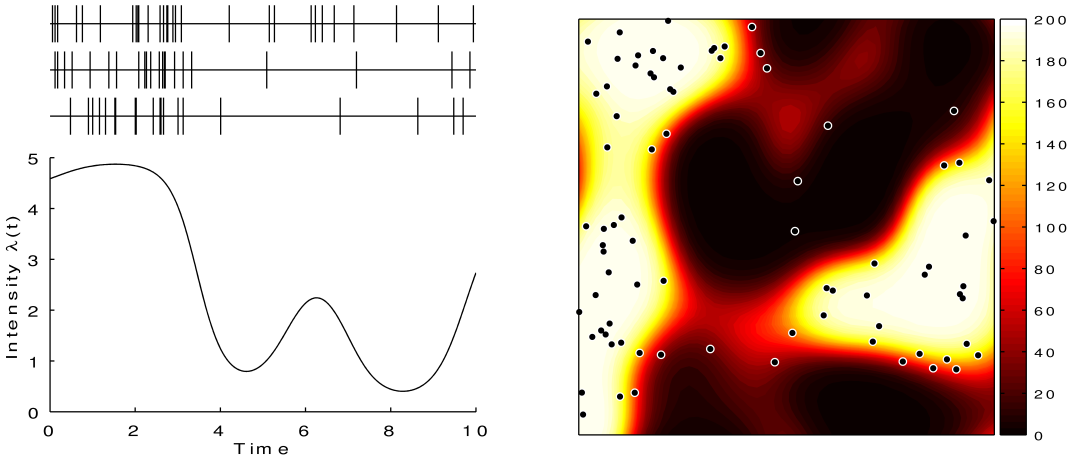
The Sigmoidal Gaussian Cox Process

This chapter reviews the Poisson process and constructs the *sigmoidal Gaussian Cox process* (SGCP), which is a Gaussian Cox process from which it is possible to generate exact data. I begin with a discussion of Poisson and Cox processes in Section 3.1. Section 3.2 reviews methods for generating Poisson event data. I introduce the sigmoidal Gaussian Cox process in Section 3.3 and present the associated generative procedure in Section 3.4. Section 3.5 examines a variety of Poisson-derived point processes and discusses how the SGCP can incorporate nonparametric inhomogeneity into them.

3.1 The Poisson and Cox Processes

As before, I consider the space $\mathcal{X} = \mathbb{R}^D$, and $\mathcal{V}$ is a bounded subset of $\mathcal{X}$. A *point process* on $\mathcal{V}$ is a random process that generates finite subsets of $\mathcal{V}$. I will denote one of these finite subsets as $\mathcal{S} \subset \mathcal{V}$ and refer to members of $\mathcal{S}$ as "events." A point process can be thought of as a random measure that takes nonnegative integer values, or a locally-finite *random counting measure*. A measure on $\mathcal{X}$ that is locally-finite has the property that the measure of any bounded subset of $\mathcal{X}$ is finite (Møller and Waagepetersen, 2004). Intuitively, a point process being locally finite means that for any bounded region of $\mathcal{X}$, the process will always realise a finite number of points. A Poisson process is a point process with a locally-integrable *intensity function* (or *rate* function) $\lambda(\boldsymbol{x}) : \mathcal{V} \to \mathbb{R}^+$, where $\mathbb{R}^+$ denotes the nonnegative real numbers. Local integrability is the property that $\int_{\mathcal{V}} \mathrm{d}\boldsymbol{x}\ \lambda(\boldsymbol{x}) < \infty$ for any bounded subset $\mathcal{V} \subset \mathcal{X}$ (Møller and Waagepetersen, 2004). The Poisson process is defined by two properties. First, the number of events in a subset $\mathcal{T}$ of $\mathcal{V}$ is Poisson-distributed with parameter

$$\Lambda_{\mathcal{T}} = \int_{\mathcal{T}} \mathrm{d}\boldsymbol{x}\ \lambda(\boldsymbol{x}). \tag{3.1}$$



(a) One-dimensional temporal Poisson process (b) Two-dimensional Poisson process

Figure 3.1: Examples of Poisson process realisations. (a) A one-dimensional Poisson process in time, along with three independent time series realisations. (b) A spatial Poisson process in two dimensions. The intensity function is shown by the colours and a realisation of points is overlaid.

As $\lambda(\boldsymbol{x})$ is locally finite, $\Lambda_{\mathcal{T}}$ is always finite. Second, the number of events in disjoint subsets of $\mathcal{V}$ are independent. This second property is called *independent scattering* (Møller and Waagepetersen, 2004).

Figure 3.1a shows an example of a one-dimensional Poisson process and three independently-drawn sets of events. Figure 3.1b shows an example of a two-dimensional Poisson process and a set of spatial events. Generally, in this chapter I will use $\lambda(t)$ when discussing the intensity function of a one-dimensional Poisson process, and use $\lambda(\boldsymbol{x})$ to refer to the intensity functions of Poisson processes with an arbitrary number of dimensions. I will use the notation $\mathcal{PP}(\mathcal{S} \mid \mathcal{V}, \lambda(\boldsymbol{x}))$ to refer to the probability density associated with a particular set of Poisson events $\mathcal{S}$ on a domain $\mathcal{V}$ with intensity function $\lambda(\boldsymbol{x})$:

$$\mathcal{PP}(\mathcal{S} = \{\boldsymbol{x}_k\}_{k=1}^{K} \mid \mathcal{V}, \lambda(\boldsymbol{x})) = \exp\left\{-\int_{\mathcal{V}} \mathrm{d}\boldsymbol{x}\ \lambda(\boldsymbol{x})\right\} \prod_{k=1}^{K} \lambda(\boldsymbol{x}_k). \tag{3.2}$$

3.2 Generating Poisson Data

There are several available methods for simulating data from a Poisson process. Their applicability depends on the specific form of $\lambda(\boldsymbol{x})$ and the domain $\mathcal{V}$.

3.2.1 Direct Sampling

The simplest case is when $\lambda(\boldsymbol{x})$ is a positive constant, i.e. $\lambda(\boldsymbol{x}) = c$, where $c > 0$. This is called the *homogeneous* Poisson process. To generate data from a homogeneous Poisson process on $\mathcal{V}$ with intensity c, first calculate the measure of the set $\mathcal{V}$, which I denote $\mu(\mathcal{V})$. Next, draw the *number* of events K from a Poisson distribution with parameter $c \cdot \mu(\mathcal{V})$, i.e. $K \sim \mathcal{Po}(c \cdot \mu(\mathcal{V}))$. Finally, draw the *locations* of the K events, $\{\boldsymbol{x}_k\}_{k=1}^K$, by distributing the points independently and uniformly within $\mathcal{V}$, i.e. $\boldsymbol{x}_k \sim \mathcal{U}(\mathcal{V})$.

For the *inhomogeneous* Poisson process, where $\lambda(\boldsymbol{x})$ varies over $\mathcal{V}$, it is possible to use a two-stage procedure to draw Poisson data on $\mathcal{V}$. First, integrate over the intensity function:

$$\Lambda_{\mathcal{V}} = \int_{\mathcal{V}} \mathrm{d}\boldsymbol{x}\ \lambda(\boldsymbol{x}). \tag{3.3}$$

Use $\Lambda_{\mathcal{V}}$ as the Poisson distribution parameter to determine the number of events in $\mathcal{V}$, via $K \sim \mathcal{Po}(\Lambda_{\mathcal{V}})$. Finally, distribute the events independently within $\mathcal{V}$ according to the probability density

$$f(\boldsymbol{x}) = \frac{1}{\Lambda_{\mathcal{V}}}\ \lambda(\boldsymbol{x})\ \boldsymbol{I}_{\mathcal{V}}(\boldsymbol{x}), \tag{3.4}$$

where $\boldsymbol{I}_{\mathcal{V}}(\boldsymbol{x})$ is a set-membership indicator function for $\mathcal{V}$ (Devroye, 1986, Theorem VI.1.5):

$$\boldsymbol{I}_{\mathcal{V}}(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \boldsymbol{x} \in \mathcal{V} \\ 0 & \text{otherwise} \end{cases}. \tag{3.5}$$

3.2.2 Time Rescaling

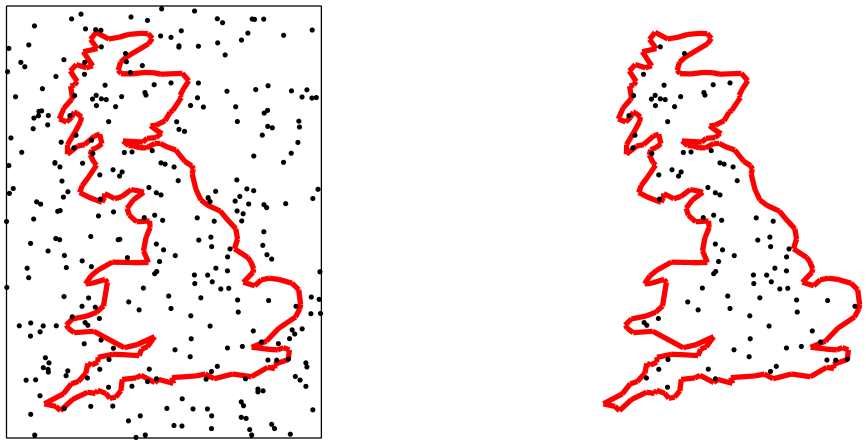
A useful property of the homogeneous Poisson process in one dimension is that the inter-event arrival times are exponentially distributed. That is, if

$$0 < t_1 < t_2 < \ldots < t_{j-1} < t_j < \ldots$$

are events from a Poisson process with constant rate c, then

$$t_j - t_{j-1} \sim \mathcal{E}x(c). \tag{3.6}$$

This *renewal-process* approach, also called the *exponential spacings method* (Devroye, 1986), provides an alternative way to generate data from a temporal homogeneous Poisson process. When a Poisson process has a time-varying intensity $\lambda(t)$, the renewal-process method can sometimes be used to generate inhomogeneous Poisson



(a) Poisson process on a bounding box.

(b) Poisson process on the irregular region.

Figure 3.2: Thinning allows simulation of Poisson processes on irregular shapes, such as this Lambert equal-area azimuthal projection of the island of Great Britain. On the left are points from a homogeneous Poisson process on a rectangle that bounds the shape. On the right, only points within the coastline have been kept.

realisations via *time rescaling* (Meyer, 1971; Papangelou, 1972). Taking the domain to be $[0, \infty)$, the *cumulative intensity function* is

$$\Lambda(t) = \int_0^t \mathrm{d}t' \; \lambda(t'). \tag{3.7}$$

If a sequence $0 < t_1 < t_2 < \ldots$ is a realisation from a unit-intensity homogeneous Poisson process, then the sequence $0 < \Lambda^{-1}(t_1) < \Lambda^{-1}(t_2) < \ldots$ is a realisation from an inhomogeneous Poisson process with cumulative intensity function $\Lambda(t)$ (Devroye, 1986, Theorem VI.1.4), where $\Lambda^{-1}(t)$ is the inverse of $\Lambda(t)$.

3.2.3 Thinning

Consider a realisation $\mathcal{S} = \{\boldsymbol{x}_k\}_{k=1}^K$ from a Poisson process with intensity $\lambda(\boldsymbol{x})$, and a function $\phi(\boldsymbol{x}) : \mathcal{V} \to [0, 1]$. For each of the K events in $\mathcal{S}$, make an independent draw from a Bernoulli distribution and remove the kth event from $\mathcal{S}$ with probability $1 - \phi(\boldsymbol{x}_k)$. Based on these independent random removals, a new set $\mathcal{S}'$ is formed. This new set $\mathcal{S}'$, composed of any remaining events, is a realisation from a Poisson process with intensity function $\lambda'(\boldsymbol{x}) = \phi(\boldsymbol{x}) \; \lambda(\boldsymbol{x})$. This procedure is called *independent thinning* (Lewis and Shedler, 1979).

Thinning bears a strong resemblance to rejection sampling, discussed in Section 2.2.1, and thinning can be used to generate Poisson process data from a complex intensity function $\lambda(\boldsymbol{x})$. Thinning requires a *dominating intensity function* $\bar{\lambda}(\boldsymbol{x})$ that provides an upper bound for $\lambda(\boldsymbol{x})$, i.e. $\bar{\lambda}(\boldsymbol{x}) \geq \lambda(\boldsymbol{x})$, $\forall \boldsymbol{x} \in \mathcal{V}$. Typically, $\bar{\lambda}(\boldsymbol{x})$ is selected so that it is easy to generate a Poisson realisation $\bar{\mathcal{S}} \sim \mathcal{PP}(\mathcal{V}, \bar{\lambda}(\boldsymbol{x}))$. The set $\bar{\mathcal{S}}$ can be turned

into a realisation $\mathcal{S}$ from $\lambda(\boldsymbol{x})$ by keeping events with independent Bernoulli probability $\phi(\boldsymbol{x}) = \lambda(\boldsymbol{x})/\bar{\lambda}(\boldsymbol{x})$.

In spatial problems, interesting domains are often irregularly-shaped. Thinning can be used to arrive at Poisson data that are drawn inside a region from which it would be difficult to sample directly. If the region of interest is $\mathcal{V}$, construct a bounding shape $\mathcal{V}' \supset \mathcal{V}$ on which it is easy to simulate Poisson data, e.g. a rectangle. Next simulate a Poisson process on $\mathcal{V}'$ and discard those events that did not fall within $\mathcal{V}$ by using $\phi(\boldsymbol{x}) = \boldsymbol{I}_{\mathcal{V}}(\boldsymbol{x})$, where $\boldsymbol{I}_{\mathcal{V}}(\boldsymbol{x})$ is a set-membership indicator function. Figure 3.2 provides an example using the coastline of Great Britain.

3.3 The Prior on Poisson Intensity Functions

As in Section 2.1, I consider a function $g(\boldsymbol{x}) : \mathcal{X} \to \mathbb{R}$, which is a realisation from a Gaussian process. I transform this function into a Poisson intensity function via

$$\lambda(\boldsymbol{x}) = \Phi(g(\boldsymbol{x}))\, \bar{\lambda}(\boldsymbol{x}), \tag{3.8}$$

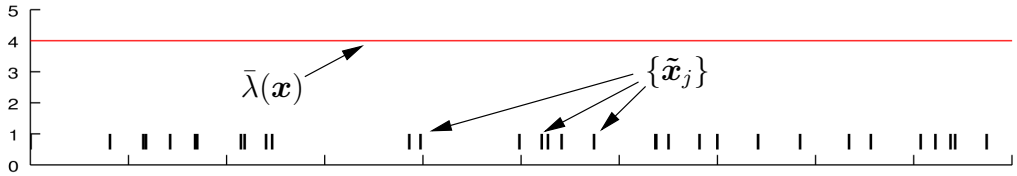
where $\Phi(\cdot)$ is a sigmoid function as in Section 2.1.1. As in Chapter 2, I will assume that $\Phi(\cdot)$ is the logistic function. The function $\bar{\lambda}(\boldsymbol{x})$ is a parametric intensity from which it is easy to generate Poisson process realisations, using, for example, the methods in Section 3.2.1. I call this prior the *sigmoidal Gaussian Cox process* (SGCP), as it is a Cox process where the intensity function is a sigmoidally-transformed draw from a Gaussian process.

Realisations from the prior implied by a Gaussian process prior on $g(\boldsymbol{x})$ and the transformation of Equation 3.8 are random positive functions bounded above by $\bar{\lambda}(\boldsymbol{x})$. As with the GPDS base density in Section 2.1.2, I allow for hyperparameters $\boldsymbol{\psi}_\lambda$ that determine the behaviour of $\bar{\lambda}(\boldsymbol{x})$. When $\boldsymbol{\psi}_\lambda$ is relevant, I will write $\bar{\lambda}(\boldsymbol{x}\,;\,\boldsymbol{\psi}_\lambda)$.

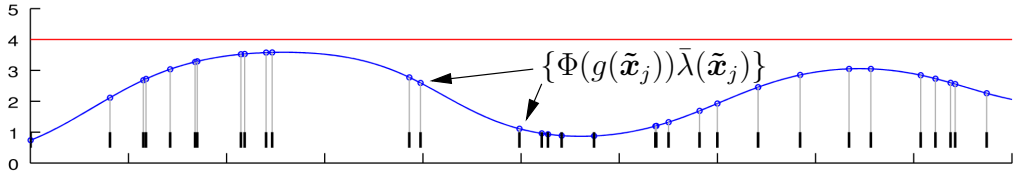
3.4 Generating Data from the Prior

I use the transformation of Equation 3.8 because it allows simulation of exact Poisson data from a random intensity function drawn from the prior provided by the Gaussian process. As in Chapter 2, by *exact* I mean that the data are not biased by an approximate sampling procedure. I use thinning, discussed in Section 3.2.3, to generate these data and augment the procedure to simultaneously sample the function $g(\boldsymbol{x})$ from the Gaussian process.

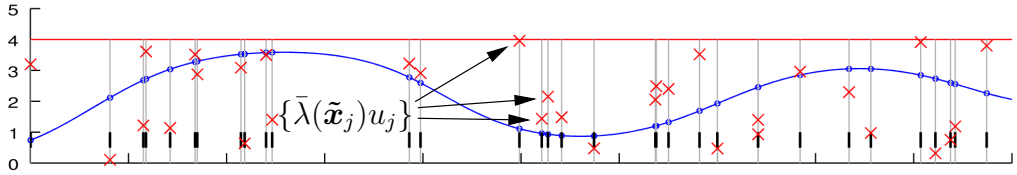
The objective is to generate a set of events $\{\boldsymbol{x}_k\}_{k=1}^{K}$ on $\mathcal{V} \subset \mathcal{X}$, where K is itself random, from a Poisson process with an intensity function $\lambda(\boldsymbol{x})$ that is the result of applying Equation 3.8 to a random function $g(\boldsymbol{x})$ drawn from the Gaussian process. This



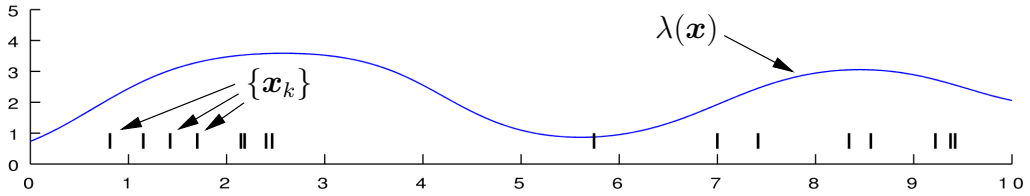
(a) Dominating intensity and initial events.



(b) Sampling the Gaussian process at the events.



(c) Sampling in the vertical coordinate for thinning.



(d) The kept events and random intensity function.

Figure 3.3: The generative procedure for the SGCP. (a) The (constant) dominating intensity $\bar{\lambda}(\boldsymbol{x}) = 4$ is shown, along with a Poisson series, $\{\tilde{\boldsymbol{x}}_j\}_{j=1}^J$, generated from it. (b) At each event, a sample is drawn from the Gaussian process, to get the set $\{g(\tilde{\boldsymbol{x}}_j)\}_{j=1}^J$. This function is squashed through the sigmoid function so that it is everywhere positive and upper-bounded by $\bar{\lambda}(\boldsymbol{x})$. (c) Variates $\{u_j\}_{j=1}^J$ are drawn uniformly on $(0, 1)$ in the vertical coordinate. d) If the jth uniform variate is greater than the random function value $\Phi(g(\tilde{\boldsymbol{x}}_j))$, then event $\tilde{\boldsymbol{x}}_j$ is discarded. The kept events are drawn from the inhomogeneous Poisson process corresponding to the random intensity $\lambda(\boldsymbol{x}) = \Phi(g(\boldsymbol{x}))\,\bar{\lambda}(\boldsymbol{x})$.

is done by first simulating a set of events $\{\tilde{\boldsymbol{x}}_j\}_{j=1}^J$ from a Poisson process on $\mathcal{V}$ with intensity $\bar{\lambda}(\boldsymbol{x})$, using the procedure of Section 3.2.1. Next, treat these $\{\tilde{\boldsymbol{x}}_j\}_{j=1}^J$ as input points for a Gaussian process and sample the function $g(\boldsymbol{x})$ at these locations to generate a corresponding set of function values, denoted $\{g(\tilde{\boldsymbol{x}}_j)\}_{j=1}^J$. Now use the thinning procedure to choose which $K \leq J$ events of $\{\tilde{\boldsymbol{x}}_j\}_{j=1}^J$ will be kept so that the remaining events, $\{\boldsymbol{x}_k\}_{k=1}^K$, are drawn from an inhomogeneous Poisson process with an intensity function $\lambda(\boldsymbol{x})$ consistent with the $\{g(\tilde{\boldsymbol{x}}_j)\}_{j=1}^J$ drawn from the Gaussian process. This is done by generating J uniform random variates on $(0, 1)$, denoted $\{u_j\}_{j=1}^J$. Only events for which $u_j < \Phi(g(\tilde{\boldsymbol{x}}_j))$ are kept. Any accepted events form the set $\{\boldsymbol{x}_k\}_{k=1}^K$. This procedure is shown in Algorithm 3.1 and graphically in Figure 3.3.

Algorithm 3.1 Generate a set of Poisson events from the SGCP prior.

Inputs:
- Domain $\mathcal{V}$
- Gaussian process covariance function $C(\boldsymbol{x}, \boldsymbol{x}' ; \boldsymbol{\theta})$
- Dominating intensity function $\bar{\lambda}(\boldsymbol{x} ; \boldsymbol{\psi}_\lambda)$

Outputs:
- Poisson events $\mathcal{S} = \{\boldsymbol{x}_k\}_{k=1}^K$ from an SGCP prior

1: $\{\tilde{\boldsymbol{x}}_j\}_{j=1}^J \sim \mathcal{PP}(\mathcal{V}, \bar{\lambda}(\boldsymbol{x} ; \boldsymbol{\psi}_\lambda))$ ▷ Draw Poisson events according to intensity $\bar{\lambda}(\boldsymbol{x} ; \boldsymbol{\psi}_\lambda)$.
2: $\{g(\tilde{\boldsymbol{x}}_j)\}_{j=1}^J \sim \mathcal{GP}(g \mid \{\tilde{\boldsymbol{x}}_j\}_{j=1}^J, \boldsymbol{\theta})$ ▷ Draw the function from the GP at the events.
3: $\mathcal{S} \leftarrow \emptyset$ ▷ Initialise the set of unthinned events.
4: **for** $j \leftarrow 1 \ldots J$ **do** ▷ Loop over the proposed events.
5: $u_j \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on the unit interval.
6: **if** $u_j < \Phi(g(\tilde{\boldsymbol{x}}_j))$ **then** ▷ Apply the thinning acceptance rule.
7: $\mathcal{S} \leftarrow \mathcal{S} \cup \tilde{\boldsymbol{x}}_j$ ▷ Add to the accepted events.
8: **end if**
9: **end for**
10: **return** $\mathcal{S}$

To generate data from the SGCP prior on an irregular region, a two-stage thinning procedure can be used. First, generate Poisson data for the bounding region as in Section 3.2.3. Next, discard events outside of the region. Sample the Gaussian process only at the remaining events and perform the accept/reject thinning as above. This procedure is provided in pseudocode in Appendix A as Algorithm A.1 (page 117).

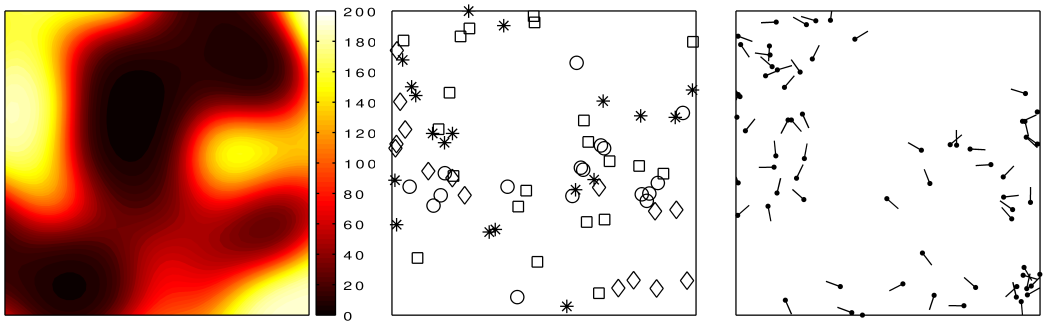
As in the Gaussian process density sampler, the generative procedure for the sigmoidal Gaussian Cox process makes it possible to simulate exact data from an infinite-dimensional random function without knowledge of the function at more than a finite number of locations and without integrating over the function. In Chapter 4 and Chapter 5, I will show that being able to generate data from the prior means that inference can be done tractably via Markov chain Monte Carlo.

3.5 Point Process Extensions to the SGCP

The Poisson process is a widely-used model for temporal and spatial data, but it is also the foundation for other types of point process with different properties. In this section I discuss how the sigmoidal Gaussian Cox process can be used to introduce nonparametric inhomogeneity into such processes to achieve marking and interaction.

3.5.1 The Marked Poisson Process

Given a set of events $\mathcal{S}$ drawn from a Poisson process, we might also wish to model characteristics of those events. For example, a Poisson process might be used to model trees in a forest, and layered upon that might be a probabilistic model for determining the species of each tree. A *marked Poisson process* is a distribution on the product space of finite subsets of the domain $\mathcal{V}$ (realisations from the Poisson process on $\mathcal{V}$),



(a) Spatial intensity function (b) Categorical marks (c) Real-valued marks

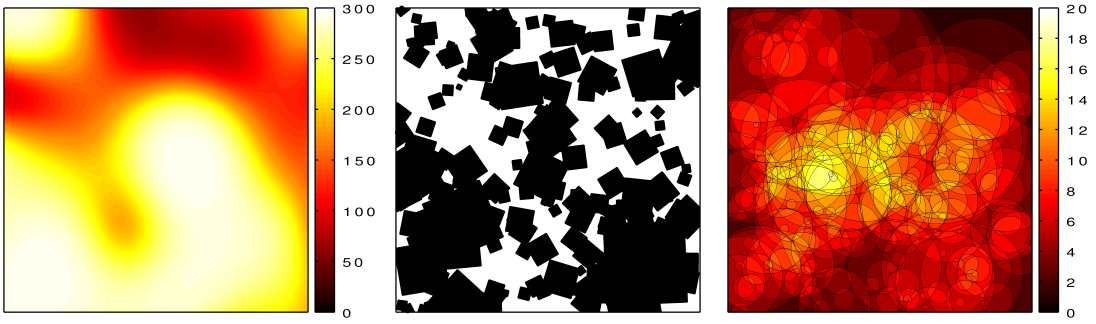
Figure 3.4: (a) A Poisson intensity function on the unit square. (b) A multitype Poisson process with four classes, shown as different marker types. This was drawn using the SGCP with the intensity function in (a). (c) A marked Poisson process with a random real-valued "orientation" characteristic. This was also drawn from an SGCP with the intensity function in (a).

and some additional space $\mathcal{Y}$ that is the *mark space*. The distribution on marks is conditioned on the event locations and the marks are independent, given the Poisson realisation. If $\mathcal{Y}$ is a finite set, then the construction is sometimes called a *multitype Poisson process* (Møller and Waagepetersen, 2004). Marked point processes with real-valued marks have been used, for example, to model the luminosity properties of galaxies (Beisbart and Kerscher, 2000). If $\boldsymbol{y}$ is a point in the mark space, i.e. $\boldsymbol{y} \in \mathcal{Y}$, then the joint distribution of a marked Poisson process realisation can be written as

$$p(\{\boldsymbol{x}_k, \boldsymbol{y}_k\}_{k=1}^K) = \mathcal{PP}(\{\boldsymbol{x}_k\}_{k=1}^K) \prod_{k=1}^K p(\boldsymbol{y}_k \,|\, \boldsymbol{x}_k). \tag{3.9}$$

From a simulation point of view, it is simple to turn the generative process for the SGCP into one that also has marks. After using Algorithm 3.1 to generate the events, draw a mark for each one from its conditional distribution. Figure 3.4b and Figure 3.4c show realisations from marked Poisson processes with categorical marks and real-valued marks, respectively. The associated intensity function is shown in Figure 3.4a. Algorithm A.2 (page 118) in Appendix A provides pseudocode for simulating a marked sigmoidal Gaussian Cox process.

The *Boolean*, or *germ-grain*, model is a marked Poisson process variant for generating random closed subsets of the domain $\mathcal{V}$. Typically, the Poisson event locations (germs) are the centres of random local compact subsets (grains), which are drawn from the marking distribution. The resulting random object is the union of these subsets. In two dimensions this can be viewed as a distribution on binary images, and in three dimensions as a model of volume occupancy. To extend the sigmoidal Gaussian Cox process to the Boolean model, generate Poisson events from a random intensity using Algorithm 3.1, then draw the marks that specify the properties of the local compact set associated with each event. Figure 3.5b shows an SGCP Boolean model with a



(a) Spatial intensity function (b) Union of random squares (c) Density from random circles

Figure 3.5: (a) A Poisson intensity function on the unit square. (b) A Boolean model realisation via the SGCP with squares of random size and orientation. The union of these squares creates a random subset of the unit square. The underlying Poisson intensity is as in (a). (c) A Boolean model realisation with overlapping circles of random radius. The density that results from accumulated set membership is shown in the colours.

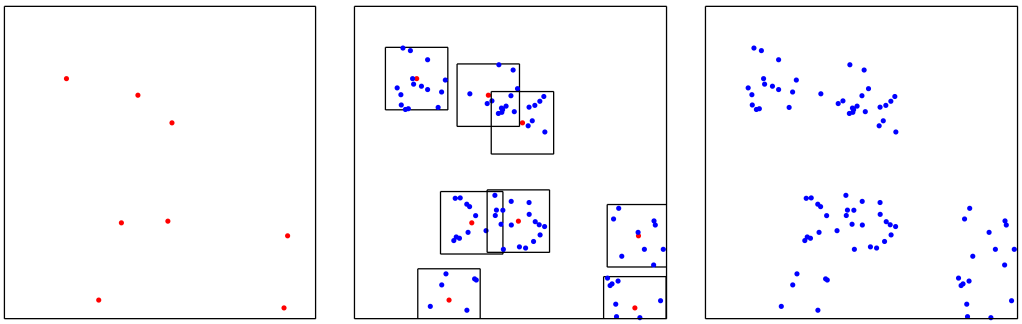
union of squares of random size and orientation. Similar models were introduced simultaneously by Kolmogorov (1937)[1], Johnson and Mehl (1939) and Avrami (1939, 1940, 1941) as descriptions of crystal growth. Figure 3.5c shows an alternative use of the Boolean model, where each local subset (in this case, determined by a circle of random size) contributes additively to a local density. The Poisson intensity that generated these processes is shown in Figure 3.5a. For an extensive discussion of the Boolean model, see Molchanov (1997).

3.5.2 Point Processes with Interaction

The defining characteristic of the Poisson process is that it has no interaction between the events — once the *number* of events has been determined, the *locations* are distributed independently. This independence is not always a realistic modeling assumption. It may be desirable for events to cluster together (*underdispersion*) or repel each other (*overdispersion*). In this section, I examine several models with interaction and show how the generative procedure of the sigmoidal Gaussian Cox process can be extended to sample from them.

The Neyman–Scott (Neyman and Scott, 1958), Hawkes (Hawkes, 1971a,b; Hawkes; Hawkes and Oakes, 1974), and Matérn (Matérn, 1960)[2] point process families are interesting from the point of view of exact simulation, because they are defined by their Poisson-based generative procedures and not in terms of their probability densities on data. This is in contrast to, for example, the Strauss process (Strauss, 1975; Kelly and Ripley, 1976) and more general Gibbs/Markov processes (see, e.g. Møller and Waagepetersen (2004), Ogato and Tanemura (1989), and Stoyan and Stoyan (1998)), which describe interactions between points via a potential function on configurations.

[1] Kolmogorov (1937) was translated into English in Kolmogorov (1992).
[2] Matérn (1960) was translated into English in Matérn (1986)



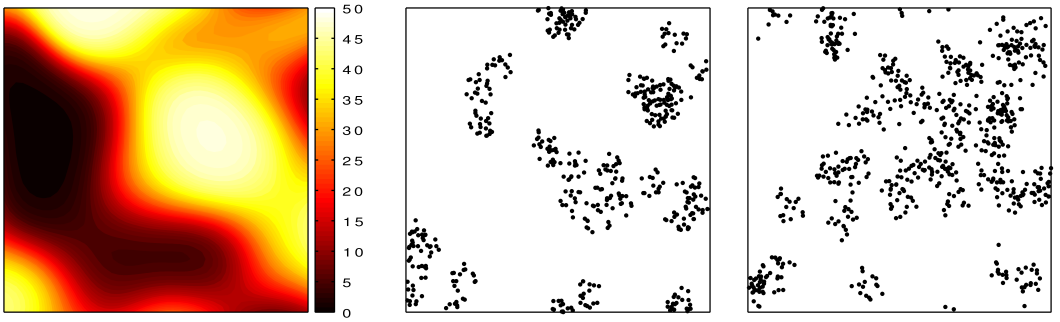
(a) Mother process realisation (b) Daughter process realisations (c) Final observed points

Figure 3.6: An example of how a mother-daughter clustering process works. (a) The mother process realises a set of points on the domain. (b) Daughter processes local to each mother point are used to generate daughter realisations. (c) Only the daughter points are kept.

There has been significant interest in perfect simulation via coupling from the past (CFTP) (Propp and Wilson, 1996) of homogeneous variants of Gibbs/Markov processes (Kendall, 1997; Häggström et al., 1999; Kendall and Møller, 2000; Wilson, 2000), but this thesis does not explore these methods.

Clustering Point Processes

Clustering point processes are those in which points tend to be more grouped than would be expected if they were from a Poisson process. A natural idea for a clustering point process is to use two stages: a "mother" process and a random set of "daughter" processes. The mother point process first realises a set of locations in the space and then the daughter processes are localised around those random locations. The realisations from the daughter processes are the observations. Figure 3.6 illustrates this idea graphically. Plants provide a motivating example for this kind of distribution: seeds tend to land and sprout near their parents. Brix and Chadœuf (2002) examine such a model for the distribution of weeds. There is a large taxonomy for these types of processes, depending on the exact nature of the mother and daughter processes, see e.g. Lawson and Dension (2002, Chapter 4). For example, if the daughter processes are Poisson, then the overall process falls into the broad class of *generalised shot noise Cox processes* described by Møller and Torrisi (2005). Here I am concerned with variants that have Poisson mother processes, as the sigmoidal Gaussian Cox process provides a drop-in replacement for the mother process to make it inhomogeneous. If the daughter processes are Poisson, then these extended models could be called "triply-stochastic" Poisson processes as they use the events from a Cox process to realise an additional random Poisson intensity.



(a) Mother SGCP intensity (b) Matérn cluster process (c) Truncated Thomas process

Figure 3.7: Realisations of Neyman–Scott processes with inhomogeneity in the mother Poisson process introduced via the SGCP. (a) The mother process intensity. (b) A Matérn cluster process realisation with daughter homogeneous intensity 2000 on a disc of radius 0.05. (c) A truncated Thomas process with isotropic Gaussians of $\sigma = 0.03$, truncated to a disc with radius 4σ.

The Neyman–Scott Process The *Neyman–Scott process* (Neyman and Scott, 1958) is the simplest example of a mother-daughter process: both mother and daughter are Poisson. Generally, the mother and daughter intensity functions are different. The daughter processes all typically have the same intensity function, except that they are centred on the mother points. As the Neyman–Scott process has Poisson daughter realisations, it is a Cox process (Stoyan and Stoyan, 1994; Møller and Waagepetersen, 2004) and the resulting intensity function has the form

$$\lambda^{\Sigma}(\boldsymbol{x}) = \sum_{j=1}^{J} \lambda^{\mathsf{dtr}}(\boldsymbol{x}\,;\, \boldsymbol{x}_j^{\mathsf{mtr}}) \tag{3.10}$$

where the set $\{\boldsymbol{x}_j^{\mathsf{mtr}}\}_{j=1}^{J}$ is drawn from the mother Poisson process. Usually, $\lambda^{\mathsf{dtr}}(\boldsymbol{x}\,;\, \boldsymbol{x}^{\mathsf{mtr}})$ is a scaled unimodal probability density function (kernel) centred on $\boldsymbol{x}^{\mathsf{mtr}}$. For example, the *Thomas process* (Thomas, 1949) is a Neyman–Scott process where the intensity function of each daughter is a scaled isotropic Gaussian distribution centred at the mother point, i.e. $\lambda^{\mathsf{dtr}}(\boldsymbol{x}\,;\, \boldsymbol{x}^{\mathsf{mtr}}) = \nu \cdot \mathcal{N}(\boldsymbol{x};\, \boldsymbol{x}^{\mathsf{mtr}}, \sigma^2 \boldsymbol{I}_D)$ for some variance σ^2 and total intensity $\nu > 0$. The *Matérn cluster process* (Matérn, 1960) is a Neyman–Scott process where each daughter intensity is uniform with intensity ν on a disc of radius z around the mother point.

The generative procedure of the sigmoidal Gaussian Cox process can be used to generate inhomogeneous Neyman–Scott cluster locations: use Algorithm 3.1 to generate the mother points, generate the daughter points as in the homogeneous case, and then keep the daughter points as the observations. Edge effects do introduce some complication, however, if mother points are allowed to be generated outside $\mathcal{V}$. Daughter points can appear inside $\mathcal{V}$ even if the mother point is outside the boundary, and so to make the claim that the generative procedure is *exact*, this must be taken into account. If the centred daughter kernel has compact support $\mathcal{Q}$, then first generate data on a

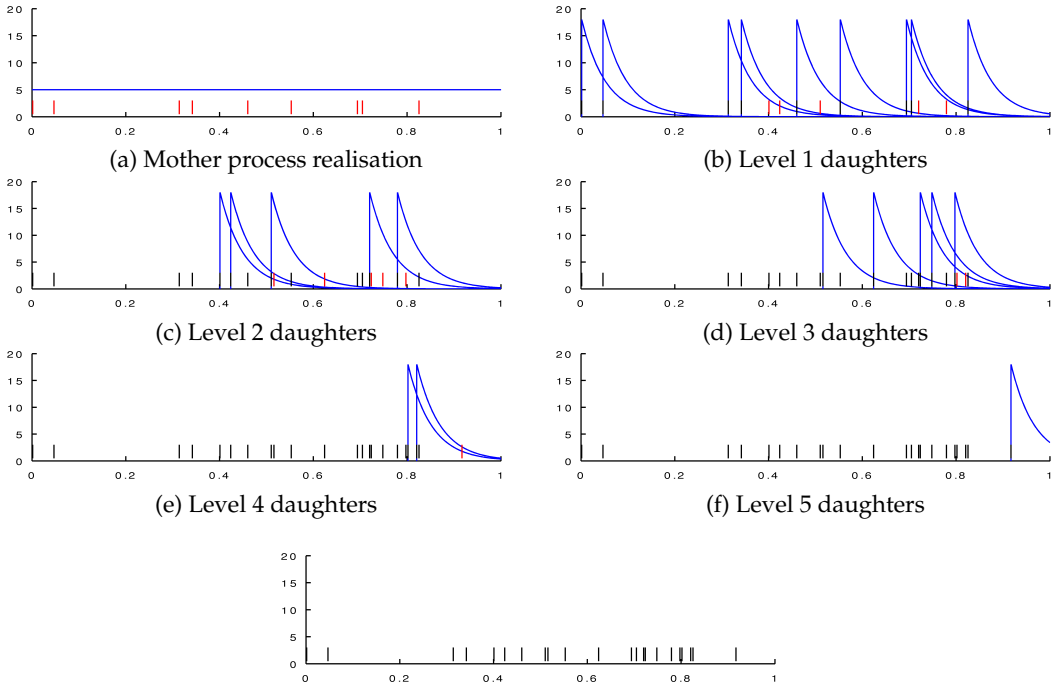
larger domain $\mathcal{V} \oplus \mathcal{Q}$, where the binary operator $\oplus$ indicates the Minkowski sum. It is not obvious how to draw exact samples from a Neyman–Scott process where the daughter kernel has unbounded support, as in the Thomas process: mother points could be arbitrarily far away and still have a non-zero probability of placing a daughter point within $\mathcal{V}$.

Algorithm A.3 (page 118) in Appendix A provides the inhomogeneous SGCP Neyman–Scott generation algorithm in pseudocode. Figure 3.7b shows an inhomogeneous Matérn cluster process on the unit square using the sigmoidal Gaussian Cox process with daughter intensity $\nu = 2000$ and radius $z = 0.05$. Figure 3.7c shows an SGCP-based Neyman–Scott process on the unit square where the daughters are isotropic Gaussians with $\sigma = 0.03$, scaled by a factor of 20 and truncated to a disc of radius $z = 4\sigma$. I refer to this tractable version of the Thomas process as the *truncated Thomas process*. Both of these realisations use the intensity in Figure 3.7a as the mother process.

These examples have used identical daughter processes, but this is not necessary. A marking process, as in Section 3.5.1 could be used to generate random daughter parameters. The approach I have presented could also be turned on its head and the sigmoidal Gaussian Cox process could be used to generate the daughter events rather than the mother events. This is a similar idea to the Archipelago cluster model I present in Chapter 8.

The Hawkes Process In the Neyman–Scott process, the mother points locate daughter processes, the daughter processes generate observations, and the process halts. The Hawkes process (Hawkes, 1971a,b; Hawkes; Hawkes and Oakes, 1974) extends this so that every daughter point also recursively spawns its own daughter Poisson process. Unlike the Neyman–Scott process, in which only the daughter points are observed, all points in the Hawkes process are observed. As long as the intensity function of each daughter Poisson process meets the requirement that $\int_{\mathcal{X}} \mathrm{d}\boldsymbol{x}\ \lambda^{\mathsf{dtr}}(\boldsymbol{x}) < 1$, then on a finite domain the Hawkes process will generate a finite number of points with probability one. The Hawkes process is referred to as a *self-exciting* or *contagious* process, and it has been applied to situations in which the presence of an event causes additional events to be more likely. Example application areas are seismology (Hawkes and Adamopoulos, 1973), neural spiking (Johnson, 1996), and financial markets (Giesecke and Kim, 2007). Figure 3.8 provides a graphical illustration of the procedure for generating data from a Hawkes process.

In the Neyman–Scott process, it is possible to avoid edge effects in simulation by using compact daughter kernels and augmenting the simulation domain for the mother process. In the Hawkes process, edge effects cannot be avoided without additional assumptions, as even compact kernels can lead to events arbitrarily far away from the original mother points. In a temporal setting this can be avoided by declaring *a pri-*



(a) Mother process realisation

(b) Level 1 daughters

(c) Level 2 daughters

(d) Level 3 daughters

(e) Level 4 daughters

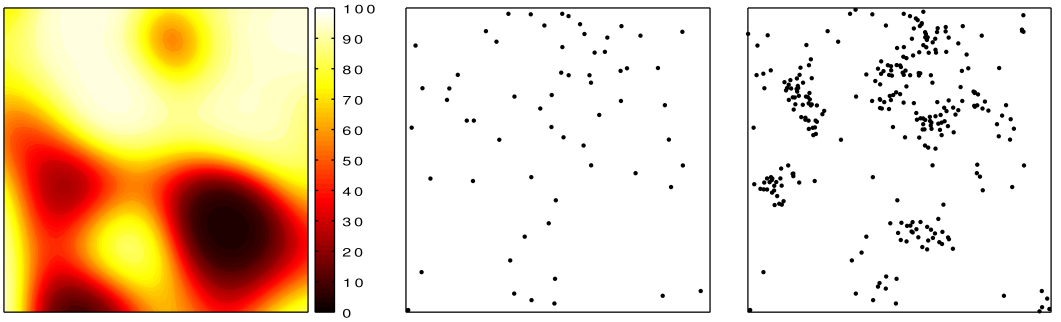
(f) Level 5 daughters

(g) Final observations

Figure 3.8: Sampling from a Hawkes process on the unit interval. In each figure, the new points are shown in red. (a) The root mother Poisson process is homogeneous with intensity 5 and there are nine events. (b) At each of the events in each realisation, a new Poisson process is instantiated, in this case with $\lambda(t) = 18e^{-20t}$. New points are drawn from these daughter processes. (c)-(f) Points and daughter processes are created recursively. (g) The Hawkes process terminates and all observations are kept.

ori that no mother events occur before $t = 0$ and that excitation is always causal, i.e. daughters can only assign nonzero intensity to times in the future. Exact simulation of a spatial Hawkes process appears to require either periodic boundary conditions or the model assumption that no mother points can occur outside $\mathcal{V}$. With compact kernels, daughters could also be limited to finite-depth recursion.

There are two places in the Hawkes process where the sigmoidal Gaussian Cox process could be introduced. The first is in the root mother process, where inhomogeneity from the SGCP could make a cascade of events more likely in some areas than in others. A realisation of this case is shown in Figure 3.9, and pseudocode is provided in Algorithm A.4 (page 119) in Appendix A. Another way to use the sigmoidal Gaussian Cox process in the Hawkes process is in the daughter processes. A single conditioning set could be used for the Gaussian process in the SGCP, resulting in Hawkes data with a single, shared random daughter intensity. If the dominating intensity $\bar{\lambda}(\boldsymbol{x})$ was chosen so that $\int_{\mathcal{X}} \mathrm{d}\boldsymbol{x}\ \bar{\lambda}(\boldsymbol{x}) \leq 1$, then the cascade would terminate with probability one for all intensities supported by the prior. Such a construction could be useful in a temporal setting if the self-excitement had an unknown time constant. For example, trades in a market might induce other trades, but there may be a propagation delay between a trade being placed and it being observed by other market participants.



(a) Mother SGCP intensity (b) Root mother realisation (c) Final Hawkes observations

Figure 3.9: A realisation from a Hawkes process on the unit square with an SGCP for the mother process. (a) The SGCP intensity function realised for the mother process. (b) The mother realisations, restricted only to the unit square. (c) All Hawkes observations. The realisation had a depth of fourteen. The daughter processes were Matérn discs of radius $z = 0.05$, each with an integrated intensity of 0.99.

Repulsive Point Processes

While clustering processes such as the Neyman–Scott and Hawkes address the notion that points in some models should tend to be near each other, *repulsive* point processes model the situation in which points should not congregate too closely. For example, many biological phenomena compete for local resources and are spatially overdispersed as a result. Glass and Tobler (1971) used a repulsive model to study the distribution of cities in Spain. In a temporal setting, neural spikes tend to be overdispersed due to the presence of refractory periods. Note that point process data can be both underdispersed and overdispersed: neural spikes might tend to come in clusters, but the individual spikes within the clusters may repel each other.

Repulsive point processes are generally categorised into two major types: *soft-core* models and *hard-core* models. In the hard-core case, there is a radius of interaction and points cannot be closer to each other than their radii allow. This can be thought of as a random placement of spheres, where the objects have an extent. In the soft-core model, objects simply *resist* being placed close to one other.

Among hard-core point process models, the most common is the *simple sequential inhibition process* (SSI) (Ripley, 1981; Stoyan and Stoyan, 1994; Venables and Ripley, 1998; Diggle, 2003). In this process, a hard-core radius z is chosen and points are added sequentially to the domain $\mathcal{V}$. At each step, a proposal is made to add a point at a random location uniformly within $\mathcal{V}$. If the proposal is within a distance z of an already-accepted point, then the new proposal is rejected. Otherwise, the point is accepted and a new proposal is made. This process continues until the desired number of points have been accepted or until no further points can be added. Penttinen and Stoyan (1989) extended the model to arbitrary overlapping sets. I will not address the SSI model in this thesis, as it is not rooted in a Poisson process and is very inflexible in

that it conditions *a priori* on the number of points in $\mathcal{V}$.

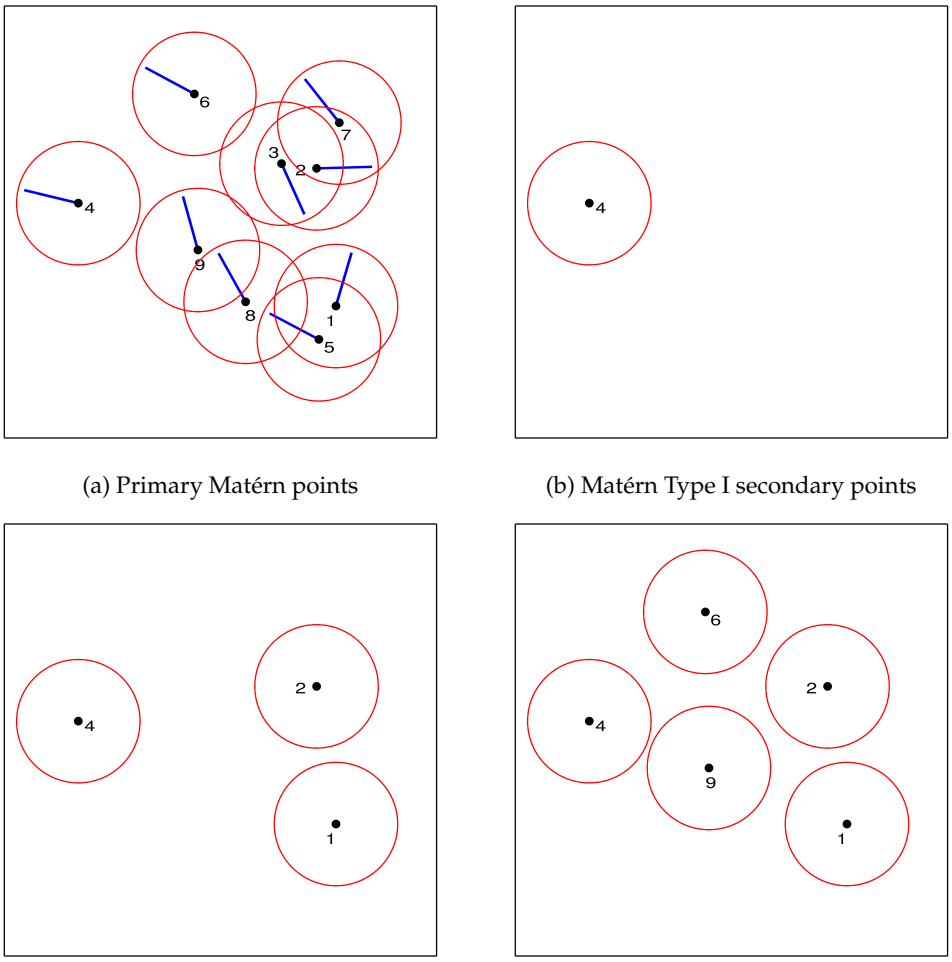
More interesting are the Poisson-based hard-core repulsive models of Matérn (1960). Matérn examined methods for two hard-core processes, now known as *Matérn Type I* and *Matérn Type II* processes. He also briefly described a third process, elaborated upon by Huber and Wolpert (2009), and now referred to as the *Matérn Type III* process. Each of these processes begins with a domain $\mathcal{V}$ and the generation of a set of Poisson events $\tilde{\mathcal{S}} = \{\tilde{\boldsymbol{x}}_j\}_{j=1}^{J}$. The events $\tilde{\mathcal{S}}$ are referred to as the *primary* points, and these are thinned to arrive at a set of *secondary* points, which are the observations $\mathcal{S} = \{\boldsymbol{x}_k\}_{k=1}^{K}$. There are three different rules for the primary-to-secondary thinning.

The Matérn Type I Process The Type I process is the simplest of the three and results in the lowest density of points. Events are thinned according to the following rule: if two points $\tilde{\boldsymbol{x}}_j$ and $\tilde{\boldsymbol{x}}_{j'}$ are closer than z to each other, then they both are excluded from the secondary set. This process has the pathological behaviour that only intermediate primary intensities are likely to result in many secondary realisations. If the primary intensity is low, then few primary events will occur, but if the intensity is high, then events are likely to overlap with other events and be thinned.

The Matérn Type II Process The Type II process is the most widely-used of the Matérn repulsive processes, and it introduces auxiliary "timestamps" for each of the primary points via marking. These timestamps are drawn uniformly and independently from $(0, 1)$, so that the process generates the set $\{\tilde{\boldsymbol{x}}_j, u_j\}_{j=1}^{J}$, where $u_j \sim \mathcal{U}(0, 1)$. The purpose of the timestamps is to induce an ordering on the primary points. The thinning rule is now that for a point to be added to the secondary set, there must be no *earlier* point which is closer than z. That is, $\tilde{\boldsymbol{x}}_j$ is thinned only if there exists a point $\tilde{\boldsymbol{x}}_{j'}$ for which $||\tilde{\boldsymbol{x}}_{j'} - \tilde{\boldsymbol{x}}_j|| < z$ and $u_{j'} < u_j$. If the point is not thinned, then it is placed in the secondary set. The Type II procedure yields a higher density than the Type I process and does not share the aforementioned pathology.

The Type II process has been generalised in several ways. Penttinen and Stoyan (1989) used the same timestamp-based thinning rule, but defined interaction as intersection of a random line segment from a marking process, rather than intersection of circles with radius $z/2$. Stoyan and Stoyan (1985) added a marking procedure so that each primary point also had a random radius z_j, with the objective of a soft-core effect. The random radii also generate the ordering of the points so that a point $\tilde{\boldsymbol{x}}_j$ is thinned only if there exists another point $\tilde{\boldsymbol{x}}_{j'}$ for which the random radius $z_{j'} > z_j$ and the points are closer than the larger radius $z_{j'}$.

The Matérn Type III Process The Type III process is exactly like the Type II process, except that a point is thinned only if it is close to an earlier point in the *secondary* set.



(a) Primary Matérn points

(b) Matérn Type I secondary points

(c) Matérn Type II secondary points

(d) Matérn Type III secondary points

Figure 3.10: Thinning in each of the Matérn process types. (a) The primary points are shown as black dots. The blue lines show the timestamps, in "clock-face ordering" so that 12:00 is earliest and 11:59 is the latest. The circles are also numbered according to the timestamps. The circles have radii of half the interaction distance. (b) Type I thinning: all circles which overlap any other circle are removed. (c) Type II thinning: when circles overlap, the later one is removed. (d) Type III thinning: a circle is removed only if it overlaps with an earlier secondary point.

This results in higher densities than the Type I or Type II variants. Huber and Wolpert (2009) develop a CFTP approach to this model, studying the Spanish towns data of Glass and Tobler (1971) and the pine sapling data of Ripley (1981). Inspired by Huber and Wolpert (2009, Figure 1), Figure 3.10 shows how a single set of points would be thinned by each of the processes.

Generalised Matérn Processes It is often desirable to have more flexibility than the hard-core process allows. For example, we might wish to have non-isotropic interactions, or to have points only repel each other stochastically. A wide variety of such soft-core models have been proposed via potential functions (Strauss, 1975; Kelly and

Ripley, 1976; Ogato and Tanemura, 1989; Stoyan and Stoyan, 1998), but there are not many that are defined in terms of the generative process. To fill this gap, I propose a general framework for soft- and hard-core repulsive Matérn processes that I call *generalised Matérn processes*. These processes are Matérn Type II or Type III processes, but rather than deterministically removing points, I propose the use of a *repulsion kernel*, $\rho(\boldsymbol{x} \nleftarrow \boldsymbol{x}') : \mathcal{X} \times \mathcal{X} \to [0, 1]$. Whereas previously an event would be thinned if it was within z of an earlier primary point (in the Type II process) or an earlier secondary point (in the Type III process), I now remove it with probability

$$p(\text{remove } \tilde{\boldsymbol{x}}_j) = 1 - \prod_{j' \text{ s.t. } u_{j'} < u_j} (1 - \rho(\tilde{\boldsymbol{x}}_j \nleftarrow \tilde{\boldsymbol{x}}_{j'})), \tag{3.11}$$

where the set of earlier points is chosen appropriately. This repulsion kernel gives earlier points a probability of vetoing a later point, depending on their relative locations. The original hard-core processes can be recovered via

$$\rho_{\mathsf{hc}}(\boldsymbol{x} \nleftarrow \boldsymbol{x}'; z) = \begin{cases} 1 & \text{if } ||\boldsymbol{x} - \boldsymbol{x}'|| < z \\ 0 & \text{otherwise} \end{cases}. \tag{3.12}$$

If the kernel $\rho(\boldsymbol{x} \nleftarrow \boldsymbol{x}')$ has unbounded support, i.e. there exists no z for which $\rho(\boldsymbol{x} \nleftarrow \boldsymbol{x}') = 0$ when $||\boldsymbol{x} - \boldsymbol{x}'|| > z$, then any point an arbitrary distance away can potentially veto any later point. This long-distance interaction causes edge effects that make perfect simulation from the prior more difficult, as I described with the Neyman–Scott and Hawkes processes. As such, I will only discuss kernels with compact support on the z-ball, i.e. $\rho(\boldsymbol{x} \nleftarrow \boldsymbol{x}') = 0, \forall \boldsymbol{x}, \boldsymbol{x}'$ such that $||\boldsymbol{x} - \boldsymbol{x}'|| > z$. I will use the notation $\mathcal{B}_z(\boldsymbol{x})$ to indicate the ball of radius z centred at $\boldsymbol{x}$.

Two such kernels that are potentially useful are the compact squared-exponential kernel:

$$\rho_{\mathsf{se}}(\boldsymbol{x} \nleftarrow \boldsymbol{x}'; z, \boldsymbol{\ell}) = \exp\left\{ -\frac{1}{2} \sum_{d=1}^{D} \frac{(x_d - x'_d)^2}{\ell_d^2} \right\} \boldsymbol{I}_{\mathcal{B}_z(\boldsymbol{x})}(\boldsymbol{x}'), \tag{3.13}$$

and the Laplacian kernel:

$$\rho_{\mathsf{lap}}(\boldsymbol{x} \nleftarrow \boldsymbol{x}'; z, \boldsymbol{\ell}) = \exp\left\{ -\sum_{d=1}^{D} \frac{||x_d - x'_d||}{\ell_d} \right\} \boldsymbol{I}_{\mathcal{B}_z(\boldsymbol{x})}(\boldsymbol{x}'). \tag{3.14}$$

It is useful to examine the effect of this soft-core behaviour under very simple circumstances. Figure 3.11 shows a Poisson time series that has been thinned via several different Matérn-like methods, along with histograms of the inter-event intervals. The timestamps used were the natural ones in the temporal setting, resulting in only causal inhibition. The hard-core processes prevent any events within intervals smaller than the hard-core radius. The soft-core processes thin out many of the smaller intervals,

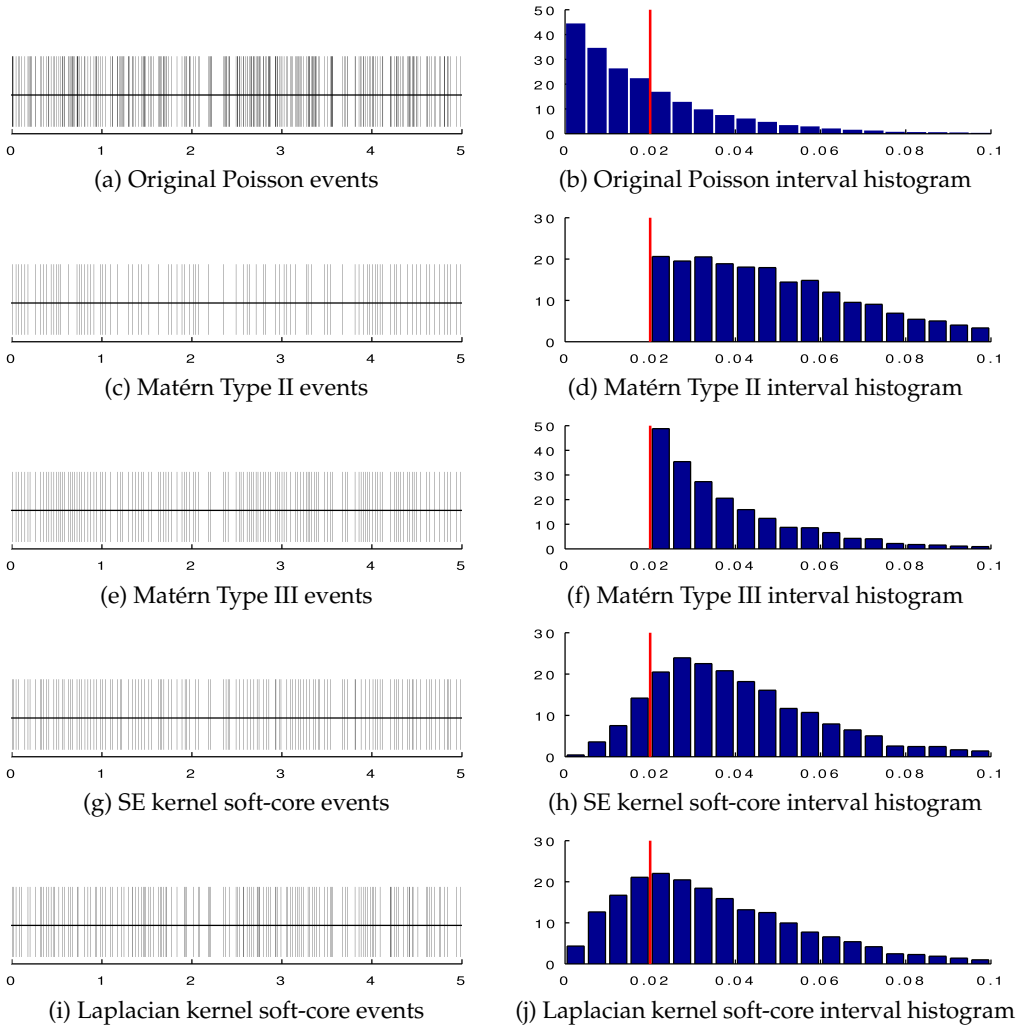
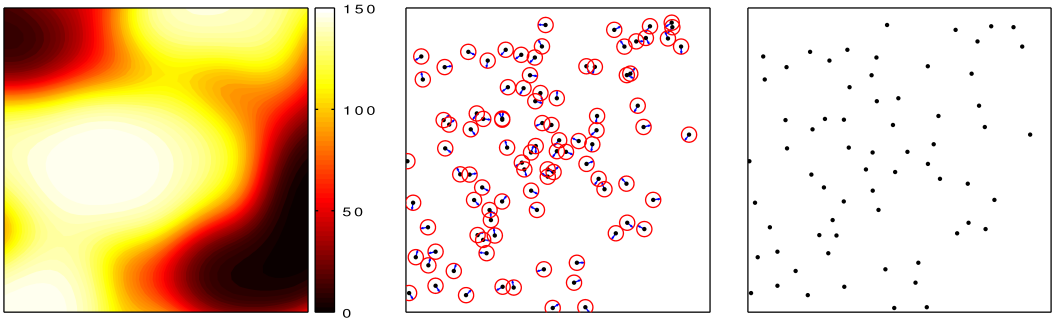


Figure 3.11: Differences in thinning for four Matérn variants. The data were generated from a Poisson process with a homogeneous rate of 50 on an interval $[0, 200]$. The left-hand figures show events from the subinterval $[0, 5]$. The right-hand figures show histograms of the intervals between events. The timestamp ordering used was the same as the arrival times. The red lines in each of the histograms shows the hard-core radius or soft-core length scale. (a,b) The original Poisson distribution. It has exponentially-distributed intervals. (c,d) The hard-core Matérn Type II process with radius $z = 0.02$. (e,f) The hard-core Matérn Type III process with radius $z = 0.02$. (g,h) A soft-core Matérn Type III process with a squared-exponential repulsion kernel of length scale $\ell = 0.02$. (i,j) A soft-core Matérn Type III process with Laplacian kernel of length scale $\ell = 0.02$.

but not all. Figure 3.11h and Figure 3.11j hint that the soft-core model may be useful for modeling the *inter-spike intervals* (ISI) of refractory neurons. The *gamma renewal process* (Yannaros, 1988) is a popular model for neural spike interval processes (e.g. Stein (1965); Teich et al. (1978); Leng et al. (2001)) and the soft-core repulsive Matérn process has qualitatively similar properties.

Simulation of Matérn Repulsive Processes The sigmoidal Gaussian Cox process can be inserted into the Matérn repulsive processes in place of the homogeneous Pois-



(a) Primary SGCP intensity (b) Primary realisation (c) Secondary realisation

Figure 3.12: A realisation from a Matérn Type III process on the unit square with an SGCP for the primary process. (a) The SGCP intensity function realised for the primary process. (b) The primary realisations shown as black dots, with red circles showing the hard cores and the blue lines showing the ordering as clock faces. (c) The secondary realisations after Type III thinning is performed.

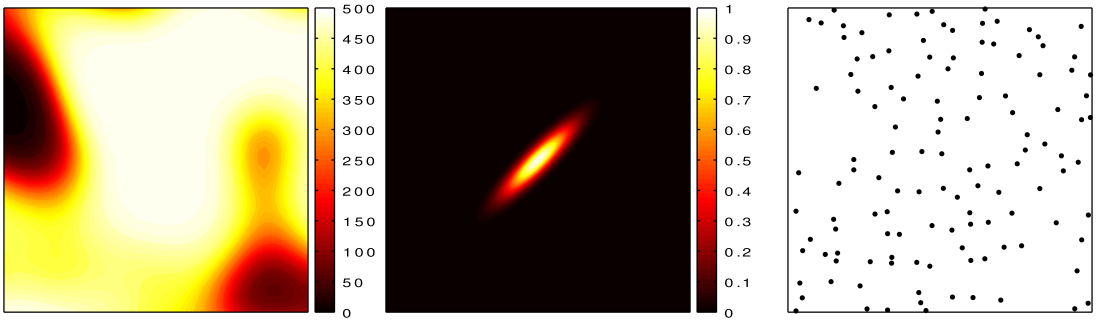
son process that generates the primary points. This allows the generation of inhomogeneous realisations with a Gaussian process prior on the intensity. As in the Neyman–Scott and Hawkes processes, however, it is necessary to address edge effects. The Type I process simply requires expanding the simulation region with the Minkowski sum, as in the Neyman–Scott process. The Type II and III processes (and the soft-core variants I propose) have similar difficulties to the Hawkes process. Simply augmenting the simulation region is not sufficient, as a sequence of overlapping regions could go arbitrarily far before a decisive veto occurs. Huber and Wolpert (2009) resolve this with periodic boundary conditions, but it could also be stated *a priori* that the primary process has zero intensity outside the region of interest. Figure 3.12 shows the result of applying the SGCP to the hard-core Matérn Type III process, using the zero-intensity boundary conditions. Pseudocode is provided for the generalised Matérn Type II process in Algorithm A.5 (page 119) and for the Type III variant in Algorithm A.6 (page 120), both in Appendix A. Figure 3.13 demonstrates the sigmoidal Gaussian Cox process applied to the soft-core Type III variant with a non-isotropic kernel

$$\rho(\boldsymbol{x} \nleftarrow \boldsymbol{x}') = \exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}')^{\mathsf{T}}\boldsymbol{W}(\boldsymbol{x}-\boldsymbol{x}')\right\}\boldsymbol{I}_{\mathcal{B}_r(\boldsymbol{x})}(\boldsymbol{x}'), \tag{3.15}$$

parameterised by positive definite $\boldsymbol{W}$, which I set to

$$\boldsymbol{W} = \begin{bmatrix} 1000 & -900 \\ -900 & 1000 \end{bmatrix}. \tag{3.16}$$

The matrix $\boldsymbol{W}$ places a Mahalanobis distance metric (Mahalanobis, 1936) on the space, from which repulsion is determined. This kernel is inspired by Vivarelli and Williams (1999), who used a similar construction for dimensionality reduction in Gaussian process regression.



(a) Primary intensity function (b) Repulsion kernel (c) Secondary points

Figure 3.13: A realisation from the soft-core Matérn Type III process, using the kernel of Equation 3.15 with $\boldsymbol{W}$ as in Equation 3.16. (a) The SGCP intensity function on the unit square. (b) The non-isotropic repulsion kernel, centred on the unit square. (c) A realisation from the process.

3.6 Summary

This chapter introduced a Cox process that arises from transforming a Gaussian process into a distribution on Poisson intensity functions. I named this process the sigmoidal Gaussian Cox process. The utility of the SGCP lies in the fact that it is possible to simulate data exactly from a Poisson process with a random intensity function drawn from the Gaussian process. I also reviewed a variety of extensions to the Poisson process for incorporation of marking, clustering and repulsion, and showed how the sigmoidal Gaussian Cox process can be extended to introduce inhomogeneity into these processes.

In Chapter 4, I develop an inference method for the sigmoidal Gaussian Cox process prior. Chapter 5 proposes an alternative, and superior, method for inference in the SGCP. This inference method can also be applied to some of the extensions described in Section 3.5. In Chapter 6, I will use these models to examine real-world point process data.

Chapter 4

Inference Via Exchange Sampling

In this chapter, I give an overview of the problem of inference with doubly-intractable posterior distributions. I present the method of *exchange sampling* (ES), which allows sampling from doubly-intractable posteriors under certain circumstances. I show how the Gaussian process density sampler of Chapter 2 and the sigmoidal Gaussian Cox process of Chapter 3 both meet exchange sampling's requirements and enable tractable inference via Markov chain Monte Carlo.

In Section 4.1, I review Markov chain Monte Carlo, before introducing the specialised MCMC method of exchange sampling in Section 4.2. In Section 4.3, I apply exchange sampling to inference in the Gaussian process density sampler, first using a simple construction for ease of understanding, and then with a more complex but efficient method. Following this, I apply exchange sampling to inference in the sigmoidal Gaussian Cox process in Section 4.4. In Section 4.5, I describe how these Markov chain Monte Carlo methods could be extended to sample from the predictive distributions.

Before discussing doubly-intractable inference, I will first review Markov chain Monte Carlo. In Section 2.2.1, I reviewed the rejection sampling method for generating data from a density function $p(\boldsymbol{x})$. This method is useful for implementing a generative process associated with the prior on density functions, as samples generated via rejection sampling are exact and independent. Rejection sampling, however, has undesirable properties in complex problems: it does not typically scale well to high dimensions and it requires specification of an upper-bounding density function. Often in Bayesian inference it is not critical that posterior samples be independent, however. For example, if we are using the samples to estimate an expectation, then correlation between the samples can be tolerated. In such cases, a broad class of efficient sampling algorithms are applicable, based on Markov chains.

4.1 Markov Chain Monte Carlo

In this section, I examine the problem of sampling efficiently from a distribution $p_\gamma(\gamma)$. I intend for γ to be interpreted as the parameters of a hierarchical model, and $p_\gamma(\gamma)$ to be treated as a posterior distribution. This is in contrast to the previous discussion of rejection sampling with a density $p(\boldsymbol{x})$. I make this change of notation to emphasise that rejection sampling is something done on the *data space*, while I use Markov chain Monte Carlo for inference in *parameter space*. Later in this thesis, this difference will become important as I will treat rejection samplers as objects to be modeled.

In Markov chain Monte Carlo, rather than specifying a computational procedure on the distribution directly, one defines a transition operator for exploring the state space Γ by moving stochastically from one point to another in time. By carefully choosing the transition operator $T(\gamma^{(t+1)} \leftarrow \gamma^{(t)})$ and ensuring that it meets certain requirements, it is possible to construct a Markov chain that has $p_\gamma(\gamma)$ as its *equilibrium* (also called *stationary* or *invariant*) distribution. MCMC methods are used by simulating a sequence of states from a Markov chain $\gamma^{(0)}, \gamma^{(1)}, \ldots, \gamma^{(T)}$, and hoping that, after running the chain long enough, the state has evolved to be very close to a draw from the equilibrium distribution. That is, one hopes that the simulated Markov chain has "forgotten" its starting state $\gamma^{(0)}$ and that the state $\gamma^{(T)}$ is representative of the distribution $p_\gamma(\gamma)$. For more comprehensive introductions to Markov chain Monte Carlo methods, see, e.g. Gelman et al. (2004, Chapter 11), MacKay (2003, Chapter 29), or Andrieu et al. (2003).

The first requirement of the transition operator is that it be Markovian: the distribution over the next state must depend only on the current state. To emphasise this, I will generally not denote the time indices explicitly and will denote the current state $\gamma^{(t)}$ by γ and the state $\gamma^{(t+1)}$ as $\hat{\gamma}$. The transition operator is then written $T(\hat{\gamma} \leftarrow \gamma)$.

Secondly, for Markov chain Monte Carlo to be useful, the transition operator must have $p_\gamma(\gamma)$ as its equilibrium distribution. This is the requirement that

$$p_\gamma(\hat{\gamma}) = \int_\Gamma \mathrm{d}\gamma \; T(\hat{\gamma} \leftarrow \gamma) \; p_\gamma(\gamma). \tag{4.1}$$

This equation can be interpreted as the requirement that when marginalising over the "origin" of the operator, assuming the origin is from $p_\gamma(\cdot)$, the resulting distribution on "destinations" must be $p_\gamma(\cdot)$. For simulation, it is useful to note that the generalised Kullback–Leibler (KL) divergence between $p_\gamma(\hat{\gamma})$ and $p_\gamma(\gamma)$ in Equation 4.1 cannot increase as the result of such a Markov transition (Cover and Thomas, 2006, Section 2.9). Therefore, regardless of the initial distribution of $\gamma^{(0)}$, the distribution over $\gamma^{(t)}$ for $t > 0$ cannot move "farther away" from the equilibrium distribution.

One way of achieving the invariance property of Equation 4.1 is to satisfy the stronger

condition of *detailed balance*:

$$T(\gamma \leftarrow \hat{\gamma})\, p_\gamma(\hat{\gamma}) = T(\hat{\gamma} \leftarrow \gamma)\, p_\gamma(\gamma) \tag{4.2}$$

for all γ and $\hat{\gamma}$ in Γ. The MCMC methods that I discuss will generally satisfy detailed balance, as in later chapters it will be a convenient identity to exploit for some Monte Carlo-based estimations. For a discussion of Markov chain Monte Carlo methods that do not satisfy detailed balance, see Neal (2004).

Finally, the Markov chain resulting from $T(\hat{\gamma} \leftarrow \gamma)$ must be *ergodic*. Ergodicity requires that the chain be *irreducible*: it is possible to eventually transition to any state where $p_\gamma(\gamma)$ is nonzero from any other supported state, given enough time. Ergodic chains must also be *aperiodic*, which is the requirement that the visitation to a given state is not limited to certain times.

Different transition operators result in different Markov chain Monte Carlo algorithms. There are many variants: slice sampling (Neal, 2003b), Gibbs sampling (Geman and Geman, 1984), and Hamiltonian Monte Carlo (Duane et al., 1987), to name a few. I will focus on the Metropolis–Hastings (MH) method (Metropolis et al., 1953; Hastings, 1970), which includes several other MCMC methods as special cases.

The Metropolis–Hastings algorithm has the transition operator

$$T(\hat{\gamma} \leftarrow \gamma) = \min\left(1, \frac{q(\gamma \leftarrow \hat{\gamma})\, p_\gamma(\hat{\gamma})}{q(\hat{\gamma} \leftarrow \gamma)\, p_\gamma(\gamma)}\right) q(\hat{\gamma} \leftarrow \gamma), \tag{4.3}$$

where $q(\hat{\gamma} \leftarrow \gamma)$ is a *proposal distribution*. Equation 4.3 corresponds to a transition with two steps. First, propose new parameters $\hat{\gamma}$ from $q(\hat{\gamma} \leftarrow \gamma)$. Next, make $\hat{\gamma}$ the new state of the Markov chain using a Bernoulli draw with probability taken from the $\min(\cdot)$ term. Otherwise, keep the current state γ. This transition operator satisfies detailed balance (Hastings, 1970). I will frequently refer to the second argument of the $\min(\cdot)$ in Equation 4.3 as the *acceptance ratio*. The Metropolis–Hastings transition operator results in the algorithm listed in Algorithm 4.1.

4.2 Doubly-Intractable Distributions and Exchange Sampling

In Section 1.3, I introduced the concept of the doubly-intractable posterior distribution. In these distributions, the likelihood function $p(\boldsymbol{x} \,|\, \gamma)$ (where $\boldsymbol{x}$ is an observed datum) is only known to within a constant, i.e. $p(\boldsymbol{x} \,|\, \gamma) = \mathcal{Z}^{-1} h(\gamma\,;\, \boldsymbol{x})$ for unknown $\mathcal{Z}$. When performing Bayesian inference, the posterior distribution has the form

$$p(\gamma \,|\, \boldsymbol{x}) = \frac{p_0(\gamma)\; \mathcal{Z}(\gamma)^{-1} h(\gamma\,;\, \boldsymbol{x})}{\int_\Gamma \mathrm{d}\gamma'\; p_0(\gamma')\; \mathcal{Z}(\gamma')^{-1} h(\gamma'\,;\, \boldsymbol{x})} \tag{4.4}$$

Algorithm 4.1 Generate R samples from $p_\gamma(\gamma)$ via Metropolis–Hastings

Inputs:
- Number of samples to draw R
- Posterior distribution $p_\gamma(\gamma)$
- Proposal distribution $q(\hat{\gamma} \leftarrow \gamma)$
- Initial state γ_0

Outputs:
- R correlated samples from $p_\gamma(\gamma)$, $\{\gamma^{(r)}\}_{r=1}^{R}$

1: $\gamma \leftarrow \gamma_0$ ▷ Initialise the Markov chain.
2: **for** $r \leftarrow 1 \dots R$ **do** ▷ Simulate R Markov transitions.
3: $\hat{\gamma} \sim q(\hat{\gamma} \leftarrow \gamma)$ ▷ Make a proposal.
4: $a_{\mathsf{mh}} \leftarrow \dfrac{q(\gamma \leftarrow \hat{\gamma})\, p_\gamma(\hat{\gamma})}{q(\hat{\gamma} \leftarrow \gamma)\, p_\gamma(\gamma)}$ ▷ Calculate acceptance ratio.
5: $u \sim \mathcal{U}(0, 1)$ ▷ Draw uniformly on $(0, 1)$.
6: **if** $u < a_{\mathsf{mh}}$ **then** ▷ Acceptance rule.
7: $\gamma \leftarrow \hat{\gamma}$ ▷ Make the proposal the current state.
8: **end if**
9: $\gamma^{(r)} \leftarrow \gamma$ ▷ Store the current state.
10: **end for**
11: **return** $\{\gamma^{(r)}\}_{r=1}^{R}$

where I write $\mathcal{Z}(\gamma)$ to make it clear that the likelihood normalisation constant (partition function) depends on the parameters γ. I use $p_0(\gamma)$ to denote the prior distribution on γ. If the parameters are actually a function, as is the case in most of the models I discuss in this thesis, then I use $\mathcal{Z}[\gamma]$ to indicate that it is a "partition functional." Some examples of models with doubly-intractable posterior distributions follow.

The Potts model The Potts model is one of the most common types of doubly-intractable models, and it motivated some of the original work in doubly-intractable inference. The Potts model is a pairwise interaction model (or *spin glass*) with S possible spin states. When neighbouring nodes have different spins, the energy increases and the probability of the overall state decreases. When there are only two types of spins, it is often called the *Ising model* (Ising, 1925)[1], the *Boltzmann machine* (Hopfield, 1982; Ackley et al., 1985; Smolensky, 1986), the *autologistic model* (Møller et al., 2004) or the *binary Markov random field* (see, e.g. Murray (2007, Chapter 1)). For D nodes, the Potts model likelihood terms are

$$h(\boldsymbol{W}\,;\, \boldsymbol{x} \in \{1, 2, \dots, S\}^D) = \exp\left\{-\frac{1}{2}\sum_{d=1}^{D}\sum_{d'=1}^{D} W_{d,d'}(\delta(x_d, x_{d'}) - 1)\right\} \tag{4.5}$$

$$\mathcal{Z}(\boldsymbol{W}) = \sum_{\boldsymbol{x}'} \exp\left\{-\frac{1}{2}\sum_{d=1}^{D}\sum_{d'=1}^{D} W_{d,d'}(\delta(x'_d, x'_{d'}) - 1)\right\}. \tag{4.6}$$

The function $\delta(\cdot, \cdot)$ is the Kronecker delta function, which is equal to one if its arguments are equal, and zero otherwise. Computing the normalisation constant of this model requires a sum over a state space that grows exponentially with the number of nodes. Many more general Markov random fields also have doubly-intractable poste-

[1] See Niss (2005) for a discussion of the history of the Ising model.

rior distributions.

The logistic Gaussian process See Section 1.3.

$$h(\boldsymbol{g}\,;\,\boldsymbol{x}) = \exp\{g(\boldsymbol{x})\} \tag{4.7}$$

$$\mathcal{Z}[\boldsymbol{g}] = \int_{\mathcal{X}} \mathrm{d}\boldsymbol{x}'\ \exp\{g(\boldsymbol{x}')\} \tag{4.8}$$

The Gaussian process density sampler See Chapter 2.

$$h(\boldsymbol{g}\,;\,\boldsymbol{x}) = \Phi(g(\boldsymbol{x}))\ \pi(\boldsymbol{x}) \tag{4.9}$$

$$\mathcal{Z}[\boldsymbol{g}] = \int_{\mathcal{X}} \mathrm{d}\boldsymbol{x}'\ \Phi(g(\boldsymbol{x}'))\ \pi(\boldsymbol{x}') \tag{4.10}$$

The log Gaussian Cox process See Section 1.4.

$$h(\boldsymbol{g}\,;\,\boldsymbol{x}) = \exp\{g(\boldsymbol{x})\} \tag{4.11}$$

$$\mathcal{Z}[\boldsymbol{g}] = \exp\left\{\int_{\mathcal{V}} \mathrm{d}\boldsymbol{x}'\ \exp\{g(\boldsymbol{x}')\}\right\} \tag{4.12}$$

The sigmoidal Gaussian Cox process See Chapter 3.

$$h(\boldsymbol{g}\,;\,\boldsymbol{x}) = \Phi(g(\boldsymbol{x}))\ \bar{\lambda}(\boldsymbol{x}) \tag{4.13}$$

$$\mathcal{Z}[\boldsymbol{g}] = \exp\left\{\int_{\mathcal{V}} \mathrm{d}\boldsymbol{x}'\ \Phi(g(\boldsymbol{x}'))\ \bar{\lambda}(\boldsymbol{x}')\right\} \tag{4.14}$$

While Metropolis–Hastings is often a workhorse for Bayesian inference with complex posterior distributions, it cannot be used directly for models with doubly-intractable posteriors. If Equation 4.4 is plugged into Equation 4.3 for $p_\gamma(\gamma)$, the acceptance ratio is

$$a_{\mathsf{mh}} = \frac{q(\gamma \leftarrow \hat{\gamma})\ p_0(\hat{\gamma})\ h(\hat{\gamma}\,;\,\boldsymbol{x})}{q(\hat{\gamma} \leftarrow \gamma)\ p_0(\gamma)\ h(\gamma\,;\,\boldsymbol{x})} \times \frac{\mathcal{Z}(\gamma)}{\mathcal{Z}(\hat{\gamma})}. \tag{4.15}$$

This acceptance ratio cannot be calculated without knowing the ratio of intractable constants that appears on the right. These constants do not cancel because they depend on the parameters. Further, the ratio in Equation 4.15 cannot be approximated in an arbitrary way (using, for example, numerical integration to estimate the constants) and still yield a Markov chain whose equilibrium distribution is the posterior of interest.

In 2004, however, Møller et al. (2004)[2] made the landmark contribution of the first method for constructing a Metropolis–Hastings Markov chain in which it is not nec-

[2] A shorter journal article was formally published in Møller et al. (2006).

essary to evaluate this intractable ratio. There is a large caveat, however: it must be possible to simulate exact data from the model for a given setting of the parameters. These data have typically been generated via coupling from the past (CFTP) (Propp and Wilson, 1996). In Møller et al. (2004), CFTP is used on the Strauss process (Strauss, 1975; Kelly and Ripley, 1976) to generate exact samples, and in Møller et al. (2006), coupling from the past is applied to the Ising model.

Following Murray (2007), I will refer to the method of Møller et al. (2004) as the *single auxiliary variable method* (SAVM). The distribution of interest is $p(\gamma \,|\, \boldsymbol{x}) \propto p(\gamma, \boldsymbol{x}) = p_0(\gamma)\, \mathcal{Z}(\gamma)^{-1}\, h(\gamma\,;\, \boldsymbol{x})$. The SAVM augments the joint distribution $p(\gamma, \boldsymbol{x})$ with an additional variable $\boldsymbol{w}$ that is on the same space as $\boldsymbol{x}$. I will refer to this new variable $\boldsymbol{w}$ as *fantasy data*. This augmented joint arises from multiplying $p(\gamma, \boldsymbol{x})$ by an arbitrary conditional distribution $f(\boldsymbol{w} \,|\, \gamma, \boldsymbol{x})$, so that the overall distribution is $p(\boldsymbol{x}, \boldsymbol{w}, \gamma) = f(\boldsymbol{w} \,|\, \gamma, \boldsymbol{x})\, p_0(\gamma)\, \mathcal{Z}(\gamma)^{-1}\, h(\gamma; \boldsymbol{x})$. Marginalising over $\boldsymbol{w}$ still gives the distribution $p(\gamma, \boldsymbol{x})$, so it is possible to simulate both γ and $\boldsymbol{w}$ and reserve the samples of γ for posterior estimation.

To sample a doubly-intractable posterior via the SAVM, one performs Metropolis–Hastings on the joint distribution over γ and $\boldsymbol{w}$, for fixed $\boldsymbol{x}$. Proposals are made in two stages. First, a new proposal for γ is drawn from the distribution $q(\hat{\gamma} \leftarrow \gamma)$. Second, a new $\boldsymbol{w}$ is proposed from the same distribution as the likelihood model, but using the newly-proposed parameter $\hat{\gamma}$, i.e. $q(\hat{\boldsymbol{w}} \leftarrow \boldsymbol{w}) = p(\hat{\boldsymbol{w}} \,|\, \hat{\gamma}) = \mathcal{Z}(\hat{\gamma})^{-1}\, h(\hat{\gamma}\,;\, \hat{\boldsymbol{w}})$. This proposal $\hat{\boldsymbol{w}}$ is generated as an exact sample from the model with parameter $\hat{\gamma}$. The overall proposal distribution is

$$q(\hat{\gamma}, \hat{\boldsymbol{w}} \leftarrow \gamma, \boldsymbol{w}) = q(\hat{\gamma} \leftarrow \gamma)\; \mathcal{Z}(\hat{\gamma})^{-1}\, h(\hat{\gamma}\,;\, \hat{\boldsymbol{w}}). \tag{4.16}$$

The Metropolis–Hastings acceptance ratio of this proposal is

$$\begin{aligned} a_{\mathsf{savm}} &= \frac{q(\gamma \leftarrow \hat{\gamma})\; f(\hat{\boldsymbol{w}} \,|\, \hat{\gamma}, \boldsymbol{x})\; p_0(\hat{\gamma})\; \cancel{\mathcal{Z}(\hat{\gamma})^{-1}}\, h(\hat{\gamma}\,;\, \boldsymbol{x})\; \cancel{\mathcal{Z}(\gamma)^{-1}}\, h(\gamma\,;\, \boldsymbol{w})}{q(\hat{\gamma} \leftarrow \gamma)\; f(\boldsymbol{w} \,|\, \gamma, \boldsymbol{x})\; p_0(\gamma)\; \cancel{\mathcal{Z}(\gamma)^{-1}}\, h(\gamma\,;\, \boldsymbol{x})\; \cancel{\mathcal{Z}(\hat{\gamma})^{-1}}\, h(\hat{\gamma}\,;\, \hat{\boldsymbol{w}})} \\ &= \frac{q(\gamma \leftarrow \hat{\gamma})\; f(\hat{\boldsymbol{w}} \,|\, \hat{\gamma}, \boldsymbol{x})\; p_0(\hat{\gamma})\; h(\hat{\gamma}\,;\, \boldsymbol{x})\; h(\gamma\,;\, \boldsymbol{w})}{q(\hat{\gamma} \leftarrow \gamma)\; f(\boldsymbol{w} \,|\, \gamma, \boldsymbol{x})\; p_0(\gamma)\; h(\gamma\,;\, \boldsymbol{x})\; h(\hat{\gamma}\,;\, \hat{\boldsymbol{w}})}, \end{aligned} \tag{4.17}$$

in which the normalisation constants cancel out.

Inspired by the SAVM, Murray et al. (2006) developed a procedure called *exchange sampling* (ES), that removes the need for the conditional distribution $f(\boldsymbol{w} \,|\, \gamma, \boldsymbol{x})$. Again, the distribution of interest is $p(\gamma \,|\, \boldsymbol{x}) \propto p(\gamma, \boldsymbol{x}) = p_0(\gamma)\, \mathcal{Z}(\gamma)^{-1}\, h(\gamma\,;\, \boldsymbol{x})$. To make an exchange sampling move, first propose new parameters $\hat{\gamma}$ from a proposal distribution $q(\hat{\gamma} \leftarrow \gamma)$. Next, draw new fantasy data from the likelihood model using the new parameter $\hat{\gamma}$. These data must be exactly from the distribution $p(\boldsymbol{w} \,|\, \hat{\gamma}) = \mathcal{Z}(\hat{\gamma})^{-1}\, h(\hat{\gamma}\,;\, \boldsymbol{w})$. Finally, propose the eponymous exchange of γ and $\hat{\gamma}$.

Algorithm 4.2 Generate R samples from $p(\gamma \mid \{\boldsymbol{x}_n\}_{n=1}^N)$ via exchange sampling

Inputs:
- Number of samples to draw R
- Observed data $\{\boldsymbol{x}_n\}_{n=1}^N$
- Likelihood model $p(\boldsymbol{x} \mid \gamma) = \mathcal{Z}(\gamma)^{-1} h(\gamma\,;\,\boldsymbol{x})$
- Prior $p_0(\gamma)$
- Proposal distribution $q(\hat{\gamma} \leftarrow \gamma)$
- Initial state γ_0

Outputs:
- R correlated samples from $p(\gamma \mid \{\boldsymbol{x}_n\}_{n=1}^N)$, $\{\gamma^{(r)}\}_{r=1}^R$

$\gamma \leftarrow \gamma_0$ ▷ Initialise the Markov chain.
for $r \leftarrow 1 \ldots R$ **do** ▷ Simulate R exchange sampling steps.
  $\hat{\gamma} \sim q(\hat{\gamma} \leftarrow \gamma)$ ▷ Make a proposal.
  for $n \leftarrow 1$ to N **do** ▷ Generate N fantasies.
    $\boldsymbol{w}_n \sim p(\boldsymbol{x} \mid \hat{\gamma})$ ▷ Generate an exact fantasy from $\hat{\gamma}$.
  end for
  $a_{\mathsf{es}} \leftarrow \dfrac{q(\gamma \leftarrow \hat{\gamma})\; p_0(\hat{\gamma})}{q(\hat{\gamma} \leftarrow \gamma)\; p_0(\gamma)} \displaystyle\prod_{n=1}^{N} \frac{h(\hat{\gamma}\,;\,\boldsymbol{x}_n)\; h(\gamma\,;\,\boldsymbol{w}_n)}{h(\gamma\,;\,\boldsymbol{x}_n)\; h(\hat{\gamma}\,;\,\boldsymbol{w}_n)}$ ▷ Calculate acceptance ratio.
  $u \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variable on $(0, 1)$.
  if $u < a_{\mathsf{es}}$ **then** ▷ Acceptance rule.
    $\gamma \leftarrow \hat{\gamma}$ ▷ Make the proposal the current state.
  end if
  $\gamma^{(r)} \leftarrow \gamma$ ▷ Store the current state.
end for
return $\{\gamma^{(r)}\}_{r=1}^R$

The joint distribution over the current state and first two proposal steps is

$$p(\gamma, \boldsymbol{x}, \hat{\gamma}, \boldsymbol{w}) = p_0(\gamma)\; p(\boldsymbol{x} \mid \gamma)\; q(\hat{\gamma} \leftarrow \gamma)\; p(\boldsymbol{w} \mid \hat{\gamma}). \tag{4.18}$$

The Metropolis–Hastings acceptance ratio of the exchange is the ratio of Equation 4.18 with γ and $\hat{\gamma}$ swapped, to Equation 4.18 as written, yielding:

$$\begin{aligned} a_{\mathsf{es}} &= \frac{p_0(\hat{\gamma})\; \cancel{\mathcal{Z}(\hat{\gamma})^{-1}}\; h(\hat{\gamma}\,;\,\boldsymbol{x})\; q(\gamma \leftarrow \hat{\gamma})\; \cancel{\mathcal{Z}(\gamma)^{-1}}\; h(\gamma\,;\,\boldsymbol{w})}{p_0(\gamma)\; \cancel{\mathcal{Z}(\gamma)^{-1}}\; h(\gamma\,;\,\boldsymbol{x})\; q(\hat{\gamma} \leftarrow \gamma)\; \cancel{\mathcal{Z}(\hat{\gamma})^{-1}}\; h(\hat{\gamma}\,;\,\boldsymbol{w})} \\ &= \frac{p_0(\hat{\gamma})\; h(\hat{\gamma}\,;\,\boldsymbol{x})\; q(\gamma \leftarrow \hat{\gamma})\; h(\gamma\,;\,\boldsymbol{w})}{p_0(\gamma)\; h(\gamma\,;\,\boldsymbol{x})\; q(\hat{\gamma} \leftarrow \gamma)\; h(\hat{\gamma}\,;\,\boldsymbol{w})}. \end{aligned} \tag{4.19}$$

Even with this simpler construction, the normalisation constants have canceled out of the acceptance ratio. To generalise this procedure to multiple i.i.d. data, generate as many exact fantasy data as there are true data. The exchange sampling algorithm for N independent data is provided in pseudocode in Algorithm 4.2.

Exchange sampling and the single auxiliary variable method are the motivation for modeling with the Gaussian process density sampler and the sigmoidal Gaussian Cox process, rather than the logistic Gaussian process and the log Gaussian Cox process. While the models I have presented are slightly more complicated than the LGP and LGCP, I have coupled my proposed priors with generative procedures that enable tractable inference via the methods presented in this section.

4.3 Exchange Sampling for GPDS Inference

Inference is the task of "inverting" a model to explain the observed data. The Gaussian process density sampler is a model for probability density functions, parameterised by a latent function $g(\boldsymbol{x})$, which has a Gaussian process prior. Given N data $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^N$, that are modeled as i.i.d. from an unknown probability density, the objective is to sample from the posterior distribution on $g(\boldsymbol{x})$, which, as before, I will treat as a vector $\boldsymbol{g}$. As the GPDS enables generation of exact data, it is possible to simulate a Markov chain on the posterior distribution $p(\boldsymbol{g} \mid \mathcal{D})$ using the exchange sampling method described in Section 4.2.

4.3.1 Independence-Chain Exchange Sampling

For clarity, I will begin by describing exchange sampling in the GPDS using the simplest possible proposals for $\boldsymbol{g}$: independent draws from the prior, i.e. $q(\hat{\boldsymbol{g}} \leftarrow \boldsymbol{g}) = p(\hat{\boldsymbol{g}} \mid \boldsymbol{\theta}) = \mathcal{GP}(\hat{\boldsymbol{g}} \mid \boldsymbol{\theta})$. This corresponds to an *independence chain Metropolis–Hastings* variant of exchange sampling. This method is inefficient, for reasons that will be explained shortly, but it is a useful starting point to see how exchange sampling can be applied to the Gaussian process density sampler. It is also useful to think of $\boldsymbol{g}$ as something that can be stored wholly in memory, even though it is an infinite-dimensional object. From an initial state $\boldsymbol{g}$, an exchange sampling step would first require drawing a new function $\hat{\boldsymbol{g}}$ from the proposal (the prior in this case). Following this, it is necessary to generate N fantasy data $\mathcal{W} = \{\boldsymbol{w}_n\}_{n=1}^N$ from the density implied by $\hat{\boldsymbol{g}}$. Section 2.2 provided a way to do both of these steps at once — sampling a function from the prior while simultaneously drawing data from the corresponding density. This is precisely running Algorithm 2.2 until exactly N data are accepted. After fantasisation, the joint distribution over $\boldsymbol{g}$, $\{\boldsymbol{x}_n\}_{n=1}^N$, $\hat{\boldsymbol{g}}$, and $\{\boldsymbol{w}_n\}_{n=1}^N$, given the hyperparameters, is

$$p(\boldsymbol{g}, \mathcal{D}, \hat{\boldsymbol{g}}, \mathcal{W} \mid \boldsymbol{\theta}, \boldsymbol{\psi}_\pi) = \mathcal{GP}(\boldsymbol{g} \mid \boldsymbol{\theta})\ \mathcal{Z}_\pi[\boldsymbol{g}]^{-N} \left[\prod_{n=1}^N \Phi(g(\boldsymbol{x}_n))\ \pi(\boldsymbol{x}_n \mid \boldsymbol{\psi}_\pi)\right] \times \mathcal{GP}(\hat{\boldsymbol{g}} \mid \boldsymbol{\theta})\ \mathcal{Z}_\pi[\hat{\boldsymbol{g}}]^{-N} \prod_{n=1}^N \Phi(\hat{g}(\boldsymbol{w}_n))\ \pi(\boldsymbol{w}_n \mid \boldsymbol{\psi}_\pi). \quad (4.20)$$

Algorithm 4.3 Simulate R steps using independence chain ES on the GPDS.

Inputs:
- Number of MCMC iterations R
- Observed data $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^N$
- Gaussian process covariance function $C(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta})$
- Base density $\pi(\boldsymbol{x} \,|\, \boldsymbol{\psi}_\pi)$

Outputs:
- R conditioning sets of function inputs and outputs $\{\boldsymbol{X}^{(r)}, \boldsymbol{G}^{(r)}\}_{r=1}^R$

$\{g(\boldsymbol{x}_n)\}_{n=1}^N \sim \mathcal{GP}(g \,|\, \mathcal{D}, \boldsymbol{\theta})$ ▷ Initialise the function at the data.
$\boldsymbol{X}^{(1)} \leftarrow \{\boldsymbol{x}_n\}_{n=1}^N$, $\boldsymbol{G}^{(1)} \leftarrow \{g(\boldsymbol{x}_n)\}_{n=1}^N$ ▷ Initialise conditioning sets.
for $r \leftarrow 1 \ldots R$ **do** ▷ Take R exchange sampling steps.
  $\{\hat{g}(\boldsymbol{x}_n)\}_{n=1}^N \sim \mathcal{GP}(g \,|\, \mathcal{D}, \boldsymbol{\theta})$ ▷ Draw a new function at the data.
  $\hat{\boldsymbol{X}} \leftarrow \{\boldsymbol{x}_n\}_{n=1}^N$, $\hat{\boldsymbol{G}} \leftarrow \{\hat{g}(\boldsymbol{x}_n)\}_{n=1}^N$ ▷ Initialise proposal conditioning sets.
  $\mathcal{W} \leftarrow \emptyset$ ▷ Initialise empty fantasy data set.
  repeat ▷ Run the rejection sampling loop.
    $\tilde{\boldsymbol{w}} \sim \pi(\boldsymbol{x} \,|\, \boldsymbol{\psi}_\pi)$ ▷ Draw a proposal from the base density.
    $\hat{g}(\tilde{\boldsymbol{w}}) \sim \mathcal{GP}(\hat{g} \,|\, \tilde{\boldsymbol{w}}, \hat{\boldsymbol{X}}, \hat{\boldsymbol{G}}, \boldsymbol{\theta})$ ▷ Draw the function value at the proposal.
    $u_{\mathsf{fant}} \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on $(0, 1)$.
    if $u_{\mathsf{fant}} < \Phi(\hat{g}(\tilde{\boldsymbol{w}}))$ **then** ▷ Rejection sampling acceptance rule.
      $\mathcal{W} \leftarrow \mathcal{W} \cup \tilde{\boldsymbol{w}}$ ▷ Keep the fantasy.
    end if
    $\hat{\boldsymbol{X}} \leftarrow \hat{\boldsymbol{X}} \cup \tilde{\boldsymbol{w}}$, $\hat{\boldsymbol{G}} \leftarrow \hat{\boldsymbol{G}} \cup \hat{g}(\tilde{\boldsymbol{w}})$ ▷ Add proposals to the conditioning sets.
  until $|\mathcal{W}| = N$ ▷ Loop until N fantasies are accepted.
  $\{g(\boldsymbol{w}_n)\}_{n=1}^N \sim \mathcal{GP}(g \,|\, \mathcal{W}, \boldsymbol{X}^{(r)}, \boldsymbol{G}^{(r)})$ ▷ Sample the current function at the fantasies.
  $a_{\mathsf{gpds-ices}} \leftarrow \prod_{n=1}^N \frac{\Phi(\hat{g}(\boldsymbol{x}_n))\ \Phi(g(\boldsymbol{w}_n))}{\Phi(g(\boldsymbol{x}_n))\ \Phi(\hat{g}(\boldsymbol{w}_n))}$ ▷ Calculate the acceptance ratio.
  $u_{\mathsf{mh}} \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on $(0, 1)$.
  if $u_{\mathsf{mh}} < a_{\mathsf{gpds-ices}}$ **then** ▷ Apply the Metropolis–Hastings acceptance rule.
    $\boldsymbol{X}^{(r+1)} \leftarrow \hat{\boldsymbol{X}}$, $\boldsymbol{G}^{(r+1)} \leftarrow \hat{\boldsymbol{G}}$ ▷ Keep the new function data.
  else
    $\boldsymbol{X}^{(r+1)} \leftarrow \boldsymbol{X}^{(r)} \cup \{\boldsymbol{w}_n\}_{n=1}^N$ ▷ Add the fantasy evaluations to the current state.
    $\boldsymbol{G}^{(r+1)} \leftarrow \boldsymbol{G}^{(r)} \cup \{g(\boldsymbol{w}_n)\}_{n=1}^N$
  end if
end for
return $\{\boldsymbol{X}^{(r)}, \boldsymbol{G}^{(r)}\}_{r=1}^R$

Now the proposal is made to exchange $\boldsymbol{g}$ and $\hat{\boldsymbol{g}}$, which has the acceptance ratio

$$
\begin{aligned}
a_{\mathsf{gpds-ices}} &= \frac{\cancel{\mathcal{GP}(\hat{\boldsymbol{g}} \,|\, \boldsymbol{\theta})}\ \cancel{\mathcal{Z}_\pi[\hat{\boldsymbol{g}}]^{-N}}\ \cancel{\mathcal{GP}(\boldsymbol{g} \,|\, \boldsymbol{\theta})}\ \cancel{\mathcal{Z}_\pi[\boldsymbol{g}]^{-N}}}{\cancel{\mathcal{GP}(\boldsymbol{g} \,|\, \boldsymbol{\theta})}\ \cancel{\mathcal{Z}_\pi[\boldsymbol{g}]^{-N}}\ \cancel{\mathcal{GP}(\hat{\boldsymbol{g}} \,|\, \boldsymbol{\theta})}\ \cancel{\mathcal{Z}_\pi[\hat{\boldsymbol{g}}]^{-N}}} \\
&\qquad \times \prod_{n=1}^N \frac{\Phi(\hat{g}(\boldsymbol{x}_n))\ \cancel{\pi(\boldsymbol{x}_n \,|\, \boldsymbol{\psi}_\pi)}\ \Phi(g(\boldsymbol{w}_n))\ \cancel{\pi(\boldsymbol{w}_n \,|\, \boldsymbol{\psi}_\pi)}}{\Phi(g(\boldsymbol{x}_n))\ \cancel{\pi(\boldsymbol{x}_n \,|\, \boldsymbol{\psi}_\pi)}\ \Phi(\hat{g}(\boldsymbol{w}_n))\ \cancel{\pi(\boldsymbol{w}_n \,|\, \boldsymbol{\psi}_\pi)}} \\
&= \prod_{n=1}^N \frac{\Phi(\hat{g}(\boldsymbol{x}_n))\ \Phi(g(\boldsymbol{w}_n))}{\Phi(g(\boldsymbol{x}_n))\ \Phi(\hat{g}(\boldsymbol{w}_n))}.
\end{aligned} \tag{4.21}
$$

Not only have the normalisation constants canceled out of this acceptance ratio, but the remaining terms depend only on evaluating the function at a finite number of locations.

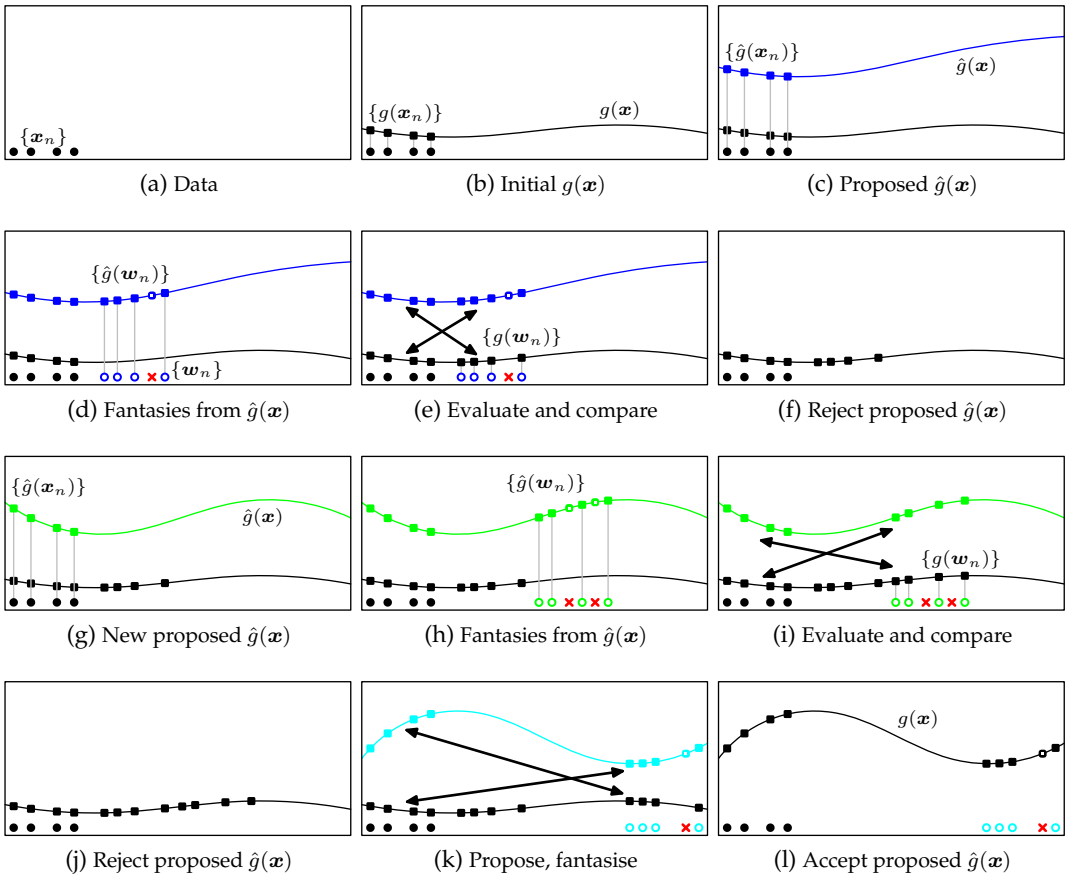
Of course, the acceptance ratio of Equation 4.21 is not the entire story. To ensure that the algorithm has the correct invariant distribution, it is also necessary to perform

some bookkeeping. Specifically, it is necessary to keep consistent track of the current and proposed functions, $\boldsymbol{g}$ and $\hat{\boldsymbol{g}}$. It is not required to know them everywhere, but anything discovered about them (i.e. sampled from the Gaussian process prior) must be retained for as long as that function is relevant to the current Markov state. Such information can accrue in two different ways. First, when fantasy data are drawn via Algorithm 2.2, there may be rejections. When fantasy proposals are rejected, the function $\hat{\boldsymbol{g}}$ is still evaluated and the input and output values are added to the conditioning set. All future evaluations of this function must be conditioned on these values. Second, it is necessary to sample the function values $\{g(\boldsymbol{w}_n)\}_{n=1}^N$, i.e. the *current* function evaluated at the *fantasy* data, to calculate the acceptance ratio in Equation 4.21. If the exchange sampling move is then rejected, these values must nevertheless be added to the conditioning set of the function that is the current Markov state. The practical effect of this bookkeeping is that the actual stored state of the Markov chain — the conditioning set of the current function — grows by N function evaluations when exchange sampling moves are rejected. Fortunately, when an exchange sampling move is accepted, the conditioning set of the old function can be discarded, due to the Markov property of the chain. The conditioning set after an acceptance possesses only 1) the true data, 2) the fantasy data, and 3) the rejections made along the way to generating fantasy data.

The key to the tractability and practicality of exchange sampling with the Gaussian process density sampler is that information about any function $g(\boldsymbol{x})$ in the Markov state is queried only when necessary. This idea of Markov chain Monte Carlo inference via sampling finite parts of an infinite-dimensional random object is known as *retrospective sampling* (Papaspiliopoulos and Roberts, 2008). The concept is to sample and store only information that is immediately relevant to the inference computation. Some probabilistic models make it easy to ask new questions about a random object, conditioning on what is already known. The Gaussian process allows this conditional sampling for $\boldsymbol{g}$ via Equation 1.5 and Equation 1.6. The effect is that inference can be performed in the Gaussian process density sampler using finite computation, even though the latent function $g(\boldsymbol{x})$ has an infinite number of dimensions. Retrospective sampling was introduced by Papaspiliopoulos and Roberts (2008), who examined the Dirichlet process. Beskos et al. (2006b) and Beskos et al. (2006a) applied retrospective sampling to inference in diffusion processes. Algorithm 4.3 shows the independence chain exchange sampling algorithm as pseudocode and Figure 4.1 illustrates the idea graphically.

4.3.2 Improving the Acceptance Rate with Conservative Proposals

In general, the independence chain exchange sampling algorithm is not expected to have a good acceptance rate. If there have been many observed data, then typical samples from the Gaussian process prior are unlikely to result in densities that ex-



(a) Data (b) Initial $g(\boldsymbol{x})$ (c) Proposed $\hat{g}(\boldsymbol{x})$

(d) Fantasies from $\hat{g}(\boldsymbol{x})$ (e) Evaluate and compare (f) Reject proposed $\hat{g}(\boldsymbol{x})$

(g) New proposed $\hat{g}(\boldsymbol{x})$ (h) Fantasies from $\hat{g}(\boldsymbol{x})$ (i) Evaluate and compare

(j) Reject proposed $\hat{g}(\boldsymbol{x})$ (k) Propose, fantasise (l) Accept proposed $\hat{g}(\boldsymbol{x})$

Figure 4.1: A cartoon of three exchange sampling transitions. In the first two transitions, the proposals are rejected to demonstrate the expanding Markov state due to retrospective sampling of $g(\boldsymbol{x})$. The third proposal is accepted and the previously-accumulated state is discarded. (a) The observed data, illustrated as • . (b) The initial $g(\boldsymbol{x})$ evaluated at the data, shown as ▪ . (c) The proposed $\hat{g}(\boldsymbol{x})$ evaluated at the data, shown as • . (d) Fantasies are drawn from $\hat{g}(\boldsymbol{x})$, illustrated as ○ . There is one rejected proposal, shown as × , with the corresponding function value illustrated as ○ . (e) $g(\boldsymbol{x})$ is evaluated at the fantasies and the two explanations are compared using Equation 4.21. (f) The proposal $\hat{g}(\boldsymbol{x})$ is rejected. The Markov state expands to include the fantasies. (g) A new $\hat{g}(\boldsymbol{x})$, shown in green, is evaluated at the data. (h) Fantasies are drawn from $\hat{g}(\boldsymbol{x})$ (two rejected fantasy proposals). (i) $g(\boldsymbol{x})$ is evaluated at the fantasies and the explanations are compared. (j) The proposal was rejected. The Markov state expands to twelve function evaluations. (k) Skipping the intermediate steps, propose, fantasise, evaluate and compare, using the function shown in cyan. (l) The proposal is accepted and made the new $g(\boldsymbol{x})$. All of the information about the old function is thrown away, but the new $g(\boldsymbol{x})$ must keep information in its conditioning set about the fantasies it generated.

plain the data well. Tuning to get efficient mixing is important in most Markov chain Monte Carlo algorithms, but in exchange sampling for the Gaussian process density sampler it is even more important: each time there is a rejection, the Markov state expands. This expanding Markov state causes subsequent calculations to have quadratic space complexity due to the need to calculate the covariance matrix, and cubic time complexity, due to the need to decompose or invert the covariance matrix.

One way to increase the acceptance rate of a Metropolis–Hastings sampler is to make more conservative proposals. If the proposals are only small perturbations of the cur-

rent state, the posterior probability is unlikely to change dramatically, and many proposals will be accepted. In the case of exchange sampling for the Gaussian process density sampler, being more conservative means altering the proposal distribution over functions to assign greater probability to functions that are "similar" to the current function.

Introducing Control Points

To make conservative proposals, I narrow the distribution on functions by introducing a set of B *control points* in $\mathcal{X}$, denoted $\mathcal{C} = \{\boldsymbol{x}_b\}_{b=1}^B$, $\boldsymbol{x}_b \in \mathcal{X}$. These control points have associated function values, which I denote $\mathcal{G} = \{g(\boldsymbol{x}_b)\}_{b=1}^B$. I will assume that $\mathcal{C}$ is a superset of the observed data, i.e. $\mathcal{D} \subseteq \mathcal{C}$. The function values at the control points are explicitly included in the Markov state and all retrospective function draws condition on these points. New discoveries about the function continue to accumulate in the conditioning sets, as before. The difference now is that the conditioning sets are initialised with the control points and small, perturbative proposals can be made on the function values at those initial points.

To make this construction explicit, Equation 4.20 is extended to

$$\begin{aligned} p(\mathcal{G}, \boldsymbol{g}_{\setminus\mathcal{C}}, \mathcal{D}, \hat{\mathcal{G}}, \hat{\boldsymbol{g}}_{\setminus\mathcal{C}}, \mathcal{W} \,|\, \mathcal{C}, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi) = \\ \mathcal{GP}(\mathcal{G} \,|\, \mathcal{C}, \boldsymbol{\theta})\ \mathcal{GP}(\boldsymbol{g}_{\setminus\mathcal{C}} \,|\, \mathcal{G}, \boldsymbol{\theta})\ \mathcal{Z}_\pi[\boldsymbol{g}]^{-N} \left[\prod_{n=1}^N \Phi(g(\boldsymbol{x}_n))\ \pi(\boldsymbol{x}_n \,|\, \boldsymbol{\psi}_\pi)\right] \\ \times\ q(\hat{\mathcal{G}} \leftarrow \mathcal{G})\ \mathcal{GP}(\hat{\boldsymbol{g}}_{\setminus\mathcal{C}} \,|\, \hat{\mathcal{G}}, \boldsymbol{\theta})\ \mathcal{Z}_\pi[\hat{\boldsymbol{g}}]^{-N} \prod_{n=1}^N \Phi(\hat{g}(\boldsymbol{w}_n))\ \pi(\boldsymbol{w}_n \,|\, \boldsymbol{\psi}_\pi), \end{aligned} \tag{4.22}$$

where $\hat{\mathcal{G}}$ indicates the proposal of the function values at the control points, i.e. $\hat{\mathcal{G}} = \{\hat{g}(\boldsymbol{x}_b)\}_{b=1}^B$, and $\boldsymbol{g}_{\setminus\mathcal{C}}$ denotes the function values, excluding those at $\mathcal{C}$. The proposal density $q(\hat{\mathcal{G}} \leftarrow \mathcal{G})$ can be chosen to take smaller steps than the prior draws of the previous section. With the joint distribution in Equation 4.22, and using the conditional retrospective exchange sampling as before, the acceptance ratio of exchanging the pair $(\mathcal{G}, \boldsymbol{g}_{\setminus\mathcal{C}})$ for $(\hat{\mathcal{G}}, \hat{\boldsymbol{g}}_{\setminus\mathcal{C}})$ is

$$a_{\mathsf{gpds-cpes}} = \frac{q(\mathcal{G} \leftarrow \hat{\mathcal{G}})\ \mathcal{GP}(\hat{\mathcal{G}} \,|\, \mathcal{C}, \boldsymbol{\theta})}{q(\hat{\mathcal{G}} \leftarrow \mathcal{G})\ \mathcal{GP}(\mathcal{G} \,|\, \mathcal{C}, \boldsymbol{\theta})} \prod_{n=1}^N \frac{\Phi(\hat{g}(\boldsymbol{x}_n))\ \Phi(g(\boldsymbol{w}_n))}{\Phi(g(\boldsymbol{x}_n))\ \Phi(\hat{g}(\boldsymbol{w}_n))}. \tag{4.23}$$

Superficially, this might seem similar to the knot-based imputation method of Tokdar (2007). However, whereas Tokdar (2007) uses knots as a finite-dimensional approximation, I am using the control points simply to constrain the proposal distribution. The control points only initialise the retrospective sampling procedure. As I enforce a Gaussian process prior on the function values of the control points, the inference procedure still yields the correct posterior distribution on the uncompromised fully-

nonparametric Gaussian process density sampler model. The number and locations of the control points are free parameters. I have not significantly explored the topic of optimal placement of control points. Intuitively, however, a set of locations with low discrepancy under the base density seems appropriate.

Underrelaxed Proposals

Gaussian process priors typically create strong correlations between the function values, in this case restricting the probability mass of $\mathcal{G}$ to a relatively narrow region. A naïve perturbative proposal, such as independent Gaussian proposals on each component, is likely to step out of the high-density region and be rejected. To combat this, it is desirable to make proposals that are invariant under the prior distribution and so cannot be vetoed by a highly-structured prior such as the Gaussian process. Invariant proposals of this type were introduced by Adler (1981) under the name *overrelaxation*, for Gibbs sampling of densities with univariate Gaussian marginals. The goal of Adler (1981) was to make the Gibbs draws more aggressive to enhance mixing and reduce random walk behaviour. Neal (1998) generalised this idea and also discussed the *underrelaxed* variant, which I describe in slightly broader terms here. Additionally, Rasmussen (1996) used a very similar idea to implement persistent momenta in Hamiltonian Monte Carlo simulations.[3]

I return to the abstract parameter space γ, examining a posterior distribution $p(\gamma \,|\, \mathcal{D}) \propto p_0(\gamma)\, p(\mathcal{D} \,|\, \gamma)$, where $p_0(\gamma)$ is the prior and $p(\mathcal{D} \,|\, \gamma)$ is the likelihood function. I define underrelaxed proposals to be direct transitions from γ to $\hat{\gamma}$ that satisfy detailed balance for the prior:

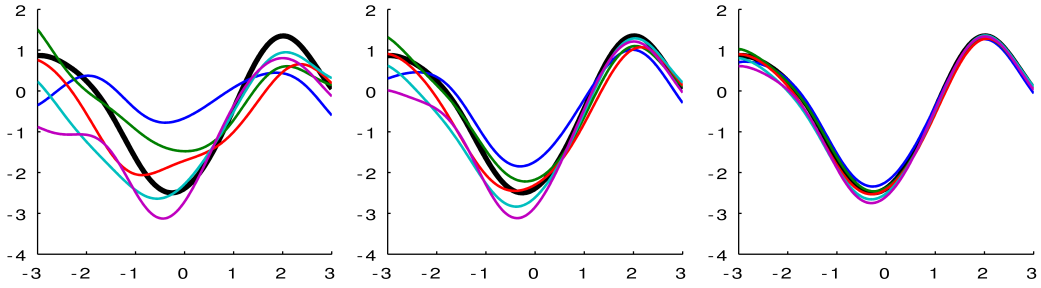
$$p_0(\gamma)\; q(\hat{\gamma} \leftarrow \gamma) = p_0(\hat{\gamma})\; q(\gamma \leftarrow \hat{\gamma}). \tag{4.24}$$

Lemma 4.1. *If the proposal $q(\hat{\gamma} \leftarrow \gamma)$ is a transition operator that satisfies detailed balance for the prior distribution $p_0(\gamma)$, then using Metropolis–Hastings acceptance ratio $a = p(\mathcal{D} \,|\, \hat{\gamma}) / p(\mathcal{D} \,|\, \gamma)$ yields an operator that satisfies detailed balance for the distribution $p_0(\gamma)\, p(\mathcal{D} \,|\, \gamma)$.*

Proof.

$$\begin{aligned}
T(\hat{\gamma} \leftarrow \gamma)\; p_0(\gamma)\, p(\mathcal{D} \,|\, \gamma) &= p_0(\gamma)\, p(\mathcal{D} \,|\, \gamma)\; q(\hat{\gamma} \leftarrow \gamma) \min\left(1, \frac{p(\mathcal{D} \,|\, \hat{\gamma})}{p(\mathcal{D} \,|\, \gamma)}\right) \\
&= p_0(\gamma)\; q(\hat{\gamma} \leftarrow \gamma) \min\left(p(\mathcal{D} \,|\, \gamma), p(\mathcal{D} \,|\, \hat{\gamma})\right) \\
&= p_0(\hat{\gamma})\; q(\gamma \leftarrow \hat{\gamma}) \min\left(p(\mathcal{D} \,|\, \gamma), p(\mathcal{D} \,|\, \hat{\gamma})\right) \quad \text{(Equation 4.24)} \\
&= p_0(\hat{\gamma})\; q(\gamma \leftarrow \hat{\gamma}) \min\left(p(\mathcal{D} \,|\, \gamma), p(\mathcal{D} \,|\, \hat{\gamma})\right) \frac{p(\mathcal{D} \,|\, \hat{\gamma})}{p(\mathcal{D} \,|\, \hat{\gamma})}
\end{aligned}$$

[3] Hamiltonian Monte Carlo will be discussed in more detail in Chapter 5.



(a) $\kappa = 0.6$ (b) $\kappa = 0.9$ (c) $\kappa = 0.99$

Figure 4.2: Examples of underrelaxed proposals for Gaussian processes, using different values of κ, as in Equation 4.27. The black line shows the initial state, $\mathcal{G}$, and the five coloured lines show different proposals, $\hat{\mathcal{G}}$. The five underlying $\mathcal{H}$ are shared across the figures.

$$= p_0(\hat{\gamma})\, p(\mathcal{D} \,|\, \hat{\gamma})\; q(\gamma \leftarrow \hat{\gamma}) \min\left(1, \frac{p(\mathcal{D} \,|\, \gamma)}{p(\mathcal{D} \,|\, \hat{\gamma})}\right)$$
$$= T(\gamma \leftarrow \hat{\gamma})\; p_0(\hat{\gamma})\, p(\mathcal{D} \,|\, \hat{\gamma})$$

□

Naturally, proposals via independent draws from the prior, as in Section 4.3.1, fit the detailed balance condition. Independence chain Metropolis–Hastings is not an interesting case, however, for the reasons discussed previously. Similarly, generating the proposal via Metropolis–Hastings on the prior also satisfies detailed balance. This procedure would result in the well-known MH *two-stage acceptance rule* (e.g. Murray (2007, Chapter 2)). I also consider this an uninteresting case for my purposes, as it only provides a way of making early rejections and does not actually increase the acceptance rate. In fact, the two-stage acceptance rule would likely *decrease* the overall acceptance rate as it would occasionally veto some proposals that would have been accepted.

More useful for this thesis are underrelaxed proposals that are conservative but that can be made directly and not as the result of an intermediate accept/reject step. Specifically, I will discuss a transition operator that is invariant under a Gaussian prior, such as that arising on the function values of the control points in the previous section.

Lemma 4.2. *Consider a transition from $\boldsymbol{z}$ to $\hat{\boldsymbol{z}}$, $q(\hat{\boldsymbol{z}} \leftarrow \boldsymbol{z})$, for $\boldsymbol{z}, \hat{\boldsymbol{z}} \in \mathbb{R}^D$, such that*

$$\hat{\boldsymbol{z}} = \kappa(\boldsymbol{z} - \boldsymbol{\mu}) + \sqrt{1 - \kappa^2}\,(\boldsymbol{\eta} - \boldsymbol{\mu}) + \boldsymbol{\mu}, \tag{4.25}$$

for $\boldsymbol{\eta} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, and $-1 < \kappa < 1$. Then $q(\hat{\boldsymbol{z}} \leftarrow \boldsymbol{z})$ satisfies detailed balance for the distribution $p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, i.e. $q(\hat{\boldsymbol{z}} \leftarrow \boldsymbol{z})$ meets the condition of Equation 4.24 where the equilibrium distribution is a D-dimensional Gaussian with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$.

Proof. Start with the joint distribution over $\boldsymbol{z}$ and $\hat{\boldsymbol{z}}$:

$$\mathcal{N}(\boldsymbol{z} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma})\; q(\hat{\boldsymbol{z}} \leftarrow \boldsymbol{z}) = \mathcal{N}(\boldsymbol{z} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma})\; \mathcal{N}(\hat{\boldsymbol{z}} \mid \boldsymbol{\mu} + \kappa(\boldsymbol{z} - \boldsymbol{\mu}), (1-\kappa^2)\boldsymbol{\Sigma}). \tag{4.26}$$

The log probability of this joint distribution is

$$-\frac{1}{2}\left[(\boldsymbol{z}-\boldsymbol{\mu})^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{z}-\boldsymbol{\mu}) + (1-\kappa^2)^{-1}(\hat{\boldsymbol{z}}-\boldsymbol{\mu}-\kappa(\boldsymbol{z}-\boldsymbol{\mu}))^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}(\hat{\boldsymbol{z}}-\boldsymbol{\mu}-\kappa(\boldsymbol{z}-\boldsymbol{\mu}))\right] + \text{const},$$

which can be rewritten as

$$-\frac{1}{2(1-\kappa^2)}\left[(\boldsymbol{z}-\boldsymbol{\mu})^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{z}-\boldsymbol{\mu}) - 2\kappa(\hat{\boldsymbol{z}}-\boldsymbol{\mu})^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{z}-\boldsymbol{\mu}) + (\hat{\boldsymbol{z}}-\boldsymbol{\mu})^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}(\hat{\boldsymbol{z}}-\boldsymbol{\mu})\right] + \text{const},$$

which is symmetric in $\boldsymbol{z}$ and $\hat{\boldsymbol{z}}$. This means that in Equation 4.26, $\boldsymbol{z}$ and $\hat{\boldsymbol{z}}$ could be exchanged without changing the value of the joint distribution. This is equivalent to detailed balance. □

To make conservative proposals, κ is chosen to be close to one. At each step, a new proposal for control points is made by perturbing the current state with a new draw from the Gaussian process, using Equation 4.25. In the notation of the previous section, the proposal is made via

$$\hat{\mathcal{G}} = \kappa\,\mathcal{G} + \sqrt{1-\kappa^2}\,\mathcal{H}, \tag{4.27}$$

where $\mathcal{H} \sim \mathcal{GP}(\mathcal{C}, \boldsymbol{\theta})$ is an independent draw from the Gaussian process prior at the control points. Figure 4.2 shows the sorts of proposals that result from underrelaxation on Gaussian processes. Additional retrospective function evaluations are drawn from the Gaussian process prior conditioning on these control points and any other values in the conditioning set. Ultimately, the exchange sampling acceptance ratio for the underrelaxed control point method is Equation 4.21. The algorithm is provided in pseudocode in Algorithm 4.4.

4.3.3 Hyperparameter Inference

One of the appealing aspects of Bayesian probabilistic modeling is the ability to construct and perform inference in hierarchical models. In the case of the Gaussian process density sampler, this corresponds to inference of the hyperparameters $\boldsymbol{\theta}$ that govern the covariance function, and ψ_π that control the behaviour of the base density.

Inferring Hyperparameters of the Covariance Function

It is possible to perform hyperparameter inference on $\boldsymbol{\theta}$ by including it in the exchange sampling Markov chain state. The transitions of Section 4.3.2 are augmented so that

Algorithm 4.4 Simulate R steps using underrelaxed control point ES on the GPDS.

Inputs:
- Number of MCMC iterations R
- Observed data $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^N$
- Control points $\mathcal{C} = \{\boldsymbol{x}_b\}_{b=1}^B$
- Underrelaxation parameter κ
- Gaussian process covariance function $C(\boldsymbol{x}, \boldsymbol{x}' ; \boldsymbol{\theta})$
- Base density $\pi(\boldsymbol{x} \,|\, \boldsymbol{\psi}_\pi)$

Outputs:
- R conditioning sets of function inputs and outputs $\{\boldsymbol{X}^{(r)}, \boldsymbol{G}^{(r)}\}_{r=1}^R$

$\mathcal{G} \sim \mathcal{GP}(g \,|\, \mathcal{C}, \boldsymbol{\theta})$ ▷ Initialise the function at the control points.
$\boldsymbol{X}^{(1)} \leftarrow \mathcal{C}$, $\boldsymbol{G}^{(1)} \leftarrow \mathcal{G}$ ▷ Initialise conditioning sets.
for $r \leftarrow 1 \ldots R$ **do** ▷ Take R exchange sampling steps.
  $\mathcal{H} \sim \mathcal{GP}(g \,|\, \mathcal{C}, \boldsymbol{\theta})$ ▷ Make the independent GP draw for the proposal.
  $\hat{\mathcal{G}} \leftarrow \kappa \mathcal{G} + \sqrt{1 - \kappa^2}\, \mathcal{H}$ ▷ Construct the actual proposal.
  $\hat{\boldsymbol{X}} \leftarrow \mathcal{C}$, $\hat{\boldsymbol{G}} \leftarrow \hat{\mathcal{G}}$ ▷ Initialise proposal conditioning sets.
  $\mathcal{W} \leftarrow \emptyset$ ▷ Initialise empty fantasy data set.
  repeat ▷ Loop to generate fantasy data.
    $\tilde{\boldsymbol{w}} \sim \pi(\boldsymbol{x} \,|\, \boldsymbol{\psi}_\pi)$ ▷ Draw a proposal from the base density.
    $\hat{g}(\tilde{\boldsymbol{w}}) \sim \mathcal{GP}(g \,|\, \tilde{\boldsymbol{w}}, \hat{\boldsymbol{X}}, \hat{\boldsymbol{G}}, \boldsymbol{\theta})$ ▷ Draw the function value at the proposal.
    $u_{\mathsf{fant}} \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on $(0, 1)$.
    if $u_{\mathsf{fant}} < \Phi(\hat{g}(\tilde{\boldsymbol{w}}))$ **then** ▷ Rejection sampling acceptance rule.
      $\mathcal{W} \leftarrow \mathcal{W} \cup \tilde{\boldsymbol{w}}$ ▷ Keep the fantasy.
    end if
    $\hat{\boldsymbol{X}} \leftarrow \hat{\boldsymbol{X}} \cup \tilde{\boldsymbol{w}}$, $\hat{\boldsymbol{G}} \leftarrow \hat{\boldsymbol{G}} \cup \hat{g}(\tilde{\boldsymbol{w}})$ ▷ Add proposals to the conditioning sets.
  until $|\mathcal{W}| = N$ ▷ Loop until N fantasies are accepted.
  $\{g(\boldsymbol{w}_n)\}_{n=1}^N \sim \mathcal{GP}(g \,|\, \mathcal{W}, \boldsymbol{X}^{(r)}, \boldsymbol{G}^{(r)})$ ▷ Sample the current function at the fantasies.
  $a_{\mathsf{gpds-ures}} \leftarrow \prod_{n=1}^N \frac{\Phi(\hat{g}(x_n))\ \Phi(g(w_n))}{\Phi(g(x_n))\ \Phi(\hat{g}(w_n))}$ ▷ Calculate the acceptance ratio.
  $u_{\mathsf{mh}} \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on $(0, 1)$.
  if $u_{\mathsf{mh}} < a_{\mathsf{gpds-ures}}$ **then** ▷ Apply the Metropolis–Hastings acceptance rule.
    $\boldsymbol{X}^{(r+1)} \leftarrow \hat{\boldsymbol{X}}$, $\boldsymbol{G}^{(r+1)} \leftarrow \hat{\boldsymbol{G}}$ ▷ Keep the new function data.
    $\mathcal{G} \leftarrow \hat{\mathcal{G}}$ ▷ Keep the new control point function values.
  else
    $\boldsymbol{X}^{(p+1)} \leftarrow \boldsymbol{X}^{(r)} \cup \{\boldsymbol{w}_n\}_{n=1}^N$ ▷ Add the fantasy evaluations to the current state.
    $\boldsymbol{G}^{(p+1)} \leftarrow \boldsymbol{G}^{(r)} \cup \{g(\boldsymbol{w}_n)\}_{n=1}^N$
  end if
end for
return $\{\boldsymbol{X}^{(r)}, \boldsymbol{G}^{(r)}\}_{r=1}^R$

before proposing control point function updates, new hyperparameters are first proposed from a distribution $q(\hat{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta})$. The control point function value proposals $\hat{\mathcal{G}}$ are made while conditioning upon the new hyperparameters. Additional retrospective function draws for $\hat{g}(\boldsymbol{x})$, including those for fantasisation, are also made using the newly proposed hyperparameters $\hat{\boldsymbol{\theta}}$. Retrospective draws of the current function $g(\boldsymbol{x})$, however, are made using the current hyperparameters $\boldsymbol{\theta}$. The exchange proposal now includes both the new function and the new hyperparameters. The pre-exchange joint

distribution, including the hyperprior on $\boldsymbol{\theta}$, is

$$p(\boldsymbol{\theta}, \mathcal{G}, \boldsymbol{g}_{\setminus\mathcal{C}}, \mathcal{D}, \hat{\boldsymbol{\theta}}, \hat{\mathcal{G}}, \hat{\boldsymbol{g}}_{\setminus\mathcal{C}}, \mathcal{W} \,|\, \mathcal{C}, \boldsymbol{\psi}_\pi) = \\ p_0(\boldsymbol{\theta})\; \mathcal{GP}(\mathcal{G} \,|\, \mathcal{C}, \boldsymbol{\theta})\; \mathcal{GP}(\boldsymbol{g}_{\setminus\mathcal{C}} \,|\, \mathcal{G}, \boldsymbol{\theta})\; \mathcal{Z}_\pi[\boldsymbol{g}]^{-N} \left[\prod_{n=1}^{N} \Phi(g(\boldsymbol{x}_n))\; \pi(\boldsymbol{x}_n \,|\, \boldsymbol{\psi}_\pi)\right] \\ \times\; q(\hat{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta})\; q(\hat{\mathcal{G}} \leftarrow \mathcal{G}; \hat{\boldsymbol{\theta}})\; \mathcal{GP}(\hat{\boldsymbol{g}}_{\setminus\mathcal{C}} \,|\, \hat{\mathcal{G}}, \hat{\boldsymbol{\theta}})\; \mathcal{Z}_\pi[\hat{\boldsymbol{g}}]^{-N} \prod_{n=1}^{N} \Phi(\hat{g}(\boldsymbol{w}_n))\; \pi(\boldsymbol{w}_n \,|\, \boldsymbol{\psi}_\pi), \quad (4.28)$$

and the acceptance ratio of exchanging the triplet $(\boldsymbol{\theta}, \mathcal{G}, \boldsymbol{g}_{\setminus\mathcal{C}})$ for $(\hat{\boldsymbol{\theta}}, \hat{\mathcal{G}}, \hat{\boldsymbol{g}}_{\setminus\mathcal{C}})$ is

$$a_{\mathsf{gpds-gphp}} = \frac{p_0(\hat{\boldsymbol{\theta}})\; \mathcal{GP}(\hat{\mathcal{G}} \,|\, \mathcal{C}, \hat{\boldsymbol{\theta}})\; q(\boldsymbol{\theta} \leftarrow \hat{\boldsymbol{\theta}})\; q(\mathcal{G} \leftarrow \hat{\mathcal{G}}; \boldsymbol{\theta})}{p_0(\boldsymbol{\theta})\; \mathcal{GP}(\mathcal{G} \,|\, \mathcal{C}, \boldsymbol{\theta})\; q(\hat{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta})\; q(\hat{\mathcal{G}} \leftarrow \mathcal{G}; \hat{\boldsymbol{\theta}})} \prod_{n=1}^{N} \frac{\Phi(\hat{g}(\boldsymbol{x}_n))\; \Phi(g(\boldsymbol{w}_n))}{\Phi(g(\boldsymbol{x}_n))\; \Phi(\hat{g}(\boldsymbol{w}_n))}. \quad (4.29)$$

The underrelaxation trick cannot be used for hyperparameter updates because the joint proposal distribution for $\hat{\boldsymbol{g}}$ and $\hat{\boldsymbol{\theta}}$ would not generally satisfy detailed balance for the corresponding joint prior. Conservative proposals are still suggested, but the prior and proposal densities must be explicitly included in the acceptance ratio. Pseudocode for taking an exchange sampling step on the hyperparameters is provided in Appendix A in Algorithm A.7 (page 121).

Inferring Hyperparameters of the Base Density

Sampling from the hyperparameters $\boldsymbol{\psi}_\pi$ of the base density can also be done by augmenting the exchange sampling Markov chain. In this case, however, it is possible to assume a fixed latent function $\boldsymbol{g}$, by using a single conditioning set for all retrospective samples, including those for generating the fantasies $\mathcal{W} = \{\boldsymbol{w}_n\}_{n=1}^N$. As in previous discussions of exchange sampling, a proposal is drawn from the distribution $q(\hat{\boldsymbol{\psi}}_\pi \leftarrow \boldsymbol{\psi}_\pi)$, followed by fantasisation using these new hyperparameters. The joint distribution over the current hyperparameters, the data, the new hyperparameters, and the fantasies is

$$p(\boldsymbol{\psi}_\pi, \mathcal{D}, \hat{\boldsymbol{\psi}}_\pi, \mathcal{W} \,|\, \boldsymbol{g}) = p_0(\boldsymbol{\psi}_\pi)\; \mathcal{Z}_\pi[\boldsymbol{g}]^{-N} \left[\prod_{n=1}^{N} \Phi(g(\boldsymbol{x}_n))\; \pi(\boldsymbol{x}_n \,|\, \boldsymbol{\psi}_\pi)\right] \\ \times\; q(\hat{\boldsymbol{\psi}}_\pi \leftarrow \boldsymbol{\psi}_\pi)\; \hat{\mathcal{Z}}_\pi[\boldsymbol{g}]^{-N} \prod_{n=1}^{N} \Phi(g(\boldsymbol{w}_n))\; \pi(\boldsymbol{w}_n \,|\, \hat{\boldsymbol{\psi}}_\pi), \quad (4.30)$$

where I use $\hat{\mathcal{Z}}_\pi[\boldsymbol{g}]$ to indicate the dependence of the normalisation constant on $\hat{\boldsymbol{\psi}}_\pi$. The acceptance ratio of the exchange of $\boldsymbol{\psi}_\pi$ for $\hat{\boldsymbol{\psi}}_\pi$ is

$$a_{\mathsf{gpds-bdhp}} = \frac{p_0(\hat{\boldsymbol{\psi}}_\pi)\; q(\boldsymbol{\psi}_\pi \leftarrow \hat{\boldsymbol{\psi}}_\pi)}{p_0(\boldsymbol{\psi}_\pi)\; q(\hat{\boldsymbol{\psi}}_\pi \leftarrow \boldsymbol{\psi}_\pi)} \prod_{n=1}^{N} \frac{\pi(\boldsymbol{x}_n \,|\, \hat{\boldsymbol{\psi}}_\pi)\; \pi(\boldsymbol{w}_n \,|\, \boldsymbol{\psi}_\pi)}{\pi(\boldsymbol{x}_n \,|\, \boldsymbol{\psi}_\pi)\; \pi(\boldsymbol{w}_n \,|\, \hat{\boldsymbol{\psi}}_\pi)}. \quad (4.31)$$

Algorithm A.8 (page 122) in Appendix A provides pseudocode for taking an exchange sampling step on the base density hyperparameters.

4.4 Exchange Sampling for SGCP Inference

Having explained exchange sampling in Section 4.2 and applied it to the Gaussian process density sampler in Section 4.3, I now apply exchange sampling to the sigmoidal Gaussian Cox process of Chapter 3. Inference in the SGCP is very similar to the GPDS, so I will skip over the independence chain approach and will start directly from the control point method.

In the domain $\mathcal{V}$, there are K observed events, $\mathcal{S} = \{\boldsymbol{x}_k\}_{k=1}^K$. Choose a set of $B \geq K$ control points $\mathcal{C} = \{\boldsymbol{x}_b\}_{b=1}^B$ that includes the observed events as a subset, i.e. $\mathcal{S} \subseteq \mathcal{C}$. Draws from the Gaussian process prior will be conditioned on the function values $\mathcal{G} = \{g(\boldsymbol{x}_b)\}_{b=1}^B$ at these points. As in Section 4.3.2, the number B and location $\mathcal{C}$ of these control points are free parameters, but they remain fixed throughout the algorithm. The first step in an exchange sampling transition is to draw new control point function values $\hat{\mathcal{G}} = \{\hat{g}(\boldsymbol{x}_b)\}_{b=1}^B$ from a proposal density $q(\hat{\mathcal{G}} \leftarrow \mathcal{G})$. Next, a fantasy set of events is drawn from the sigmoidal Gaussian Cox process using Algorithm 3.1, as described in Section 3.4, while conditioning on $\mathcal{C}$ and $\hat{\mathcal{G}}$. Unlike the Gaussian process density sampler, the actual number of fantasy events may not be the same as in the original data. I will denote the number of fantasy events as $\hat{K}$. After the fantasies are generated, the current function $g(\boldsymbol{x})$ is retrospectively sampled at the fantasy events. Finally, the proposal to exchange the pair $(\mathcal{G}, \boldsymbol{g}_{\setminus\mathcal{C}})$ for $(\hat{\mathcal{G}}, \hat{\boldsymbol{g}}_{\setminus\mathcal{C}})$ is accepted or rejected, where $\boldsymbol{g}_{\setminus\mathcal{C}}$ denotes the values of the function $g(\boldsymbol{x})$, excluding those at the control points. As in Gaussian process density sampler inference, all retrospective evaluations, such as from thinned events during generation of fantasies, must be kept in the Markov state. The control points are simply a way to make more conservative proposals. Also as in the GPDS case, rejecting the exchange causes expansion of the Markov state, via retrospective sampling of the *current* function at the *fantasy* events.

The joint distribution over current and proposed functions, the K observed events $\{\boldsymbol{x}_k\}_{k=1}^K$, and the $\hat{K}$ fantasy events $\{\boldsymbol{w}_k\}_{k=1}^{\hat{K}}$ — the SGCP equivalent of

Equation 4.22 — is

$$p(\mathcal{G}, \boldsymbol{g}_{\setminus\mathcal{C}}, K, \{\boldsymbol{x}_k\}_{k=1}^{K}, \hat{\mathcal{G}}, \hat{\boldsymbol{g}}_{\setminus\mathcal{C}}, \hat{K}, \{\boldsymbol{w}_k\}_{k=1}^{\hat{K}} \,|\, \boldsymbol{\theta}, \boldsymbol{\psi}_\lambda) = \mathcal{GP}(\mathcal{G} \,|\, \mathcal{C}, \boldsymbol{\theta})\, \mathcal{GP}(\boldsymbol{g}_{\setminus\mathcal{C}} \,|\, \mathcal{G}, \boldsymbol{\theta})$$
$$\times \exp\left\{-\int_{\mathcal{V}} \mathrm{d}\boldsymbol{x}'\, \Phi(g(\boldsymbol{x}'))\, \bar{\lambda}(\boldsymbol{x}' \,;\, \boldsymbol{\psi}_\lambda)\right\}\left[\prod_{k=1}^{K} \Phi(g(\boldsymbol{x}_k))\, \bar{\lambda}(\boldsymbol{x}_k \,;\, \boldsymbol{\psi}_\lambda)\right]$$
$$\times\, q(\hat{\mathcal{G}} \leftarrow \mathcal{G})\, \mathcal{GP}(\hat{\boldsymbol{g}}_{\setminus\mathcal{C}} \,|\, \hat{\mathcal{G}}, \boldsymbol{\theta})$$
$$\times \exp\left\{-\int_{\mathcal{V}} \mathrm{d}\boldsymbol{x}'\, \Phi(\hat{g}(\boldsymbol{x}'))\, \bar{\lambda}(\boldsymbol{x}' \,;\, \boldsymbol{\psi}_\lambda)\right\}\prod_{k=1}^{\hat{K}} \Phi(\hat{g}(\boldsymbol{w}_k))\, \bar{\lambda}(\boldsymbol{w}_k \,;\, \boldsymbol{\psi}_\lambda). \quad (4.32)$$

The proposal to exchange the pair $(\mathcal{G}, \boldsymbol{g}_{\setminus\mathcal{C}})$ for the pair $(\hat{\mathcal{G}}, \hat{\boldsymbol{g}}_{\setminus\mathcal{C}})$ has acceptance ratio

$$a_{\mathsf{sgcp-cpes}} = \frac{q(\mathcal{G} \leftarrow \hat{\mathcal{G}})\; \mathcal{GP}(\hat{\mathcal{G}} \,|\, \mathcal{C}, \boldsymbol{\theta})}{q(\hat{\mathcal{G}} \leftarrow \mathcal{G})\; \mathcal{GP}(\mathcal{G} \,|\, \mathcal{C}, \boldsymbol{\theta})} \left[\prod_{k=1}^{K} \frac{\Phi(\hat{g}(\boldsymbol{x}_k))}{\Phi(g(\boldsymbol{x}_k))}\right] \prod_{k=1}^{\hat{K}} \frac{\Phi(g(\boldsymbol{w}_k))}{\Phi(\hat{g}(\boldsymbol{w}_k))}, \quad (4.33)$$

where the difficult integrals have canceled out.

It is also possible in the sigmoidal Gaussian Cox process to apply the underrelaxed proposal trick that I discussed in Section 4.3.2. If the proposals for new control point function values are made using Equation 4.27, then after generating the fantasy events, the acceptance ratio is

$$a_{\mathsf{sgcp-ures}} = \left[\prod_{k=1}^{K} \frac{\Phi(\hat{g}(\boldsymbol{x}_k))}{\Phi(g(\boldsymbol{x}_k))}\right] \prod_{k=1}^{\hat{K}} \frac{\Phi(g(\boldsymbol{w}_k))}{\Phi(\hat{g}(\boldsymbol{w}_k))}. \quad (4.34)$$

The underrelaxed control point exchange sampling algorithm for the sigmoidal Gaussian Cox process is provided as pseudocode in Algorithm 4.5.

4.4.1 Hyperparameter Inference

As with the Gaussian process density sampler in Section 4.3.3, it is possible to augment the exchange sampling Markov chain to sample from the Gaussian process hyperparameters $\boldsymbol{\theta}$ and any hyperparameters $\boldsymbol{\psi}_\lambda$ governing the dominating intensity $\bar{\lambda}(\boldsymbol{x} \,;\, \boldsymbol{\psi}_\lambda)$.

Inferring Hyperparameters of the Covariance Function

Sampling from Gaussian process hyperparameters in the sigmoidal Gaussian Cox process is similar to the method described in Section 4.3.3. Propose new hyperparameters $\hat{\boldsymbol{\theta}}$, draw new control point function values $\hat{\mathcal{G}}$, then generate fantasy events conditioned upon these two proposals. The exchange proposes to swap the triplet $(\boldsymbol{\theta}, \mathcal{G}, \boldsymbol{g}_{\setminus\mathcal{C}})$

Algorithm 4.5 Simulate R steps using underrelaxed control point ES on the SGCP.

Inputs:
- Number of MCMC iterations R
- Domain $\mathcal{V}$
- Observed data $\{\boldsymbol{x}_k\}_{k=1}^K$
- Control points $\mathcal{C} = \{\boldsymbol{x}_b\}_{b=1}^B$
- Underrelaxation parameter κ
- Gaussian process covariance function $C(\boldsymbol{x}, \boldsymbol{x}' ; \boldsymbol{\theta})$
- Dominating intensity $\bar{\lambda}(\boldsymbol{x} ; \boldsymbol{\psi}_\lambda)$

Outputs:
- R conditioning sets of function inputs and outputs $\{\boldsymbol{X}^{(r)}, \boldsymbol{G}^{(r)}\}_{r=1}^R$

$\mathcal{G} \sim \mathcal{GP}(g \,|\, \mathcal{C}, \boldsymbol{\theta})$ ▷ Initialise the function at the control points.
$\boldsymbol{X}^{(1)} \leftarrow \mathcal{C}$, $\boldsymbol{G}^{(1)} \leftarrow \mathcal{G}$ ▷ Initialise conditioning sets.
for $r \leftarrow 1 \ldots R$ **do** ▷ Take R exchange sampling steps.
  $\mathcal{H} \sim \mathcal{GP}(g \,|\, \mathcal{C}, \boldsymbol{\theta})$ ▷ Make the independent GP draw for the proposal.
  $\hat{\mathcal{G}} \leftarrow \kappa \mathcal{G} + \sqrt{1 - \kappa^2}\, \mathcal{H}$ ▷ Construct the actual proposal.
  $\hat{\boldsymbol{X}} \leftarrow \mathcal{C}$, $\hat{\boldsymbol{G}} \leftarrow \hat{\mathcal{G}}$ ▷ Initialise proposal conditioning sets.
  $\{\tilde{\boldsymbol{w}}_j\}_{j=1}^J \sim \mathcal{PP}(\bar{\lambda}(\boldsymbol{x} ; \boldsymbol{\psi}_\lambda), \mathcal{V})$ ▷ Draw from the bounding intensity.
  $\{\hat{g}(\tilde{\boldsymbol{w}}_j)\}_{j=1}^J \sim \mathcal{GP}(g \,|\, \{\tilde{\boldsymbol{w}}_j\}_{j=1}^J, \hat{\boldsymbol{X}}, \hat{\boldsymbol{G}})$ ▷ Sample the proposed function at the events.
  $\hat{\boldsymbol{X}} \leftarrow \hat{\boldsymbol{X}} \cup \{\tilde{\boldsymbol{w}}_j\}_{j=1}^J$, $\hat{\boldsymbol{G}} \leftarrow \hat{\boldsymbol{G}} \cup \{g(\tilde{\boldsymbol{w}}_j)\}_{j=1}^J$ ▷ Update the proposed conditioning set.
  $\mathcal{W} \leftarrow \emptyset$ ▷ Initialise fantasy set.
  for $j \leftarrow 1 \ldots J$ **do** ▷ Loop over the events.
    $u_{\mathsf{fant}} \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on $(0, 1)$.
    if $u_{\mathsf{fant}} < \Phi(\hat{g}(\tilde{\boldsymbol{w}}_j))$ **then** ▷ Apply the thinning rule.
      $\mathcal{W} \leftarrow \mathcal{W} \cup \tilde{\boldsymbol{w}}_j$ ▷ Add the event to the fantasy set.
    end if
  end for
  $\{g(\boldsymbol{w}_k)\}_{k=1}^{\hat{K}} \sim \mathcal{GP}(g \,|\, \mathcal{W}, \boldsymbol{X}^{(r)}, \boldsymbol{G}^{(r)})$ ▷ Sample the current function at the fantasies.
  $a_{\mathsf{sgcp-ures}} \leftarrow \left[\prod_{k=1}^K \frac{\Phi(\hat{g}(\boldsymbol{x}_k))}{\Phi(g(\boldsymbol{x}_k))}\right] \left[\prod_{k=1}^{\hat{K}} \frac{\Phi(g(\boldsymbol{w}_k))}{\Phi(\hat{g}(\boldsymbol{w}_k))}\right]$ ▷ Calculate the acceptance ratio.
  $u_{\mathsf{mh}} \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on $(0, 1)$.
  if $u_{\mathsf{mh}} < a_{\mathsf{sgcp-ures}}$ **then** ▷ Apply the Metropolis–Hastings acceptance rule.
    $\boldsymbol{X}^{(r+1)} \leftarrow \hat{\boldsymbol{X}}$, $\boldsymbol{G}^{(r+1)} \leftarrow \hat{\boldsymbol{G}}$ ▷ Keep the new function data.
    $\mathcal{G} \leftarrow \hat{\mathcal{G}}$ ▷ Keep the new control points.
  else
    $\boldsymbol{X}^{(r+1)} \leftarrow \boldsymbol{X}^{(r)} \cup \{\boldsymbol{w}_k\}_{k=1}^{\hat{K}}$ ▷ Add the fantasy evaluations to the current state.
    $\boldsymbol{G}^{(r+1)} \leftarrow \boldsymbol{G}^{(r)} \cup \{g(\boldsymbol{w}_k)\}_{k=1}^{\hat{K}}$
  end if
end for
return $\{\boldsymbol{X}^{(r)}, \boldsymbol{G}^{(r)}\}_{r=1}^R$

for $(\hat{\boldsymbol{\theta}}, \hat{\mathcal{G}}, \hat{\boldsymbol{g}}_{\setminus \mathcal{C}})$, with acceptance ratio

$$a_{\mathsf{sgcp-gphp}} = \frac{p_0(\hat{\boldsymbol{\theta}})\, \mathcal{GP}(\hat{\mathcal{G}} \,|\, \mathcal{C}, \hat{\boldsymbol{\theta}})\, q(\boldsymbol{\theta} \leftarrow \hat{\boldsymbol{\theta}})\, q(\mathcal{G} \leftarrow \hat{\mathcal{G}}; \boldsymbol{\theta})}{p_0(\boldsymbol{\theta})\, \mathcal{GP}(\mathcal{G} \,|\, \mathcal{C}, \boldsymbol{\theta})\, q(\hat{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta})\, q(\hat{\mathcal{G}} \leftarrow \hat{\mathcal{G}}; \hat{\boldsymbol{\theta}})} \times \left[\prod_{k=1}^K \frac{\Phi(\hat{g}(\boldsymbol{x}_k))}{\Phi(g(\boldsymbol{x}_k))}\right] \prod_{k=1}^{\hat{K}} \frac{\Phi(g(\boldsymbol{w}_k))}{\Phi(\hat{g}(\boldsymbol{w}_k))}. \quad (4.35)$$

This algorithm is provided as pseudocode in Appendix A as Algorithm A.9 (page 123).

Inferring Hyperparameters of the Dominating Intensity

It is also possible to sample from the hyperparameters $\boldsymbol{\psi}_\lambda$ that govern the dominating intensity $\bar{\lambda}(\boldsymbol{x}\,;\,\boldsymbol{\psi}_\lambda)$. This is analogous to sampling from the base density hyperparameters in Section 4.3.3. If the current hyperparameters are $\boldsymbol{\psi}_\lambda$, then a new proposal $\hat{\boldsymbol{\psi}}_\lambda$ is drawn from the distribution $q(\hat{\boldsymbol{\psi}}_\lambda \leftarrow \boldsymbol{\psi}_\lambda)$. It is not necessary to propose a new function — the current conditioning set can be used to generate the fantasies. The joint distribution over the current and proposed hyperparameters, and the observed and fantasy data, is

$$\begin{aligned} p(\boldsymbol{\psi}_\lambda, K, \{\boldsymbol{x}_k\}_{k=1}^K, \hat{\boldsymbol{\psi}}_\lambda, \hat{K}, \{\boldsymbol{w}_k\}_{k=1}^{\hat{K}} \,|\, \boldsymbol{g}) = \\ p_0(\boldsymbol{\psi}_\lambda)\ \exp\left\{-\int_{\mathcal{V}} \mathrm{d}\boldsymbol{x}'\ \Phi(g(\boldsymbol{x}'))\ \bar{\lambda}(\boldsymbol{x}'\,;\,\boldsymbol{\psi}_\lambda)\right\}\left[\prod_{k=1}^{K} \Phi(g(\boldsymbol{x}_k))\ \bar{\lambda}(\boldsymbol{x}_k\,;\,\boldsymbol{\psi}_\lambda)\right] \\ \times\ q(\hat{\boldsymbol{\psi}}_\lambda \leftarrow \boldsymbol{\psi}_\lambda)\ \exp\left\{-\int_{\mathcal{V}} \mathrm{d}\boldsymbol{x}'\ \Phi(g(\boldsymbol{x}'))\ \bar{\lambda}(\boldsymbol{x}'\,;\,\hat{\boldsymbol{\psi}}_\lambda)\right\}\prod_{k=1}^{\hat{K}} \Phi(g(\boldsymbol{w}_k))\ \bar{\lambda}(\boldsymbol{w}_k\,;\,\hat{\boldsymbol{\psi}}_\lambda). \end{aligned} \tag{4.36}$$

The acceptance ratio of exchanging $\boldsymbol{\psi}_\lambda$ for $\hat{\boldsymbol{\psi}}_\lambda$ is

$$a_{\mathsf{sgcp-bihp}} = \frac{q(\boldsymbol{\psi}_\lambda \leftarrow \hat{\boldsymbol{\psi}}_\lambda)\ p_0(\hat{\boldsymbol{\psi}}_\lambda)}{q(\hat{\boldsymbol{\psi}}_\lambda \leftarrow \boldsymbol{\psi}_\lambda)\ p_0(\boldsymbol{\psi}_\lambda)} \left[\prod_{k=1}^{K} \frac{\bar{\lambda}(\boldsymbol{x}_k\,;\,\hat{\boldsymbol{\psi}}_\lambda)}{\bar{\lambda}(\boldsymbol{x}_k\,;\,\boldsymbol{\psi}_\lambda)}\right] \prod_{k=1}^{\hat{K}} \frac{\bar{\lambda}(\boldsymbol{w}_k\,;\,\boldsymbol{\psi}_\lambda)}{\bar{\lambda}(\boldsymbol{w}_k\,;\,\hat{\boldsymbol{\psi}}_\lambda)}. \tag{4.37}$$

This algorithm is provided as pseudocode in Appendix A as Algorithm A.10 (page 124).

4.5 Predictive Samples

Commonly when performing inference, we are interested in the predictive distribution. This is the distribution that arises on the data space after integrating out the posterior distribution over parameters. It can be viewed as the distribution over the *next* datum, having seen a set of data, and taking into account uncertainty. While this distribution is not available in closed form for the Gaussian process density sampler or the sigmoidal Gaussian Cox process, it is possible to generate samples from it. To generate predictive samples, simulate the exchange sampling Markov chain algorithm and after each update, run the generative procedure using the current state to initialise the conditioning set. This is done exactly as it was when generating fantasy data using Algorithm 2.2 and Algorithm 3.1, but starting with the conditioning set that is the current state of the Markov chain.

4.6 Summary

In this chapter I presented one method of inference for the Gaussian process density sampler model and the sigmoidal Gaussian Cox process model. This method, exchange sampling, relied on the ability to generate exact data from the models for a given setting of the parameters. The result was a Metropolis–Hastings sampler in which acceptance ratios could be calculated without estimation of a ratio of intractable normalisation constants. Remarkably, it is possible to construct a Markov chain with the Bayesian posterior as its equilibrium distribution for both models, even though the objects of inference have infinite dimensions. I also showed that for both models it is possible to perform hyperparameter inference and discussed how one can sample from the predictive distribution.

In Chapter 5, I will present an alternate method for inference in these two models and show how, for the Gaussian process density sampler and the sigmoidal Gaussian Cox process, it is preferable to exchange sampling.

Chapter 5

Inference Via Latent Histories

In this chapter, I present a method for inference in the Gaussian process density sampler and the sigmoidal Gaussian Cox process that is superior to the exchange sampling approach of Chapter 4. This method takes advantage of the generative procedures of the GPDS and the SGCP to construct latent variable models in which inference can be performed via Markov chain Monte Carlo, without it being necessary to compute ratios of intractable normalisation constants.

I begin with a discussion of latent history inference in Section 5.1. In Section 5.2, I show how the latent history method can be applied to the Gaussian process density sampler. I apply the method to the sigmoidal Gaussian Cox process in Section 5.3. In Section 5.4, I discuss how the latent history approach differs from exchange sampling, and why it is an improvement. I describe how the latent history method can be extended to allow inference in SGCP-derived interacting point processes in Section 5.5.

5.1 Modeling the Latent History of the Generative Procedure

Modeling data with a generative prior is, in effect, asserting that the observed data are the result of running the relevant generative procedure. Typically, nothing is known about the various intermediate states that led to the final observations, but it is nevertheless possible to model these unknown states with latent variables. One can generate samples from the posterior distribution over this *latent history* of the generative procedure, and then keep samples from the parameters of interest.

Generative models as a concept are as old as probabilistic modeling itself. The idea captured by "inference via latent histories," however, is to model some specific computational procedure that is not necessarily motivated by a natural process. In Chapter 2 and Chapter 3, I presented generative processes based on rejection sampling and thinning, respectively. These are "artificial" in the sense that it is not realistic to think that variation in bus arrival rates is the result of the independent removal of some

buses. Still, the Gaussian process density sampler and the sigmoidal Gaussian Cox process allow the construction of models that are tractable, but that are no less reasonable explanations for the data than their cousins the logistic Gaussian process and the log Gaussian Cox process.

Inference by Markov chain Monte Carlo of the history of a probabilistic computational procedure has been studied previously. Beskos et al. (2006b) sampled from the state of a rejection sampler for diffusions. Murray (2007), who coined the phrase "latent history," modeled data as having been the result of a Markov chain which had provably mixed via coupling from the past (Propp and Wilson, 1996). Another example is Huber and Wolpert (2009), who model the history of the Matérn Type III process to perform tractable inference. The Church programming language (Goodman et al., 2008) also exploits this idea, by treating probabilistic procedures as first class objects on which inference can be performed.

5.2 Latent History Inference in the GPDS

To perform inference in the Gaussian process density sampler via the latent history method, I model the N data $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^N$ as having been generated exactly as in Algorithm 2.2. The unknown quantities in this model are 1) the number of rejected proposals, which I denote as M; 2) the locations of the rejected proposals, which I denote as $\mathcal{M} = \{\boldsymbol{x}_m\}_{m=1}^M$; 3) the values of the function at the data, denoted $\mathcal{G}_N = \{g(\boldsymbol{x}_n)\}_{n=1}^N$, and at the rejections, denoted $\mathcal{G}_M = \{g(\boldsymbol{x}_m)\}_{m=1}^M$. The joint distribution over the data and the ordered history of the GPDS generative procedure, given the hyperparameters, is

$$p(\mathcal{D}, \mathcal{G}_N, \mathcal{M}, \mathcal{G}_M \mid \boldsymbol{\theta}, \boldsymbol{\psi}_\pi) = \mathcal{GP}(\mathcal{G}_N, \mathcal{G}_M \mid \mathcal{D}, \mathcal{M}, \boldsymbol{\theta}) \times \left[\prod_{n=1}^N \Phi(g(\boldsymbol{x}_n))\, \pi(\boldsymbol{x}_n \mid \boldsymbol{\psi}_\pi)\right] \prod_{n=1}^M (1 - \Phi(g(\boldsymbol{x}_m)))\, \pi(\boldsymbol{x}_m \mid \boldsymbol{\psi}_\pi). \quad (5.1)$$

To sample from this joint distribution, I suggest a Gibbs-like alternation between 1) modification of the number of rejections M; 2) updating the rejection locations $\mathcal{M}$; 3) modification of the latent function values $\mathcal{G}_M$ and $\mathcal{G}_N$. When discussing the latent history, I imply that an ordering is maintained over the sequence of acceptances and rejections in $\mathcal{D}$ and $\mathcal{M}$. Due to exchangeability, however, it is not necessary to actually maintain this ordering. At any time, a reshuffling of the latent history could be proposed, subject to it ending in an acceptance, and this proposal would always be accepted, as the two permutations have the same probability under the model. When thinking about the computation, it is often still useful to imagine an actual ordering, but it will turn out that no computations actually depend on the ordering.

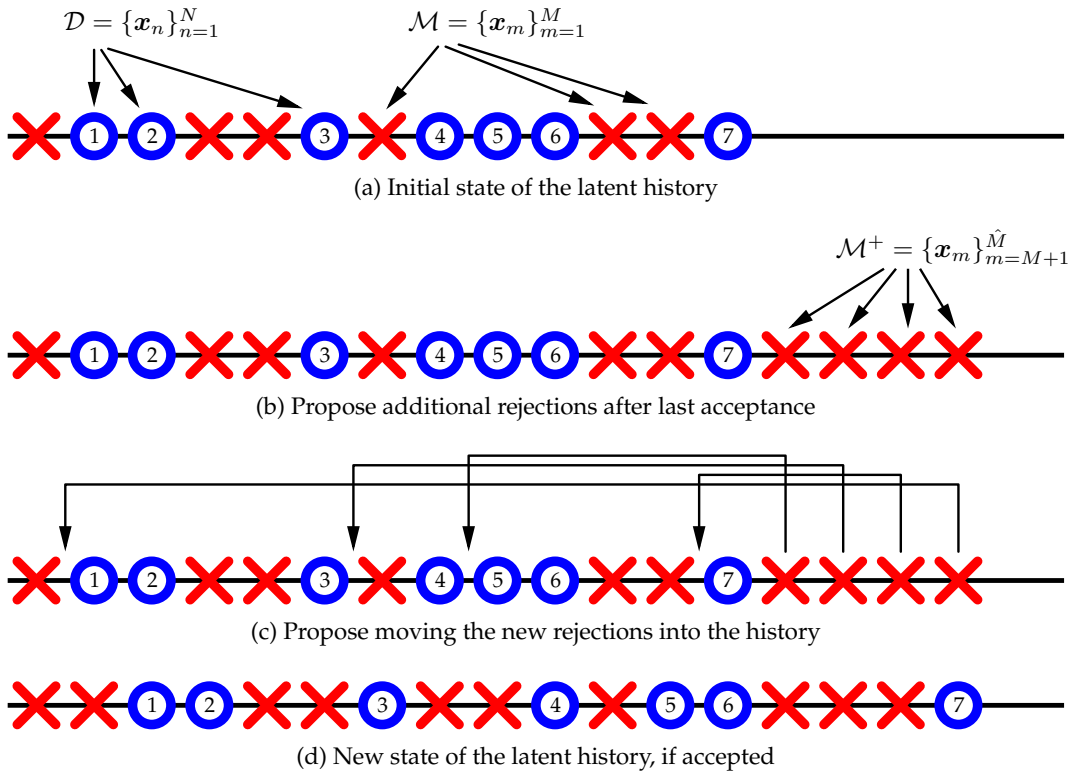


Figure 5.1: A cartoon of how new rejections are inserted into the latent history. There are seven data. (a) There are six latent rejections in the initial state. (b) Four additional rejections are proposed after the seventh accepted datum. Each of these has associated with it a function value drawn from the Gaussian process. (c) Locations for the new rejections in the latent history are proposed. (d) If accepted, there is a new history, now with ten latent rejections.

5.2.1 Sampling the Number of Latent Rejections

Propose a new number of latent rejections $\hat{M}$ by drawing it from a proposal density $q(\hat{M} \leftarrow M)$. If $\hat{M}$ is greater than M, it is also necessary to propose the new rejection locations and function values that will be added to the latent state. We can take advantage of the exchangeability of the process to generate the new rejections: we imagine these proposals were made *after* the last observed datum was accepted, and the proposal is to label them rejections and move them *before* the last datum. This idea is shown graphically in Figure 5.1. If $\hat{M}$ is less than M, do the opposite by proposing to move some rejections to after the last acceptance.

When proposing additional rejections, it is also necessary to propose times for them among the current latent history. There are $\binom{\hat{M}+N-1}{\hat{M}-M}$ such ways to insert these additional rejections into the existing latent history, such that the sampler terminates after the Nth acceptance. When removing rejections, there are $\binom{M}{M-\hat{M}}$ possible sets that can be moved to after the last acceptance. As mentioned before, none of the computation depends directly upon the ordering, due to exchangeability. It is, however, still necessary to account for the implicit ordering when constructing the proposals.

Upon simplification, the proposal ratios for both addition and removal of rejections are identical:

$$\overbrace{\frac{q(M \leftarrow \hat{M})\binom{\hat{M}+N-1}{\hat{M}-M}}{q(\hat{M} \leftarrow M)\binom{\hat{M}}{\hat{M}-M}}}^{\hat{M}>M} = \overbrace{\frac{q(M \leftarrow \hat{M})\binom{M}{M-\hat{M}}}{q(\hat{M} \leftarrow M)\binom{M+N-1}{M-\hat{M}}}}^{\hat{M}<M} = \frac{q(M \leftarrow \hat{M})M!(\hat{M}+N-1)!}{q(\hat{M} \leftarrow M)\hat{M}!(M+N-1)!}. \tag{5.2}$$

When inserting rejections, propose the locations of the additional proposals, denoted $\mathcal{M}^+$, and the corresponding values of the latent function, denoted $\mathcal{G}_M^+$. These $\mathcal{M}^+$ are generated by making $\hat{M} - M$ independent draws from the base density. The $\mathcal{G}_M^+$ are then drawn jointly from the Gaussian process prior, conditioned on all of the current latent state, i.e. $\mathcal{G}_M^+ \sim \mathcal{GP}(\mathcal{M}^+, \mathcal{M}, \mathcal{G}_M, \mathcal{D}, \mathcal{G}_N, \boldsymbol{\theta})$. This is the intermediate state shown in Figure 5.1b (although the function draws are not illustrated), with joint probability

$$\begin{aligned} p(\mathcal{D}, \mathcal{M}, \mathcal{M}^+, \mathcal{G}_N, \mathcal{G}_M, \mathcal{G}_M^+ \,|\, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi) &= \mathcal{GP}(\mathcal{G}_M, \mathcal{G}_N, \mathcal{G}_M^+ \,|\, \mathcal{D}, \mathcal{M}, \mathcal{M}^+, \boldsymbol{\theta}) \\ &\times \left[\prod_{n=1}^{N} \Phi(g(\boldsymbol{x}_n))\ \pi(\boldsymbol{x}_n \,|\, \boldsymbol{\psi}_\pi)\right] \left[\prod_{m=1}^{M} (1 - \Phi(g(\boldsymbol{x}_m)))\ \pi(\boldsymbol{x}_m \,|\, \boldsymbol{\psi}_\pi)\right] \prod_{m=M+1}^{\hat{M}} \pi(\boldsymbol{x}_m \,|\, \boldsymbol{\psi}_\pi). \end{aligned} \tag{5.3}$$

The joint distribution in Equation 5.3 expresses the probability of all the base density draws, the values of the function draws from the Gaussian process, the acceptance probabilities of the data, and the rejection probabilities of the latent rejections, *before* labeling the new proposals $\mathcal{M}^+$ as rejections. When an insertion proposal is made, exchangeability allows the ordering to be shuffled without changing the probability; the only change is that it is necessary now to account for labeling the new points as rejections. In the acceptance ratio, all terms except for this "labeling probability" cancel. The reverse proposal is similar, however, I denote the removed proposal locations as $\mathcal{M}^-$ and the corresponding function values as $\mathcal{G}_M^-$. The overall acceptance ratios for insertions or removals are

$$a_{\mathsf{gpds-num}} = \begin{cases} \dfrac{q(M \leftarrow \hat{M})\ M!\ (\hat{M}+N-1)!}{q(\hat{M} \leftarrow M)\ \hat{M}!\ (M+N-1)!} \displaystyle\prod_{\boldsymbol{x} \in \mathcal{M}^+} (1 - \Phi(g(\boldsymbol{x}))) & \text{if } \hat{M} > M \\ \dfrac{q(M \leftarrow \hat{M})\ M!\ (\hat{M}+N-1)!}{q(\hat{M} \leftarrow M)\ \hat{M}!\ (M+N-1)!} \displaystyle\prod_{\boldsymbol{x} \in \mathcal{M}^-} (1 - \Phi(g(\boldsymbol{x})))^{-1} & \text{if } \hat{M} < M. \end{cases} \tag{5.4}$$

A simple and convenient way of implementing this procedure is to make limited proposals that either insert or delete only one latent rejection at a time. In practice, I have found this to work well. I define a function $\zeta(M, N) : \mathbb{N} \times \mathbb{N}^+ \to (0, 1]$ that is the Bernoulli probability of making a proposal to insert a new latent rejection. It is, of

course, necessary that $\zeta(0, N) = 1$. With this limited proposal, the first case of Equation 5.4 (proposing one new latent rejection, i.e. $\hat{M} = M + 1$) can be written as

$$a_{\mathsf{gpds-ins}} = \frac{(1 - \zeta(M+1, N))\ (M+N)\ (1 - \Phi(g(\boldsymbol{x}^+)))}{\zeta(M, N)\ (M+1)}, \tag{5.5}$$

where $\boldsymbol{x}^+$ is the proposed rejection location. The location $\boldsymbol{x}^+$ is drawn from the base density $\pi(\boldsymbol{x} \,|\, \boldsymbol{\psi}_\pi)$. In the second case, if there is at least one latent rejection in the current history ($M > 0$), then the deletion of a single rejection is proposed, i.e. $\hat{M} = M - 1$. This deletion proposal has Metropolis–Hastings acceptance ratio

$$a_{\mathsf{gpds-del}} = \frac{\zeta(M-1, N)\ M}{(1 - \zeta(M, N))\ (M+N-1)\ (1 - \Phi(g(\boldsymbol{x}^-)))}, \tag{5.6}$$

where $\boldsymbol{x}^-$ is the location of the proposed removal. The rejection to remove is chosen uniformly from among the M currently in the history. In my experience applying the latent history method to the examples in Chapter 6, it appears that the specific form of $\zeta(M, N)$ does not significantly impact performance and I have had success with $\zeta(M, N) = \frac{1}{2}$. It has, however, often been beneficial to make several (≈ 10) of these proposals for each of the other proposals mentioned in this section.

5.2.2 Sampling the Locations of Latent Rejections

This section discusses Markov chain Monte Carlo transitions on the locations of the latent rejections, denoted $\mathcal{M} = \{\boldsymbol{x}_m\}_{m=1}^M$. When making these moves, we condition on the rest of the latent history: the number of rejections M and the latent function $g(\boldsymbol{x})$. Given the latent function, the locations of the rejections are independent. Iterate over each of the M rejections and sample its location using Metropolis–Hastings. For the mth rejection $\boldsymbol{x}_m$, first propose a new location $\hat{\boldsymbol{x}}_m$ from a distribution $q(\hat{\boldsymbol{x}}_m \leftarrow \boldsymbol{x}_m)$. Then sample the function at this location to find $g(\hat{\boldsymbol{x}}_m)$. Draw this value from the Gaussian process, conditioning on the rest of the state, i.e. $g(\hat{\boldsymbol{x}}_m) \sim \mathcal{GP}(\hat{\boldsymbol{x}}_m, \mathcal{D}, \mathcal{G}_N, \mathcal{M}, \mathcal{G}_M, \boldsymbol{\theta})$. Reject or accept this proposal according to MH acceptance ratio

$$a_{\mathsf{gpds-loc}} = \frac{q(\boldsymbol{x}_m \leftarrow \hat{\boldsymbol{x}}_m)\ (1 - \Phi(g(\hat{\boldsymbol{x}}_m)))\ \pi(\hat{\boldsymbol{x}}_m \,|\, \boldsymbol{\psi}_\pi)}{q(\hat{\boldsymbol{x}}_m \leftarrow \boldsymbol{x}_m)\ (1 - \Phi(g(\boldsymbol{x}_m)))\ \pi(\boldsymbol{x}_m \,|\, \boldsymbol{\psi}_\pi)}. \tag{5.7}$$

These moves could also be done with an aggregate proposal for all of M latent rejection locations at once, via $q(\hat{\mathcal{M}} \leftarrow \mathcal{M})$, where $\hat{\mathcal{M}} = \{\hat{\boldsymbol{x}}_m\}_{m=1}^M$. This Metropolis–Hastings proposal has acceptance ratio

$$a_{\mathsf{gpds-locs}} = \frac{q(\mathcal{M} \leftarrow \hat{\mathcal{M}})}{q(\hat{\mathcal{M}} \leftarrow \mathcal{M})} \prod_{m=1}^{M} \frac{(1 - \Phi(\hat{g}(\boldsymbol{x}_m)))\ \pi(\hat{\boldsymbol{x}}_m \,|\, \boldsymbol{\psi}_\pi)}{(1 - \Phi(g(\boldsymbol{x}_m)))\ \pi(\boldsymbol{x}_m \,|\, \boldsymbol{\psi}_\pi)}. \tag{5.8}$$

As in most implementations of Metropolis–Hastings, choosing an appropriate proposal distribution $q(\hat{\boldsymbol{x}}_m \leftarrow \boldsymbol{x}_m)$ is important to efficiency. For perturbative proposals, this corresponds to choosing a reasonable "step size" for proposed moves, e.g. the σ of a Gaussian proposal. In the Gaussian process density sampler, when using a covariance function such as the squared-exponential, it is possible to select this step size somewhat automatically, based on the length scale ℓ of the GP, e.g. $\sigma = c \cdot \ell$, for some "hyper-step-size" c. In Section 5.2.4, I discuss sampling from the hyperparameters of the Gaussian process. Adapting the length scale of a stationary covariance function is equivalent to inferring an appropriate distance metric for the data space. It is useful to take advantage of this information to improve mixing of the rejection locations. Each time the length scale is updated, the step size of $q(\hat{\boldsymbol{x}}_m \leftarrow \boldsymbol{x}_m)$ can also be updated. This adaptation is Markovian, so it does not sacrifice the validity of the Markov chain's equilibrium distribution.

5.2.3 Sampling the Latent Function

Conditioned on the number M and locations $\mathcal{M}$ of the latent rejections, it is also necessary to sample from the latent function at both the data and rejection locations, denoted $\mathcal{G}_N$ and $\mathcal{G}_M$, respectively. The conditional joint posterior distribution is

$$p(\mathcal{G}_N, \mathcal{G}_M \,|\, \mathcal{M}, \mathcal{D}, \boldsymbol{\theta}) \propto \mathcal{GP}(\mathcal{G}_N, \mathcal{G}_M \,|\, \mathcal{D}, \mathcal{M}, \boldsymbol{\theta}) \times \left[\prod_{n=1}^{N} \Phi(g(\boldsymbol{x}_n))\right] \prod_{m=1}^{M} \left(1 - \Phi(g(\boldsymbol{x}_m))\right). \quad (5.9)$$

In Section 4.3.2, I discussed a complex underrelaxation scheme, based on control points, to make effective proposals for the function $g(\boldsymbol{x})$ in the Gaussian process density sampler. In the latent history approach to inference, however, it is not necessary to use control points and underrelaxation to achieve efficient sampling. Instead, we can sample from Equation 5.9 using *Hamiltonian Monte Carlo* (HMC).

Hamiltonian (or *hybrid*) Monte Carlo (Duane et al., 1987) is a Metropolis–Hastings sampling method that takes advantage of gradient information to make efficient proposals and reduce random walk behaviour. The idea is to augment the state of the Markov chain with random "momentum" variables and then simulate the resulting Hamiltonian system in fictitious time using, for example, Euler integration. The *leapfrog* method described in Neal (1996), Rasmussen (1996) and MacKay (2003, Chapter 30) is easy to implement and is effective for simulating these dynamics.

For numerical reasons, when sampling from a function with a Gaussian process prior, as in Equation 5.9, it is useful to perform gradient calculations in the "whitened" space resulting from applying to the function values the inverse Cholesky decomposition of the covariance matrix. That is, if $\boldsymbol{C}$ is the positive-definite matrix result-

ing from applying the covariance function to the union of data $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^N$ and latent rejections $\mathcal{M} = \{\boldsymbol{x}_m\}_{m=1}^M$, and $\boldsymbol{g}_{N,M} = \mathsf{vec}(\mathcal{G}_N \cup \mathcal{G}_M)$ is the vector of function values at those locations, then first transform $\boldsymbol{g}_{N,M}$ with the matrix $\boldsymbol{L}^{-1}$, where $\boldsymbol{L}$ is the Cholesky decomposition of $\boldsymbol{C}$, i.e. $\boldsymbol{C} = \boldsymbol{L}\boldsymbol{L}^\mathsf{T}$.

Algorithm 5.1 implements the latent history algorithm in pseudocode, using a simple $q(\hat{M} \leftarrow M)$ that proposes increasing or decreasing the number of latent rejections M by one, with probabilities $\zeta(M, N)$ and $1 - \zeta(M, N)$, respectively.

5.2.4 Sampling the Gaussian Process Hyperparameters

The joint distribution in Equation 5.1 depends only on the Gaussian process hyperparameters $\boldsymbol{\theta}$ through the GP prior term. Conditioning on the latent rejections and the data, as well as the current function values, it is possible to sample from the hyperparameters using the Gaussian process marginal likelihood, as in Equation 1.3. As suggested by Williams and Rasmussen (1996) and Neal (1998), Hamiltonian Monte Carlo is a useful method for efficient sampling from Gaussian process hyperparameters due to the frequent availability of analytic gradients. See Rasmussen (1996, Chapter 4) for a more comprehensive discussion of this topic.

5.2.5 Sampling the Base Density Hyperparameters

As with exchange sampling in Section 4.3.3, it is appealing to sample from the hyperparameters $\boldsymbol{\psi}_\pi$ governing the base density $\pi(\boldsymbol{x} \mid \boldsymbol{\psi}_\pi)$. Conditioned on the number of rejections M, the rejection locations $\mathcal{M}$, and the function values $\mathcal{G}_N$ and $\mathcal{G}_M$, the joint distribution in Equation 5.1 depends on $\boldsymbol{\psi}_\pi$ only through the products of base densities. This makes intuitive sense, as the generative model is that the union of the data and rejections are i.i.d. from the base density. Therefore, samples from the base density hyperparameters can be drawn using any convenient Markov chain Monte Carlo method. Conditioned on all other aspects of the latent history, $\pi(\boldsymbol{x} \mid \boldsymbol{\psi}_\pi)$ can be treated as its own model, with $\mathcal{D} \cup \mathcal{M}$ being the observed data. For example, with a simple exponential-family base density, it is convenient to select a conjugate hyperprior. The hyperparameters $\boldsymbol{\psi}_\pi$ can then be updated using Gibbs sampling, as in Gelman et al. (2004), for example.

5.2.6 Generating Predictive Samples

As discussed in Section 4.5, the predictive distribution plays an important role in Bayesian modeling. To generate a predictive sample from the Gaussian process density sampler under the latent history scheme, take the current latent history and run Algorithm 2.2 further until another (the $N+1$th) datum has been accepted. If this is

Algorithm 5.1 Generate R samples from the latent history of the GPDS

Inputs:
- Number of MCMC iterations R
- Observed data $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^N$
- Gaussian process covariance function $C(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta})$
- Base density $\pi(\boldsymbol{x} \mid \boldsymbol{\psi}_\pi)$
- Location proposal density $q(\hat{\boldsymbol{x}}_m \leftarrow \boldsymbol{x}_m)$
- Insert proposal probability function $\zeta(M, N)$

Outputs:
- R samples of the latent history $\{\mathcal{M}^{(r)}, \mathcal{G}_N^{(r)}, \mathcal{G}_M^{(r)}\}_{r=1}^R$

$\mathcal{M} \leftarrow \emptyset$, $\mathcal{G}_M \leftarrow \emptyset$ ▷ Start out with no latent rejections.
$\mathcal{G}_N \sim \mathcal{GP}(g \mid \mathcal{D}, \boldsymbol{\theta})$ ▷ Initialise the function at the data.
for $r \leftarrow 1 \ldots R$ **do** ▷ Take R MCMC steps on the latent history.
  $u_\zeta \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on $(0, 1)$.
  if $u_\zeta < \zeta(|\mathcal{M}|, N)$ **then** ▷ Decide whether to insert or delete.
    $\boldsymbol{x}^+ \sim \pi(\boldsymbol{x} \mid \psi_\pi)$ ▷ Draw a proposed rejection location.
    $g(\boldsymbol{x}^+) \sim \mathcal{GP}(g \mid \boldsymbol{x}^+, \mathcal{D}, \mathcal{M}, \mathcal{G}_M, \mathcal{G}_N, \boldsymbol{\theta})$ ▷ Draw the proposed function value.
    $a_{\mathsf{gpds-ins}} \leftarrow \dfrac{(1 - \zeta(|\mathcal{M}| + 1, N))\ (|\mathcal{M}| + N)\ (1 - \Phi(g(\boldsymbol{x}^+)))}{\zeta(|\mathcal{M}|, N)\ (|\mathcal{M}| + 1)}$ ▷ Acceptance ratio.
    $u_{\mathsf{ins}} \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on $(0, 1)$.
    if $u_{\mathsf{ins}} < a_{\mathsf{gpds-ins}}$ **then** ▷ Metropolis–Hastings acceptance rule.
      $\mathcal{M} \leftarrow \mathcal{M} \cup \boldsymbol{x}^+$, $\mathcal{G}_M \leftarrow \mathcal{G}_M \cup g(\boldsymbol{x}^+)$ ▷ Add this new rejection.
    end if
  else if $|\mathcal{M}| > 0$ **then**
    $m \sim \lceil \mathcal{U}(0, |\mathcal{M}|) \rceil$ ▷ Select one of the M rejections at random.
    $a_{\mathsf{gpds-del}} = \dfrac{\zeta(|\mathcal{M}| - 1, N)\ |\mathcal{M}|}{(1 - \zeta(|\mathcal{M}|, N))\ (|\mathcal{M}| + N - 1)\ (1 - \Phi(g(\boldsymbol{x}_m)))}$ ▷ Acceptance ratio.
    $u_{\mathsf{del}} \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on $(0, 1)$.
    if $u_{\mathsf{del}} < a_{\mathsf{gpds-del}}$ **then** ▷ Metropolis–Hastings acceptance rule.
      $\mathcal{M} \leftarrow \mathcal{M} \backslash \boldsymbol{x}_m$, $\mathcal{G}_M \leftarrow \mathcal{G}_M \backslash g(\boldsymbol{x}_m)$ ▷ Remove the mth rejection.
    end if
  end if
  for $m \leftarrow 1 \ldots M$ **do** ▷ Loop over the latent rejections.
    $\hat{\boldsymbol{x}}_m \sim q(\hat{\boldsymbol{x}}_m \leftarrow \boldsymbol{x}_m)$ ▷ Propose a new location.
    $g(\hat{\boldsymbol{x}}_m) \sim \mathcal{GP}(g \mid \hat{\boldsymbol{x}}_m, \mathcal{D}, \mathcal{M}, \mathcal{G}_N, \mathcal{G}_M, \boldsymbol{\theta})$ ▷ Draw a function value from the GP.
    $a_{\mathsf{gpds-loc}} = \dfrac{q(\boldsymbol{x}_m \leftarrow \hat{\boldsymbol{x}}_m)\ \pi(\hat{\boldsymbol{x}}_m)\ (1 - \Phi(g(\hat{\boldsymbol{x}}_m)))}{q(\hat{\boldsymbol{x}}_m \leftarrow \boldsymbol{x}_m)\ \pi(\boldsymbol{x}_m)\ (1 - \Phi(g(\boldsymbol{x}_m)))}$ ▷ Acceptance ratio.
    $u_{\mathsf{loc}} \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate from $(0, 1)$.
    if $u_{\mathsf{loc}} < a_{\mathsf{gpds-loc}}$ **then** ▷ Metropolis–Hastings acceptance rule.
      $\boldsymbol{x}_m \leftarrow \hat{\boldsymbol{x}}_m$, $g(\boldsymbol{x}_m) \leftarrow g(\hat{\boldsymbol{x}}_m)$ ▷ Update the rejection.
    end if
  end for
  $\mathcal{G}_N$,$\mathcal{G}_M \sim \mathsf{HMC}(\mathcal{G}_N, \mathcal{G}_M \mid \mathcal{D}, \mathcal{M}, \boldsymbol{\theta})$ ▷ Update function via Hamiltonian Monte Carlo.
  $\mathcal{M}^{(r)} \leftarrow \mathcal{M}$, $\mathcal{G}_N^{(r)} \leftarrow \mathcal{G}_N$, $\mathcal{G}_M^{(r)} \leftarrow \mathcal{G}_M$ ▷ Store the current estimate of the latent history.
end for
return $\{\mathcal{M}^{(r)}, \mathcal{G}_N^{(r)}, \mathcal{G}_M^{(r)}\}_{r=1}^R$

done for each Markov chain Monte Carlo step in the latent history inference, then the samples kept will be from the predictive distribution. This, in effect, integrates over the posterior distribution on the latent function, and any hyperparameters included in the model.

5.3 Latent History Inference in the SGCP

Inference can be performed in the sigmoidal Gaussian Cox process using the latent history method in a very similar way to the Gaussian process density sampler. Given a set of K observed events $\mathcal{S} = \{\boldsymbol{x}_k\}_{k=1}^K$, the task is to sample from the posterior distribution on latent functions $g(\boldsymbol{x})$ that gave rise to the unknown intensity function $\lambda(\boldsymbol{x})$. Using the SGCP to model the data is equivalent to asserting that Algorithm 3.1 was used to generate the data. The unknowns in this case are 1) the number of events that were thinned, denoted M; 2) the locations of the thinned events, denoted $\mathcal{M} = \{\boldsymbol{x}_m\}_{m=1}^M$; 3) the values of the latent function $g(\boldsymbol{x})$ at the observed events, denoted $\mathcal{G}_K = \{g(\boldsymbol{x}_k)\}_{k=1}^K$, and at the thinned events, denoted $\mathcal{G}_M = \{g(\boldsymbol{x}_m)\}_{m=1}^M$. This set of latent variables is nearly identical to that used to model the latent history of the Gaussian process density sampler. The joint distribution on the history is different, however:

$$p(\mathcal{S}, \mathcal{G}_K, \mathcal{M}, \mathcal{G}_M \,|\, \mathcal{V}, \boldsymbol{\theta}, \boldsymbol{\psi}_\lambda) = \mathcal{GP}(\mathcal{G}_K, \mathcal{G}_M \,|\, \mathcal{S}, \mathcal{M}, \boldsymbol{\theta}) \, \exp\left\{-\bar{\Lambda}_{\mathcal{V}}(\boldsymbol{\psi}_\lambda)\right\} \times \left[\prod_{k=1}^K \bar{\lambda}(\boldsymbol{x}_k \,;\, \boldsymbol{\psi}_\lambda) \, \Phi(g(\boldsymbol{x}_k))\right] \prod_{m=1}^M \bar{\lambda}(\boldsymbol{x}_m \,;\, \boldsymbol{\psi}_\lambda) \, (1 - \Phi(g(\boldsymbol{x}_m))), \quad (5.10)$$

where

$$\bar{\Lambda}_{\mathcal{V}}(\boldsymbol{\psi}_\lambda) = \int_{\mathcal{V}} \mathrm{d}\boldsymbol{x} \, \bar{\lambda}(\boldsymbol{x} \,;\, \boldsymbol{\psi}_\lambda). \quad (5.11)$$

I assume that $\bar{\lambda}(\boldsymbol{x} \,;\, \boldsymbol{\psi}_\lambda)$ has been chosen so that the integral in Equation 5.11 is easy to compute. As in the previous section with the Gaussian process density sampler, this latent history can be sampled using Markov chain Monte Carlo in several stages.

5.3.1 Sampling the Number of Thinned Events

Metropolis–Hastings can be used to sample from the number of thinned events M. Use a function $\zeta(M, K)$, as introduced in Section 5.2.1, to provide a Bernoulli probability of making a proposal to insert a new thinned event into the latent history. An

insertion move consists of proposing a new $\boldsymbol{x}^+$ from the density

$$q_{\boldsymbol{x}^+}(\boldsymbol{x}) = \frac{\bar{\lambda}(\boldsymbol{x}\,;\,\boldsymbol{\psi}_\lambda)}{\bar{\Lambda}_{\mathcal{V}}(\boldsymbol{\psi}_\lambda)} \boldsymbol{I}_{\mathcal{V}}(\boldsymbol{x}), \tag{5.12}$$

followed by a draw from the Gaussian process to find $g(\boldsymbol{x}^+)$, conditioned on $\mathcal{G}_K$ and $\mathcal{G}_K$, i.e. $g(\boldsymbol{x}^+) \sim \mathcal{GP}(\boldsymbol{x}^+, \mathcal{S}, \mathcal{M}, \mathcal{G}_K, \mathcal{G}_M)$. The proposal distribution of this new location $\boldsymbol{x}^+$ and associated function value $g(\boldsymbol{x}^+)$, taken together, is

$$q_{\mathsf{ins}}(\mathcal{M} \cup \boldsymbol{x}^+ \leftarrow \mathcal{M}) = \zeta(M, K)\, \frac{\bar{\lambda}(\boldsymbol{x}^+\,;\,\boldsymbol{\psi}_\lambda)}{\bar{\Lambda}_{\mathcal{V}}(\boldsymbol{\psi}_\lambda)} \boldsymbol{I}_{\mathcal{V}}(\boldsymbol{x}^+)\, \mathcal{GP}(g(\boldsymbol{x}^+) \,|\, \boldsymbol{x}^+, \mathcal{S}, \mathcal{M}, \mathcal{G}_K, \mathcal{G}_M). \tag{5.13}$$

A deletion move is proposed when the Bernoulli coin flip does not indicate an insertion. To delete a latent thinned event that is already in the history, first select the event m uniformly from the M events currently in the state. This proposal distribution is

$$q_{\mathsf{del}}(\mathcal{M} \backslash \boldsymbol{x}_m \leftarrow \mathcal{M}) = \frac{1 - \zeta(M, K)}{M}. \tag{5.14}$$

Incorporating the joint distribution in Equation 5.10, the Metropolis–Hastings acceptance ratios of the two types of proposals are

$$a_{\mathsf{sgcp-ins}} = \frac{(1 - \zeta(M+1, K))\, \bar{\Lambda}_{\mathcal{V}}(\boldsymbol{\psi}_\lambda)\, (1 - \Phi(g(\boldsymbol{x}^+)))}{\zeta(M, K)\, (M+1)} \tag{5.15}$$

$$a_{\mathsf{sgcp-del}} = \frac{\zeta(M-1, K)\, M}{(1 - \zeta(M, K))\, \bar{\Lambda}_{\mathcal{V}}(\boldsymbol{\psi}_\lambda)\, (1 - \Phi(g(\boldsymbol{x}_m)))}. \tag{5.16}$$

5.3.2 Sampling the Locations of Thinned Events

Given the number of thinned events M, sample next from the posterior distribution on the locations of the events, $\mathcal{M} = \{\boldsymbol{x}_m\}_{m=1}^M$. As when updating the rejection locations in the Gaussian process density sampler, Metropolis–Hastings can be used to perform this sampling. Iterate over each of the M thinned events and propose a new location $\hat{\boldsymbol{x}}_m$ via the proposal density $q(\hat{\boldsymbol{x}}_m \leftarrow \boldsymbol{x}_m)$. Then draw a function value $g(\hat{\boldsymbol{x}}_m)$ from the Gaussian process, conditioned on the current state $\mathcal{G}_K$ and $\mathcal{G}_M$, including the mth one. The Metropolis–Hastings acceptance ratio for this proposal is

$$a_{\mathsf{sgcp-loc}} = \frac{q(\boldsymbol{x}_m \leftarrow \hat{\boldsymbol{x}}_m)\, \bar{\lambda}(\hat{\boldsymbol{x}}_m\,;\,\boldsymbol{\psi}_\lambda)\, (1 - \Phi(g(\hat{\boldsymbol{x}}_m)))}{q(\hat{\boldsymbol{x}}_m \leftarrow \boldsymbol{x}_m)\, \bar{\lambda}(\boldsymbol{x}_m\,;\,\boldsymbol{\psi}_\lambda)\, (1 - \Phi(g(\boldsymbol{x}_m)))}. \tag{5.17}$$

Typically, perturbative proposals on the order of the Gaussian process length scale are appropriate for these Metropolis–Hastings steps, as discussed in Section 5.2.2. If the

move is accepted, the old values $\boldsymbol{x}_m$ and $g(\boldsymbol{x}_m)$ can safely be discarded.

5.3.3 Sampling the Latent Function

As in the Gaussian process density sampler, Hamiltonian Monte Carlo (Duane et al., 1987) is useful for inference of the values of the function at the observed events and at the thinned events. HMC enables the use of gradient information to make improved proposals. As discussed previously, it is beneficial to whiten the space for improved numerical conditioning. The conditional distribution on the function values is

$$p(\mathcal{G}_K, \mathcal{G}_M \,|\, \mathcal{V}, \mathcal{S}, \mathcal{M}, \boldsymbol{\theta}) \propto \mathcal{GP}(\mathcal{G}_K, \mathcal{G}_M \,|\, \mathcal{S}, \mathcal{M}, \boldsymbol{\theta}) \times \left[\prod_{k=1}^{K} \Phi(g(\boldsymbol{x}_k))\right] \prod_{m=1}^{M} (1 - \Phi(g(\boldsymbol{x}_m))). \quad (5.18)$$

5.3.4 Sampling the Gaussian Process Hyperparameters

Inference of the Gaussian process hyperparameters can be done easily using the standard Hamiltonian Monte Carlo methods discussed in Section 5.2.4. As in the Gaussian process density sampler, conditioned on the latent history, the Gaussian process marginal likelihood allows convenient sampling from GP hyperparameters.

5.3.5 Sampling the Dominating Intensity Hyperparameters

Sampling from the dominating intensity hyperparameters $\boldsymbol{\psi}_\lambda$ helps ensure that $\bar{\lambda}(\boldsymbol{x}\,;\,\boldsymbol{\psi}_\lambda)$ is not limiting the maximum intensity of the Poisson model artificially. Alternatively, adapting the dominating intensity can prevent an excessive number of thinned events being required for the model, if $\bar{\lambda}(\boldsymbol{x}\,;\,\boldsymbol{\psi}_\lambda)$ is providing only a loose bound on the true intensity $\lambda(\boldsymbol{x})$. The conditional posterior distribution over the dominating intensity hyperparameters, writing out the integral for $\bar{\Lambda}_{\mathcal{V}}(\boldsymbol{\psi}_\lambda)$, is

$$p(\boldsymbol{\psi}_\lambda \,|\, \mathcal{V}, \mathcal{S}, \mathcal{M}) \propto \exp\left\{-\int_{\mathcal{V}} \mathrm{d}\boldsymbol{x}'\; \bar{\lambda}(\boldsymbol{x}'\,;\,\boldsymbol{\psi}_\lambda)\right\} \left[\prod_{k=1}^{K} \bar{\lambda}(\boldsymbol{x}_k\,;\,\boldsymbol{\psi}_\lambda)\right] \prod_{m=1}^{M} \bar{\lambda}(\boldsymbol{x}_m\,;\,\boldsymbol{\psi}_\lambda). \quad (5.19)$$

Sampling from this conditional posterior via Markov chain Monte Carlo is exactly as it would be if this was a simple parametric Poisson model. One convenient form of the dominating intensity is to simply make it constant, i.e. $\bar{\lambda}(\boldsymbol{x}\,;\,\boldsymbol{\psi}_\lambda) = \lambda_\star$, so that $\boldsymbol{\psi}_\lambda = \lambda_\star$ and $\bar{\Lambda}_{\mathcal{V}}(\boldsymbol{\psi}_\lambda) = \lambda_\star \mu(\mathcal{V})$. In this case, the gamma distribution is a conditionally-conjugate prior for $\lambda_\star$. If the gamma prior parameters are α_0 and β_0,

Algorithm 5.2 Generate R samples from the latent history of the SGCP

Inputs:
- Number of MCMC iterations R
- Observed events $\mathcal{S} = \{\boldsymbol{x}_k\}_{k=1}^K$
- Gaussian process covariance function $C(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta})$
- Dominating intensity gamma prior parameters α_0 and β_0
- Location proposal density $q(\hat{\boldsymbol{x}}_m \leftarrow \boldsymbol{x}_m)$
- Insert proposal probability function $\zeta(M, K)$

Outputs:
- R samples of the latent history $\{\mathcal{M}^{(r)}, \mathcal{G}_K^{(r)}, \mathcal{G}_M^{(r)}\}_{r=1}^R$
- R samples of the dominating intensity $\{\lambda_\star^{(r)}\}_{r=1}^R$

$\mathcal{M} \leftarrow \emptyset$, $\mathcal{G}_M \leftarrow \emptyset$ ▷ Start out with no thinned events.
$\mathcal{G}_K \sim \mathcal{GP}(g \mid \mathcal{S}, \boldsymbol{\theta})$ ▷ Initialise the function at the data.
$\lambda_\star \sim \mathcal{GA}(\alpha_0, \beta_0)$ ▷ Draw the initial dominating intensity.
for $r \leftarrow 1 \dots R$ **do** ▷ Take R MCMC steps on the latent history.
  $u_\zeta \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on $(0, 1)$.
  if $u_\zeta < \zeta(|\mathcal{M}|, K)$ **then** ▷ Decide whether to insert or delete.
    $\boldsymbol{x}^+ \sim \mathcal{U}(\mathcal{V})$ ▷ Draw a new location uniformly in $\mathcal{V}$.
    $g(\boldsymbol{x}^+) \sim \mathcal{GP}(g \mid \boldsymbol{x}^+, \mathcal{S}, \mathcal{M}, \mathcal{G}_M, \mathcal{G}_K, \boldsymbol{\theta})$ ▷ Draw the proposed function value.
    $a_{\mathsf{sgcp-ins}} \leftarrow \dfrac{(1 - \zeta(|\mathcal{M}| + 1, K))\ \lambda_\star \mu(\mathcal{V})\ (1 - \Phi(g(\boldsymbol{x}^+)))}{\zeta(|\mathcal{M}|, K)\ (|\mathcal{M}| + 1)}$ ▷ Acceptance ratio.
    $u_{\mathsf{ins}} \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on $(0, 1)$.
    if $u_{\mathsf{ins}} < a_{\mathsf{sgcp-ins}}$ **then** ▷ Metropolis–Hastings acceptance rule.
      $\mathcal{M} \leftarrow \mathcal{M} \cup \boldsymbol{x}^+$, $\mathcal{G}_M \leftarrow \mathcal{G}_M \cup g(\boldsymbol{x}^+)$ ▷ Add this new rejection.
    end if
  else
    $m \sim \lceil \mathcal{U}(0, |\mathcal{M}|) \rceil$ ▷ Select one of the M thinned events at random.
    $a_{\mathsf{sgcp-del}} = \dfrac{\zeta(|\mathcal{M}| - 1, K)\ |\mathcal{M}|}{(1 - \zeta(|\mathcal{M}|, K))\ \lambda_\star \mu(\mathcal{V})\ (1 - \Phi(g(\boldsymbol{x}_m)))}$ ▷ Acceptance ratio.
    $u_{\mathsf{del}} \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on $(0, 1)$.
    if $u_{\mathsf{del}} < a_{\mathsf{sgcp-del}}$ **then** ▷ Metropolis–Hastings acceptance rule.
      $\mathcal{M} \leftarrow \mathcal{M} \backslash \boldsymbol{x}_m$, $\mathcal{G}_M \leftarrow \mathcal{G}_M \backslash g(\boldsymbol{x}_m)$ ▷ Remove the mth thinned event.
    end if
  end if
  for $m \leftarrow 1 \dots M$ **do** ▷ Loop over the thinned events.
    $\hat{\boldsymbol{x}}_m \sim q(\hat{\boldsymbol{x}}_m \leftarrow \boldsymbol{x}_m)$ ▷ Propose a new location.
    $g(\hat{\boldsymbol{x}}_m) \sim \mathcal{GP}(g \mid \hat{\boldsymbol{x}}_m, \mathcal{S}, \mathcal{M}, \mathcal{G}_K, \mathcal{G}_M, \boldsymbol{\theta})$ ▷ Draw a function value from the GP.
    $a_{\mathsf{sgcp-loc}} = \dfrac{q(\boldsymbol{x}_m \leftarrow \hat{\boldsymbol{x}}_m)\ (1 - \Phi(g(\hat{\boldsymbol{x}}_m)))}{q(\hat{\boldsymbol{x}}_m \leftarrow \boldsymbol{x}_m)\ (1 - \Phi(g(\boldsymbol{x}_m)))}$ ▷ Acceptance ratio.
    $u_{\mathsf{loc}} \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on $(0, 1)$.
    if $u_{\mathsf{loc}} < a_{\mathsf{sgcp-loc}}$ **then** ▷ Metropolis–Hastings acceptance rule.
      $\boldsymbol{x}_m \leftarrow \hat{\boldsymbol{x}}_m$, $g(\boldsymbol{x}_m) \leftarrow g(\hat{\boldsymbol{x}}_m)$ ▷ Update the thinned event location.
    end if
  end for
  $\mathcal{G}_K$,$\mathcal{G}_M \sim \mathsf{HMC}(\mathcal{G}_K, \mathcal{G}_M \mid \mathcal{S}, \mathcal{M}, \boldsymbol{\theta})$ ▷ Update function via Hamiltonian Monte Carlo.
  $\alpha \leftarrow \alpha_0 + |\mathcal{M}| + K$, $\beta \leftarrow \beta_0 + \mu(\mathcal{V})$ ▷ Posterior parameters for the dominating intensity.
  $\lambda_\star \sim \mathcal{GA}(\alpha, \beta)$ ▷ Resample the dominating intensity.
  $\mathcal{M}^{(r)} \leftarrow \mathcal{M}$, $\mathcal{G}_K^{(r)} \leftarrow \mathcal{G}_K$, $\mathcal{G}_M^{(r)} \leftarrow \mathcal{G}_M$ ▷ Store the current estimate of the latent history.
  $\lambda_\star^{(r)} \leftarrow \lambda_\star$ ▷ Store the current estimate of the dominating intensity.
end for
return $\{\mathcal{M}^{(r)}, \mathcal{G}_N^{(r)}, \mathcal{G}_M^{(r)}, \lambda_\star^{(r)}\}_{r=1}^R$

then $\lambda_\star$ can be resampled from a gamma distribution with parameters

$$\alpha_{\mathsf{post}} = \alpha_0 + K + M \tag{5.20}$$

$$\beta_{\mathsf{post}} = \beta_0 + \mu(\mathcal{V}). \tag{5.21}$$

Algorithm 5.2 shows in pseudocode the algorithm for a constant-rate dominating intensity function. It also includes inference of the bounding intensity with a gamma prior, as above.

5.4 Latent History Inference Versus Exchange Sampling

Having now examined the latent history method, how does it compare with the exchange sampling method of Chapter 4? Modeling of densities, whether PDFs or Poisson intensities, is fundamentally different from regression. In regression and classification, one conditions on having seen data in the input space when performing inference and prediction. In these cases, it is necessary only to model the function at places where data have been observed, or at predictive query locations. In density modeling, however, the places with low density are just as important to the model as those with high density. Unfortunately, it is unlikely to have observed data in regions with low density, so a representation of the function only at locations where there are data is not adequate for the inference we wish to perform. One might think of defining a density as analogous to putting up a tent: pinning the canvas down with pegs (or stakes) is just as important as putting up poles. In exchange sampling, the "pegs" are inferred implicitly as rejections along the way to generating fantasy data. At each exchange sampling step, a new tent is constructed — complete with its own pegs — and asked to explain the data. In the latent history model, however, the tent is modified one piece at a time: pegs and poles are inserted, removed, and adjusted gradually to explain the data.

It is possible also to see that the latent history model is likely to require fewer samples from the Gaussian process as it proceeds. Consider the Gaussian process density sampler: when the latent history method is at equilibrium, its state will have some typical number of latent rejections M. This is about the same number of rejections as would be expected to occur during an exchange sampling fantasy. However, to find the acceptance ratio in exchange sampling it is *also* necessary to evaluate against the observed data after fantasising. This means that the Gaussian process in exchange sampling requires at least $2N + M$ evaluations to make a Metropolis–Hastings move, while the latent history method requires only $N + M$. This does not even consider the expansion of state that occurs when exchange sampling rejects a proposal, and additional fantasy data are incorporated into the Markov state. As the time complexity of computation in the Gaussian process grows cubically in the number of data, exchange

sampling can become rapidly more expensive.

Another reason that the latent history method is preferable to exchange sampling is that it requires less bookkeeping about the function $g(\boldsymbol{x})$. The state of the exchange sampling Markov chain is the uncountably-infinite object $g(\boldsymbol{x})$. The innovation of the method is that through retrospective sampling we are able to make Metropolis–Hastings moves with only a finite number of computations. This retrospective sampling, however, means that information discovered about a particular $g(\boldsymbol{x})$ must be retained for as long as that function is relevant to the current Markov state. In contrast, the state of the Markov chain when performing latent history inference only includes $g(\boldsymbol{x})$ at the latent rejections or thinned events. That is, rather than an uncountably-infinite object $g(\boldsymbol{x})$, the Gaussian process in the latent history model conditions on a finite set of points in the input space. This means that the values of the function do not need to be kept in memory, except for at the data and at the locations of the rejections or thinned events. This contrast can also be seen in the difference between the joint distributions that I use to describe the two inference methods for the Gaussian process density sampler. In exchange sampling, when writing Equation 4.20, I use $\boldsymbol{g}$ to denote $g(\boldsymbol{x})$ as an infinite vector. When writing the posterior distribution on the latent history, however, I do not need to denote an infinite function. Equation 5.1 only defines a distribution on the function values at the data and the latent rejections.

Finally, while the latent history method enables efficient Hamiltonian Monte Carlo sampling of the latent function values, it is not clear how to combine HMC with exchange sampling. The underrelaxed proposal methods can be interpreted as a one-step Hamiltonian simulation on the Gaussian process prior, but this effectively removes the likelihood term from the gradient calculation. If there are enough data for the posterior to be significantly more peaked than the prior, then only small proposal steps have a good probability of being accepted.

5.5 Inference in Poisson-Derived Interacting Point Processes

In Section 3.5, I discussed several ways to extend the Poisson process to construct point processes with interaction. I showed that when these clustering or repulsive processes are generative, the sigmoidal Gaussian Cox process provides a means to introduce nonparametric inhomogeneity. In this section I look at how the latent history inference method can be extended to two of these cases.

5.5.1 Inference in the Neyman–Scott Process

The Neyman–Scott process, discussed in Section 3.5.2, is a way to generate events that tend to be closer to each other than would be expected from a Poisson process.

It consists of a mother Poisson process, whose realisations are not observed, but are used to place localised independent daughter Poisson processes. I used the sigmoidal Gaussian Cox process in place of a homogeneous Poisson mother process, to allow for variation in the placement of the clusters. The data, as before, are K events in $\mathcal{V}$, denoted $\mathcal{S} = \{\boldsymbol{x}_k\}_{k=1}^{K}$. I am considering the daughter processes to have local intensity functions $\lambda^{\mathsf{dtr}}(\boldsymbol{x}\,;\,\boldsymbol{x}^{\mathsf{mtr}})$ with compact support $\mathcal{Q}$. The objective is to infer the intensity of the mother process $\lambda^{\mathsf{mtr}}(\boldsymbol{x})$, on which I have placed an SGCP prior. It is also desirable to know the number of mother points, which I will denote as J, and their locations, which I denote as $\mathcal{S}^{\mathsf{mtr}} = \{\boldsymbol{x}_j^{\mathsf{mtr}}\}_{j=1}^{J}$. Due to the superposition property of Poisson processes, conditioned on the mother points, the daughter points are from a Poisson process with intensity $\lambda^{\Sigma}(\boldsymbol{x})$, given by

$$\lambda^{\Sigma}(\boldsymbol{x}) = \sum_{j=1}^{J} \lambda^{\mathsf{dtr}}(\boldsymbol{x}\,;\,\boldsymbol{x}_j^{\mathsf{mtr}}). \tag{5.22}$$

I use the notation $\mathsf{SGCP}(\mathcal{S} \mid \mathcal{V}, \boldsymbol{\theta}, \boldsymbol{\psi}_\lambda)$ to indicate an SGCP prior on a set of events $\mathcal{S}$, and write the joint distribution over the mother and daughter points as

$$p(\mathcal{S}^{\mathsf{mtr}}, \mathcal{S} \mid \mathcal{V}, \boldsymbol{\psi}_\lambda, \boldsymbol{\theta}, \lambda^{\mathsf{dtr}}(\cdot)) = \mathsf{SGCP}(\mathcal{S}^{\mathsf{mtr}} \mid \mathcal{V} \oplus \mathcal{Q}, \boldsymbol{\theta}, \boldsymbol{\psi}_\lambda)\ \mathcal{PP}(\mathcal{S} \mid \mathcal{V}, \lambda^{\Sigma}(\boldsymbol{x})). \tag{5.23}$$

By using the aggregated intensity function $\lambda^{\Sigma}(\boldsymbol{x})$ that results from superposition, it is possible to marginalise over the assignments of daughters to mothers. Conditioned on the mother points $\mathcal{S}^{\mathsf{mtr}}$, inference of the latent mother intensity is exactly as in Section 5.3. Additional work must be done, however, to sample from the number and location of the mother points themselves.

Sampling the Number of Mother Points

To sample the number J of mother points in $\mathcal{S}^{\mathsf{mtr}}$, I define birth and death proposals for the mother points on $\mathcal{V} \oplus \mathcal{Q}$. In its simplest form, the choice between proposing to insert or delete a mother could be done via a Bernoulli coin flip with probability $\frac{1}{2}$. When introducing a new mother point, condition on the latent function $g(\boldsymbol{x})$, the current mother points, and the set of thinned events currently in the history of the sigmoidal Gaussian Cox process. Draw the location of the new mother, $\hat{\boldsymbol{x}}^{\mathsf{mtr}}$ uniformly on the domain $\mathcal{V} \oplus \mathcal{Q}$, and draw the associated function values $g(\hat{\boldsymbol{x}}^{\mathsf{mtr}})$ from the Gaussian process, conditioning on everything currently known about $g(\boldsymbol{x})$ in the sigmoidal Gaussian Cox process. This proposal is similar to proposing a thinned event in the SGCP, except in this case it is an unthinned (but unobserved) event. The Metropolis–Hastings acceptance ratio of the proposed birth is then

$$a_{\mathsf{ns-ins}} = \frac{\mu(\mathcal{V} \oplus \mathcal{Q})\ \bar{\lambda}(\hat{\boldsymbol{x}}^{\mathsf{mtr}}\,;\,\boldsymbol{\psi}_\lambda)\ \Phi(g(\hat{\boldsymbol{x}}^{\mathsf{mtr}}))}{(J+1)\ \exp\left\{\Lambda_{\mathcal{V}}^{\mathsf{dtr}}(\hat{\boldsymbol{x}}^{\mathsf{mtr}})\right\}} \prod_{k=1}^{K} \left(1 + \frac{\lambda^{\mathsf{dtr}}(\boldsymbol{x}_k\,;\,\hat{\boldsymbol{x}}^{\mathsf{mtr}})}{\sum_{j=1}^{J} \lambda^{\mathsf{dtr}}(\boldsymbol{x}_k\,;\,\boldsymbol{x}_j^{\mathsf{mtr}})}\right), \tag{5.24}$$

where

$$\Lambda_{\mathcal{V}}^{\mathsf{dtr}}(\hat{\boldsymbol{x}}^{\mathsf{mtr}}) = \int_{\mathcal{V}} \mathrm{d}\boldsymbol{x}\ \lambda^{\mathsf{dtr}}(\boldsymbol{x}\,;\,\hat{\boldsymbol{x}}^{\mathsf{mtr}}). \tag{5.25}$$

When proposing to remove a mother, I select one uniformly from among the J in $\mathcal{S}^{\mathsf{mtr}}$. The acceptance ratio of a proposal to remove the jth mother point is

$$a_{\mathsf{ns-del}} = \frac{J\ \exp\left\{\Lambda_{\mathcal{V}}^{\mathsf{dtr}}(\boldsymbol{x}_j^{\mathsf{mtr}})\right\}}{\mu(\mathcal{V}\oplus\mathcal{Q})\ \bar{\lambda}(\boldsymbol{x}_j^{\mathsf{mtr}}\,;\,\boldsymbol{\psi}_\lambda)\ \Phi(g(\boldsymbol{x}_j^{\mathsf{mtr}}))} \times \prod_{k=1}^{K}\left(1 - \frac{\lambda^{\mathsf{dtr}}(\boldsymbol{x}_k\,;\,\boldsymbol{x}_j^{\mathsf{mtr}})}{\sum_{j'=1}^{J}\lambda^{\mathsf{dtr}}(\boldsymbol{x}_k\,;\,\boldsymbol{x}_{j'}^{\mathsf{mtr}})}\right). \tag{5.26}$$

Even though the daughter intensities are modeled as being identical, except for their locations, the $\exp\{\Lambda_{\mathcal{V}}^{\mathsf{dtr}}(\cdot)\}$ terms are still necessary in these acceptance ratios because mother points are allowed to be in the larger domain $\mathcal{V}\oplus\mathcal{Q}$. This is done to account for edge effects. The aggregate intensity function $\lambda^{\Sigma}(\boldsymbol{x})$, however, only models daughter realisations in $\mathcal{V}$, as that is how the likelihood is defined. The data say nothing about potential daughters outside of $\mathcal{V}$. As not all of the support of a given daughter process may be within $\mathcal{V}$, it is important to account for this when sampling the mother point configuration.

Sampling the Mother Locations

Conditioned on the number of mothers J, we can also sample from the mother locations $\mathcal{S}^{\mathsf{mtr}}$, given the rest of the history of the sigmoidal Gaussian Cox process. Iterate over each mother, and draw a proposal to move the jth mother point from location $\boldsymbol{x}_j^{\mathsf{mtr}}$ to $\hat{\boldsymbol{x}}_j^{\mathsf{mtr}}$ from a density $q(\hat{\boldsymbol{x}}_j^{\mathsf{mtr}} \leftarrow \boldsymbol{x}_j^{\mathsf{mtr}})$, sampling the function value $g(\hat{\boldsymbol{x}}_j^{\mathsf{mtr}})$ from the Gaussian process. The acceptance ratio of this proposal is

$$a_{\mathsf{ns-loc}} = \frac{q(\boldsymbol{x}_j^{\mathsf{mtr}} \leftarrow \hat{\boldsymbol{x}}_j^{\mathsf{mtr}})\ \bar{\lambda}(\hat{\boldsymbol{x}}_j^{\mathsf{mtr}}\,;\,\boldsymbol{\psi}_\lambda)\ \Phi(g(\hat{\boldsymbol{x}}_j^{\mathsf{mtr}}))\ \exp\left\{\Lambda_{\mathcal{V}}^{\mathsf{dtr}}(\boldsymbol{x}_j^{\mathsf{mtr}})\right\}}{q(\hat{\boldsymbol{x}}_j^{\mathsf{mtr}} \leftarrow \boldsymbol{x}_j^{\mathsf{mtr}})\ \bar{\lambda}(\boldsymbol{x}_j^{\mathsf{mtr}}\,;\,\boldsymbol{\psi}_\lambda)\ \Phi(g(\boldsymbol{x}_j^{\mathsf{mtr}}))\ \exp\left\{\Lambda_{\mathcal{V}}^{\mathsf{dtr}}(\hat{\boldsymbol{x}}_j^{\mathsf{mtr}})\right\}} \times \prod_{k=1}^{K}\frac{\lambda^{\mathsf{dtr}}(\boldsymbol{x}_k\,;\,\hat{\boldsymbol{x}}_j^{\mathsf{mtr}}) + \sum_{j'\neq j}^{J}\lambda^{\mathsf{dtr}}(\boldsymbol{x}_k\,;\,\boldsymbol{x}_{j'}^{\mathsf{mtr}})}{\sum_{j'=1}^{J}\lambda^{\mathsf{dtr}}(\boldsymbol{x}_k\,;\,\boldsymbol{x}_{j'}^{\mathsf{mtr}})}. \tag{5.27}$$

The posterior distribution the latent history of the SGCP-derived Neyman–Scott process does not depend on $g(\boldsymbol{x})$ at any locations other than the latent thinned events and the latent mother points. Therefore, as discussed in Section 5.4 for the Gaussian process density sampler, if mother moves are accepted, then the old function values can be discarded. Similarly, if the mother move is rejected, the proposed function value does not need to be retained.

5.5.2 Inference in the Generalised Matérn Type III Process

Among the Matérn repulsive processes described in Section 3.5.2, the one capable of the highest densities is the Type III variant. In that section, I also presented a softened generalisation, which contains the original Matérn hard-core process as a special case. I showed how the sigmoidal Gaussian Cox process could be used to make this process inhomogeneous. This section provides an overview of how to extend latent history inference in the SGCP to the generalised inhomogeneous Matérn Type III process. I will assume *a priori* that all primary and secondary points are limited to $\mathcal{V}$, as by periodic boundary conditions. I will denote the repulsion kernel as $\rho(\boldsymbol{x} \nleftarrow \boldsymbol{x}')$.

The observations are K secondary points, which I denote $\mathcal{S} = \{\boldsymbol{x}_k\}_{k=1}^K$. These points were the ones left after a possibly-larger set of primary points were thinned. I model these primary points as having been drawn from a sigmoidal Gaussian Cox process. The objective is to infer the latent function $g(\boldsymbol{x})$ that gave rise to the intensity function of the primary process. I model an additional J events that *were not* thinned by the SGCP (becoming primary points), but *were* thinned by the repulsion kernel (so they were not observed). I denote these thinned primary points as $\mathcal{S}^{\mathsf{p}\setminus\mathsf{s}} = \{\boldsymbol{x}_j^{\mathsf{p}\setminus\mathsf{s}}\}_{j=1}^J$. I use "$\mathsf{p}\setminus\mathsf{s}$" to mean "in the primary set, excluding those in the secondary set." As discussed in Section 3.5.2, the original Matérn model uses timestamps to induce an ordering on the primary events. Since the timestamps in the generative process are i.i.d., this is equivalent to a uniform prior distribution over permutations of the $K + J$ primary events. I will represent this permutation directly in the model and denote it by Π, rather than model the timestamps as done by Huber and Wolpert (2009). I define the set $\mathcal{P}_\Pi(k)$ to be the indices of the events in the secondary set that, according to Π, appear "earlier" than the member with index k. The set $\mathcal{P}_\Pi^{\mathsf{p}\setminus\mathsf{s}}(j)$ is defined in the same way, but j indexes the thinned secondary points in $\mathcal{S}^{\mathsf{p}\setminus\mathsf{s}}$. More formally, I write these as set functions:

$$\mathcal{P}_\Pi(k) = \{k' : \Pi(\boldsymbol{x}_{k'}) < \Pi(\boldsymbol{x}_k), \boldsymbol{x}_{k'} \in \mathcal{S}, \boldsymbol{x}_k \in \mathcal{S}\} \tag{5.28}$$

$$\mathcal{P}_\Pi^{\mathsf{p}\setminus\mathsf{s}}(j) = \{k : \Pi(\boldsymbol{x}_k) < \Pi(\boldsymbol{x}_j^{\mathsf{p}\setminus\mathsf{s}}), \boldsymbol{x}_k \in \mathcal{S}, \boldsymbol{x}_j^{\mathsf{p}\setminus\mathsf{s}} \in \mathcal{S}^{\mathsf{p}\setminus\mathsf{s}}\}. \tag{5.29}$$

I define these two set functions so that it is easy to denote the events in $\mathcal{S}$ that had the power to veto another point in $\mathcal{S}$ or in $\mathcal{S}^{\mathsf{p}\setminus\mathsf{s}}$, given the permutation Π. Having defined this set of latent variables, the joint distribution over the latent history — the observed secondary points, the thinned primary points, and the ordering — is

$$p(\mathcal{S}, \mathcal{S}^{\mathsf{p}\setminus\mathsf{s}}, \Pi, J, K \mid \mathcal{V}, \boldsymbol{\theta}, \psi_\lambda) = \mathsf{SGCP}(\mathcal{S} \cup \mathcal{S}^{\mathsf{p}\setminus\mathsf{s}} \mid \mathcal{V}, \boldsymbol{\theta}, \psi_\lambda) \; \frac{1}{(K+J)!}$$
$$\times \left[\prod_{k=1}^{K} \prod_{k' \in \mathcal{P}_\Pi(k)} (1 - \rho(\boldsymbol{x}_k \nleftarrow \boldsymbol{x}_{k'}))\right] \prod_{j=1}^{J} \left(1 - \prod_{k \in \mathcal{P}_\Pi^{\mathsf{p}\setminus\mathsf{s}}(j)} (1 - \rho(\boldsymbol{x}_j^{\mathsf{p}\setminus\mathsf{s}} \nleftarrow \boldsymbol{x}_k))\right). \tag{5.30}$$

Given the set $\mathcal{S}^{\mathsf{p}\setminus\mathsf{s}}$, inference of the latent function and events thinned in the first (SGCP) stage is exactly as in Section 5.3. It is also necessary, however, to sample from the number and location of the thinned primary events $\mathcal{S}^{\mathsf{p}\setminus\mathsf{s}}$, and the permutation Π.

Sampling the Number of Thinned Primary Events

We can make insertion or deletion proposals based on the outcome of a Bernoulli coin flip with probability $\frac{1}{2}$. When inserting a new event into the set $\mathcal{S}^{\mathsf{p}\setminus\mathsf{s}}$, also propose its location $\hat{\boldsymbol{x}}^{\mathsf{p}\setminus\mathsf{s}}$ uniformly from $\mathcal{V}$. Draw a function value $g(\hat{\boldsymbol{x}}^{\mathsf{p}\setminus\mathsf{s}})$ from the Gaussian process, conditioning upon the rest of $g(\boldsymbol{x})$ currently stored in the sigmoidal Gaussian Cox process state. Also propose a location of the event in the ordering Π, selecting uniformly from among the $K + J + 1$ possibilities. I use $\mathcal{P}_{\Pi}^{\mathsf{p}\setminus\mathsf{s}}(\star)$ to denote the set of secondary events that would appear before the new proposed event. The Metropolis–Hastings acceptance ratio of this new configuration with an additional member of $\mathcal{S}^{\mathsf{p}\setminus\mathsf{s}}$ is

$$a_{\mathsf{mat-ins}} = \frac{\mu(\mathcal{V})\,(J+K+1)}{J+1}\bar{\lambda}(\hat{\boldsymbol{x}}^{\mathsf{p}\setminus\mathsf{s}}\,;\,\boldsymbol{\psi}_\lambda)\,\Phi(g(\hat{\boldsymbol{x}}^{\mathsf{p}\setminus\mathsf{s}})) \times \left[1 - \prod_{k\in\mathcal{P}_{\Pi}^{\mathsf{p}\setminus\mathsf{s}}(\star)} (1 - \rho(\hat{\boldsymbol{x}}^{\mathsf{p}\setminus\mathsf{s}} \nleftarrow \boldsymbol{x}_k))\right]. \quad (5.31)$$

When proposing a removal, choose uniformly from among the J members of $\mathcal{S}^{\mathsf{p}\setminus\mathsf{s}}$. When removing the jth event, leave the ordering of the remaining events unchanged. The MH acceptance ratio of the proposal to remove the jth member of $\mathcal{S}^{\mathsf{p}\setminus\mathsf{s}}$ is

$$a_{\mathsf{mat-del}} = \frac{J}{\mu(\mathcal{V})\,(J+K)}\bar{\lambda}(\boldsymbol{x}_j^{\mathsf{p}\setminus\mathsf{s}}\,;\,\boldsymbol{\psi}_\lambda)^{-1}\,\Phi(g(\boldsymbol{x}_j^{\mathsf{p}\setminus\mathsf{s}}))^{-1} \times \left[1 - \prod_{k\in\mathcal{P}_{\Pi}^{\mathsf{p}\setminus\mathsf{s}}(j)} (1 - \rho(\boldsymbol{x}_j^{\mathsf{p}\setminus\mathsf{s}} \nleftarrow \boldsymbol{x}_k))\right]^{-1}. \quad (5.32)$$

Sampling the Locations of the Thinned Primary Events

Conditioned on the number of members in $\mathcal{S}^{\mathsf{p}\setminus\mathsf{s}}$ and the permutation Π, we can sample from the locations of the $\mathcal{S}^{\mathsf{p}\setminus\mathsf{s}}$ by iterating over each one and making a Metropolis–Hastings move. Draw a proposal for a new location for the jth event by drawing it from a density $q(\hat{\boldsymbol{x}}_j^{\mathsf{p}\setminus\mathsf{s}} \leftarrow \boldsymbol{x}_j^{\mathsf{p}\setminus\mathsf{s}})$. Then draw an associated value for the latent function $g(\hat{\boldsymbol{x}}_j^{\mathsf{p}\setminus\mathsf{s}})$ from the Gaussian process, conditioning on the rest of the history. The

MH acceptance ratio of this proposal is

$$a_{\mathsf{mat-loc}} = \left(q(\boldsymbol{x}_j^{\mathsf{p}\backslash\mathsf{s}} \leftarrow \hat{\boldsymbol{x}}_j^{\mathsf{p}\backslash\mathsf{s}})\ \bar{\lambda}(\hat{\boldsymbol{x}}_j^{\mathsf{p}\backslash\mathsf{s}};\ \boldsymbol{\psi}_\lambda)\ \Phi(g(\hat{\boldsymbol{x}}_j^{\mathsf{p}\backslash\mathsf{s}})) \left[1 - \prod_{k \in \mathcal{P}_\Pi^{\mathsf{p}\backslash\mathsf{s}}(j)} (1 - \rho(\hat{\boldsymbol{x}}_j^{\mathsf{p}\backslash\mathsf{s}} \nleftarrow \boldsymbol{x}_k)) \right] \right) \Bigg/ \left(q(\hat{\boldsymbol{x}}_j^{\mathsf{p}\backslash\mathsf{s}} \leftarrow \boldsymbol{x}_j^{\mathsf{p}\backslash\mathsf{s}})\ \bar{\lambda}(\boldsymbol{x}_j^{\mathsf{p}\backslash\mathsf{s}};\ \boldsymbol{\psi}_\lambda)\ \Phi(g(\boldsymbol{x}_j^{\mathsf{p}\backslash\mathsf{s}})) \left[1 - \prod_{k \in \mathcal{P}_\Pi^{\mathsf{p}\backslash\mathsf{s}}(j)} (1 - \rho(\boldsymbol{x}_j^{\mathsf{p}\backslash\mathsf{s}} \nleftarrow \boldsymbol{x}_k)) \right] \right). \tag{5.33}$$

Here again, the latent history method allows the old values of the function to be discarded after an acceptance.

Sampling the Ordering of Primary Events

It is necessary to propose changing the ordering Π of the primary events, in order to sample from the posterior distribution on the latent history of the generalised Matérn process. This can be done while conditioning on all of the rest of the state: the number and locations of the thinned primary events $\mathcal{S}^{\mathsf{p}\backslash\mathsf{s}}$, the number and locations of the events thinned during the sigmoidal Gaussian Cox process generation, and the latent function $g(\boldsymbol{x})$. I suggest simple local proposals that only swap two adjacent members of the primary set. Choose uniformly and randomly from among the $J + K - 1$ such proposals. These proposals allow ergodic Markov chains on permutations, as it is possible to eventually transition between any two orderings (Cormen et al., 2001). For this inference task, these local swaps are appealing because they do not change the ordering relationship of the pair with all of the other primary events. I will use the symbol $\blacktriangleleft$ to refer to the index of the "early" event in the pair and the symbol $\blacktriangleright$ to denote the index of the "late" event in the pair. So, in the sequence of primary events, there is a subsequence of size two — a pair of adjacent events — $\boldsymbol{x}_\blacktriangleleft$ immediately followed by an event $\boldsymbol{x}_\blacktriangleright$. There are four possible cases, depending on the sets to which these events belong.

Case 1: Both are Secondary Events If both the early and late events are observed and in the secondary set, i.e. $\boldsymbol{x}_\blacktriangleleft \in \mathcal{S}$ and $\boldsymbol{x}_\blacktriangleright \in \mathcal{S}$, then the MH acceptance ratio of swapping their ordering is a comparison of their respective vetoes:

$$a_1 = \frac{1 - \rho(\boldsymbol{x}_\blacktriangleleft \nleftarrow \boldsymbol{x}_\blacktriangleright)}{1 - \rho(\boldsymbol{x}_\blacktriangleright \nleftarrow \boldsymbol{x}_\blacktriangleleft)}. \tag{5.34}$$

Case 2: Neither is a Secondary Event If neither is a secondary event, i.e. $\boldsymbol{x}_\blacktriangleleft \in \mathcal{S}^{\mathsf{p}\backslash\mathsf{s}}$ and $\boldsymbol{x}_\blacktriangleright \in \mathcal{S}^{\mathsf{p}\backslash\mathsf{s}}$, then the joint probability of the latent history does not change when the two events are swapped and $a_2 = 1$.

Case 3: The Early Event is Secondary, the Late Event is Not If the early event is in the secondary set and the late event is not in the secondary set, i.e. $\boldsymbol{x}_{\blacktriangleleft} \in \mathcal{S}$ and $\boldsymbol{x}_{\blacktriangleright} \in \mathcal{S}^{\mathsf{p}\setminus\mathsf{s}}$, then the Metropolis–Hastings acceptance ratio of the swap is

$$a_3 = \frac{1 - \prod_{k \in \mathcal{P}_\Pi(\blacktriangleleft)} (1 - \rho(\boldsymbol{x}_{\blacktriangleright}^{\mathsf{p}\setminus\mathsf{s}} \nleftarrow \boldsymbol{x}_k))}{1 - \prod_{k \in \mathcal{P}_\Pi^{\mathsf{p}\setminus\mathsf{s}}(\blacktriangleright)} (1 - \rho(\boldsymbol{x}_{\blacktriangleright}^{\mathsf{p}\setminus\mathsf{s}} \nleftarrow \boldsymbol{x}_k))}. \tag{5.35}$$

Case 4: The Late Event is Secondary, the Early Event is Not This is the opposite of the third case. The early event is not in the secondary set, but the late event is, i.e. $\boldsymbol{x}_{\blacktriangleleft} \in \mathcal{S}^{\mathsf{p}\setminus\mathsf{s}}$ and $\boldsymbol{x}_{\blacktriangleright} \in \mathcal{S}$. The MH ratio of exchanging their order is the inverse of a_3 in Equation 5.35:

$$a_4 = \frac{1 - (1 - \rho(\boldsymbol{x}_{\blacktriangleright}^{\mathsf{p}\setminus\mathsf{s}} \nleftarrow \boldsymbol{x}_{\blacktriangleleft})) \prod_{k \in \mathcal{P}_\Pi^{\mathsf{p}\setminus\mathsf{s}}(\blacktriangleright)} (1 - \rho(\boldsymbol{x}_{\blacktriangleright}^{\mathsf{p}\setminus\mathsf{s}} \nleftarrow \boldsymbol{x}_k))}{1 - \prod_{k \in \mathcal{P}_\Pi^{\mathsf{p}\setminus\mathsf{s}}(\blacktriangleright)} (1 - \rho(\boldsymbol{x}_{\blacktriangleright}^{\mathsf{p}\setminus\mathsf{s}} \nleftarrow \boldsymbol{x}_k))}. \tag{5.36}$$

5.6 Summary

This chapter proposed a method of inference in the Gaussian process density sampler and the sigmoidal Gaussian Cox process that models the latent history of their respective generative procedures. Despite having doubly-intractable posterior distributions, the availability of an exact generative procedure for data enables a latent variable model that is tractable via Markov chain Monte Carlo. I also showed that this latent history method could be applied to inhomogeneous variants of the Neyman–Scott and generalised Matérn processes, when they are based on sigmoidal Gaussian Cox processes. I will apply the latent history approach to data in Chapter 6 and show empirically that it has superior properties to the exchange sampling approach of Chapter 4. In Chapter 7, I will discuss additional topics related to latent history inference, and in Chapter 8 I will infer the latent history of the Archipelago generative model for semi-supervised learning.

Chapter 6

Application Examples

In this chapter, I apply to data the computational methods that I have developed in this thesis. I look at several synthetic and real-world data sets. In Section 6.1, I study a set of density modeling problems and compare the Gaussian process density sampler to related methods. In Section 6.2, I apply the sigmoidal Gaussian Cox process to several data sets and compare it to other methods.

6.1 Density Modeling Examples

In Chapter 2, I presented the Gaussian process density sampler as an alternative to the logistic Gaussian process for nonparametric modeling of probability density functions. In Chapter 4 and Chapter 5, I discussed two methods for performing inference in the GPDS. In this section, I apply these methods and compare them empirically.

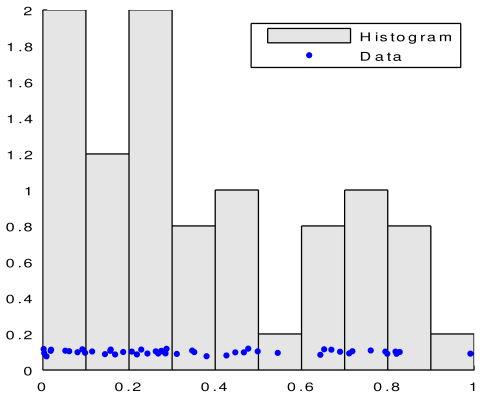
6.1.1 Bounded Univariate Density

I first look at a simple case: data from an unknown univariate density with known finite support. I study the same density on $[0, 1]$ which was examined by Lenk (1991) and Tokdar (2007) for the logistic Gaussian process:

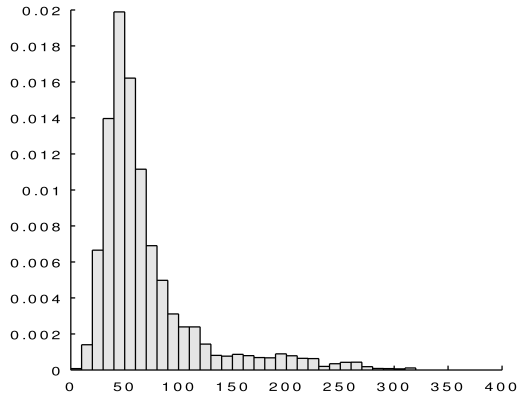
$$f_{\mathsf{lenk}}(x) \propto \frac{3}{4} \cdot 3 \exp\{-3x\} + \frac{1}{4} \cdot \left(\frac{\pi}{32}\right)^{-\frac{1}{2}} \exp\{-32(x - \frac{3}{4})^2\}. \tag{6.1}$$

This density is a mixture of an exponential and a Gaussian on $[0, 1]$ and data from it are considered to be difficult to model due to the Gaussian bump. I generated a set of 50 independent data from this density. The data, and a histogram, are shown in Figure 6.1a. The true density is shown in blue in Figure 6.1b.

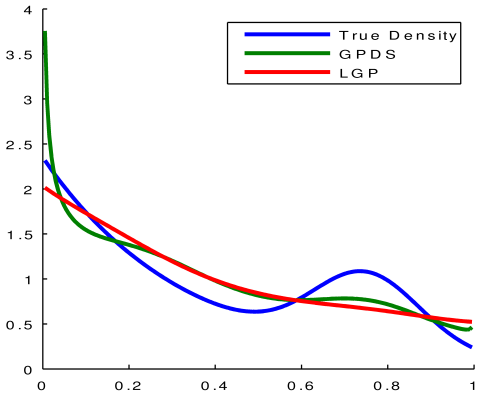
I performed inference in the Gaussian process density sampler using the latent history method, with a squared-exponential covariance function and a beta distribution for



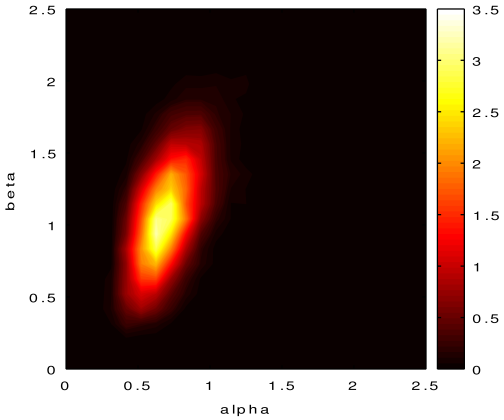
(a) Observed data and histogram



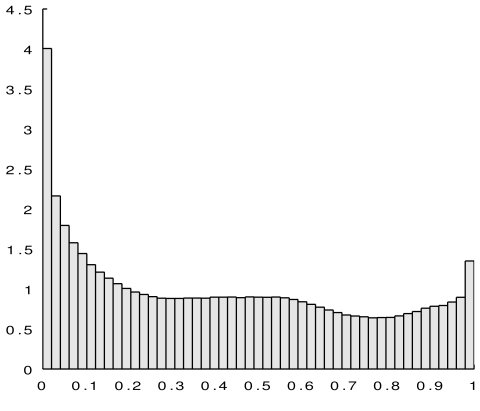
(d) Histogram of the number of rejections



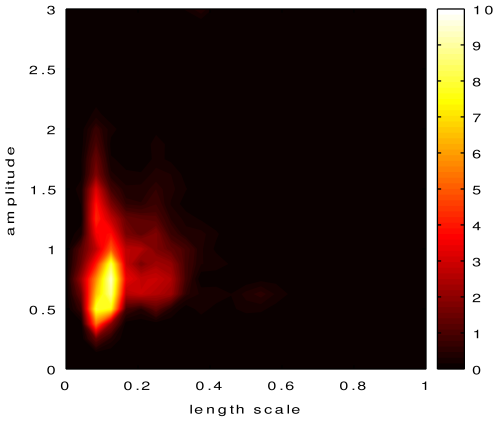
(b) Probability density functions



(e) Histogram of base density hyperparameters

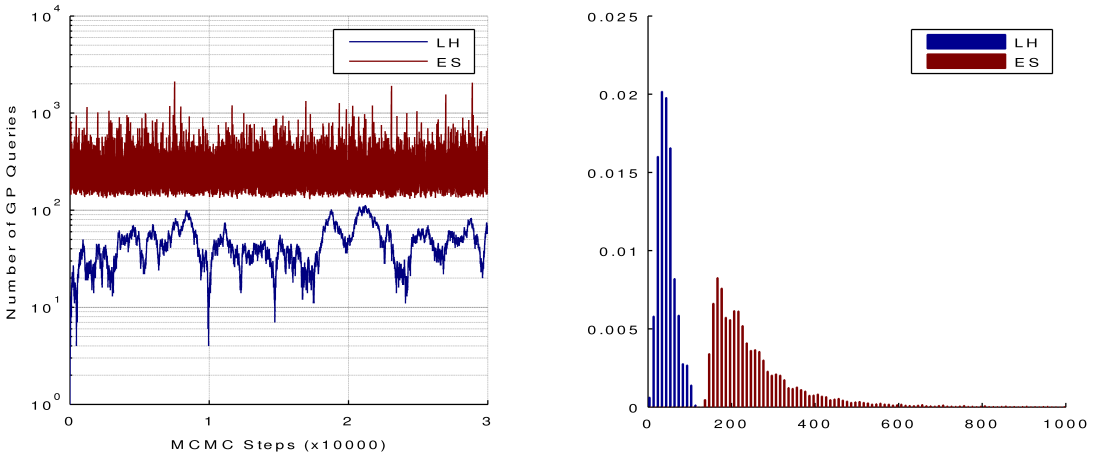


(c) Histogram of the locations of rejections



(f) Histogram of GP hyperparameters

Figure 6.1: (a) The 50 observed data, and histogram. (b) The true density in blue, with the Gaussian process density sampler estimate in green, and the logistic Gaussian process estimate (with 150 knots) in red. (c) A histogram of the locations of rejections during the latent history inference. (d) A histogram of the number of rejections in the latent history. (e) A histogram of the α and β parameters for the beta base density. (f) A histogram of the length scale ℓ and amplitude ω hyperparameters for the squared-exponential covariance function.



(a) Trace of Markov state on MCMC run

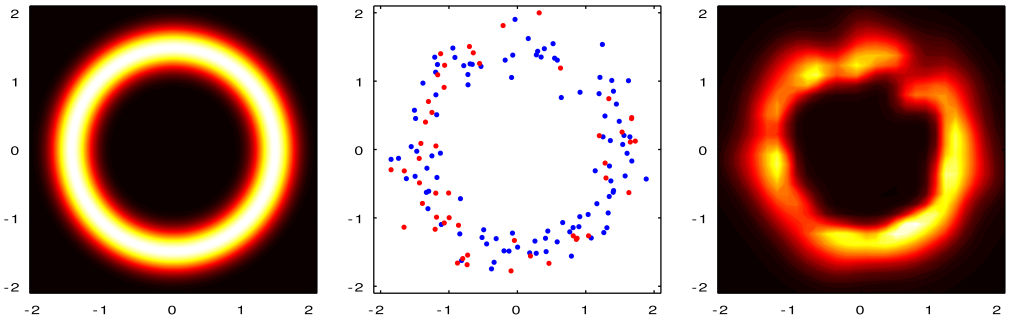
(b) Histogram of Markov state on MCMC run

Figure 6.2: These figures compare the number of data kept in the Gaussian process over the course of 30K MCMC samples. Both the exchange sampling and latent history methods used identical data and hyperparameters. (a) A trace of the number of data in the Markov state. For the latent history method this is the number of data plus the number of latent rejections. For the exchange sampling method, this is the size of the current conditioning set. (b) A histogram of the values in the trace.

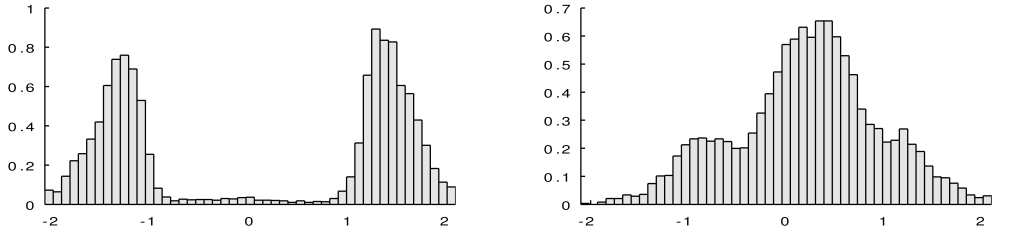
the base density. I performed hyperparameter inference for both the base density and the Gaussian process. I also applied the logistic Gaussian process to the data, using the method of Tokdar (2007) with 100 uniformly-spaced knots, plus the 50 data. Figure 6.1b shows the predictive distribution from both the GPDS and the LGP. The Gaussian process density sampler does a better job of capturing the bump, but creates an unwanted spike near $x = 0$, due to the shape of the beta base density.

Figure 6.1c is a histogram of the locations of the latent rejections inferred during the Markov chain Monte Carlo inference of the Gaussian process density sampler latent history, and Figure 6.1d is a histogram of the number of rejections. Figure 6.1e and Figure 6.1f are histograms of the parameters of the beta base density and the GP hyperparameters, respectively.

In Section 5.4, I discussed in theoretical terms how one might expect exchange sampling to compare to the latent history method for inference in the Gaussian process density sampler. I argued that the number of data stored in the Markov state would tend to be greater with exchange sampling. Specifically, I am concerned with the number of unique locations at which any particular function in the Markov state has been evaluated (sampled from the Gaussian process). In the exchange sampling method, this is the size of the conditioning set. For the latent history inference, this number is the sum of the number of data and the number of rejections. In Figure 6.2a, I plot a trace over 30K Markov chain Monte Carlo transitions of the size of the Gaussian process state for each method. Figure 6.2b shows the histograms for these data. The exchange sampling algorithm was run using the underrelaxed



(a) True ring PDF (b) Ring training and test data (c) Ring predictive distribution



(d) Conditional predictive samples at $x_1 = 0$ (e) Conditional predictive samples at $x_1 = 1.5$

Figure 6.3: (a) The probability density function of the ring data, as given in Equation 6.2. (b) The 100 training data are shown in blue and the 50 test data are shown in red. (c) The predictive density from latent history inference with the Gaussian process density sampler. (d) Predictive samples from x_2, conditioned on $x_1 = 0$. (e) Predictive samples from x_2, conditioned on $x_1 = 1.5$. These two figures are vertical "slices" through the predictive distribution in the middle of the ring and at the edge of the ring.

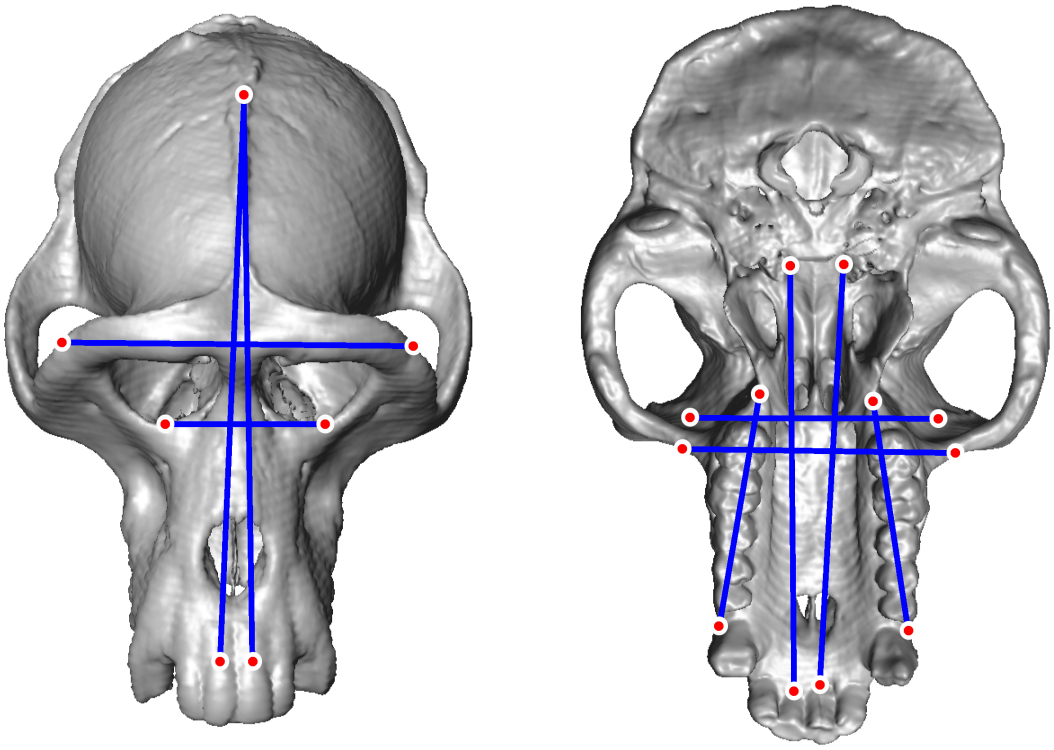
method with $\kappa = 0.9$ and a uniformly-spaced grid of 25 control points, plus the data. The latent history algorithm was run with $\zeta(M, N) = \frac{1}{2}$ and one insert or delete proposal per MCMC step. In both cases, the hyperparameters for the base density and Gaussian process were fixed and identical. Note that the plots reflect the fact that the minimum number of data the exchange sampling algorithm can have is $125 = 50$ data $+ 50$ fantasies $+ 25$ control points. In this example, even the worst-case state of the latent history method was smaller than the best-case state of the exchange sampling algorithm.

6.1.2 Bivariate Ring Density

I applied the Gaussian process density sampler and the latent history inference method to data from a density with the shape of a ring:

$$f_{\text{ring}}(\boldsymbol{x}) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \mathrm{d}\chi \, \mathcal{N}\left(\boldsymbol{x}\,;\, \boldsymbol{\mu} = \frac{3}{2}\begin{bmatrix}\cos\chi \\ \sin\chi\end{bmatrix}, \sigma \boldsymbol{I}_2\right), \tag{6.2}$$

for $\sigma = 0.2$. This density is shown in Figure 6.3a, and the 100 independent training data are shown in Figure 6.3b. I generated an additional 50 test data to evaluate the predictive log probability, and compared the GPDS to 1) a Parzen window method



(a) *Macaca mulatta* superior view (b) *Macaca mulatta* inferior view

Figure 6.4: Ten linear distances between anatomical landmarks, superimposed on superior and inferior views of a rhesus macaque (*Macaca mulatta*). The landmark names are listed in Appendix B in Table B.1 (page 125).

with Gaussian kernels and bandwidth set via ten fold cross-validation on the log likelihood (Rosenblatt, 1956; Parzen, 1962); 2) a Dirichlet process mixture of Gaussians (Rasmussen, 2000; Neal, 2000); and 3) the Dirichlet diffusion tree method (Neal, 2003a). Radford Neal's software for flexible Bayesian modeling (FBM)[1] was used to evaluate the two Dirichlet methods. The GPDS predictive density is shown in Figure 6.3c and the numerical results appear in Table 6.1. The Bayesian methods, including the Gaussian process density sampler, make significantly better predictions than the frequentist Parzen window method.

These ring data highlight a useful aspect of the Gaussian process density sampler construction: it is convenient to draw from the conditional predictive distribution. If the D dimensions of $\mathcal{X}$ are divided into two sets, $\boldsymbol{x}_A$ and $\boldsymbol{x}_B$, so that we have $\boldsymbol{x} = [\boldsymbol{x}_A \; \boldsymbol{x}_B]$, then one might wish to find

$$p(\boldsymbol{x}_A \,|\, \mathcal{D}, \boldsymbol{x}_B) = \int \mathrm{d}\boldsymbol{g} \; p(\boldsymbol{x}_A \,|\, \boldsymbol{g}, \boldsymbol{x}_B) \, p(\boldsymbol{g} \,|\, \mathcal{D}). \tag{6.3}$$

This can be motivated by the supervised learning problem where the model has been trained on the joint distribution over features and labels, and one wishes to make

[1] `http://www.cs.toronto.edu/~radford/fbm.software.html`

	GPDS	Parzen	DPMM	DFT
Ring	0.439	-2.253	-0.236	0.730
Macaque Trial 1	-15.285	-15.443	-15.267	-15.259
Macaque Trial 2	-15.044	-15.742	-15.176	-15.136
Macaque Trial 3	-14.863	-15.254	-15.077	-14.775

Table 6.1: This table shows the mean log probability of held-out test data for each of the data sets and compared models. The compared models are the Gaussian process density sampler (GPDS), the Parzen window method (Parzen), the Dirichlet process mixture of Gaussians (DPMM) and the Dirichlet diffusion tree (DFT).

predictions of the label given a previously-unseen set of features. Generating samples from this distribution is exactly as in Section 4.5 and Section 5.2.6 — draw new fantasy data after each MCMC step — with the difference being that the fantasy proposals are drawn from $\pi(\boldsymbol{x}_A \mid \boldsymbol{x}_B, \boldsymbol{\psi}_\pi)$ rather than $\pi(\boldsymbol{x} \mid \boldsymbol{\psi}_\pi)$. The histograms in Figure 6.3d and Figure 6.3e are examples of the conditional predictive distribution, where the predictions are "sliced" by conditioning on one dimension of the data.

6.1.3 Macaque Skull Data

I also studied the Gaussian process density sampler on a real-world task of skull reconstruction. I modeled the joint distribution of ten measurements of linear distances between anatomical landmarks on 228 rhesus macaque (*Macaca mulatta*) skulls. These linear distances were generated from three-dimensional coordinate data taken by a single observer from dried skulls using a digitiser (Willmore et al., 2005).[2] Linear distances are commonly used in morphological studies as they are invariant under rotation and translation of the objects being compared (Lele and Richtsmeier, 2001). Figure 6.4 shows superior and inferior views of a computed tomography (CT) scan reconstruction of a macaque skull, along with the ten linear distances used. The anatomical landmarks associated with each distance are given in Appendix B in Table B.1 (page 125). Each skull was measured three times in different trials, and these were modeled separately. 200 randomly-selected skulls were used as a training set and 28 were used as a test set. To be as fair as possible, the data were logarithmically transformed and whitened as a preprocessing step, to have zero sample mean and spherical sample covariance. The results are shown in Table 6.1. The tests indicate that the Gaussian process density sampler is competitive with the other Bayesian methods, all of which outperform the frequentist estimator.

[2]I thank Katherine Willmore, the Caribbean Primate Research Center, the University of Puerto Rico, Medical Sciences Campus, Laboratory of Primate Morphology and Genetics, and the National Institutes of Health (Grant RR03640 to CPRC) for contributing these data.

6.2 Poisson Process Examples

I applied the sigmoidal Gaussian Cox process and latent history inference to two types of problems. I compared the SGCP to other equivalent methods on synthetic data, and I applied the SGCP to two real-world data sets: one temporal and one spatial.

6.2.1 Univariate Synthetic Data

The objective of using synthetic data was to compare to other methods of estimating Poisson intensity function, where the ground truth was known. I created three one-dimensional data sets using the following intensity functions:

1. A sum of an exponential and a Gaussian bump:

$$\lambda_1(t) = 2\exp\{-t/15\} + \exp\{-((t-25)/10)^2\}, \tag{6.4}$$

on the interval $[0, 50]$. There are 53 training events.

2. A sinusoid with increasing frequency:

$$\lambda_2(t) = 5\sin(t^2)+6, \tag{6.5}$$

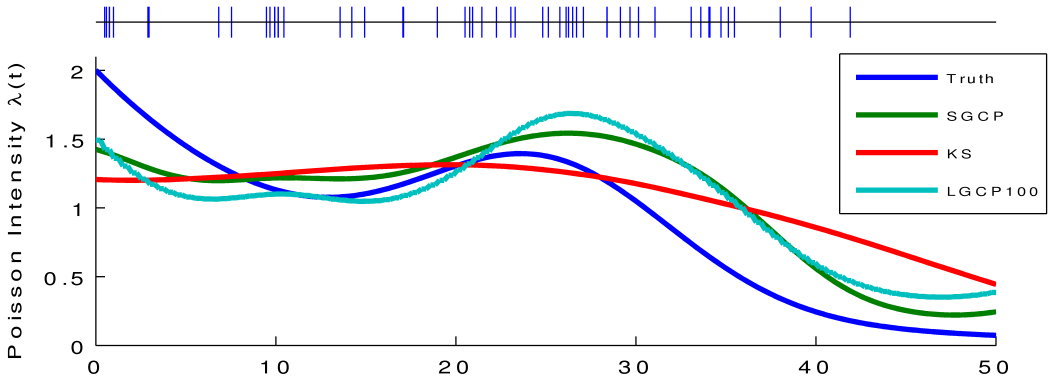
on $[0, 5]$. There are 29 training events.

3. $\lambda_3(t)$ is the piecewise linear function shown in Figure 6.5c, on the interval $[0, 100]$. There are 235 training events.

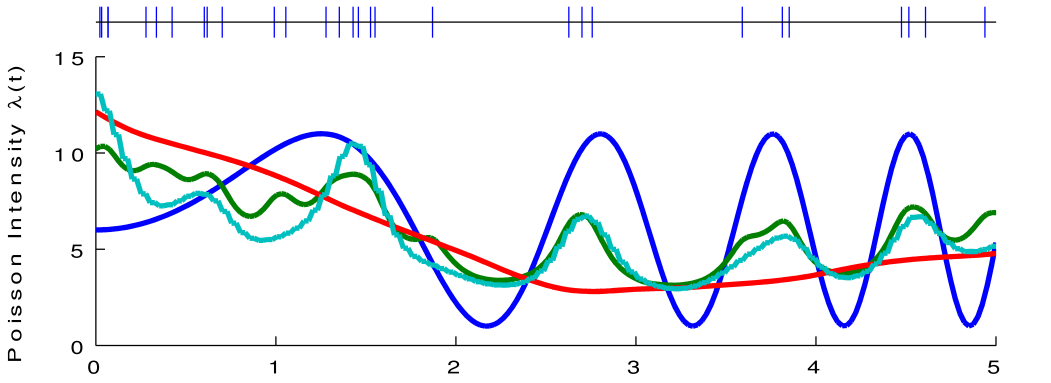
I compared the sigmoidal Gaussian Cox process to the classical kernel smoothing (KS) approach of Diggle (1985). I performed edge-corrected kernel smoothing using a quartic kernel and the recommended mean-square minimisation technique for bandwidth selection. I also compared to the most closely-related nonparametric Bayesian technique, the log Gaussian Cox process of Rathbun and Cressie (1994) and Møller et al. (1998). To implement this method, I used discretisation to make a finite-dimensional approximation and applied Markov chain Monte Carlo. I ran the log Gaussian Cox process method with 10 bins (LGCP10), 25 bins (LGCP25) and 100 bins (LGCP100).

I used the squared-exponential kernel for both the SGCP and the LGCP, and sampled from the hyperparameters for both models.

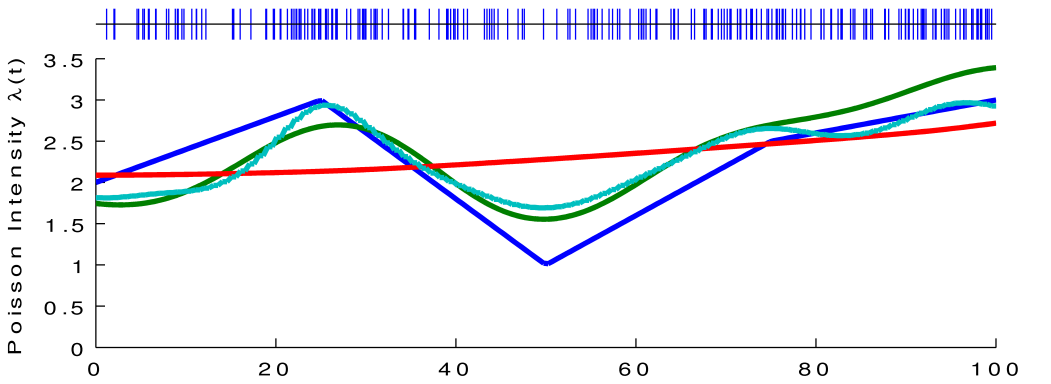
I report the numerically-estimated $\mathcal{L}_2$ distance between the mean $\lambda(t)$ provided by each method and the known true function. I also report the mean log predictive probability of ten additional held-out time series generated from the same (known) $\lambda(t)$. These predictive probabilities were calculated by numerical integrations of the functions. These results are provided in Table 6.2, and the resulting estimates (excluding



(a) Synthetic data from $\lambda_1(t)$, with intensity estimates

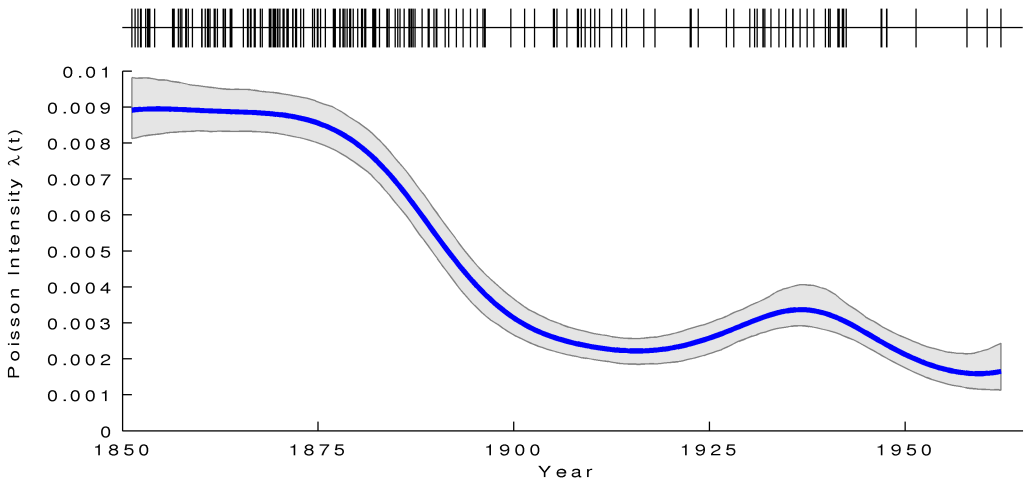


(b) Synthetic data from $\lambda_2(t)$, with intensity estimates

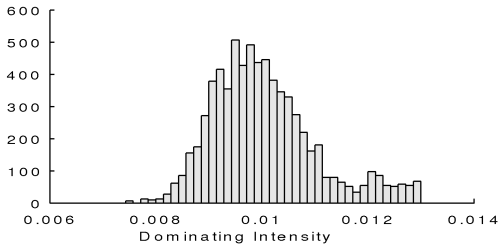


(c) Synthetic data from $\lambda_3(t)$, with intensity estimates

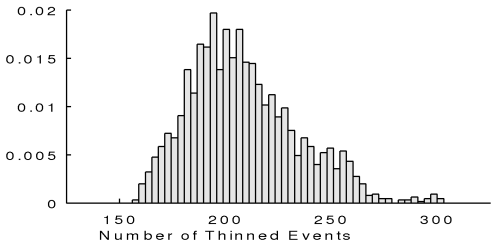
Figure 6.5: These figures show the three sets of synthetic Poisson data from known intensities, along with the predictive estimates for the sigmoidal Gaussian Cox process (SGCP), the frequentist kernel smoothing method (KS), and the log Gaussian Cox process with 100 bins (LGCP100).



(a) Coal mine disaster data and intensity estimate



(b) Histogram of dominating intensity $\lambda_\star$



(c) Histogram of number of thinned events

Figure 6.6: (a) The coal mine disaster data, estimated intensity and quartile error bars. (b) A histogram of the values of the dominating intensity $\lambda_\star$ throughout the MCMC run. (c) A histogram of the number of thinned events modeled by the latent history.

LGCP10 and LGCP25) are shown in Figure 6.5. For these evaluations, the sigmoidal Gaussian Cox process appears to outperform the competing methods.

6.2.2 Coal Mine Disaster Data

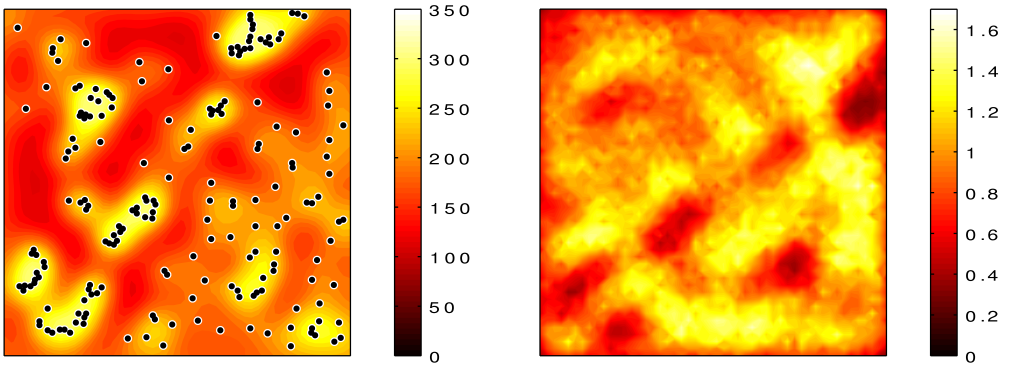
I ran the Markov chain Monte Carlo inference procedure on the classic coal mine disaster data of Jarrett (1979). These data are the dates of 191 coal mine explosions that killed ten or more men in Britain between 15 March 1851 and 22 March 1962. Figure 6.6a shows the events along the top, and the inferred mean intensity function. Also shown are approximate quartile error bars. In Figure 6.6b is a histogram of the inferred upper bounding intensity, $\lambda_\star$. Figure 6.6c is a histogram of the number of latent thinned events, M.

6.2.3 Redwoods Data

I used a standard data set from spatial statistics to demonstrate the sigmoidal Gaussian Cox process in two dimensions. These data are the locations of redwood trees studied by Ripley (1977) and others. There are 195 points and, as in previous studies, they

		SGCP	KS	LGCP10	LGCP25	LGCP100
$\lambda_1(s)$	ℓ_2	4.20	6.65	5.96	6.12	5.44
	lp	-45.11	-46.41	-46.00	-46.80	-45.24
$\lambda_2(s)$	ℓ_2	38.38	73.71	70.34	53.27	43.51
	lp	24.45	28.19	23.36	22.89	25.29
$\lambda_3(s)$	ℓ_2	11.41	30.56	90.76	22.14	10.79
	lp	-43.39	-46.47	-53.67	-52.31	-47.16

Table 6.2: This table shows the results of empirical comparison of the sigmoidal Gaussian Cox process (SGCP) versus the frequentist kernel smoothing method (KS), the the log Gaussian Cox process with various levels of discretisation (10 bins: LGCP10, 25 bins: LGCP25, 100 bins: LGCP100).



(a) Redwood data and intensity estimate (b) Histogram of locations of thinned events

Figure 6.7: (a) The redwood data, scaled to the unit square, and the estimated intensity. (b) A histogram of the locations of thinned events inferred during the MCMC run.

have been scaled to the unit square. Figure 6.7a shows the data along with the inferred mean intensity function. These data are useful for examining the placement of latent thinned events. Figure 6.7b shows a histogram of where the these events tended to be located during the MCMC run. As expected, it is approximately a "negative" of the mean intensity; the thinned events are moving to places where it is necessary to "peg down" the intensity function.

6.3 Summary

In this chapter, I applied the inference methods of Chapter 4 and Chapter 5 to the Gaussian process density sampler and the sigmoidal Gaussian Cox process. For both models, I performed inference on both synthetic and real-world data and found my proposed approaches to be competitive with existing methods.

Chapter 7

Model Extensions

This chapter presents some extensions to the Gaussian process density sampler and the sigmoidal Gaussian Cox process. Section 7.1 discusses how to make Monte Carlo estimates of the normalised predictive density for the GPDS. Section 7.2 deals with avoidance of computational bottlenecks when performing inference in the Gaussian process density sampler. Finally, Section 7.3 considers how to improve the performance of latent history inference in the sigmoidal Gaussian Cox process.

7.1 Estimation of Normalised Predictive Probabilities

As discussed in Section 4.5 and Section 5.2.6, both exchange sampling and the latent history inference methods can yield predictive samples from the Gaussian process density sampler. It is also natural, however, to require that a density model provide an estimate of the *value* of the normalised predictive density. It is possible to use the Rao-Blackwellisation method of Chib and Jeliazkov (2001) to perform this estimation.

The desired quantity in this case is

$$p(\boldsymbol{x} \mid \mathcal{D}) = \int \mathrm{d}\boldsymbol{g} \int \mathrm{d}\boldsymbol{\theta} \int \mathrm{d}\boldsymbol{\psi}_\pi \; p(\boldsymbol{x} \mid \boldsymbol{g}, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi)\, p(\boldsymbol{g}, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi \mid \mathcal{D}), \tag{7.1}$$

which integrates out the posterior on parameters to find the predictive distribution at $\boldsymbol{x}$ after seeing data $\mathcal{D}$. Consider making a Metropolis–Hastings transition from $\boldsymbol{x}$ to $\boldsymbol{x}'$ on the distribution $p(\boldsymbol{x} \mid \boldsymbol{g}, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi)\, p(\boldsymbol{g}, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi \mid \mathcal{D})$ that appears inside the integral. Metropolis–Hastings satisfies detailed balance, giving the identity

$$p(\boldsymbol{x} \mid \boldsymbol{g}, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi)\, p(\boldsymbol{g}, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi \mid \mathcal{D})\, T(\boldsymbol{x}' \leftarrow \boldsymbol{x}) = p(\boldsymbol{x}' \mid \boldsymbol{g}, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi)\, p(\boldsymbol{g}, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi \mid \mathcal{D})\, T(\boldsymbol{x} \leftarrow \boldsymbol{x}'). \tag{7.2}$$

Using $\pi(\boldsymbol{x}' \,|\, \boldsymbol{\psi}_\pi)$ as the proposal distribution, the transition operator is

$$T(\boldsymbol{x}' \leftarrow \boldsymbol{x}) = \pi(\boldsymbol{x}' \,|\, \boldsymbol{\psi}_\pi)\ \min\left(1, \frac{p(\boldsymbol{x}' \,|\, \boldsymbol{g}, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi)\, p(\boldsymbol{g}, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi \,|\, \mathcal{D})\, \pi(\boldsymbol{x} \,|\, \boldsymbol{\psi}_\pi)}{p(\boldsymbol{x} \,|\, \boldsymbol{g}, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi)\, p(\boldsymbol{g}, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi \,|\, \mathcal{D})\, \pi(\boldsymbol{x}' \,|\, \boldsymbol{\psi}_\pi)}\right). \tag{7.3}$$

Conditioned on the latent function and hyperparameters, the generative procedure of Section 2.2 provides a distribution over data space:

$$p(\boldsymbol{x} \,|\, \boldsymbol{g}, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi) = \pi(\boldsymbol{x} \,|\, \boldsymbol{\psi}_\pi)\, \Phi(g(\boldsymbol{x})). \tag{7.4}$$

Using Equation 7.3 and Equation 7.4, the identity of Equation 7.2 can now be rewritten as

$$\begin{aligned} p(\boldsymbol{x}, \boldsymbol{g}, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi \,|\, \mathcal{D})\, \pi(\boldsymbol{x}' \,|\, \boldsymbol{\psi}_\pi)\ \min\left(1, \frac{\Phi(g(\boldsymbol{x}'))}{\Phi(g(\boldsymbol{x}))}\right) = \\ p(\boldsymbol{x}', \boldsymbol{g}, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi \,|\, \mathcal{D})\, \pi(\boldsymbol{x} \,|\, \boldsymbol{\psi}_\pi)\ \min\left(1, \frac{\Phi(g(\boldsymbol{x}))}{\Phi(g(\boldsymbol{x}'))}\right). \end{aligned} \tag{7.5}$$

Integrate both sides of this identity over $\boldsymbol{x}'$, $\boldsymbol{g}$, $\boldsymbol{\theta}$, and $\boldsymbol{\psi}_\pi$, giving

$$\begin{aligned} \int \mathrm{d}\boldsymbol{\theta} \int \mathrm{d}\boldsymbol{\psi}_\pi \int \mathrm{d}\boldsymbol{g}\ \int \mathrm{d}\boldsymbol{x}'\ p(\boldsymbol{x}, \boldsymbol{g}, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi \,|\, \mathcal{D})\, \pi(\boldsymbol{x}' \,|\, \boldsymbol{\psi}_\pi)\ \min\left(1, \frac{\Phi(g(\boldsymbol{x}'))}{\Phi(g(\boldsymbol{x}))}\right) = \\ \int \mathrm{d}\boldsymbol{\theta} \int \mathrm{d}\boldsymbol{\psi}_\pi \int \mathrm{d}\boldsymbol{g}\ \int \mathrm{d}\boldsymbol{x}'\ p(\boldsymbol{x}', \boldsymbol{g}, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi \,|\, \mathcal{D})\, \pi(\boldsymbol{x} \,|\, \boldsymbol{\psi}_\pi)\ \min\left(1, \frac{\Phi(g(\boldsymbol{x}))}{\Phi(g(\boldsymbol{x}'))}\right). \end{aligned} \tag{7.6}$$

Now observe that

$$p(\boldsymbol{x}, \boldsymbol{g}, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi \,|\, \mathcal{D}) = p(\boldsymbol{x} \,|\, \mathcal{D})\, p(\boldsymbol{g}, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi \,|\, \boldsymbol{x}, \mathcal{D}), \tag{7.7}$$

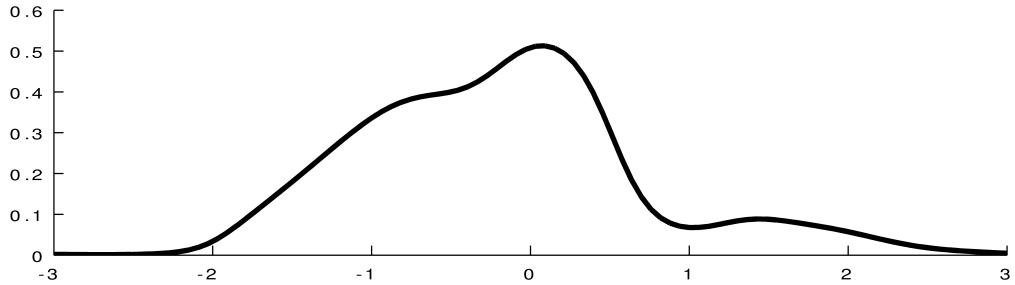
and so the predictive density can be found via

$$p(\boldsymbol{x} \,|\, \mathcal{D}) = \frac{\displaystyle\int \mathrm{d}\boldsymbol{\theta} \int \mathrm{d}\boldsymbol{\psi}_\pi \int \mathrm{d}\boldsymbol{g} \int \mathrm{d}\boldsymbol{x}'\ p(\boldsymbol{\theta}, \boldsymbol{\psi}_\pi, \boldsymbol{g}, \boldsymbol{x}' \,|\, \mathcal{D})\ \pi(\boldsymbol{x} \,|\, \boldsymbol{\psi}_\pi)\ \min\left(1, \frac{\Phi(g(\boldsymbol{x}))}{\Phi(g(\boldsymbol{x}'))}\right)}{\displaystyle\int \mathrm{d}\boldsymbol{\theta} \int \mathrm{d}\boldsymbol{\psi}_\pi \int \mathrm{d}\boldsymbol{g}\ p(\boldsymbol{\theta}, \boldsymbol{\psi}_\pi, \boldsymbol{g} \,|\, \boldsymbol{x}, \mathcal{D}) \int \mathrm{d}\boldsymbol{x}'\ \pi(\boldsymbol{x}' \,|\, \boldsymbol{\psi}_\pi)\ \min\left(1, \frac{\Phi(g(\boldsymbol{x}'))}{\Phi(g(\boldsymbol{x}))}\right)}. \tag{7.8}$$

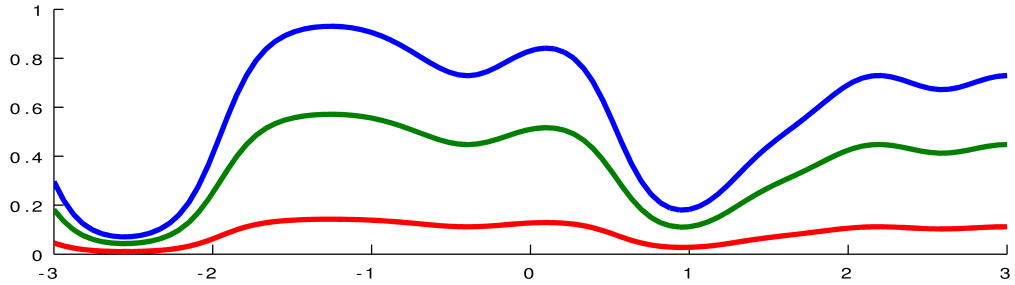
Both the numerator and the denominator in Equation 7.8 are expectations. The top is an expectation under the posterior and the bottom is an expectation under the posterior where the data have been augmented with $\boldsymbol{x}$:

$$p(\boldsymbol{x} \,|\, \mathcal{D}) = \frac{\mathbb{E}_{p(\boldsymbol{g}, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi, \boldsymbol{x}' | \mathcal{D})}\left[\pi(\boldsymbol{x} \,|\, \boldsymbol{\psi}_\pi)\ \min\left(1, \frac{\Phi(g(\boldsymbol{x}))}{\Phi(g(\boldsymbol{x}'))}\right)\right]}{\mathbb{E}_{p(\boldsymbol{g}, \boldsymbol{\theta}, \boldsymbol{\psi}_\pi | \mathcal{D}, \boldsymbol{x})}\left[\mathbb{E}_{\pi(\boldsymbol{x}' \,|\, \boldsymbol{\psi}_\pi)}\left[\min\left(1, \frac{\Phi(g(\boldsymbol{x}'))}{\Phi(g(\boldsymbol{x}))}\right)\right]\right]}. \tag{7.9}$$

The numerator can be estimated directly as part of the Markov chain Monte Carlo inference. After each Markov step, generate a predictive sample $\boldsymbol{x}'$ and record the



(a) Probability density function



(b) Latent functions that lead to the same PDF

Figure 7.1: A demonstration of redundancy in the map between realisations from the Gaussian process prior and probability density functions. The base density is a standard normal. (a) A probability density function. (b) Three Φ-squashed functions that all result in the probability density function shown in (a).

transition probabilities. The denominator requires a Markov chain to be run with a data set augmented by the predictive location $\boldsymbol{x}$. At each step in the Markov chain, a sample $\boldsymbol{x}'$ is generated from the base density and the transition probabilities are evaluated.

7.2 Improving Performance of Inference in the GPDS

The major bottleneck for inference in the Gaussian process density sampler is the cubic time complexity of the linear algebraic computations necessary for the Gaussian process. While the number of data are fixed, the latent history algorithm samples from the posterior distribution on the number of latent rejections. These latent rejections contribute to the cubic scaling, and one way to improve performance of the inference in the GPDS would be to limit the posterior probability assigned to latent histories with many rejections. It would be preferable to do this without compromising the correctness of the inference algorithm or requiring a finite-dimensional approximation. To this end, it is significant to note that multiple latent functions $g(\boldsymbol{x})$, when pushed through the transformation of Equation 2.1, lead to the same density $f(\boldsymbol{x})$. Figure 7.1 provides an example of several latent functions yielding the same density. Figure 7.1b shows three Φ-squashed functions, and Figure 7.1a shows the probability density function to which they all lead.

Consider the result of applying Algorithm 2.2 to the red function in Figure 7.1b.

Roughly, the red line hovers around 0.1, which means that nine out of ten proposals will be rejected when generating data. If we had data from the density in Figure 7.1a, then when performing latent history inference we would sometimes consider the hypothesis of the red line, as its likelihood is the same as that of the blue and green lines. However, latent history inference would require about nine times as many rejections as data in order to explain the red line. This many rejections would combine with the cubic computational complexity of the Gaussian process to make inference expensive. We would prefer a stronger prior than the vanilla Gaussian process, in order to remove this kind of redundancy and instead cause the model to choose functions like the blue and green line over the red line.

One way to achieve this is to use a *tied* Gaussian process prior. In the tied GP, I add a new "hyperparameter" to the Gaussian process that is the location of a point at which the function $g(\boldsymbol{x})$ is forced to be zero. That is, I introduce a hyperparameter $\boldsymbol{x}_{\mathsf{tied}}$ and require that $g(\boldsymbol{x}_{\mathsf{tied}}) = 0$. I use $\pi(\boldsymbol{x})$ as the prior on $\boldsymbol{x}_{\mathsf{tied}}$, and include it in the Markov chain Monte Carlo inference. This additional constraint does not seriously compromise the Gaussian process prior, but helps prevent functions such as the red one in Figure 7.1b from appearing during inference. Note that in Figure 7.1b, with a tied Gaussian process, both the blue and green lines are still good hypotheses, as their latent functions both include an $\boldsymbol{x}$ where $g(\boldsymbol{x}) = 0$, i.e. where $\Phi(g(\boldsymbol{x})) = \frac{1}{2}$.

7.3 Improving Performance of Inference in the SGCP

Modeling a large number of latent thinned events in the sigmoidal Gaussian Cox process presents similar difficulties to the latent rejections in the Gaussian process density sampler. However, latent thinned events in the SGCP are not due to redundancy in the function mapping, but to the dominating intensity $\bar{\lambda}(\boldsymbol{x})$. If $\bar{\lambda}(\boldsymbol{x})$ is a very loose bound on the true intensity $\lambda(\boldsymbol{x})$, then the latent history method will require many thinned events to explain the data. As with the latent rejections, cubic time complexity of computations in the Gaussian process make undesirable the inclusion of superfluous latent thinned events. Preventing very loose $\bar{\lambda}(\boldsymbol{x})$ is a question of choice of hyperparameters $\boldsymbol{\psi}_\lambda$. For the simple case of a constant dominating intensity $\lambda_\star$, a weak gamma prior may not be enough to prevent the occasional departure into loose-bound territory. In such cases, it may be desirable to use a truncated gamma prior that only offers support for $\lambda_\star$ up to some limit. This has the practical effect of placing a soft upper-bound on the number of latent thinned events that the latent history method could be required to model.

7.4 Summary

This chapter examined a few peripheral topics related to inference in the Gaussian process density sampler and the sigmoidal Gaussian Cox process. I presented a method for estimating the value of the normalised density under the GPDS. I also discussed ways to prevent the GPDS and SGCP models from incorporating large quantities of latent data.

Chapter 8

Archipelago: Nonparametric Bayesian Semi-Supervised Learning

This chapter presents a nonparametric Bayesian method for solving the problem of *semi-supervised learning*. I extend the Gaussian process density sampler of Chapter 2 to model multiple, coupled densities in a classifier and call this new model *Archipelago*. I extend the latent history inference method of Chapter 5 for this new approach and apply it to both synthetic and real-world data.

8.1 The Semi-Supervised Learning Problem

Semi-supervised learning (SSL) algorithms solve the problem of classification under the circumstance that only a subset of the training data is labeled. In contrast to the purely-supervised setting, semi-supervised learning assumes that the probability density of the data is important to discovering the decision boundary. Semi-supervised learning is motivated by the situation where copious training data are available, but hand-labeling the data is expensive.

From a Bayesian perspective, the natural way to perform semi-supervised learning is with a generative model of the data. We explicitly model the densities that gave rise to the observations, integrating over the class assignments for unlabeled data. We can then integrate over any parameters of the density model and arrive at predictive classifications of unseen data.

We would like to perform this kind of modeling while making the fewest possible assumptions about the densities in question. Nonparametric Bayesian methods are appealing in this regard, as they incorporate an infinite number of parameters into the model. This prevents us from having to make difficult choices about dimensionality.

Unfortunately, the nature of many nonparametric Bayesian density models makes

them ill-suited for the semi-supervised setting. Specifically, the main assumption in SSL is that data of the same class will cluster together; the labeled data provide the appropriate class and the unlabeled data determine the cluster boundary. The most common nonparametric Bayesian density model, the Dirichlet process mixture model (DPMM) (Escobar and West, 1995; Rasmussen, 2000), suffers from the problem that the mixture components are located independently. There is no tendency for mixture models to form contiguous densities. If we take the natural approach of using a DPMM to flexibly model each class density in a semi-supervised learning problem, the mixtures will grab unlabeled data away from other classes.

Due to these difficulties with specifying a flexible density model, discriminative methods are more common than generative models for semi-supervised learning. The Bayesian discriminative model of Lawrence and Jordan (2005) takes advantage of the Gaussian process to construct a nonparametric model for binary SSL. Similarly, Chu et al. (2007) and Sindhwani et al. (2007) also specify nonparametric Bayesian discriminative models with Gaussian processes that exploit graph-based side information.

In this chapter, I extend the Gaussian process density sampler model of Chapter 2 to address the semi-supervised learning problem. This results in a fully-Bayesian generative approach to semi-supervised learning that uses Gaussian processes to specify the prior on class densities. I call the model *Archipelago* as it performs Bayesian clustering with infinite-dimensional density models, but it prefers contiguous densities. These clusters can form irregular shapes, like islands in a chain.

8.2 The Archipelago Model

As in previous chapters, I consider a model for data that live in a space $\mathcal{X}$. There are K possible discrete labels for these data. I condition on K, under the assumption that, when performing inference, at least one of each class has been observed. Let $\boldsymbol{x}$ be a point in $\mathcal{X}$. I define K scalar functions (one for each class) $\{g_k(\boldsymbol{x})\}_{k=1}^K$, where $g_k(\cdot) : \mathcal{X} \to \mathbb{R}$. Conditioned on a point $\boldsymbol{x}$, these functions are used with a softmax function to construct a categorical distribution for $c \in 1, 2, \ldots, K$, the class label of $\boldsymbol{x}$:

$$p(c \,|\, \boldsymbol{x}, \{g_k(\boldsymbol{x})\}_{k=1}^K) = \frac{\exp\{g_c(\boldsymbol{x})\}}{\Omega(\boldsymbol{x})}, \tag{8.1}$$

where I use $\Omega(\boldsymbol{x}) = \sum_{k=1}^K \exp\{g_k(\boldsymbol{x})\}$ for notational convenience. I combine this discriminative classifier with a density model for $\boldsymbol{x}$ that is also constructed from the $g_k(\boldsymbol{x})$:

$$p(\boldsymbol{x} \,|\, \{g_k(\boldsymbol{x})\}_{k=1}^K) = \frac{1}{\mathcal{Z}_\pi[\boldsymbol{\Omega}]} \frac{\Omega(\boldsymbol{x})\ \pi(\boldsymbol{x})}{1 + \Omega(\boldsymbol{x})} \tag{8.2}$$

where $\pi(\boldsymbol{x})$ is the base density as in the Gaussian process density sampler. The constant $\mathcal{Z}_\pi[\boldsymbol{\Omega}]$ is the normalisation given by

$$\mathcal{Z}_\pi[\boldsymbol{\Omega}] = \int_{\mathcal{X}} \mathrm{d}\boldsymbol{x}\ \frac{\Omega(\boldsymbol{x})\ \pi(\boldsymbol{x})}{1+\Omega(\boldsymbol{x})}. \tag{8.3}$$

Multiplying Equation 8.1 by Equation 8.2 constructs a joint distribution for the location and label together, given the functions $\{g_k(\boldsymbol{x})\}_{k=1}^K$:

$$p(\boldsymbol{x}, c \,|\, \{g_k(\boldsymbol{x})\}_{k=1}^K) = \frac{1}{\mathcal{Z}_\pi[\boldsymbol{\Omega}]}\ \frac{\exp\{g_c(\boldsymbol{x})\}\ \pi(\boldsymbol{x})}{1+\Omega(\boldsymbol{x})}. \tag{8.4}$$

From the point of view of the semi-supervised learning problem, this construction is appealing because Equation 8.2 can be viewed as marginalising out the class label in Equation 8.4. This formulation is in contrast to typical Bayesian generative methods for SSL which would construct $p(\boldsymbol{x}, c)$ from $p(\boldsymbol{x} \,|\, c)\, p(c)$. While it is easy to marginalise out the class label c in $p(c \,|\, \boldsymbol{x})\, p(\boldsymbol{x})$, it is not necessarily so easy for $p(\boldsymbol{x} \,|\, c)\, p(c)$. The result of the Archipelago construction is a set of K *coupled densities* which are each intractable individually, but whose sum is tractable.

Introducing K independent Gaussian process priors on the functions $\{g_k(\boldsymbol{x})\}_{k=1}^K$ makes the model nonparametric. As in the Gaussian process density sampler, there may also be Gaussian process hyperparameters $\boldsymbol{\theta}$ governing the covariance function, and base density hyperparameters $\boldsymbol{\psi}_\pi$.

8.3 Generating Data from the Prior

The significance of coupling the likelihood of Equation 8.4 with Gaussian process priors on the $\{g_k(\boldsymbol{x})\}_{k=1}^K$ is that it is possible to define a fully-generative process for arriving at labeled data from a set of K random coupled densities. In other words, it is possible to define a process that is equivalent to drawing a set of data from the density in Equation 8.2 with a random $\Omega(x)$ arising from the Gaussian process priors on the $\{g_k(\boldsymbol{x})\}_{k=1}^K$, and then labeling that data according to Equation 8.1.

To generate a labeled datum, first draw a proposal $\tilde{\boldsymbol{x}}$ from the base density $\pi(\boldsymbol{x})$. Then draw a function value from each of the Gaussian processes at at the point $\tilde{\boldsymbol{x}}$. I denote these function draws as $\{g_k(\tilde{\boldsymbol{x}})\}_{k=1}^K$. Use the function values to partition the unit interval $[0, 1]$ into $K+1$ non-overlapping segments with boundaries $0, \xi_1, \xi_2, \ldots, \xi_K, 1$ where

$$\xi_k = \xi_{k-1} + \frac{\exp\{g_k(\tilde{\boldsymbol{x}})\}}{1+\Omega(\tilde{\boldsymbol{x}})} \tag{8.5}$$

and $\xi_0 = 0$. Draw a uniform random variate u on $(0, 1)$ and either assign the datum the label k if it falls into the kth partition, i.e. $\xi_{k-1} < u < \xi_k$, or reject the proposal if it is in

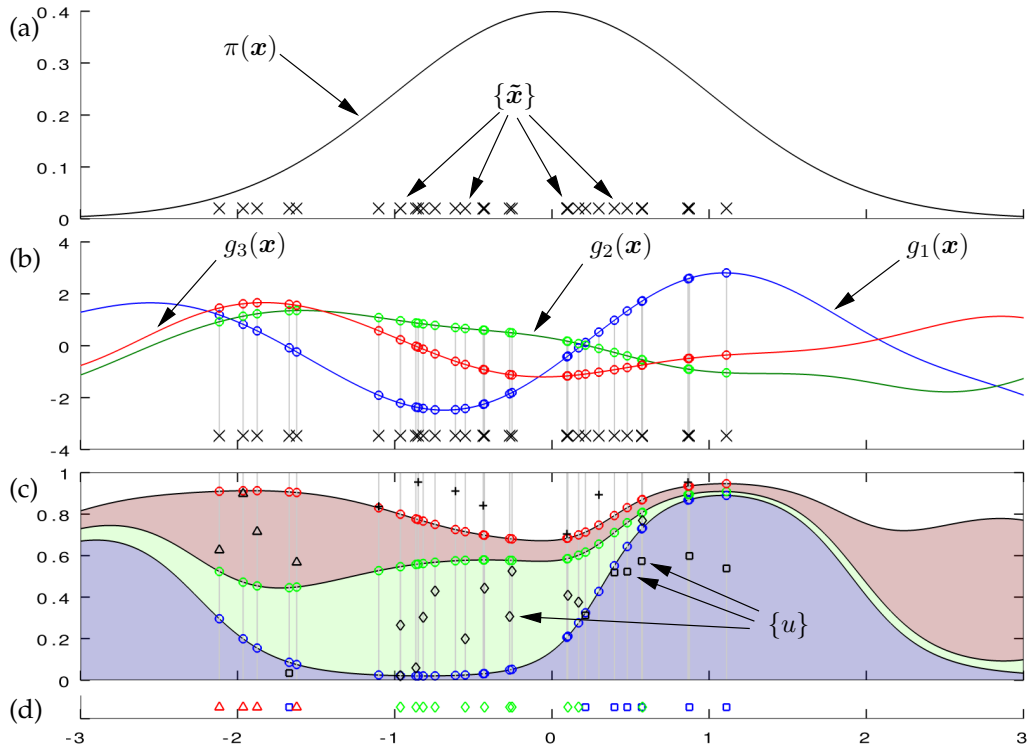


Figure 8.1: The generative process for Archipelago in one dimension. (a) Proposals are drawn from the base density $\pi(\boldsymbol{x})$, which here is a Gaussian with zero mean and unit variance. (b) Next, each of the $K = 3$ functions is sampled from the Gaussian process at the proposal locations. (c) The functions are used to partition the unit interval into $K + 1$ slices at each proposal location and uniform variates u are drawn in the vertical coordinate. (d) If u falls into the topmost partition, the proposal is rejected. Otherwise, it is accepted and assigned the class label based on the partition in to which u falls.

the topmost partition, i.e. $u \geq \xi_K$. If the $\tilde{\boldsymbol{x}}$ is rejected, make another proposal and continue until an acceptance. When making these new proposals, sample conditionally from the Gaussian process, incorporating the function draws from previous proposals. As in the Gaussian process density sampler, I refer to these as the K *conditioning sets*, with inputs $\boldsymbol{X}_k$ and outputs $\boldsymbol{G}_k$. Continue this rejection/acceptance process, generating as many data as necessary, "discovering" the set of functions $\{g_k(\boldsymbol{x})\}_{k=1}^K$ during the iterations. This procedure is shown as pseudocode in Algorithm 8.1 and graphically in Figure 8.1.

This generative procedure is infinitely exchangeable and the data are exact, just as in the Gaussian process density sampler. It is also not necessary to determine the $\{g_k(\boldsymbol{x})\}_{k=1}^K$ over any more than a finite number of points in $\mathcal{X}$, nor is it necessary to determine the normalisation constant $\mathcal{Z}_\pi[\boldsymbol{\Omega}]$ in order to generate these data.

Algorithm 8.1 Generate N labeled data from the Archipelago model

Inputs:
- Number of samples to draw N
- Gaussian process covariance function $C(\boldsymbol{x}, \boldsymbol{x}' ; \boldsymbol{\theta})$
- Gaussian process hyperparameters $\{\boldsymbol{\theta}_k\}_{k=1}^K$
- Base density $\pi(\boldsymbol{x} \mid \boldsymbol{\psi}_\pi)$

Outputs:
- N labeled data $\mathcal{D} = \{\boldsymbol{x}_n, c_n\}_{n=1}^N$ from a density drawn from the prior.

for $k \leftarrow 1 \ldots K$ **do**
  $\boldsymbol{X}_k \leftarrow \emptyset$, $\boldsymbol{G}_k \leftarrow \emptyset$ ▷ Initialise K empty conditioning sets.
end for
$\mathcal{D} \leftarrow \emptyset$ ▷ Initialise the set to be returned.
repeat ▷ Run the rejection sampler.
  $\tilde{\boldsymbol{x}} \sim \pi(\boldsymbol{x} \mid \boldsymbol{\psi}_\pi)$ ▷ Draw a proposal from the base density.
  $\Omega(\tilde{\boldsymbol{x}}) \leftarrow 0$ ▷ Initialise the sum of functions.
  for $k \leftarrow 1 \ldots K$ **do** ▷ Loop over the K functions.
    $g_k(\tilde{\boldsymbol{x}}) \sim \mathcal{GP}(g \mid \tilde{\boldsymbol{x}}, \boldsymbol{X}_k, \boldsymbol{G}_k, \boldsymbol{\theta}_k)$ ▷ Draw the function from the GP.
    $\Omega(\tilde{\boldsymbol{x}}) \leftarrow \Omega(\tilde{\boldsymbol{x}}) + \exp\{g_k(\tilde{\boldsymbol{x}})\}$ ▷ Update the total.
    $\boldsymbol{X}_k \leftarrow \boldsymbol{X}_k \cup \tilde{\boldsymbol{x}}$, $\boldsymbol{G}_k \leftarrow \boldsymbol{G}_k \cup g_k(\tilde{\boldsymbol{x}})$ ▷ Update the conditioning sets.
  end for
  $u \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform variate on $(0, 1)$.
  $\xi_0 \leftarrow 0$ ▷ Initialise the partitions.
  for $k \leftarrow 1 \ldots K$ **do** ▷ Loop over the partitions.
    $\xi_k \leftarrow \xi_{k-1} + \dfrac{\exp\{g_k(\tilde{\boldsymbol{x}})\}}{1 + \Omega(\tilde{\boldsymbol{x}})}$ ▷ Calculate the next partition.
    if $u < \xi_k$ **then** ▷ Test if the variate falls into this partition.
      $\mathcal{D} \leftarrow \mathcal{D} \cup \{\tilde{\boldsymbol{x}}, k\}$ ▷ Accept this proposal and store the label.
      break ▷ Break out of the loop.
    end if
  end for
until $|\mathcal{D}| = N$ ▷ Loop until N proposals are accepted.
return $\mathcal{D}$

8.4 Latent History Inference in Archipelago

I have so far described a nonparametric generative model for labeled data. I now show how this model can be used to perform semi-supervised learning. The semi-supervised inference task in Archipelago is to find the predictive distribution over the class label of some unlabeled datum $\boldsymbol{x}_\star$, given N labeled data $\{\boldsymbol{x}_n, c_n\}_{n=1}^N$ and J unlabeled data $\{\boldsymbol{x}_j\}_{j=1}^J$, marginalising out the unknown latent functions $\{g_k(\boldsymbol{x})\}_{k=1}^K$. This is the *inductive* interpretation of semi-supervised learning: $\boldsymbol{x}_\star$ is not known in advance and is not included in the unlabeled data for inference. I mention this to contrast Archipelago with the more restricted class of *transductive* models, which attempt to find the correct class labels for the J unlabeled examples. If, as in previous chapters, I treat the function $g_k(\boldsymbol{x})$ as an infinite vector $\boldsymbol{g}_k$, then the objective is to integrate out these K vectors:

$$p(c_\star \mid \boldsymbol{x}_\star, \{\boldsymbol{x}_n, c_n\}_{n=1}^N, \{\boldsymbol{x}_j\}_{j=1}^J, \{\boldsymbol{\theta}_k\}_{k=1}^K) = \int \mathrm{d}\boldsymbol{g}_1 \cdots \int \mathrm{d}\boldsymbol{g}_K \; p(c_\star \mid \boldsymbol{x}_\star, \{\boldsymbol{g}_k\}_{k=1}^K) \; p(\{\boldsymbol{g}_k\}_{k=1}^K \mid \{\boldsymbol{x}_n, c_n\}_{n=1}^N, \{\boldsymbol{x}_j\}_{j=1}^J, \{\boldsymbol{\theta}_k\}_{k=1}^K) \quad (8.6)$$

where the posterior distribution on $\{\boldsymbol{g}_k\}_{k=1}^K$ arises from Equation 8.2 and Equation 8.4 along with the Gaussian process priors and is proportional to

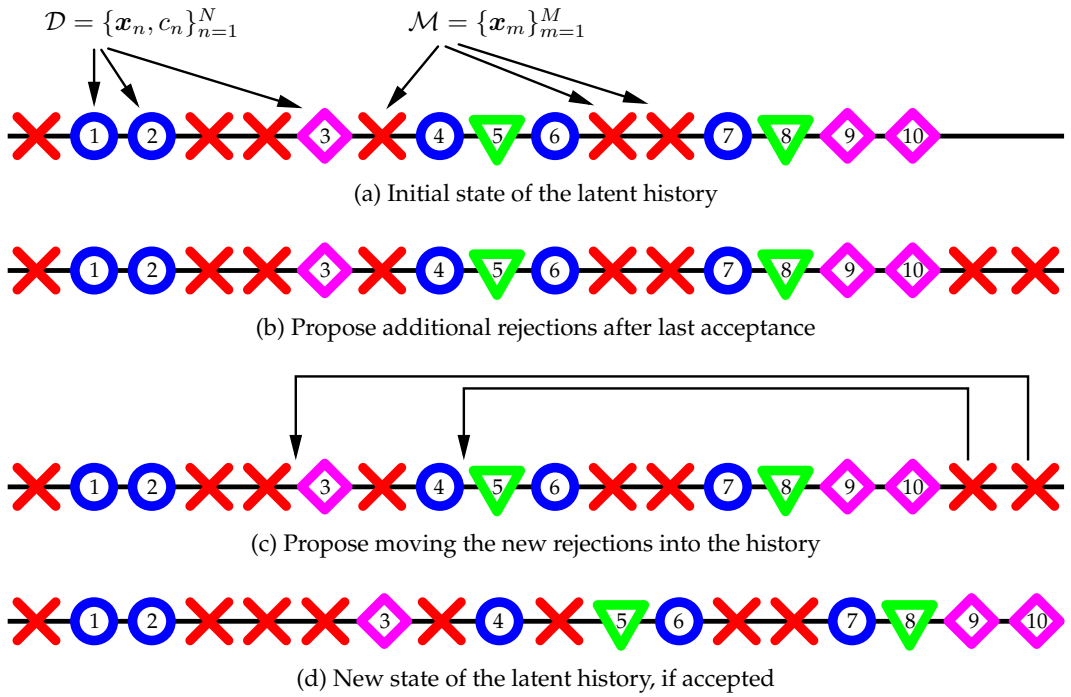
$$p(\{\boldsymbol{g}_k\}_{k=1}^K, \{\boldsymbol{x}_n, c_n\}_{n=1}^N, \{\boldsymbol{x}_j\}_{j=1}^J \mid \{\boldsymbol{\theta}_k\}_{k=1}^K) = \mathcal{Z}_\pi[\boldsymbol{\Omega}]^{-N-J} \prod_{k=1}^K \mathcal{GP}(\boldsymbol{g}_k \mid \{\boldsymbol{x}_n\}_{n=1}^N, \{\boldsymbol{x}_j\}_{j=1}^J, \boldsymbol{\theta}_k) \times \prod_{n=1}^N \frac{\exp\{g_{c_n}(\boldsymbol{x}_n)\}\ \pi(\boldsymbol{x}_n)}{1+\Omega(\boldsymbol{x}_n)} \prod_{j=1}^J \frac{\Omega(\boldsymbol{x}_j)\ \pi(\boldsymbol{x}_j)}{1+\Omega(\boldsymbol{x}_j)}. \quad (8.7)$$

Given a set of samples of $\{g_k(\boldsymbol{x}_\star)\}_{k=1}^K$, one can estimate the distribution in Equation 8.6 via a sum, but acquiring samples from the posterior on the $\boldsymbol{g}_k$ is difficult because it is doubly-intractable. As in Chapter 5 for the Gaussian process density sampler and the sigmoidal Gaussian Cox process, I construct a Markov chain Monte Carlo inference algorithm on the latent history of the generative process. The unknowns in this case are 1) the number of latent rejections M; 2) the locations of the latent rejections $\mathcal{M} = \{\boldsymbol{x}_m\}_{m=1}^M$; and 3) the values of the functions at the rejections, at the labeled data and at the unlabeled data. The true labels of the unlabeled data are also unknown, but these quantities are easily marginalised over. As it is possible to run the generative algorithm without calculating $\mathcal{Z}_\pi[\boldsymbol{\Omega}]$ and without knowing the functions $g_k(\boldsymbol{x})$ everywhere, it will be possible to construct a latent history model that inherits these properties for inference.

Integrating out the classes of the unlabeled data, as in Equation 8.2, the joint distribution over the function values, the number and location of the latent rejections, and the fixed data is

$$p(\{\boldsymbol{g}_k\}_{k=1}^K, \{\boldsymbol{x}_m\}_{m=1}^M, \{\boldsymbol{x}_j\}_{j=1}^J, \{\boldsymbol{x}_n, c_n\}_{n=1}^N \mid \{\boldsymbol{\theta}_k\}_{k=1}^K, \boldsymbol{\psi}_\pi) = \prod_{k=1}^K \mathcal{GP}(\{g_k(\boldsymbol{x}_n)\}_{n=1}^N, \{g_k(\boldsymbol{x}_j)\}_{j=1}^J, \{g_k(\boldsymbol{x}_m)\}_{m=1}^M \mid \{\boldsymbol{x}_m\}_{m=1}^M, \{\boldsymbol{x}_j\}_{j=1}^J, \{\boldsymbol{x}_n\}_{n=1}^N, \boldsymbol{\theta}_k) \times \prod_{n=1}^N \frac{\exp\{g_{c_n}(\boldsymbol{x}_n)\}\ \pi(\boldsymbol{x}_n \mid \boldsymbol{\psi}_\pi)}{1+\Omega(\boldsymbol{x}_n)} \prod_{j=1}^J \frac{\Omega(\boldsymbol{x}_j)\ \pi(\boldsymbol{x}_j \mid \boldsymbol{\psi}_\pi)}{1+\Omega(\boldsymbol{x}_j)} \prod_{m=1}^M \frac{\pi(\boldsymbol{x}_m \mid \boldsymbol{\psi}_\pi)}{1+\Omega(\boldsymbol{x}_m)}. \quad (8.8)$$

The joint distribution in Equation 8.8 is proportional to the posterior distribution over the latent history, and I sample from it with three kinds of Markov transitions: updating the number of latent rejections, updating the rejection locations, and updating the function values. This is very similar to the latent history Markov chain Monte Carlo discussed in Chapter 5 for the GPDS and SGCP.



(a) Initial state of the latent history

(b) Propose additional rejections after last acceptance

(c) Propose moving the new rejections into the history

(d) New state of the latent history, if accepted

Figure 8.2: Inserting new latent rejections into the Archipelago latent history. (a) There are six latent rejections and ten labeled data, with $K = 3$, represented as blue circles, magenta diamonds and green triangles. The rejections are in red. (b) Two new rejections are proposed after the last labeled datum. (c) It is proposed to move the two rejections into randomly-selected times in the history. (d) If accepted, there is a new history with eight latent rejections.

8.4.1 Sampling the Number of Latent Rejections

I take advantage of the infinite exchangeability of the generative process to construct a Metropolis–Hastings move that can add or remove rejections from the latent history. The model for inference is that the data came about as the result of Algorithm 8.1, stopping when there were $N + J$ data accepted. The unlabeled data were generated by the J labels being discarded, and the joint distribution in Equation 8.8 integrates out the unknown label. Due to infinite exchangeability, a reordering of the latent history can be proposed at any time and this move will always be accepted. Thus to insert a new latent rejection, propose moving a rejection that occurred after the $N+J$th acceptance to be moved to sometime before it. This idea is shown graphically in Figure 8.2. To delete a rejection, make the opposite type of transition: select one of the current rejections at random and propose moving it to after the $N+J$th acceptance.

As in the Gaussian process density sampler it is not necessary in practice to actually maintain an ordering. Use the function $\zeta(M, N + J)$, introduced in Section 5.2.1, as the Bernoulli probability of making an insertion proposal versus a deletion proposal. When deciding to add an additional rejection, first propose a new location for it in $\mathcal{X}$, denoted $\hat{\boldsymbol{x}}$, by drawing it from $\pi(\boldsymbol{x} \,|\, \boldsymbol{\psi}_\pi)$. Then draw each of the K function values at that location, $\{g_k(\hat{\boldsymbol{x}})\}_{k=1}^K$, conditioning on all of the current function values. Finally,

accept or reject with Metropolis–Hastings acceptance ratio

$$a_{\mathsf{arch-ins}} = \frac{(1 - \zeta(M+1, N+J))\ (M+N+J)}{\zeta(M, N+J)\ (M+1)\ (1+\Omega(\hat{\boldsymbol{x}}))}. \tag{8.9}$$

If proposing a deletion, choose an index m uniformly from among the M rejections and remove the mth rejection with Metropolis–Hastings acceptance ratio

$$a_{\mathsf{arch-del}} = \frac{\zeta(M-1, N+J)\ M\ (1+\Omega(\boldsymbol{x}_m))}{(1 - \zeta(M, N+J))\ (M+N+J-1)}. \tag{8.10}$$

As in Chapter 5, in practice it is often reasonable to simply set $\zeta(M, N+J) = \frac{1}{2}$. More sophisticated proposals may yield improvements in mixing, but this is a topic for future study. Typically, I make several of these transitions (≈ 10) for each of the transitions in Section 8.4.2 and Section 8.4.3.

8.4.2 Sampling the Locations of Latent Rejections

Conditioned on the current function values and the number of rejections M, it is also necessary to update the locations of the rejections. Iterate over each of the M rejections and make a Metropolis–Hastings proposal with two stages. For the mth rejection, first draw a new location $\hat{\boldsymbol{x}}_m$ from a proposal density $q(\hat{\boldsymbol{x}}_m \leftarrow \boldsymbol{x}_m)$. Second, draw a new set of K function values from the Gaussian process at this location, conditioning on the rest of the latent history, yielding $\{g_k(\hat{\boldsymbol{x}}_m)\}_{k=1}^K$. Accept or reject the proposal according to the acceptance ratio

$$a_m = \frac{q(\boldsymbol{x}_m \leftarrow \hat{\boldsymbol{x}}_m)\ \pi(\hat{\boldsymbol{x}}_m \,|\, \boldsymbol{\psi}_\pi)\ (1+\Omega(\boldsymbol{x}_m))}{q(\hat{\boldsymbol{x}}_m \leftarrow \boldsymbol{x}_m)\ \pi(\boldsymbol{x}_m \,|\, \boldsymbol{\psi}_\pi)\ (1+\Omega(\hat{\boldsymbol{x}}_m))}. \tag{8.11}$$

8.4.3 Sampling the Latent Functions

To sample from the latent functions, iterate over each of the K functions and use Hamiltonian Monte Carlo (Neal, 1997) for efficient sampling, as in Section 5.2.3. For the kth function, the log conditional posterior distribution is

$$\begin{aligned}
&\ln p(\{g_k(\boldsymbol{x}_n)\}_{n=1}^N, \{g_k(\boldsymbol{x}_j)\}_{j=1}^J, \{g_k(\boldsymbol{x}_m)\}_{m=1}^M \,|\, \{\boldsymbol{x}_m\}_{m=1}^M, \{\boldsymbol{x}_n, c_n\}_{n=1}^N, \{\boldsymbol{x}_j\}_{j=1}^J, \theta_k) = \\
&\quad \ln \mathcal{GP}(\{g_k(\boldsymbol{x}_n)\}_{n=1}^N, \{g_k(\boldsymbol{x}_j)\}_{j=1}^J, \{g_k(\boldsymbol{x}_m)\}_{m=1}^M \,|\, \{\boldsymbol{x}_m\}_{m=1}^M, \{\boldsymbol{x}_n\}_{n=1}^N, \{\boldsymbol{x}_j\}_{j=1}^J, \theta_k) \\
&+ \sum_{n=1}^N [\delta(c_n, k)\, g_k(\boldsymbol{x}_n) - \ln(1+\Omega(\boldsymbol{x}_n))] + \sum_{j=1}^J \ln \frac{\Omega(\boldsymbol{x}_j)}{1+\Omega(\boldsymbol{x}_j)} - \sum_{m=1}^M \ln(1+\Omega(\boldsymbol{x}_m)) + \text{const},
\end{aligned} \tag{8.12}$$

where $\delta(\cdot, \cdot)$ is the Kronecker delta function.

8.4.4 Sampling the Hyperparameters

An appealing feature of Bayesian inference methods is the ability to perform hierarchical inference. In this case, it is desirable to sample from the hyperparameters, $\{\boldsymbol{\theta}_k\}_{k=1}^K$, governing the Gaussian process covariance functions. Conditioned on the number and locations of the rejections, and on the function values, this is done as described in Section 5.2.4. Sampling from the base density hyperparameters is performed as in Section 5.2.5.

8.4.5 Making Predictions

With samples from the posterior distribution over the function values, the integral in Equation 8.6 can now be approximated. At each Markov chain Monte Carlo transition after burn-in, sample the function values $\{g_k(\boldsymbol{x}_\star)\}_{k=1}^K$ from the Gaussian process, conditioned on the current state of the history. From these values, the predictive distribution can be approximated via a mixture of categorical distributions arising from the softmax functions.

We might also be interested in the class-conditional predictive distributions. These K distributions are the ones that arise on data space, conditioning on membership in class k, but integrating out the latent function and hyperparameters. While not available in closed form, it is straightforward to generate predictive fantasies. At each MCMC step after burn-in, run the generative process forward from the current state until a datum is accepted that is a member of class k.

8.5 Empirical Results

I now compare the Archipelago model and inference method to three other multiclass Gaussian process classification approaches: the standard softmax GP classifier (Williams and Barber, 1998), the probit classifier of Girolami and Rogers (2006), and the multiclass extension (Rogers and Girolami, 2007) of the Null Category Noise Model of Lawrence and Jordan (2005). I present comparisons of these models on three two-dimensional toy problems and two real-world datasets. Note that unlabeled data were ignored in the softmax and probit models.

8.5.1 Toy Pinwheel Data

The toy problems are noisy pinwheels with three, four and five classes, shown in Figure 8.3a, Figure 8.3d, and Figure 8.3g, respectively. There are 50 data points in each set and the algorithms were run with 1, 2, 4, and 8 labeled data in each class. 300 held-out data for each class were used to evaluate the predictive distributions by error rate

and perplexity. Inference for each method was performed using Markov chain Monte Carlo, and each used an isotropic squared-exponential covariance function. For each method I also sampled from the length-scale and amplitude hyperparameters using Hamiltonian Monte Carlo. The Archipelago base density was a bivariate Gaussian. Figure 8.3 shows the predictive modes for the Archipelago and softmax models, as well as the entropy of the Archipelago marginal predictive distribution as a function of space. Numerical results are in Table 8.1.

8.5.2 Wine Data

The wine data are thirteen-dimensional, with three classes from Aeberhard et al. (1992) via Asuncion and Newman (2007). As in Rogers and Girolami (2007), I translated and scaled the inputs to have zero mean and unit variance. I used an automatic relevance determination (ARD) covariance function (Rasmussen and Williams, 2006) and HMC for inference of length scales and amplitude. I divided the data into two sets of 89 for training and testing, with the classes divided approximately evenly. I ran four sets of experiments with each method, as with the toy data, with 1, 2, 4, and 8 labeled data in each class. I used a multivariate Gaussian for the Archipelago base density. The results are given in Table 8.1.

8.5.3 Oil Pipe Data

The oil pipe data are included with software for Williams and Barber (1998). The task is to classify the type of flow in a pipe (laminar, hydrogenous, or amorphous), using a set of fourteen gamma ray readings. I split the 400 data into 134 training and 266 testing. I used an ARD covariance function and followed the same procedures as in Section 8.5.2. The results are given in Table 8.1.

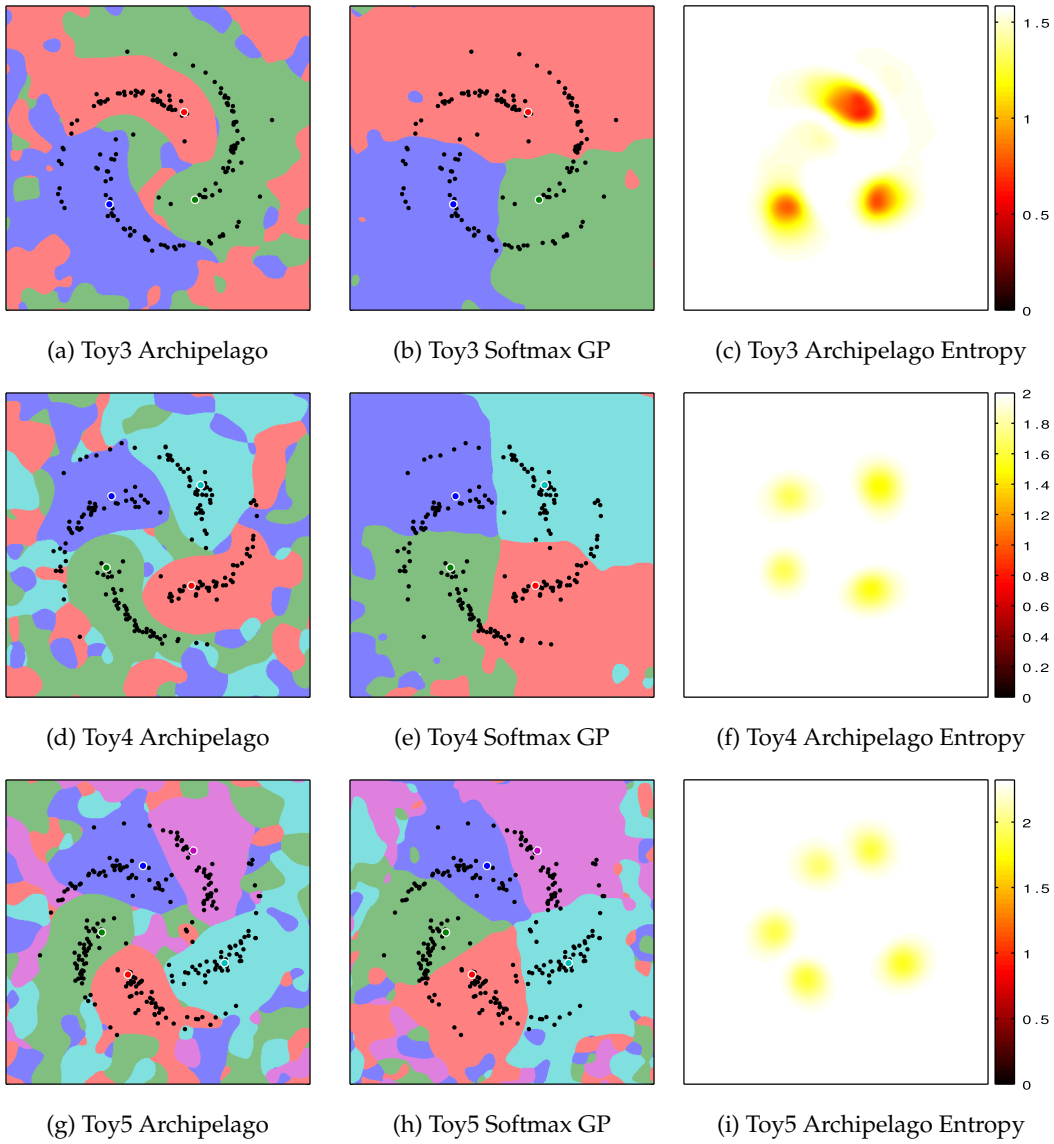
8.6 Discussion

Archipelago combines ideas both from Gaussian process density modeling and Gaussian process classification into a single model. When using a kernel such as the squared-exponential, it is expressing the idea that similar data should have similar probabilities *and* have similar class assignments. In contrast, a semi-supervised learning approach with a mixture model for each class must maintain both a class assignment and a mixture component assignment for each datum, and the notion of smoothness does not extend beyond the simple parametric form of the components.

The most relevant Bayesian model to Archipelago is the discriminative Null Category Noise Model (NCNM) (Lawrence and Jordan, 2005) and the multi-class extension of Rogers and Girolami (2007). The NCNM enforces the idea of "margin" in a

			Archipelago	Softmax GP	Probit GP	NCNM
Toy3	1	error	0.167	0.347	0.290	0.612
		perplexity	1.461	2.588	1.631	14.120
	2	error	0.111	0.312	0.279	0.462
		perplexity	1.565	2.769	1.350	4.801
	4	error	0.042	0.118	0.142	0.114
		perplexity	1.092	1.310	1.041	1.002
	8	error	0.037	0.056	0.137	0.070
		perplexity	1.070	1.065	1.043	1.000
Toy4	1	error	0.120	0.305	0.416	0.554
		perplexity	2.478	3.436	1.239	16.160
	2	error	0.092	0.188	0.253	0.429
		perplexity	2.471	2.861	7.896	8.257
	4	error	0.032	0.119	0.077	0.056
		perplexity	1.469	1.505	1.343	1.113
	8	error	0.025	0.061	0.088	0.0533
		perplexity	1.135	1.076	1.086	1.009
Toy5	1	error	0.183	0.192	0.515	0.482
		perplexity	3.766	4.519	3.533	294.758
	2	error	0.124	0.181	0.178	0.365
		perplexity	2.811	2.008	1.476	11.669
	4	error	0.081	0.159	0.113	0.093
		perplexity	2.345	1.730	1.375	1.210
	8	error	0.051	0.095	0.073	0.051
		perplexity	2.105	1.290	1.201	1.093
Wine	1	error	0.141	0.260	0.303	0.393
		perplexity	1.928	2.790	1.769	31.832
	2	error	0.135	0.292	0.213	0.393
		perplexity	2.654	2.760	2.851	2.799
	4	error	0.090	0.146	0.101	0.101
		perplexity	1.198	2.440	1.387	1.153
	8	error	0.090	0.067	0.067	0.067
		perplexity	1.084	1.250	1.224	1.045
Oil Pipe	1	error	0.538	0.568	0.432	0.650
		perplexity	2.997	3.024	5.631	3.56K
	2	error	0.297	0.331	0.308	0.365
		perplexity	2.451	2.576	3.499	1.024
	4	error	0.211	0.215	0.271	0.176
		perplexity	1.658	1.864	2.558	1.301
	8	error	0.068	0.053	0.068	0.038
		perplexity	1.241	1.137	1.232	1.002

Table 8.1: Results from four methods applied to test data: Archipelago, the softmax GP, the probit GP and the Null Category Noise Model (NCNM). Six problems were evaluated, with 1, 2, 4, and 8 labeled data per class. Test-set error and perplexity are shown.



(a) Toy3 Archipelago (b) Toy3 Softmax GP (c) Toy3 Archipelago Entropy

(d) Toy4 Archipelago (e) Toy4 Softmax GP (f) Toy4 Archipelago Entropy

(g) Toy5 Archipelago (h) Toy5 Softmax GP (i) Toy5 Archipelago Entropy

Figure 8.3: These figures show results of applying Archipelago to "pinwheel" data with only one labeled datum per class (highlighted). The top row has three classes, the middle row has four classes and the bottom has five. The figures on the left show the modes of the class assignments from Archipelago. The middle column shows the class assignment modes using a softmax Gaussian process. The right column shows the entropies of the predictive distribution produced by Archipelago.

Gaussian process classifier by requiring a null-class transition between any two "real" classes. The latent rejections play a similar role in Archipelago by allowing regions with mass under $\pi(\boldsymbol{x} \mid \boldsymbol{\psi}_\pi)$ to have no data. However, Archipelago models data in the null class explicitly as part of inference, and the rejections give Archipelago its ability to model arbitrarily complex densities. Also, in contrast to Archipelago, the Null Category Noise Model requires setting the null-category width and *a priori* determination of labeling probability. In almost all of the tests, Archipelago had lower test classification error than the NCNM. In the regime with few labels, Archipelago significantly outperformed the other methods on test error. As Archipelago is a generative model

and the others are discriminative, this difference is expected.

8.7 Computational Considerations

Gaussian processes have relatively high computational costs, and the Archipelago model inherits them. With a naïve implementation, the time complexity per MCMC step is $O(K(N+J+M)^3)$ and the space complexity is $O(K(N+J+M)^2)$. There is a large literature on sparse approximations to Gaussian processes, e.g. Quiñonero-Candela and Rasmussen (2005), and this is an area for future research. Mixing of Markov chains is an additional concern, but I have not found this to be a problem in practice due to the use of Hamiltonian Monte Carlo.

As the computational complexity grows rapidly with the number of latent rejections, minimising these rejections is paramount. The number of latent rejections relates directly to the mismatch between the base density $\pi(\boldsymbol{x} \mid \boldsymbol{\psi}_\pi)$ and the true data density. Base density hyperparameter inference can lower the computational burden by adapting $\pi(\boldsymbol{x} \mid \boldsymbol{\psi}_\pi)$ to the data, but it is important to choose an appropriate parametric family. In high dimensions this choice is difficult to make, and in the absence of significant domain knowledge I expect that Archipelago will not work well in high dimensions. This problem in high-dimensions is common to generative approaches to semi-supervised learning, and it is unclear how to avoid it.

8.8 Summary

I have presented a nonparametric Bayesian method for semi-supervised learning that is an extension of the Gaussian process density sampler of Chapter 2. I used a latent history method for inference, similar to that presented for the GPDS and sigmoidal Gaussian Cox process in Chapter 5. Empirical analyses on synthetic and real-world data indicate that the Archipelago model successfully incorporates information from the unlabeled data. The approach is competitive with other Bayesian approaches to semi-supervised learning.

Chapter 9

Conclusions and Future Work

This chapter discusses some possible future directions for work on the Gaussian process density sampler, the sigmoidal Gaussian Cox process, and the Archipelago semi-supervised learning model. I also briefly review the contributions of the thesis.

9.1 Modeling Other Spaces with the GPDS and Archipelago

One of the interesting aspects of the Gaussian process density sampler and Archipelago is that they can be used on any space on which it is possible to define a positive definite kernel function. I have presented these methods with $\mathcal{X}$ as $\mathbb{R}^D$, but this is not necessary. There is a growing literature describing positive definite kernels on many different types of data (e.g. Haussler (1999), Kondor and Lafferty (2002) and Rasmussen and Williams (2006, Section 4.4)). Some of these spaces are countable and in these cases the Gaussian process density sampler provides a nonparametric prior on probability mass functions, rather than density functions.

9.1.1 Permutation Kernels

Modeling a distribution on the space of permutations has similar difficulties to the Potts model discussed in Section 4.2. If there are K items in our set, then there are $K!$ possible permutations. The naïve approach to representing a distribution on permutations is to define a categorical distribution on all $K!$ possible situations. For K larger than about 14, however, it becomes impossible to hold this distribution in memory, and normalising the distribution becomes an intractable problem.

Kondor (2008) proposes a class of positive definite kernels on permutation groups. It may be possible to combine such kernels with the Gaussian process density sampler to construct tractable Bayesian models on permutations with $K > 14$, without

approximation. This would enable inferences of the form "given a set of observed permutations, fantasise additional permutations from the same distribution."

9.1.2 String Kernels

Given an alphabet, a *string* is a sequence of members from that alphabet, possibly with repeats. Several different proposals have been made of positive definite kernels on strings, e.g. Haussler (1999), Watkins (2000), and Lodhi et al. (2002). These strings can represent a wide variety of data structures and have been used to model, for example, text (Collins and Duffy, 2001) and protein structure (Leslie et al., 2004). It is possible that the Gaussian process density sampler and Archipelago could be used as a nonparametric Bayesian prior for mass functions on these domains.

9.1.3 Set Kernels

Closely related to string kernels are kernel functions on sets. The space of interest in this case is the space of unordered subsets of some other space. Kondor and Jebara (2003), for example, describe a positive definite kernel on finite subsets of $\mathbb{R}^D$. This can be thought of as a kernel between "bags of data." It is interesting to consider that the Gaussian process density sampler, defined on the space of finite subsets of $\mathbb{R}^D$, is a point process as described in Section 3.1. The specific characteristics of this point process depend heavily on the choice of base density and the set kernel, and identifying the properties of this point process is a potential topic for further study.

9.1.4 Graph Kernels

Many data are best described by graphs, and several authors, e.g., Kondor and Lafferty (2002), Gärtner et al. (2003), and Borgwardt (2007), have proposed positive definite kernels on the space of graphs. Modeling of protein interactions has been of particular interest in this domain (Borgwardt et al., 2005). A common task in modeling proteins is to predict their function with a classifier. The Archipelago model is promising in this regard, as it could potentially allow proteins with unknown function to help guide classification.

9.2 Inclusion of Covariates

There has been significant recent interest in methods for introducing dependency between nonparametric processes (MacEachern, 2000; Müller et al., 2004; Srebro and Roweis, 2005; Dunson and Park, 2008; Griffin and Steel, 2009). The motivation for

these models has been to find a way to tie together multiple nonparametric distributions. For example, the distribution over a set of economic indicators may vary in time. However, we expect that this distribution will change very slowly, and we would like to take advantage of this additional structure when performing inference. The Gaussian process density sampler can potentially be extended to include covariates by augmenting the input space for the GP prior. The specific details of this construction would be a topic for future work, but it appears to be straightforward to tie multiple Gaussian process density samplers together via shared inputs.

9.3 Doubly-Nonparametric Density Modeling

When presenting the Gaussian process density sampler and the Archipelago model, I have made the assumption that $\pi(\boldsymbol{x})$ is a model with a finite set of parameters. However, this is not strictly necessary. Rather, what is required is that it be possible to generate exact data from $\pi(\boldsymbol{x})$. Dirichlet process mixture models (Escobar and West, 1995; Rasmussen, 2000) are nonparametric models from which it is possible to generate exact data. Potentially, the Gaussian process density sampler and the DPMM could be combined to create a model with two nonparametric aspects. It appears that inference via the latent history method would be a direct extension of the vanilla GPDS case, as long as component membership labels are explicitly modeled. This would enable $\pi(\boldsymbol{x})$ to be very flexible and provide a tighter bound on the true density.

9.4 Archipelago with Dependent Latent Functions

In the Archipelago model, I used K independent Gaussian processes as part of the model. This independence is not required. We might, for example, believe that data from some classes tend to co-occur with data from another class. Dependency between the Gaussian processes could be introduced to capture this kind of prior information, using, for example, the latent factor model of Teh et al. (2005) or the multi-task learning approach of Bonilla et al. (2008).

9.5 Summary of Contributions

Kernel-based methods for nonparametric inference of densities and point processes have been of significant interest to statistical modelers for several decades. Bayesian approaches to this problem, while desirable in the abstract, have been difficult due to the need to integrate over infinite-dimensional random functions. A variety of approximations have been introduced in the literature over the years.

This thesis contributes the first fully-nonparametric and non-approximate Bayesian models for kernel-based MCMC inference of densities and point process intensities. The methods I have presented combine generative models with recent advances in Markov chain Monte Carlo to achieve tractability. I also proposed a soft-core generalisation of the Matérn Type III repulsive process and showed how inference could be performed in that model with the sigmoidal Gaussian Cox process as the primary process. In addition, I extended the latent history inference approach in the SGCP to allow for an inhomogeneous Neyman–Scott process. Finally, I presented a nonparametric Bayesian model for semi-supervised learning and showed that it is competitive with other Bayesian approaches to this problem.

Appendix A

Additional Algorithms

Algorithm A.1 Generate Poisson events from the SGCP on an irregular region.

Inputs:
- Bounding domain $\mathcal{V}$
- Region indicator function $\boldsymbol{I}(\boldsymbol{x})$
- Dominating intensity function $\bar{\lambda}(\boldsymbol{x}\,;\,\boldsymbol{\psi}_\lambda)$
- Gaussian process covariance function $C(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta})$

Outputs:
- Poisson events $\mathcal{S} = \{\boldsymbol{x}_k\}_{k=1}^{K}$ from an SGCP prior

$\{\tilde{\boldsymbol{x}}_j\}_{j=1}^{J} \sim \mathcal{PP}(\bar{\lambda}(\boldsymbol{x}\,;\,\boldsymbol{\psi}_\lambda), \mathcal{V})$ ▷ Draw Poisson events on $\mathcal{V}$ according to intensity $\bar{\lambda}(\boldsymbol{x})$.
$\hat{\mathcal{V}} \leftarrow \emptyset$ ▷ Initialise the first set of unthinned events.
for $j \leftarrow 1 \ldots J$ **do** ▷ Loop over the initial events.
  if $\boldsymbol{I}(\tilde{\boldsymbol{x}}_j)$ **then** ▷ Test the region membership.
    $\hat{\mathcal{V}} \leftarrow \hat{\mathcal{V}} \cup \tilde{\boldsymbol{x}}_j$ ▷ Keep if inside the irregular region.
  end if
end for
$\hat{J} \leftarrow |\hat{\mathcal{V}}|$ ▷ Get the number of events left inside the region.
$\{g(\tilde{\boldsymbol{x}}_j)\}_{j=1}^{\hat{J}} \sim \mathcal{GP}(g \mid \{\tilde{\boldsymbol{x}}_j\}_{j=1}^{\hat{J}}, \boldsymbol{\theta})$ ▷ Draw the function from the GP at the events.
$\mathcal{S} \leftarrow \emptyset$ ▷ Initialise the set of unthinned events.
for all $\tilde{\boldsymbol{x}}_j \in \hat{\mathcal{V}}$ **do** ▷ Loop over the events in the region.
  $u \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on $(0, 1)$.
  if $u < \Phi(g(\tilde{\boldsymbol{x}}_j))$ **then** ▷ Apply the thinning acceptance rule.
    $\mathcal{S} \leftarrow \mathcal{S} \cup \tilde{\boldsymbol{x}}_j$ ▷ Add to the accepted events.
  end if
end for
return $\mathcal{S}$

Algorithm A.2 Generate marked Poisson events from the SGCP.

Inputs:
- Domain $\mathcal{V}$
- Marking distribution $p(\boldsymbol{y} \mid \boldsymbol{x})$
- Dominating intensity function $\bar{\lambda}(\boldsymbol{x}\,;\,\boldsymbol{\psi}_\lambda)$
- Gaussian process covariance function $C(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta})$

Outputs:
- Marked Poisson events $\mathcal{S} = \{\boldsymbol{x}_k, \boldsymbol{y}_k\}_{k=1}^{K}$ from an SGCP prior

$\{\tilde{\boldsymbol{x}}_j\}_{j=1}^{J} \sim \mathcal{PP}(\bar{\lambda}(\boldsymbol{x}\,;\,\boldsymbol{\psi}_\lambda), \mathcal{V})$ ▷ Draw Poisson events according to intensity $\bar{\lambda}(\boldsymbol{x})$.
$\{g(\tilde{\boldsymbol{x}}_j)\}_{j=1}^{J} \sim \mathcal{GP}(g \mid \{\tilde{\boldsymbol{x}}_j\}_{j=1}^{J}, \boldsymbol{\theta})$ ▷ Draw the function from the GP at the events.
$\mathcal{S} \leftarrow \emptyset$ ▷ Initialise the set of unthinned events.
for $j \leftarrow 1 \ldots J$ **do** ▷ Loop over the proposed events.
  $u \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on $(0, 1)$.
  if $u < \Phi(g(\tilde{\boldsymbol{x}}_j))$ **then** ▷ Apply the thinning acceptance rule.
    $\tilde{\boldsymbol{y}} \sim p(\boldsymbol{y} \mid \tilde{\boldsymbol{x}}_j)$ ▷ Draw from the marking distribution.
    $\mathcal{S} \leftarrow \mathcal{S} \cup (\tilde{\boldsymbol{x}}_j, \tilde{\boldsymbol{y}})$ ▷ Add the pair to the accepted events.
  end if
end for
return $\mathcal{S}$

Algorithm A.3 Generate a Neyman–Scott realisation from the SGCP.

Inputs:
- Domain $\mathcal{V}$
- Daugher intensity $\lambda^{\mathsf{dtr}}(\boldsymbol{x}\,;\,\boldsymbol{x}^{\mathsf{mtr}})$ with compact support $\mathcal{Q}$
- Dominating mother intensity function $\bar{\lambda}(\boldsymbol{x}\,;\,\boldsymbol{\psi}_\lambda)$
- Gaussian process covariance function $C(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta})$

Outputs:
- Neyman–Scott events $\mathcal{S} = \{\boldsymbol{x}_k\}_{k=1}^{K}$ from an SGCP prior

$\{\tilde{\boldsymbol{x}}_j\}_{j=1}^{J} \sim \mathcal{PP}(\bar{\lambda}(\boldsymbol{x}\,;\,\boldsymbol{\psi}_\lambda), \mathcal{V} \oplus \mathcal{Q})$ ▷ Draw Poisson events according to intensity $\bar{\lambda}(\boldsymbol{x})$.
$\{g(\tilde{\boldsymbol{x}}_j)\}_{j=1}^{J} \sim \mathcal{GP}(g \mid \{\tilde{\boldsymbol{x}}_j\}_{j=1}^{J}, \boldsymbol{\theta})$ ▷ Draw the function from the GP at the events.
$\mathcal{S} \leftarrow \emptyset$ ▷ Initialise the set of unthinned events.
for $j \leftarrow 1 \ldots J$ **do** ▷ Loop over the proposed events.
  $u \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on $(0, 1)$.
  if $u < \Phi(g(\tilde{\boldsymbol{x}}_j))$ **then** ▷ Thin to get mother point.
    $\{\boldsymbol{x}_m\}_{m=1}^{M} \sim \mathcal{PP}(\lambda^{\mathsf{dtr}}(\boldsymbol{x}\,;\,\tilde{\boldsymbol{x}}_j), \tilde{\boldsymbol{x}}_j \oplus \mathcal{Q})$ ▷ Draw daughter points centred at $\tilde{\boldsymbol{x}}_j$.
    for $m \leftarrow 1 \ldots M$ **do** ▷ Loop over the daughter events.
      if $\boldsymbol{I}_{\mathcal{V}}(\boldsymbol{x}_n)$ **then** ▷ Test if the daughter is inside $\mathcal{V}$.
        $\mathcal{S} \leftarrow \mathcal{S} \cup \boldsymbol{x}_m$ ▷ Keep the daughter point.
      end if
    end for
  end if
end for
return $\mathcal{S}$

Algorithm A.4 Generate a Hawkes realisation from the SGCP.

Inputs:
- Domain $\mathcal{V}$ with periodic boundaries
- Daugher intensity $\lambda^{\mathsf{dtr}}(\boldsymbol{x}\,;\,\boldsymbol{x}^{\mathsf{mtr}})$
- Dominating mother intensity function $\bar{\lambda}(\boldsymbol{x}\,;\,\boldsymbol{\psi}_\lambda)$
- Gaussian process covariance function $C(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta})$

Outputs:
- Hawkes events $\mathcal{S} = \{\boldsymbol{x}_k\}_{k=1}^K$ from an SGCP prior

$\{\tilde{\boldsymbol{x}}_j\}_{j=1}^J \sim \mathcal{PP}(\bar{\lambda}(\boldsymbol{x}\,;\,\boldsymbol{\psi}_\lambda), \mathcal{V})$ ▷ Draw Poisson events according to intensity $\bar{\lambda}(\boldsymbol{x})$.
$\{g(\tilde{\boldsymbol{x}}_j)\}_{j=1}^J \sim \mathcal{GP}(g \mid \{\tilde{\boldsymbol{x}}_j\}_{j=1}^J, \boldsymbol{\theta})$ ▷ Draw the function from the GP at the events.
$\mathcal{S} \leftarrow \emptyset$ ▷ Initialise the set of unthinned events.
for $j \leftarrow 1 \dots J$ **do** ▷ Loop over mother proposals.
  $u \sim \mathcal{U}(0,1)$ ▷ Draw a uniform random variate on $(0,1)$.
  if $u < \Phi(g(\tilde{\boldsymbol{x}}_j))$ **then** ▷ Thin to get mother point.
    $\mathcal{S} \leftarrow \mathcal{S} \cup \tilde{\boldsymbol{x}}_j$ ▷ Add the mother point to the set.
    $\mathcal{S}' \leftarrow$ DAUGHTER($\tilde{\boldsymbol{x}}_j$) ▷ Simulate the daughter cascade.
    $\mathcal{S} \leftarrow \mathcal{S} \cup \mathcal{S}'$ ▷ Add the daughter points to the set.
  end if
end for
return $\mathcal{S}$
procedure DAUGHTER($\boldsymbol{x}'$)
  $\{\tilde{\boldsymbol{x}}_m\}_{m=1}^M \sim \mathcal{PP}(\lambda^{\mathsf{dtr}}(\boldsymbol{x}\,;\,\boldsymbol{x}'), \mathcal{Q})$ ▷ Draw daughters centred at $\boldsymbol{x}'$.
  $\mathcal{S}' \leftarrow \{\tilde{\boldsymbol{x}}_m\}_{m=1}^M$ ▷ Store the daughter points.
  for $m \leftarrow 1 \dots M$ **do** ▷ Loop over the daughter points.
    $\mathcal{S}'' \leftarrow$ DAUGHTER($\tilde{\boldsymbol{x}}_m$) ▷ Recurse on the daughters.
    $\mathcal{S}' \leftarrow \mathcal{S}' \cup \mathcal{S}''$ ▷ Collect cascaded daughter points.
  end for
  return $\mathcal{S}'$
end procedure

Algorithm A.5 Generate a generalised Matérn Type II realisation from the SGCP.

Inputs:
- Domain $\mathcal{V}$ with periodic boundaries
- Repulsion kernel $\rho(\boldsymbol{x} \nleftarrow \boldsymbol{x}')$
- Dominating primary intensity function $\bar{\lambda}(\boldsymbol{x}\,;\,\boldsymbol{\psi}_\lambda)$
- Gaussian process covariance function $C(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta})$

Outputs:
- Generalised Matérn Type II events $\mathcal{S} = \{\boldsymbol{x}_k\}_{k=1}^K$ from an SGCP prior

$\{\tilde{\boldsymbol{x}}_j\}_{j=1}^J \sim \mathcal{PP}(\bar{\lambda}(\boldsymbol{x}\,;\,\boldsymbol{\psi}_\lambda), \mathcal{V})$ ▷ Draw Poisson events according to intensity $\bar{\lambda}(\boldsymbol{x})$.
▷ (I assume a random ordering of the events.)
$\{g(\tilde{\boldsymbol{x}}_j)\}_{j=1}^J \sim \mathcal{GP}(g \mid \{\tilde{\boldsymbol{x}}_j\}_{j=1}^J, \boldsymbol{\theta})$ ▷ Draw the function from the GP at the events.
$\mathcal{S}' \leftarrow \emptyset$ ▷ Initialise the set of primary events.
for $j \leftarrow 1 \dots J$ **do** ▷ Loop over the proposed events.
  $u_1 \sim \mathcal{U}(0,1)$ ▷ Draw a uniform random variate on $(0,1)$.
  if $u_1 < \Phi(g(\tilde{\boldsymbol{x}}_j))$ **then** ▷ Thin to get primary point.
    $p_{\mathsf{keep}} \leftarrow 1$ ▷ Initialise the probability of keeping this point.
    for $j' \leftarrow 1 \dots j-1$ **do** ▷ Loop over older primary points.
      $p_{\mathsf{keep}} \leftarrow p_{\mathsf{keep}} \times (1 - \rho(\tilde{\boldsymbol{x}}_j \nleftarrow \tilde{\boldsymbol{x}}_{j'}))$ ▷ Accumulate repulsion probabilities.
    end for
    $u_2 \sim \mathcal{U}(0,1)$ ▷ Draw a uniform random variate on $(0,1)$.
    if $u_2 < p_{\mathsf{keep}}$ **then** ▷ Apply Type II primary-to-secondary thinning rule.
      $\mathcal{S} \leftarrow \mathcal{S} \cup \tilde{\boldsymbol{x}}_j$ ▷ Store the secondary point.
    end if
  end if
end for
return $\mathcal{S}$

Algorithm A.6 Generate a generalised Matérn Type III realisation from the SGCP.

Inputs:
- Domain $\mathcal{V}$ with periodic boundaries
- Repulsion kernel $\rho(\boldsymbol{x} \nleftarrow \boldsymbol{x}')$
- Dominating primary intensity function $\bar{\lambda}(\boldsymbol{x}\,;\,\boldsymbol{\psi}_\lambda)$
- Gaussian process covariance function $C(\boldsymbol{x}, \boldsymbol{x}'\,;\,\boldsymbol{\theta})$

Outputs:
- Generalised Matérn Type III events $\mathcal{S} = \{\boldsymbol{x}_k\}_{k=1}^K$ from an SGCP prior

$\{\tilde{\boldsymbol{x}}_j\}_{j=1}^J \sim \mathcal{PP}(\bar{\lambda}(\boldsymbol{x}\,;\,\boldsymbol{\psi}_\lambda), \mathcal{V})$ ▷ Draw Poisson events according to intensity $\bar{\lambda}(\boldsymbol{x})$.
▷ (I assume a random ordering of the events.)
$\{g(\tilde{\boldsymbol{x}}_j)\}_{j=1}^J \sim \mathcal{GP}(g \mid \{\tilde{\boldsymbol{x}}_j\}_{j=1}^J, \boldsymbol{\theta})$ ▷ Draw the function from the GP at the events.
$\mathcal{S}' \leftarrow \emptyset$ ▷ Initialise the set of primary events.
for $j \leftarrow 1 \ldots J$ **do** ▷ Loop over the proposed events.
  $u_1 \sim \mathcal{U}(0,1)$ ▷ Draw a uniform random variate on $(0,1)$.
  if $u_1 < \Phi(g(\tilde{\boldsymbol{x}}_j))$ **then** ▷ Thin to get primary point.
    $p_{\mathsf{keep}} \leftarrow 1$ ▷ Initialise the probability of keeping this point.
    for all $\boldsymbol{x}' \in \mathcal{S}$ **do** ▷ Loop over older secondary points.
      $p_{\mathsf{keep}} \leftarrow p_{\mathsf{keep}} \times (1 - \rho(\tilde{\boldsymbol{x}}_j \nleftarrow \boldsymbol{x}'))$ ▷ Accumulate repulsion probabilities.
    end for
    $u_2 \sim \mathcal{U}(0,1)$ ▷ Draw a uniform random variate on $(0,1)$.
    if $u_2 < p_{\mathsf{keep}}$ **then** ▷ Apply primary-to-secondary thinning rule.
      $\mathcal{S} \leftarrow \mathcal{S} \cup \tilde{\boldsymbol{x}}_j$ ▷ Store the secondary point.
    end if
  end if
end for
return $\mathcal{S}$

Algorithm A.7 Take an exchange sampling step on GP hyperparameters for the GPDS

Inputs:
- Observed data $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^N$
- Control points $\mathcal{C} = \{\boldsymbol{x}_b\}_{b=1}^B$
- Control point proposal distribution $q(\hat{\mathcal{G}} \leftarrow \mathcal{G}\,;\,\boldsymbol{\theta})$
- Hyperparameter proposal distribution $q(\hat{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta})$
- Hyperparameter prior $p_0(\boldsymbol{\theta})$
- Current conditioning sets $\boldsymbol{X}$ and $\boldsymbol{G}$
- Gaussian process covariance function $C(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta})$
- Base density $\pi(\boldsymbol{x} \,|\, \boldsymbol{\psi}_\pi)$
- Current Gaussian process hyperparameters $\boldsymbol{\theta}$

Outputs:
- New Gaussian process hyperparameters $\boldsymbol{\theta}$
- New conditioning sets $\boldsymbol{X}$ and $\boldsymbol{G}$

$\hat{\boldsymbol{\theta}} \sim q(\hat{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta})$ ▷ Draw the hyperparameter proposal.
$\hat{\mathcal{G}} \sim q(\hat{\mathcal{G}} \leftarrow \mathcal{G}; \hat{\boldsymbol{\theta}})$ ▷ Draw new control point values.
$\hat{\boldsymbol{X}} \leftarrow \mathcal{C}$, $\hat{\boldsymbol{G}} \leftarrow \hat{\mathcal{G}}$ ▷ Initialise conditioning sets.
repeat
  $\tilde{\boldsymbol{w}} \sim \pi(\boldsymbol{x} \,|\, \boldsymbol{\psi}_\pi)$ ▷ Draw a proposal from the base density.
  $\hat{g}(\tilde{\boldsymbol{w}}) \sim \mathcal{GP}(g \,|\, \tilde{\boldsymbol{w}}, \hat{\boldsymbol{X}}, \hat{\boldsymbol{G}}, \hat{\boldsymbol{\theta}})$ ▷ Draw the function value at the proposal.
  $u_{\mathsf{fant}} \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on $(0, 1)$.
  if $u_{\mathsf{fant}} < \Phi(\hat{g}(\tilde{\boldsymbol{w}}))$ **then** ▷ Rejection sampling acceptance rule.
    $\mathcal{W} \leftarrow \mathcal{W} \cup \tilde{\boldsymbol{w}}$ ▷ Keep the fantasy.
  end if
  $\hat{\boldsymbol{X}} \leftarrow \hat{\boldsymbol{X}} \cup \tilde{\boldsymbol{w}}$, $\hat{\boldsymbol{G}} \leftarrow \hat{\boldsymbol{G}} \cup \hat{g}(\tilde{\boldsymbol{w}})$ ▷ Add proposals to the conditioning sets.
until $|\mathcal{W}| = N$ ▷ Loop until N fantasies are accepted.
$\{g(\boldsymbol{w}_n)\}_{n=1}^N \sim \mathcal{GP}(g \,|\, \mathcal{W}, \boldsymbol{X}^{(p)}, \boldsymbol{G}^{(p)}, \boldsymbol{\theta})$ ▷ Sample the current function at the fantasies.

$$a_{\mathsf{gpds-gphp}} \leftarrow \frac{p_0(\hat{\boldsymbol{\theta}})\; \mathcal{GP}(\hat{\mathcal{G}} \,|\, \mathcal{C}, \hat{\boldsymbol{\theta}})\; q(\boldsymbol{\theta} \leftarrow \hat{\boldsymbol{\theta}})\; q(\mathcal{G} \leftarrow \hat{\mathcal{G}}; \boldsymbol{\theta})}{p_0(\boldsymbol{\theta})\; \mathcal{GP}(\mathcal{G} \,|\, \mathcal{C}, \boldsymbol{\theta})\; q(\hat{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta})\; q(\hat{\mathcal{G}} \leftarrow \mathcal{G}; \hat{\boldsymbol{\theta}})} \prod_{n=1}^{N} \frac{\Phi(\hat{g}(\boldsymbol{x}_n))\; \Phi(g(\boldsymbol{w}_n))}{\Phi(g(\boldsymbol{x}_n))\; \Phi(\hat{g}(\boldsymbol{w}_n))}$$

$u_{\mathsf{mh}} \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on $(0, 1)$.
if $u_{\mathsf{mh}} < a_{\mathsf{gpds-gphp}}$ **then** ▷ Apply the Metropolis–Hastings acceptance rule.
  $\boldsymbol{\theta} \leftarrow \hat{\boldsymbol{\theta}}$ ▷ Keep the new hyperparameters.
  $\boldsymbol{X} \leftarrow \hat{\boldsymbol{X}}$, $\boldsymbol{G} \leftarrow \hat{\boldsymbol{G}}$ ▷ Keep the new conditioning sets.
  $\mathcal{G} \leftarrow \hat{\mathcal{G}}$ ▷ Keep the new control points.
else
  $\boldsymbol{X} \leftarrow \boldsymbol{X} \cup \{\boldsymbol{w}_n\}_{n=1}^N$, $\boldsymbol{G} \leftarrow \boldsymbol{G} \cup \{g(\boldsymbol{w}_n)\}_{n=1}^N$ ▷ Update the current conditioning set.
end if
return $\boldsymbol{\theta}$, $\boldsymbol{X}$, $\boldsymbol{G}$

Algorithm A.8 Take an ES step on base density hyperparameters for the GPDS

Inputs:
- Observed data $\{\boldsymbol{x}_n\}_{n=1}^N$
- Hyperparameter proposal distribution $q(\hat{\boldsymbol{\psi}}_\pi \leftarrow \boldsymbol{\psi}_\pi)$
- Hyperparameter prior $p_0(\boldsymbol{\psi}_\pi)$
- Current conditioning sets $\boldsymbol{X}$ and $\boldsymbol{G}$
- Gaussian process covariance function $C(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta})$
- Gaussian process hyperparameters $\boldsymbol{\theta}$
- Base density $\pi(\boldsymbol{x} \mid \boldsymbol{\psi}_\pi)$
- Current base density hyperparameters $\boldsymbol{\psi}_\pi$

Outputs:
- New base density hyperparameters $\boldsymbol{\psi}_\pi$
- New conditioning sets $\boldsymbol{X}$ and $\boldsymbol{G}$

$\hat{\boldsymbol{\psi}}_\pi \sim q(\hat{\boldsymbol{\psi}}_\pi \leftarrow \boldsymbol{\psi}_\pi)$ ▷ Draw the hyperparameter proposal.
repeat
  $\tilde{\boldsymbol{w}} \sim \pi(\boldsymbol{x} \mid \hat{\boldsymbol{\psi}}_\pi)$ ▷ Draw a proposal from the base density.
  $g(\tilde{\boldsymbol{w}}) \sim \mathcal{GP}(g \mid \tilde{\boldsymbol{w}}, \boldsymbol{X}, \boldsymbol{G}, \boldsymbol{\theta})$ ▷ Draw the function value at the proposal.
  $u_{\mathsf{fant}} \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on $(0, 1)$.
  if $u_{\mathsf{fant}} < \Phi(g(\tilde{\boldsymbol{w}}))$ **then** ▷ Rejection sampling acceptance rule.
    $\mathcal{W} \leftarrow \mathcal{W} \cup \tilde{\boldsymbol{w}}$ ▷ Keep the fantasy.
  end if
  $\boldsymbol{X} \leftarrow \boldsymbol{X} \cup \tilde{\boldsymbol{w}}$, $\boldsymbol{G} \leftarrow \boldsymbol{G} \cup g(\tilde{\boldsymbol{w}})$ ▷ Add proposals to the conditioning sets.
until $|\mathcal{W}| = N$ ▷ Loop until N fantasies are accepted.

$$a_{\mathsf{gpds-bdhp}} \leftarrow \frac{p_0(\hat{\boldsymbol{\psi}}_\pi)\; q(\boldsymbol{\psi}_\pi \leftarrow \hat{\boldsymbol{\psi}}_\pi)}{p_0(\boldsymbol{\psi}_\pi)\; q(\hat{\boldsymbol{\psi}}_\pi \leftarrow \boldsymbol{\psi}_\pi)} \prod_{n=1}^{N} \frac{\pi(\boldsymbol{x}_n \mid \hat{\boldsymbol{\psi}}_\pi)\; \pi(\boldsymbol{w}_n \mid \boldsymbol{\psi}_\pi)}{\pi(\boldsymbol{x}_n \mid \boldsymbol{\psi}_\pi)\; \pi(\boldsymbol{w}_n \mid \hat{\boldsymbol{\psi}}_\pi)}$$

$u_{\mathsf{mh}} \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on $(0, 1)$.
if $u_{\mathsf{mh}} < a_{\mathsf{gpds-bdhp}}$ **then** ▷ Apply the Metropolis–Hastings acceptance rule.
  $\boldsymbol{\psi}_\pi \leftarrow \hat{\boldsymbol{\psi}}_\pi$ ▷ Keep the new hyperparameters.
end if
return $\boldsymbol{\psi}_\pi$, $\boldsymbol{X}$, $\boldsymbol{G}$

Algorithm A.9 Take an exchange sampling step on GP hyperparameters for the SGCP

Inputs:
- Observed data $\mathcal{S} = \{\boldsymbol{x}_k\}_{k=1}^{K}$
- Control points $\mathcal{C} = \{\boldsymbol{x}_b\}_{b=1}^{B}$
- Control point proposal distribution $q(\hat{\mathcal{G}} \leftarrow \mathcal{G}\,;\,\boldsymbol{\theta})$
- Hyperparameter proposal distribution $q(\hat{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta})$
- Hyperparameter prior $p_0(\boldsymbol{\theta})$
- Current conditioning sets $\boldsymbol{X}$ and $\boldsymbol{G}$
- Gaussian process covariance function $C(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta})$
- Dominating intensity $\bar{\lambda}(\boldsymbol{x}\,;\,\boldsymbol{\psi}_\lambda)$
- Current Gaussian process hyperparameters $\boldsymbol{\theta}$

Outputs:
- New Gaussian process hyperparameters $\boldsymbol{\theta}$
- New conditioning sets $\boldsymbol{X}$ and $\boldsymbol{G}$

$\hat{\boldsymbol{\theta}} \sim q(\hat{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta})$ ▷ Draw the hyperparameter proposal.
$\hat{\mathcal{G}} \sim q(\hat{\mathcal{G}} \leftarrow \mathcal{G}; \hat{\boldsymbol{\theta}})$ ▷ Draw new control point values.
$\hat{\boldsymbol{X}} \leftarrow \mathcal{C}$, $\hat{\boldsymbol{G}} \leftarrow \hat{\mathcal{G}}$ ▷ Initialise conditioning sets.
$\{\tilde{\boldsymbol{w}}_j\}_{j=1}^{J} \sim \mathcal{PP}(\bar{\lambda}(\boldsymbol{x}\,;\,\boldsymbol{\psi}_\lambda), \mathcal{V})$ ▷ Draw from the dominating intensity.
$\{\hat{g}(\tilde{\boldsymbol{w}}_j)\}_{j=1}^{J} \sim \mathcal{GP}(g \,|\, \{\tilde{\boldsymbol{w}}_j\}_{j=1}^{J}, \hat{\boldsymbol{X}}, \hat{\boldsymbol{G}}, \hat{\boldsymbol{\theta}})$ ▷ Sample the proposed function at the events.
$\hat{\boldsymbol{X}} \leftarrow \hat{\boldsymbol{X}} \cup \{\tilde{\boldsymbol{w}}_j\}_{j=1}^{J}$, $\hat{\boldsymbol{G}} \leftarrow \hat{\boldsymbol{G}} \cup \{g(\tilde{\boldsymbol{w}}_j)\}_{j=1}^{J}$ ▷ Update the proposed conditioning set.
$\mathcal{W} \leftarrow \emptyset$ ▷ Initialise fantasy set.
for $j \leftarrow 1 \ldots J$ **do** ▷ Loop over the events.
    $u_{\mathsf{fant}} \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on $(0, 1)$.
    if $u_{\mathsf{fant}} < \Phi(\hat{g}(\tilde{\boldsymbol{w}}_j))$ **then** ▷ Apply the thinning rule.
        $\mathcal{W} \leftarrow \mathcal{W} \cup \tilde{\boldsymbol{w}}_j$ ▷ Add the event to the fantasy set.
    end if
end for
$\{g(\boldsymbol{w}_k)\}_{k=1}^{\hat{K}} \sim \mathcal{GP}(g \,|\, \mathcal{W}, \boldsymbol{X}^{(r)}, \boldsymbol{G}^{(r)}, \boldsymbol{\theta})$ ▷ Sample the current function at the fantasies.

$$a_{\mathsf{sgcp-gphp}} \leftarrow \frac{p_0(\hat{\boldsymbol{\theta}})\ \mathcal{GP}(\hat{\mathcal{G}} \,|\, \mathcal{C}, \hat{\boldsymbol{\theta}})\ q(\boldsymbol{\theta} \leftarrow \hat{\boldsymbol{\theta}})\ q(\mathcal{G} \leftarrow \hat{\mathcal{G}}; \boldsymbol{\theta})}{p_0(\boldsymbol{\theta})\ \mathcal{GP}(\mathcal{G} \,|\, \mathcal{C}, \boldsymbol{\theta})\ q(\hat{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta})\ q(\hat{\mathcal{G}} \leftarrow \mathcal{G}; \hat{\boldsymbol{\theta}})} \prod_{n=1}^{N} \frac{\Phi(\hat{g}(\boldsymbol{x}_n))\ \Phi(g(\boldsymbol{w}_n))}{\Phi(g(\boldsymbol{x}_n))\ \Phi(\hat{g}(\boldsymbol{w}_n))}$$

$u_{\mathsf{mh}} \sim \mathcal{U}(0, 1)$ ▷ Draw a uniform random variate on $(0, 1)$.
if $u_{\mathsf{mh}} < a_{\mathsf{sgcp-gphp}}$ **then** ▷ Apply the Metropolis–Hastings acceptance rule.
    $\boldsymbol{\theta} \leftarrow \hat{\boldsymbol{\theta}}$ ▷ Keep the new hyperparameters.
    $\boldsymbol{X} \leftarrow \hat{\boldsymbol{X}}$, $\boldsymbol{G} \leftarrow \hat{\boldsymbol{G}}$ ▷ Keep the new conditioning sets.
    $\mathcal{G} \leftarrow \hat{\mathcal{G}}$ ▷ Keep the new control points.
else
    $\boldsymbol{X} \leftarrow \boldsymbol{X} \cup \{\boldsymbol{w}_n\}_{n=1}^{N}$, $\boldsymbol{G} \leftarrow \boldsymbol{G} \cup \{g(\boldsymbol{w}_n)\}_{n=1}^{N}$ ▷ Update the current conditioning set.
end if
return $\boldsymbol{\theta}$, $\boldsymbol{X}$, $\boldsymbol{G}$

Algorithm A.10 Take an ES step on bounding intensity hyperparameters for the SGCP

Inputs:
- Observed data $\{\boldsymbol{x}_k\}_{k=1}^K$
- Hyperparameter proposal distribution $q(\hat{\boldsymbol{\psi}}_\lambda \leftarrow \boldsymbol{\psi}_\lambda)$
- Hyperparameter prior $p_0(\boldsymbol{\psi}_\lambda)$
- Current conditioning sets $\boldsymbol{X}$ and $\boldsymbol{G}$
- Gaussian process covariance function $C(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta})$
- Dominating intensity $\bar{\lambda}(\boldsymbol{x}\,;\,\boldsymbol{\psi}_\lambda)$
- Current dominating intensity hyperparameters $\boldsymbol{\psi}_\lambda$

Outputs:
- New base density hyperparameters $\boldsymbol{\psi}_\lambda$
- New conditioning sets $\boldsymbol{X}$ and $\boldsymbol{G}$

$\hat{\boldsymbol{\psi}}_\lambda \sim q(\hat{\boldsymbol{\psi}}_\lambda \leftarrow \boldsymbol{\psi}_\lambda)$ ▷ Draw the hyperparameter proposal.
$\{\tilde{\boldsymbol{w}}_j\}_{j=1}^J \sim \mathcal{PP}(\bar{\lambda}(\boldsymbol{x}\,;\,\hat{\boldsymbol{\psi}}_\lambda), \mathcal{V})$ ▷ Draw from the dominating intensity.
$\{\hat{g}(\tilde{\boldsymbol{w}}_j)\}_{j=1}^J \sim \mathcal{GP}(g \,|\, \{\tilde{\boldsymbol{w}}_j\}_{j=1}^J, \boldsymbol{X}, \boldsymbol{G}, \boldsymbol{\theta})$ ▷ Sample the proposed function at the events.
$\boldsymbol{X} \leftarrow \boldsymbol{X} \cup \{\tilde{\boldsymbol{w}}_j\}_{j=1}^J$, $\boldsymbol{G} \leftarrow \boldsymbol{G} \cup \{g(\tilde{\boldsymbol{w}}_j)\}_{j=1}^J$ ▷ Update the proposed conditioning set.
$\mathcal{W} \leftarrow \emptyset$ ▷ Initialise fantasy set.
for $j \leftarrow 1 \ldots J$ **do** ▷ Loop over the events.
    $u_{\mathsf{fant}} \sim \mathcal{U}(0,1)$ ▷ Draw a uniform random variate on $(0,1)$.
    if $u_{\mathsf{fant}} < \Phi(\hat{g}(\tilde{\boldsymbol{w}}_j))$ **then** ▷ Apply the thinning rule.
        $\mathcal{W} \leftarrow \mathcal{W} \cup \tilde{\boldsymbol{w}}_j$ ▷ Add the event to the fantasy set.
    end if
end for

$$a_{\mathsf{sgcp-bihp}} \leftarrow \frac{p_0(\hat{\boldsymbol{\psi}}_\lambda)\; q(\boldsymbol{\psi}_\lambda \leftarrow \hat{\boldsymbol{\psi}}_\lambda)}{p_0(\boldsymbol{\psi}_\lambda)\; q(\hat{\boldsymbol{\psi}}_\lambda \leftarrow \boldsymbol{\psi}_\lambda)} \left[\prod_{k=1}^{K} \frac{\bar{\lambda}(\boldsymbol{x}_n\,;\,\hat{\boldsymbol{\psi}}_\lambda)}{\bar{\lambda}(\boldsymbol{x}_n\,;\,\boldsymbol{\psi}_\lambda)}\right] \prod_{k=1}^{\hat{K}} \frac{\bar{\lambda}(\boldsymbol{w}_n\,;\,\boldsymbol{\psi}_\lambda)}{\bar{\lambda}(\boldsymbol{w}_n\,;\,\hat{\boldsymbol{\psi}}_\lambda)}$$

$u_{\mathsf{mh}} \sim \mathcal{U}(0,1)$ ▷ Draw a uniform random variate on $(0,1)$.
if $u_{\mathsf{mh}} < a_{\mathsf{sgcp-bihp}}$ **then** ▷ Apply the Metropolis–Hastings acceptance rule.
    $\boldsymbol{\psi}_\lambda \leftarrow \hat{\boldsymbol{\psi}}_\lambda$ ▷ Keep the new hyperparameters.
end if
return $\boldsymbol{\psi}_\lambda$, $\boldsymbol{X}$, $\boldsymbol{G}$

Appendix B

Additional Tables

Distance	Landmark 1	Landmark 2
1	Bregma	Right superior incisor
2	Bregma	Left superior incisor
3	Right fronto-zygomatic junction	Left fronto-zygomatic junction
4	Right zygomaxillare superior	Left zygomaxillare superior
5	Right basi-occipital synchondrosis	Right superior incisor
6	Left basi-occipital synchrondrosis	Left superior incisor
7	Right zygomaxillare inferior	Left zygomaxillare inferior
8	Right posterior canine	Right maxillary tuberosity
9	Left posterior canine	Left maxillary tuberosity
10	Right pterion anterior	Left pterion anterior

Table B.1: This table identifies the anatomical landmarks associated with the linear distances used for the modeling of *Macaca mulatta* skulls.

Symbol Glossary

α	The shape parameter for the gamma distribution.
β	The inverse scale parameter for the gamma distribution.
$\boldsymbol{\ell}$	The length scale parameter for covariance functions and repulsion kernels. Has components ℓ_d. See Section 1.2.1.
Γ	An abstract parameter space. See Section 2.2.3 and Section 4.1.
$\Gamma(z)$	The gamma function.
γ	An abstract parameter in the space Γ. See Section 2.2.3 and Section 4.1.
κ	Underrelaxation parameter. Restricted to $(-1, 1)$, but typically close to 1 for underrelaxed proposals. See Section 4.3.2.
$\Lambda(t)$	The cumulative intensity function. See Section 3.2.2.
$\Lambda^{-1}(t)$	The inverse cumulative intensity function. See Section 3.2.2.
$\lambda(\boldsymbol{x})$	A Poisson intensity function.
$\mu(\mathcal{T})$	The measure of the set $\mathcal{T}$.
ν	Poisson rate of a Neyman–Scott daughter process. See Section 3.5.2.
$\Omega(\boldsymbol{x})$	The sum of exponentiated latent functions in the Archipelago model. See Section 8.2.
ω	The amplitude parameter for the squared exponential covariance function. See Section 1.2.1.
$\oplus$	The Minkowski sum, a binary operator.
$\Phi(z)$	A function $\Phi(\cdot) : \mathcal{X} \to (0, 1)$, typically a sigmoid. In this thesis, generally taken to be a logistic function. See Section 2.1.1.
$\phi(\boldsymbol{x})$	A function used for independent thinning. See Section 3.2.3.
$\boldsymbol{\psi}_\lambda$	Hyperparameters of the dominating intensity function $\bar{\lambda}(\boldsymbol{x}\,;\,\boldsymbol{\psi}_\lambda)$. See Section 3.3.
$\rho(\boldsymbol{x} \nleftarrow \boldsymbol{x}')$	A repulsion kernel. See Section 3.5.2.
σ	The standard deviation of a univariate Gaussian distribution.
$\boldsymbol{\theta}$	Hyperparameters of the Gaussian process covariance function. See Section 1.2.
ξ_k	The partition edges in the Archipelago generative model. See Section 8.3.

$\zeta(\cdot, \cdot)$	The Bernoulli probability of proposing to insert a new rejection or thinned event into the latent history model.
a	The Metropolis–Hastings acceptance ratio.
B	The number of control points when using exchange sampling, indexed by b. See Section 4.3.2.
$\mathcal{B}_z(\boldsymbol{x})$	The z-ball centred at $\boldsymbol{x}$. See Section 3.5.2.
$C(\boldsymbol{x}, \boldsymbol{x}\,;\,\boldsymbol{\theta})$	The Gaussian process covariance function $C : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. A positive semidefinite function parameterised by $\boldsymbol{\theta}$. See Section 1.2.
$\mathcal{C}$	The set of control points input locations, i.e. $\mathcal{C} = \{\boldsymbol{x}_b\}_{b=1}^{B}$. See Section 4.3.2.
D	Dimensionality of the space $\mathcal{X}$, indexed by d.
$\mathcal{D}$	The observed data when performing inference with the GPDS. Typically, considered to be N i.i.d. data, i.e. $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^{N}$.
$\mathcal{E}x(x \mid \lambda)$	The exponential distribution with parameter λ.
$f(\boldsymbol{x})$	A probability density function, typically random with a logistic Gaussian process or Gaussian process density sampler prior.
$\boldsymbol{g}$	A vector of infinite length used to denote $g(\boldsymbol{x})$ when it is an object of inference. See Section 1.3.
$\boldsymbol{g}_{\setminus\mathcal{C}}$	The vector $\boldsymbol{g}$, excluding the values at the control points. See Section 4.3.2.
$\boldsymbol{G}$	The conditioning set of outputs for a Gaussian process. See Section 2.2.2.
$\mathcal{GP}(\boldsymbol{g} \mid \boldsymbol{\theta})$	A Gaussian process prior on the function $\boldsymbol{g}$, with hyperparameter $\boldsymbol{\theta}$
$\mathcal{GA}(x \mid \alpha, \beta)$	The gamma distribution with shape parameter α and inverse scale parameter β.
$\mathcal{G}$	The control point function values, i.e. $\mathcal{G} = \{g(\boldsymbol{x}_b) : \boldsymbol{x}_b \in \mathcal{C}\}$. See Section 4.3.2.
$\mathcal{G}_M$	The values of $g(\boldsymbol{x})$ at the rejections or thinned events, when performing latent history inference.
$\mathcal{G}_N$	The values of $g(\boldsymbol{x})$ at the observed data in the GPDS.
$g(\boldsymbol{x})$	A function $g : \mathcal{X} \to \mathbb{R}$ that has a Gaussian process prior. See Section 1.2.
$\mathcal{H}$	The independent draw from a Gaussian process that is used for an underrelaxed proposal. See Section 4.3.2.
$h(\gamma\,;\,\boldsymbol{x})$	The tractable portion of a doubly-intractable likelihood. See Section 4.2.
$\boldsymbol{I}_D$	The $D \times D$ identity matrix.
$\boldsymbol{I}_{\mathcal{V}}(\boldsymbol{x})$	The indicator function for set membership in $\mathcal{V}$. It returns one if $\boldsymbol{x} \in \mathcal{V}$, or zero otherwise. See Section 3.2.1.
K	When discussing point process models, this a number of events. In the Archipelago model, this is the number of classes. Indexed by k.

M	The number of latent rejections, when performing GPDS latent history inference (See Section 5.2.). The number of latent thinned events when performing SGCP latent history inference (See Section 5.3.).
$\mathcal{M}$	The set of latent rejections, or latent thinned events.
N	The number of i.i.d. data generated or observed in the GPDS. The number of distinct sets of events observed in the SGCP. Indexed by n.
$\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma})$	The Gaussian (Normal) distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$.
$\mathcal{PP}(\mathcal{S} \mid \mathcal{V}, \lambda(\boldsymbol{x}))$	A Poisson process on $\mathcal{V}$ with intensity $\lambda(\cdot)$.
$\mathcal{Po}(k \mid \lambda)$	The Poisson distribution with parameter λ.
$\mathcal{Q}$	The support of a daughter or repulsion kernel in an interacting point process. See Section 3.5.2.
$q(\boldsymbol{x})$	A rejection sampling proposal density. See Section 2.2.1.
$q(\hat{\gamma} \leftarrow \gamma)$	A Metropolis–Hastings proposal density. See Section 4.1.
R	The number of iterations of a Monte Carlo procedure, indexed by r.
$\mathbb{R}$	The real line.
$\mathcal{S}$	A finite set of events, as from a point process. See Section 3.1.
$T(\hat{\gamma} \leftarrow \gamma)$	A Markov chain transition operator. See Section 4.1.
t	A temporal event.
$\mathcal{U}(\mathcal{V})$	The uniform distribution on the set $\mathcal{V}$.
$\mathcal{U}(a, b)$	The uniform distribution on (a, b).
u	A uniform variate on a bounded interval, typically $(0, 1)$.
$\mathcal{V}$	The domain of interest for a point process. See Section 3.1.
$\boldsymbol{W}$	The positive definite matrix determining the Mahalanobis distance in a quadratic repulsion kernel. See Section 3.5.2.
$\boldsymbol{w}_n$	The nth fantasy datum when using exchange sampling for the GPDS. See Section 4.3.1.
$\mathcal{W}$	A set of fantasy data.
$\boldsymbol{X}$	A conditioning set of inputs for a Gaussian process. See Section 2.2.2.
$\boldsymbol{x}$	A member of $\mathcal{X}$.
$\mathcal{X}$	The data space for density models and the event space for point process models. Typically this is $\mathbb{R}^D$.
$\mathcal{Y}$	The mark space of a marked Poisson process. See Section 3.5.1.
$\mathcal{Z}[\boldsymbol{g}]$	The normalisation constant for a doubly-intractable model that depends on a function $\boldsymbol{g}$.
$\mathcal{Z}_\pi[\boldsymbol{g}]$	The normalisation constant for the GPDS likelihood model. See Section 2.1.

Bibliography

Petter Abrahamsen. A review of Gaussian random fields and correlation functions. Technical Report 917, Norwegian Computing Center, Oslo, Norway, April 1997. (page 4)

Milton Abramowitz and Irene A. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Dover, New York, ninth edition, 1964. (page 15)

David H. Ackley, Geoffrey E. Hinton, and Terrence J. Sejnowski. A learning algorithm for Boltzmann machines. *Cognitive Science*, 9(1):147–169, 1985. (pages 8 and 46)

Ryan Prescott Adams and Zoubin Ghahramani. Archipelago: Nonparametric Bayesian semi-supervised learning. In Leon Bottou and Michael Littman, editors, *Proceedings of the 26th International Conference on Machine Learning*, 2009. To Appear. (page 12)

Ryan Prescott Adams and Oliver Stegle. Gaussian process product models for nonparametric nonstationarity. In Andrew McCallum and Sam Roweis, editors, *Proceedings of the 25th International Conference on Machine Learning*, pages 1–8, 2008.

Ryan Prescott Adams, Iain Murray, and David J. C. MacKay. Nonparametric Bayesian density modeling with Gaussian processes. In *ICML/UAI/COLT Workshop on Nonparametric Bayes*, 2008. (page 12)

Ryan Prescott Adams, Iain Murray, and David J. C. MacKay. The Gaussian process density sampler. In *Advances in Neural Information Processing Systems 21*, Cambridge, MA, 2009a. MIT Press. (page 12)

Ryan Prescott Adams, Iain Murray, and David J. C. MacKay. Nonparametric Bayesian density modeling with Gaussian processes. In Preparation, 2009b. (page 12)

Ryan Prescott Adams, Iain Murray, and David J. C. MacKay. Tractable nonparametric Bayesian inference in Poisson processes with Gaussian process intensities. In Leon Bottou and Michael Littman, editors, *Proceedings of the 26th International Conference on Machine Learning*, 2009c. To Appear. (page 12)

Stephen L. Adler. Over-relaxation method for the Monte Carlo evaluation of the partition function for multiquadratic actions. *Physics Review D*, 23(12):2901–2904, 1981. (page 55)

S. Aeberhard, D. Coomans, and O. de Vel. Comparison of classifiers in high dimensional settings. Technical Report 92-02, Department of Computer Science, James Cook University, North Queensland, 1992. (page 109)

Christophe Andrieu, Nando de Freitas, Arnaud Doucet, and Michael I. Jordan. Introduction to MCMC for machine learning. *Machine Learning*, 50(1):5–43, 2003. (page 44)

Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, 1950. (page 5)

A. Asuncion and D.J. Newman. UCI machine learning repository, 2007. (page 109)

Melvin Avrami. Kinetics of phase change I: General theory. *Journal of Chemical Physics*, 7(12):1103–1112, 1939. (page 31)

Melvin Avrami. Kinetics of phase change II: Transformation-time relations for random distribution of nuclei. *Journal of Chemical Physics*, 8(2):212–224, 1940. (page 31)

Melvin Avrami. Kinetics of phase change III: Granulation, phase change, and microstructure. *Journal of Chemical Physics*, 9(2):177–184, 1941. (page 31)

Claus Beisbart and Martin Kerscher. Luminosity- and morphology-dependent clustering of galaxies. *The Astrophysical Journal*, 545(1):6–25, December 2000. (page 30)

José M. Bernardo. The concept of exchangeability and its applications. *Far East Journal of Mathematical Sciences*, 4:111–121, 1996. (page 19)

Alexandros Beskos, Omiros Papaspiliopoulos, and Gareth O. Roberts. Retrospective exact simulation of diffusion sample paths with applications. *Bernoulli*, 12(6):1077–1098, 2006a. (page 52)

Alexandros Beskos, Omiros Papaspiliopoulos, Gareth O. Roberts, and Paul Fearnhead. Exact and computationally efficient likelihood-based estimation for discretely observed diffusion processes. *Journal of the Royal Statistical Society, Series B*, 68:333–382, 2006b. (pages 52 and 66)

Edwin V. Bonilla, Kian Ming A. Chai, and Christopher K. I. Williams. Multi-task Gaussian process prediction. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing System 20*, pages 153–160. MIT Press, Cambridge, MA, 2008. (page 115)

Karsten M. Borgwardt. *Graph Kernels*. PhD thesis, Ludwig-Maximilians University, Munich, 2007. (page 114)

Karsten M. Borgwardt, Cheng Soon Ong, Stefan Schönauer, S. V. N. Vishwanathan, Alex J. Smola, and Hans-Peter Kriegel. Protein function prediction via graph kernels. *Bioinformatics*, 21(1):47–56, 2005. (page 114)

Anders Brix and Joël Chadœuf. Spatio-temporal modelling of weeds by shot-noise G Cox processes. *Biometrical Journal*, 44(1):83–99, 2002. (page 32)

Emery N. Brown. *Theory of Point Processes for Neural Systems*, chapter 14, pages 691–726. Elsevier, Paris, 2005. (page 9)

Siddhartha Chib and Ivan Jeliazkov. Marginal likelihood from the Metropolis–Hastings output. *Journal of the American Statistical Association*, 96(453):270–281, 2001. (page 95)

Wei Chu, Vikas Sindhwani, Zoubin Ghahramani, and S. Sathiya Keerthi. Relational learning with Gaussian processes. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 289–296, Cambridge, MA, 2007. MIT Press. (page 101)

Michael Collins and Nigel Duffy. Convolution kernels for natural language. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing System 14*, pages 625–632, Cambridge, MA, 2001. MIT Press. (page 114)

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, Cambridge, MA, second edition, 2001. (pages 21 and 83)

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. John Wiley and Sons, Inc., Hoboken, New Jersey, second edition, 2006. (page 44)

David R. Cox. Some statistical methods connected with series of events. *Journal of the Royal Statistical Society, Series B*, 17(2):129–164, 1955. (page 9)

Noel A. C. Cressie. *Statistics for Spatial Data*. Wiley-Interscience, New York, 1991. (page 3)

Lehel Csató. *Gaussian Processes - Iterative Sparse Approximations*. PhD thesis, Aston University, Birmingham, UK, March 2002. (page 10)

John P. Cunningham, Krishna V. Shenoy, and Maneesh Sahani. Fast Gaussian process methods for point process intensity estimation. In Andrew McCallum and Sam Roweis, editors, *Proceedings of the 25th International Conference on Machine Learning*, pages 192–199, 2008a. (pages 11 and 12)

John P. Cunningham, Byron M. Yu, Krishna V. Shenoy, and Maneesh Sahani. Inferring neural firing rates from spike trains using Gaussian processes. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, Cambridge, MA, 2008b. MIT Press. (page 9)

Luc Devroye. *Non-Uniform Random Variate Generation*. Springer–Verlag, New York, 1986. (pages 17, 25, and 26)

Peter Diggle. A kernel method for smoothing point process data. *Applied Statistics*, 34 (2):138–147, 1985. (pages 1 and 91)

Peter Diggle. *Statistical Analysis of Spatial Point Patterns*. Oxford University Press, Oxford, 2003. (page 36)

Ilaria DiMatteo, Christopher R. Genovese, and Robert E. Kass. Bayesian curve-fitting with free-knot splines. *Biometrika*, 88(4):1055–1071, 2001. (page 2)

Simon Duane, A. D. Kennedy, Brian J. Pendleton, and Duncan Roweth. Hybrid Monte Carlo. *Physics Letters B*, 195(2):216–222, 1987. (pages 45, 70, and 75)

David B. Dunson and Ju-Hyun Park. Kernel stick-breaking processes. *Biometrika*, 95 (2):307–323, 2008. (page 114)

Michael D. Escobar and Mike West. Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, 90(430):577–588, June 1995. (pages 2, 101, and 115)

Thomas S. Ferguson. A Bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1(2):209–230, 1973. (pages 2 and 16)

Thomas S. Ferguson. Prior distributions on spaces of probability measures. *The Annals of Statistics*, 2(4):615–629, 1974. (page 2)

Thomas Gärtner, Peter Flach, and Stefan Wrobel. On graph kernels: Hardness results and efficient alternatives. In *Proceedings of the 16th Annual Conference on Computational Learning Theory and 7th Kernel Workshop*, pages 129–143. Springer–Verlag, August 2003. (page 114)

Andrew Gelman, John B. Carlin, Hal S. Stern, and Donald B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, New York, second edition, 2004. (pages 44 and 71)

Stuart Geman and Donald Geman. Stochastic relaxation, Gibbs distribution and Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984. (page 45)

Jayanta K. Ghosh and R. V. Ramamoorthi. *Bayesian Nonparametrics*. Springer-Verlag, Berlin, 2003. (page 2)

Mark N. Gibbs. *Bayesian Gaussian Processes for Regression and Classification*. PhD thesis, University of Cambridge, Cambridge, 1997. (page 4)

Kay Giesecke and Baeho Kim. Estimating tranche spreads by loss process simulation. In S. G. Henderson, B. Biller, M. H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, editors, *Proceedings of the 2007 Winter Simulation Conference*, pages 967–975, 2007. (page 34)

Mark Girolami and Simon Rogers. Variational Bayesian multinomial probit regression with Gaussian process priors. *Neural Computation*, 18(8):1790–1817, 2006. (page 108)

Leon Glass and Waldo R. Tobler. Uniform distribution of objects in a homogeneous field: cities on a plain. *Nature*, 233(5314):67–68, September 1971. (pages 36 and 38)

Noah D. Goodman, Vikash K. Mansinghka, Daniel M. Roy, Keith Bonawitz, and Joshua B. Tenenbaum. Church: a language for generative models. In *Proceedings of the 24th Annual Conference on Uncertainty in Artificial Intelligence*, 2008. (page 66)

Peter J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model selection. *Biometrika*, 82:711–732, 1995. (page 1)

Phil C. Gregory and Thomas J. Loredo. A new method for the detection of a periodic signal of unknown shape and period. *The Astrophysical Journal*, 398:146–168, October 1992. (page 9)

Jim E. Griffin and Mark F. J. Steel. Time-depenent stick-breaking processes. Technical report, Institute of Mathematics, Statistics and Actuarial Science, University of Kent, 2009. (page 114)

Thomas L. Griffiths and Zoubin Ghahramani. Infinite latent feature models and the Indian buffet process. In *Advances in Neural Information Processing Systems 18*, pages 475–482, Cambridge, MA, 2006. MIT Press. (page 2)

Olle Häggström, Marie-Colette N. M. van Lieshout, and Jesper Møller. Characterization results and Markov chain Monte Carlo algorithms including exact simulation for some spatial point processes. *Bernoulli*, 5(4):641–659, 1999. (page 32)

W. Keith Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, April 1970. (page 45)

David Haussler. Convolution kernels on discrete structures. Technical Report UCSC-CRL-99-10, University of California at Santa Cruz, 1999. (pages 113 and 114)

Alan G. Hawkes. Point spectra of some mutually exciting point processes. *Journal of the Royal Statistical Society, Series B*, 33(3):438–443, 1971a. (pages 31 and 34)

Alan G. Hawkes. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1):83–90, April 1971b. (pages 31 and 34)

Alan G. Hawkes. Spectra of some mutually exciting point processes with associated variables. In Peter A. W. Lewis, editor, *Stochastic Point Processes*, pages 261–271. John Wiley and Sons, New York. (pages 31 and 34)

Alan G. Hawkes and Leonidas Adamopoulos. Cluster models for earthquakes – regional comparisons. *Bulletin of the International Statistical Institute*, 45:454–461, 1973. (page 34)

Alan G. Hawkes and David Oakes. A cluster representation of a self-exciting process. *Journal of Applied Probability*, 11(3):493–503, September 1974. (pages 31 and 34)

Juha Heikkinen and Elja Arjas. Modeling a Poisson forest in variable elevations: a nonparametric Bayesian approach. *Biometrics*, 55:738–745, 1999. (page 9)

Edwin Hewitt and Leonard Jimmie Savage. Symmetric measures on Cartesian products. *Transactions of the American Mathematical Society*, 80:470–501, 1955. (page 19)

Philip Holgate. Studies in the history of probability and statistics XXXIX: Buffon's cycloid. *Biometrika*, 68(3):712–716, December 1981. (page 18)

John J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the USA*, 79 (8):2554–2558, April 1982. (page 46)

Mark L. Huber and Robert L. Wolpert. Perfect simulation of Matérn type III repulsive point processes. In Preparation, 2009. (pages 37, 38, 41, 66, and 81)

Hemant Ishwaran and Lancelot F. James. Gibbs sampling methods for stick-breaking priors. *Journal of the American Statistical Association*, 96(453):161–173, March 2001. (page 2)

Ernst Ising. Beitrag zur theorie des ferromagnetismus. *Zeitschrift für Physik*, 31:253–258, 1925. (page 46)

R. G. Jarrett. A note on the intervals between coal-mining disasters. *Biometrika*, 66(1): 191–193, 1979. (page 93)

Don H. Johnson. Point process models of single-neuron discharges. *Journal of Computational Neuroscience*, 3(4):275–299, December 1996. (page 34)

William A. Johnson and Robert F. Mehl. Reaction kinetics in processes of nucleation and growth. *Transactions of the American Institute of Mining, Metallurgical and Petroleum Engineers*, 135:410–458, 1939. (page 31)

Francis P. Kelly and Brian D. Ripley. A note on Strauss's model for clustering. *Biometrika*, 63(2):357–360, August 1976. (pages 31, 38, and 48)

Wilfrid S. Kendall. Perfect simulation for spatial point processes. In *Proceedings of the 51st Session of the International Statistical Institute: Simulation of Stochastic Processes in Engineering, Istanbul*, pages 163–166, 1997. (page 32)

Wilfrid S. Kendall and Jesper Møller. Perfect simulation using dominating processes on ordered spaces, with application to locally stable point processes. *Advances in Applied Probability*, 32(3):844–865, 2000. (page 32)

Andrey N. Kolmogorov. Zur statistik der kristallisationsvorgänge in metallen. *Izvestiya Rossiiskoi Akademii Nauk, Seriya Matematicheskaya*, 1(3):355–359, 1937. (page 31)

Andrey N. Kolmogorov. *Selected Works of A. N. Kolmogorov, Volume II: Probability Theory and Mathematical Statistics*. Mathematics and Its Applications. Springer-Verlag, Berlin, 1992. (page 31)

Risi I. Kondor. *Group Theoretical Methods in Machine Learning*. PhD thesis, Columbia University, New York, NY, May 2008. (page 113)

Risi I. Kondor and Tony Jebara. A kernel between sets of vectors. In T. Fawcett and N. Mishra, editors, *Proceedings of the 20th International Conference on Machine Learning*, pages 361–368, 2003. (page 114)

Risi I. Kondor and John D. Lafferty. Diffusion kernels on graphs and other discrete structures. In Claude Sammut and Achim G. Hoffmann, editors, *Proceedings of the 19th International Conference on Machine Learning*, pages 315–322, 2002. (pages 113 and 114)

Athanasios Kottas and Bruno Sansó. Bayesian mixture modeling for spatial Poisson process intensities, with applications to extreme value analysis. *Journal of Statistical Planning and Inference*, 137:3151–3163, 2007. (page 2)

Michael Lavine. Some aspects of Pólya tree distributions for statistical modelling. *The Annals of Statistics*, 20(3):1222–1235, 1992. (page 2)

Michael Lavine. More aspects of Pólya tree distributions for statistical modelling. *The Annals of Statistics*, 22(3):1161–1175, 1994. (page 2)

Neil D. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, 2005. (pages 10 and 11)

Neil D. Lawrence and Michael I. Jordan. Semi-supervised learning via Gaussian processes. In L. Saul, Y. Weiss, and L. Bouttou, editors, *Advances in Neural Information Processing Systems 17*, Cambridge, MA, 2005. MIT Press. (pages 101, 108, and 109)

Andrew B. Lawson and David G. T. Dension. *Spatial Cluster Modelling*. Monographs on Statistics and Applied Probability. Chapman and Hall/CRC, Boca Raton, FL, 2002. (page 32)

Georges-Louis Leclerc de Buffon. Essai d'arithmétique morale. In *Supplément à l'Histoire Naturelle*, volume 4. Imprimerie Royale, Paris, 1777. (page 18)

Subhash R. Lele and Joan T. Richtsmeier. *An Invariant Approach to Statistical Analysis of Shapes*. Chapman and Hall/CRC Press, London, 2001. (page 90)

Gareth Leng, Colin H. Brown, Philip M. Bull, David Brown, Sinead Scullion, James Currie, Ruth E. Blackburn-Monroe, Jianfeng Feng, Tatsushi Onaka, Joseph G. Verbalis, John A. Russell, and Mike Ludwig. Responses of magnocellular neurons to osmotic stimulation involves coactivation of excitatory and inhibitory input: an experimental and theoretical analysis. *The Journal of Neuroscience*, 21(17):6967–6977, September 2001. (page 40)

Peter J. Lenk. The logistic normal distribution for Bayesian, nonparametric, predictive densities. *Journal of the American Statistical Association*, 83(402):509–516, 1988. (page 7)

Peter J. Lenk. Towards a practicable Bayesian nonparametric density estimator. *Biometrika*, 78(3):531–543, 1991. (pages 7, 8, and 85)

Peter J. Lenk. Bayesian semiparametric density estimation and model verification using a logistic-Gaussian process. *Journal of Computational and Graphical Statistics*, 12 (3):548–565, 2003. (page 8)

Tom Leonard. Density estimation, stochastic processes and prior information. *Journal of the Royal Statistical Society, Series B*, 40(2):113–146, 1978. (page 7)

Christina S. Leslie, Eleazar Eskin, Adiel Cohen, Jason Weston, and William Stafford Noble. Mismatch string kernels for discriminative protein classification. *Bioinformatics*, 20(4):467–476, 2004. (page 114)

Peter A. W. Lewis and Gerald S. Shedler. Simulation of a nonhomogeneous Poisson process by thinning. *Naval Research Logistics Quarterly*, 26:403–413, 1979. (page 26)

D. V. Lindley. Approximate Bayesian methods. In J. M. Bernardo, M. H. DeGroot, D. V. Lindley, and A. F. M. Smith, editors, *Bayesian Statistics*, pages 223–237. Valencia University Press, Valencia, 1980. (page 1)

Albert Y. Lo. On a class of Bayesian nonparametric estimates: I. density estimates. *The Annals of Statistics*, 12(1):351–357, March 1984. (page 2)

Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Chris Watkins. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444, February 2002. (page 114)

Steven N. MacEachern. Dependent nonparametric processes. Technical report, Department of Statistics, The Ohio State University, 2000. (page 114)

David J. C. MacKay. Bayesian interpolation. *Neural Computation*, 4(3):415–447, 1992a. (page 2)

David J. C. MacKay. A practical Bayesian framework for backpropagation networks. *Neural Computation*, 4(3):448–472, 1992b. (page 1)

David J. C. MacKay. Bayesian neural networks and density networks. *Nuclear Instruments and Methods in Physics Research, Section A*, 354(1):73–80, 1995. (page 11)

David J. C. MacKay. Introduction to Gaussian processes. In C.M. Bishop, editor, *Neural Networks and Machine Learning*, NATO ASI Series, pages 133–166. Kluwer, 1998a. (page 5)

David J. C. MacKay. Choice of basis for Laplace approximation. *Machine Learning*, 33 (1):77–86, 1998b. (page 1)

David J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, Cambridge, 2003. (pages 17, 18, 44, and 70)

Prasanta C. Mahalanobis. On the generalised distance in statistics. *Proceedings of the National Institute of Sciences of India*, 2(1):49–55, 1936. (page 41)

Bertil Matérn. Spatial variation. *Meddelanden från Statens Skogsforskningsinstitut (Reports of the Forest Research Institute of Sweden)*, 49(5), 1960. (pages 31, 33, and 37)

Bertil Matérn. *Spatial Variation*. Lecture Notes in Statistics. Springer-Verlag, Berlin, second edition, 1986. (page 31)

R. Daniel Mauldin, William D. Sudderth, and S. C. Williams. Pólya trees and random distributions. *The Annals of Statistics*, 20(3):1203–1221, September 1992. (page 2)

Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21(6):1087–1092, 1953. (page 45)

Paul-André Meyer. Démonstration simplifiée d'un théorème de Knight. *Séminaire des Probabilités de Strasbourg V*, 191:191–195, 1971. (page 26)

Thomas P. Minka. *A Family of Algorithms for Approximate Bayesian Inference*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, January 2001. (page 1)

Ilya Molchanov. *Statistics of the Boolean Model for Practitioners and Mathematicians*. Probability and Statistics. John Wiley and Sons, New York, 1997. (page 31)

Jesper Møller and Giovanni Luca Torrisi. Generalised shot noise Cox processes. *Advances in Applied Probability*, 37(1):48–74, 2005. (page 32)

Jesper Møller and Rasmus Plenge Waagepetersen. *Statistical Inference and Simulation for Spatial Point Processes*. Monographs on Statistics and Applied Probability. Chapman and Hall/CRC, Boca Raton, FL, 2004. (pages 9, 23, 24, 30, 31, and 33)

Jesper Møller, Anne Randi Syversveen, and Rasmus Plenge Waagepetersen. Log Gaussian Cox processes. *Scandinavian Journal of Statistics*, 25:451–482, 1998. (pages 9, 10, 12, and 91)

Jesper Møller, Anthony N. Pettitt, Robert Reeves, and Kasper K. Berthelsen. An efficient Markov chain Monte Carlo method for distributions with intractable normalising constants. Technical Report R-2004-02, Department of Mathematical Sciences, Aalborg University, 2004. (pages 46, 47, and 48)

Jesper Møller, Anthony N. Pettitt, Robert Reeves, and Kasper K. Berthelsen. An efficient Markov chain Monte Carlo method for distributions with intractable normalising constants. *Biometrika*, 93(2):451–458, 2006. (pages 47 and 48)

Peter Müller, Fernando Quintana, and Gary Rosner. A method for combining inference across related nonparametric Bayesian models. *Journal of the Royal Statistical Society, Series B*, 66(3):735–749, 2004. (pages 16 and 114)

Iain Murray. *Advances in Markov Chain Monte Carlo Methods*. PhD thesis, Gatsby Computational Neuroscience Unit, University College London, London, 2007. (pages 46, 48, 56, and 66)

Iain Murray, Zoubin Ghahramani, and David J.C. MacKay. MCMC for doubly-intractable distributions. In *Proceedings of the 22nd Annual Conference on Uncertainty in Artificial Intelligence*, pages 359–366, 2006. (pages 8 and 48)

Radford M. Neal. Priors for infinite networks. Technical Report CRG-TR-94-1, Department of Computer Science, University of Toronto, 1994. (page 3)

Radford M. Neal. *Bayesian Learning for Neural Networks*. Springer–Verlag, Berlin, 1996. (pages 5 and 70)

Radford M. Neal. Monte Carlo implementation of Gaussian process models for Bayesian regression and classification. Technical Report 9702, Department of Statistics, University of Toronto, 1997. (page 107)

Radford M. Neal. Supressing random walks in Markov chain Monte Carlo using ordered overrelaxation. In *Learning in Graphical Models*, pages 205–225. Kluwer Academic Publishers, Dordrecht, 1998. (pages 55 and 71)

Radford M. Neal. Markov chain sampling methods for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9(2):249–265, 2000. (page 89)

Radford M. Neal. Defining priors for distributions using Dirichlet diffusion trees. Technical Report 0104, Department of Statistics, University of Toronto, 2001. (page 2)

Radford M. Neal. Density modeling and clustering using Dirichlet diffusion trees. In *Bayesian Statistics 7*, pages 619–629, 2003a. (pages 2 and 89)

Radford M. Neal. Slice sampling (with discussion). *The Annals of Statistics*, 31(3):705–767, 2003b. (page 45)

Radford M. Neal. Improving asymptotic variance of MCMC estimators: non-reversible chains are better. Technical Report 0406, Department of Statistics, University of Toronto, 2004. (page 45)

Roger B. Nelsen. *An Introduction to Copulas*. Springer–Verlag, Berlin, second edition, 2007. (page 15)

Jerzy Neyman and Elizabeth L. Scott. Statistical approach to cosmology. *Journal of the Royal Statistical Society, Series B*, 20(1):1–43, 1958. (pages 31 and 33)

Martin Niss. History of the Lenz–Ising model 1920-1950: From ferromagnetic to cooperative phenomena. *Archive for History of Exact Sciences*, 59(3):267–318, March 2005. (page 46)

Yosihiko Ogato and Masaharu Tanemura. Likelihood estimation of soft-core interaction potentials for Gibbsian point patterns. *Annals of the Institute of Statistical Mathematics*, 41(3):583–600, 1989. (pages 31 and 39)

Anthony O'Hagan. Curve fitting and optimal design for prediction. *Journal of the Royal Statistical Society, Series B*, 40(1):1–42, 1978. (pages 2 and 3)

Christopher J. Paciorek. *Nonstationary Gaussian Processes for Regression and Spatial Modelling*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, 2003. (page 4)

Fredos Papangelou. Integrability of expected increments of point processes and a related random change of scale. *Transactions of the American Mathematical Society*, 165:483–506, March 1972. (page 26)

Omiros Papaspiliopoulos and Gareth O. Roberts. Retrospective Markov chain Monte Carlo methods for Dirichlet process hierarchical models. *Biometrika*, 95(1):169–186, 2008. (page 52)

Emanuel Parzen. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, 1962. (pages 1 and 89)

Antti Penttinen and Dietrich Stoyan. Statistical analysis for a class of line segment processes. *Scandinavian Journal of Statistics*, 16(2):153–168, 1989. (pages 36 and 37)

Jim Pitman and Marc Yor. The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *The Annals of Probability*, 25:855–900, 1997. (page 2)

James G. Propp and David B. Wilson. Exact sampling with coupled Markov chains and applications to statistical mechanics. *Random Structures and Algorithms*, 9(1&2): 223–252, 1996. (pages 32, 48, and 66)

Joaquin Quiñonero-Candela and Carl Edward Rasmussen. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6: 1935–1959, December 2005. (page 112)

Carl Edward Rasmussen. *Evaluation of Gaussian Processes and Other Methods for Non-Linear Regression*. PhD thesis, University of Toronto, Toronto, Ontario, March 1996. (pages 55, 70, and 71)

Carl Edward Rasmussen. The infinite Gaussian mixture model. In S.A. Solla, T.K. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems 12*, pages 554–560, Cambridge, MA, 2000. MIT Press. (pages 2, 89, 101, and 115)

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006. (pages 3, 4, 5, 14, 109, and 113)

Stephen L. Rathbun and Noel Cressie. Asymptotic properties of estimators for the parameters of spatial inhomogeneous Poisson point processes. *Advances in Applied Probability*, 26(1):122–154, March 1994. (pages 9 and 91)

Brian D. Ripley. Modelling spatial patterns (with discussion). *Journal of the Royal Statistical Society, Series B*, 39:172–212, 1977. (page 93)

Brian D. Ripley. *Spatial Statistics*. Probability and Mathematical Statistics. John Wiley and Sons, New York, 1981. (pages 36 and 38)

Christian P. Robert and George Casella. *Monte Carlo Statistical Methods*. Springer–Verlag, Berlin, second edition, 2004. (page 17)

Simon Rogers and Mark Girolami. Multi-class semi-supervised learning with the ϵ-truncated multinomial probit Gaussian process. In *JMLR Workshop and Conference Proceedings: Gaussian Processes in Practice*, volume 1, pages 17–32, 2007. (pages 108 and 109)

Murray Rosenblatt. Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics*, 27(3):832–837, September 1956. (pages 1 and 89)

Gideon Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464, 1978. (page 1)

Vikas Sindhwani, Wei Chu, and S. Sathiya Keerthi. Semi-supervised Gaussian process classifiers. In *International Joint Conference on Artificial Intelligence*, pages 1059–1064, 2007. (page 101)

Paul Smolensky. Information processing in dynamical systems: foundations of harmony theory. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Volume 1*, pages 194–281. MIT Press, Cambridge, MA, 1986. (pages 8 and 46)

Edward Snelson, Carl Edward Rasmussen, and Zoubin Ghahramani. Warped Gaussian processes. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, Cambridge, MA, 2004. MIT Press. (page 14)

Nathan Srebro and Sam Roweis. Time-varying topic models using dependent Dirichlet processes. Technical Report UTML TR 2005-003, Department of Computer Science, University of Toronto, Toronto, ON, 2005. (page 114)

Richard B. Stein. A theoretical analysis of neuronal variability. *Biophysical Journal*, 5 (2):173–194, March 1965. (page 40)

Dietrich Stoyan and Helga Stoyan. On one of Matérn's hard-core point process models. *Mathematische Nachrichten*, 122(1):205–214, 1985. (page 37)

Dietrich Stoyan and Helga Stoyan. *Fractals, Random Shapes and Point Fields*. Probability and Mathematical Statistics. John Wiley and Sons, New York, 1994. (pages 9, 33, and 36)

Dietrich Stoyan and Helga Stoyan. Non-homogeneous Gibbs process models for forestry – a case study. *Biometrical Journal*, 40(5):521–531, 1998. (pages 31 and 39)

David J. Strauss. A model for clustering. *Biometrika*, 62(2):467–475, August 1975. (pages 31, 38, and 48)

Yee Whye Teh, Matthias Seeger, and Michael I. Jordan. Semiparametric latent factor models. In R. G. Cowell and Z. Ghahramani, editors, *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, pages 333–340, 2005. (page 115)

Malvin Carl Teich, Leonard Matin, and Barry I. Cantor. Refractoriness in the maintained discharge of the cat's retinal ganglion cell. *Journal of the Optical Society of America*, 68(3):386–402, 1978. (page 40)

Marjorie Thomas. A generalization of Poisson's binomial limit for use in ecology. *Biometrika*, 36(1):18–25, June 1949. (page 33)

Daniel Thorburn. A Bayesian approach to density estimation. *Biometrika*, 73(1):65–75, 1986. (page 7)

Michael E. Tipping and Christopher M. Bishop. Probabilistic principal components analysis. *Journal of the Royal Statistical Society, Series B*, 61(3):611–622, 1999. (page 11)

Surya T. Tokdar. *Exploring Dirichlet Mixture and Logistic Gaussian Process Priors in Density Estimation, Regression and Sufficient Dimension Reduction*. PhD thesis, Purdue University, West Lafayette, Indiana, USA, August 2006. (page 7)

Surya T. Tokdar. Towards a faster implementation of density estimation with logistic Gaussian process priors. *Journal of Computational and Graphical Statistics*, 16(2):1–23, 2007. (pages 8, 54, 85, and 87)

Surya T. Tokdar and Jayanta K. Ghosh. Posterior consistency of logistic Gaussian process priors in density estimation. *Journal of Statistical Planning and Inference*, 137: 34–42, 2007. (pages 7 and 16)

Vladmir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, Berlin, 1995. (page 11)

William N. Venables and Brian D. Ripley. *Modern Applied Statistics with S-PLUS*. Statistics and Computing. Springer–Verlag, Berlin, 1998. (page 36)

Francesco Vivarelli and Christopher K. I. Williams. Discovering hidden features with Gaussian process regression. In M. J. Kearns, S. A. Solla, and D. A. Cohn, editors, *Advances in Neural Information Processing Systems 11*, Cambridge, MA, 1999. MIT Press. (page 41)

John von Neumann. Various techniques used in connection with random digits. *National Bureau of Standards, Applied Mathematics Series*, 12:36–38, 1951. (page 18)

Stephen G. Walker, Paul Damien, Purushottam W. Laud, and Adrian F. M. Smith. Bayesian nonparametric inference for random distributions and related functions. *Journal of the Royal Statistical Society, Series B*, 61(3):485–527, 1999. (page 2)

Christopher J. C. H. Watkins. Dynamic alignment kernels. In A. Smola and P. Bartlett, editors, *Advances in Large Margin Classifiers*, chapter 3, pages 39–50. MIT Press, Cambridge, MA, 2000. (page 114)

Norbert Wiener. *Extrapolation, Interpolation and Smoothing of Stationary Time Series*. MIT Press, Cambridge, MA, 1949. (page 3)

Christopher K. I. Williams and David Barber. Bayesian classification with Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1342–1351, 1998. (pages 108 and 109)

Christopher K. I. Williams and Carl Edward Rasmussen. Gaussian processes for regression. In D.S. Touretzky, M.C. Mozer, and M.E. Hasselmo, editors, *Advances in Neural Information Processing Systems 8*, pages 514–520, Cambridge, MA, 1996. MIT Press. (pages 3 and 71)

Katherine E. Willmore, Christian P. Klingenberg, and Benedikt Hallgrimsson. The relationship between fluctuating asymmetry and environmental variance in rhesus macaque skulls. *Evolution*, 59(4):898–909, 2005. (page 90)

David B. Wilson. How to couple from the past using a read-once source of randomness. *Random Structures and Algorithms*, 16(1):85–113, 2000. (page 32)

Nikos Yannaros. On Cox processes and gamma renewal processes. *Journal of Applied Probability*, 25(2):423–427, June 1988. (page 40)