Bayesian Methods for High Frequency Financial Time Series Analysis

James Murphy

27th August 2010

0.1 Abstract

In this report we review the literature on financial time series modelling, Markov chain Monte Carlo (MCMC) methods and particle filtering (sequential Monte Carlo) methods. We then describe the application of reversible jump MCMC to the estimation of a two factor jump-diffusion model and show some results. A particle filter for on-line estimation of the model is described. The report concludes with a description of proposed future work.

Contents

Chapter 1

Introduction

The aim of this work is to apply Bayesian state estimation methods to financial time series, specifically high frequency foreign exchange data. We aim to develop a model that will be useful in a trading application, which requires the model to have some mechanism for recognizing patterns in the price data which can be used to generate trading opportunities. To this end we propose a relatively simple state space model of the process that incorporates a 'trend' term in order to attempt to capture any price trend in the data.

We develop Bayesian estimation methods to determine the underlying state and the model parameters of this model for a given series of observations. A Markov chain Monte Carlo (MCMC) method is used to estimate the parameters of the system on batch data. A particle filter is then developed in order to perform on-line estimation of the underlying state of a model with the parameter estimates generated by the MCMC estimation.

1.1 Structure of the Report

This report is split into a further six chapters.

Chapter 2 reviews the the literature surrounding the modelling of financial time series, MCMC methods and particle filtering (sequential Monte Carlo) methods.

Chapter 3 presents our state space model for asset prices and derives transition and observation functions for it.

Chapter 4 gives details of an MCMC algorithm for estimating the hidden state and parameters of our model.

Chapter 5 develops particle filter methods for on-line estimation of the hidden state of our model.

Chapter 6 shows some results from the estimation of our model.

Chapter 7 outlines a plan of future work in this area.

Chapter 2

Literature Survey

Our literature survey can be broadly divided into the application area, high frequency finance, and the solution techniques we use, Bayesian state space methods. The latter can be sub-divided into Markov Chain Monte Carlo (MCMC) methods, useful for batch processing and parameter estimation, and particle filter methods, useful for online state estimation.

2.1 High Frequency Finance

High frequency finance is the study of the behaviour of asset prices when considered at frequencies typically higher than daily. Such studies have become more possible with increased computing power and better data collection. Increasingly high frequency finance refers to studies of so-called *tick data*, which is data collected at all (public) transaction times. Such data is sometimes referred to as ultra high frequency and is the upper frequency limit of financial data. In heavily traded markets there can be several hundred (irregularly spaced) ticks per hour.

Tick data itself consists of a time stamp indicating when a trade took place, along with the price at which the trade was conducted. It might also contain data about the volume of the trade and further information about the *order book*, which records outstanding *bids* and *offers* (or *asks*) in the market. A bid is an indication that a market participant is interested in buying at the specified price, and an offer indicates that a participant is interested in selling the asset at the stated price.

Before describing the models and statistical observations about asset returns, it is worth describing, precisely, what is meant by a return (as it is a surprisingly ambiguous term responsible for considerable confusion). In almost all academic finance papers, return from time t to $t + \Delta t$ is defined as the change in the log price of an asset over that period:

$$r(t, \Delta t) = \ln S(t + \Delta t) - \ln S(t), \tag{2.1}$$

where $S(t)$ is the price of the asset at time t.

2.1.1 Models

The earliest work in the mathematical study of finance was that of Bachelier (1900). This started from the assumption that returns in a period were independent and identically distributed (iid), leading to a Brownian motion model for stock prices as a natural consequence of the central limit theorem. The work was, in fact, one of the earliest independent expositions of the random walk as a stochastic process (Davis and Etheridge (2006) give an interesting description of the early development of financial modelling and stochastic analysis). The important mean-variance framework of Markowitz (1952) does not explicitly require normally distributed returns but does rely on returns being sufficiently characterized by their first and second moments to make sensible portfolio decisions; Gaussian returns obviously satisfy this.

Perhaps the earliest systematic criticism of this method of modelling asset prices came from Kendall and Hill (1953), who demonstrated that the empirical variance of weekly asset prices (including Chicago wheat prices and British stock indices) was significantly different in the first half of the series than in the second, suggesting the variance of the series was not constant.

A different concern was addressed by Mandelbrot (1963), who noted that (weekly) asset returns had a distribution that, though it resembled the Gaussian bell-shaped curve, had many more outliers than would be expected from a Gaussian random variable (i.e. return distributions were more peaked in the middle with longer and thicker tails than Gaussian distributions). Since the assumption of identically distributed returns was retained a *stable* asset return distribution was proposed, so that the sum of the individual returns would have the same distribution as the returns themselves. The family of stable (or α-*stable*) distributions was discovered by Levy (1925) and can be parameterized by the α parameter, which gives a measure of the weight of a distribution's tails compared to its centre (with small α having the heaviest tails). The stable distributions have α parameters between 0 and 2 and do not, in general, have closed form density functions, although the family includes the Gaussian ($\alpha = 2$), Cauchy ($\alpha = 1$) and Levy ($\alpha = 1/2$) distributions as special cases. None of the stable distributions other than the Gaussian have a defined variance.

Mandelbrot (1963) (and the follow up work of Fama (1965)) purport to show that the variance of cotton prices does not converge (in the sense that the realized variance over an increasingly large window does not converge to a particular value). This conclusion has been criticized (e.g. Akgiray and Booth (1988)) because the observation is also consistent with time-varying volatility and does not necessarily imply divergent variance. This is then used to justify a stable distribution as the distribution of asset prices.

The heaviness (or thickness) of the tails of the distribution has been a recurring concern of asset price modellers. Numerous heavy-tailed return distributions have been proposed (see below), but a key development in its investigation has been the application of *extreme value theory* (EVT) to financial returns by Jansen and De Vries (1991). This quantifies the amount of probability in the tails of the distribution by considering the distribution of the extrema of the first n elements of a sequence M_n and m_n:

$$M_n = \max(X_1, X_2, ..., X_n) \tag{2.2}$$

$$m_n = \min(X_1, X_2, ..., X_n). \tag{2.3}$$

The limiting distribution of M_n as $n \to \infty$ is given by

$$P(a_n(M_n - b_n) \leq x) \xrightarrow{w} G(x), \tag{2.4}$$

where a_n and b_n are normalizing constants, $\xrightarrow{w}$ signifies weak convergence and $G(x)$ is a limiting distribution. Perhaps surprisingly, for iid sequences if the normalizing constants exist, the limiting $G(x)$ distribution converges to one of three distributions (Fisher and Tippett (1928)):

Gumbel:

$$G(x) = \exp(-e^{-x}) \tag{2.5}$$

Frechet:

$$G(x) = \begin{cases} \exp(-x^{-\alpha}) & x > 0 \\ 0 & x \leq 0 \end{cases} \tag{2.6}$$

Weibull:

$$G(x) = \begin{cases} 1 & x > 0 \\ \exp(-(-x)^{-\alpha}) & x \leq 0 \end{cases} \tag{2.7}$$

The distribution to which $G(x)$ converges is determined by the shape of the tail of the distribution of the individual series members, X_i. If the distribution of the X_i's has finite support then convergence will be to the Weibull distribution. If it has a power-law tail with exponent α then convergence will be to the Frechet distribution. Power-law tails with tail exponent α are defined as tails that are proportional to $x^{-\alpha}$ (so that $\lim_{x\to\infty} x^{\alpha} p(X > x)$ converges to a finite non-zero limit). Otherwise the distribution will converge to the Gumbel distribution which "plays a role for extreme values analogous to that of the normal distribution for sums of random variables: it is the typical limit for the distribution of iid extremes" (Cont (2001)).

The value of α is known as the *tail index* or *tail exponent* of the distribution, with distributions that converge to the Gumbel distribution having a tail index defined as ∞ and being known as having *exponential tails*. Such distributions include many common 'thin-tailed' distributions such as the normal, log-normal, exponential and finite Gaussian mixtures. For modelling heavy tails distributions with power-law tails are necessary, leading to power-law tails. These include distributions such as the Student-t, Pareto, inverse-Gaussian and hyperbolic. They also include the stable distributions with $\alpha < 2$, which Levy (1925) showed to have tails asymptotically equivalent to a Pareto law with tail index equal to their α parameter. The tail index of a distribution determines which moments are defined. Distributions with ∞ tail index have all moments defined but for those with finite α the moments greater than α are divergent.

With the introduction of EVT to finance, considerable effort has been made to establish the tail index of returns. A very common way of doing this is the Hill estimator (Hill (1975)), which is based on maximum likelihood. This estimator has the advantage of asymptotic normality, meaning that standard hypothesis testing methods can be used. This can be done using several methods: ... Tail estimation methods ... These estimators have the advantage of asymptotic normality, which means that standard hypothesis testing methods can be used. Clauset et al. (2009) covers the practical issues of estimating tail indices in detail.

The tail index of various assets at various frequencies has been studied by Jansen and De Vries (1991) (S&P500, daily, $\alpha \approx 3-5$), Longin (1996) (NYSE, daily, $\alpha \approx 3-4$), Lux (2001) (DAX, minute, $\alpha \approx 2-3.5$), **?** (FX, 10m - 6h, $\alpha \approx 3-5$) and Hauksson et al. (2001) (FX, 10m - 2wks, $\alpha \approx 3-4.5$), amongst others. Almost all of the established tail indices lie in the range 2-5 and according to **?** and Hauksson et al. (2001) appear to be reasonably stable over various frequencies. This implies lighter tails than the stable distributions proposed by Mandelbrot (1963), but heavier tails than the normal distribution. However, Weron (2003) cautions that since the convergence results are asymptotic, data generated from a stable distribution with $\alpha < 2$ can appear in finite samples to have tail indices greater than 2, so that the results above do not necessarily preclude a stable distribution and could be consistent with a stable distribution with $\alpha \approx 1.8$.

One of the attractions of the stable distributions as asset return distributions is that they can be seen as a consequence of a generalized central limit theorem (CLT). However, convergence under generalized CLTs can be very slow and the tail index does not seem to be consistent across time scales for real data (e.g. in Plerou et al. (1999)), so it is not clear that this is a good justification. For tick-data no central limit theorem should be in operation since the tick-to-tick moves are atomic and not the consequence of multiple aggregated moves. There is therefore no particular reason to expect ultra-high frequency returns to be drawn from a stable distribution. The observation of aggregate Gaussianity (e.g. Cont (2001)) (i.e. the tendency for returns to look more Gaussian on longer time scales) adds weight to the idea that the tail index of the high-frequency returns is greater than 2, since in that case the classical CLT holds (even when the returns are not identically distributed), so one might expect long-term returns, as the sum of tick-by-tick moves, to look increasingly Gaussian.

The thickness of the tails does not specify the particular distribution from which returns are drawn, so there has been work fitting a range of distributions to the observations. These include the Student-t (Blattberg and Gonedes (1974)), hyperbolic (Eberlein et al. (1998)), normal inverse Gaussian (Barndorff-Nielsen (1997a)) and exponentially truncated stable (Cont et al. (1997)) distributions. Cont (2001) suggests that the choice between distributions is a "matter of analytical and numerical tractability".

The Fisher-Tippet theorem that underpins EVT, and the central limit theorem (and its generalized counterpart) that underly the idea of stably-distributed returns are predicated on the idea of independent, identically distributed returns (though the identicality requirement can be relaxed in the case of the ordinary central limit theorem if moments higher than 2 are defined), but this assumption has been challenged in the analysis of financial data.

Furthermore, it is not clear whether modelling financial returns with leptokurtic distributions that do not have exponential-law tails (for example Gaussian mixtures) is wholly inappropriate. Realistically, financial returns do not have an infinite range (for example, exchange rates are unlikely ever to reach 0) and so, except in the case of catastrophe modelling (for example in insurance) it might be acceptable to model the distribution correctly in some reasonable albeit wide range. This might then admit, for example, Gaussian mixture models, at least in contexts where one is aware of their limitations for extreme event modelling. For example, Behr and Pötter (2009) finds a two-component Gaussian mixture to be adequate for monthly returns (though less so for daily returns).

The serial correlation of returns was investigated early on in finance, but their presence was largely dismissed (e.g. Kendall and Hill (1953)). The subject was then somewhat neglected in the 1960s and 1970s in favour of distributional modelling. Fama (1970) found that there was very little serial correlation in stock returns, as did Pagan (1996). However, Greene and Fielitz (1977) finds evidence of long-range dependence in stock returns by estimating the Hurst exponent of stock returns, which he finds to be in the region of 0.7, indicating long-memory behaviour. Though stock returns themselves were found not to exhibit serial correlation several authors identified the fact that two (non-linear) measures of the size of returns, absolute return and squared return, do show serial correlation (Engle (1982), Ding et al. (1993), Granger and Ding (1994), Lo (1991), amongst others). These studies show that the autocorrelation between absolute returns (and squared returns) remains positive and decays slowly as a function of the lag, with the greatest correlation being at lag 1. Ding et al. (1993) and Granger and Ding (1994) study the correlations of powers of the absolute return and find that it is highest for the first power (absolute return) and decays slowly thereafter. Granger and Ding (1996), amongst others, observes that the decay of a power of the absolute return with lag resembles a power law.

A further dependence effect that has been noted in stock returns is the so-called *leverage effect*, in which the volatility of a stock tends to increase with negative price movements. It was first observed by Black (1976) and is studied further in Bouchaud (2001) and in high frequency data in Bollerslev et al. (2006).

It has also been noted that volatility is related to the volume of stocks traded (for example Clark (1973) and Andersen and Bollerslev (1997b)). Clark (1973) suggests that volume of trade is an indication of the type of news that is driving the market and uses trading volume as a proxy for market activity as a way to reduce seasonal effects, which can be significant, particularly intraday foreign exchange returns (Andersen and Bollerslev (1997a)).

The earliest attempt to model serial correlation in asset prices was that of Engle (1982) who introduced the *AutoRegressive Conditional Heteroskadastic* (ARCH) model. ARCH models allow for serial correlation and long-range persistence of volatility by introducing a scaling process h_t for the process variance which, in an ARCH(q) process, depends on the previous q values of h_t. This scales an iid innovation process, which has typically been Gaussian but need not be. The innovation is then added to a mean to give the final process. For example, for returns r_t coming from an ARCH(q) process, we write

$$r_t = \mu + \epsilon_t, \tag{2.8}$$

$$\epsilon_t = \sqrt{h_t} z_t, \tag{2.9}$$

$$h_t = \alpha_0 + \sum_{j=1}^{q} \alpha_j \epsilon_{t-j}^2, \tag{2.10}$$

$$z_t \sim D(0, 1), \tag{2.11}$$

where D is the innovation distribution. ARCH processes have constant unconditional variance and exhibit *leptokurticity*, that is, higher kurtosis (fourth moment) than the Gaussian distribution. The model is important because it was the first to introduce the idea of variable variance to the econometric literature.

Bollerslev (1986) extended the ARCH model to allow dependence not only on the realized variance but also on previous conditional variance, changing the

h_t process above to

$$h_t = \alpha_0 + \sum_{j=1}^{q} \alpha_j \epsilon_{t-j}^2 + \sum_{j=1}^{p} \beta_j h_{t-j}. \tag{2.12}$$

This process is the GARCH(p, q) process and was introduced to overcome some of the shortcomings of the ARCH model, especially concerning its estimation in empirical settings. Bollerslev (1986) shows that the GARCH(p, q) model with $p > 0$ can be viewed as an ARCH(∞) model. The GARCH process, and especially the simple GARCH(1,1) process have had enormous success in financial econometrics. Engle (2001) describes some applications of ARCH/GARCH models in finance. An important real-world use of a GARCH model is in J.P. Morgan's *RiskMetrics* software, which is used for *Value at Risk* (VaR) calculations. A very large number of extensions to the GARCH model have been proposed, including an integrated GARCH process (which has a unit root so is non-stationary), the IGARCH, and a nonlinear variant, NGARCH. Hansen and Lunde (2005) lists sixteen variants of the basic ARCH/GARCH process, which they also compare on real data. They find that for daily foreign exchange data a GARCH(1,1) model performed as well as a number of other more sophisticated GARCH-type models, but that for daily IBM stock returns a model that could also accommodate the leverage effect was more accurate.

Whilst ARCH models arose from the attempts of econometricians to model the behaviour of asset prices and other econometric variables, a parallel development of models was taking place in the field of option pricing. The groundbreaking option pricing formula of Black and Scholes (1973), based on the idea of the absence of arbitrage (risk free profits) in financial markets, used a simple model of asset prices based on *geometric Brownian motion* (GBM) (simply an exponentiated random walk, giving log-normal returns). Though this simple model did not exhibit many of the stylized facts of stock returns that had been observed up to that point, the fact that a closed-form formula for option prices had been found was of such significance that it gave renewed vigour to random-walk models of asset prices (and their geometric variants).

A big difference between the 'econometric' asset price models (such as GARCH) and those used in option pricing was that whereas the former were discrete time models the latter were continuous in time. GARCH and other time series models move from one price point to the next according to a set of (stochastic) difference equations. Their prices are only defined at a discrete (countable) set of price points. Continuous-time models, on the other hand, are described by a set of (stochastic) differential equations that describe the *instantaneous* evolution of the system. Their prices are defined at every (uncountable) point in time. The two views are connected, of course, since the continuous-time formulation can be seen as the infinite limit (infinitesimal time steps) of the discrete process or the discrete process can be seen as the value of the continuous process at certain points in time. Either view can be seen as an approximation of the other. In, say, describing the physics of motion a discrete system is clearly an approximation to the 'true' (in the usual physical sense of 'nearly true') underlying continuous system. In finance, in which the ultimate temporal limit of asset prices are their tick prices, defined at a series of points, it is much less clear that the discrete system is the approximation and in fact, seem more the case that continuous-time formulations are an approximation of the underlying discrete system in order

to make them analytically more tractable. Nonetheless, particularly in option pricing work, the continuous-time formulation *is* more tractable and continuous time models have gained great significance. Such models are most usually described as a set of stochastic differential equations. For example, the GBM process used in the original option pricing work has the following stochastic differential form

$$dS_t = \mu S_t dt + \sigma S_t dW_t, \tag{2.13}$$

where S_t is the asset price at time t, dW_t is the differential of a standard Gaussian process and μ and σ are the constant expected return and volatility of the asset, respectively.

In attempting to correct the inaccuracies in option pricing, most famously the *volatility smile* observed by e.g. Schmalensee and Trippi (1978), more sophisticated models of stock prices still consistent with the option pricing framework of Black and Scholes (1973) and Cox et al. (1979) (who introduced the binomial asset pricing model where asset prices evolve on according to a binomial tree of up and down moves) were developed. This led to the jump-diffusion model of Merton (1976) and the the stochastic volatility models of Hull and White (1987) and Heston (1993).

Stochastic volatility models extend the GBM model in equation 2.13 by making the volatility process stochastic, giving the following differential form

$$dS_t = \mu S_t dt + \sqrt{v_t} S_t dW_t \tag{2.14}$$

$$dv_t = \alpha(S,t)dt + \beta(S,t)dB_t, \tag{2.15}$$

where α and β are functions of the asset price and time and dB_t and dW_t are differentials of (possibly correlated) standard Gaussian processes. The model of Heston (1993) has the following differential form for variance

$$dv_t = \theta(\omega - v_t)dt + \xi\sqrt{v_t}dB_t. \tag{2.16}$$

The Hull and White (1987) model has the form

$$dv_t = \theta(\omega - v_t)dt + \xi v_t dB_t. \tag{2.17}$$

Nelson (1990) shows that under certain conditions the continuous limit of the GARCH(1,1) process has the same volatility process as the stochastic volatility model of Hull and White (1987), though with a price process that has has a volatility-dependent μ. Duan (1995) produced an option-pricing formula for underlying assets following a GARCH(1,1) process.

Kim et al. (1998) compare a Gaussian GARCH model, the stochastic volatility model of Hull and White (1987) and a Student-t based GARCH model on daily foreign exchange data using Markov Chain Monte Carlo (see section 2.2) to calibrate the models. They find that the stochastic volatility model performs somewhat better than the Gaussian GARCH model, but that both are beaten by the Student-t based GARCH model. Taylor (1994) also compares a range of stochastic volatility models. Andersen and Bollerslev (1998) finds that ARCH-type models povide a good predictor of future volatility, and criticizes much of the earlier work suggesting otherwise on statistical grounds, arguing that their tests for predictive power were badly formed. A more appropriate test is proposed. Carnero et al. (2004) show that the relationship between kurtosis,

persistence of volatility shocks and first order correlation of squares is different in GARCH and (discrete) autoregressive stochastic volatility models, which they suggest could explain why Gaussian stochastic volatility models perform better than Gaussian GARCH models.

An alternative to stochastic volatility models of the types described above is a regime-switching model in which volatility can take a number of discrete levels. Such a model allows for the very rapid changes of volatility sometimes seen in real market prices (e.g. after a crash, see below). Elliott et al. (1998) use a *hidden Markov model* (HMM) formulation (a hidden state model in which the state can only take a finite number of values) to estimate such a discrete volatility process. Rossi and Gallo (2006) use a similar model and find that "it does comparatively well in short-term volatility forecasting", whilst being reasonably parsimonious in its parameterization.

Merton (1976) introduced jump-diffusion models in order to improve the accuracy of option prices. The idea of 'jumping' only makes sense in the continuous context, in which case it would mean that the continuous-time series contained price discontinuities, unlike Brownian motion, which is continuous in both space and time. If we accept a continuous approximation, an obvious question, and one asked by Eraker (2004), is, "do stock prices jump?" The meaning of the question is not obvious: of course stock prices jump since in reality they are discrete. We can interpret the question as meaning, "does adding price discontinuities to a continuous time approximation of stock prices improves its ability to model them?" Eraker (2004) examines two measures of "ability to model stock prices": the ability of the model to fit observed option prices and the ability of the model to describe observed returns. He concludes that jumps (in both the price and volatility process) help with the latter but do not add much to the former, as assessed on daily S&P500 option and closing prices

Bates (1996) was perhaps the first to provide an option pricing formula for a model that combined stochastic volatility and jumps in the price process. He claims that the model fits the option price data in dollar/Deutsche Mark options better than a stochastic volatility model alone. Bates (2000) and Pan (2002) employ similar models and also conclude that the inclusion of jumps helps explain option prices, especially after the 1987 crash.

Duffie et al. (2000) introduces two stochastic volatility models with jumps in both the price and volatility processes. One of the models has jumps in the two processes with independent arrival times and sizes, whereas the other has jumps co-located in both processes with correlated sizes. This latter model can be seen as an attempt to model the leverage effect of Black (1976), mentioned above, since it allows volatility to jump suddenly in the face of a price crash. Both Duffie et al. (2000) and Eraker et al. (2003) find that such models provide a better fit to option prices than a stochastic volatility models with jumps in the price process alone. Eraker et al. (2003) use an MCMC method on daily return data to calibrate his models, whereas Duffie et al. (2000) calibrate their model on option price data. Eraker et al. (2003) find that the model with correlated jumps has more success in replicating some features of option prices. The reason for the success of stochastic volatility models with jumps is that they can recreate the high levels of skewness and kurtosis seen after sudden market movements, whereas without jumps, since the price/volatility can only vary as a diffusion sudden changes in the skew/kurtosis cannot be produced.

The stochastic volatility model without jumps described above can produce

unconditional return distributions that are heavy tailed (Bates (1996)), however since the price process is driven by Gaussian noise, the distribution of the returns conditional on the volatility is itself Gaussian. The introduction of jumps in the price and volatility processes can make the conditional distributions non-normal. A somewhat alternative (but related) idea is to use a non-Gaussian proccess for the volatility, the price or both. This changes the nature of the stochastic integration involved in integrating the system. This could be circumvented in finite-activity jump models (e.g. models driven by Poisson jumps) where the jumps could be represented as a simple sum and integrated separately. A different (and mathematically more involved) approach is to model the price and volatility processes as Lévy processes.

Lévy processes are a set of processes with independent identically distributed returns that are infinitely divisible. An infinitely divisible variable X is one for which, for every natural number n there are n iid variables whose sum has the same distribution as X. A full definition of such processes is given in Barndorff-Nielsen and Shepherd (2001). The Brownian motion is one such process (and the only continuous one; all others have jumps in their sample paths), but there are an entire class of processes that fit the definition. In finance three other Lévy processes have proved popular (Barndorff-Nielsen and Shephard (2006)): the jump-diffusion process (Merton (1976)), the variance gamma process (Madan and Seneta (1990)), and the zero mean normal inverse Gaussian process (Barndorff-Nielsen (1997b)). Though the processes have iid increments, Carr and Wu (2004) showed that stochastic volatility could be incorporated through a time-change function. Lévy processes, then, provide quite a general framework with which to build financial models, allowing both alternative stock return distributions (conditional and unconditional) and stochastic volatility. Thus, they present a possible way of unifying the work on distributional modelling and that on stochastic volatility of the past 30 years. The class of processes is, however, restrictive enough to permit an integral calculus based on the decomposition of the process into a Brownian motion, a compound Poisson process and a pure-jump martingale (see e.g. Barndorff-Nielsen et al. (2001)), meaning that models involving them are mathematically somewhat tractable (for example sometimes permiting closed-form integration). Barndorff-Nielsen and Shephard (2006) give likelihood methods for the estimation of a Lévy process based on the normal inverse Gaussian distribution. They find that they fit well to real foreign exchange data. Lévy process modelling appears to be becoming more popular in finance, but does increase the mathematical complexity of models. The area is an active area of current research and warrants further investigation and a more extensive survey in the near future.

2.1.2 Stylized facts

Having reviewed a number of models and studies of asset returns, there are a number of commonly observed statistical features that seem to be common across a range of time periods and asset classes. These so-called *stylized facts* are given here, based on the survey papers of Granger and Ding (1994), Granger et al. (2000), Guillaume et al. (1997) and Cont (2001)) and on the modelling work described above.

Heavy tails (leptokurtosis) Heavy tails are noted in the distributions of

both conditional and unconditional returns.

Autodependence Whilst there appears to be no autocorrelation in returns themselves there does seem to be significant correlation in absolute returns and their powers, which decays with increasing lag.

Skewness Large downward movements are observed in stock prices, but equally large upward movements are not. This effect is not present in foreign exchange.

Aggregational Gaussianity On long time scales returns tend toward being normally distributed. On short time scales they are significantly non-normal.

Leverage Effect Volatility is often seen to be negatively correlated with returns, so that it increases after periods of asset price declines.

Volume/Volatility correlation Trading volume is positively correlated with all measures of volatility.

2.2 Markov Chain Monte Carlo

Markov Chain Monte Carlo (MCMC) methods were first introduced by Metropolis et al. (1953). They are used for sampling from a *target* distribution, which might only be known up to proportionality and from which it might be difficult to sample. They work by setting up a Markov chain with the target distribution as its stationary distribution. This chain can then be run for a large number of steps and will (under certain mild conditions given by Roberts and Smith (1994)) converge to the target distribution, meaning that the distribution of the state of the chain over time will be equal to the target distribution (or a normalized version of it, if the target distribution does not integrate to 1) and thus, samples from the state-sequence of the chain will be drawn from the target distribution.

2.2.1 Samplers

Metropolis et al. (1953) showed how to sample from such a target distribution using a much simpler (symmetric) *proposal* distribution from which samples could be easily drawn and the Markov chain idea. Hastings (1970) generalizes this to use asymmetric proposals and so the algorithm is often known as *Metropolis-Hastings*. It works by drawing a sample from the proposal distribution and then accepting this proposal as the next point in the chain with a probability given by

$$\alpha(X, Y) = \min\left(1, \frac{\pi(Y)q(X \mid Y)}{\pi(X)q(Y \mid X)}\right), \tag{2.18}$$

where X is the current state of the chain, Y is the proposal, π is the target density, q is the proposal density and α is the acceptance probability. Using this sets up a Markov chain with the target distribution as its stationary distribution (this can be shown via the *detailed balance equations* of the Markov chain

- see, for example, the first chapter of Gilks et al. (1996)). Once we have established that the stationary distribution is our target distribution we can call on a fundamental result of Markov chains that finite state-space irreducible, aperiodic chains have a unique stationary distribution to which they converge as the number of steps goes to infinity (see e.g. Levin et al. (2009)). The extension of this result to countably infinite and uncountably infinite (e.g. continuous) state spaces is outlined in Tierney (1996) and Rosenthal (1995). The number of steps taken to reach a point where the distribution of samples being generated from the chain is approximately the stationary distribution is known as the *mixing time* of the chain.

The Metropolis-Hastings algorithm, then, gives us a way of drawing samples from a target distribution. The applications of this general statistical problems was perhaps first realized by Gelfand and Smith (1990), although it had been previously used for the restoration of images (Geman and Geman (1984)). The basic idea of applying MCMC methods to statistical problems is to derive an expression for the probability of the parameters given the data. This will be a multi-dimensional probability density function with the dimensionality determined by the number of unknown parameters. This is then used as the target distribution of an MCMC process. After running the MCMC process until it approximates its stationary distribution, samples will be drawn from the normalized target distribution, giving a Monte-Carlo approximation of the pdf of the parameters. This fits particularly neatly with the Bayesian approach to inference since the pdf of the parameters given the data is simply the posterior probability of the parameters given the data. Here the term 'parameters' is used in the general sense of anything unknown in the model under consideration (so it can include hidden state at a series of times or missing data values). This means that if the model is, for example, a state-space model then the dimensionality of the parameter space (and thus the state space of the Markov chain) can be large.

In some problems the dimensionality of the state space might be unknown, for example in model selection problems where the number of parameters of the model is itself one of the unknown parameters. In this case the simple MCMC method described above needs to be adapted in such a way as to allow a variable-dimensionality state space to be explored. This can be done through the use of *reversible jump* MCMC (RJMCMC), first introduced by Green (1995). In this method the proposal function must be able to move (reversibly) between state-spaces of different dimensions, allowing the entire space to be explored. This can be considered as multiple fixed-dimension MCMCs that have the ability to transition between each other (with the correct probability).

Though any proposal distribution will give a Markov chain with the target distribution as its stationary distribution, the time that it takes to converge to that distribution depends on the relationship between the target and the proposal functions. The time taken to explore the entire support of that distribution once reached also depends on this relationship. For this reason a large number of schemes have been developed for coming up with good proposal functions.

If the dimensionality of the state space is large then it can be difficult to generate good proposals that will be accepted, since any one of the dimensions of the state could take the proposal to an area of low probability and cause it to be rejected. Instead of updating the entire state space at each step, it is perfectly possible to update just a single component or subset of components

of the state. This is done is exactly the same way as a full state update, with all other state components retaining their previous value. This was, in fact, the original scheme proposed by Metropolis et al. (1953). The acceptance proposal for the site update is

$$\alpha(X_i, X_{-i}, Y) = \min\left(1, \frac{\pi(Y_i \mid X_{-i})q(X_i \mid Y_i, X_{-i})}{\pi(X_i \mid X_{-i})q(Y_i \mid X_i, X_{-i})}\right), \tag{2.19}$$

where X_i is the component (or block of components) being updated, X_{-i} is the state vector without the components in X_i, Y_i is the new proposal for the components being updated and $\pi(X_i \mid X_{-i})$ is the *full conditional* of the X_i component, i.e. the probability of X_i given the rest of the state. (This is easily derived from the original Metropolis-Hastings acceptance probability in equation 2.18 by noting that $\pi(Y) = \pi(Y_i|Y_{-i})\pi(Y_{-i})$ by the chain law of probability (and similarly for $\pi(X)$) and then noting that since the components other than i do not change from the current state to the proposal, $Y_{-i} = X_{-i}$, the the $\pi(X_{-i})$ and $\pi(Y_{-i})$ terms cancel in the product).

A special version of site-by-site updating is the Gibbs sampler, introduced by Geman and Geman (1984) . This uses the conditional probability of the site being updates as the proposal function, i.e. $q(Y_i \mid X_i, X_{-i}) = \pi(Y_i \mid X_{-i})$. Putting this into equation 2.19 causes the entire fraction to cancel, giving an acceptance probability of 1 at all times. This means that if the full conditionals are available and easy to sample from it is possible to use them to make proposals that are always accepted. This sampling scheme has been used extensively, especially in early MCMC applications, for example Gelfand and Smith (1990) and Spiegelhalter et al. (1994), the latter of which introduced the popular BUGS software for Gibbs sampling. One reason for this popularity is an absence of parameters that need to be tuned.

In a block-update method, choosing the blocks to sample can have a significant influence on the time taken to explore the state space of the stable distribution (or *mixing time*). Gilks et al. (1996) suggests that blocking together highly correlated components may improve mixing, though this depends on the choice of proposal.

Another simple proposal scheme is the *independence sampler*, first discussed by Hastings (1970) and generalized by Tierney (1994), in which the proposal function does not depend on the current state. Liu (1996) looks at the performance of this algorithm in comparison to importance sampling and rejection sampling and finds that it works well when the proposal distribution is similar to the target distribution, but preferably with heavier tails (Gilks et al. (1996)). Roberts (1996) comments that "it is rare for the independence sampler to be useful as a stand-alone algorithm," but that "within a hybrid strategy which combines and mixes different MCMC methods, the method is extremely easy to implement and often very effective."

As hinted by the preceding quote, all of these updates methods lead to a chain with the same stationary distribution so they can be used together to increase the variety of the update function and to avoid the pathologies of any particular method (e.g. Brooks (1998)).

Clearly the proposal function and its relation to the target function are of crucial importance in ensuring good results from MCMC algorithms. However, when an MCMC algorithm is started it may know practically nothing about

the shape of the target distribution. Progression of the MCMC chain reveals information about the target function and so there is an obvious temptation to use this additional information to refine the shape of the proposal function (for example reducing the width of the proposal distribution if only short steps are being accepted, or adding a new component to a mixture model proposal if a new mode of the target function is found). This is the idea behind *adaptive MCMC* algorithms. However, caution must be exercised since naive adaptation can destroy the *ergodicity* of the chain, meaning that the estimator of the expectation of a function g under the target distribution π given by

$$\mathbb{E}_\pi(g) \approx \bar{g}_N = \frac{1}{N} \sum_{n=M+1}^{M+N} g(X_n), \tag{2.20}$$

(where M is the burn in period and X_n is the n^{th} sample in the chain), is no longer a consistent estimator. Gelfand and Sahu (1994) give an example of infinite adaptation destroying ergodicity and that paper proposes a very simple solution: stop adaptation after some fixed time period.

Two early algorithms that allows continuous adaptation without damaging ergodicity are the adaptive direction sampling (ADS) and the related adaptive Metropolis sampling (AMS) algorithms of Gilks and Roberts (1996) and Roberts and Smith (1994). ADS works by augmenting the state space to include a (fixed size) set of previous state points upon which adaptation is based. This is done by sampling along a line between the existing point and one of the previously visited states, with the idea that such a line is increasingly likely to traverse the high probability regions of the target. The algorithms were observed to have "inconsistent performance" on some problems on which traditional methods performed well. These algorithms are closely related to Skilling's leapfrog method (see MacKay (2003), section 30.4).

In general, any algorithm that can incorporate the adaptive parameters into the state space will retain ergodicity because it will still obey the Markov condition (Gilks et al. (1998)).

A different approach is taken by the *regeneration* algorithm of Gilks et al. (1998). This is based on the idea of Markov chain regeneration which, in discrete state spaces, occurs when the chain revisits some nominated state, and which with some work can be extended to continuous state space Markov chains. At such regeneration points the sample path of the Markov chain to the next regeneration is independent of the sample path from the previous regeneration point to this one. This allows the transition kernel of the chain to be adapted at that time using the entire history of the chain up to that point as a basis for the adaptation. The algorithm allows an unlimited amount of adaptation but in high dimensional spaces is limited by the difficulty of obtaining frequent regeneration (Gilks et al. (1998)).

An important development in adaptive MCMC was the *adaptive Metropolis* algorithm of Haario et al. (2001) and the corresponding analysis that showed that, though non-Markovian, it retained ergodicity. This algorithm uses a Gaussian proposal with a covariance matrix that can adapt based on the *entire* state space history seen up to that point. The adaptation is not free, however, but must be performed according to a specific formula that allows the ergodicity property of the resulting chain to be proven. Atchadé and Rosenthal (2005)

relax some of the conditions of this and give an adaptive version of the standard random walk Metropolis algorithm.

The work of Haario et al. (2001) led to several efforts to place minimal bounds on the adaptation process in order to ensure ergodicity. Andrieu and Moulines (2006) establishes ergodicity and some other results for chains with vanishing adaptation and shows that adaptive versions of the random walk Metropolis and independent Metropolis-Hastings algorithm can be constructed to satisfy the conditions given. Roberts and Rosenthal (2005) gave slightly weaker conditions on the convergence of the transition kernels of the Markov chain that still ensured ergodicity (though lost some other properties).

The establishment of bounds on the adaptation process within which ergodicity is maintained has lead to numerous proposed adaptive schemes within those bounds. Haario et al. (2006) proposed a single component version of his active Metropolis algorithm. Roberts and Rosenthal (2008) propose two further schemes: state dependent scaling, which attempts to alter the scale of a Gaussian proposal to be optimal at the current point, and regionally adapted Metropolis, which divides the state space into a number of disjoint regions and then attempts to scale Gaussian proposals in each of these regions to be optimal. Haario et al. (2006) proposes delayed rejection adaptive Metropolis, in which if a proposal is rejected a number of refined proposals can be submitted, allowing local adaptation, whilst still retaining the desired stationary distribution. The adaptive version of this algorithm builds this delayed rejection strategy into an adaptive Metropolis algorithm, giving both local and global adaptability and an ability to counter poor calibration of either.

Several algorithms are based on the results of Roberts et al. (1997) and Roberts and Rosenthal (2001) that shows that under certain conditions the optimal acceptance rate for high dimensional MCMC methods with Gaussian proposal and target is 0.234.

Adaptive MCMC continues to be an active area of both theoretical and applied research, for example Atchadé and Fort (2009) develops some further limit theorems for adaptive MCMC and Tian et al. (2009) apply adaptive MCMC to image restoration.

Along with adaptive MCMC other methods for improving the efficiency of MCMC exist. Overrelaxation (Alder (1981)), generalized by Neal (1998) aims to suppress random walk behaviour in order to improve performance in highly correlated systems. *Slice sampling* (Neal (1997) and Neal (2003)) belongs to the family of auxiliary variable methods, where the state space is augmented with additional information. In this case, it is a u coordinate, giving a state space (x, u), where u corresponds to the value of the target function at x. The aim is then to sampling from a uniform distribution over the (hyper-)volume enclosed by the function and the state space axes (e.g. for a one dimensional function, the area under the curve). Discarding the u component of the state space then gives a set of x samples drawn from the target distribution.

The *Hamiltonian* or *hybrid* Monte-Carlo of Duane et al. (1987) uses gradient information about the target in addition to the target function itself in order to direct the motion of the MCMC random walk. The method uses a state augmented with momentum variables. An step in the chain can either randomly update these or use them to direct the current motion according to mechanical principles, with the target function acting as an energy function (hence its gradient being used to update the momentum from step to step). This method,

perhaps due to its physical origin, is more popular in the physics literature than in the statistical literature.

Such methods hint at the link between MCMC methods which aim to generate a set of representative samples from a target function and optimization methods, which aim to maximize or minimize that function. Since multimodal distributions can cause difficulties for MCMC algorithms (since it is tricky to come up with a proposal that is both good at exploring the areas around the modes and at transitioning between them, as the two distances might be on significantly different scales), it seems appropriate to attempt to adapt optimization methods useful in such applications to MCMC. *Simulated annealing* (first proposed by Kirkpatrick et al. (1983)) is an optimization method designed to cope with multimodal distributions by performing a series of random walk Metropolis runs on a changing objective function controlled by a decreasing 'temperature' parameter. At high temperatures (at the start of the algorithm) the objective function is 'smoothed', making hollows shallower and hills less steep and allowing transitions to be made that would be unlikely with the true objective function. This allows a wide area of the parameter space to be explored. As the temperature cools, the landscape of the objective function becomes more defined and the algorithm tends to find the minima of the objective function. For optimization, the cooling continues until the algorithm reaches a minima and no longer moves. A naive attempt to apply this algorithm to MCMC would stop cooling when the objective function reaches the true objective distribution and run a random walk from there. This, however, produces biased samples, but Marinari and Parisi (1992) proposed a modified scheme called *simulated tempering* which removes this bias, allowing simulated annealing ideas to be used in MCMC schemes.

A different idea is that of *exact sampling* or *coupling from the past*, originally proposed by Propp and Wilson (1996). This is based around the idea of *coupled* Markov chains (ones that share a random number generator). The idea is that once two coupled chains get into the same state, since they share a random number generator, they will never separate in the future. This is called *coalescence*. If coupled chains were started from all possible start states then, after they had coalesced, they must be in a state from the stationary distribution since any possible starting value arrives there. However, we cannot just simulate forward until we get coalescence because the conditions under which coalescence occur might themselves affect the distribution (e.g. if points only coalesce when one state element reaches 100 then sampling after seeing coalescence will deliver biased samples). Instead, we simulate to time 0 from some point in the past, $-T$. If by time 0 coalescence has not occurred then we go back further (e.g. to $-2T$) and try again. Once we do see coalescence at time 0 we can take the time 0 sample as an exact sample from the stationary distribution. However, it is not feasible to simulate from all starting points (since there are far too many), so the method only works if we can use a small number of chains to bound the entire ensemble. For example, if we know that one dimensional chains never cross then we can just simulate the chains from the most extreme starting points and when they have coalesced know that all chains would have done so. This algorithm is more easily applied to finite state spaces, but coalescence is related to the idea of regeneration used by Gilks et al. (1998) in continuous spaces. Brooks et al. (2006) links the idea of perfect sampling to regenerative versions of simulated tempering schemes (and relax the backward-time idea described above so that

forward-time simulation can be used).

Finally, a recent introduction to the MCMC sampling field has been the particle filter sampler of Andrieu et al. (2010). This is based on the use of a particle filter (see section 2.3) with the target distribution of the MCMC algorithms as *its* target distribution. This can then be used as a proposal function for the MCMC algorithm, since the optimal proposal function is the target function itself. Andrieu et al. (2010) report good results for this algorithm, which requires only relatively simpler particle filter proposals to be designed.

2.2.2 Diagnostics

The methods above allow MCMC samples to be generated from a target distribution, once the chain has converged to the stationary distribution. However there still remain the crucial questions of how long convergence will take, and how to tell when it is arrived at. The former of these is very difficult to answer and is, in general, model dependent. All irreducible aperiodic Markov chains will converge to their stationary distribution, but the speed with which they do so depends on a number of things including their starting point, the sampler being used, the MCMC scheme in use and the shape of the target distribution. Cowles and Carlin (1996) observes that attempts to bound MCMC convergence "typically involve sophisticated mathematics, as well as laborious calculations that must be repeated for every model".

The second question is whether we can tell when an MCMC method has converged to its target distribution. Such MCMC diagnostics must always be heuristic, since in general we do not know anything about the target distribution and so any test of 'convergence' can only measure convergence between MCMC steps and not against the true distribution and so can easily misdiagnose slowly mixing chains as converged. The problem is analogous to that of detecting the optimum in global optimization. It is easy to tell that a local optimum has been reached but essentially impossible (in reasonable time) to determine whether or not it is the global optimum. Such problems "lead many theoreticians to conclude that all diagnostics are fundamentally unsound" (Cowles and Carlin (1996)). Nevertheless, there have been a very large number of suggestions for ways in which MCMC convergence can be detected. Cowles and Carlin (1996) lists thirteen different convergence diagnostics from the literature each with their own pathological cases and successes. Brooks and Gelman (1998) and Mengersen and Robert (1999) both examine a range of convergence diagnostics. Many diagnostics (such as Gilks et al. (1996)) are based on the statistical comparison of the output of multiple chains with different starting points; if they are sufficiently similar, convergence is assumed. Visual inspection remains popular, with Nylander et al. (2008) and Peltonen et al. (2009) providing some enhanced visualization methods for assessing convergence. Cowles et al. (1999) looks at possible biases introduced to MCMC samples via the use of a number of different convergence diagnostics. Flegal and Jones (2010) gives a recent guide to the analysis of MCMC output. The conclusion of several of these studies seems to be that MCMC diagnostics can be useful, should be used with caution, all have faults and to be most effective should be used together so as to minimize their chance of serious misdiagnosis.

2.3 Particle Filters

Particle filters, or *sequential Monte Carlo* (SMC) methods as they are also known, are a class of estimation techniques based on the use of discrete samples to approximate distributions of interest. Unlike MCMC methods they can be updated sequentially, as more data becomes available, and so are of particular interest for filtering and on-line estimation (i.e. estimation performed as data arrives). They are particularly suitable for Bayesian filtering problems for state space models, where the observations are viewed as being generated from a model with known state transition probabilities and observation probabilities, dependent on the current (and perhaps preceding) states (and subject to noise). At any point in time such models will have an expected prior distribution for the state (based on observations seen up to that point), which when a new observation arrives can be updated using the model specification to give a posterior estimate of the underlying system state at that time.

Clearly, probability distributions play a central role in such state inference. In some circumstances the probability distributions will be well known and easily parameterizable, for example in the case when the state transition and observation functions are linear and Markovian, the noise is Gaussian and the original prior distribution is Gaussian then all the prior and posterior distributions will also be Gaussian (and we can use the Kalman filter to extract the underlying state). However, such cases are the exception rather than the rule and in the general case the distributions involved can become very complicated.

Particle filters tackle this problem by using a discrete sample based representation of the underlying probability distributions. The advantage of using this approach is the ability to represent arbitrary probability distributions (though perhaps arbitrarily poorly). Early methods used grid-based approximations, either based on point masses on regular grids Bucy and Senne (1971), or on spline approximations of various complexities, e.g. Wang and Klein (1976). Limited computational resources in the 1970s meant work then tended to focus on more sophisticated interpolation schemes to reduce the storage burden of dense grids Kramer and Sorenson (1988). The early 1980s saw a hiatus of interest in such approximations, but by the late 1980s increases in computing power, particularly storage, motivated a renewed look at simpler methods Kramer and Sorenson (1988). Kitagawa (1987) used piecewise linear approximations and Kramer and Sorenson (1988) proposed piecewise constant approximations to the pdfs of interest. A general problem with such methods is that the construction of the grid, especially when the density functions involved are multimodal or have sharp peaks, can be significantly nontrivial (Kitagawa (1987), Kramer and Sorenson (1988), Gordon et al. (1993)).

An alternative way to approximate distribution functions, first introduced to these problems by Gordon et al. (1993), is to use a collection of weighted or unweighted random samples. Using random samples to make inferences is the essence of Monte-Carlo inference and hence such techniques are often known as sequential Monte Carlo (SMC) methods. The individual samples are frequently referred to as *particles* (e.g. Kitagawa (1996)) and the term "particle filter" was first used by Carpenter et al. (1999). A number of ways of updating these sample-based representations have been proposed, leading to a range of algorithms for sequential state inference.

Particle filters have been applied to a wide range of problems including

tracking e.g. Gordon et al. (1993), multiple object tracking e.g. Khan et al. (2005), calibration of stochastic volatility models e.g. Pitt and Shephard (1999), seasonal adjustment in econometrics e.g. Kitagawa (1996), audio processing, computer vision, control and others e.g. Doucet et al. (2001). Much of the methodology outlined in this review can also be applied to smoothing problems, see e.g. Cappé et al. (2007).

2.3.1 Basic Algorithm

Monte-Carlo methods approximate distributions by randomly placing a finite number of samples throughout the space of interest in accordance with the distribution being approximated. A variation on this is to allow the samples to be weighted. This has the advantage that by choosing the weights correctly any distribution can be approximated, even if the random samples themselves are drawn from a different distribution. This is useful because as noted in the MCMC section 2.2 above, sampling from an arbitrary distribution can be difficult.

The idea of importance sampling can be applied in the sequential setting of state estimation. Here, the samples are drawn from some known and easy to sample distribution, and the weights are calculated in such a way as to make the sample/weight combination approximate the required posterior distribution. Doucet et al. (2000) set out a framework within which much of the work on sequential Monte-Carlo can be understood, shown in algorithm 2.3.1.

Algorithm 1 Sequential Importance Resampling (SIR) filter

```
Initialize
Draw N samples form proposal function q
Calculate weights so this approximates initial prior
for t = 0 to T do
    (Resample if required)
    Propagate samples: draw N samples from proposal function q
    Weight samples so that they approximate posterior
    (Do any inference using posterior)
end for
```

Doucet et al. (2000) show, based on a theorem from Kong et al. (1994), that without the resampling step this algorithm is degenerate in the sense that the variance of the weights increases (stochastically) over time, meaning in practice that a very few samples will acquire large weights whilst the weights of the remainder will approach zero.

This algorithm generalizes earlier work because it does not specify the form of the importance function from which samples are drawn or the method of resampling. The following section reviews variants of this basic algorithm.

2.3.2 Variants

The first use of random samples to represent pdfs of interest in sequential inference problems was in Gordon et al. (1993). This introduced the *bootstrap filter* variant of the algorithm, which used the state transition prior as the importance function. This, along with the observation function is known from

the specification of the model and leads to a particularly simple form of update equation for the weights. Other early methods that effectively use the state transition function as the importance function are the Monte-Carlo filter of Kitagawa (1996), the Condensation method of Blake and Isard (1997) and the Sequential Imputations method of Liu and Chen (1995).

The resampling method used by these algorithms is straightforward. Resampled particles are drawn from the set of existing samples with probability equal to their weight (i.e. they are drawn from the discrete distribution represented by the samples). All but Liu and Chen (1995) resample on every iteration. These latter authors introduce a heuristic quantity called the effective sample size, intended to measure the equivalent number of equally weighted particles represented by the current cohort. Once this quantity falls below some value resampling is performed. Carpenter et al. (1999) note that even when resampling does take place any inference should be made using the original samples and weights as resampling introduces additional noise. These authors also note that resampling using stratified sampling will theoretically achieve greater accuracy (in terms of the variance of quantities estimated using the samples, sometimes known as Monte-Carlo variance) than simple sampling.

Using the state transition function as the importance function is often inefficient in simulations as the state space is explored without any knowledge of the [most recent] observations (Doucet et al. (2000)). An obvious alternative, and one that can be shown to minimize the Monte-Carlo variance, is to use the conditional expectation of the state at time t, given the previous state and the new observation at time t. This has been used in e.g. Kong et al. (1994), though the closed-form sequential derivation of such an importance function is only generally feasible for the class of models having nonlinear transition and observation equations subject to Gaussian noise (Doucet et al. (2000)). In order to generalize this method Doucet et al. (2000) propose a local linearization of the state space model to make it locally Gaussian ("in a similar way to the Extended Kalman Filter"). This allows an optimal importance function for the localized model to be derived.

A different approach to improving filter performance is that taken by Pitt and Shephard (1999) and their auxiliary particle filter. The essence of this approach is the use of a biased resampling procedure so that particles with a higher probability of surviving into the future are more likely to be propagated. The particle weights are then chosen to remove bias. Candy (2009) notes that if the process is severely nonlinear or contaminated by high process noise then the performance of the auxiliary particle filter might be worse than that of the bootstrap filter but if this is not the case then it will be less sensitive to outliers and is likely to be superior. Godsill and Clapp (2001) compare the auxiliary and bootstrap filters using a time-varying autoregressive test model and conclude that the auxiliary particle filter gives a slight but systematic improvement in performance.

As far back as Gordon et al. (1993) it was noted that resampling can cause the diversity of particles to be curtailed, especially when there is a poor overlap between the likelihood of the latest observation and the predicted state distribution. This is because during the resampling stage relatively few particles have a very high probability of being resampled as the parents of the next stage, as only a handful lie in the region of interest, and so these are likely to be resampled multiple times, reducing diversity. Gordon et al. (1993) suggested roughening

the samples by adding random noise. A related solution of Musso et al. (2001) is to resample from a regularized distribution rather than the discrete distribution represented by the weighted particles. This is a continuous distribution constructed by placing a kernel function at the location of each of the samples, thus increasing the diversity of the resampled particles. An alternative way of increasing diversity is to use a Markov chain sampling method to draw new samples from the required distribution (see next section).

Worth noting is the idea of marginalization Cappé et al. (2007) of systems of interest, if the system permits, whereby parts of the system that are conditionally linear can be separated from the non-linear parts. The non-linear parts can then be tackled using particle filters whereas exact solutions can be found for the linear Gaussian parts using Kalman filters.

A recent development is the *variable rate* particle filter of Godsill and Vermaak (2005). These filters are designed to work with systems that undergo fairly regular (and therefore easy to filter) state transitions for most of the time, interspersed with significant changes in system evolution. An example of such behaviour is that of self-propelled objects (such as aircraft) that mostly obey standard dynamics, but occasionally execute sharp maneuvers by exerting a large sudden impulse.

Specialized particle filters have also been applied to multi-target tracking in which a group of objects moving together (for example aircraft in formation) are tracked both independently and as a group (e.g. Khan et al. (2005) and Vermaak et al. (2005)). The method has recently been applied to the tracking of a group of stock returns of related stocks by Pang et al. (2008)

2.3.3 MCMC and Particle Filtering

As described above in section 2.2, MCMC methods can be used to generate (dependent) samples from an almost arbitrary distribution that is the stationary distribution of a Markov chain. The technique has several applications in the particle filtering setting.

Following on from the previous section, MCMC methods can be used to increase sample diversity after resampling using an algorithm called resample-move Gilks and Berzuini (2001) and Berzuini and Gilks (2001). This uses an MCMC algorithm to generate a new sample directly from the posterior distribution, using each resampled particle as a starting point for the Markov chain. **?** proposes a slightly different MCMC scheme that does not perform resampling before the move step. In general, since the pre-move samples should come from the posterior distribution only a single iteration of MCMC is required (Cappé et al. (2007)). A problem, especially for multimodal distributions or ones for which the regions of high likelihood of the new observation data are far from the pre-move sample points, is that the chain can take a large number of iterations to explore its state space (mix). Godsill and Clapp (2001) suggest the refinement of creating a sequence of intermediate bridging distributions through annealing of the target distribution, so as to encourage more rapid mixing.

A sequential use of MCMC that does not fit into the weighted sampling framework is given by Khan et al. (2005). This uses an unweighted particle representation of the posterior density of the state given the observations. When new data becomes available it uses an MCMC chain to generate a set of samples from the new posterior distribution. The integration necessary to derive the new

posterior sequentially from the previous one is possible owing to the discrete approximation.

A related method is the so-called Practical filter of Polson et al. (2008). This uses a particle approximation of the smoothing distribution of the state posterior at $t-k$ given observations up to $t-1$, and then runs a number of MCMC chains, starting one at each of the particle positions, to draw samples from the posterior of the state from $t-k+1$ to t.

A very recent algorithm using MCMC is that of **?**. This again uses a particle set to represent state posteriors but, after making an observation at time t, runs an MCMC chain to estimate the joint distribution of the state at t and $t-1$.

The problem is turned on its head somewhat by Del Moral et al. (2006) who use a sequential Monte-Carlo method incorporating MCMC moves as an algorithm to sample from a given distribution.

2.3.4 Parameter Estimation

Particle filters have been particularly successful for certain difficult state space model problems. However, all such models have parameters that dictate the evolution of the system and the generation of its observations. In order to filter observations from a system to reveal the underlying state, tese parameters need to be known and this means estimating them unless they are known in advance.

The simplest way to add parameters into the particle filter framework is to augment the state space with additional elements, θ to represent the parameters. However, as was noted earlier, successive particle representations either degenerate into a single point without resampling or, with resampling, can lose diversity. This is not necessarily a problem in state estimation where successive states are closely linked and observations help tune the filter's estimate, but are difficult to deal with in parameter estimation where each observation only reveals a little about the parameters and excessive early convergence to a value denies the filter an opportunity to fully explore the parameter space. This difficulty was recognized by Gordon et al. (1993), who introduced the idea of adding noise to sampled states in order to reduce degeneracy. West (1993) used kernel methods (i.e. mixture models based around a particular kernel function) as a way to reintroduce variance into state (and parameter) samples. Liu and West (2001) give a generalized version of these algorithms in an auxiliary particle filter, but acknowledge that the method "suffers from the obvious drawback that it "throws away" information about parameters in assuming them to time-varying when they are, in fact, fixed." More exotic approaches include the replacement of the 'prediction' step (where the prior is determined) with mutation and crossover operations from a genetic algorithm Higuchi (1997) and a smoothing approach by Kitagawa (1998). Perhaps a better alternative is provided by MCMC rejuvination of particles using the method of Gilks and Berzuini (2001). This maintains particle diversity using an MCMC chain to generate new samples from the current (particle approximation of the) posterior distribution, and so does so without introducing any artificial parameter dynamics.

Another standard technique for statistical parameter estimation is *maximum likelihood estimation* (MLE). In the particle filter context, this views the filter's posterior distributions as likelihood functions of the observations and parameters, which it then attempts to maximize over the parameters. One problem with this approach is that particle filters are random sampling methods and so the

evaluation of the likelihood is always an approximation and prone to sampling error (Kitagawa (1998)). A further problem is that each run of the particle filter (to evaluate the likelihood function) is computationally expensive, so its use as an objective function in an optimization algorithm can take a long time. Nonetheless, Andrieu and Doucet (2003) use the *Expectation-Maximization* (EM) algorithm to perform MLE estimation using particle filters. Poyiadjis et al. (2005) and Doucet and Tadić (2003) devise a method for estimating the gradient of the particle filter likelihood function and then use gradient ascent methods to do both recursive and batch parameter estimation using MLE.

Other recent ideas for parameter estimation in sequential Monte Carlo methods include the SMC sampler algorithm of Del Moral et al. (2006), which uses a sequential sampling algorithm related to MCMC algorithms in order to draw samples from a sequence of target distributions. This can be used in the same way as an MCMC algorithm to estimate parameters. The practical filter of Polson et al. (2008) uses a rolling window fixed-lag MCMC as a sequential smoother and parameter estimator for each particle in the particle ensemble, giving a diverse set of estimates for the state and parameters at each point. This is similar to the fixed-lag smoothing algorithm of Clapp and Godsill (1999), but uses MCMC to generate diverse samples. Polson et al. (2008) also use the idea of sufficient statistics for the state and parameter space (first applied in the sequential setting by Fearnhead (2002)) in order to reduce the dimensionality of the target distribution being sampled.

Parameter estimation in sequential filtering methods is still an active area of research with several related but distinct approaches proposed in the recent literature.

2.3.5 Theoretical Results

There are a number of theoretical results on the convergence of sequential Monte-Carlo algorithms. Geweke (1989) sets out convergence conditions and numerical accuracy results for Monte-Carlo integration with importance sampling for Bayesian inference, though he does not treat sequential estimation. A difficulty of obtaining theoretical results for sequential methods is that through resampling or MCMC generation the particles interact and so classical limit theorems relying on independence do not apply (Crisan and Doucet (2002)). **?** gives what claim to be the first mathematically well-founded convergence results for interacting particle approximations. Crisan and Doucet (2002) give a number of convergence results and the conditions under which they hold for interacting particle systems. Chopin (2004) gives a central limit theorem for sequential Monte-Carlo methods that applies to a large class of methods "under minimal assumptions on the distributions" and looks at the asymptotic long-term stability of the algorithms.

There have been a number of empirical studies of the performance of particle filter methods, including Pang et al. (2008) and Godsill and Clapp (2001), though these inevitably tend to be based around a single model.

Chapter 3

Model

The model we propose is a simple two-factor model, with a 'value' x and 'trend' $\dot{x}$ process, both of which are mean-reverting with constant volatility. Though this model cannot reproduce many of the features observed in real data (see section 2.1), it could be sufficient for a basic trading application and so is worth investigating. Both processes are subject to two different types of random innovation: Gaussian noise and jumps. Observed prices (y) are modelled as noisy observations of the value process. This leads to the following model

$$d\dot{x} = \lambda_1 \dot{x} dt + \sigma_1 dW_t^{(1)} + dJ_t^{(1)} \tag{3.1}$$

$$dx = \lambda_2 x dt + \sigma_2 dW_t^{(2)} + dJ_t^{(2)} \tag{3.2}$$

$$y = x + \sigma_{obs}\epsilon \tag{3.3}$$

where

$dW_t^{(1)}$, $dW_t^{(2)}$ are independent Gaussian noise processes
$dJ_t^{(1)}$, $dJ_t^{(2)}$ are jump processes (see below)
ϵ is Gaussian noise, distributed $\mathcal{N}(0, 1)$

The jump processes we use have jump times that are distributed as a Poisson distribution and jump sizes that are distributed as a Gaussian. This means for simplicity we can view each jump process as having two parameters, the jump rate, λ_{jump} and the jump size variance σ_{jump}. This assumes that the mean jump size is 0 and that the jumps are not correlated. However, as we shall see, both of these assumptions can easily be relaxed, as can that on the jump time distribution and that on jump size distribution.

In the model thus described there are nine unknown parameters: two mean reversion coefficients (λ_1 and λ_2), two Gaussian noise variances (σ_1 and σ_2), two jump rates (parameters of the respective Poisson processes) ($\lambda_{jump}^{(1)}$ and $\lambda_{jump}^{(2)}$), two jump size variances ($\sigma_{jump}^{(1)}$ and $\sigma_{jump}^{(2)}$) and finally the observation noise variance (σ_{obs}).

The model in equations 3.1 and 3.2 allows some key features that we are interested in to be described (see Figure 3.1). These include changes in trend without a jump in the price, a jump in the price only and a jump in both the price and trend. The inclusion of jumps in the price process can also allow the

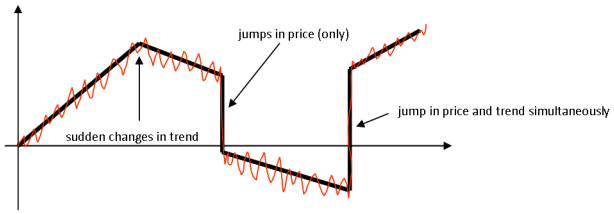

Figure 3.1: Features that can be described using our model

modelling of some types of heavy-tailed return distributions, depending on their distribution.

In its present form the model does not allow for the modelling of stochastic volatility or heteroskadasticity (volatility clustering), which has been observed in real-world financial data. Some form of heteroskedasticity could be incorporated in the model with little additional work by changing the jump time distribution in such a way as to allow jump clustering.

A further obvious improvement to the model would be to introduce mean levels to both the price and trend processes (in the present formulation they have zero means). This would allow a mean level of the price other than zero and would allow a trend that mean reverts to some positive or negative underlying trend value. Care would need to be taken when having a mean reverting price and a non-zero mean reverting trend, as the two interact with each other. For example, consider the zero-noise case with a positive-mean trend but zero-mean price. The equilibrium price of this system is that at which the mean reverting tendency of the price exactly counteracts the trend, and this is not at zero.

3.1 State Space Model

Having devised a model the next problem is fitting it to some data. We would like to be able to fit it to observed data, giving estimates for the state (trend and value) processes with time, as well as estimates for the model parameters. We will approach this problem from a Bayesian perspective, with the aim of applying Bayesian inference techniques (particularly Markov chain Monte Carlo and Particle Filtering - see sections 2.2 and 2.3) to the problem.

Since the evolution of the system in equations 3.1 and 3.2 depends only on its current state and not on any information from further back in time it fits into the Markov class of models. Diagramatically we can represent it as shown in figure 3.2, where each state depends only on the one preceding it and each observation depends only on the current state at the time the observation is made. This figure looks discrete, but can represent our continuous system since we can think of the states shown as being those at the time of the observations and varying continuously between observations.

However, when trying to solve the system (i.e. to extract information about underlying states and parameters, given a series of observations) it can be helpful to think of it as being a discrete system, being in particular states at the observation times (that we would like to discover, or at least gain information about). We can then define a 'transition function' that describes how one state

evolves to the next and an 'observation function', describing how observations derive from the underlying state.

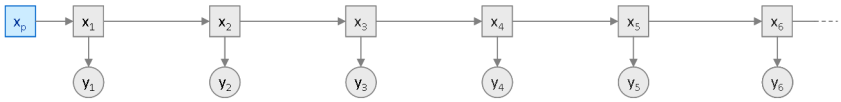

Figure 3.2: Graphical representation of Markov model structure

The system in equations 3.1 to 3.3 can easily be fitted into this way of thinking. The 'value' x and 'trend' $\dot{x}$ processes together are the state of the system and y is its observation, defined at certain observation times. The observation function is then simply the x process with additive Gaussian noise, as defined in equation 3.3. The state transition function is defined by the right hand side of the SDEs in equations 3.1 and 3.2.

Since we want to define a discrete time transition function (e.g. from t to $t+1$) we need to integrate the SDEs to derive the transition function itself. With a generalized jump process included in the SDEs, this is not, in general, possible in closed form. One way to proceed would be to numerically integrate the system and use the resulting approximation as the transition function. For many SDEs this is the only solution, but it has the disadvantage of being inexact and of introducing discretization errors. More importantly it means that solutions to the SDE can no longer be expressed in neat closed form distributions, but rather have to represented by a collection of samples of (approximate) realizations of the SDE evolution over the time period in question. This can lead to a large increase in computation time because thousands of realizations might need to be calculated for each step in order to represent the distribution with sufficient accuracy.

As previously stated, no closed form of the jump system in equations 3.1 and 3.2 exists in general. However, if we remove the jumps the system is Gaussian and relatively simple. It is so simple, in fact, that it can be solved in closed form using Itô calculus (see below). But given that our system does have jumps, does this observation actually help? Well, yes. If we condition on jump times we can effectively think of part of the system as if the jump times are known and then we are able to separate the system into a tractable linear Gaussian bit (between the jumps) and a nonlinear, non-Gaussian bit (the jumps).

One way of thinking of the system conditioned on the jump times is as shown in figure 3.3. Here we can think of the jump times being fixed and given. All the bits of the system between jumps behave according to the system equations with the jumps removed. This allows us to define a transition function for those sections based on the Gaussian-only SDEs, which we can solve. From one side of the jump to the other we have a different transition function, based on the jump-size distribution. That still leaves us with the problem of finding the jump times but this too will turn out to be possible (at least in the sense of estimating their distribution).

Alternatively, we can think of the jump conditioned system as shown in figure 3.4. In this view we only worry about the state at the time of the jumps. This makes the logical structure similar to that in the original Markov model of figure 3.2, only this time the state times are the (unknown) jump times

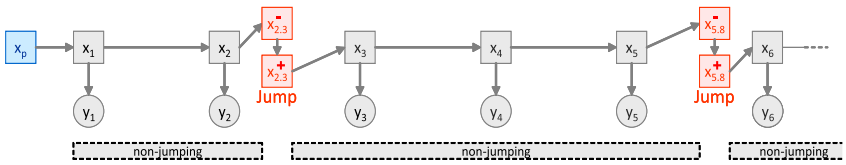

Figure 3.3: Model structure with random jumps

rather than times known *a priori* (regularly spaced in the case of figure 3.2) and the observations dependent on each state are a whole series of the regular observations viewed as a single block. The 'observation' dependent on each state is then a *neighbourhood* of observed values, the limits of which are defined by the preceding and following jump (or the start and end of the data series for the first and last neighbourhoods, respectively). Because it transforms the jumping system into a form very similar to the original Markov formulation of figure 3.2 this alternative view is useful in formulating the methods necessary estimate the state and parameters of the model. It can be viewed as a 'variable rate' version of the original Markov model, with state information being held at variable time points determined by the data rather than at time points fixed in advance.

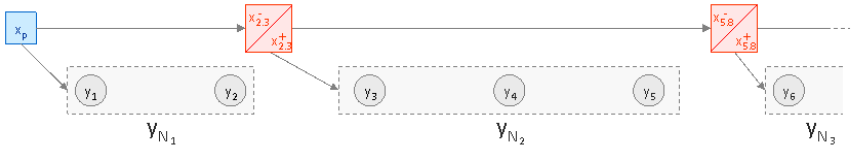

Figure 3.4: The jump-conditioned system considering only the state at jump times. The state might need to be represented both before and after the jump. Note the similarity in structure to the original Markov model in figure 3.2

If the jump size distribution is non-Gaussian, then even if every transition between one jump and the next is Gaussian, the distribution of the state will not be Gaussian between those jumps and might well not have any closed-form representation. One approach to get around this is to use a sample-based representation of the post-jump distribution. In that case we have a number of samples that approximate the post-jump state distribution (found by sampling from the jump-size distribution). Each of these is a point, rather than a distribution, and so feeding them through the Gaussian section will result in a Gaussian distribution for each of them at the end of the section. This can then be sampled, along with the next jump distribution to generate a starting sample for the following section, etc., until we reach the end of the series. An alternative approach, that we will return to later, is to approximate the non-Gaussian jump-size distribution with a Gaussian mixture model, which, going through a series of Gaussian transitions remains a Gaussian mixture. In our initial work we use a Gaussian jump-size distribution, meaning that the state distributions remain Gaussian throughout.

3.2 State Transition Functions

To define a system of the type depicted in figures 3.2, 3.3 and 3.4 the meaning of the arrows in the diagram must be defined. Each arrow represents a function that maps from a state (or prior distribution) to either another state, in which case we call it a *state transition function* or to an observable manifestation of the system, in which case we call it an *observation function.*

The model that we are interested in defining is that defined by equations 3.1-3.3 and depicted in figures 3.3 and 3.4. The representation of the former figure breaks apart the jump and non-jump state transitions and treats every observation as the direct observation of an underlying state defined at the same time as the observation. This is the easiest set-up on which to start to define the transition and observation functions. We will then extend this to deal with the variable rate formulation of figure 3.4.

3.2.1 Notation

First, let us define some notation.

- Let τ_i be the time of the i^{th} jump.
- Let X_t be the state of the system at time t (i.e. X_t incorporates both x_t and $\dot{x}_t$).
- Let X_t^- and X_t^+ represent the system state instantaneously before and after t, respectively. At times other than jump times, $X_t^- = X_t^+$, whereas at jump times X_t^- is the state just before the jump and X_t^+ is the state just after the jump. We can define these precisely as $X_t^- = \lim_{s \to t^-} X_s$ and $X_t^+ = \lim_{s \to t^+} X_s$.
- Let y_t be the observation of the system (defined by equation 3.3) at time t.

3.2.2 Jump Transition

The jump transition in the model (i.e. the transition between $X_{\tau_i}^-$ and $X_{\tau_i}^+$ is easy to define. It is Gaussian of the form

$$p(X_{\tau_i}^+ | X_{\tau_i}^-) \sim \mathcal{N}\left(X_{\tau_i}^-, \begin{pmatrix} \sigma_{jump}^{(1)} & 0 \\ 0 & \sigma_{jump}^{(2)} \end{pmatrix} \right). \tag{3.4}$$

There is one complication: is a 'jump' a jump in the x value process, the $\dot{x}$ trend process, or both? This is a modelling question and we choose to define three possible jump types: an x jump, in which we replace $\sigma_{jump}^{(1)}$ in equation 3.4 with zero, an $\dot{x}$ jump, in which we replace $\sigma_{jump}^{(2)}$ with zero and a co-located jump in both x and $\dot{x}$, which takes the form shown in equation 3.4. This could easily be extended to use a Gaussian mixture model for the jump distribution. It is also straightforward to give the jumps a non-zero mean size, allowing either positive or negative jumps to be larger and more common.

The other jump related aspect of the model is the jump-time distribution. This can be chosen to be any distribution, but in this work we use the Poisson distribution.

$$\tau_i \sim \text{Poisson}(\lambda_{jump}) \tag{3.5}$$

For financial data a more sophisticated jump-time distribution allowing for jump clustering might be more appropriate.

3.2.3 Non-jump transition

The non-jumping portion of the model is defined by the system of equations 3.1 and 3.2 with the jump processes removed. This system can be re-written in matrix-vector form as

$$\begin{pmatrix} dx \\ d\dot{x} \end{pmatrix} = \begin{pmatrix} \lambda_1 & 1 \\ \lambda_2 & 0 \end{pmatrix} \begin{pmatrix} x \\ \dot{x} \end{pmatrix} + \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix} \begin{pmatrix} dW_1 \\ dW_2 \end{pmatrix} \tag{3.6}$$

For notational simplicity, define

$$X = \begin{pmatrix} x \\ \dot{x} \end{pmatrix}, \quad A = \begin{pmatrix} \lambda_1 & 1 \\ \lambda_2 & 0 \end{pmatrix}, \quad b = \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix}, \tag{3.7}$$

and note that the Brownian motion components dW_1 and dW_2 are independent. The solution to this system (from time 0 to T) is given by the following stochastic integral. This is derived by solving the above system of SDEs (some guidance on doing this is given in **?**).

$$X_T = e^{AT} \left[X_0 + \int_0^T e^{-At} b dW \right], \tag{3.8}$$

where e^{At} is the matrix exponential defined as

$$e^{At} = I + At + \frac{(At)^2}{2!} + \frac{(At)^3}{3!} + ..., \tag{3.9}$$

analogously to the scalar exponential.

If the matrix A is diagonalizable (distinct eigenvalues are a sufficient condition for this), we can calculate these powers efficiently since

$$A = PDP^{-1}, \tag{3.10}$$

where D is a diagonal matrix, so

$$A^2 = PDP^{-1}PDP^{-1} = PDDP^{-1} = PD^2P^{-1}. \tag{3.11}$$

Since D is a diagonal matrix, powers of it are simply diagonal matrices with the non-zero elements raised to the appropriate power. Using the definition of the matrix exponential from equation 3.9 we obtain

$$e^{At} = I + PDP^{-1}t + \frac{PD^2P^{-1}t^2}{2!} + \frac{PD^3P^{-1}t^3}{3!} + ... \tag{3.12}$$

$$= P\left(I + Dt + \frac{D^2t^2}{2!} + \frac{D^3t^3}{3!} + ... \right) P^{-1} \tag{3.13}$$

$$= Pe^{Dt}P^{-1} \tag{3.14}$$

The matrix exponential of a diagonal matrix is itself a diagonal matrix, as can be seen from equation 3.9, since each power of a diagonal matrix is diagonal and e^{Dt} is simply a weighted sum of such matrices, so

$$e^{Dt} = \begin{pmatrix} e^{d_1 t} & & & \\ & e^{d_2 t} & & \\ & & \ddots & \\ & & & e^{d_N t} \end{pmatrix}, \tag{3.15}$$

where d_1 to d_N are the diagonal entries of D.

If X_0 is deterministic or Gaussian distributed then the solution of the system given in equation 3.8 is itself Gaussian distributed. We can see this because e^{AT} is deterministic if A is, and under reasonable assumptions, which the integrand in equation 3.8 meets, the stochastic integral is itself a normally distributed random variable, with mean 0. It can be seen intuitively by thinking of the stochastic integral as the (limit of the) weighted sum of a series of Gaussian random variables (since the increments of the Brownian motion dW are themselves Gaussian), which therefore has a Gaussian distribution, since a sum of Gaussians is itself Gaussian. Gaussian distributions can be determined fully by their first two moments, so if the right hand side of equation 3.8 is Gaussian, we just need to find its expectation and covariance.

$$\mathbb{E}(X_T) = \mathbb{E}(e^{AT} X_0) + \mathbb{E} \int_0^T e^{At} b dW \tag{3.16}$$

$$= e^{AT} \mathbb{E}(X_0) \tag{3.17}$$

The variance-covariance of X_T is given by

$$\operatorname{cov}(X_T) = \mathbb{E}\left((X_T - \mathbb{E}(X_T)) (X_T - \mathbb{E}(X_T))' \right) \tag{3.18}$$

$$= \mathbb{E}\left(X_T X_T' - 2 X_T \mathbb{E}(X_T)' + \mathbb{E}(X_T) \mathbb{E}(X_T)' \right) \tag{3.19}$$

$$= \mathbb{E}(X_T X_T') - 2\mathbb{E}(X_T) \mathbb{E}(X_T)' + \mathbb{E}(X_T) \mathbb{E}(X_T)' \tag{3.20}$$

$$= \mathbb{E}(X_T X_T') - \mathbb{E}(X_T) \mathbb{E}(X_T)' \tag{3.21}$$

$$= \mathbb{E}\left(e^{AT} \left[X_0 + \int_0^T e^{-At} b dW \right] \left(e^{AT} \left[X_0 + \int_0^T e^{-At} b dW \right] \right)' \right) - e^{AT} \mathbb{E}(X_0) (e^{AT} \mathbb{E}(X_0))' \tag{3.22}$$

$$= e^{AT} \left[\mathbb{E}\left\{ X_0 X_0' + X_0 \left(\int_0^T e^{-At} b dW \right)' + \left(\int_0^T e^{-At} b dW \right) X_0' + \left(\int_0^T e^{-At} b dW \right) \left(\int_0^T e^{-At} b dW \right)' \right\} - \mathbb{E}(X_0) \mathbb{E}(X_0)' \right] (e^{AT})'. \tag{3.23}$$

By independence of X_0 and $\int_0^T e^{-At} dW$, and the fact that the latter integral has an expectation of 0 in all components,

$$\mathbb{E}\left(X_0 \left(\int_0^T e^{-At} b dW \right)' \right) = \mathbb{E}(X_0) \mathbb{E}\left(\int_0^T e^{-At} b dW \right), \\ = 0. \tag{3.24}$$

Clearly this also applies to the transpose of this term, which appears in equation 3.23. Noting also that

$$\mathbb{E}\left(X_0 X_0'\right) - \mathbb{E}\left(X_0\right)\mathbb{E}\left(X_0\right)' = \mathrm{cov}(X_0), \tag{3.25}$$

we can simplify equation 3.23 to

$$\mathrm{cov}(X_T) = e^{AT}\left[\mathbb{E}\left(\left(\int_0^T e^{-At} b dW\right)\left(\int_0^T e^{-At} b dW\right)'\right) + \mathrm{cov}(X_0)\right](e^{AT})'. \tag{3.26}$$

This just leaves us to deal with a single expectation, but it is the most complicated as it involves stochastic integrals. Since the stochastic integral $\int_0^T e^{-At} b dW$ is a (column) vector, the term inside the expectation is a matrix, which we will denote Q. To simplify notation, let the matrix $B = e^{-At} b$. Then consider the i^{th} component (notated $(.)_i$) of the stochastic integral

$$\left(\int_0^T e^{-At} b dW\right)_i = \sum_j \int_0^T B_{ij} dW_j, \tag{3.27}$$

where dW_j is the j^{th} component of the multidimensional Brownian motion process represented by dW. So, the $(i,j)^{th}$ component of the matrix Q is given by

$$Q_{ij} = \left(\sum_m \int_0^T B_{im} dW_m\right)\left(\sum_n \int_0^T B_{jn} dW_n\right). \tag{3.28}$$

We are interested in the expectation of this. Since the integrals are driven by components of Brownian motion which are assumed to be independent then they themselves are independent zero-mean Gaussian random variables and the expectation of their product is zero, except in the case where $m = n$, i.e. where both integrals are driven by the same component of the Brownian motion. This means that when we take expectations, all the cross terms equal zero and we are only left with the square terms, so

$$\mathbb{E}\; Q_{ij} = \mathbb{E}\left(\sum_m \left\{\int_0^T B_{im} dW_m \int_0^T B_{jm} dW_m\right\}\right). \tag{3.29}$$

Now, considering just one of the integral products in equation 3.29 and writing the stochastic integrals as limits of sums, we get

$$\mathbb{E}\int_0^T B_{im} dW_m \int_0^T B_{jm} dW_m = \mathbb{E}\left(\lim_{\mathrm{mesh}(\tau)\to 0}\left(\sum_k B_{im}(\tau_k)\Delta W_{m,\tau_k}\right)\left(\sum_k B_{jm}(\tau_k)\Delta W_{m,\tau_k}\right)\right), \tag{3.30}$$

where τ is a division of the period from 0 to T such that $0 = \tau_0 < \tau_1 < ... < \tau_N = T$, and $\mathrm{mesh}(\tau)$ is the maximum spacing between adjacent points. $\Delta W_m, \tau_k$ is the change in the m^{th} component of the multivariate Brownian motion from τ_{k-1} to τ_k.

Taking the product of the sums, putting the expectation inside the summation and noting that all terms contain

$$\mathbb{E}\left(\Delta W_{m,\tau_k} W_{m,\tau_l}\right) = \begin{cases} 0 & k \neq l \\ \tau_k - \tau_{k-1} & k = l \end{cases}, \tag{3.31}$$

we get

$$\mathbb{E}\int_0^T B_{im} dW_m \int_0^T B_{jm} dW_m = \lim_{\text{mesh}(\tau_i)\to 0} \sum_k B_{im}(\tau_k) B_{jm}(\tau_k)(\tau_k - \tau_{k-1}) \tag{3.32}$$

$$= \int_0^T B_{im} B_{jm} dt. \tag{3.33}$$

We have now succeeded in expressing the stochastic integral in terms of a deterministic one. Putting this back into equation 3.29, we obtain

$$\mathbb{E}\, Q_{ij} = \sum_m \int_0^T B_{im} B_{jm} dt \tag{3.34}$$

$$= \left(\int_0^T BB' dt\right)_{ij}. \tag{3.35}$$

Replacing Q and B by their definitions, we get

$$\mathbb{E}\left(\int_0^T e^{-At} b dW\right)\left(\int_0^T e^{-At} b dW\right)' = \int_0^T e^{-At} bb' (e^{-At})' dt \tag{3.36}$$

This is actually a multivariate instance of the Itô isometry, applied to a deterministic process.

Putting this back into the expression for the covariance of X_T in equation 3.26 gives the deterministic expression

$$\text{cov}(X_T) = e^{AT}\left[\int_0^T e^{-At} bb' (e^{-At})' dt + \text{cov}(X_0)\right](e^{AT})'. \tag{3.37}$$

Because X_T is Gaussian this, along with the expression for the expectation in X_T in equation 3.17, fully determines the distribution of X_T.

We can introduce correlation between the underlying Brownian motion components that drive the system in equation 3.6 by replacing b with the Cholesky decomposition of the required covariance matrix Σ_{bm}, so that $bb' = \Sigma_{bm}$.

Coming up with expressions for the distribution of X_T allows us to define a transition function for the Gaussian (inter-jump) portion of the process. To define this we need to know the distribution of the state at time s given that we know the distribution of the state at time r. To use the above derivation we simply use $(s - r)$ in place of the time step T, and use X_r in place of the starting point X_0. The above derivation assumed that the starting point X_0 was a random variable, but when calculating the conditional probability of X_s *given* X_r, we can assume that we know X_r as a deterministic quantity. Since $\mathbb{E}(X_r) = X_r$ if X_r is deterministic, the expectation operator disappears from

the right hand side of 3.17. Furthermore, the $\text{cov}(X_0)$, calculated in equation 3.25 reduces to 0 (since a deterministic quantity has no variance). The transition function is therefore given by

$$p(X_s|X_r) \sim \mathcal{N}\left(e^{A(s-r)}X_r,\ e^{A(s-r)}\left[\int_0^{s-r} e^{-At}bb'(e^{-At})'dt\right](e^{A(s-r)})'\right). \tag{3.38}$$

3.2.4 Calculating the covariance matrix

In order to calculate the covariance matrix of the above transition density, we must calculate the integral

$$\Sigma^{(T)}_{\text{diffuse}} = \int_0^T e^{-At}bb'(e^{-At})'dt. \tag{3.39}$$

where the parameter T determines the length of the diffusion period.

If the matrix A is diagonalizable, then we can write it as

$$A = UDU^{-1}, \tag{3.40}$$

where D is a diagonal matrix. Then, as in equation 3.14 we can write

$$e^{-At} = e^{-UDU^{-1}t}, \tag{3.41}$$
$$= Ue^{-Dt}U^{-1}, \tag{3.42}$$

where e^{-Dt} is a diagonal matrix of the form of equation 3.15. So, putting this into equation 3.39, we get

$$\Sigma^{(T)}_{\text{diffuse}} = \int_0^T (Ue^{-Dt}U^{-1}b)(Ue^{-Dt}U^{-1}b)'dt \tag{3.43}$$
$$= \int_0^T Ue^{-Dt}U^{-1}bb'(U^{-1})'e^{-Dt}U^{-1}dt \tag{3.44}$$
$$= U\left(\int_0^T e^{-Dt}C'e^{-Dt}dt\right)U^{-1}, \tag{3.45}$$

where $C = U^{-1}bb'(U^{-1})$. The last step above is possible because the matrix A and thus U does not depend on t in our system. Now,

$$e^{-Dt}\begin{pmatrix} -- C_{1\cdot} -- \\ -- C_{2\cdot} -- \\ \vdots \\ -- C_{n\cdot} -- \end{pmatrix} = \begin{pmatrix} -- e^{-d_1 t}C_{1\cdot} -- \\ -- e^{-d_2 t}C_{2\cdot} -- \\ \vdots \\ -- e^{-d_n t}C_{n\cdot} -- \end{pmatrix} = E, \tag{3.46}$$

where $C_{1\cdot}$ to $C_{n\cdot}$ are the rows of C and

$$e^{-Dt}Ce^{-Dt} = Ee^{-Dt} = \begin{pmatrix} | & | & & | \\ e^{-d_1 t}E_{\cdot 1} & e^{-d_2 t}E_{\cdot 2} & \cdots & e^{-d_n t}E_{\cdot n} \\ | & | & & | \end{pmatrix} \tag{3.47}$$

$$= \begin{pmatrix} e^{-d_1 t}e^{-d_1 t}C_{11} & e^{-d_1 t}e^{-d_2 t}C_{12} & \cdots \\ e^{-d_2 t}e^{-d_1 t}C_{21} & \ddots & \\ \vdots & & \end{pmatrix}, \tag{3.48}$$

i.e.

$$(e^{-Dt}Ce^{-Dt})_{ij} = e^{-(d_i+d_j)t}C_{ij}. \tag{3.49}$$

Now, we actually want to calculate the integral of this, so let $F = \int_0^T (e^{-Dt}Ce^{-Dt})dt$, then

$$F_{ij} = \int_0^T (e^{-Dt}Ce^{-Dt})_{ij}dt \tag{3.50}$$

$$= C_{ij}\int_0^T e^{-(d_i+d_j)t}dt \tag{3.51}$$

$$= \frac{-C_{ij}}{d_i+d_j}\left[e^{-(d_i+d_j)t}\right]_0^T \tag{3.52}$$

$$= \frac{-C_{ij}}{d_i+d_j}\left[e^{-(d_i+d_j)T} - 1\right]. \tag{3.53}$$

The second step above only works on the condition that $(d_i + d_j) \neq 0$. If $(d_i + d_j) = 0$ then $F_{ij} = C_{ij}\int_0^T dt = C_{ij}T$. This expression for F allows us to calculate $\Sigma^{(T)}_{\text{diffuse}}$ when A is diagonalizable, which is very often the case (distinct eigenvalues are sufficient to ensure this), as

$$\Sigma^{(T)}_{\text{diffuse}} = UFU^{-1}. \tag{3.54}$$

In the case that A is not diagonalizable then we can still calculate this integral by putting A into Jordan normal form (where the Jordan matrix J has entries only on the diagonal and on the first superdiagonal with those on the superdiagonal only being 1 or 0)

$$A = UJU^{-1} \tag{3.55}$$

with

$$J = D + N, \tag{3.56}$$

where D is a diagonal matrix and N is a nilpotent matrix such that $N^{k+1} = 0$ for some $k \in \mathbb{Z}^+$. Analogous to equation 3.42, we can write

$$e^{-At} = Ue^{-(D+N)t}U^{-1} \tag{3.57}$$

$$= Ue^{-Dt}e^{-Nt}U^{-1}. \tag{3.58}$$

The last step is only possible if D and N commute, but this is always the case, since D is diagonal. Now,

$$e^{-Nt} = I - Nt + \frac{N^2t^2}{2!} - \frac{N^3t^3}{3!} + \cdots + (-1)^k\frac{N^kt^k}{k!} \tag{3.59}$$

is a finite series since N is nilpotent. In the 2×2 case, as in our system, $N = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$ and so $N^2 = \mathbf{0}$, meaning that

$$e^{-Nt} = I - Nt. \tag{3.60}$$

Following the working as in the diagonalizable case, we can get an expression for $\Sigma^{(T)}_{\text{diffuse}}$ similar to that in equation 3.54:

$$\Sigma^{(T)}_{\text{diffuse}} = U \left(\int_0^T e^{-Dt} e^{-Nt} C' e^{-Dt} (e^{-Nt})' dt \right) U^{-1}, \tag{3.61}$$

with $C = U^{-1} b b' (U^{-1})'$ as before. We need to calculate the inner integral, so let

$$L = \int_0^T e^{-Dt} e^{-Nt} C' e^{-Dt} (e^{-Nt})' dt \tag{3.62}$$

In the 2×2 case, this becomes (letting $E = e^{-Dt} C e^{-Dt}$)

$$\begin{aligned} L &= \int_0^T (I - Nt) E (I - Nt)' dt && (3.63) \\ &= \int_0^T E - NtE - E(Nt)' + Nt^2 EN' dt && (3.64) \\ &= \int_0^T E dt - N \int_0^T tE dt - \left(\int_0^T tE dt \right) N' + N \left(\int_0^T t^2 E dt \right) N'. && (3.65) \end{aligned}$$

The first of these integrals is the F matrix as in 3.50. If we call the second one G and the final one H, then we can find expressions for their components through integration by parts. This yields

$$G_{ij} = \frac{-C_{ij}}{(d_i + d_j)^2} \left[((d_i + d_j)T + 1) e^{-(d_i + d_j)T} - 1 \right], \tag{3.66}$$

and

$$H_{ij} = \frac{-C_{ij}}{(d_i + d_j)^3} \left[((d_i + d_j)T((d_i + d_j)T + 2) + 2) e^{-(d_i + d_j)T} - 2 \right], \tag{3.67}$$

where d_i is the i^{th} diagonal entry of D, and $d_i + d_j \neq 0$. In the latter case

$$G_{ij} = C_{ij} \frac{T^2}{2}, \tag{3.68}$$

and

$$H_{ij} = C_{ij} \frac{T^3}{3}. \tag{3.69}$$

We can now express $\Sigma^{(T)}_{\text{diffuse}}$ for the 2×2 case in terms we can calculate:

$$\Sigma^{(T)}_{\text{diffuse}} = U(F - NG - GN' + NHN')U^{-1}, \tag{3.70}$$

and, in fact, this includes the diagonalizable case, since in that case, $N = 0$ and we get the same expression for $\Sigma^{(T)}_{\text{diffuse}}$ as we had before in equation 3.54.

The extension to the $n \times n$ case simply requires more terms in the expansion of e^{Nt} given in equation 3.60.

3.3 Observation Function

The observation function gives the probability of seeing a particular observation, given that we are in a particular state. In our system, the observation function is defined by equation 3.3, which states how an observation at time t, y_t derives from the state at time t, X_t. Since the observation is the value process x_t perturbed by Gaussian noise, the observation function involving X_t and y_t is simply the univariate Gaussian distribution given by

$$p(y_t|X_t) \sim \mathcal{N}(x_t, \sigma^2_{obs}). \tag{3.71}$$

In a standard model, such as that shown in figure 3.2, this is sufficient, but we want to consider a model in the form of figure 3.4 with states only recorded at jump times. In this case we need an observation function that gives the probability of a neighbourhood of observations given the state (or its distribution) at the jump time. The neighbourhood for a particular jump consists of all observations occurring after that jump but before the following one. This means that it is impossible to define this function without knowing the following jump time, as the end point of the neighbourhood of observations dependent on a particular state depends on the time of the successor state. If there is no successor state then the neighbourhood runs to the end of the time series.

The problem that we are therefore trying to solve is that of finding the probability of a neighbourhood of observations given what we know about the state at the jump times.

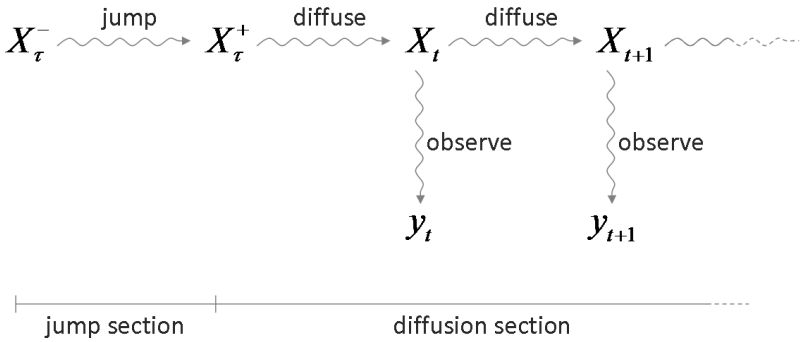

Figure 3.5: The problem of finding an observation function in the variable rate model

If we assume that the jump-size is Gaussian, then figure 3.5 shows that the process that generates the observations from the state before the jump X_τ^- is a series of Gaussian steps: a jump from X_τ^- to X_τ^+, a diffusion from X_τ^+ to X_t, Gaussian noise from X_t to Y_t, a diffusion from X_t to X_{t+1}, Gaussian noise from X_{t+1} to Y_{t+1}, etc. This means that on the condition that the distribution of X_τ^- is Gaussian, the observations are Gaussian and the distribution of the series of observations must be a multivariate Gaussian of dimensionality equal to the number of observations in the neighbourhood. This reduces the problem to that of calculating the expectation and covariance of this multivariate Gaussian,

which gives the pdf of the observations, given the pre-jump state (or its distribution). In order to make the derivation as general as possible (and because we will need it later), we will assume that we only have distributional information about X_τ^-, but that it is Gaussian and we know its mean and covariance.

The model can be extended to encompass jump-sizes and pre-jump states distributed according to Gaussian mixture models (GMMs), since the result of applying a series of Gaussian or GMM steps to a GMM is another GMM. This is because we can apply exactly the same logic as in the derivation of the observation function to each component of the GMM, yielding an expectation and covariance for observations deriving from that component. The final model is then a mixture of these Gaussians in the same ratio as in the original GMM. If the pre-jump or jump-size distribution were to be GMM, then the resulting observation function will simply be a multivariate GMM.

From the state transition derivation we know that from a Gaussian distributed initial state X_0 the state distribution after time T of diffusion is a Gaussian distribution specified by equations 3.17 and 3.37 for the expectation and covariance, respectively. When calculating the observation function, the distribution of the state at the start of the diffusion section is given by the initial distribution, followed by the jump distribution. The distribution of the state at the start of the diffusion section is therefore given by the distribution before the jump, plus the jump-size distribution. This is the sum of two independent multivariate Gaussians and the following lemma is useful.

Lemma 1. *The sum of two independent multivariate Gaussian distributions* $X \sim \mathcal{N}(\mu_X, \Sigma_X)$ *and* $Y \sim \mathcal{N}(\mu_Y, \Sigma_Y)$ *is a multivariate Gaussian distribution distributed* $X + Y \sim \mathcal{N}(\mu_X + \mu_Y, \Sigma_X + \Sigma_Y)$.

This allows us to derive the distribution of the state at the start of the diffusion section, X_τ^+

$$X_\tau^+ \sim \mathcal{N}\left(\mu_{X_\tau^-} + \mu_{\text{jump}}, \Sigma_{X_\tau^-} + \Sigma_{\text{jump}}\right). \tag{3.72}$$

where $\mu_{X_\tau^-}$ and $\Sigma_{X_\tau^-}$ are the mean and covariance of the state before the jump X_τ^- and μ_{jump} and Σ_{jump} are the mean and covariance of the jump-size distribution.

We can now apply the derivation of the transition function in the diffusion section, given in equations 3.17 and 3.37 to state the distribution of the state at a time $\tau < t < \tau'$, where τ' is the time of the next jump after that at τ

$$X_t \sim \mathcal{N}\left(e^{AT}(\mu_{X_\tau^-} + \mu_{\text{jump}}), e^{AT}\left[\Sigma_{\text{diffuse}}^{(t-\tau)} + \Sigma_{X_\tau^-} + \Sigma_{\text{jump}}\right](e^{AT})'\right) \tag{3.73}$$

where $\Sigma_{\text{diffuse}}(t-\tau)$ is defined as the integral in equation 3.39, using limits τ to t and $T = t - \tau$.

Now, we want to calculate the distribution of the neighbourhood of observations $y_N = \{y_i : \tau \leq i \leq \tau'\}$ given the pre-jump state X_τ^-. Since the observation is an observation of the value process x_t (the first element of the X_t state vector) perturbed by Gaussian noise, each individual observation has a Gaussian distribution (since it is the sum of the Gaussian distributed state process and

Gaussian noise). We need to calculate the expected value and autocovariances of these observations. The expected value of each of the observations is simply

$$\begin{aligned}\mathbb{E}(y_t) &= \mathbb{E}\left(x_t + \epsilon_{obs}\right) && (3.74)\\ &= \mathbb{E}\left(x_t\right) && (3.75)\\ &= (e^{AT}(\mu_{X_\tau^-} + \mu_{\text{jump}}))_1 && (3.76)\end{aligned}$$

where ϵ_{obs} is the Gaussian observation noise from equation 3.3 and $(\cdot)_1$ means the first component of a vector.

We now want to calculate the autocovariances $\text{cov}(y_t, y_s)$ for all observations in the neighbourhood y_N. Since we do not have an expression for x_t other than as the first component in the X_t vector, we introduce Z_t, a dummy observation of both x_t and $\dot{x}_t$ perturbed by mean 0, independent Gaussian noise

$$Z_t = X_t + \epsilon_{obs}^{(t)}, \tag{3.77}$$

with

$$\epsilon_{obs}^{(t)} \sim \mathcal{N}\left(\mathbf{0}, \Sigma_{obs}^{(t)}\right), \tag{3.78}$$

where

$$\Sigma_{obs}^{(t)} = \begin{pmatrix}\sigma_{obs}^2 & 0\\ 0 & \sigma_{obs\dot{x}}^2\end{pmatrix} \tag{3.79}$$

Then, letting J_τ be the jump size at time τ, so that $X_\tau^+ = X_\tau^- + J_\tau$,

$$\begin{aligned}\mathbb{E}\left(Z_t\right) &= \mathbb{E}\left(X_t\right) && (3.80)\\ &= e^{AT}\mathbb{E}\left(X_\tau^+\right) && (3.81)\\ &= e^{AT}\mathbb{E}\left(X_\tau^- + J_\tau\right) && (3.82)\\ &= e^{AT}(\mu_{X_\tau^-} + \mu_{\text{jump}}). && (3.83)\end{aligned}$$

We now want to calculate $\text{cov}(Z_t, Z_s)$. This is a covariance matrix giving

$$\text{cov}(Z_t, Z_s) = \begin{pmatrix}\text{cov}((Z_t)_1, (Z_s)_1) & \text{cov}((Z_t)_1, (Z_s)_2)\\ \text{cov}((Z_t)_2, (Z_s)_1) & \text{cov}((Z_t)_2, (Z_s)_2)\end{pmatrix} \tag{3.84}$$

where $(Z_t)_1$ is the first component of Z_t (i.e. y_t), etc. Of the entries in this matrix, it is just the first that we are interested in, since this is $\text{cov}(y_t, y_s)$. Assuming, without loss of generality, that $s \leq t$,

$$\begin{aligned}\text{cov}(Z_t, Z_s) &= \mathbb{E}\left((Z_t - \mathbb{E}\left(Z_t\right))(Z_s - \mathbb{E}\left(Z_s\right))'\right) && (3.85)\\ &= \mathbb{E}\left(Z_t Z_s' - Z_t\mathbb{E}\left(Z_s\right)' - \mathbb{E}\left(Z_t\right)Z_s' + \mathbb{E}\left(Z_t\right)\mathbb{E}\left(Z_s\right)'\right) && (3.86)\\ &= \mathbb{E}\big[(X_t + \epsilon_{obs}^{(t)})(X_s + \epsilon_{obs}^{(s)})' - (X_t + \epsilon_{obs}^{(t)})\mathbb{E}\left(X_s\right)'\\ &\qquad - \mathbb{E}\left(X_t\right)\left(X_s + \epsilon_{obs}^{(s)} + \mathbb{E}\left(X_t\right)\mathbb{E}\left(X_s\right)\right] && (3.87)\\ &= \mathbb{E}\big[X_t X_s' + X_t\epsilon_{obs}^{(s)\prime} + \epsilon_{obs}^{(t)}X_s' + \epsilon_{obs}^{(t)}\epsilon_{obs}^{(s)\prime} - X_t\mathbb{E}\left(X_s\right)'\\ &\qquad + \epsilon_{obs}^{(t)}\mathbb{E}\left(X_s\right)' - \mathbb{E}\left(X_t\right)X_s' - \mathbb{E}\left(X_t\right)\epsilon_{obs}^{(s)} + \mathbb{E}\left(X_t\right)\mathbb{E}\left(X_s\right)'\big]. && (3.88)\end{aligned}$$

Since $\epsilon_{obs}^{(t)}$ and $\epsilon_{obs}^{(s)}$ are random, mean 0 variables, independent of all others, the expectation of all terms involving them is 0, except when $t = s$, so

$$\begin{aligned}\operatorname{cov}(Z_t, Z_s) &= \mathbb{E}\big[X_t X_s' - X_t \mathbb{E}\left(X_s\right)' - \mathbb{E}\left(X_t\right) X_s' + \mathbb{E}\left(X_t\right)\mathbb{E}\left(X_s\right)'\big] \\ &\quad + \mathbf{1}_{\{t=s\}}\mathbb{E}\left(\epsilon_{obs}^{(t)}\epsilon_{obs}^{(s)\prime}\right) \qquad (3.89)\\ &= \mathbb{E}\left(X_t X_s'\right) - \mathbb{E}\left(X_t \mathbb{E}\left(X_s\right)'\right) - \mathbb{E}\left(\mathbb{E}\left(X_t\right) X_s'\right) + \mathbb{E}\left(X_t\right)\mathbb{E}\left(X_s\right)' \\ &\quad + \mathbf{1}_{\{t=s\}}\mathbb{E}\left(\epsilon_{obs}^{(t)}\epsilon_{obs}^{(s)\prime}\right). \qquad (3.90)\end{aligned}$$

Since the expectations are all taken at the same time, the expectations $\mathbb{E}\left(X_t\right)$ and $\mathbb{E}\left(X_s\right)$ that occur inside other expectations can be removed outside. Furthermore, $\mathbb{E}\left(\epsilon_{obs}^{(t)}\epsilon_{obs}^{(s)\prime}\right) = \Sigma_{obs}^{(t)}$, so we get

$$\operatorname{cov}(Z_t, Z_s) = \mathbb{E}\left(X_t X_s'\right) - \mathbb{E}\left(X_t\right)\mathbb{E}\left(X_s\right)' + \mathbf{1}_{\{t=s\}}\Sigma_{obs}^{(t)}. \qquad (3.91)$$

To make further progress, we must expand X_t and X_s according to the solution in equation 3.8. In the case of X_t, this expansion is

$$X_t = e^{AT}\left[X_\tau^+ + \int_0^T e^{-A\tilde{t}} b dW\right], \qquad (3.92)$$

with $T = t - \tau$ and $\tilde{t}$ denotes time in the integral. Defining $S = s - \tau$ gives a similar result for X_s. By taking expectations of equation 3.92 and using the equivalence in equation 3.81 we can also note that

$$\mathbb{E}\left(X_t\right)\mathbb{E}\left(X_s\right)' = e^{AT}\mathbb{E}\left(X_\tau^+\right)\mathbb{E}\left(X_\tau^+\right)'(e^{AS})'. \qquad (3.93)$$

Putting both of these facts into equation 3.91 gives

$$\begin{aligned}\operatorname{cov}(Z_t, Z_s) &= \mathbb{E}\left\{e^{AT}\left(X_\tau^+ + \int_0^T e^{-A\tilde{t}} b dW\right)\left[e^{AS}\left(X_\tau^+ + \int_0^S e^{-A\tilde{t}} b dW\right)\right]'\right\} \\ &\quad - e^{AT}\mathbb{E}\left(X_\tau^-\right)\mathbb{E}\left(X_\tau^-\right)'(e^{AS})' + \mathbf{1}_{\{t=s\}}\Sigma_{obs}^{(t)} \qquad (3.94)\\ &= e^{AT}\Bigg\{\mathbb{E}\left(X_\tau^+ X_\tau^{+\prime}\right) - \mathbb{E}\left(X_\tau^+\right)\mathbb{E}\left(X_\tau^+\right)' \\ &\quad + \mathbb{E}\left(X_\tau^+\left[\int_0^S e^{-A\tilde{t}} b dW\right]'\right) + \mathbb{E}\left(\int_0^T e^{-A\tilde{t}} b dW (X_\tau^+)'\right) \\ &\quad + \mathbb{E}\left(\int_0^T e^{-A\tilde{t}} b dW \left[\int_0^S e^{-A\tilde{t}} b dW\right]'\right)\Bigg\}(e^{AS})' + \mathbf{1}_{\{t=s\}}\Sigma_{obs}^{(t)}. \qquad (3.95)\end{aligned}$$

The first two terms inside the brackets in equation 3.95 are $\operatorname{cov}(X_\tau^+)$, which, using lemma 1 and the fact that the jump size J_τ is Gaussian and independent of the Gaussian pre-jump state X_τ^-, can be written as

$$\begin{aligned}\operatorname{cov}(X_\tau^+) &= \operatorname{cov}(X_\tau^- + J_\tau) \qquad (3.96)\\ &= \operatorname{cov}(X_\tau^-) + \operatorname{cov}(J_\tau) \qquad (3.97)\\ &= \Sigma_{X_\tau^-} + \Sigma_{\text{jump}} \qquad (3.98)\end{aligned}$$

Returning to equation 3.95, X_τ^+ is independent of both $\int_0^T e^{-A\tilde{t}} b dW$ and $\int_0^S e^{-A\tilde{t}} b dW$, so

$$\mathbb{E}\left(X_\tau^+ \left[\int_0^S e^{-A\tilde{t}} b dW\right]'\right) = \mathbb{E}\left(X_\tau^+\right)\mathbb{E}\left(\left[\int_0^S e^{-A\tilde{t}} b dW\right]'\right) \tag{3.99}$$

$$= \mathbb{E}\left(X_\tau^+\right).0 \tag{3.100}$$

$$= 0, \tag{3.101}$$

and similarly for $\mathbb{E}\left(\int_0^T e^{-A\tilde{t}} b dW (X_\tau^+)'\right)$.

We can also show that

$$\mathbb{E}\left(\int_0^T e^{-A\tilde{t}} b dW \left[\int_0^S e^{-A\tilde{t}} b dW\right]'\right) = \int_0^S \left(e^{-A\tilde{t}} b\right)\left(e^{-A\tilde{t}} b\right)' d\tilde{t} \tag{3.102}$$

$$= \Sigma_{\text{diffuse}}^{(S)} \tag{3.103}$$

So, putting the results in equations 3.98, 3.101 and 3.103 back into equation 3.95 leaves us with

$$\text{cov}(Z_t, Z_s) = e^{AT}\left\{\text{cov}(X_\tau^-) + \Sigma_{\text{jump}} + \Sigma_{\text{diffuse}}^{(S)}\right\}(e^{AS})' + \mathbf{1}_{\{t=s\}}\Sigma_{obs}^{(t)}, \tag{3.104}$$

which means that we now know the means and covariances of the Z_t observation vectors. By taking the first components of these, which are the means and covariances of the y_t observations, and arranging them as follows, we can get the means and covariances of the y_t observations in an observation neighbourhood:

$$\mu_{obs} = \begin{pmatrix} \mathbb{E}(y_1) \\ \mathbb{E}(y_2) \\ \vdots \\ \mathbb{E}(y_n) \end{pmatrix} = \begin{pmatrix} (\mathbb{E}(Z_1))_1 \\ (\mathbb{E}(Z_2))_1 \\ \vdots \\ (\mathbb{E}(Z_n))_1 \end{pmatrix} \tag{3.105}$$

and

$$\Sigma_{obs} = \begin{pmatrix} \text{cov}(y_1, y_1) & \text{cov}(y_1, y_2) & \dots & \text{cov}(y_1, y_n) \\ \text{cov}(y_2, y_1) & \ddots & & \vdots \\ \vdots & & & \\ \text{cov}(y_n, y_1) & \dots & & \text{cov}(y_n, y_n) \end{pmatrix} \tag{3.106}$$

$$= \begin{pmatrix} (\text{cov}(Z_1, Z_1))_{11} & (\text{cov}(Z_1, Z_2))_{11} & \dots & (\text{cov}(Z_1, Z_n))_{11} \\ (\text{cov}(Z_2, Z_1))_{11} & \ddots & & \vdots \\ \vdots & & & \\ (\text{cov}(Z_n, Z_1))_{11} & \dots & & (\text{cov}(Z_n, Z_n))_{11} \end{pmatrix} \tag{3.107}$$

These specify the mean and covariance of the multivariate Gaussian probability density function for the observations in an observation neighbourhood. Their derivation depends on the assumption that the distributions of the jump and of the pre-jump state are both Gaussian (though both of these distributional requirements can be relaxed to Gaussian mixture models).

Chapter 4

MCMC Model Calibration

Markov Chain Monte Carlo (MCMC) is a method for drawing samples from a distribution. The distribution must be specified by a function that is proportional to its density, but the normalizing factor (that makes the integral over the whole density equal to 1) need not be known. This latter fact is a significant boon when using the method in Bayesian estimation settings where often probabilities can be expressed quite simply up to proportionality, but where finding a normalizing constant can be tricky.

The method is reviewed in section 2.2, but the basic idea is to set up a Markov chain that has the required distribution as its stationary distribution. The chain can then be started from any state from which a state in the stationary distribution can be reached and it will, after a number of iterations, converge so that the states that the chain reaches are samples drawn from the stationary distribution. The amount of time necessary for the chain to converge to the required stationary distribution is potentially difficult to judge and there are numerous heuristic methods of doing so, a very useful one of which is visual inspection.

Setting up a Markov chain with a given stationary distribution is remarkably easy and can be done with the Metropolis-Hastings algorithm. All this algorithm requires is that we know the distribution we are trying to draw from (up to proportionality) and that we can draw samples from some known distribution. This latter distribution can be a simple one such as the Gaussian distribution and the samples drawn act as *proposals* for the next step in the chain. The proposal density usually has some sort of dependence on the current value of the chain, so for example it might be the current value of the chain plus some Gaussian noise. Sometimes these proposals are accepted, in which case the proposed value becomes the next step in the chain and sometimes they are rejected in which case the next step in the chain is simply the current value. Their acceptance or rejection is random, with the probability of acceptance being given by

$$p_{\text{accept}} = \min\left(\frac{p(x')q(x^t \mid x')}{p(x^t)q(x' \mid x^t)}, 1\right), \tag{4.1}$$

where $p(x)$ is the probability density from which we are trying to sample (to proportionality); $q(x \mid y)$ is the proposal density given that the current state is y; x^t is the current state of the chain; and x' is the proposal.

The proposal density again need only be known to proportionality; this is obvious because it is in a ratio with itself in equation 4.1, so any constant of proportionality in p or q would cancel. The minimum requirement on the proposal density is that it allows all states of the chain to be reached.

4.1 Variable Rate MCMC

In order for such a method to be useful for state estimation in models such as that in section 3 we must express the probability of all possible state sequences as a single probability density. Thanks to the structure of state space models this is quite straightforward.

Basic MCMC requires that we find the system state at the time of each observation. In contrast, the system described in section 3 and illustrated in figure 3.4 only requires knowledge of the state at the jump times. Furthermore, under Gaussian assumptions about the jump size, the only piece of state information that we require to be represented is the jump time itself, as the rest can be extracted through use of the Kalman filter (described below in section 4.2). This means that what we are actually interested in finding is the probability density over sets of jump times $\tau_{1:N}$ (where N is the unknown number of jumps), given the observations $y_{1:T}$, which we can write this as $p(\tau_{1:N} \mid y_{1:T})$.

In order to be able to write an expression for this distribution we must express it in terms of quantities that we know, such as the observation and state transition functions. First, we apply Bayes' Theorem to get

$$p(\tau_{1:N} \mid y_{1:T}) = \frac{p(y_{1:T} \mid \tau_{1:N})p(\tau_{1:N})}{p(y_{1:T})} \tag{4.2}$$

$$\propto p(y_{1:T} \mid \tau_{1:N})p(\tau_{1:N}), \tag{4.3}$$

where in the second step we put the $p(y_{1:T})$ in the denominator into the constant of proportionality (which we can do because $y_{1:T}$ are given and so $p(y_{1:T})$ is constant).

We must now deal with each of the two terms on the right hand side of equation 4.3. Before embarking on this, it is helpful to divide the observations into neighbourhoods dependent on jump times, as shown in figure 3.4. Let us call y_{N_i} as the neighbourhood of the i^{th} jump, encompassing all observations that occur after the i^{th} jump but before the $(i+1)^{th}$ jump. Then, define y_{N_0} as the neighbourhood of observations running from the start of the time series to just before the first jump and y_{N_N} as the neighbourhood of observations running from just after the last jump to the end of the set of observations. Noting that the union of all neighbourhoods is equal to the entire set of observations (i.e. $y_{N_{0:N}} = y_{1:T}$) we get

$$p(y_{1:T} \mid \tau_{1:N}) = p(y_{N_{0:N}} \mid \tau_{1:N}). \tag{4.4}$$

It is also useful to introduce conditional independence in the context of graphical models. In graphical terms, an item A is independent of an item B conditional on C if the only link from A to B goes through C. In this case $p(A \mid B, C) = p(A \mid C)$. In the case of our model (shown in figure 4.1) this means that given its predecessor and successor states, a state is independent

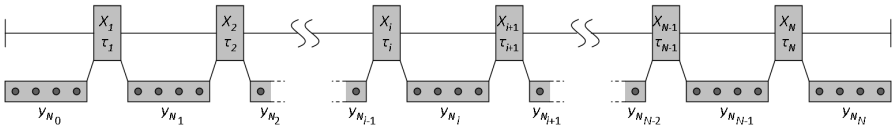

Figure 4.1: Graphical representation of the dependence structure of the variable rate model

of all others, and that given the state at either end of it, a neighbourhood of observations is independent of all other states and observations.

Returning to equation 4.3, the second term on the right can be dealt with by first applying the chain rule of probability and then the conditional independence of states to get

$$p(\tau_{1:N}) = p(\tau_N \mid \tau_{1:N-1})p(\tau_{N-1} \mid \tau_{1:N-2}) \dots p(\tau_2 \mid \tau_1)p(\tau_1) \tag{4.5}$$

$$= p(\tau_N \mid \tau_{N-1})p(\tau_{N-1} \mid \tau_{N-2}) \dots p(\tau_2 \mid \tau_1)p(\tau_1) \tag{4.6}$$

$$= p(\tau_1) \prod_{i=2}^{N} p(\tau_i \mid \tau_{i-1}). \tag{4.7}$$

Here each of the functions in the product on the right are state transition functions, which we know from the model definition (given by the jump time distribution).

Some progress with the first term on the right hand side of equation 4.3 can be made by applying the chain rule of probability to the neighbourhoods of observations to get

$$p(y_{N_{0:N}} \mid \tau_{1:N}) = p(y_{N_N} \mid \tau_{1:N}, y_{N_{0:N-1}})p(y_{N_{N-1}} \mid \tau_{1:N}, y_{N_{0:N-2}}) \dots \\ \dots p(y_{N_1} \mid \tau_{1:N}, y_{N_0})p(y_{N_0} \mid \tau_{1:N}) \tag{4.8}$$

$$= p(y_{N_0} \mid \tau_{1:N}) \prod_{i=1}^{N} p(y_{N_i} \mid \tau_{1:N}, y_{N_{0:i-1}}). \tag{4.9}$$

We cannot simplify this expression more at this stage, since a neighbourhood of observations is independent of all others only given the preceding and following state: the partial state information in $\tau_{1:N}$ is *not* sufficient, we also require some information about the X part of the state.

Putting equations 4.7 and 4.9 into equation 4.3 gives us an expression for the jump time distribution given the observations:

$$p(\tau_{1:N} \mid y_{1:T}) \propto p(\tau_1)p(y_{N_0} \mid \tau_{1:N}) \prod_{i=2}^{N} p(\tau_i \mid \tau_{i-1}) \prod_{i=1}^{N} p(y_{N_i} \mid \tau_{1:N}, y_{N_{0:i-1}}). \tag{4.10}$$

The final term is the only part of this for which we do not yet have an expression, but we will address this in the following section.

4.2 Rao-Blackwellization

We have previously asserted that given Gaussian assumptions about the jump-size and prior distributions, it is only necessary to store jump time information

and that the rest of the state information (i.e. the value of X: x and $\dot{x}$) can be derived. This section explains how this is done using a technique called *Rao-Blackwellization* (see e.g. Cappé et al. (2007)). It also addresses the problem of calculating the final term in equation 4.10.

The first step is to divide the state into *linear Gaussian* and *non-linear Gaussian* (i.e. not linear or not Gaussian or both) parts. In our case the linear Gaussian part of the state is comprised of the trend and value processes, x and $\dot{x}$, which we denote X. Since there is an instantaneous jump at the times at which we want to calculate the state, the state at that time can take two values, the pre-jump state X^- and the post-jump state X^+. We will calculate the pre-jump state at each jump time, so that each transition consists of a jump, then a diffusion. The non-linear part of the state is made up of the jump times, τ. This allows us to write the state transition and observation functions in the following way:

$$X^-_{\tau_i} = A(\tau_i)(X^-_{\tau_{i-1}} + J_i) + \epsilon^X_i \tag{4.11}$$

$$y_{N_i} = B(\tau_i)(X^-_{\tau_i} + J_i) + \epsilon^Y_i, \tag{4.12}$$

where $A(\tau_i)$ and $B(\tau_i)$ are matrices dependent on the jump time τ_i, y_{N_i} is the i^{th} observation neighbourhood, J_i is the size of the i^{th} jump, ϵ^X_i and ϵ^Y_i are mean zero Gaussian noise of appropriate dimension (see below).

The jump size J_i is Gaussian and, if we assume it has zero mean we can incorporate it into the noise terms in equations 4.11 and 4.12 to give

$$X^-_{\tau_i} = A(\tau_i)X^-_{\tau_{i-1}} + \widetilde{\epsilon^X_i} \tag{4.13}$$

$$y_{N_i} = B(\tau_i)X^-_{\tau_i} + \widetilde{\epsilon^Y_i}, \tag{4.14}$$

In fact we do not need to assume that the jump size has zero mean to get the system into this form, but doing so with a non-zero jump mean is a little more complicated and involves augmenting the state space with a dummy constant state.

The advantage of writing the system in this form is that, if we fix the jump times τ_i then the system in equations 4.13 and 4.14 is linear Gaussian (since the A and B matrices become constant). This means that if we have distributional information about $X^-_{\tau_{i-1}}$ we can use the Kalman filter to derive distributional information about $X^-_{\tau_i}$ and y_{N_i}. Furthermore, at the start of the problem, we *do* have distributional information about the initial value of X (through its prior), so (if this prior is Gaussian) we can find the distribution of $X^-_{\tau_1}$, then using that $X^-_{\tau_2}$, etc., until we obtain the value of $X^-_{\tau_i}$ for all i from 1 to N.

Kalman Filter

The Kalman filter can be though of as a *prediction-correction* algorithm, in which an initial 'prediction' for the state, given by the state prior, is 'corrected' once an observation has been made using equation 4.14. This 'corrected' state is then used as the basis of another prediction for the state at the next step using equation 4.14 and the process continues. At the prediction stage, the prediction can be used, along with the observation equation 4.14 to make a best estimate of the next observation given all the previous observations to this point. This step is known as 'prediction error decomposition' and will yield the crucial terms

we need to completely specify the right hand side of equation 4.10. Since the entire system is linear Gaussian all the distributions involved are Gaussian and so we need only track the first (mean) and second (covariance) moments of the distributions involved. Figure 4.2 shows the process.

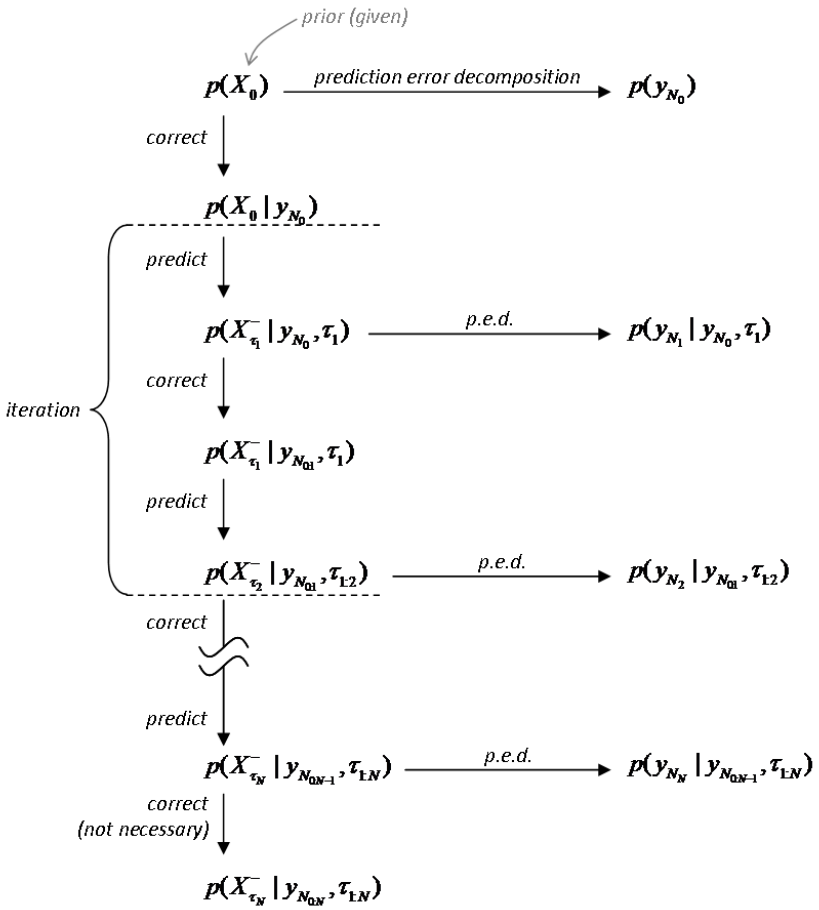

Figure 4.2: The Kalman filter process to extract state information and observation probabilities

Let us focus on the i^{th} iteration of the Kalman filter.

Initial information We start with the estimate for the mean and covariance of $X_{\tau_i}^-$ from the previous $(i-1)^{\text{th}}$ step. Denote these $\mu_i^{(p)}$ and $\Sigma_i^{(p)}$, respectively. On the first iteration ($i = 0$) these are the state priors μ_{prior} and Σ_{prior}. On later iterations these are the predictions made by the 'predict' step of the preceding iteration (see below).

Correct step In the correct step, we use the new observation neighbourhood y_{N_i} to refine our previous estimates of the mean and covariance of $X_{\tau_i}^-$. This is done using the following updates (which come from the standard

derivation of the Kalman filter (see e.g. Candy (2009))

$$\mu_i = \mu_i^{(p)} + K_i(y_{N_i} - B_i\mu_i^{(p)}) \tag{4.15}$$

$$\Sigma_i = (I - K_iB_i)\Sigma_i^{(p)} \tag{4.16}$$

$$K_i = \Sigma_i^{(p)}B_i'(B_i\Sigma_i^{(p)}B_i' + C_v)^{-1} \tag{4.17}$$

where K_i is known as the *Kalman gain*, B_i is shorthand for the $B(\tau_i)$ matrix from equation 4.12, I is the 2×2 identity matrix and C_v is the covariance of $\widetilde{\epsilon_i^{obs}}$.

The estimates that are derived for μ_i and Σ_i are the 'final' best estimates of these quantities.

Predict step In the predict step we use the current best state estimate along with the state transition function (equation 4.13) to make a prediction of the distribution of the next state. The best prediction of the next state is simply the current state transformed by the state transition, giving

$$\mu_{i+1}^{(p)} = A_{i+1}\mu_i \tag{4.18}$$

$$\Sigma_{i+1}^{(p)} = A_{i+1}\Sigma_iA_{i+1}' + C_u, \tag{4.19}$$

where A_{i+1} is shorthand for the $A(\tau_{i+1})$ matrix from equation 4.13 and C_u is the covariance of $\widetilde{\epsilon_i^X}$.

Prediction error decomposition The prediction error decomposition now allows us to derive an expression for the probability of an observation neighbourhood given all the previous ones. It does this using the state prediction above and the fact that the new observation neighbourhood is conditionally independent of all its predecessors given the state immediately preceding it. This means that, since our estimate of that state was the best we could make, using that to estimate the next observation probability is no worse than using the entire set of past observations (this is an example of the Rao-Blackwell theorem, applied in the context of sampling problems by Casella and Robert (1996)). Estimating the observation from the preceding state can be done easily by transforming the state using the observation function in equation 4.14 to give

$$\mu_{y_{N_i}} = B_i\mu_i^{(p)} \tag{4.20}$$

$$\Sigma_{y_{N_i}} = B_i\Sigma_i^{(p)}B_i' + C_v. \tag{4.21}$$

Note that here we have given the prediction error decomposition for the $(i-1)^{\text{th}}$ iteration to avoid unnecessary subscripts.

This step allows us to write an expression for the final term of equation 4.10 as

$$p(y_{N_i} \mid y_{N_{0:i-1}}, \tau_{1:N}) \sim \mathcal{N}(\mu_{y_{N_i}}, \Sigma_{y_{N_i}}). \tag{4.22}$$

This is a multivariate normal distribution with $|y_{N_i}|$ dimensions.

As shown in figure 4.2, if there are N jumps then we must perform $N+1$ prediction error decompositions, since there are $N+1$ neighbourhoods of observations and observation terms in equation 4.10.

The matrices above have the following dimensions: A_i is 2×2, B_i is $M \times 2$, C_u is 2×2, C_v is $M \times M$, X_i^- is 2×1, y_{N_i} is $M \times 1$, ϵ_i^X is 2×1 and ϵ_i^{obs} is $M \times 1$, where M is the number of observations in the neighbourhood y_{N_i}.

This leaves us to specify the quantities A_i, B_i, C_u and C_v. We can do this by considering the mean and covariance of X_{τ_i} and y_{N_i} as expressed in equations 4.13 and 4.14, respectively, and then comparing this to the means and covariances calculated for these quantities in the derivation of the state transition and observation equations in section 3. We can then choose the unknown quantities to make these two expressions equivalent.

Considering state transition first, we can get an expression for the expectation of $X_{\tau_i}^-$ by taking expectations of equation 4.13, giving

$$\mathbb{E}\left(X_{\tau_i}^-\right) = A_i \mathbb{E}\left(X_{\tau_{i-1}}^-\right). \tag{4.23}$$

The covariance of $X_{\tau_i}^-$ is given by

$$\text{cov}(X_{\tau_i}^-) = A_i \text{cov}\left(X_{\tau_{i-1}}^-\right) A_i' + \text{cov}\left(\widetilde{\epsilon_i^X}\right). \tag{4.24}$$

If we compare these with the earlier expressions for the mean and covariance of $X_{\tau_i}^-$ expressed in terms of its predecessor $X_{\tau_{i-1}}^-$, given in equations 3.17 and 3.37, we see that there $X_{\tau_i}^-$ was distributed as

$$X_{\tau_i}^- \sim \mathcal{N}\left(e^{AT}\mathbb{E}(X_{\tau_{i-1}}^-), e^{AT}\left[\Sigma_{\text{diffuse}}^{(T)} + \Sigma_{\text{jump}} + \text{cov}(X_{\tau_{i-1}}^-)\right](e^{AT})'\right), \tag{4.25}$$

where $T = \tau_i - \tau_{i-1}$ and $\Sigma_{\text{diffuse}}^{(T)}$ is defined in equation 3.39. (This comes from equation 3.37 because the X_0 in that corresponds to the state at the start of the diffusion section, here $X_{\tau_{i-1}}^+$ which has covariance given by equation 3.98)

Equating the expressions for mean and covariance in 4.23 and 4.24 with those in equation 4.25 we get expressions for A_i and $C_u = \text{cov}\left(\widetilde{\epsilon_i^X}\right)$:

$$A_i = e^{AT} \tag{4.26}$$

$$C_u = e^{AT}\left[\Sigma_{\text{diffuse}}^{(T)} + \Sigma_{\text{jump}}\right](e^{AT})' \tag{4.27}$$

We now turn our attention to B_i and C_v, and the observation equation. In section 3, we did not calculate a convenient form for the observation mean and covariance directly, but instead worked with an augmented observation Z_t, which included a dummy observation of the unobserved trend process $\dot{x}$. In equation 4.14, the form of the B_i matrix determines what we observe (in our case its second column will be zero since we do not observe the $\dot{x}$ process). We can make progress by considering the form of the observation at time t in the observation neighbourhood as given by equation 4.14:

$$Z_t = B_i^t X_i + \widetilde{\epsilon_{i,t}^Y}, \tag{4.28}$$

where B_i^t is a 2×2 matrix, the first row of which is the row of B_i corresponding to the observation at time t and the second row of which corresponds to the dummy observation so can be anything. $\widetilde{\epsilon_{i,t}^Y}$ is a 2×1 vector of noise, the first

element of which is the noise corresponding to the observation at time t in $\widetilde{\epsilon_i^Y}$. Z_t is then a 2×1 vector, the first element of which is the observation at time t and the second element of which is the dummy observation of $\dot{x}$ at time t.

We can now calculate the expectation of Z_t and its covariance with any other observation in the neighbourhood Z_s in terms of the state from which it derives $X_{\tau_i}^-$ and then equate these expressions to those previously calculated in section 3. The expectation of Z_t from equation 4.14 is given by

$$\mathbb{E}\left(Z_t\right) = B_i^t \mathbb{E}\left(X_{\tau_i}^-\right). \tag{4.29}$$

The covariance of Z_t with any other observation Z_s is given by

$$\begin{aligned}
\operatorname{cov}(Z_t, Z_s) &= \mathbb{E}\left(\left(B_i^t X_{\tau_i}^- + \widetilde{\epsilon_{i,t}^Y}\right)\left(B_i^s X_{\tau_i}^- + \widetilde{\epsilon_{i,s}^Y}\right)'\right) - \mathbb{E}\left(Z_t\right)\mathbb{E}\left(Z_s\right) && (4.30)\\
&= \mathbb{E}\left(B_i^t X_{\tau_i}^- X_{\tau_i}^{-\prime} B_i^s + B_i^t X_{\tau_i}^- \widetilde{\epsilon_{i,s}^Y}' + \widetilde{\epsilon_{i,t}^Y} X_{\tau_i}^{-\prime} B_i^{s\prime} + \widetilde{\epsilon_{i,t}^Y}\widetilde{\epsilon_{i,s}^Y}'\right)\\
&\quad - B_i^t \mathbb{E}\left(X_{\tau_i}^-\right)\mathbb{E}\left(X_{\tau_i}^-\right)' B_i^{s\prime} && (4.31)\\
&= B_i^t \operatorname{cov}\left(X_{\tau_i}^-\right) B_i^s + \operatorname{cov}\left(\widetilde{\epsilon_{i,t}^Y}, \widetilde{\epsilon_{i,s}^Y}\right). && (4.32)
\end{aligned}$$

The final term $\operatorname{cov}\left(\widetilde{\epsilon_{i,t}^Y}, \widetilde{\epsilon_{i,s}^Y}\right)$ is a 2×2 matrix which, by the definition of $\widetilde{\epsilon_{i,t}^Y}$, has the covariance between the elements of the noise $\widetilde{\epsilon_i^Y}$ corresponding to the observation at t and at s as its first element. This first element is the element of C_v corresponding to observations at time t and s, which we will denote $C_v^{(t,s)}$, so

$$C_v^{(t,s)} = \left(\operatorname{cov}\left(\widetilde{\epsilon_{i,t}^Y}, \widetilde{\epsilon_{i,s}^Y}\right)\right)_{11} \tag{4.33}$$

Equating the expression for the expectation of Z_t given in equation 4.29 with its counterpart from section 3 in equations 3.83 (and noting that we have assumed the jump has zero mean) we get

$$B_i = \begin{pmatrix} e^{AT_1} & 0 \\ e^{AT_2} & 0 \\ \vdots & \vdots \\ e^{AT_M} & 0 \end{pmatrix} \tag{4.34}$$

where $T_j = t_j - \tau_i$, where t_j is the time of the the j^{th} observation in the neighbourhood. The second column is entirely zero since the second component of $X_{\tau_i}^-$, the trend $\dot{x}$ process, does not play any part in the observation.

Similarly, equating the covariance expression in equation 4.32 with that in equation 3.104 in section 3 we get

$$\operatorname{cov}\left(\widetilde{\epsilon_{i,t}^Y}, \widetilde{\epsilon_{i,s}^Y}\right) = e^{AT}\left[\Sigma_{\text{jump}} + \Sigma_{\text{diffuse}}^{(\min(S,T))}\right](e^{AS})' + \mathbf{1}_{t=s}\Sigma_{obs}^{(t)}, \tag{4.35}$$

where $T = t - \tau_i$, $S = s - \tau_i$ and where s and t correspond to observation times in the observation neighbourhood N_i. By equation 4.33 this specifies the entire C_v matrix as a matrix of the first elements of the above covariance for every s and t in the neighbourhood.

This now gives us the entire set of unknown quantities that we need to use the Kalman filter for state inference.

4.2.1 Intermediate State Distribution

The filter developed up to now allows us to infer the distribution of the pre-jump state at each of the jump times. However, sometimes it might be useful to be able to infer the state at the observation times. This can be done using a second Kalman filter that we apply *within* an observation neighbourhood. Figure 4.3 shows the step-by-step logic of this filter. As shown there, since we are only interested in state inference, we do not need to calculate the prediction error decomposition as we are only interested in state inference.

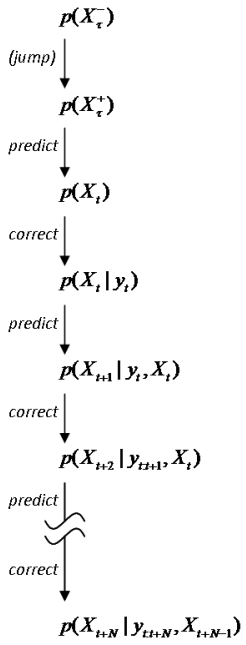

Figure 4.3: A Kalman filter to infer state distribution information within an observation neighbourhood

Initially we know the distribution of the pre-jump state $X_{\tau_i}^-$. We use the jump distribution to find the distribution of the post-jump state $X_{\tau_i}^+$. From this starting point, the state transition from one observation time to the next is diffusion-only. This means that the state transition is the same as in the jump-to-jump filter above, but with the jump covariance Σ_{jump} removed from the expression in equation 4.27.

The observation function is the much simpler

$$y_i = Bx_i + \epsilon_i^{obs}, \tag{4.36}$$

where $B = [1 \;\; 0]$ and $\epsilon_i^{obs} \sim \mathcal{N}(0, \sigma_{obs}^2)$.

Running the standard Kalman filter using these state transition and observation functions will yield the state at all observation times in a neighbourhood. The jump-to-jump Kalman filter must be run prior to this in order to get the starting points for the neighbourhood states.

4.3 Proposals

In order to build a Markov chain that will give samples from the now fully specified distribution in equation 4.10, we need a proposal function from which we can draw samples that generates a proposal for the next state of the chain. In the case of our MCMC, the chain state consists of the entire sequence of jump times, $\tau_{1:N}$ along with their number N. Therefore, any proposal function must be able to move from one sequence of jump times to another and these moves together must be able to reach all possible sequences of jump times.

To this end we propose a set of simple proposal steps to create a new proposal of state times. These are shown in figure 4.4 and consist of move steps, in which one jump time is altered locally, birth steps, in which a new jump time is created, and death steps, in which an existing jump time is removed.

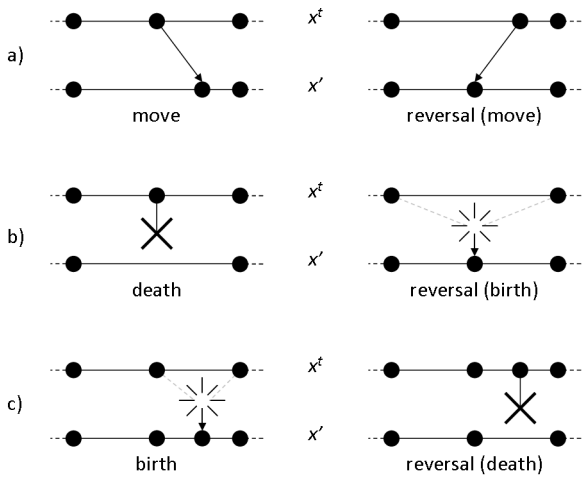

Figure 4.4: The three basic types of proposal for state sequence updates, along with their reversals: (a) move, (b) death and (c) birth.

One crucial feature of these proposals is that they must be able to cover the entire space of possible jump numbers and times. It is obvious that the proposals in figure 4.4 do this, since any starting sequence of jump times can be transformed to any other sequence through a series of moves, births and deaths.

The other key feature that we require from the proposals is that they must be *reversible*. This means that if the probability of a proposal from the current state is non-zero, then there must also be a non-zero probability of being able to make a transition back from the proposal to the current state. Mathematically, we require

$$q(x' \mid x^t) > 0 \Leftrightarrow q(x^t \mid x') > 0 \tag{4.37}$$

where x' is the proposal and x^t is the current state and $q(\cdot \mid y)$ is the proposal density function from current state y.

The reason for requiring reversibility is immediately obvious from the acceptance probability in Metropolis-Hastings, given in equation 4.1. The presence of

$q(x^t \mid x')$ in the numerator makes it clear that if this is zero, no proposal will ever be accepted.

In practice reversibility in our model means that the way in which we generate moves, births and deaths must be such that it is possible to go back the other way, and we must be able to calculate the probability densities of these reverse steps. A step that would not be reversible, for example, would be to move a jump time to the point at which the observation process changed most (see figure 4.5a). In this case there would be no probability of generating the reverse proposal, since the largest change in observation is not at the original state point. In contrast, an example of a reversible step would be to move the position of a jump according to a Gaussian distribution (figure 4.5b). Then, wherever the step proposes to move the jump time to, there is always some probability of generating the original state if starting from the new proposal.

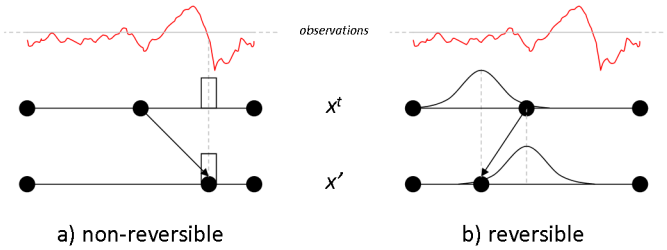

Figure 4.5: A non-reversible (a) and a reversible jump move (b)

The proposals we use are as follows (illustrated in figure 4.4).

Move For the move proposal, we choose a jump-time at random and then move it by adding a sample from a Gaussian mixture model to it.

$$q_{\text{move}}(x' \mid x^t) = p_{\text{move}} \frac{1}{N} \text{GMM}(\tau_i' \mid \tau_i^t) \tag{4.38}$$

where p_{move} is the probability of making a move step, τ_i is the jump time that moves and $\text{GMM}(\tau_i' \mid \tau_i^t)$ is the pdf of a GMM centred at τ_i^t, evaluated at τ_i'.

The reverse of a move proposal is simply another move: that of the point from its new position back to its old position.

Death The death proposal is obvious: choose a jump time at random and remove it from the state sequence.

$$q_{\text{death}}(x' \mid x^t) = p_{\text{death}} \frac{1}{N} \tag{4.39}$$

where p_{death} is the probability of making a death step.

The reverse of a death proposal is a birth proposal occurring between the two states on either side of the killed state and leading to a birth in the position of the killed state.

Birth A birth proposal can add a new jump time between any pair of states (or between the sequence start time and the first state, or the last state and

the sequence end time). The location of the new state is chosen uniform randomly between the neighbouring states.

$$q_{\text{birth}}(x' \mid x^t) = p_{\text{death}} \frac{1}{(N+1)(\tau_{i+1} - \tau_i)} \tag{4.40}$$

where p_{birth} is the probability of making a birth step, and where the birth between x' and x^t occurs between jumps τ_i and τ_{i+1} (with τ_0 defined at $t_{\min}$ and τ_{N+1} defined as $t_{\max}$).

The reversal of a birth proposal is a death proposal that proposes the death of the newly created state.

We can then define the reversal of a step as

$$q(x^t \mid x') = \begin{cases} q_{\text{move}}(x^t \mid x') & x^t \to x' \text{ is a move} \\ q_{\text{birth}}(x^t \mid x') & x^t \to x' \text{ is a death} \\ q_{\text{death}}(x^t \mid x') & x^t \to x' \text{ is a birth} \end{cases} \tag{4.41}$$

A practical issue with this scheme is that GMM moves can generate a new jump time anywhere, including before the start of the time series, after the end of the time series and at times beyond the 'following' jump or before the 'preceding' one. This is resolved by defining the probability of any such sequence as 0. This means that whilst such a sequence can be proposed, thanks to the Metropolis-Hastings acceptance probability in equation 4.1, it will never be accepted, since the acceptance probability will always be 0.

4.4 Parameter Estimation

So far we have only dealt with estimation of the state sequence. The parameters of the model have not been mentioned and we have implicity assumed that they are known. Unfortunately, this is not the case. Indeed, the entire point of the MCMC estimation is parameter inference.

MCMC is a way of drawing samples from a complicated multivariate distribution. In the case of model state estimation (as seen above), we arrange that this distribution is the probability density function of the unknown states given the observations, which we can write as $p(X \mid y)$. In order to incorporate parameter estimation into the system, we extend the dimensionality of the space from which we draw the samples from the number of unknown states to the number of unknown states plus the number of parameters. The distribution we want to draw from in this extended space is the joint probability density of the unknown states *and* unknown parameters, given the observations. We can write this distribution as $p(X, \theta \mid y)$. Each MCMC sample is a sample in this extended space and, once the chain converges, will be drawn from the required distribution, giving us an estimate of the probability of state and parameter values.

Increasing dimensionality inevitably increases the complexity of the function from which we are trying to sample, meaning that the chain is likely to take longer to converge. For this reason, we want to minimize the number of dimensions in which we need to sample. This is why it is useful in the above state estimation to reduce the state space being sampled to only the jump times

(plus the number of jumps). By doing this, we reduce the dimensionality from $2 \times T$ in a conventional MCMC (two dimensional state by T observation times) to $N + 1$ where N is the number of jumps.

Each step of the MCMC chain can propose an update for the entire state space, or a just part of it. All that we require is that the range of updates must be sufficient to cover the entire space of interest (for example, if we never updated the parameters, we could not hope to estimate them), although some update schemes might converge more rapidly than others. We have already seen how to sample from the state distribution (here the jump-time distribution) above, so when adding parameters to the sample space it is convenient to split the updates into two types: state and parameter. The state update must draw from the pdf of the state given the parameters and observations $p(\tau_{1:N} \mid \theta, y_{1:T})$. With the parameter implicit, this is exactly what we have described in the above sections. The parameter update must draw from the pdf of the parameters given the states and observations $p(\theta \mid \tau_{1:N}, y_{1:T})$. It is this latter update with which the following subsection deals.

By using a combination of these two block updates, one for states and the other for parameters, the MCMC will be able, when converged, to sample from the required joint distribution $p(\tau_{1:N}, \theta \mid y_{1:T})$. There is, however, no restriction on the sequence in which these updates are combined other than that both must be made at some points.

4.4.1 Parameter Update

The parameter update consists of sampling a new set of parameters drawn from the distribution $p(\theta \mid \tau_{1:N}, y_{1:T})$. In fact, just as we divided the state from the parameters, we can divide the parameters into smaller groups and in a single update just update a subset of the entire parameter set. Consider a division of the parameters into two sets, $\theta = \{\theta_1, \theta_r\}$.

As with the state estimation, our aim is to derive a distribution for the parameters of interest θ_1 in terms of things that we can evaluate. We can then draw samples from this distribution using Metropolis-Hastings and use the sample so created as the next step in the MCMC chain. The distribution that we want to sample from is $p(\theta_1 \mid \tau_{1:N}, y_{1:T}, \theta_r)$. We can use Bayes' Theorem twice (putting terms involving only given quantities into the proportionality constant) to get

$$p(\theta_1 \mid \tau_{1:N}, y_{1:T}, \theta_r) \propto p(y_{1:T} \mid \tau_{1:N}, \theta) p(\theta_1 \mid \tau_{1:N}, \theta_r) \tag{4.42}$$

$$\propto p(y_{1:T} \mid \tau_{1:N}, \theta) p(\tau_{1:N} \mid \theta) p(\theta_1 \mid \theta_r). \tag{4.43}$$

Noting that the first two terms on the right-hand side are those in equations 4.9 and 4.7, respectively, with the parameters θ that were implicit there made explicit, we can write

$$p(\theta_1 \mid \tau_{1:N}, y_{1:T}, \theta_r) \propto$$
$$p(\theta_1 \mid \theta_r) p(\tau_1 \mid \theta) p(y_{N0} \mid \tau_{1:N}) \prod_{i=2}^{N} p(\tau_i \mid \tau_{i-1}, \theta) \prod_{i=1}^{N} p(y_{N_i} \mid \tau_{1:N}, y_{N_{0:i-1}}). \tag{4.44}$$

The right hand side of this expression is the same as that of equation 4.10 multiplied by $p(\theta_1 \mid \theta_r)$, i.e. the prior of θ_1, and with the dependence on the parameters θ made explicit. So, this gives the pdf from which we are trying to sample and we can use exactly the same methods as above in order to evaluate it.

4.4.2 Implementation

All that we need now in order to be able to use MCMC to sample from this distribution are proposal functions for individual or blocks of parameters and parameter (block) priors.

We chose initially to update the parameters individually, using Gaussian mixtures with appropriate width as their proposal functions.

The priors were initially chosen to be uninformative. (They were actually chosen to be the uninformative improper prior $p(\theta_1 \mid \theta_r) = 1$, for all values of θ_1).

These proposals and priors have not been found to give good results for the problem of parameter estimation when the jump-times are also unknown (see section 6). Improvement of this parameter update scheme is currently an on-going area of research (see below) and is key to making the system work.

4.4.3 Improvements

The simple MCMC scheme described above does not work well for our parameter estimation problem (see section 6). Since this is the purpose of the MCMC set-up, this is a grave problem. It is therefore important that we understand why it is not working and what possible approaches we can take to overcome this.

MCMC is as an algorithm that draws samples from a high-dimensional probability density function. In the case of a parameter block of p parameters, this function has a p-dimensional range and is given by equation 4.44. The purpose of this is to gain information about the shape of the function through the samples drawn: enough samples should do a reasonable job of representing the probability density function in question. This, in turn, gives us probability information about where the true values of the parameters are likely to lie.

As in optimization methods, it is intuitive to think of the function being sampled from as a multi-dimensional landscape, around which the MCMC algorithm walks. If the proposal function is symmetrical and does not depend on the local target function, then from equation 4.1 the probability of a move to a higher valued part of the objective is more likely than one to a low valued one (though not inevitable, since the proposal is random).

MCMC, then, is an algorithm that walks around a target function in order to illustrate its shape. The shape of the function matters significantly, just as it does in optimization methods. Complicated target functions with lots of local maxima spaced far apart are pathological cases for MCMC (and many optimization techniques), since it is likely that the algorithm will get stuck in one of them and not be able to escape. To overcome this the proposal function should have the possibility of making proposals that can reach from one maxima to another, even in the case of the most isolated maxima. This, however, requires some prior knowledge of the shape of the objective function in order to tailor the

proposal function, and it is hard to create a proposal function that can explore the local maxima and also transit between them.

A number of approaches for improving the performance of the MCMC algorithm in the shape of tricky target functions are available, many of which were reviewed in section 2.2. We hope to implement some of these in our future work (see section 7) in order to improve MCMC performance in our problem. The difficulties we have encountered underline the fact that getting MCMC algorithms to work can be difficult and they cannot be treated as a statistical magic bullet for solving all estimation problems.

4.5 Numerical Stability

There are a number of numerical issues that make the implementation of the above MCMC scheme tricky. This section briefly covers the solutions to them.

4.5.1 Numerical Precision

The pdf functions above take tiny values throughout most of their domain. This is problematic for the finite representations of numbers used in computers and means that values below a certain size get truncated to zero. However, since, when using Metropolis-Hastings, we need to divide by these small numbers this is not acceptable. To get around this the logarithms of all probability density values are used throughout the calculations.

4.5.2 Covariance Calculation

The calculation of the covariance matrices in the state transition equation 4.27 and observation equation 4.35 can cause numerical problems if the diffusion section is long (T is large). This is because the matrix e^{AT} becomes very poorly conditioned for large T. This problem can be overcome by using the following recursion to calculate the covariance of the state vector after time T given initial covariance Σ_0:

$$e^{AT}\Sigma_0 e^{AT} = \Sigma(T) = \begin{cases} e^{AT}\Sigma_0 e^{AT} & \text{cond}\left(e^{AT}\right) < \text{cond}_{\max} \\ \exp\left(\frac{AT}{2}\right)\Sigma\left(\frac{T}{2}\right)\exp\frac{AT}{2} & \text{otherwise} \end{cases} \tag{4.45}$$

where $\text{cond}_{\max}$ is chosen to be a condition number where the calculation does not present numerical problems (we used 10^5).

In the case of the observation function we can use the above method if we decompose the calculation as

$$e^{AT}\Sigma_0 e^{AT} = e^{AT}\Sigma_0 e^{AT} e^{A(S-T)}, \tag{4.46}$$

with $T < S$, so that the first part can be calculated recursively.

Chapter 5

Particle Filter

MCMC provides a powerful method for estimating hidden state and parameters in a batch context, but it is sometimes useful to be able to process information online as it arrives. This is obviously true in financial applications such as trading, where decisions need to be made as soon as information is available. On the other hand, online methods cannot hope to be as accurate as batch methods since when estimating the current state they do not have the advantage of knowing future observations.

As our model is not linear Gaussian, a particle filter (see section 2.3) is an appropriate estimation technique. The basic idea of the particle filter is to use a collection of weighted samples to represent the current state distribution. When a new observation becomes available the particles are updated using the a Bayesian update derived from the observation and state transition functions. Here we extend the basic particle filter to the variable rate setting, with the times at which state information is held being determined by jump times. Much of the machinery developed for the MCMC estimation above can be reused in order to do this.

5.1 Basic Particle Filter

As in MCMC, the only state that we will explicitly represent will be the non-Gaussian part, the jump times. The remaining state (the x and $\dot{x}$ components) will be determined using a Kalman filter, as in section 4.2. In the context of a particle filter this means that a 'particle' (a single sample in the collection of samples that we use to represent the unknown state) will be a collection of jump times, from the start of the time series to the present time.

Let us assume that we have a weighted collection of such particles representing our estimate of the system state sequence up to time t, so that each particle is a set of jump times which, when taken together are a good approximation to our 'best estimate' of the probability density function of the jump times (see figure 5.1. The basic particle filter step is, upon the arrival of a new observation at time $t + \Delta t$, to update this collection so that it represents our estimate of the pdf of the jump times up to $t + \Delta t$.

The first stage in the update step is to sample a new element of the state sequence at $t + \Delta t$ from some sample distribution for each of the particles. The

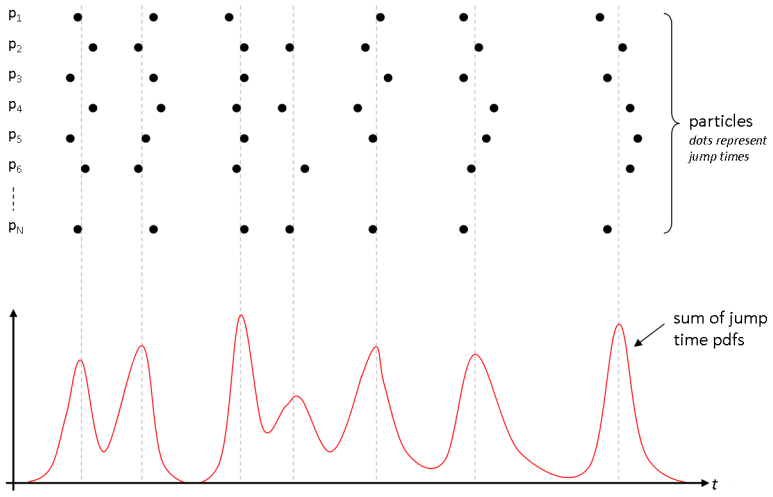

Figure 5.1: Representing the pdf of jump times using particles

new sample is appended to the particle's existing state sequence.

The sample distribution can be arbitrary, but it is useful to pick one that is both easy to sample from and likely to generate samples close to the likely state values at $t + \Delta t$. So, for example, in a tracking application it is useful to generate samples around where you expect the target might be at $t + \Delta t$, rather than completely at random. This is because when the new samples are weighted, if most are a very poor estimate of the state they will receive near zero weights, meaning that just a handful of samples will have to represent the bulk of the probability mass. On the other hand, if the new samples are good estimates of the state then the weight will be more evenly distributed, giving a more nuanced representation of the state distribution. Figure 5.1 illustrates this. The sample distribution is known as the *importance distribution* because of its similarity to importance sampling in other Monte-Carlo methods. In these the importance function attempts to place samples in the areas that contribute most to the final answer (for example the high valued parts of a function being numerically integrated). Since samples need to be drawn from it, it must be easy to sample from.

After samples have been drawn, the next stage is to weight them so that the samples represent the posterior distribution of the state after incorporating the new observation into the state estimate. Note that the particles can have different weights at each observation time, so that along with the particles themselves we also obtain a sequence of weights for each particle with an entry at each observation time.

We want to approximate a distribution $p(x \mid y)$ with a weighted collection of samples $\{x^i\}$ drawn from the sample distribution $q(x \mid y)$. (In the particle filter context, x here represents the state sequence, y represents the observation sequence and x^i represents a particle, itself being a particular state sequence)

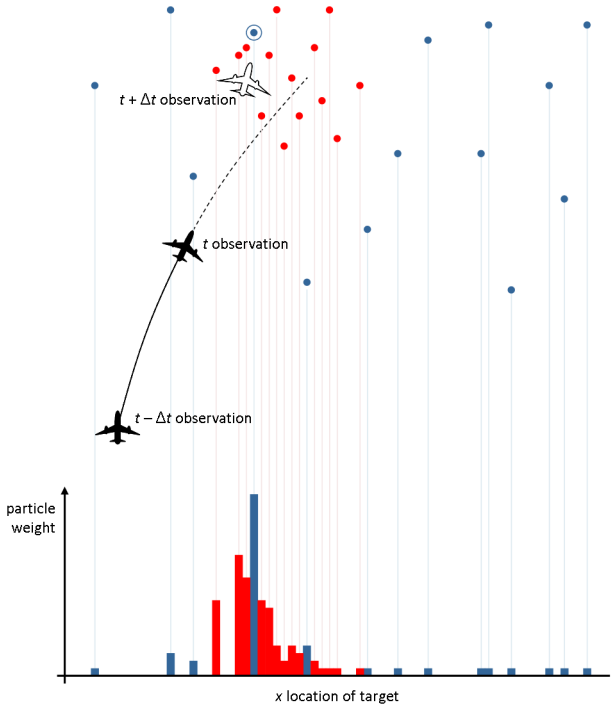

Figure 5.2: Illustration of the impact of the choice of importance function. The red points are drawn from a good importance function at t, whereas the blue dots are drawn from a poor importance function. With the blue importance function most of the weight attaches to the circled point, whereas with the red function the weight is more evenly distributed and the location representation in the lower graph is a more useful representation of the posterior pdf.

Mathematically, we want

$$\sum_{\{x^i\}} W(x^i \mid y)\delta_{x^i} \approx p(x \mid y) \tag{5.1}$$

where $W(x^i \mid y)$ is the weight of the sample at x^i and δ_{x^i} is a delta function at the point x^i. In the limit we want

$$\lim_{|\{x^i\}|\to\infty} \sum_{\{x^i\}} W(x^i \mid y)\delta_{x^i} = p(x \mid y). \tag{5.2}$$

Now, if we consider a region of radius ϵ around a point x^*, we require

$$\int_{|x-x^*|<\epsilon} W(x \mid y)q(x \mid y)dx = \int_{|x-x^*|<\epsilon} p(x \mid y)dx, \tag{5.3}$$

which is satisfied if

$$W(x \mid y) = \frac{p(x \mid y)}{q(x \mid y)}. \tag{5.4}$$

Applying Bayes' theorem to this we get

$$W(x \mid y) = \frac{p(y \mid x)p(x)}{p(y)q(x \mid y)} \tag{5.5}$$

$$\propto \frac{p(y \mid x)p(x)}{q(x \mid y)}. \tag{5.6}$$

If we then define

$$W^*(x \mid y) = \frac{p(y \mid x)p(x)}{q(x \mid y)}, \tag{5.7}$$

we can use the fact that the weights must sum to 1 to retrieve the values of $W(x \mid y)$ from this form that avoids the troublesome $p(y)$ term using

$$W(x_i \mid y) = \frac{W^*(x^i \mid y)}{\sum_{x^i} W^*(x^i \mid y)}. \tag{5.8}$$

We would like a way to be able to update the weight of a particle sequentially, so that given a particle, its latest sample and its previous weight we can calculate its new weight. If we write the state sequence of the i^{th} particle as $x^i_{0:t}$, the observation sequence as $y_{0:t}$ and the unnormalized weight of a particle i at time t as $W^*_i(t)$, then, from equation 5.7 we get

$$W^*_i(t) = \frac{p(y_{0:t} \mid x^i_{0:t})p(x^i_{0:t})}{q(x^i_{0:t} \mid y_{0:t})} \tag{5.9}$$

$$= \frac{p(y_{0:t-1}, y_t \mid x^i_{0:t-1}, x_t)p(x^i_{0:t-1}, x_t)}{q(x^i_{0:t-1}, x_t \mid y_{0:t-1}, y_t)} \tag{5.10}$$

$$= \frac{p(y_{0:t-1} \mid y_t, x^i_{0:t-1}, x_t)p(y_t \mid x^i_{0:t-1}, x_t)p(x^i_t \mid x^i_{0:t-1})p(x^i_{0:t-1})}{q(x_t \mid x^i_{0:t-1}, y_{0:t})q(x^i_{0:t-1} \mid y_{0:t})}, \tag{5.11}$$

where the final step above is made by several applications of the chain rule of probability. If we now use the conditional independence given by the model structure in figure 3.2 and make the requirement that $q(x^i_{0:t-1} \mid y_{0:t}) = q(x^i_{0:t-1} \mid y_{0:t-1})$, i.e. that the proposal at time $t-1$ does not depend on the observation at time t (a very mild requirement), we can simplify equation 5.11 to

$$W^*_i(t) = \frac{p(y_{0:t-1} \mid x^i_{0:t-1})p(x^i_{0:t-1})}{q(x^i_{0:t-1} \mid p(y_{0:t-1})} \times \frac{p(y_t \mid x^i_t)p(x^i_t \mid x^i_{t-1})}{q(x^i_t \mid y_{0:t}, x^i_{0:t-1})} \tag{5.12}$$

$$= W^*_i(t-1) \times \frac{p(y_t \mid x^i_t)p(x^i_t \mid x^i_{t-1})}{q(x^i_t \mid y_{0:t}, x^i_{0:t-1})}. \tag{5.13}$$

This gives us a recursion with which to update the particle weight. The terms in the rightmost fraction of equation 5.13 are all things that are specified by the model: $p(y_t \mid x^i_t)$ is the observation function, $p(x^i_t \mid x^i_{t-1})$ is the state transition function and $q(x^i_t \mid y_{0:t}, x^i_{0:t-1})$ is the sample function.

Things are simplified further if we choose the state transition function to be the sample function. Such a choice gives a filter known as a *bootstrap filter* (Gordon et al. (1993)). Then, equation 5.13 simplifies to

$$W^*_i(t) = W^*_i(t-1) \times p(y_t \mid x^i_t), \tag{5.14}$$

and the particle filter algorithm can be given as

Algorithm 2 Basic bootstrap particle filter (no resampling)

while observations available **do**
  for all particles x^i **do**
    Sample: $x_t^i \sim p(\cdot \mid x_{t-1}^i)$
    Re-weight particle: $W_i^*(t) = W_i^*(t-1) \times p(y_t \mid x_t^i)$
  end for
  $t = t + 1$
end while

5.2 Variable Rate Particle Filter

We want to adapt the basic particle filter given in algorithm 3 to work with our variable rate problem, where the state sequence is a sequence of jump times. Since a jump does not always take place between the last observation and the current one, for a particular particle this sequence only changes if that particle experiences a jump between the last observation time and the current one.

We can do this by only updating the state sequence of a particle when the sample function generates a jump for that particle between the previous and current observation times. If, as in the bootstrap filter above, we use the state transition function as the sample function, that means sampling from the jump-time distribution in order to update the particles' state sequences.

The particle weight at time t is given by equation 5.9 adapted for the jump time sequence to be the particle state:

$$W_i^*(t) = \frac{p(y_{1:t} \mid \tau_{1:N_t^i}^i) p(\tau_{1:N_t^i}^i)}{q(\tau_{1:N_t^i}^i \mid y_{1:t})} \tag{5.15}$$

$$= \frac{p(y_t \mid y_{1:t-1}, \tau_{1:N_t^i}^i) p(y_{1:t-1} \mid \tau_{1:N_t^i}^i) p(\tau_{1:N_t^i}^i)}{q(\tau_{1:N_t^i}^i \mid y_{1:t})} \tag{5.16}$$

where N_t^i is the number of jump times in the state sequence of the i^{th} particle at time t.

Now, if there is no jump for particle i from time $t-1$ to t then $N_t^i = N_{t-1}^i$ and so

$$p(\tau_{1:N_t^i}^i) = p(\tau_{1:N_{t-1}^i}^i), \tag{5.17}$$

$$p(y_{1:t-1} \mid \tau_{1:N_t^i}^i) = p(y_{1:t-1} \mid \tau_{1:N_{t-1}^i}^i). \tag{5.18}$$

If $q(\tau_{1:N_t^i}^i \mid y_{1:t}) = q(\tau_{1:N_t^i}^i \mid y_{1:t-1})$, which is a mild condition on the sample function that it not be dependent on the new observation (obviously satisfied by the bootstrap filter, where the sample function does not depend on the observations at all), then

$$W_i^*(t) = W_i^*(t-1) \times p(y_t \mid y_{1:t-1}, \tau_{1:N_t^i}^i). \tag{5.19}$$

Here y_t is the last observation in the final neighbourhood of observations for particle i; we will return to this below.

If, on the other hand, particle i's state sequence does contain a jump from time $t-1$ to t then $N_t^i = N_{t-1}^i + 1$. Then

$$p(\tau^i_{1:N_t^i}) = p(\tau^i_{N_t^i} \mid \tau^i_{N_{t-1}^i}) p(\tau^i_{1:N_{t-1}^i}), \tag{5.20}$$

$$p(y_{1:t-1} \mid \tau^i_{1:N_t^i}) = p(y_{1:t-1} \mid \tau^i_{1:N_{t-1}^i}), \tag{5.21}$$

$$q(\tau^i_{1:N_t^i} \mid y_{1:t}) = q(\tau^i_{N_t^i} \mid \tau^i_{1:N_t^i-1}, y_{1:t}) q(\tau^i_{1:N_t^i-1} \mid y_{1:t}) \tag{5.22}$$

The second of the above observations can be seen by noting that the last observation at $t-1$ is before the jump at $\tau^i_{1:N_t^i}$. So, if $q(\tau^i_{1:N_t^i-1} \mid y_{1:t}) = q(\tau^i_{1:N_t^i-1} \mid y_{1:t-1})$ similar to before, then

$$W_i^*(t) = W_i^*(t-1) \times \frac{p(y_t \mid y_{1:t-1}, \tau^i_{1:N_t^i}) p(\tau^i_{N_t^i} \mid \tau^i_{N_{t-1}^i})}{q(\tau^i_{N_t^i} \mid \tau^i_{1:N_t^i-1}, y_{1:t})}, \tag{5.23}$$

which, if we are using the bootstrap filter where $q(\tau^i_{N_t^i} \mid \tau^i_{1:N_t^i-1}, y_{1:t}) = p(\tau^i_{N_t^i} \mid \tau^i_{N_{t-1}^i})$, gives

$$W_i^*(t) = W_i^*(t-1) \times p(y_t \mid y_{1:t-1}, \tau^i_{1:N_t^i}), \tag{5.24}$$

just as in equation 5.19. Here y_t is the first observation in a new neighbourhood.

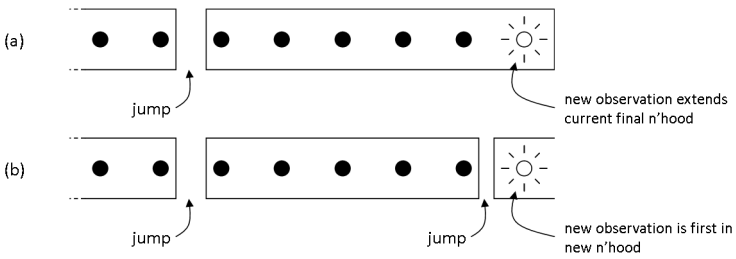

Figure 5.3: The neighbourhood to which the new observation belongs if (a) there is no jump from $t-1$ to t and (b) there is a jump from $t-1$ to t

These equations tell us how to update the particle weights, but we must still calculate the observation function $p(y_t \mid y_{1:t-1}, \tau^i_{1:N_t^i})$. This is straightforward, however, since it is simply the marginal corresponding to the last observation of the observation probability for the final observation neighbourhood, as calculated in equation 4.22. As we saw in section 4.2 this can be calculated using the Kalman filter.

Putting this all together gives us a variable rate particle filter algorithm:

In the first step of the above algorithm we have to sample from jump time distribution given that there has been no jump up to the last observation time (in this case $t-1$). This is given by

$$p(\tau_i \mid t-1) = \frac{p(\tau_i \mid \tau_{i-1})}{p(\tau_i > t-1)}. \tag{5.25}$$

In our initial work we use the 'memoryless' exponential distribution as the jump time distribution, so this problem is trivial (it is simply the exponential distribution from $t-1$), but with other distributions it might be more involved.

Algorithm 3 Variable rate bootstrap particle filter (no resampling)

while observations available **do**
  for all i in particles **do**
    Sample: $\tau^* \sim p_{\text{jump-time}}(\cdot \mid t-1)$ {sample from jump-time distribution}
    if $\tau^* < t$ **then**
      $N_t^i = N_{t-1}^i + 1$ {increment number of jumps for this particle}
      $\tau_{N_t^i}^i = \tau^*$ {add the new jump to the jump times for this particle}
    end if
    Re-weight particle: $W_i^*(t) = W_i^*(t-1) \times p(y_t \mid y_{1:t-1}, \tau_{1:N_t^i}^i)$
  end for
  $t = t + 1$
end while

The variable rate filter described here makes the assumption that there is at most one jump in each inter-observation period. For most distributions of jump-times, including our exponential distribution, this need not be the case. However, if the expected inter-jump time is much longer than the expected inter-observation time then the at-most-one-jump assumption is almost always true and, even when not is still a reasonably good approximation. If we were dealing with a case where observations were much less frequent that the expected jump time then the algorithm given could easily be modified to include the possibility of arbitrarily many jumps between observations. This also applies to the MCMC estimation method.

5.3 Resampling

Unfortunately, the above algorithms have a fatal problem. As more observations arrive and more steps of the algorithm are taken it can be shown that the variance of the importance weights increases (Doucet et al. (2000)), and can only continue to do so. This means that eventually (and in practice often quite quickly) almost all of the weight ends up being placed on a single particle, leading to a very poor representation of the state pdf we are trying to estimate (since it essentially being represented as a delta function at the location of the single 'good' particle). This is shown in figure 5.4.

The problem can be overcome by *resampling*, so that from one stage to the next 'good' particles (having high weight) are propagated giving rise to many offspring whereas 'bad' particles are allowed to die out. When we resample we want the new set of particles to be drawn from the current posterior probability distribution. Since this distribution is represented by our weighted collection of particles we can use this as the distribution from which to draw our new samples. There are several approaches to resampling (see section 2.3), but the simplest is to duplicate particles based on their current weight. Under this type of scheme, particle weights are used as a probability with which to select the particle for propagation to the next round, with selection multiple times being possible. The new particle set is thus a series of samples from our representation of the posterior distribution and so is itself a representation of the posterior distribution.

For the variable rate filter above we use a resampling scheme based on a

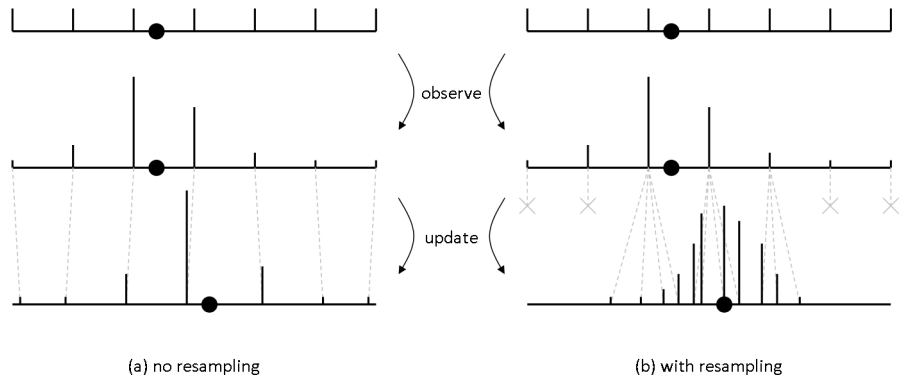

Figure 5.4: Evolution of the particle filter (a) without resampling and (b) with resampling. Resampling concentrates particles in the areas of highest probability (but note how range is reduced).

similar idea, taken from Godsill et al. (2007). Here the idea is to duplicate every particle a deterministic number of times based on its weight, so that the number of offspring for particle i in the next generation M_{t+1}^i is

$$M_{t+1}^i = \max\left(1, \lfloor M^{\text{tgt}} W_i(t) \rfloor\right), \tag{5.26}$$

where M^{tgt} is a target number of particles (i.e. roughly the number of particles we want to exist). The weight of each particle in the new set is given by

$$W_j(t) = \frac{W_i(t)}{\max\left(1, \lfloor M^{\text{tgt}} W_i(t) \rfloor\right)} \qquad \forall j \in \{\text{offspring of } i\} \tag{5.27}$$

$$= \begin{cases} W^i(t) & W_i(t) < 1/M^{\text{tgt}} \\ W_i(t)/M^{\text{tgt}} & W_i(t) \geq 1/M^{\text{tgt}} \end{cases} \qquad \forall j \in \{\text{offspring of } i\} \tag{5.28}$$

Under a naive version of this scheme, the number of particles is non-decreasing since every particle is propagated at least once. However, only some of a particle's offspring will experience a jump in the period between $t-1$ and t, and those that do not will all be the same (since their jump time sequences will be those of their parent particle). So, the increase in particle numbers can be mitigated by collapsing all non-jumping offspring of a particle to a single particle. This collapsed particle represents a number of offspring, so its weight must be multiplied by the number of particles it represents. Adding this method of resampling to algorithm 5.2 gives algorithm 5.3.

Resampling has the obvious advantage of making the particle filter algorithm work for more than a few steps. However, since it causes heavily weighted particles to be duplicated at the expense of more lightly weighted ones it means that particle diversity can be reduced as time goes on, with only particles from the most probable regions managing to survive and propagate. The duplication of particles also means that the particles are not independent of each other and this makes it much harder to provide useful theoretical bounds on the algorithm performance (Crisan and Doucet (2002)).

Algorithm 4 Variable rate bootstrap particle filter (with resampling)

P_0 = initial (prior) particle distribution {particle set}
$W_i(0) = {}^1/_{|P_0|}$ $\quad \forall i \in P_0$
$t = 1$
while observations available **do**
  for all $p \in P_{t-1}$ **do**
    $M_t^p = \max(1, \lfloor M^{\text{tgt}} W_p(t-1) \rfloor)$ {number of offspring for particle p}
    $M_0 = 0$ {number of non-jumping offspring}
    $J_p = \emptyset$ {set of jumping offspring}
    for $i = 1$ to M_t^p **do**
      Sample: $\tau^* \sim p_{\text{jump-time}}(\cdot \mid t-1)$ {sample from jump-time distribution}
      if $\tau^* < t$ **then**
        q = new particle
        $q.\tau = p.\tau \cup \tau^*$ {q has parent's jumps plus new jumps}
        $J_p = J_p \cup q$ {add q to set of jumping particles}
      else
        $M_0 = M_0 + 1$ {increment count of non-jumping offspring}
      end if
    end for
    Reset $M_t^p = |J_p| + 1$ {since we collapse all non-jumping particles}
    for all $q \in J_p$ **do**
      $W_q(t-1) = \frac{W_p(t-1)}{M_t^p}$ {assign an even weight to each offspring}
    end for
    r = new particle {particle r represents all non-jumping offspring}
    $r.\tau = p.\tau$ {r jumps as parent's}
    $W_r(t-1) = M_0 \frac{W_p(t-1)}{M_t^p}$ {assign weight to represent M_0 non-jumping particles}
    $C_p = J_p \cup \{r\}$ {C_p is set of children of p}
    *** End of resampling ***
    for all $q \in C_p$ **do**
      Re-weight particle: $W_q^*(t) = W_q(t-1) \times p(y_t \mid y_{1:t-1}, \tau^i_{1:N_t^i})$
    end for
  end for
  $P_t = \bigcup_{p \in P_{t-1}} C_p$ {new particle set is offspring of all old particles}
  $W_p(t) = \frac{W_p^*(t)}{\sum_{p \in P_t} W_p^*(t)}$ $\quad \forall p \in P_t$ {normalize particle weights}
  $t = t + 1$
end while

5.4 Parameters

In our proposed system, parameter estimation is not the primary function of the particle filter, instead the MCMC estimation is intended to estimate parameters over a preceding batch of observations and the particle filter is then intended to estimate the hidden state as observations arrive.

There are two issues relating to parameters in the particle filter that we need to address. The first is incorporating the parameter estimates produced by the MCMC estimation into the particle filter. The second is allowing the particle filter to update the parameter estimates based on further observations.

MCMC produces a series of samples drawn from the posterior distribution of the hidden states and true parameters. That means that its parameter estimates take the form of a series of samples which, taken together, represent the posterior parameter distribution. The simplest way of incorporating the parameter estimates from MCMC into the particle filter is simply to use the *maximum a posteori* (MAP) parameter estimate (i.e. the parameter values that maximize the probability density marginalized for the particular parameter).

If we want to incorporate the more nuanced sample-based parameter estimates into the particle filter we must extend the filter itself by extending its state-space to include the parameters. This will also address our second problem of allowing the particle filter to update the parameter estimates. The idea is to include the parameters θ in the state held by each particle. By initializing the particle collection with a selection of parameter values representative of the distribution determined by MCMC, the MCMC parameter estimation can be incorporated into the start of the filter.

Each update step of the filter must include a step which samples new parameter values for each particle in the next generation. Since the post-resampling particles have a distribution representative of the pre-resampling particles it is possible to separate the jump-time and parameter sampling and resampling steps to allow the preservation of the ability to collapse non-jumping particles to a single particle. This is not necessary; new jump times and parameter values could be sampled for all offspring and resampling could occur after this, however in this case further steps would need to be taken to prevent particle number explosion. The particle weights are then updated as before, with the observation function value for a particular particle being calculated using that particle's parameter values.

Parameter estimation in particle filters is difficult (see section 2.3), so as these methods have not yet been implemented they are not covered here in further detail.

Chapter 6

Results

The results in this section show the output from running the MCMC algorithm described in the previous sections on data generated from model described in equations 3.1-3.3. The data was generated using the integral form of the system given in equation 3.38. In all the results shown the jumps in the data were co-located in both processes, but the sizes of the jumps were independent.

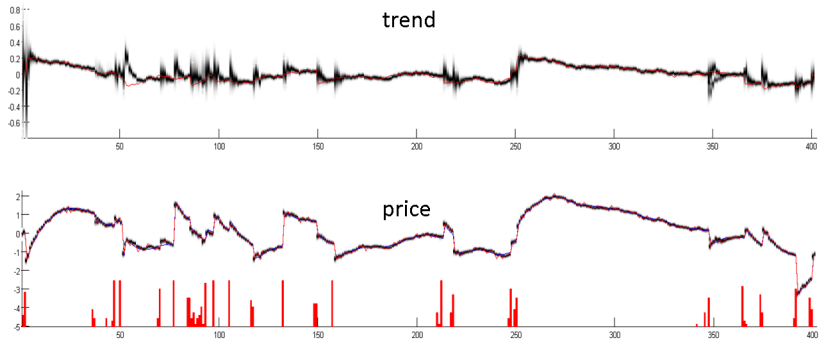

Figure 6.1: Determination of jump times with fixed correct parameter values. The red lines show the true value of the processes. The grey shading shows the probability density of the MCMC estimate of the state at each point. The red bars on the price graph are a histogram of estimated jump times.

The first set of results in figure 6.1 shows the performance of the system in estimating the number of jumps and their times in a data series of 400 observations generated from the state space model. The parameters are given and fixed and the initial jump distribution was 30 evenly spaced jumps. In this case, the MCMC estimator does a reasonable job of finding the jump times (shown by the red bars, which give a histogram of the estimated jump times over the MCMC samples. The estimator here was run for 2000 iterations, with a burn-in period of 500 iterations. An interesting feature of these results is the ambiguity between fast-reverting trend changes and jumps. This can be seen by the presence of large spikes in the estimate of the trend variable which quickly decay away at the point of each jump in the 'price' process. This could cause prob-

lems for the MCMC estimation process because ambiguity can lead to ridges and multiple modes in the posterior probability function. In this case a ridge might run between a short jump in the price process with a large jump in the trend process and a large jump in the price process with no jump in the trend process. Misidentifying jumps in the trend process could also lead to problems correctly identifying the mean-reversion rate in that process (which has been one of the problems with parameter estimation - see below) since misidentifying a large jump in the trend process would also require a misidentification of the mean reversion speed to be much greater than its true value.

The second set of results (figure 6.2) shows the estimation of parameters of the model when the jump times are given. Here the MCMC chain was run for just over 2000 iterations and started with randomly chosen values (in a reasonable range) for each parameter. The parameters were updated individually, one at a time, in random order. The priors on the parameters were uninformative. Figure 6.2 shows the entire evolution of each parameter value throughout the chain. The diagram appears to show that the parameter estimates for all parameters have converged within about 1000 iterations. They all appear to be close to their true values (shown by the red lines). This shows that with the jump times given the MCMC estimator is capable of finding parameter values.

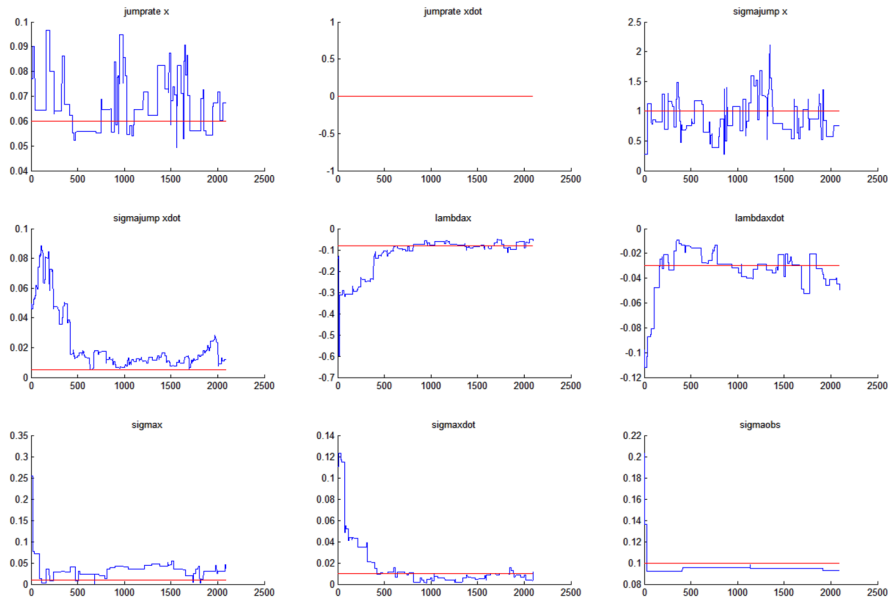

Figure 6.2: Parameter estimation when the correct jump times are given. The parameters shown are, from the top left to bottom right are λ^x_{jump}, ($\lambda^{\dot{x}}_{\text{jump}}$,), σ^x_{jump}, $\sigma^{\dot{x}}_{\text{jump}}$,$\lambda_x$, $\lambda_{\dot{x}}$, σ_x, $\sigma_{\dot{x}}$ and σ_{obs}. $\lambda^{\dot{x}}_{\text{jump}}$ was defined as zero because jumps were co-located, so driven off a single Poisson process. The red lines show the true parameters of the process used to generate the data.

The final two sets of results in figures 6.3 and 6.4 show the attempted estimation of jump times and sizes and system parameters by the MCMC estimator.

The data used here was again generated from the state space model and contained 400 evenly spaced observations.

For each jump-time updating step of the chain, around five parameter updating steps of the chain were made. This was found to lead to more stable estimations through experiment, since otherwise the number of jumps seemed to collapse rapidly to only a handful.

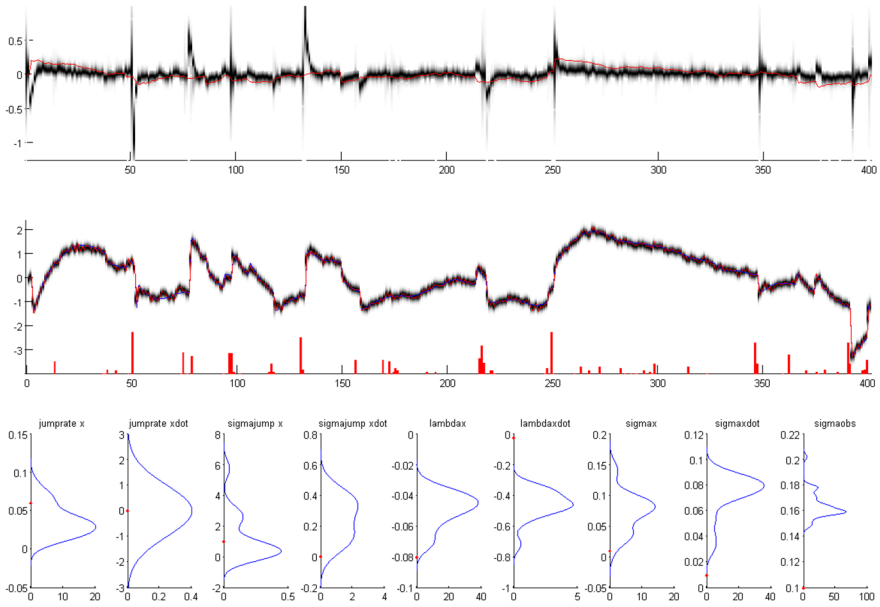

Figure 6.3: State and jump-time estimation for the MCMC estimator attempting to estimate both state (including jump times) and parameter values. The results shown are those after a 1000 iteration burn-in. The graphs in the bottom panel show kernel smoothed probability density estimates for the parameters (in the same order as in figure 6.2), with the red dots showing their true values.

Figure 6.3 shows the trend and price process and the MCMC estimation of them. Again the amiguity between price and trend jumps is apparent and is indeed more pronounced. From the histogram of jump times underneath the price process, it appears that the jump times have been estimated quite well. Indeed, aside from the ambiguity between the price and trend jumps, the state estimation appears to be quite good.

Unfortunately, the parameter sequences shown in figure 6.4 tell a different story. The MCMC was run for around 4000 steps and, as with the previous set of results, the priors on the parameters were uninformative. Here some of the parameters still appear to converge within about 1200 steps ($\lambda^x_{\text{jump}}, \sigma^x_{\text{jump}}$ and λ_x) but several other parameters, notably $\sigma_{\dot{x}}$ and $\lambda_{\dot{x}}$ do not appear to show any sign of converging. $\lambda_{\dot{x}}$ has reached a value of nearly -1 by the end of the run (and other experiments showing similar results suggest that it will continue beyond that). This is saying that the trend process mean-reverts in less than one unit of time, which contributes to (or is perhaps fed by) the ambiguity between

jumps in the trend and price process. Once the mean reversion rate reaches such levels, the trend process is effectively useless as a trend process, since any innovation in it decays away immediately. It becomes, in effect, a second jump process. Numerous experiments with similar results have been conducted.

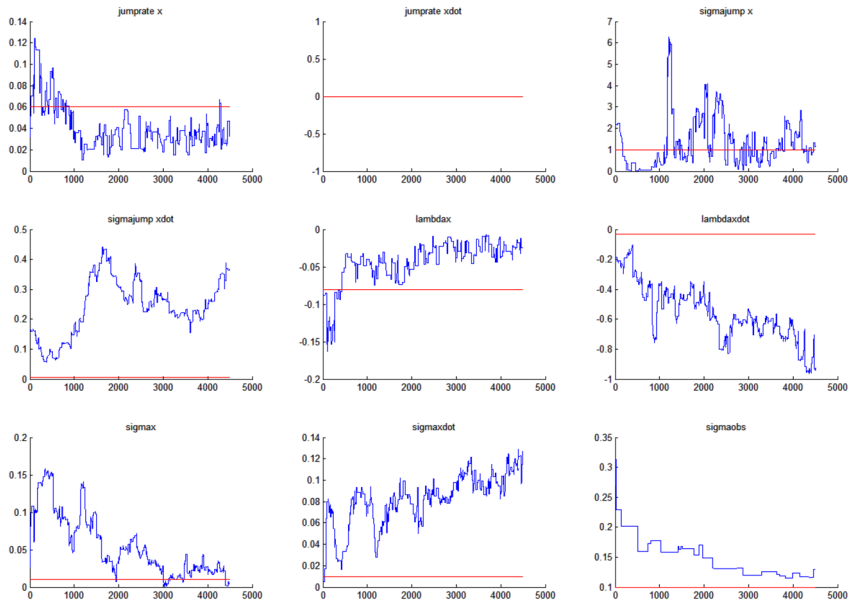

Figure 6.4: Parameter estimation by the MCMC estimator attempting to estimate both state and parameters. The order of the figures is the same as in figure 6.2, with the red lines showing the parameter values for the model from which the data was generated.

In many of the experiments conducted the parameters $\sigma_{\dot{x}}$ (the trend variance) and $\lambda_{\dot{x}}$ (the mean reversion rate of the trend) appeared to be negatively correlated. Figure 6.5 examines this relationship for the two parameters, with all jump times and other parameters being fixed to the values from which the data was generated. The plot shows a fairly marked ridge in the objective function with respect to these two parameters, suggesting that they are indeed correlated (negatively in this case). Such ridges in the function being targeted are known to cause difficulties with MCMC estimators and can be a sign of over-parameterization of the model (meaning that the parameters can be ambiguous). In the case of figure 6.5 the posterior probability still has a maximum near the correct values and does not seem to be multi-modal, so an MCMC method would still be expected to find the correct value of these two parameters if started from here (and indeed experiments have shown that this is the case). However, we have also noticed posterior probabilities with respect to these two parameters looking something like the sketch shown in figure 6.6. In this case the MCMC estimator can easily find itself in the much larger area of high probability in the lower-right corner. The function appears to continue to increase for a very long way in this area and so the parameter values then tend

to diverge from their true values as seen in figure 6.4. This seems to be a problem in our estimator. It might be possible to solve it with informative priors, or a more advanced MCMC algorithm (see the following section on future work).

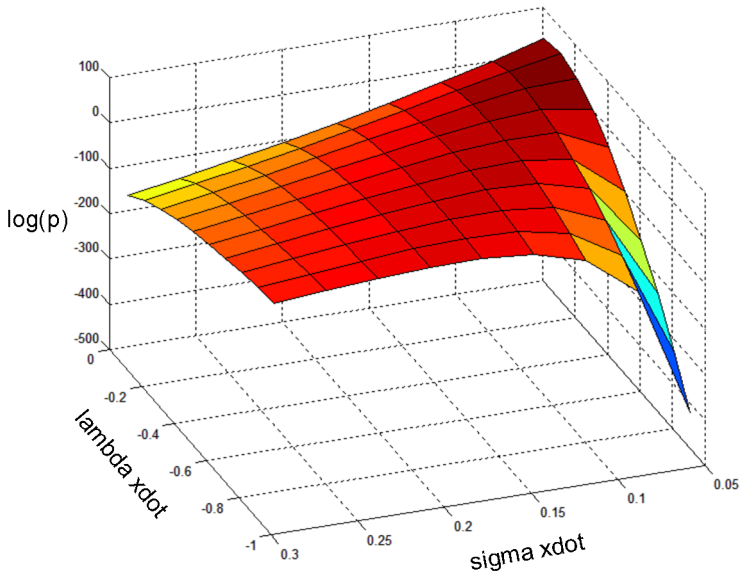

Figure 6.5: The (log) posterior probability function of a sample with the correct jump times for a range of parameter values for $\lambda_{\dot{x}}$ and $\sigma_{\dot{x}}$ (other parameters were fixed at the value used to generate the data). The true values for the parameters are $\lambda_{\dot{x}} = -0.03$ and $\sigma_{\dot{x}} = 0.01$

Estimations were attempted with troublesome blocks of parameters (those that seemed to exhibit strong correlation, such as those in figure 6.5) being updated together, but this yielded similar results to those shown in figure 6.4.

In all of the above MCMC estimations, each step of the MCMC chain took around a third of a second on a desktop PC.

Since the MCMC estimator has not yet proved capable of estimating the parameters of the process even of data known to be generated from the same model as the state space model of the estimator, it has not been applied to real data. See the following section for suggestions as to how to remedy this problem.

The results of this work so far can only be described as disappointing but obviously we hope to remedy this with future work.

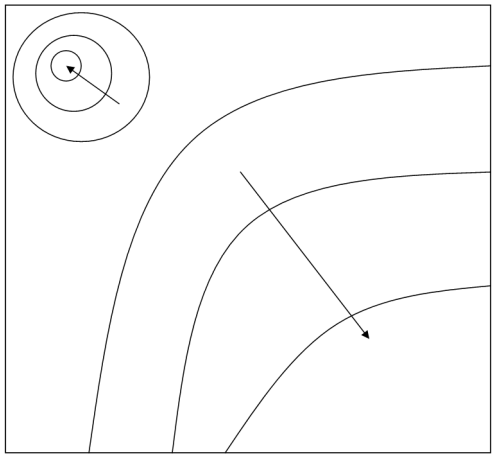

Figure 6.6: Sketch of the contours of the posterior probability as a function of $\lambda_{\dot{x}}$ and $\sigma_{\dot{x}}$ for some parameter/state values. The arrows point to areas of higher probability. The true parameters lie near the top left corner.

Chapter 7

Future Work

This section contains some ideas for future extensions to the work presented here. These have been divided according to the time-scales on which they are expected to be tackled.

7.1 Near Term

The near-term extensions can be divided into two main areas: making the current system work properly, and enhancing the current system to work with (and effectively model) real data of interest.

7.1.1 Making It Work

The current MCMC system has difficulty identifying both parameters and jump times simultaneously and does not seem to converge in this case. There are a number of possible improvements that might help with this. One option is to impose a more informative prior on the model parameters, as the current prior is uninformative. Possible priors that would be useful include priors on the mean reversion coefficients to 'encourage' them into the region below between 0 and -0.25, particularly in the trend process. This is where they would intuitively be expected to be since more negative values lead to almost full mean reversion within the space of a few observations and so it is possible that jumps in the value process are being incorrectly ascribed to large jumps in the trend process, which quickly decay. Another useful prior might be to link the jump and diffusion variances, making the jump variance some multiple k of the diffusion variance, and then placing a prior distribution on the value of k to make large multiples more likely (since we want the 'jumps' to be clearly distinguished from the normal moves of the diffusion process). This is a form of reparameterization of the model, and it might be possible to find others so as to reduce correlation of model parameters (as suggested in Gilks et al. (1996)).

The use of adaptive MCMC methods as in Roberts and Rosenthal (2008) and Haario et al. (2006) might improve the proposal distributions by discovering and taking account of the covariance that exists between the model parameters. Simulated tempering algorithms, such as that of Brooks et al. (2006) (which applies to reversible jump methods) might also be of use in achieving convergence.

The particle MCMC method of Andrieu et al. (2010) might be an alternative to adaptive methods in order to improve proposal generation within our MCMC algorithm. An alternative to MCMC sample generation altogether might be the SMC sampler of Del Moral et al. (2006); it would be interesting to see how this sampler compared to MCMC algorithms. Since the SMC sampler can be adapted for sequential sampling it might prove an effective alternative to the current MCMC/particle filter combination, but this depends on its ability to do batch parameter estimation as effectively as MCMC (since there is a wealth of historical data that we want to use in model calibration).

Since in our current work we have not successfully managed MCMC parameter estimation, we have not yet attempted to incorporate MCMC parameter estimates into the particle filter. This is obviously a key step in building an on-line estimation system. It is also likely to be necessary to allow the particle filter to at least update its parameter estimates. The practical filter of Polson et al. (2008) or one of the MCMC particle methods of e.g. Gilks and Berzuini (2001) or Cappé et al. (2007) might be most effective in doing this, although the dynamic parameter method of e.g. Liu and West (2001) might be simpler to implement and perhaps sufficient for quite short runs of the filter. Maximum likelihood methods are not appropriate for an on-line application because they are batch rather than sequential methods.

7.1.2 Real Data

The application of the model and the estimation techniques is to the processing of high frequency financial data, particularly foreign exchange data at a trade-by-trade (tick) level. An obvious next step, made even more necessary by the troublesome and disputed nature of financial return data (see section 2.1), is a statistical examination of the data of interest.

One key feature of the data that is essential to understand is the nature of its tails. To this end we will attempt to estimate the tail index of the data using the estimator of Hill (1975). In light of this, the data should be examined to see which moments appear to converge and how fast they do so. This should give some insight into which moments of the data should be possible to estimate and model and which of these estimates should be stable (see for example Cont (2001)). Another interesting feature of the tails is whether or not they are symmetric. According to the literature we might expect the left tail to be heavier than the right tail (although this effect has been reported to be absent in foreign exchange data).

It will also be necessary to study the overall shape of the distribution. We want to determine whether it is skewed and what its overall shape is. An interesting question is whether the data appears to be from a power-law distribution - Clauset et al. (2009) provide a framework for testing and estimating power laws, which may be of use here. Both the unconditional shape of the distribution and its shape conditional on volatility should be examined. The latter can be achieved by first estimating the volatility with GARCH or stochastic volatility type models.

Another feature of the data that is important to understand is its autodependency structure. The autocorrelation and partial autocorrelation of the returns at various lags will be examined. The same analysis will be applied to the absolute returns and to the square of the returns and possibly also of higher

powers of the absolute returns, as in Ding et al. (1993). According to the literature we should expect to find a decaying autocorrelation in these autocorrelations and we will attempt to verify the power-decay behaviour observed by Liu et al. (1997). A different way of looking at the data is as self-similar, in the manner of e.g. Muzy et al. (2000). Taking this view, it will be interesting to look at the Hurst exponent of the time series as a way of measuring its autodependency.

The realized variance of the return data is of interest because it is a common estimator of volatility. It can be troublesome to estimate at the highest frequencies due to market microstructure noise Zhang et al. (2005) and with intermittent data Hansen and Lunde (2005). The methods of doing so for ultra-high frequecny data are likely to have relevance to the estimation of other powers of the absolute return in such data.

Financial return data including intraday foreign exchange data has been reported to exhibit strong seasonality (see e.g. Dacorogna and Gencay (2001) and section 2.1) and so an examination of the seasonality our data is necessary. In fact, the removal of seasonal effects before the analysis of the data is traditional in econometric analysis (since the aim is to create a stationary time series, which can be analysed using standard methods).

The idea of this statistical study of the data is to form an opinion about it and then to use this to narrow the space of models that capture those features of interest that we have identified.

7.2 Medium Term

In the medium term we would like to build and estimate models which fit well to real data.

7.2.1 Models

After conducting a thorough statistical analysis of the data the aim will be to create a model that can describe the most important features of interest in it, whilst still being analytically tractable. There are numerous models already in the literature (see section 2.1) and so these, possibly with significant adaptations or in some combination, could form the basis of the new model. The stochastic volatility jump-diffusion models of e.g. Duffie et al. (2000) are good possible candidates.

From the literature we can expect to observe time varying persistent volatility (heteroskedasticity) and so some manner of introducing this into our model will probably be necessary. The traditional approach has been the stochastic volatility approach of e.g. Heston (1993) or the GARCH approach of Bollerslev (1986). The former of these has found more success in the option pricing literature whilst the second has found more success in econometrics but complicates option pricing (see e.g. Duan (1995)). Both approaches could be investigated to introduce time varying persistent volatility.

An alternative approach might be to use a jump process as a mechanism for introducing heteroskedasticity into the model. In this approach a jump time distribution that allowed clustering of jumps could be used to have a similar effect to a stochastic volatility model. The feasibility of this and its ability to

fit the data need to be investigated, but it has the advantage that it can be implemented in the current framework as presented in this report.

Another issue our model might have to address is the (likely) heavy tails of the distribution of returns. Traditional approaches to this have been the use of various distributions with heavy tails (see section 2.1), starting with the work of Mandelbrot (1963). In the current framework a move away from the Gaussian diffusion sections would considerably increase both the mathematical complexity (the processes involved are no longer Gaussian and so their integration becomes much trickier) and the computational burden of estimating the model, since the Kalman filter could no longer be used between jumps. Several approaches exist for dealing with this.

Sticking within the finite jump diffusion framework(i.e. diffusion plus jumps of which there are almost surely a finite amount in any finite interval), it might be possible to accommodate the observed statistical features of the returns process by altering the jump-size distribution to fit with the data. This could give a jump-diffusion process power-law tails, for example. It is also possible that multiscale jumps with only a finite number of scales could provide a simple way to approximate heavy-tailed distributions, but these (and even infinite mixtures) suffer from the theoretical limitation imposed by extreme value theory which would mean that their tails (and thus the tails of the entire process) would still be light. However, modelling jumps with a different jump size distribution (for example an exponential distribution) is possible in the framework given in this report if approximate the jump-sizes using a sample-based representation.

So, it might be possible to produce a relatively simple model within a modified version of the current framework that could model heavy tails and volatility clustering by suitable modifications to the jump process.

Alternatively, if this does not prove successful, Lévy processes might be appropriate. These include processes such as the Variance Gamma process and the log-stable processes as well as the inverse Gaussian processes used by Barndorff-Nielsen (1997a). Li et al. (2008) uses such processes as a model for stock price returns and estimates them using an MCMC method. Though Lévy processes are themselves iid, it is possible to build stochastic volatility into the models using a stochastic time change (Carr and Wu (2004)). Lévy processes also have the advantage that a well-known integral calculus exists for them (e.g. Barndorff-Nielsen and Shepherd (2001)), based around their decomposition into a diffusion process, a point process and a pure jump martingale with countable jumps, which means that processes based on them are mathematically tractable.

A more detailed survey of the literature surrounding Lévy processes in finance will be necessary as will a significant investment of time in understanding the associated mathematics.

It might be more satisfactory to model the individual ticks directly, taking a discrete approach to modelling the process as in e.g. Rydberg and Shephard (2003). In such models the connection between volatility and volume is relevant and might be sufficient to account for observed seasonality and stochastic volatility. It might be possible to build discrete models using *hidden Markov models* (HMM) and then use regime switching to model trends in the data.

7.2.2 Model Evaluation Framework

Since we plan to develop models of the data, we would like to be able to evaluate their quality. A possible way to this in a Bayesian context is to use the *Bayes factor*, reviewed in detail by Kass and Raftery (1995). These are the basis of the Bayesian approach to hypothesis testing, which in the context of model evaluation can be used to assess the evidence in favour of one model compared to another. They are the ratio of the Bayesian posterior probabilities of one hypothesis to the other given the data. If we assume an uninformative prior this is equal to ratio of the probability of the data given each of the models, and so is similar in practice to the traditional likelihood ratio test.

The problem with such hypothesis testing approaches is that they are fundamentally parametric, requiring models of the data to evaluate against each other. An alternative that does not require the model be tested against another model is to look at the residuals of the process when compared to the model and see whether they have any structure. If the model is correctly specified the residuals should correspond to the model's noise terms and so should come from the same distribution as those. If they exhibit, for example, serial correlations, then it suggests that an effect is present which is not being modelled. Such techniques as examining the autocorrelation of the residuals are standard in time series econometrics (see for example Lütkepohl and Krätzig (2004)).

7.3 Longer Term

In the longer term we would like to take the models developed for real data and extend them in two main directions: trading models and multivariate models.

7.3.1 Trading Models

One possible application of the models developed is to trading. This requires adding a decision layer on top of the model to make optimal decisions about buying and selling. With state space models this, in its simplest form, is relatively easy to construct, since we have a probabilistic estimate of the current state, from which, in conjunction with the transition and observation model, we can make a probabilistic estimate of future observations (prices). The simplest trading model might look one period ahead, making a decision to buy if there was a high estimated probability of the next observed price being above the current one, sell if there was a high estimated probability of it being below the current one and do nothing if the estimated probability was inconclusive. The model could be enhanced by adding in control over the *value at risk* (VaR) that it was allowed (based on its current estimates) and by imposing restrictions on how long it could keep open positions. This might lead to a multistage stochastic optimization system in the spirit of e.g. Consigli and Dempster (1998).

Since trading is a possible application of the model, this should be bourne in mind during its development. A trading model must be able to form an opinion about whether it can make money by buying or selling in future periods. This means that it *must not* be a (discounted) martingale! Hence, for a trading model it is important to include some sort of 'trend' component that allows for some dependency in the returns over time. If there is no dependence of future prices on the past (not just in an absolute or squared sense) then it is simply

not possible to make profitable trading systems. This may indeed be the case, but if we intend to try, we should not build a model that precludes it. The existence of high-frequency algorithmic trading funds suggests that profitable trading systems might be possible, though it does not necessarily mean it is possible in a single asset context.

Any trading models should be evaluated for profitability and risk against a common benchmark for the market in question the period of the test. It should also be evaluated against simple trading rules in order to gauge whether it can outperform relatively naive analysis (for example moving average based rules, which popular in technical analysis - see Brock et al. (1992)). The very existence and persistence of such rules might suggest profitable trading opportunities or might even create them. Since finance is a system composed of agents trading against each other, if enough of them believe something it might *become* true through their actions (e.g. bubble behaviour).

7.3.2 Multivariate Models

A different direction in which to develop the model is to a multivariate setting. In this case a group of assets and their codependency need to be modelled. There is a huge literature on the estimation of asset covariance (e.g. Voev and Lunde (2006) using high-frequency data), for example, and the more general codependency modelling provided by copula models (an introduction to which is given by Cherubini et al. (2004)) has recently attracted considerable attention. Bayesian estimation of state space models has been applied to multiple target tracking (e.g. Vermaak et al. (2005)) and this has been extended to stocks by Pang et al. (2008), so these approaches might be applicable to other assets.

An important aspect of any such modelling would be a comprehensive review of the literature on the stylized facts of multivariate models. This should then be followed by a statistical analysis of the data of interest. Then models could be proposed and tested against the data and each other, as with the univariate models. Finally, the a trading system could be extended to work on a multi asset model.

Bibliography

V. Akgiray and G.G. Booth. The stable-law model of stock returns. *Journal of Business & Economic Statistics*, 6(1):51–57, 1988.

S. L. Alder. Over-relaxation method for the monte-carlo evaluation of the partition function for multiquadratic actions. *Physical Review D - Particles and Fields*, 23 (12):2901–2904, 1981.

T.G. Andersen and T. Bollerslev. Intraday periodicity and volatility persistence in financial markets. *Journal of empirical finance*, 4(2-3):115–158, 1997a.

T.G. Andersen and T. Bollerslev. Heterogeneous information arrivals and return volatility dynamics: uncovering the long-run in high frequency returns. *Journal of finance*, 52(3):975–1005, 1997b.

T.G. Andersen and T. Bollerslev. Answering the skeptics: Yes, standard volatility models do provide accurate forecasts. *International Economic Review*, 39(4):885–905, 1998.

C. Andrieu and A. Doucet. Online expectation-maximization type algorithms for parameter estimation in general state space models. In *Proc. IEEE ICASSP*, volume 6, 2003.

C. Andrieu and É. Moulines. On the ergodicity properties of some adaptive MCMC algorithms. *The Annals of Applied Probability*, 16(3):1462–1505, 2006.

C. Andrieu, A. Doucet, and R. Holenstein. Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342, 2010.

Y. Atchadé and G. Fort. Limit theorems for some adaptive MCMC algorithms with subgeometric kernels. *Bernoulli*, 2009.

Y.F. Atchadé and J.S. Rosenthal. On adaptive markov chain monte carlo algorithms. *Bernoulli*, 11(5):815–828, 2005.

L. Bachelier. Théorie de la Spéculation, 1900. 1900.

O.E. Barndorff-Nielsen. Normal inverse Gaussian distributions and stochastic volatility modelling. *Scandinavian journal of statistics*, 24(1):1–13, 1997a.

O.E. Barndorff-Nielsen. Processes of normal inverse Gaussian type. *Finance and stochastics*, 2(1):41–68, 1997b.

O.E. Barndorff-Nielsen and N. Shephard. Impact of jumps on returns and realised variances: econometric analysis of time-deformed Levy processes. *Journal of econometrics*, 131(1-2):217–252, 2006.

O.E. Barndorff-Nielsen, T. Mikosch, and S.I. Resnick. *Lévy processes: theory and applications*. Birkhauser, 2001.

Ole E. Barndorff-Nielsen and Neil Shepherd. *Modelling by Levy processes for financial econometrics (in Levy Processes: Theory and Applications)*, pages 283–318. Applied Probability and Statistics. Springer / Birkhauser, 2001.

D.S. Bates. Jumps and stochastic volatility: Exchange rate processes implicit in Deutsche Mark options. *Review of financial studies*, 9(1):69–107, 1996.

D.S. Bates. Post-'87 crash fears in the S&P 500 futures option market. *Journal of econometrics*, 94(1-2):181–238, 2000.

A. Behr and U. Pötter. Alternatives to the normal model of stock returns: Gaussian mixture, generalised logF and generalised hyperbolic models. *Annals of finance*, 5(1):49–68, 2009.

C. Berzuini and W. Gilks. *RESAMPLE-MOVE Filtering with Cross-Model Jumps (in Sequential Monte Carlo Methods in Practice)*, chapter 6, pages 117–138. Springer, 2001.

F. Black. Studies of stock price volatility changes. In *Proceedings of the 1976 meetings of the business and economic statistics section, American Statistical Association*, volume 177, page 81, 1976.

F. Black and M. Scholes. The pricing of options and corporate liabilities. *Journal of political economy*, 81(3), 1973.

A. Blake and M. Isard. The condensation algorithm-conditional density propagation and applications to visual tracking. *Advances in Neural Information Processing Systems*, pages 361–367, 1997.

R.C. Blattberg and N.J. Gonedes. A comparison of the stable and student distributions as statistical models for stock prices. *Journal of Business*, pages 244–280, 1974.

T. Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics*, 31(3):307–327, 1986.

T. Bollerslev, J. Litvinova, and G. Tauchen. Leverage and volatility feedback effects in high-frequency data. *Journal of Financial Econometrics*, 4(3):353, 2006.

J-P Bouchaud. Power laws in economics and finance. *Quantitative Finance*, 1: 105–12, 2001.

W. Brock, J. Lakonishok, and B. LeBaron. Simple technical trading rules and the stochastic properties of stock returns. *Journal of Finance*, 47(5):1731–1764, 1992.

S. Brooks. Markov chain Monte Carlo method and its application. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 47(1):69–100, 1998.

S.P. Brooks and A. Gelman. General methods for monitoring convergence of iterative simulations. *Journal of Computational and Graphical Statistics*, 7 (4):434–455, 1998.

S.P. Brooks, Y. Fan, and J.S. Rosenthal. Perfect forward simulation via simulated tempering. *Communications in Statistics-Simulation and Computation*, 35(3):683–713, 2006.

R.S. Bucy and K.D. Senne. Digital synthesis of non-linear filters. *Automatica*, 7(3):287–298, 1971.

J.V. Candy. *Bayesian signal processing: classical, modern, and particle filtering methods*. Wiley-Interscience, 2009.

O. Cappé, S.J. Godsill, and E. Moulines. An overview of existing methods and recent advances in sequential Monte Carlo. *Proceedings of the IEEE*, 95(5): 899–924, 2007.

M. Carnero, D. Peña, and E. Ruiz. Persistence and kurtosis in GARCH and stochastic volatility models. *Journal of Financial Econometrics*, 2(2):319, 2004.

J. Carpenter, P. Clifford, and P. Fearnhead. Improved particle filter for nonlinear problems. *IEE Proceedings-Radar, Sonar and Navigation*, 146(1):2–7, 1999.

P. Carr and L. Wu. Time-changed Lévy processes and option pricing. *Journal of Financial Economics*, 71(1):113–142, 2004.

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

U. Cherubini, E. Luciano, W. Vecchiato, and G. Cherubini. *Copula methods in finance*. John Wiley & Sons Chichester, 2004.

N. Chopin. Central limit theorem for sequential Monte Carlo methods and its application to Bayesian inference. *Annals of Statistics*, 32(6):2385–2411, 2004.

T.C. Clapp and S.J. Godsill. Fixed-lag blind equalization and sequence estimation in digital communications systems using sequential importance sampling. In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing- Proceedings*, volume 5, pages 2495–2498. Citeseer, 1999.

P.K. Clark. A subordinated stochastic process model with finite variance for speculative prices. *Econometrica: Journal of the Econometric Society*, 41(1): 135–155, 1973.

A. Clauset, C.R. Shalizi, and M.E.J. Newman. Power-law distributions in empirical data. *SIAM review*, 51(4):661–703, 2009.

G. Consigli and M.A.H. Dempster. Dynamic stochastic programming for asset-liability management. *Annals of Operations Research*, 81:131–162, 1998.

R. Cont. Empirical properties of asset returns: stylized facts and statistical issues. *Quantitative Finance*, 1(2):223–236, 2001.

R. Cont, M. Potters, and J.P. Bouchaud. Scaling in stock market data: stable laws and beyond. *Science & Finance (CFM) working paper archive*, 1997.

M.K. Cowles and B.P. Carlin. Markov Chain Monte Carlo convergence diagnostics: a comparative review. *Journal of the American Statistical Association*, 91(434), 1996.

M.K. Cowles, G.O. Roberts, and J.S. Rosenthal. Possible biases induced by MCMC convergence diagnostics. *Journal of Statistical Computation and Simulation*, 64(1):87–104, 1999.

J.C. Cox, S.A. Ross, and M. Rubinstein. Option pricing: A simplified approach. *Journal of financial Economics*, 7(3):229–263, 1979.

D. Crisan and A. Doucet. A survey of convergence results on particle filtering methods for practitioners. *IEEE Transactions on signal processing*, 50(3): 736–746, 2002.

M.M. Dacorogna and R. Gencay. *An introduction to high-frequency finance*. Academic Press, 2001.

M. Davis and A. Etheridge. *Louis Bachelier's Theory of speculation*. Princeton and Oxford: Princeton University Press, 2006.

P. Del Moral, A. Doucet, and A. Jasra. Sequential monte carlo samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(3):411–436, 2006.

Z. Ding, C.W.J. Granger, and R.F. Engle. A long memory property of stock market returns and a new model. *Journal of empirical finance*, 1(1):83–106, 1993.

A. Doucet and V.B. Tadić. Parameter estimation in general state-space models using particle methods. *Annals of the institute of Statistical Mathematics*, 55 (2):409–422, 2003.

A. Doucet, S. Godsill, and C. Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and computing*, 10(3):197–208, 2000.

A. Doucet, N. De Freitas, and N. Gordon. *Sequential Monte Carlo methods in practice*. Springer Verlag, 2001.

J.C. Duan. The GARCH option pricing model. *Mathematical Finance*, 5(1): 13–32, 1995.

AD Duane et al. Hybrid monte carlo. *Physics letters B*, 195(2):216–222, 1987.

D. Duffie, J. Pan, and K. Singleton. Transform analysis and asset pricing for affine jump-diffusions. *Econometrica*, 68(6):1343–1376, 2000.

E. Eberlein, U. Keller, and K. Prause. New insights into smile, mispricing, and value at risk: The hyperbolic model. *The Journal of Business*, 71(3):371–405, 1998.

R.J. Elliott, W.C. Hunter, and B.M. Jamieson. Drift and volatility estimation in discrete time. *Journal of Economic Dynamics and Control*, 22(2):209–218, 1998.

R. Engle. GARCH 101: The use of ARCH/GARCH models in applied econometrics. *Journal of Economic Perspectives*, 15(4):157–168, 2001.

R.F. Engle. Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. *Econometrica: Journal of the Econometric Society*, 50(4):987–1007, 1982.

B. Eraker. Do stock prices and volatility jump? Reconciling evidence from spot and option prices. *The Journal of Finance*, 59(3):1367–1404, 2004.

B. Eraker, M. Johannes, and N. Polson. The impact of jumps in volatility and returns. *The Journal of Finance*, 58(3):1269–1300, 2003.

E.F. Fama. The behavior of stock-market prices. *Journal of business*, 38(1), 1965.

E.F. Fama. Efficient capital markets: A review of theory and empirical work. *Journal of finance*, 25(2):383–417, 1970.

P. Fearnhead. Markov chain Monte Carlo, sufficient statistics, and particle filters. *Journal of Computational and Graphical Statistics*, 11(4):848–862, 2002.

R.A. Fisher and L.H.C. Tippett. On the estimation of the frequency distributions of the largest or smallest member of a sample. In *Proceedings of the Cambridge Philosophical Society*, volume 24, pages 180–190, 1928.

J.M. Flegal and G.L. Jones. Output analysis for Markov chain Monte Carlo simulations. *Handbook of Markov Chain Monte Carlo. Springer-Verlag*, 2010.

A.E. Gelfand and S.K. Sahu. On Markov chain Monte Carlo acceleration. *Journal of Computational and Graphical Statistics*, 3(3):261–276, 1994.

A.E. Gelfand and A.F.M. Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American statistical association*, 85(410): 398–409, 1990.

S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

J. Geweke. Bayesian inference in econometric models using Monte Carlo integration. *Econometrica: Journal of the Econometric Society*, pages 1317–1339, 1989.

W.R. Gilks and C. Berzuini. Following a moving targetMonte Carlo inference for dynamic Bayesian models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(1):127–146, 2001.

W.R. Gilks and G.O. Roberts. *Strategies for improving MCMC (in Markov Chain Monte Carlo in Practice)*, chapter 6, pages 89–114. Chapman & Hall, 1996.

W.R. Gilks, W.R. Gilks, S. Richardson, and D.J. Spiegelhalter. *Markov chain Monte Carlo in practice*. Chapman & Hall/CRC, 1996.

W.R. Gilks, G.O. Roberts, and S.K. Sahu. Adaptive markov chain monte carlo through regeneration. *Journal of the American Statistical Association*, 93 (443):1045–1054, 1998.

S. Godsill and J. Vermaak. Variable rate particle filters for tracking applications. In *2005 IEEE/SP 13th Workshop on Statistical Signal Processing*, pages 1280–1285, 2005.

Simon Godsill and Tim Clapp. *Improvement strategies for Monte Carlo particle filters (in Sequential Monte Carlo Methods in Practice)*, chapter 7, pages 139–158. Springer, 2001.

S.J. Godsill, J. Vermaak, W. Ng, and J.F. Li. Models and algorithms for tracking of maneuvering objects using variable rate particle filters. *Proceedings of the IEEE*, 95(5):925–952, 2007.

N.J. Gordon, D.J. Salmond, and A.F.M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *IEE Proceedings*, volume 140, pages 107–113, 1993.

CWJ Granger and Z. Ding. Stylized facts on the temporal and distributional properties of daily data from speculative markets, University of California San Diego, Department of Economics. 1994.

C.W.J. Granger and Z. Ding. Varieties of long memory models. *Journal of Econometrics*, 73(1):61–77, 1996.

C.W.J. Granger, S. SPEAR, and Z. Ding. Stylized facts on the temporal and distributional properties of absolute returns: An update. In *Proceedings of the Hong Kong International Workshop on Statistics and Finance: an interface: Centre of Financial Time Series, The University of Hong Kong, 4-8 July 1999*, page 97. Imperial College Pr, 2000.

P.J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82(4):711, 1995.

M.T. Greene and B.D. Fielitz. Long-term dependence in common stock returns. *Journal of Financial Economics*, 4(3):339–349, 1977.

Dominique M. Guillaume, Michel M. Dacorogna, Rakhal R. Dave, Ulrich A. Muller, Richard B. Olsen, and Olivier V. Pictet. From the bird's eye to the microscope: A survey of new stylized facts of the intra-daily foreign exchange markets. *Finance and Stochastics*, 1:95–129, 1997.

H. Haario, M. Laine, A. Mira, and E. Saksman. DRAM: efficient adaptive MCMC. *Statistics and Computing*, 16(4):339–354, 2006.

Heikki Haario, Eero Saksman, and Johanna Tamminen. An adaptive metropolis algorithm. *Bernoulli*, 7(2)(2):223–242, April 2001.

P.R. Hansen and A. Lunde. A forecast comparison of volatility models: Does anything beat a GARCH (1, 1)? *Journal of Applied Econometrics*, 20(7): 873–889, 2005.

W.K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

H.A. Hauksson, M. Dacorogna, T. Domenig, U. Muller, and G. Samorodnitsky. Multivariate extremes, aggregation and risk estimation. *Quantitative Finance*, 1(1):79–95, 2001.

S.L. Heston. A closed-form solution for options with stochastic volatility with applications to bond and currency options. *Review of financial studies*, 6(2): 327–343, 1993.

T. Higuchi. Monte Carlo filter using the genetic algorithm operators. *Journal of Statistical Computation and Simulation*, 59(1):1–23, 1997.

B.M. Hill. A simple general approach to inference about the tail of a distribution. *The Annals of Statistics*, 3(5):1163–1174, 1975.

J. Hull and A. White. The pricing of options on assets with stochastic volatilities. *Journal of finance*, 42(2):281–300, 1987.

D.W. Jansen and C.G. De Vries. On the frequency of large stock returns: Putting booms and busts into perspective. *The Review of Economics and Statistics*, 73(1):18–24, 1991.

R.E. Kass and A.E. Raftery. Bayes factors. *Journal of the American Statistical Association*, 90(430), 1995.

M.G. Kendall and A.B. Hill. The analysis of economic time-series-part i: Prices. *Journal of the Royal Statistical Society. Series A (General)*, 116(1):11–34, 1953.

Z. Khan, T. Balch, and F. Dellaert. MCMC-based particle filtering for tracking a variable number of interacting targets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1805–1918, 2005.

S. Kim, N. Shepherd, and S. Chib. Stochastic volatility: likelihood inference and comparison with ARCH models. *Review of Economic studies*, 65(3):361–393, 1998.

S. Kirkpatrick, C.D. Gelatt Jr, and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671, 1983.

G. Kitagawa. Non-Gaussian state-space modeling of nonstationary time series. *Journal of the American statistical association*, 82(400):1032–1041, 1987.

G. Kitagawa. Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of computational and graphical statistics*, 5(1):1–25, 1996.

G. Kitagawa. A self-organizing state-space model. *Journal of the American Statistical Association*, 93(443):1203–1215, 1998.

A. Kong, J.S. Liu, and W.H. Wong. Sequential imputations and Bayesian missing data problems. *Journal of the American Statistical Association*, 89(425): 278–288, 1994.

S.C. Kramer and H.W. Sorenson. Recursive Bayesian estimation using piecewise constant approximations. *Automatica*, 24(6):789–801, 1988.

D.A. Levin, Y. Peres, and E.L. Wilmer. *Markov chains and mixing times*. American Mathematical Society, 2009.

Paul Levy. *Calcul des probabilites*. Gauthier-Villars, 1925.

H. Li, M. Wells, and L. Yu. A MCMC Analysis of Time-Changed Lévy Processes of Stock Return Dynamics. *Review of Financial Studies*, 21:2345–2378, 2008.

Jane Liu and Mike West. *Combined parameter and state estimation in simulation-based filtering (in Sequential Monte Carlo Methods in Practice)*, chapter 10, pages 197–224. Springer, 2001.

J.S. Liu. Metropolized independent sampling with comparisons to rejection sampling and importance sampling. *Statistics and Computing*, 6(2):113–119, 1996.

J.S. Liu and R. Chen. Blind Deconvolution Via Sequential Imputations. *Journal of the American Statistical Association*, 90(430), 1995.

Y. Liu, P. Cizeau, M. Meyer, C.K. Peng, and H. Eugene Stanley. Correlations in economic time series. *Physica A: Statistical and Theoretical Physics*, 245 (3-4):437–440, 1997.

A.W. Lo. Long-term memory in stock market prices. *Econometrica: Journal of the Econometric Society*, 59(5):1279–1313, 1991.

M. Longin. The asymptotic distribution of extreme stock market returns. *Journal of Business*, pages 383–408, 1996.

H. Lütkepohl and M. Krätzig. Applied time series econometrics. 2004.

T. Lux. The limiting extremal behaviour of speculative returns: an analysis of intra-daily data from the Frankfurt Stock Exchange. *Applied Financial Economics*, 11(3):299–315, 2001.

D.J.C. MacKay. *Information theory, inference, and learning algorithms*. Cambridge Univ Pr, 2003.

D.B. Madan and E. Seneta. The variance gamma (VG) model for share market returns. *Journal of Business*, 63(4):511–524, 1990.

B. Mandelbrot. The variation of certain speculative prices. *Journal of business*, 36(4), 1963.

E. Marinari and G. Parisi. Simulated tempering: a new Monte Carlo scheme. *EPL (Europhysics Letters)*, 19:451, 1992.

H. M. Markowitz. Portfolio selection. *The Journal of Finance*, 7:77–91, 1952.

K.L. Mengersen and C.P. Robert. MCMC convergence diagnostics: a reviewww. page 415, 1999.

R.C. Merton. Option pricing when underlying stock returns are discontinuous. *Journal of Financial Economics*, 3(1-2):125–144, 1976.

N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, E. Teller, et al. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087, 1953.

C. Musso, N. Oudjane, and F. LeGland. Improving regularised particle filters. *Sequential Monte Carlo methods in practice*, pages 247–271, 2001.

JF Muzy, J. Delour, and E. Bacry. Modelling fluctuations of financial time series: from cascade process to stochastic volatility model. *The European Physical Journal B-Condensed Matter and Complex Systems*, 17(3):537–548, 2000.

R.M. Neal. Markov chain Monte Carlo methods based on slicing the density function. *Technical Report 9722 Department of Statistics University of Toronto*, 1997.

R.M. Neal. Suppressing random walks in Markov chain Monte Carlo using ordered overrelaxation. *Learning in graphical models*, pages 205–225, 1998.

R.M. Neal. Slice sampling. *Annals of Statistics*, 31(3):705–741, 2003.

D.B. Nelson. ARCH models as diffusion approximations. *Journal of Econometrics*, 45(1-2):7–38, 1990.

J.A.A. Nylander, J.C. Wilgenbusch, D.L. Warren, and D.L. Swofford. AWTY(are we there yet?): a system for graphical exploration of MCMC convergence in Bayesian phylogenetics. *Bioinformatics*, 24(4):581, 2008.

A. Pagan. The econometrics of financial markets. *Journal of empirical finance*, 3(1):15–102, 1996.

J. Pan. The jump-risk premia implicit in options: evidence from an integrated time-series study. *Journal of Financial Economics*, 63(1):3–50, 2002.

S.K. Pang, J. Li, and S. Godsill. Models and algorithms for detection and tracking of coordinated groups. In *IEEE Aerospace Conference*, pages 1–17, 2008.

J. Peltonen, J. Venna, and S. Kaski. Visualizations for assessing convergence and mixing of Markov chain Monte Carlo simulations. *Computational Statistics & Data Analysis*, 53(12):4453–4470, 2009.

M.K. Pitt and N. Shephard. Filtering Via Simulation: Auxiliary Particle Filters. *Journal of the American Statistical Association*, 94(446):590–591, 1999.

V. Plerou, P. Gopikrishnan, L.A. Nunes Amaral, M. Meyer, and H.E. Stanley. Scaling of the distribution of price fluctuations of individual companies. *Physical Review E*, 60(6):6519–6529, 1999.

N.G. Polson, J.R. Stroud, and P. Müller. Practical filtering with sequential parameter learning. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(2):413–428, 2008.

G. Poyiadjis, A. Doucet, and S.S. Singh. Maximum likelihood parameter estimation in general state-space models using particle methods. *Proc of the American Stat. Assoc*, 2005.

J.G. Propp and D.B. Wilson. Exact sampling with coupled Markov chains and applications to statistical mechanics. *Random structures and Algorithms*, 9 (1-2):223–252, 1996.

G. O. Roberts and J. S. Rosenthal. Coupling and ergodicity of adaptive mcmc. 2005.

G. O. Roberts and J. S. Rosenthal. Examples of adaptive mcmc. 2008.

G.O. Roberts. *Strategies for improving MCMC (in Markov Chain Monte Carlo in Practice)*, chapter 3, pages 45–58. Chapman & Hall, 1996.

G.O. Roberts and J.S. Rosenthal. Optimal scaling for various Metropolis-Hastings algorithms. *Statistical Science*, 16(4):351–367, 2001.

G.O. Roberts and A.F.M. Smith. Simple conditions for the convergence of the Gibbs sampler and Metropolis-Hastings algorithms. *Stochastic Processes and their Applications*, 49(2):207–216, 1994.

G.O. Roberts, A. Gelman, and W.R. Gilks. Weak convergence and optimal scaling of random walk metropolis algorithms. *Annals of Applied Probability*, 7(1):110–120, 1997.

J.S. Rosenthal. Minorization Conditions and Convergence Rates for Markov Chain Monte Carlo. *Journal of the American Statistical Association*, 90(430), 1995.

A. Rossi and G.M. Gallo. Volatility estimation via hidden Markov models. *Journal of Empirical Finance*, 13(2):203–230, 2006.

T.H. Rydberg and N. Shephard. Dynamics of trade-by-trade price movements: decomposition and models. *Journal of Financial Econometrics*, 1(1):2, 2003.

R. Schmalensee and R.R. Trippi. Common stock volatility expectations implied by option premia. *Journal of Finance*, 33(1):129–147, 1978.

DJ Spiegelhalter, A. Thomas, NG Best, and WR Gilks. BUGS: Bayesian inference using Gibbs sampling, Version 0.30. *Biostatistics Unit. Cambridge: MRC*, 1994.

S.J. Taylor. Modeling stochastic volatility: A review and comparative study. *Mathematical Finance*, 4(2):183–204, 1994.

H. Tian, T. Shen, B. Hao, Y. Hu, and N. Yang. Image Restoration Based on Adaptive MCMC Particle Filter. In *Image and Signal Processing, 2009. CISP'09. 2nd International Congress on*, pages 1–5. IEEE, 2009.

L. Tierney. Markov chains for exploring posterior distributions. *the Annals of Statistics*, 22(4):1701–1728, 1994.

L. Tierney. *Strategies for improving MCMC (in Markov Chain Monte Carlo in Practice)*, chapter 4, pages 89–114. Chapman & Hall, 1996.

J. Vermaak, S.J. Godsill, and P. Perez. Monte Carlo filtering for multi-target tracking and data association. *IEEE Transactions on Aerospace and Electronic systems*, 41(1):309–332, 2005.

V. Voev and A. Lunde. Integrated covariance estimation using high-frequency data in the presence of noise. *Journal of Financial Econometrics*, 2006.

A.H. Wang and R.L. Klein. Implementation of non-linear estimators using monospline. In *1976 IEEE Conference on Decision and Control including the 15th Symposium on Adaptive Processes*, volume 15, 1976.

R. Weron. Levy-stable distributions revisited: tail index > 2 does not exclude the Levy-stable regime. *Econometrics*, 2003.

M. West. Approximating posterior distributions by mixtures. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 409–422, 1993.

L. Zhang, P.A. Mykland, and Y. Ait-Sahalia. A tale of two time scales. *Journal of the American Statistical Association*, 100(472):1394–1411, 2005.