Flexible Birth-Death MCMC Sampler for Changepoint Models

vorgelegt von
Diplom-Informatiker
Florian Stimberg
geb. in Berlin

von der Fakultät IV - Elektrotechnik und Informatik
der Technischen Universität Berlin
zur Erlangung des akademischen Grades
Dr. rer. nat
genehmigte Dissertation

Promotionsausschuss:

Vorsitzender:	Prof. Dr. Olaf Hellwich
Gutachter:	Prof. Dr. Manfred Opper
Gutachter:	Prof. Dr. Guido Sanguinetti
Gutachter:	Prof. Dr. Klaus Obermayer

Tag der wissenschaftlichen Aussprache: 9. November 2015

Berlin 2016

Acknowledgements

First, I want to thank my supervisor Manfred Opper for managing to always have time for discussions, helpful advices and interesting conversations over the last 5 years.

Andreas Ruttor was always there when I had questions and had the patience to answer them all. I don't know how I would have coped without his help.

Many thanks to Klaus Obermayer for agreeing to be part of my reviewing committee.

I was always happy to meet Andrea Ocone and Botond Cseke. Working together with them was a blast and the time together off work was even better. The same is true for Guido Sanguinetti, who I also like to thank for joining my reviewing committee.

I would also like to thank Ludovica Bachschmid Romana, Philipp Batz, Gina Grünhage, Cordula Lippke, Alex Susemihl, Sebastian Thiel and Fritz Wysotzki for the talks over lunch and between work which made dealing with the stressful times that much easier.

I'd like to thank the FWE group for letting me waste precious hours, which I could probably have used more efficiently.

I can't thank my parents enough for all their support over the years. I would certainly not have gotten to this point without them. My brothers Till and Marcel always helped me with their 6 and 4 years of experience they have ahead of me (at least).

Finally, I want to thank my wife Anja for always being there for me and making the tough times easier and the good times even better.

Zusammenfassung

Diese Arbeit beschreibt eine flexible Architektur eines Markov Chain Monte Carlo Samplers, der Bayessche Inferenz für eine Vielzahl von Changepoint-Modellen erlaubt. Die Struktur dieser Klasse von Modellen besteht aus zwei stochastischen Prozessen. Der erste Prozess wird entweder direkt beobachtet oder indirekt durch, möglicherweise verrauschte, Beobachtungen. Der zweite Prozess ist unbeobachtet und bestimmt die Parameter des beobachteten Prozesses. Die Hauptannahme unserer Modellklasse ist, dass der versteckte Prozess stückweise konstant ist, d.h. er springt zwischen diskreten Zuständen.

Als beobachteter Prozess diskutieren wir hauptsächlich den Ornstein-Uhlenbeck und Poisson Prozess. Der versteckte Prozess kann eine feste Anzahl von Zuständen haben oder eine unbekannte Anzahl. Im zweiten Fall basiert das Modell auf einem versteckten Chinese Restaurant Prozess und ermöglicht so Bayessche Inferenz über die Anzahl der Zustände des versteckten Parameterprozesses. Der Sampler wendet einen Metropolis-Hastings Random Walk auf den versteckten Prozess an indem Birth-Death Schritte vorgeschlagen werden. Die Arbeit präsentiert unterschiedliche Modifikationen des Pfades des versteckten Prozesses. Die Struktur des Samplers ist sehr flexibel und lässt sich, im Vergleich zu anderen Algorithmen, die für ein spezifisches Modell maßgeschneidert sind, einfach an verschiedene Kombinationen von beobachteten und versteckten Prozessen anpassen.

Angewandt auf Genexpressionsdaten ermöglicht der Sampler Bayessche Inferenz für komplexere Modelle als vorherige Methoden. Der berechnete Bayes Faktor deutet an, dass unser Modell, welches es erlaubt die Stärke des intrinsischen Rauschens zu variieren, die Daten besser erklärt als das vorherige Modell. Der Sampler wird für Genexpressionsdaten von Hefezellen benutzt und die Ergebnisse mit denen einer variationellen Näherung verglichen. Der Posterior scheint genauer in der Vorhersage der Aktivierungszeitpunkte der Transkriptionsfaktoren zu sein als es die Näherung zeigt. Die Ergebnisse des Chinese Restaurant Prozess Samplers auf den gleichen Messungen von Hefezellen unterstützt die vorherige Annahme über die Anzahl der Transkriptionsfaktoren, die in die Kontrolle der untersuchten Gene involviert sind.

Die Anpassung des Samplers an Markov modulierte Poisson Prozesse beschleunigt die Inferenz und dies wird gezeigt, indem die Zeit zur Berechnung eines unkorrelierten Samples mit einem exakten Gibbs Sampler verglichen wird. Ein Modell, welches einen beobachteten Poisson Prozess mit dem Chinese Restaurant Prozess verbindet wird anschließend benutzt um versteckte Zustände in der Rate von neuronalen Spike-Daten zu finden und sie mit dem Stimulus zu verbinden. Die Vorteile des Modells beim finden und bestimmen von neuronalen Bursts wird diskutiert und mit Modellen verglichen, die eine kontinuierliche Poisson Rate annehmen.

Abstract

This thesis describes a flexible architecture for a Markov chain Monte Carlo sampler which allows Bayesian posterior inference for a variety of changepoint models. The structure of this class of models consists of two stochastic processes. The first process is either observed directly or indirectly through, possibly noisy, observations. The second process is not observed and governs the parameters of the observed process. The main assumption for our class of models is that the hidden process is piecewise constant, i.e. it jumps between discrete states.

As the observed process, we discuss mainly the Ornstein-Uhlenbeck and Poisson process. The hidden process can have a fixed number of states, or an unknown number of states. The latter model is based on a hidden Chinese restaurant process and allows Bayesian inference over the number of states of the hidden parameters. The sampler applies a Metropolis-Hastings random walk on the hidden jump process through proposed birth-death moves. Different kinds of proposal moves on the path of the hidden process are presented. The structure of the sampler makes it very flexible and easy to modify to other combinations of observed and hidden processes compared to other inference methods which are tailor-made for a specific model.

Applied to gene expression data the sampler allows Bayesian posterior inference on a more complex model than in previous work. We compute the Bayes factor which indicates that our model, which allows the strength of the system noise to switch, is better in explaining the data. The sampler is used on gene expression data from yeast cells and the results are compared to a variational approximation. The posterior is more confident about the times of transcriptional activity than the approximation suggests. The results from the Chinese restaurant process sampler on the same yeast dataset support the initial assumption about the number of transcription factors involved in the control of the examined genes.

When the sampler is used on financial data, changepoints are revealed which can be connected to historic events. This is shown both for the Ornstein-Uhlenbeck model as well as a Cox-Ingersoll-Ross model used in a different thesis.

Modifying the sampler to work on Markov modulated Poisson processes allows for very fast posterior inference and this is shown when the time to get an uncorrelated sample is compared to an exact Gibbs sampler for the model. A model combining an observed Poisson process with the Chinese restaurant process is then utilized to find hidden states in the rate of neuronal spike trains and linked to the stimulus. The model's advantages in finding and estimating bursting of neurons is discussed and compared to a model which assumes a continuous Poisson rate.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

When forecasting tomorrow's weather, the simplest approach is to assume persistence: The weather tomorrow is going to be the same as today. While this method can often be very accurate, there usually comes a time when the conditions change. Finding such *changepoints* is of high interest to a myriad of different scientific fields, among them network traffic analysis (Blazek et al., 2001), climate science (Reeves et al., 2007), ecological modeling (Qian et al., 2003), finance (Preis et al., 2011) or the often cited coal-mining disaster dataset of Jarrett (1979). Therefore changepoint detection has been an active field of research for a long time with Page (1954) being an early example[1].

In contrast to many other approaches (e.g. Fearnhead and Liu, 2011; Giordani and Kohn, 2008; Wyse et al., 2011) our class of models assumes that the observations come from a stochastic process with jumping parameters, instead of assuming that the observations are i.i.d from a changing distribution. Additionally, we are not resorting to discretizing schemes and fully work in continuous time.

We are interested in a Bayesian approach to the problem and this usually makes it impossible to get analytical results for models of non-trivial complexity. Markov chain Monte Carlo (MCMC) methods have made Bayesian inference possible for models where analytical methods are unfeasible and have been applied successfully in a wide range of fields, such as physics (von Toussaint, 2011), finance (Glasserman, 2003) and biology (Manly, 2006).

This thesis presents a general MCMC sampler architecture which allows efficient Bayesian inference for a class of changepoint models consisting of two stochastic processes, one observed and one hidden. The hidden process is piecewise constant and jumps between different levels thereby controlling the parameters of the observed process. We investigate in detail models where the observed process is of the Ornstein-Uhlenbeck type and

[1]For a good overview of previous work in the field see Chen and Gupta (2011).

where it is a Poisson process. Three different types of hidden jump processes are formulated: A Markov jump process (MJP) with a fixed number of states, a MJP where the parameters of each segment are i.i.d. from a common probability distribution and a jump process whose parameters are drawn from a Chinese restaurant process, which means that the parameter jump between an unknown number of reusable states. Our sampler applies a Gibbs sampling approach to the inference task, alternating between sampling from the conditional posteriors of the parameters and the jump process. For the latter part, birth-death steps, i.e. adding, removing or shifting times of jumps, are used to modify the current path of the hidden jump process and serve as a proposal in a Metropolis-Hastings setting. This approach resembles the reversible jump algorithm of Green (1995) but in contrast to them, we formulate our model in a way that birth-death moves stay inside the model and therefore makes the approach more flexible. The overall structure of our sampler does not depend on the processes used in a specific model, making it uncomplicated to adapt it to different model combinations. We focus on gene expression and neuronal spiking data for applications of our method, as well as demonstrating its usefulness for the analysis of financial data series.

To summarize, the contributions of this thesis are as follows: 1) it describes a class of changepoint models where a stochastic process is driven by hidden jumps of its parameters, including a model where the number of states is unknown beforehand, 2) it presents a general MCMC sampler structure for Bayesian inference in these models and 3) it applies the sampler to datasets from systems biology, neurobiology and finance and compares the results to different approaches to the task.

1.1 Organization of the Thesis

A short introduction to the concepts and models used in this thesis is given in *chapter 2*. Bayesian inference, Stochastic processes and MCMC sampling are explained and the chapter concludes with a very brief description of the main applications in this thesis: transcriptional regulation and neuronal spiking.

Before diving into details about the individual model combinations we present a non-formal overview of the different model components and the general structure of the sampler in *chapter 3*. The aim of this chapter is to clarify the class of models we are looking at and to preview which processes we will use. The following chapters 4 and 5 deal with specific observed processes combined with the different hidden processes.

In *chapter 4* models where the observed process is of the Ornstein-Uhlenbeck form are discussed. The chapter is divided into sections for the different kinds of hidden processes which are combined with the Ornstein-Uhlenbeck process, starting from a simple binary telegraph process in section 4.1 and ending with the flexible Chinese restaurant process in section 4.4. The main application in this part of the thesis are gene expression datasets but in section 4.3 the sampler is applied on stock index data. Much of this chapter is based on work previously published in Stimberg et al. (2011b), Stimberg et al. (2011a) and Stimberg et al. (2012).

The observed process is switched to a Poisson process for *chapter 5*. For a hidden MJP with fixed dimensionality the model becomes the well studied Markov modulated Poisson process (MMPP). Our sampler is compared to the exact Gibbs sampler and found to improve its results under certain conditions. The Poisson process is then combined with the Chinese restaurant process to find discrete states in neuronal spiking data from the primary visual cortex. This part of the chapter is based on Stimberg et al. (2014).

Two more model combinations are investigated in *chapter 6*. The application of our sampler to a Cox-Ingersoll-Ross model with changing parameters is summarized from Herrmann (2014) and a small experiment with a multivariate Ornstein-Uhlenbeck process is presented. To conclude the main part of the thesis section 6.3 gives a brief guideline how to extend the sampler to work with other model combinations not considered in this thesis.

We summarize and discuss the results of the thesis in *chapter 7* and compare it to related work. Finally, the thesis ends by highlighting possible directions of further research.

1.2 Notation

1.2.1 Variables

$X(t)$	Observed process.
$X_{0:T}$	Path of the process from time 0 to T.
$\mathbf{D}$	Dataset of the observations.
t_i	Times of observations.
$\mu(t)$	Hidden jump process whose value is either binary or a non-negative integer.
τ_i	Times of jumps in the hidden process.
$\theta(t)$	Process of the parameters.
c	Number of jumps in the hidden process.
n	Number of data points.
N	Dimensionality of the observed process.
f,f_+,f_-	Jump rates of the hidden process.
α	Concentration parameters of the Chinese restaurant process.
A,b	Parameters of the Ornstein-Uhlenbeck process' drift.
λ	Parameter of the OU process' drift or rate of a Poisson process.
σ^2	Strength of the system noise.
σ^2_{obs}	Variance of the Gaussian observation noise.

1.2.2 Abbreviations

MC	Monte Carlo
MCMC	Markov chain Monte Carlo
MH	Metropolis-Hastings
OU	Ornstein-Uhlenbeck
CIR	Cox-Ingersoll-Ross
MJP	Markov jump process
MMPP	Markov modulated Poisson process
ODE	Ordinary differential equation
SDE	Stochastic differential equation
CP	Changepoint process
CRP	Chinese restaurant process

1.2.3 Probability Distributions

$$
\begin{aligned}
\mathcal{N}(x;\mu,\sigma^2) &= \frac{1}{\sqrt{2\sigma^2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \\
\mathrm{Exp}(x;\lambda) &= \lambda \exp(-\lambda x) \\
\mathrm{Gamma}(x;a,b) &= \frac{1}{\Gamma(a)b^a} x^{a-1} \exp\left(-\frac{x}{b}\right) \\
\mathrm{Poisson}(k;\lambda) &= \frac{\lambda^k}{k!} \exp(-\lambda)
\end{aligned}
$$

Chapter 2

Background

This chapter gives an introduction to the methods and applications used in this thesis. It is intended to give readers not familiar with the topics a short overview to understand the later chapters. For in-depth information about the subjects the reader is referred to the references given in the relevant sections.

2.1 Methods

As this thesis deals with Bayesian inference for stochastic processes using Markov chain Monte Carlo methods, these three topics are explained first.

2.1.1 Bayesian Inference

While a detailed and thorough discussion on Bayesian inference is far beyond the scope of this thesis, a very short introduction about the approach is given for readers not familiar with it. The Bayesian approach differs from the so-called frequentist one not in the mathematical foundation of probability theory, but in the way probabilities are interpreted. While in a frequentist view a probability is defined as a "limiting frequency in independent repetitions of a random experiment" (Jaynes and Bretthorst, 2003, p. 270), Bayesians use probabilities to describe "degrees of belief" (Barber, 2012, p. 5) and the mathematical laws of probability theory allow us to calculate how these beliefs should behave. Or as Bernardo and Smith (2009, p. 4) write: "Bayesian Statistics offers a rationalist theory of personalistic beliefs in contexts of uncertainty, [...]".

From a Bayesian perspective inference starts with the prior belief about the subject before any data is observed. We have a belief about what model M describes the process which we want to observe and given the model we have a belief about the parameters θ in that model.

These beliefs are represented by probability distributions $P(M)$ and $P(\theta|M)$, respectively. If we are very sure a model or parameter value is correct the distributions would be very narrow, if we are unsure the distributions would be flat, i.e. have a high variance.

The "Bayesian" aspect comes from Bayes rule[1], which can be used to describe how a prior belief will change when we observe data D:

$$P(\theta|D,M) = \frac{P(D|\theta,M)P(\theta|M)}{P(D|M)}. \tag{2.1}$$

$P(\theta|M)$ is called the prior (belief about the parameters), $P(D|\theta,M)$ the likelihood (of the data given the parameters) and $P(\theta|D,M)$ the posterior (belief about the parameters). The denominator $P(D|M)$ in (2.1) is called the evidence or marginal likelihood and can be rewritten as

$$P(D|M) = \int P(D|\theta,M)P(\theta|M)d\theta \tag{2.2}$$

by marginalizing out the parameters (Kruschke, 2011, p. 58). Usually the evidence only interests us if we want to do model comparison, at least in the context of this thesis because it gets canceled out of all our calculations otherwise[2].

If we are not comparing different models but are interested in parameter inference for a specific one, we usually suppress M for the sake of clarity. In these cases what interests us is the posterior over the parameters $P(\theta|D)$ which describes how our belief has been updated by observing the data. This means instead of a point estimate of the most likely set of parameter values which generated the data, we get a probability distribution. We can still look at the point which maximizes it (the maximum-a-posteriori or MAP estimate) but having a full distribution gives us an estimate of how sure we should be that the parameter values were in a specified region. Because of this the results in this thesis are usually given either as plots of the posterior distribution over parameters or the distribution is described by expected values over it such as the mean and variance.

Bayesian Model Comparison

Founded on the work of Jeffreys (1935) the Bayesian framework allows us to compare how well different models explain a dataset. One advantage over classical hypothesis testing is that Bayesian model comparison automatically prevents overfitting by penalizing too complicated models (Kass and Raftery, 1995). The central unit in this framework is the

[1]Named after Thomas Bayes who most likely lived from 1702 to 1761 according to Dale (1991) and whose work on the theorem was posthumously published in Bayes and Price (1763).

[2]See section 2.1.3 for an explanation.

	$BF_{1,2}$	< 1	Evidence against M_1.
$1 <$	$BF_{1,2}$	$< 10^{\frac{1}{2}}$	Evidence supports M_1 but not worth more than a bare mention.
$10^{\frac{1}{2}} <$	$BF_{1,2}$	< 10	Evidence supporting M_1 substantial.
$10 <$	$BF_{1,2}$	$< 10^{\frac{3}{2}}$	Evidence supporting M_1 strong.
$10^{\frac{3}{2}} <$	$BF_{1,2}$	$< 10^2$	Evidence supporting M_1 very strong.
$10^2 <$	$BF_{1,2}$		Evidence supporting M_1 decisive.

Table 2.1 Interpretation for the value of the Bayes factor defined by (2.3).

Bayes factor:

$$BF_{1,2} = \frac{P(D|M_1)}{P(D|M_2)}, \tag{2.3}$$

which compares the likelihood of the data being generated by model M_1 and M_2. As we know from (2.2) the marginal likelihoods are computed by marginalizing over all possible parameter values and are therefore independent of the parameters. If we have prior belief about how likely the models are, these can be incorporated to get the posterior odds of the models

$$PO_{1,2} = \frac{P(D|M_1)}{P(D|M_2)}\frac{P(M_1)}{P(M_2)}. \tag{2.4}$$

As we indicated above, the marginal likelihoods normally don't need to be computed when posterior inference is done and, as Calderhead and Girolami (2009) highlighted, some methods to compute this quantity can be treacherous. If $BF_{1,2}$ is larger than 1 then the data favors the model M_1. Furthermore (Jeffreys, 1998, p. 432) provided a guideline to interpret the results which can be seen in table 2.1.

2.1.2 Stochastic Processes

Stochastic processes can be seen as a generalization of vectors of random variables. Instead of a finite number of random variables $\mathbf{X} = (x_1, \ldots, x_n)$ a random process is a collection of random variables $X(t)$ with $t \in T$ and $T \subseteq \mathbb{R}$ (Stirzaker, 2005, pp. 45 ff.) [3].

Usually t will be the time which makes the stochastic process describe the evolution of a random variable (or a vector of random variables) over time. A specific value of $X(t)$ is called the state of the process at time t and the space of possible values that can be assigned to $X(t)$ (e.g. $\mathbb{R}^n$ for a n-dimensional continuous state process) is referred to as the state space.

[3] Sometimes this definition is broadened to include random fields where the "time" can be a multidimensional vector in $\mathbb{R}$.

A realization of a stochastic process is called a (sample) path. Both the time and the state can be discrete or continuous.

Markov Processes

An important subclass of stochastic processes are the Markov processes. Almost all the processes in this thesis are Markov processes and this holds true in many other applications as well (Stirzaker, 2005; Van Kampen, 2011). A Markov process is a process which satisfies the Markov property i.e. for any set of times $t_1, \dots, t_k$ with $t_1 < \cdots < t_k$ the process fulfills

$$P(t_k, X(t_k) | t_{k-1}, X(t_{k-1}); \dots; t_1, X(t_1)) = P(t_k, X(t_k) | t_{k-1}, X(t_{k-1})) \tag{2.5}$$

(Van Kampen, 2011, p. 73).

Informally this means that given a history of exact observations, the future evolution of a process only depends on the last observation. Markov processes are uniquely defined through $P(t_1, X(t_1))$ and the transition probability $P(t_k, X(t_k) | t_{k-1}, X(t_{k-1}))$ (Honerkamp, 1994) and this allows us to rewrite the joint density over observations $X(t_1)$, $X(t_2)$, ..., $X(t_{n-1})$, $X(t_n)$ as

$$\begin{aligned} &P(t_{,1}, X(t_1); t_2, X(t_2); \dots; t_{n-1}, X(t_{n-1}); t_n, X(t_n)) \\ =&P(t_1, X(t_1)) P(t_2, X(t_2) | t_1, X(t_1)) \dots P(t_n, X(t_n) | t_{n-1}, X(t_{n-1})) \end{aligned} \tag{2.6}$$

An important property of all Markov processes is that the transition density fulfills the Chapman-Kolmogorov equation

$$P(t_k, X(t_k) | t_{k-1}, X(t_{k-1})) = \int_{-\infty}^{\infty} P(t_j, X(t_j) | t_{k-1}, P(X(t_{k-1})) P(t_k, X(t_k) | t_j, P(X(t_j)) dX(t_j), \tag{2.7}$$

which means that the transition probability from one state to another can be written as a product of two transition densities with the unknown state in the middle marginalized out (Kloeden and Platen, 1992, p. 35).

Diffusion processes

Diffusion processes are special cases of Markov processes with a continuous state space. The simplest diffusion process and the basis for all the others is the Wiener process (or Brownian motion). A Wiener process $W(t)$ is a continuous stochastic process with the property that all its increments are independent and normally distributed with zero mean and variance proportional to the time difference, i.e.

$$W(t+s) - W(s) \sim \mathcal{N}(0, \sigma^2 t), \tag{2.8}$$

for some $0 < \sigma^2 < \infty$ (Stirzaker, 2005, pp. 219 ff.).

A diffusion process $X(t)$ can be defined by stochastic differential equations (SDEs) of the form

$$dX = a(t,X)dt + (b(t,X))^{1/2}dW, \tag{2.9}$$

where dW is the increment of the standard Wiener process, $a(t,X)$ is called the drift function and $b(t,X)$ the diffusion function.

Depending on the drift and diffusion function a closed form solution for $X(t)$ might be obtainable (Kloeden and Platen, 1992, p. 104-105). If this is not the case a diffusion process can always be approximately simulated using the Euler-Maruyama approximation, which is a generalization of Euler's method for ordinary differential equations. It discretizes the time and iteratively draws the next values of the process from

$$X(t_k) \sim \mathcal{N}\left(X(t_{k-1}) + a(t_{k-1}, X(t_{k-1}))\Delta t, b(t_{k-1}, X(t_{k-1}))\Delta t\right), \tag{2.10}$$

where $\Delta t = t_k - t_{k-1}$ is the time step and the approximation becomes exact for $\Delta t \to 0$ (Kloeden and Platen, 1992, pp. 305 ff.).

The time evolution of the transition distribution $P(X(t) = y|X(s) = x) = P_{t,y,s,x}$ of a diffusion process can be described by the Fokker-Planck or Kolmogorov forward equation

$$\frac{\partial P_{t,y,s,x}}{\partial t} = -\frac{\partial}{\partial y}[a(t,y)P_{t,y,s,x}] + \frac{1}{2}\frac{\partial^2}{\partial y^2}[b(t,y)P_{t,y,s,x}], \tag{2.11}$$

this can also be formulated backward in time

$$\frac{\partial P_{t,y,s,x}}{\partial s} = -a(s,x)\frac{\partial P_{t,y,s,x}}{\partial x} - \frac{1}{2}b(s,x)\frac{\partial^2 P_{t,y,s,x}}{\partial x^2}, \tag{2.12}$$

which is the Kolmogorov backward equation (Kloeden and Platen, 1992, p. 37).

Ornstein-Uhlenbeck Process

The Ornstein-Uhlenbeck (OU) Process is a mean reverting Markov process with a linear drift and constant diffusion function and was first introduced by Uhlenbeck and Ornstein (1930) as a model for the dynamics of gas molecules. Besides their initial application OU processes haven been used in many fields such as neurobiology (Ricciardi and Sacerdote, 1979), finance (Marsh and Rosenfeld, 1983) or genetics (Dunlop et al., 2008).

In its one dimensional form the OU process is defined by a stochastic differential equation of the form

$$dX = (b - \lambda X)dt + \sigma dW, \tag{2.13}$$

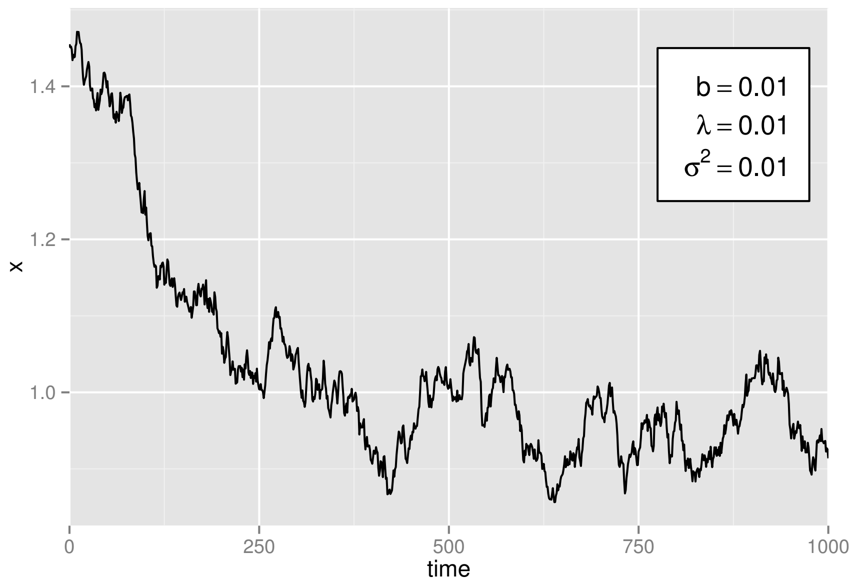


Fig. 2.1 Example of the path drawn from an Ornstein-Uhlenbeck process. The mean is at $b/\lambda = 1$.

with parameters b, $\lambda > 0$ and $\sigma > 0$ and where dW models a Wiener process. In this parametrization the mean of the process will converge to b/λ, λ is the rate of convergence and σ the strength of the noise[4].

The SDE (2.13) is solved by

$$X(t_k) = X(t_{k-1})\exp(-\lambda t) + \frac{b}{\lambda}(1-\exp(-\lambda t)) + \sigma\int_{t_{k-1}}^{t_k} \exp(-\lambda(t_k - s))dW_s, \tag{2.14}$$

and because this is an integral of a deterministic function with respect to a Wiener process we know this is a Gaussian process with transition density

$$P(t_k, X(t_k)|t_{k-1}, X(t_{k-1})) = \mathcal{N}(X(t_k); m(\Delta t, X(t_{k-1})), v(\Delta, X(t_{k-1}))), \tag{2.15}$$

with mean and variance

$$m(\Delta t, X(t_{k-1})) = X(t_{k-1})\exp(-\lambda\Delta t) + \frac{b}{\lambda}(1-\exp(-\lambda\Delta t)) \tag{2.16}$$

$$v(\Delta t, X(t_{k-1})) = \frac{\sigma^2}{2\lambda}(1-\exp(-2\lambda\Delta t)) \tag{2.17}$$

[4]Often a different parametrization is used: $dx = a(b-x)dt + \sigma dW$, where the mean of the process converges to b.

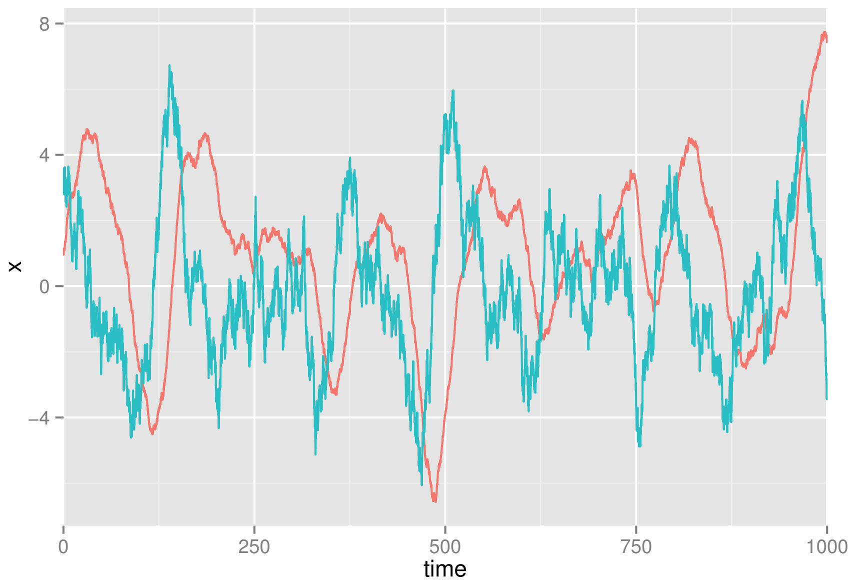


Fig. 2.2 Example of the path drawn from an multivariate Ornstein-Uhlenbeck process. The first dimension is in red, the second in blue. The parameters used were $\mathbf{B} = \begin{bmatrix} 0.01 \\ 0.05 \end{bmatrix}$, $\Lambda = \begin{bmatrix} 0 & -0.05 \\ 0.04 & 0.02 \end{bmatrix}$ and $\Sigma = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.5 \end{bmatrix}$.

and where $\Delta t = t_k - t_{k-1}$ (Steele, 2001, pp. 138 ff.)[5].

The Ornstein-Uhlenbeck process can be extended to multiple dimensions. If we assume the dimensions are independent then it is just a set of stochastic differential equations like (2.13) with (possibly) different parameters for each dimension. But it is also possible to define a multidimensional Ornstein-Uhlenbeck process where the different dimensions are coupled. The stochastic differential equation then becomes

$$d\mathbf{X} = (\mathbf{B} - \Lambda\mathbf{X})dt + \Sigma d\mathbf{W}, \tag{2.18}$$

where $\mathbf{X}$ and $\mathbf{B}$ are m-dimensional row vectors while Λ and Σ are $m \times m$ matrices. The transition density then becomes

$$P(t_k, \mathbf{X}(t_k) | t_{k-1}, \mathbf{X}(t_{k-1})) = \mathcal{N}(\mathbf{X}(t_k); \mathbf{M}(\Delta t, \mathbf{X}(t_{k-1})), \mathbf{V}(\Delta t, \mathbf{X}(t_{k-1}))), \tag{2.19}$$

with mean

$$\mathbf{M}(t, \mathbf{X}(t_{k-1})) = \exp(-\Delta t\Lambda)\mathbf{X}(t_{k-1}) + (\mathbf{I}_{m\times m} - \exp(-\Delta t\Lambda))\Lambda^{-1}\mathbf{B}, \tag{2.20}$$

[5] We generalized the solution because Steele (2001) use an SDE with $b = 0$.

where $\mathbf{I}_{m\times m}$ is the $m \times m$ identity matrix, $\exp(\cdot)$ is the matrix exponential.

The variance is

$$\mathbf{V}(t,\mathbf{X}(t_{k-1})) = \mathbf{U}\mathbf{D}^{-1} \tag{2.21}$$

$$\mathbf{U} = \mathbf{R}(1:m,1:m) \tag{2.22}$$

$$\mathbf{D} = \mathbf{R}(m+1:2*m,1:m) \tag{2.23}$$

$$\mathbf{R} = \exp(\Delta t\Psi)\begin{bmatrix} \mathbf{0}_{m\times m} \\ \mathbf{I}_{m\times m} \end{bmatrix} \tag{2.24}$$

$$\Psi = \begin{bmatrix} -\Lambda & \Sigma\Sigma' \\ \mathbf{0}_{m\times m} & \Lambda' \end{bmatrix}, \tag{2.25}$$

with and $\mathbf{R}(r_b : r_e, c_b : c_e)$ being a sub-matrix of $\mathbf{R}$ ranging from row r_b to r_e and column c_b to c_e.

For a derivation of this see section A.5 of the appendix.

Cox-Ingersoll-Ross Process

The Cox-Ingersoll-Ross (CIR) model was introduced by Cox et al. (1985) to describe the time evolution of interest rates. It is similar to the Ornstein-Uhlenbeck process with the difference, that the diffusion depends on the value of the process in such a way that the process can never become negative for positive values of b and λ.

$$dX = (b-\lambda X)dt + \sigma\sqrt{X}dW, \tag{2.26}$$

The transition density $P(t_k, X(t_k)|t_{k-1}, X(t_{k-1})$ of the CIR process has mean and variance

$$m(\Delta t, X(t_{k-1})) = X(t_k)\exp -\lambda\Delta t + \frac{B}{\lambda}(1-\exp(-\lambda\Delta t)) \tag{2.27}$$

$$v(\Delta t, X(t_{k-1})) = X(t_k)\frac{\sigma^2}{\lambda}(\exp(-\lambda\Delta t) - \exp(-2\lambda\Delta t)) + \frac{B\sigma^2}{2\lambda}(1-\exp(-\lambda\Delta t))^2. \tag{2.28}$$

While, like an OU process, the CIR process is driven by a Wiener process it is not a Gaussian process. The transition density has the form

$$P(t_k, X(t_k)|t_{k-1}, X(t_{k-1})) = c\exp(-u - cX(t_k))\left(\frac{cX(t_k)}{u}\right)^{q/2} I_q(2\sqrt{ucX(t_k)}), \tag{2.29}$$

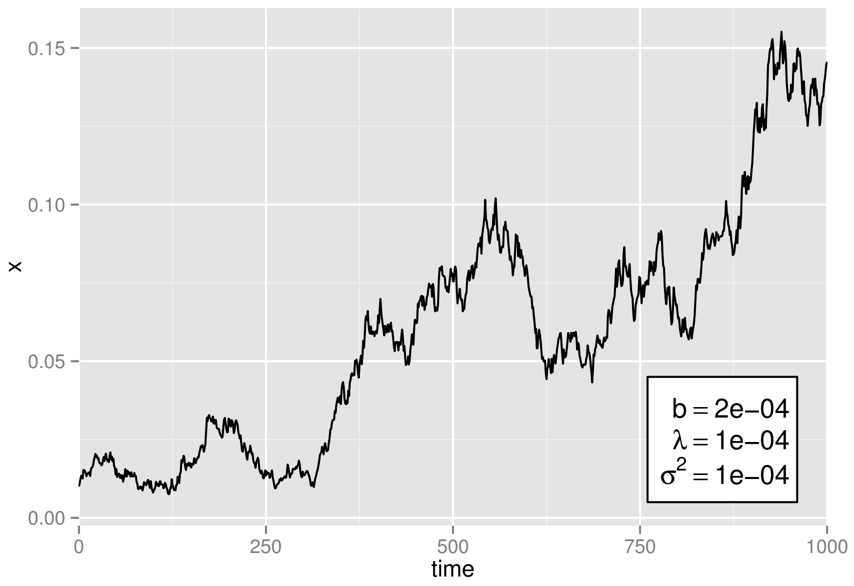


Fig. 2.3 Example of the path drawn from an Cox-Ingersoll-Ross process.

where

$$c = \frac{2\lambda}{\sigma^2(1 - \exp(-\lambda\Delta t))} \tag{2.30}$$

$$u = cX(t_{k-1})\exp(-\lambda\Delta t) \tag{2.31}$$

$$q = \frac{2b}{\sigma^2} - 1, \tag{2.32}$$

and with $I_q(\cdot)$ being the modified Bessel function of the first kind of order q (Cox et al., 1985). This is equivalent to a noncentral chi-square distribution with $2q+2$ degrees of freedom and noncentrality parameter $2u$:

$$P(X(t_k)|X(t_{k-1})) = \chi^2\left(2cX(t_k); 2q+2, 2uX(t_{k-1})\right). \tag{2.33}$$

Markov Jump Processes

A jump process is defined over a finite or countable set of states, therefore the main difference between diffusion and jump processes is that the former have continuous and the latter discrete states. The process starts in some state x_0 and stays in it until after some positive time τ_1 it jumps to a new state x_1 and stays in it until some positive time $\tau_2 > \tau_1$, then it jumps to state x_2 and so on. This means the process is piecewise constant and its state can be defined by (Hoel et al., 1972, p. 84)

$$X(t) = \begin{cases} x_0, & 0 \geq t > \tau_1 \\ x_1, & \tau_1 \geq t > \tau_2 \\ x_2, & \tau_2 \geq t > \tau_3 \\ \vdots \end{cases} . \tag{2.34}$$

If a process is non-explosive, i.e. if

$$\lim_{n \to \infty} \tau_n = \infty, \tag{2.35}$$

then $X(t)$ is defined for all $t \geq 0$ (Hoel et al., 1972, p. 86).

A Markov jump process (MJP) is a jump process which fulfills the Markov property. This means that the distribution over the time to the next jump, may depend at most on the last state of the process. When a jump happens the process chooses a new state according to its transition distribution $P(x_i|x_{i-1})$. As for diffusion processes we can formulate the Chapman-Kolmogorov equation for Markov jump processes as well:

$$P(t_k, X(t_k)|t_{k-1}, X(t_{k-1})) = \sum_{X(t_j)} P(t_j, X(t_j)|t_{k-1}, P(X(t_{k-1}))P(t_k, X(t_k)|t_j, P(X(t_j)), \tag{2.36}$$

with $0 \leq t_{k-1} \leq t_j \leq t_k$.

A special case of the Markov jump process is the telegraph process. It only has 2 states and switches between them at each jump.

Poisson Process

The Poisson process is a counting process, i.e. a jump process whose state starts at 0 and grows by 1 at each jump. For a Poisson process the waiting times between jumps are exponentially distributed with rate parameter λ and this leads to the state $X(t)$ being Poisson distributed with parameter λt (Hoel et al., 1972, p. 95). A generalization of a Poisson process where the time between jumps is not necessarily exponentially distributed is called a renewal process. In some applications the state values are ignored because we are only interested in the times of the jumps.

In order to simulate the path of a Poisson process it is only necessary to draw the time until the next jump from an exponential distribution until the time is past the end time T.

Poisson processes have a very broad range of applications from modeling the scoring of goals in soccer (Heuer et al., 2010) to network access data (Balachandran et al., 2002).

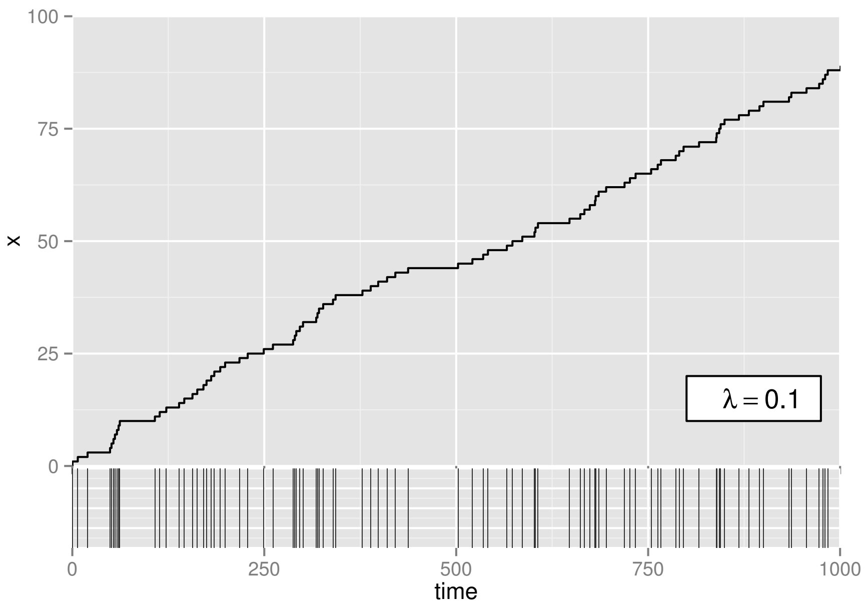


Fig. 2.4 Example of the path drawn from a Poisson process. The top shows the path as a counting process while in the bottom each event is drawn as a vertical line.

2.1.3 Markov Chain Monte Carlo Sampling

Many problems in areas such as statistical physics, Bayesian inference or computational biology, involve having to solve multi-dimensional integrals, which most of the time are analytically intractable (Kalos and Whitlock, 2008; von Toussaint, 2011; Wakefield, 2007). There are a number of numerical approximations for one-dimensional integrals which can be extended to the multi-dimensional settings but they suffer from the so called *curse of dimensionality* which leads to the computational costs growing exponentially with the dimensionality (see e.g. Gamerman and Lopes, 2006; Liu, 2008).

The Monte Carlo Method of Metropolis and Ulam (1949) overcomes this problem because the error rate of Monte Carlo methods is independent of the dimensionality and shrinks proportional to the square root of the number of samples (Kalos and Whitlock, 2008, pp. 77-79).

The Monte Carlo method is based on the idea that an integral over a probability distribution $P(x)$

$$E\left(f(x)\right) = \int P(x) f(x) dx \tag{2.37}$$

can be approximated by

$$E(f(x)) \approx \frac{1}{m}\sum_{i=1}^{m} f(x_i) \tag{2.38}$$

where $X = (x_1, \ldots, x_m)$ is a sequence of random numbers distributed according to $P(x)$.

If $P(x)$ is a simple distribution, e.g. Gaussian or gamma, then there are direct methods to get independent identical distributed (i.i.d.) samples from it (Gamerman and Lopes, 2006, pp. 12-13) but for more complicated cases i.d.d samples are seldom obtainable directly.

Rejection and Importance Sampling

One method to obtain i.i.d. samples from a distribution $P(x)$ when direct sampling is not possible, is called *rejection sampling*. Rejection sampling needs a so called proposal distribution $Q(x)$ for which i.i.d. samples can be generated. This proposal distribution must fulfill the property that there exists a finite constant $c > 1$ for which

$$cQ(x) \geq P(x), \forall x. \tag{2.39}$$

For each sample generated from the proposal distribution we draw a random number u uniformly distributed between 0 and 1 and accept the sample if

$$u < P(x)/cQ(x) \tag{2.40}$$

is true (Liu, 2008, p. 24). The average acceptance rate of rejection sampling is $1/c$ and this means that we might be drawing a large number of proposals from $Q(x)$ to get one sample from $P(x)$ if c is large.

Importance sampling (Liu, 2008, pp. 31 ff.) on the other hand doesn't reject samples but draws them from $Q(x)$ and assigns a weight $w = P(x)/Q(x)$ to them, basically weighting samples in regions where $Q(x)$ underestimates $P(x)$ higher and vice versa. If $Q(x)$ isn't chosen carefully the estimator can be dominated by few samples with large weights and in the worst case it can have a small empirical variance while still being far from the true value (Bishop, 2007, p. 534).

Markov Chain Monte Carlo Sampling

While both rejection and importance sampling have been shown to work well for many applications they tend to loose efficiency fast when the dimensionality of the target distribution grows (MacKay, 2002, p. 363-365). For problems such as this, a special class of Monte

Carlo algorithms, called Markov chain Monte Carlo (MCMC) methods, tends to be more effective. They are based on creating a Markov chain of random samples whose stationary (or invariant) distribution is the target distribution $P(x)$. The first MCMC method was introduced by Metropolis et al. (1953) for problems in statistical physics and henceforward called the Metropolis algorithm (Kendall et al., 2005, p. ix). This approach is based on drawing a sample x^* from a proposal distribution $Q(x^*|x_i)$ depending on the last sample x_i. Metropolis et al. (1953) required that Q has to be symmetric, i.e. it must fulfill $Q(x^*|x_i) = Q(x_i|x^*)$.

The next sample in the Markov chain then becomes

$$x_{i+1} = \begin{cases} x_* & \text{if } u < \min\left(\frac{P(x^*)}{P(x_i)}, 1\right) \\ x_i & \text{otherwise,} \end{cases} \tag{2.41}$$

where $u \sim U(0,1)$ is a uniformly distributed random number[6].

This means that a proposal is drawn from $Q(x^*|x_i)$ and accepted with probability $\min(P(x^*)/P(x_i), 1)$ but in contrast to rejection sampling a rejection means that the last sample is reused instead of drawing samples until one is accepted.

A very important advantage of the Metropolis algorithm is that only ratios of the target distribution $P(x)$ have to be computed and therefore any normalization factors can be ignored.

Hastings (1970) generalized the Metropolis algorithm by no longer requiring Q to be symmetric, leading to the next sample being

$$x_{i+1} = \begin{cases} x_* & \text{if } u < \min\left(\frac{Q(x_i|x^*)}{Q(x^*|x_i)} \frac{P(x^*)}{P(x_i)}), 1\right) \\ x_i & \text{otherwise.} \end{cases} \tag{2.42}$$

The algorithm is called the Metropolis-Hastings algorithm and because of its very wide applicability has been called one of the most important algorithms for science and engineering in the 20th century (Dongarra and Sullivan, 2000).

The proof that $P(x)$ is in fact the stationary distribution of the Markov chain has two steps: First it must be shown that a stationary distribution exists and secondly it must be shown that there exists only one stationary distribution, i.e. that it is unique (Bishop, 2007, p. 540). The existence of a stationary distribution can be shown using a property called detailed balance

[6]The minimization is not necessary if the method is implemented, instead it only needs to be checked if u is smaller than $P(x^*)/P(x_i)$.

(Rubinstein and Kroese, 2008, p. 168), which means that the chain's transition probabilities $T(x_{i+1}|x_i)$ satisfy

$$P(x_i)T(x_{i+1}|x_i) = P(x_{i+1})T(x_i|x_{i+1}), \tag{2.43}$$

with respect to the distribution $P(x)$ (Bishop, 2007, p. 540).

A Markov chain which fulfills detailed balance is said to be reversible and while the Metropolis-Hastings algorithm fulfills detailed balance with respect to the target distribution this is not a necessary condition to ensure the existence of a stationary distribution (MacKay, 2002, p. 374) and some approaches have used non-reversible chains to reduce the random walk behavior of the sampler (see e.g. Fernandes and Weigel, 2011).

The uniqueness of the stationary distribution can be guaranteed by showing that the Markov chain is ergodic, i.e. that

$$P_t(x) \to P(x) \text{ as } t \to \infty, \text{ for any } P_0(x), \tag{2.44}$$

where $P(x)$ is the stationary distribution and $P_t(x)$ is the distribution over the state of the Markov chain at time t. (MacKay, 2002, p. 373) lists two possible cases how ergodicity might not be respected by the Markov chain: Either certain areas of the probability space are not reachable from all starting positions, i.e. there exist two or more subspaces that are not reachable from each other, or that there are starting positions which lead to periodic limit-cycles. When choosing the proposal distribution one has to make sure that both these cases do not occur. When $P(x)$ is the Markov chain's stationary distribution and ergodicity is satisfied $P(x)$ is called the equilibrium distribution of the Markov chain (Bishop, 2007, p. 540).

Gibbs Sampler

A special case of the MH algorithm, called Gibbs Sampler, was introduced by Geman and Geman (1984). A Gibbs sampler allows to draw samples from complicated joint distributions $P(\mathbf{X}) = P(X_1,\ldots,X_n)$ by drawing one dimension from the conditional distribution $P(X_j|X_{-j})$ and holding the rest fixed. This is then repeated for all dimensions, either deterministically or in random order (Rubinstein and Kroese, 2008, p. 177).

In order to prove that the Gibbs sampler samples from the desired distribution, it first has to be shown that the target distribution is a stationary distribution of the Markov chain. In each step of the Gibbs sampler the marginal distribution over all the dimensions which remain fixed ($P(X_{-j})$) is clearly invariant. The transition distribution in each step is given by the conditional distribution $P(X_j|X_{-j})$ and combined with $P(X_{-j})$ this gives the joint distribution $P(\mathbf{X})$ and therefore leaves it invariant (Bishop, 2007, p. 544).

Ergodicity is satisfied when the conditional distributions are positive over all possible values of X_j but this is not a necessary condition.

If the next dimension to sample from is drawn randomly the Gibbs sampler can be interpreted as a MH algorithm. The proposal first draws the dimension and then samples from its conditional distribution. If we insert this into the acceptance probability (2.42) the acceptance will always be 1 (Bishop, 2007, p. 544).

The Gibbs sampler described so far only samples one dimension in each step. This is often called a single-site Gibbs sampler. Jensen and Kong (1995) introduced the blocking-Gibbs sampler which separates the dimensions into, possibly overlapping, sets. One sampling step takes one of these sets and samples all the dimensions in it conditioned on the current value of all the other dimensions. In one full pass of the sampler, each set has been used at least once and because the union of all the sets has to contain all dimensions each dimension has been resampled. Obviously, the advantage of this approach over a single-site Gibbs sampler is that the samples will be less correlated because more than one dimension changes in each step. The downside is that it is usually easier to sample from a one-dimensional conditional density than sampling multiple dimensions at once.

Often (as in this work) Metropolis-Hastings and Gibbs samplers are combined, so that in each step only a subset of the variables is sampled using a Metropolis-Hastings step instead of directly from the conditional distribution. This is sometimes called a Metropolis-within-Gibbs sampler but the validity of that name is disputed (Brooks et al., 2011, pp. 105-106).

Convergence and Correlation

While the MH and Gibbs sampler algorithms guarantee that with a suitable proposal distribution the Markov chain's stationary distribution is the target distribution, they may need some time to converge from the (possibly random) starting position. Because of this, using all the samples to compute expectations could lead to the arbitrary first sample to have a, possibly large, effect on the results, which is not desired. In order to avoid this effect it is advisable to drop a certain number of samples in the beginning (called the burn-in) and not use them for any calculations.

There are a multitude of methods to analyze if the chain has converged, but Cowles and Carlin (1996) showed that all of them still can fail and there is no guarantee that the chain really has converged. The recommendation of Cowles and Carlin (1996) is to not rely on automated methods or a single measure to assure convergence. In this work we manually decided on the size of burn-in by inspecting the trace plots of the parameters and let the sampler run with multiple random starting values for a short time before the main sampling run to see that they all lead to the same region of high probability. While this method may circumvent some problems, it should be clear that, as Geyer (1992) writes, it is always possible to construct an example where this method fails.

One problem for many of the methods to assess convergence is that they are ill-equipped to deal with samples which change in dimensionality or are paths of variables over time, as is the case in this work. Many of the methods are only meant for univariate samples (Cowles and Carlin, 1996).

Besides the problem of convergence it should be clear that MCMC methods don't generate i.i.d. samples because a sample always depends on its predecessor[7]. If the samples are highly correlated the sampler is said to have a slow mixing rate. The choice of the proposal distribution is especially important here. Normally one would assume that a high acceptance ratio should be desirable but there is a trade-off to consider: The acceptance probability usually becomes very high when the proposal only makes small steps in the state space leading to many different but highly correlated samples. If very large steps are made the accepted samples are less correlated but more samples are rejected, leading to many identical samples and an overall high sample correlation (Bishop, 2007, pp. 541-542). Roberts et al. (1997) e.g. concluded that for a multidimensional Gaussian proposal density with a diagonal covariance matrix the asymptotically optimal acceptance probability is ≈ 0.234.

In this work, in order to asses the quality of the mixing, we compute the integrated autocorrelation time

$$\tau_{iat} = 1 + 2\sum_{i=1}^{\infty} Corr_i[\mathbf{X}], \tag{2.45}$$

with $Corr_i[X]$ being the (normalized) autocorrelation of the samples at lag i:

$$Corr_i[\mathbf{X}] = \sum_{j=1}^{\infty} \frac{Cov[X_j, X_{j+i}]}{Var[\mathbf{X}]}. \tag{2.46}$$

When we divide the total number of samples by the integrated autocorrelation time, sometimes also referred to as the inefficiency factor, we get the effective number of samples (Berg and Billoire, 2008). One approach to obtain i.i.d samples would be to run multiple randomly initialized chains in parallel until convergence and then only use one sample per chain (Tierney, 1994). Obviously this will give very few samples compared to the invested computational costs. A less extreme approach is to thin the data by using only every n-th sample, where n is larger than the integrated autocorrelation time. While this might be

[7]Even if the proposal doesn't!

desirable for some applications in most cases thinning is not necessary (Geyer, 1992) and even counterproductive if we want to get as much information as possible (Link and Eaton, 2012). Even if n highly correlated samples don't carry much more information than each of them on their own, they can't contain less. It is just important to remember that the effective information in the samples is smaller than their number suggests. In this work we sometimes use thinning in order to avoid memory and computation time problems because the dimensionality and number of samples can become very large. This shouldn't be understood as claiming that the thinned samples can be seen as i.i.d samples from the posterior.

2.2 Applications

The model described in this thesis is very flexible and allows for different types of stochastic processes to be put in with minimal modification necessary. This means that it can be applied to a large variety of applications where we have time course data and want to know when the parameters of the model switch and what states the system has. In this section the two main applications demonstrated in this work are briefly explained.

2.2.1 Transcriptional Regulation

Genes are sequences of bases in the DNA or RNA. Crick et al. (1961) discovered that genes start and end with a specific triple of bases, called a start and end codon, respectively. In between these every triple of bases codes for an amino acid and thus genes can be seen as the codes for proteins. The process of generating an equivalent mRNA copy of a gene in a DNA sequence is called transcription and it is performed by the enzyme RNA polymerase (Kleinsmith and Kish, 1995, p. 81). The mRNA copy is then used by a ribosome for synthesizing the coded protein, this process is called translation (Alberts, 1989, p. 104). Transcription factors are proteins which have a DNA binding domain which allows them to bind to specific DNA sequences next to genes, including the promoter region, which is needed for the transcription process to start. By binding to these parts of the DNA the transcription factors can up- or down-regulate the transcription rate of the corresponding gene (Latchman, 1997). Transcription factors can only bind to specific binding sites and this allows the gene expression to be controlled in reaction to outside stimuli, e.g. temperature, by producing specific transcription factors, which then change the transcription rate of genes to respond to the changed surroundings (Sorger, 1991). Usually, multiple transcription factors are involved in determining the expression level of a gene allowing for a highly complex regulation (Alberts, 1989, pp. 554 ff.).

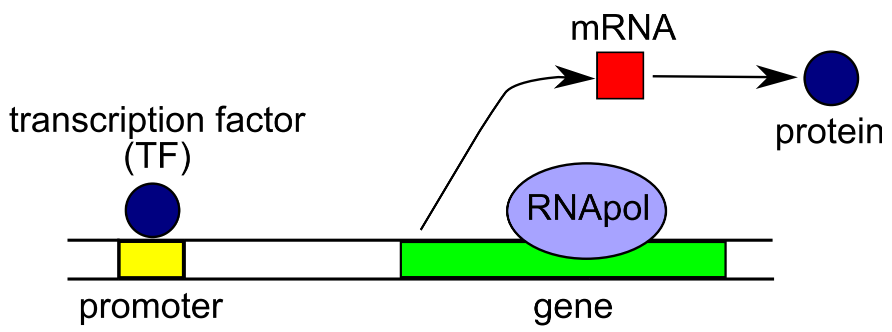


Fig. 2.5 Schematic of the process of transcriptional regulation based on figure 1.1 in Ocone (2013).

A simplified version of the process of transcriptional regulation is shown in figure 2.5.

2.2.2 Neuronal Spiking

Neuronal spikes, often called action potentials in the neurobiology literature, are all-or-none electrical signals triggered in the origin of a neuron's axon (Kandel et al., 2000, p. 21). Spikes are sudden changes in the membrane potential of a neuron and are initialized when the membrane potential surpasses a certain threshold. For the giant squid's axon (from which Hodgkin and Huxley (1939) made the first published intracellular recording) this threshold is 15mV above the membrane's resting potential (Dowling, 1992, p. 80), which is usually between -60mv and -70mv (Kandel et al., 2000, p. 126). When the membrane potential is below this threshold, the cell can be described by a passive electrical circuit but when it is surpassed voltage-gated ion channels open. This allows Na+ ions to rapidly move into the cell and starts a cascading effect because the influx of Na+ ions raises the membrane potential further, opening more ion channels which speeds up the influx of Na+ (Kandel et al., 2000, pp. 150 ff.). After about a millisecond of rising membrane potential the Na+ channels start to inactivate and K+ channels open to let K+ ions move out of the cell and lower the membrane potential. This process leads to the membrane potential being lowered past the resting potential and the period it takes for the membrane to reach the resting potential again is called the relative refractory period. While during the absolute refractory period, directly after the action potential, stimulation of the neuron will never lead to another spike, the relative refractory period needs a higher stimulation to start the process again (Kandel et al., 2000, p. 157). Figure 2.6 shows how the membrane potential changes during a neuronal spike.

As all-or-none signals, the information a spike transfers is not in its form but in the pathway it is traveling along and in its time (Kandel et al., 2000, p. 22). Therefore it makes

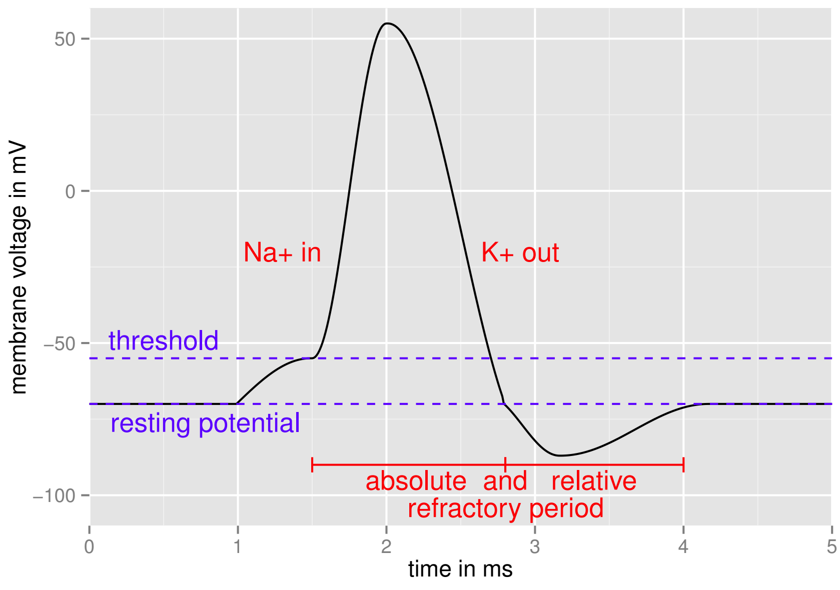


Fig. 2.6 Shematic of the change of membrane potential during an action potential. Based on figure from Dowling (1992, p. 80).

sense to model spikes as point processes, e.g. as a Poisson process. But Poisson processes are not an ideal model for neural spiking times (Barbieri et al., 2001) and one reason for this is the refractory period, in which they are less likely to fire than the exponentially distributed waiting times of a Poisson process would suggest (Kass and Ventura, 2001). Despite this, Poisson processes have been used extensively to analyze spiking data (e.g. Nawrot et al., 1999; Perkel et al., 1967).

Chapter 3

General Model & Sampler

This thesis deals with models where the parameters of a stochastic process change over time. These changes are sudden, i.e. the parameters jump between discrete values, and they are not directly observed. Our goal is to infer the path of the parameters over time from the, often noisy, observations of the observed process. Neither the number of jumps nor their times are known beforehand, and in the most recent model even the dimensionality of the hidden state space is not known beforehand.

3.1 General Model Description

The observable process $X(t)$ is based on a set of parameters θ which change over time. We write $\theta_{0:T}$ for the path of the parameters from time $t = 0$ to $t = T$. $\theta_{0:T}$ is piecewise constant and has c jumps at times $\tau_1, \ldots, \tau_c$ which partition it into $c+1$ segments. For better readability we define $\tau_0 = 0$ and $\tau_{c+1} = T$.

Given observations $\mathbf{D} = (d_1, \ldots, d_n)$ at times $\mathbf{t} = (t_1, \ldots, t_n)$, we are interested in the posterior

$$P(\theta_{0:T}|\mathbf{D}) = \frac{P(\mathbf{D}|\theta_{0:T})P(\theta_{0:T})}{P(\mathbf{D})}, \tag{3.1}$$

where $P(\mathbf{D}|\theta_{0:T})$ is the likelihood of the data giving the current path of the parameters and $P(\theta_{0:T})$ is the prior probability over the parameter path. Computing the evidence

$$P(\mathbf{D}) = \int P(\mathbf{D}|\theta_{0:T})P(\theta_{0:T})d\theta_{0:T} \tag{3.2}$$

is a non-trivial task. Luckily, we are using MCMC methods and the evidence cancels out in the acceptance probability. Therefore the computationally demanding part is calculating the likelihood of the data, conditioned on the current set of parameters $P(\mathbf{D}|\theta_{0:T})$.

Our model is build around two stochastic processes: $X(t)$ is the observable process which defines how the data is generated given the parameter values at a time and $\theta(t)$ is the hidden jump process of the parameters. The model is very flexible because different processes can be chosen both for the observable and the hidden process without the need to change the overall structure of the algorithm. We first present the different types of processes used in this work, then explain the general structure of the sampler before going into the details for different combinations of hidden and observed processes and their applications in the following chapters.

3.1.1 Types of Jump Process

We described that $\theta(t)$ is piecewise constant but there are several ways to specify what values the parameters can take after a jump. In this thesis three variants are described and examined. In all versions we assumed that the time until a new jump is exponentially distributed with parameter $f(\theta)$ which might depend on the last state of θ, i.e.

$$P(\tau_i|\tau_{i-1}) = f(\theta(\tau_{i-1}))\exp\left(-f(\theta(\tau_{i-1}))(\tau_i - \tau_{i-1})\right), \qquad \forall i \in \{1,\dots,c\}. \tag{3.3}$$

This is the same as to assume that the jump times are drawn from a Poisson process with rate $f(\theta)$. Additionally, in all our models we have a prior distribution over the parameters $P_\theta(\cdot)$ from which we assume the parameters for each state are drawn.

Fixed Number of States

In this case, it is known beforehand how many states the parameters can have. The probability distribution of the waiting time until the next jump depends on the current state, as does the probability distribution over the new state after a jump. This means that the process is a Markov jump process with a finite number of states.

This model was used in Stimberg et al. (2011a) and Stimberg et al. (2011b) for describing the dynamics of gene expression data and in that context the model had m independent telegraph processes $\mu(t)$. A telegraph process has only two states: 0 and 1. It switches from 0 to 1 with rate f_+ and from 1 to 0 with f_-.

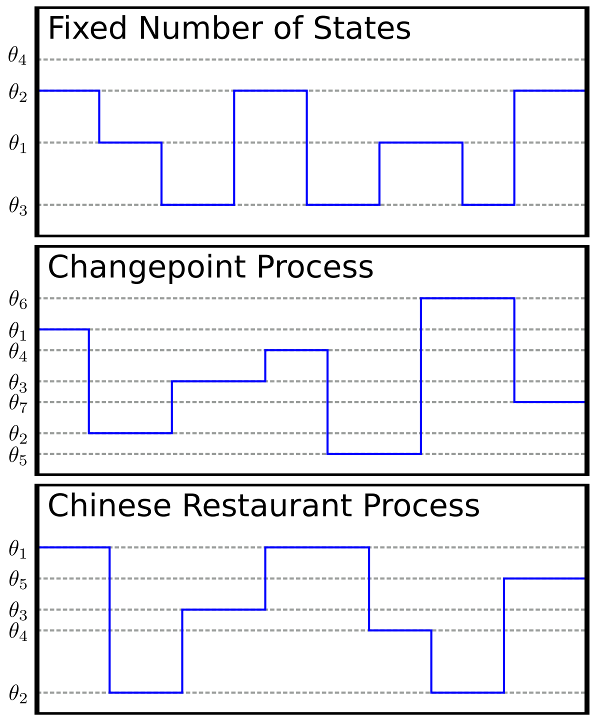


Fig. 3.1 The three types of hidden processes used in this thesis. The blue line is the path of the parameters $\theta_{0:T}$ while the dashed vertical line represents the parameter values at the different states. The process in the top with a fixed number of states is the only model where a state can exist while not being used, as it is the case for θ_4 in the example. The changepoint process in the middle creates a new state after every jump and the Chinese restaurant process in the bottom has an unknown number of states beforehand but after they are created they can be reused.

Changepoint Process

In this model, after each jump a new value for the parameters is drawn from a continuous probability distribution, therefore no parameter value is used for more than one segment. The number of changepoints until time t is counted by a Poisson process $\mu(t)$ and in our case its jump rate is independent of the state, i.e. it is always f. At every jump we draw a new parameter value from $P_\theta(\cdot)$ therefore the parameter values of the segments are all independent. This means that if there are c jumps, $c+1$ distinct parameter vectors are drawn from $P_\theta(\cdot)$ and we define $\theta(t) = \theta_{\mu(t)}$. In Stimberg et al. (2011a) this model was used to model stock index data. In many cases when talking about changepoint processes a model similar to the following is assumed (e.g. Chib, 1998; Fearnhead, 2006). Because of this we will refer to this model as the *changepoint model* in the remainder of this thesis, despite all the other models having changepoints as well.

Chinese Restaurant Process

Our last model can be seen as a combination of the former two. At each jump a new value is drawn from a Chinese Restaurant process. This means either a completely new value is drawn, similar to the changepoint process, or a value is selected from the set of values already assigned to former segments. Therefore a finite number of states exist, but the exact number is unknown beforehand and in contrast to the changepoint process model, the states are reusable. This is the same model as in Stimberg et al. (2012) and Stimberg et al. (2014) where it was used on gene expression and neuronal spiking data, respectively.

A priori we have an unknown number of discrete states $\theta_1, \dots, \theta_k$ and let π be the probability of a state being visited after a jump. We assume that π is drawn from a Dirichlet process. A Dirichlet process is described by a concentration parameter α and a base distribution $P_\theta(\cdot)$ which is the prior distribution over the parameters in our case. If we integrate π out we get a Chinese restaurant process (CRP) with the same parameters as the Dirichlet process (Teh, 2010).

In contrast to the other models the CRP is *not* a Markov process because the probability over the next state depends on the complete history of the process.

Figure 3.1 shows exemplary paths for the three types of hidden processes.

3.1.2 Types of Observed Process

$X(t)$ can be one of many types of stochastic processes but in this thesis we focus on Poisson and Ornstein-Uhlenbeck processes. Additionally, we summarize the results of Herrmann (2014), which modified the Ornstein-Uhlenbeck model to use a Cox-Ingersoll-Ross process as the observable process $X(t)$. A short explanation of how to customize the model to different processes, both hidden and observed, is given in section 6.3.

Poisson Process

Poisson processes are often used to model event data (Scargle, 1998; Wang et al., 2001) because they produce points in time without any information[1] attached to them. For an introduction to Poisson processes see section 2.1.2. In this thesis we assume that we get complete and exact data from the Poisson process, i.e. that for every event we have the exact time of its occurrence, without any noise. There are alternative models where the

[1]There is extensive literature on *marked point processes* (Jacobsen, 2006; Last and Brandt, 1995; Quick et al., 2014), including the marked Poisson process, which are point processes where a random element is associated with each event time. They are not used in this thesis but applying the sampler to work on models where the parameters of marked point processes change over time could be an interesting direction for the future.

data is binned and the observations are either binary (events in a bin or not) or the number of events in a bin is observed. Our sampler also works with these kind of models because the likelihood of the data given the hidden parameter process is still easily calculated as described in Sherlock (2006).

A Poisson process is completely defined by its rate parameter λ therefore this is the only parameter which is governed by the hidden process. A Poisson process where the rate is a stochastic process itself is called a Cox process and was first introduced by Cox (1955). The special case where the rate follows a Markov jump process is called a Markov modulated Poisson process (MMPP) and is widely used (see e.g. Rydén, 1996; Salvador et al., 2003; Yoshihara et al., 2001).

In Stimberg et al. (2014) a model where the rate $\lambda(t)$ is coming from a Chinese restaurant process is introduced and applied to neural spiking data. See chapter 5 for a detailed description of the models with Poisson data.

Ornstein-Uhlenbeck Process

In many applications we do not only have timed events but measurements of a process at discrete times. Often we cannot assume that the process is observed without error. This means that compared to the Poisson process model we have another layer of abstraction. Instead of knowing $X(t)$ directly we have observations $\mathbf{D} = (d_1, \dots, d_n)$ at discrete times $t_1, \dots, t_n$. A popular model for dynamic systems is the Ornstein-Uhlenbeck model. If the observations were exact we could directly calculate the likelihood using the transition probability of the OU process

$$\begin{aligned} P(\mathbf{D}|\theta_{0:T}) &= P(d_1|\theta_{0:T}) \prod_{i=2}^{n} P(d_i|d_{i-1}, \theta_{0:T}) \\ &= P(X(t_1|\theta_{0:T})) \prod_{i=2}^{n} P(X(t_i)|X(t_{i-1}, \theta_{0:T})). \end{aligned} \tag{3.4}$$

If we are not able to get exact observations, we assume the observations are corrupted, e.g. by i.i.d. Gaussian noise with variance σ_o^2

$$P(d_i|X(t_i)) = \mathcal{N}(d_i; X(t_i), \sigma_o^2). \tag{3.5}$$

We now need to compute the likelihood of the data giving a path of the hidden jump process of the parameters $P(\mathbf{D}|\theta_{0:T})$. Without any information about $X(t)$ this likelihood does not factorize over the observations. There are two approaches in computing it: Either we sample a path $X_{0:T}$ from $P(X_{0:T}|\mathbf{D}, \theta_{0:T})$, thereby introducing another Monte Carlo-step, and then compute

$$P(\mathbf{D}|X_{0:T},\theta_{0:T}) = \prod_{i=1}^{n} P(d_i|X(t_i),\theta_{0:T}), \tag{3.6}$$

or we marginalize out $X_{0:T}$ directly

$$P(\mathbf{D}|\theta_{0:T}) = \int P(\mathbf{D}|X_{0:T},\theta_{0:T})P(X_{0:T}|\theta_{0:T})dX_{0:T}. \tag{3.7}$$

In Stimberg et al. (2011b) the first version was used but luckily all transition densities are Gaussian and together with the Gaussian observation model we can integrate out $X_{0:T}$ analytically.

If $X(t)$ has multiple dimensions we distinguish between two cases: Either the individual dimensions are independent and the likelihood over all dimensions is the product of the individual likelihoods or the dimensions depend on each other and all probabilities are multivariate Gaussians.

Stimberg et al. (2011a,b) and Stimberg et al. (2012) use an Ornstein-Uhlenbeck process whose dimensions are independent and apply it to gene expression data as well as stock index data. In chapter 4 inference for models with OU data will be explained in detail and section 6.2 presents a small example of the sampler used on a model using a multivariate OU process.

Cox-Ingersoll-Ross Process

In Herrmann (2014) our model was modified to include a Cox-Ingersoll-Ross process. As described in section 2.1.2, the difference between the Ornstein-Uhlenbeck and the Cox-Ingersoll-Ross model is that the latter's diffusion depends on the value of the process itself. This complicates inference because the transition density is no longer Gaussian. For simplicity Herrmann (2014) assumed that the observations were exact. Thus the likelihood can be calculated from the transition density as in (3.4). While the transition density is available in close form it is a non-central chi-square distribution whose evaluation has very high computational costs. Therefore a Gaussian approximation was used and showed promising results. The model and algorithm was then applied on the EUR/USD-exchange rate to find changepoints corresponding to decisions made by the Federal Reserve of the United States during the subprime mortgage crisis in 2008.

3.2 General Sampler Architecture

Our aim is to sample from the posterior $P(\theta_{0:T}|\mathbf{D})$, i.e. we want to estimate the posterior distribution over the path of the parameters $\theta_{0:T}$ given a set of observations $\mathbf{D}$. We assume that the parameters are piecewise constant but in contrast to a lot of other changepoint models (e.g. Chib, 1998; Ko et al., 2015; Wang et al., 2004) we do not assume that changepoints only can happen at discrete times, e.g. the observation times. This means that $\theta_{0:T}$ is an infinite dimensional object and we cannot use classical methods for hidden Markov models. A possible approach would be to discretize time but this would introduce another source of approximation and the need to choose the level of discretization. Large time steps lead to high errors while small steps can become computationally demanding. Instead we stay in continuous time and note that a path of the hidden process is fully defined by the number and position of the jumps and the parameter values for each segment. We apply a Metropolis-within Gibbs approach as following:

Algorithm 1 General structure of the sampler algorithm.

```
$\theta_{0:T} \leftarrow$ random initialization
for 1 to $s$ do
    1. Sample parameter values $\theta$, given the jump times $\tau$ and data $\mathbf{D}$
    2. Sample jump times $\tau$ and the parameter values $\theta$, given the data $\mathbf{D}$
end for
```

The algorithm generates n samples by alternating between sampling the parameter values θ[2] conditioned on the jump times τ and the data $\mathbf{D}$, and sampling the jump times and the parameter values θ given the data $\mathbf{D}$. These steps are overlapping for some of the models but this is not a problem for a Gibbs sampler (see section 2.1.3).

3.2.1 Sampling the Parameters

The first step is usually performed as a Gibbs sampler as well, by cycling through all the parameters and sampling them from the conditional distribution given all the other parameters, the jump times and the data. If the parameters are all conditionally independent of each other, given the jump times and the data, this procedure is equivalent to sampling from the combined conditional distribution over all parameters. In some cases the parameters can be directly sampled from the true conditional densities because they have a simple form, e.g. are Gaussian or gamma distributed. If that is not the case a Metropolis-Hastings sampler is

[2]These include parameters which are constant, i.e. are not affected by the hidden jump process.

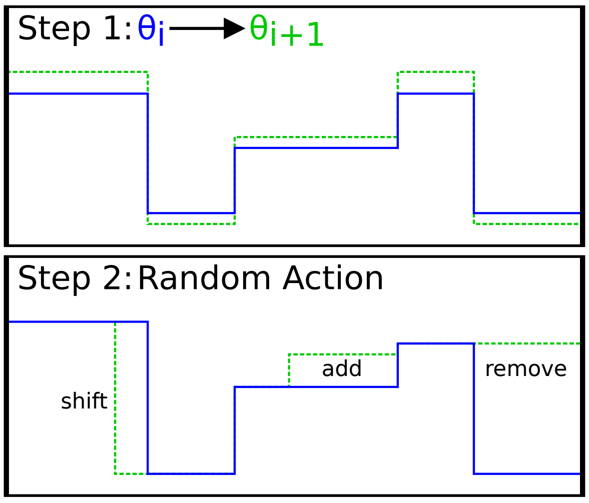


Fig. 3.2 The two steps of the Metropolis-within-Gibbs sampler. In the first step the parameter values are sampled either directly from the conditional densities or by a Metropolis-Hastings update. This includes parameters which remain constant over time. The second step changes the number of times of the jumps in the parameters by randomly applying one of multiple actions. The three actions shown here are common to all models but additional actions are introduced for some of the models to accelerate convergence times.

used either doing a random walk or a random walk on the logarithm if the parameters have to be positive.

3.2.2 Sampling the Jump Times

The more complicated and more interesting step in algorithm 1 is the second one. Here we perform a Metropolis-Hastings random walk on the path $\theta_{0:T}$ by proposing small changes through a number of possible actions. These actions depend on the type of the hidden process but always include

1. Shifting a jump in time
2. Adding a jump
3. Removing a jump

which is similar to the birth-death approach of Rotondi (2002) but is put in a more general framework and expanded to more complicated models in this thesis. The proposed path is then accepted with the Metropolis-Hastings acceptance ratio. Figure 3.2 illustrates the two steps on an example. While this approach might seem simple and not practical it is fast to compute, very flexible and easy to adjust to a number of different models, and able to swiftly reach regions of high probability in the space of all possible hidden paths $\theta_{0:T}$ even for complicated models. It can outperform exact Gibbs sampling approaches which sample a complete new path every time and have to be completely tailored to the specific model[3].

As discussed in section 2.1.3, a requisite for the Metropolis-Hastings algorithm to work is that the proposal distribution is able to reach every point in the sample space in a finite number of steps. This is clearly satisfied by the algorithm because through adding and removing jumps at random times all possible paths of $\theta_{0:T}$ can be reached.

3.3 Label-Switching

One problem for models such as this is label switching, i.e. the possibility to switch the state indices and corresponding parameter values without actually changing the path of the parameters $\theta_{0:T}$. This is only a problem for a fixed number of states, because in this model the state index can have a meaning. In the case of the changepoint and CRP process the indices are always numbered in ascending order by the time of their appearance. If the model is one-dimensional, the state indices can be sorted by the value of the parameters. When the random actions invalidate this sorting it is automatically restored.

For the fixed number of states model, where we have multiple telegraph processes, as described in section 3.1.1, the problem is that they can be exchanged together with their parameters without changing the likelihood of the data. One method to avoid this is to use prior information and have different prior densities over the parameters' values associated with the different telegraph processes. This leads to different posterior probabilities for the switched paths. Another way is to include knowledge that certain dimensions of the observed process are only influenced by one of the telegraph processes, i.e. setting certain parameters to 0 and not sampling them.

[3]See section 6.3 for a short explanation on this.

Chapter 4

Applications using the Ornstein-Uhlenbeck Process

Most of this chapter is based on Stimberg et al. (2011b), Stimberg et al. (2011a) and Stimberg et al. (2012) and therefore represents work done in collaboration with Manfred Opper, Andreas Ruttor and Guido Sanguinetti.

4.1 Fixed Number of States

The first model uses an Ornstein-Uhlenbeck process whose parameters are controlled by a telegraph process. This is a special case of models with a fixed state space dimensionality, i.e. where the exact number of states are known beforehand.

4.1.1 Switching Model

Our model is based on Sanguinetti et al. (2009), who modified the ODE model of transcriptional regulation of Barenco et al. (2006) to have telegraph processes represent the transcription factor activity. The model was then further expanded in Opper et al. (2010) to include system noise. Our model consists of an Ornstein-Uhlenbeck process whose drift and diffusion depend on a shared telegraph process $\mu(t)$. The telegraph process has two parameters f_+ and f_- representing the transition rates from 0 to 1 and 1 to 0, respectively. The OU process is defined through the following stochastic differential equation

$$dX = (b + A\mu(t) - \lambda X)dt + \sigma_{\mu(t)}dW. \tag{4.1}$$

In Opper et al. (2010) σ, the strength of the system noise, was constant but our sampler allows us to do inference even if σ switches between different values.

This model can be easily extended to have multiple dimensions $\mathbf{X} = (X_1, \dots, X_N)$, each with their own set of parameters, which are independent conditioned on the path of the telegraph process $\mu_{0:T}$. As our first application is on single-dimensional data and to avoid cluttered notation we start by describing only the single dimensional model and expand on it in section 4.2.

If we want to view (4.1) in the form of the general formulation of our models in section 3.1 $\theta(t)$ is a telegraph process which jumps between states $(b, \lambda, \sigma_0^2, f_+)$ and $(b+A, \lambda, \sigma_1^2, f_-)$.

For state and parameter inference purposes, $X(t)$ is observed at discrete points $\mathbf{t} = t_1, \dots, t_n$ in time and the observations $\mathbf{D} = (d_1, \dots, d_n)$ are corrupted by i.i.d Gaussian noise with variance σ_{obs}^2. The $\mu(t)$-process on the other hand is unobserved and can only be inferred from the observations of $X(t)$.

We assume Gaussian priors over b and A For the system noise parameters σ_0^2 and σ_1^2 we used a Gaussian prior which was truncated to be non-negative or a Gamma prior. On the other hand λ, f_+ and f_- are assumed to be gamma distributed a-priori. While it is possible to specify a prior over the observation noise σ_{obs}^2 and infer its value, as was done for the other parameters, we decided to let σ_{obs}^2 remain fixed.

If we set $\theta = (b, A, \lambda, \sigma_0, \sigma_1, f_+, f_-)$ then the joint probability becomes

$$P(\mathbf{D}, X_{0:T}, \mu_{0:T}, \theta) = P(\theta) P(\mu_{0:T}|\theta) P(X_{0:T}|\mu_{0:T}, \theta) P(\mathbf{D}|X_{0:T}). \tag{4.2}$$

We will go into details on the different parts which make up the joint probability in the following.

Prior Probability

The prior probability is made up from the prior probability over the path of the telegraph process $\mu_{0:T}$ and the the prior probabilities over the values of the parameters[1].

$$P(\theta) P(\mu_{0:T}|\theta) = P(\mu_{0:T}|f_+, f_-) P(b) P(A) P(\lambda) P(\sigma_0^2) P(\sigma_1^2) P(f_+) P(f_-). \tag{4.3}$$

[1]To simplify the notation here we write $P(b)$, $P(\mu_{0:T}|f_+, f_-)$ etc. instead of $P_b(b)$ and $P_\mu(\mu_{0:T}|f_+, f_-)$.

The prior over a μ path in the time interval $[\tau_0 = 0, \tau_{c+1} = T]$ with jumps at times $\tau_1 < \cdots < \tau_c$ is given by (Wilkinson, 2006, p. 221)

$$P(\mu|f_+, f_-) = P(\mu_{\tau_0}) f_1^{\lceil \frac{c}{2} \rceil} f_2^{\lfloor \frac{c}{2} \rfloor} \exp(-\sum_{i=1}^{\lceil \frac{c}{2} \rceil} f_1 \Delta\tau_{2i-1} - \sum_{i=1}^{\lfloor \frac{c}{2} \rfloor} f_2 \Delta\tau_{2i} - f_3 \Delta\tau_{c+1}), \tag{4.4}$$

with

$$\Delta\tau_i = \tau_i - \tau_{i-1},$$
$$f_1 = \begin{cases} f_+, \text{if } \mu(0) = 0 \\ f_-, \text{if } \mu(0) = 1 \end{cases}, f_2 = \begin{cases} f_-, \text{if } \mu(0) = 0 \\ f_+, \text{if } \mu(0) = 1 \end{cases}, f_3 = \begin{cases} f_+, \text{if } \mu(T) = 0 \\ f_-, \text{if } \mu(T) = 1 \end{cases}.$$

This can be easily shown by constructing the process from the exponential waiting time with rates f_+ and f_-. The prior probability over a path $\mu_{0:T}$ just becomes a product over exponential distributions and the probability that there is no jump between the last jump time and T.

Likelihood

We are interested in the likelihood function, conditioned on a path of the switching process and a set of parameters

$$\begin{aligned} P(\mathbf{D}|\mu_{0:T}, \theta) &= \int P(X_{0:T}|\mu_{0:T}, \theta) P(\mathbf{D}|X_{0:T}) dX_{0:T} \\ &= \int P(X(t_1)) \prod_{i=2}^{n} P(X(t_i)|X(t_{i-1}, \mu_{0:T}, \theta) P(d_i|X(t_i)) dX_{0:T}. \end{aligned} \tag{4.5}$$

If $\mu(t)$ is constant between t_{i-1} and t_i then from the solution of the Ornstein-Uhlenbeck process (Gardiner, 2009, p. 73) we obtain its transition probability

$$\begin{aligned} P(X(t_i)|X(t_{i-1}), \mu_{0:T}, \theta) &= \mathcal{N}(X(t_i)|m(t_{i-1}, t_i), v(t_{i-1}, t_i)) \\ m(t_{i-1}, t_i) &= X(t_{i-1}) \exp(-\lambda \Delta t_i) + \frac{b + A\mu_(t_{i-1})}{\lambda}(1 - \exp(-\lambda \Delta t_i)) \\ v(t_{i-1}, t_i) &= \frac{\sigma^2_{\mu(t_{i-1})}}{2\lambda}(1 - \exp(-2\lambda \Delta t_i)) \end{aligned}$$

where $\Delta t_i = t_i - t_{i-1}$.

With (4.6) we can write (4.5) as

$$P(\mathbf{D}|\mu_{0:T},\theta) = \int \mathcal{N}(d_1|X(t_1),\sigma^2_{obs}) \prod_{i=2}^{n} \mathcal{N}(X(t_i)|\alpha_i X(t_{i-1})+\beta_i,\xi_i)\mathcal{N}(d_i|X(t_i),\sigma^2_{obs}) dX_{0:T}, \tag{4.6}$$

with

$$\alpha_i = \exp(-\lambda \Delta t_i) \tag{4.7}$$

$$\beta_i = \frac{b + A\mu(t_{i-1})}{\lambda}(1-\alpha_i) \tag{4.8}$$

$$\xi_i = \frac{\sigma^2_{\mu(t_{i-1})}}{2\lambda}(1-\alpha_i^2), \tag{4.9}$$

if there are no jumps between t_i and t_{i-1}. Otherwise if there are jumps at $t_{i,1},\ldots,t_{i,k-1}$ then β_i and ξ_i can be computed iteratively by

$$\beta_{i,j} = \beta_{i,j-1}\alpha_{i,j} + \frac{b + A\mu(t_{i,j-1})}{\lambda}(1-\alpha_{i,j}) \tag{4.10}$$

$$\xi_{i,j} = \xi_{i,j-1}\alpha^2_{i,j} + \frac{\sigma^2_{\mu(t_{i,j-1})}}{2\lambda}(1-\alpha^2_{i,j}), \tag{4.11}$$

where $\alpha_{i,j} = \exp(-\lambda(t_{i,j} - t_{i,j-1}))$, $\beta_{i,0} = \xi_{i,0} = 0$, $\beta_{i,k} = \beta_i$, $\xi_{i,k} = \xi_i$, $t_{i,0} = t_{i-1}$ and $t_{i,k} = t_i$.

This enables us to solve the integral in (4.6) by setting $Z_n = P(\mathbf{D}|\mu_{0:T},\theta)$ and computing it recursively through

$$Z_i = Z_{i-1}\mathcal{N}(d_i|m_{Z_i} - d_i, v_{Z_i} + \sigma^2_{obs}) \tag{4.12}$$

$$m_{Z_i} = m_{i-1}\alpha_i + \beta_i \tag{4.13}$$

$$v_{Z_i} = \xi_i + v_{i-1}\alpha_i^2 \tag{4.14}$$

$$m_i = \frac{\sigma^2_{obs} m_{Z_i} + d_i v_{Z_i}}{\sigma^2_{obs} + v_{Z_i}} \tag{4.15}$$

$$v_i = \frac{\sigma^2_{obs} v_{Z_i}}{\sigma^2_{obs} + v_{Z_i}}, \tag{4.16}$$

with the start values m_1 and v_1 being the first observation and the observation noise variance, respectively.

4.1.2 Sampler

As discussed in section 3.2, we use a Gibbs sampler which switches between two different steps: sampling the jump process and sampling the parameter values. As initialization we set the starting parameters either by hand or use the means of the priors, while $\mu(t)$ is drawn from the prior conditioned on the starting parameters.

Sampling the Parameters

The parameters to sample are b, A, λ, σ_0, σ_1, f_+ and f_-.

Sampling the Transition rates f_+ and f_- Given the path $\mu_{0:T}$ the posterior distributions over f_+ and f_- are independent of the data D, therefore

$$P(f_+|\mu_{0:T}) \propto P(f_+)P(\mu_{0:T}|f_+,f_-) \tag{4.17}$$

$$P(f_-|\mu_{0:T}) \propto P(f_-)P(\mu_{0:T}|f_+,f_-). \tag{4.18}$$

From (4.4) we get

$$P(f_+|\mu_{0:T}) \propto P(f_+) \begin{cases} f_+^{\lceil \frac{c}{2} \rceil} \exp\left(-\sum_{i=1}^{\lceil \frac{c}{2} \rceil} f_+\Delta\tau_{2i-1} - f_+\Delta\tau_{c+1}\right), & \text{if } \mu(0)=0 \;\&\; \mu(T)=0,\\ f_+^{\lceil \frac{c}{2} \rceil} \exp\left(-\sum_{i=1}^{\lceil \frac{c}{2} \rceil} f_+\Delta\tau_{2i-1}\right), & \text{if } \mu(0)=0 \;\&\; \mu(T)=1,\\ f_+^{\lfloor \frac{c}{2} \rfloor} \exp\left(-\sum_{i=1}^{\lfloor \frac{c}{2} \rfloor} f_+\Delta\tau_{2i-1} - f_+\Delta\tau_{c+1}\right), & \text{if } \mu(0)=1 \;\&\; \mu(T)=0,\\ f_+^{\lfloor \frac{c}{2} \rfloor} \exp\left(-\sum_{i=1}^{\lfloor \frac{c}{2} \rfloor} f_+\Delta\tau_{2i-1}\right), & \text{if } \mu(0)=1 \;\&\; \mu(T)=1, \end{cases} \tag{4.19}$$

and

$$P(f_-|\mu_{0:T}) \propto P(f_-) \begin{cases} f_-^{\lfloor \frac{c}{2} \rfloor} \exp\left(-\sum_{i=1}^{\lfloor \frac{c}{2} \rfloor} f_-\Delta\tau_{2i-1}\right), & \text{if } \mu(0)=0 \;\&\; \mu(T)=0,\\ f_-^{\lfloor \frac{c}{2} \rfloor} \exp\left(-\sum_{i=1}^{\lfloor \frac{c}{2} \rfloor} f_-\Delta\tau_{2i-1} - f_-\Delta\tau_{c+1}\right), & \text{if } \mu(0)=0 \;\&\; \mu(T)=1,\\ f_-^{\lceil \frac{c}{2} \rceil} \exp\left(-\sum_{i=1}^{\lceil \frac{c}{2} \rceil} f_-\Delta\tau_{2i-1}\right), & \text{if } \mu(0)=1 \;\&\; \mu(T)=0,\\ f_-^{\lceil \frac{c}{2} \rceil} \exp\left(-\sum_{i=1}^{\lceil \frac{c}{2} \rceil} f_-\Delta\tau_{2i-1} - f_-\Delta\tau_{c+1}\right), & \text{if } \mu(0)=1 \;\&\; \mu(T)=1. \end{cases} \tag{4.20}$$

This has the form of a gamma distribution and if we choose gamma priors over f_+ and f_- then the posterior will be gamma distributed as well (George et al., 1993).

Sampling b and A If we look at the calculation of the likelihood in section 4.1.1, b and A are linear in $\beta_{i,j}$ which is linear in m_i which is linear in m_{Z_i}. This means that b and A

appear linear in the log-likelihood and we can use the same iterative approach which is used to compute the likelihood to propagate the mean and variance depending on b or A forward. For the detailed calculations see section A.3 of the appendix.

Sampling σ^2 and λ For both σ^2 parameters and λ the likelihood (4.5) does not have a simple form we can directly sample from, instead we apply a random walk Metropolis-Hastings algorithm.

Because σ_0^2, σ_1^2 and λ all have to be positive we use a random walk on the logarithm, i.e. we draw a proposal from a log-normal density with the current parameter value as its mode

$$Q(\lambda^*|\lambda) = \frac{1}{\lambda^* \sigma_{lrw}\sqrt{2\pi}} \exp\left(-\frac{(\log(\lambda^*) - \log(\lambda))^2)}{2\sigma_{lrw}^2}\right), \tag{4.21}$$

and accordingly for σ_0 and σ_1. In contrast to the normal random walk this proposal is not symmetric and the acceptance probability becomes

$$\min\left(1, \frac{P(\mathbf{D}|\mu_{0:T}, \theta^*)}{P(\mathbf{D}|\mu_{0:T}, \theta} \frac{P(\lambda^*)}{P(\lambda)} \frac{Q(\lambda|\lambda^*)}{Q(\lambda^*|\lambda)}\right), \tag{4.22}$$

where

$$\frac{Q(\lambda|\lambda^*)}{Q(\lambda^*|\lambda)} = \frac{\lambda^*}{\lambda}. \tag{4.23}$$

One advantage of the log random walk is that it scales automatically because the stepsize is determined by σ_{lrw} which is the standard deviation of the Gaussian *in log-space*. If not otherwise specified $\sigma_{lrw} = 0.1$ was chosen in this thesis.

Sampling the Jump Process

We sample from the posterior of the jump process given the parameter values and the data by applying a Metropolis-Hastings random walk. The sampler proposes a small modification of the current jump process and then accepts it with the MH acceptance probability. The modification is done by randomly choosing one of five actions. We first describe exactly what each of the actions do and then formulate their acceptance probabilities.

Shifting the time of a jump One of the jumps is chosen with equal probability and the new time of the jump is drawn from a Gaussian distribution with standard deviation σ_t, centered around the current time of the jump and truncated at the neighboring jumps. σ_τ was chosen by hand in our case and should be in the same order of magnitude that is is expected as the time between jumps. If we are unsure about what value to set σ_τ to, a higher value is

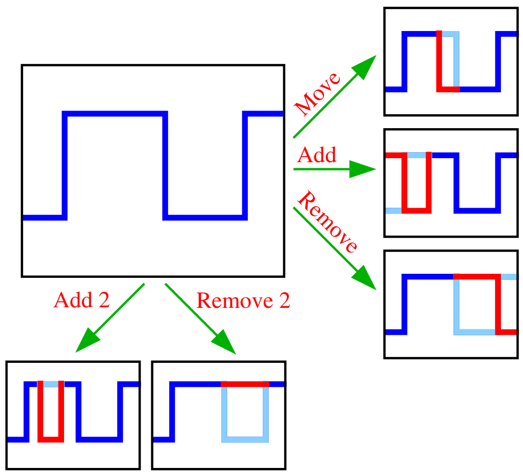


Fig. 4.1 Generating a proposal path by modifying the current path of the telegraph process with one out of five possible actions. The part of the path which was left unchanged is drawn in dark blue, the modified part is red, while the old path which differs from the proposal is in light blue.

preferable, because in the limit this will let the truncated Gaussian density become a uniform one. We could from the start draw the jump times from a uniform distribution between the neighboring jumps but this approach can lead to long convergence times, especially when there are long time periods without jumps.

Adding a jump When we add a new jump its time is drawn uniformly from the interval $[0,T]$. In the case of the telegraph process a new jump means either the whole path after it or before it will be inverted. We choose one of both options with probability 0.5.

Removing a jump One of the jumps is chosen at random with probability $1/c$ and removed. The path is inverted before or after the removed jump with equal probability. This action can only be chosen if there is at least one jump in the current path.

These three actions are enough to explore the space of all possible paths but the acceptance rates for adding or removing only one jump will be low most of the time, because a large part of the process will be inverted. To get faster convergence two more actions are introduced.

Adding two jumps We add two neighboring jumps by adding the first one uniformly over the whole time span, as is done for the action of adding a single jump. The time of the second jump is then drawn uniformly from the interval between the first added jump and the time of the next jump (or T if the jump was added at the end).

action	probability
shift jump time	$q_{sh} = 0.5$
add single jump	$q_a = 0.05$
remove single jump	$q_r = 0.05$
shift jump time	$q_{a2} = 0.2$
shift jump time	$q_{r2} = 0.2$

Table 4.1 Probability distribution over the 5 possible actions.

Removing two jumps One of the jumps (except the last one) is randomly chosen with equal probability $1/(c-1)$ and is removed together with the following jump. This action can only be chosen if there are at least two jumps in the current path.

Because the telegraph process is only changed between the two added or removed jumps both these actions normally have a higher acceptance rate than the corresponding actions, which only add or remove a single jump.

All 5 possible actions and their effects on the path are shown in figure 4.1. The probability distribution from which the next action is drawn was set manually. Table 4.1 shows the distribution we used if not specified otherwise. As shifting the time of a jump and removing a jump is only possible when there is at least one jump and removing two jumps can only be chosen when there are two or more jumps the distribution is normalized to include only the possible actions.

Acceptance Probabilities

After manipulating the $\mu_{0:T}$ path to generate a proposal $\mu^*_{0:T}$ we accept it with probability

$$p_{MH} = \min\left(1, \frac{P(D|\mu^*_{0:T},\theta)}{P(D|\mu_{0:T},\theta)} \frac{P(\mu^*_{0:T}|f_+,f_-)}{P(\mu_{0:T}|f_+,f_-)} \frac{Q(\mu_{0:T}|\mu^*_{0:T})}{Q(\mu^*_{0:T}|\mu_{0:T})}\right). \tag{4.24}$$

The likelihood ratio doesn't depend on the action chosen but both the prior and the proposal ratio do. We set

$$\Psi = \frac{P(\mu^*_{0:T}|f_+,f_-)}{P(\mu_{0:T}|f_+,f_-)} \frac{Q(\mu_{0:T}|\mu^*_{0:T})}{Q(\mu^*_{0:T}|\mu_{0:T})}, \tag{4.25}$$

as the part of the acceptance probability which depends on the action. The equations in the following arise from the description of the proposal process and the prior likelihood over a path (4.4).

Shifting the time of a jump For simplicity we assume that the switching rates are symmetrical, but this is not a requirement. This means that moving the position of a jump does not change the prior, thus it cancels out in Ψ but the proposal density is not symmetrical because it is truncated at the neighboring changepoints. If we move a changepoint from time τ to τ^* and the proposal density is truncated by $\tau_{\min}$ and $\tau_{\max}$, then we get

$$\Psi = \frac{\Phi((\tau_{\max} - \tau)/\sigma_\tau) - \Phi((\tau_{\min} - \tau)/\sigma_\tau)}{\Phi((\tau_{\max} - \tau^*)/\sigma_\tau) - \Phi((\tau_{\min} - \tau^*)/\sigma_\tau)}, \tag{4.26}$$

where σ_τ is the standard deviation of the Gaussian and $\Phi(\cdot)$ is the cumulative distribution function of the standard normal distribution.

Adding a jump When adding a jump the ratio becomes

$$\Psi = \frac{q_r T}{q_a(c+1)} \frac{f}{1}, \tag{4.27}$$

where f is either f_+ or f_- depending on weather a jump from 0 to 1 or 1 to 0 was added and q_r and q_a are the probabilities to remove and add a jump, respectively.

Removing a jump The ratio for removing a jump is the inverse of the ratio for adding a jump but with c instead of $c+1$:

$$\Psi = \frac{q_a(c)}{q_r T} \frac{1}{f}, \tag{4.28}$$

where f is f_+ if the jump we removed went from 0 to 1 and f_- if it was from 1 to 0.

Adding two jumps When adding two jumps we draw twice from different uniform densities and this leads to

$$\Psi = \frac{q_{r2} T \Delta t_{\text{next}}}{q_{a2}(c+1)} \frac{f_+ f_-}{1}, \tag{4.29}$$

where Δt_{next} is the time between the first added jump and the next jump (or T) and q_{a2} and q_{r2} are the probabilities to add two jumps and remove two jumps, respectively.

Removing two jumps Removing two jumps chooses one out of the $c-1$ pairs of neighboring jumps and the ratio becomes

$$\Psi = \frac{q_{a2}(c-1)}{q_{r2} T \Delta t_{\text{next}}} \frac{1}{f_+ f_-}. \tag{4.30}$$

Samples from the OU Posterior

As described in section 4.1.1 we integrate out the path $X_{0:T}$ in our computation of the likelihood. While this improves the convergence of the sampler[2] we no longer get samples from the posterior over the observed process $X(t)$. But we might still be interesting in looking at the estimates at the time of the noisy observations or in seeing how the process behaves between the observations. Conditioned on a particular sample of the parameters θ and the jump process $\mu(t)$, we are able to generate exact samples from $X(t)$ without any discretization error (see Archambeau et al., 2008). For more details on this see the appendix A.1. Depending on the time resolution we are interested in, this can be computationally very demanding therefore it makes sense to only use thinned out samples from the parameter and jump process posterior.

4.1.3 Results

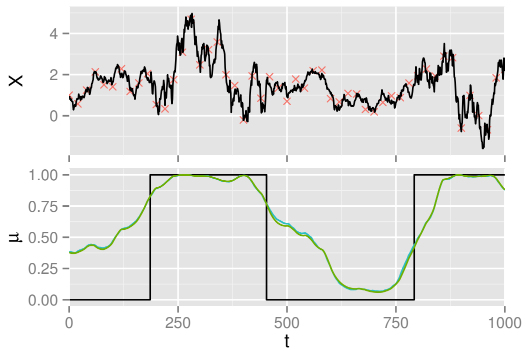
We first verify our sampler on synthetic data and then apply it to real gene expression data and analyze the benefits of letting the amplitude of the system noise switch.

One-Dimensional Synthetic Data

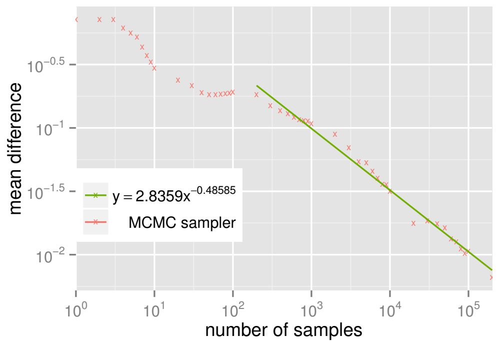
We generate synthetic data from the model. Because there is only a single telegraph process controlling the parameters of a one dimensional OU process we have the possibility to compare the sampler's results to a numerically computed exact solution.As described in Stimberg et al. (2011a) exact inference can be done by a smoothing algorithm similar to the forward-backward approach (Baum et al., 1970) used for state inference in hidden Markov models. Because the method involves numerically solving partial differential equations by integrating on a grid it is only feasible for low-dimensional systems and only for the specific model of the hidden process with a fixed number of states. Nevertheless it allows us to check that the sampler truly converges towards the exact solution for these models.

We use one-dimensional data with one switching process which only affected the system noise (i.e. $A = 0$). This makes inferring the hidden jump process especially hard because only the variance of the OU process is influenced by the jumps. For synthetic results on data where A was inferred as well see section C.1 of the appendix. To make our comparison to the exact solution be solely based on the sampling of the jump process we fixed all parameters to their true values ($b = 0.02, \lambda = 0.02$, $\sigma_0^2 = 0.01$, $\sigma_1^2 = 0.09$, $f_+ = f_- = 0.005$, $\sigma_{obs}^2 = 0.05$). In figure 4.2a the data is shown together with the results of the posterior inference for the μ

[2]This technique of analytically integrating out variables to improve the convergence is called Rao-Blackwellization (Casella and Robert, 1996).



(a) Data and posterior results, both from the sampler and the numerical exact solution, for a model where only the strength of the system noise is controlled by the switching process. (*top*) Noisy observations as red crosses, true $X(t)$ process as black line. (*bottom*) True $\mu(t)$ process as black line, exact $\mu(t)$ posterior mean as a green line and sampled $\mu(t)$ posterior mean as a blue line.



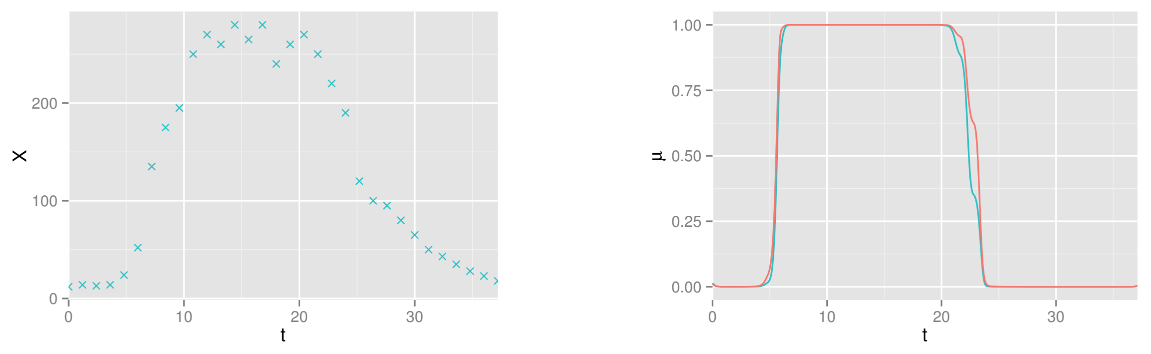
(b) Convergence of the MCMC sampler's results towards the exact solution. A power law model of the form $y = ax^b$ was fitted to the difference after the initial 100 samples and shows that the sampler's results converge towards the numerically estimated exact solution, as described by Stimberg et al. (2011a).

Fig. 4.2 Comparison of the MCMC sampler results with the numerical exact solution for synthetic OU data from the switching model.

process. After 200,000 samples the results of the exact inference and the sampler are almost indistinguishable and figure 4.2b shows that the convergence is roughly proportional to the square root of the number of samples which is what is expected for Monte Carlo methods (see Kalos and Whitlock, 2008, pp. 77-79). These results indicate that our MCMC sampler manages to create samples from the desired posterior distribution after an initial burn-in period and converges there reasonably fast.

ComS Protein Expression Data

According to recent studies, made possible by the progress in microscopy technology, stochasticity plays an important role in biochemical networks (Shahrezaei and Swain, 2008). An important distinction in this context is made between intrinsic and extrinsic noise (Elowitz et al., 2002). The former results from fluctuations due to the low number of molecules in the system, while the latter is a consequence of the system being influenced from external events. There is still no consensus how to characterize the difference between intrinsic and extrinsic noise formally. A widely shared belief is that the different types of noise either differ in their amplitude or in their spectral characteristics (Eldar and Elowitz, 2010). We use our model on real gene expression data subject to extrinsic noise in *Bacillus subtilis* (Suël et al., 2006)



(a) Measurements of the fluorescence intensity for the ComS protein over 36 hours.

(b) Posterior probability of ComK activation over time for the model with fixed σ^2 (*blue*) and the model with switching σ^2 (*red*).

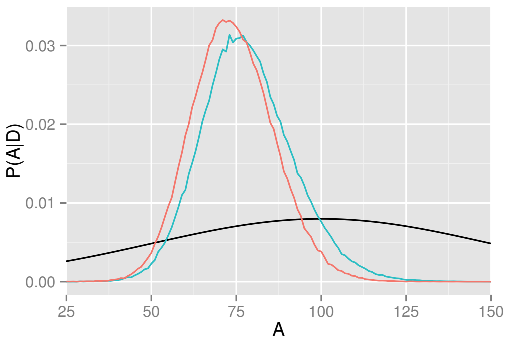
Fig. 4.3 Results of the inference for the ComS Protein expression data.

to investigate these questions. The data consists of fluorescence levels of the protein ComS taken from a single cell using time-lapse microscopy over 36 hours. The protein's production rate was influenced by extrinsic noise through the competence transcription factor ComK, which plays an important role in controlling ComS' expression (Turgay et al., 1997). When the expression level of the ComS protein is up-regulated the cell undergoes competence, a state in which the cell is able to take up extracellular DNA (Solomon and Grossman, 1996). Figure 4.3a shows the observations of the ComS fluorescence.

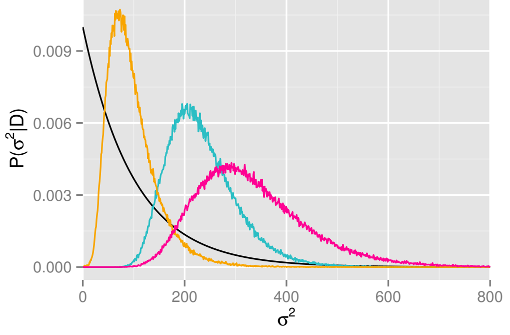
If we model ComS gene expression data by an OU process defined by (4.1) the parameters have biological interpretations. The base transcription rate of the ComS gene is represented by b. When the ComK transcription factor activates the transcription rate becomes $A+b$, therefore A represents the effect of the transcription factor: A negative value of A means it is down-regulating the gene ComS gene, a positive value of A means it is up-regulating. The degradation rate of the ComS protein is represented by λ and σ_0 and σ_1 model the strength of the intrinsic noise level in the transcription process without and with ComK being active, respectively. The rate of ComK activation is described by f_+ and the rate to inactivate by f_-.

Compared to previous models, e.g. the one used in Opper et al. (2010), our model is able to have the strength of the system noise depend on the unobserved ComK activity[3]. Using this advantage we want to compare two different models: The first model allows both A and σ^2 to depend on the state of μ, i.e. both the strength and the noise level of the gene expression depend on the ComK activity. In the second model the noise level remains

[3]The mean field approximation used in Opper et al. (2010) assumes that the Gaussian process approximating the true posterior has the same variance as the original process.



(a) Prior (*black*) and posterior density over A for the model without switching system noise (*blue*) and with switching system noise (*red*).



(b) Prior (*black*) and posterior density over σ for the model with fixed system noise (*blue*) and with switching system noise (*orange*: σ_0^2, *magenta*: σ_1^2).

Fig. 4.4 Parameter posteriors for the ComS Protein expression data.

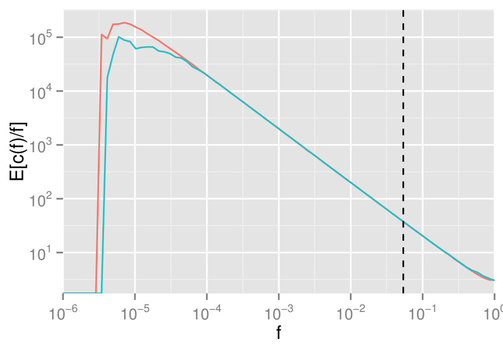
parameters	prior	hyper-parameters
f_+	fixed	$f_+ = 0.05$
f_-	fixed	$f_- = 0.05$
b	Gaussian	mean $= 0$, std. $= 50$
A	Gaussian	mean $= 100$, std. $= 50$
σ_0^2	Gamma	shape $= 1$, scale $= 100$
σ_1^2	Gamma	shape $= 1$, scale $= 100$
λ	Gamma	shape $= 1$, scale $= 100$

Table 4.2 Parameters and prior distributions for the ComS dataset.

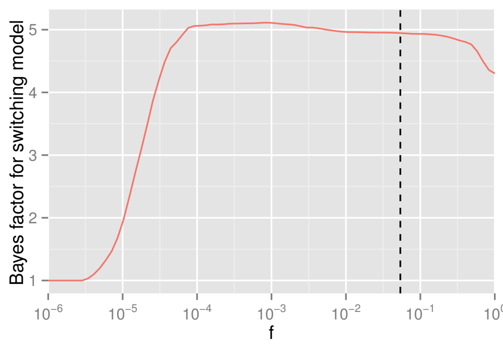
constant and only A switches with ComK[4]. For both models we let our sampler generate 510,000 samples and discarded the first 10,000 as burn-in. This took about 1 minute on a Intel Xeon CPU with 2.40 GHz. For the parameter values and priors for this simulation see table 4.2.

The posterior probability over time that ComK is active is shown in figure 4.3b and it is evident that both models predict that ComK is activated around 5 hrs and deactivated around 23 hrs into the experiment. The posteriors over A are very similar as well, as can be seen in figure 4.4a. Our first model produces two well separated posteriors for σ_0^2 and σ_1^2 (see figure 4.4b) which supports the belief that it better explains the data by allowing the noise to switch together with the activation profile of ComK. For the posterior densities of the b and λ parameters see appendix C.2.

[4]This model is identical to the one used by Opper et al. (2010).



(a) Expected value of c/f over f on a log-log scale for the model with switching σ^2 (red) and the model with fixed σ^2 (blue).



(b) Bayes factor between the model with switching σ^2 and the model with fixed σ^2 over f.

Fig. 4.5 Bayes factor estimation between models with and without switching system noise. The vertical dashed line is at $f = 0.0538$ where the prior mean number of jumps would be exactly 2.

To further look into this we want to compute the Bayes factor between both models we investigated. Our sampler produces samples from the posterior over the parameters given the data and one could think that these samples could be used to compute the evidence of the models and thus the Bayes factor. One approach to this is called the harmonic mean estimator (Raftery et al., 2007). Unfortunately, even for simple models, the harmonic mean estimator can give biased results, and even worse the estimator can have a low variance, suggesting that the computed Bayes factor is very accurately approximated by the samples (Calderhead and Girolami, 2009; Vyshemirsky and Girolami, 2008)[5]. A better method to compute Bayes factors is the thermodynamic integrator as described by Calderhead and Girolami (2009) which has been successfully used for Bayesian model selection (Goggans and Chi, 2004). Thermodynamic integration defines the power posterior

$$P_t(\theta|D) \propto P(D|\theta)^t P(\theta) \tag{4.31}$$

which interpolates between the prior ($t = 0$) and the posterior ($t = 1$). The evidence $P(D)$ can then be estimated by sampling from the power posterior for different temperatures t. We take a similar approach but instead of integrating over the temperature we can integrate over the jump rate. If, as we have done in our simulations, we fix f_+ and f_- to a fixed value $f = F$

[5]Another method using samples from the parameter prior doesn't have that problem but for a high-dimensional parameter space it becomes infeasible.

we can express the log evidence as

$$\log P(D|f=F) = \int_0^F \left(E_{P(\mu_{0:T},\theta|D,f=f')} \left[\frac{c}{f'} \right] \right) df' - TF + \log P(D|f=0). \tag{4.32}$$

where $c(f')$ is the expected number of posterior jumps with the jump rate set to f'. For a derivation of this result see section A.4 of the appendix. To compute this we ran the sampler for 100 values from $f = 1.5 \cdot 10^{-8}$ to $f = 1$ for both models and then computed the expected value of $c(f|D)/f$. In Figure 4.5b the results are plotted on a log-log scale showing that for small f the switching model seems to be above the non-switching, while for higher values the opposite is true. To get the log-evidence $\log P(D)$ we need to integrate this over f. Figure 4.5b shows the Bayes factor over f, we see that after the first jumps appear the switching model is clearly favored but for values over $f > 0.3$ it starts to drop. This is not surprising because a high jump rate allows to explain the different levels of system noise by jumping between the states more often. For $f = 1$, the highest value we used, the prior mean number of jumps is almost 40 while we saw only 2 clear jumps in figure 4.3b. Even for this value the Bayes factor in favor of the switching of σ^2 is ≈ 4.3 which according to table 2.1 represents *substantial* evidence that the model with switching system noise is a better model for the data.

While the OU model seems to be a good fit to the biological system of transcription it should not be forgotten that it is only an approximation of the true process. One major approximation is that we assume the fluorescence level is continuous, but it is a measure of the number of molecules, which is clearly a discrete variable. The models of Barenco et al. (2006) and Sanguinetti et al. (2009) approximate this by applying the system size expansion of Van Kampen (2011), which assumes that the number of molecules is large. In bacteria transcription and translation are tightly coupled (Gowrishankar and Harinarayanan, 2004) therefore it is feasible to assume the production of protein molecules as events coming from a Poisson process.

In our model the birth rate of ComS molecules would be b while ComK is inactive and switch to $A+b$ when ComK activates. The death rate is assumed to be λ in both cases. This birth-death model suggests that the steady state protein levels would be Poisson distributed with parameter

$$\rho_0 = \frac{b}{\lambda} \tag{4.33}$$

when ComK is inactive and parameter

$$\rho_1 = \frac{A+b}{\lambda} \tag{4.34}$$

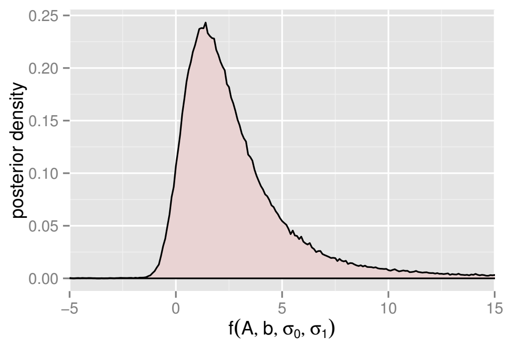


Fig. 4.6 Posterior density over $f(A,b,\sigma_0,\sigma_1)$ defined by (4.36). The probability mass is clearly not centered around 0, which would be expected according to the birth-death assumption the model is based on.

when it is active(Lafuerza and Toral, 2011). The observations are only proportional to the actual molecule count because they are measured in arbitrary units of fluorescence which doesn't allow us to test if the mean and variance are equal as a Poisson distribution would suggest. Instead we look at a quantity which is independent of the fluorescence units

$$\frac{\frac{mean(X_1)}{stdev(X_1)}}{\frac{mean(X_0)}{stdev(X_0)}} = \sqrt{\frac{\rho_1}{\rho_0}} = \sqrt{\frac{A+b}{b}}, \tag{4.35}$$

where X_0, and X_1 are the fluorescence levels when ComK is inactive and active, respectively. This is the ratio of the signal to noise ratio in both states and while it should emerge from the underlying birth-death model it is not enforced by our Ornstein-Uhlenbeck model.

In order to test if the simple birth-death assumption is true for the data we plot the difference between the posterior estimate of the ratio of the signal to noise ratios in both states and the prediction of this quantity from the birth-death model. This is a function of A, b, σ_0 and σ_1:

$$f(A,b,\sigma_0,\sigma_1) = \frac{\frac{A+b}{\sigma_1}}{\frac{b}{\sigma_0}} - \sqrt{\frac{A+b}{b}}. \tag{4.36}$$

If the birth-death model is correct the samples of $f(A,b,\sigma_0,\sigma_1)$ should be clustered around zero, meaning that the posterior estimate fits the model's prediction. As we can see in figure 4.6 this is not the case, instead almost all the samples are positive indicating that the predictions of the steady state birth-death models are not supported by the data.

One explanation for this difference could be that the continuous approximation of the chemical master equation is very inaccurate because the number of molecules is low. This would be very surprising because SDEs have been widely recommended for this purpose (Wilkinson, 2006). Another explanation would be that in contrast to our model's assumption not only the expression rate and amplitude of the noise are affected by the activation of ComK but the degradation rate λ as well. If corresponding to the system noise the protein degradation rates would be λ_0 and λ_1 in the case of ComK being inactive and active, respectively, then equation (4.36) would be

$$f(A,b,\sigma_0,\sigma_1) = \frac{\frac{A+b}{\lambda_1\sigma_1}}{\frac{b}{\lambda_0\sigma_0}} - \sqrt{\frac{\frac{A+b}{\lambda_1}}{\frac{b}{\lambda_0}}}. \tag{4.37}$$

This would suggest that ComK regulates the ComS expression both transcriptionally and post-transcriptionally. Our sampler works with such a model and it would be an interesting future direction but for the moment we are more interested in models with a more complex structure of the hidden process and decide to head in that direction instead.

4.2 Multiple Switching Processes

So far our model has been one-dimensional and has had one latent binary state telegraph process. We now extend it to have a higher dimensional state space and a multidimensional OU process as the observed process.

4.2.1 Model

It would be straightforward to extend this model to more states but for our application in systems biology a different strategy is more promising. Instead of a single telegraph process, we have multiple ones each with their own jump rates f_+, f_-. In the setting of transcriptional regulation each of these switching processes represents the binary (active / inactive) state of a transcription factor. The observed process has multiple dimensions which are independent conditioned on the hidden telegraph processes. When there are multiple telegraph processes there can be combinatorial effects, i.e. if two telegraph processes are active the effects are not just added but there might be a non-linear or multiplicative effect.

Formally, we assume N SDEs of the OU type:

$$dX_i = (\mu(t)^\top \mathbf{A}_i \mu(t) + b_i - \lambda_i X_i)dt + \sigma_i dW_i, \text{ for } i = 1, \ldots, N \tag{4.38}$$

where $\mu(t) \in \{0,1\}^k$ and $\mathbf{A}_i$ is a k-by-k matrix with $(\mathbf{A}_i)_{j,l}$ fixed to zero for $l > j$ to avoid ambiguity. To illustrate this on an example: If $k = 3$ and the first and the third of the switching processes are active at the current time we get

$$\begin{aligned}
\mu(t)^\top \mathbf{A}_i \mu(t) &= \begin{pmatrix} 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} (\mathbf{A}_i)_{1,1} & 0 & 0 \\ (\mathbf{A}_i)_{2,1} & (\mathbf{A}_i)_{2,2} & 0 \\ (\mathbf{A}_i)_{3,1} & (\mathbf{A}_i)_{3,2} & (\mathbf{A}_i)_{3,3} \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \\
&= (\mathbf{A}_i)_{1,1} + (\mathbf{A}_i)_{3,3} + (\mathbf{A}_i)_{3,1},
\end{aligned} \tag{4.39}$$

which is the sum of the effects from the first and third switching process being active on their own ($(\mathbf{A}_i)_{1,1}$ and $(\mathbf{A}_i)_{3,3}$) and the combinatorial effect of both of them being active at the same time ($(\mathbf{A}_i)_{3,1}$). This is a generalization of the model used in Opper and Sanguinetti (2010) which was only formulated for exactly two switching processes.

We could extend the effects of the switching processes to the λ_i and σ_i parameters like we did in the previous section but we restrict the model here to the switching of the production rate A_i because we are mainly interested in seeing how the sampler's results compare to the variational approximation of Opper and Sanguinetti (2010).

Likelihood

The calculation of the likelihood $P(\mathbf{D}|\mu_{0:T}, \theta)$ remains the same as described in section 4.1.1 with the only difference that $A\mu(t)$ has to be replaced by $\mu(t)^\top \mathbf{A}_i \mu(t)$ everywhere. As described before, conditioned on $\mu_{0:T}$ the individual dimensions of the OU process are independent in this model, which means the likelihood factorizes over the dimensions.

4.2.2 Sampler

The sampling of the parameters follows the same pattern as in section 4.1.2 because conditioned on the hidden jump process, the different dimensions of the OU process are independent. The entries of the $\mathbf{A}_i$ matrices all are linear in (4.38) and therefore the likelihood is Gaussian with respect to them and they can be each sampled in a Gibbs step similar to the single A parameter in the one-dimensional model.

Sampling of the multiple switching processes is performed in Gibbs steps. For each switching process a separate proposal is generated by applying one of the five proposal actions and rejected or accepted with acceptance probabilities as specified in section 4.1.2.

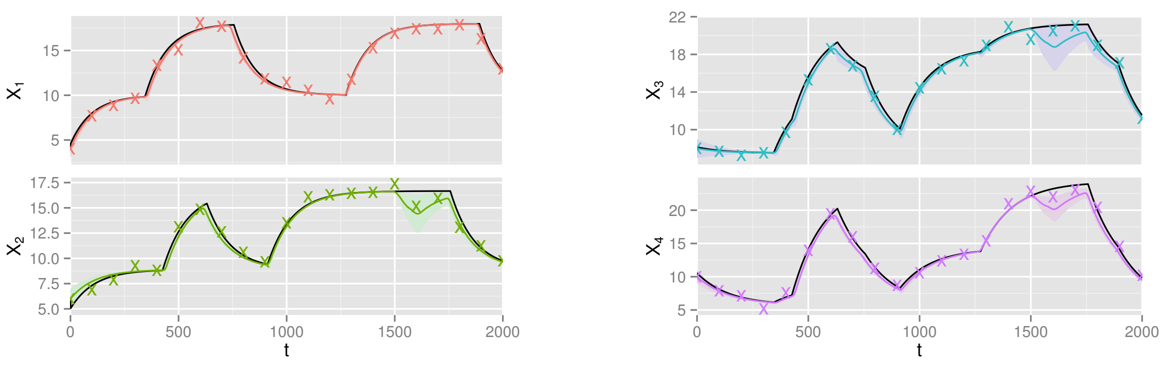


Fig. 4.7 Four-dimensional Ornstein-Uhlenbeck process with two telegraph processes controlling A. Observations are plotted as crosses. The black line is the true $X(t)$ process, while the colored lines are the posterior mean with two times the standard deviation as a confidence interval around it.

4.2.3 Results

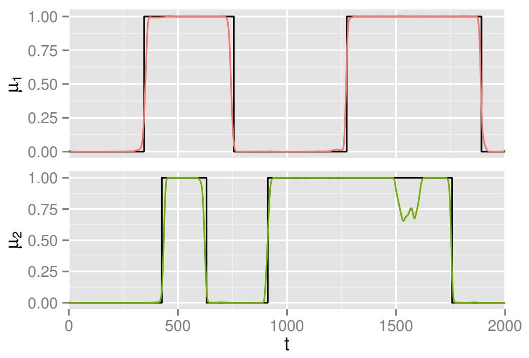
With multiple switching processes and a multidimensional observed process

$$\mathbf{X}(t) = (X_1(t), \ldots, X_N(t)) \tag{4.40}$$

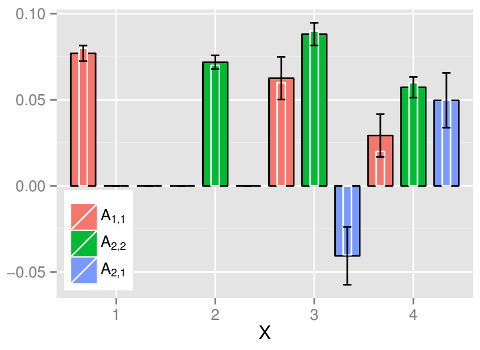
the exact solution from Stimberg et al. (2011a) we used to verify our sampler's results in section 4.1.3 is no longer feasible. We would need to numerically integrate on a high-dimensional grid. Instead we use the results on synthetic data to verify that our posterior estimates fit the true hidden process and parameters. After this the sampler is used on gene expression data from yeast cells which was used by Opper and Sanguinetti (2010) as well to see how it compares to the variational approximation when applied to real data.

Multi-Dimensional Synthetic Data

We generated a four-dimensional dataset controlled by two telegraph processes. The sampler knew that the first and second dimension of the OU process were only affected by the first and second telegraph process, respectively, i.e. all other entries of $\mathbf{A}_1$ and $\mathbf{A}_2$ were fixed to 0. The observations are plotted in figure 4.7 while the posterior probability of the telegraph processes being in state 1 over time is shown in figure 4.8a and compared to the true process. The posterior almost perfectly matches the true process. The results of the parameter inference for A are shown in figure 4.8b. The posterior estimates match the true values and the parameter $(\mathbf{A}_i)_{2,1}$, which is only used when both telegraph processes are active at the same time, has the highest variance. This is not surprising as it depends on the state of both switching processes.



(a) Posterior probability for both telegraph processes to be switched on as the red and green line respectively. The true path of the telegraph processes is drawn in black.



(b) Posterior probabilities over A. The bars with a black outline are the posterior mean, with two times the standard deviation as error bars. The white outlined bars are the true values.

Fig. 4.8 Posterior over the telegraph processes' activation and the A parameter for the four-dimensional OU synthetic dataset.

Yeast Cell Gene Expression Data

The data is a subset of the data from Tu et al. (2005) who measured the gene expression of yeast cells going through metabolic cycles using microarrays. Three cycles were induced by alternating between forced starvation and providing glucose to the yeast. We chose the same set of 10 genes as Opper and Sanguinetti (2010), who used the ChIP-on-chip[6] experiments of Harbison et al. (2004) and Lee et al. (2002) to select the genes which are influenced by the two transcription factors FHL1 and RAP1. FHL1 and RAP1 are known to play important roles in the control of ribosomal protein production (Schawalder et al., 2004).

The knowledge that three of the genes are solely regulated by FHL1 and two only by RAP1 was included in the model to avoid identifiability problems. We did not sample the parameters for this data but used the maximum-likelihood results of the variational approximation of Opper and Sanguinetti (2010). This was done because we are mostly interested in the inference of the transcription factors' activity and using the same parameters allows us only to focus on the quality of that part of the approximation.

As can be seen in figure 4.9, the posterior over $\mu(t)$ from our sampler is more confident about when predicting the activation of the transcription factors, especially for FHL1 besides the short segments of transition the posterior is always near 0 or 1. The activation profile of FHL1, especially fits very well to the experimental setup consisting of 3 phases with and

[6]ChIP-on-chip combines the methods of **ch**romatin **i**mmuno**p**recipitation and DNA microarrays for in vivo experiments of protein-DNA interaction. For more information see Aparicio et al. (2001).

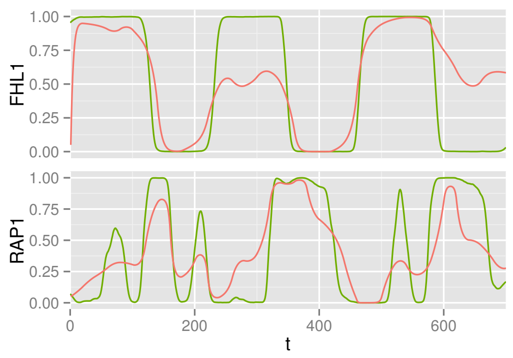


Fig. 4.9 Comparison of the posterior profile of TF activity obtained through our MCMC algorithm (green line) and the variational approximation of Opper and Sanguinetti (2010) (red line). The sampler used the maximum likelihood parameters from the variational approximation to make the comparison solely based on inference of the TF's profiles.

without the supply of glucose each. The longer phases of RAP1 activation in all 3 cases start just about or a little bit before FHL1 is deactivated.

4.3 Changepoint Process

There are many applications where the number of hidden states affecting the observed variables is not known beforehand. Furthermore, the assumption that we might revisit a former state might not be fulfilled because there are an enormous number of latent factors responsible for the outcome and even if an important parameter is reset to a former value it is almost impossible that the whole environment is in the same state as well.

Stock market data or in general financial data certainly fulfills these properties. Besides very rare events, e.g. the fake news story which let the stock markets tumble instantly in 2013, just to revert back minutes later when it was exposed (Moore and Roberts, 2013), the stock markets don't switch between a fixed set of states. We make minor changes to our model to get a general changepoint hidden process.

4.3.1 Model

In our previous switching model $\mu(t)$ was a continuous time process consisting of, possibly multiple, telegraph processes switching between two states. At each time t $\mu(t)$ was a binary vector which lead to a specific drift and diffusion term of the OU-process. Instead of a binary vector, in the general changepoint model $\mu(t)$ is a non-negative integer representing a specific set of parameters. This enables us to have an unlimited and variable number of possible

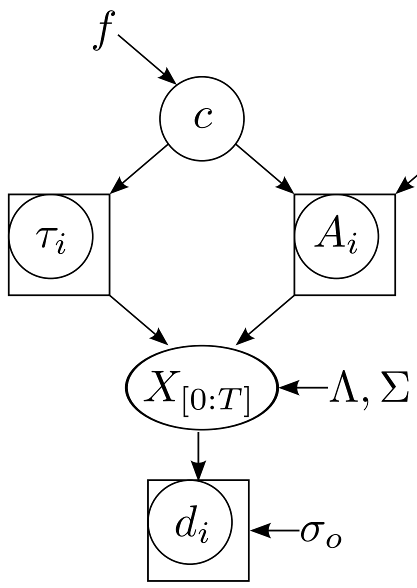


Fig. 4.10 The generative model of the Ornstein-Uhlenbeck process driven by a hidden changepoint process.

states. As we are changing the general concept of the hidden process, we reformulate the entire model.

We assume that c, the number of jumps of $\mu(t)$ in the time interval $[0:T]$, is Poisson distributed with parameter f

$$c \sim \text{Poisson}(fT), \tag{4.41}$$

with mean value fT. If we condition on the number of changepoints c their positions $\tau_1, \ldots, \tau_c \in [0:T]$ are independently and uniformly distributed random variables (Gelenbe, 1979):

$$\tau_i | c \sim U(0,T). \tag{4.42}$$

The model is visualized in figure 4.10.

We assume τ_i are sorted in ascending order so that they divide $[0:T]$ into $c+1$ segments with $[\tau_{i-1}:\tau_i]$ being the i-th segment, where we have defined $\tau_0 = 0$ and $\tau_{c+1} = T$ to simplify the notation. Altogether this is equivalent as defining the jump times $\tau_1, \ldots, \tau_c$ to come from a Poisson process with constant rate f in the time interval $[0:T]$ (Ross, 1983). The index of the Poisson process determines which parameter set is active at that time. That means for c jumps there have to be $c+1$ different parameter sets which we assume are all drawn from the same prior distribution $P(\theta)$. As before, we choose λ to be constant over time, therefore the OU process is defined by N SDEs of the form

$$dX = (A_{\mu(t)} - \lambda X)dt + \sigma_{\mu(t)} dW, \tag{4.43}$$

where we omitted the indices for the N dimensions for the sake of clarity.

Another way to view the model is to describe A and σ as a multidimensional jump process $\theta(t)$ where at each jump a new parameter vector is drawn from $P(\theta)$.

In this model we no longer have the possibility to have different priors for the different states, instead there is one overall prior for all the parameters. One disadvantage over the switching model is that we can no longer specify that a jump only has an effect on a subset of the dimensions of the OU process but the model can emulate this behavior by choosing parameter values similar to the ones before the jump for these dimensions.

Likelihood

The change of the hidden process has only a very minor effect on the likelihood $P(D|\mu_{0:T}, \theta)$. The calculation remains the same as described in section 4.1.1, but with $A_{\mu(t)}$ substituted for $A\mu(t) + b$ everywhere.

4.3.2 Sampler

The overall sampling algorithm stays the same as for the switching process (see section 4.1.2) but some changes in the details of the proposal actions need to be addressed.

Proposal Actions

For the changepoint model it is no longer necessary to allow for adding or removing two jumps at a time. For the switching case this was necessary because adding or removing only one jump changes a large part of the process and thus will result in low acceptance rates. When adding a jump to the changepoint process it is only changed between the added or removed jump time and the next or last jump time[7]. Our algorithm therefore only has three different actions remaining.

Shifting the time of a jump No changes are necessary for this action, the only difference is the new likelihood ratio as described in section 4.3.1.

[7]If we are representing the $\mu(t)$ process as a Poisson process all state indices after the added or removed jump would be changed to keep the state indices in ascending order but this does *not* change the parameter values after the affected segment, which influence the likelihood and therefore the acceptance probability.

Adding a jump A key difference to the switching process model is that adding a jump will generate a new set of parameters for the new segment. This means we need to sample a new parameter set for the segment from a proposal distribution and change the acceptance probability accordingly. The proposal distribution's support has to be at least the parameter prior's otherwise the action of adding a jump could not reverse all possible action of removing a jump.

An obvious choice would be to sample the parameters from the prior but this can only be effective if the prior is very narrow. Instead for A we draw the new value from the posterior of the parameters in the segment similar to what is done when resampling all the A parameters. For σ^2 we make a log random walk on the current value to get a new proposed one. We decide randomly if the new segment starts or ends with the added jump and get

$$\frac{P(\mu^*|\Theta)^*}{P(\mu|\Theta)}\frac{Q(\mu|\mu^*)}{Q(\mu^*|\mu)} = \frac{fP(A_{k+1},\sigma^2_{k+1})}{1}\frac{q_rT}{Q(A_{k+1},\sigma^2_{k+1})(c+1)q_a}, \tag{4.44}$$

with $Q(A_{k+1},\sigma^2_{k+1})$ being the proposal distribution we draw the parameters from and $\Theta = (\mathbf{A},\sigma^2,\lambda,f)$.

Removing a jump When removing a jump we decide randomly to either use the next segment's or the last segment's parameter values for the whole segment. The acceptance probability has to be changed to reflect the change in the action of adding a jump in order to guarantee reversibility:

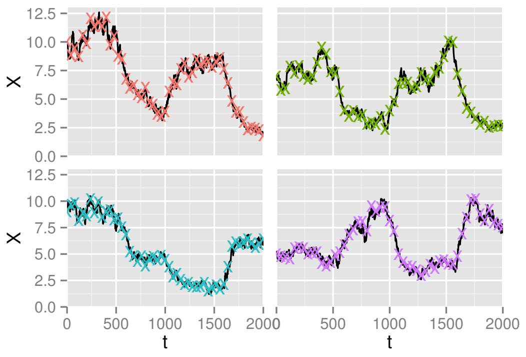
$$\frac{P(\mu^*|\Theta)}{P(\mu|\Theta)}\frac{Q(\mu|\mu^*)}{Q(\mu^*|\mu)} = \frac{1}{fP(A_i,\sigma^2_i)}\frac{Q(A_i,\sigma^2_i)cq_aq_n}{q_rT}. \tag{4.45}$$

4.3.3 Results

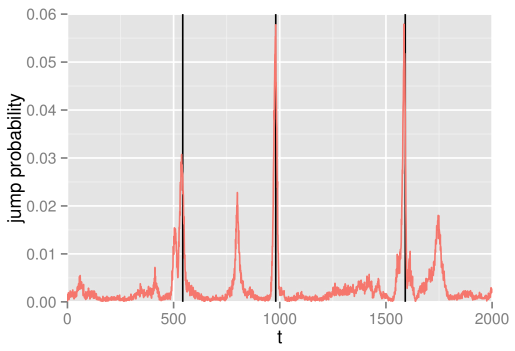
Before we apply the changepoint model to stock index data we first test it on a synthetic dataset.

Synthetic Data

We generated data from a four-dimensional OU process whose A and σ^2 parameters were controlled by a changepoint process $\mu(t)$. In figure 4.11a the observations are plotted for all four dimensions together with the true process $X(t)$. Figure 4.11b shows that the 3 most likely jump times fit the real jumps. As $\mu(t)$ now only represents a state index it is hard to interpret the posterior over its path as we did for the switching model. Instead we plot

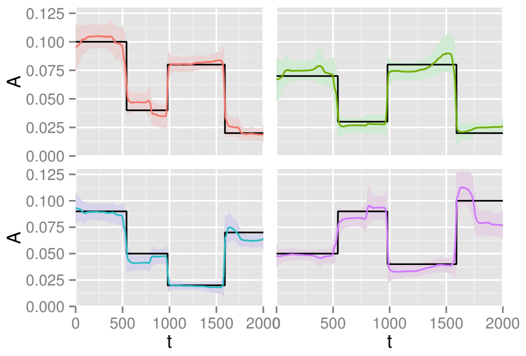


(a) Four-dimensional Ornstein-Uhlenbeck process with A and σ^2 controlled by a changepoint process. The true $X(t)$ process is drawn in black while the noisy observations are represented by colored crosses.

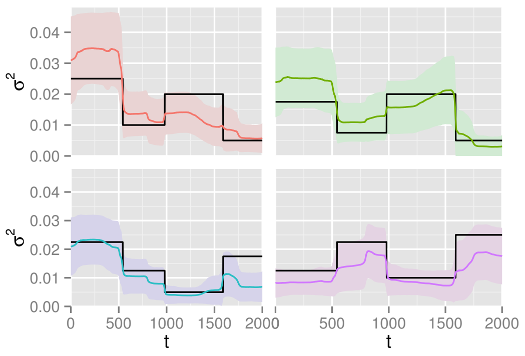


(b) Posterior probability of a jump happening in the time interval (here $\Delta t = 1$) vs. time drawn as a red line with the true jumps times represented by the vertical black lines.

Fig. 4.11 Synthetic changepoint data and jump posterior.



(a) Posterior of A over time for all four dimensions. The black line is the actual path of A. The colored line is the posterior mean with a 95% confidence interval around it.



(b) Posterior of σ^2 over time for all four dimensions. The black line is the actual path of σ^2. The colored line is the posterior mean with a 95% confidence interval around it.

Fig. 4.12 Parameter posteriors for synthetic changepoint data.

parameters	prior	hyper-parameters
f	fixed	$f = 0.1$
A	Gaussian	mean $= 4,300$, std. $= 5,000$
σ^2	Gaussian	mean $= 250,000$, std. $= 40,000$
λ	Gamma	shape $= 1.2$, scale $= 1.0$

Table 4.3 Parameters and prior distributions for the DAX dataset.

the posterior over the path of the parameters which are controlled by $\mu(t)$, namely $A(t)$ and $\sigma^2(t)$. The posteriors are plotted in figure 4.12 and while the posterior mean over $A(t)$ fits the true path very closely, inference for the system noise parameter σ^2 seems to be harder.

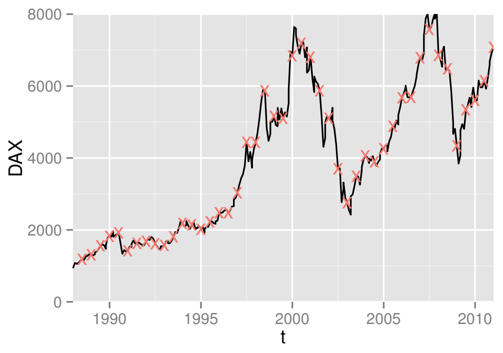
Stock Index Data

We use data taken from the German stock index (DAX) which is comprised of the 30 biggest publicly traded German companies. While the Ornstein-Uhlenbeck model with linear noise is usually not used to model stock prices directly [8] the addition of jumping parameters makes it very flexible and stock prices do exhibit mean reverting properties (Chiang et al., 1995; Lo and MacKinlay, 1988).

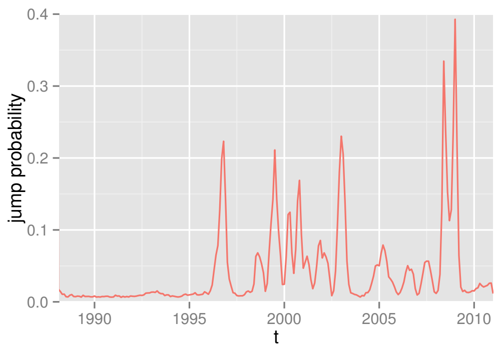
Figure 4.13a shows the data we used. While stock prices are available for a very high time resolution we only used quarterly from 1988 to 2011 for two reasons. Firstly, the lower number of observations allows faster inference and secondly we show that the algorithm is able to find changepoints even for low resolution data. We generated 510,000 Monte Carlo samples from which the first 10,000 were dropped as burn-in. For the parameters and priors we used see table 4.3.

On average 9.7 jumps were found. The probability of a jump occurring over time is plotted in figure 4.13b and it is visible that some jumps are very clearly timed. When we look at the posterior of the parameters over time (figure 4.14) we again see it is very distinct for A and a lot less informative for σ^2. Most interestingly, we can find historic events happening around the time A jumps, like the introduction of the German version of the NASDAQ index (Neuer Markt) in 1997, the bursting of the dot-com bubble in 2000 or the recent global financial crisis.

[8]OU processes whose diffusion is driven by a Levy process have been used (Jongbloed et al., 2005; Onalan, 2009) to model financial data and OU processes have been used to model the volatility of other stochastic process (Chronopoulou and Viens, 2012; Fouque et al., 2000).

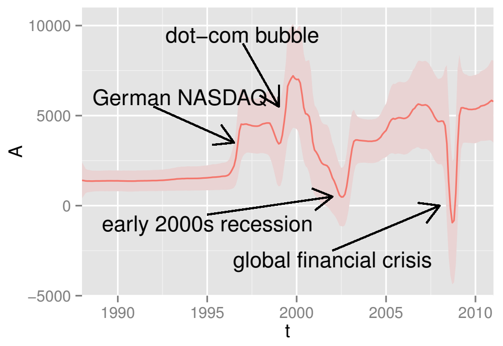


(a) Data points taken from the German stock index (DAX) from 1988 to 2011. The closing value of the stock index for every month is in black and the quarterly data used in the sampler is drawn as red crosses.

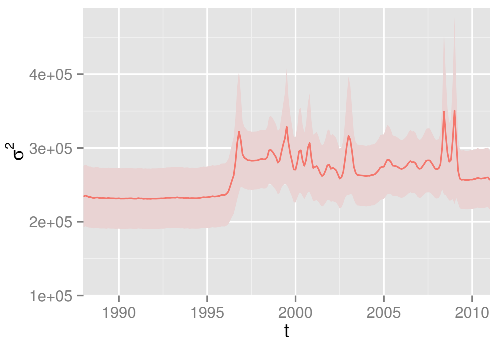


(b) Posterior probability of a jump over time for the German stock index (DAX) data.

Fig. 4.13 German stock index (DAX) data and jump probability.



(a) Posterior mean probability of A over time with two times the standard deviation as a confidence interval. Notable economic events are correlated with jumps in A are highlighted.



(b) Posterior mean probability of σ^2 over time with two times the standard deviation as a confidence interval.

Fig. 4.14 Posterior of the parameters over time for the German stock index (DAX) data.

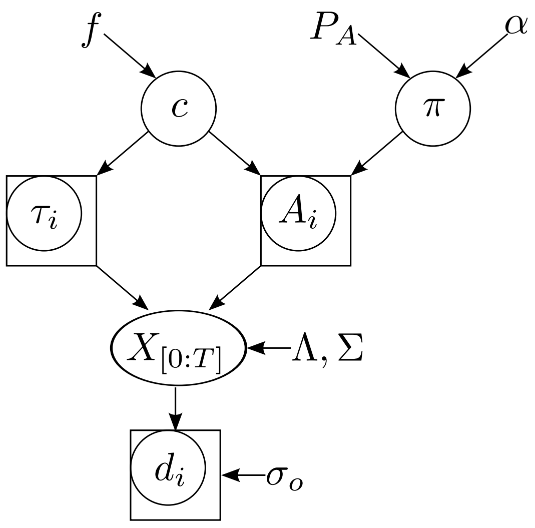


Fig. 4.15 The generative model of the Ornstein-Uhlenbeck process driven by a hidden Chinese restaurant process.

4.4 Chinese Restaurant Process

When changing our model from a fixed number of sets to a more flexible changepoint model, we lost the ability to have the hidden process revisit already used states. This made sense considering our application on financial data but in other fields it would be better if former states can be reused. Besides being a better model for some applications, like the gene expression data we examined in section 4.1.3 and 4.2.3, there are other benefits. With reusable states our model will have less parameters allowing faster inference. Additionally, less states mean we have more data per state, resulting in better estimates of the parameters.

We had these advantages in the original switching model but more often than not we do not know the exact number of hidden states beforehand but still want to have the benefits of reusable states. For these cases we formulated a model which combines both the switching and the changepoint model's advantages using a process usually called the Chinese restaurant process (Pitman and Picard, 2006, pp. 54 ff.).

4.4.1 Model

Our model still is described by a set of Ornstein-Uhlenbeck type SDEs. As in section 4.2 we let only the production rate A depend on time, therefore our system has the form

$$d\mathbf{X} = ((\mathbf{A}(t))_i - \Lambda)dt + \Sigma d\mathbf{W}, \tag{4.46}$$

with fixed decay and diffusion parameters

$$\Lambda = \operatorname{diag}(\lambda_1, \dots, \lambda_N)^\top, \tag{4.47}$$

$$\Sigma = \operatorname{diag}(\sigma_1, \dots, \sigma_N)^\top, \tag{4.48}$$

and time-dependent function

$$\mathbf{A}(t) = (A_1(t), \dots, A_N(t))^\top. \tag{4.49}$$

The times of the changepoints are still coming from a Poisson process with constant rate f but instead of drawing the parameter sets for each segment from the prior over the time dependent parameters $P_A(\cdot)$ itself they are now drawn from an unknown distribution π. We assume π comes from a Dirichlet process

$$\pi \sim DP(\alpha, P_A), \tag{4.50}$$

with concentration parameter α and the prior $P_A(\cdot)$ as its base distribution.

If we integrate out the unknown distribution π we get a Chinese restaurant process with the same parameters as the Dirichlet process (Teh, 2010) from which we can sample sequentially. Conditioned on the previous segments, the value of the $i+1$-th segment

$$A_{i+1} | A_1, \dots, A_i \sim \frac{1}{\alpha + i} \left(\alpha P_A + \sum_{l=1}^{i} \delta_{A_l} \right) \tag{4.51}$$

is either sampled from the base distribution $P_A(\cdot)$ with probability $\alpha/(\alpha+i)$ or a parameter set which was already used in a previous segment is reused. In the latter case the parameter set to use is selected with equal probability from all the segments which means the Chinese restaurant process leads to a "rich-get-richer" effect (Ghahramani, 2005). Altogether, this leads to the following prior probability over a path $A_{0:T}$

$$P(A_{(0:T)} | f, \alpha, P_A) \propto f^c e^{-fT} \alpha^s \frac{\prod_{j=1}^{s} \left(P_A(A_j)(\#_j - 1)! \right)}{\prod_{i=0}^{c} (\alpha + i)}, \tag{4.52}$$

where $_j$ is the number of times state j has been assigned to a segment.

See figure 4.15 for a graphical representation of the complete model.

While until now our hidden processes have fulfilled the Markov property it is important to note that the Chinese restaurant process is *not* a Markov process because the parameter value of a segment depends on the complete history of the process up to that point. The model also allows that a jump does not change the state. This has to be kept in mind when looking at the number of jumps in the posterior but in our experience such jumps are usually very rare unless a very high jump rate is chosen.

4.4.2 Sampler

As for the changepoint model our general sampling strategy does not need to be adjusted. The only part of the sampler we need to modify is the sampling of the time and type of jumps. By changing the existing proposal actions and introducing a new one we are able to get samples for the new model.

Sampling The Parameters

If a Gaussian is chosen as $P_A(\cdot)$ the posterior over the individual values of A is Gaussian as well and we directly sample from it, as described in section 4.1.2. For the λ and σ^2 parameters we again apply a Gaussian random walk on their log values.

Proposal Actions

In the changepoint model of section 4.3 it was clear that when a jump occurs a new set of parameters was drawn, which was completely independent of the former sets. The Chinese restaurant process provides the possibility that a new time interval reuses an existing parameter set. Therefore when adding or removing changepoints it is now important to distinguish between different cases. In order to get faster convergence we introduce the possibility to switch the state of an existing interval with either a new parameter set or an existing one.

Adding a jump When adding a jump it is now randomly decided if the new segment gets an already existing state assigned to it or if a new state is introduced. We could incorporate the α parameter in the probability of a new parameter set to be created but we decided to use a fixed probability $q_n = 0.1$ in our experiments if not otherwise specified. This has the reason that a prior based probability might be very small and it may take a long time until a new parameter set is proposed even though the data might be very well explained by it. The trade-off in this case is that a new set might be proposed and rejected very often but even if all these proposals are rejected they make up only 10% of the proposals which add a jump. If a new parameter set is proposed we use the posterior over A for this segment to draw a suitable parameter set. As before we decide with equal probability if we want to change the parameter value of the segment directly after or before the new jump.

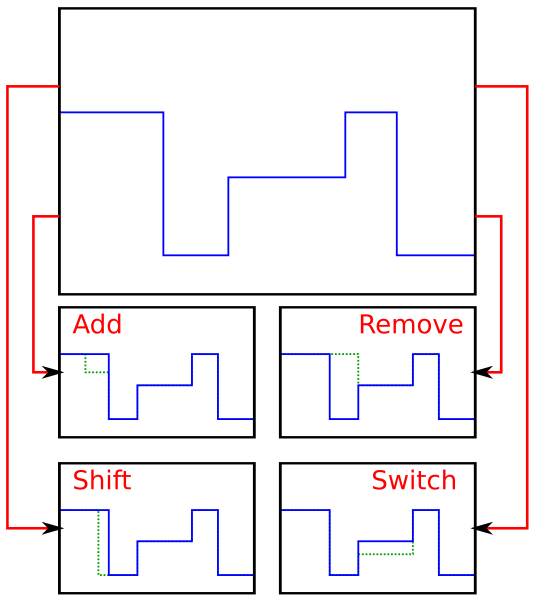


Fig. 4.16 The 4 actions to modify the hidden path. The old path is drawn in blue, while the modified part is drawn in green.

Removing a jump When a jump is removed we have to distinguish between the case that the number of states stays the same and the case that the last instance of an state has been removed. In contrast to the switching model this has an effect on the acceptance probability of the proposal because the dimensionality of the parameters change. As for the other models it is randomly decided if the process after or before the removed jump defines what parameters will be used in the joint segment.

Switching a state In theory, this action is not necessary to be able to reach all possible hidden process paths, because the same effect can be achieved by removing a jump and adding a new one. Unfortunately, this would only happen very rarely and lead to very slow convergence times. Instead we introduce a new action which randomly selects one of the jump process' segments with equal probability and changes what state it is associated with. We decide either to set the segment to an already existing state (with equal probability) or to draw a new state in the same way as we do when adding a jump.

Figure 4.16 shows the 4 actions' effects on a path.

Acceptance Probabilities

The acceptance probabilities resemble the changepoint process‘ but now the effects of the Chinese restaurant process have to be incorporated and we need to distinguish between different cases for all actions besides the shifting of jump times.

As before, the acceptance probability is

$$P_{MH} = \min\left(1, \frac{P(\mathbf{D}|A_{0:T}*,\Theta)}{P(\mathbf{D}|A_{0:T},\Theta)} \frac{Q(A_{0:T}|A^*_{0:T})}{Q(A^*_{0:T}|A_{0:T})} \frac{P(A^*_{0:T}|\Theta)}{P(A_{0:T}|\Theta)}\right), \tag{4.53}$$

with $\Theta = (\Lambda, \Sigma, \sigma_{obs}, f)$ and where $P(\mathbf{D}|A_{0:T},\Theta)$ and $P(\mathbf{D}|A^*_{0:T},\Theta)$ remain as described in section 4.1.1 just with $A(t)$ instead of $A\mu(t)+b$ everywhere. The prior over $A_{0:T}$ and the proposal probabilities Q are again different for the actions and we again define

$$\Psi = \frac{Q(A_{0:T}|A^*_{0:T})}{Q(A^*_{0:T}|A_{0:T})} \frac{P(A^*_{0:T}|\Theta)}{P(A_{0:T}|\Theta)}. \tag{4.54}$$

Shifting the time of a jump The acceptance probability for shifting a jump stays the same as in (4.26).

Adding a jump When adding a jump point we have to distinguish between adding a new and an existing set of parameters. In the first case the acceptance ratio becomes

$$\Psi = \frac{q_r T}{q_a q_n (c+1) Q(A_{k+1})} \frac{f \alpha P_A(A_{k+1})}{\alpha + c + 1}, \tag{4.55}$$

where k is the number of parameter sets and c the number of jumps *before* the action was applied. q_a and q_r are the probabilities to choose the action to add and remove a jump, respectively.

If on the other hand the parameter set for the new interval already exists the ratio is

$$\Psi = \frac{q_r T k}{q_a (1-q_n)(c+1)} \frac{f \#_i}{\alpha + c + 1}, \tag{4.56}$$

where $\#_i$ is the number of times the parameter set A_i is used *before* applying the action.

Removing a jump Removing a changepoint whose state was the last instance of its kind ($\#_i = 1$) leads to

$$\Psi = \frac{q_a q_n c Q(A_i)}{q_r T} \frac{\alpha + c}{f \alpha P_A(A_i)}. \tag{4.57}$$

When θ_i still occurs after removing the changepoint ($\#_i > 1$) the ratio becomes

$$\Psi = \frac{q_a (1-q_n) c}{q_r T k} \frac{\alpha c}{f(\#_i - 1)}. \tag{4.58}$$

Switching a state If we switch the current parameter set A_i of a segment to A_j there are four different cases to consider:

1. The old parameter set is still used ($\#_i > 1$) and the new parameter set is already active in another interval ($\#_j > 0$)
$$\Psi = \frac{\#_j}{\#_i - 1}. \tag{4.59}$$

2. The old parameter set is still used ($\#_i > 1$) and the new parameter set has not been in use ($\#_j = 0$)
$$\Psi = \frac{(1-q_n)}{q_n Q(A_j)(k+1)} \frac{\alpha P_A(A_j)}{\#_i - 1}. \tag{4.60}$$

3. The old parameter set vanishes ($\#_i = 1$) while the new parameter is already used ($\#_j > 0$)
$$\Psi = \frac{q_n Q(A_j)k}{(1-q_n)} \frac{\#_j}{\alpha P_A(A_i)}. \tag{4.61}$$

4. The old parameter set vanishes ($\#_i = 1$) and the new parameter set has not been in use ($\#_j = 0$)
$$\Psi = \frac{Q(A_i)}{Q(A_j)} \frac{P_A(A_j)}{P_A(A_i)}. \tag{4.62}$$

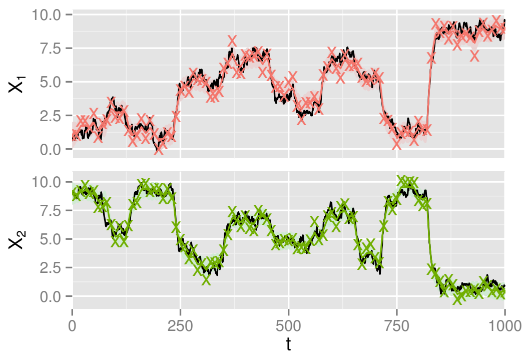
After an action the the parameter sets A_i are renumbered in ascending order from start so that we again have a legitimate path from our model.

4.4.3 Results

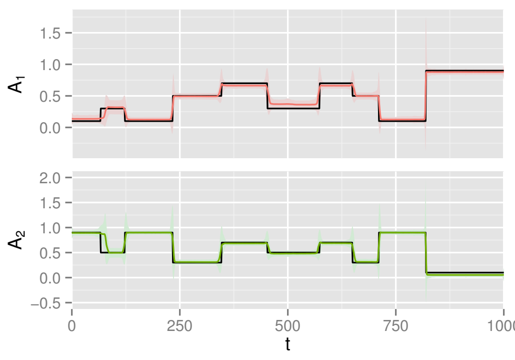
We first again have a look at the performance of our sampler on synthetic data to see that the estimated posterior distributions are reasonable in relation to the true hidden process. After this the yeast gene expression data which was used in section 4.2.3 is revisited to see if the more general CRP model can confirm the assumption about the number of states which had to be made for the switching model.

Synthetic Data

We generated a synthetic dataset of a two-dimensional OU process with a hidden CRP process defining the value of A over time. The CRP process has 10 segments which are assigned to 5 different states. As the base distribution P_A we used a Gaussian with mean and standard derivation 0.5. In figure 4.17a the observations along with the posterior over the $X(t)$ process are compared to the true $X(t)$ process. The sampler is able to restore the



(a) Noisy observations and posterior over $X(t)$. The actual path of $X(t)$ is drawn as a black line with colored crosses representing the noisy observations. The posterior mean over $X(t)$ is plotted as a colored line with a 95% confidence interval around it.



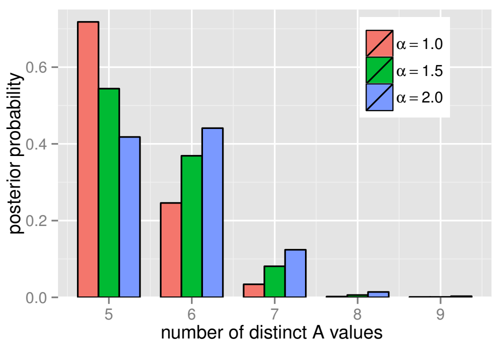
(b) Posterior over $A(t)$ for the synthetic CRP data. The true paths are the black lines, while the colored lines are the posterior mean with a 95% confidence interval around them.

Fig. 4.17 True paths and posterior densities over $X(t)$ and $A(t)$ for a synthetic two-dimensional OU model driven by a hidden Chinese restaurant process with 5 states.

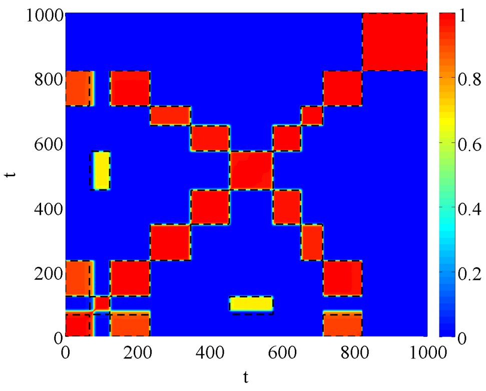
original path of X accurately and ignores outliers resulting from the observation noise. The path of A can be seen in figure 4.17b and the posterior fits the true path well. The rise of the confidence interval near the jumps follows from the sampler switching between allocating that region to the state before and after the jump. We are most interested if inference on the number of distinct states works well. Figure 4.18a shows that for all three values of α the sampler assumes that at least 5 states are necessary to explain the data. For the smaller two α values the correct number has the highest posterior probability while for $\alpha = 2.0$ 6 states are a little bit more likely. For 9 jumps the prior mean number states are 2.9, 3.5 and 4 for α equal to 1, 1.5 and 2, respectively.

The index of a state is not necessarily meaningful between samples with a different number of states but we can visualize how likely two points in time will be associated with the same state. This is shown as a heatmap in figure 4.18b and compared with the areas where the states are equal in the true hidden process. While the posterior mostly fits the true process here for the segment between $70 < t < 120$ and $450 < t < 570$ the posterior probability to be in the same state is only around 65% but this can be explained by the few and very noisy observations during these segments. These segments are also the primary reason for getting more than the true number of states in some of the posterior samples because each segment is assigned its own state when they are not assigned a shared one.

One of the advantages of the CRP model compared to a simple changepoint model as in section 4.3 should be that the parameter inference is improved, especially for small segments,

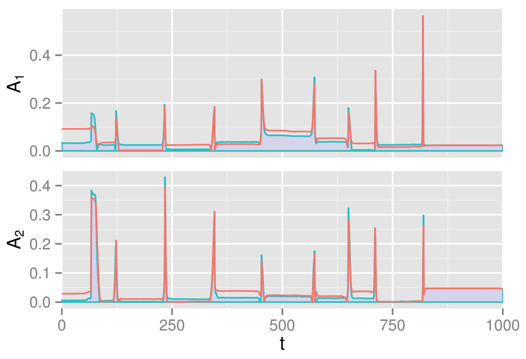


(a) Posterior probability over the number of states for the synthetic CRP dataset with 3 different values of α. The true number of states was 5.

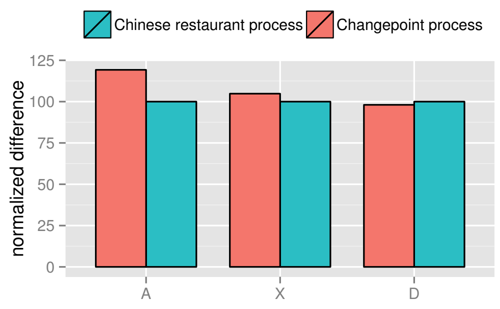


(b) Heatmap for the synthetic CRP dataset. The color represents the posterior probability that the state of the CRP is the same at both times. The perfect heatmap would have probability 1.0 (red) inside the black dashed lines and probability 0 (blue) everywhere else.

Fig. 4.18 Posterior over the number of states and the reuse of states.



(a) Difference over time between the true path of $A(t)$ and the posterior of the changepoint sampler (red line) and the CRP sampler (blue area) for the synthetic CRP dataset.



(b) Difference between the true path of $A(t)$ and $X(t)$ and the posterior mean integrated over time for the CRP and the changepoint sampler. Additionally, the sum of the difference between the observations and the posterior mean of $X(t)$ at the observation times. All values are normalized to 100 for the CRP. For the unnormalized values see table 4.4.

Fig. 4.19 Comparison of the CRP sampler and the changepoint sampler on the synthetic CRP dataset. For the CRP sampler $\alpha = 1.5$ was used.

parameter	sampler	difference
$\mathbf{A}(t)$	CRP	62.7
	CP	74.8
$\mathbf{X}(t)$	CRP	710
	CP	744
$\mathbf{D}$	CRP	56.3
	CP	55.2

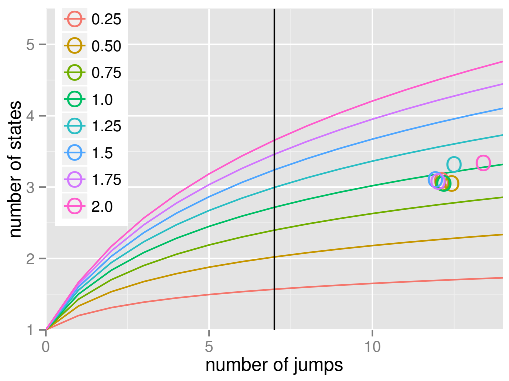
Table 4.4 Comparison of the differences to the true values for toy data generated from the CRP model when using the sampler for the CRP and CP model. The differences are integrated over time for $\mathbf{A}(t)$ and $\mathbf{X}(t)$ and calculated as the difference between the observations and $\mathbf{X}(t)$ at the observation times for $\mathbf{D}$.

as it uses the observations from all segments assigned to a state for inference instead of inferring the parameters for each segment on their own. For this purpose we used the general changepoint model where the unknown distribution π from our CRP model is replaced by the base distribution P_A and applied our sampler to it. In figure 4.19a the difference between the posterior mean of the changepoint and the CRP model to the true path of A is shown. As expected the CRP model gives superior results especially for short segments whose state is used multiple times, as can be seen between $650 < t < 710$. Figure 4.19b and table 4.4 compare the errors for $A(t)$ and $X(t)$ integrated over time and the difference between the posterior mean and the observations for both models. A is approximated around 15% better with the CRP model and X about 5%. Not surprisingly the simplified changepoint model has a slightly better fit to the data because it is able to change its parameters to best fit the data for every segment on its own.

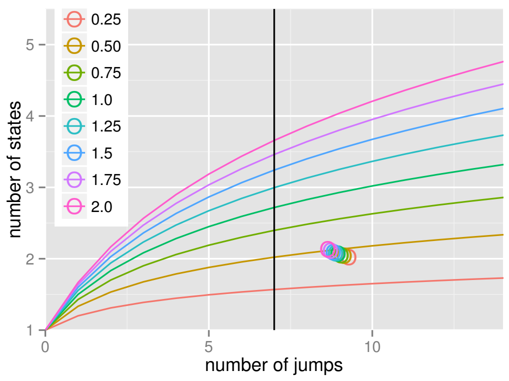
Besides these results the CRP model has a computational advantage as well. The time to get half a million samples was roughly half as long compared to the changepoint model. This is mainly because the CRP model has fewer distinct states which need to be updated in every run.

The Yeast Gene Expression Revisited

In section 4.2.3 we used gene expression data of yeast cells taken from Tu et al. (2005) and tried to infer the activity of two transcription factors which are known to control the 10 genes which were considered. For the switching model we used this knowledge when defining that two binary telegraph processes, representing the transcription factors *FHL1* and *RAP1*, are responsible for the changes in the expression levels of the transcription factors over time. By applying our new Chinese restaurant process model on the same data we hope that the



(a) Comparison of the prior and posterior mean for the yeast data showing the robustness of the results for different values of α. The colored lines show the prior mean over the number of jumps while the circles are the posterior mean number of states over the posterior mean number of jumps. The vertical line is the prior mean number of jumps resulting from $f = 0.01$ as the exponential jump rate.



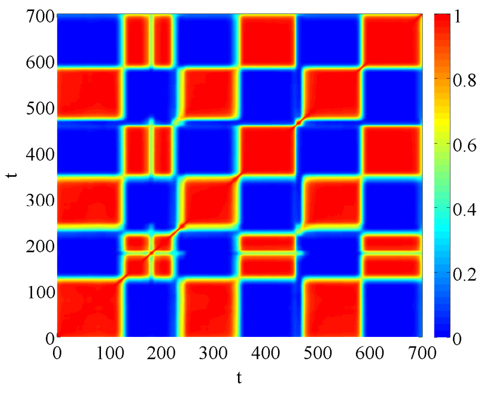
(b) Comparison of the prior and posterior mean for a subset of the yeast data with only 3 genes, which are known to be only regulated by the *FHL1* transcription factor. The lines are the prior mean number of states over the number of jumps and the circles represent the posterior mean number of states over the posterior mean number of jumps. The prior mean number of jumps resulting from the jump rate being $f = 0.01$ is shown as a vertical line.

Fig. 4.20 Robustness of the yeast cell data results for different values of α.

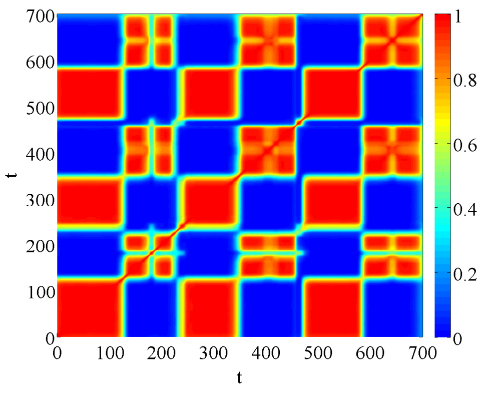
posterior estimate will point to two transcription factors with two states (i.e. 4 states overall) controlling the 10 genes we examined.

As the data is averaged over multiple cells, we can set the system noise variance to zero and we use the same values for the degradation rate λ_i and observation noise variance σ^2_{obs} as in Opper and Sanguinetti (2010). For the base distribution P_A we chose a Gaussian with zero mean and a standard deviation of 0.25.

We generated 1 million samples for 8 different values of α and this took about half an hour on an Intel Xeon CPU with 2.40 GHz. In figure 4.20a it can be seen that the results are very stable over the different values of α. For $\alpha = 1.0$ the model predicted with $\approx 96\%$ certainty that there are 3 distinct states in the data while 4 or more states were predicted in the rest of the samples. If we are assuming binary states for the transcription factors, as we did in the switching model, this would indicate two transcription factors controlling the expression levels of the 10 genes in the data. This fits with the biological explanation of Opper and Sanguinetti (2010) that the 10 genes are controlled by the transcription factors *FHL1* and *RAP1*. To test our model further we used it on data from 3 of the 10 genes which are known to be only regulated by *FHL1*. As can be seen in 4.20b this leads to the posterior mean number of states now being almost exactly 2 as would be expected for only one transcription factor being involved.



(a) Same-state heatmap for a low value of $\alpha = 1$ resulting in fewer distinct states on average.



(b) Same-state heatmap for a high value of $\alpha = 2$ resulting in more distinct states on average.

Fig. 4.21 Heatmaps for the full yeast dataset (10 genes) with the color representing the probability of the hidden process at the two times (x and y coordinate) being in the same state.

As for the synthetic data we can visualize how the states are reused over time in a heatmap. In figure 4.21 this is shown for $\alpha = 1.0$ and $\alpha = 2.0$, showing that the higher value of α leads to a more detailed structure and clear representation of the periodic nature of the experiment[9].

[9]The yeast cells were starved and then given sugar and this process was repeated 3 times over the whole experiment.

Chapter 5

Applications using the Poisson Process

While homogeneous Poisson processes are very restrictive and seldom useful on complex data sets, inhomogeneous Poisson processes (i.e. a Poisson process whose rate changes over time) allow to describe a variety of event-time data. A popular variant of an inhomogeneous Poisson process is the Markov modulated Poisson process (MMPP). An MMPP is a Poisson process whose rate depends on a hidden Markov jump process (MJP). Bayesian inference for MMPPs has been done by an approximate Gibbs sampler (Scott, 1999; Scott and Smyth, 2003), which assumes that the jumps of the Markov process can only happen at event times, or by a Metropolis-Hastings random walk on the parameters, integrating out the hidden MJP (Kou et al., 2005). Other approaches for inference of general MJPs include uniformization (Rao and Teh, 2011). On the other hand, Fearnhead and Sherlock (2006) proposed an exact Gibbs sampler which alternates between sampling a path for the hidden MJP at the Poisson event times given the parameters and sampling the parameters given the current path of the MJP. Advantages of this approach are that it additionally delivers the posterior distribution over the Markov process and requires no tuning of random walk parameters. Its disadvantage is that its computational complexity scales linearly with the number of Poisson events, even if the actual number of jumps in the rate process is small. We assume that after modifying our sampler to work with Poisson process observations it will cope better with such data sets because the likelihood computation can be sped up enormously in comparison to the approach of Fearnhead and Sherlock (2006) and Sherlock (2006). The parameter sampling step on the other hand remains the same as in their work, i.e. all parameters can be sampled directly from their conditional posteriors. In section 3.1.2 we already established that we will only use data where the observations are assumed to be exact. In Sherlock (2006) two different observation models are described. Both bin the data and either the number of events in a bin is known or there is a binary indicator if events have happened inside the binned time. Sherlock (2006) describes the likelihood for these models and our algorithm can be adapted to them easily. The second half of this chapter uses the Chinese restaurant process from section 4.4 for Poisson data to create a more flexible model than the common MMPP. This part is largely based on Stimberg et al. (2014) and therefore represents joint work with Manfred Opper and Andreas Ruttor.

5.1 Switching Process

As for the Ornstein-Uhlenbeck process in section 4.1 we start with a model where the number of states is fixed. The model described here is a special case of a popular model Markov modulated Poisson process (MMPP) (Fischer and Meier-Hellstern, 1993; Neuts, 1979). There have been multiple approaches to Bayesian inference for this model, most similar to ours are the exact Gibbs sampler of Fearnhead and Sherlock (2006) and Sherlock (2006) as well as the MCMC algorithm by Rao and Teh (2013) which uses uniformization.

5.1.1 Model

We observe event times $d_1,\ldots,d_n$ in ascending order which we assume are coming from a Poisson process. The rate $\lambda(t)$ of the Poisson process jumps at specific points in time $\tau_1,\ldots,\tau_c$ and remains constant between these jumps. In our model we assume that the rate can only jump between two rates: λ_0 and λ_1. As before the rate to jump from state 0 to state 1 is f_+ and the rate to jump from 1 to 0 is f_-. For a detailed formulation of a more general MMPP with an arbitrary, but fixed, number of states see Sherlock (2006). For the two state model the rate at time t is defined as

$$\lambda(t) = \mu(t)\lambda_1 + (1-\mu(t))\lambda_0. \tag{5.1}$$

The prior probability over a path of the switching process is the same as in (4.4) from section 4.1.1. Introducing ζ_i as the overall time spent in state i by μ, which is needed to compute the likelihood as well, we are able to simplify this and write the prior probability as:

$$P(\mu_{0:T}|f_0,f_1) \propto P(\mu(0)|f_0,f_1)\prod_{i=0,1} f_i^{c_i} e^{-\zeta_i f_i}, \tag{5.2}$$

where c_i is the number of jumps from state i to state $1-i$ and we set $f_0 = f_+$ and $f_1 = f_-$. To compare our algorithm to the exact Gibbs sampler of Sherlock (2006) we adopt their choice of the prior probability over the starting state of $\mu(t)$:

$$P(\mu(0)|f_0,f_1) = \frac{\mu(0)f_1 + (1-\mu(0))f_0}{f_0+f_1}. \tag{5.3}$$

Likelihood

As our observed process has changed, the likelihood of the observations given the parameter jump process needs to be updated. Luckily, a Markov modulated Poisson process leads to a very simple likelihood term. If we have s states then the likelihood becomes

$$P(D|\lambda_{0:T}) = \prod_{i=1}^{s} \lambda_i^{n_i} \exp(-\zeta_i \lambda_i), \tag{5.4}$$

where λ_i is the value of $\lambda_{0:T}$ when in state i and n_i is the number of Poisson events in our data while $\lambda_{0:T}$ is in state i. We will describe later how this likelihood can be calculated very efficiently.

5.1.2 Sampler

The structure of our sampler is very flexible and adjusting it to observations from a Poisson process instead of an OU does not require any changes of the general algorithm.

Sampling the Parameters

When looking at the likelihood (5.4) we see that with respect to the λ_i parameters it is proportional to a gamma distribution with parameters $n_i + 1$ and ζ_i. If we choose a conjugate gamma prior with hyper-parameters a_{λ_i} and b_{λ_i} we get

$$\begin{aligned} P(\lambda_i|D, \mu_{0:T}) &\propto \text{Gamma}(\lambda_i; n_i + 1, 1/\zeta_i)\text{Gamma}(\lambda_i; a_{\lambda_i}, b_{\lambda_i}) \\ &\propto \text{Gamma}(\lambda_i; a_{\lambda_i} + n_i, b_{\lambda_i}/(1 + \zeta_i b_{\lambda_i})) \end{aligned} \tag{5.5}$$

The same would be true for the f_i parameters when looking at (5.2), as described in section 4.1.2, but in contrast to our OU model we now follow Fearnhead and Sherlock (2006) in choosing the prior probability over the first state (5.3) to depend on the f_i parameters. Even with a gamma prior $P(f_i) = \text{Gamma}(f_i; a_{f_i}, b_{f_i})$ the posterior is not gamma distributed anymore

$$\begin{aligned} P(f_i|D, \mu_{0:T}) &\propto P(\mu(0)|f_0, f_1)\text{Gamma}(f_i; c_i + 1, \zeta_i)\text{Gamma}(f_i; a_{f_i}, b_{f_i}) \\ &\propto P(\mu(0)|f_0, f_1)\text{Gamma}(f_i; a_{f_i} + c_i, b_{f_i}/(1 + \zeta_i b_{f_i})) \end{aligned} \tag{5.6}$$

As proposed by Fearnhead and Sherlock (2006) we use a rejection sampler with $\text{Gamma}(a_{f_i} + c_i, b_{f_i} + \zeta_i)$ as a proposal distribution and the new values f_0^*, f_1^* are accepted with probability $P(\mu(0)|f_0^*, f_1^*)$.

Sampling the Jump Process

The set of proposal actions we use is the same as for the switching OU model in section 4.1.2, i.e. *shift*, *add*, *remove*, *add two* and *remove two* jumps. After the proposal path is generated it is accepted with the Metropolis-Hastings acceptance probability

$$P_{MH} = \min\left(1, \frac{P(\mu^*_{0:T}|D,\Theta)}{P(\mu_{0:T}|D,\Theta)} \frac{Q(\mu_{0:T}|\mu^*_{0:T})}{Q(\mu^*_{0:T}|\mu_{0:T})}\right), \tag{5.7}$$

where the posterior ratio follows from (5.2) and (5.4):

$$\begin{aligned}\frac{P(\mu^*_{0:T}|D,\Theta)}{P(\mu_{0:T}|D,\Theta)} &= \frac{P(\mu^*_{0:T}|f_0,f_1)P(D|\mu^*_{0:T},\lambda_0,\lambda_1)}{P(\mu_{0:T}|f_0,f_1)P(D|\mu_{0:T},\lambda_0,\lambda_1)}\\ &= \frac{P(\mu^*(0))}{P(\mu(0))} \prod_{i=0,1} f_i^{c_i^*-c_i} \lambda_i^{n_i^*-n_i} e^{-(f_i+\lambda_i)(\tau_i^*-\tau_i)}\end{aligned}$$

and with $\Theta = \{f_0, f_1, \lambda_0, \lambda_1\}$.

The posterior ratio can be calculated very efficiently because we only need to know how many Poisson events occur during the segments of $\mu_{0:T}*$ and $\mu_{0:T}$, how often the process changes state and how much time it spends in each state. In order to avoid iterating over all the data for each proposal, we compute the index of the next event in the data for a fine time grid before the sampler starts. This ensures that the computational time is most likely linear in the number of jumps while the one-time costs for calculating the grid are neglectable. Additionally, we only need to compute the likelihood ratio over segments which changed in the proposal because the unchanged parts cancel each other out. For details about how the calculation of the likelihood is implemented see section B.2 of the appendix.

5.1.3 Comparison to exact Gibbs sampler

In Sherlock (2006) the exact Gibbs sampler of Fearnhead and Sherlock (2006) was compared to a number of Metropolis-Hastings random walk algorithms which work directly on the parameters (as e.g. Kou et al., 2005, does). The exact Gibbs sampler performed better or comparable for a range of data sets, therefore we use it as the only comparison for our algorithm's performance. As mentioned before, our sampler is very fast even for a large dataset when the number of jumps in the Poisson rate is comparable small. We expect our sampler to outperform the exact Gibbs sampler for datasets with these properties and will verify this claim using synthetic datasets. Our work was done in parallel to Rao and Teh (2013) who did a similar comparison showing that the exact Gibbs sampler looses efficiency

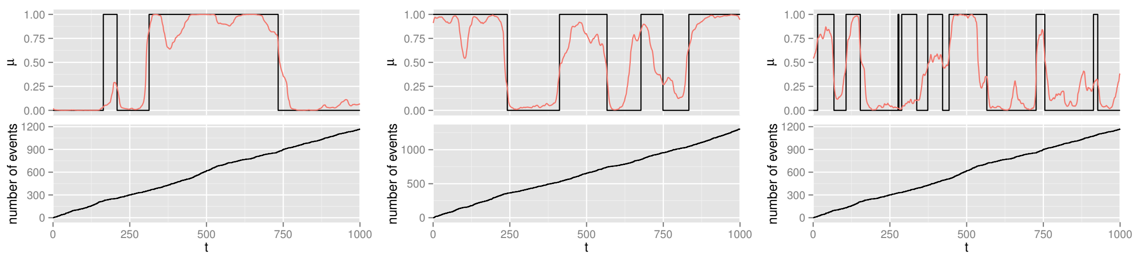


Fig. 5.1 Poisson event data (black line, bottom), true MJP path (black line, top) and MJP posterior (solid red) for 3 data sets with f_i = 0.005 (left), 0.01 (middle) and 0.02 (right). The Poisson rates of the observed process were $\lambda_0 = 1.0$ and $\lambda_0 = 1.5$ for all 3 sets.

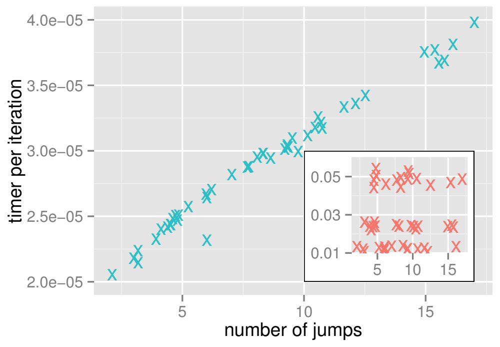
when the number of data points grows compared to the number of jumps in the hidden process.

Fearnhead and Sherlock (2006) sample from the true posterior $P(\mu_{0:T}|D,\Lambda)$ by applying the forward-backward algorithm of Baum et al. (1970) in continuous time. They start by sampling the state of the MJP at each of the Poisson events. Then for each interval between the events, the hidden process is sampled given the start and the end state. Naturally, this approach leads to high computational costs linear in the number of events. As described in section 5.1.2 the computational costs of our algorithm are linear in the number of jumps of the MJP. This would suggest that our algorithm performs especially well for $n \gg c$, where $n = n_0 + n_1$ is the number of total Poisson events and $c = c_0 + c_1$ is the number of jumps in the MJP.

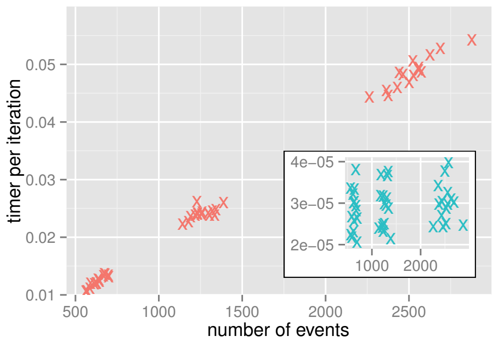
To test this we generated 45 synthetic datasets. We used 3 different sets for the Poisson rates of the observed process ($\lambda_0 = 0.5, \lambda_1 = 0.75$, $\lambda_0 = 1.0, \lambda_1 = 1.5$ and $\lambda_0 = 2.0, \lambda_1 = 3.0$) and 3 different sets for the jump rates of the hidden process ($f_0 = f_1 = 0.005, f_0 = f_1 = 0.01$, $f_0 = f_1 = 0.02$). For each of the 9 possible parameter combinations the hidden MJP and the MMPPs were then sampled from the prior. Figure 5.1 shows the data, the hidden MJP and the posterior over the MJP for three of these data sets.

To achieve identifiably we ensured that $\lambda_1 \geq \lambda_0$ by switching the parameters and the jump process if necessary. For our sampler this was done during the simulation while for the exact Gibbs sampler the parameters where switched, if necessary, after the simulation.

For the exact Gibbs sampler we used the program written by Chris Sherlock which is implemented in C. Our random walk sampler was written in C++ and for all simulations both programs were run on an Intel Xeon CPU with 2.40 GHz. Figure 5.2a shows that, as expected, the computational costs of our algorithm are linear in the number of jumps in the posterior over the path of the MJP. For the exact Gibbs sampler such a correlation cannot be



(a) Time per iteration over posterior mean jumps in the hidden Markov process, for the exact Gibbs sampler (red crosses) and the random walk sampler (blue crosses).



(b) Time per iteration over the number of Poisson events in the data, for the exact Gibbs sampler (red crosses) and the random walk sampler (blue crosses).

Fig. 5.2 Comparison of the time per iteration between the exact Gibbs sampler and our random walk algorithm.

observed. The number of Poisson events seems to have a similar effect on the exact Gibbs sampler, but not the random walk sampler (see figure 5.2b).

It is clear that our sampler is significantly faster per iteration (≈ 1020 times faster for $f_i = 0.005$, ≈ 860 times faster for $f_i = 0.01$ and ≈ 700 times faster for $f_i = 0.02$, averaged over 5 datasets), because a complete Gibbs update is slower than computing a random-walk proposal and the Metropolis-Hastings acceptance ratio. However, the samples produced by our method are more highly correlated than those of the exact Gibbs sampler. To assess this we computed the integrated autocorrelation time, as described in section 2.1.3, for the samples of each of the 4 parameters. The integrated autocorrelation time is a measure of inefficiency for Monte Carlo samples and can be interpreted as the number of correlated samples that correspond to one uncorrelated sample.

The autocorrelation functions of the four parameters for one of the datasets are depicted in figure 5.3. As predicted, the random walk sampler generates considerably more correlated samples than the exact Gibbs sampler. This impression is confirmed when we compare the integrated autocorrelation times in table 5.1. For the exact Gibbs sampler the integrated autocorrelation times are 8 to 135 times lower than for our sampler. We follow Sherlock (2006) and multiply the integrated autocorrelation time with the time it takes to compute one sample as a performance measure. The product can be considered as the time it takes to produce one uncorrelated sample.

The comparison for this value are shown in figure 5.4. For $f_i = 0.005$ our sampler performed around 33 times better than the exact Gibbs sample when averaged over all 4

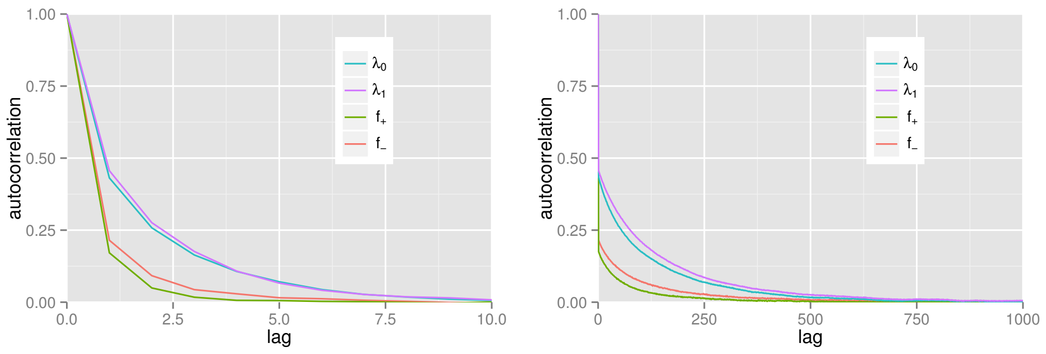


Fig. 5.3 Autocorrelation of the posterior samples for the four parameters from one of the data sets with $f_i = 0.005$, $\lambda_0 = 1.0$ and $\lambda_1 = 1.5$. The exact Gibbs sampler's results are on the left and the random walk sampler's results are on the right.

Parameter values				IAT RW				IAT GI			
f_+	f_-	λ_0	λ_1	f_+	f_-	λ_0	λ_1	f_+	f_-	λ_0	λ_1
0.005	0.005	0.5	0.75	50.0	40.5	47.5	39.0	2.34	2.47	4.05	4.81
0.005	0.005	1.0	1.5	92.8	76.6	99.0	103	1.84	3.31	6.91	3.70
0.005	0.005	2.0	2.0	44.7	40.6	95.0	90.7	1.37	1.33	1.71	1.37
0.01	0.01	0.5	0.75	169	168	228	169	4.35	4.10	7.88	5.69
0.01	0.01	1.0	1.5	153	153	247	182	2.92	2.53	5.01	4.30
0.01	0.01	2.0	2.0	207	241	262	199	1.92	1.79	2.30	2.01
0.02	0.02	0.5	0.75	754	717	837	774	8.55	8.80	12.7	14.7
0.02	0.02	1.0	1.5	417	408	405	340	5.67	4.75	5.99	6.12
0.02	0.02	2.0	2.0	439	297	499	335	3.88	3.40	4.20	4.36

Table 5.1 Integrated autocorrelation time (IAT) for the samples from our random walk sampler (RW) and the exact Gibbs sampler of Fearnhead and Sherlock (2006) (GI). The values were averaged over 5 datasets for each parameter combination.

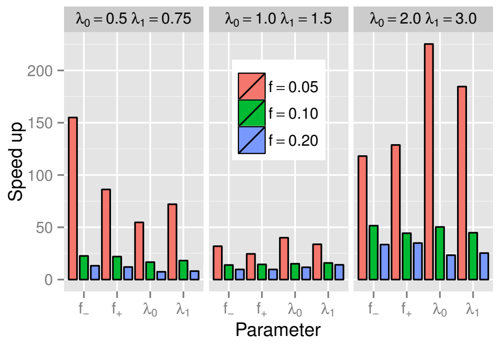


Fig. 5.4 The time needed to compute one uncorrelated sample of the parameters with the exact Gibbs sampler divided by the same value for the random walk Sampler. The bars are average values over 5 datasets for each parameter configuration.

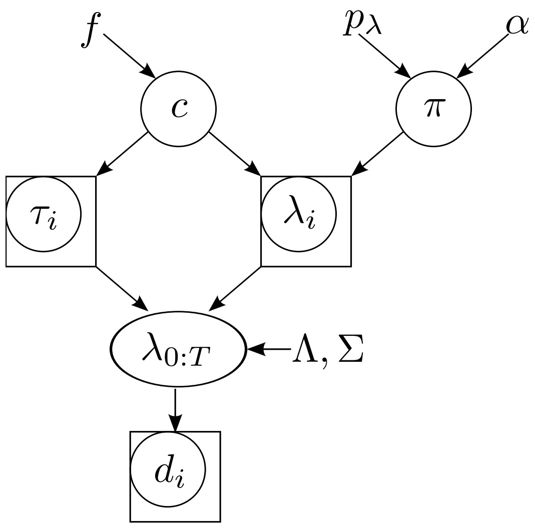


Fig. 5.5 The generative model of the Poisson process driven by a hidden Chinese restaurant process.

parameters. The advantage declines for higher values of f_i but at $f_i = 0.02$ it is still 11 times better.

Although we apply a random walk sampler the need for tuning is minimal. The only parameters affecting the sampler are the distribution over the possible actions to generate a new proposal path and the variance of the Gaussian used to shift the time of a jump. The former should be fairly independent of the data and the latter can be chosen by looking at the time span of the data and a rough estimate of the number of jumps in the MJP. We chose both values by hand but combining our algorithm with an adaptive MCMC approach might further enhance the efficiency of our sampler.

5.2 Chinese Restaurant Process

As for the Ornstein-Uhlenbeck process in section 4.4, we now change the model of the hidden process from a Markov jump process with a fixed number of states to a Chinese restaurant process where the number of states is estimated as part of the Bayesian inference.

5.2.1 Model

The model is very similar to the one in section 4.4, with the differences that we only have one parameter for the observed process (the Poisson rate λ) and the observations are without noise. The model is visualized in figure 5.5 and we quickly recap it here: The hidden process is constructed from drawing c jump times from a Poisson process with constant rate f. The rate of the observed process, $\lambda_{0:T}$, is separated into $c+1$ parts by this and during each of these segments it stays constant in one state λ_i. The unknown probability π from which we draw the λ_i value for each segment comes from a Dirichlet process with concentration parameter α and base distribution P_λ. Integrating out π gives a Chinese restaurant process with the same parameters. This means that after each jump the value of λ to be used in the following segment is either drawn from the base distribution (with probability $\alpha/(\alpha+i)$, where i is the number of segments until the jump) or it is drawn from the λ values of the previous segments with equal probability.

The prior probability over a path $\lambda_{0:T}$ therefore is

$$P(\lambda_{(0:T)}|f,\alpha,P_\lambda) \propto f^c e^{-fT} \alpha^s \frac{\prod_{j=1}^{s} \left(P_\lambda(\lambda_j)(\#_j - 1)!\right)}{\prod_{i=0}^{c}(\alpha+i)}, \tag{5.8}$$

and the likelihood of the data given a path of $\lambda_{0:T}$ remains as in (5.4):

$$P(D|\lambda_{0:T}) = \prod_{i=1}^{s} \lambda_i^{n_i} \exp(-\zeta_i \lambda_i), \tag{5.9}$$

but now the number of states s varies between samples.

5.2.2 Sampler

The sampler for this model is a mixture of the sampler applied to the Ornstein-Uhlenbeck model with a hidden Chinese restaurant process, described in section 4.4.2, and the sampler used for the MMPP model from section 5.1.2.

Sampling the Parameters

The parameter sampling follows the same strategy as for the MMPP model: We assume Gamma priors over the λ_i with shape parameter a_λ and scale parameter b_λ and this leads to gamma posteriors similar to (5.5)

$$\begin{aligned} P(\lambda_i|D,\mu_{0:T}) &\propto \text{Gamma}(\lambda_i; n_i+1, 1/\zeta_i)\text{Gamma}(\lambda_i; a_\lambda, b_\lambda) \\ &\propto \text{Gamma}(\lambda_i; a_\lambda + n_i, b_\lambda/(1+\zeta_i b_\lambda)) \end{aligned} \tag{5.10}$$

from which we can directly sample. In contrast to the MMPP model, the state of the first segment does not depend on the jump rate f. Because of this if we assume a conjugate gamma prior over the jump rate

$$P(f) = \text{Gamma}(f, a_f, b_f), \tag{5.11}$$

the posterior is again gamma distributed, as was the case for the OU models:

$$\begin{aligned} P(f|D,\lambda_{0:T}) &\propto \text{Gamma}(f_i;c_i+1,\zeta_i)\text{Gamma}(f_i;a_f,b_f) \\ &\propto \text{Gamma}(f_i;a_f+c_i,b_f/(1+\zeta_i b_f)). \end{aligned} \tag{5.12}$$

Sampling the Jump Times

As in section 4.4.2 we use the 4 actions to *shift* the time of a jump, *add* a jump, *remove* a jump and *switch* the state of a segment. As our model is built around single-dimensional data, i.e. there is only one Poisson process with rate $\lambda(t)$ which generates the data, we can always order our states by ascending λ values. This ordering allows us to introduce two new actions: *Joining* two neighboring states and the reverse action of *dividing* a state into two. As was the case for the actions of adding and removing two jumps for the switching models, these new actions are not necessary for a valid sampler but they improve the mixing. In this case they allow the sampler to change the number of states more easily. With only the previous actions reducing the dimensionality of the state space could take a lot of steps if the states were used in many segments, because the state in each of these segments would need to be changed through individual actions. The data we studied in section 4.4 did not have states who were used as often as the data we will be looking at here, so the problem did not arise before. In the following all the proposal actions are described.

Shifting the time of a jump This action remains the same as before, we still draw from a truncated Gaussian with standard deviation σ_t.

Adding a jump The time of the new jump is still drawn uniformly from the observed time period and with probability q_n a new value of λ is added, otherwise one of the already used values is reused for the new segment.

Removing a jump One of the jumps is chosen with equal probability and removed.

Switching a state One of the segments is chosen at random and its state is either switched to an already used one or a new value of λ is drawn. As when adding a jump the latter option is chosen with probability q_n.

If adding a jump or switching the state of a segment creates a new state and a corresponding new value of λ, then it is drawn from the conditional density

$$\begin{aligned} P(\lambda^*_{s+1}|\mathbf{Y},\lambda_{(0:T)}) &\propto \mathrm{Gamma}(\lambda^*_{s+1};a,b)\,\mathrm{Gamma}(\lambda^*_{s+1};n_{s+1}+1,1/\zeta_{s+1}) \\ &\propto \mathrm{Gamma}\left(\lambda^*_{s+1};a+n_{s+1},b/(1+zeta_{s+1}b)\right). \end{aligned} \tag{5.13}$$

When the segment uses an already existing value of λ, the state to use is drawn from a discrete distribution whose probabilities are proportional to (5.13), but with n_{s+1} and ζ_{s+1} being the number of Poisson events and the time in the modified segment, respectively.

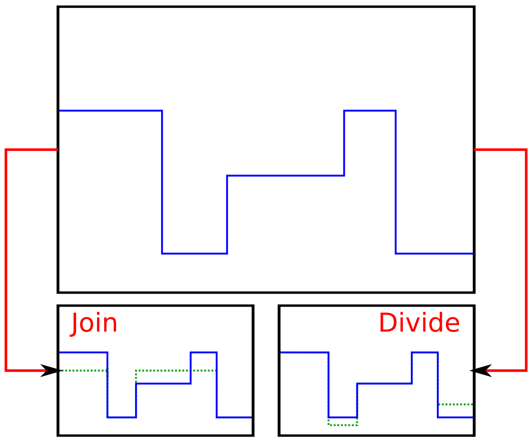


Fig. 5.6 The new actions of joining two states and dividing a state into two new states. The old path is drawn in blue, while the modified part is drawn in green.

Joining two states We draw two neighboring states (with respect to their λ values) at random and join them into a new state. All the segments which were assigned to the two states are assigned to the new state and the λ value of the joined state is the geometrical mean of the two states' λ values:

$$\lambda^*_j = \sqrt{\lambda_{i_1}\lambda_{i_2}}. \tag{5.14}$$

Dividing a state We randomly choose one of the states with at least two segments assigned to it. Then a small factor $\varepsilon > 1$ is drawn from a shifted exponential distribution and is used to create the new λ-values by multiplying and dividing the λ value of the chosen state with it. This leads to two new states with

$$\lambda_{j_1}^* = \lambda_i \varepsilon \tag{5.15}$$

$$\lambda_{j_2}^* = \frac{\lambda_i}{\varepsilon} \tag{5.16}$$

as their Poisson rates. The distribution ε is drawn from is truncated to assure that λ_{j_1} and λ_{j_2} are between the neighboring values of λ[1]. As a last step the segments used by the old state are randomly assigned to the new states with probability proportional to (5.13). If it happens, that by the last segment all other segments are assigned to only one of the new states then the last segment is assigned to the other state. It is important to note that this approach allows every possible assignment which uses both new states to be drawn and that there is exactly one way for each possible assignment to be drawn. This makes the Metropolis-Hastings acceptance probabilities for both the join and the divide action simple to calculate.

The first 4 actions remain the same as in figure 4.16 and the new join and divide actions are demonstrated in figure 5.6.

Acceptance Probabilities

After an action is chosen and the path $\lambda_{0:T}$ is modified to create the proposal $\lambda_{0:T}^*$ we accept it with probability

$$P_{\mathrm{MH}} = \min\left(1, \frac{P(D|\lambda_{(0:T)}^*)}{P(D|\lambda_{(0:T)})} \frac{Q(\lambda_{(0:T)}|\lambda_{(0:T)}^*)}{Q(\lambda_{(0:T)}^*|\lambda_{(0:T)})} \frac{P(\lambda_{(0:T)}^*|f,\alpha,P_\lambda)}{P(\lambda_{(0:T)}|f,\alpha,P_\lambda)}\right). \tag{5.17}$$

As before, the likelihood ratio is computed the same way independent of the action chosen to create the proposal path but the proposal and prior ratios

$$\Psi = \frac{Q(\lambda_{(0:T)}|\lambda_{(0:T)}^*)}{Q(\lambda_{(0:T)}^*|\lambda_{(0:T)})} \frac{P(\lambda_{(0:T)}^*|f,\alpha,P_\lambda)}{P(\lambda_{(0:T)}|f,\alpha,P_\lambda)} \tag{5.18}$$

depend on the chosen proposal action.

Shifting the time of a jump The acceptance ratio stays as in (4.26).

Adding a jump As for the CRP model with an OU process we need to distinguish two cases when adding a jump: Either we add a new value λ_{s+1} or we reuse an existing one. In

[1]If this was not assured, the join action could not reverse all possible outcomes of the divide action.

the first case the ratio is

$$\Psi = \frac{q_r T}{q_a q_n (c+1)\gamma^*(\lambda_{s+1})} \frac{f\alpha}{(\alpha+c+1)}, \tag{5.19}$$

with q_n as the probability to add a new value λ_{s+1} and $\gamma^*(\lambda_i)$ being the gamma density at λ_i with shape n_i^*+1 and inverse scale τ_i^*. This is proportional to the likelihood of the data given the parameter λ_i and the new path $\lambda^*_{(0:T)}$.

When we instead reuse an old state λ_i, we get

$$\Psi = \frac{q_r T}{q_a(1-q_n)(c+1)p^*_{\text{seg}}(i)} \frac{f(\#_i^*-1)}{(\alpha+c+1)}, \tag{5.20}$$

where $\#_i$ is the number of segments which use λ_i in the old path and $p^*_{\text{seg}}(i)$ denotes the probability to choose λ_i for the segment (see section B.1 for details).

Removing a jump When removing a jump, we either remove the last instance of a state i ($\#_i^* = 0$):

$$\Psi = \frac{q_a q_n c\gamma(\lambda_i)}{q_r T} \frac{(\alpha+c)}{f\alpha}, \tag{5.21}$$

or the state of the removed segment still exists in the proposed path: ($\#_i^* > 0$):

$$\Psi = \frac{q_a(1-q_n)c p_{\text{seg}}(i)}{q_r T} \frac{(\alpha+c)}{f(\#_i-1)}. \tag{5.22}$$

Switching the state of a segment The state assigned to a segment is switched from λ_i to λ_j and we differentiate between four cases:

1. λ_i is still used in the proposal ($\#_i^* > 0$) and λ_j is already assigned to another segment ($\#_j > 0$):

$$\Psi = \frac{p_{\text{seg}}(i)}{p^*_{\text{seg}}(j)} \frac{(\#_j^*-1)}{(\#_i-1)}. \tag{5.23}$$

2. λ_i is still used in the proposal ($\#_i^* > 0$) and we introduce a new value λ_j ($\#_j = 0$):

$$\Psi = \frac{(1-q_n)p_{\text{seg}}(i)}{q_n\gamma^*(\lambda_j)} \frac{\alpha}{(\#_i-1)}. \tag{5.24}$$

3. λ_i is no longer used in the proposal ($\#_i^* = 0$) and λ_j is already used in another segment ($\#_j > 0$):

$$\Psi = \frac{q_n \gamma^*(\lambda_i)}{(1-q_n)p^*_{\text{seg}}(j)} \frac{(\#_j^* - 1)}{\alpha}. \tag{5.25}$$

4. λ_i is no longer used in the proposal ($\#_i^* = 0$) and we introduce a new value λ_j ($\#_j = 0$):

$$\Psi = \gamma(\lambda_i)/\gamma^*(\lambda_j). \tag{5.26}$$

Joining two states Joining two neighboring states i_1 and i_2 into a new state j leads to

$$\Psi = \frac{q_d p_{\text{par}} p_\varepsilon(\varepsilon)(s-1)}{q_j s^*_{>1}} \frac{\varepsilon}{2\lambda_j} \frac{p_\lambda(\lambda_j^*)(\#_j^* - 1)!}{\alpha p_\lambda(\lambda_{i_1}) p_\lambda(\lambda_{i_2})(\#_{i_1} - 1)!(\#_{i_2} - 1)!}, \tag{5.27}$$

where q_d and q_j are the probabilities to choose the *divide* and *join* action, respectively, p_ε is the density from which the factor $\varepsilon > 1$ is drawn and p_{par} is the probability to assign the segments between states i_1 and i_2 like they are in the original path (see section B.1 for details). $s^*_{>1}$ is the number of states in the proposal with more than one segment assigned. $\varepsilon/(2\lambda_j)$ is a Jacobian factor resulting from using the distribution over ε rather than that of the λ-values when applying the join and divide actions.

Dividing a state For dividing state i into the new states j_1 and j_2 we get

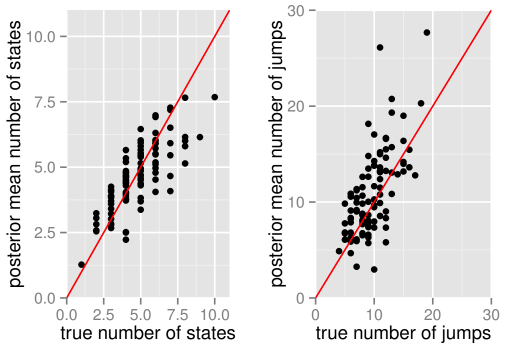
$$\Psi = \frac{q_j s_{>1}}{q_d p_{\text{par}} p_\varepsilon(\varepsilon) s} \frac{2\lambda_i}{\varepsilon} \frac{\alpha p_\lambda(\lambda_{j_1}^*) p_\lambda(\lambda_{j_2}^*)(\#_{j_1}^* - 1)!(\#_{j_2}^* - 1)!}{p_\lambda(\lambda_i)(\#_i - 1)!}. \tag{5.28}$$

5.2.3 Results

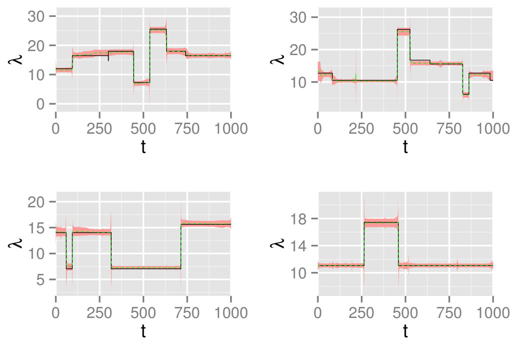
Before applying the sampler on neuronal spiking data from the primary visual cortex, we evaluate the sampler's performance on toy datasets.

Synthetic Data

We are especially interested in how good the Chinese restaurant process model is able to estimate the number of distinct states in the data. To investigate this we sampled 100 datasets from the prior with the jump rate and concentration parameter fixed to $f = 0.02$ and $\alpha = 3.0$, respectively. For each dataset 1.1 million samples were generated from which the first $100,000$ were dropped as burn-in. For each dataset this took around 25 seconds on average using an Intel Xeon CPU with 2.40 GHz without any parallelization.



(a) Posterior mean vs. true number of states (*left*) and jumps (*right*) for 100 data sets drawn from the prior. The red line shows the identity function which would represent the posterior mean perfectly predicting the true number of jumps and states in the data.



(b) Posterior of $\lambda(t)$ vs. t for the first 4 out of 100 toy data sets. The true path of $\lambda(t)$ is plotted as a black line, while the posterior mean is drawn as a dashed green lined surrounded by a 95% confidence interval.

Fig. 5.7 Results for Poisson toydata with a hidden Chinese restaurant process.

As we can see in figure 5.7a the true number of jumps and states in the data seems to be captured well by our posterior results, but when the number of distinct states becomes large the posterior mean underestimate the true value. This is not surprising, because the more jumps and states there are the more likely it is that the states are only active for a short time or that states have very similar λ values associated with them. The values chosen for f and α make the algorithm expect a smaller number of jumps and states and if there is not enough (or even none) data to support a larger number of states then the posterior will not represent it. For a diffusion process like the OU process, we would expect to still find these states when we have denser observations. For point processes this would only be possible if we observe the system over a longer time period and the state is revisited because the number of observations is based on the rate of the process.

To show that the posterior is able to reconstruct the hidden path $\lambda_{0:T}$ figure 5.7b compares it with the true path for 4 of the 100 datasets.

For the experiments the sampler used the true value of $\alpha = 3$ but for real datasets α has to be set manually and we often might only have a general idea of how many states we will be expecting. To ensure that the posterior results are robust we let the sampler run with 100 different α values (from 0.1 to 10 in steps of 0.1) for the first 4 datasets. As can be seen in figure 5.8 the model predicts a similar number of states even when the α parameter is very different from the true value. For all 4 datasets and over the whole range of α values the absolute difference between the posterior mean number of states and the true number is only larger than 1 for the first dataset and only for very low values of α. These results indicate

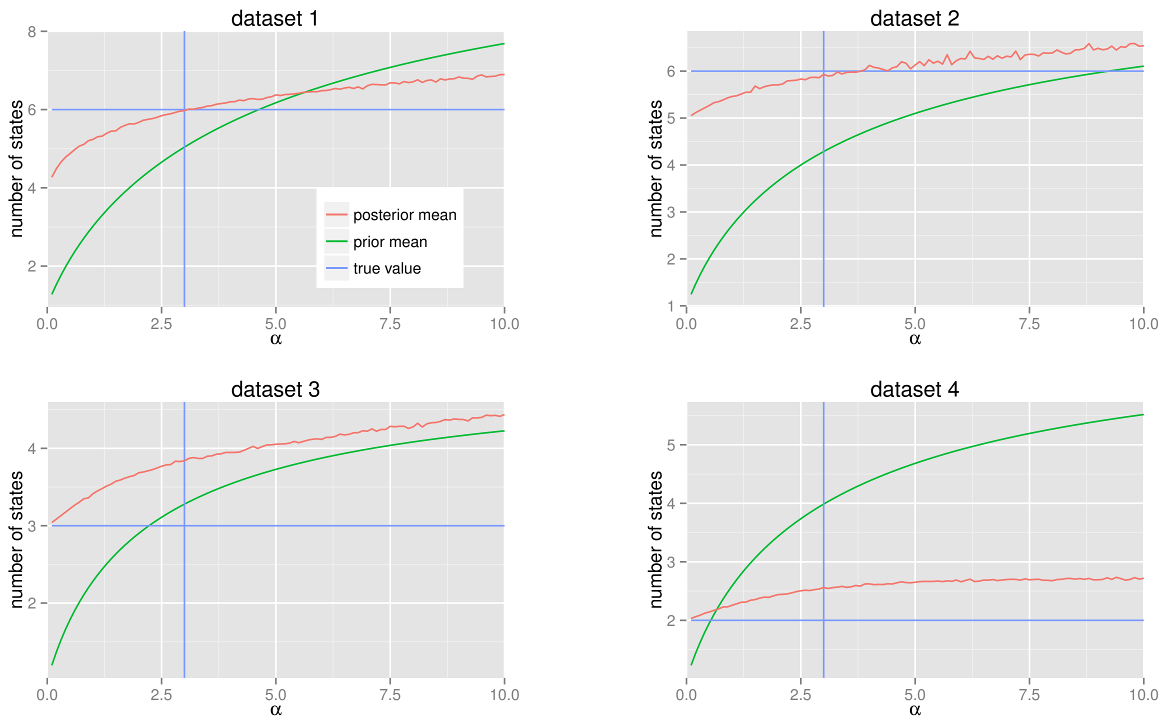


Fig. 5.8 Posterior mean number of states (red line) for the first 4 of the 100 toy datasets with α varying from 0.1 to 100. The prior mean number of states is plotted as a green line and the true number of states and the true value of α in the set is indicated by horizontal and vertical blue lines.

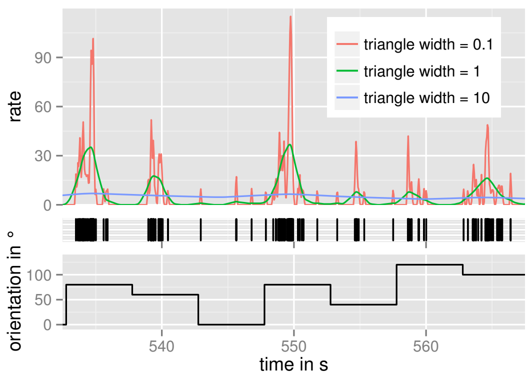


Fig. 5.9 Stimulus and spiking data for a part of the recordings from the first neuron. (*top*) Mean rates computed by using a moving triangle function with three different parameters. (*middle*) Spiking data with every vertical line representing the time of a spike. (*bottom*) Orientation of the moving bar stimulus.

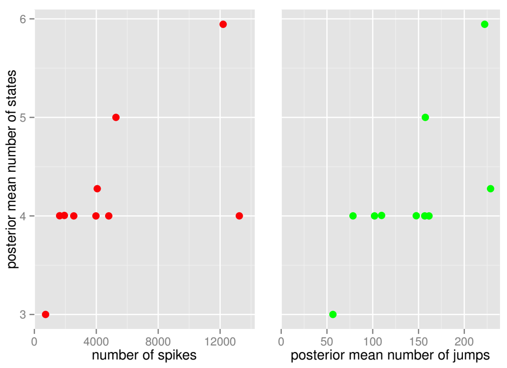


Fig. 5.10 Posterior mean number of states and jumps for all 10 neurons. (*left*) Posterior mean number of states vs. number of spikes in the data for all neurons. (*right*) Posterior mean number of states vs. the posterior mean number of jumps.

that it is sufficient if we have a broad idea of the value of α or the number of states we expect in the data, similar to what we saw when using the CRP model on gene expression data in section 4.4.3. Another interesting observation is that when we look at dataset 1 and 4 the number of states in the posterior seems to converge to a value below the prior mean for α becoming very large. This would implicate that even if we choose a too high value of α the model will not suffer from overfitting by creating a large number of states unless the data forces it to.

Application to Neural Spiking Data

While Poisson models have been used extensively to model neuronal data (e.g. Nawrot et al., 1999; Perkel et al., 1967) it has been established that Poisson processes are not an ideal model for spike trains from single neurons (Barbieri et al., 2001). One of the reasons, which has been explained in section 2.2.2, is the refractory period of neurons. In a Poisson process the waiting time to the next event is exponentially distributed. This means that in a Poisson model spikes can be generated directly after one another whereas neurons cannot create another action potential for a brief period after they have fired. This should not be a problem in our case, because we do not use our model to simulate neuronal spiking but as an inference tool. A second reason why the Poisson model is seen as flawed is bursting (Kass et al., 2005), i.e. rapid firing of neurons during a short period of time. This would be a problem if we

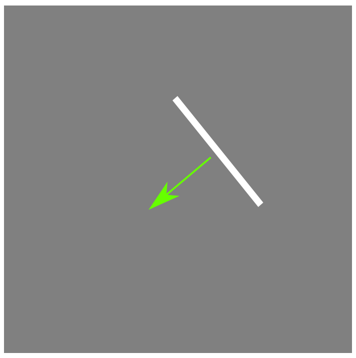


Fig. 5.11 Example of the moving bar stimulus utilized in the neuronal measurements. The green arrow represents the movement direction.

modeled a neuron by a Poisson process with a constant rate but bursting is essentially what our model is made for: sudden changes in the spiking rate of a neuron.

The dataset examined in the following was obtained from multi-site silicon electrodes in the primary visual cortex (V1) of an anesthetized cat. For further information on the experimental setup see Blanche et al. (2005). The data consists of spike trains from 10 different neurons while bars of varying orientation moved through the visual field of the cat (see figure 5.11 for an example of the stimulus). The orientation of the bars ranged from 0° to 340° in steps of 20°. While the order of the orientation was randomized every orientation was shown 8 times for 5 seconds each. Over the whole experiment this results in 720 seconds of total recording time. As the stimulus is discrete in its orientation, we expect to find discrete states in the response of the neurons. A section of the spiking times from one neuron together with the orientation of the stimulus is shown in figure 5.9.

A very simple approach for estimating the spiking rate of neurons is sliding a triangle function over the data and counting the number of spikes under it, weighted by the value of the function at their time (applied by e.g. Nelson et al., 2009; Wang et al., 2005). A problem of this approach is that the width of the window has to be chosen very carefully. As can be seen in figure 5.9, a small window makes it possible to find short periods of high spiking activity (the bursting behavior mentioned previously) but also leads to the estimated rate jumping even when a single spike occurs. A wider window, on the other hand, will not be strongly influenced by spontaneous spiking, but will smooth out bursts and therefore underestimate their rate or even make it hard to recognize them. Our model is able to find bursting periods and cluster them by their spiking rate, but at the same time the spikes between bursts are explained by ground states with lower rates, but longer durations.

We ran our sampler separately for all 10 neurons with an exponential prior over f with 10^{-4} as the mean jump rate[2] and the concentration parameter $\alpha = 0.1$. We chose a low value for α because we expect a large number of jumps (overall the orientation of the stimulus changes 143 times) but expect a lot of the segments to have similar states. Another reason for our choice of α is that our model is very flexible and a too high value of α can lead to overfitting by creating a large number of states to explain random variations and the approximations made by the model. To rule out that this has a strong effect on the results we ran a second simulation with a ten times higher prior mean for f (10^{-3}) and $\alpha = 0.5$. This lead to almost the same posterior number of states and only a slightly higher number of jumps, of which a large fraction had no impact because the state was left unchanged. For more information about this second run see the appendix C.3. For the base distribution over the firing rate P_λ we chose an exponential distribution with a prior mean of 10^6 which makes it a fairly uninformative prior, because the duration of a single spike is in the order of 1ms (Huttenlocher, 1967) which gives an upper bound for the firing rate at around $1000/\text{s}$. For each neuron we generated 110 million samples and dropped the first 10 million as burn-in. Per neuron this took between 80 and 325 minutes on an Intel Xeon CPU with 2.4 GHz. The posterior estimate had converged after a tenth of that time at the latest.

While this may seem like a long time it has to be remembered that to obtain similar results with methods which assume a fixed number of states, Bayes factors would need to be computed for different dimensionalities of the state space. This is a much more complicated task than just estimating the posterior for a range of dimensionalities and would require more computationally demanding approaches, e.g. a bridge sampler (Meng and Wong, 1996), in order to get appropriate results. On top of that the range has to be decided beforehand making it necessary to at least know the minimum and maximum number of states to investigate. In contrast to this, our sampler typically gave a good estimate of the number of states just a few seconds into the simulation. We only need to run the simulation longer if we are interested in a very accurate estimate of the posterior distribution over the number of states.

Despite the number of spikes differing widely between the recordings of the 10 neurons (from 725 to 13244) the posterior mean number of states in the results are very similar, ranging from 3 to 6 states but mostly clustered around 4, as can be seen in figure 5.10. In our model the mean number of states grows with the number of jumps[3] and this behavior seems to be replicated in the data, but it is clear that the posterior differs strongly from the prior—a priori the expected number of states is under 2—which indicates that the posterior results are dominated by the likelihood and not the prior.

[2]All rates for this experiment are in jumps per seconds.

[3]See the prior mean curve in figure 4.20 for how the prior behaves relative to the number of jumps.

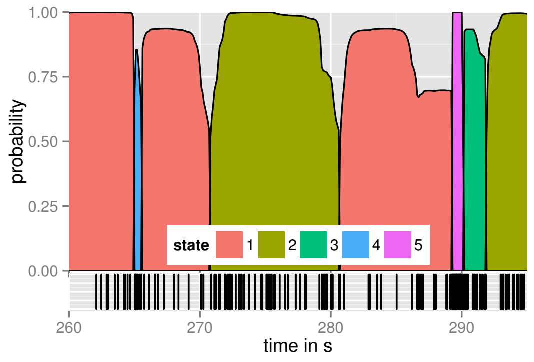


Fig. 5.12 Detail of the posterior states for one of the neurons. In the bottom the spiking times are depicted as vertical black lines. The colored areas in the top represent the state with the highest posterior probability. The probability of that state is plotted as the height of the area. The states are ordered by their firing rate λ in ascending order.

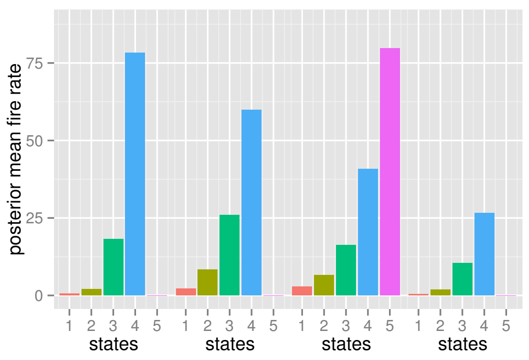


Fig. 5.13 Posterior mean firing rates λ_i at the MAP number of states for 4 of the neurons.

Figure 5.12 shows a small time frame of the spiking data from one of the neurons together with the MAP state over time and its posterior probability. While the states with low firing rates persist over multiple seconds, the bursting states are only active during very short time periods. The value of the firing rates in similar position seems to be similar over the different neurons, i.e. the rates of low firing base states and high firing burst states are in similar orders of magnitude. This is shown in figure 5.13 but it has to be remembered that the orientation is not the only feature of the stimulus that determines the neurons' reaction. The position of the stimulus in the visual field and phenomenons such as synaptic short-term plasticity (Varela et al., 1997) also influence the firing rate which could explain, why only some of the neurons reach the highest bursting rate.

The high-firing bursting states are only active for short periods of times and as figure 5.14 shows they are clearly orientation dependent. This is not surprising as it is widely known that neuron's in mammals primary visual cortex are orientation dependent (Hubel and Wiesel, 1959; Priebe and Ferster, 2008) but it is interesting that our model is able to pick up discrete states corresponding to the orientation of the stimulus. The state with the highest firing rate seems to be concentrated on a range of about 60° while the lower bursting states cover neighboring orientations. The lower bursting states (and with a lower probability the higher ones) are also active for the preferred stimulus rotated by 180° which is a bar of the same orientation moving in the opposite direction. The base state with the lowest firing rate

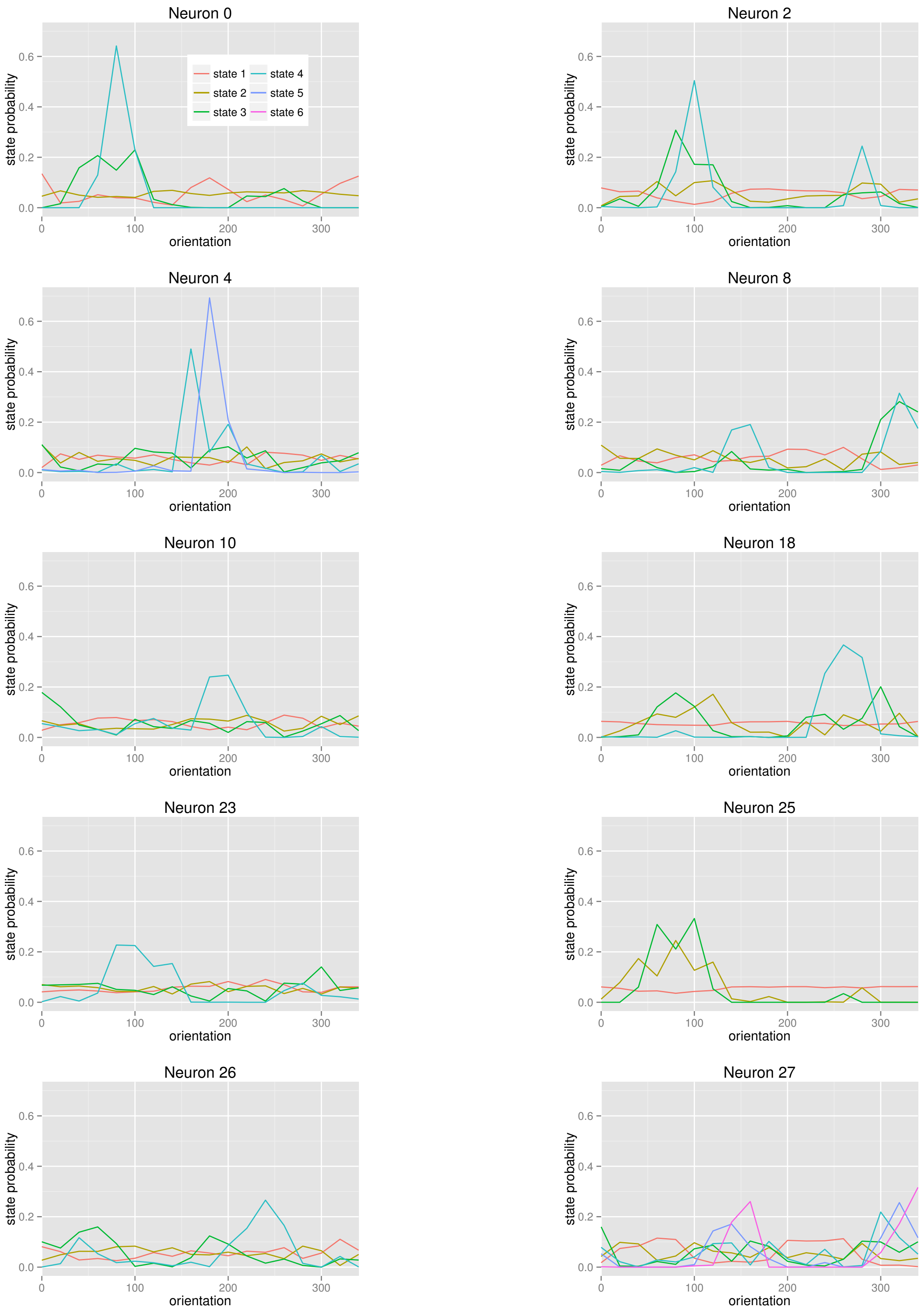


Fig. 5.14 Probability distribution over the orientation of the stimulus while a state is active for all 10 neurons. A probability p at orientation o for state i means that the probability of the stimulus having the orientation o while being in state i is p. States are ordered by the firing rate λ in ascending order. All results are calculated conditioned on the maximum-a-posteriori number of states.

might indicate inhibition, because it is mostly active between the favored orientation and the reversed one.

To test if our assumption of discrete firing rates conserves important properties of the data, we computed the tuning curves for all 10 neurons. This was achieved by counting the number of spikes which were observed while an orientation was presented and dividing it by the 40 seconds each orientation was shown over the experiment. Additionally, we computed the tuning curves obtained from our posterior results, by calculating the average posterior mean rate for each orientation of the stimulus. Both tuning curves are compared for all 10 neurons in figure 5.15 and are found to be mostly overlapping. This result suggests that our assumption does retain the important aspects of the data, instead of forcing suboptimal firing rates on specific time segments because only a limited set of states is available.

While finding bursts of neuronal activity may seem like a simple task, naive approaches, like smoothing the data to get a mean firing rate over time, can fail easily, if the time resolution is not chosen carefully. Even then there is always a trade-off between a good estimate of the bursting rates and a too sensitive method whose results are influenced by single spikes (as seen in figure 5.9). Because of these challenges there has been extensive work on the strategies to identify and measure neuronal bursts (e.g. Chiappalone et al., 2005; Kaneoke and Vitek, 1996). The most widely used model seems to be the Poisson surprise model of Legéndy and Salcman (1985). For this model the surprise value S is computed, which describes how unlikely it is for a number of spikes to occur in a time frame, when assuming that they are generated from a homogeneous Poisson process. Gourévitch and Eggermont (2007) dropped the Poisson assumption in their rank surprise model but both models only looked at clusters of spikes and decided if the cluster represented a burst or not. This binary assumption is also maintained in the model of Tokdar et al. (2010), who use a Gibbs sampler similar to Fearnhead and Sherlock (2006) for posterior inference. In contrast to these approaches our model is not only able to distinguish between burst and non-burst phases, but can find different levels of bursting and non-bursting states and therefore allows better estimation of the firing rates associated to a specific stimulus.

The data from all 10 neurons was used separately to be able to order the states by their λ firing rates. Using all the data combined as a 10 dimensional Poisson process modified by a shared Chinese restaurant process would have been possible as well[4] but this would lead to a large number of states because there are 18 different stimuli and they are in different parts of the visual field. Looking at single neurons this would mean that despite a very large number of states and therefore different spiking rates only a handful of them would represent

[4]Albeit, only with some changes to the join and divide action because the notion of neighboring states would have to be defined differently.

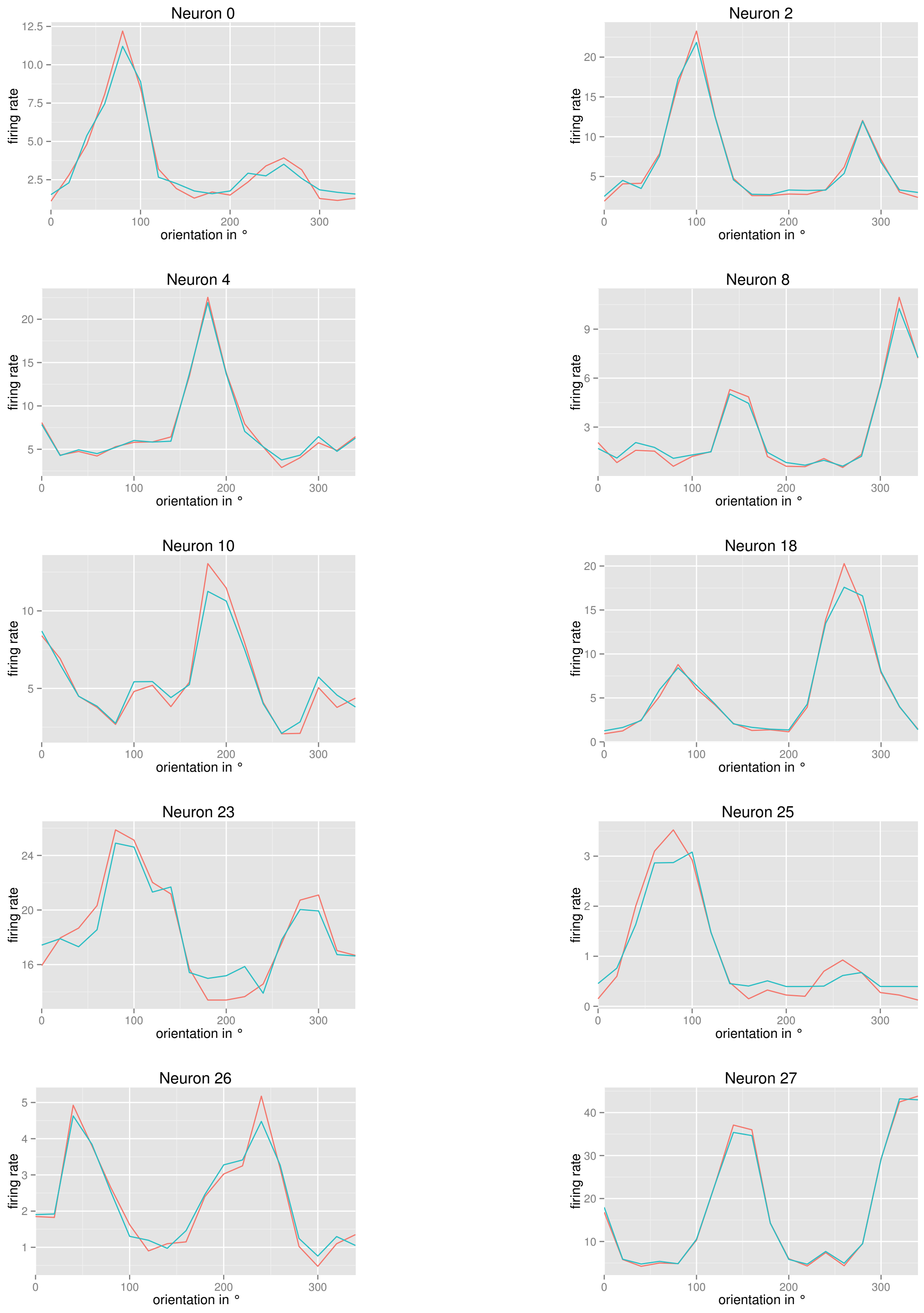


Fig. 5.15 Tuning curve for all 10 neurons from the data and the posterior mean. The red line denotes the tuning cure calculated from the data by counting the number of spikes while an orientation is shown and dividing by the duration the orientation was shown. The red line represents the average posterior mean firing rate from our sampler's results vs. the orientation of the stimulus.

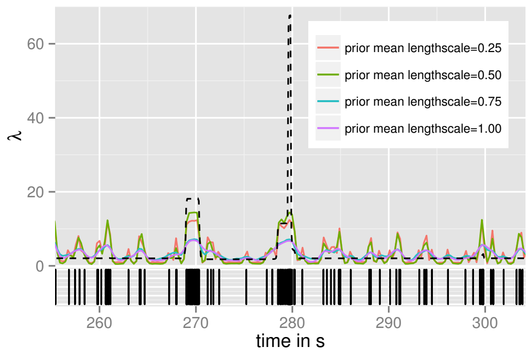


Fig. 5.16 Results of the SGCP Sampler on a small part of the data of one neuron. The black dashed line shows the posterior mean from our random walk sampler while the colored lines are results from the SGCP sampler with different hyper-parameters The spiking times are plotted as vertical black lines in the bottom.

different behavior for each neuron. The rest of the states would just be needed to explain the state of other neurons in the population. The large number of states would slow down the inference process, and even worse, lead to lower quality estimates of the base firing rates of the neurons because they would be spread out over multiple states. Inference in a model with multiple neuronal spike trains would make sense, if we had a large number of neurons and subsets of them shared very similar orientation tuning curves. This was not the case for the data used in this chapter.

Comparison with a Model Assuming a Continuous Rate

A major assumption of our model is that the Poisson rates are discrete values therefore it is interesting to compare our method's results to models which assume a continuous rate. There have been multiple methods for inference on inhomogeneous Poisson processes driven by a continuous rate (Arkin and Leemis, 2000; Zhao and Xie, 1996). One of the more recent ones was proposed by Adams et al. (2009). Their model assumes Gaussian process prior over a transformation of the intensity function $\lambda(t)$ and they describe a MCMC sampler for posterior inference. When applying the sampler to the neural data used in this thesis the computational costs are extremely demanding. To cope with this we restricted the inference task to a small time window of the spike train from only one of the neurons.

The results of the Sigmoidal Gaussian Cox Process (SGCP) model of Adams et al. (2009) are shown in figure 5.16 for 4 different values of the length scale hyper-parameters and compared to the results from our model. The results from the SGCP model have similar problems to the moving average approach described earlier and shown in figure 5.9: Either the bursts are smoothed out or the method becomes so sensitive that even single spikes strongly influence the estimated firing rate.

The format of the neural data seems to be especially bad for the performance of the SGCP algorithm because of its structure. SGCP uses uniformization, a principle introduced by Grassmann (1977), which allows to sample from inhomogeneous Poisson processes by first sampling from a homogeneous one. The rate of the homogeneous Poisson process is required to be an upper bound to the inhomogeneous rate. A sample from the latter can be generated by thinning the Poisson events with a certain probability, which depends on the rate of both the inhomogeneous and the homogeneous process. The SGCP sampler uses uniformization by creating an extension of the original dataset from a homogeneous Poisson process, with the rate being an estimate of the overall maximum rate over the whole time period, and the added data points being thinned out afterwards. In the case of the neural data the maximum rate has to be the spiking rate during the strongest bursts, but this leads to a very large number of (later thinned out) data points in the times between bursts.

We started by using a flat prior over the maximum rate, but this led to a very low value compared to the bursting rates we observe in the data (see figure 5.16). When we tried to fix the maximum rate around our inferred highest bursting rate the algorithm became extremely slow, taking hours to even generate 100 samples on a standard computer, while only being applied to less than a tenth of the data for a single neuron.

Acknowledgements

The Neural data used in section 5.2.3 was recorded by Tim Blanche in the laboratory of Nicholas Swindale, University of British Columbia, and downloaded from the NSF-funded CRCNS Data Sharing website.

Chapter 6

Applications with Other Processes

This chapter consists of short descriptions of the sampler being applied to models which use the Cox-Ingersoll-Ross process or the multivariate Ornstein-Uhlenbeck process as the observed processes. The last section gives a brief guide how the samplers needs to be adapted to model combinations, which were not part of this thesis.

6.1 Cox-Ingersoll-Ross Process

This section summarizes the work of Herrmann (2014) who modified our sampler to work with data generated by a Cox-Ingersoll-Ross (CIR) (Cox et al., 1985) process with the parameters jumping according to the changepoint model of section 4.3.

6.1.1 Model

As described in section 2.1.2 the CIR process resembles the Ornstein-Uhlenbeck process with the important difference that the strength of the diffusion depends on the value of the process. We here use the form of the SDE from Herrmann (2014)

$$dX = \lambda(A - X)dt + \sigma\sqrt{X}dW \tag{6.1}$$

which is the same as in (2.26) but with $A = b/\lambda$.

The small difference to the OU process complicates inference because instead of an easy to calculate Gaussian density, the transition density of the CIR process is a noncentral chi-square distribution which is computationally very expensive to evaluate (L'Ecuyer and Owen, 2010, p. 301). Additionally, when assuming noisy observations the unobserved values of the process at the time of the observations cannot be marginalized out analytically as has been done for the OU process in section 4.1.1. Therefore an additional Gibbs step to sample the unobserved process would be necessary (Stramer et al., 2010).

Because of these restrictions there have been numerous attempts to approximate the transition density and likelihood (Malham and Wiese, 2013; Sahai and Ojeda, 2003) to enable efficient Bayesian inference of the CIR model. In Herrmann (2014) all the observations were assumed to be exact and the transition density was approximated by a Gaussian distribution with the same mean and variance as the noncentral chi-square distribution. This allows to calculate the likelihood without problems when there are no jumps in the parameters between observations:

$$\begin{aligned} P(D|\theta_{0:T}) &= P(d_1|\theta(t_1))\prod_{i=2}^{n} P(d_i|d_{i-1},\theta(t_{i-1})) \\ &= P(X(t_1)|\theta(t_1))\prod_{i=2}^{n} P(X(t_i)|X(t_{i-1}),\theta(t_{i-1})). \end{aligned} \tag{6.2}$$

If there are jumps between the observations, the unknown state of the CIR process $X(\tau_j)$ at the point of the jump has to be marginalized out:

$$P(X(t_i)|X(t_{i-1})) = \int_0^\infty P(X(t_i)|X(\tau_j))P(X(\tau_j)|X(t_{i-1}))dX(\tau_j) \tag{6.3}$$

For the OU process this was possible as we have shown in section 4.1.1. When using the moment matching approximation for the CIR process on the other hand, the variance of the Gaussian transition density depends on the value of the process:

$$v(\Delta t, X(t_{k-1})) = X(t_k)\frac{\sigma^2}{\lambda}(\exp(-\lambda\Delta t) - \exp(-2\lambda\Delta t)) + \frac{B\sigma^2}{2\lambda}(1-\exp(-\lambda\Delta t))^2. \tag{6.4}$$

This means that the integral in (6.3) is no longer solvable analytically. Herrmann (2014) decided to numerically approximate the integral by the Gauss-Kronrod approach (Dahlquist and Björck, 2008, p. 573).

6.1.2 Sampler

The sampler is very similar to the sampler for the hidden changepoint process from section 4.3. The Gibbs step of sampling the jump times uses the same 3 actions of *adding*, *removing* and *shifting* a jump. Naturally the acceptance probability has to be computed using the approximation of the CIR process' likelihood. According to Herrmann (2014) using the moment matching approximation instead of the exact transition density gives accurate results for $A\lambda \gg \sigma^2$.

Sampling the parameters

While for the OU process A had a Gaussian posterior density if the prior was chosen accordingly, this is no longer the case for the CIR process. A, λ and σ are all restricted to non-negative values and therefore they are all updated by doing a random walk on their logarithm, i.e. drawing proposals from the log-normal distribution

$$Q(A^*|A,\sigma_A) = \frac{1}{A^*\sqrt{2\pi\sigma_A^2}} \exp\left(-\frac{\log(A^*)-\log(A)}{2\sigma_A^2}\right), \tag{6.5}$$

and accordingly for λ and σ.

Sampling f again can be done by directly sampling from the gamma posterior because a conjugate gamma prior was chosen.

6.1.3 Application to Exchange Rate Data

One advantage of the CIR over the OU model is that the system noise automatically scales with the value of the process. This behavior is expected for many applications and we can let the system noise strength σ^2 be unaffected by the hidden jump process without losing its ability to scale. CIR processes have been used to model currency exchange rates (Ewald and Wang, 2010) and therefore Herrmann (2014) applied them to the EUR/USD exchange rates from the beginning of 2007, before the global financial crisis, until the beginning of 2013.

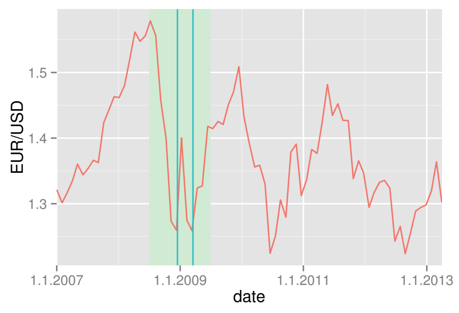
4 different datasets were compiled from the raw data, each spanning a different time period and time scale (6 years, 1 year, 5 month and 3 days). For the sampler the times were then scaled to $0 < t < 1000$ for all 4 datasets. The Gaussian prior over A was chosen as

$$P(A) = \mathcal{N}(A; m_A, v_A) m_A = \frac{\max(\mathbf{D}) + \min(\mathbf{D})}{2} v_A = (m_A + 1.5\max(\mathbf{D}))^2, \tag{6.6}$$

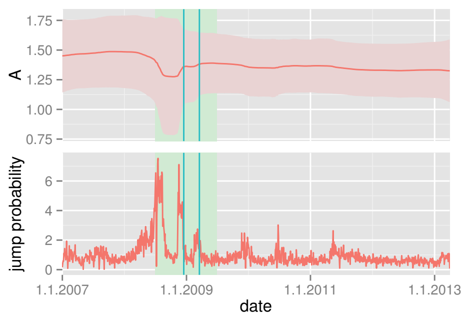
where $\mathbf{D}$ represents the whole raw dataset.

Figure 6.1 shows the posterior results from the sampler for all 4 datasets. On all but the smallest time scale a jump was found prior to two important events during that crisis[1]: On the 16th of December 2008 the Federal Reserve of the United States (FED) set the overnight federal funds rate to between zero and 0.25%. Roughly 3 months later, on the 18th of March

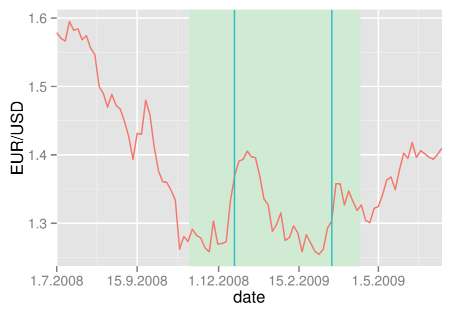
[1]The 3 day period in figure 6.1g and 6.1h did not include the second event.



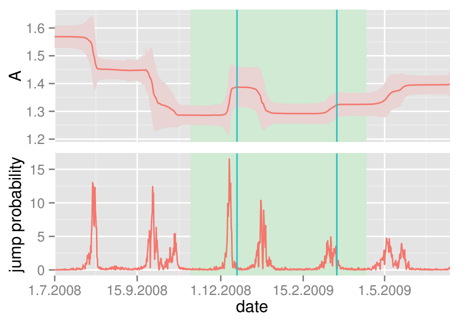
(a)



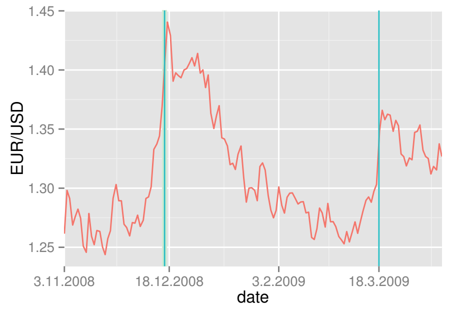
(b)



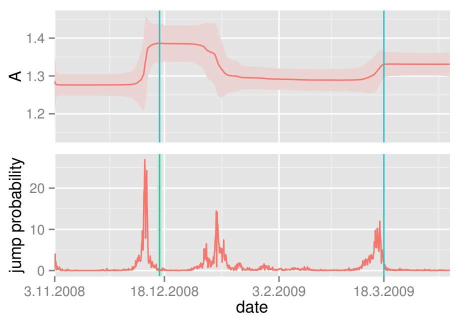
(c)



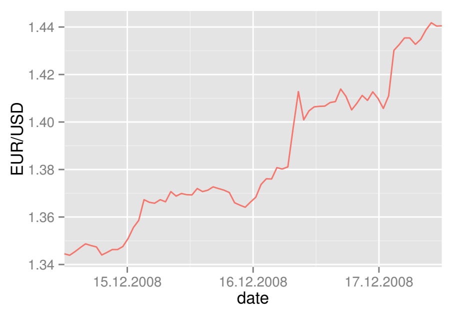
(d)



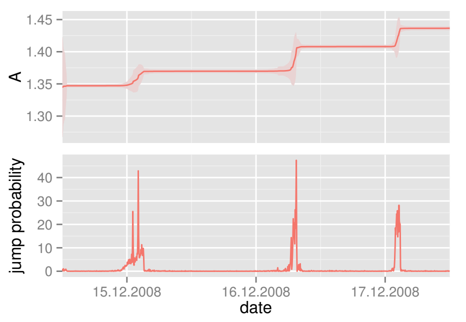
(e)



(f)



(g)



(h)

Fig. 6.1 Results from the 4 datasets of different time periods. The data is plotted on the left as a red line. On the top right the posterior mean of A vs. t is plotted in red with a 95% confidence interval surrounding it. On the bottom right the probability of a jump over t is drawn as a red line. The green shaded areas denote the time span of the next dataset and the blue vertical lines are the launch of the Federal Reserve's quantitative ease program on the 16th December 2008 and its expansion on the 18th March 2009. The figures were reproduced with the data from Herrmann (2014).

2009, the FED announced an expansion of the quantitative easing program (Gance, 2014). The jumps are always found prior to the events because they were either known beforehand or similar actions were expected.

6.2 Multivariate Ornstein-Uhlenbeck Process

While in chapter 4 we already talked extensively about models using a multidimensional Ornstein-Uhlenbeck process as its observed process, in all these cases the different dimensions of the OU process were independent, conditioned on the path of the hidden parameter process. This works well for the application to systems biology and the joint hidden jump process allows for coupling between the dimensions but there are other models in which this form of coupling is integrated in the stochastic differential equations of the non-changepoint model. One example are dynamical systems where the dimensions can represent e.g. the velocity and position of an object.

6.2.1 Model

Our model is a linear mass-spring-damper system, because they play an important role in modeling robot motion (Morita and Sugano, 1995; Xu et al., 2011) and other fields like computer graphics (Irving, 2007).

If $x(t)$ represents the displacement and $v(t)$ the velocity, then the state-space description of the system becomes

$$\begin{aligned} dx(t) &= v(t)dt \\ dv(t) &= (-\lambda x(t) - \gamma v(t) + b)dt, \end{aligned} \tag{6.7}$$

where γ represents the damping, λ the stiffness[2] and b an external force (Williams and Lawrence, 2007, pp. 5-7).

To modify this to our model class we let both λ and γ switch and account for noise in the dynamics of the velocity by including a Wiener process dW:

$$\begin{aligned} dx(t) &= v(t)dt \\ dv(t) &= (-\lambda(t)x(t) - \gamma(t)v(t) + b)dt + \sigma dW, \end{aligned} \tag{6.8}$$

If we define $\mathbf{X}(t) = (x(t), v(t))^\top$ we can write the SDEs in matrix form

$$d\mathbf{X}(t) = (\mathbf{B} - \Lambda(t)\mathbf{X}(t))dt + \mathbf{\Sigma} d\mathbf{W} \tag{6.9}$$

[2]We set the mass to $m = 1$ for simplicity.

with

$$\Lambda(t) = \begin{pmatrix} 0 & -1 \\ \lambda(t) & \gamma(t) \end{pmatrix}, \mathbf{B} = \begin{pmatrix} 0 \\ b \end{pmatrix} \text{ and } \mathbf{\Sigma} = \begin{pmatrix} 0 & 0 \\ 0 & \sigma \end{pmatrix}. \tag{6.10}$$

We choose a Chinese restaurant process to draw new parameters after each jump, as it is our most flexible model and we are interested in using it on data where we don't know the exact dimensionality of the hidden state space.

6.2.2 Sampler

We can reuse our sampler for the independent Ornstein-Uhlenbeck processes in section 4.4 without making many changes.

Sampling of the Jump Times

The sampling of the jump times is exactly as described in section 4.4, since we have multiple dimension we cannot have a linear ordering of the states and therefore do not use the *join* and *divide* action used for the Chinese restaurant process driven Poisson process in section 5.2. The only change to the acceptance probabilities of the proposal actions is that we use the likelihood ratio for the multivariate OU process. The calculation of the likelihood function by integrating out the unknown state of the process $X(t)$ at the time of the observations is very similar to the univariate OU process. For a derivation see section A.5 in the appendix.

An important point to remember is that calculating the likelihood for a multivariate OU process is slower than for a comparable OU process with conditional independent dimensions because it includes the calculation of matrix exponentials which are computationally costly.

Sampling the parameters

While we fix all parameters besides Λ in our example, we will discuss how the other parameters can be sampled as well.

The elements of $\mathbf{B}$ are linear in the log-likelihood and can be sampled directly from a multivariate Gaussian similar to the A and b parameters in chapter 4. The derivation is not part of this sampler but should be fairly easy to obtain from the calculation of the multivariate OU likelihood in section A.5 and the description of sampling A and b directly in section A.3.

The values of Λ and $\mathbf{\Sigma}$ can be sampled by doing a Metropolis random walk update on their components. If the parameters need to be positive we can sample from a log-normal distribution with its mode at the current value. If this is not required we use a simple Gaussian random walk.

Dataset	Number of Jumps		Number of States	
	True	Posterior Mean	True	Posterior Mean
1	3	3.4	2	2.0
2	4	5.6	3	3.0
3	8	8.8	4	3.9

Table 6.1 Posterior mean and true number of jumps and states for the synthetic multivariate OU datsets.

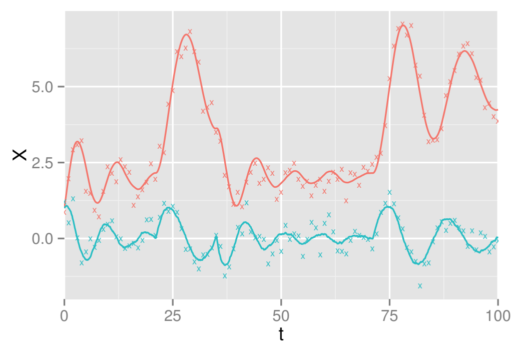
As for the previous models, the jumping rate of the hidden process f can be directly sampled from its posterior gamma distribution (5.12) if we assume conjugate gamma priors.

6.2.3 Results & Possible Applications

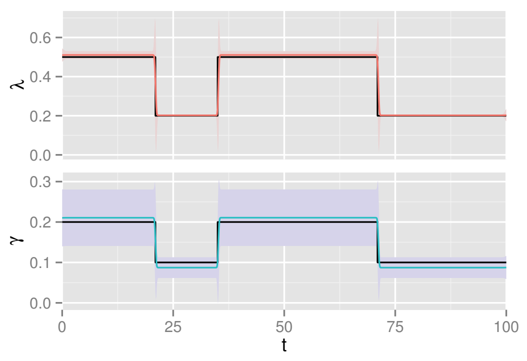
We generated 3 datasets with 2, 3 and 4 states. Besides the jumping parameters $\lambda(t)$ and $\gamma(t)$ all parameters were the same for all 3 datasets and left fixed to their true values for the simulation. The simulation parameters were $b = 1.0$, $\sigma = 0.04$ and $\sigma_{obs} = 0.1$. As the jumps processes were generated by hand there were no true values for f and α but we fixed them to $f = 0.05$ and $\alpha = 2.0$ for all 3 datasets. The base distribution over the values of $\lambda(t)$ and $\gamma(t)$ were set to Gaussian distributions with zero mean and variance 10.

In figure 6.2 the datasets as well as the posterior results are shown. We generated 2.1 million samples for each dataset and dropped the first 100,000 samples as burn-in. Running on an Intel Xeon CPU with 2.40 GHz this took roughly 5.5, 8.5 and 12 hours for the dataset with 2, 3 and 4 states, respectively. The path of $\lambda(t)$ is almost perfectly matched. $\gamma(t)$ on the other hand, is not as accurately predicted. The broader posterior over $\gamma(t)$ can be explained by the fact that $v(t)$ oscillates around zero which makes it hard to predict the value of γ because its influence goes to zero as $v(t)$ goes to zero. This can be especially seen when looking at the middle segment of the second dataset in figure 6.2d which has a very high variance. When we look at the corresponding observations in figure 6.2c we can see that they are clustered around zero for $v(t)$. The posterior mean number of jumps and states is compared to the true values in table 6.1 and besides a tendency to overestimate the number of jumps they are very close.

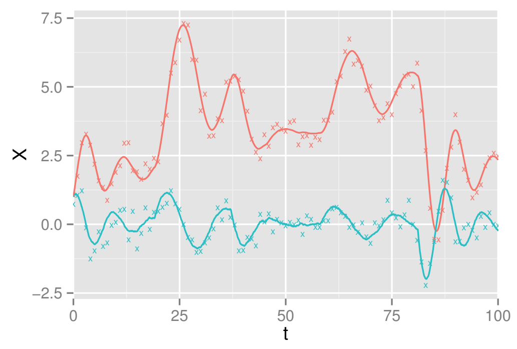
While this is only a short experiment on synthetic data, it looks very promising and could be further expanded on. We motivated the multivariate OU process by the use of mass-spring-damper systems in robotics and that field would certainly be an interesting application in the future.



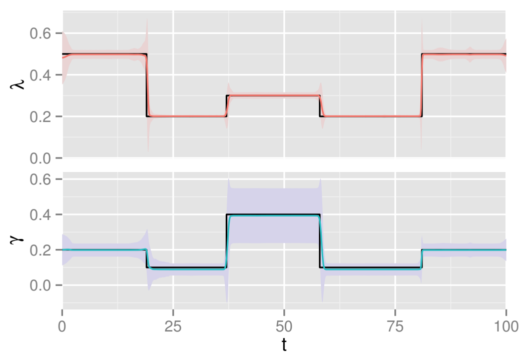
(a)



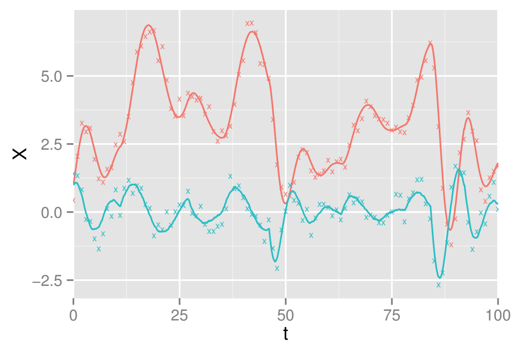
(b)



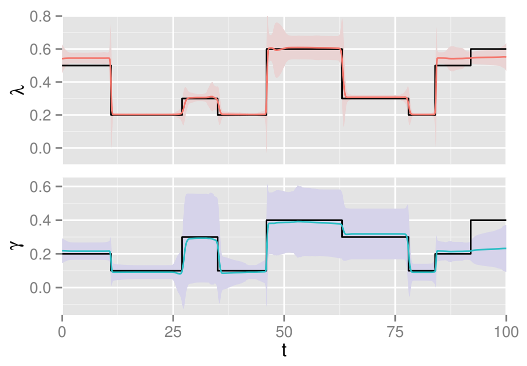
(c)



(d)



(e)



(f)

Fig. 6.2 Results for synthetic data from a multivariate OU process. On the *left* the true path of $X(t)$ is plotted as a solid line and the noisy observations are plotted as crosses. Red always represents $x(t)$ and blue $v(t)$. On the *right* we show the path of $\lambda(t)$ (*top*) and $\gamma(t)$ (*bottom*). The true path is drawn as a solid black line while the posterior mean is a colored line with a 95% confidence interval surrounding it.

6.3 Modifying the Sampler to Other Processes

The structure of our sampling algorithm makes it easy to adapt it to different combinations of observed and hidden processes. This section aims to be a brief guide about how to modify the sampler to work with process combinations which were not discussed in this thesis.

6.3.1 The Likelihood

Most of the sampler's computation time is needed for calculating the likelihood for the Metropolis-Hastings acceptance rate. If the parameters can be sampled directly, their conditional distribution arises from the form of the likelihood as well. For all models in chapter 4 and 5, we could analytically compute the likelihood given a set of observations **D** by marginalizing over the path of the observed $X_{0:T}$[3]

$$P(\mathbf{D}|\theta_{0:T}) = \int P(\mathbf{D}|X_{0:T}, \theta_{0:T}) P(X_{0:T}|\theta_{0:T}) dX_{0:T}. \tag{6.11}$$

If the model is modified to another observed process, for which the integral can not be solved analytically, we suggest two possibilities: Either the likelihood is approximated, as proposed for the CIR process in section 6.1, or another Gibbs step is implemented, during which a path $X_{0:T}$ is sampled from $P(X_{0:T}|\mathbf{D}, \theta_{0:T})$. Instead of the likelihood $P(\mathbf{D}|\theta_{0:T})$, the acceptance probability would then include $P(X_{0:T}|\theta_{0:T})$.

Another important factor is the actual implementation of the likelihood calculation. In the case of an observed Poisson process in chapter 5, large parts of the likelihood canceled each other out when the acceptance probability of a proposal was calculated. Because the computation of the likelihood function usually makes up a large part of the overall computation time, optimization schemes, like e.g. preprocessing the data or saving partial results for later reuse, can speed up the sampler enormously.

6.3.2 Sampling the Jump Process

What ever kind of jump process is chosen, the three proposal actions of adding, removing or shifting a jump are always necessary. If adding a single jump often leads to low acceptance rates because it changes a large part of the path, as was observed for the switching model in section 4.1, proposal actions which add jumps but leave most of the hidden process' path intact should be introduced. For models using the Chinese restaurant process the action of switching the state of a segment is important to speed up the convergence of the sampler. If

[3]For the Poisson model the observations contained complete information over the path, therefore this was not necessary.

the number of segments is high, joining and dividing states allows the sampler to more often change the dimensionality of the hidden process.

It would certainly be interesting to implement different forms of the hidden process, e.g. have non-exponential waiting times for the jumps or a different prior distribution over the number of states than the CRP implies. Such changes would only affect one part of our sampler: the acceptance probabilities of the proposal actions when sampling the jump process. All the other parts of the sampler are conditioned on a particular path of the jump process and therefore independent from its prior distribution. Similar to the likelihood, it is beneficial to look for any parts of the prior probability over the hidden process which cancel out in the acceptance probability.

For some choice of observed or hidden process there might be proposal actions not discussed in this thesis, which accelerate the convergence of the sampler. When creating a new proposal action one has to be very careful to not invalidate the sampler. If the action cannot reverse itself, an inverse action has to be introduced, similar to the add-remove and join-divide pairs. When defining how these actions work on the path it is important to check that every possible outcome of the proposal action can be reverted by its inverse action and vice versa. If this is not the case the sampler will no longer have the target distribution as its equilibrium distribution.

6.3.3 Sampling the Parameters

If it is possible to directly sample from the conditional posterior of the parameters, conditioned on the current jump process and the observations, then this should be the preferred approach. The only exception would be if there is evidence that using an indirect sampling method is able to produce samples of the same quality faster.

For parameters where the conditional posterior does not have a simple form we applied a Metropolis-Hastings random walk. When the parameters were defined on the real line we chose a Gaussian random walk, when they had to be positive a random walk on their logarithm was implemented, i.e. the proposal was a log-normal distribution with its mode on the last value. More complex sampling strategies like adaptive MCMC (Andrieu and Thoms, 2008) or Hybrid Monte Carlo (Duane et al., 1987) could be employed as well if they promise to improve the quality of the samples.

Chapter 7

Conclusion

This thesis introduced a framework for a MCMC sampler which is able to sample from the posterior distribution of a range of changepoint models. We introduced the class of models we are interested in, which consists of an observed stochastic process whose parameters undergo sudden changes at random times. We introduced different forms of the hidden jump process including a form based on a Chinese restaurant process (CRP) which allows inference over the number of states.

Our sampler was demonstrated to be very flexible and to work well on many different combinations of observed and hidden process. We showed that the sampler's results improve on previous approximative approaches and open up possibilities for complex models of gene expression to be investigated which were not tractable before. Additionally, the CRP model allowed us to verify assumptions about the number of transcription factors involved in gene expression data from yeast cells. Besides the application to gene expression data, our sampler was able to correctly capture known historic events for two different finance datasets.

After modifying our sampler to Poisson process observations, it proved to outperform other sampling algorithms on datasets generated by a Markov modulated Poisson process. The Poisson process observation model was combined with the CRP prior and applied to neuronal spiking data. The posterior results produced by the sampler managed to find discrete bursting states in the spiking data and connect them to the stimulus in the experiment. Our sampler was proven to enable efficient Bayesian inference for a broad range of changepoint models and we gave a brief explanation how to modify it to combinations of stochastic processes not covered in this thesis.

7.1 Related Work

Changepoint analysis is a very active field and other approaches to it have been mentioned throughout this thesis. Before we draw our conclusions we want to discuss and compare recent work in the field which has not been, or only briefly, mentioned.

7.1.1 General Changepoint Models

While birth-death moves have been used for changepoint estimation since the well known reversible jump sampler of Green (1995), our model is different in several aspects. In the reversible jump setting a change in the number of jumps was treated as switching to a different model. In contrast, our model formulation allows both the number of changepoints and the number of states (in the CRP model) to vary *inside* the model, because we treat the hidden jump process as a infinite dimensional path object. Furthermore, we introduce more complex proposal actions than Green (1995) and broaden the scope of its application by allowing diffusion processes to be modified by the hidden jump process.

Fearnhead (2006) also works with an unknown number of changepoint but these changepoints can only happen at discrete points in time. Conditioned on the number and position of changepoints, as well as the parameters in each segment, the observations in our class of models come from a Markov process. However, in the model of Fearnhead (2006) the observations are i.i.d., conditioned on the hidden process. They describes a method to directly sample from the posterior of their model based on the forward-backward algorithm. This approach was extended in Fearnhead and Liu (2011) to allow dependencies across segments but the observations inside a segment remain i.i.d. and therefore cannot be produced by a stochastic process, as is the case in our OU model.

Fixing the number of changepoints beforehand leads to undesired effects according to Koop and Potter (2009). Their solution is to allow changepoints to be able to occur outside of the scope of the data. This seems less elegant than our approach of directly sampling the number of changepoints inside the model.

A model where Gaussian processes describe latent forces in robot movements was proposed by Alvarez et al. (2010). The latent forces can switch between different latent functions but in contrast to our model the number of jumps is fixed beforehand and states cannot be reused.

An approach for changepoint inference by using a nested Laplace approximation instead of MCMC sampling was described by Wyse et al. (2011). As expected, the approximation is faster than MCMC methods and it allows for dependencies between observations from the same segment. However, the algorithm requires time discretization and only delivers results conditioned on the (approximated) maximum-a-posteriori number of jumps.

One example of an online changepoint algorithm is presented by Grande (2014). Their algorithm describes the data stream by switching Gaussian processes and decides online if a changepoint improves the likelihood significantly. As an online method, the approach is very fast but does *not* perform Bayesian inference. The results from our sampler were used by Grande (2014) to measure the quality of their algorithm.

7.1.2 Methods Based On The Dirichlet Process

All the models described so far only work with either a fixed number of states or non-reusable states. In the following we discuss work, which tries to estimate the number of states as well, similar to our CRP model.

The infinite HMM, an extension of a HMM with a countably infinite number of states, was proposed by Beal et al. (2001). It enables HMMs to have a flexible number of states by utilizing a Dirichlet process to integrate out the infinite number of parameters. In contrast to our model, it is based on discrete time and there is no observed process on top of the hidden state, just an emission distribution with parameters depending on the hidden state.

The concept of the infinite HMM was generalized by Teh et al. (2006), who proposed a hierarchical Dirichlet process (HDP). The HDP is used to model groups of data by a set of Dirichlet processes which are coupled through their base distribution which itself is drawn from a Dirichlet process. In Teh et al. (2006) it was shown that the infinite HMM of Beal et al. (2001) is a special case of the HDP.

Fox et al. (2011) extend the HDP to a so-called sticky HDP, which has a hyper-parameter controlling the probability of self-transitions. They combine the sticky HDP with dynamical linear systems similar to what we did in section 6.2. However, the important difference to our model is that their model is still defined in discrete time and therefore has the form of an HMM, while our model is defined in continuous time, which e.g. does not require us to choose the grain size of the time discretization while we can still benefit from a high time resolution when necessary.

Another difference is the flexibility of our model: Plugging in different observed or hidden processes would in many cases only require modifications of the prior or likelihood ratios in the acceptance probabilities. The sampler of Fox et al. (2011) could not be as easily adapted. Additionally, the algorithm of Fox et al. (2011) samples a complete path in every iteration. Especially for a very fine time discretization this can be computationally very expensive. If the posterior is very specific, each iteration will only result in small differences of the hidden path, similar to our random walk approach, but with higher computational costs.

Our sampler framework aims to be very easily modified to work on a large variety of models. Naturally, there are many algorithms which are made specifically for a particular problem. We give an overview over recent methods which focus on applications which are also part of this thesis.

7.1.3 Markov Modulated Poisson Processes

Scott (1999) applied a two-state MMPP to fraud detection in network traffic data. Inference was done by a Gibbs sampler after approximating the MMPP as a discrete time HMM. A different approach was taken by Kou et al. (2005) who integrate out the hidden MJP completely. It is not possible to directly sample from the resulting posterior probabilities over the parameters. Therefore, their approach is an adaptive random walk sampler directly on the parameters. Fearnhead and Sherlock (2006) demonstrated that their exact Gibbs sampler provides better samples than the algorithm of Kou et al. (2005). However, the downside of the exact Gibbs sampler in contrast to our approach is that its computation costs grow linearly with the number of Poisson events, as shown in section 5.1.3. The same point was made by Rao and Teh (2013). Their algorithm uses uniformization to get samples from the posterior MJP without the computation time scaling with the number of observations. In contrast to our approach, their algorithm has not been shown to work with an observed diffusion process or a different hidden process than the classical MJP.

7.1.4 Systems Biology

The model of transcriptional regulation in chapter 4 is based on Sanguinetti et al. (2009) and Opper et al. (2010) who derived it by extending the model of Barenco et al. (2006) to include a binary telegraph process which represents the on and off switching of a transcription factor. Opper et al. (2010) present an exact solution as well as a mean field approximation and demonstrate its use both on synthetic and real data, including the ComS protein data.

In Opper and Sanguinetti (2010) multiple telegraph processes with combinatorial effects were introduced and a variational approximation derived. In contrast to our model, they did not include a system noise term.

For the switching model a very similar approach to ours was developed in parallel by Jenkins et al. (2013). However, they formulate their model in a reversible jump setting and do not introduce any further proposal actions than Green (1995) did.

Ocone and Sanguinetti (2011) and Ocone and Sanguinetti (2013) augment the previous models to hierarchical models representing feed-forward loop motifs in gene expression, i.e. a master transcription factor controls a slave transcription factor, which controls a number of target genes. The master transcription factor is a binary telegraph process as in the previous models but the slave transcription factor has a continuous concentration value and only influences the target genes if the concentration rises above a threshold. Variational approximations are derived for these models. To our knowledge our sampling algorithm can not be extended to these kinds of models without using an approximation of the likelihood as we can never exactly predict if the continuous value surpasses the threshold between observations.

7.1.5 Finance

Many changepoint models with financial applications assume that the observations in a segment are i.i.d. from a distribution whose parameters change over time. Examples for this approach are Giordani and Kohn (2008) and Allen et al. (2013). The latter apply a non-parametric approach to model the unknown densities and find the maximum likelihood changepoint positions. Applied to daily data from multiple stock indices from 2003 to 2013, Allen et al. (2013) find changepoints and connect them to events during the global financial crisis similar to our results for the DAX dataset in section 4.3.

A switching model with observations being produced by a CIR process was introduced by Dahlquist and Gray (2000). In contrast to the work summarized in section 6.1, their model was only described in discrete time for a fixed number of states. Additionally, they performed maximum-likelihood estimation instead of Bayesian inference.

7.1.6 Neuron Spiking

The aim of Pillow et al. (2011) is inferring the position of changepoints in a stimulus, given a neuron population's spiking response. The stimulus is assumed to come from a multivariate distribution whose parameters undergo sudden changes. The paper proposes a maximization approach to obtain the maximum-a-posteriori estimate and a Gaussian approximation to infer the full posterior distribution. Their model shares with Putzky et al. (2014) that the spiking data is not assumed to be generated by a Poisson process. Instead the generalized linear model of Nelder and Wedderburn (1972) is applied, which has been used extensively in the analysis of spiking data. Putzky et al. (2014) combine a Markov decision tree with binary splits with the generalized linear model to represent the hidden states of neurons. They employ a variational expectation maximization algorithm for inference and apply it to spiking data from the primary visual cortex of an anesthetized macaque with a drifting grating stimulus.

7.2 Discussion & Outlook

The overview of other approaches of changepoint estimation should emphasize that the choice of continuous time and observations which are not i.i.d distinguishes our work from most models in the field. While our birth-death approach might appear very simple and inefficient we showed that it can outperform exact Gibbs sampling and is easily adapted to different models. Before the current algorithm we tried out a sampler which drew a complete new path by using the posterior transition rates derived in section A.2 of the appendix. This was only an approximation to drawing from the correct conditional probability because the transition rates were assumed to be piecewise constant. The proposal worked well most of the time, but sometimes little details in the posterior got lost in the approximation. When despite of the approximation, a path including these details was proposed, it was immediately accepted and the sampler got stuck with it for thousands of iterations. This was the result of the proposal ratio in the Metropolis-Hastings acceptance probability being extremely low. This observation should caution against applying too complex and not well understood proposal mechanisms.

Regarding the CRP model, α is the parameter which controls the dimensionality of the jump process. Its choice can be interpreted not only as a prior estimate of the number of states but as the perspective from which we want to look at the data. Lower values of α represent an interest in large scale changes, ignoring the small details. However, this does not work if there is a lot of data which suggests otherwise. As an example we used our Poisson CRP sampler on access log data of the TU Berlin's website. The dataset consists of over 9 million timed accesses over the period of a week. When looking at the empirical rate, it is obvious that the changes are continuous and not sudden. Even for extremely low values of α the CRP did not stop adding states until we stopped the algorithm. The preliminary results showed that the process was fitting a step function to the continuous rate.

Although our prior over the CRP jump process does not induce state-dependent transition rates[1], this does not hold for the posterior. If needed, we can compute the posterior transition matrix with state-dependent jump rates from the samples.

An interesting approach for further research could be to look at different prior distributions over the parameters, e.g. to have a formulation which allows an unknown number of

[1]This is only true if we treat jumps which end up in the same state as state transitions. Since these kind of jumps are rarely seen in the posterior this should not change the statetment.

reusable states but does not facilitate the "rich-get-richer" effect of the CRP. Additionally, the assumption of exponential waiting times is not necessary for our sampler to work. If we would assume that the time to the next jump is e.g. gamma distributed, the only change in the sampling algorithm would be in the acceptance probabilities.

Considering further applications on neural data it might be interesting to change the observation model to come from a more complex model than a Poisson process. The model could be more closely based on neurobiology, e.g. the generalized linear or an integrate and fire model. The major disadvantage of such an approach is that we would lose the ability to calculate the likelihood as quickly as for the Poisson model.

A more general improvement of the sampler could be to apply adaptive MCMC methods to sample the parameters whose conditional posteriors are not of a simple form we can directly sample from. This could also reduce the necessary effort when applying the sampler to new datasets.

As far as further applications to real world data go, the task of segmenting robot motion data we suggested in section 6.2 looks very promising. This would be the same task as in Alvarez et al. (2010) but their model did not allow for the number of states to be unknown. The results could be used to learn motion primitives for a framework similar to that of Paraschos et al. (2013). The same model could further be applied to motion data from other fields, e.g. motion capturing or the analysis of the bee waggle dance.

Bibliography

Adams, R. P., Murray, I., and MacKay, D. J. C. (2009). Tractable Nonparametric Bayesian Inference in Poisson Processes with Gaussian Process Intensities. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 9–16, New York, NY, USA. ACM.

Alberts, B. (1989). *Molecular Biology of the Cell*. Garland Pub., 2 edition.

Allen, D. E., McAleer, M., Powell, R. J., and Kumar-Singh, A. (2013). Nonparametric Multiple Change Point Analysis of the Global Financial Crisis. *Available at SSRN 2270029*.

Alvarez, M., Peters, J. R., Lawrence, N. D., and Schölkopf, B. (2010). Switched Latent Force Models for Movement Segmentation. In *Advances in Neural Information Processing Systems*, pages 55–63.

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

Aparicio, O., Geisberg, J. V., and Struhl, K. (2001). *Chromatin Immunoprecipitation for Determining the Association of Proteins with Specific Genomic Sequences In Vivo*, chapter 17. John Wiley & Sons, Inc.

Archambeau, C., Opper, M., Shen, Y., Cornford, D., and Shawe-Taylor, J. S. (2008). Variational Inference for Diffusion Processes. In Platt, J., Koller, D., Singer, Y., and Roweis, S., editors, *Advances in Neural Information Processing Systems 20*, pages 17–24. Curran Associates, Inc.

Arkin, B. L. and Leemis, L. M. (2000). Nonparametric Estimation of the Cumulative Intensity Function for a Nonhomogeneous Poisson Process from Overlapping Realizations. *Management Science*, 46(7):989–998.

Balachandran, A., Voelker, G. M., Bahl, P., and Rangan, P. V. (2002). Characterizing User Behavior and Network Performance in a Public Wireless LAN. *SIGMETRICS Perform. Eval. Rev.*, 30(1):195–205.

Barber, D. (2012). *Bayesian Reasoning and Machine Learning*. Cambridge University Press.

Barbieri, R., Quirk, M. C., Frank, L. M., Wilson, M. A., and Brown, E. N. (2001). Construction and analysis of non-Poisson stimulus-response models of neural spiking activity. *J. Neurosci. Methods*, 105(1):25–37.

Barenco, M., Tomescu, D., Brewer, D., Callard, R., Stark, J., and Hubank, M. (2006). Ranked prediction of p53 targets using hidden variable dynamic modeling. *Genome biology*, 7(3):R25.

Baum, L. E., Petrie, T., Soules, G., and Weiss, N. (1970). A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains. *The Annals of Mathematical Statistics*, 41(1):164–171.

Bayes, T. and Price, R. (1763). An Essay towards Solving a Problem in the Doctrine of Chances. By the Late Rev. Mr. Bayes, F. R. S. Communicated by Mr. Price, in a Letter to John Canton, A. M. F. R. S. *Philosophical Transactions*, 53:370–418.

Beal, M. J., Ghahramani, Z., and Rasmussen, C. E. (2001). The Infinite Hidden Markov Model. In *Advances in neural information processing systems*, pages 577–584.

Berg, B. A. and Billoire, A. (2008). Markov Chain Monte Carlo Simulations. *Wiley Encyclopedia of Computer Science and Engineering*.

Bernardo, J. and Smith, A. (2009). *Bayesian Theory*. Wiley Series in Probability and Statistics. Wiley.

Bishop, C. M. (2007). *Pattern Recognition and Machine Learning*. Springer-Verlag, New York, Inc., New York, USA, 1st edition.

Blanche, T. J., Spacek, M. A., Hetke, J. F., and Swindale, N. V. (2005). Polytrodes: High-Density Silicon Electrode Arrays for Large-Scale Multiunit Recording. *Journal of Neurophysiology*, 93(5):2987–3000.

Blazek, R. B., Kim, H., Rozovskii, B., and Tartakovsky, A. (2001). A novel approach to detection of denial-of-service attacks via adaptive sequential and batch-sequential change-point detection methods. In *Proceedings of IEEE systems, man and cybernetics information assurance workshop*, pages 220–226. Citeseer.

Brooks, S., Gelman, A., Jones, G., and Meng, X. (2011). *Handbook of Markov Chain Monte Carlo*. Chapman & Hall/CRC Handbooks of Modern Statistical Methods. CRC Press.

Calderhead, B. and Girolami, M. (2009). Estimating Bayes factors via thermodynamic integration and population {MCMC} . *Computational Statistics & Data Analysis*, 53(12):4028 – 4045.

Casella, G. and Robert, C. P. (1996). Rao-Blackwellisation of sampling schemes. *Biometrika*, 83(1):81–94.

Chen, J. and Gupta, A. (2011). *Parametric Statistical Change Point Analysis: With Applications to Genetics, Medicine, and Finance*. SpringerLink : Bücher. Birkhäuser Boston.

Chiang, R., Liu, P., and Okunev, J. (1995). Modelling mean reversion of asset prices towards their fundamental value. *Journal of Banking & Finance*, 19(8):1327–1340.

Chiappalone, M., Novellino, A., Vajda, I., Vato, A., Martinoia, S., and van Pelt, J. (2005). Burst detection algorithms for the analysis of spatio-temporal patterns in cortical networks of neurons . *Neurocomputing*, 65–66(0):653–662.

Chib, S. (1998). Estimation and comparison of multiple change-point models. *Journal of Econometrics*, 86(2):221 – 241.

Chronopoulou, A. and Viens, F. (2012). Estimation and pricing under long-memory stochastic volatility. *Annals of Finance*, 8(2-3):379–403.

Cowles, M. K. and Carlin, B. P. (1996). Markov Chain Monte Carlo Convergence Diagnostics: A Comparative Review. *Journal of the American Statistical Association*, 91:883–904.

Cox, D. R. (1955). Some Statistical Methods Connected with Series of Events. *Journal of the Royal Statistical Society. Series B*, 17(2):129–164.

Cox, J. C., Ingersoll, Jonathan E., J., and Ross, S. A. (1985). A Theory of the Term Structure of Interest Rates. *Econometrica*, 53(2):385–407.

Crick, F. H. C., Barnett, L., Brenner, S., and Watts-Tobin, R. J. (1961). General Nature of the Genetic Code for Proteins. *Nature*, 192:1227–1232.

Dahlquist, G. and Björck, Å. (2008). *Numerical Methods in Scientific Computing: Volume 1*. SIAM e-books. Society for Industrial and Applied Mathematics (SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104).

Dahlquist, M. and Gray, S. F. (2000). Regime-switching and interest rates in the European monetary system . *Journal of International Economics*, 50(2):399 – 419.

Dale, A. (1991). Thomas Bayes: a biographical sketch. In *A History of Inverse Probability*, volume 16 of *Studies in the History of Mathematics and Physical Sciences*, pages 1–15. Springer US.

Dongarra, J. and Sullivan, F. (2000). Guest Editors' Introduction: The Top 10 Algorithms. *Computing in Science & Engineering*, 2(1):22–23.

Dowling, J. E. (1992). *Neurons and Networks: An Introduction to Neuroscience*. Belknap Press of Harvard University Press.

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

Dunlop, M., Cox, R., Levine, J., Murray, R., and Elowitz, M. (2008). Regulatory activity revealed by dynamic correlations in gene expression noise. *Nature Genetics*, 40:1493–1498.

Eldar, A. and Elowitz, M. B. (2010). Functional roles for noise in genetic circuits. *Nature*, 467(7312):167–173.

Elowitz, M. B., Levine, A. J., Siggia, E. D., and Swain, P. S. (2002). Stochastic gene expression in a single cell. *Science*, 297(5584):1129–1131.

Ewald, C.-O. and Wang, W.-K. (2010). Irreversible investment with Cox–Ingersoll–Ross type mean reversion . *Mathematical Social Sciences*, 59(3):314 – 318.

Fearnhead, P. (2006). Exact and efficient Bayesian inference for multiple changepoint problems. *Statistics and Computing*, 16(2):203–213.

Fearnhead, P. and Liu, Z. (2011). Efficient Bayesian analysis of multiple changepoint models with dependence across segments. *Statistics and Computing*, 21(2):217–229.

Fearnhead, P. and Sherlock, C. (2006). An exact Gibbs sampler for the Markov-modulated Poisson process. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(5):767–784.

Fernandes, H. C. and Weigel, M. (2011). Non-reversible Monte Carlo simulations of spin models. *Computer Physics Communications*, 182(9):1856–1859. Computer Physics Communications Special Edition for Conference on Computational Physics Trondheim, Norway, June 23-26, 2010.

Fischer, W. and Meier-Hellstern, K. (1993). The Markov-modulated Poisson process (MMPP) cookbook . *Performance Evaluation*, 18(2):149–171.

Fouque, J.-P., Papanicolaou, G., and Sircar, K. R. (2000). Mean-reverting stochastic volatility. *International Journal of theoretical and applied finance*, 3(01):101–142.

Fox, E., Sudderth, E., Jordan, M., and Willsky, A. (2011). Bayesian nonparametric inference of switching dynamic linear models. *Signal Processing, IEEE Transactions on*, 59(4):1569–1585.

Gamerman, D. and Lopes, H. F. (2006). *Markov Chain Monte Carlo: Stochastic Simulation for Bayesian Inference*. Chapman and Hall/CRC, 2nd edition.

Gance, P. (2014). Timeline: Fed's Bernanke saw U.S. economy through turbulent times. *Reuters*, 30rd January 2014. Available: http://www.reuters.com/article/2014/01/30/us-usa-fed-bernanke-timeline-idUSBREA0T0AL20140130 [Last accessed: 26th June 2015].

Gardiner, C. (2009). *Stochastic Methods: A Handbook for the Natural and Social Sciences*. Springer Series in Synergetics. Springer Berlin Heidelberg, 4 edition.

Gelenbe, E. (1979). On the Optimum Checkpoint Interval. *J. ACM*, 26(2):259–270.

Geman, S. and Geman, D. (1984). Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 6(6):721–741.

George, E. I., Makov, U. E., and Smith, A. F. M. (1993). Conjugate Likelihood Distributions. *Scandinavian Journal of Statistics*, 20(2):147–156.

Geyer, C. J. (1992). Practical Markov Chain Monte Carlo. *Statistical Science*, 7(4):473–483.

Ghahramani, Z. (2005). Nonparametric Bayesian methods. Tutorial presentation at the UAI Conference in Edinburgh, Scotland [Last accessed: 13th May 2015].

Gillespie, D. T. (1977). Exact Stochastic Simulation of Coupled Chemical Reactions. *The Journal of Physical Chemistry*, 81(25):2340–2361.

Giordani, P. and Kohn, R. (2008). Efficient Bayesian inference for multiple change-point and mixture innovation models. *Journal of Business & Economic Statistics*, 26(1).

Glasserman, P. (2003). *Monte Carlo Methods in Financial Engineering*. Applications of mathematics. Springer.

Goggans, P. M. and Chi, Y. (2004). Using thermodynamic integration to calculate the posterior probability in Bayesian model selection problems. In *AIP Conference Proceedings*, volume 707, pages 59–66. IOP INSTITUTE OF PHYSICS PUBLISHING LTD.

Gourévitch, B. and Eggermont, J. J. (2007). A nonparametric approach for detection of bursts in spike trains. *Journal of Neuroscience Methods*, 160(2):349–358.

Gowrishankar, J. and Harinarayanan, R. (2004). Why is transcription coupled to translation in bacteria? *Molecular Microbiology*, 54(3):598–603.

Grande, R. C. (2014). *Computationally efficient Gaussian Process changepoint detection and regression*. PhD thesis, Massachusetts Institute of Technology.

Grassmann, W. (1977). Transient solutions in Markovian queueing systems. *Computers & Operations Research*, 4(1):47–53.

Green, P. J. (1995). Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82(4):711–732.

Grewal, M. and Andrews, A. (2011). *Kalman Filtering: Theory and Practice Using MATLAB*. Wiley.

Harbison, C. T., Gordon, D. B., Lee, T. I., Rinaldi, N. J., Macisaac, K. D., Danford, T. W., Hannett, N. M., Tagne, J.-B., Reynolds, D. B., Yoo, J., et al. (2004). Transcriptional regulatory code of a eukaryotic genome. *Nature*, 431(7004):99–104.

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

Herrmann, J. (2014). Bayesian Inference for a Cox-Ingersoll-Ross Model with changing Parameters and Application to Finance Data. Master thesis, TU Berlin.

Heuer, A., Müller, C., and Rubner, O. (2010). Soccer: Is scoring goals a predictable Poissonian process? *EPL (Europhysics Letters)*, 89(3):38007.

Hodgkin, A. L. and Huxley, A. F. (1939). Action potentials recorded from inside a nerve fibre. *Nature*, 144(3651):710–711.

Hoel, P., Port, S., and Stone, C. (1972). *Introduction to Stochastic Processes*. The Houghton Mifflin Series in Statistics. Houghton Mifflin Comp.

Honerkamp, J. (1994). *Stochastic dynamical systems: concepts, numerical methods, data analysis*. VCH.

Hubel, D. H. and Wiesel, T. N. (1959). Receptive fields of single neurones in the cat's striate cortex. *The Journal of physiology*, 148(3):574–591.

Huttenlocher, P. R. (1967). Development of cortical neuronal activity in the neonatal cat. *Experimental Neurology*, 17(3):247–262.

Irving, G. (2007). *Methods for the physically based simulation of solids and fluids*. PhD thesis, Stanford University.

Jacobsen, M. (2006). *Point Process Theory and Applications: Marked Point and Piecewise Deterministic Processes*. Probability and Its Applications. Birkhäuser Boston.

Jarrett, R. (1979). A note on the intervals between coal-mining disasters. *Biometrika*, 66(1):191–193.

Jaynes, E. and Bretthorst, G. (2003). *Probability Theory: The Logic of Science*. Cambridge University Press.

Jeffreys, H. (1935). Some Tests of Significance, Treated by the Theory of Probability. *Mathematical Proceedings of the Cambridge Philosophical Society*, 31:203–222.

Jeffreys, H. (1998). *The Theory of Probability*. OUP Oxford.

Jenkins, D. J., Finkenstädt, B., and Rand, D. A. (2013). A temporal switch model for estimating transcriptional activity in gene expression. *Bioinformatics*, 29(9):1158–1165.

Jensen, C. S. and Kong, A. (1995). Blocking Gibbs Sampling in Very Large Probabilistic Expert Systems. *Internat. J. Human–Computer Studies*, 42:647–666.

Jongbloed, G., Van Der Meulen, F. H., Van Der Vaart, A. W., et al. (2005). Nonparametric inference for Lévy-driven Ornstein-Uhlenbeck processes. *Bernoulli*, 11(5):759–791.

Kalos, M. and Whitlock, P. (2008). *Monte Carlo Methods*. Wiley.

Kandel, E., Schwartz, J., and Jessell, T. (2000). *Principles of neural science*. McGraw-Hill, Health Professions Division, fourth edition.

Kaneoke, Y. and Vitek, J. (1996). Burst and oscillation as disparate neuronal properties. *Journal of Neuroscience Methods*, 68(2):211–223.

Kass, R. E. and Raftery, A. E. (1995). Bayes Factors. *Journal of the American Statistical Association*, 90(430):773–795.

Kass, R. E. and Ventura, V. (2001). A Spike-Train Probability Model. *Neural Computation*, 13(8):1713–1720.

Kass, R. E., Ventura, V., and Brown, E. N. (2005). Statistical Issues in the Analysis of Neuronal Data. *J Neurophysiol*, 94(1):8–25.

Kendall, W., Liang, F., and Wang, J. (2005). *Markov Chain Monte Carlo: Innovations and Applications*. Institute for Mathematical Sciences lecture notes series. World Scientific.

Kleinsmith, L. and Kish, V. (1995). *Principles of cell and molecular biology*. HarperCollins, 2 edition.

Kloeden, P. and Platen, E. (1992). *Numerical Solution of Stochastic Differential Equations*. Applications of mathematics : stochastic modelling and applied probability. Springer.

Ko, S. I. M., Chong, T. T. L., and Ghosh, P. (2015). Dirichlet Process Hidden Markov Multiple Change-point Model. *Bayesian Anal.*, 10(2):275–296.

Koop, G. and Potter, S. M. (2009). Prior Elicitation In Multiple Change-Point Models. *International Economic Review*, 50(3):751–772.

Kou, S. C., Sunney Xie, X., and Liu, J. S. (2005). Bayesian analysis of single-molecule experimental data. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 54(3):469–506.

Kruschke, J. (2011). *Doing Bayesian Data Analysis: A Tutorial with R and BUGS*. Academic Press.

Lafuerza, L. F. and Toral, R. (2011). Exact solution of a stochastic protein dynamics model with delayed degradation. *Phys. Rev. E*, 84:051121.

Last, G. and Brandt, A. (1995). *Marked Point Processes on the Real Line: The Dynamical Approach*. Graduate Texts in Mathematics. Springer.

Latchman, D. S. (1997). Transcription factors: An overview. *The International Journal of Biochemistry & Cell Biology*, 29(12):1305 – 1312.

L'Ecuyer, P. and Owen, A. B. (2010). *Monte Carlo and Quasi-Monte Carlo Methods 2008*. Mathematics and Statistics. Springer Berlin Heidelberg.

Lee, T. I., Rinaldi, N. J., Robert, F., Odom, D. T., Bar-Joseph, Z., Gerber, G. K., Hannett, N. M., Harbison, C. T., Thompson, C. M., Simon, I., Zeitlinger, J., Jennings, E. G., Murray, H. L., Gordon, D. B., Ren, B., Wyrick, J. J., Tagne, J.-B., Volkert, T. L., Fraenkel, E., Gifford, D. K., and Young, R. A. (2002). Transcriptional Regulatory Networks in Saccharomyces cerevisiae. *Science*, 298(5594):799–804.

Legéndy, C. R. and Salcman, M. (1985). Bursts and recurrences of bursts in the spike trains of spontaneously active striate cortex neurons. *Journal of Neurophysiology*, 53(4):926–939.

Link, W. A. and Eaton, M. J. (2012). On thinning of chains in MCMC. *Methods in Ecology and Evolution*, 3(1):112–115.

Liu, J. S. (2008). *Monte Carlo strategies in scientific computing*. Springer.

Lo, A. W. and MacKinlay, A. C. (1988). Stock Market Prices Do Not Follow Random Walks: Evidence from a Simple Specification Test. *Review of Financial Studies*, 1(1):41–66.

MacKay, D. J. C. (2002). *Information Theory, Inference & Learning Algorithms*. Cambridge University Press, New York, NY, USA.

Malham, S. J. A. and Wiese, A. (2013). Chi-Square Simulation of the CIR Process and the Heston Model. *International Journal of Theoretical and Applied Finance*, 16(03):1350014.

Manly, B. F. (2006). *Randomization, Bootstrap and Monte Carlo Methods in Biology, Third Edition*. Chapman & Hall texts in statistical science series. Taylor & Francis.

Marsh, T. A. and Rosenfeld, E. R. (1983). Stochastic Processes for Interest Rates and Equilibrium Bond Prices. *The Journal of Finance*, 38(2):635–646.

Meng, X.-L. and Wong, W. H. (1996). Simulating Ratios of Normalizing Constants via a Simple Identity: a Theoretical Exploration. *Statistica Sinica*, pages 831–860.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6):1087–1092.

Metropolis, N. and Ulam, S. M. (1949). The Monte Carlo Method. *Journal of the American Statistical Association*, 44(247):335–341.

Moore, H. and Roberts, D. (2013). AP Twitter hack causes panic on Wall Street and sends Dow plunging. *The Guardian*, 23rd April 2013. Available: http://www.theguardian.com/business/2013/apr/23/ap-tweet-hack-wall-street-freefall [Last accessed: 11th May 2015].

Morita, T. and Sugano, S. (1995). Design and development of a new robot joint using a mechanical impedance adjuster. In *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on*, volume 3, pages 2469–2475. IEEE.

Nawrot, M., Aertsen, A., and Rotter, S. (1999). Single-trial estimation of neuronal firing rates: From single-neuron spike trains to population activity. *Journal of Neuroscience Methods*, 94:81–92.

Nelder, J. A. and Wedderburn, R. W. M. (1972). Generalized Linear Models. *Journal of the Royal Statistical Society. Series A (General)*, 135(3):370–384.

Nelson, P. C., Smith, Z. M., and Young, E. D. (2009). Wide-Dynamic-Range Forward Suppression in Marmoset Inferior Colliculus Neurons Is Generated Centrally and Accounts for Perceptual Masking. *J. Neurosci.*, 29(8):2553–2562.

Neuts, M. F. (1979). A versatile Markovian point process. *Journal of Applied Probability*, pages 764–779.

Ocone, A. (2013). *Variational inference for Gaussian-jump processes with application in gene regulation*. PhD thesis, The University of Edinburgh.

Ocone, A. and Sanguinetti, G. (2011). Reconstructing transcription factor activities in hierarchical transcription network motifs. *Bioinformatics*, 27(20):2873–2879.

Ocone, A. and Sanguinetti, G. (2013). A stochastic hybrid model of a biological filter. *arXiv preprint arXiv:1308.5338*.

Onalan, O. (2009). Financial modelling with Ornstein-Uhlenbeck processes driven by Lévy process. In *Proceedings of the World Congress on Engineering*, volume 2, pages 1–3.

Opper, M., Ruttor, A., and Sanguinetti, G. (2010). Approximate inference in continuous time Gaussian-Jump processes. In Lafferty, J., Williams, C., Shawe-Taylor, J., Zemel, R., and Culotta, A., editors, *Advances in Neural Information Processing Systems 23*, pages 1831–1839. Curran Associates, Inc.

Opper, M. and Sanguinetti, G. (2010). Learning combinatorial transcriptional dynamics from gene expression data. *Bioinformatics*, 26(13):1623–1629.

Page, E. (1954). Continuous Inspection Schemes. *Biometrika*, pages 100–115.

Paraschos, A., Daniel, C., Peters, J., and Neumann, G. (2013). Probabilistic movement primitives. In *Advances in Neural Information Processing Systems*, pages 2616–2624.

Perkel, D. H., Gerstein, G. L., and Moore, G. P. (1967). Neuronal spike trains and stochastic point processes. I. The single spike train. *Biophysical Journal*, 7(4):391–418.

Petersen, K. B. and Pedersen, M. S. (2012). The Matrix Cookbook. Version 20121115.

Pillow, J. W., Ahmadian, Y., and Paninski, L. (2011). Model-based decoding, information estimation, and change-point detection techniques for multineuron spike trains. *Neural Computation*, 23(1):1–45.

Pitman, J. and Picard, J. (2006). *Combinatorial Stochastic Processes*. Combinatorial Stochastic Processes: École D'Été de Probabilités de Saint-Flour XXXII - 2002. Springer.

Preis, T., Schneider, J. J., and Stanley, H. E. (2011). Switching processes in financial markets. *Proceedings of the National Academy of Sciences*, 108(19):7674–7678.

Priebe, N. J. and Ferster, D. (2008). Inhibition, spike threshold, and stimulus selectivity in primary visual cortex. *Neuron*, 57(4):482–497.

Putzky, P., Franzen, F., Bassetto, G., and Macke, J. H. (2014). A Bayesian model for identifying hierarchically organised states in neural population activity. In *Advances in Neural Information Processing Systems*, pages 3095–3103.

Qian, S. S., King, R. S., and Richardson, C. J. (2003). Two statistical methods for the detection of environmental thresholds. *Ecological Modelling*, 166(1):87–97.

Quick, H., Holan, S. H., Wikle, C. K., and Reiter, J. P. (2014). Bayesian Marked Point Process Modeling for Generating Fully Synthetic Public Use Data with Point-Referenced Geography. *arXiv preprint arXiv:1407.7795*.

Raftery, A. E., Newton, M. A., Satagopan, J. M., and Krivitsky, P. N. (2007). Estimating the integrated likelihood via posterior simulation using the harmonic mean identity. *Bayesian Statistics*, 8:1–45.

Rao, V. and Teh, Y. W. (2011). Fast MCMC sampling for Markov jump processes and continuous time Bayesian networks. In *Proceedings of the International Conference on Uncertainty in Artificial Intelligence*.

Rao, V. and Teh, Y. W. (2013). Fast MCMC Sampling for Markov Jump Processes and Extensions. *J. Mach. Learn. Res.*, 14(1):3295–3320.

Reeves, J., Chen, J., Wang, X. L., Lund, R., and Lu, Q. Q. (2007). A review and comparison of changepoint detection techniques for climate data. *Journal of Applied Meteorology and Climatology*, 46(6):900–915.

Ricciardi, L. M. and Sacerdote, L. (1979). The Ornstein-Uhlenbeck process as a model for neuronal activity. *Biological Cybernetics*, 35(1):1–9.

Roberts, G. O., Gelman, A., and Gilks, W. R. (1997). Weak convergence and optimal scaling of random walk Metropolis algorithms. *Ann. Appl. Probab.*, 7(1):110–120.

Ross, S. M. (1983). *Stochastic Processes.* John Wiley & Sons.

Rotondi, R. (2002). On the influence of the proposal distributions on a reversible jump MCMC algorithm applied to the detection of multiple change-points. *Computational Statistics & Data Analysis*, 40(3):633–653.

Rubinstein, R. Y. and Kroese, D. P. (2008). *Simulation and the Monte Carlo Method.* Wiley Series in Probability and Statistics. Wiley.

Ruttor, A., Sanguinetti, G., and Opper, M. (2009). Approximate inference for stochastic reaction processes. In *Learning and Inference in Computational Systems Biology*, pages 189–205. MIT Press.

Rydén, T. (1996). An EM algorithm for estimation in Markov-modulated Poisson processes. *Computational Statistics & Data Analysis*, 21(4):431 – 447.

Sahai, H. and Ojeda, M. M. (2003). A comparison of approximations to percentiles of the noncentral chi2-distribution. *Revista de Matemática: Teoría y Aplicaciones*, 10(1-2):57–76.

Salvador, P., Valadas, R., and Pacheco, A. (2003). Multiscale Fitting Procedure Using Markov Modulated Poisson Processes. *Telecommunication Systems*, 23(1-2):123–148.

Sanguinetti, G., Ruttor, A., Opper, M., and Archambeau, C. (2009). Switching regulatory models of cellular stress response. *Bioinformatics*, 25(10):1280–1286.

Scargle, J. D. (1998). Studies in Astronomical Time Series Analysis. V. Bayesian Blocks, a New Method to Analyze Structure in Photon Counting Data. *The Astrophysical Journal*, 504(1):405.

Schawalder, S. B., Kabani, M., Howald, I., Choudhury, U., Werner, M., and Shore, D. (2004). Growth-regulated recruitment of the essential yeast ribosomal protein gene activator Ifh1. *Nature*, 432(7020):1058–1061.

Scott, S. L. (1999). Bayesian Analysis of a Two-State Markov Modulated Poisson Process. *Journal of Computational and Graphical Statistics*, 8(3):662–670.

Scott, S. L. and Smyth, P. (2003). The Markov modulated Poisson process and Markov Poisson cascade with applications to web traffic data. In Bayarri, M. J., Berger, J. O., Bernardo, J. M., Dawid, A. P., Heckerman, D., Smith, A. F. M., and West, M., editors, *Bayesian Statistics 7, Proceedings of the Seventh Valencia International Meeting*, pages 671–680. Oxford University Press.

Shahrezaei, V. and Swain, P. (2008). The stochastic nature of biochemical networks. *Curr. Opin. in Biotech.*, 19(4):369–374.

Sherlock, C. (2006). *Methodology for inference on the Markov modulated Poisson process and theory for optimal scaling of the random walk Metropolis.* PhD thesis, Lancaster University.

Solomon, J. M. and Grossman, A. D. (1996). Who's competent and when: regulation of natural genetic competence in bacteria. *Trends in Genetics*, 12(4):150 – 155.

Sorger, P. K. (1991). Heat shock factor and the heat shock response. *Cell*, 65(3):363–366.

Steele, J. (2001). *Stochastic Calculus and Financial Applications*. Applications of mathematics : stochastic modelling and applied probability. Springer.

Stimberg, F., Opper, M., Sanguinetti, G., and Ruttor, A. (2011a). Inference in continuous-time change-point models. In Shawe-Taylor, J., Zemel, R., Bartlett, P., Pereira, F., and Weinberger, K., editors, *Advances in Neural Information Processing Systems 24*, pages 2717–2725. Curran Associates, Inc.

Stimberg, F., Ruttor, A., and Opper, M. (2011b). Bayesian Inference for Models of Transcriptional Regulation Using Markov Chain Monte Carlo Sampling. In Koeppl, H., Aćimović, J., Kesseli, J., Mäki-Marttunen, T., Larjo, A., and Yli-Harja, O., editors, *Proceedings of the 8th International Workshop on Computational Systems Biology (WCSB)*, TICSP series # 57, pages 169—-172, Zürich, Switzerland. Tampere University of Technology, Tampere, Finland.

Stimberg, F., Ruttor, A., and Opper, M. (2012). Bayesian Inference for Change Points in Dynamical Systems with Reusable States - a Chinese Restaurant Process Approach. In Lawrence, N. D. and Girolami, M. A., editors, *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics (AISTATS-12)*, volume 22, pages 1117–1124.

Stimberg, F., Ruttor, A., and Opper, M. (2014). Poisson Process Jumping between an Unknown Number of Rates: Application to Neural Spike Data. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K., editors, *Advances in Neural Information Processing Systems 27*, pages 730–738. Curran Associates, Inc.

Stirzaker, D. (2005). *Stochastic Processes and Models*. Stochastic Processes and Models. Oxford University Press.

Stramer, O., Bognar, M., and Schneider, P. (2010). Bayesian Inference for Discretely Sampled Markov Processes with Closed-Form Likelihood Expansions. *Journal of Financial Econometrics*, 8(4):450–480.

Suël, G. M., Garcia-Ojalvo, J., Liberman, L. M., and Elowitz, M. B. (2006). An excitable gene regulatory circuit induces transient cellular differentiation. *Nature*, 440(7083):545–550.

Teh, Y. W. (2010). Dirichlet Processes. In *Encyclopedia of Machine Learning*. Springer.

Teh, Y. W., Jordan, M. I., Beal, M. J., and Blei, D. M. (2006). Hierarchical dirichlet processes. *Journal of the american statistical association*, 101(476).

Tierney, L. (1994). Markov Chains for Exploring Posterior Distributions. *The Annals of Statistics*, 22(4):1701–1728.

Tokdar, S., Xi, P., Kelly, R., and Kass, R. (2010). Detection of bursts in extracellular spike trains using hidden semi-Markov point process models. *Journal of Computational Neuroscience*, 29(1-2):203–212.

Tu, B. P., Kudlicki, A., Rowicka, M., and McKnight, S. L. (2005). Logic of the yeast metabolic cycle: temporal compartmentalization of cellular processes. *Science (New York, N.Y.)*, 310(5751):1152–8.

Turgay, K., Hamoen, L. W., Venema, G., and Dubnau, D. (1997). Biochemical characterization of a molecular switch involving the heat shock protein ClpC, which controls the activity of ComK, the competence transcription factor of Bacillus subtilis. *Genes & Development*, 11(1):119–128.

Uhlenbeck, G. E. and Ornstein, L. S. (1930). On the Theory of the Brownian Motion. *Phys. Rev.*, 36:823–841.

Van Kampen, N. (2011). *Stochastic Processes in Physics and Chemistry*. North-Holland Personal Library. Elsevier Science.

Varela, J. A., Sen, K., Gibson, J., Fost, J., Abbott, L. F., and Nelson, S. B. (1997). A Quantitative Description of Short-Term Plasticity at Excitatory Synapses in Layer 2/3 of Rat Primary Visual Cortex. *The Journal of Neuroscience*, 17(20):7926–7940.

von Toussaint, U. (2011). Bayesian inference in physics. *Rev. Mod. Phys.*, 83:943–999.

Vyshemirsky, V. and Girolami, M. A. (2008). Bayesian ranking of biochemical system models. *Bioinformatics*, 24(6):833–839.

Wakefield, J. (2007). A Bayesian Measure of the Probability of False Discovery in Genetic Epidemiology Studies. *The American Journal of Human Genetics*, 81(2):208–227.

Wang, H., Zhang, D., and Shin, K. (2004). Change-point monitoring for the detection of DoS attacks. *Dependable and Secure Computing, IEEE Transactions on*, 1(4):193–208.

Wang, M.-C., Qin, J., and Chiang, C.-T. (2001). Analyzing Recurrent Event Data With Informative Censoring. *Journal of the American Statistical Association*, 96(455):1057–1065.

Wang, X., Lu, T., Snider, R. K., and Liang, L. (2005). Sustained firing in auditory cortex evoked by preferred stimuli. *Nature*, 435(7040):341–346.

Wilkinson, D. (2006). *Stochastic modelling for systems biology*. Chapman and Hall/CRC mathematical and computational biology series. Chapman and Hall/CRC, Boca Raton, London, New York.

Williams, R. and Lawrence, D. (2007). *Linear State-Space Control Systems*. John Wiley & Sons.

Wyse, J., Friel, N., et al. (2011). Approximate simulation-free Bayesian inference for multiple changepoint models with dependence within segments. *Bayesian Analysis*, 6(4):501–528.

Xu, Z., Todorov, E., Dellon, B., and Matsuoka, Y. (2011). Design and analysis of an artificial finger joint for anthropomorphic robotic hands. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 5096–5102. IEEE.

Yoshihara, T., Kasahara, S., and Takahashi, Y. (2001). Practical Time-Scale Fitting of Self-Similar Traffic with Markov-Modulated Poisson Process. *Telecommunication Systems*, 17(1-2):185–211.

Zhao, M. and Xie, M. (1996). On Maximum Likelihood Estimation for a General Non-Homogeneous Poisson Process. *Scandinavian Journal of Statistics*, 23(4):597–607.

Appendix A

Detailed Calculations

A.1 Exact sampling of the Ornstein-Uhlenbeck-Process

Given the parameters Θ and the μ-process we can draw samples from the posterior distribution of X directly. The posterior is a Markov process with a time dependent drift Archambeau et al. (2008). The backward equation is solved to compute the posterior drift, which is then used to sample a new discrete path for X.

A.1.1 Backward Equation

We define

$$\psi_{t_i}(X) \triangleq P(\{X_{t>t_i}\}|X_{t_i} = X) \tag{A.1}$$

and use the backward equation adapted to the switching model by Sanguinetti et al. (2009), which simplifies to

$$\frac{\partial \psi}{\partial t} = -\frac{\partial \psi}{\partial X}(A\mu + b - \lambda X), \tag{A.2}$$

because we sample X given a path of μ. With the starting condition $\psi_T(X) = \mathcal{N}(d_k, s^2)$ this can be solved backwards in time by

$$\begin{aligned}\psi_{t_i}(X) &= \mathcal{N}(X; m(t_i), v(t_i)) \\ m(t_i) &= m(t_{i+1})\exp(-\lambda\Delta t) + \frac{A\mu_{t_i} + b}{\lambda}(1 - \exp(-\lambda\Delta t)) \\ v(t_i) &= v(t_{i+1})\exp(-2\lambda\Delta t) + \frac{\sigma^2}{2\lambda}(1 - \exp(-2\lambda\Delta t))\end{aligned} \tag{A.3}$$

with $\Delta t = t_i - t_{i+1}$. Additionally $\psi_t(X)$ has the following jump conditions at the measurements:

$$\begin{aligned} m(t_i^-) &= \frac{v(t_i^-)}{v(t_i^+)} m(t_i^+) + \frac{v(t_i^-)}{s^2} d_{t_i} \\ v(t_i^-) &= \frac{v(t_i^+) s^2}{v(t_i^+) + s^2}, \end{aligned} \tag{A.4}$$

where t_i^- and t_i^+ are the times before and after the measurement, respectively.

Clearly this only holds if μ is constant between t_i and t_{i+1}. If this is not the case, the variance is not affected, but the mean has to be computed iteratively at all the jump times $t_{i,j}$, where μ changes states, by applying

$$m(t_{i,j}) = m(t_{i,j+1}) \exp(-\lambda \Delta t) + \frac{A\mu_{t_{i,j}} + b}{\lambda}(1 - \exp(-\lambda \Delta t_{i,j})), \tag{A.5}$$

with $\Delta t_{i,j} = t_{i,j} - t_{i,j+1}$.

A.1.2 Posterior Drift

Ruttor et al. (2009) showed that the posterior drift $\tau(X,t)$ can be computed with the solution to the backwards equation by applying

$$\begin{aligned} \tau(X,t) &= A\mu_t + b - \lambda X + \sigma^2 \frac{d}{dX} \ln(\psi_t(X)) \\ &= A\mu_t + b - \lambda X + \sigma^2 (\frac{m(t) - X}{v(t)}) \end{aligned}$$

The posterior process is now an Ornstein-Uhlenbeck-process with time-dependent drift $\tau(X,t)$ and diffusion σ^2. Given a starting condition we can iteratively sample $X_{t_{i+1}}$ from

$$P_{post}(X_{t_{i+1}} | X_{t_i}, \mu, \Theta, \mathbf{D}) = \mathcal{N}(X_{t_{i+1}}; m_f, v_f), \tag{A.6}$$

where m_f and v_f are solutions to

$$\begin{aligned} \frac{dm_f}{dt} &= \tau(m_f(t), t), m_f(t_i) = X_{t_i} \\ \frac{dv_f}{dt} &= 2v_f(t)(-\lambda - \frac{\sigma^2}{v(t)}) + \sigma^2, v_f(t_i) = 0. \end{aligned} \tag{A.7}$$

This can be done analytically and therefore we can get exact samples from X, independent of the time step Δt.

A.2 Posterior Transition Rates of the Telegraph process

As shown by Sanguinetti et al. (2009) the transition rates of the posterior process are given by

$$\begin{aligned} g_+(t_i) &= f_+ P(X_{t_{i+1}}|X_{t_i}, \mu = 1, \Theta) \frac{\psi_{t_{i+1}}(1)}{\psi_{t_i}(0)} \\ g_-(t_i) &= f_- P(X_{t_{i+1}}|X_{t_i}, \mu = 0, \Theta) \frac{\psi_{t_{i+1}}(0)}{\psi_{t_i}(1)} \end{aligned} \tag{A.8}$$

where the marginal likelihood $\psi_t(\mu)$ is defined as

$$\psi_{t_i}(u) \overset{\wedge}{=} P(\{X_{t>t_i}\}|\mu(t_i) = u). \tag{A.9}$$

As in section A.1 we assume that, between two data points, ψ satisfies the backward equation from Sanguinetti et al. (2009)

$$\frac{\partial \psi(u)}{\partial t} = \sum_{u' \neq u} f(u'|u) \left(\psi_t(u) - \psi_t(u')\right), \tag{A.10}$$

which conditioned on a specific path for X is solved backwards in time by

$$\psi_{t_i}(u) = P(X_{t_{i+1}}|X_{t_i}, \mu(t_i) = u, \Theta) \sum_{u'} \exp(-f(u'|u)\Delta t) \psi_{t_{i+1}}(u')) \tag{A.11}$$

with $\Delta t = t_{i+1} - t_i$ and $P(X_{t_{i+1}}|X_{t_i}, \mu, \Theta)$ as in (4.6) and the starting values $\psi_T(u) = 1 \forall u$.

Because the values of ψ are often extremely small or large it is best to work with the logarithm of ψ:

$$\begin{aligned} \ln(\psi_{t_i}(u)) = {} & \ln(P(X_{t_{i+1}}|X_{t_i}, \mu = u, \Theta)) + \ln(\psi_{t_{i+1}}(u)) \\ & + \ln \sum_{u' \neq u} \Big\{ \left[1 - \exp(-f(u'|u)\Delta t)\right] \exp\left[\ln(\psi_{t_{i+1}}(u')) - \ln(\psi_{t_{i+1}}(u))\right] \\ & + \exp\Big(- \sum_{u' \neq u} f(u'|u)\Delta t\Big) \Big\} \end{aligned} \tag{A.12}$$

Using ψ we can compute the posterior transition rates according to (A.8) and sample a new continuous time path for μ using a modified Gillespie algorithm (Gillespie, 1977).

Since the posterior transition rates are only piecewise constant, we draw the time for the next state change using the current transition rate. If that time is outside the time interval of the current transition rate, nothing happens in this interval and the process is repeated from the start of the new interval.

A.3 Sampling A and b Directly

We describe the computation for A, the computation for b is analogous.

For the likelihood we iteratively computed the mean and variance of a Gaussian, starting from the first observations. The mean of the Gaussian had the form $m+m_A A$ and therefore we compute m and m_A through the process and update it at the observations.

We initialize our mean and variance to

$$m = d_1 \tag{A.13}$$
$$m_A = 0 \tag{A.14}$$
$$v = \sigma_{obs}. \tag{A.15}$$

When there are no jumps between observations we compute

$$\alpha = \exp(-\lambda \Delta t) \tag{A.16}$$
$$\beta = \frac{b(1-\alpha)}{\lambda} \tag{A.17}$$
$$\xi = (1-\alpha^2)\frac{\sigma^2}{2\lambda}, \tag{A.18}$$

where Δt is the difference between the observations.

Then at the observations we update our temporary variables

$$m_{tmp} = \alpha m + \beta \tag{A.19}$$
$$m_{tmpA} = \alpha m_A + \beta \mu(t) \tag{A.20}$$
$$v_{tmp} = \xi + \alpha^2 v \tag{A.21}$$

and our mean and variance

$$m = \frac{\sigma_{obs}^2 m_{tmp} + v_{tmp} d_i}{\sigma_{obs}^2 + v_{tmp}} \tag{A.22}$$

$$m_A = \frac{\sigma_{obs}^2 m_{tmpA}}{\sigma_{obs}^2 + v_{tmp}} \tag{A.23}$$

$$v = \frac{\sigma_{obs}^2 v_{tmp}}{\sigma_{obs}^2 + v_{tmp}} \tag{A.24}$$

and multiply our current Gaussian over A with

$$\mathcal{N}\left(\frac{d_i - m_{tmp}}{m_{tmpA}}; \frac{\sigma_{obs}^2 + v_{tmp}}{m_{tmpA}^2}\right) \tag{A.25}$$

If we choose a Gaussian prior over b and A then their posterior densities will be Gaussian as well and we can directly sample from them.

A.4 Computing Bayes Factors for the Switching Model

We want to estimate the Bayes factor for a switching model with a fixed jump rate $f = F$. We can fully describe a path $\mu_{0:T}$ by the number of jumps and the time of the jumps:

$$\mu_{0:T} = (c, \tau_1, \ldots, \tau_c). \tag{A.26}$$

In order to get the Bayes factors we calculate the evidence as a function of f:

$$\begin{aligned} P(D|f) &= \int\int P(D|\mu_{0:T}, \theta, f) P(\mu_{0:T}|, f) P(\theta) d\mu_{0:T} d\theta \\ &= \int\int P(D|\mu_{0:T}, \theta, f) f^c e^{-fT} P(\theta) d\mu_{0:T} d\theta, \end{aligned} \tag{A.27}$$

where $\theta = (A, b, \lambda, \sigma^2)$ for the model without switching diffusion and $\theta = (A, b, \lambda, \sigma_0^2, \sigma_1^2)$ for the model with switching diffusion. We calculate the derivative of the evidence with respect to the jump rate f

$$\begin{aligned}
\frac{dP(D|f)}{df} &= \int\int P(D|\mu_{0:T},\theta,f)P(\mu_{0:T}|f)P(\theta)d\mu_{0:T}d\theta \\
&= \int\int P(D|\mu,\theta,f)\frac{df^c e^{-fT}}{df}P(\theta)d\mu_{0:T}d\theta \\
&= \int\int P(D|\mu_{0:T},\theta,f)(cf^{c-1}e^{-fT} - Tf^c e^{-fT})P(\theta)d\mu_{0:T}d\theta \\
&= \int\int P(D|\mu_{0:T},\theta,f)(f^{c-1}e^{-fT})(c-fT)P(\theta)d\mu_{0:T}d\theta \\
&= \int\int P(D|\mu_{0:T},\theta,f)(f^c e^{-fT})(\frac{c}{f}-T)P(\theta)d\mu_{0:T}d\theta \\
&= \int\int P(D|\mu_{0:T},\theta,f)P(\mu_{0:T}|f)P(\theta)(\frac{c}{f}-T)d\mu_{0:T}d\theta \qquad \text{(A.28)} \\
&= \int\int P(D,\mu_{0:T},\theta|f)(\frac{c}{f}-T)d\mu_{0:T}d\theta \\
&= \int\int P(D|f)P(\mu_{0:T},\theta|D,f)(\frac{c}{f}-T)d\mu_{0:T}d\theta \\
&= P(D|f)\int\int P(\mu_{0:T},\theta|D,f)(\frac{c}{f}-T)d\mu_{0:T}d\theta \\
&= P(D|f)E_{P(\mu_{0:T},\theta|D,f)}\left[\frac{c}{f}-T\right] \\
&= P(D|f)\left(E_{P(\mu_{0:T},\theta|D,f)}\left[\frac{c}{f}\right]-T\right)
\end{aligned}$$

In summary we know that

$$\begin{aligned}
\frac{dP(D|f)}{df} &= P(D|f)\left(E_{P(\mu_{0:T},\theta|D,f)}\left[\frac{c}{f}\right]-T\right) \\
\Leftrightarrow \frac{1}{P(D|f)}\frac{dP(D|f)}{df} &= E_{P(\mu_{0:T},\theta|D,f)}\left[\frac{c}{f}\right]-T \qquad \text{(A.29)} \\
\Leftrightarrow \frac{d\log P(D|f)}{df} &= E_{P(\mu_{0:T},\theta|D,f)}\left[\frac{c}{f}\right]-T
\end{aligned}$$

We integrate both sides over $f = f'$ from 0 to F

$$\begin{aligned}
\Leftrightarrow \int_0^F \frac{d\log P(D|f=f')}{df'}df' &= \int_0^F \left(E_{P(\mu_{0:T},\theta|D,f=f')}\left[\frac{c}{f'}\right]-T\right)df' \\
\Leftrightarrow \left[\log P(D|f=f')\right]_0^F &= \int_0^F \left(E_{P(\mu_{0:T},\theta|D,f=f')}\left[\frac{c}{f'}\right]\right)df' - TF \qquad \text{(A.30)} \\
\Leftrightarrow \log P(D|f=F) - \log P(D|f=0) &= \int_0^F \left(E_{P(\mu_{0:T},\theta|D,f=f')}\left[\frac{c}{f'}\right]\right)df' - TF
\end{aligned}$$

and get the following equation to compute the desired log evidence for a fixed jump rate of $f = F$:

$$\log P(D|f = F) = \int_0^F \left(E_{P(\mu_{0:T},\theta|D,f=f')} \left[\frac{c}{f'} \right] \right) df' - TF + \log P(D|f = 0). \tag{A.31}$$

If we are calculating Bayes factors we are only looking at differences between log evidences for different models, therefore $-TF$ can be ignored as it does not depend on the model. The same is true for $\log P(D|f = 0)$ as with $f = 0$ there will be no jumps and the models we are comparing only differ when jumps exist.

To approximate the integral on the right side we let the sampler run for different values of f and from the samples compute $E_{P(\mu_{0:T},\theta|D,f)} \left[\frac{c}{f} \right]$ as the posterior mean number of jumps divided by f. We do not formally prove that the expection does not diverge for $f \to 0$ but this is evident when looking at the results in figure 4.5a.

A.5 Multivariate Ornstein-Uhlenbeck Process Likelihood

We have a m-dimensional multivariate Ornstein-Uhlenbeck Process which is defined by the SDE

$$d\mathbf{X} = (\mathbf{B} - \Lambda X)dt + \mathbf{\Sigma} d\mathbf{W}, \tag{A.32}$$

with $\mathbf{B}$ being a m-dimensional column vector and Λ and $\mathbf{\Sigma}$ being $m \times m$ matrices.

A.5.1 Transition Density

To compute the likelihood we need the transition density of the process which solves (A.32). In order to get the mean of the transitional density we first solve the ODE

$$\frac{d\mathbf{X}}{dt} = (\mathbf{B} - \Lambda \mathbf{X}), \tag{A.33}$$

which is (A.32) with $\mathbf{\Sigma} d\mathbf{W}$ dropped.

This is a nonhomogeneous linear system of differential equations which, with initial condition $\mathbf{X}(t_1) = \mathbf{X}_1$ is solved by

$$\begin{aligned}
\mathbf{X}(t) &= e^{-(t-t_1)\Lambda}\mathbf{X}_1 + \int_{t_1}^{t} e^{-(t-s)\Lambda}\mathbf{B}ds \\
&= e^{-(t-t_1)\Lambda}\mathbf{X}_1 + \int_{t_1}^{t} e^{-t\Lambda}e^{s\Lambda}\mathbf{B}ds \\
&= e^{-(t-t_1)\Lambda}\mathbf{X}_1 + e^{-t\Lambda}\int_{t_1}^{t} e^{s\Lambda}ds\mathbf{B} \\
&= e^{-(t-t_1)\Lambda}\mathbf{X}_1 + e^{-t\Lambda}\left[e^{s\Lambda}\Lambda^{-1}\right]_{t_1}^{t}\mathbf{B} \\
&= e^{-(t-t_1)\Lambda}\mathbf{X}_1 + e^{-t\Lambda}\left(e^{t\Lambda}\Lambda^{-1} - e^{t_1\Lambda}\Lambda^{-1}\right)\mathbf{B} \\
&= e^{-\Delta t\Lambda}\mathbf{X}_1 + \left(I - e^{-\Lambda\Delta t}\right)\Lambda^{-1}\mathbf{B},
\end{aligned}$$

where we defined $\Delta t = t - t_1$.

To get the covariance matrix of the transition probability we use the calculation from Grewal and Andrews (2011) which is done for a general stochastic process with time dependent drift and diffusion (Grewal and Andrews, 2011, pp. 144 ff.). When we set the drift and observation noise to zero and use our notation, equation (4.37) in Grewal and Andrews (2011) becomes

$$\frac{\mathbf{V}(t,\mathbf{X}(t_{k-1}))}{dt} = \mathbf{\Sigma}\mathbf{W}\mathbf{\Sigma}^{\top} \tag{A.34}$$

and its solution in equation (4.76) of Grewal and Andrews (2011) is the covariance of our transition density

$$\begin{aligned}
\mathbf{V}(t,\mathbf{X}(t_{k-1})) &= \mathbf{U}\mathbf{D}^{-1} \\
\mathbf{U} &= \mathbf{R}(1:m,1:m) \\
\mathbf{D} &= \mathbf{R}(m+1:2*m,1:m) \\
\mathbf{R} &= \exp(\Delta t\Psi)\begin{bmatrix} \mathbf{0}_{m\times m} \\ \mathbf{I}_{m\times m} \end{bmatrix} \\
\Psi &= \begin{bmatrix} -\Lambda & \Sigma\Sigma' \\ \mathbf{0}_{m\times m} & \Lambda' \end{bmatrix},
\end{aligned} \tag{A.35}$$

with and $\mathbf{R}(r_b : r_e, c_b : c_e)$ being a sub-matrix of $\mathbf{R}$ ranging from row r_b to r_e and column c_b to c_e.

Therefore (A.32) describes a stochastic process with transition density

$$P(\mathbf{X}(t)|\mathbf{X}_0) = \mathcal{N}(\mathbf{X}(t)|m,\mathbf{\Sigma}_t), \tag{A.36}$$

where the mean m is the solution of the corresponding ODE we calculated in (A.34) and covariance matrix $\mathbf{\Sigma}_t$ as specified in (A.35).

A.5.2 Likelihood function

Assume we have noisy observations $\mathbf{D} = (\mathbf{d}_1, \cdots, \mathbf{d}_n)$ of the process taken at times $\mathbf{t} = (t_1, \cdots, t_n)$ and corrupted by i.i.d. Gaussian noise with covariance matrix $\mathbf{\Sigma}_o$.

We are interested in the likelihood of the data conditioned on the parameters $\Theta = (\mathbf{B}, \Lambda, \mathbf{\Sigma}, \mathbf{\Sigma}_o)$:

$$P(\mathbf{D}|\Theta) = \int P(\mathbf{D}, \mathbf{X}|\Theta) d\mathbf{X} \tag{A.37}$$

where $\mathbf{X} = (X_1, \cdots, X_n)$ are the (unknown) true value of the process at times $\mathbf{t}$.

$$\begin{aligned} P(\mathbf{D}|\Theta) &= \int P(\mathbf{D}, \mathbf{X}|\Theta) d\mathbf{X} && \text{(A.38)} \\ &= \int P(\mathbf{D}|\mathbf{X}, \Theta) P(\mathbf{X}|\Theta) d\mathbf{X} && \text{(A.39)} \\ &= \int P(d_1|X_1, \Theta) \prod_{i=1}^{n-1} P(X_i|X_{i-1}, \Theta) P(d_i|X_i, \Theta) d\mathbf{X}, && \text{(A.40)} \end{aligned}$$

where

$$\begin{aligned} P(d_i|X_i, \Theta) &= \mathscr{N}(d_i|X_i, \mathbf{\Sigma}_o) && \text{(A.41)} \\ P(X_i|X_{i-1}, \Theta) &= \mathscr{N}(X_i|m, \mathbf{\Sigma}_t). && \text{(A.42)} \end{aligned}$$

We can compute this iteratively, starting by integrating out X_1.

$$\int P(d_1|X_1, \Theta) P(Y_1|X_1, \Theta) dX_1 = \int \mathscr{N}(d_1|X_1, \mathbf{\Sigma}_o) \mathscr{N}(Y_1|m, \mathbf{\Sigma}_t) dX_1 \tag{A.43}$$

We can rewrite this as a product of two Gaussian densities over X_1 multiplied by a factor which doesn't depend on X_1

$$\begin{aligned} &\int \mathscr{N}(d_1|X_1, \mathbf{\Sigma}_o) \mathscr{N}(X_2|m, \mathbf{\Sigma}_t) dX_1 && \text{(A.44)} \\ &= \int \mathscr{N}(X_1|d_1, \mathbf{\Sigma}_o) \mathscr{N}(X_2|e^{-\Delta t \Lambda} X_1 + \left(I - e^{-\Lambda \Delta t}\right) \Lambda^{-1} \mathbf{B}, \mathbf{\Sigma}_t) dX_1 && \text{(A.45)} \\ &= \int \mathscr{N}(X_1|d_1, \mathbf{\Sigma}_o) \mathscr{N}(e^{-\Delta t \Lambda} X_1|X_2 - \left(I - e^{-\Lambda \Delta t}\right) \Lambda^{-1} \mathbf{B}, \mathbf{\Sigma}_t) dX_1 && \text{(A.46)} \end{aligned}$$

The last Gaussian can be rewritten as (Petersen and Pedersen, 2012, 8.1.5.)

$$
\begin{aligned}
&= \int \mathcal{N}\left(X_1|d_1, \mathbf{\Sigma}_o\right) \frac{1}{|\det(e^{-\Delta t \Lambda})|} \\
&\mathcal{N}\left(X_1|e^{\Delta t \Lambda}\left[X_2 - \left(I - e^{-\Lambda \Delta t}\right)\Lambda^{-1}\mathbf{B}\right], \left[(e^{-\Delta t \Lambda})^\top \mathbf{\Sigma}_t^{-1} e^{-\Delta t \Lambda}\right]^{-1}\right) dX_1
\end{aligned}
\tag{A.47}
$$

and the integral over a product of two Gaussians is easily computed (Petersen and Pedersen, 2012, 8.1.8.)

$$
= \frac{1}{|e^{-\operatorname{tr}(\Delta t \Lambda)}|}\mathcal{N}(d_1|e^{\Delta t \Lambda}\left[X_2 - \left(I - e^{-\Lambda \Delta t}\right)\Lambda^{-1}\mathbf{B}\right], \mathbf{\Sigma}_o + \left[(e^{-\Delta t \Lambda})^\top \mathbf{\Sigma}_t^{-1} e^{-\Delta t \Lambda}\right]^{-1}). \tag{A.48}
$$

We now rewrite the Gaussian distribution as a distribution over X_2 and multiply it with $P(X_2|d_1|\mathbf{\Sigma}_o)$ and $P(X_3|m, \mathbf{\Sigma}_t)$ and integrate out X_2 and then repeat this iteratively until we are at the last observation.

Appendix B

Details of the sampler

B.1 Poisson CRP Sampler: Assigning a λ value to a segment

If we reuse an existing state when **adding** a jump or **switching** the state of a segment we choose the new state i randomly with probability proportional to

$$P(\lambda_i|\mathbf{Y},\lambda_{(0:T)}) = \text{Gamma}\left(\lambda_i; a+n_{seg}, \frac{b}{\tau_{seg}b+1}\right), \tag{B.1}$$

where n_{seg} is the number of Poisson events during the segment, τ_{seg} is the width (in time units) of the segment and a and b come from the base distribution $p_\lambda(\lambda) = \text{Gamma}(\lambda; a, b)$.

Therefore the probability to choose state i becomes

$$p_{seg}(i) = \frac{\lambda_i^{a+n_{seg}-1}\exp\left(-\frac{\tau_{seg}b+1}{b}\lambda_i\right)}{\sum_{j=1}^{s}\lambda_j^{a+n_{seg}-1}\exp\left(-\frac{\tau_{seg}b+1}{b}\lambda_j\right)}. \tag{B.2}$$

After **dividing** a state i into two new states j_1 and j_2 the segments of the original state must be assigned to the new states. For segment l the probability to be assigned to state j_1 is

$$p'_{seg}(l, j_1) = \frac{\lambda_{j_1}^{a+n_l-1}\exp\left(-\frac{\tau_l b+1}{b}\lambda_{j_1}\right)}{\lambda_{j_1}^{a+n_l-1}\exp\left(-\frac{\tau_l b+1}{b}\lambda_{j_1}\right)+\lambda_{j_2}^{a+n_l-1}\exp\left(-\frac{\tau_l b+1}{b}\lambda_{j_2}\right)}, \tag{B.3}$$

and accordingly for state j_2. Let $\omega_1 \dots \omega_{\#_i} \in \{j_1, j_2\}$ be the assignments of the $\#_i$ segments of state i to the new states then we get

$$p_{par} = \prod_{l=1}^{\#_i} p'_{seg}(l, \omega_l). \tag{B.4}$$

If one of the new states is assigned to all segments but the last one, then the last segment is automatically assigned to the other state thereby setting $p_{seg}(\#_i, \omega_{\#_i}) = 1$.

B.2 Poisson Likelihood Calculation

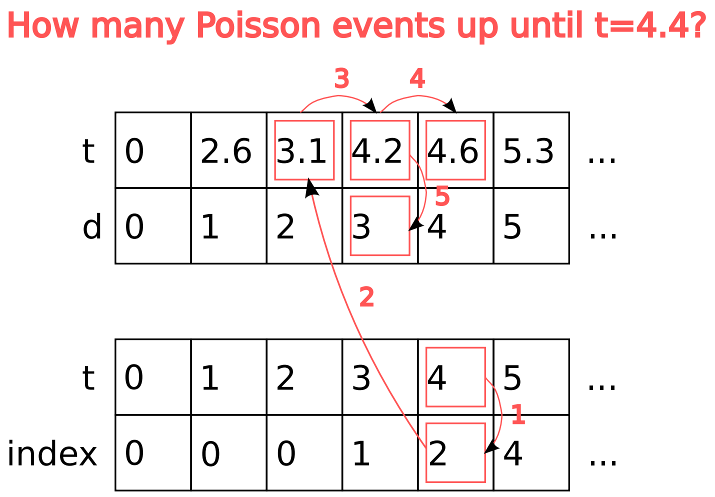

How many Poisson events up until t=4.4?
t 0 2.6 3.1 4.2 4.6 5.3 ...
d 0 1 2 3 4 5 ...
t 0 1 2 3 4 5 ...
index 0 0 0 1 2 4 ...
1
2
3
4
5

Fig. B.1 Nearest observations grid data structure for fast calculation of the likelihood for Poisson data. The observation data is on the top and the grid data structure on the bottom.

When starting the sampler we generate a data structure which creates a grid over the whole time span of the data. For the explanation we assume here that the grid is generated for $\Delta t = 1$ but the level of granularity can be chosen freely. We explain how this works on the example in figure B.1. If we need the number of Poisson events up until $t = 4.4$ we look at the last grid point before that time, which is $t = 4$ in the example. For each time in the grid structure the index in the dataset of the last observation before that time is saved (Step 1). In the example for $t = 4$ the grid saves the index 2 which is the last observation that happened before $t = 4$ (Step 2). We know that the observation the grid is pointing to is prior to the time we are interested in, therefore we look at the next observations and check if its time is larger than the time we are interested in (Step 3). This is not the case, so we iterate one observation further and check again (Step 4). This time we know see that the observation happened after

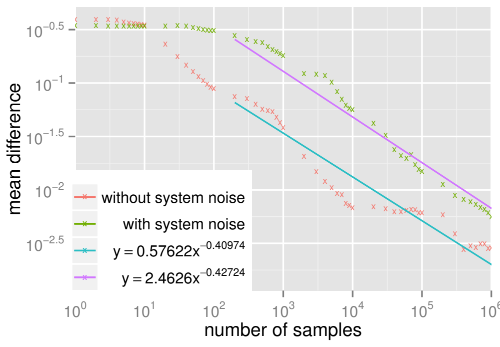
$t = 4.4$. This means the observation before is the last observation before our time. We get the number of Poisson events up until $t = 4.4$ from this observation (Step 5).

Because the grid needs only to be calculated once, we can use a very fine discretization. The finer the discretization is, the less steps are needed to find the correct observation. In practice we get the observation after step 3 in the example. This means even for very large datasets we do not need to iterate over the data but get the number of Poisson events up to a time at constant computational costs. It has to be noted that this procedure does *not* induce a time discretization into the sampler. All calculations still are in continuous time and no approximation error is added. The time discretization only speeds up the retrieval of the correct observation.

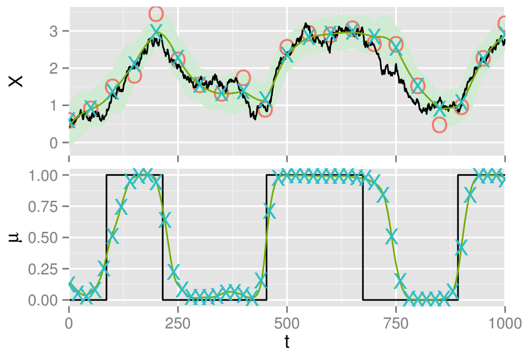
Appendix C

Further Results

C.1 Toy Switching Data



(a) Mean absolute difference between the exact posterior expectation value of μ and the result of the MCMC sampler as a function of the number of iterations. The straight lines are a least squares fit of $y = ax^b$ to the data.



(b) Comparison of the results for the small toy model with system noise. The exact solution (green line) and the MCMC results (blue crosses) are shown for the posterior over $X(t)$ (top) and $\mu(t)$ (bottom). The true values are represented by black lines and the noisy observations by red circles.

Fig. C.1 Further results for synthetic data from the one-dimensional switching model.

C.2 ComS Gene Expression

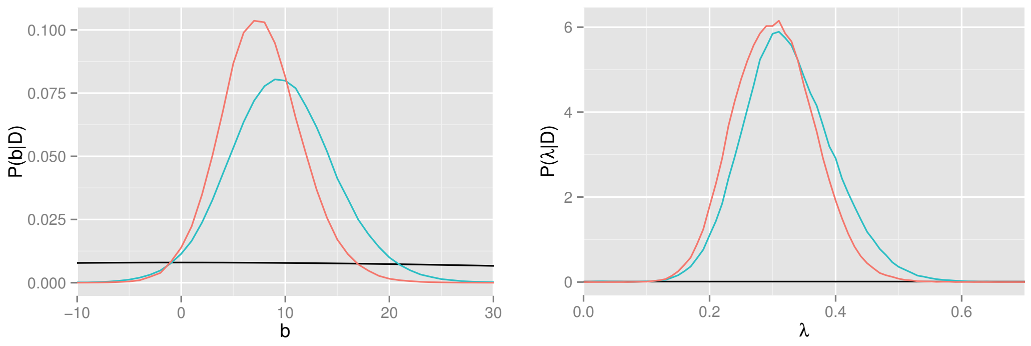


Fig. C.2 Posterior and prior densities of b and λ parameter for the ComS expression data in section 4.1.3. For the model with fixed σ^2 the posterior density is plotted in blue, for the model with switching σ^2 the posterior density is plotted in red. The prior density is plotted in black but for λ we used an exponential distribution with mean 100 which is practically a uniform distribution for the plotted range.

C.3 V1 Neuron Spiking Data

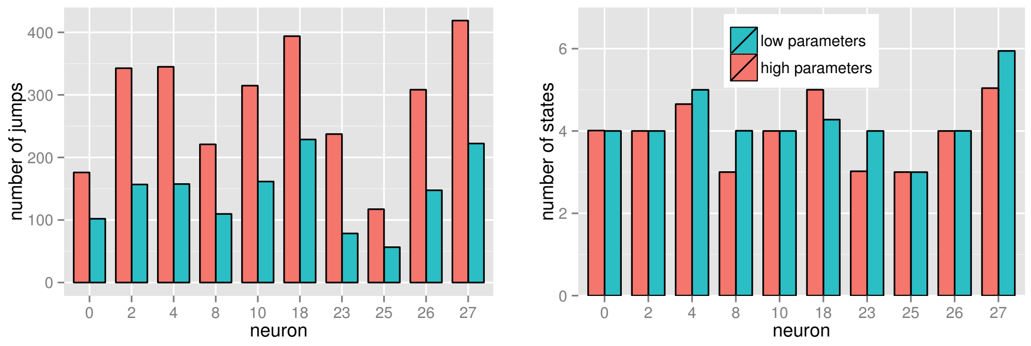


Fig. C.3 Posterior mean number of jumps (*left*) and states (*right*) with low parameters (prior mean of f 10^-4, $\alpha = 0.1$) and high parameters (prior mean of f 10^-3, $\alpha = 0.5$) for the neuron spiking data from section refsec:CRPPoissonNeuralData. The number of states seems to be mostly unaffected while the number of jumps is roughly doubled. Many of the new jumps are jumps between the same state, e.g. for neuron 8 only about 2% of the jumps do not change the state for the low parameter values. For the high parameters this is the case for about 16% of the jumps.