Statistical Inference for Partially Observed Diffusion Processes

PhD thesis by

Anders Christian Jensen

Department of Mathematical Sciences
University of Copenhagen
Denmark

PhD School of Science - Faculty of Science - University of Copenhagen

Anders Christian Jensen
Department of Mathematical Sciences
University of Copenhagen
Universitetsparken 5
DK-2100 København Ø
Denmark

anders@math.ku.dk
htpp://www.math.ku.dk/~anders

PhD thesis submitted to the PhD School of Science, Faculty of Science, University of Copenhagen, Denmark, in February 2014.

Academic advisor:	Susanne Ditlevsen University of Copenhagen, Denmark
Assessment Committee:	Adeline Samson l'Université Joseph Fourier, Grenoble
	Niels Richard Hansen (chair) Department of Mathematical Sciences, University of Copenhagen
	Erik Lindström Lund University

ISBN 978-87-7078-975-2

Preface

This dissertation is submitted in partial fulfillment of the requirements for the Ph.D. degree at the Faculty of Science, University of Copenhagen, Denmark. The work was carried out at the Department of Mathematical Sciences, University of Copenhagen, from March 2010 to February 2014. It was financed by the Department of Mathematical Sciences, and the grant of S. Ditlevsen from the Danish Council for Independent Research | Natural Sciences.

I would like to express my gratitude to my supervisor Susanne Ditlevsen, for scientific advise and her never failing positivity toward her students. Also a very special thanks to Omiros Papaspilioupoulos who has been a big inspiration and for his patience during numerous Skype conversations; always encouraging and full of support and good ideas. Great thanks are also due to Mathieu Kessler for introducing me to the world of Bayesian statistics and MCMC methods and for a really nice time during my stay at the Universidad Polytécnica de Cartagena.

To everyone at the Department of Mathematics, thank you for creating a stimulating environment. Special thanks are due to Massimiliano Tamborrino for interesting conversations about everything and nothing and for his support during the last hectic weeks.

Finally I would like to thank my family for their help and patience all the way from the start until the hectic periods at the final stage.

Summary

This thesis is concerned with parameter estimation for multivariate diffusion models. It gives a short introduction to diffusion models, and related mathematical concepts. We then introduce the method of prediction-based estimating functions and describe in detail the application for a two-dimensional Ornstein-Uhlenbeck where one coordinate is completely unobserved. This model does not have the Markov property and it makes parameter inference more complicated. Next we take a Bayesian approach and introduce some basic Markov Chain Monte Carlo methods. In chapter five and six we describe a Bayesian method to perform parameter inference in multivariate diffusion models that may be only partially observed. The methodology is applied to the stochastic FitzHugh-Nagumo model and the two-dimensional Ornstein-Uhlenbeck process. Chapter seven focus on parameter identifiability in the partially observed Ornstein-Uhlenbeck process, while chapter eight describes the details of an `R`-package that was developed in relations to the application of the estimation procedure of chapters five and six.

Dansk resumé

Denne PhD afhandling omhandler parameter estimation for flerdimensionelle diffusionsmodeller. Der gives en introduktion til diffusionsmodeller og relaterede matematiske begreber. Derefter introduceres de såkaldte prædiktionsbaserede estimationsfunktioner, og der præsenteres en anvendelse heraf på en todimensional Ornstein-Uhlenbeck proces, hvor en af koordinaterne er uobserveret. Da denne model ikke er markov er parameterestimation ganske kompliceret. Vi skifter derefter fokus til et Bayesiansk setup og introducer nogle fundamentale Bayesianske metoder. I kapitel fem og seks introduceres en Bayesiansk metode til parameterestimation som kan bruges i flerdimensionelle diffusionsmodeller som potentielt er partielt observerede. I afhandlingen demonstreres anvendelser heraf, på den stokastiske FitzHugh-Nagumo model og på den todimensionelle Ornstein-Uhlenbeck model. Kapitel otte beskriver detaljerne omkring implementeringen af estimationsmetoderne, i en `R`-pakke som er udviklet som en del af denne afhandling.

Contents

1 Introduction

Diffusion models form a flexible class of statistical models which can be used to model complicated dynamics that are somehow influenced by noise. One such example is excitability - a phenomenon observed in a variety of natural systems, such as neuronal dynamics, ion channels, chemical reactions, or climate dynamics (Lindner et al. (2004); Keener and Sneyd (2009); Berglund and Gentz (2006)). The stochastic FitzHugh-Nagumo model is a prominent example of a two dimensional model, representing an excitable system. To validate the practical use of a model, the first step is to estimate model parameters from experimental data. This is, however not an easy task because of the inherent linearity necessary to produce the excitable dynamics. The estimation procedure may be further complicated by the fact that some coordinates are completely unobserved or because coordinates may operate on different time scales.

Parameter estimation for diffusions has been an active area of research within the last twenty years. For a frequentistic approach see for instance Ait-Sahalia (2002, 2008); Durham and Gallant (2002); Beskos et al. (2009); Sørensen (2000). For Bayesian approaches see Golightly and Wilkinson (2008); Elerian et al. (2001); Roberts and Stramer (2001); Papaspiliopoulos and Roberts (2012). See also Sørensen (2004) for a review. Many papers deal with parameter estimation mainly from a theoretical point of view and examples and illustrations are typically one dimensional. From a practical point of view there is still a great deal of work left for the end user in terms of programming as the theoretical procedures do not always fit directly into the framework of existing software solutions - especially when the models are multidimensional. Together with the improvement in computer speed and storage, complicated and computationally heavy algorithms can now be implemented on most computers. Even so, it is not always straightforward to implement these algorithms, as some methods require a large amount of programming partly due to the complexity of the theoretical method but also in order to minimize computation time.

During my PhD i have dealt with parameter estimation for multivariate diffusions and chapter 2 gives an introduction to diffusions and some basic concepts needed in the subsequent chapters. Starting from a frequentistic point of view I have been working with prediction-based estimating functions (described in i.e. Sørensen (1999, 2000)) and they are the topic of chapter 3. This approach to parameter estimation turned out to be quite difficult and cumbersome to apply in practice to the models of interest. The main problem was related to the computation of the optimal weight matrix and this problem was circumvented using simulations to approximate the weight. However, the results were not all too promising and it motivated us to search for more optimal methods and strategies for estimation.

Thus, focus was changed toward Bayesian methods for parameter inference and these are introduced in chapter 4. In chapter 5 focus is on Bayesian parameter estimation for multidimensional diffusions where all coordinates are (partially) observed. The chapter is an updated version of Jensen et al. (2012) which builds upon the theoretical ideas from Roberts and Stramer (2001).

Chapter 6 expand the ideas from chapter 5 and focus on parameter estimation for multidimensional diffusions when some coordinates are completely unobserved. From a practical

point of view, this makes parameter estimation more difficult because less information is available compared to the fully observed case. Furthermore, parameter identification problems arise, which are difficult to handle in general, without taking into account the specific features of the model. For the FitzHugh–Nagumo model we build upon the original procedure in order to include also the scenario with latent coordinates and a small simulation study is included to demonstrate the performance of the estimation method. Chapter 7 deals with parameter identifiability for the two-dimensional Ornstein-Uhlenbeck process.

The approach to parameter estimation in chapters 5 and 6 is highly computer intensive and it requires a great amount of programming. This motivated the development of an `R`-package, `BIPOD` (Bayesian Inference for Partially Observed Diffusions) which implements these estimation procedures. The first version was implemented directly in `R` and it was very slow. In order to improve performance in terms of speed, it was necessary to set up the programming in a faster language than `R` and the key components are programmed in `C++`. Doing so decreased computation times considerably, sometimes by a factor of 40 compared to a direct implementation in `R`.

The current version of the software supports the stochastic FitzHugh-Nagumo model, a modified version of the FitzHugh-Nagumo model and the Ornstein–Uhlenbeck. Support for the Cox–Ingersoll–Ross model is work in progress.

2 Diffusions

2.1 Stochastic differential equations

This chapter introduces stochastic differential equations (SDE)'s and presents some of the tools needed in the subsequent chapters. A complete introduction to SDE's requires a fair amount of measure theory, and stochastic integration theory. This is however not the focus of this thesis, but the reader is referred to e.g. Rogers and Williams (2000), Karatzas and Shreve (1991) or Jacobsen (2008).

It suffices to state that throughout we will consider filtered probability spaces $(\Omega, \mathcal{F}, (\mathcal{F}_t), P)$ satisfying the following (usual) conditions:

- $(\Omega, \mathcal{F}, P)$ is complete
- $N \in \mathcal{F}_0$ for all $N \in \mathcal{F}$ with $P(N) = 0$
- $\mathcal{F}_t = \cap_{s>t} \mathcal{F}_s$ for all $t \geq 0$

Consider the equation

$$\mathrm{d}V_t = b(V_t; \theta)\, \mathrm{d}t + \Sigma(V_t; \sigma)\, \mathrm{d}W_t, \tag{2.1}$$

where W_t is an m-dimensional Brownian motion with respect to P, and θ and σ are p_1 and p_2-dimensional parameters, respectively. The functions b and Σ take values in (a subset of) $\mathbb{R}^d$ and the set of $d \times m$ matrices, respectively. Let U be $\mathcal{F}_0$-measurable (often a constant) and assume that the stochastic process (V_t) is adapted with respect to $\mathcal{F}_t$. We say that (V_t) solves the SDE (2.1) with initial condition U, if

$$V_0 = U \text{ a.s.} \tag{2.2}$$

$$V_t = V_0 + \int_0^t b(V_s; \theta)\, \mathrm{d}s + \int_0^t \Sigma(V_s; \sigma)\, \mathrm{d}W_s, \tag{2.3}$$

where the first integral is the Lebesgue integral and the second integral is the Itô integral. Unless otherwise stated, we shall consider $U = v_0$ as fixed.

SDE's appear naturally in many different situations, and for many theoretical and practical purposes it is of interest to estimate the parameters guiding the solution process. A strong Markov process solving an SDE is also known as an (Itô) diffusion and the functions b and Σ are known as the drift and diffusion coefficients, respectively.

A necessary condition for the existence of a solution to (2.1) is that

$$\int_0^t |b(V_s; \theta)| + |\Sigma(V_s; \sigma)|^2\, \mathrm{d}s < \infty,$$

for all $t > 0$, with the matrix norm $|\Sigma|^2 := \mathrm{Tr}(\Sigma\Sigma^T)$. There are Lipschitz type conditions ensuring existence and uniqueness of a solution, though such conditions are often too restrictive to hold for many practical diffusion models. See e.g. Jacobsen (2008) or Rogers and Williams (2000). Throughout we will assume sufficient regularity of b and Σ such that a unique weak non-explosive solution to (2.1) exists.

Example 2.1 (The Ornstein-Uhlenbeck process)**.** One of the simplest diffusion models is the Ornstein-Uhlenbeck process, defined by the equation

$$\mathrm{d}V_t = -B(V_t - A)\,\mathrm{d}t + \Sigma\,\mathrm{d}W_t, \tag{2.4}$$

where A and B are real $d \times 1$ and $d \times d$ matrices, respectively. The diffusion coefficient Σ is a $d \times d$ matrix and W_t is a d-dimensional Brownian motion. To ensure that the solution is non-explosive, it is required that the real part of the eigenvalues of B are positive. See Jacobsen (1991).

In the one-dimensional case the condition simplifies to $B > 0$ and all parameters have a very direct interpretation: The diffusion coefficient Σ is measuring the 'size' of the noise, A is the asymptotic mean level for the process and B decides how fast the system reacts to perturbations; that is, how quickly the model returns to values around its asymptotic mean. In Figure 2.1 left is shown a simulated path from the one-dimensional Ornstein-Uhlenbeck process with parameters $A = 0,\ B = 1,\ \Sigma = 1/2$.

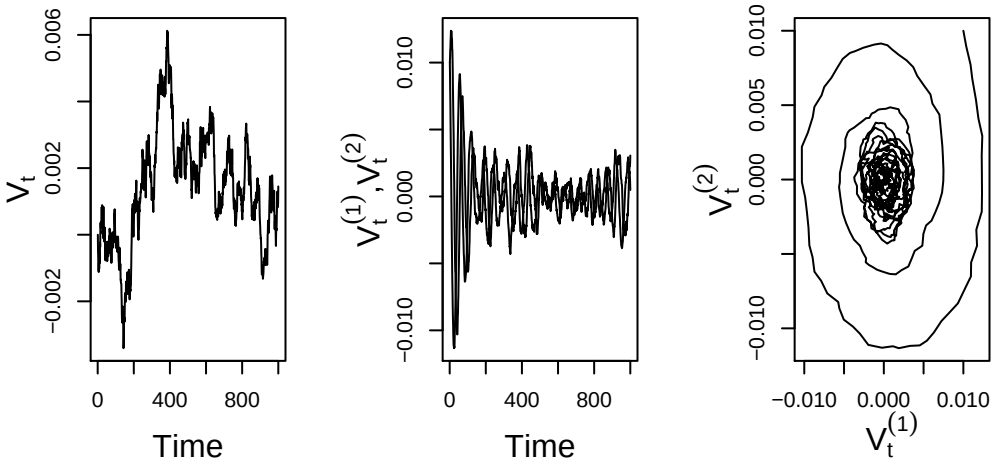

Figure 2.1: *Simulation of two Ornstein-Uhlenbeck processes with positive real part of the eigenvalue of B. Left: 1 dimensional model. Middle and right: Time plot and phase portrait of the two dimensional Ornstein-Uhlenbeck model.*

♦

In the multidimensional version of the Ornstein-Uhlenbeck process, the eigenvalues may very well be complex. In this case the process spiral toward the asymptotic mean level A. This phenomenon is shown in Figure 2.1 middle and right for parameter values

$$A = \begin{pmatrix} 0 \\ 0 \end{pmatrix},\ B = \begin{pmatrix} 1 & -10 \\ 10 & 1 \end{pmatrix},\ \Sigma = \begin{pmatrix} 1/2 & 0 \\ 0 & 3/10 \end{pmatrix}.$$

The Ornstein-Uhlenbeck process is one of the few diffusions where the transition density can be derived explicitly. One way to do this is via the Itô formula.

2.1.1 The Itô formula

The Itô formula is very useful as it describes the result of transforming a diffusion (or more generally, a continuous semi-martingale).

Theorem 2.2. *Let*

$$\mathrm{d}V_t = b(V_t;\theta)\,\mathrm{d}t + \Sigma(V_t;\sigma)\,\mathrm{d}W_t$$

and define $g : \mathbb{R}^d \times \mathbb{R}_+ \mapsto \mathbb{R}^q$ *to be a two times continuous differentiable function. Let* $Z_t^{(k)} = g_k(V_t, t)$ *and define* $Z_t = g(V_t, t)$*. Then the k'th coordinate of the transformed process can be written*

$$\mathrm{d}Z_t^{(k)} = \frac{\partial g_k}{\partial t}(V_t,t)\,\mathrm{d}t + \sum_{i=1}^{d}\frac{\partial g_k}{\partial x_i}(V_t,t)\,\mathrm{d}V_t^{(i)} + \frac{1}{2}\sum_{i,j=1}^{d}\frac{\partial^2 g_k}{\partial x_i \partial x_j}(V_t,t)\,\mathrm{d}V_t^{(i)}\,\mathrm{d}V_t^{(j)}. \quad (2.5)$$

Proof. See Øksendal (2007) for a sketch of the proof in the one-dimensional case. □

Note that the function g may depend on the parameters θ or σ but for notational simplicity this is hidden from the notation.

Equation (2.5) is written in terms of increments with respect to V_t. Sometimes it is more useful to express the formula, using increments with respect to the Brownian motion W_t:

$$\mathrm{d}Z_t^{(k)} = \left(\frac{\partial g_k}{\partial t}(V_t,t) + \sum_{i=1}^{d} b_i(V_t;\theta)\frac{\partial g_k}{\partial x_i}(V_t,t) + \frac{1}{2}\sum_{i,j=1}^{d}\Gamma_{ij}(V_t,\sigma)\frac{\partial^2 g_k}{\partial x_i \partial x_j}(V_t,t)\right)\mathrm{d}t + \sum_{i=1}^{d}\sum_{l=1}^{m}\Sigma_{il}(V_t;\sigma)\frac{\partial g_k}{\partial x_i}(V_t,t)\,\mathrm{d}W_t^{(l)},$$

where b_i is the i'th coordinate of the drift function, Σ_{ij} is the (i,j) coordinate of the diffusion coefficient and

$$\Gamma_{ij}(V_t;\sigma) := \sum_{l=1}^{m}\Sigma_{il}(V_t;\sigma)\Sigma_{jl}(V_t;\sigma).$$

Example 2.3 (Derivation of the solution to the one-dimensional Ornstein-Uhlenbeck process)**.** Assume

$$\mathrm{d}V_t = -\beta(V_t - \alpha)\,\mathrm{d}t + \sigma\,\mathrm{d}W_t,\ \beta > 0,\ \alpha \in \mathbb{R}, \sigma > 0,\ V_0 = v_0,$$

which is the one-dimensional Ornstein-Uhlenbeck process. The solution for this process can be derived using Itô's formula on the function $g(x_t, t) = x_t e^{\beta t}$. With $Z_t := g(V_t, t)$, it follows that

$$\begin{aligned} \mathrm{d}Z_t &= \left(\beta V_t e^{\beta t} - e^{\beta t}\beta(V_t - \alpha)\right)\,\mathrm{d}t + e^{\beta t}\sigma\,\mathrm{d}W_t \\ &= \alpha\beta e^{\beta t}\,\mathrm{d}t + e^{\beta t}\sigma\,\mathrm{d}W_t. \end{aligned}$$

Then upon integration

$$V_t e^{\beta t} = v_0 + \alpha\beta\int_0^t e^{\beta s}\,\mathrm{d}s + \sigma\int_0^t e^{\beta s}\,\mathrm{d}W_s,$$

such that

$$V_t = v_0 e^{-\beta t} + \alpha(1 - e^{-\beta t}) + \sigma\int_0^t e^{-\beta(t-s)}\,\mathrm{d}W_s.$$

From this expression it is easy to derive the conditional moments that define the distribution of the process. The first conditional moment is

$$E(V_t \mid V_0 = v_0) = v_0 e^{-\beta t} + \alpha(1 - e^{-\beta t}),$$

and the second central moment (using Itô's isometry):

$$\mathrm{Var}(V_t \mid V_0 = v_0) = \frac{\sigma^2}{2\beta}(1 - e^{-2\beta t}).$$

As the process is Gaussian and time homogeneous we obtain the transition density

$$p_t(x, y) = \left(\pi\sigma^2(1 - e^{-2\beta t})/\beta\right)^{-1/2}\exp\left(\frac{-\beta(y - xe^{\beta t} - \alpha(1 - e^{-\beta t}))^2}{\sigma^2(1 - e^{-2\beta t})}\right).$$

♦

2.1.2 Reducible diffusions and the Lamperti transform

A key concept in the methods to be described in chapter 5 and chapter 6 is that of a reducible diffusion. A diffusion is reducible if there exists a one-to-one function g that transforms the original diffusion V into another diffusion Z with unit diffusion coefficient and this function is termed the Lamperti transform. Using Itô's formula

$$\begin{aligned} \mathrm{d}Z_t^{(i)} &= \nabla g_i(g^{-1}(Z_t))^T b(g^{-1}(Z_t);\theta)\,\mathrm{d}t \\ &\quad + \frac{1}{2}\,\mathrm{Tr}\left\{\nabla^2 g_i(g^{-1}(Z_t))\Sigma(g^{-1}(Z_t);\sigma)\Sigma(g^{-1}(Z_t);\sigma)^T\right\}\,\mathrm{d}t \\ &\quad + \nabla g_i(g_i^{-1}(Z_t))^T\Sigma(g_i^{-1}(Z_t);\sigma)\,\mathrm{d}W_t, \end{aligned}$$

where ∇g_i and $\nabla^2 g_i$ are the Jacobian and the Hessian of g_i, respectively. Unit diffusion for Z is obtained exactly when

$$\nabla g(x)\Sigma(x;\sigma) = I_d, \tag{2.6}$$

where I_d is the d-dimensional identity matrix. Because Σ is non singular, this is equivalent to

$$\frac{\partial g_i(x)}{\partial x_j} = \Sigma^{-1}_{ij}(x;\sigma). \tag{2.7}$$

When the second partial derivatives of g exist and are continuous, such that the partial derivatives can be interchanged, this condition can be translated into a necessary and sufficient condition on the diffusion matrix Σ:

$$\frac{\partial \Sigma^{-1}_{ij}(x;\sigma)}{\partial x_k} = \frac{\partial \Sigma^{-1}_{ik}(x;\sigma)}{\partial x_j}, \quad i,j,k = 1,2,\ldots,d. \tag{2.8}$$

Clearly condition (2.8) is necessary, but it is also sufficient: Choosing g such that

$$g_i(x) = \int^{x_j} \Sigma^{-1}_{ij}(u;\sigma)\,\mathrm{d}u_j,$$

for all $j = 1,2,\ldots,d$, will imply (2.7) and therefore also (2.6). Condition (2.8) is given in Ait-Sahalia (2008) where also more general Σ are considered. The lower limit of the integral is not specified as it is not important.

When Σ is diagonal, and Σ_{ii} depends on x only through x_i, then the transformation given by

$$g_i(x) = \int^{x_i} \Sigma^{-1}_{ii}(u;\sigma)\,\mathrm{d}u_i$$

satisfies (2.6) and g is one-to-one. It follows that

$$\mathrm{d}Z^{(i)}_t = \left(\Sigma^{-1}_{ii}(g_i^{-1}(Z^{(i)}_t);\sigma)b_i(g^{-1}(Z_t);\theta) - \frac{1}{2}\frac{\partial}{\partial x_i}\Sigma_{ii}(g_i^{-1}(Z^{(i)}_t);\sigma)\right)\,\mathrm{d}t + \,\mathrm{d}W^{(i)}_t. \tag{2.9}$$

In general, when (2.6) is satisfied, we obtain a diffusion Z

$$\mathrm{d}Z_t = \alpha(Z_t;\theta,\sigma)\,\mathrm{d}t + \,\mathrm{d}W_t,$$

where the new drift function α depends on both θ and σ. It also involves g^{-1}, for which a feasible expression is not always easy to find, but in the following two examples it is possible.

Example 2.4 (Constant diffusion)**.** Let $\Sigma(x;\sigma) := \Sigma$. Then $g(x) = \Sigma^{-1}x$, and

$$\mathrm{d}Z_t = \Sigma^{-1}b(\Sigma V_t;\theta)\,\mathrm{d}t + \,\mathrm{d}W_t.$$

♦

Example 2.5 (Square root diffusion)**.** Consider $\Sigma_{ij}(x;\sigma) = \sigma_i\sqrt{x_i}\mathbf{1}_{(i=j)}$ for $x_i > 0$. Taking $g_i(x) = 2\sqrt{x_i}/\sigma_i$ gives $g_i^{-1}(g(x)) = g_i^2(x)\sigma_i^2/4$, and

$$\mathrm{d}Z_t^{(i)} = \frac{1}{2Z_t^{(i)}\sigma_i^2}\left(4b_i(g^{-1}(Z_t);\theta) - \sigma_i^2\right)\,\mathrm{d}t + \,\mathrm{d}W_t^{(i)}.$$

♦

2.1.3 The Girsanov Theorem

In both frequentistic and Bayesian statistics, the likelihood function is of great interest in order to perform parameter inference. It is defined via the joint density of the observed data, which is given with respect to some dominating measure. For discretely observed data the density is typically given with respect to either the counting measure or the Lebesgue measure. For continuously observed data the dominating measure is, in nature, infinite dimensional and for a diffusion process as in (5.1), the likelihood for θ can be derived using the Girsanov theorem. This theorem provides an expression for the likelihood (or the Radon-Nikodym derivative) of one Itô process with respect to another when the probability measures related to the two processes are not mutually singular.

Theorem 2.6 (Girsanov)**.** *Let $(\Omega, \mathcal{F}, \mathcal{F}_{t\geq 0}, P_0)$ be a filtered probability space and define the drift-less d-dimensional Itô process,*

$$\mathrm{d}V_t = \Sigma(V_t;\sigma)\,\mathrm{d}W_t, \quad 0 \leq t \leq T,$$

such that W_t is a d-dimensional Brownian motion with respect to P_0. Suppose there exists 'suitable' functions h and b such that

$$\Sigma(V_t,\sigma)h(V_t;\theta) = -b(V_t;\theta).$$

Define for $0 \leq t \leq T$,

$$M_t := e^{-\int_0^t h(V_s;\theta)^T\,\mathrm{d}W_s - \frac{1}{2}\int_0^t (h^T h)(V_s;\theta)\,\mathrm{d}s},$$

and let

$$\mathrm{d}P_b := M_T\,\mathrm{d}P_0, \quad \text{on } \mathcal{F}_T.$$

If M_t is a martingale with respect to P_0 and $\mathcal{F}_t$, then P_b is a probability measure on $\mathcal{F}_T$, and

$$\mathrm{d}V_t = b(V_t;\theta)\,\mathrm{d}t + \Sigma(V_t;\sigma)\,\mathrm{d}\tilde{W}_t,$$

where $\tilde{W}_t$ is a d-dimensional Brownian motion with respect to P_b.

See Øksendal (2007) for a proof. It is important to realize that W_t is the P_0 Brownian motion and that M_T is the Radon-Nikodym derivative of $\mathrm{d}P_b/\,\mathrm{d}P_0$ on $[0,T]$. For a formal definition of 'suitable' processes, see def. 3.3.2 in Øksendal (2007). Note that M_t is the continuous time likelihood relative to the measures P_b and P_0.

2.1.4 Diffusion bridges

A diffusion bridge is a diffusion conditioned on start and end point. If the diffusion bridge, started in v_0, is conditioned to hit v_T at time T, the corresponding bridge process can be written as

$$\mathrm{d}V_t = (b(V_t;\theta) + \Gamma(V_t;\sigma)\nabla_{V_t}\log(p_{t,T}(V_t, v_T, \theta)))\ \mathrm{d}t + \Sigma(V_t;\sigma)\ \mathrm{d}W_t, \tag{2.10}$$

as noted in Papaspiliopoulos et al. (2013) and references therein. Here $p_{t,T}(x,y)$ is the transition density for (2.1) from x to y at time t and T, and $\nabla_x g(x,y)$ is the partial derivative of g with respect to x and $\Gamma = \Sigma\Sigma^T$. From a practical point of view this is not so useful, as it involves an explicit expression for the transition density which in most situations is intractable. However, for the Brownian motion the corresponding bridge (ending at $V_T = v_T$ at time T) is

$$\mathrm{d}V_t = \frac{1}{T-t}(v_T - V_t)\ \mathrm{d}t + \ \mathrm{d}W_t,$$

where W_t is the Brownian motion from the unconditional process. Any skeleton of this bridge can be sampled exactly, as the distribution for any $0 \leq s < t \leq T$ is Gaussian with

$$(V_t \mid v_s, v_T) \sim \mathcal{N}\left(v_s + \frac{t-s}{T-s}(v_T - v_s); \frac{(T-t)(t-s)}{T-s}\right), \tag{2.11}$$

which does not depend on V_0, unless $s = 0$.

The Brownian bridge $(V_t)_{0\leq t\leq T}$ can be transformed into a standard Brownian bridge, which is starting and ending at 0 on the interval from 0 to T: Define the function h by

$$h(V_t; t, v_0, v_T) = V_t - \left(1 - \frac{t}{T}\right)v_0 - \frac{t}{T}v_T, \quad 0 \leq t \leq T.$$

It can easily be verified that $h(V_t; t, v_0, v_T)$ is a Brownian bridge starting and ending at 0. This implies that conditional on the endpoints, v_0, v_T, there is a one-to-one correspondence between the original Brownian motion and the one starting and ending at 0.

2.2 The continuous time likelihood for a diffusion bridge

In general it is very difficult to simulate a diffusion bridge, except for special cases as in section (2.1.4), where the transition density for the Brownian bridge was found. As will be discussed in chapter 4 some diffusion processes can be simulated using only samples of a Brownian bridge and knowledge about the Radon-Nikodym derivative linking the Brownian bridge with the diffusion bridge of interest. It is therefore important to have an expression for the Radon-Nikodym derivative of a bridged diffusion with respect to a

Brownian bridge. Consider all continuous functions on the interval from 0 to T starting and ending in v_0 and v_T, respectively. Let $\mathbb{P}(0, T, v_0, v_T; \theta, \sigma)$ denote the measure of the process

$$\mathrm{d}V_t = b(V_t; \theta)\ \mathrm{d}t + \Sigma(V_t; \sigma)\ \mathrm{d}W_t,\ V_T = v_T,\ 0 \leq t \leq T, \tag{2.12}$$

and let $\mathbb{Q}(0, T, v_0, v_T; \sigma)$ denote the measure of

$$\mathrm{d}V_t = \Sigma(V_t; \sigma)\ \mathrm{d}W_t,\ 0 \leq t \leq T,\ V_T = v_T,\ 0 \leq t \leq T. \tag{2.13}$$

The Radon-Nikodym derivative between these two measures are given in Papaspiliopoulos and Roberts (2012) and it has the form

$$\frac{\mathrm{d}\mathbb{P}(0, T, v_0, v_T; \theta, \sigma)}{\mathrm{d}\mathbb{Q}(0, T, v_0, v_T; \sigma)} = \frac{\varphi_{0,T}(z_0, z_T)}{p_{0,T}(z_0, z_T)} G(0, T, V_{[0;T]}, b, \Gamma; \theta, \sigma). \tag{2.14}$$

Here $p_{0,T}(v_0, v_T)$ and $\varphi_{0,T}(v_0, v_T)$ are the transition densities for (2.12) and (2.13), respectively, where we have not conditioned on the endpoint v_T, and

$$G(0, T, V_{[0;T]}, b, \Gamma; \theta, \sigma) \tag{2.15}$$

$$= \exp\left(\int_0^T b(V_s; \theta, \sigma)^T \Gamma(V_s; \sigma)^{-1}\ \mathrm{d}V_s - \frac{1}{2}\int_0^T b(V_s; \theta, \sigma)^T \Gamma(V_s; \sigma)^{-1} b(V_s; \theta, \sigma)\ \mathrm{d}s\right) \tag{2.16}$$

is the Radon-Nikodym derivative related to the measures of the unconditional processes. For a detailed derivation of the likelihood, see also Papaspiliopoulos et al. (2013).

2.3 Models

In this section we introduce the models that will be considered in subsequent chapters for parameter estimation.

2.3.1 The Ornstein-Uhlenbeck process

The Ornstein-Uhlenbeck process is especially interesting in its own right as one can compute the transition density. The one-dimensional model was already introduced in Example 2.1 and 2.3 and the multi-dimensional version is

$$\mathrm{d}V_t = -B(V_t - A)\ \mathrm{d}t + \Sigma\ \mathrm{d}W_t, \quad V_0 = v_0, \quad t \in [0, T]. \tag{2.17}$$

Using the Itô formula as in Example 2.3, or referring to Kloeden et al. (2003), p. 73ff, it follows that the solution to (2.17) is given by

$$V_t = e^{-Bt} v_0 + \int_0^t e^{-B(t-s)} A\ \mathrm{d}s + \int_0^t e^{-B(t-s)} \Sigma\ \mathrm{d}W_s. \tag{2.18}$$

To keep things as simple as possible the asymptotic mean level A is now set to 0. In this case the defining moments for the Gaussian distribution are

$$\begin{aligned}\mathrm{E}(V_t \mid V_s) &= e^{-Bt}V_s,\\ \mathrm{Cov}(V_s, V_t) &= \int_0^s e^{-B(s-u)}\Sigma\Sigma^T e^{-B^T(t-u)}du,\end{aligned}$$

for $0 < s \leq t \leq T$, where $\mathrm{Cov}(V_t, V_t) = \mathrm{Var}(V_t)$. Therefore the transition density is given by

$$p_{s,t}(x,y) = (2\pi)^{-d/2}\det\left(\mathrm{Var}(V_t \mid V_s)\right)^{-1/2} e^{-\frac{1}{2}(x-y)^T \mathrm{Var}(V_t|V_s)^{-1}(x-y)}.$$

Note that in the one-dimensional case, where $\Sigma = \sigma$ and $B = \beta$, the moments simplify to

$$\mathrm{E}(V_t \mid V_s) = e^{-\beta(t-s)}V_s, \tag{2.19}$$

$$\mathrm{Cov}(V_s, V_t) = \frac{\sigma^2}{2\beta}\left(e^{-\beta(t-s)} - e^{-\beta(s+t)}\right). \tag{2.20}$$

Using (2.10), the bridge (conditioned on $V_0 = v_0$ and $V_T = v_T$) corresponding to the diffusion (2.17) (with $A = 0$) is given by

$$\begin{aligned}\mathrm{d}V_t &= \left(b(V_t;\theta) + \Gamma(V_t;\sigma)\nabla_{V_t}\log(p_{t,T}(V_t, v_T))\right)\,\mathrm{d}t + \Sigma(V_t;\sigma)\,\mathrm{d}W_t\\ &= \left[-BV_t + \Sigma\Sigma^T\,\mathrm{Var}(V_T \mid V_t)^{-1}(v_T - V_t)\right]\,\mathrm{d}s + \Sigma\,\mathrm{d}W_t.\end{aligned}$$

This expression can be used for simulation of the multidimensional Ornstein-Uhlenbeck bridge via e.g. the Euler scheme or higher order simulation schemes. Another approach which does not rely on approximations, is to use rules for conditional Gaussian distributions directly on $(V_t)_{t\geq 0}$. It follows that

$$\begin{aligned}\mathrm{E}(V_s \mid V_t, V_0) &= \mathrm{E}(V_s \mid V_0) + \mathrm{Cov}(V_t, V_s \mid V_0)\,\mathrm{Var}(V_t \mid V_0)^{-1}\left(V_t - \mathrm{E}(V_t \mid V_0)\right),\\ \mathrm{Var}(V_s \mid V_t, V_0) &= \mathrm{Var}(V_s \mid V_0) - \mathrm{Cov}(V_t, V_s \mid V_0)\,\mathrm{Var}(V_t \mid V_0)^{-1}\,\mathrm{Cov}(V_t, V_s \mid V_0),\end{aligned}$$

where all moments can be evaluated using (2.19) and (2.20).

As the distribution of $(V_t)_{0\leq t\leq T}$ conditional on V_0 and V_T is Gaussian, the conditional mean and variance define the transition density.

Conditions for stationarity

Consider again the general model in (2.17). The process is stationary when the real parts of the eigenvalues of B are strictly positive, see Jacobsen (1991), Theorem 6.2. In the two-dimensional case, the two eigenvalues λ_+ and λ_- are

$$\lambda_\pm = \frac{\mathrm{Tr}(B) \pm \sqrt{\mathrm{Tr}(B)^2 - 4\det(B)}}{2}.$$

Conditions for stationarity now depend on the sign of $\mathrm{Tr}(B)^2 - 4\det(B)$.

If $\mathrm{Tr}(B)^2 - 4\det(B) \leq 0$: Then $\mathrm{Re}(\lambda_\pm) = \mathrm{Tr}(B)/2$, and the stationarity condition is $\mathrm{Tr}(B) > 0$.

If $\mathrm{Tr}(B)^2 - 4\det(B) > 0$: Then $\mathrm{Re}(\lambda_-) < \mathrm{Re}(\lambda_+)$, so it suffices to focus on $\mathrm{Re}(\lambda_-)$. Thus

$$\begin{aligned} \mathrm{Re}(\lambda_-) &= \frac{\mathrm{Tr}(B) - \sqrt{\mathrm{Tr}(B)^2 - 4\det(B)}}{2} > 0 \\ &\Updownarrow \\ \det(B) &> 0, \end{aligned}$$

when $\mathrm{Tr}(B)^2 - 4\det(B) > 0$. Thus the process is stationary if

$$\begin{aligned} \mathrm{Tr}(B) > 0 \quad &\text{when } \mathrm{Tr}(B)^2 - 4\det(B) \leq 0; \\ \det(B) > 0 \quad &\text{when } \mathrm{Tr}(B)^2 - 4\det(B) > 0. \end{aligned}$$

2.3.2 The FitzHugh-Nagumo model

The FitzHugh-Nagumo model is a two-dimensional model where the coordinates represent the membrane potential of a neuron and a (latent) recovery variable modeling the ion channel kinetics. It was first studied without noise, that is without any diffusion term, but in order to account for various sources of noise it is natural to extend the model with the diffusion term to obtain an SDE. The FitzHugh-Nagumo model is one of the simplest models that can exhibit either excitatory or oscillatory behavior. It is defined in slightly different ways in the literature and it has been studied in e.g. FitzHugh (1961); Nagumo et al. (1962); Gerstner and Kistler (2002); Izhikevich (2007); Jensen et al. (2012). The deterministic model is given as

$$\frac{\mathrm{d}}{\mathrm{d}t}x_t = \frac{1}{\varepsilon}\left(x_t - x_t^3 - y_t + s\right) \tag{2.21}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}y_t = \gamma x_t - y_t + \beta, \tag{2.22}$$

with $\varepsilon > 0$ and $\gamma, s, \beta \in \mathbb{R}$. In the modeling of neuronal spike generation in axons, x describes the membrane potential and y is a recovery variable. The parameter ε is a time scale separator, typically smaller than one, such that x is the fast, and y is the slow variable. Furthermore, s denotes the input current and γ and β determines the location of the fixed point(s). Note that the process has no dimension, but x and y must be on the same (dimensionless) scale because of the common ε in the first coordinate of the drift function.

The x and y null-clines are found by setting (2.21) and (2.22) equal to zero:

$$\begin{aligned} y_t &= x_t - x_t^3 + s \\ y_t &= \gamma x_t + \beta. \end{aligned}$$

Since the null-clines are linear and cubic in x_t the model has at most three fixed points and when $\gamma > 1$ there is exactly one, located at the intersection of the two null-clines. Depending on the parameter values, this fixed point is either stable or unstable and the model exhibits qualitatively different behavior for different values of the parameter β. Figure 2.2 shows phase and time plots for two sets of parameter values, leading to excitatory and oscillatory behavior, respectively. In both cases there is only one fixed point. The pa-

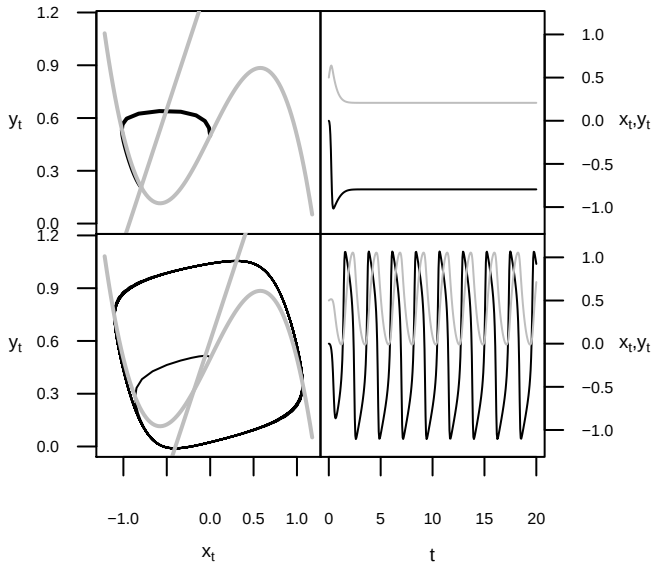

Figure 2.2: *Deterministic FitzHugh-Nagumo model. Left: Phase portraits with x and y null-clines (gray) and simulated trajectory (black). Right: Time plots of x (black) and y (gray). For all plots* $\varepsilon = 0.1, s = 0.5, \gamma = 1.5$. *Top:* $\beta = 1.4$, *excitatory behavior, fixed point is stable. Bottom:* $\beta = 0.6$, *oscillatory behavior, fixed point is unstable.*

rameters in the upper panels are chosen such that the fixed point is stable and the model spikes one time and then relaxes to the resting state and stays there. In the lower panels the fixed point is unstable and a limit cycle with spikes appears. A detailed exposition of the dynamics of the model can be found in Gerstner and Kistler (2002); Izhikevich (2007).

Stochastic extension

We include additive noise in both coordinates and obtain the following stochastic model:

$$\mathrm{d}X_t = \frac{1}{\varepsilon}\left(X_t - X_t^3 - Y_t + s\right)\,\mathrm{d}t + \sigma_1\,\mathrm{d}W_t^{(1)} \tag{2.23}$$

$$\mathrm{d}Y_t = \left(\gamma X_t - Y_t + \beta\right)\,\mathrm{d}t + \sigma_2\,\mathrm{d}W_t^{(2)}, \tag{2.24}$$

where $(W_t^{(1)}, W_t^{(2)})^T$ is a two-dimensional standard Brownian motion and $t \in [0, T]$, $(X_0, Y_0) = (x_0, y_0)$. The parameters of the model are $(\theta, \sigma)^T$ with the drift parameter $\theta = (\varepsilon, s, \gamma, \beta)^T \in \mathbb{R}_+ \times \mathbb{R}^3$ and diffusion parameter $\sigma = (\sigma_1, \sigma_2)^T \in \mathbb{R}_+^2$.

With $V_t = (X_t, Y_t)$ and

$$b(V_t; \theta) = \begin{pmatrix} \frac{1}{\varepsilon}\left(V_t^{(1)} - \left(V_t^{(1)}\right)^3 - V_t^{(2)}\right) + s \\ \gamma V_t^{(1)} - V_t^{(2)} + \beta \end{pmatrix},$$

$$\Sigma(\sigma) = \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix},$$

the model is on the general form from (2.1).

The qualitative behavior of the stochastic model is different from the deterministic model because the random perturbations can affect the system and lead to emergence of repeated spiking activity, also when the fixed point is stable, see Figure 2.3.

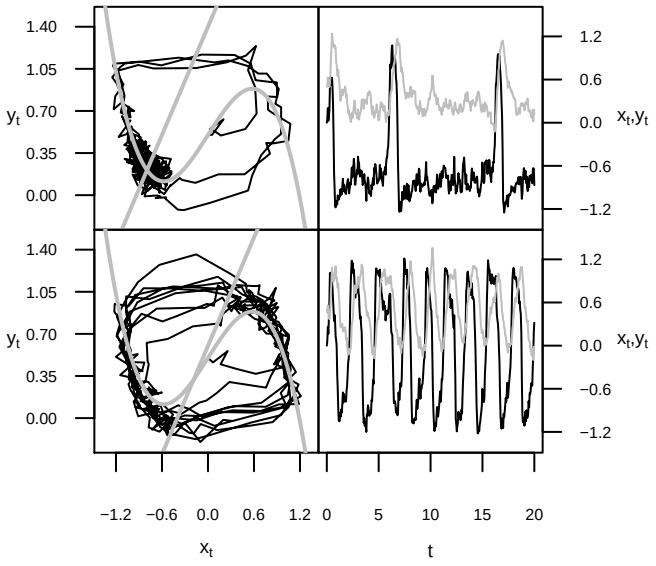

Figure 2.3: *Stochastic FitzHugh-Nagumo model. Left: Phase portraits with x and y null-clines (gray) and simulated trajectory (black). Right: Time plots of x (black) and y (gray). For all plots* $\varepsilon = 0.1, s = 0.5, \gamma = 1.5, \sigma_1 = 0.5, \sigma_2 = 0.3$. *Top:* $\beta = 1.4$, *excitatory behavior, fixed point is stable. Bottom:* $\beta = 0.6$, *oscillatory behavior, fixed point is unstable.*

The FitzHugh-Nagumo model has some natural restrictions when it comes to modeling real data. The local minimum and maximum are always located at $\pm 1/\sqrt{3}$ respectively, with a distance between min and max equal to $2/\sqrt{3} \approx 1.15$. Since the data trajectory

tends to circle along the legs and around the knees of the cubic null-cline it is not possible to have x values that in absolute values are much larger than 1 (relative to the size of the noise). This problem can potentially be solved by a transformation of data. Another option is to consider an extension of the FitzHugh-Nagumo model as in the next section.

2.3.3 The extended FitzHugh-Nagumo model

For the FitzHugh-Nagumo model the local minimum and maximum is located at $x = \{\pm 1/\sqrt{3}\}$, respectively, with a distance between min and max, $L \approx 2/\sqrt{3} \approx 1.15$. Since the data trajectory tends to circle along the legs and around the knees of the cubic null-cline it is not possible to have x values that in absolute values are much larger than 1.

The range for a real data set do not always satisfy this rather strict condition even after transformations, and the model must be modified to accommodate the specific situation. The obvious approach is to add another parameter in front of either x or x^3 in the first coordinate. (To preserve linearity in the drift, we do not use a parameter α/ε but rather just α.) In the first case we get (after a re-parametrization of ε into $1/\varepsilon$)

$$\mathrm{d}X_t = (-\alpha X_t^3 + \varepsilon(X_t - Y_t) + s)\,\mathrm{d}t + \sigma_1\,\mathrm{d}W_t^{(1)} \tag{2.25}$$

$$\mathrm{d}Y_t = (\gamma X_t - Y_t + \beta)\,\mathrm{d}t + \sigma_2\,\mathrm{d}W_t^{(2)} \tag{2.26}$$

with $\alpha, \epsilon > 0, \gamma > 1$, and $\beta, s \in \mathbb{R}$. For this model the minimum and maximum are located at $x = \{\pm\sqrt{\varepsilon/3\alpha}\}$. the spiking behavior of the original FitzHugh-Nagumo-model is retained if the two null-clines intersect close to the left knee of the first null-cline. Hence we need $\gamma x_1 + \beta = y_1$, for $(x_1, y_1) = (\sqrt{\varepsilon/(3\alpha)}, -(\alpha/\varepsilon)x_1^3 + x_1 + s/\varepsilon)$. For reasonable values of the diffusion matrix, this will give rise to the spiking behavior already described in the FitzHugh-Nagumo model.

In the case where $\alpha \approx 0$ this model is approximately a two-dimensional Ornstein-Uhlenbeck model as in (2.17) with $A = (0, 0)^T$,

$$B = \begin{pmatrix} -\frac{1}{\varepsilon} & \frac{1}{\varepsilon} \\ -\gamma & 1 \end{pmatrix},$$

and diffusion coefficient

$$\sigma = \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix}.$$

The eigenvalues of B are

$$\lambda_\pm = \frac{1 - \frac{1}{\varepsilon} \pm \sqrt{(1 - \frac{1}{\varepsilon})^2 - 4\frac{(\gamma - 1)}{\varepsilon}}}{2}.$$

Note that the model (2.17) is parameterized with a minus in front of B and for $0 < \varepsilon < 1, \gamma > 1$, the eigenvalues of B always have positive real parts, meaning that the diffusion will be stationary.

3

Estimating functions

3.1 Estimating Functions

Let $\theta \in \Theta$ be a p-dimensional unknown parameter and consider d-dimensional observations $X_0, X_{t_1}, \ldots, X_{t_n} \in D \subset \mathbb{R}^d$ from some distribution. The t_i denotes time, and for notational simplicity we primarily write X_i instead of X_{t_i}, and assume that we have equidistant observations with time step $\Delta = t_i - t_{i-1}$ between observations. These observations could in the simplest case be iid. variables but they are typically samples from some stochastic process.

In many situations the preferred strategy for parameter estimation in statistical models is maximum likelihood estimation because of the nice properties this estimator possesses. However, it is not always possible to apply this strategy to a specific model, either because the likelihood function is unknown or because it is too computational costly to evaluate. To illustrate, consider a two-dimensional continuous time Markov model, with transition density $p(\Delta, x, y; \theta)$ for going from x to y in the time period Δ. If both coordinates are observed, the likelihood, $L(\theta)$, decomposes into a product of conditionals, such that

$$L(\theta) = \prod_{i=1}^{n} p(\Delta, X_{i-1}, X_i; \theta),$$

with X_0 considered fix. If one coordinate is completely unobserved, inference should be based solely on the marginal distribution of the observed coordinate and this likelihood is typically no longer Markovian. In this case, one may take a more general approach and use estimating functions.

Definition 3.1. *An estimating function is a function of the unknown parameter and data:*

$$(\theta, X_0, X_1, \ldots, X_n) \mapsto G_n(\theta, X_0, X_1, \ldots, X_n).$$

An estimator related to the estimating function is a solution to

$$G_n(\theta) = 0. \tag{3.1}$$

Definition 3.1 is quite general and is also known as the generalized method of moments; see e.g. Hansen (1982). Here we shall focus on the class of estimating functions of the following form

$$G_n(\theta) = \frac{1}{n} \sum_{i=s+1}^{n} g(X_{i-s}, \ldots, X_i; \theta),$$

with s a fixed integer and g a function with values in $\mathbb{R}^p$, where p is the dimension of the parameter θ. Thus each term in the sum depends on the last $s+1$ observations up to time point t_i, and we want to find a parameter vector θ, such that $G_n(\theta) = 0$.

Results on existence and uniqueness of a solution to (3.1) can be found in Sørensen (2012).

3.2 Martingale estimating functions

Consider the diffusion

$$\mathrm{d}X_t = b(X_t;\theta)dt + \Sigma(X_t;\theta)\,\mathrm{d}W_t,$$

where Σ is a $d \times d$ dimensional matrix and W is a d-dimensional Brownian motion. Note that we do not distinguish between drift and diffusion parameters. Assume this process is observed for $X_0, X_1, \ldots, X_n$ and the aim is to estimate the parameter θ guiding the process.

A martingale estimating function G_n is simply an estimating function as in definition 3.1 with the property

$$\mathrm{E}_\theta\left(G_n(\theta) \mid \mathcal{F}_{n-1}\right) = G_{n-1}(\theta), \quad n = 1, 2, \ldots,$$

where $\mathcal{F}_n$ is the σ-algebra generated by $X_0, X_1, \ldots, X_n$.

Martingale estimating functions are particularly interesting from a mathematical point of view because a well developed asymptotic theory is already in place for such functions, see e.g. Hall and Heyde (1980).

In the following focus is on martingale estimating functions on the form

$$G_n(\theta) = \sum_{i=1}^{n} a(X_{i-1};\theta)h(X_{i-1}, X_i;\theta), \tag{3.2}$$

where a is a weight matrix with dimensions $p \times N$ and $h = (h_1, \ldots, h_N)^T$ is an N dimensional function satisfying for all real valued h_k functions

$$\int_D h_k(x, y;\theta)p(\Delta, x, y;\theta)\,\mathrm{d}y = 0,$$

for all $\Delta > 0, \theta \in \Theta$ and $x \in D$. Here D denotes the state space of X and p is the time homogeneous transition density going from x to y by time Δ. One argument for choosing this class of estimating functions is that the score function (which would be the optimal choice), under weak regularity conditions, is a martingale. In order to apply martingale estimating functions to a given problem, it is necessary to compute the weight matrix a in (3.2). Under certain differentiability and integrability conditions on h_j given as Condition 1.8 in Sørensen (2012), the optimal weight matrix a^* is given by

$$a^*(x;\theta) = B_h(x;\theta)V_h(x;\theta)^{-1},$$

with

$$\begin{aligned} B_h(x;\theta) &= \int_D \partial_\theta h(x, y;\theta)^T p(x, y;\theta)\,\mathrm{d}y, \\ V_h(x;\theta) &= \int_D h(x, y;\theta)h(x, y;\theta)^T p(\Delta, x, y;\theta)\,\mathrm{d}y. \end{aligned}$$

According to Sørensen (2012) most martingale estimating function in the literature can be written as

$$G_n(\theta) = \sum_{i=1}^{n} a(X_{i-1};\theta)\left(f(X_i;\theta) - \pi_\Delta^\theta(f(\theta))(X_{i-1})\right), \tag{3.3}$$

where

$$\pi_s^\theta(f)(x) = \int_D f(y)p(s,x,y;\theta)\,\mathrm{d}y = \mathrm{E}_\theta(f(X_s) \mid X_0 = x).$$

In the one dimensional case ($d = 1$) Kessler and Sørensen (1999) gave an explicit expression, under mild regularity conditions, for $f(X_i;\theta) - \pi_\Delta^\theta(f(\theta))(X_{i-1})$ in (3.3), when the f_j's are eigenfunctions for the generator of the diffusion. That is, the f_j's should satisfy

$$-\lambda_j f(x) = \sum_{k=1}^{d} b_k(x;\theta)\partial_{x_k} f(x) + \frac{1}{2}\sum_{k,l=1}^{d} (\Sigma\Sigma^T)_{kl}(x;\theta)\partial^2_{x_k x_l} f(x),$$

for some real λ_j.

3.3 Prediction-based estimating functions

When the approach with martingale estimating functions is difficult to apply, for example when only a subset of the coordinates in a diffusion is observed, another alternative is to use prediction-based estimating functions. They were proposed in Sørensen (2000) and recently reviewed in Sørensen (2011) and Sørensen (2012). They define a class of estimators that generalize the class of martingale estimating functions in order to make frequentistic inference about the parameters of a stochastic process. We assume that we have observations $X_i, i = 1,\ldots,N$, from a stationary stochastic process governed by a parameter vector $\theta \in \Theta$. The measure related to this process is denoted P_θ.

The procedure involves the definition of $N \in \mathbb{N}$ freely chosen functions f_j

$$f_j : \mathbb{R}^{s+1} \mapsto \mathbb{R},\ j = 1,\ldots,N,$$

potentially depending on $s+1, s \in \mathbb{N}_0$, consecutive observations of data, $(X_i, X_{i-1},\ldots,X_{i-s})$, $i = s,\ldots,N$. These functions can be defined more generally by allowing for dependence on θ as well, but to simplify notation we avoid this. In either case, the f functions should satisfy that

$$\mathrm{E}_\theta\left(f_j(X_{s+1},X_s,\ldots,X_1)\right)^2 < \infty.$$

The set of functions of $X_1,\ldots,X_{i-1}$ with finite variance under P_θ define an L_2 space, L^θ_{i-1}. Let $\mathcal{P}^\theta_{i-1,j}$ denote a closed (linear) subspace of L^θ_{i-1}. The aim is to find the best prediction of $f_j(X_i,\ldots,X_{i-s})$ in $\mathcal{P}^\theta_{i-1,j}$ and minimize a weighted sum of differences between each

f_j and these projections. The set $\mathcal{P}^{\theta}_{i-1,j}$ can be thought of as a set of predictors for $f_j(X_i, \ldots, X_{i-s})$ based on $X_1, \ldots, X_{i-1}$.

We define the prediction-based estimating function as

$$G_n(\theta) = \sum_{i=s+1}^{n} \sum_{j=1}^{N} \Pi_j^{(i-1)} \left(f_j(X_i, X_{i-1}, \ldots, X_{i-s}) - \breve{\pi}_j^{(i-1)}(\theta) \right), \tag{3.4}$$

with $p \times 1$ dimensional weights $\Pi_j^{(i-1)}$ and the l'th coordinate $\left(\Pi_j^{(i-1)}\right)_l \in \mathcal{P}^{\theta}_{i-1,j}$. Furthermore $\breve{\pi}_j^{(i-1)}(\theta)$ is the orthogonal projection in L^{θ}_{i-1} of $f_j(X_i, \ldots, X_{i-s})$, onto the subspace $\mathcal{P}^{\theta}_{i-1,j}$. Hence $\breve{\pi}_j^{(i-1)}$ solves

$$\mathrm{E}_\theta \left(\pi_j^{(i-1)} \left(f_j(X_i, \ldots, X_{i-s}) - \breve{\pi}_j^{(i-1)}(\theta) \right) \right) = 0, \text{ for all } \pi_j^{(i-1)} \in \mathcal{P}^{\theta}_{i-1,j}.$$

For most applications it is practical to consider finite dimensional predictor spaces $\mathcal{P}_{i-1,j}$. Let the dimension be $q_j + 1$. In order to introduce a more compact matrix type notation, let $h_{jk} : \mathbb{R}^r \mapsto \mathbb{R}, j = 1, \ldots, N, k = 0, \ldots, q_j$ with $r \geq s$ and define a set of linearly independent vectors that span the predictor space $\mathcal{P}_{i-1,j}$:

$$Z_{jk}^{(i-1)} = h_{jk}(X_{i-1}, \ldots, X_{i-r}).$$

Furthermore we write $Z_j^{(i-1)} = (Z_{j0}, Z_{j1}^{(i-1)}, \ldots, Z_{jq_j}^{(i-1)})^T$, with $Z_{j0} = 1$, and typically $h_{jk}(x)$ will be the projection onto the k'th coordinate. In some situations we do not include Z_{j0} in $Z_j^{(i-1)}$; see the application in section 3.4. Let 0_k denote the k dimensional zero vector and define

$$Z^{(i-1)} = \begin{pmatrix} Z_1^{(i-1)} & 0_{q_1+1} & \cdots & 0_{q_1+1} \\ 0_{q_2+1} & Z_2^{(i-1)} & \cdots & 0_{q_2+1} \\ \vdots & \vdots & & \vdots \\ 0_{q_N+1} & 0_{q_N+1} & \cdots & Z_N^{(i-1)} \end{pmatrix},$$

of dimension $\sum_{j=1}^{N}(q_j + 1) \times N$, and

$$\begin{aligned} F(X_i, \ldots, X_{i-s}) &= (f_1(X_i, \ldots, X_{i-s}), f_2(X_i, \ldots, X_{i-s}), \ldots, f_N(X_i, \ldots, X_{i-s}))^T, \\ \breve{\pi}^{(i-1)}(\theta) &= (\breve{\pi}_1^{(i-1)}(\theta), \breve{\pi}_2^{(i-1)}(\theta), \ldots \breve{\pi}_N^{(i-1)}(\theta))^T. \end{aligned}$$

We can now define the $\sum_{j=1}^{N}(q_j + 1)$ dimensional vector

$$H^{(i)}(\theta) = Z^{(i-1)} \left(F(X_i, \ldots, X_{i-s}) - \breve{\pi}^{(i-1)}(\theta) \right).$$

Consider the $p \times 1$ dimensional weights $\Pi_j^{(i-1)}$. As noted earlier, each coordinate is an element of $\mathcal{P}^{\theta}_{i-1,j}$ which is spanned by the elements of $Z_j^{(i-1)}$, and can therefore be written

as

$$\left(\Pi_j^{(i-1)}\right)_l = \sum_{k=0}^{q_j} a_{ljk}(\theta) Z_{jk}^{(i-1)},\ l = 1, \ldots, p.$$

The $Z_{jk}^{(i-1)}$'s depend on index i and they are already incorporated into $H^{(i)}(\theta)$ through $Z^{(i-1)}$. With all this notation in place we can rewrite the p-dimensional prediction-based estimating equation as

$$G_n(\theta) = A(\theta) \sum_{i=r+1}^{n} H^{(i)}(\theta), \tag{3.5}$$

where

$$A(\theta) = \begin{pmatrix} a_{110}(\theta) & \cdots & a_{11q_1}(\theta) & \cdots & \cdots & a_{1N0}(\theta) & \cdots & a_{1Nq_N}(\theta) \\ \vdots & & \vdots & & & \vdots & & \vdots \\ a_{p10}(\theta) & \cdots & a_{p1q_1}(\theta) & \cdots & \cdots & a_{pN0}(\theta) & \cdots & a_{1pq_N}(\theta) \end{pmatrix}.$$

The expression for $H^{(i)}(\theta)$ involves the orthogonal projection $\breve{\pi}_j^{(i-1)}(\theta)$ and in order to compute it, we use the following lemma.

Lemma 3.2. *Let V be a Hilbert space with an inner product given by $\langle \cdot, \cdot \rangle$. and let U be a finite dimensional subspace of V, spanned by the basis $(Z_i)_i$. The orthogonal projection of any $x \in V$ onto U, π^x is given by*

$$\pi^x = C^{-1} b,$$

where $C_{ij} = \langle Z_i, Z_j \rangle$ and $b_i = \langle x, Z_i \rangle$ for all $i, j \leq \dim(U)$.

Proof. The orthogonal projection is characterized by the normal equation:

$$\langle x - \pi^x, v \rangle = 0, \quad \forall v \in U. \tag{3.6}$$

Using that there exist $(v_i)_i$ such that $v = \sum_i v_i Z_i$, it follows that

$$\begin{gathered} \langle x, v \rangle = \langle \pi^x, v \rangle \\ \Downarrow \\ \sum_i v_i \langle x, Z_i \rangle = \sum_i v_i \langle \pi^x, Z_i \rangle \quad \forall v \in U. \end{gathered}$$

Therefore

$$\langle x, Z_i \rangle = \langle \pi^x, Z_i \rangle \quad \forall i.$$

Since $\pi^x \in U$ we can write $\pi^x = \sum_j \pi_j^x Z_j$, where subscript j denotes the j'th coordinate, and it follows that

$$\underbrace{\langle x, Z_i \rangle}_{b_i} = \sum_j \pi_j^x \underbrace{\langle Z_j, Z_i \rangle}_{C_{ij}} \quad \forall i, \tag{3.7}$$

which is the coordinate-wise version of the vector equality

$$b = C\pi^x \Rightarrow C^{-1}b = \pi^x.$$

Note that the C matrix is invertible because the Z_i's are linearly independent. □

Using Lemma 3.2 to find $\breve{\pi}_j^{(i-1)}$, we get that the projection onto the space spanned by $Z_{j1}^{(i-1)}, \ldots, Z_{jq_j}^{(i-1)}$ is given by $C_{kj}(\theta) = \mathrm{E}_\theta(Z_k^{(i-1)} Z_j^{(i-1)})$ and $b_j(\theta) = \mathrm{E}_\theta(Z_j^{(i-1)} f_j(X_i))^T$ such that

$$\breve{\pi}_j^{(i-1)}(\theta) = \breve{a}_j^T Z_j^{(i-1)},$$

with $\breve{a}_j(\theta)^T = (\breve{a}_{j0}(\theta), \breve{a}_{j*}(\theta)^T)$ and $\breve{a}_{j*}(\theta) = C_j(\theta)^{-1} b_j(\theta)$. Note that

$$\breve{a}_{j0}(\theta) = \mathrm{E}_\theta\left(f_j(X_{s+1}, \ldots, X_1; \theta)\right) - \sum_{k=1}^{q_j} \breve{a}_{jk}(\theta)\, \mathrm{E}_\theta(Z_{jk}^{(r)}).$$

This means that we have an expression for the predictors, and thus of $H^{(i)}(\theta)$, when the f functions have been chosen. Now we would like to find the 'best' choice of the weight matrix $A(\theta)$. The notion 'best' refers to the Godambe optimal estimating function within the class of estimating functions given by (3.5): It is the function that minimizes the mean square error to the score function, which would ideally have defined the optimal estimating equation. Under regularity conditions given in Sørensen (2011), the Godambe optimal weight $A^*(\theta)$ can be found in the following way for general f's that may depend on θ. Let

$$\begin{aligned}
U(\theta)^T &= \mathrm{E}_\theta\left(Z^{(i-1)} \partial_{\theta^T} F(X_i, \ldots, X_{i-s})\right), \\
\bar{M}_n(\theta) &= \mathrm{E}_\theta\left(H^{(r+1)}(\theta) H^{(r+1)}(\theta)^T\right) \\
&\quad + \sum_{k=1}^{n-r-1} \frac{n-r-k}{n-r} \left\{ \mathrm{E}_\theta\left(H^{(r+1)}(\theta) H^{(r+1+k)}(\theta)^T\right) \right. \\
&\quad \left. + \mathrm{E}_\theta\left(H^{(r+1+k)}(\theta) H^{(r+1)}(\theta)^T\right) \right\}, \\
D(\theta) &= \mathrm{E}_\theta\left(Z^{(i-1)} (Z^{(i-1)})^T\right).
\end{aligned}$$

Here $\partial_\theta^T F$ denotes the partial derivative of F with respect to all entries of θ. Then

$$A^*(\theta) = (U(\theta) - \partial_\theta \breve{a}(\theta)^T D(\theta)) \bar{M}_n(\theta)^{-1}, \tag{3.8}$$

where

$$\breve{a}(\theta) = (\breve{a}_1(\theta), \breve{a}_2(\theta), \ldots, \breve{a}_N(\theta)).$$

As already mentioned we only consider f's independent of θ and therefore we get that $U(\theta) = 0$.

One of the regularity conditions requires that $p \leq N + \sum_{j=1}^{N} q_j$. Note that A is a $p \times \left(N + \sum_{j=1}^{N} q_j\right)$ matrix since it would not make sense to perform estimation with p parameters from a system with less than p equations. Unfortunately it is very difficult to compute $A^*(\theta)$ due to the partial derivative involved in $U(\theta)$ and it may also be complicated to recompute A^* for each new evaluation in the optimization procedure. There are some steps that can be taken in order to simplify matters. First, one could compute the derivative of $\breve{a}(\theta)$ only once, for some initial value of θ, and then reuse this matrix. This can be done at the cost of some efficiency. Secondly, one could compute $\bar{M}_n$ for only the first two terms ($n = r + 2$) and as an approximation assume that the correlation between H^i and H^{i+j} is zero for any $j \geq 2$.

3.3.1 Differentiation in R

Algebraic differentiation and evaluation is not a strong feature of R, although it would be convenient in order to automatize the procedure of evaluating the derivative of $\breve{a}(\theta)$ from (3.8), with respect to θ. One option is to use another program such as Maple and then import the result back to R. This is not a practical solution because the estimating function must be evaluated for many different values of θ. It is however possible to use R directly. Following the idea of Niels Richard Hansen (personal correspondence) one could do the following: Define all functions of interest in a list only once, say f and g.

```
simpleFun <- list(f=quote(theta1+3*theta2),g=quote(theta3+theta1))
```

Next, create a function that changes input from f, g notation to `theta` notation and use `deriv()` to evaluate the result:

```
compDeriv <- function(expr,arglist,...) {
    compFun <- do.call("substitute",list(substitute(expr),arglist))
    return(deriv(compFun,...))
}
```

Finally make a function that finds the partial derivatives as a function of the parameters:

```
gradComp <- compDeriv(f^2*g,simpleFun,c("theta1","theta2","theta3"),TRUE)
> gradComp(2,3,4)
[1] 726
attr(,"gradient")
     theta1 theta2 theta3
[1,]    253    396    121
```

This approach works fine for simple compositions of functions. The drawback is that it does not support derivatives of composite functions like for instance $\partial_x f(g(x))$. Thus for applications the length of the expressions would easily turn into unmanageable sizes. On the technical side, another potential problem with this solution is that the combined behavior of `do.call()` and `substitute()` may change in future `R` versions, thus causing the function to behave unexpectedly. (As of version 3.0.2 it works fine, though.)

3.4 Prediction-based Estimating Functions for the partially observed Ornstein-Uhlenbeck process

Consider the stationary two-dimensional OU model with asymptotic mean equal to zero:

$$\mathrm{d}V_t = -BV_t \,\mathrm{d}t + \Sigma \,\mathrm{d}W_t, \tag{3.9}$$

where $V_t = (X_t, Y_t)^T$ and

$$B = \begin{pmatrix} \beta_{11} & \beta_{12} \\ \beta_{21} & \beta_{22} \end{pmatrix}, \; \Sigma = \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix}. \tag{3.10}$$

Let $\theta = (\beta_{11}, \beta_{12}, \beta_{21}, \beta_{22}, \sigma_1^2, \sigma_2^2)^T$ denote the parameters in the model and assume discrete observations from the first coordinate are available at time points $t_0, t_1, \ldots, t_n$ with $t_i - t_{i-1} = \Delta$. As will be shown in chapter 7, not all parameters are identifiable from the first marginal only, and we will assume that enough parameters are known a priori in order to make the unknown parameters identifiable. In this section we write down the expressions involved in applying prediction-based estimating functions to the two dimensional Ornstein-Uhlenbeck model. First we define the functions to predict

$$\begin{aligned} f_1(x) &= x, \\ f_2(x) &= x^2, \end{aligned}$$

such that the number of functions to be predicted is $N = 2$. Next, define the space from which the f functions are to be predicted:

$$\begin{aligned} Z_1^{(i-1)} &= (X_{i-1}, X_{i-2})^T, \\ Z_2^{(i-1)} &= (1, X_{i-1}, X_{i-1}^2, X_{i-2}, X_{i-2}^2)^T. \end{aligned}$$

We do not include an intercept in $Z_1^{(i-1)}$ because the X_i's have mean zero by assumption. Then the dimension of the predictor spaces $\mathcal{P}_{i-1,j}^{\theta}$ become $0 + q_1 = 2$, and $1 + q_2 = 5$.

Note for example that

$$\begin{aligned} Z_1^{(2)} &= (Z_{11}^{(1)}, Z_{21}^{(1)}) = (X_1, X_0), \\ Z_2^{(2)} &= (Z_{20}^{(1)}, Z_{21}^{(1)}, \ldots, Z_{24}^{(1)}) = (1, X_1, X_1^2, X_0, X_0^2). \end{aligned}$$

Note also that $N - 1 + q_1 + q_2 = 7$. This number should be larger than the number of parameters to be estimated in the model.

The OU-process is assumed stationary and we can therefore denote for all $i, h > 0$

$$\begin{aligned} \mathrm{E}_\theta(X_i) &= 0, \\ \mathrm{Var}_\theta(X_i) &:= \gamma, \\ \mathrm{E}_\theta(X_i^p X_{i+h}^q) &:= \nu(h,p,q). \end{aligned}$$

For example $\mathrm{Cov}_\theta(X_0, X_1) = \nu(1,1,1)$ because $\mathrm{E}_\theta(X_i) = 0$, and $\gamma = \nu(0,1,1)$.

We get that

$$C_1(\theta) = \mathrm{Cov}\begin{pmatrix} X_1 \\ X_0 \end{pmatrix} = \begin{pmatrix} \gamma & \nu(1,1,1) \\ \nu(1,1,1) & \gamma \end{pmatrix}.$$

Also, for $i = 1, 2$

$$b_1(\theta)_i = \mathrm{Cov}_\theta(Z_{1i}^{(1)}, f_1(X_2)) = \mathrm{Cov}_\theta(X_{2-i}, X_2),$$

so that

$$b_1(\theta) = \begin{pmatrix} \nu(1,1,1) \\ \nu(2,1,1) \end{pmatrix}.$$

Then

$$\breve{a}_{1*}(\theta) = C_1(\theta)^{-1} b_1(\theta) = \left(\nu(1,1,1)^2 - \gamma^2\right)^{-1} \begin{pmatrix} \nu(1,1,1)\left(\nu(2,1,1) - \gamma\right) \\ \nu(1,1,1)^2 - \gamma\nu(2,1,1) \end{pmatrix}.$$

For the first entry of $\breve{a}_1(\theta)$ it holds that

$$\breve{a}_{10}(\theta) = \mathrm{E}\left(f_1(X_1)\right) - \breve{a}_{11}(\theta)\,\mathrm{E}_\theta(X_1) - \breve{a}_{12}(\theta)\,\mathrm{E}_\theta(X_2) = 0,$$

and we do not include it in $\breve{a}_1(\theta)$. Thus $\breve{a}_1(\theta) = \breve{a}_{1*}(\theta)$.

Similarly for $j = 2$, where we use that all odd moments of X are zero, as well as all moments of $X_i^p X_{i+h}^q$, where $p + q$ odd:

$$C_2(\theta) = \mathrm{Cov}\begin{pmatrix} X_1 \\ X_1^2 \\ X_0 \\ X_0^2 \end{pmatrix} = \begin{pmatrix} \gamma & 0 & \nu(1,1,1) & 0 \\ 0 & 2\gamma^2 & 0 & \nu(1,2,2) - \gamma^2 \\ \nu(1,1,1) & 0 & \gamma & 0 \\ 0 & \nu(1,2,2) - \gamma^2 & 0 & 2\gamma^2 \end{pmatrix}. \tag{3.11}$$

Also, for $i = 1, 2, 3, 4$

$$b_2(\theta)_i = \mathrm{Cov}_\theta(Z_{2i}^{(1)}, f_2(X_2)),$$

so that

$$b_2(\theta) = \begin{pmatrix} 0 \\ \nu(1,2,2) - \gamma^2 \\ 0 \\ \nu(2,2,2) - \gamma^2 \end{pmatrix}. \tag{3.12}$$

Then

$$\breve{a}_{2*}(\theta) = \left(-3\gamma^4 + \nu(1,2,2)^2 - 2\nu(1,2,2)\gamma^2\right)^{-1} \times \begin{pmatrix} 0 \\ \left(\nu(1,2,2) - \gamma^2\right)\left(\nu(2,2,2) - 3\gamma^2\right) \\ 0 \\ \nu(1,2,2)^2 - 2\nu(1,2,2)\gamma^2 + 3\gamma^4 - 2\gamma^2\nu(2,2,2) \end{pmatrix},$$

and

$$\breve{a}_{20}(\theta) = \gamma\left(1 - \breve{a}_{22}(\theta) - \breve{a}_{24}(\theta)\right) = \frac{\gamma\left(3\gamma^2 - \nu(2,2,2)\right)}{\gamma^2 + \nu(1,2,2)}.$$

The first entry of $\breve{a}_2(\theta)$ is 1, so that $\breve{a}_2(\theta) = (1, \breve{a}_{2*}(\theta))^T$, and we can now compute $\breve{\pi}_j(\theta), j = 1, 2$:

$$\breve{\pi}_j(\theta) = \breve{a}_j(\theta)^T Z_j^{(i-1)}.$$

Hence the estimating equation can be written

$$\begin{aligned} G_n(\theta) &= A_n^*(\theta) \sum_{i=3}^{n} Z_j^{(i-1)} \left(F(X_i) - \breve{\pi}^{(i-1)}(\theta)\right) \\ &= A_n^*(\theta) \sum_{i=3}^{n} \begin{pmatrix} X_i X_{i-1} - \breve{\pi}_1^{(i-1)}(\theta) X_{i-1} \\ X_i X_{i-2} - \breve{\pi}_1^{(i-1)}(\theta) X_{i-2} \\ X_i^2 - \breve{\pi}_2^{(i-1)} \\ X_i^2 X_{i-1} - \breve{\pi}_2^{(i-1)} X_{i-1} \\ X_i^2 X_{i-1}^2 - \breve{\pi}_2^{(i-1)} X_{i-1}^2 \\ X_i^2 X_{i-2} - \breve{\pi}_2^{(i-1)} X_{i-2} \\ X_i^2 X_{i-2}^2 - \breve{\pi}_2^{(i-1)} X_{i-2}^2 \end{pmatrix} \\ &= A_n^*(\theta) \sum_{i=3}^{n} \begin{pmatrix} X_{i-1}\left(X_i - \breve{a}_1(\theta)^T Z_1^{(i-1)}\right) \\ X_{i-2}\left(X_i - \breve{a}_1(\theta)^T Z_1^{(i-1)}\right) \\ X_i^2 - \breve{a}_2(\theta)^T Z_2^{(i-1)} \\ X_{i-1}\left(X_i^2 - \breve{a}_2(\theta)^T Z_2^{(i-1)}\right) \\ X_{i-1}^2\left(X_i^2 - \breve{a}_2(\theta)^T Z_2^{(i-1)}\right) \\ X_{i-2}\left(X_i^2 - \breve{a}_2(\theta)^T Z_2^{(i-1)}\right) \\ X_{i-2}^2\left(X_i^2 - \breve{a}_2(\theta)^T Z_2^{(i-1)}\right) \end{pmatrix}. \end{aligned} \tag{3.13}$$

Note that the sum runs from $r+1 = 2+1$, where r is the number of lags used in the predictions.

The calculations related to $A_n^*(\theta)$ are quite complicated, but it involves the derivative with respect to θ of

$$\breve{a}(\theta) = \begin{pmatrix} \frac{\nu(1,1,1)(\nu(2,1,1)-\gamma)}{\nu(1,1,1)^2-\gamma^2} \\ \frac{\nu(1,1,1)^2-\gamma\nu(2,1,1)}{\nu(1,1,1)^2-\gamma^2} \\ \frac{\gamma\left(3\gamma^2-\nu(2,2,2)\right)}{\gamma^2+\nu(1,2,2)} \\ 0 \\ \frac{\left(\nu(1,2,2)-\gamma^2\right)\left(\nu(2,2,2)-3\gamma^2\right)}{-3\gamma^4+\nu(1,2,2)^2-2\nu(1,2,2)\gamma^2} \\ 0 \\ \frac{\nu(1,2,2)^2-2\nu(1,2,2)\gamma^2+3\gamma^4-2\gamma^2\nu(2,2,2)}{-3\gamma^4+\nu(1,2,2)^2-2\nu(1,2,2)\gamma^2} \end{pmatrix}. \tag{3.14}$$

Since two of the entries are zero, the rank of $\partial_\theta \breve{a}(\theta)$ will never be larger than five in this setting.

3.4.1 Moment calculations

In the following we write $V_{t_i} = (V_i^{(1)}, V_i^{(2)}) = (X_i, Y_i)$ and change between the left and right hand side notation. In order to compute $G_n(\theta)$ in (3.13) we need to compute the terms involved in $\breve{a}_j(\theta)$. This entails computing elements of the form

$$\mathrm{E}_\theta(X_i^p X_{i-h}^q X_{i-h-k}^r X_{i-h-k-l}^s), \tag{3.15}$$

for $p, q, r, s \in \{0, 1, 2\}$ and $i > h > k > l > 0$.

For the OU-model the first coordinate is

$$V_t^{(1)} = \sum_{j=1}^{2} \left(e^{-B(t-t_0)}\right)_{1j} V_{t_0}^{(j)} + \int_{t_0}^{t} \left(e^{-B(t-s)}\Sigma\right)_{1j} \mathrm{d}W_s^{(j)},$$

and we denote, because of stationarity, for all $t \geq 0$

$$\mathrm{E}_\theta(V_t) = 0, \quad \mathrm{Var}_\theta(V_t) = \begin{pmatrix} \gamma & \gamma_{12} \\ \gamma_{12} & \gamma_{22} \end{pmatrix}.$$

We will write

$$\left(e^{-Bt}\right)_{ij} = a_{ij}(t),$$

and let

$$\left(\mathrm{Var}_\theta(V_t \mid V_{t-s})\right)_{ij} = \omega_{ij}(t-s).$$

We shall repeatedly make use of the binomial theorem in the following to compute the moments from (3.15), and we therefore state it here:

$$(a+b)^n = \sum_{i=0}^{n} \begin{pmatrix} n \\ i \end{pmatrix} a^{n-i} b^i.$$

Single moments

For $X_i \sim \mathcal{N}(0,\gamma), \forall i$, all higher order moments of X_i can be described as functions of γ.

$$\mathrm{E}_\theta(X_i^p) = \begin{cases} \gamma^{p/2}(p-1)!!, & p \text{ even} \\ 0, & p \text{ odd} \end{cases}$$

where $(2n-1)!! = \prod_{i=1}^{n}(2i-1)$, for $n \in \mathbb{N}_0$. For convenience let $(-1)!! := 1$. Thus

$$\begin{aligned} \mathrm{E}_\theta(X_i^2) &= \gamma, \\ \mathrm{E}_\theta(X_i^4) &= 3\gamma^2, \\ \mathrm{E}_\theta(X_i^6) &= 15\gamma^3, \\ \mathrm{E}_\theta(X_i^8) &= 105\gamma^4. \end{aligned}$$

Mixed moments of order 2: $\mathrm{E}\left(X_i^p X_{i-h}^q\right)$

Let $q, p \in \mathbb{N}_0$ and $i > h$, and note that for even m,

$$\mathrm{E}\left[\left(\sum_{j=1}^{2}\int_{(i-h)\Delta}^{i\Delta} a_{1j}(i\Delta - s)\sigma_j \ \mathrm{d}W_s^{(j)}\right)^m\right] = \omega_{11}(h\Delta)^{m/2}(m-1)!!, \tag{3.16}$$

so that for p even,

$$\begin{aligned} &\mathrm{E}\left(X_i^p \mid V_{i-h}\right) \\ &= \mathrm{E}\left[\left(\sum_{j=1}^{2} a_{1j}(h\Delta)V_{i-h}^{(j)} + \int_{(i-h)\Delta}^{i\Delta} a_{1j}(i\Delta - u)\sigma_j \ \mathrm{d}W_u^{(j)}\right)^p \mid V_{i-h}\right] \\ &= \sum_{m=0}^{p}\binom{p}{m}\mathrm{E}\left[\left(\sum_{j=1}^{2} a_{1j}(h\Delta)V_{i-h}^{(j)}\right)^{p-m}\left(\sum_{j=1}^{2}\int_{(i-h)\Delta}^{i\Delta} a_{1j}(i\Delta - u)\sigma_j \ \mathrm{d}W_u^{(j)}\right)^m \mid V_{i-h}\right] \\ &= \sum_{m=0}^{p}\binom{p}{m}\left(\sum_{j=1}^{2} a_{1j}(h\Delta)V_{i-h}^{(j)}\right)^{p-m}\mathrm{E}\left[\left(\sum_{j=1}^{2}\int_{(i-h)\Delta}^{i\Delta} a_{1j}(i\Delta - u)\sigma_j \ \mathrm{d}W_u^{(j)}\right)^m\right] \\ &= \sum_{m=0}^{p}\binom{p}{m}\left(\sum_{j=1}^{2} a_{1j}(h\Delta)V_{i-h}^{(j)}\right)^{p-m}\omega_{11}(h\Delta)^{m/2}(m-1)!!1_{(m \text{ even})} \\ &= \sum_{m=0}^{\lfloor p/2\rfloor}\binom{p}{2m}\left(\sum_{j=1}^{2} a_{1j}(h\Delta)V_{i-h}^{(j)}\right)^{p-2m}\omega_{11}(h\Delta)^{m}(2m-1)!!, \end{aligned}$$

where $\lfloor x \rfloor = a$ means that $a \in \mathbb{N}_0$ and $x - 1 \leq a \leq x$. This leads to

$$\begin{aligned}
&\mathrm{E}\left(X_i^p X_{i-h}^q\right) \\
&= \mathrm{E}\left(X_{i-h}^q \mathrm{E}\left(X_i^p \mid V_{i-h}\right)\right) \\
&= \mathrm{E}\left(X_{i-h}^q \sum_{m=0}^{\lfloor p/2 \rfloor} \binom{p}{2m} \left(\sum_{j=1}^{2} a_{1j}(h\Delta) V_{i-h}^{(j)}\right)^{p-2m} \omega_{11}(h\Delta)^m (2m-1)!!\right) \\
&= \sum_{m=0}^{\lfloor p/2 \rfloor} \binom{p}{2m} \omega_{11}(h\Delta)^m (2m-1)!! \, \mathrm{E}\left(X_{i-h}^q \left(\sum_{j=1}^{2} a_{1j}(h\Delta) V_{i-h}^{(j)}\right)^{p-2m}\right) \\
&= \sum_{m=0}^{\lfloor p/2 \rfloor} \sum_{l=0}^{p-2m} g(p,m,l) \omega_{11}(h\Delta)^m a_{11}(h\Delta)^{p-2m-l} a_{12}(h\Delta)^l \, \mathrm{E}\left(X_{i-h}^{q+p-2m-l} \left(V_{i-h}^{(2)}\right)^l\right),
\end{aligned}$$

where

$$g(p,m,l) = \binom{p-2m}{l} \binom{p}{2m} (2m-1)!!.$$

This leaves yet another mixed term to be evaluated.

First we need to evaluate the following conditional moment:

$$\begin{aligned}
&\mathrm{E}\left[\left(V_i^{(2)}\right)^q \mid V_i^{(1)}\right] \\
&= \mathrm{E}\left[\left(V_i^{(2)} - \mathrm{E}\left[V_i^{(2)} \mid V_i^{(1)}\right] + \mathrm{E}\left[V_i^{(2)} \mid V_i^{(1)}\right]\right)^q \mid V_i^{(1)}\right] \\
&= \sum_{m=0}^{q} \binom{q}{m} \mathrm{E}\left[\left\{V_i^{(2)} - \mathrm{E}\left[V_i^{(2)} \mid V_i^{(1)}\right]\right\}^{q-m} \mid V_i^{(1)}\right] \left\{\mathrm{E}\left[V_i^{(2)} \mid V_i^{(1)}\right]\right\}^m \\
&= \sum_{m=0}^{q} \binom{q}{m} \mathrm{Var}\left(V_i^{(2)} \mid V_i^{(1)}\right)^{(q-m)/2} (q-m-1)!! 1_{(q-m \text{ even})} \left(\frac{\gamma_{12} V_i^{(1)}}{\gamma}\right)^m \\
&= \sum_{m=0}^{q} \binom{q}{m} \left(\gamma_{22} - \frac{\gamma_{12}^2}{\gamma}\right)^{(q-m)/2} \left(\frac{\gamma_{12} V_i^{(1)}}{\gamma}\right)^m (q-m-1)!! 1_{(q-m \text{ even})}. \qquad (3.17)
\end{aligned}$$

Here we have used the well known formula for finding the conditional mean and variance of a multivariate Gaussian distribution. Thus

$$\begin{aligned}
&\mathrm{E}\left(X_i^p \left(V_i^{(2)}\right)^q\right) \\
&= \mathrm{E}\left[X_i^p \mathrm{E}\left[\left(V_i^{(2)}\right)^q \mid X_i\right]\right] \\
&= \sum_{m=0}^{q} \binom{q}{m} \left(\gamma_{22} - \frac{\gamma_{12}^2}{\gamma}\right)^{(q-m)/2} \left(\frac{\gamma_{12}}{\gamma}\right)^m (q-m-1)!! \, \mathrm{E}\left[X_i^{p+m}\right] 1_{(q-m \text{ even})}
\end{aligned}$$

In conclusion, for $p, q \in \mathbb{N}_0, h < i$,

$$
\begin{aligned}
&\mathrm{E}\left(X_i^p X_{i-h}^q\right) \\
&= \sum_{m=0}^{\lfloor p/2 \rfloor} \sum_{l=0}^{p-2m} g(p,m,l)\omega_{11}(h\Delta)^m a_{11}(h\Delta)^{p-2m-l} a_{12}(h\Delta)^l \,\mathrm{E}\left(X_{i-h}^{q+p-2m-l}\left(V_{i-h}^{(2)}\right)^l\right) \\
&= \sum_{m=0}^{\lfloor p/2 \rfloor} \sum_{l=0}^{p-2m} g(p,m,l)\omega_{11}(h\Delta)^m a_{11}(h\Delta)^{p-2m-l} a_{12}(h\Delta)^l \\
&\quad \sum_{n=0}^{l} \binom{l}{n} \left(\gamma_{22} - \frac{\gamma_{12}^2}{\gamma}\right)^{(l-n)/2} \left(\frac{\gamma_{12}}{\gamma}\right)^n (l-n-1)!! \\
&\quad \mathrm{E}\left(X_{i-h}^{q+p-2m-l+n}\right) 1_{(l-n \text{ even})} 1_{(q+p-2m-l+n \text{ even})} \\
&= \sum_{m=0}^{\lfloor p/2 \rfloor} \sum_{l=0}^{p-2m} \sum_{n=0}^{l} \tilde{g}(p,m,l,n)\omega_{11}(h\Delta)^m a_{11}(h\Delta)^{p-2m-l} a_{12}(h\Delta)^l \\
&\quad \left(\gamma_{22} - \frac{\gamma_{12}^2}{\gamma}\right)^{(l-n)/2} \left(\frac{\gamma_{12}}{\gamma}\right)^n \mathrm{E}\left(X_{i-h}^{q+p-2m-l+n}\right) 1_{(q+p-2m-l+n \text{ even})} 1_{(l-n \text{ even})},
\end{aligned}
$$

where

$$
\tilde{g}(p,m,l,n) = \binom{p-2m}{l}\binom{p}{2m}\binom{l}{n}(2m-1)!!(l-n-1)!!.
$$

Example 3.3.

$$
\mathrm{E}_\theta(X_1 X_{1+h}) = a_{11}(h\Delta)\gamma + a_{12}(h\Delta)\gamma_{12}.
$$

♦

Example 3.4.

$$
\begin{aligned}
\mathrm{E}_\theta(X_1^2 X_{1+h}^2) &= 3a_{11}(h\Delta)^2\gamma^2 + 6a_{11}(h\Delta)a_{12}(h\Delta)\gamma_{12}\gamma \\
&\quad + a_{12}(h\Delta)^2\gamma_{22}\gamma + 2a_{12}(h\Delta)^2\gamma_{12}^2 + \omega_{11}(h\Delta)\gamma.
\end{aligned}
$$

♦

Mixed moments of order 3: $\mathrm{E}\left(X_i^p X_{i-h}^q X_{i-h-k}^r\right)$

Assume $i > h > k$ and $i > h + k$. Using (3.16) we get,

$$
\begin{aligned}
&\mathrm{E}\left(X_i^p X_{i-h}^q X_{i-h-k}^r\right) \\
&= \mathrm{E}\left[\left(\sum_{j=1}^{2} a_{1j}(h\Delta)V_{i-h}^{(j)} + \int_{(i-h)\Delta}^{i\Delta} a_{1j}(i\Delta - u)\sigma_j \ \mathrm{d}W_u^{(j)}\right)^p X_{i-h}^q X_{i-h-k}^r\right] \\
&= \sum_{m=0}^{p} \binom{p}{m} \omega_{11}(h\Delta)^{m/2}(m-1)!!\,\mathrm{E}\left[\left(\sum_{j=1}^{2} a_{1j}(h\Delta)V_{i-h}^{(j)}\right)^{p-m} X_{i-h}^q X_{i-h-k}^r\right] 1_{(m \text{ even})} \\
&= \sum_{m=0}^{p} \binom{p}{m} \omega_{11}(h\Delta)^{m/2}(m-1)!!1_{(m \text{ even})} \\
&\sum_{s=0}^{p-m} \binom{p-m}{s} a_{11}(h\Delta)^{p-m-s} a_{12}(h\Delta)^s\,\mathrm{E}\left[X_{i-h}^{p-m-s+q}\left(V_{i-h}^{(2)}\right)^s X_{i-h-k}^r\right] \\
&= \sum_{m=0}^{\lfloor p/2 \rfloor} \sum_{s=0}^{p-2m} g(p,m,s)\omega_{11}(h\Delta)^m a_{11}(h\Delta)^{p-2m-s} a_{12}(h\Delta)^s\,\mathrm{E}\left[X_{i-h}^{p-2m-s+q}\left(V_{i-h}^{(2)}\right)^s X_{i-h-k}^r\right].
\end{aligned}
$$

Using (3.17) we obtain

$$
\begin{aligned}
&\mathrm{E}\left[X_i^p\left(V_i^{(2)}\right)^s X_{i-k}^r\right] \\
&= \mathrm{E}\left[X_i^p X_{i-k}^r\,\mathrm{E}\left[\left(V_i^{(2)}\right)^s \mid V_i^{(1)}\right]\right] \\
&= \sum_{m=0}^{s} \binom{s}{m} (s-m-1)!!\left(\gamma_{22} - \frac{\gamma_{12}^2}{\gamma}\right)^{\frac{s-m}{2}} \left(\frac{\gamma_{12}}{\gamma}\right)^m \mathrm{E}\left[X_i^{p+m} X_{i-k}^r\right] 1_{(s-m \text{ even})}.
\end{aligned}
$$

In conclusion:

$$
\begin{aligned}
&\mathrm{E}\left(X_i^p X_{i-h}^q X_{i-h-k}^r\right) \\
&= \sum_{m=0}^{\lfloor p/2 \rfloor} \sum_{s=0}^{p-2m} \sum_{l=0}^{s} \tilde{g}(p,m,s,l)\omega_{11}(h\Delta)^m a_{11}(h\Delta)^{p-2m-s} a_{12}(h\Delta)^s \\
&\left(\gamma_{22} - \frac{\gamma_{12}^2}{\gamma}\right)^{\frac{s-l}{2}} \left(\frac{\gamma_{12}}{\gamma}\right)^l \mathrm{E}\left[X_{i-h}^{p-2m-s+q+l} X_{i-h-k}^r\right] 1_{(s-l \text{ even})}.
\end{aligned}
$$

Mixed moments of order 4: $\mathrm{E}\left(X_i^p X_{i-h}^q X_{i-h-k}^r X_{i-h-k-l}^s\right)$

Let $i > h > k > l, i > h + k, i > h + k + l, h > k + l$. We only need these moments for $p, q, r, s \in \{1, 2\}$.

Assume $p = 1$. Then, using (3.17) it follows that,

$$\begin{aligned}
&\mathrm{E}\left(X_i X_{i-h}^q X_{i-h-k}^r X_{i-h-k-l}^s\right) \\
&= \mathrm{E}\left[\left(\sum_{j=1}^{2} a_{1j}(h\Delta)V_{i-h}^{(j)} + \int_{(i-h)\Delta}^{i\Delta} a_{1j}(i\Delta - u)\sigma_j \ \mathrm{d}W_u^{(j)}\right) X_{i-h}^q X_{i-h-k}^r X_{i-h-k-l}^s\right] \\
&= \mathrm{E}\left[a_{11}(h\Delta)X_{i-h}^{q+1} X_{i-h-k}^r X_{i-h-k-l}^s + a_{12}(h\Delta)V_{i-h}^{(2)} X_{i-h}^q X_{i-h-k}^r X_{i-h-k-l}^s\right] \\
&= \mathrm{E}\left(X_{i-h}^{q+1} X_{i-h-k}^r X_{i-h-k-l}^s\right)\left(a_{11}(h\Delta) + a_{12}(h\Delta)\frac{\gamma_{12}}{\gamma}\right).
\end{aligned}$$

Assume $p = 2$. Now

$$\begin{aligned}
&\mathrm{E}\left(X_i^2 X_{i-h}^q X_{i-h-k}^r X_{i-h-k-l}^s\right) \\
&= \mathrm{E}\left[\left(\sum_{j=1}^{2} a_{1j}(h\Delta)V_{i-h}^{(j)} + \int_{(i-h)\Delta}^{i\Delta} a_{1j}(i\Delta - u)\sigma_j \ \mathrm{d}W_u^{(j)}\right)^2 X_{i-h}^q X_{i-h-k}^r X_{i-h-k-l}^s\right] \\
&= a_{11}^2(h\Delta)\,\mathrm{E}\left[X_{i-h}^{q+2} X_{i-h-k}^r X_{i-h-k-l}^s\right] + a_{12}^2(h\Delta)\,\mathrm{E}\left[\left(V_{i-h}^{(2)}\right)^2 X_{i-h}^q X_{i-h-k}^r X_{i-h-k-l}^s\right] \\
&+ 2a_{11}(h\Delta)a_{12}(h\Delta)\,\mathrm{E}\left[X_{i-h}^{q+1} V_{i-h}^{(2)} X_{i-h-k}^r X_{i-h-k-l}^s\right] \\
&+ \omega_{11}(h\Delta)\,\mathrm{E}\left(X_{i-h}^q X_{i-h-k}^r X_{i-h-k-l}^s\right) \\
&= \mathrm{E}\left(X_{i-h}^q X_{i-h-k}^r X_{i-h-k-l}^s\right)\left(\omega_{11}(h\Delta) + a_{12}^2(h\Delta)\left(\gamma_{22} - \frac{\gamma_{12}}{\gamma}\right)\right) \\
&+ \mathrm{E}\left(X_{i-h}^{q+2} X_{i-h-k}^r X_{i-h-k-l}^s\right)\left(a_{11}(h\Delta) + a_{12}(h\Delta)\frac{\gamma_{12}}{\gamma}\right)^2.
\end{aligned}$$

Now we have expressions for all mixed moments needed to compute $G_n(\theta)$ from (3.13).

3.5 Implementation

The method of prediction-based estimating functions was implemented in `R` and tested on simulated data. We simulated the Ornstein-Uhlenbeck process from (3.9) and (3.10), for 1.000 observations with equidistant time step between observations, $\Delta = 0.1$, with

$$B = \begin{pmatrix} 5 & 1 \\ -2 & 1 \end{pmatrix}, \Sigma = \begin{pmatrix} 0.5 & 0 \\ 0 & 0.3 \end{pmatrix}.$$

The second coordinate was assumed to be unobserved. The most involved computations relates to finding $A^*(\theta)$ from (3.8). To simplify computations we approximated $\bar{M}_n(\theta)$ with the empirical covariance matrix of $H(\theta)$ computed for 10.000 data sets with true parameter values. The term $\partial_\theta \breve{a}(\theta)$ was found using Maple and evaluated in the true parameters. This

procedure is obviously not possible for real data, but is was done in order to investigate how the procedure would behave for a fixed weight matrix, which potentially could simplify computations.

The three drift parameters β_{11}, β_{12} and β_{21} were estimated simultaneously for 5.000 data sets. For each data set the estimating function $G_{1000}(\theta)$ was 3×1 dimensional and the solution was found by taking the sum of squared coordinates and finding the minimum. In some cases this resulted in lack of convergence of the algorithm searching for the minimum, and such estimates were removed from the final sample resulting in 3783 sets of estimates. Visual inspection of some of the data sets where convergence was not reached, suggested that this was an error that could be attributed to the 'squaring and summing' procedure.

Figure 3.1 shows histograms of the marginal distribution of the three parameter estimators, with the red horizontal line denoting the true parameter value. The results are not too

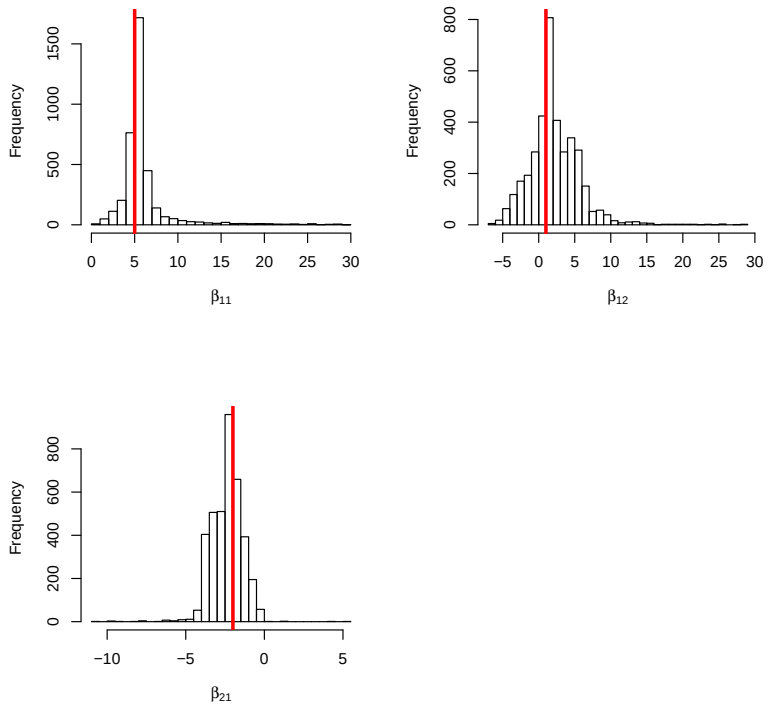

Figure 3.1: *histograms of the estimates of the three parameters* $\beta_{11}, \beta_{12}, \beta_{21}$. *Vertical lines denote the true parameter value.*

convincing, as the variances of the estimators are relatively large.

Figure 3.2 shows a pairs plot of the estimates for the three parameters. The red lines are the true parameter values, the green are the mean of the estimates and the blue lines

denote the medians of the estimates. Clearly there is a strong negative correlation between β_{12} and β_{21}.

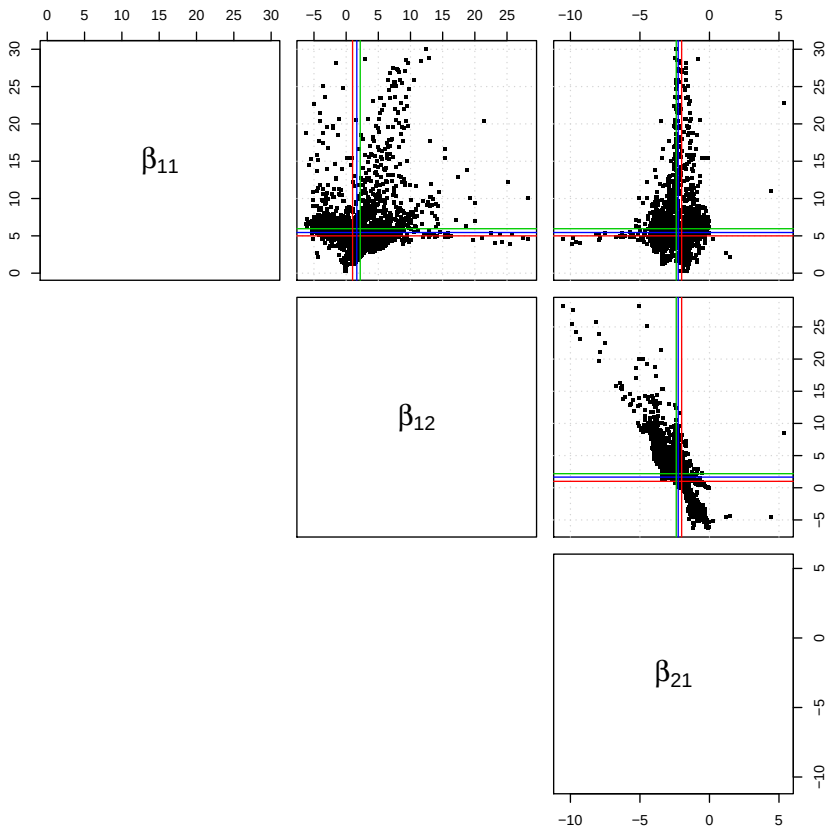

Figure 3.2: *Pairs plot of estimates of the three drift parameters β_{11}, β_{12} and β_{21}. Red lines are true parameter values, green are means of estimators, and blue lines denote the sample medians of the estimators.*

In conclusion the method seems to work, although not as convincing as one might hope. Thus we decide to try a Bayesian approach, which is the subject of chapters 4, 5 and 6.

4

Bayesian statistics and MCMC methods

4.1 The basic Bayesian framework

In this chapter we give a short introduction to some of the Bayesian ideas that are used in the subsequent chapters. We will introduce the basic Bayesian framework and then move on to some of the important MCMC methods typically used in the Bayesian setting.

Let θ denote the parameter of a statistical model with observations $X = \{X_1, \ldots, X_n\}$ and let π be generic notation for a distribution. The Bayesian approach requires one to specify the prior belief in θ via a probability distribution on θ. After the data is observed, this prior knowledge is updated using Bayes formula, to obtain the posterior distribution of θ conditional on the observed data.

$$\pi(\theta \mid X) = \frac{\pi(x \mid \theta)\pi(\theta)}{\int \pi(X, \theta)\, \mathrm{d}\theta}. \tag{4.1}$$

Here $\pi(\theta)$ denotes the prior distribution of θ, $\pi(X \mid \theta)$ is the distribution of data given θ (the likelihood), $\pi(\theta \mid x)$ is the posterior of θ and finally $\pi(X, \theta)$ is the simultaneous distribution of data and θ.

Hence the result of estimating θ in the Bayesian framework comes down to estimating the posterior distribution of θ, which is fundamentally different from the frequentistic approach where the parameter is considered (unknown but) fixed. In practice it is common to report functionals of the posterior distribution, such as the median or the mean.

If the prior is uninformative, that is, when the prior is almost flat, such that any value of θ a priori is equally likely to occur, it follows from (4.1) that

$$\pi(\theta \mid X) \propto \pi(x \mid \theta),$$

because the denominator does not depend on θ. Here proportionality is with respect to θ. This means that the posterior is proportional to the likelihood and the posterior mode (if it exists) will correspond to the maximum likelihood estimate.

4.2 Importance sampling

Assume we are interested in some quantity $\mathrm{E}_f(h(X))$, where f is a density function. For example, taking $h(x) = x^p$ we can get all (existing) moments of $h(X)$, or with $h(x) = 1_A(x)$ we get $P(h(X) \in A)$ for some measurable set A. We will assume that h is a square integrable function with respect to f, and let $\mathcal{X}$ denote the state space of X.

We shall address the problem of computing $\mathrm{E}_f(h(X))$ by simulation when moments of $h(X)$ under f are difficult to evaluate theoretically. In many applications f is difficult or even impossible to sample from, but sometimes there exists a distribution with density g such that $g \gg f$ from which sampling is feasible. The idea is to use samples from g to evaluate the quantity of interest. The g density may be defined on a larger state space $\bar{\mathcal{X}} \supseteq \mathcal{X}$ and in this case we define $f(x) = 0$ for $x \notin \mathcal{X}$.

Thus we consider a sample $X_1, \ldots, X_n$ distributed according to some distribution with density g. Importance sampling is a method to evaluate functional moments from the distribution f using samples from the distribution g. The idea is to express the object of interest as

$$\mathrm{E}_f(h(X)) = \int_{\mathcal{X}} \frac{f(x)h(x)g(x)}{g(x)} \, \mathrm{d}x.$$

By the strong law of large numbers

$$\frac{1}{n}\sum_{i=1}^{n} \frac{f(X_i)h(X_i)}{g(X_i)} \to \mathrm{E}_g\left(\frac{f(X_i)h(X_i)}{g(X_i)}\right) = \int_{\mathcal{X}} f(x)h(x) \, \mathrm{d}x \quad a.s.$$

Even though the empirical average of the fraction fh/g converges, it does not mean that all g's are equally good candidates to sample from. In order to assess whether a given g is a good choice we make the following observation related to the variance:

$$\mathrm{E}_g\left\{\left(\frac{f(X_i)h(X_i)}{g(X_i)}\right)^2\right\} = \int_{\mathcal{X}} \frac{f(x)^2h(x)^2}{g(x)} \, \mathrm{d}x,$$

which must be finite in order for fh/g to have finite variance. It follows, since h is square integrable with respect to f, that if f/g is bounded, the variance is finite. The following example from Robert and Casella (2004) illustrates the methodology.

Example 4.1 (Cauchy tail probabilities)**.** Consider as quantity of interest

$$p = \int_2^{\infty} \frac{1}{\pi(1+x^2)} \, \mathrm{d}x.$$

This immediately suggests the estimators

$$\hat{p}_1 = \frac{1}{n}\sum_{i=1}^{n} 1_{(X_i>2)}, \quad \hat{p}_2 = \frac{1}{2n}\sum_{i=1}^{n} 1_{(|X_i|>2)}.$$

One may rewrite p as

$$p = \frac{1}{2} - \int_0^2 \frac{1}{\pi(1+x^2)} \, \mathrm{d}x = \int_0^{\frac{1}{2}} \frac{x^{-2}}{\pi(1+x^{-2})} \, \mathrm{d}x,$$

giving rise to estimators

$$\hat{p}_3 = \frac{1}{2} - \frac{1}{n}\sum_{i=1}^{n} \frac{2}{\pi(1+U_i^2)}, \quad \hat{p}_4 = \frac{1}{2n}\sum_{i=1}^{n} \frac{1}{\pi(1+Y_i^{-2})},$$

where U_i and Y_i are uniformly distributed on $[0,2]$ and $[0,\frac{1}{2}]$, respectively.

One can compute the variance of these estimators and see that

$$\mathrm{Var}(\hat{p}_1) \approx \frac{0.13}{n}, \ \mathrm{Var}(\hat{p}_2) \approx \frac{0.05}{n}, \ \mathrm{Var}(\hat{p}_3) \approx \frac{0.029}{n}, \ \mathrm{Var}(\hat{p}_4) \approx \frac{0.000096}{n}.$$

All four estimators converge toward p as n tends to infinity, but they have very different variances. Interestingly the estimators $\hat{p}_1$ and $\hat{p}_2$, that samples from the Cauchy distribution, are the worst of the four options, while $\hat{p}_4$ is much better than all the other three.

♦

The point of example 4.1 is that it may sometimes be more efficient to sample from a distribution different from f in order to obtain the better estimator. Next, we give a theoretical result about the optimal choice of the proposal density g. Note that it is not a result to be used in applications, as it involves the integral of hf, which is the quantity of interest.

Theorem 4.2. *The choice of g that minimizes the variance of the estimator*

$$\frac{1}{m}\sum_{i=1}^{m}\frac{f(X_i)}{g(X_i)}h(X_i),$$

is given by

$$g^*(x) = \frac{|h(x)|f(x)}{\int |h(z)|f(z)\ dz}.$$

Proof. The proof is straightforward, focusing of the essential part of the variance as a squared mean. Using Jensen's inequality leads to the result. □

4.3 The Metropolis-Hastings Algorithm

The Metropolis-Hastings algorithm first described in Metropolis et al. (1953) and later generalized in Hastings (1970), is a powerful and simple simulation technique used to obtain samples from a given distribution. It works under weak regularity conditions and requires only partial knowledge of the distribution of interest, in the sense that any normalizing constant of the density can be ignored. Chib and Greenberg (1995) gives a detailed introduction. Assume we want to sample from the target distribution Π with density π. The idea of the Metropolis-Hastings algorithm is to sample from another distribution (the candidate distribution) where sampling is easier, and then accept or reject these samples according to information about the candidate and the target distributions.

The main component of the algorithm is a transition kernel, P, that satisfies

$$\Pi(A) = \int P(x, A)\pi(x)\ \mathrm{d}x, \tag{4.2}$$

for all measurable sets A. If we can find such a P then Π is the invariant distribution for P. We consider P on the form

$$P(x, A) = \int_A p(x, y)\ \mathrm{d}y + r(x)\delta_x(A),$$

for some integrable function p where $r(x) = 1 - \int p(x, y)\ \mathrm{d}y$ is the probability of staying in x, $p(x, x) = 0$, and δ_x is the Dirac measure.

The detailed balance equation states that

$$\pi(x)p(x, y) = \pi(y)p(y, x),$$

and if this is satisfied by $p(x, y)$, it follows, by an interchanging of integrals, that P and π satisfies (4.2). Consider now a candidate distribution, with density $q(x, y)$, specifying the transition density for moving from x to y. If, for $x \neq y$,

$$\pi(x)q(x, y) > \pi(y)q(y, x), \tag{4.3}$$

we move from x to y too often to satisfy the detailed balance equation. To correct for this, we introduce the probability $\alpha(x, y)$, such that the move from x to y is made according to

$$\tilde{p}(x, y) = q(x, y)\alpha(x, y).$$

For $\tilde{p}$ to satisfy the detailed balance equation, it must hold that

$$\pi(x)q(x, y)\alpha(x, y) = \pi(y)q(y, x)\alpha(y, x).$$

If (4.3) is true we want to maximize the probability of moving from y to x, so we put $\alpha(y, x) = 1$, which implies

$$\alpha(x, y) = \frac{\pi(y)q(y, x)}{\pi(x)q(x, y)}.$$

Similar considerations apply if (4.3) is reversed. Thus

$$\alpha(x, y) = \begin{cases} \min\left(\frac{\pi(y)q(y,x)}{\pi(x)q(x,y)}, 1\right), & \pi(x)q(x, y) > 0 \\ 1, & \pi(x)q(x, y) = 0 \end{cases}. \tag{4.4}$$

This motivates the definition of the transition kernel for the Metropolis-Hastings algorithm:

$$P(x, A) := \int_A q(x, y)\alpha(x, y)\ \mathrm{d}y + \tilde{r}(x)\delta_x(A),$$

with $\tilde{r}(x) := 1 - \int q(x, y)\alpha(x, y)\ \mathrm{d}y$, and it follows now that P has Π as invariant distribution. Under mild regularity conditions the algorithm generates (correlated) samples with a distribution that converges to the invariant distribution; See Meyn and Tweedie (1993). The algorithm is summarized in table 4.1.

The distribution q is known as the instrumental/proposal distribution and f is the target density. Note that it suffices to know the ratio f/q only up to proportionality. A general advice for implementation of the Metropolis-Hastings algorithm is to compute acceptance probabilities on the log scale as opposed to the original scale. In this way one may avoid running into numerical problems related to large values arisen from evaluation of the exponential function.

Initialize
1) Initialize $x^{(0)}$.
Iterate
At iteration $t+1$:
Sample $Z \sim q(x^{(t+1)}, x^{(t)})$ and $U \sim [0,1]$
if $U < \alpha(x^{(t)}, Z)$ put $x^{(t+1)} := Z$
if $U \geq \alpha(x^{(t)}, Z)$ put $x^{(t+1)} := x^{(t)}$

Table 4.1: The Metropolis-Hastings algorithm

Independence sampler

If the proposal is on the form $q(x, y) = q(y)$ such that q does not depend on the current state of the Markov chain we have an independence sampler. In this case it is not possible to change the acceptance probability of the sampler, thus it is important to choose q not 'too far' from the target distribution.

Random walk sampler

If the proposal $q(x, y) = q(|x-y|)$ is symmetric in x and y we have a random walk sampler, and the typical example is where $q(x, y)$ is the Gaussian density with mean x and some covariance Ω. For this proposal one can adjust the step size, in order to obtain an optimal acceptance rate, by adjusting Ω; see Roberts et al. (1997) where it is shown that under regularity conditions the optimal rate is 0.234 for one dimensional targets. The Ω may be tuned in order to control the step size of the proposal. Consider for instance a univariate Gaussian target initiated around its mean. Too large steps will increase the rejection rate and it may be difficult for the chain to visit the tails. On the other hand too small steps will increase the acceptance rate but it will take a long time to visit the tails. In general the random walk will accept all proposals to states with larger density and only go the states with smaller densities with a certain probability.

A drawback of the random walk occurs if the state space is bounded, in which case the random walk will reject all proposals beyond the boundary. In practice, if the sampler never gets close to the boundary, the problem can be neglected. Alternatively one may do a one-to-one transformation of the variable that is to be sampled, to circumvent the boundary problem.

4.3.1 Simulation of diffusion bridges

Direct simulation of a diffusion bridge is not easy. However if the distribution of the bridge has a Radon-Nikodym derivative with respect to a Brownian bridge, one may use the Metropolis-Hastings algorithm to sample diffusion bridges, using Brownian bridge

proposals. It is important that the target f and the proposal g are densities with respect to the same (dominating) measure λ. Assume $h > 0$ is also a density with respect to λ. Then if we express f with respect to the measure with density $h\,\mathrm{d}\lambda$, then

$$\frac{f(y)q(x \mid y)}{f(x)q(y \mid x)} = \frac{\frac{f(y)}{h(y)}q(x \mid y)}{\frac{f(x)}{h(x)}q(y \mid x)}.$$

This acceptance probability is clearly different from (4.4), unless q is also expressed with respect to the same dominating measure. This is almost obvious when the dominating measure is the Lebesgue measure. In the setting of continuous time processes there is no 'standard' dominating measure.

Example 4.3 (Sampling an Ornstein-Uhlenbeck bridge)**.** In order to be able to compare simulated bridges to true bridges we use the Ornstein-Uhlenbeck process as our target, because for this process we can find the marginal distributions. Therefore consider the 2-dimensional OU model from (2.17) with

$$A = \begin{pmatrix} 0 \\ 0 \end{pmatrix},\ B = \begin{pmatrix} 4 & 1 \\ 2 & 3 \end{pmatrix},\ \Sigma = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},$$

for $0 \leq t \leq 1$, starting and ending at $(0,0)^T$. The Metropolis-Hastings algorithm was run for 20.000 iterations with an acceptance rate of 18%, using proposals from the standard Brownian bridge. Figure 4.2 shows comparisons between the marginal density of the true OU bridge for the first coordinate, for each 0.1 time step and the output from the Metropolis-Hastings sampler. Comparison of the results from the second coordinate is similar (not shown). Thus the marginal distributions from the simulations are reasonably close to the true marginal OU distributions.

4.4 Gibbs sampling

The Gibbs sampler, proposed in Geman and Geman (1984), effectively breaks down a multivariate simulation problem to a series of one dimensional simulation problems. For a detailed introduction see for example Casella and George (1992). Let d denote the number of components that are to be sampled and assume that the marginal conditional distributions, generically denoted by π, exists. The Gibbs sampler iteratively simulate variables from the conditional distributions of each component conditional on the current state of all other $d-1$ components, thereby updating one coordinate at the time. The result is a correlated sample that approximates the simultaneous distribution of all d components. The algorithm is given in table 4.3.

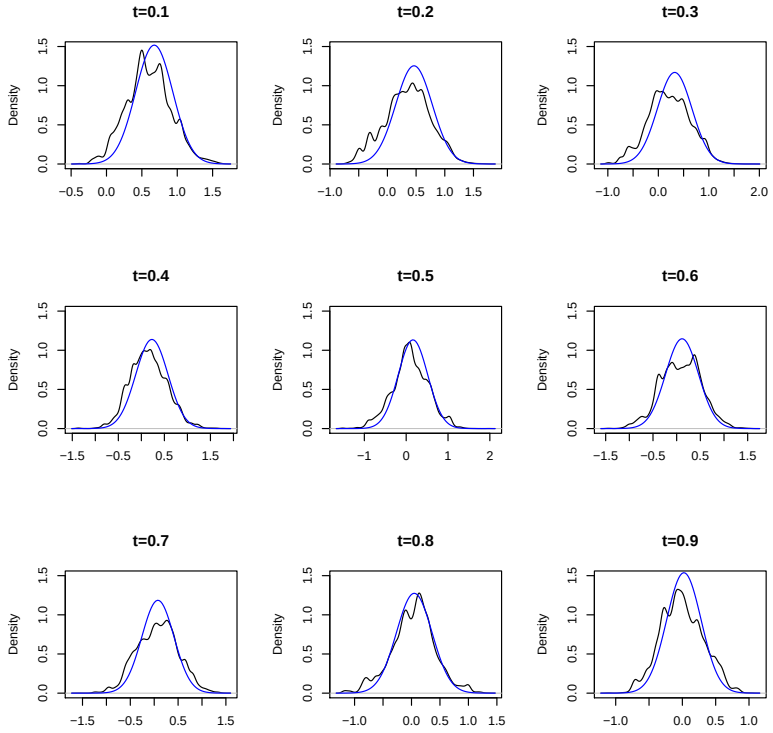

Figure 4.2: *Comparison between marginal distributions of the OU process and the output from the Metropolis-Hastings sampler.*

Initialize

1) Initialize $x_1^{(1)}, x_2^{(1)}, \ldots, x_d^{(1)}$.

Iterate

At iteration $t+1$:

Sample $X_1^{(t+1)} \sim \pi(\cdot \mid x_2^{(t)}, x_3^{(t)}, \ldots, x_d^{(t)})$

Sample $X_2^{(t+1)} \sim \pi(\cdot \mid x_1^{(t+1)}, x_3^{(t)}, \ldots, x_d^{(t)})$

$\vdots$

Sample $X_d^{(t+1)} \sim \pi(\cdot \mid x_1^{(t+1)}, x_2^{(t+1)}, \ldots, x_{d-1}^{(t+1)})$

Table 4.3: The Gibbs sampler

5

Parameter estimation for multidimensional diffusions, fully observed

This chapter is an updated version of the article Jensen et al. (2012). The algorithms described in that paper was implemented directly in `R` , resulting in relatively slow computation times. In this chapter all results and figures are produced using the newly developed `BIPOD`-package for `R` , which is described more detailed in chapter 8.2. Since the random number generator is different in the two approaches, the figures differ slightly from the ones in the article.

Excitability is observed in a variety of natural systems, such as neuronal dynamics, cardiovascular tissues, or climate dynamics. The stochastic FitzHugh-Nagumo model is a prominent example representing an excitable system. To validate the practical use of a model, the first step is to estimate model parameters from experimental data. This is not an easy task because of the inherent non-linearity necessary to produce the excitable dynamics, and because the two coordinates of the model are moving on different time scales. In this chapter we propose a Bayesian framework for parameter estimation, which can handle multi-dimensional non-linear diffusions with time scale separation. The estimation method is illustrated on simulated data.

An excitable system is characterized by a resting state from which it only escapes if perturbed by a sufficiently large stimulus. Weak stimuli only result in a small amplitude linear response, whereas strong stimuli cause a highly non-linear response, where the system variables make a large excursion through state space, whereafter it returns to its resting state after a refractory period. Under a continuous stimulus, the system can enter into an oscillatory mode. Thus, an excitable system operates close to a bifurcation point, and is sensitive to small perturbations, e.g. caused by noise. It is observed in many natural systems, such as neuronal dynamics, ion channels, chemical reactions, climate dynamics or wildfires (Lindner et al., 2004; Keener and Sneyd, 2009; Berglund and Gentz, 2006). Noise can have a dramatic effect on excitable systems, inducing stochastic limit cycles on otherwise stable dynamics. A prototype of an excitable system is the FitzHugh-Nagumo model, a minimal representation of more realistic excitable systems, like the Hodgkin-Huxley model, modeling the firing mechanisms in a neuron (FitzHugh, 1961; Nagumo et al., 1962; Hodgkin and Huxley, 1952). It is a generalization of the van der Pol equations. It allows for coordinates to evolve on different time scales, and the time scale separation parameter is essential for the understanding of the dynamical behavior of the system. The larger the time scale separation, the more an all-or-nothing response is observed to a perturbation, mimicking the response of the simpler threshold models, like leaky integrate-and-fire models (Gerstner and Kistler, 2002; Ditlevsen and Greenwood, 2013).

The FitzHugh-Nagumo model is defined by two coupled differential equations, representing the neuronal membrane potential and a recovery variable, respectively, where the recovery variable models the channel kinetics. Extending the model by adding a noise term governed by Brownian motion results in a diffusion process. The noise term in the FitzHugh-Nagumo model accounts for various sources of noise affecting the neuronal behavior, like random opening and closing of ion channels or noisy presynaptic currents Gerstner and Kistler (2002).

Diffusions are defined through stochastic differential equations, and for all but a few models an explicit expression for the transition density is unattainable. This problem complicates parameter estimation. Though many methods deal with this problem (see Sørensen (2004) and also Hindriks et al. (2011); Kleinhans (2012)), they tend to be highly complicated to implement and apply in practice, especially when the dimension of the diffusion is larger than one. The Euler-Maruyama, the Milstein and other schemes offer easy-to-implement approximations to the transition density. However, if the time-step between observations is too large, the approximation will be inaccurate.

Within the last decade, novel Bayesian methods have been developed which can be used for statistical inference, see Roberts and Stramer (2001); Papaspiliopoulos and Roberts (2012); Beskos et al. (2006); Papaspiliopoulos et al. (2012); Wu and Noé (2011). We describe a Markov Chain Monte Carlo method and adapt it to the two-dimensional stochastic FitzHugh-Nagumo model for parameter inference. Our approach involves imputation of data from the distribution of the underlying diffusion process, and application of a Gibbs sampler to iteratively update parameters and imputed data. We apply an independent Metropolis-Hastings-step to update the imputed data conditional on parameters, and sample the parameters conditional on the imputed data directly. Parameter sampling relies on a Gaussian prior for the parameters, and when this assumption is not met a Gaussian random walk Metropolis-Hastings-step may be applied, see section 5.2.2 for details.

Typically, in experimental settings only the slow variable of the membrane potential is observed through intracellular recordings in single neurons, whereas the channel kinetics are unobserved. This largely complicates the statistical inference, e.g. because the observed process is no longer Markov. One approach to this problem is to assume the channel kinetics known (Huys et al., 2006; Huys and Paninski, 2009). We will not assume the channel kinetics known, but instead assume that the recovery variable is observed.

Our methodology may be extended to the partially observed case and this is the topic of chapter 6. However, the first goal, which is achieved in this chapter, is to make the statistical procedure work in a computationally efficient manner in the two-dimensional non-linear model with time scale separation.

The effects of noise on the FitzHugh-Nagumo model have been extensively studied (see e.g. Lindner and Schimansky-Geier (1999); Lindner and Schimansky-Geier (2000); Lindner et al. (2004); Lee DeVille et al. (2005); Berglund and Gentz (2006)), whereas papers devoted to its comparison with experimental data are rare. Here we use simulated data to estimate parameters of the stochastic FitzHugh-Nagumo model from discrete observations of the state variables.

Section 5.2 describes the estimation procedure in the case where the diffusion coefficient is assumed to be known. This approach simplifies the exposition and speeds up the practical implementation considerably. Section 5.3 deals with the procedure when also the diffusion coefficient is estimated, and section 5.4 includes a small simulation study.

5.1 Statistical model

Consider the model

$$\mathrm{d}V_t = b(V_t;\theta)\,\mathrm{d}t + \Sigma(\sigma)\,\mathrm{d}B_t, \quad V_0 = v_0, \tag{5.1}$$

with V_t a d-dimensional stochastic process, B_t a d-dimensional standard Brownian motion, and functions b and Σ taking values in $\mathbb{R}^d$ and the set of $d \times d$ matrices, respectively. We will also assume that $\Gamma := \Sigma\Sigma^T$ is invertible. Note that the diffusion matrix in (5.1) does not depend on V_t. For the models we consider it is not a problem, and one can easily extend the methodology to include also state depend diffusion coefficients as long as Σ is invertible in both arguments. The only complication is that the Lamperti transform, that one needs to apply, gets more complicated.

We assume equidistant observations

$$D_n = \{V_{t_0}, V_{t_1}, \ldots, V_{t_n}\},$$

with $t_i - t_{i-1} = \Delta$, and $t_0 = 0, t_n = T$, and aim to make inference for the parameters θ and σ governing the diffusion (5.1). The observation times are assumed equidistant only for notational simplicity and because it is consistent with the type of experimental design for which the analysis in this chapter is relevant for; it will be clear that our methodology does not require this assumption.

Even though the theory is presented for multi-dimensional models with any dimension d, we will focus solely on two-dimensional models for applications. For models where the drift is not linear in θ, the estimation procedure we describe will still work, at the cost of an additional Metropolis-Hastings-step, and this approach is described in section 5.2.2.

5.2 Estimation of drift parameters with known diffusion

The aim of this section is to estimate the parameter vector θ governing the drift, while σ is assumed known. This assumption simplifies the exposition, while at the same time, it provides the foundation of the methodology used in the setting where also the diffusion is unknown. For notational simplicity σ is neglected in this section.

Since the model is Markov, the distribution of θ conditional on the observed data D_n is

$$\begin{aligned} p(\theta \mid D_n) &\propto p(\theta)p(D_n \mid \theta) \\ &= p(\theta)\prod_{i=1}^{n} p(V_{t_i} \mid V_{t_{i-1}}, \theta), \end{aligned} \tag{5.2}$$

where $p(\theta)$ is the prior distribution of θ, $p(D_n \mid \theta)$ is the distribution of the observed data given θ and $p(V_{t_i} \mid V_{t_{i-1}}, \theta)$ is the transition density of the process V, from $V_{t_{i-1}}$ to

V_{t_i}, conditional on θ. Following standard convention in Bayesian statistics, proportionality is understood with respect to the argument of the density on the left hand side of the equation, i.e. θ in the above equation. For all but a few models, the transition density is not explicitly known and this is indeed a problem in the FitzHugh-Nagumo model. To overcome this difficulty, consider a theoretical data augmentation step that imputes a latent data path $\bar{V}_i$ on each interval (t_i, t_{i+1}), distributed according to the underlying model (5.1). Denote the collection of latent paths $\bar{V} := \cup_{i=0}^{n-1} \bar{V}_i$, and change focus to the posterior of θ and the imputed data conditional on the observed data:

$$p(\theta, \bar{V} \mid D_n). \tag{5.3}$$

A Gibbs sampler can be applied to obtain a sample from (5.3) from which marginal inference about the posterior of θ can be drawn. The algorithm alternates between updating the parameter and the latent data while keeping the other fixed. After a suitable burn-in period, the result is a sample $(\theta^{(k)}, \bar{V}^{(k)})_k$ from the distribution $p(\theta, \bar{V} \mid D_n)$, from which summarized inference about the posterior of θ can be drawn, e.g. mean, variance and tail probabilities. The algorithm is given in table 5.1.

Initialize
1) Initialize $\theta^{(0)}$ and imputed data $\bar{V}^{(0)}$.
Iterate
At iteration k:
2a) Sample $\bar{V}^{(k)}$ from $p(V \mid \theta^{(k-1)}, D_n)$.
2b) Sample $\theta^{(k)}$ from $p(\theta \mid \bar{V}^{(k)}, D_n)$.

Table 5.1: Gibbs sampler for $p(\theta, \bar{V} \mid D_n)$.

A few remarks are in order: On the theoretical level the imputed paths $\bar{V}$ are infinite dimensional. Therefore, each path is projected onto a discrete subset of the continuous path, and the accuracy of the algorithm depends to some extend on the accuracy of this approximation. The paths sampled from step 2a) in table 5.1 are conditioned on the observed data D_n, and therefore a method for simulation of processes conditional on both start and endpoint (diffusion bridges) is needed. Simulating a diffusion bridge is in general not an easy task though much progress in this area has been achieved in the last decade; see Beskos et al. (2006); Papaspiliopoulos and Roberts (2012); Bladt and Sørensen (2014); Lindström (2012).

5.2.1 Sampling the latent path

Due to the Markov property, the following relation holds,

$$p(\bar{V} \mid \theta, D_n) = \prod_{i=1}^{n} p(\bar{V}_i \mid V_{t_{i-1}}, V_{t_i}, \theta), \tag{5.4}$$

implying that paths $\bar{V}_i$ may be sampled independently. Direct simulation of these bridges (or the projection thereof) is not feasible. Instead, we follow Roberts and Stramer (2001) and use a Metropolis-Hastings-step for this simulation, whereby we propose paths from an alternative simpler model and accept them with the appropriate probability. Without loss of generality, focus on a single term of the form

$$p(\bar{V} \mid V_0, V_T, \theta) \tag{5.5}$$

from (5.4). This is a diffusion bridge, and in Roberts and Stramer (2001) it is shown that one can simulate these bridges with a Metropolis-Hastings-step, using proposals from a Brownian bridge. Samples from the latter are easily generated, since the transition density of a Brownian bridge is for $t > s$,

$$\begin{aligned} &p(V_t \mid V_s = v_s, V_0 = v_0, V_T = v_T) \\ &\sim \varphi\left(V_t \mid v_s + \frac{t-s}{T-s}(v_T - v_s); \frac{(T-t)(t-s)}{T-s}\Gamma\right), \end{aligned} \tag{5.6}$$

where $\varphi(\cdot \mid \mu; \Omega)$ denotes the Gaussian density with mean μ and covariance matrix Ω. To each proposal a weight is assigned, the logarithm of which is given by

$$\log\left(\psi(\bar{V}, \theta)\right) = \int_0^T b(\bar{V}_u; \theta)^T \Gamma^{-1} \, \mathrm{d}\bar{V}_u - \frac{1}{2}\int_0^T b(\bar{V}_u; \theta)^T \Gamma^{-1} b(\bar{V}_u; \theta) \, \mathrm{d}u. \tag{5.7}$$

This is precisely (proportional to) the Radon-Nikodym derivative between the target diffusion bridge measure and the proposal Brownian bridge measure. The algorithm for simulating the bridge $\bar{V}$ is given in table 5.2. Note that step 2) in table 5.2 involves an approximation of the integral in (5.7). More details on diffusion bridge simulation can be found in Papaspiliopoulos and Roberts (2012).

Initialize

1) Initialize a skeleton path $(\bar{V}^M)_0$ according to (5.6), and compute the weight $\psi_0 = \psi((\bar{V}^M)_0, \theta)$, using (5.7).

Iterate

2) Update the current value of ψ_k according to $(\bar{V}^M)_k$.

3) Generate a proposal skeleton path, $\tilde{V}^M$ according to (5.6), and compute the weight $\tilde{\psi} = \psi(\tilde{V}^M, \theta)$, using (5.7).

4) Let $(\bar{V}^M)_{k+1} = \begin{cases} \tilde{V}^M & \text{with prob. } \min\left(1, \frac{\tilde{\psi}}{\psi_k}\right) \\ (\bar{V}^M)_k & \text{otherwise} \end{cases}$.

Table 5.2: Metropolis-Hastings-step for sampling from the diffusion bridge (5.5).

5.2.2 Sampling the drift parameter

It follows directly from Theorem 2.6 (Cameron-Martin-Girsanov), that

$$\log(p(V \mid \theta)) = \int_0^T b(V_s;\theta)^T \Gamma^{-1}(\sigma) \left(\mathrm{d}V_s - \frac{1}{2} b(V_s;\theta)\, \mathrm{d}s \right). \tag{5.8}$$

In general, this likelihood is not easy to evaluate directly. One can use the Euler-Maruyama scheme to obtain a Gaussian approximation of the density. This approximation can be made arbitrarily accurate, as the Euler-Maruyama approximation can approximate the true process arbitrarily well when the discretization goes to 0. In theory, it is not a problem to allow for arbitrarily small Δ, but in applications one must consider computer and time limitations. In the following we treat the sampling of drift parameters differently according to whether the drift is linear in the parameters.

Linear drift

When the model is linear in the drift parameters it can be written as

$$b(V_t;\theta) = f_0(V_t) + \sum_{i=1}^{p_1} \theta_i f_i(V_t), \tag{5.9}$$

with f_i a $d \times 1$ vector for $i = 0, \ldots, p_1$. In this case, taking a Gaussian prior, the prior and the posterior distribution will be conjugate, i.e., the prior and the posterior belong to the same family of distributions. To see this, first note that

$$p(\theta \mid \bar{V}, D_n) \propto p(\theta) p(V \mid \theta),$$

where V denotes the union of observed and imputed data.

Then

$$\int_0^T b(V_s;\theta)^T \Gamma^{-1}\, \mathrm{d}V_s = \sum_{i=1}^{p_1} \theta_i I_i + C_1,$$

$$\int_0^T b(V_s;\theta)^T \Gamma^{-1} b(V_s;\theta)\, \mathrm{d}s = \sum_{i,j=1}^{p_1} \theta_i \theta_j R_{ij} + \; 2 \sum_{i=1}^{p_1} \theta_i \int_0^T f_i(V_s)^T \Gamma^{-1} f_0(V_s)\, \mathrm{d}s + C_2,$$

where for $i, j = 1, \ldots, p_1$

$$I_i := \int_0^T f_i(V_s)^T \Gamma^{-1}\, \mathrm{d}V_s, \tag{5.10}$$

$$R_{ij} := \int_0^T f_i(V_s)^T \Gamma^{-1} f_j(V_s)\, \mathrm{d}s \tag{5.11}$$

$$C_1 := \int_0^T f_0(V_s)^T \Gamma^{-1}\, \mathrm{d}V_s$$

$$C_2 := \int_0^T f_0(V_s)^T \Gamma^{-1} f_0(V_s)\, \mathrm{d}s,$$

and these expressions do not depend on θ. It follows that

$$\begin{aligned}\log(p(V \mid \theta)) &= \int_0^T b(V_s;\theta)^T \Gamma^{-1} \, \mathrm{d}V_s - \frac{1}{2}\int_0^T b(V_s;\theta)^T\Gamma^{-1}b(V_s;\theta)\,\mathrm{d}s \\ &= -\frac{1}{2}\left(\sum_{i,j=1}^{p_1}\theta_i\theta_j R_{ij} - 2\sum_{i=1}^{p_1}\theta_i F_i - 2C_1 + C_2\right) \\ &= -\frac{1}{2}\left(\theta^T R\theta - 2\theta^T F - 2C_1 + C_2\right).\end{aligned}$$

where

$$F_i := \left(I_i - \int_0^T f_i(V_s)^T\Gamma^{-1}f_0(V_s)\,\mathrm{d}s\right),$$

and $F := (F_i)_{i=1,\ldots,p_1}$ and $R := (R_{ij})_{i,j=1,\ldots,p_1}$. So the log-likelihood from (5.8) is exponentially quadratic in θ, and this means that with a Gaussian prior for $\theta \sim \mathcal{N}(\mu, \Psi)$, the posterior is conjugate

$$\pi(\theta \mid D_n, \tilde{V}_{(0,T]}, \sigma) \sim \mathcal{N}(\tilde{\mu}, \tilde{\Psi}), \tag{5.12}$$

where

$$\tilde{\Psi} = (R + \Psi^{-1})^{-1}, \tag{5.13}$$

$$\tilde{\mu} = \tilde{\Psi}(F + \Psi^{-1}\mu). \tag{5.14}$$

If there is no prior information about the parameters, it is natural to choose a prior distribution with large variance. If the variance of the prior is taken to be infinite, the posterior distribution is completely determined by the terms from the Radon-Nikodym derivative:

$$\begin{aligned}\tilde{\Psi} &= R^{-1}, \\ \tilde{\mu} &= R^{-1}F.\end{aligned}$$

In this case, the k'th iteration of the Gibbs sampler simulates drift parameters as

$$\theta^{(k)} \sim \mathcal{N}_{p_1}(R^{-1}F; R^{-1}). \tag{5.15}$$

Having identified the moments in (5.13) and (5.14) is highly appealing, since it allows for direct sampling from $p(\theta \mid \bar{V}, D_n)$. In practice, the integrals in (5.10) and (5.11) are approximated by Riemann sums, and the accuracy will depend on the sparsity of the imputed data.

In models where the assumptions of a Gaussian prior or linearity in the drift parameters are not met, the distribution $p(\theta \mid \bar{V}, D_n)$ could be approximated by a Metropolis-Hastings-step. This approach is described next.

Non-linear drift

If a re-parametrization to obtain linearity in the drift parameters is not feasible, one can still obtain approximate samples from the distribution $p(\theta \mid \bar{V}, D_n)$. One approach is to use a Metropolis-Hastings-step, though it requires additional computational time. The interval $[t_i, t_{i+1}]$ is split into M sub-intervals defined by the time points $t_i^m := t_i + m\Delta/M, m = 0, \ldots, M$, such that $t_i^0 = t_i$ and $t_i^M = t_{i+1}$. Assume the imputation of $M-1$ data points between each pair of successive observations and denote the collection of imputed data in the interval (t_i, t_{i+1}) by $\bar{V}_i^M$. Thus we have $n+1$ observations, with $M-1$ imputed values in each of the n intervals. By the Markov property it follows that

$$\begin{aligned} p(\theta \mid \bar{V}, D_n) &\approx p(\theta \mid \cup_{i=0}^{n-1} \bar{V}_i^M, D_n) \\ &\propto p(\theta) p(\cup_{i=0}^{n-1} \bar{V}_i^M \mid D_n, \theta) \\ &= p(\theta) \prod_{i=0}^{n-1} \prod_{j=1}^{M} p(V_{t_i^j} \mid V_{t_i^{j-1}}, \theta). \end{aligned} \tag{5.16}$$

Compared to (5.2), each transition now occurs on the time scale of Δ/M instead of Δ and therefore, when M is large enough, an Euler-Maruyama approximation is reasonable:

$$V_{t_i^j} \approx V_{t_i^{j-1}} + b(V_{t_i^{j-1}}; \theta)\Delta + \Sigma(\sigma)\Delta B_{t_i^j},$$

where $\Delta B_{t_i^j} = B_{t_i^j} - B_{t_i^{j-1}} \sim \varphi(\cdot \mid 0; I_d\Delta)$. Therefore

$$p(V_{t_i^j} \mid V_{t_i^{j-1}}, \theta) \approx \varphi(V_{t_i^j} \mid \mu_i; \Gamma(\sigma)\Delta),$$

where $\mu_i = V_{t_i^{j-1}} + b(V_{t_i^{j-1}}; \theta)\Delta$.

The density $p(\theta \mid \bar{V}, D_n)$ is therefore approximately known up to a proportionality constant, and for simulations it is thus natural to use a Metropolis-Hastings-step. Motivated by (5.16) define the proportional target distribution f by

$$f(\theta) = p(\theta) \prod_{i=0}^{n-1} \prod_{j=1}^{M} \varphi(V_{t_i^j} \mid V_{t_i^{j-1}}; \theta).$$

We suggest a Gaussian random walk for the proposal distribution $\varphi(\cdot \mid \theta^{(k-1)}; \Omega)$, to propose a new θ. Note that for this approach, parameters which are restricted to a true subset of $\mathbb{R}$ requires a re-parameterization. The Metropolis-Hastings-step is given in table 5.3.

5.3 Estimation of both drift and diffusion parameters

When estimating diffusion parameters an important point related to the dependence between parameters and imputed data must be made. The quadratic variation identity im-

Initialize

1) Initialize $\theta^{(0)}$.

Iterate

2a) Generate a proposal $\tilde{\theta}$ from $\varphi(\tilde{\theta} \mid \theta^{(k)}; \Omega)$.

2b) Let $\theta^{(k+1)} = \begin{cases} \tilde{\theta} & \text{with prob. } \min\left(1, \frac{f(\tilde{\theta})}{f(\theta^{(k)})}\right) \\ \theta^{(k)} & \text{otherwise} \end{cases}$.

Table 5.3: Metropolis-Hastings-step for sampling from $p(\theta \mid \bar{V}, D_n)$.

plies that for any $t > 0$

$$\lim_{M\to\infty} \sum_{i=1}^{M} \left(V_{ti/M} - V_{t(i-1)/M}\right)\left(V_{ti/M} - V_{t(i-1)/M}\right)^T$$
$$= \int_0^t \Gamma(\sigma)\, \mathrm{d}s \quad \text{in probability.}$$

Thus, an observed path of V completely identifies σ, and if $\Sigma(\sigma) = \Sigma$ is constant the limit is just $t\Gamma(\sigma)$. When dealing with discrete-time observations D_n, there is only finite information about σ, hence there will be statistical error associated with its estimation. However, the identity implies that we cannot hope to apply a Gibbs sampler which iteratively would update paths and σ. Any value of σ would generate a path whose quadratic variation would return exactly the same value for σ, hence it will be impossible to explore the posterior distribution of σ in this way. Of course, in practice we only generate finite-dimensional projections of the paths, hence we would not observe this reducible behavior. Nevertheless, it is obvious, and actually proved in Roberts and Stramer (2001), that the Gibbs sampler which updates σ and a projection of the path based on M intermediate values for each pair of observations, has mixing time which is $\mathcal{O}(M)$. Therefore it becomes worse as we reduce the approximation bias.

However, this problem is easy to overcome by a simple transformation as in Roberts and Stramer (2001). The original article describes it for one-dimensional diffusions, but the extension is immediate for the multi-dimensional setting we consider here: We apply the one-to-one Lamperti transformation (see section 2.1.2)

$$x \mapsto \Sigma^{-1}(\sigma)x$$

to the process V, and obtain a new diffusion Z which has the form

$$\mathrm{d}Z_t = \alpha(Z_t, \theta, \sigma)\, \mathrm{d}t + \mathrm{d}B_t, \quad Z_0 = \Sigma^{-1}(\sigma)V_0, \tag{5.17}$$

where $\alpha(Z_t, \theta, \sigma) = \Sigma^{-1}(\sigma)b(\Sigma(\sigma)Z_t, \theta)$.

Sampling $\bar{V}$ is equivalent to sampling Z conditionally on $Z_{t_{i-1}} = \Sigma^{-1}(\sigma)V_{t_{i-1}}$ and $Z_{t_i} = \Sigma^{-1}(\sigma)V_{t_i}$, and the quadratic variation of Z is now independent of σ. However, there

is again a perfect dependence between Z and σ via the endpoints of Z: for given σ, Z has σ-dependent endpoints, which then perfectly determine σ in the following iteration. Therefore we need to remove the dependence on the endpoints as well.

Define $\tilde{V}$ as

$$\tilde{V}_t = Z_t - \left(1 - \frac{t - t_{i-1}}{\Delta}\right) Z_{t_{i-1}} - \frac{t - t_{i-1}}{\Delta} Z_{t_i} \tag{5.18}$$

for $t_{i-1} \leq t \leq t_i$. Note that $\tilde{V}_{t_{i-1}} = \tilde{V}_{t_i} = 0$, and $\bar{V}$ can be reconstructed from $\tilde{V}$ and σ by inverting the two transformations: adding first the endpoints to obtain Z and scaling by $\Sigma(\sigma)$ to obtain $\bar{V}$. To understand the intuition behind this transformation, consider the measure of the process in (5.17), without the drift α, but conditional on $Z_{t_{i-1}}$ and Z_{t_i}. Under this measure, Z is a Brownian bridge starting and ending at $Z_{t_{i-1}}$ and Z_{t_i}, respectively. Tilting it linearly as in (5.18) makes $\tilde{V}$ a standard Brownian bridge. This construction effectively allow us to sample Z from (5.17), using proposals from a Brownian bridge.

Next we describe the individual steps for the Gibbs sampler. We apply it to $(\theta, \sigma, \tilde{V})$ and not $(\theta, \sigma, \bar{V})$. This approach will ensure that $\tilde{V}$ and (θ, σ) are independent under the proposal for updating $\tilde{V}$, and circumvent the problem with reducible behavior of the Gibbs sampler. The parameters θ and σ are updated separately in order to take advantage of the simple Gaussian conditional posterior for θ.

5.3.1 Sampling the latent path

As in section 5.2.1 we write $p(\tilde{V} \mid \theta, D_n)$ as a product of densities and for simplicity we focus on a single term of the form $p(\tilde{V} \mid V_0, V_T, \theta, \sigma)$. Continuing along the lines of section 5.2.1 we note that $p(\tilde{V} \mid V_0, V_T, \theta, \sigma)$ can be simulated using a Metropolis-Hastings step with proposals from a Brownian bridge. Thus for each proposal $\tilde{V}$ we assign a weight, ϕ, given by

$$\log\left(\phi(\tilde{V}, \theta, \sigma)\right) = \int_0^T \alpha(Z_s, \theta, \sigma)^T \, \mathrm{d}Z_s - \frac{1}{2} \int_0^T \alpha(Z_s, \theta, \sigma)^T \alpha(Z_s, \theta, \sigma) \, \mathrm{d}s, \tag{5.19}$$

where $\tilde{V}$ and Z are linked by the relation (5.18).

The algorithm for updating the bridge $\tilde{V}$ is given in table 5.4.

5.3.2 Sampling the drift parameter

Sampling from $p(\theta \mid \tilde{V}, D_n, \sigma)$ is carried out in complete analogy to subsection 5.2.2 and (5.12), using the process $\Sigma(\sigma) Z_t$ instead of $\bar{V}_t$.

Initialize

1) Initialize a skeleton path $(\tilde{V}^M)_0$, sampled as a standard Brownian bridge starting and ending at 0. Compute Z according to (5.18) and approximate the weight ϕ in (5.19), w_0.

Iterate

2) Generate a proposal skeleton path, $\tilde{V}_P^M$ sampled as a standard Brownian bridge starting and ending at 0. Compute $\tilde{Z}$ according to (5.18) and approximate the weight ϕ in (5.19), $\tilde{w}$.

3) Let $(\tilde{V}^M)_{k+1} = \begin{cases} \tilde{V}_P^M & \text{with prob. } \min\left(1, \frac{\tilde{w}}{w_k}\right) \\ (\tilde{V}^M)_k & \text{otherwise} \end{cases}$.

Table 5.4: Metropolis-Hastings-step for sampling from the diffusion bridge (5.17).

5.3.3 Sampling the diffusion parameter

The priors of θ and σ are assumed to be independent, and therefore

$$p(\sigma \mid \tilde{V}, D_n, \theta) \propto p(\tilde{V} \mid \theta, \sigma, D_n) p(D_n \mid \theta, \sigma) p(\sigma). \tag{5.20}$$

Now redefine (5.19) for the specific time points t_{i-1} and t_i:

$$\log\left(\phi_i(\tilde{V}, \theta, \sigma)\right) = \int_{t_{i-1}}^{t_i} \alpha(Z_s, \theta, \sigma)^T \, \mathrm{d}Z_s - \frac{1}{2} \int_{t_{i-1}}^{t_i} \alpha(Z_s, \theta, \sigma)^T \alpha(Z_s, \theta, \sigma) \, \mathrm{d}s.$$

Using (2.14) yields

$$\begin{aligned} &p(\tilde{V} \mid \theta, \sigma, D_n) \\ &= \prod_{i=1}^{n} \frac{\varphi(\Sigma^{-1}(\sigma) V_{t_i} \mid \Sigma^{-1}(\sigma) V_{t_{i-1}}; \Delta I_d)}{p_{t_{i-1}, t_i}(\Sigma^{-1}(\sigma) V_{t_{i-1}}, \Sigma^{-1}(\sigma) V_{t_i})} \phi_i(\tilde{V}, \theta, \sigma), \end{aligned}$$

where p is the transition density of (5.1), and I_d is the d dimensional identity matrix. Furthermore, with a change of variables,

$$\begin{aligned} &p(D_n \mid \theta, \sigma) \\ &= |\det(\Sigma^{-1}(\sigma))|^n \prod_{i=1}^{n} p_{t_{i-1}, t_i}(\Sigma^{-1}(\sigma) V_{t_{i-1}}, \Sigma^{-1}(\sigma) V_{t_i}), \end{aligned}$$

so we obtain

$$\begin{aligned} &p(\sigma \mid \tilde{V}, D_n, \theta) \\ &\propto p(\sigma) |\det(\Sigma^{-1}(\sigma))|^n \cdot \\ &\prod_{i=1}^{n} \varphi(\Sigma^{-1}(\sigma) V_{t_i} \mid \Sigma^{-1}(\sigma) V_{t_{i-1}}; \Delta I_d) \phi_i(\tilde{V}, \theta, \sigma), \end{aligned} \tag{5.21}$$

which can be evaluated using a Riemann approximation of ϕ_i. Applying a Metropolis-Hastings-step to sample from the distribution proportional to (5.21) is straightforward. We use a Gaussian random walk, on the transformed diffusion parameter $\bar{\sigma} = \log(\sigma)$ in order to account for the restriction to positive values in the original parametrization. Therefore, define the proportional target function $s(\log(\sigma))$ as the right hand side of (5.21), and let $\varphi(\cdot \mid \bar{\sigma}^{(k)}; \Omega_2)$ be the Gaussian proposal distribution, used to propose an update of $\bar{\sigma}$ from $\bar{\sigma}^{(k)}$ to $\bar{\sigma}^{(k+1)}$. The Metropolis-Hastings-step is summarized in table 5.5.

Initialize

1) Initialize $\bar{\sigma}^{(0)}$.

Iterate

2a) Generate a proposal $\tilde{\sigma}$ from $\varphi(\cdot \mid \bar{\sigma}^{(k)}; \Omega_2)$.

2b) Let $\bar{\sigma}^{(k+1)} = \begin{cases} \tilde{\sigma} & \text{with prob. } \min\left(1, \frac{s(\tilde{\sigma})}{s(\bar{\sigma}^{(k)})}\right) \\ \bar{\sigma}^{(k)} & \text{otherwise} \end{cases}$.

Table 5.5: Metropolis-Hastings-step for sampling from $p(\sigma \mid \tilde{V}, D_n, \theta)$.

5.4 Simulation study for the FitzHugh-Nagumo model

For the FitzHugh-Nagumo model (2.23)-(2.24), the re-parametrization

$$(\tilde{\varepsilon}, \tilde{s}, \gamma, \beta) = (1/\varepsilon, s/\varepsilon, \gamma, \beta) \tag{5.22}$$

makes the model linear in the drift parameters. Note that ε is assumed positive, therefore in principle the Gaussian prior should be truncated at 0. However, for small ε the effect of the truncation is negligible and can be omitted.

Six data sets were generated, with parameter values resembling the situation in figure 2.3 with excitatory ($\beta = 1.4$) and oscillatory ($\beta = 0.6$) behavior, respectively. The choice of parameter values for the excitatory data is inspired by the values used in Lindner and Schimansky-Geier (1999). Parameters are given in table 5.6. Data was generated by

	ε	s	γ	β	σ_1	σ_2
Oscillatory	0.1	0.5	1.5	0.6	0.5	0.3
Excitatory	0.1	0.5	1.5	1.4	0.5	0.3

Table 5.6: Parameter values for simulation study.

thinning simulations from an Euler-Maruyama-scheme of the FitzHugh-Nagumo model (2.23)-(2.24), with 20,000 observations and a time step of 0.001.

Two data sets were generated using subsamples from the FitzHugh-Nagumo data for every 100th observation such that the sample size was $n = 200$ and time step between consecutive observations was $\Delta = 0.1$, implying a sample interval length of 20 time units. To investigate data with higher frequency, an additional data set was created for the excitatory setting, using subsamples for every 10th observation, leading to a step size of $\Delta = 0.01$.

Finally, three data sets were generated, to evaluate the estimation procedure for different values of ε. We used ε equal to $0.5, 0.05, 0.01$ and sample size $n = 200$ and $\Delta = 0.1$. All other parameters were as in the excitatory data.

For all six data sets, four data points were imputed between consecutive observations ($M = 5$) and we used 100,000 iterations of the Gibbs sampler. In all settings, the prior for $\theta = (\tilde{\varepsilon}, \tilde{s}, \gamma, \beta)$ was taken to be independent Gaussian. In the estimation procedure, the prior of θ enters only in the posterior distribution of $(\theta \mid \bar{V}, D_n)$, and with the variance of the prior taken to be infinite, the prior contributes no information to the posterior. The prior for $\log(\sigma)$ was taken to be independent Gaussian with mean $(\log(0.3)+2, \log(0.5)+1)$ and variance $(5, 5)$.

For the Metropolis-Hastings-step, the covariance matrix of the random walk proposal, was set to $\text{diag}(0.03, 0.0075)$ for the data sets with sample size $n = 200$. For each iteration a proposal parameter is either accepted or rejected, and acceptance rates were 20% and 21% for oscillatory and excitatory data, respectively.

5.4.1 Estimation of the drift parameters

Figure 5.7 shows density plots of the marginal posterior of each of the four drift parameters, for the setting from table 5.6, with $n = 200$. The black curves represents the excitatory data, and the gray curves represents oscillatory data. The vertical lines denote the parameter values used to generate data. For the oscillatory data the posterior distribution is more narrow and the modes are closer to the true parameter values, indicating that the estimation procedure performs better for the oscillatory data. In the lower right panel the dashed gray line separates the domain of β that leads to either excitatory or oscillatory behavior.

Figure 5.8 shows trace plots of the posterior for θ for the oscillatory data ($n = 200$). The plot was thinned before plotting and contains only every 10th iteration of the Gibbs sampler. For all four parameters the chain quickly reaches a stable regime, and trace plots for the excitatory data show similar characteristics.

Figure 5.9 shows autocorrelation plots of the posterior for θ for the oscillatory data ($n = 200$). It is desirable that the autocorrelations die out quickly to obtain a variance close to that provided by independent sampling from the target. For γ and β the autocorrelation goes to zero very fast, but less so for $\tilde{\varepsilon}$ and $\tilde{s}$. The mixing of all parameters is much improved in higher frequency data (not shown). For the excitatory data, the conclusions remain the same. Increasing the frequency of data ($n = 2000$), while keeping the sample length constant, has little effect in the precision of the estimates of θ, as we expect from

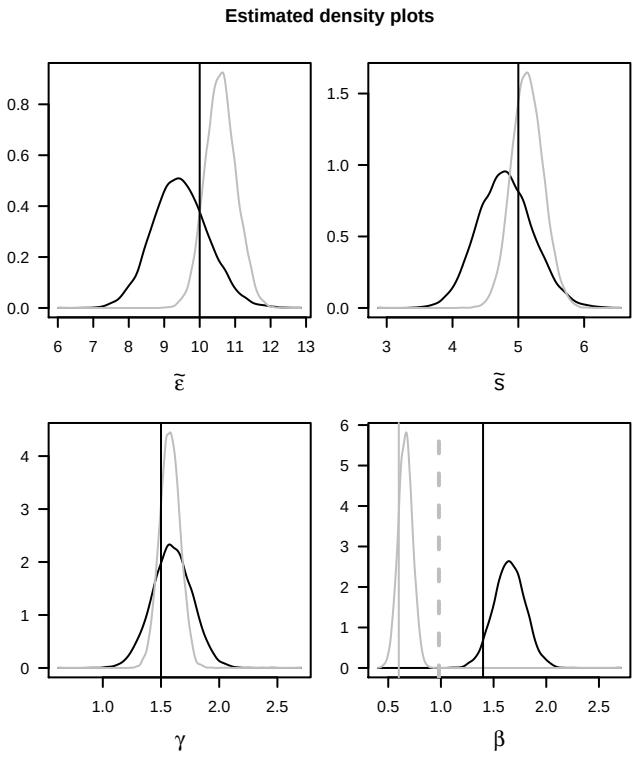

Figure 5.7: *Density plots of the sample posterior of the drift parameters* ($n = 200$). *Black and gray curves represents excitatory and oscillatory data and vertical lines denotes parameter values used to generate simulated data. Dashed line represents parameter value where the regime changes between oscillatory and excitatory behavior. The burn in period was 1000 iterations.*

the theory anyway. Improved statistical precision for θ is achieved by increasing the time period of observation.

5.4.2 Estimation of the diffusion parameters

Figure 5.10 shows density plots of the posterior for σ for both oscillatory and excitatory data ($n = 200$). Also included is the situation where sampling frequency is increased to $n = 2000$ in the excitatory regime. Black and gray solid lines represents excitatory and oscillatory data, respectively, with $n = 200$. Black dashed line denotes excitatory data with $n = 2000$. For all data sets, the mode of the posterior distribution is the same, but the tails are larger for smaller sample size.

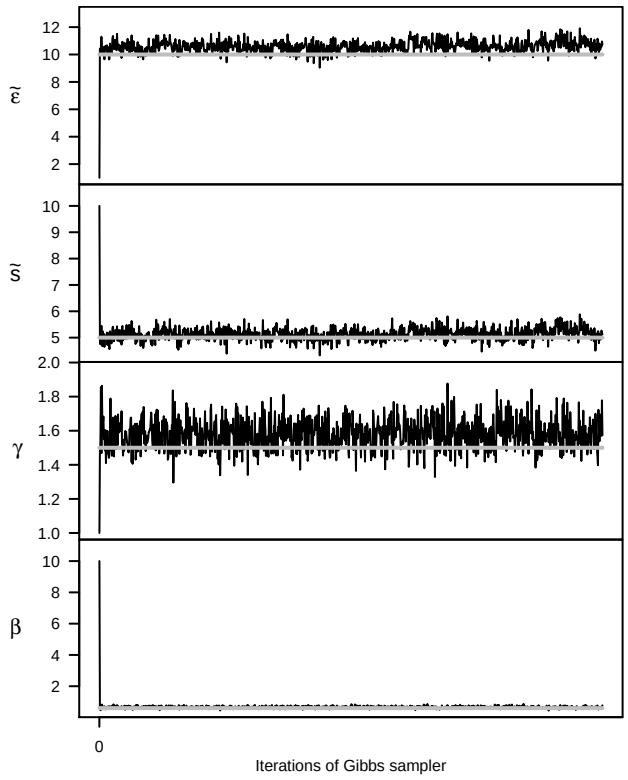

Figure 5.8: *Trace plot for the four drift parameters. Horizontal lines denote parameter values used to generate simulated data. Data was thinned before plotting.*

5.4.3 Changing the time scale parameter ε

The performance of the algorithm depends strongly on the size of the time scale separation. If the separation is large, it may become difficult to extract information from both coordinates in the system. Figure 5.11 shows four density plots based on data sets with $\tilde{\varepsilon} = 2, 10, 50$ and 100, and all other parameters as in the excitatory setting from table 5.6 and $n = 200$. Clearly the estimates get worse for large values of $\tilde{\varepsilon}$.

5.4.4 Practical comments

A Gaussian random walk was used as proposal for updating the diffusion parameters. In order to tune the covariance matrix for the proposal, the Gibbs sampler ran for 10,000 iterations, with a unit proposal variance, leading to a very low acceptance rate for the parameters. Taking the diagonal of the empirical correlation matrix, $\hat{\Psi}$, after a suitable burn in period, we obtained a rough relation between the diagonal elements of the covariance matrix. Thus, the covariance matrix was taken to be on the form $\lambda \cdot \text{ diag}(\hat{\Psi})$ for some $\lambda > 0$. Finally λ was tuned until the acceptance rate was relatively close to 0.23 as

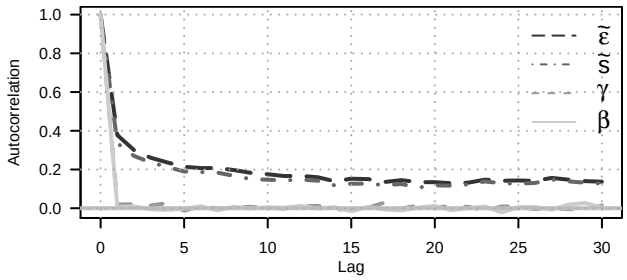

Figure 5.9: *Marginal autocorrelation plot for output of the Gibbs sampler. The burn in period was 1000 iterations.*

suggested in Roberts and Rosenthal (2001).

5.5 Discussion

We have introduced a Bayesian approach to parameter estimation in multivariate diffusion models and the method has been applied to the FitzHugh-Nagumo model for estimation of both drift and diffusion parameters. To the best of our knowledge, not many papers have previously focused on parameter estimation in the FitzHugh-Nagumo model or other excitatory models.

A few comment regarding the performance of the algorithm must be made. First, it is sensitive to the size of the time scale separation, but it is expected that performance will improve further if the latent paths are updated using proposals that resembles the true paths 'better' than the Brownian bridge. Second, figure 5.7 suggests that the estimation procedure performs better for data in the oscillatory regime than data in the excitatory regime with respect to all four drift parameters. This may intuitively be explained by the fact that in the oscillatory setting, in the limit of no noise, the drift is observable, whereas in the excitatory regime, only the location of the steady state can be observed. Thus, less information about the drift is available in the latter case, even if the noise makes some inference possible.

In this chapter we have focused on the setting where all coordinates are discretely observed without measurement noise, and the diffusion matrix Γ is of full rank. In some applications this is not the case. The methodology described here can be extended to work when the observations are contaminated by measurement noise or when not all coordinates are observed. The latter setting is the topic of chapter 6.

If only a subset of the coordinates includes noise (the hypo-elliptic setting) the problem becomes much harder. The methodology breaks down, as it relies on the equivalence of (Gaussian) measures. To solve this problem, one could make an approximation that includes a small amount of noise, however, this may result in numerical instabilities when

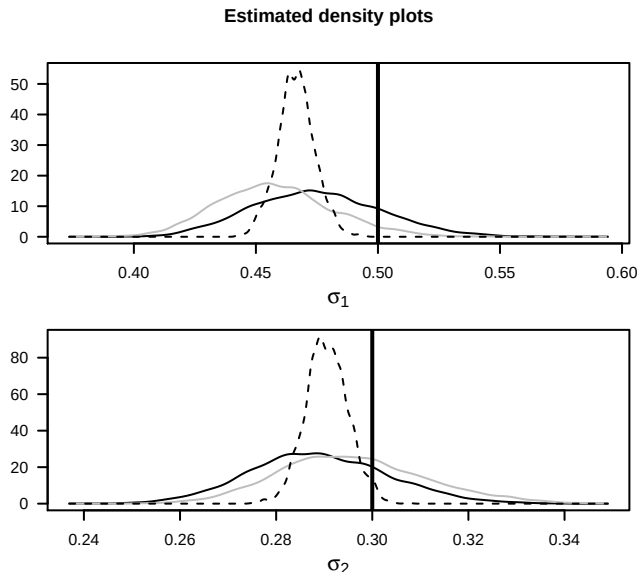

Figure 5.10: *Density plots of the sample posterior of the diffusion parameters. Solid and gray curves represent excitatory and oscillatory data respectively for* $n = 200$*. Black dashed curve represents excitatory data for* $n = 2000$*. Vertical lines denote true parameter values.*

inverting the diffusion matrix Γ. To effectively deal with the hypo-elliptic case, more sophisticated methods are required. See for instance Pokern et al. (2009) and Samson and Thieullen (2012).

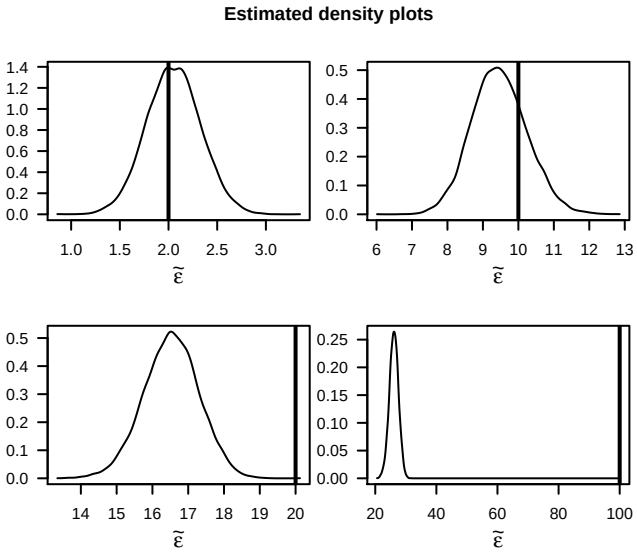

Figure 5.11: *Density plots for output of the Gibbs sampler for different values of $\tilde{\varepsilon}$. Vertical lines denote true values of $\tilde{\varepsilon}$.*

6

Parameter estimation for multidimensional diffusions, partially observed

In this chapter we build upon the ideas of chapter 5 and describe a data augmentation scheme and a Gibbs sampler which can be used for parameter estimation in discretely observed multivariate Ito diffusions with (potentially) latent coordinates. It is required that the Lamperti transform of the diffusion exists. For data augmentation, we apply a method described by Roberts and Stramer (2001) and Papaspiliopoulos and Roberts (2012) among others. The general idea is (theoretically) to impute continuous time data, distributed according to the continuous time SDE, and then use a Gibbs sampler to iteratively update imputed data and parameters.

We consider discretely observed data where some coordinates are completely unobserved. This makes parameter estimation more difficult because less information is available compared to the fully observed case. Furthermore, parameter identification problems arise, which are difficult to handle in general, without taking into account the specific features of the model. If there are no latent coordinates the framework reduce to the setup considered in chapter 5.

The estimation procedure is implemented in a new `R`-package called `BIPOD` and it currently works for the 2-dimensional Ornstein-Uhlenbeck process, the FitzHugh-Nagumo model and the extended FitzHugh-Nagumo model, with more models to come. See chapter 8 for details.

6.1 Statistical model and notation

Consider the (multivariate) Ito diffusion model as in chapter 5

$$\mathrm{d}V_t = b(V_t;\theta)\,\mathrm{d}t + \Sigma(V_t;\sigma)\,\mathrm{d}B_t, \tag{6.1}$$

with V_t a d-dimensional stochastic process, B_t a d-dimensional standard Brownian motion, and functions b and Σ taking values in $\mathbb{R}^d$ and the set of $d \times d$ matrices, respectively. We will assume that a solution to (6.1) exists, that Σ is uniformly invertible in both arguments, and that the dimension of θ and σ is p_1 and p_2, respectively.

6.1.1 Latent coordinates

In many applications of model (6.1) it is natural to consider latent coordinates. Therefore the process V is split in two compartments, $V_t = (V_t^{(1)}, V_t^{(2)})^T$ with $V_t^{(1)}$ of dimension $d_1 > 0$ and $V_t^{(2)}$ of dimension $d_2 \geq 0$, with $d_1 + d_2 = d$. The aim is to make inference for the parameters θ and σ governing the diffusion (6.1) given a discretely observed sample from $V^{(1)}$. In many cases this will not provide enough information to avoid model identification problems: Consider a two-dimensional version of (6.1) with constant diffusion coefficient, and

$$b(x,t) = \begin{pmatrix} b_1(x_1) + x_2 + \mu_1 \\ b_2(x_1) + x_2 + \mu_2 \end{pmatrix}.$$

Then the process $\tilde{V} := (V^{(1)}, V^{(2)} + \gamma)^T$, gives rise to a model with the same diffusion and the new drift becomes

$$\tilde{b}(x,t) = \begin{pmatrix} b_1(x_1) + x_2 + (\mu_1 + \gamma) \\ b_2(x_1) + x_2 + (\mu_2 + \gamma) \end{pmatrix},$$

which is just a re-parametrization of the original model $\mu_i \mapsto \mu_i + \gamma$. If $\gamma := -\mu_1$, there is no information about μ_1 in the first coordinate. Thus, if only $V_t^{(1)}$ is (discretely) observed it is impossible to identify μ_1 without additional information about the parameters guiding the latent coordinates $V_t^{(2)}$, either by assuming knowledge about $V_t^{(2)}$ or some of the parameters. If $d_2 = 0$ all coordinates are discretely observed.

Define the discrete sample $D_n = (D_n^{(1)}, D_n^{(2)})$, where

$$D_n^{(i)} = \{V_0^{(i)}, V_1^{(i)}, \ldots, V_n^{(i)}\}, \ i = 1, 2,$$

with $V_i^{(i)} := V_{t_i}^{(i)}$, and $t_0 = 0 < t_1 < \ldots < t_n = T$ equidistant time points such that $t_i - t_{i-1} = \Delta$. In the following we think of $D_n^{(1)}$ as discretely observed and $D_n^{(2)}$ as unobserved. Equidistant time steps are only for notational simplicity. Single subscripts V_i denote the process V at time t_i and interval subscripts $V_{(i;j)}$ will be used to denote the process V in the time interval $(t_i; t_j)$.

6.2 The estimation procedure

The aim is to estimate drift and diffusion parameters, θ and σ, in model (6.1) from discrete observations $D_n^{(1)}$. The model is defined in continuous time and it is therefore natural to consider data augmentation to obtain continuous time augmented data.

Recall the Lamperti transformed process

$$\mathrm{d}Z_t = \alpha(Z_t; \theta, \sigma)\, \mathrm{d}t + \ \mathrm{d}W_t, \tag{6.2}$$

from section 2.1.2. Since V and Z are linked one-to-one (conditional on σ) by the Lamperti transform, we can update Z instead of V in the Gibbs sampler.

Following the approach from chapter 5 it seems natural to construct a three component Gibbs sampler that updates drift parameters, diffusion parameters, and the latent path iteratively, always conditional on the observed data. We will pursue this approach, although it is not straightforward to update the entire latent path in one step as in chapter 5. This is because we have to update the endpoints $Z_0^{(2)}$ and $Z_n^{(2)}$ and for other reasons which will be described below. Thus we break down the step, to update the path, into four separate steps.

First step is to update the latent path between observation times $t_0, t_1, \ldots, t_n$ for both path components. Every step of the Gibbs sampler conditions on the current value of all

other components, such that this step is exactly the same as described in chapter 5 and it will not be the focus here.

In the second step we update the latent path component at observation times $t_1, t_2, \ldots, t_{n-1}$, except for the start and end point t_0 and t_n. The update of the latent component at these two time points are handled separately in step three and four. It is important to state that in both step two three and four, we do not update just one point of the path, as this would lead to discontinuities. Instead we update a piece of the latent path. In the following we give the details of step two three and four, but first we recall the linear transformation from (5.18)

$$\tilde{V}_t = Z_t - \left(1 - \frac{t - t_{i-1}}{\Delta}\right) Z_{t_{i-1}} - \frac{t - t_{i-1}}{\Delta} Z_{t_i}, \tag{6.3}$$

which transformed a Brownian bridge starting and ending at Z_{i-1} and Z_i into a standard Brownian bridge, starting and ending in 0. We used this transformation in order to be able to update the latent path on a common state space (continuous functions starting and ending at zero). Without this transformation the endpoints of $Z_{(i;i+1)}$ would change from one update to the next, when σ was updated.

6.2.1 Updating the latent path component at observation times

In order to update $Z_i^{(2)}$, the latent component at time t_i, $i \in \{1, \ldots, n-1\}$, we consider the latent component on the double interval $(t_{i-1}; t_{i+1})$. If we are able to update on this interval we may proceed by updating $Z_{(i;i+2)}^{(2)}$ next, where $Z_{(i;i+1)}^{(2)}$ was already updated in the previous step, and so on. Thus we update in overlapping intervals, and it suffices to focus on a single double interval in order to update the latent component $Z_i^{(2)}$.

To clarify, we want to sample $Z_{(i-1;i+1)}^{(2)}$ conditional on $Z_{i-1}, Z_{i+1}, Z_{(i-1;i+1)}^{(1)}$ and the parameters σ and θ, i.e. $\pi(Z_{(i-1;i+1)}^{(2)} \mid Z_{i-1}, Z_{i+1}, Z_{(i-1;i+1)}^{(1)}; \sigma, \theta)$. Note that this conditional distribution is proportional to $\pi(Z_{(i-1;i+1)} \mid Z_{i-1}, Z_{i+1}; \sigma, \theta)$, when considered as a function of $Z_{(i-1;i+1)}^{(2)}$.

At first sight one could try to proceed almost as in chapter 5: Sample a skeleton of a d_2-dimensional standard Brownian bridge for the latent component $\bar{V}_{(i-1;i+1)}^{(2)}$ from time t_{i-1} to time t_{i+1} starting and ending at zero. Then transform this process to start and end at $Z_{i-1}^{(2)}$ and $Z_{i+1}^{(2)}$ at times t_{i-1} and t_{i+1} as in a d_2-dimensional version of (6.3). This path, combined with the fixed value of the first component $Z_{(i-1;i+1)}^{(1)}$, is then used as the proposal for the Metropolis-Hastings algorithm. The problem with this approach becomes apparent in the next iteration of the Gibbs sampler, where we update $Z_{(i;i+1)}$. In chapter 5 we saw that this step involves the tilting from the standard Brownian bridge $\bar{V}_{(i;i+1)}$ to $Z_{(i;i+1)}$, which is a problem when $\bar{V}_i$ is no longer 0, because it was updated in the previous step. Hence $\bar{V}_{(i;i+1)}$ is no longer defined on the space of continuous functions starting and ending in 0 at times t_i and t_{i+1}, and the algorithm breaks down.

Independence sampler

Instead of proposing the Brownian bridge from 0 to 0 and then transforming it, we may propose directly from the d_2-dimensional Brownian bridge starting and ending at $Z^{(2)}_{i-1}$ and $Z^{(2)}_{i+1}$, respectively and then combine this path with the current value of $Z^{(1)}_{(i-1;i+1)}$ to get a d-dimensional path. That is, with $\tilde{Z}^{(2)}_{(i-1;i+1)}$ the proposal, we construct $\tilde{Z}_{(i-1;i+1)} := (Z^{(1)}_{(i-1;i+1)}, \tilde{Z}^{(2)}_{(i-1;i+1)})^T$. If the proposal is accepted in the Metropolis-Hastings-algorithm, we update the latent path $\bar{V}^{(2)}_{(t_{i-1};t_{i+1})}$ and $\bar{V}^{(2)}_{(t_i;t_{i+1})}$ using (6.3).

In order to compute the acceptance ratio of the Metropolis-Hastings-algorithm, we find the Radon-Nikodym derivative of the proposed process with respect to the Brownian bridge measure. It is proportional to

$$\begin{aligned} &G(t_{i-1}, t_{i+1}, Z_{[i-1;i+1]}\alpha, I_d; \theta, \sigma) \\ &= \exp\left(\int_{t_{i-1}}^{t_{i+1}} \alpha(Z_s;\theta,\sigma)^T \, \mathrm{d}Z_s - \frac{1}{2}\int_{t_{i-1}}^{t_{i+1}} \alpha(Z_s;\theta,\sigma)^T\alpha(Z_s;\theta,\sigma) \, \mathrm{d}s\right), \end{aligned} \tag{6.4}$$

as stated in section 2.2. Recall again that we only update the second path component so the first component of the proposal is kept fixed at the current value of $Z^{(1)}_{(i-1;i+1)}$.

Conditional on the endpoints $Z^{(2)}_{i-1}$ and $Z^{(2)}_{i+1}$ the proposal is sampled independently of any parameters so the proposal density is proportional to one and it does not change the acceptance probability of the Metropolis-Hastings-algorithm. The procedure is summarized in table 6.1.

Initialize

1) Initialize a skeleton path $Z_{(i-1;i+1)}$ starting and ending at Z_{i-1} and Z_{i+1}.

Iterate (step k)

2) For the current value of $Z_{(i-1;i+1)}$ approximate the weight in (6.4), and denote it w_k. Generate a proposal skeleton path, $\tilde{Z}^{(2)}_{(i-1;i+1)}$ according to the Brownian bridge distribution in (2.11) and approximate the weight in (6.4), denoted $\tilde{w}$, using $\tilde{Z}_{(i-1;i+1)} := (Z^{(1)}_{(i-1;i+1)}, \tilde{Z}^{(2)}_{(i-1;i+1)})^T$.

3) Let $(Z_{(i-1;i+1)})_{k+1} = \begin{cases} \tilde{Z}_{(i-1;i+1)} & \text{with prob. } \min\left(1, \frac{\tilde{w}}{w_k}\right) \\ (Z_{(i-1;i+1)})_k & \text{otherwise} \end{cases}$.

4) Update $\bar{V}_{(i-1;i+1)}$ according to (6.3).

Table 6.1: Independent Metropolis-Hastings sampler, generating latent component at times $t_1, \ldots, t_{n-1}$.

In order to reduce computational costs and speed up the process, one may realize that it is not necessary to sample the entire path $Z^{(2)}_{(i-1;i+1)}$, because the current value of the

standard Brownian bridges $\tilde{V}_{(i-1;i)}$ and $\tilde{V}_{(i;i+1)}$ can be reused. Thus it suffices to sample the single point $Z_i^{(2)}$, which, under the proposal, is distributed as the middle point of the Brownian bridge starting and ending at $Z_{i-1}^{(2)}$ and $Z_{i+1}^{(2)}$ at times t_{i-1} and t_{i+1}, respectively:

$$Z_i^{(2)} \sim \mathcal{N}\left(\frac{Z_{i-1}^{(2)} + Z_{i+1}^{(2)}}{2}; \frac{\Delta}{2} I_{d_2}\right). \tag{6.5}$$

Changing just this one point gives rise to a completely new path $Z_{(i-1;i+1)}$, since, by construction, $Z_{(i-1;i)}$ and $Z_{(i;i+1)}$ are given by (6.3) which, in this setting, can be rewritten as

$$Z_t^{(2)} = \tilde{V}_t^{(2)} + \left(1 - \frac{t - t_{i-1}}{\Delta}\right) Z_{t_{i-1}}^{(2)} + \frac{t - t_i}{\Delta} Z_{t_{i+1}}^{(2)}, t_{i-1} \leq t \leq t_i. \tag{6.6}$$

The procedure is summarized in table 6.2.

Initialize

1) Initialize a starting value for $Z_i^{(2)}$.

Iterate (step k)

2) For the current value of $Z_{(i-1;i+1)}$ approximate the weight in (6.4), and denote it w_k. Sample $\tilde{Z}_i^{(2)}$ according to (6.5) and compute $\tilde{Z}_{(i-1;i)}^{(2)}$ using the current value of $Z_{(i-1;i)}^{(1)}$, $\tilde{V}_{(i-1;i)}$ and (6.6). Repeat for $(i; i+1)$ and approximate the weight in (6.4), denoted by $\tilde{w}$, using $\tilde{Z}_{(i-1;i+1)} := (\tilde{Z}_{(i-1;i+1)}^{(1)}, \tilde{Z}_{(i-1;i+1)}^{(2)})^T$.

3) Let $(Z_{(i-1;i+1)})_{k+1} = \begin{cases} \tilde{Z}_{(i-1;i+1)}, & \text{with prob. } \min\left(1, \frac{\tilde{w}}{w_k}\right) \\ (Z_{(i-1;i+1)})_k, & \text{otherwise} \end{cases}$.

4) Update $\bar{V}_{(i-1;i+1)}$ according to (6.3).

Table 6.2: Improved version of independent Metropolis-Hastings sampler, generating latent component at times $t_1, \ldots, t_{n-1}$.

Random walk sampler

There are other ways to sample the proposal than the independence sampler. For $s \in (t_{i-1}; t_{i+1})$ and $\rho \in [0, 1]$, define the proposal as

$$\tilde{Z}_s = \left(\left(1 - \frac{s - t_{i-1}}{t_{i+1} - t_{i-1}}\right) Z_{i-1} + \frac{s - t_{i-1}}{t_{i+1} - t_{i-1}} Z_{i+1}\right)(1 - \rho) + \rho Z_s + \sqrt{1 - \rho^2}\tilde{V}_s. \tag{6.7}$$

Conditional on Z_{i-1} and Z_{i+1} the distribution of $\tilde{Z}_t$ is the same as the distribution of Z_t under the proposal, and the random walk proposal is therefore as valid as the independence sampler. Note the two extreme values of ρ: If $\rho = 1$ then $\tilde{Z}_t = Z_t$ and the path remains

the same always. If $\rho = 0$, we are back at the independence sampler. The best value of ρ depends on the type of model, but based solely on practical experience it is suggested to use values around 0.5, which typically yields better results than the value 0. Note that this type of proposal can also be used to update the latent component in chapter 5, even though it is not described therein.

6.2.2 Updating the endpoints of the latent component

We sample $Z_0^{(2)}$ by proposing according to the distribution $\mathcal{N}(Z_1; \Delta)$. This is motivated by the fact that a time reversed Brownian motion has the same distribution as the original Brownian motion. We also put a prior distribution on $Z_0^{(2)}$ and assume that it is Gaussian. Analogously to the argument given above we update the entire latent path in $[0, t_1)$ by updating $Z_0^{(2)}$ and then obtaining $Z_{(0,1)}^{(2)}$ via the transformation of $\bar{V}_{(0,1)}^{(2)}$ given in (6.6). Combining the proposal with the value of $Z_{(0,1)}^{(1)}$ we can compute the product of the prior and the Radon-Nikodym-derivative for use in the acceptance probability of the Metropolis-Hastings-algorithm.

The last endpoint is updated as the first endpoint except we do not include a prior. Thus we sample $Z_n^{(2)} \sim \mathcal{N}(Z_{n-1}^{(2)}; \Delta)$ and re-compute $Z_{(n-1;n)}^{(2)}$ from $\bar{V}_{(n-1;n)}^{(2)}$. Combining this proposal with the fixed value of $Z_{(n-1;n)}^{(1)}$ we can compute the Radon-Nikodym-derivative for use in the acceptance probability of the Metropolis-Hastings-algorithm.

6.3 Simulation study

We will now focus on the application of the described Bayesian methods to perform parameter estimation in some 2-dimensional diffusion models where the second coordinate is unobserved. Some general remarks are in place which apply to all simulations. Several simulations were carried out for different models and parameter values and here we only show a small sample of these. In general the performance in both the FitzHugh-Nagumo-model and the extended version was good. the same was true for the Ornstein-Uhlenbeck, although to a lesser extend.

For each data set we used 100 observations with a time step between observations of size 0.1. Drift parameters were updated using uninformative prior although performance only improved by choosing reasonable priors (not shown). The diffusion parameters were updated with a random walk Metropolis-Hastings-step, after a re-parametrization to allow for negative parameter values. Thus we used the log-transformation. In all simulations the diffusion matrix was $\Sigma = \text{diag}(0.5, 0.3)$ and the prior, on the log scale was taken to be $\mathcal{N}(\log(0.5) + 2, \log(0.3) + 1);\ \text{diag}(5, 5))$.

The random walk covariance was adjusted to obtain an acceptance rate between 15% and 60%. Initial value for the latent path was set to 1 for all time points, and the prior distribution on $Z_0^{(2)}$ was $\mathcal{N}(0.2; 10)$.

4 data points were imputed between consecutive observations. Increasing this number lead in general to more accurate estimates but also longer computation times and a stronger autocorrelation meaning that the chain should run for a longer period in order to explore the state space. Each chain was run for 30.000 iterations and the output was saved for every 10'th iteration leading to a sample size for each simulation of 3.000.

6.3.1 The FitzHugh-Nagumo model

In this section the FitzHugh-Nagumo model is re-parameterized to obtain linearity in the drift parameters. Thus we use ε instead of $1/\varepsilon$. The model is

$$\mathrm{d}X_t = \varepsilon\left(X_t - X_t^3 - Y_t + s\right)\,\mathrm{d}t + \sigma_1\,\mathrm{d}W_t^{(1)}, \tag{6.8}$$

$$\mathrm{d}Y_t = (\gamma X_t - Y_t + \beta)\,\mathrm{d}t + \sigma_2\,\mathrm{d}W_t^{(2)}. \tag{6.9}$$

Data from the FitzHugh-Nagumo model was simulated for two different sets of parameter values $\varepsilon = 10, \theta_2 = 5, \theta_3 = 1.5, \sigma_1 = 0.5, \sigma_2 = 0.3$ fixed and $\theta_4 = 0.6$ or 1.4, as shown in Figure 2.3 but for half the time length. Both data sets were simulated in order to investigate the performance of the estimation procedure for both excitatory data and oscillatory data.

Identifiability of parameters

When only the first coordinate of the FitzHugh-Nagumo model is observed, not all parameters can be identified. First, the transformation $Y_t \mapsto Y_t - s/\varepsilon$ leads to the model

$$\begin{aligned}\mathrm{d}X_t &= \varepsilon\left(X_t - X_t^3 - Y_t\right)\,\mathrm{d}t + \sigma_1\,\mathrm{d}B_t^{(1)},\\ \mathrm{d}Y_t &= \left(\gamma X_t - Y_t + \beta - \frac{s}{\varepsilon}\right)\,\mathrm{d}t + \sigma_2\,\mathrm{d}B_t^{(2)}.\end{aligned}$$

On the other hand, the transformation $Y_t \mapsto Y_t - \beta$ leads to

$$\begin{aligned}\mathrm{d}X_t &= \varepsilon\left\{\left(X_t - X_t^3 - Y_t\right) - \varepsilon\beta + s\right\}\,\mathrm{d}t + \sigma_1\,\mathrm{d}B_t^{(1)}\\ \mathrm{d}Y_t &= (\gamma X_t - Y_t)\,\mathrm{d}t + \sigma_2\,\mathrm{d}B_t^{(2)}.\end{aligned}$$

It follows that when Y is unobserved it is impossible to distinguish between the two models and one can therefore not identify both s and β. Therefore we decide to fix β.

In Figure 6.3 is shown trace plots for the oscillatory data set. With these parameters the data set contains approximately 3.5 oscillations. Even though the parameters are initiated far from their optimal values they move fast to the stationary regime. For the s parameter there seems to be a strong autocorrelation and this is verified by an autocorrelation plot (not shown).

Not only the parameters are updated in the Gibbs sampler, but also the latent coordinate. We store for each observation the latent coordinate at times $t_0, t_1, \ldots, t_n$. Figure 6.4 shows

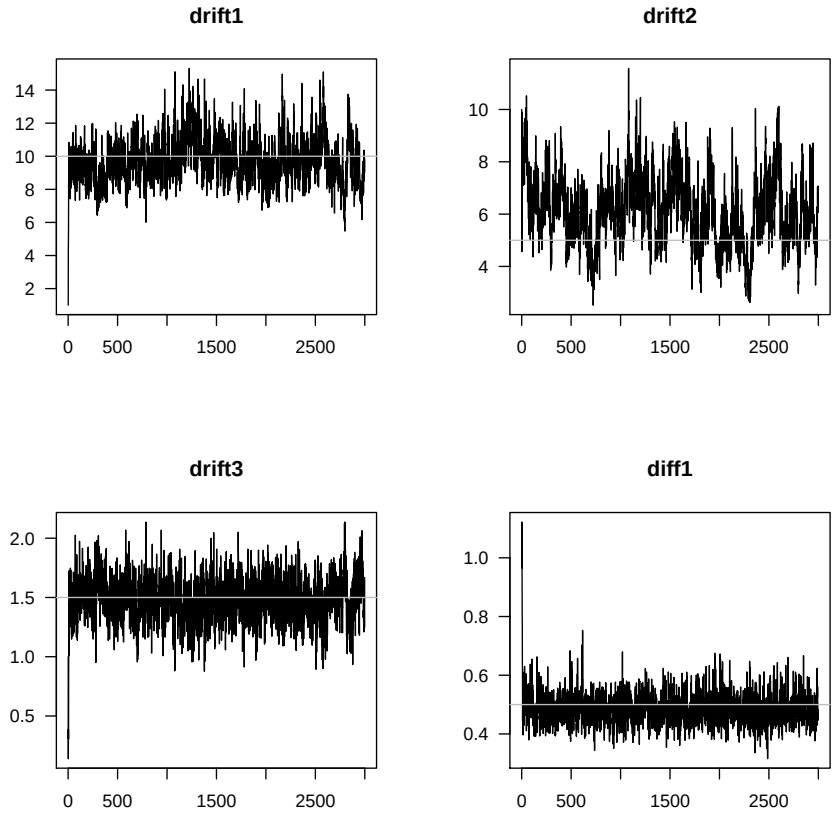

Figure 6.3: *Trace plots for oscillatory data from the FitzHugh-Nagumo model. Gray lines denote parameter values used to generate simulated data. drift1=ε, drift2=s, drift3=γ, diff1=σ_1.*

path output from the Gibbs sampler. That is the imputed data for the latent coordinate at observation times $Z_i^{(2)}, i = 0, \ldots, n$. Black line denotes the true data points and the gray lines are the 5% and 95% quantile of the marginal sample distribution. It seems that the algorithm is able to replicate the latent data points quite well even though it was started as a straight line with value equal to 1.

For the excitatory data the estimation procedure also performed well even though the data only contained one spike. The trace plot is shown in Figure 6.5. The autocorrelation was slightly larger for excitatory data than oscillatory data. See Figure 6.6. This is in line with the conclusion from chapter 5, stating that in the fully observed case, performance was better for the oscillatory data compared to the excitatory data.

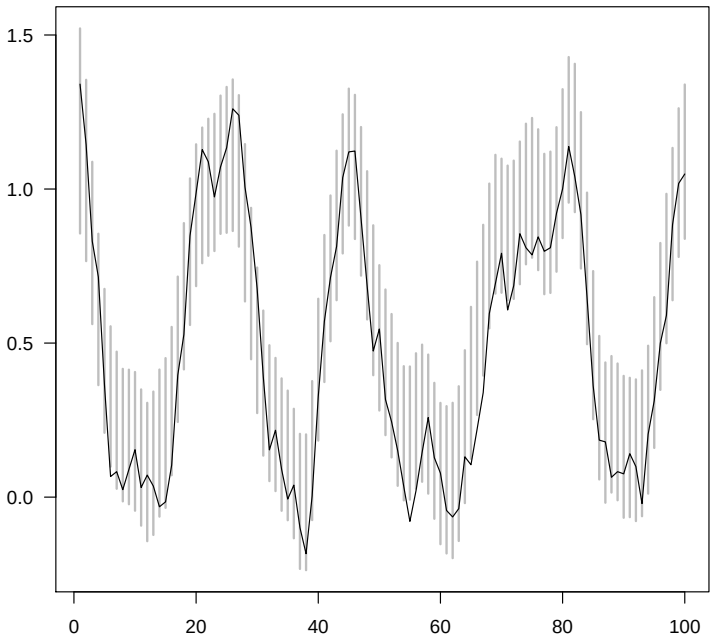

Figure 6.4: *Sampled points $D_n^{(2)}$ from Gibbs sampler. Black: True data. Gray lines: 5% to 95% quantile of marginal sample.*

6.3.2 The two-dimensional Ornstein-Uhlenbeck model

The two dimensional Ornstein-Uhlenbeck process is given by

$$\mathrm{d}V_t = -B(V_t - A)\,\mathrm{d}t + \Sigma\,\mathrm{d}B_t,$$

with the restriction that the real part of the eigenvalues of the drift matrix B should be positive to obtain stationarity. For the simulations we chose

$$A = \begin{pmatrix} 6 \\ 8 \end{pmatrix}, \quad B = \begin{pmatrix} 5 & 2 \\ 4 & 6 \end{pmatrix}, \quad \Sigma = \begin{pmatrix} 0.5 & 0 \\ 0 & 0.3 \end{pmatrix},$$

with eigenvalues $3 \pm \sqrt{2}$.

Figure 6.7 shows trace plots for the Ornstein-Uhlenbeck model. It seems that B_{21} is very difficult to estimate compared to B_{11} and B_{12}. This is confirmed by the autocorrelation plot that shows a huge autocorrelation for B_{21} and almost none for B_{11} and B_{12}. For the Ornstein-Uhlenbeck process it was more difficult to estimate parameters, compared to the FitzHugh-Nagumo model. In the Ornstein-Uhlenbeck case, the output of the Gibbs sampler

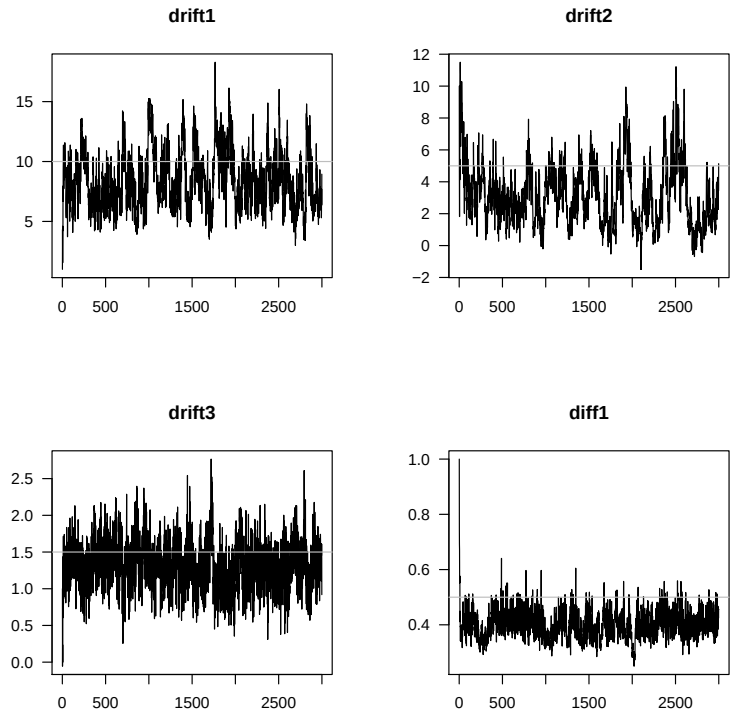

Figure 6.5: *Trace plots for excitatory data from the FitzHugh-Nagumo model. Gray lines denote parameter values used to generate simulated data. drift1=ε, drift2=s, drift3=γ, diff1=σ_1.*

was not always stable. Occasionally the output from the Gibbs sampler drifted away from the state of the stationary distribution (not shown). This could either be attributed to a programming bug or too little information in data to estimate three parameters at once.

6.3.3 The extended FitzHugh-Nagumo model

Consider again the extended version of the FitzHugh-Nagumo model.

$$\begin{aligned} \mathrm{d}X_t &= (-\alpha X_t^3 + \varepsilon(X_t - Y_t) + s)\,\mathrm{d}t + \sigma_1\,\mathrm{d}W_t^{(1)} \\ \mathrm{d}Y_t &= (\gamma X_t - Y_t + \beta)\,\mathrm{d}t + \sigma_2\,\mathrm{d}W_t^{(2)} \end{aligned}$$

with $\alpha, \epsilon > 0, \gamma > 1$, and $\beta, s \in \mathbb{R}$. For simulations we chose

$$\alpha = 8\ \varepsilon = 12,\ s = 5,\ \gamma = 1.5,\ \beta = 0.5,\ \sigma_1 = 0.5,\ \sigma_2 = 0.3.$$

and estimated $\alpha, \varepsilon, \beta$ and σ_1. Figure 6.8 shows trace plots for the four parameters after a burn in period of around 100 iterations. The samples are located around the value that

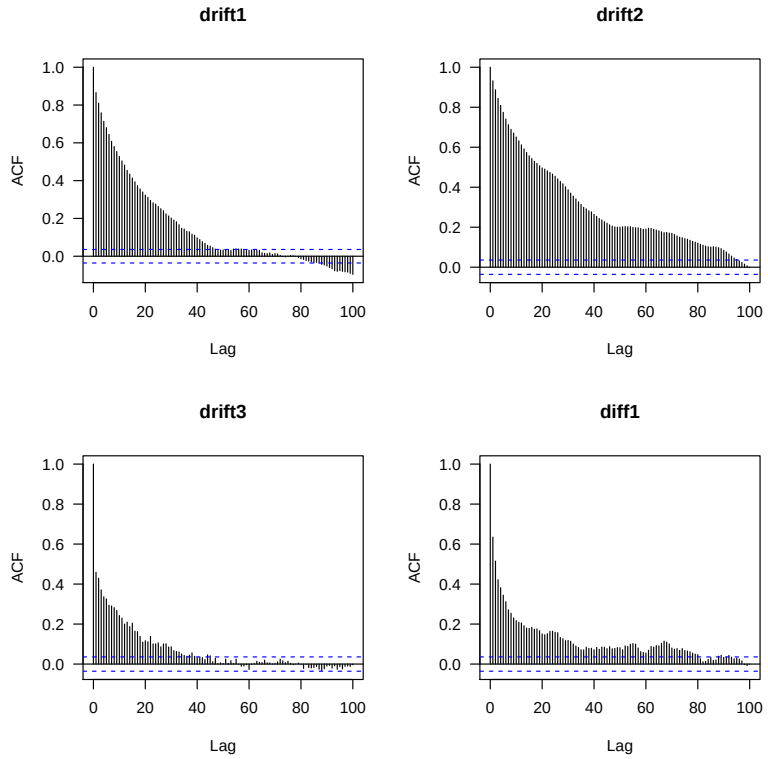

Figure 6.6: *Autocorrelation plot for the estimated parameters of the excitatory FitzHugh-Nagumo model. drift1=ε, drift2=s, drift3=γ, diff1=σ_1*

generated the simulated data, but there seems to be some very slow oscillations indicating a large autocorrelation.

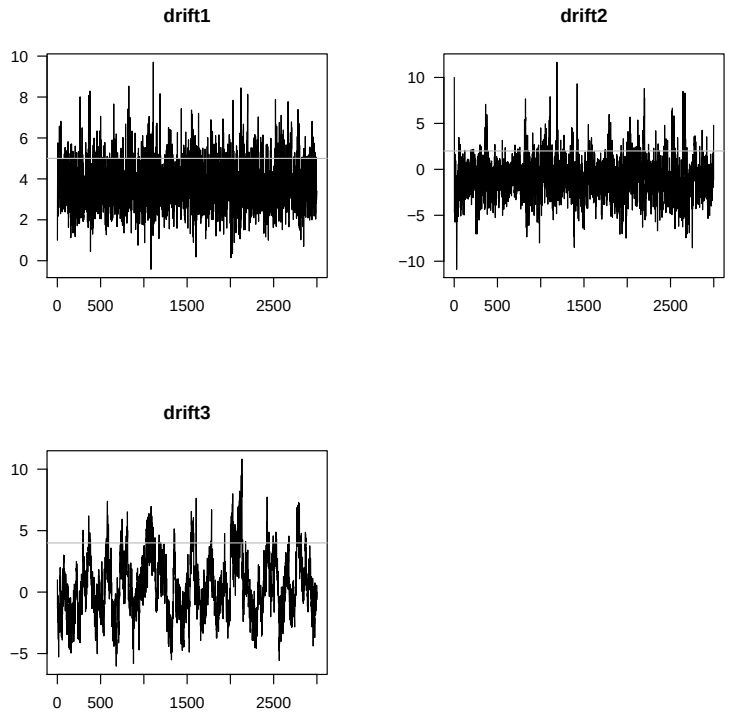

Figure 6.7: *Trace plots for the Ornstein-Uhlenbeck model. drift1=*B_{11}*, drift2=*B_{12}*, drift3=*B_{21}*.*

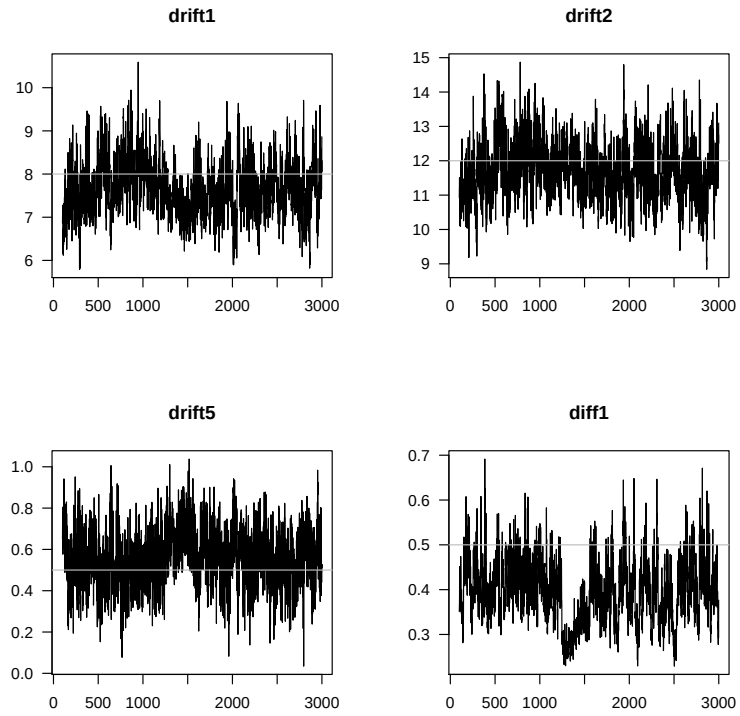

Figure 6.8: *Trace plots for the extended FitzHugh-Nagumo model. drift1=α, drift2=ε, drift5=β, diff1=σ_1.*

7

Parameter identifiability for partially observed diffusions

In this chapter we repeatedly use the notion of partially and fully observed data. Partially observed data refers to the scenario where the statistical model has two compartments, where one is completely unobserved, and the other is discretely observed; just as in chapter 6. The fully observed case refers to discretely observed data where both compartments are discretely observed, and it was the case in chapter 5.

Parameter estimation for partially observed diffusions presents an additional challenge compared to the scenario where all coordinates of the process is observed. In the fully observed case the model is typically parameterized such that all parameters are identifiable. In the partially observed model, the first problem is to recognize which parameters can be estimated at all, since one can only hope to infer about parameters from the distribution of the marginal component. Consider the simple example $Z_t = (X_t, Y_t)$ where the X and Y components are mutually independent. Clearly, if only X_t is observed, one can never make inference about the parameters controlling Y_t. In more complicated models, where the components are not independent, one typically finds that not all parameters are identifiable and it may be necessary to fix or standardize some parameters in order to identify others. Different parameterizations may be natural depending on whether one has access to data from the fully or the partially observed model. As an example, consider the drift matrix of the Ornstein-Uhlenbeck model: In the fully observed case it is typically parameterized directly in terms of the entries of the drift matrix, but in the marginally observed case it is more natural to consider the eigenvalues and the entries of the eigenvectors as parameters. Note that the eigenvectors only contain two parameters because the eigenvector is only identified up to proportionality. Eigenvalues also contain only two parameters - either complex conjugate or two reals.

Another issue arise in the case where the process is only partially observed, which may also appear in the fully observed case: Assume we have a parametrization for the partially observed case that is not over-parameterized, potentially by fixing some of the parameters from the fully observed model. From a practical point of view, it is very interesting to know how much information is available about the parameters of interest. Consider again the zero mean Ornstein-Uhlenbeck model with unit diffusion matrix: If the entry of the drift matrix b_{12} is zero, then the first marginal is just a one-dimensional Ornstein-Uhlenbeck with parameters b_{11} and σ_1, making inference about b_{21}, b_{22} impossible in the marginal model. If $b_{12} \neq 0$ is numerically small (relative to the other parameters), only a small amount of information is available about b_{21}, b_{22}. Even though information about the parameters of the latent component theoretically may be extracted from the marginal distribution, it is not always possible to do so in practice.

For general diffusions it is not easy to make strong statements about identifiability. In this chapter we will primarily focus on parameter identification for the partially observed two-dimensional Ornstein-Uhlenbeck process. However we start with a general remark about linear transformations.

7.1 Linear transformation of latent coordinate

Consider the stochastic differential equation

$$\mathrm{d}V_t = b(V_t;\theta)\,\mathrm{d}t + \Sigma(V_t;\sigma)\,\mathrm{d}W_t,$$

as in chapter 6. We assume again that $V_t = (V_t^{(1)}, V_t^{(2)})^T$ with $V_t^{(1)}$ discretely observed and $V_t^{(2)}$ unobserved. Such models are typically specified so that over parameterization is not a problem. However, this does not automatically carry over to the case where one of the compartments is unobserved.

Since the second component is unobserved, we consider transformations of the form $g(x,y) = (x, g_2(x,y))$. With $z = (x,y)$ we may write $g(z)$ or $g(x,y)$ interchangeably to simplify notation. Using Ito's formula the components of the transformed drift $\tilde{b}$ become

$$\tilde{b}_k(Z_t;\theta,\sigma) = \sum_{i=1}^{d} b_i(g^{-1}(Z_t);\theta)\frac{\partial g_k}{\partial x_i}(g^{-1}(Z_t)) + \frac{1}{2}\sum_{i,j=1}^{d}\Gamma_{ij}(g^{-1}(Z_t);\sigma)\frac{\partial^2 g_k}{\partial x_i \partial x_j}(g^{-1}(Z_t)),$$

where Z_t denotes the transformed process $k = 1,2$ and $\Gamma = \Sigma\Sigma^T$. The g function may also depend on the parameters, but this is hidden for notational simplicity. The diffusion function is given by

$$\tilde{\Sigma}(Z_t;\theta,\sigma) = \nabla g_k(g^{-1}(Z_t))\Sigma(g^{-1}(Z_t)).$$

Notice that in order to satisfy the quadratic variation identity in both parameterizations, it should hold that

$$\left(\Sigma\Sigma^T\right)_{11} = \left(\tilde{\Sigma}\tilde{\Sigma}^T\right)_{11}. \tag{7.1}$$

Furthermore, for the marginal distribution of the first component to remain unchanged it should hold that

$$\tilde{b}_1(Z_t;\theta,\sigma) = b_1(V_t;\theta). \tag{7.2}$$

Example 7.1. When b is affine in the parameters, (7.2) is satisfied when $g(x,y) = (x, \alpha + \beta y)$. In this case

$$\tilde{b}_1(Z_t;\theta,\sigma) = \sum_{i=1}^{d} b_i\left(Z_t^{(1)}, \frac{Z_t^{(2)} - \alpha}{\beta};\theta\right),$$
$$\tilde{b}_2(Z_t;\theta,\sigma) = \sum_{i=1}^{d} \beta b_i\left(Z_t^{(1)}, \frac{Z_t^{(2)} - \alpha}{\beta};\theta\right),$$

and the new diffusion function $\tilde{\Sigma}$ becomes

$$\tilde{\Sigma}_{ij}\left(Z_t;\sigma\right) = \Sigma_{ij}\left(Z_t^{(1)}, \frac{Z_t^{(2)} - \alpha}{\beta};\sigma\right)\beta^{1_{(i=2)}}.$$

Thus in order to satisfy (7.1), it is sufficient that $\Sigma_{1j}(Z_t^{(1)}, Z_t^{(2)}; \sigma)$ does not depend on $Z_t^{(2)}$ for $j = 1, 2$.

♦

7.2 The two-dimensional Ornstein-Uhlenbeck process

Consider the process

$$\mathrm{d}V_t = -BV_t \,\mathrm{d}t + \Sigma \,\mathrm{d}W_t,$$

where B and Σ are 2×2-matrices and V_t, W_t are 2×1-matrices with W_t a two-dimensional standard Brownian motion.

From Example 7.1 it follows that a linear transformation of the latent coordinate in the two-dimensional Ornstein-Uhlenbeck model case gives a transformed model, Z_t,

$$\mathrm{d}Z_t = -B \begin{pmatrix} 1 & \frac{1}{\beta} \\ \beta & 1 \end{pmatrix} \left[Z_t - \begin{pmatrix} 1 & 0 \\ 0 & \beta \end{pmatrix} A + \begin{pmatrix} \alpha & 0 \\ 0 & \alpha\beta \end{pmatrix} B \begin{pmatrix} 1 \\ \frac{1}{\beta} \end{pmatrix} \right] \mathrm{d}t + \Sigma \begin{pmatrix} 1 & 0 \\ 0 & \beta \end{pmatrix} \mathrm{d}W_t,$$

and the parameters change from

$$\{b_{ij}, a_i, \sigma_i \mid i, j = 1, 2\}$$

to

$$\{b_{11}, b_{22}, b_{12}/\beta, b_{21}\beta, a_1 + b_{12}\alpha/\beta, a_2\beta + b_{22}\alpha, \sigma_1, \beta\sigma_2\}.$$

Another more thorough way to approach the problem of parameter identifiability is to look directly at the marginal distribution of the observed compartment as in Jacobsen (2011). Here the first marginal for the two-dimensional Ornstein-Uhlenbeck was investigated to find out which drift and diffusion matrices that were equivalent to a diagonal drift matrix and arbitrary diffusion matrix, in the sense of same first marginal distribution. The expression for the distribution of the first marginal is given in (7.4).

Define $\Gamma = \Sigma\Sigma^T$ and let

$$C := \int_0^\infty e^{-Bs} \Gamma e^{-B^T s} \,\mathrm{d}s,$$

the variance of the stationary process. Assume the initial distribution is $\mathcal{N}(0, C)$ such that the process is stationary with $\mathrm{E}(V_s) = 0$ and cross covariance

$$\mathrm{E}(V_s V_{s+t}^T) = Ce^{-B^T t}, \quad s, t \geq 0. \tag{7.3}$$

In order to evaluate (7.3) it is necessary to compute the matrix exponential, and since $\lambda_1 \neq \lambda_2$ and both are different from 0, one can find P such that $B = PDP^{-1}$, with $D = \operatorname{diag}(\lambda_j)_{j=1,2}$.

Then

$$\begin{aligned} Ce^{-B^T t} &= \int_0^\infty Pe^{-Ds}P^{-1}\Gamma(P^{-1})^T e^{-Ds}P^T \,\mathrm{d}s (P^{-1})^T e^{-Dt}P^T \\ &= P\int_0^\infty e^{-Ds}P^{-1}\Gamma(P^{-1})^T e^{-Ds}\,\mathrm{d}s\; e^{-Dt}P^T. \end{aligned}$$

For the matrix integral we get that the (m,n)'th coordinate is

$$\begin{aligned} \left(\int_0^\infty e^{-Ds}P^{-1}\Gamma(P^{-1})^T e^{-Ds}\,\mathrm{d}s\right)_{mn} &= \sum_{k,l=1}^2 \int_0^\infty \left(e^{-Ds}\right)_{mk}\left(P^{-1}\Gamma(P^{-1})^T\right)_{kl}\left(e^{-Ds}\right)_{ln}\,\mathrm{d}s \\ &= \left(P^{-1}\Gamma(P^{-1})^T\right)_{mn}\frac{1}{\lambda_m+\lambda_n}, \end{aligned}$$

since

$$\begin{aligned} \int_0^\infty (e^{-Ds})_{mk}(e^{-Ds})_{ln}\,\mathrm{d}s &= \int_0^\infty e^{-(\lambda_m+\lambda_n)s}1_{(m=k,l=n)}\,\mathrm{d}s \\ &= \frac{1}{\lambda_m+\lambda_n}1_{(m=k,l=n)}. \end{aligned}$$

Hence

$$\begin{aligned} \left(Ce^{-B^T t}\right)_{ij} &= \sum_{m,n,r} P_{im}\left(P^{-1}\Gamma(P^{-1})^T\right)_{mn}\frac{1}{\lambda_m+\lambda_n}\left(e^{-Dt}\right)_{nr}P^T_{rj} \\ &= \sum_{m,n} P_{im}\left(P^{-1}\Gamma(P^{-1})^T\right)_{mn}\frac{1}{\lambda_m+\lambda_n}e^{-\lambda_n t}P^T_{nj} \\ &= \sum_{m,n,k,r} P_{im}P^{-1}_{mk}P_{jn}P^{-1}_{nr}\Gamma_{kr}\frac{1}{\lambda_m+\lambda_n}e^{-\lambda_n t}. \end{aligned}$$

The marginal distribution of the first coordinate $V_t^{(1)}$ is completely specified by $(Ce^{-B^T t})_{11}$:

$$(Ce^{-B^T t})_{11} = \sum_{k,m,n,r}^2 \frac{1}{\lambda_r+\lambda_k}e^{-\lambda_k t}\Gamma_{nm}P_{1r}(P^{-1})_{rn}P_{1k}(P^{-1})_{km}. \tag{7.4}$$

We now focus on which matrices $\tilde{B}, \tilde{\Gamma}$ that lead to the same marginal cross covariance as B and Γ. First of all, the quadratic variation identity will identify Σ_{11} and therefore it must hold that $\tilde{\Gamma}_{11} = \Gamma_{11}$. From (7.4) it follows that the eigenvalues must be the same for B and $\tilde{B}$, and they are

$$\begin{aligned} \lambda &= \frac{\beta_{11}+\beta_{22} \pm \sqrt{(\beta_{11}+\beta_{22})^2 - 4(\beta_{11}\beta_{22}-\beta_{12}\beta_{21})}}{2} \\ &= \frac{\operatorname{Tr}(B) \pm \sqrt{\operatorname{Tr}(B)^2 - 4\det(B)}}{2}. \end{aligned} \tag{7.5}$$

It follows that uniqueness of the eigenvalues is equivalent to

$$\det(B) = \det(\tilde{B}) \text{ and } \operatorname{Tr}(B) = \operatorname{Tr}(\tilde{B}).$$

Therefore we can potentially write a candidate matrix $\tilde{B}$ in two ways because of the symmetry of the problem. First as

$$\tilde{B}_0(B, h, \delta). = \begin{pmatrix} b_{11} + h & \delta \\ 0 & b_{22} - h \end{pmatrix}, \quad \delta \in \mathbb{R}, \tag{7.6}$$

and second

$$\tilde{B}(B, h, \alpha) := \begin{pmatrix} b_{11} + h & \frac{b_{12}b_{21} - h(b_{11} - b_{22}) - h^2}{\alpha} \\ \alpha & b_{22} - h \end{pmatrix}, \quad \alpha \neq 0, h \in \mathbb{R}. \tag{7.7}$$

Note that $\tilde{B}(B, 0, b_{21}) = B$, and assume also that $\tilde{B}(B, h, \alpha) \neq 0$ as this would correspond to (7.6) with the coordinates switched. Complex h values are not allowed to avoid a complex $\tilde{B}$ matrix. Here we focus on candidates of the second form (7.7).

In order to find the matrix $\tilde{P}$ that is used to diagonalize $\tilde{B}$ we find that for an eigenvector $(v_1, v_2)^T$ and an eigenvalue λ,

$$\begin{aligned} (b_{11} + h)v_1 + \frac{b_{12}b_{21} - h(b_{11} - b_{22}) - h^2}{\alpha} v_2 &= \lambda v_1, \\ \alpha v_1 + (b_{22} - h)v_2 &= \lambda v_2. \end{aligned}$$

Since the eigenvectors are only unique up to a proportionality constant, we are free to choose $\tilde{P}_{11} = \tilde{P}_{22} = 1$. It follows that for $\lambda_1 - b_{22} + h \neq 0$,

$$\tilde{P} = \begin{pmatrix} 1 & \frac{\lambda_2 - b_{22} + h}{\alpha} \\ \frac{\alpha}{\lambda_1 - b_{22} + h} & 1 \end{pmatrix},$$

such that

$$\tilde{P}^{-1} = \frac{\lambda_1 - b_{22} + h}{\lambda_1 - \lambda_2} \begin{pmatrix} 1 & -\frac{\lambda_2 - b_{22} + h}{\alpha} \\ -\frac{\alpha}{\lambda_1 - b_{22} + h} & 1 \end{pmatrix},$$

Now one can see that

$$\tilde{P}_{ij} = \left(\frac{\lambda_j - b_{22} + h}{\alpha} \right)^{j-i}, \tag{7.8}$$

$$\tilde{P}^{-1}_{ij} = \frac{\lambda_1 - b_{22} + h}{\lambda_1 - \lambda_2} \left(\frac{\lambda_j - b_{22} + h}{\alpha} \right)^{j-i} (-1)^{i+j}. \tag{7.9}$$

In order to compute (7.4) the following lemma is useful.

Lemma 7.2. *Let $a_1, a_2 \neq 0$ and let $i, r, n, j, k, m \in \{1, 2\}$. Define*

$$\gamma_1(i,j,k,m,n,r) = (2-r)(1-i) + (r-1)(n-2) + (2-k)(1-j) + (k-1)(m-2),$$
$$\gamma_2(i,j,k,m,n,r) = (r-1)(2-i) + (2-r)(n-1) + (k-1)(2-j) + (2-k)(m-1).$$

Then

$$a_r^{r-i} a_n^{n-r} a_k^{k-j} a_m^{m-k} = a_1^{\gamma_1(i,j,k,m,n,r)} a_2^{\gamma_2(i,j,k,m,n,r)}.$$

Proof. First note that $a_n^{n-r} = a_{3-r}^{n-r}$, such that

$$a_r^{r-i} a_n^{n-r} a_k^{k-j} a_m^{m-k} = a_r^{r-i} a_{3-r}^{n-r} a_k^{k-j} a_{3-k}^{m-k}.$$

Also

$$a_r^{r-i} = a_1^{(2-r)(1-i)} a_2^{(r-1)(2-i)},$$

such that

$$a_r^{r-i} a_{3-r}^{n-r} = a_1^{(2-r)(1-i)+(r-1)(n-2)} a_2^{(r-1)(2-i)+(2-r)(n-1)}.$$

This motivates the definition of γ_1, γ_2 and the result follows. □

An short calculation shows that $\gamma_1 \in \{-2, -1, 0\}$ and $\gamma_2 \in \{0, 1, 2\}$ with $\gamma_1 + \gamma_2 = n + m - i - j$.

Now it follows from lemma 7.2 and equation (7.8) and (7.9) that

$$\begin{aligned}
&\tilde{P}_{ir}(\tilde{P}^{-1})_{rn}\tilde{P}_{jk}(\tilde{P}^{-1})_{km} \\
&= (\lambda_1 - b_{22} + h)^{\gamma_1(i,j,k,m,n,r)+2} (\lambda_2 - b_{22} + h)^{\gamma_2(i,j,k,m,n,r)} \frac{\alpha^{-(n+m-i-j)}}{(\lambda_1 - \lambda_2)^2} (-1)^{k+m+n+r},
\end{aligned}$$

with $0^0 := 1$. To emphasize the dependence on α and h, define

$$g(i,j,k,m,n,r,h,\alpha,B) := \tilde{P}_{ir}(\tilde{P}^{-1})_{rn}\tilde{P}_{jk}(\tilde{P}^{-1})_{km}.$$

Then

$$\left(Ce^{-\tilde{B}^T t}\right)_{11} = \sum_{k,m,n,r}^{2} \frac{e^{-\lambda_k t}\tilde{\Gamma}_{nm}}{\lambda_r + \lambda_k} g(1,1,k,m,n,r,h,\alpha,B). \tag{7.10}$$

So for a given pair (B, Γ), a candidate pair $(\tilde{B}, \tilde{\Gamma})$ with the same first marginal distribution should satisfy for each $k = 1, 2$,

$$\sum_{m,n,r}^{2} \frac{1}{\lambda_r + \lambda_k} \left\{ \tilde{\Gamma}_{nm} g(1,1,k,m,n,r,h,\alpha,B) - \Gamma_{nm} g(1,1,k,m,n,r,0,b_{21},B) \right\} = 0, \tag{7.11}$$

where $\tilde{\Gamma}_{11} = \Gamma_{11}$, due to the quadratic variation identity.

Solutions to (7.11) with respect to either α, h or $\tilde{\Sigma}$ can easily be found using a computer with the following approach.

For notational simplicity, define $P = (v_1, v_2)^T$ where $v_1 = (1, x_1)^T$ and $v_2 = (x_2, 1)^T$, and let $\Gamma := \Sigma\Sigma^T = \begin{pmatrix} c_1 & c_2 \\ c_2 & c_3 \end{pmatrix}$. We compute

$$\begin{aligned}
&\mathrm{Cov}(V_s^{(1)}, V_{s+t}^{(1)}) \\
&= \begin{pmatrix} 1 & 0 \end{pmatrix} P \int_0^\infty e^{-Ds} P^{-1} \Gamma \left(P^{-1}\right)^T e^{-Ds}\, ds\; e^{-Dt} P^T \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\
&= (1 - x_1 x_2)^{-2} \left[\left(\frac{c_1 - 2x_2 c_2 + x_2^2 c_3}{2\lambda_1} + \frac{x_2 c_2 - x_1 x_2 c_1 + x_1 x_2^2 c_2 - x_2^2 c_3}{\lambda_1 + \lambda_2} \right) e^{-\lambda_1 t} + \right. \\
&\left. \left(\frac{x_1^2 x_2^2 c_1 - 2x_1 x_2^2 c_2 + x_2^2 c_3}{2\lambda_2} + \frac{x_2 c_2 - x_1 x_2 c_1 + x_1 x_2^2 c_2 - x_2^2 c_3}{\lambda_1 + \lambda_2} \right) e^{-\lambda_2 t} \right].
\end{aligned}$$

A candidate $\tilde{B}$ would lead to the same expression with x_1, x_2 exchanged with $\tilde{x}_1$ and $\tilde{x}_2$, so we obtain two equations:

$$\begin{aligned}
&(1 - x_1 x_2)^{-2} \left[(\lambda_1 + \lambda_2) \left(c_1 - 2x_2 c_2 + x_2^2 c_3 \right) + 2\lambda_1 \left(x_2 c_2 - x_1 x_2 c_1 + x_1 x_2^2 c_2 - x_2^2 c_3 \right) \right], \\
&= (1 - \tilde{x}_1 \tilde{x}_2)^{-2} \left[(\lambda_1 + \lambda_2) \left(c_1 - 2\tilde{x}_2 c_2 + \tilde{x}_2^2 c_3 \right) + 2\lambda_1 \left(\tilde{x}_2 c_2 - \tilde{x}_1 \tilde{x}_2 c_1 + \tilde{x}_1 \tilde{x}_2^2 c_2 - \tilde{x}_2^2 c_3 \right) \right]
\end{aligned} \tag{7.12}$$

and

$$\begin{aligned}
&(1 - x_1 x_2)^{-2} \left[(\lambda_1 + \lambda_2) \left(x_1^2 x_2^2 c_1 - 2x_1 x_2^2 c_2 + x_2^2 c_3 \right) + 2\lambda_2 \left(x_2 c_2 - x_1 x_2 c_1 + x_1 x_2^2 c_2 - x_2^2 c_3 \right) \right] \\
&= (1 - \tilde{x}_1 \tilde{x}_2)^{-2} \left[(\lambda_1 + \lambda_2) \left(\tilde{x}_1^2 \tilde{x}_2^2 c_1 - 2\tilde{x}_1 \tilde{x}_2^2 c_2 + \tilde{x}_2^2 c_3 \right) + 2\lambda_2 \left(\tilde{x}_2 c_2 - \tilde{x}_1 \tilde{x}_2 c_1 + \tilde{x}_1 \tilde{x}_2^2 c_2 - \tilde{x}_2^2 c_3 \right) \right],
\end{aligned} \tag{7.13}$$

where

$$\tilde{x}_1 = \frac{b_{12} x_1 - h}{\tilde{b}_{12}}, \quad \tilde{x}_2 = \frac{b_{21} x_2 + h}{\alpha}, \tag{7.14}$$

and

$$\tilde{b}_{12} = \frac{b_{12} b_{21} - h(b_{11} - b_{22}) - h^2}{\alpha}. \tag{7.15}$$

Substituting (7.14) and (7.15) into (7.12) and (7.13), we obtain two equations in $\tilde{b}_{12}, \tilde{b}_{21}, c_{12}, c_{22}$ and h.

For fixed α and h we have two equations with two unknowns, $\tilde{\Gamma}_{21} = \tilde{\Gamma}_{12}$ and $\tilde{\Gamma}_{22}$. In Jacobsen (2011) $\tilde{\Gamma}_{12}$ and $\tilde{\Gamma}_{22}$, with $\tilde{\Gamma}_{11} = 1$, was characterized for the diffusion with diagonal B and unit Γ. It turns out that the solutions $(\tilde{\Gamma}_{12}, \tilde{\Gamma}_{22})$ span a one-dimensional affine subspace of $\mathbb{R}^2$. As a special case, the situation where

$$\tilde{B} = \begin{pmatrix} \lambda_1 & \delta \\ 0 & \lambda_2 \end{pmatrix},$$

was considered and the solution subspace was given as

$$\tilde{\Gamma}_{12} = \frac{\delta}{2(\lambda_1 - b_{22})} \left(\frac{\lambda_1}{\lambda_2} - 1 \right) \tilde{\Gamma}_{22}, \tag{7.16}$$

Note that this corresponds to the situation from (7.6), though $\tilde{B}_{12} = 0$ and the function g is not yet defined in this case. One can find

$$P_{1r}(P^{-1})_{rn} P_{1k}(P^{-1})_{km} = \left(\frac{\delta}{\lambda_2 - b_{11} - h} \right)^{n+m-2} 1_{(r \leq n)} 1_{(k \leq m)}, \tag{7.17}$$

and verify the claim (7.16) by rearranging (7.11), where the first g is substituted by (7.17).

Example 7.3. Let $\Sigma = \tilde{\Sigma} = \operatorname{diag}((2,1))$ and

$$B = \begin{pmatrix} 3 & 2 \\ 1 & 4 \end{pmatrix}.$$

Tedious computations or a computer program like Maple, can show that the auto covariance function for (V_t) is

$$\begin{aligned} \operatorname{Cov}(V_0, V_t) &= \begin{pmatrix} \frac{16}{105} e^{-5t} + \frac{13}{21} e^{-2t} & \frac{16}{105} e^{-5t} - \frac{13}{42} e^{-2t} \\ \frac{2}{35} e^{-5t} - \frac{3}{14} e^{-2t} & \frac{2}{35} e^{-5t} + \frac{3}{28} e^{-2t} \end{pmatrix} \\ &\approx \begin{pmatrix} 0.152 e^{-5t} + 0.619 e^{-2t} & 0.152 e^{-5t} - 0.310 e^{-2t} \\ 0.057 e^{-5t} - 0.214 e^{-2t} & 0.057 e^{-5t} + 0.107 e^{-2t} \end{pmatrix}, \end{aligned}$$

with

$$\operatorname{Cov}(V_0, V_0) = \begin{pmatrix} \frac{27}{35} & -\frac{11}{70} \\ -\frac{11}{70} & \frac{23}{140} \end{pmatrix} \approx \begin{pmatrix} 0.771 & -0.157 \\ -0.157 & 0.164 \end{pmatrix}.$$

Furthermore, with h equal to

$$h = \frac{1}{3} \left(377 + 12\sqrt{987} \right)^{1/3} + \frac{1}{3 \left(377 + 12\sqrt{987} \right)^{1/3}} + \frac{2}{3} \approx 3.737,$$

we get that

$$\tilde{B} \approx \begin{pmatrix} 6.737 & -8.229 \\ 1 & 0.263 \end{pmatrix},$$

with exact values that are possible to compute. The $\tilde{B}$ matrix induces a new process $(\tilde{V}_t)$ with covariance function

$$\operatorname{Cov}(\tilde{V}_0, \tilde{V}_t) \approx \begin{pmatrix} 0.152 e^{-5t} + 0.619 e^{-2t} & 0.032 e^{-5t} + 0.356 e^{-2t} \\ -0.550 e^{-5t} + 0.939 e^{-2t} & -0.116 e^{-5t} + 0.540 e^{-2t} \end{pmatrix},$$

with

$$\operatorname{Cov}(\tilde{V}_0, \tilde{V}_0) \approx \begin{pmatrix} 0.771 & 0.389 \\ 0.389 & 0.424 \end{pmatrix}.$$

See figure 7.1 where both V_t and $\tilde{V}_t$ are simulated for 400 observations, showing both scatter plot, time plot and density plots.

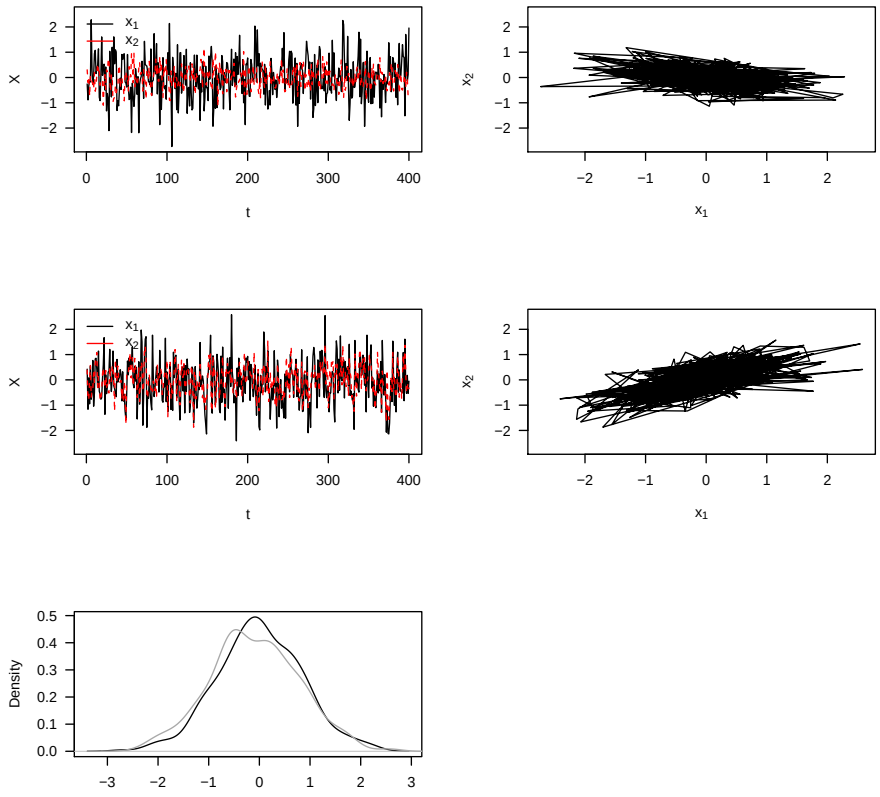

Figure 7.1: *Top: Simulations from* (V_t)*. Middle: Simulations from* $(\tilde{V}_t)$*. Bottom: Density plots of first marginal for* (V_t) *(black) and* $(\tilde{V}_t)$ *(gray).*

In conclusion one should always be careful when it comes to parameter estimation in models with latent components, because the natural model formulation, corresponding to the fully observed case might contain parameters that are no identifiable.

8

Computer implementation

In this chapter we focus on the implementational aspects of the statistical methods described in chapter 5 and 6. In general, implementation of computer intensive methods can be a highly time consuming task for several reasons. For instance, one has to choose how to structure the program and decide which programming language to use for the implementation. The choice of language should in principle entail a good relation between ease of implementation, speed, and perhaps also user friendliness. There are many options when it comes to choosing the right language for statistical computing, including `R`, `(WIN/open)BUGS`, `Python`, `JAVA`, `julia`, `C++` and many more, and they all have their own strengths and drawbacks. Some are more difficult to debug than others, and some are specialized in accomplishing specific types of operations such as linear algebra. It is therefore important to consider if the task is to be carried out only once (or a small number of times) relative to the amount of time it takes to do the implementation (including the unavoidable and time consuming task of debugging). For example, implementing a Gibbs sampler requires one to perform calculations based on an iterative procedure and thus lends itself toward a loop statement. In our setting, each step in the loop is relatively time consuming implying that the language should be fast. Initially, part of the algorithm was coded directly in `R`, leading to a very slow program. After switching to `C++` the speed up was between a factor 20 and 40.

Another important aspect is whether the final software should be portable to other computers with different compilers and/or operating systems. Additionally, one must also be familiar with the estimation procedure and understand the potential limitations or restrictions of the methodology. For instance, it does not always make sense to estimate all parameters in a partially observed model even though the software does not produce an error cf. example 7.2. This is of course a design issue, which could be solved with a thorough number of sanity checks, but it is easy to make a design failing to satisfy all reasonable checks.

Here we focus on the choice made in relation to this thesis, namely a combination of `C++` and `R`. The former has the advantage of being very fast and capable of handling large data structures, whereas the latter one is excellent for graphics. Also `R` is an interpreted language, making debugging and direct manipulation of objects easier than in `C++`. Furthermore `R` directly supports all the (graphical) tools needed to evaluate results from the MCMC routine, and it is available across different platforms (Windows, Linux, Mac) for others to use. In relation to this thesis is developed a piece of software which is collected in an `R` package. The package is available on `CRAN` (the official `R`-library repository) and can also be found on `www.math.ku.dk/~anders`.

8.1 Developing `R`-packages in Windows

This section is a brief description on how to create and maintain an `R` package with included `C++` code on a Windows computer. It is targeted at people with little or no experience in this matter, hence we will not go far into the technical details. There are many different ways to accomplish the task, and in the following is outlined one specific solution. There

are many others. For detailed help the main reference is 'Writing R Extensions' (R Core Team, 2013).

A nice feature of R is the option to create a package containing all functionality related to a certain task, and the ease of which one can share this software to other computers and operating systems. Furthermore, the R language is a very nice tool for statistical inference, although it has its limitations, one of them being the lack of speed for certain tasks. The general progress in computational power within the last few decades, has made implementation of highly computer intensive methods possible. However this progress comes with a few challenges. For a person with programming knowledge limited to basic use of R, it can be a challenge to start developing an R package, though it need not be. The optimal development work flow is not entirely obvious, because all changes to the package has to be compiled before use, and this is different from the behavior of a standard interactive R session. For one, it makes debugging more cumbersome. On top of this comes the need to include code from another programming language which one needs to be familiar with.

Since the methods from chapter 5 and 6 are multivariate, a natural way to structure the implementation is via matrix operations. The C++ Armadillo library (Sanderson, 2010) supports matrix operations and has syntax that is somewhat familiar to R users. In order to establish the connection between R and C++, including the Armadillo library, we use the two R packages Rcpp (Eddelbuettel and Francois, 2011) and RcppArmadillo (Eddelbuettel and Sanderson, 2014), which are designed to facilitate the gap between C++ and R.

8.1.1 Preliminaries

Some of the essential tools for building R packages are not pre-installed in a default Windows setup. This can be accomplished by downloading and installing Rtools which is a freely available collection of files that makes it possible to create and compile R-packages. It includes a compiler (MinGW), Perl and some UNIX command line tools. Rtools must be installed in C:/Rtools which will require Windows administrator rights. Next, one must ensure that the following lines are included in the search path:

```
 c:\Rtools\bin;
c:\Rtools\perl\bin;
c:\Rtools\MinGW\bin;
c:\progra~1\R\R_VERS\R\bin\i386
```

where the last path must be modified according to the location of R, and i386 should be interchanged with x64 if one prefer the 64 bit version of R instead of 32 bit. In this way, R is able to locate all the files needed to use Rtools. It is also recommended to have a working installation of latex, but this is only to create the help files. The final preparation step is to install the R packages Rcpp and RcppArmadillo. To ease the process of updating a package, it is also recommended to install devtools which is designed to do exactly this.

8.1.2 Creating and building the package

An R package consists of a collection of files and folders ordered in a specific way. This file structure can be generated automatically and the standard way to do this is to use the R function `package.skeleton`. However, if the package should contain C++ code and link to the Armadillo library, it is recommended to use the `RcppArmadillo.package.skeleton` instead. This function automatically creates a package folder with three sub-folders, src, R and man, which are more or less self explanatory: The man folder is for documentation files for the functions in the package, the R folder is for the actual R code, and the src folder holds the C++ source code. Three example files are automatically generated: The `rcpparma_hello_world.R` function which calls `rcpparma_hello_world.cpp` which in turn calls `rcpparma_hello_world.h`. By examining and modifying these files, basic functionality can be obtained. Note that the R function use the R function `.Call` to call a C++ function.

The package folder contains three files: A read-and-delete-me file, a DESCRIPTION file with details about the package version, author etc., and NAMESPACE which defines which of the package functions are made available to the R user. When using the `RcppArmadillo.package.skeleton` the default is that all functions starting with a letter becomes available. Thus to hide a function, which is only for internal use, one may explicitly ensure that it is not included or hide it by starting the function name with a period. If the package use functionality from other R packages, it should be specified in the DESCRIPTION file. Previously one had to manually edit the DESCRIPTION file and the src/MAKEVARS to specify R library dependencies and locations of certain C++ libraries. This procedure is now fully automatized: The `RcppArmadillo.package.skeleton` automatically links to `Rcpp` and `RcppArmadillo` and ensures that R is able to find the `BLAS` and `LAPACK` libraries which are necessary for some C++ functionality.

Next step is to create the actual R files that should be part of the package. This can be done directly in `RcppArmadillo.package.skeleton` via the `list` argument: Start a clean R session and load exactly the code that should be in the package. List the objects to include in a character vector as in Listing 8.1.

Listing 8.1: Example for creating a minimal R package

```
library(RcppArmadillo)
a.fun <- function(x){x^2}
rcppArmadillo.package.skeleton(name="NameOfPackage",list=c("a.fun"))
```

This will create the folder structure for the package. Additional arguments can be given; see the help page. One should also remember to update and edit the documentation files in the /man directory.

When additional functionality is to be included the package must be completely unloaded first, then compiled and re-installed. After closing R, this is done from the command line using

```
R CMD INSTALL --build "PathToPackage/NameOfPackage"
```

This will create and install a zip file containing the package for use on Windows. For optional arguments to `R CMD INSTALL` see

```
R CMD INSTALL --help
```

This is however, not a fast way to debug or test new code. The `R` package `devtools` can help to perform these steps automatically using the function `load_all`. Updating package version and date must be done manually in the two files 'DESCRIPTION' and 'man/NameOfPackage-package.Rd'.

8.2 The BIPOD-package

The `BIPOD` (Bayesian Inference for Partially Observed Diffusions) package implements the Bayesian methods from chapter 5 and 6 in order to make parameter inference for diffusion models. The implementation is relatively computer intensive and in order to keep computational time at a minimum we take advantage of `C++` which runs the MCMC procedure considerably faster than `R` itself is able to do. As a positive side effect there is also access to functions that can do fast simulations of 1 or 2-dimensional versions of the Ornstein-Uhlenbeck process, the Brownian bridge and three additional processes via the Euler-Maruyama scheme. All computational heavy code is written in `C++` while the `R` functions mainly work as wrappers.

The package contains five functions and full details can be found in the manual in appendix 10.A. The `ShowModels` function prints a given model description to the screen. This is mainly relevant when specifying a list of possible known parameters that should not be estimated. The result is printed to the screen showing how the parameters are ordered and how many drift and diffusion parameters one may use. For the current implementation the diffusion matrix must be diagonal. See Listing 8.2 for an example.

Listing 8.2: Application of the ShowModels functions

```
> ShowModels("FHN5")
$Model
     [,1]
[1,] "dX_t = ( - drift1 * X_t^3 + drift2 * ( X_t - Y_t ) + drift3 ) dt + diff1
    dB1_t"
[2,] "dY_t = drift4 * X_t - Y_t + drift5 ) dt + diff2 dB2_t"

$Ndrift
[1] 5

$Ndiff
[1] 2
```

The `BBSim` simulates Brownian bridges and `DiffSim` simulates two-dimensional diffusion processes. Both functions are faster than a naive `R`-implementation as they rely on `C++`-code. The following code generates an Euler-Maruyama approximation to a sample path from the FitzHugh-Nagumo model starting in $(1, 1)$. The time step is 0.001 but the output is thinned such that the actual time step is 0.1. The code is given in Listing (8.3). For the Ornstein-Uhlenbeck data is simulated from the exact distribution.

Listing 8.3: Simulation of data from the FitzHugh-Nagumo model

```
DiffSim(n      = 10000,
        start  = c(1,1),
        Delta  = .001,
        driftpar = c(10,5,1.5,0.6),
        Sigma  = diag(c(.5,.3)),
        thin   = 100,
        Model  = "FHN")
```

The main function is `Estfun` and it runs a Gibbs sampler to estimate parameters in a discretely observed two-dimensional diffusion where one coordinate potentially is unobserved. The current implementation handles the following models:

- The 2 dimensional Ornstein-Uhlenbeck process
- The stochastic FitzHugh-Nagumo model
- The extended stochastic FitzHugh-Nagumo model

The `Estfun` function takes 19 arguments. The first is `data` which takes the observed data. If given as a vector it is treated as the first coordinate of the process with the second coordinate latent, and if it is given as a two column matrix both coordinates are treated as fully observed. The next argument `Delta` is needed in order to know the time step between observations. The number of imputed data points is given by `ImputeN`, and `GibbsN` specifies the number of iterations of the Gibbs sampler. The `parKnown` argument is a named list with the parameters (if any) that are fixed and names can be found via the `ShowModels` function. The arguments `Start` and `LatentPathStart` are the starting values for the parameters of the Gibbs sampler and the latent path, respectively. Thus `LatentpathStart` should be a vector as long as `data` or a single number. In the latter case the path starts as the horisontal line through `LatentpathStart`.

Prior distributions on the parameters are specified separately for drift and diffusion. The drift has three options: It can be Gaussian and sampled directly, because the posterior is also Gaussian. It can be sampled using an MH step or the prior can be improper such that sampling is performed as described in (5.15). The prior of the diffusion parameters is assumed to be a two-dimensional Gaussian distribution with mean `diffPriorMean` and covariance `DiffPriorCovar`, while `DiffRW` is the covariance matrix of the random walk which updates the diffusion parameters. The `Model` argument specifies which model to use for the estimation procedure.

If the second coordinate is unobserved, the `LatentMeanY0` and `LatentVarY0` give the prior Gaussian distribution on the latent second coordinate at time 0. Finally `RWrhoPaths` and `RWrho2PathPoints` are values between 0 and 1 controlling the update of the latent path. See (6.7).

The result is an object of class `BIPOD`, which is a list with several entries, including output from the Gibbs sampler. The output related to the parameters are in $Drift and $Diff respectively. For general information about computation time and parameters given to the function, see $Info.

The package also supports some simple graphical aspects of the output. The `plot.BIPOD` function is a wrapper function for several standard plot functions. Its primary argument is `type` which is one of (trace, hist,acp, pairs,SDtrace, accept, movie,cover). It's use is shown in Listing (8.4), where we start by simulating a data set and next use the `Estfun` for for estimation.

Listing 8.4: Parameter estimation for the FitzHugh-Nagumo model

```
Data <- DiffSim(n      = 10000,
                start  = c(0,0),
                Delta  = .001,
                driftpar = c(10,5,1.5,.6),
                Sigma  = diag(c(.5,.3)),
                seed   = 1,
                thin   = 100,
                Model  = "FHN")

Result <- Estfun(data   = Data[,1],
                 Delta    = .001*100,
                 ImputeN  = 20,
                 seed     = 1,
                 GibbsN   = 10000,
                 parKnown = list(
                     "drift3" = 1.5,
                     "drift4" = .6,
                     "diff2" = diffPar[2,2]
                     ),
                 Start    = c(1,10,1,10,1,1),
                 diffPriorMean = c(log(.5)+2,log(.3)+1),
                 diffPriorCovar= diag(c(5,5)),
                 driftPriorMean = NULL,
                 driftPriorCovar = NULL,
                 driftRW = NULL,
                 diffRW = diag(c(.03,.0075)),
                 LatentPathStart = 1,
                 RWrho2PathPoints=.5,
                 RWrhoPaths = .5,
                 Model="FHN",
                 LatentMeanY0 = .2,
                 LatentVarY0 = 10)
```

```
plot(Result,type="trace",theta=c(10,5,1.5,.6,.5,.3))
plot(Result,type="hist",theta=c(10,5,1.5,.6,.5,.3),subset=c(1,2,5),interval=100)
plot(Result,type="movie",truepath=Data[,2])
```

9 Outlook

In this thesis the primary focus has been parameter estimation for multivariate diffusions. We have given an introduction to some of the useful tools needed in order to perform parameter inference in such models, both from a frequentistic and a Bayesian point of view. Special attention has been given to the stochastic FitzHugh-Nagumo model and the two-dimensional Ornstein-Uhlenbeck model and for the latter model we investigated identifiability issues related to the situation where one coordinate is completely unobserved. Identifiability problems arise in many context, and it is not an easy problem to solve for diffusions in general, because explicit expressions for the transition density are typically unknown. The approach we have taken for the Ornstein-Uhlenbeck does not apply directly to other models, and different approaches must be pursued. In the same setup with the Ornstein-Uhlenbeck process we also applied the method of prediction-based estimating functions to estimate some of the parameters in the model. The results were not so convincing, and it would require a certain amount of work, to use this approach in other models.

Taking a different approach to parameter estimation for both the FitzHugh-Nagumo model, the Ornstein-Uhlenbeck and the extended Ornstein-Uhlenbeck, we used Bayesian methods to perform parameter inference. For all three models we have implemented a computer intensive method for parameter estimation and collected everything in an `R` package. The implemented method can be expanded to work for a larger class of diffusion models. The ultimate aim is to be able to feed the computer any diffusion function, any drift function and a data set, and then perform parameter estimation. One can come a long way using `R` where it is easy to pass functions as arguments. It is a little more difficult in `C++` if the code must be compiled beforehand as is the case in an `R` package. Using the current methodology it would also require that the computer could invert the diffusion function, or alternatively, that one could specify it manually.

10 Appendix

10.A BIPOD manual

Package 'BIPOD'

February 3, 2014

Type Package

Title BIPOD (Bayesian Inference for Partially Observed diffusions)

Version 0.2.0

Date 2014-01-26

Author Anders Chr. Jensen

Maintainer Anders Chr. Jensen <anders@math.ku.dk>

Description Bayesian parameter estimation for (partially observed) two-dimensional diffusions

License GPL-3

Depends Rcpp (>= 0.10.6), RcppArmadillo (>= 0.4.000)

LinkingTo Rcpp, RcppArmadillo

R topics documented:

`BIPOD-package` *Bayesian parameter estimation in two-dimensional diffusion models with affine drift.*

Description

This package use data augmentation and a Gibbs sampler to sample from the joint posterior of parameters and augmented data. The main function is 'Estfun' which produce an object of class 'BIPOD'. See the help page for 'Estfun' for an example.

Details

Package:	BIPOD
Type:	Package
Version:	0.2.0
Date:	2014-01-26
License:	GPL-3

BBSim DiffSim Estfun plot.BIPOD ShowModels

Author(s)

Anders Chr. Jensen

Maintainer: Anders Chr. Jensen <anders@math.ku.dk>

References

'Importance sampling techniques for estimation of diffusion models' by Papaspiliopoulos and Roberts, 'Statistical Methods for Stochastic Differential Equations', Monographs on Statistics and Applied Probability, Chapman and Hall, 2012 and 'Markov chain Monte Carlo approach to parameter estimation in the FitzHugh-Nagumo model' by Jensen et al, Phys. Rev. E 2012.

`BBSim` *Function for simulation of p dimensional Brownian bridge*

Description

Simulation of p-dimensional driftless SDE with constant diffusion, conditional on end points: dV_t=Sigma dW_t, conditional on V_0 and V_T. This function makes a call to C++ and it is therefore relatively fast.

Usage

```
BBSim(start, end, n, Sigma=diag(2), T, t0 = 0,seed = 1)
```

Arguments

`start`	Numerical vector of length p: Starting point for the process
`end`	Numerical vector of length p: Ending point for the process
`n`	Positive integer: Number of time points where the process is simulated
`Sigma`	p*p matrix: The diffusion matrix for the process
`T`	Positive number: End of time interval.
`t0`	Non negative number, defaults to 0. Start of time interval.
`seed`	Integer, defaults to 1. Specifies seed for random generator. If <=0 it is set randomly.

Details

An n*p matrix with columns representing simulations for each coordinate.

Value

An n*p matrix

Examples

```
(tmp <- BBSim(start = c(1,2),
              end   = c(3,5),
              n     = 10,
              Sigma = diag(2),
              T     = 2,
              t0    = 0,
              seed  = 1))
matplot(tmp,type="l")
```

`DiffSim` *Simulation of a 2-dimensional diffusion process. See the Model argument for options.*

Description

Function for simulation of 2-dimensional diffusion processes, using the Euler-maruyama scheme.

Usage

```
DiffSim(n, start, Delta, driftpar, Sigma, seed=NULL, thin=1, Model)
```

Arguments

`n`	positive integer: Length of simulation.
`start`	Numerical vector: Starting point for the simulation.
`Delta`	Numerical: Time interval between observations.
`driftpar`	Numerical vector. Parameters of the FitzHugh-Nagumo model.
`Sigma`	2*2 diffusion matrix.
`seed`	Integer: Gives the seed for the random number generator.
`thin`	Integer: Output only every 'thin' simulation.
`Model`	Character specifying the model. Currently one of 'OU', 'FHN', 'FHN5' and 'CIR'.

Value

An (n/thin) by 2 matrix.

Examples

```
FH <- DiffSim(n       = 10000,
              start   = c(1,1),
              Delta   = .001,
              driftpar = c(10,0.6,1.5,0.0),
              Sigma   = diag(c(.5,.3)),
              seed    = 1,
              thin    = 100,
              Model   = "FHN")
matplot(FH,type="l")
```

Estfun *Parameter estimation for some two dimensional diffusions.*

Description

Applies a Gibbs sampler to parameters and augmented data for two-dimensional stochastic differential equations. Currently the Ornstein-Uhlenbeck, the stochastic FitzHugh-Nagumo model and the extended FitzHugh-Nagumo model are implemented.

Usage

```
Estfun(data, Delta, ImputeN = 5, seed, GibbsN = 1000,
       parKnown = list(), Start = c(0,0,0,0,1,1), diffPriorMean,
       diffPriorCovar, diffRW=diag(2), LatentPathStart,Model = NULL,
       driftPriorMean, driftPriorCovar, driftRW, LatentMeanY0 = 0,
       LatentVarY0 = 1, RWrhoPaths = 1, RWrho2PathPoints = 1)
```

Arguments

data	Data to estimate parameters from. Matrix or numeric. Dimensions must be n*1 or n*2 depending on whether second coordinate is observed.
Delta	Positive numeric: Time between observations.
ImputeN	Positive integer>=3: M-2 is the number of imputed data points between consecutive observed data.
seed	Positive integer giving the seed for the random number generator. Defaults to random.
GibbsN	Positive integer: Number of iterations of the Gibbs sampler.
parKnown	List of named values for the known drift and diffusion parameters.
Start	Numerical vector with starting values for the drift and diffusion parameters in the Gibbs sampler.
diffPriorMean	numerical vector of length 2. Prior mean for diffusion coefficients
diffPriorCovar	2*2 matrix. Prior variance for diffusion coefficients.
diffRW	Random walk variance for the MH step for the diffusion ocefficients.
LatentPathStart	Numeric of length one or same length as Data. Starting value for the latent path. If LatentPathStart is a single number then all starting values take this value.

`Model`	Charater, specifying the model. Currently the only options are 'OU','FHN' and 'FHN5'.
`driftPriorMean`	prior mean for the drift parameters
`driftPriorCovar`	Prior covariance for the drift parameters
`driftRW`	Covariance matrix for the RW update of the drift parameters
`LatentMeanY0`	Prior mean for the first data point of the unobserved coordinate.
`LatentVarY0`	Prior variance for the first data point of the unobserved coordinate. If 0, the point is fixed at first value of LatentPathStart.
`RWrhoPaths`	Numeric in [0,1]. Parameter for random walk update of the latent path between observation times. The value 0 samples a BB, the value 1 keeps the current value of the (skeleton) path
`RWrho2PathPoints`	Parameter for random walk update of the latent coordinate at observation times. The value 0 samples a middle point of a BB, the value 1 keeps the current value of the points

Details

More details for the help page will be added soon.

Value

An object of class BIPOD.

`Drift`	Output of the Gibbs sampler for the drift parameters.
`Diff`	Output of the Gibbs sampler for the diffusion parameters.
`AccRate1`	Accept/reject (1/0) for each path interval and each iteration of the sampler.
`AccRate2`	Accept/reject (1/0) for each path endpoint of the latent coordinate and each iteration of the sampler. Only valid if second coordinate is latent.
`LatentPath`	Output of the Gibbs sampler for the endpoints of the latent path. Only valid when one coordinate is observed.
`diffAcc`	Accept/reject (1/0) for the MH step of the diffusion coefficient.
`Info`	List with information about the estimated model.
`driftPriormu`	Prior mean of the drift parameters.
`driftPriorOmega`	Prior variance in the drift parameters.
`driftRW`	Random Walk variance for updating drift parameters.

Author(s)

Anders Chr. Jensen

Examples

```
Data <- DiffSim(n=10000,
                start=c(0,0),
                Delta=.001,
                driftpar=c(10,5,1.5,.6),
                Sigma=diag(c(.5,.3)),
                seed=1,
                thin=100,
                Model="FHN")

A <- Estfun(data    = Data[,1],
            Delta   = .001*100,
            ImputeN = 10,
            seed    = 2,
            GibbsN  = 2000,
            parKnown = list("drift3"=1.5,"drift4"=.6,"diff2"=.3),
            Start=c(10,10,10,10,1,1),
            diffPriorMean= c(0,0),
            diffPriorCovar= diag(2),
            diffRW = diag(c(.01,.02)),
            LatentPathStart = .5,
            Model="FHN",
            driftPriorMean = NULL,
            driftPriorCovar = NULL,
            driftRW = diag(4),
            LatentMeanY0 = 0,
            LatentVarY0 = 1,
            RWrhoPaths = 0,
            RWrho2PathPoints = 0)

class(A);names(A)
plot(A,type="trace",interval=1,theta=c(10,5,1.5,.6,.5,.3),subset=c(1,2,5))
### plot(A,type="movie",truepath=Data[,2],speed=.01,BY=10,interval=1)
```

---

plot.BIPOD *Plot function for class BIPOD*

Description

Graphical summaries of output from Gibbs sampler

Usage

```
## S3 method for class 'BIPOD'
plot(x, theta = NULL, subset = NULL, type, lag = 20,
       interval = NULL, treshold = 0.1, speed = 0.1, truepath = NULL,
       BY = 1, prop = c(.05,.95), diffPriorMean = NULL,
       diffPriorCovar = NULL, log = FALSE, ...)
```

Arguments

x	x: An object of class BIPOD.
theta	Optional numerical vector with true parameter values.

subset	Numeric vector. Which parameters should be used for plotting. Defaults to all.
type	Character choosing plotting type: Either "trace", "hist", "acp", "pairs","SDtrace", "accept", "movie" or "cover". See details.
lag	Positive integer: Number of lags used in autocorrelation plot
interval	Positive integer or numericla vector: If integer, used as burn in for the Gibbs sampler. If vector, used to subsample Gibbs output. Defaults to no subsampling and no burn in.
treshold	Positive numeric: Cut off for acceptance rate.
speed	Positive number used for type="movie": How much time to pause between each frame of the movie?
truepath	Vector or matrix with latent data. Optional.
BY	integer: Only relevant for type="movie". How many frames to skip for each iteration?
prop	Numeric vector with values between 0 and 1: Only relevant for type="cover". Specifies quantiles for the sampled paths
diffPriorMean	Numeric of length 2, only for type=="hist". The prior mean for the diffusion coefficients.
diffPriorCovar	2*2 matrix, only for type=="hist". The prior covariance for the diffusion coefficients.
log	Boolean, only for type=="hist". If TRUE, the prior density and the estimate of the diffusion coefficients are log transformed before plotting.
...	Additional arguments to be passed to matplot, density or acf.

Details

Different 'type'-argument gives different plots. More details to come...

`ShowModels` *Prints form of supported stochastic differential equations*

Description

Print function displaying the model structures currently supported. Used to fix the parametrization of the parameters.

Usage

```
ShowModels(Model)
```

Arguments

Model	Character specifying the model. Current options are "OU" for the Ornstein Uhlenbeck process, "FHN" for the stochastic FitzHugh-Nagumo process "FHN5" for the extended FitzHugh-Nagumo model and "CIR" for the Cox-Ingersoll-Ross model.

Details

This function is used to identity the parameter names in the supported models. This is necessary when specifying the "parKnown" argument in the "Estfun" function.

Value

List with three entries:

`Model`	A 2*1 Matrix with character entries.
`Ndrift`	Numeric giving the number of drift parameters
`Ndiff`	Numeric giving the number of diffusion parameters

Examples

```
ShowModels(Model="FHN")
```

Index

Bibliography

Ait-Sahalia, Y. (2002). Maximum likelihood estimation of discretely sampled diffusions: a closed-form approximation approach. *Econometrica*, 70(1):223–262. 2

Ait-Sahalia, Y. (2008). Closed-form likelihood expansions for multivariate diffusions. *The annals of statistics*, 36(2):906–937. 2, 10

Berglund, N. and Gentz, B. (2006). *Noise-induced phenomena in slow-fast dynamical systems*. Probability and its Applications (New York). Springer-Verlag London Ltd., London. A sample-paths approach. 2, 50, 51

Beskos, A., Papaspiliopoulos, O., and Roberts, G. (2009). Monte Carlo maximum likelihood estimation for discretely observed diffusion porcesses. *The Annals of Statistics*, pages 223–245. 2

Beskos, A., Papaspiliopoulos, O., and Roberts, G. O. (2006). Retrospective exact simulation of diffusion sample paths with applications. *Bernoulli*, 12(6):1077–1098. 51, 53

Bladt, M. and Sørensen, M. (2014). Simple simulation of diffusion bridges with application to likelihood inference for diffusions. To appear in Bernoulli. 53

Casella, G. and George, E. I. (1992). Explaining the gibbs sampler. *The American Statistician*, 46(3):167–174. 45

Chib, S. and Greenberg, E. (1995). Understanding the Metropolis-Hastings Algorithm. *The American Statistician*, 49(4):327–335. 42

Ditlevsen, S. and Greenwood, P. (2013). The Morris-Lecar neuron model embeds a leaky integrate-and-fire model. *Journal of Mathematical Biology*, 67(2):239–259. 50

Durham, G. B. and Gallant, A. R. (2002). Numerical Techniques for Maximum Likelihood Estimation of Continuous-Time Diffusion Processes. *Journal of Business & Economic Statistics*, 20(3):pp. 297–316. 2

Eddelbuettel, D. and Francois, R. (2011). Rcpp: Seamless `R` and `C++` Integration. *Journal of Statistical Software*, 40(8):1–18. 95

Eddelbuettel, D. and Sanderson, C. (2014). RcppArmadillo: Accelerating `R` with high-performance `C++` linear algebra. *Computational Statistics and Data Analysis*, 71:1054–1063. 95

Elerian, O., Chib, S., and Shephard, N. (2001). Likelihood Inference for Discretely Observed Nonlinear Diffusions. *Econometrica*, 69(4):959–993. 2

FitzHugh, R. (1961). Impulses and physiological states in theoretical models of nerve membrane. *Biophys. J.*, 1:445–466. 15, 50

Geman, S. and Geman, D. (1984). Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Trans. Pattern Anal. Machine Intell*, 6:721–741. 45

Gerstner, W. and Kistler, W. (2002). *Spiking Neuron Models*. Cambridge University Press. 15, 16, 50

Golightly, A. and Wilkinson, D. J. (2008). Baysian inference for nonlinear multivariate diffusion models observed with error. *Computational Statistics & Data Analysis*, 52:1674–1693. 2

Hall, P. and Heyde, C. C. (1980). *Martingale limit theory and its application*. Probability and mathematical statistics : a series of monographs and textbooks. Academic Press. 21

Hansen, L. P. (1982). Large sample properties of generalized method of moments estimators. *Econometrica*, 50(4):1029–1054. 20

Hastings, W. K. (1970). Monte Carlo Sampling Methods Using Markov Chains and Their Applications. *Biometrika*, 57(1):97–109. 42

Hindriks, R., Jansen, R., Bijma, F., Mansvelder, H., de Gunst, M., and van der Vaart, A. (2011). Unbiased estimation from time series with application to hippocampal field potentials in vitro. *Physical Review E*, 84. 51

Hodgkin, A. L. and Huxley, A. F. (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117(4):500–544. 50

Huys, Q. J. M., Ahrens, M. B., and Paninski, L. (2006). Efficient estimation of detailed single-neuron models. *J Neurophysiol*, 96(2):872–90. 51

Huys, Q. J. M. and Paninski, L. (2009). Smoothing of, and parameter estimation from, noisy biophysical recordings. *PLoS Comput Biol*, 5(5):e1000379. 51

Izhikevich, E. M. (2007). *Dynamical Systems in Neuroscience*. The MIT Press, Cambridge, Massachusetts. 15, 16

Jacobsen, M. (1991). Homogeneous Gaussian Diffusions in Finite Dimensions. Preprint, Institute of Mathematical Statistics, University of Copenhagen. 7, 14

Jacobsen, M. (2008). Stokastiske integraler. Lecture notes in Danish. 6

Jacobsen, M. (2011). The first marginal in 2-dimensional OU. Personal correspondance. 86, 90

Jensen, A. C., Ditlevsen, S., Kessler, M., and Papaspiliopoulos, O. (2012). Markov chain Monte Carlo approach to parameter estimation in the FitzHugh-Nagumo model. *Phys. Rev. E*, 86:041114. 2, 15, 50

Karatzas, I. and Shreve, S. (1991). *Brownian motion and stochastic calculus*. Springer, 2nd edition. 6

Keener, J. P. and Sneyd, J. (2009). *Mathematical physiology. II. , Systems physiology*. Interdisciplinary applied mathematics. Springer, New York, London. 2, 50

Kessler, M. and Sørensen, M. (1999). Estimating Equations Based on Eigenfunctions for a Discretely Observed Diffusion Process. *Bernoulli*, 5(2):299–314. 22

Kleinhans, D. (2012). Estimation of drift and diffusion functions from time series data: A maximum likelihood framework. *Physical Review E*, 85(2):026705. 51

Kloeden, P. E., Platen, E., and Schurz, H. (2003). *Numerical solution of SDE through computer experiments*. Springer. 13

Lee DeVille, R. E., Vanden-Eijnden, E., and Muratov, C. B. (2005). Two distinct mechanisms of coherence in randomly perturbed dynamical systems. *Phys. Rev. E*, 72:031105. 51

Lindner, B., Ojalvo, G. J., Neiman, A., and Geier, S. L. (2004). Effects of noise in excitable systems. *Physics Reports-Review Section Of Physics Letters*, 392:321–424. 2, 50, 51

Lindner, B. and Schimansky-Geier, L. (1999). Analytical approach to the stochastic FitzHugh-Nagumo system and coherence resonance. *Physical Review E*, 60(6). 51, 61

Lindner, B. and Schimansky-Geier, L. (2000). Coherence and stochastic resonance in a two-state system. *Physical Review E*, 61:6103–6110. 51

Lindström, E. (2012). A regularized bridge sampler for sparsely sampled diffusions. *Statistics and Computing*, 22(2):615–623. 53

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. 42

Meyn, S. and Tweedie, R. (1993). *Markov Chains and Stochastic Stability*. Springer-Verlag. 43

Nagumo, J., Arimoto, S., and Yoshizawa, S. (1962). An Active Pulse Transmission Line Simulating Nerve Axon. *Proceedings of the IRE*, 50(10):2061–2070. 15, 50

Øksendal, B. (2007). *Stochastic Differential equations*. Springer. 8, 11

Papaspiliopoulos, O., Pokern, Y., Roberts, G. O., and Stuart, A. M. (2012). Nonparametric estimation of diffusions: a differential equations approach. *Biometrika*, 99(3):511–531. 51

Papaspiliopoulos, O. and Roberts, G. O. (2012). Importance sampling techniques for estimation of diffusion models. In *Statistical Methods for Stochastic Differential Equations*, pages 311–337. Monographs on Statistics and Applied Probability, Chapman and Hall. 2, 13, 51, 53, 54, 70

Papaspiliopoulos, O., Roberts, G., O., and Stramer, O. (2013). Data Augmentation for Diffusions. *Journal of Computational and Graphical Statistics*, 22(3):665–688. 12, 13

Pokern, Y., Stuart, A. M., and Wiberg, P. (2009). Parameter estimation for partially observed hypoelliptic diffusions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(1):49–73. 66

R Core Team (2013). Writing R Extensions. `http://cran.revolution-computing.com/doc/manuals/R-exts.pdf`. Accessed: 2014-01-03. 95

Robert, C. P. and Casella, G. (2004). *Monte Carlo Statistical Methods*. Springer. 41

Roberts, G. O., Gelman, A., and Gilks, W. R. (1997). Weak convergence and optimal scaling of random walk Metropolis algorithms. *Annals of Applied Probability*, 7:110–120. 44

Roberts, G. O. and Rosenthal, J. S. (2001). Optimal Scaling for Various Metropolis-Hastings Algorithms. *Statistical Science*, 16(4):351–367. 65

Roberts, G. O. and Stramer, O. (2001). On inference for partially observed nonlinear diffusion models using the Metropolis-Hastings algorithm. *Biometrika*, 88(3):603–621. 2, 51, 54, 58, 70

Rogers, L. C. G. and Williams, D. (2000). *Diffusions, Markov processes, and martingales. Vol. 2*. Cambridge Mathematical Library. Cambridge University Press. Itô calculus, Reprint of the second (1994) edition. 6

Samson, A. and Thieullen, M. (2012). A contrast estimator for completely or partially observed hypoelliptic diffusion. *Stochastic Processes and their Applications*, 122(7):2521 – 2552. 66

Sanderson, C. (2010). Armadillo: An Open Source `C++` Linear Algebra Library for Fast Prototyping and Computationally Intensive Experiments. Technical report, NICTA. 95

Sørensen, H. (2004). Parametric Inference for Diffusion Processes Observed at Discrete Points in Time: a Survey. *International Statistical Review*, 72(3):337–354. 2, 51

Sørensen, M. (1999). On Asymptotics of Estimating Functions. *Brazilian Journal of Probability and Statistics*, 13:111–136. 2

Sørensen, M. (2000). Prediction-based estimating functions. *Econometrics Journal*, 3:123–147. 2, 22

Sørensen, M. (2011). Prediction-based estimating functions: Review and new developments. *Brazilian Journal of Probability and Statistics*, 25(3):362–391. 22, 25

Sørensen, M. (2012). Estimating functions for diffusion-type processes. In *Statistical Methods for Stochastic Differential Equations*, pages 1–99. Monographs on Statistics and Applied Probability, Chapman and Hall. 20, 21, 22

Wu, H. and Noé, F. (2011). Bayesian framework for modeling diffusion processes with nonlinear drift based on nonlinear and incomplete observations. *Phys. Rev. E*, 83:036705. 51