



Recursive Bayesian Methods for Sequential Parameter-State Es

by Yinan Huang

This thesis/dissertation document has been electronically approved by the following individuals:

Wells, Martin Timothy (Chairperson)

Nussbaum, Michael (Minor Member)

Jarrow, Robert A. (Minor Member)

RECURSIVE BAYESIAN METHODS FOR SEQUENTIAL PARAMETER-STATE ESTIMATION

A Dissertation

Presented to the Faculty of the Graduate School
of Cornell University

in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy

by

Yinan Huang

August 2010

© 2010 Yinan Huang
ALL RIGHTS RESERVED

RECURSIVE BAYESIAN METHODS FOR SEQUENTIAL PARAMETER-STATE ESTIMATION

Yinan Huang, Ph.D.

Cornell University 2010

A central theme in applied and computational statistics is the accurate and efficient methods of inference. The Bayesian paradigm performs inference based on the posterior distribution of unknown quantities. Throughout decades, there has been an enormous literature on computational Bayesian methods. Practical implementations, while successful to different degrees, usually impose certain restrictions on the specific model structure. As more applications rely on complex model dynamics, more challenges remain to tackle the curse of high dimensionality and the analytical intractability of many non-Gaussian distributions.

This thesis builds on existing research in the field of sequential Bayesian estimation for a general class of state-space models. We establish recursive Bayesian simulation algorithms to estimate parameters and states for a variety of diffusion and jump stochastic models. Our main work and contribution are two-fold.

First, we build a particle filter framework for Levy-type state-space models. Particle filters are efficient numerical simulation techniques ideally suitable for highly nonlinear models, with a significant computational advantage over the standard Markov Chain Monte Carlo. Our particle filters can effectively estimate parameters and state variables for non-Gaussian dynamics. We perform empirical testing on financial time series, and find that certain Levy-type small jump processes can be a substitute of the usual Brownian motion-based random walk models. In addition, we propose a general Variational Bayes Particle Filter framework.

It is applicable to a wider class of models with a large number of dimensions.

Secondly, we build a Variational Bayes estimator for Hidden Markov Models with observational jumps. This is a typical setup for numerous biostatistical data analysis, where huge amounts of streaming data need to be sequentially filtered for potential evidence of the existence of quantitative traits or genetic features. Our algorithm works to identify and classify different responses. The hidden Markov estimator is robust and highly adaptable.

In addition, this thesis also includes a self-contained chapter on the technique of Markovian projection. It reduces a complicated multi-dimensional dynamics to a one-dimensional simple Markovian process with identical marginal distributions, therefore keeping certain path-independent expectation values invariant. The projection has certain implications in the pricing of European-style options in financial mathematics. We provide a theorem generalizing existing results to the general Levy jump models, and discuss calibration issues.

BIOGRAPHICAL SKETCH

Yinan Huang was born in Hefei, Anhui province in central China. He grew up in the capital city of Beijing. Yinan did his undergraduate studies in the Department of Electronic Engineering at Tsinghua university. After earning a Bachelor of Engineering degree, he came to Cornell university to pursue a Ph.D. degree in Operations Research. He has a great interest in the field of applied probability and statistics, and its numerous applications in financial mathematics. Since 2007, he has been doing research under the supervision of Professor Martin Wells. Upon graduation in the summer of 2010, Yinan will join Morgan Stanley as a desk strategist, working in the Securitized Products group.

to my parents and my wife, for their love and support

ACKNOWLEDGEMENTS

This thesis is the result of my research work at Cornell university during the five years 2005 to 2010. At this special moment, I would like to express my sincere gratitude and thanks to the many faculty, friends and family members.

First and foremost, I want to thank my thesis advisor Professor Martin Wells. I feel so fortunate and grateful to have had the wonderful opportunity of working with Marty throughout my stay here in Cornell. He is such a brilliant scholar and an amazing mentor. The discussions with him are always illuminating to me, and in all ups and downs of my exploration into statistics, in particular during the moments of frustration and anxiety, Marty is always the person so kind and encouraging to me, which is the most precious support and guidance. My Ph.D. research would not have been possible without him. He sets a role model for me not only in academic research, but also in my life.

I want to thank Professor Robert Jarrow for bringing me into the field of mathematical finance. His deep wisdom about the markets, glowing insights and ideas give me eye-opening impressions, and have helped shape my mind towards a career in quantitative finance. I also want to thank Professor Michael Nussbaum for serving on my committee. His inspiring, well-designed course in mathematical statistics motivates me to pursue my research in statistical methods.

My thanks also go to many friends I have met during my years in Ithaca, to name a few, Haizhi Lin, Leon Chen, Tuohua Wu, Fan Zhu, Jie Chen, Chao Ding. In particular, the graduate students from my department form a very cheering atmosphere together, making life much more enjoyable and rewarding.

Last but not least, I will have my special thanks to my family, my loving wife Sherry and my parents. I dedicate this thesis to you all.

TABLE OF CONTENTS

| | |
|---|-----------|
| Biographical Sketch | iii |
| Dedication | iv |
| Acknowledgements | v |
| Table of Contents | vi |
| List of Figures | viii |
| 1 Introduction | 1 |
| 1.1 Inference: the Bayesian perspective and challenges | 1 |
| 1.2 Distributional approximation methods: Analytics vs Simulation | 2 |
| 1.3 Simulation methods: Offline vs Online | 5 |
| 1.4 Outline of the dissertation | 6 |
| 2 Variational Bayes estimation for hidden Markov models | 8 |
| 2.1 Introduction | 8 |
| 2.2 The Variational Bayes framework | 9 |
| 2.2.1 The VB Method | 9 |
| 2.2.2 Online VB Parameter Estimation | 12 |
| 2.2.3 Online VB Parameter-State Joint Estimation | 16 |
| 2.3 Marginalized Variational Bayes | 18 |
| 2.3.1 Connections to the EM algorithm | 18 |
| 2.3.2 Marginalized Variational Bayes Estimation | 24 |
| 2.4 Hidden Markov Models (HMM) | 27 |
| 2.4.1 Inference for HMMs | 29 |
| 2.5 Variational Bayes Hidden Markov Model | 30 |
| 2.5.1 VB-HMM model formulation | 30 |
| 2.5.2 Model Specifics | 32 |
| 2.6 Case study: jump detection in gene sequences | 34 |
| 2.6.1 Overview | 34 |
| 2.6.2 The dataset | 35 |
| 2.6.3 Estimation Results and Discussion | 36 |
| 3 Particle filtering methods for state-parameter estimation | 39 |
| 3.1 Introduction | 39 |
| 3.2 The particle filtering framework | 41 |
| 3.2.1 Basic concepts in random sampling | 41 |
| 3.2.2 The Particle Filter(PF) | 45 |
| 3.2.3 Extensions of standard particle filtering | 54 |
| 3.3 The variational Bayes particle filter | 59 |
| 3.3.1 VB-PF formulation | 61 |
| 3.4 Joint Parameter-State Estimation for state-space models | 63 |
| 3.4.1 Discrete-time State Space models | 63 |
| 3.4.2 Continuous-time Diffusion models | 64 |

| | | |
|----------|---|------------|
| 3.5 | Simulation Results | 66 |
| 3.5.1 | The classic Kalman filter | 66 |
| 3.5.2 | Particle filter simulations | 68 |
| 3.5.3 | VB-PF simulation | 74 |
| 4 | Dimensional Reduction: Markovian Projection for Levy Jump models | 84 |
| 4.1 | Introduction | 84 |
| 4.2 | The general Markovian Projection framework | 85 |
| 4.3 | Markovian Projection for Poisson Jump processes | 88 |
| 4.4 | Markovian Projection for Levy processes | 90 |
| 4.5 | Calibration Procedures | 92 |
| 4.5.1 | Calibration to local speed | 93 |
| 4.5.2 | From local speed to stochastic speed | 96 |
| A | Proof of Theorem 4.1 | 98 |
| | Bibliography | 103 |

LIST OF FIGURES

| | | |
|------|--|----|
| 2.1 | A Simplified HMM schematic | 28 |
| 2.2 | Original p value sequence of the dog weight dataset. There are 42,396 data points, within the [0,1] range. Jump identification on the raw data is nearly impossible. | 37 |
| 2.3 | Detected jumps with VB-HMM estimation. 8 most significant jumps are detected. The vertical scale corresponds to the choice of a particular HMM, with state 1 being normal and state 2 possible genetic mutational. | 37 |
| 2.4 | Original logarithm p values. The vertical scale corresponds to the logarithms of the raw data. The jumps are clustered around 8 significant regions. Vb-HMM essentially switches between 2 states sequentially | 38 |
| 3.1 | 1-dim state filtering with Kalman Filter, all known parameters | 67 |
| 3.2 | Fast convergence for posterior variance with Kalman Filter, all known parameters | 68 |
| 3.3 | 1-dim state filtering with USPF, exact parameter | 69 |
| 3.4 | 1-dim state filtering with USPF, unknown parameter μ | 69 |
| 3.5 | 2-dim state filtering with USPF, exact parameter | 70 |
| 3.6 | 2-dim state filtering with USPF, unknown parameter μ | 70 |
| 3.7 | 1-dim state filtering with MRBPF, unknown parameter μ | 72 |
| 3.8 | 1-dim state filtering with MRBPF, unknown parameter κ | 73 |
| 3.9 | 1-dim state filtering with MRBPF, exact parameter | 73 |
| 3.10 | Kalman Filter with single unknown parameter: posterior distribution of b | 77 |
| 3.11 | Kalman Filter with single unknown parameter: sequential estimates of b | 78 |
| 3.12 | Kalman Filter with multiple unknown parameter: posterior distribution of a | 81 |
| 3.13 | Kalman Filter with multiple unknown parameter: posterior estimates of σ_{\max} | 82 |
| 3.14 | Kalman Filter with multiple unknown parameter: sequential estimate of a | 82 |
| 3.15 | Kalman Filter with multiple unknown parameter: sequential estimate of σ_{\max} | 83 |

CHAPTER 1

INTRODUCTION

1.1 Inference: the Bayesian perspective and challenges

In everyday life, we are faced with the perennial question of data and models. Given observable data phenomenon, which is usually some partial and imperfect representation of the object of interest, it is of paramount importance to devise accurate and efficient methods to obtain knowledge about the driving mechanics. In statistics terminology, we use inference to denote this generic operation of learning. Typically, the underlying dynamics is random, governed by unknown, and even unknowable forces, which are further polluted by noise through observation. The resulting data is therefore invariably stochastic, and probabilities are used to define and manipulate quantities of data.

The Bayesian philosophy builds on the concept of probabilistic beliefs, and the calculus of conditional probabilities to represent and update our knowledge about the data and models. Given an unknown parameter θ , an observation D , we have the *a priori* belief $p(\theta)$, our *a priori* understanding and estimate of the system. Our beliefs about the data are completely characterized by the parametric probabilistic observation model $p(D|\theta)$. By constructing the posterior, or, a posteriori distribution, $p(\theta|D)$, we have incorporated everything we can possibly know from the data D . The Bayesian rule stipulates:

$$p(\theta|D) \propto p(\theta)p(D|\theta). \tag{1.1}$$

Bayesian theory has been successfully applied and has become a central topic in

statistical inference. In many scenarios, the Bayesian solution, represented through the simple posterior $p(\theta|D)$, is viewed as optimal. It also addresses to a large extent the problem of model selection and overfitting that often appears in classic frequentist statistics. In theory, the Bayesian posterior has captured all information in the data relating to the parameter of interest, and no further improvements can be made. In practice, however, the computation of this posterior stands out as a real numerical challenge. For almost all practical applications, the integrals involved are analytically intractable, and are further complicated by the presence of high dimensionality, latent states and non-linearity.

1.2 Distributional approximation methods: Analytics vs Simulation

As described above, it is the case that a full Bayesian analysis, namely the calculation of the exact joint posterior, is often intractable. We are faced with the problem of approximating complex integrals, and this reduces to a problem of distributional approximation. We can replace the true posterior $p(\theta|D)$ with a proxy $\hat{p}(\theta|D)$, and work with \hat{p} instead. There are two fundamentally different but related approaches.

Using a deterministic approximation, $\hat{p}(\theta|D)$ is obtained by application of a deterministic rule to $p(\theta|D)$. In other words, $\hat{p}(\theta|D)$ is uniquely functionally determined by $p(\theta|D)$. Clearly, if we choose $\hat{p} = p$, then we get the best possible approximate as an identity, but this is intractable. Rather, there should be some subspace of available and “easy” functions to choose from. Mathematically, we

can formulate it as an abstract optimization problem:

$$\widehat{p}(\theta|D) = \arg \min_{f \in \mathbb{F}} D(\widehat{p}(\theta|D)||f). \quad (1.2)$$

where \mathbb{F} represents a subspace of distribution functions such that all functions within \mathbb{F} are considered tractable, and $D(\cdot||\cdot)$ is a distance measure defined on \mathbb{F} . The optimal function that minimizes this distance metric is the “best” approximate proxy.

There are a high level of flexibility in such an optimization. Both the metric D and the subspace \mathbb{F} have different options. In Bayesian statistics, it is customary to use the Kullback-Leibler divergence $KL(\cdot||\cdot)$. The following are deterministic methods that all fall into this category: certain equivalence, which includes maximum likelihood and maximum *a posteriori* point inference as special cases; the Laplace transform; and fixed-form approximation.

Our focus throughout this paper is Variational Bayes (VB) approximation, which is a deterministic, free-form approximation method. As will become apparent later, Variational Bayes imposes conditional independence among the component dimensions, thereby factorizing a complex, multi-dimensional distribution into the product of simpler, lower-dimensional marginals. \mathbb{F} in this case is the subspace of conditionally posterior independent distributions. Through VB, a large variety of models can be well approximated and computed efficiently.

On the other hand, stochastic distributional approximation doesn’t aim to derive an invariant fixed functional form \widehat{p} as the proxy. On the contrary, it produces a random sample of realizations of $p(\theta|D)$, and use the empirical distribution of that sample as approximation. This is in essence a nonparametric method. Each time the sample is different, while the main task remains to numerically generate large numbers of random numbers. By the law of large numbers, the closeness of

\hat{p} to the true posterior is guaranteed. Such is the foundation of Bayesian Monte Carlo methods.

For notational completeness, we have, for instance, a set of independent, identically distributed (i.i.d.) samples $\theta^{(i)}$:

$$\theta^{(i)} \sim p(\theta|D) \tag{1.3}$$

$$\theta = \theta^{(1)}, \dots, \theta^{(n)}. \tag{1.4}$$

The approximating distribution is:

$$\hat{p}(\theta|D) = \frac{1}{n} \sum_{i=1}^n \delta(\theta - \theta^{(i)}). \tag{1.5}$$

For more refined extension, each sample can be added using a kernel density, resulting in a smoothed \hat{p} .

There has been a huge and accumulating literature on Monte Carlo methods in Bayesian statistics. This simulation-based approach bypasses the difficult part of analytics and high-dimensional integrals through a unified, standard framework. Much of the existing work is concentrated on the reduction of sample variances, with techniques such as importance sampling, stratified sampling, control variates and hierarchical sampling. Another stream of more theoretical research focuses on the convergence rate of Monte Carlo methods, as these numerical programs, while correct, can be notoriously slow to execute. One disadvantage occurs when there are multiple random factors to simulate, and the computational burden as a bottleneck will easily overwhelm.

Our research in this thesis deals with particle filters (PF), which are a class of numerically-efficient Monte Carlo methods for Bayesian estimation. More specifically, a particle filter maintains and updates an ensemble of samples to represent

the time evolution of posterior distributions. The PF framework is ideally suited to time series data, as the computational advantage becomes more pronounced.

1.3 Simulation methods: Offline vs Online

Within the general category of simulation methods, a clear distinction is made between offline and online estimation algorithms. Offline, or parallel methods, assume that all data D have been collected before performing any inference. This batch-typed mode completes one giant Bayesian update step in one shot. It is based on the ability to draw samples from proposal densities that mimic the full posterior. The well-known Markov Chain Monte Carlo methodology belongs to this class.

To be precise, we generate a chain of samples, starting from x_1 , such that the next sample is drawn randomly based on the previous sample. In this way, correlated samples are obtained; they are however, close to being i.i.d with respect to the true distribution. The power of MCMC lies in the versatility of proposal distributions and guaranteed convergence to the target.

In recent years, the alternative class of simulation, sequential Monte Carlo has attracted more and more attention from different research areas, with successful applications to such diverse fields as signal processing, machine learning, biology and wireless communications. The main attraction of sequential, or online, methods is that they allow on-line estimates by cascading successive Bayesian estimates in a recursive manner, therefore significantly reduces execution speed. To be precise, sequential Monte Carlo tends to lose a small amount of accuracy as a trade-off for computational simplicity, when compared with MCMC.

Particle filters are sequential Monte Carlo methods applied to Bayesian filtering. In real-world applications, the underlying model typically has multiple parameters and unobservable state variables, leading to a highly nonlinear environment. The analytics is daunting to the point that even simple approximations seem impractical, and numerical methods are the preferred approach. Within them, particle filters offer a promising compromise. PF-based algorithms, at the same time standardized and seemingly routine, entail lots of improvising and tailor-making to adapt to specific models and problems.

1.4 Outline of the dissertation

Throughout this dissertation, we revolve around the central theme: to develop tools for computational Bayesian estimations. A key characteristic is the reconciliation of high dimensionality, nonlinearity and computational complexity within a model. To address this challenge, we have worked on Variational Bayes and Particle filters, and combined the two approaches together to unify merits. The detailed organization of the thesis is as follows.

Chapter 2 builds on the Variational Bayesian framework. We establish a general framework for joint estimation of state and parameters, including persistent latent states such as volatility as well as stand-alone states such as isolated jumps. We develop a marginalized Variational Bayes algorithm to exploit analytical tractability along certain dimensions. For a practical application, the variational Bayes algorithm is applied to hidden Markov models in biostatistical data analysis. Our estimation shows that the VB-HMM approach can single out large atypical gene expression differences in a long gene data sequence.

Chapter 3 deals with the particle filtering framework. We perform extensive simulations on the various types of particle filters, including extensions to Rao-Blackwellized and unscented transformations. We propose and implement a Variational Bayes Particle Filter (VB-PF) that incorporates both frameworks for comprehensive filtering. Simulation results indicate that particle filters are a viable approach for the joint identification of state and parameters.

Chapter 4 includes some theoretical work in dimension reduction. It introduces the technique of Markovian projection to reduce a multi-dimensional process to a single-dimensional, Markovian process while maintaining marginal distributions. Such a transformation will keep all path-independent quantities unchanged, and hence have implications for certain type of European-style derivatives pricing. We extend the known theorem in Brownian motion diffusions to jump-diffusions and further to general Levy-type jump processes, and obtain parallel results.

CHAPTER 2

VARIATIONAL BAYES ESTIMATION FOR HIDDEN MARKOV MODELS

2.1 Introduction

The primary problem in Bayesian inference is that of the computation of posterior distribution. It is a difficult quantity to compute because of the high dimensions involved. In this chapter, we develop efficient methods based on the concept and techniques of Variational Bayes (VB) approximation. Variational Bayes imposes conditional posterior independence between subsets of parameters, and find the optimal approximate distribution by a particular measure of divergence. Functional optimization yields a known functional form for each marginal distribution, which are called VB-marginals. The shaping parameters associated with each of these VB-marginals are expressed through particular moments of others. In this way, the approximation is intertwined in the sense that the determination of individual distributions depend on the determination of other distributions. Such mutual interactions of VB-marginals via their respective moments presents an obstacle to evaluation. Usually, analytical solution is available only under very special and simple models, and in general, a generic iterative algorithm for the evaluation of VB-moments and shaping parameters must be employed. Through an iterative VB algorithm, all parameters are estimated and marginal distributions can be approximated.

In this chapter, we explore the various aspects of Variational Bayes. We derive a complete framework of online Variational Bayes for the joint identification of parameter and states. This procedure allows for fast estimation in many state-

space models with latent states and possibly jumps. Along the lines of marginalized particle filtering, we propose a marginalized Variational Bayes to further exploit the analytical structure within a complex joint distribution. Only the states are considered to be conditionally independent; the parameters, however, can have closed-form posterior, making further independence unnecessary.

As an illustration of the power of Variational Bayes, we investigate a common type of state-space models, the hidden Markov model (HMM). HMM has a rich structure with flexible inference methods. We use Variational Bayes for HMM state estimation. In particular, we apply this model to the genetic data sequences arising in biostatistics. The algorithm successfully picks up the most distinctive portions of “jump”-style p-values, indicating possible presence of genetic significance.

The variational Bayes method has gained attention and popularity over the past decade. There has been a growing literature on the topic, mostly focusing on particular applications. [1] and [2] are definite references for the introduction of generic Variational Bayesian methods. [6] considers the standard mixture model. More recently, [4] works with incomplete data and graphical models. [50] proposes an parameter expanded version of Variational Bayes to speed up the algorithm.

2.2 The Variational Bayes framework

2.2.1 The VB Method

The idea of variational approximation has been around in different research disciplines for well over a decade. In the machine learning community, [40] and

[41] are early pioneering works of the use of VB-approximation. The focus is on learning complex systems such as neural networks, [22], and the name is under mean Field Theory, [53]. Meanwhile, statistical physics has been concerned with a similar problem of high-dimensional distribution approximation, see [47] and [46] for a comprehensive survey. The exact probability model, $f(\theta)$, is replaced with an approximation $\tilde{f}(\theta)$. The optimal such approximate distribution can be chosen using the variational method by seeking a free-form solution within the approximating class that minimizes some measure of distance from f to \tilde{f} . The Kullback-Leibler divergence is the preferred measure due to its properties and connection with the physical entropy, which can be interpreted as a relative entropy.

With the introduction of graphical models, [1] and [16], there is a strong communication between the machine learning community and the statistical physics community, and Variational Bayes methods grow out of this interplay. The well-established Expectation-Maximization (EM) algorithm, [11] was re-derived in [45] using Kullback-Leibler approximation. The EM algorithm has long been used to find Maximum Likelihood Estimation (MLE) solutions in high-dimensional problems where the exact likelihood is intractable. Such a connection justifies the use of Variational Bayes to approximate distributions and compute point estimates.

Given an unknown multivariate parameter vector $\theta = (\theta^1, \theta^2)$ and observation data y , we aim to find the posterior distribution $f(\theta|y)$. The VB theory provides a practical framework to approximate this distribution by enforcing posterior conditional independence:

$$f(\theta|y) \approx \tilde{f}(\theta|y) = \tilde{f}(\theta^1|y)\tilde{f}(\theta^2|y).$$

The Variational Bayes method finds the best approximating distribution \tilde{f} in the sense of minimizing the Kullback-Leibler distance. Here, we have an iterative

algorithm to compute the marginal distributions.

1. Get the joint distribution $f(\theta, y) = f(\theta)f(y|\theta)$.
2. Partition the parameter space into marginals, here simply $\theta = (\theta^1, \theta^2)$ as the default decomposition. At this step, we need to check for separable-in-parameters. That is, $\log f(\theta, y)$ must be separable as a product of individual likelihoods for each parameter.

$$\log f(\theta^1, \theta^2, y) = g(\theta^1, y) \cdot h(\theta^2, y). \quad (2.1)$$

3. Write down the VB-marginals \tilde{f} . The marginals are intertwined through the corresponding VB-moments:

$$\begin{aligned} \tilde{f}(\theta^1|y) &\propto \exp E_{\tilde{f}(\theta^2|y)}(\log f(\theta^1, \theta^2, y)) \\ &\propto \exp(g(\theta^1, y) \widehat{h(\theta^2, y)}) \\ \tilde{f}(\theta^2|y) &\propto \exp E_{\tilde{f}(\theta^1|y)}(\log f(\theta^1, \theta^2, y)) \\ &\propto \exp(\widehat{g(\theta^1, y)} h(\theta^2, y)). \end{aligned}$$

4. Identify a standard distributional form. One key to a successful VB approximation is to use simple, standard distributions for the VB-marginals. Clearly, this is a strong requirement. Once the parametric family to which the VB-marginals belong are identified, the shaping parameters can be expressed using the VB-moments. On the other hand, VB-moments come from the corresponding VB-marginals, all expressed using the shaping parameters. This is essentially a mutual dependence that can be calculated using an iterative algorithm.
5. Formulate the necessary VB-moments.

6. Establish a system of equations. This (nonlinear) system solves for the VB-moments and the shaping parameters.
7. Solve the equations through Iterated VB (IVB) algorithm. An IVB cycle is similar to an iteration in the EM algorithm used for Maximum Likelihood estimation.

2.2.2 Online VB Parameter Estimation

In the previous chapter, we established the generic framework of the Variational Bayes method for posterior approximation. In offline estimation, there are three possible scenarios.

1. The posterior distribution $f(\theta|y)$ has an analytical form. This is the ideal case, which lays foundation of all Bayesian calculations. Here the central concept is Bayesian conjugacy. Equivalently, we say that the observation link $f(y|\theta)$ is a conjugate channel, meaning that it admits a conjugate prior-posterior pair. Naturally, the prior $f(\theta)$ is chosen as that particular conjugate prior that fits the channel, and then the posterior $f(\theta|y)$ can be obtained in closed-form by simply updating the corresponding parameters. We only need to maintain a small set of sufficient statistics. The Exponential family provides a rich collection of conjugate channels. Throughout the dissertation, we can, in some sense, use the phrases “conjugate”, “sufficient statistics”, “exponential family distributions” almost interchangeably.
2. The posterior distribution $f(y|\theta)$ is not a conjugate channel, but the joint distribution $f(\theta, y)$ satisfies a weaker condition of “separable-in-parameter”, as in (2.1). Here we cannot expect to find a closed-form expression for the

posterior. The VB method fits perfectly under this scenario. It uses IVB cycles as described earlier to iteratively calculate the VB-moments and the parameters for VB-marginals.

3. The joint distribution $f(\theta, y)$ is not separable-in-parameters. Even the VB semi-analytics doesn't apply. We have to resort to more generic simulation methods. A popular approach is Monte Carlo simulation to sample from the prior, calculate the likelihood, and construct the empirical posterior by a large random sample.

We now turn to the associated problem of Sequential Bayesian estimation. Given the time $t - 1$ estimate $f(\theta|y^{t-1})$, we calculate the time t estimate $f(\theta|y^t)$. The recursive algorithm is based on the fundamental Bayesian relation:

$$f(\theta|y^t) \propto f(\theta|y^{t-1})f(y_t|\theta, y^{t-1}). \quad (2.2)$$

As we will see, there are again parallel cases to consider, depending on the level of analytical tractability for the channel $f(y_t|\theta, y^{t-1})$.

1. $f(y_t|\theta, y^{t-1})$ is a conjugate channel. Clearly, this is the perfect case where closed-form solutions for the posterior $f(\theta|y^t)$ exists, and can be expressed using low-dimensional sufficient statistics. The classic Kalman filter belongs to this category. It is essentially based on linear channels with a conjugate normal prior-posterior pair. The efficiency of Kalman filter lies in the updating of only the first two moments. More generally, such a conjugate channel typically belongs to the broad Dynamic Exponential Family. The sequential Bayesian estimation is very straightforward. It involves simple recursive updating equations for the sufficient statistics.

Sometimes, even if we can get the analytical expression of the joint posterior

$f(\theta|y^t)$, where as always $\theta = (\theta^1, \theta^2)$, the marginalization of the posterior is still not tractable. In this case, we need a VB step to obtain the approximates $\tilde{f}(\theta^1|y^t)$ and $\tilde{f}(\theta^2|y^t)$.

2. $f(y_t|\theta, y^{t-1})$ is not a conjugate channel, but satisfies “separable-in-parameter” condition. Moreover, the individual approximate channels $\tilde{f}(y_t|\theta^i, y^{t-1})$ are all conjugate. This is the perfect setting for the VB methodology. We elaborate this scenario in Section 4.2.
3. $f(y_t|\theta, y^{t-1})$ is not even “separable-in-parameter”. A trick is to introduce latent states l_t , and make the enlarged likelihood $f(y_t, l_t|\theta, y^{t-1})$ “separable-in-parameter”. We are therefore back to the previous case, but still need additional requirements in order to perform the VB analytics. Section 4.3 tackles this special scenario.
4. For all other cases, we need simulation-based methods to approximate the joint posterior. Particle filters are a powerful family of algorithms for sequential Bayesian estimation in this aspect. For the subsequent chapter, we will combine PF with VB to establish a general framework that combines the merits of both.

The VB derivation

Here we provide a derivation of the VB method for online estimation, given that the observation channel can be split into individual channels.

We aim to approximate $f(\theta^1, \theta^2|y^t) \propto \tilde{f}(\theta^1|y^t)\tilde{f}(\theta^2|y^t)$. According to the fundamental VB theorem, we use the equivalent joint $f(\theta^1, \theta^2, y^t)$, assuming it satisfies

the usual “separable-in-parameter”,

$$\begin{aligned}
\tilde{f}(\theta^1|y^t) &\propto \exp E_{\tilde{f}(\theta^2|y^t)}(\log f(\theta^1, \theta^2, y^t)) \\
&= \exp E_{\tilde{f}(\theta^2|y^t)}(\log f(\theta^1, \theta^2, y_t|y^{t-1})) \\
&= \exp E_{\tilde{f}(\theta^2|y^t)}(\log f(\theta^1, \theta^2|y^{t-1}) + \log f(y_t|\theta, y^{t-1})).
\end{aligned}$$

For offline approximation, this is end of story, since the prior $f(\theta|y^{t-1})$ is assumed known (there is no time dimension in offline estimation). Here, however, we use the previous VB-approximation, that is, $f(\theta^1, \theta^2|y^{t-1}) \propto \tilde{f}(\theta^1|y^{t-1})\tilde{f}(\theta^2|y^{t-1})$. Plugging back into the expression above, we have:

$$\begin{aligned}
\tilde{f}(\theta^1|y^t) &= \exp E_{\tilde{f}(\theta^2|y^t)}(\log f(y_t|\theta, y^{t-1}) + \log \tilde{f}(\theta^1|y^{t-1})) + \log \tilde{f}(\theta^2|y^{t-1}) \\
&= \exp E_{\tilde{f}(\theta^2|y^t)}(\log f(y_t|\theta, y^{t-1}) + \log \tilde{f}(\theta^1|y^{t-1})) \\
&= \exp E_{\tilde{f}(\theta^2|y^t)}(\log f(y_t|\theta, y^{t-1})) \cdot \tilde{f}(\theta^1|y^{t-1}).
\end{aligned}$$

Where $\tilde{f}(\theta^2|y^t)$ is constant term in the expectation. We define the individual subchannel for the first parameter component $\tilde{f}(y_t|\theta^1, y^{t-1})$ as:

$$\tilde{f}(y_t|\theta^1, y^{t-1}) = \exp E_{\tilde{f}(\theta^2|y^t)}(\log f(y_t|\theta, y^{t-1})). \quad (2.3)$$

The VB equations can be written as:

$$\tilde{f}(\theta^1|y^t) = \tilde{f}(y_t|\theta^1, y^{t-1}) \cdot \tilde{f}(\theta^1|y^{t-1}). \quad (2.4)$$

Similarly for the other subchannel:

$$\tilde{f}(\theta^2|y^t) = \tilde{f}(y_t|\theta^2, y^{t-1}) \cdot \tilde{f}(\theta^2|y^{t-1}). \quad (2.5)$$

In general, for a parameter vector θ of size K , we can make a partition according to the index space $I_k = \{1, \dots, K\}$. Each subchannel with index subset $I \subset I_k$

can be computed by holding all other parameter components fixed, and compute:

$$\tilde{f}(\theta^I|y^t) = \tilde{f}(y_t|\theta^I, y^{t-1}) \cdot \tilde{f}(\theta^I|y^{t-1}). \quad (2.6)$$

with

$$\tilde{f}(y_t|\theta^I, y^{t-1}) = \exp E_{\tilde{f}(\theta^{I_k-I}|y^t)}(\log f(y_t|\theta, y^{t-1})). \quad (2.7)$$

2.2.3 Online VB Parameter-State Joint Estimation

In a more realistic setting, sequential estimation involves unknown persistent states as well as parameters. Persistent states are also called time-varying parameters. Such a joint identification of parameters and states are a very challenging problem, as it is almost invariably high-dimensional, and the model specification becomes all the more important. In this subsection, we derive a generic framework using online VB method, and identify the necessary conditions to carry out the semi-analytics.

The VB derivation

We consider a generic case. In addition to time-invariant parameters θ and persistent states x_t , there are other unobservable, “latent” states l_t . They can be unknown jump sizes or hidden indices for categorical distributions. They can come out of an artificial construct to make the overall observation model “separable-in-parameter”, similar to the auxiliary slack variables in linear programming to convert the model into standard form. Either way, the introduction of l_t poses still more challenges, and a recursive algorithm must effectively filter out its influence, while making a sequential estimate of (θ, x_t) .

The prior and observation are given by transition distributions $f(\theta)$, $f(l_t|\theta)$, $f(x_t|x_{t-1}, \theta)$ and $f(y_t|x_t, l_t, \theta)$.

The VB-based online method will iteratively solve for $f(\theta, x_t|y^t)$.

Before working out the expressions, a choice needs to be made for the proper partitioning of the joint space. The distribution $f(\theta, x_t, x_{t-1}, l_t|y^t)$ is high-dimensional. Here for brevity, we assume all parameters can be grouped together. Moreover, we need to keep l_t separate, as there is actually no need to propagate $\tilde{f}(l_t|y^t)$ forward in time. Also, due to its persistent nature, x_t and x_{t-1} should be treated as a whole.

Therefore, we enforce conditional posterior independence among θ , l_t and (x_t, x_{t-1}) .

The VB decomposition becomes:

$$f(\theta, x_t, x_{t-1}, l_t|y^t) = \tilde{f}(\theta|y^t)\tilde{f}(l_t|y^t)\tilde{f}(x_t, x_{t-1}|y^t). \quad (2.8)$$

Assuming that the joint distribution $f(\theta, x_t, x_{t-1}, l_t|y^t)$ satisfies the usual separability condition (but not just “separable-in-parameter”, but separable along the state dimensions), we can write the term for θ :

$$\begin{aligned} \tilde{f}(\theta|y^t) &= \exp E_{x_t, x_{t-1}, l_t} \log f(\theta, x_t, x_{t-1}, l_t|y^t) \\ &= \exp E_{x_t, x_{t-1}, l_t} \log f(\theta, x_t, x_{t-1}, l_t, y_t|y^{t-1}) \\ &= \exp E_{x_t, x_{t-1}, l_t} \log(f(\theta, x_{t-1}|y^{t-1}) \cdot f(l_t|\theta) \cdot f(x_t|x_{t-1}, \theta) \cdot f(y_t|x_t, \theta, l_t)). \end{aligned}$$

Expanding $f(\theta, x_{t-1}|y^{t-1})$ using the time $t - 1$ VB marginals, $f(\theta, x_{t-1}|y^{t-1}) =$

$\tilde{f}(\theta|y^{t-1}) \cdot \tilde{f}(x_{t-1}|y^{t-1})$, we have

$$\begin{aligned} \tilde{f}(\theta|y^t) &= \exp E_{x_t, x_{t-1}, l_t}(\log \tilde{f}(\theta|y^{t-1}) + \log \tilde{f}(x_{t-1}|y^{t-1}) + \dots \\ &\quad \log f(l_t|\theta) + \log f(x_t|x_{t-1}, \theta) + \log f(y_t|x_t, \theta, l_t)) \\ &= \tilde{f}(\theta|y^{t-1}) \exp E_{x_t, x_{t-1}, l_t} \\ &\quad (\log f(l_t|\theta) + \log f(x_t|x_{t-1}, \theta) + \log f(y_t|x_t, \theta, l_t)). \end{aligned}$$

Similarly for l_t and (x_t, x_{t-1}) :

$$\begin{aligned} \tilde{f}(l_t|y^t) &= \exp E_{x_t, x_{t-1}, \theta}(\log f(l_t|\theta) + \log f(y_t|x_t, \theta, l_t)) \\ \tilde{f}(x_t, x_{t-1}|y^t) &= \tilde{f}(x_{t-1}|y^{t-1}) \exp E_{l_t, \theta}(\log f(x_t|x_{t-1}, \theta) \\ &\quad + \log f(y_t|x_t, \theta, l_t)). \end{aligned}$$

In order to use the VB algorithm, we have identified 3 auxiliary subchannels. The overall schematic is in figure below. We need conjugacy conditions for the subchannels. Also, some of the marginals from the joint distribution needs separability conditions.

2.3 Marginalized Variational Bayes

2.3.1 Connections to the EM algorithm

Frequentist Estimation: the EM algorithm

One of the central themes in applied statistics is parameter estimation. Given observed data Y , the task is to estimate unknown parameter θ . Maximum likeli-

hood Estimation (MLE) provides the underpinning philosophy for the frequentist approach. The MLE paradigm works on the marginal likelihood $p(Y|\theta)$, or equivalently, the log-likelihood $L(\theta) = \log p(Y|\theta)$, and performs a global optimization over the parameter space.

$$\theta_{MLE} = \arg \max_{\theta} \log p(Y|\theta) \quad (2.9)$$

In many circumstances, there are unobservable states X in the system. Estimating θ_{MLE} in the presence of X becomes a challenging task, since $p(Y|\theta)$ requires the marginalization step, a key integral calculation that is often intractable. As a result, numerical methods such as the EM algorithm [11], [43] and [37].

The fundamental idea behind the Expectation-Maximization (EM) algorithm is to find a lower bound for $L(\theta)$, and iteratively improves and optimizes over the bound. To construct such a bound, we propose an auxiliary density $q(x)$ over X , and construct an auxiliary bivariate function $F(q, \theta)$ such that $L(\theta) \geq F(q, \theta)$ for all proposals q . Here F is a functional with respect to q . More specifically:

$$\begin{aligned} L(\theta) &= \log p(y|\theta) \\ &= \log \int_x p(y, x|\theta) dx \\ &= \log \int_x q(x) \frac{p(y, x|\theta)}{q(x)} dx \\ &\geq \int_x q(x) \log \frac{p(y, x|\theta)}{q(x)} dx \equiv F(q, \theta) \end{aligned}$$

where inequality follows from the concavity of the log function and Jensen's inequality. At the last step, it is defined:

$$F(q, \theta) = \int_x q(x) \log \frac{p(y, x|\theta)}{q(x)} dx \quad (2.10)$$

Different choices of q leads to different lower bounds, which can be optimized more easily. The EM algorithm iteratively optimizes over q and θ :

1. Initialize: Choose a suitable initial guess $\theta^{(0)}$
2. Iteration: For each iteration i , maximize along one dimension, while holding the other fixed at the previous value:

(a) The E-Step: $q^{(i)} = \arg \max_q F(q, \theta^{(i)})$

This step returns a new proposal density. $q^{(i)} = p(x|y, \theta^{(i)})$ such that the state posterior is optimal. See derivations below.

(b) The M-Step: $\theta^{(i+1)} = \arg \max_{\theta} F(q^{(i)}, \theta)$

This step is the key frequentist methodology. A maximizer $\theta^{(i+1)}$ is chosen, rather than a whole parameter posterior distribution. Hence EM is still non-Bayesian.

In the E-Step, it turns out that the posterior $p(x|y, \theta^{(i)})$ gives the exact optimizing q . To verify this, put $p(x|y, \theta^{(i)})$ into $F(q, \theta)$ to get tight bound as equality. A rigorous proof follows from simple variational calculus.

Define the Lagrangian $F_{lag}(q, \theta, \lambda)$ as $F_{lag}(q, \theta) = F(q, \theta) + \lambda(\int_x q(x)dx - 1)$. Setting the functional derivative to 0:

$$\frac{\partial}{\partial q} F_{lag} = \log \frac{p(x, y|\theta)}{q(x)} - 1 + \lambda = 0.$$

therefore,

$$q(x) \propto p(x, y|\theta) = p(x|y, \theta).$$

The crucial requirement for EM algorithm is thus a tractable state posterior. The E-Step can now be simplified. Notice that

$$\begin{aligned} F(q^{(i)}, \theta) &= \int_x p(x|y, \theta^{(i)}) \log \frac{p(x, y|\theta)}{p(x|y, \theta^{(i)})} dx \\ &= \int_x p(x|y, \theta^{(i)}) \log p(x, y|\theta) dx - \int_x p(x|y, \theta^{(i)}) \log p(x|y, \theta^{(i)}) dx. \end{aligned}$$

The second term on the right hand side does not depend on θ , so it is irrelevant for the M-Step. The first term becomes $E_{p(x|y,\theta^{(i)})} \log p(x, y|\theta)$, hence the name of Expectation for E-Step.

In essence, the EM algorithm bypasses the very hard problem of working directly with $p(y|\theta)$, and replaces it with a potentially simpler problem of characterizing state posterior $p(x|y, \theta)$. Under many circumstances, it is possible to get near-analytical results for this density.

There is yet another way of writing the $F(q, \theta)$ expression that yields further insight. We have:

$$F(q, \theta) = \log p(y|\theta) - KL(q \parallel p(x|y, \theta)). \quad (2.11)$$

where

$$KL(f \parallel g) \equiv \int_x f(x) \log \frac{f(x)}{g(x)} dx. \quad (2.12)$$

is the Kullback-Leibler distance from f to g , assuming both are proper probability density functions. Now the lower bound becomes trivially apparent, since KL distance is always nonnegative! The maximization of $F(q, \theta)$ is equivalent to minimization of the KL distance from $q(x)$ to $p(x|y, \theta)$, which is attained for 0 when the two coincide.

While Expectation-Maximization has been well-established in the applied statistics literature, it is only a particular realization of a more general optimization technique. By first setting up a lower bound of the marginal likelihood and then optimize over such a bound, EM effectively opens a way to a broad category of algorithms collectively called “optimization transfer”. When we try to optimize a complex objective function, it is often desirable to instead work with a simpler surrogate. The MM algorithm is introduced to tackle this prob-

lem. MM stands for minorization-maximization for “argmax”-type problems and majorization-minimization for “argmin”-type problems.

The crucial aspect of an MM algorithm is to construct a convenient surrogate objective function, which can facilitate optimization greatly. Usually this step invokes some form of convexity. For example, EM is in fact a special MM instance, where convexity is established through the negative logarithmic function.

The Variational-Bayes EM (VBEM) algorithm

The classic EM algorithm works alternatively on x_t and θ . For simple models, we can derive the exact analytical expression for $p(x|y, \theta)$. In other words, the optimization over q in the E-Step attains its theoretical optimal value. This is a free-form optimization with no constraint on the proposal density. For more complex models or typical multi-dimensional models, it is unrealistic to expect such tractability. Rather, unconstrained functional optimization becomes infeasible.

The Variational Bayes idea builds on this generalization, and seeks to have a well-behaved approximately optimal q . It assumes the additional requirement of conditional posterior independence. Instead of finding the best possible function $q(x_1, \dots, x_m)$ over the entire m -dimensional function space, VB operates on a smaller subspace, where q takes the factored form:

$$q(x) = q(x_1, \dots, x_m) = \prod_{i=1}^m q_i(x_i). \quad (2.13)$$

This factorization is also known as mean field approximation in statistical physics.

For the multi-dimensional case with states $x = (x_1, \dots, x_m)$, we have a proposal density $\prod_{i=1}^m q_i(x_i)$, effectively proposing m marginal densities. The lower bound

becomes:

$$L(\theta) \geq \int_x \prod_{i=1}^m q_i(x_i) \log \frac{p(y, x|\theta)}{\prod_{i=1}^m q_i(x_i)} dx \equiv F(q_1, \dots, q_m, \theta). \quad (2.14)$$

For each iteration, we have:

1. The E-Step: $(q_1^{(i)}, \dots, q_m^{(i)}) = \arg \max_{q_1, \dots, q_m} F(q_1, \dots, q_m, \theta^{(i)})$.
2. The M-Step: $\theta^{(i+1)} = \arg \max_{\theta} F(q_1^{(i)}, \dots, q_m^{(i)}, \theta)$.

Again, define the Lagrangian $F_{lag}(q_1, \dots, q_m, \theta)$. as

$$F_{lag}(q_1, \dots, q_m, \theta) = F(q_1, \dots, q_m, \theta) + \sum_{i=1}^m \lambda_i \left(\int_{x_i} q_i(x_i) dx_i - 1 \right).$$

Using variational calculus to have:

$$\frac{\partial}{\partial q_k} F_{lag} = \int_{x_i, i \neq k} \prod_{i=1, i \neq k}^m q_i \log \frac{p(y, x|\theta)}{\prod_{i=1}^m q_i} dx - 1 + \lambda_i = 0.$$

Upon simplification to get:

$$E_{\sim i} \log p(y, x|\theta) - \log q_i(x_i) = 0.$$

So we have:

$$q_i(x_i) = \exp E_{\sim i} \log p(y, x|\theta), \quad (2.15)$$

where $E_{\sim i}$ takes expectation on all indices other than i . In the $m = 1$ special case, this is reduced to the trivial optimal distribution $q(x) = p(x|y, \theta)$, as the exact posterior.

The system of equations (2.15) forms the IVB cycle, which is the VB-E step. Iteratively, after $q_i^{(i)}, \dots, q_m^{(i)}$ are obtained, the VB-M step can be performed to find the optimizer for θ .

2.3.2 Marginalized Variational Bayes Estimation

A full Bayesian approach treats parameter and states in a uniform way. In contrast to the frequentist EM-class algorithms, numerical Bayesian methods do not have the M-Step over a point estimate of the “best” parameters. Instead, the quantity of interest is the Bayesian likelihood $L = \log p(y)$, or for model selection purposes, $L = \log p(y|\mathcal{M})$.

Classical VB: a brief review

Here we briefly review the Variational Bayes method for completeness. The idea is to find a lower bound of the likelihood. In the trivial case, we seek an unrestricted proposal distribution $q(x, \theta)$. The usual Jensen inequality argument leads to:

$$L = \log p(y) = \log \int_x \int_\theta p(y, x, \theta) dx d\theta = \log \int_x \int_\theta q(x, \theta) \frac{p(y, x, \theta)}{q(x, \theta)} dx d\theta \geq F(q),$$

with $F(q) = \int_x \int_\theta q(x, \theta) \log \frac{p(y, x, \theta)}{q(x, \theta)} dx d\theta$. So that

$$L \geq F(q) = L - KL(q \parallel p(x, \theta|y)).$$

Clearly, the optimal q is just the joint posterior $p(x, \theta|y)$. This naive application of variational calculation gives no gain, as we already know that the posterior contains all information in a Bayesian estimation setting. The problem here is due to the full free-form optimization over q . Without any constraint, q will point to the exact optimal, which is in most cases intractable.

Classical VB instead operates on a smaller feasible region, and propose $q_x(x)q_\theta(\theta) \approx p(x, \theta|y)$, such that

$$q_x, q_\theta = \arg \min_{q_x, q_\theta} KL(q_x(x)q_\theta(\theta) \parallel p(x, \theta|y)).$$

The optimal proposals follow an iterated version in an IVB-cycle.

Marginalized VB (MVB)

In this section, we propose a modified version of Variational Bayes, called the Marginalized Variational Bayes (MVB). The fundamental idea is to further exploit the structure within the system. In many cases, the model has additional levels of analytical tractability. Recall in the particle filtering framework, a brute-force particle representation will require a multi-dimensional particle, which can be highly inefficient and under-representative. When some component dimensions can be analytically marginalized out, then a more efficient scheme, the Rao-Blackwellized Particle Filter (RBPF) can be employed. RBPF assumes that certain dimensions (parameters or states) can have a closed-form posterior, then it is no longer necessary to use particles along these dimensions. In this way, the effective number of dimensions required by particle representation is significantly reduced. RBPF achieves a much higher computational efficiency as compared with the traditional general-purpose particle filter.

Marginalized VB is established along similar insights. For traditional VB, the very first step involves a partitioning of the unknown dimensions. For example, for joint parameter-state estimation, it is customary to impose conditional posterior independence between parameter θ and state x . This is a very restrictive assumption, as the state dynamics almost invariably contains unknown parameters, and their coupling to each other can be substantial.

Another potential problem for VB is again its general-purpose framework. Regardless of the actual number of dimensions for θ and x , VB seeks a unified estimation procedure that often results in a huge IVB cycle. This is acceptable in some

aspects, as essentially parameters and states are identified in a uniform and joint manner. In other cases, however, it is too much to maintain such a “parallel” estimator. Rather, we would break θ and x sequentially into blocks, and estimate one block after another. By doing so, it is crucial that other blocks can be integrated out. The Marginalized VB achieves this, at least in theory.

There has been some modified algorithms in the VB literature. [56] and the book [55] has some summary for the various types of approximations. In particular, the restricted VB is used extensively in this thesis. Instead of optimizing over the individually free-form subspace $(q_1(x_1), q_2(x_2))$, Restricted VB imposes a fixed-form along certain dimensions, and optimize over the (yet smaller) subspace of $(q_1(x_1), \overline{q_2}(x_2))$. Here $\overline{q_2}(x_2)$ is a pre-determined functional form, usually taken as from some simple parametric family. Indeed, our VB-PF can be seen as a special type of restriction, where along the complex dimensions, we don’t assume any fixed form, but rather adopt the general particle representation, i.e, use empirical distribution to approximate that dimension.

As a first step towards a full-blown MVB, consider a simple-minded application. Given (y, x, θ) , if we can marginalize θ analytically, then $p(\theta|x, y)$ is known, and there is no need or benefit in choosing the proposal as $q_x(x)q_\theta(\theta)$. Instead, we attempt to propose $q_x(x)p(\theta|x, y) \approx p(x, \theta|y)$. This is equivalent to proposing $q_x(x) \approx p(x|y)$ given no other constraints. By using variational calculus, it is easy to see that this scenario reduces to a trivial result. The optimizing q_x exactly equals the intended $p(x|y)$, which leads to no additional gain.

The trouble in here is that we are actually not doing any partitioning, since θ is not part of the approximation. The power of VB is only present when there are at least two unknown blocks, each depending on the others in an intertwined

manner. As a result, we re-establish the true Marginalized Variational Bayes as follows:

Given $(x_1, \dots, x_m, \theta)$, MVB assumes the proposal

$$p(\theta|x_1, \dots, x_m) \prod_{i=1}^m q_i(x_i) \approx p(x_1, \dots, x_m, \theta|y).$$

The optimization is taken over all state variables only, since the parameter full conditional posterior is completely known. The derivation here exactly follows our standard technique shown before, and the resulting proposal densities are, no surprisingly:

$$q_i(x_i) = \exp E_{\sim_i} \log p(x_1, \dots, x_m, y). \quad (2.16)$$

Notice the only difference here is the disappearance of θ in the calculations. Moreover, the IVB cycle now focuses on the states. Unlike the full VB framework, MVB does not assume posterior independence between states and parameters. It also does not return an estimate of parameters directly. Rather, it seeks a better estimate of states. When the states are estimated already, they can be fed into the known $p(\theta|x, y)$ to subsequently estimate model parameters.

2.4 Hidden Markov Models (HMM)

A very common type of state-space model is the Hidden Markov Model, where states evolve according to some latent Markov process, and the observations are conditionally independent given their respective states. In mathematical terms, in a HMM (X, Y) , there are two probabilistic mechanisms in HMM:

- Transition: $p(x_t|x_{t-1})$ is the Markov link, also called the prior or state equation. Notice that $p(\cdot|x_{t-1})$ is a natural candidate as a proposal density for

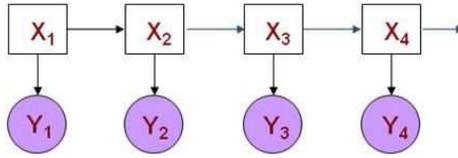


Figure 2.1: A Simplified HMM schematic

state X_t . It contains no information regarding the most recent observation y_t .

- Likelihood: $p(y_t|x_t)$ is the observation equation. We use the likelihood to adjust the weights, and since y_t is directly observable, we never sample from this likelihood distribution.

Together with an initial distribution $p(x_1)$, these two equations completely characterize the dynamics of the states and the observations. The Markovian structure in the HMM model has nice properties that leads to simplifications in subsequent calculations.

A simple schematic of the HMM is shown below.

2.4.1 Inference for HMMs

The complete-data likelihood of a sequence of T observations in an HMM is:

$$p(x_{1:T}, y_{1:T}) = p(x_1)p(y_1|x_1) \prod_{t=2}^T p(x_t|x_{t-1})p(y_t|x_t). \quad (2.17)$$

The probability of the observations $y_{1:T}$ is computed as a sum over all hidden states:

$$p(y_{1:T}) = \sum_{x_{1:T}} p(x_{1:T}, y_{1:T}). \quad (2.18)$$

Given a state transition matrix Q , with $Q = \{q_{jj'}\} = \{p(x_t = j'|x_{t-1} = j)\}$, a symbol observation matrix R , with $R = \{r_{jm}\} = \{P(y_t = m|x_t = j)\}$, and initial state vector distribution π , with $\pi = \{\pi_j\} = \{p(x_1 = j)\}$. These parameters $\theta = (Q, R, \pi)$ satisfies normalization constraints:

$$\sum_{j'=1}^k q_{jj'} = 1, \quad (2.19)$$

$$\sum_{m=1}^p r_{jm} = 1, \quad (2.20)$$

$$\sum_{j=1}^k \pi_j = 1. \quad (2.21)$$

We have:

$$p(x_1|\pi) = \prod_{j=1}^k \pi_j^{x_{1,j}}, \quad (2.22)$$

$$p(x_t|x_{t-1}, Q) = \prod_{j=1}^k \prod_{j'=1}^k q_{jj'}^{x_{t,j}x_{t-1,j'}}, \quad (2.23)$$

$$p(y_t|x_t, R) = \prod_{j=1}^k \prod_{m=1}^p r_{jm}^{x_{t,j}y_{t,m}}. \quad (2.24)$$

and finally get the expression for the log complete-data likelihood:

$$\begin{aligned}
\log p(x_{1:T}, y_{1:T} | \theta) &= \sum_{j=1}^k x_{1:j} \log \pi_j + \sum_{t=2}^T \sum_{j=1}^k \sum_{j'=1}^k x_{t-1,j} \log q_{jj'} x_{t,j'} \\
&+ \sum_{t=1}^T \sum_{j=1}^k \sum_{m=1}^p x_{t,j} \log r_{jm} y_{t,m} \\
&= x_1^T \log \pi + \sum_{t=2}^T x_{t-1}^T \log Q x_t + \sum_{t=1}^T x_t^T \log(R y_t).
\end{aligned}$$

This provides the basis for inference in an HMM. To perform the classical EM algorithm, an M-step is carried out, followed by an E-step to complete one iteration. For details, refer to the well-known Baum-Welch algorithm for completeness. The fundamental trick here is an application of dynamic programming, using the fact of conditional independence inherent in an HMM.

2.5 Variational Bayes Hidden Markov Model

2.5.1 VB-HMM model formulation

A HMM is a stochastic process generated by a stationary, first-order Markov chain whose sequence cannot be directly observed. Instead, we observe a noisy version of the state sequence. As described in the previous section, the states evolve according to its transition matrix $Q = \{q_{jj'}\}$. The T observations $y_i, i = 1, \dots, T$ are generated by such a Markov process. The probability density for y_i at time i , given that the system is in state j , is $p(y_i | x_i = j) = p(y_i | \theta_j)$, where θ_j are the unknown parameters associated with the observation model at state j . Q is also part of the unknown parameters. For notational simplicity, here we assume y_i s are univariate, although the generalization to multivariate data is straightforward.

There are different choices for the observation model $p(y_i|x_i = j)$. Throughout this chapter, we will use two prototype models: Poission Jump-Diffusion and infinite-activity Levy jump. The derivations for the Vb-HMM is similar as follows.

According to the VB philosophy, conditional independence is imposed among different dimensions. Here we choose a factorized prior density:

$$p(Q, \theta) = p(Q)p(\theta). \quad (2.25)$$

In the case of jump-diffusions, we have Poisson-typed finite-activity jumps $I_i J_i$ at time i . The Bernoulli indicator variable $I_i = 1$ indicates the presence of a single jump at i , and J_i denotes its amplitude. A typical choice is $J_i \sim \text{exp}(\lambda)$ are i.i.d exponential jumps with some known λ . Here both I and J are latent state sequences that needs to be inferred.

The joint density of all variables are now written as:

$$p(y, x, \theta, I, J) = \prod_{i=1}^T \prod_{j=1}^k \{p(I_i)p(J_i)p(y_i|I_i, J_i, \theta_j)\}^{x_{i,j}} \cdot \prod_{i=1}^{T-1} \prod_{j_1} \prod_{j_2} (q_{j_1 j_2})^{x_{i,j_1} x_{i+1,j_2}} p(\theta)p(Q).$$

where $x_{i,j}$ is an indicator for which state the chain is in for a given observation, that is, $x_{i,j} = 1$ iff $x_i = j$, and 0 otherwise. The θ_j s are assumed prior independence to get:

$$p(\theta) = \prod_{j=1}^k p(\theta_j). \quad (2.26)$$

The VB framework requires further independence among the state variables x and the parameters θ , so we have

$$q(x, \theta, Q) = q(x) \prod_j q(\theta_j) \prod_{j_1} q(Q_{j_1}). \quad (2.27)$$

Our objective is to find the estimates for the posterior state distribution x , which provides information about the presence of Poission-typed jumps. This corresponds

to an estimate of the current latent state that the hidden Markov chain is in. In order to attain conjugacy, the observation link $p(y_i|I_i, J_i, \theta_j)$ needs to belong to a suitable exponential family. We have actually $p(y_i|I_i, J_i, \theta_j)$ is Gaussian diffusion. Then, the estimate $q(x)$ can be found by standard forward-backward computations involving the Baum-Welch procedure.

2.5.2 Model Specifics

For the transition kernel, Bayesian statistics offers a standard conjugacy formulation. We choose an independent Dirichlet prior for the transition probabilities of each state j :

$$p(Q) = \prod_{j_1} Dir(q_{j_1} | \lambda_{j_1 j_2}). \quad (2.28)$$

with given known hyperparameters $\{\tilde{\lambda}\}$. Conditional on the Poisson jump, the observation becomes a Gaussian link with:

$$p(y_i|I_i, J_i, \theta_j) = N(y_i | \mu_j + I_i J_i, \sigma_j^2). \quad (2.29)$$

The means μ_j are chosen as independent univariate Gaussian conjugate priors, and the variances σ_j^2 have inverse Gamma priors. We have:

$$p\left(\frac{1}{\sigma_j^2}\right) = Gamma\left(\cdot \mid \frac{1}{2}\tilde{\xi}, \frac{1}{2}\tilde{\eta}\right),$$

and

$$p(\mu_j | \sigma_j^2) = N\left(\cdot \mid \tilde{\alpha}, \frac{1}{\tilde{\beta}\sigma_j^2}\right).$$

for given known hyperparameters $\tilde{\xi}$ and $\tilde{\eta}$. With these choices, the variational posteriors maintain identical distributional forms, respectively, with updated parameters. More specifically, all transition probability parameters still follow Dirichlet

distribution:

$$q(Q) = \prod_j Dir(q_{j_1} | \lambda_{j_1 j_2}^{\text{new}}),$$

with

$$\lambda_{j_1 j_2}^{\text{new}} = \lambda_{j_1 j_2}^{\sim} + \sum_{i=1}^{T-1} q(x_i = j_1, x_{i+1} = j_2). \quad (2.30)$$

The Gaussian link is similarly updated:

$$q\left(\frac{1}{\sigma_j^2}\right) = Gamma(\cdot | \frac{1}{2} \xi_j^{\text{new}}, \frac{1}{2} \eta_j^{\text{new}})$$

and

$$p(\mu_j | \sigma_j^2) = N(\cdot | \alpha_j^{\text{new}}, \frac{1}{\beta_j^{\text{new}} \sigma_j^2})$$

with:

$$\beta_j^{\text{new}} = \beta_j + \sum_{i=1}^T q_{ij} \quad (2.31)$$

$$\alpha_j^{\text{new}} = \frac{1}{\beta_j^{\text{new}}} (\beta_j \alpha_j + \sum_{i=1}^T q_{ij} (y_i - I_i J_i)) \quad (2.32)$$

$$\xi_j^{\text{new}} = \xi_j + \sum_{i=1}^T q_{ij} \quad (2.33)$$

$$\eta_j^{\text{new}} = \eta_j + \sum_{i=1}^T q_{ij} y_i^2 + \beta_j \alpha_j^2 - \beta_j^{\text{new}} \alpha_j^{\text{new}2} \quad (2.34)$$

To complete the computation, we need the auxiliary variables $q(x_i = j_1, x_{i+1} = j_2)$ and $q_{ij} = q(x_i = j)$. These come from standard HMM inference procedures based on forward backward calculations.

2.6 Case study: jump detection in gene sequences

2.6.1 Overview

In this section, we apply our VB-HMM framework to genetic data. This is part of an ongoing research called Genome-wide association (GWA). GWA utilize a large number of genetic variants data, called single nucleotide polymorphisms (SNPs), across the entire genome in order to identify genetic basis mutations for some particular trait of interest. Data for the trait of interest are collected on the order of tens of thousands. The traditional methodology of testing SNPs one at a time is quite challenging due to the heavy computational burden. Traditional Bayesian methodology based on MCMC is based on processing the entire dataset as a whole, thus usually requiring parallel computing and enormous amounts of CPU time. Overall, statistical inference of potential jump variations poses a curse of high dimensionality.

Our Variational Bayes Hidden Markov Model estimator (VB-HMM) provides one alternative possibility of scaling this challenge to estimate posterior distributions. Specifically, the dataset is viewed as a sequential flow of some underlying stochastic process with normal diffusive and jumpy components, respectively. Since we are mostly interested in identifying “significant” jumps, it is natural to model the hidden states as a straightforward 2-state Markov chain, alternating between normal and mutational. This is not a strict restriction, and can be easily extended to multi-state chains in subsequent studies. Due to the experimental nature of data collection and quality control, the effective data points are observed with some random error. We have performed estimation using two different error modeling: the Poisson jump-diffusion and infinite-activity Levy jumps. The model shaping pa-

rameters, along with unknown states of the volatility and jump time/amplitudes, need to be estimated. Clearly, such a complex HMM setup makes it extremely difficult to perform simple likelihood-based inference. Standard MCMC or even particle filters can be too time-consuming.

In contrast, the VB-HMM algorithm is inherently sequential. At each time point, it iteratively updates only a handful of VB-moments and parameter estimates, and propagate the system of marginal equations forward. We use conjugate prior/posterior pairs when possible. For transition matrices, Dirichlet distribution is applied. For normal observation with unknown variance, the Normal-Inverse-Gamma distribution is used. Again, this can be generalized to other non-standard distributions where only a particle representation is feasible. In that case, we switch to the VB-PF framework, using particles parsimoniously. Throughout, VB-HMM methods prove to be quite efficient. Typical computational time on a dataset of size 50,000 is within an hour, as compared to possibly days for traditional methods.

2.6.2 The dataset

This study focuses on the body weight data from domestic dogs. Many examples of the traits with underlying genetic variants have been identified through the past decade, including color, hair length and skeletal size. In particular, body weight is an important trait with a huge amount of variation. Simply put, different dogs can have weights that are vastly different. Biostatisticians have been working to identify certain genetic loci that accounts for the traits inheritance.

The Bayesian analysis of genetic data takes as input the P-value sequence of SNP data. P-values offer a summarized way of understanding the genetic expres-

sion along the genome. Our dataset contains 42,396 p-values for a dog body weight observation sequence. The data are extracted from 844 dogs from 76 breeds and their breed averages of body weight are used as phenotypes. This dataset has been used to determine significant genetic jumps that might be responsible for the differences in body weights. The original dataset is dense and undetectable. There has been parallel research along this line, particularly the thesis work of Li. In [34], he uses various Bayesian computational methods to scan for body weight data, and finds six strongest hits occurring at CFA 15.44226659, CFA X.106866624, CFA 10.11440860, CFA X.86813164, CFA 4.42351982 and CFA 7.46842856.

We convert raw data into logarithms, and perform variational Bayes sequential estimation on the transformed data. The sequence is assumed to follow a 2-state hidden Markov chain, and observed through Gaussian noise with additive, exponential Poisson jumps. The inference is based on the VB-HMM algorithm as detailed in this chapter. Our simulation result is shown in the following figures. It clearly captures certain “significant” jumps, where a genetic mutation is most likely.

2.6.3 Estimation Results and Discussion

The main results are reported here.

We found 8 “significant” trait loci along the given dataset, corresponding to CFA X.866, CFA X.104, CFA 10.145, CFA 14.896, CFA 5.278, CFA 5.409, CFA 14.122, CFA 20.808. For comparison, we see the first 4 loci matches very closely. The significance of the remaining jumps are somewhat more blurring.

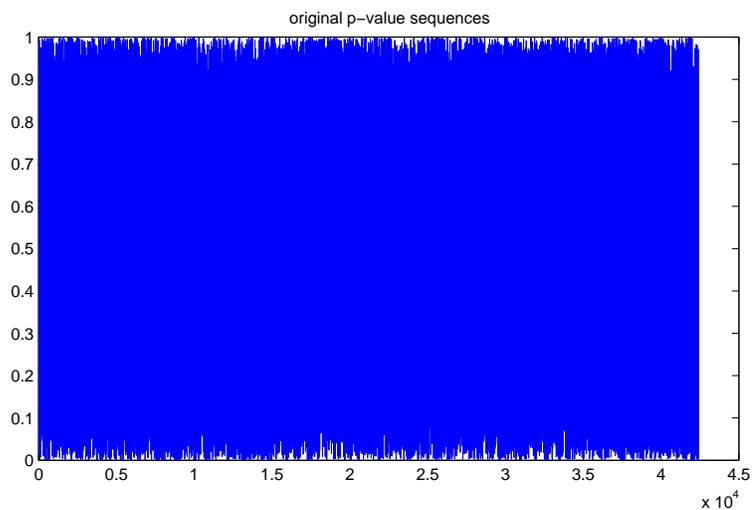


Figure 2.2: Original p value sequence of the dog weight dataset. There are 42,396 data points, within the $[0,1]$ range. Jump identification on the raw data is nearly impossible.

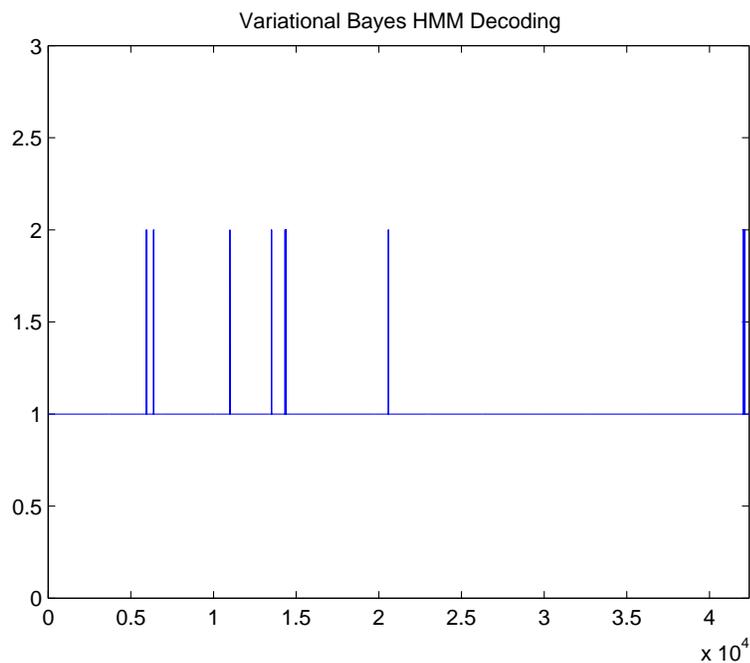


Figure 2.3: Detected jumps with VB-HMM estimation. 8 most significant jumps are detected. The vertical scale corresponds to the choice of a particular HMM, with state 1 being normal and state 2 possible genetic mutational.

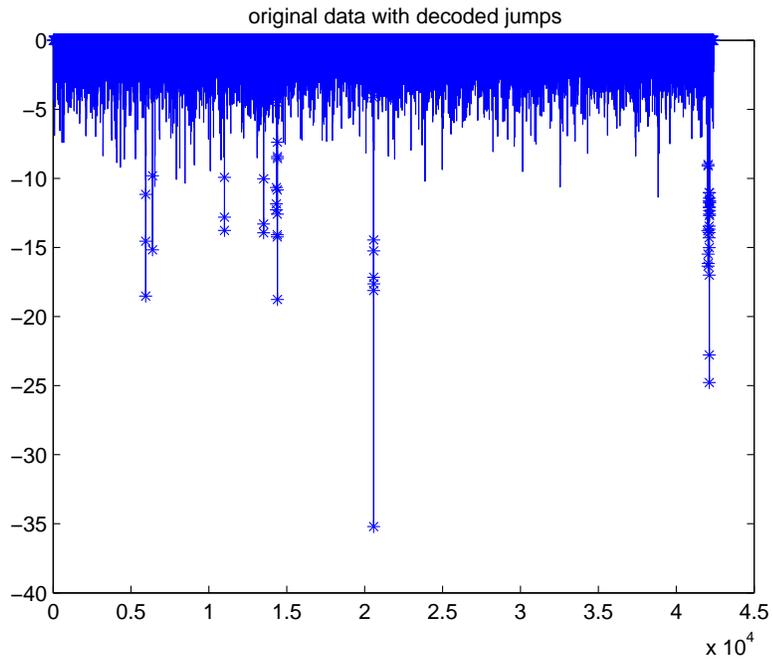


Figure 2.4: Original logarithm p values. The vertical scale corresponds to the logarithms of the raw data. The jumps are clustered around 8 significant regions. Vb-HMM essentially switches between 2 states sequentially

CHAPTER 3
PARTICLE FILTERING METHODS FOR STATE-PARAMETER
ESTIMATION

3.1 Introduction

Many problems require estimation of a system that changes over time and observed with noise. The state-space approach focuses on the the state vector of a system. Such a state representation is very convenient; for example, in financial econometrics, volatility and jumps are typical states. The state representation provides a convenient way to incorporate multivariate data and nonlinear/non-Gaussian processes.

To make inference in state-space modeling, we need a model of the evolution of the state itself through time, the so-called system model, and secondly, an observation model relating states to actual observable data. These models are represented in a probabilistic form. The flexible nature of state-space modeling poses problems for efficient inference. This is made still more challenging by the presence of unknown parameters, which can be thought of as “static” states.

In the Bayesian approach, the posterior distribution is all that is needed to make inference. In practice, it is highly desirable to have an estimate of the posterior that gets updated with the arrival of new data. In other words, the estimates are sequentially updated, without having to go from scratch with all available data. This is the fundamental idea of Bayesian filtering. The celebrated Kalman filter is one classical example that deals with the simplest of all: linear Gaussian in both system and observation models. For more general dynamics, particle filters offer

a practical generalization. They are attractive alternatives to MCMC due to the relative low computational overhead.

In this chapter, we discuss the methodology and implications of particle filtering. We conducted extensive simulation for standard and extended versions of particle filters. In many scenarios, a brute-force application of particle filtering is still too computational intensive, especially in the presence of a high dimensional unknown parameter-state vector. We combine Variational Bayes with particle filtering to construct a general-purpose VB-PF framework. Simulation results indicate that VB-PF can effectively perform inference with multiple unknown parameters, a situation where traditional particle filters lack accuracy and MCMC approaches are too time-consuming.

The basic mechanism of particle filtering works like this: The state space is partitioned into different portions. Particles, which are random samples drawn according to some probability measure, represent an approximate posterior density by point-mass histogram. The particle system evolves through time according to the state model, and then weights are adjusted to correct according to the observation model.

Historically called the bootstrap filter or CONDENSATION algorithm [26], there is a huge literature on particle filters over the past decade. [15] is the classic reference that introduces the concept of particle filtering into the Bayesian simulation community. [8], [30] and [31] are early contributions. See [59] for a comprehensive survey.

3.2 The particle filtering framework

3.2.1 Basic concepts in random sampling

Definition of a random Sample

Statistical simulation is the process of constructing a discrete sample to approximate an unknown distribution. We are interested in obtaining a sample of size N (denoted N -sample henceforth). Let (x_1, \dots, x_N) be the N -sample. For a complete characterization of the sample, there are two aspects:

- the target distribution $p(x)$;
- the weights associated with the sample. We specify explicitly two types of samples based on their associated weights;
 - Uniform N -sample, with weights $(\frac{1}{N}, \dots, \frac{1}{N})$ corresponding to the discrete uniform distribution. This is the default case, and also provides the basis for the histogram.
 - Weighted N -sample, with weights (w_1, \dots, w_N) corresponding to the categorical distribution. This is the most general case. It can be seen as an instance of the multinomial distribution.

We will write

$$\begin{pmatrix} x_1 & \cdots & x_N \\ w_1 & \cdots & w_N \end{pmatrix} \sim p(x).$$

Resampling

Resampling here means the conversion between a uniform N -sample and its equivalent weighted N -sample. We use the following two-way conversions frequently:

- **Weighted-to-Uniform:** Given a weighted N -sample

$$\begin{pmatrix} x_1 & \cdots & x_N \\ w_1 & \cdots & w_N \end{pmatrix} \sim p(x),$$

we resample according to the multinomial(categorical) distribution defined by the weights (w_1, \dots, w_N) a total of N times, and obtain a new N -sample (x'_1, \dots, x'_N) , then

$$\begin{pmatrix} x'_1 & \cdots & x'_N \\ \frac{1}{N} & \cdots & \frac{1}{N} \end{pmatrix} \sim p(x).$$

- **Uniform-to-Weighted:** Given a uniform N -sample

$$\begin{pmatrix} x_1 & \cdots & x_N \\ \frac{1}{N} & \cdots & \frac{1}{N} \end{pmatrix} \sim p(x),$$

we resample according to the multinomial(categorical) distribution defined by the weights (w_1, \dots, w_N) a total of N times, and obtain a new N -sample (x'_1, \dots, x'_N) , then

$$\begin{pmatrix} x'_1 & \cdots & x'_N \\ \frac{1}{w_1} & \cdots & \frac{1}{w_N} \end{pmatrix} \sim p(x).$$

Importance Sampling (IS)

The fundamental issue in simulation is that the target distribution $p(x)$ is usually difficult to sample from. We first focus on the 1-dimensional scalar case.

Assume a single random variable X has distribution $p(x) = CF(x)$ for some easy-to-evaluate known function $F(x)$ and some hard-to-compute unknown normalizing constant C . We usually cannot directly get a sample for X , so we choose a proposal density $q(x)$, which is easy to sample from, and obtain a N -sample based on $q(\cdot)$.

So naturally we have

$$\begin{pmatrix} x_1 & \cdots & x_N \\ \frac{1}{N} & \cdots & \frac{1}{N} \end{pmatrix} \sim q(\cdot).$$

Such a uniform N -sample is only a representation of the proposal. To get back to the target distribution, we simply adjust each sample point x_i by its weight $w_i = \frac{p(x_i)}{q(x_i)}$. In Bayesian calculations, we will sometimes use $=$ and \propto interchangeably.

Then

$$\begin{pmatrix} x_1 & \cdots & x_N \\ w_1 & \cdots & w_N \end{pmatrix} \sim p(\cdot).$$

We can add an optional step of Weighted-to-Uniform Resampling to get:

$$\begin{pmatrix} x'_1 & \cdots & x'_N \\ \frac{1}{N} & \cdots & \frac{1}{N} \end{pmatrix} \sim p(\cdot),$$

and finally back to discrete uniform. In order to have fast convergence and good approximation, it is critical to choose an appropriate proposal $q(\cdot)$.

Sampling from a Mixture distribution

In many situations, the target distribution $p(x)$ can be decomposed into a number of more basic distributions $p_i(x)$ as “building blocks”, each of which can then be sampled from. This is ideally represented in a mixture distribution, where:

$$p(x) = \sum_{i=1}^n \lambda_i p_i(x)$$

Assume each p_i can be directly sampled from or using IS with suitable proposals, then there are a few methods to obtain a N -sample for the overall target p . Here for simplicity, we first normalize the coefficients to have a valid weight: $\lambda_i = \frac{\lambda_i}{\sum_i \lambda_i}$ such that $\sum_{i=1}^n \lambda_i = 1$.

- (Two-level sampling) Draw from a multinomial distribution with weight λ_i and obtain (m_1, \dots, m_n) such that $\sum_{i=1}^n m_i = N$. Draw a uniform m_i -sample from distribution $p_i(\cdot)$, then combine to get a uniform N -sample for $p(\cdot)$
- (One-level sampling) Choose a deterministic number m_i of samples for each individual distribution, such that $\frac{m_i}{m_j} = \frac{\lambda_i}{\lambda_j}$, the relative frequency of actual number of samples is equal to the relative ratio of weights. Draw a uniform m_i -sample from distribution $p_i(\cdot)$, then combine to get a uniform N -sample for $p(\cdot)$. This simplifies the two-level sampling while obtaining the correct sample asymptotically.

As a special case to the second method, assume $n = N$, and all $\lambda = 1$. We only need to draw a single point $x^{(i)}$ from $p_i(\cdot)$ for each distribution ($i = 1, \dots, N$).

Then

$$\begin{pmatrix} x^{(1)} & \dots & x^{(N)} \\ \frac{1}{N} & \dots & \frac{1}{N} \end{pmatrix} \sim p(x).$$

This provides the basis for particle filtering.

Sequential Importance Sampling(SIS)

In the above, we have summarized the basics for sampling from a single 1-dim scalar distribution via IS. Now it is natural to consider a sequence of pdfs $p_t(x)$, ($t = 1, \dots, T$) and devise ways to sample from all of these distributions sequentially.

In the Bayesian context, the $p_i(\cdot)$ s are posterior densities given the observed data. Denote the unobserved state sequence x_t , the observed data sequence y_t , where ($t = 1, \dots, T$). Also denote $y^t = (y_1, \dots, y_t)$ is the accumulated data up to time t . Then our filtering problem is formulated as drawing samples from $p(x_t|y^t)$.

3.2.2 The Particle Filter(PF)

There is a vast literature on analysis of state space models using Markov Chain Monte Carlo (MCMC) methods, see [59] for an extensive survey. In financial applications, [14] implemented MCMC to estimate a stochastic volatility model with jumps in both returns and volatility, and [39] apply MCMC to estimate the Levy jump stochastic volatility models. Given an entire sample of available data, their methods deliver quite accurate parameter estimates. MCMC methods work for a large variety of complex dynamics, although it usually needs to be customized for specific models.

However, the MCMC approach is inherently a batch method, which has to restart the estimation procedure from scratch every time a new piece of data is obtained. In this way, it is highly computationally intensive. In many real world applications, we are more interested in sequentially learning about the models. With the arrival of new information, we would ideally like to have our estimates updated in real-time. In this sense, we turn to the alternative approach of simulation methods: Sequential Monte Carlo (SMC). A particle filter is a class of SMC methods that sequentially updates the posterior estimates. It provides a trade-off to achieve time efficiency at the expense of some loss of accuracy at still acceptable level.

A Particle Filter is a sequential Bayesian simulation algorithm to sample from the filtered distributions $p(x_t|y^t)$ for $(t = 1, \dots, T)$. The key is a recursive updating relationship between two successive densities, moving from $p(x_{t-1}|y^{t-1})$ to $p(x_t|y^t)$. Depending on the order of calculation, PF is classified into two basic structures, the Standard Particle Filter (SPF) and the Auxiliary Particle Filter (APF). We here include a brief summary of the two structures, and highlight their close connections. All calculations invariably start from $p(x_t|y^t)$, and relate it to its predecessor.

In this thesis, we develop a specialized version of particle filter algorithm to perform state filtering and parameter estimation for various models. The algorithm combines the exact parameter learning algorithm proposed in [27] and the optimal filtering methodology in [28]. We propose a Metropolis-Hastings MCMC for each time that also includes the auxiliary index for the particles. We include data augmentation to reduce discretization errors, following the steps outlined in [20]. The models we consider include stochastic volatility diffusions, and will include infinite-activity Levy process models of financial returns.

We build on a growing literature of particle filter applications. [10] provides the general framework for the two tasks of state filtering and parameter learning in a discrete-time setting. [36] uses the same framework to estimate the latent volatility process, but assumes known parameters or estimate parameters elsewhere. [18] deals with the continuous-time case, where parameters are estimated before states are filtered. [25] applies particle filtering to discrete-time GARCH regime switching models, and uses a different parameter updating scheme. [33] deals with the infinite activity Levy models, again only with state filtering. Our paper adds to the current research in this direction and will provide some viable methodology for more general continuous-time models.

Particle Filters: Pure State Filtering

Assume we have a state space model. Denote $x^t = \{x_0, x_1, \dots, x_t\}$ as the vector of states up to time t . Denote $y^t = \{y_0, y_1, \dots, y_t\}$ as the accumulated data up to time t . Given all observations y^T , the joint estimation problem involves the complete characterization of the joint posterior density $p(\theta, x^T | y^T)$. This is usually intractable, and MCMC methods are applied to sample from this high dimensional density. MCMC algorithm reads all y^T as input, and effectively produce a smoothed estimation of all states and the parameters. As pointed out in the Introduction, this is often undesirable or impractical, as we need real time updates of state filtering and parameter estimates.

In the sequential estimation setting, we are interested in the marginal densities $p(x_t | y^t)$ and $p(\theta | y^t)$. Even these significantly reduces dimensions, it is usually still analytically intractable, and we are required to perform sequential simulation, drawing samples from the sequence of posterior densities $p(x_t, \theta | y^t)$, for $t \in \{0, 1, \dots, T\}$. The particle filter is a sequential Monte Carlo method to sample from these distributions. The fundamental idea of particle filter is to use a group of samples (N particles) to approximate the target density, and evolve the particles forward in time. Suppose we have N particles $x_t^{(i)}$, $i \in \{1, \dots, N\}$, representing $p(x_t | y^t)$. The key is then a recursive updating scheme between two successive densities.

In this subsection, we consider pure state filtering, namely, moving from $p(x_t | y^t)$ to $p(x_{t+1} | y^{t+1})$. Depending on the order of calculation, there are two basic structures.

Particle Filters: State Filtering + Parameter Estimation

When the parameters are also unknown, particle filters need to jointly estimate state and parameters. This is a difficult problem, since state propagation depends on the parameters. There have been much research into this joint identification problem. In the original pathbreaking paper, [19] simply adds an artificial evolution to the parameters, thus treating parameters as part of an enlarged state vector. This method is straightforward to implement, but it is less than accurate because, after all, parameters are not states. The artificial dynamics imposed on the parameters increase the noise and variance of the estimates, and the quality of estimates tend to deteriorate with the passage of time. To correct for this “overdispersion”, [38] introduce kernel based smoothing with normal kernels. Their method removes the problem of information loss over time via shrinkage of the normal variances.

An alternative approach is proposed in [57], on the basis that the posterior for parameters $p(\theta|x_t, y^t)$ is analytically tractable, and it depends on the observed data and latent states only through a set of sufficient statistics. In many applications, sufficient statistics can be formulated through the use of conjugate priors in the first place. This method is further exploited in [27] where parameter learning is made theoretically sound and exact. APF structure is applied as the fundamental algorithm for state filtering.

This exact parameter learning framework under APF is attractive, but there are still difficulties. One problem arises when the state posterior $p(x_{t+1}|x_t^{(i)}, y_{t+1})$ is hard to sample from, making step 2 difficult. In comparison, [20] uses a brute-force MCMC at each time t to draw samples from the full posterior, before propagating to $t + 1$. They essentially make no distinction among the N particles, but their

method first draw the parameter samples based on kernel smoothing normal densities, therefore possibly losing accuracy. Along this line, we modify the framework by combining the merits in [28] and [18]. Specifically, we keep the APF structure to exploit its “look-ahead” smoothing property, while use a MCMC Metropolis-Hastings step to effectively sample from the target distribution and obtain the new set of particles.

Sequential Importance Sampling Resampling(SISR-PF)

In standard particle filtering, the particles are first sampled according to a particular proposal density, essentially an importance sampling step, followed by recalculation of the updated weights. In order to maintain a uniform sample after each time step, an additional resampling step follows. Let

$$\begin{aligned}
p(x_t|y^t) &= p(x_t, y_t) = \int p(x_t, y^t, x_{t-1}) dx_{t-1} \\
&= \int p(y^{t-1}, x_{t-1}, x_t, y_t) dx_{t-1} \\
&= \int p(y^{t-1}) p(x_{t-1}|y^{t-1}) p(x_t|y^{t-1}, x_{t-1}) p(y_t|x_t, y^{t-1}, x_{t-1}) dx_{t-1} \\
&= \int C p(x_{t-1}|y^{t-1}) p(x_t|x_{t-1}) p(y_t|x_t) dx_{t-1}.
\end{aligned}$$

By recursion, X_{t-1} is represented by a uniform N -sample:

$$\begin{pmatrix} x_{t-1}^{(1)} & \cdots & x_{t-1}^{(N)} \\ \frac{1}{N} & \cdots & \frac{1}{N} \end{pmatrix} \sim p(x_{t-1}|y^{t-1}).$$

Equivalently, the discrete uniform density function can be written as:

$$p(x_{t-1}|y^{t-1}) = \frac{1}{N} \sum_{i=1}^N \delta(x_{t-1} = x_{t-1}^{(i)}) = \sum_{i=1}^N \delta(x_{t-1} = x_{t-1}^{(i)}). \quad (3.1)$$

where again, the normalizing constant can be suppressed throughout Bayesian calculations. Upon substitution of (3.1), we get:

$$\begin{aligned}
 p(x_t|y^t) &= \int \sum_{i=1}^N \delta(x_{t-1} = x_{t-1}^{(i)}) p(x_t|x_{t-1}) p(y_t|x_t) dx_{t-1} \\
 &= \sum_{i=1}^N p(x_t|x_{t-1}^{(i)}) p(y_t|x_t).
 \end{aligned} \tag{3.2}$$

The equation (3.2) is the fundamental equation for SPF. Clearly, $p(x_t|y^t)$ is represented as a mixture distribution with $n = N$ and equal weights. The problem of sampling from $p(x_t|y^t)$ is reduced to obtaining a single sample from $p(\cdot|x_{t-1}^{(i)})p(y_t|\cdot)$. According to IS theory, it is sufficient to choose a proposal $q(\cdot)$, and draw a sample $x_t^{(i)}$ from $q(\cdot)$, and attach to it the weight

$$w_t^{(i)} = \frac{p(x_t^{(i)}|x_{t-1}^{(i)})p(y_t|x_t^{(i)})}{q(x_t^{(i)})}.$$

How does one find a suitable proposal density $q(x_t)$? Under the SPF framework, there are many possible choices for the proposal. The most common in the literature is the simple transition density $p(x_t|x_{t-1})$, as long as it can be directly sampled from. Here the weights calculation is greatly simplified to be left with only the likelihood. In sum,

1. Draw 1 sample point $x_t^{(i)}$ from $p(\cdot|x_{t-1})$;
2. Assign weight $p(y_t|x_t^{(i)})$, normalize the weights. Then

$$\left(\begin{array}{ccc} x_t^{(1)} & \cdots & x_t^{(N)} \\ p(y_t|x_t^{(1)}) & \cdots & p(y_t|x_t^{(N)}) \end{array} \right) \sim p(x_t|y^t).$$

Notice that our derivation of (3.2) presupposes a uniform N -sample for the previous posterior, as expressed in (3.1), while after the two steps above, each sample point has its own weight. As a result, it is mandatory to reset all weights back to discrete uniform.

3. Weighted-to-Uniform Resampling to get

$$\begin{pmatrix} x_t^{(1)} & \cdots & x_t^{(N)} \\ \frac{1}{N} & \cdots & \frac{1}{N} \end{pmatrix} \sim p(x_t|y^t).$$

Auxiliary Particle Filter (APF)

In comparison to the standard SISR-PF, the Auxiliary Particle Filter will not perform any importance sampling. Instead, APF first adjusts the “predictive” weights associated with each current sample. It already transforms a uniform sample to a biased one, where the bias comes from the corrections due to the current most recent observation. Then APF performs an exact sampling step, restoring uniformity on the resulting samples. Let

$$\begin{aligned} p(x_t|y^t) &= p(x_t, y_t) = \int p(x_t, y^t, x_{t-1}) dx_{t-1} \\ &= \int p(y^{t-1}, x_{t-1}, y_t, x_t) dx_{t-1} \\ &= \int p(y^{t-1}) p(x_{t-1}|y^{t-1}) p(y_t|y^{t-1}, x_{t-1}) p(x_t|y_t, y^{t-1}, x_{t-1}) dx_{t-1} \\ &= \int C p(x_{t-1}|y^{t-1}) p(y_t|x_{t-1}) p(x_t|x_{t-1}, y_t) dx_{t-1}. \end{aligned}$$

Again, upon substitution of (3.1), we get:

$$\begin{aligned} p(x_t|y^t) &= \int \sum_{i=1}^N \delta(x_{t-1} = x_{t-1}^{(i)}) p(x_{t-1}|y^{t-1}) p(y_t|x_{t-1}) p(x_t|x_{t-1}, y_t) dx_{t-1} \\ &= \sum_{i=1}^N p(y_t|x_{t-1}^{(i)}) p(x_t|x_{t-1}^{(i)}, y_t). \end{aligned} \tag{3.3}$$

(3.3) is the fundamental equation for APF. It is instructive to compare (3.3) with (3.2). For the APF framework, the posterior $p(x_t|y^t)$ is still represented as a mixture distribution with $n = N$, but this time with weights related to the previous particle value $x_{t-1}^{(i)}$. Also, x_t appears only through one function $p(\cdot|x_{t-1}^{(i)}, y_t)$,

although it is unclear how this function can be easily handled. Following the sampling scheme for such a mixture distribution, we have the APF algorithm:

1. Draw 1 sample point $x_t^{(i)}$ from $p(\cdot|x_{t-1}^{(i)}, y_t)$;
2. Assign weight $p(y_t|x_{t-1}^{(i)})$, normalize the weights. Then

$$\begin{pmatrix} x_t^{(1)} & \cdots & x_t^{(N)} \\ p(y_t|x_{t-1}^{(1)}) & \cdots & p(y_t|x_{t-1}^{(N)}) \end{pmatrix} \sim p(x_t|y^t);$$

3. Weighted-to-Uniform Resampling.

The two structures of the Particle Filter appears almost identically symmetric. However, in most literature, the APF is formulated through a different expression. We first perform Uniform-to-Weighted resampling to adjust the previous posterior so that $x_{t-1}^{(i)}$ has weight $w^{(i)}$ (to be determined). Then we have

$$p(x_{t-1}|y^{t-1}) = \sum_{i=1}^N w^{(i)} \delta(x_{t-1} = x_{t-1}^{(i)}).$$

Now the update posterior becomes:

$$\begin{aligned} p(x_t|y^t) &= \int C p(x_{t-1}|y^{t-1}) p(y_t|x_{t-1}) p(x_t|x_{t-1}, y_t) dx_{t-1} \\ &= \int \sum_{i=1}^N w^{(i)} \delta(x_{t-1} = x_{t-1}^{(i)}) p(x_{t-1}|y^{t-1}) p(y_t|x_{t-1}) p(x_t|x_{t-1}, y_t) dx_{t-1} \\ &= \sum_{i=1}^N w^{(i)} p(y_t|x_{t-1}^{(i)}) p(x_t|x_{t-1}^{(i)}, y_t). \end{aligned} \tag{3.4}$$

According to the Uniform-to-Weighted resampling weight formula, if we resample according to multinomial distribution given by $p(y_t|x_{t-1}^{(i)})$, then $w^{(i)} = \frac{1}{p(y_t|x_{t-1}^{(i)})}$ to yield the simple result:

$$p(x_t|y^t) = \sum_{i=1}^N p(x_t|x_{t-1}^{(i)}, y_t).$$

The equivalent APF algorithm becomes:

1. Uniform-to-Weighted resampling according to $p(y_t|x_{t-1}^{(i)})$; Still call the new sample $x_{t-1}^{(i)}$
2. Draw 1 sample $x_t^{(i)}$ from $p(\cdot|x_{t-1}^{(i)}, y_t)$. Now there is no need to assign weights to $x_t^{(i)}$, they together already satisfy the uniform final condition:

$$\begin{pmatrix} x_t^{(1)} & \cdots & x_t^{(N)} \\ \frac{1}{N} & \cdots & \frac{1}{N} \end{pmatrix} \sim p(x_t|y^t).$$

Comparison of SISR-PF and APF

Both SPF and APF are derived through a manipulation of the joint density $p(y^{t-1}, x_{t-1}, y_t, x_t)$ and expansion by Markovian properties. The APF can be seen as a special case of SPF where the proposal is chosen to be the conditional posterior $p(x_t|x_{t-1}^{(i)}, y_t)$. Indeed, APF provides the best possible proposal for SPF, since any proposal $q(\cdot)$ can at most depend on the most recent state $x_{t-1}^{(i)}$ and the newest observation y_t , hence $q(\cdot) = q(\cdot, x_{t-1}^{(i)}, y_t)$. The conditional density optimally uses all available information in the two known quantities.

In sum, SPF with transition proposal is easiest and simplest to implement, but it suffers from relatively poor quality, since $p(x_t|x_{t-1})$ may be far from $p(x_t|x_{t-1}, y_t)$ depending on the actual value of observation. On the contrary, APF is most accurate with the best proposal density, but it is almost always most analytically intractable with the difficult-to-approximate function $p(x_t|x_{t-1}^{(i)}, y_t)$ involved and the difficult-to-calculate weights for the predictive likelihood $p(y_t|x_{t-1})$.

3.2.3 Extensions of standard particle filtering

The Unscented Standard Particle Filter(USPF)

From previous discussions, there is a natural trade-off of complexity versus accuracy between the two prevailing PF frameworks. SPF and APF represent the two extremes. For many applications, it is desirable to obtain both approximately good accuracy as well as relatively simple implementation. This puts more requirement on the choice of the proposal density $q(\cdot, x_{t-1}^{(i)}, y_t)$ to be used for the i -th individual sampling of the target mixture distribution (hence the i -th particle of the eventual N -sample). The idea of normal proposal has appeared in some literature before. For purely linear system, Kalman filter provides the classical solution to posterior estimation. For a wide class of mildly nonlinear systems, we can use a best-possible normal approximate to the posterior as the proposal. The normal distribution is best-possible in the sense that it approximately matches the posterior mean and variance. There are two aspects in this:

1. Determine the mean and variance through a nonlinear transformation.
2. Propagate the updated mean and variance estimate recursively through time.

The goal of the two steps is to obtain estimates

$$m_t^{(i)} \simeq E(x_t | x_{t-1}^{(i)}, y_t);$$

and

$$v_t^{(i)} \simeq Var(x_t | x_{t-1}^{(i)}, y_t).$$

Then choose normal proposal $q(\cdot, x_{t-1}^{(i)}, y_t) \sim N(m_t^{(i)}, v_t^{(i)})$, draw a single sample $x_t^{(i)}$, and adjust weights accordingly. In our algorithm, step 1 is done through some

form of a “moment matching” method called the “Scaled Unscented Transformation” (SUT), and step 2 is done through the formal Kalman Filtering framework.

Piecing all things together, we can write down the complete algorithm for the unscented standard particle filter: For each time $t = 2, \dots, N$, perform:

SUT For each particle $x_{t-1}^{(i)}$, use SUT to propagate through to time t .

KF For $x_{t-1}^{(i)}$, apply Kalman filtering to get the updated estimates $m_t^{(i)}$ and $v_t^{(i)}$.

SPF Draw sample $x_t^{(i)}$ by normal proposal $q(\cdot)$, normalizing weights, Weighted-to-Uniform resampling.

HMM and the USPF algorithm

More realistic applications involve a state-space model with unknown parameters. We first keep things simple, and consider a model with 1-dim state x and 1 single unknown parameter θ . This is a generalization of HMM:

Transition $p(x_t|x_{t-1}, \theta)$.

Likelihood $p(y_t|x_t, \theta)$.

The goal is to recursively establish $p(\theta, x_t|y^t)$. First, the N -sample particle representation applies to the 2-dimensional (x_t, θ) pair. The discrete uniform N -sample approximation to $p(\theta, x_{t-1}|y^{t-1})$ is

$$p(\theta, x_{t-1}|y^{t-1}) = \sum_{i=1}^N \delta(x_{t-1} = x_{t-1}^{(i)}, \theta = \theta^{(i)}). \quad (3.1)$$

We use the SPF structure throughout the following derivation;

$$p(\theta, x_t | y^t) \equiv p(x_t | y^t) p(\theta | x_t, y^{(t)}).$$

Assume that there is a sufficient statistic s_t for parameter θ at time t , such that

$$p(\theta | x_t, y^{(t)}) = p(\theta | s_t).$$

s_t then can be recursively calculated given its predecessor s_{t-1} , the most recent state x_t and the newest observation y_t :

$$s_t = S(s_{t-1}, x_t, y_t)$$

for some known and easy-to-evaluate deterministic function S . The second term:

$$\begin{aligned} p(x_t | y^t) &= p(x_t, y_t) = \int p(x_t, y^t, x_{t-1}, \theta) dx_{t-1} d\theta \\ &= \int p(y^{t-1}, x_{t-1}, \theta, x_t, y_t) dx_{t-1} d\theta \\ &= \int p(y^{t-1}) p(\theta, x_{t-1} | y^{t-1}) p(x_t | y^{t-1}, x_{t-1}, \theta) p(y_t | x_t, y^{t-1}, x_{t-1}, \theta) dx_{t-1} d\theta \\ &= \int C p(\theta, x_{t-1} | y^{t-1}) p(x_t | x_{t-1}, \theta) p(y_t | x_t, \theta) dx_{t-1} d\theta \\ &= \int \sum_{i=1}^N \delta(x_{t-1} = x_{t-1}^{(i)}, \theta = \theta^{(i)}) p(x_t | x_{t-1}, \theta) p(y_t | x_t, \theta) dx_{t-1} d\theta \\ &= \sum_{i=1}^N p(x_t | x_{t-1}^{(i)}, \theta^{(i)}) p(y_t | x_t, \theta^{(i)}). \end{aligned} \tag{3.2}$$

Finally we have the recursion:

$$p(\theta, x_t | y^t) = \sum_{i=1}^N p(y_t | x_t, \theta^{(i)}) p(x_t | x_{t-1}^{(i)}, \theta^{(i)}) p(\theta | S(s_{t-1}^{(i)}, x_t, y_t)). \tag{3.3}$$

Again, as always for PF applications, sampling from the joint density is equivalent to sampling from a uniform mixture distribution. The problem is hence reduced to: draw one (\cdot, \circ) pair from the individual distribution $p(\cdot | x_{t-1}^{(i)}, \theta^{(i)}) p(y_t | \cdot, \theta^{(i)}) p(\circ | S(s_{t-1}^{(i)}, \cdot, y_t))$, where for notational clarity, \cdot and \circ represents x_t and θ “to be sampled”, respectively. This is a complicated function of

either x_t or θ , and some appropriate IS proposal has to be applied to get reasonable samples.

Sampling from a target 2-dim density $p(x, y) = CF(x, y)$ parallels the 1-dim case. We sample $(x^{(i)}, y^{(i)})$ from some proposal $q(x, y)$, and assign weight $w^{(i)} = \frac{p(x^{(i)}, y^{(i)})}{q(x^{(i)}, y^{(i)})}$. Focus on a special case, one conditional density $p_{y|x}$ is completely known and can be directly sampled from:

$$p(x, y) = CF(x)p_{y|x}(y|x).$$

Here it suffices to propose a 1-dim density $q_x(x)$ for x :

- Draw $x^{(i)}$ from $q_x(x)$.
- Draw $y^{(i)}$ from $p_{y|x}(y|x^{(i)})$.
- Assign weight $w^{(i)} = \frac{F(x^{(i)})}{q_x(x^{(i)})}$.

Back to $p(\cdot|x_{t-1}^{(i)}, \theta^{(i)})p(y_t|\cdot, \theta^{(i)})p(\circ|S(s_{t-1}^{(i)}, \cdot, y_t))$, θ has a known conditional density.

For x_t , the USPF uses a normal proposal. The complete algorithm is:

- Draw a single $x_t^{(i)}$, assuming fixed parameter $\theta^{(i)}$:
 - SUT** For each particle $x_{t-1}^{(i)}$, use SUT to propagate through to time t .
 - KF** For $x_{t-1}^{(i)}$, apply Kalman filtering to get the updated estimates $m_t^{(i)} \simeq E(x_t|x_{t-1}^{(i)}, \theta^{(i)}, y_t)$ and $v_t^{(i)} \simeq Var(x_t|x_{t-1}^{(i)}, \theta^{(i)}, y_t)$.
 - SPF** Draw sample $x_t^{(i)}$ by normal proposal $q(\cdot, x_{t-1}^{(i)}, \theta^{(i)}, y_t) \sim N(m_t^{(i)}, v_t^{(i)})$.
- Update $s_t^{(i)} = S(s_{t-1}^{(i)}, x_t^{(i)}, y_t)$.
- Draw a single $\theta^{(i)}$ from the conditional $p(\circ|s_t^{(i)})$.
- Assign the weight

$$w^{(i)} = \frac{p(y_t|x_t^{(i)}, \theta^{(i)})p(x_t^{(i)}|x_{t-1}^{(i)}, \theta^{(i)})}{q(x_t^{(i)}, x_{t-1}^{(i)}, \theta^{(i)}, y_t)}.$$

Modified Rao-Blackwellized Particle Filter (MRBPF)

The Rao-Blackwellized Particle Filter is a class of algorithms designed for multi-dimensional state-space models, where some states have special structure to exploit. Those states are conditionally Gaussian linear, so that a standard Kalman filter may be applied. In this subsection, we use this structure for parameter estimation. The parameter is assumed to follow an infinitesimally perturbed linear Gaussian process, and therefore can be estimated via KF after the other nonlinear state has been estimated. The overall algorithm goes like this:

- $t=1$: Set the initial mean and variance of θ e_1 and v_1 according to prior distribution $p(\theta)$; Draw initial N particles of x by the prior $p(x_1)$;
- for $t = 2, \dots, T$;

Regeneration For each particle $i = 1, \dots, N$, draw $\theta^{(i)} \sim N(e_{t-1}, v_{t-1})$.

SUT For each particle $x_{t-1}^{(i)}$, use SUT to propagate through to time t .

KF For $x_{t-1}^{(i)}$, apply Kalman filtering to get the updated estimates $m_t^{(i)} \simeq E(x_t | x_{t-1}^{(i)}, \theta^{(i)}, y_t)$ and $v_t^{(i)} \simeq Var(x_t | x_{t-1}^{(i)}, \theta^{(i)}, y_t)$.

SPF Draw sample $x_t^{(i)}$ by normal proposal $q(\cdot, x_{t-1}^{(i)}, \theta^{(i)}, y_t) \sim N(m_t^{(i)}, v_t^{(i)})$.

Normalization Assign weight

$$w^{(i)} = \frac{p(y_t | x_t^{(i)}, \theta^{(i)}) p(x_t^{(i)} | x_{t-1}^{(i)}, \theta^{(i)})}{q(x_t^{(i)}, x_{t-1}^{(i)}, \theta^{(i)}, y_t)}.$$

Resampling Draw $x_t^{(i)}$ by the multinomial distribution.

KF Set $\bar{x}_t = \frac{1}{N} \sum_{i=1}^N x_t^{(i)}$. Compute e_t and v_t by standard KF.

3.3 The variational Bayes particle filter

In previous chapters, we discussed the powerful algorithms of variational Bayes and particle filters, respectively. Both methods extend the reach of sequential Bayesian to address more general state-space models with unknown parameters. At the same time, both methods have apparent limitations, which leave room for further improvements.

Variational Bayes (VB) is essentially a semi-analytical method, in that it trades the high-dimensional intractability for low-dimensional tractability, at the cost of some loss of accuracy. It works well for complex models with multiple unknown parameters and/or states. From a theoretical point of view, VB runs parallel to the standard EM algorithm. Optimization is carried out on a low marginal once at a time, keeping other dimensions fixed, and approximation is achieved through iterations. In order to formulate the IVB cycles, it is crucial that the original model specification has certain structural form such as separability and conjugacy. These assumptions put a virtual limit for the practical use of VB methods. Especially in the joint identification of parameter and states, it is normal to have such conditions satisfied only along certain dimensions. Highly nonlinearity can cause VB to break down, where simulation seems to be the only feasible alternative.

Particle filters are specifically designed to deal with the generic nonlinear models. It does not impose any restrictions on the model, and can at least in theory be applied to any dynamics. The particles constitute the empirical distribution of the posterior, which is conjugate prior to any channel. This general flexibility of particle representation comes at a price. Instead of using a few sufficient statistics, here particle filters require a full description of the posterior through a relatively large particle set, and each particle needs to be propagated. This is a

severe bottleneck for multidimensional models. It is very challenging to sample a N -dimensional space efficiently, where the number of necessary particles usually grow exponentially with N .

There has been some work in the particle filter literature to incorporate some analytics to enhance accuracy while keeping the computational cost at moderate level. For example, Rao-Blackwellized particle filters (RB-PF) assume a partition of state space (x^1, x^2) , such that $f(x_t^1 | x_t^2, y^t)$ is a conjugate channel and thus can be propagated analytically. On a separate line of development, Storvik introduced particle filters for parameter estimation where the parameter marginal $f(\theta | x_t, y^t)$ admits a conjugate channel. Both can be seen as special cases of a more generalized combination of VB and PF.

In this chapter, we introduce the Variational Bayes Particle Filter (VB-PF). For a multidimensional, nonlinear model, it uses the usual VB partitioning to break up dimensions into conditionally independent VB-marginals. For each VB-marginal, if the corresponding subchannel is conjugate, then proceed with analytics to propagate conjugate pair by updating sufficient statistics. Otherwise, use particle representation for the individual dimension. Different VB-marginals are connected through VB-moments, which can be calculated using particle approximation. This is still an iterative calculation scheme.

In this manner, traditional VB can be seen as a special case of VB-PF where all subchannels are analytically tractable. On the other extreme end, traditional PF is also a special case of VB-PF where no partition is involved, and the entire joint posterior is approximated. Given K unknown parameters $(\theta^1, \dots, \theta^K)$, PF will need N particles $(\theta^{1(i)}, \dots, \theta^{K(i)})$ for the joint distribution. This usually requires a huge N to have any reasonable approximation. VB-PF would instead approximate

each dimension by N particles, such that dimension k corresponds to the N 1-dim samples $\theta^{k(i)}$.

3.3.1 VB-PF formulation

Consider a discrete jump model:

$$X_t = aX_{t-1} + \sigma_X \epsilon_t^X \quad (3.4)$$

$$Y_t = bX_t + \sigma_Y \epsilon_t^Y + J_t \quad (3.5)$$

$$J_t = \gamma_G G_t + \sigma_G \sqrt{G_t} \epsilon_t^J \quad (3.6)$$

$$G_t \sim \text{Gamma}(\alpha_G, \beta_G) \quad (3.7)$$

For simplicity, assume the unknown parameters are $\Theta = (\gamma_G, \sigma_G)$. The persistent state x_t and latent states J_t, G_t are also unknown, making the overall estimation a high-dimension problem. We use VB-PF for the online estimation of parameter Θ .

First, decompose into VB-marginals

$$f(\gamma_G, \sigma_G, X_t, X_{t-1}, J_t, G_t | y^t) = \tilde{f}(\gamma_G, \sigma_G | y^t) \tilde{f}(x_t, x_{t-1} | y^t) \tilde{f}(J_t, G_t | y^t)$$

First term:

$$\begin{aligned} \tilde{f}(\theta | y^t) &= \exp E_{\hat{x}_t, \hat{x}_{t-1}, \hat{J}_t, \hat{G}_t} \log f(\theta, x_t, x_{t-1}, J_t, G_t | y^t) \\ &= \exp E_{\hat{x}_t, \hat{x}_{t-1}, \hat{J}_t, \hat{G}_t} \log \tilde{f}(\theta | y^{t-1}) \tilde{f}(x_{t-1} | y^{t-1}) f(G_t) f(J_t | G_t, \theta) \\ &\quad \cdot f(x_t | x_{t-1}) f(y_t | x_t, J_t) \\ &= \tilde{f}(\theta | y^{t-1}) \exp E_{\hat{x}_t, \hat{x}_{t-1}, \hat{J}_t, \hat{G}_t} \log f(J_t | G_t, \theta) \\ &= \tilde{f}(\theta | y^{t-1}) \exp E_{\hat{x}_t, \hat{x}_{t-1}, \hat{J}_t, \hat{G}_t} \log \left(\frac{1}{\sigma_G^2} \right)^{\frac{1}{2}} e^{-\frac{(J_t - \gamma_G G_t)^2}{2\sigma_G^2 G_t}}. \end{aligned}$$

simplifies to:

$$\tilde{f}(\gamma_G, \sigma_G | y^t) = \tilde{f}(\gamma_G, \sigma_G | y^{t-1}) \left(\frac{1}{\sigma_G^2}\right)^{\frac{1}{2}} e^{-\frac{1}{2\sigma_G^2}(\gamma_G^2 G_t - 2J_t \gamma_G + J_t^2 \frac{1}{G_t})}.$$

This subchannel has a conjugate link, so we naturally choose the normal-inverse-gamma conjugate prior:

$$\tilde{f}(\gamma_G, \sigma_G | y^{t-1}) \sim \mathcal{NIG}(A_{t-1}^J, B_{t-1}^J, \lambda_{t-1}^J, C_{t-1}^J),$$

where

$$\begin{aligned} A_t^J &= A_{t-1}^J + \frac{1}{2} \\ C_t^J &= C_{t-1}^J + \widehat{G}_t \\ \lambda_t^J &= \frac{1}{C_t^J} (C_{t-1}^J \lambda_{t-1}^J - \frac{1}{2} (-2\widehat{J}_t)) \\ B_t^J &= B_{t-1}^J + \frac{1}{2} ((\lambda_{t-1}^J)^2 C_{t-1}^J + \frac{\widehat{J}_t^2}{G_t} - (\lambda_t^J)^2 C_t^J). \end{aligned}$$

The necessary VB-moments are \widehat{G}_t , \widehat{J}_t , and $\frac{\widehat{J}_t^2}{G_t}$.

Second term:

$$\begin{aligned} \tilde{f}(J_t, G_t | y^t) &= \exp E_{\widehat{x}_t, \widehat{x}_{t-1}, \widehat{\gamma}_G, \widehat{\sigma}_G} \log f(J_t, G_t | \theta) f(y_t | x_t, J_t) \\ &= \exp E_{\widehat{x}_t, \widehat{x}_{t-1}, \widehat{\gamma}_G, \widehat{\sigma}_G} \log G_t^{\alpha_G - 1} e^{-\beta_G G_t} e^{-\frac{(J_t - \gamma_G G_t)^2}{2\sigma_G^2 G_t}} e^{-\frac{(y_t - b x_t - J_t)^2}{2\sigma_Y^2}} \\ &= G_t^{\alpha_G - 1} e^{-(\beta_G + \frac{1}{2} \frac{\widehat{\gamma}_G^2}{\sigma_G^2}) G_t} \cdot e^{-\frac{1}{2\sigma_Y^2} J_t^2 + (\frac{\widehat{\gamma}_G}{\sigma_G} + \frac{y_t - b \widehat{x}_t}{\sigma_Y^2}) J_t} \cdot e^{-\frac{1}{2\sigma_G^2} \frac{J_t^2}{G_t}}. \end{aligned}$$

We can separate out the two states G_t and J_t

$$G_t \sim \text{Gamma}(\alpha_G, \beta_G + \frac{1}{2} \frac{\widehat{\gamma}_G^2}{\sigma_G^2}) \quad (3.8)$$

$$J_t | G_t \sim N\left(\frac{\frac{\widehat{\gamma}_G}{\sigma_G} + \frac{y_t - b \widehat{x}_t}{\sigma_Y^2}}{\frac{1}{\sigma_Y^2} + \frac{1}{\sigma_G^2} \frac{1}{G_t}}, \frac{1}{\frac{1}{\sigma_Y^2} + \frac{1}{\sigma_G^2} \frac{1}{G_t}}\right). \quad (3.9)$$

In light of the first step, where we need VB-marginals involving $\widehat{\frac{J_t^2}{G_t}}$, it is analytically intractable to use conjugate pairs. Instead, it is straightforward to simulate the joint distribution of G_t, J_t and represent them as particles. At this stage, we generate N pairs of $(G^{(i)}, J^{(i)})$, and can easily calculate $\widehat{\frac{J_t^2}{G_t}} = \frac{1}{N} \sum_{i=1}^N (J^{(i)})^2 \frac{1}{G^{(i)}}$.

The necessary VB-moments are \widehat{x}_t , $\widehat{\frac{1}{\sigma_G^2}}$, $\widehat{\frac{\gamma_G}{\sigma_G^2}}$ and $\widehat{\frac{\gamma_G^2}{\sigma_G^2}}$. The parameter VB-moments come exactly from step 1:

$$\widehat{\frac{1}{\sigma_G^2}} = \frac{A_t^J}{B_t^J} \quad (3.10)$$

$$\widehat{\frac{\gamma_G}{\sigma_G^2}} = \lambda_t^J \frac{A_t^J}{B_t^J} \quad (3.11)$$

$$\widehat{\frac{\gamma_G^2}{\sigma_G^2}} = (\lambda_t^J)^2 \frac{A_t^J}{B_t^J} + \frac{1}{C_t^J}. \quad (3.12)$$

3.4 Joint Parameter-State Estimation for state-space models

3.4.1 Discrete-time State Space models

Discrete-time state space models, or in its simplified version, hidden Markov models (HMM), are a convenient means for studying dynamics systems. Define a discrete time horizon $t \in \{0, 1, \dots, T\}$. Given parameters θ , a state space model consists of two basic components: state x_t and observation y_t . The dynamics are given by two equations, the state transition equation and the observation equation:

$$x_t = f(x_{t-1}, v_t, \theta) \quad (3.13)$$

$$y_t = g(x_t, w_t, \theta) \quad (3.14)$$

where we have:

- Transition: $p(x_t|x_{t-1}, \theta)$ is the Markov link, defined by function f . It is also called the prior or state equation. The state x_t constitute a Markov chain of first order. v_t is the random effect due to transition noise.
- Likelihood: $p(y_t|x_t, \theta)$ is the observation likelihood, defined by function g . w_t is the random effect due to observation noise.

Together with a prior distribution for the parameters $p(\theta)$ and an initial distribution of the state $p(x_0|\theta)$, these two equations completely characterize the dynamics of the states and the observations. The Markovian structure in the HMM model has nice properties that leads to simplifications in subsequent calculations.

3.4.2 Continuous-time Diffusion models

A large class of models can be cast as a state space model. We first consider a generic diffusion model driven by d -dimensional Brownian motion. The stochastic differential equation is:

$$dY_t = \mu(Y_t, \Theta)dt + \beta^{\frac{1}{2}}(Y_t, \Theta)dW_t \quad (3.15)$$

where Y_t is a d -dimensional continuous-time process, Θ is the vector of unknown parameters. $\mu(Y_t, \Theta)$ and $\beta(Y_t, \Theta)$ are the drift and volatility components of Y_t , with dimensions d and $d \times d$, respectively. Usually, Y_t consists of both observable and unobservable components. We write $Y_t = (X_t, Z_t)'$, where X_t is the observable part with dimension d_1 , Z_t corresponds to the unobservable part with dimension d_2 . $d = d_1 + d_2$.

Although the underlying dynamics is modeled in continuous-time, usually we only have discrete observations. Assume X is observed at equi-distant times

$\{t_0, t_1, \dots, t_T\}$. The observation interval is $\Delta_o = t_j - t_{j-1}, j \in \{1, \dots, T\}$. The link between discrete-time and continuous-time models is the Euler discretization. Euler method requires a step size of Δ_e sufficiently small in order to maintain a good approximation, so $\Delta_e < \Delta_o$. Therefore, it is necessary to introduce additional latent states plugged in-between every two successive observation points. It is equivalent to artificially increase the sampling frequency, only this time we don't know the intermediate observations, but have to estimate them as if they are latent states, too. To be precise, we set $\Delta_e = \Delta_o/m$ for some fixed chosen integer m , and insert $m - 1$ points within each observation interval. The discretized time-series dynamics become:

$$\Delta Y_t = \mu(Y_t, \Theta)\Delta t + \beta^{\frac{1}{2}}(Y_t, \Theta)\Delta W_t \quad (3.16)$$

with $\Delta t = \Delta_e$ and $\Delta W_t \sim N(0, I_d \Delta t)$ is a d -dimensional standard Brownian motion.

Given times t_j and t_{j+1} , we have observations X_j and X_{j+1} . Within the interval (t_j, t_{j+1}) , we have $m - 1$ unknown vectors Y^1, Y^2, \dots, Y^{m-1} , where for notational simplicity, we have suppressed subscripts j . Also, at time t_{j+1} , we only have the X -component of Y_{j+1} available, so Z_{j+1} is included in the latent states. So, if we define $L_{j+1} = (Y^1, Y^2, \dots, Y^{m-1}, Z_{j+1})$, then (L_j, X_j) becomes a discrete-time state space model. This is what we call a Partially-Observable Discretely-Observed State Space model. We will use this formulation throughout the thesis.

3.5 Simulation Results

3.5.1 The classic Kalman filter

We start with the simplest example for sequential Bayesian filtering, the classical Kalman filter. For notational simplicity, we only give the 1-dim scalar case. Vector-based Kalman filters can be similarly established using matrix algebra. Kalman filter works for the linear observation model with additive Gaussian noise. Assuming all parameters are known, we have:

$$x_t = ax_{t-1} + \sigma_x \epsilon_t^x \quad (3.17)$$

$$y_t = bx_t + \sigma_y \epsilon_t^y \quad (3.18)$$

where $\epsilon_t^x, \epsilon_t^y$ are iid standard normals. Initial state distribution $x_0 \sim N(\mu_0, \sigma_0^2)$. The linear Gaussian observation channel belongs to the Conjugate-Exponential family, with normal conjugate pairs. So, all posterior distributions can be made normal. Denote $p(x_t|y^t) \sim N(\mu_t, \sigma_t)$. We have, from basic Bayesian theory:

$$\begin{aligned} \text{prior} & : X \sim N(\mu, \sigma^2) \\ \text{observation} & : Y|X \sim N(aX, \tilde{\sigma}^2) \\ \text{posterior} & : X|Y \sim N\left(\frac{\frac{\mu}{\sigma^2} + \frac{aY}{\tilde{\sigma}^2}}{\frac{1}{\sigma^2} + \frac{a^2}{\tilde{\sigma}^2}}, \frac{1}{\frac{1}{\sigma^2} + \frac{a^2}{\tilde{\sigma}^2}}\right). \end{aligned}$$

Here, we have exactly this link:

$$\begin{aligned} \text{prior} & : p(x_t|y^{t-1}) \sim N(a\mu_{t-1}, a^2\sigma_{t-1}^2 + \sigma_x^2) \\ \text{observation} & : p(y_t|x_t) \sim N(bx_t, \sigma_y^2) \\ \text{posterior} & : p(x_t|y^t) \sim N(\mu_t, \sigma_t^2) \end{aligned}$$

where

$$\mu_t = \frac{\frac{a\mu_{t-1}}{a^2\sigma_{t-1}^2 + \sigma_x^2} + \frac{by_t}{\sigma_y^2}}{\frac{1}{a^2\sigma_{t-1}^2 + \sigma_x^2} + \frac{b^2}{\sigma_y^2}}, \sigma_t^2 = \frac{1}{\frac{1}{a^2\sigma_{t-1}^2 + \sigma_x^2} + \frac{b^2}{\sigma_y^2}}. \quad (3.19)$$

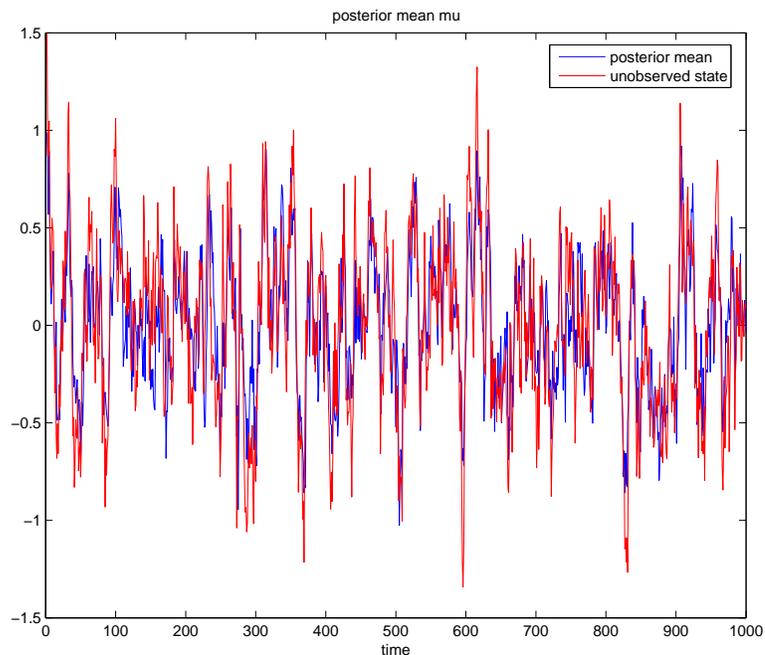


Figure 3.1: 1-dim state filtering with Kalman Filter, all known parameters

The chosen parameters are $a = 0.8$, $b = 0.7$, $\sigma_x = 0.25$, $\sigma_y = 0.3$; initial value $\mu_0 = 1$, $\sigma_0^2 = 0.05$. The simulation results are in figures (3.1) and (3.2):

Clearly, here the equation for σ_t^2 is quadratic, and has a fixed point. Recursively solving for σ_t^2 from σ_{t-1}^2 will converge to the unique fixed point solution very quickly. On the other hand, posterior mean μ_t has significant variance, which comes from the actual fluctuations of the persistent state x_t and observed through y_t . μ_t can very accurately track x_t , thus showing the high efficiency of Kalman filtering.

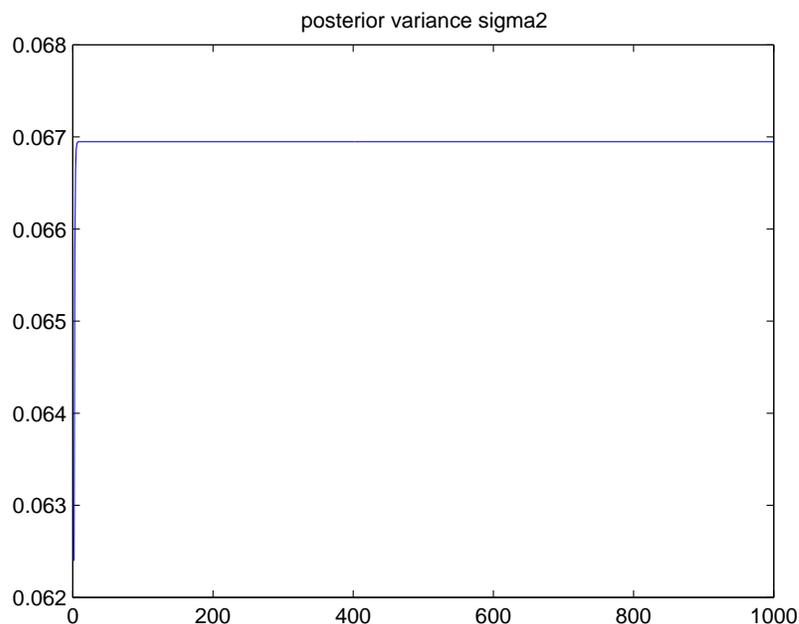


Figure 3.2: Fast convergence for posterior variance with Kalman Filter, all known parameters

3.5.2 Particle filter simulations

Unscented PF

We look at a simple application of USPF to a 1-dim HMM. The discrete non-linear model is:

$$\begin{cases} x_t = x_{t-1} + \kappa(\theta - x_{t-1}) + \sigma_v \sqrt{x_{t-1}} \varepsilon_t^x \\ y_t = \mu + x_t^2 + \sigma_y \varepsilon_t^y \end{cases} \quad \text{model 1.0}$$

where $\varepsilon_t^y, \varepsilon_t^x$ are correlated $N(0, 1)$ noise with coefficient ρ . Here the parameters are $\kappa = 0.05$, $\theta = 2$, $\sigma_v = 0.1$, $\sigma_y = 0.03$, $\rho = 0.3$, $\mu = 0.15$. We see from (3.3) that the state filtering is quite accurate.

If some parameter is unknown, then the state filtering quality gets poorer, see (3.4).

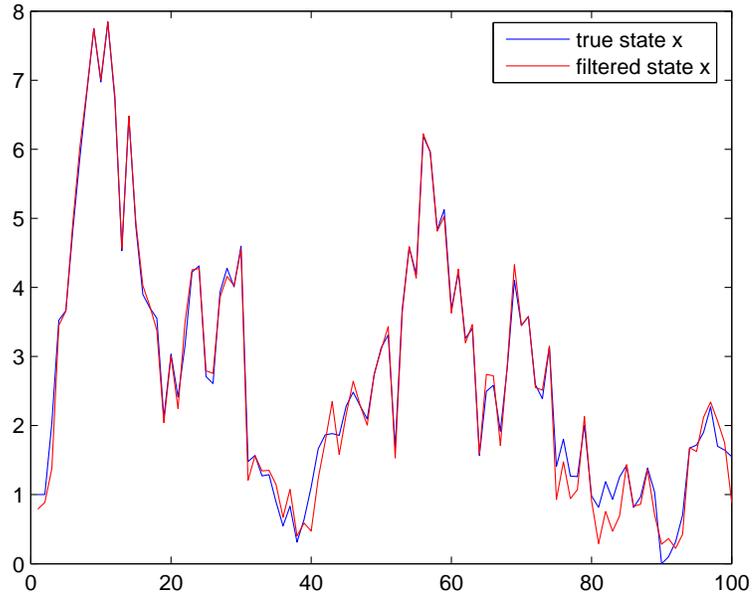


Figure 3.3: 1-dim state filtering with USPF, exact parameter

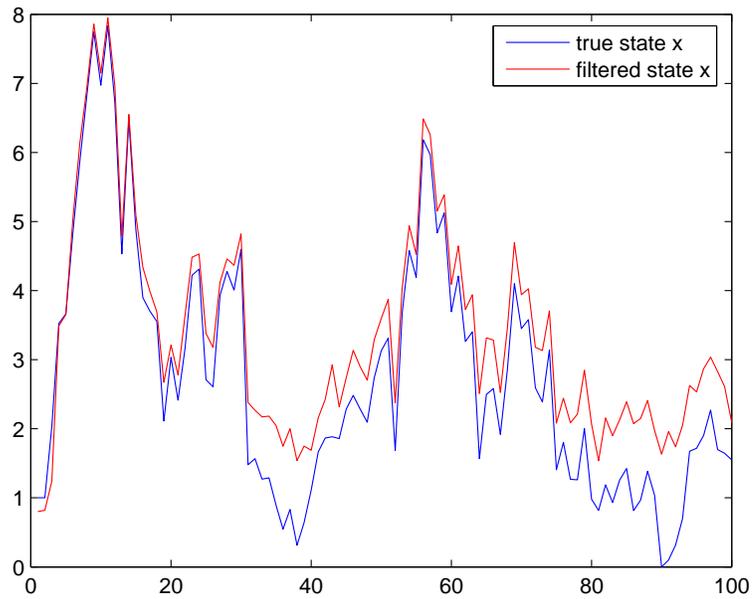


Figure 3.4: 1-dim state filtering with USPF, unknown parameter μ

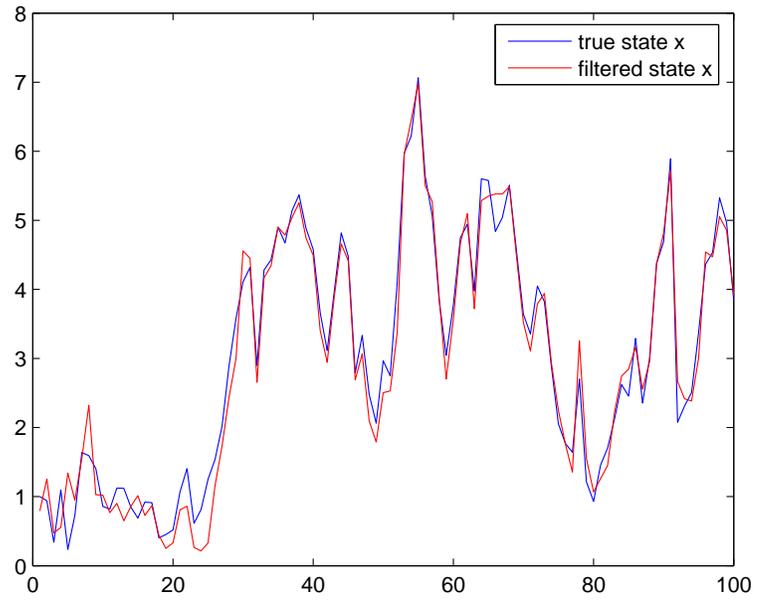


Figure 3.5: 2-dim state filtering with USPF, exact parameter

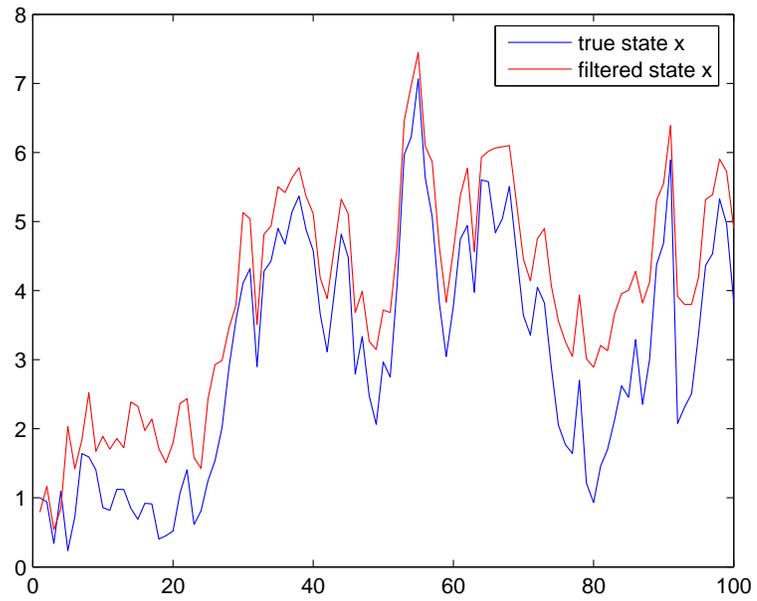


Figure 3.6: 2-dim state filtering with USPF, unknown parameter mu

Modified RBPF

We look at a simple application of MRBPF to a 1-dim HMM with 1 unknown parameter. The discrete nonlinear model is:

$$\begin{cases} x_t = x_{t-1} + \kappa(\theta - x_{t-1}) + \sigma_x \sqrt{|x_{t-1}|} \varepsilon_t^x \\ y_t = \mu + \sigma_y \sqrt{|x_t|} \varepsilon_t^y \end{cases} \quad \text{model 2.0}$$

where $\varepsilon_t^y, \varepsilon_t^x$ are independent $N(0, 1)$ noise. Here the parameters are $\kappa = 0.05$, $\theta = 2$, $\sigma_x = 0.2$, $\sigma_y = 0.1$, and we will estimate μ . We perform 100 simulations. Each simulation uses the exact true value $\mu = 0.15$ for a single sample path $T = 100$. The priors are $p(x) \sim N(\theta = 2, 1)$ and $p(\mu) \sim N(0, 0.1)$. We treat μ as a static state process, with $\sigma_\varepsilon = 10^{-6}$, and $\varepsilon_t^\mu \sim N(0, 1)$. In this way, x follows its own state process, and the posterior of x is estimated by a N -particle sample. μ is conditionally linear Gaussian. The posterior of μ is estimated by a standard Kalman filter.

$$\mu_t = \mu_{t-1} + \sigma_\varepsilon \varepsilon_t^\mu.$$

We see from figure (3.7) that the parameter estimation is accurate. For this particular instance, the mean of 100 μ estimates is 0.1506, the standard deviation of the estimates is 0.0140. The problem with this model is: the state estimation is quite bad.

Consider a different model,

$$\begin{cases} x_t = x_{t-1} + \kappa(\theta - x_{t-1}) + \sigma_x \sqrt{|x_{t-1}|} \varepsilon_t^x \\ y_t = \mu + x_t^2 + \sigma_y \sqrt{|x_t|} \varepsilon_t^y \end{cases} \quad \text{model 2.1}$$

Now estimate κ . where $\varepsilon_t^y, \varepsilon_t^x$ are independent $N(0, 1)$ noise. Here the parameters are $\mu = 0.05$, $\theta = 2$, $\sigma_x = 0.2$, $\sigma_y = 0.5$, and we will estimate κ . We perform 100 simulations. Each simulation uses the exact true value $\kappa = 0.25$ for a single

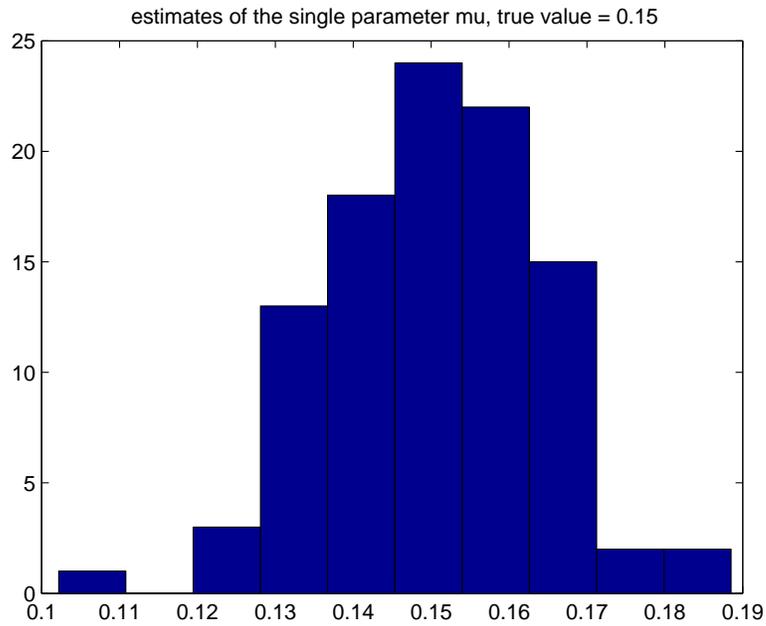


Figure 3.7: 1-dim state filtering with MRBPF, unknown parameter mu

sample path $T = 100$. The priors are $p(x) \sim N(\theta = 2, 1)$ and $p(\kappa) \sim N(0.5, 0.1)$.

We treat

$$\kappa_t = \kappa_{t-1} + \sigma_\varepsilon \varepsilon_t^\kappa$$

We see from figure (3.8) that the parameter estimation is moderately acceptable.

For this particular instance, the mean of 100 κ estimates is 0.2817, the standard deviation of the estimates is 0.0802. Here the state filtering becomes moderately

reasonable:

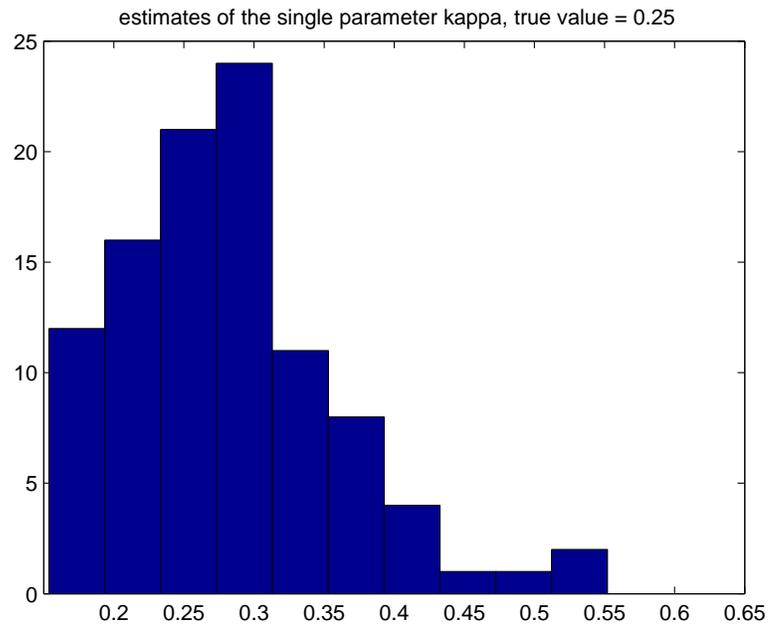


Figure 3.8: 1-dim state filtering with MRBPF, unknown parameter kappa

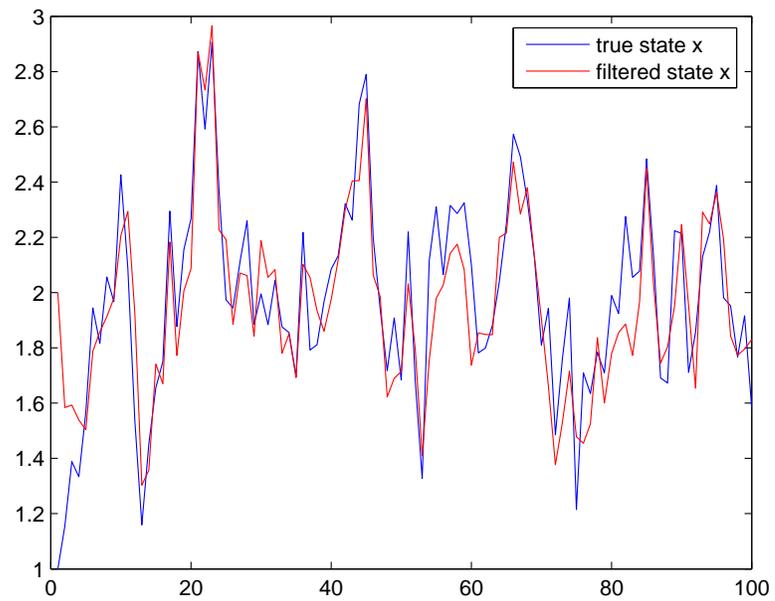


Figure 3.9: 1-dim state filtering with MRBPF, exact parameter

3.5.3 VB-PF simulation

Kalman Filter with single unknown parameter

Next, we move into the first non-trivial case: Recursive estimation of parameters in the presence of unknown state. We start with the modified Kalman filter, with a single unknown parameter b as in the observation model. So, $\theta = b$. We cannot directly apply Kalman filtering, due to the presence of unknown b . Instead, we use the standard Variational Bayes to jointly identify θ, x_t sequentially.

The model is the same as (3.17) and (3.18), while the derivation framework is fundamentally different and much more involved.

Firstly, according to the VB algorithm, we impose conditional independence on the parameter $\theta = b$ and state x_t to get:

$$f(b, x_t, x_{t-1}|y^t) \approx \tilde{f}(b|y^t)\tilde{f}(x_t, x_{t-1}|y^t).$$

The first term:

$$\tilde{f}(b|y^t) = \tilde{f}(b|y^{t-1}) \exp E_{\widehat{x}_t, \widehat{x}_{t-1}} \log f(x_t, y_t|x_{t-1}, b),$$

where

$$\log f(x_t, y_t|x_{t-1}, b) = -\frac{x_t^2 - 2ax_t x_{t-1} + a^2 x_{t-1}^2}{2\sigma_x^2} - \frac{y_t^2 - 2bx_t y_t + b^2 x_t^2}{2\sigma_y^2}.$$

Upon taking expectation with respect to $\widehat{x}_t, \widehat{x}_{t-1}$, the above is simplified to:

$$\exp E_{\widehat{x}_t, \widehat{x}_{t-1}} \log f(x_t, y_t|x_{t-1}, b) = \exp\left(-\frac{b^2 \widehat{x}_t^2 - 2b\widehat{x}_t y_t}{2\sigma_y^2}\right).$$

This is a normal channel that admits a normal conjugate pair, so we can set the conjugate prior as $\tilde{f}(b|y^{t-1}) \sim N(\mu_{t-1}^b, \sigma_{t-1}^{b2})$. The recursive updating for the

posterior mean and variance are:

$$\mu_t^b = \frac{\frac{\mu_{t-1}^b}{\sigma_{t-1}^{b^2}} + \frac{\widehat{x}_t y_t}{\sigma_y^2}}{\frac{1}{\sigma_{t-1}^{b^2}} + \frac{\widehat{x}_t^2}{\sigma_y^2}}, \sigma_t^{b^2} = \frac{1}{\frac{1}{\sigma_{t-1}^{b^2}} + \frac{\widehat{x}_t^2}{\sigma_y^2}}. \quad (3.20)$$

Compare (3.20) with the known parameter case (3.19), it is clear that here the posterior updating formula is time-dependent, and explicitly depends on the estimated moments \widehat{x}_t^2 . We can expect $\sigma_t^{b^2}$ to converge much more slowly.

At this step, the necessary VB-moments are \widehat{x}_t^2 and \widehat{x}_t .

The second term:

$$\tilde{f}(x_t, x_{t-1}|y^t) = \tilde{f}(x_{t-1}|y^{t-1}) \exp E_{\widehat{b}} \log f(x_t, y_t|x_{t-1}, b)$$

where, after simplification:

$$\exp E_{\widehat{b}} \log f(x_t, y_t|x_{t-1}, b) = \exp \left(- \left(\frac{x_t^2 - 2ax_t x_{t-1} + a^2 x_{t-1}^2}{2\sigma_x^2} + \frac{\widehat{b}^2 x_t^2 - 2\widehat{b}y_t x_t}{2\sigma_y^2} \right) \right).$$

This is again a normal link, but (x_t, x_{t-1}) follows a bivariate normal distribution.

Denote $\tilde{f}(x_{t-1}|y^{t-1}) \sim N(\mu_{t-1}^x, \sigma_{t-1}^{x^2}) = \exp \left(- \frac{(x_{t-1} - \mu_{t-1}^x)^2}{2\sigma_{t-1}^{x^2}} \right)$, we have

$$\tilde{f}(x_t, x_{t-1}|y^t) = \exp E_{\widehat{b}} \log f(x_t, y_t|x_{t-1}, b) \tilde{f}(x_{t-1}|y^{t-1}) \quad (3.21)$$

$$= e^{-\left(\frac{1}{2} X^T C X + \lambda^T X\right)} \quad (3.22)$$

$$\equiv e^{-\frac{1}{2} (X - \mu)^T \Sigma^{-1} (X - \mu)} \quad (3.23)$$

where we define:

$$C = \begin{pmatrix} \frac{1}{\sigma_x^2} + \frac{\widehat{b}^2}{\sigma_y^2}, & -\frac{a}{\sigma_x^2} \\ -\frac{a}{\sigma_x^2}, & \frac{a^2}{\sigma_x^2} + \frac{1}{\sigma_{t-1}^{x^2}} \end{pmatrix}, X = \begin{pmatrix} x_t \\ x_{t-1} \end{pmatrix}, \lambda = \begin{pmatrix} -\frac{\widehat{b}y_t}{\sigma_y^2} \\ -\frac{\mu_{t-1}^x}{\sigma_{t-1}^{x^2}} \end{pmatrix}. \quad (3.24)$$

Solve for the posterior mean and variance for (x_t, x_{t-1}) :

$$\mu = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} = -C^{-1}\lambda \quad (3.25)$$

$$\Sigma = \begin{pmatrix} \Sigma_{11}, \Sigma_{12} \\ \Sigma_{21}, \Sigma_{22} \end{pmatrix} = C^{-1}. \quad (3.26)$$

Then, we can get the necessary VB-moments as:

$$\hat{x}_t = \mu_1 \quad (3.27)$$

$$\hat{x}_t^2 = \mu_1^2 + \Sigma_{11}. \quad (3.28)$$

After N iterations of the IVB cycle, only $\Sigma_{11} = \sigma_t^{x2}$ and $\mu_1 = \mu_t^x$ need to be stored for $\tilde{f}(x_t|y^t)$ and propagated forward.

This subchannel requires VB-moments \hat{b} and \hat{b}^2 , which can be computed since b has posterior normal distribution:

$$\hat{b} = \mu_t^b \quad (3.29)$$

$$\hat{b}^2 = \mu_t^{b2} + \sigma_t^{b2}. \quad (3.30)$$

Now we can outline the complete VB estimation procedure for this model. At time $t - 1$, we already have $\tilde{f}(x_{t-1}|y^{t-1}) \sim N(\mu_{t-1}^x, \sigma_{t-1}^{x2})$ and $\tilde{f}(b|y^{t-1}) \sim N(\mu_{t-1}^b, \sigma_{t-1}^{b2})$. For time t , set initial entry-point condition μ_t^b, σ_t^{b2} . Then iterate N steps:

1. Compute \hat{b} and \hat{b}^2 , using (3.29) and (3.30);
2. Compute C and λ by (3.24), Compute μ and Σ by (3.25) and (3.26);
3. Compute \hat{x}_t and \hat{x}_t^2 by (3.27) and (3.28);
4. Compute μ_t^b, σ_t^{b2} by (3.20).

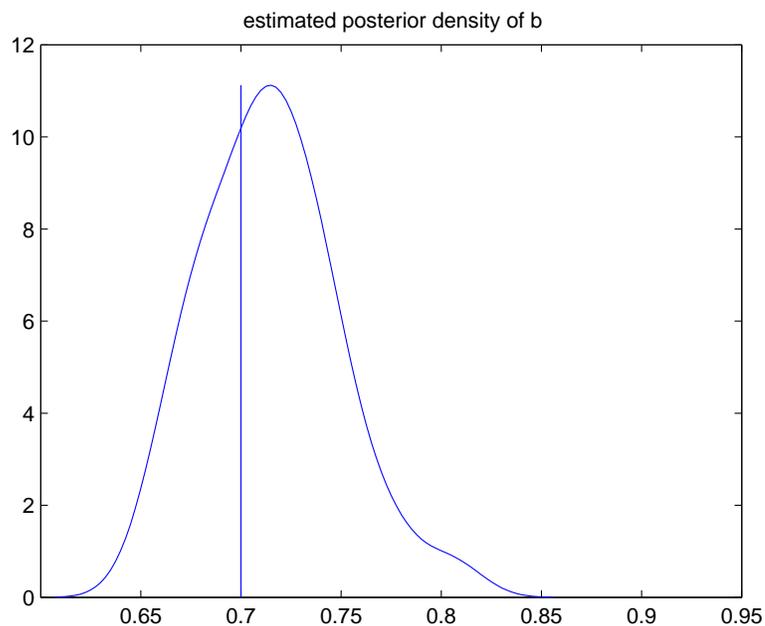


Figure 3.10: Kalman Filter with single unknown parameter: posterior distribution of b

The chosen parameters are $a = 0.8$, $b = 0.7$, $\sigma_x = 0.25$, $\sigma_y = 0.3$; initial value $\mu_0 = 1$, $\sigma_0^2 = 0.05$. Here the VB algorithm sequentially estimates b . The simulation results are in Figures (3.10) and (3.11):

From the simulation, it can be shown that the posterior distribution approximately centers around the true value of $b = 0.7$, with standard deviation 0.03. The sequential estimates converge to the true value after around 1000 time steps. Within each time, the IVB takes 5 iterations, and the outputs μ_t^b , σ_t^{b2} , $\mu_t^x = \mu_1$ and $\sigma_t^{x2} = \Sigma_{11}$ propagates to the next time.

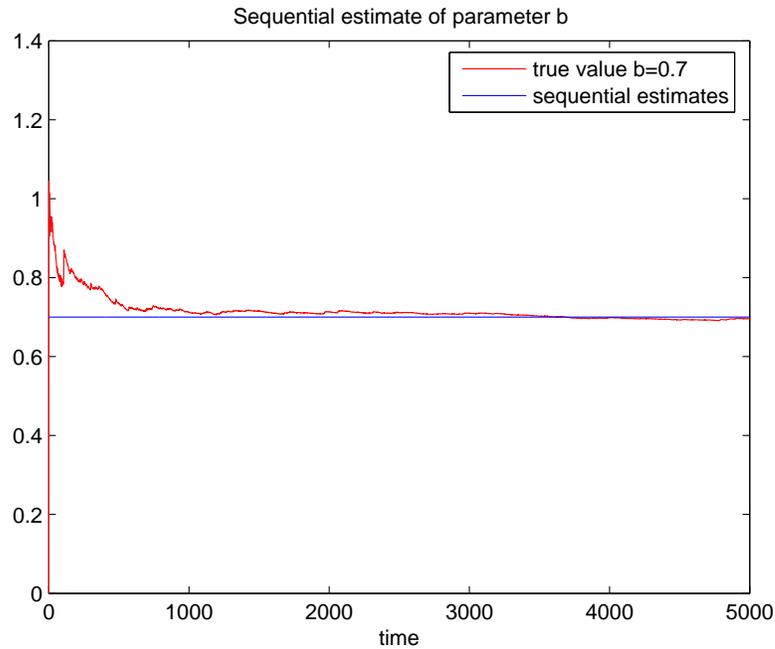


Figure 3.11: Kalman Filter with single unknown parameter: sequential estimates of b

Kalman Filter with multiple unknown parameters

As a further example for the VB framework, here we have the same model as (3.17) and (3.18), but allows for multiple unknown parameters. Specifically, set $\theta = (a, \sigma_x)$. Both unknowns are within the state dynamics, where the state itself must be inferred. This adds to the challenge of sequential identification.

Firstly, according to the VB algorithm, we impose conditional independence to get:

$$f(\theta, x_t, x_{t-1}|y^t) \approx \tilde{f}(\theta|y^t)\tilde{f}(x_t, x_{t-1}|y^t).$$

The first term:

$$\begin{aligned}
\tilde{f}(\theta|y^t) &= \tilde{f}(\theta|y^{t-1}) \exp E_{\widehat{x}_t, \widehat{x}_{t-1}} \log f(x_t, y_t|x_{t-1}, \theta) \\
&= \tilde{f}(\theta|y^{t-1}) \exp E_{\widehat{x}_t, \widehat{x}_{t-1}} \\
&\quad \left(-\frac{x_t^2 - 2ax_t x_{t-1} + a^2 x_{t-1}^2}{2\sigma_x^2} - \frac{y_t^2 - 2bx_t y_t + b^2 x_t^2}{2\sigma_y^2} - \frac{1}{2} \log \sigma_x^2 \right) \\
&= \tilde{f}(\theta|y^{t-1}) \left(\frac{1}{\sigma_x^2} \right)^{\frac{1}{2}} e^{-\frac{a^2 \widehat{x}_{t-1}^2 - 2a \widehat{x}_t \widehat{x}_{t-1} + \widehat{x}_t^2}{2\sigma_x^2}}.
\end{aligned}$$

We resort to the Normal-Inverse-Gamma conjugate prior, setting $\tilde{f}(a, \sigma_x^2|y^{t-1}) \sim \mathcal{NIG}(A_{t-1}, B_{t-1}, \lambda_{t-1}, C_{t-1})$, so that $\tilde{f}(a, \sigma_x^2|y^t) \sim \mathcal{NIG}(A_t, B_t, \lambda_t, C_t)$ The updating scheme is:

$$A_t = A_{t-1} + \frac{1}{2}; \quad (3.31)$$

$$C_t = C_{t-1} + \widehat{x}_{t-1}^2; \quad (3.32)$$

$$\lambda_t = \frac{1}{C_t} (C_{t-1} \lambda_{t-1} - \frac{1}{2} (-2 \widehat{x}_t \widehat{x}_{t-1})); \quad (3.33)$$

$$B_t = B_{t-1} + \frac{1}{2} (\lambda_{t-1}^2 C_{t-1} + \widehat{x}_t^2 - \lambda_t^2 C_t). \quad (3.34)$$

The necessary VB-moments are \widehat{x}_{t-1}^2 , $\widehat{x}_t \widehat{x}_{t-1}$ and \widehat{x}_t^2 .

Second term:

$$\begin{aligned}
\tilde{f}(x_t, x_{t-1}|y^t) &= \tilde{f}(x_{t-1}|y^{t-1}) \exp E_{\widehat{a}, \widehat{\sigma}_x^2} \log f(x_t, y_t|x_{t-1}, \theta) \\
&= \tilde{f}(x_{t-1}|y^{t-1}) \\
&\quad \cdot \exp \left\{ -\left(x_t^2 \left(\frac{1}{2\sigma_x^2} + \frac{b^2}{2\sigma_y^2} \right) - \frac{\widehat{a}}{\sigma_x^2} x_t x_{t-1} + \frac{\widehat{a}^2}{\sigma_x^2} x_{t-1}^2 - \frac{b}{\sigma_y^2} y_t x_t \right) \right\}.
\end{aligned}$$

Following similar derivation as in the single parameter case, we can get

$$\begin{aligned}
\tilde{f}(x_t, x_{t-1}|y^t) &= e^{-\left(\frac{1}{2} X^T C X + \lambda^T X\right)} \\
&\equiv e^{-\frac{1}{2} (X - \mu)^T \Sigma^{-1} (X - \mu)},
\end{aligned}$$

where:

$$C = \begin{pmatrix} \frac{\widehat{1}}{\sigma_x^2} + \frac{b^2}{\sigma_y^2}, & -\frac{\widehat{a}}{\sigma_x^2} \\ -\frac{\widehat{a}}{\sigma_x^2}, & \frac{\widehat{a}^2}{\sigma_x^2} + \frac{1}{\sigma_{t-1}^2} \end{pmatrix}, X = \begin{pmatrix} x_t \\ x_{t-1} \end{pmatrix}, \lambda = \begin{pmatrix} -\frac{by_t}{\sigma_y^2} \\ -\frac{\mu_{t-1}^x}{\sigma_{t-1}^2} \end{pmatrix}. \quad (3.35)$$

Solve for the posterior mean and variance for (x_t, x_{t-1}) :

$$\mu = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} = -C^{-1}\lambda. \quad (3.36)$$

$$\Sigma = \begin{pmatrix} \Sigma_{11}, \Sigma_{12} \\ \Sigma_{21}, \Sigma_{22} \end{pmatrix} = C^{-1}. \quad (3.37)$$

Then, the VB-moments involving x_t can be computed as:

$$\widehat{x_t^2} = \mu_1^2 + \Sigma_{11}. \quad (3.38)$$

$$\widehat{x_{t-1}^2} = \mu_2^2 + \Sigma_{22}. \quad (3.39)$$

$$\widehat{x_t x_{t-1}} = \mu_1 \mu_2 + \Sigma_{12}. \quad (3.40)$$

The necessary VB-moments involving the parameters can be computed from the \mathcal{NIG} distribution, $\tilde{f}(a, \sigma_x^2 | y^t) \sim \mathcal{NIG}(A_t, B_t, \lambda_t, C_t)$:

$$\frac{\widehat{1}}{\sigma_x^2} = \frac{A_t}{B_t}. \quad (3.41)$$

$$\frac{\widehat{a}}{\sigma_x^2} = \lambda_t \frac{A_t}{B_t}. \quad (3.42)$$

$$\frac{\widehat{a^2}}{\sigma_x^2} = \lambda_t^2 \frac{A_t}{B_t} + \frac{1}{C_t}. \quad (3.43)$$

At time $t - 1$, $\tilde{f}(x_{t-1} | y^{t-1}) \sim N(\mu_{t-1}^x, \sigma_{t-1}^{x2})$ and $\tilde{f}(a, \sigma_x^2 | y^{t-1}) \sim \mathcal{NIG}(A_{t-1}, B_{t-1}, \lambda_{t-1}, C_{t-1})$. For time t , set initial entry-point condition $(A_t, B_t, \lambda_t, C_t)$. Then iterate N steps:

1. Compute $\frac{\widehat{1}}{\sigma_x^2}$, $\frac{\widehat{a}}{\sigma_x^2}$ and $\frac{\widehat{a^2}}{\sigma_x^2}$, using (3.41), (3.42) and (3.43);

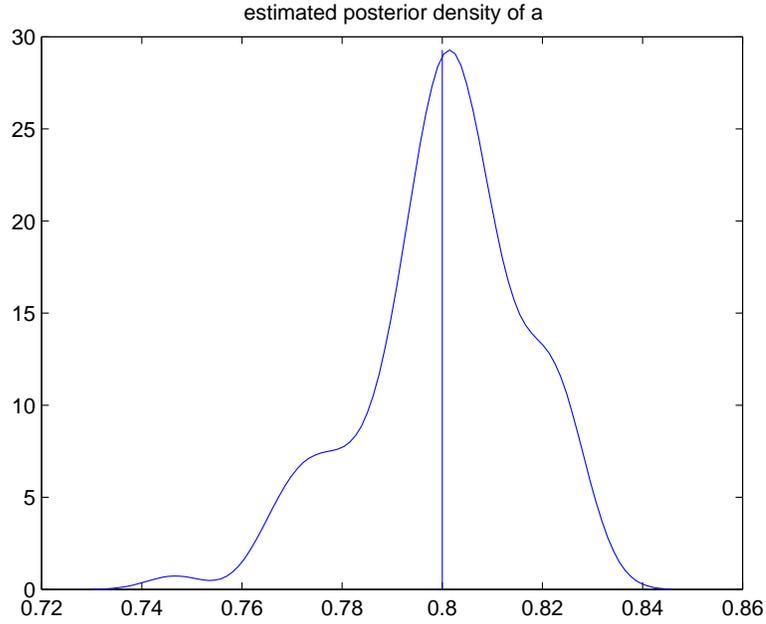


Figure 3.12: Kalman Filter with multiple unknown parameter: posterior distribution of a

2. Compute C and λ by (3.35), Compute μ and Σ by (3.36) and (3.37);
3. Compute \widehat{x}_t^2 , \widehat{x}_{t-1}^2 and $\widehat{x_t x_{t-1}}$ by (3.38), (3.39) and (3.40);
4. Compute $(A_t, B_t, \lambda_t, C_t)$ by (3.31) through (3.34).

The chosen parameters are $a = 0.8$, $b = 0.7$, $\sigma_x = 1.0$, $\sigma_y = 0.3$. Initial values are chosen as $\mu_1^x = 1$, $\sigma_1^{x^2} = 0.8$, $A_1 = 11$, $B_1 = 8$ (corresponding to initial estimate of $\sigma_x = \frac{B}{A-1} = 0.8$), $\lambda_1 = 0.9$, and $C_1 = 10$. The algorithm runs for $T = 2000$, with 100 sample paths. Within each time step, the IVB cycle has 10 iterations. Here the VB algorithm sequentially estimates (a, σ_x^2) . The simulation results are in figures (3.12) through (3.15):

Again, the VB algorithm captures the true values of the parameters very well, with standard deviations 0.016, 0.021, respectively, after 2000 time steps. The accuracy is mainly due to the analytical tractability of the partitioned marginals.

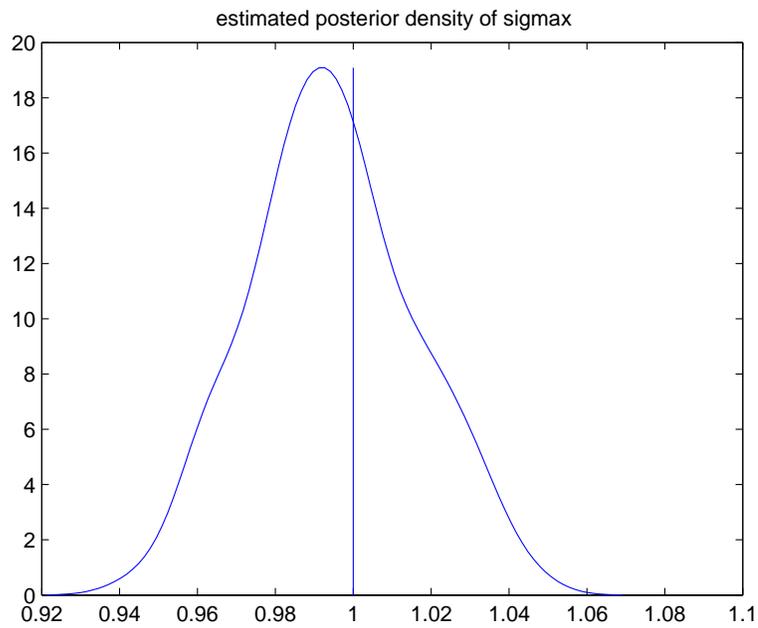


Figure 3.13: Kalman Filter with multiple unknown parameter: posterior estimates of sigmax

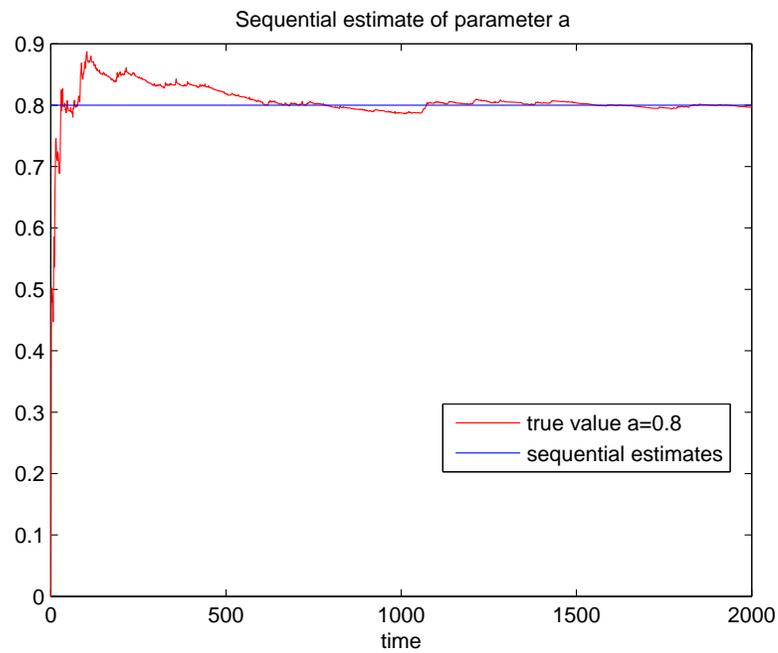


Figure 3.14: Kalman Filter with multiple unknown parameter: sequential estimate of a

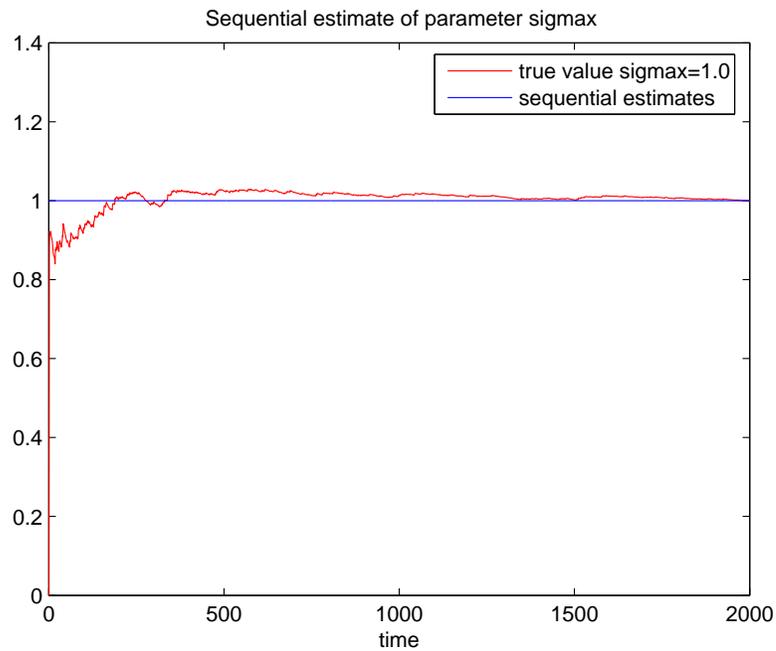


Figure 3.15: Kalman Filter with multiple unknown parameter: sequential estimate of sigmax

It is crucial that VB marginals have nice conjugate properties, so that the VB-moments are relatively easy to compute. In all these cases, there exist closed form equations for all the necessary VB-moments. This is fortunate, but not guaranteed for more general model dynamics.

DIMENSIONAL REDUCTION: MARKOVIAN PROJECTION FOR LEVY JUMP MODELS

4.1 Introduction

Local volatility models, introduced by [12], are widely used by financial practitioners to model the stochastic evolution of stock prices. These models provide a significant improvement over the classic Black-Scholes in that the volatility becomes a deterministic function of time and the underlying state. In this way, local volatility models can be calibrated to exactly match the volatility surface of vanilla options, without introducing a new random factor. Moreover, the famous Dupire formula makes calibration easy and fast.

At the same time, local volatility models have a couple of drawbacks that limit their practical use. There are two significant problems outstanding. Firstly, local volatility dynamics has the same number of factors as the underlying. This contradicts many empirical evidence, which suggest that there should be additional random effects. In some sense, local volatility models only captures the “static” distributional properties of the underlying, but not quite its dynamic behavior. Secondly, the local uncertainty is still conditionally normal. In reality, random fluctuations often exhibit fat tails and non-normality.

There has been tremendous efforts to address the above problems, with different angles of approach. The popular class of stochastic volatility models, as introduced by [23] is a natural extension to multi-factor modeling. More recently, [5] extends local volatility to local Levy models, and obtained parallel results for

the general jump setting. The local volatility function is replaced by its jump equivalent counterpart: the local speed function. Here we combine these two ideas to construct a stochastic speed Levy model with two independent factors.

A natural issue arising from multifactor models is the difficulty in calibration. Here we follow a useful technique called Markovian projection. This technique is based on a fundamental result by [21], and was first applied to quantitative finance recently by [48]. The basic idea is to reduce a multifactor model to a single factor model by matching only the one-dimensional marginal distribution at maturity, therefore effectively matching the prices of all vanilla options. [52] further discussed the implementation of this idea in the diffusive setting, and [35] extends the projection to a jump-diffusion model. Here we completes the projection in its general Levy jump model.

The outline of the chapter is as follows. In Section 2, we define our stochastic speed Levy model, and discuss its general properties. In Section 3, we discuss the calibration procedures. Section 4 contains some numerical results. Section 5 concludes. The derivations are given in the Appendix.

4.2 The general Markovian Projection framework

In this section, we provide a general introduction to the concept of Markovian projection in the diffusive setting. Suppose a stochastic process F_t driven by Brownian motion diffusions. In financial modeling, this quantity typically take the meaning of forward prices, stock prices or market interest rates. Such an Ito process can be written generically as:

$$dF_t = \Sigma_t dW_t \tag{4.1}$$

Here W_t is standard Brownian motion. It is written as 1-dimension, but this is not a restriction. Simple stochastic calculus shows that if we have d diffusion drivers dW_t^i , the quantity of interest becomes $dF_t = \sum_{i=1}^d \Sigma_t^i dW_t^i$. Here $(\Sigma_t^1, \dots, \Sigma_t^d)$ is a d -dimensional volatility process, and (dW_t^1, \dots, dW_t^d) is a d -dimensional Brownian motion with pairwise correlation ρ_{ij} , then a simple transformation will reduce the model to (4.1):

$$\begin{aligned}\Sigma_t^2 &\equiv \sum_{i,j=1}^d \Sigma_t^i \Sigma_t^j \rho_{ij} \\ dW_t &\equiv \Sigma_t^{-1} \left(\sum_{i=1}^d \Sigma_t^i dW_t^i \right)\end{aligned}$$

So (4.1) already encompasses a wide variety of different and complex dynamics.

The important observation here is that the volatility Σ_t is almost arbitrary. In particular, Σ_t can depend on additional sources of randomness, and hence not measurable with respect to \mathcal{F}_t , the usual filtration by W_t . As a result, F_t has complex dynamics, which is usually not Markovian. This is in sharp contrast to the simple diffusion process, where the strong Markovian property facilitates analysis enormously.

At the same time, for many practical applications, we would ideally prefer to work with Markovian processes. A quintessential example is in financial derivatives modeling. When stock prices S_t follow a risk-neutral local volatility process, $dS = rSdt + \sigma(S, t)Sd\widetilde{W}$, it is Markovian. The well-known Dupire formula will solve out the local volatility function $\sigma(S, t)$ given European call option prices $C(T, K)$. In this way, there exists a one-to-one correspondence among $C(T, K)$, $\sigma(S, t)$ and the marginal transition probability density $p(x, T; S, t)$. This is a highly desirable property that is missing in the more complicated stochastic volatility models, where option pricing gets significantly more involved.

A central observation that motivates the projection method, or, dimension reduction, is that European vanilla option prices depend only on the marginal distribution of the underlying process, not on the entire sample path. In other words, such prices are invariant under a “change of dynamics” as long as the marginal distribution is kept unchanged. The underlying processes can take various and distinct joint distribution over different time horizon, but the marginal distribution can be made the same. In view of the connection between a marginal density and a particular Markovian process formulation, it follows naturally that we can replace a complex, generally non-Markovian process with a simple, Markovian process to maintain marginal invariance. From a practical point of view, for example in option pricing, such a transform will yield identical results, while much simpler to implement. We use the loosely defined term “Markovian projection” to denote such a transform technique.

Back to financial applications, it suggests that we can substitute a local volatility (LV) model for a stochastic volatility (SV) model in order to perform many option pricing related tasks. Often, SV models are hard to calibrate, so it is beneficial to instead calibrate to its LV counterpart, the so-called projected process, and then “inversely mapped” LV back to get the SV fitted parameters.

At the heart of Markovian projection, Gyongy’s theorem lays the theoretical foundation to justify such a transformation:

Theorem *Let X_t be given by:*

$$dX_t = \alpha(t)dt + \beta(t)dW_t. \tag{4.2}$$

where $\alpha(t)$ and $\beta(t)$ are generic stochastic processes such that the SDE (4.2) admits a unique solution. Then there exists a “Markovian projected” process Y_t such that X_t and Y_t have identical 1-dimensional distributions. In particular, Y_t follows a

diffusion SDE:

$$dY_t = a(Y_t, t)dt + b(Y_t, t)dW_t, \quad (4.3)$$

where $a(x, t)$, $b(x, t)$ are determined by $\alpha(t)$ and $\beta(t)$ through:

$$a(x, t) = E(\alpha(t)|X_t = x); \quad (4.4)$$

$$b^2(x, t) = E(\beta^2(t)|X_t = x). \quad (4.5)$$

Markovian projection gives the exact transformation functions. In practice, the real challenge comes in as a computational problem to calculate the conditional expectations. There has been a line of research in the literature to approximate these expectations for relatively simple models. It uses a linear approximation to have analytical tractability. The Gaussian-style approximations work reasonably well.

4.3 Markovian Projection for Poisson Jump processes

To further motivate our subsequent extension of the Markovian projection method, and put it in context, here we re-establish an intermediate extension to Poisson jump processes. When there are finite-activity jumps in the dynamics, the determination of Markovian property still looks similar. Parallel to the local volatility model, we define a counting process M_t with “local intensity” function $\Lambda(M_t, t)$. Here Λ depends explicitly on the underlying, hence making M_t Markovian. Over time interval $(t, t + \delta t)$, we have $P(M_{t+\delta t} - M_t = 1) = \Lambda(M_t, t)\delta t + o(\delta t)$.

Next, we introduce a doubly-stochastic counting process N_t with “stochastic intensity” process λ_t . Here λ_t possibly introduces additional randomness, so N_t is

non-Markovian. A typical example for N_t is the Cox process, which generalizes Poisson process by having a time-dependent, stochastic arrival rate. Over $(t, t+\delta t)$, $P(N_{t+\delta t} - N_t = 1) = \lambda(t)\delta t + o(\delta t)$.

The projection from N_t down to M_t is first established in [35]. We have:

$$\Lambda(x, t) = E(\lambda_t | N_t = x). \quad (4.6)$$

Here is a sketch of proof:

Define $p(x, t) \equiv P(N_t = x)$ as the transition probability density indexed by the terminal variables. Using a first-step analysis argument from t to $t + dt$, the Forward Kolmogorov equation becomes:

$$\frac{\partial}{\partial t} p(x, t) = -\Lambda(x, t)p(x, t) + \Lambda(x-1, t)p(x-1, t). \quad (4.7)$$

Summing over the equation with k running from 0 to x , we get:

$$\Lambda(x, t) = -\frac{1}{p(x, t)} \frac{\partial}{\partial t} \sum_{k=0}^x p(k, t) = -\frac{1}{p(x, t)} \frac{\partial}{\partial t} P(N_t \leq x). \quad (4.8)$$

From the definition of stochastic intensity, we have:

$$P(N_t \leq x) = E \sum_{k=0}^x \frac{1}{k!} \left(\int_0^t \lambda_u du \right)^k \exp\left(- \int_0^t \lambda_u du\right).$$

Taking partial derivative to get:

$$\begin{aligned} \frac{\partial}{\partial t} P(N_t \leq x) &= -E\left(\lambda_t \int_0^t \lambda_u du\right)^k \exp\left(- \int_0^t \lambda_u du\right) \\ &= E\left(E\left(-\lambda_t 1_{N_t=x} | \lambda_\tau, 0 \leq \tau \leq t\right)\right) \\ &= E\left(-\lambda_t 1_{N_t=x}\right) = -E(\lambda_t | N_t = x) P(N_t = x). \end{aligned}$$

Rewrite the final line:

$$\frac{\partial}{\partial t} P(N_t \leq x) = -E(\lambda_t | N_t = x) P(N_t = x). \quad (4.9)$$

Comparing (4.7) with (4.9) to yield the final result (4.6)

4.4 Markovian Projection for Levy processes

We start with notations. Let S_t denote the price of the stock at time $t, 0 < t < T$. Let r denote the continuously compounded risk-free interest rate. Define $F_t = S_t e^{-rt}$ as the discounted stock price. Then under the risk neutral measure, F_t is a martingale. We assume here the stock dynamics is completely driven by compensated Levy jumps, and write:

$$dF_t = F_{t-} \int (e^y - 1)(\mu - \nu)(dt, dy), \quad (4.10)$$

where $\mu(dt, dy)$ is the counting measure associated with the Levy jumps in the logarithm of the discounted stock price. It is more convenient to write in terms of the logarithmic price. $\nu(dt, dy)$ is the compensated Levy measure combined with a speed function. Precisely, the local speed Levy model is:

$$F_t = F_0 \exp(X_t) \quad (4.11)$$

$$dX_t = \alpha(X_{t-}, t)dt + \int y(\mu - \nu)(dt, dy) \quad (4.12)$$

$$\nu(dt, dy) = b(X_{t-}, t)k(y)dydt \quad (4.13)$$

where $\alpha(X_{t-}, t)$ is the drift term to ensure that F_t is a martingale, $k(y)$ is the jump size distribution, $b(X_{t-}, t)$ is the local speed function dependent on space and time. Here is a brief derivation of $\alpha(X_{t-}, t)$. Given the dynamics of dX_t , it is straightforward to get the dynamics of F_t by Ito's formula for semimartingales:

$$dF_t = F_{t-}(\alpha(X_{t-}, t)dt + \int (e^y - y - 1)\nu(dt, dy) + \int (e^y - 1)(\mu - \nu)(dt, dy)).$$

The drift term is set at 0, hence:

$$\alpha(X_{t-}, t)dt + \int (e^y - y - 1)\nu(dt, dy) = 0.$$

Plugging into above the expression of $\nu(dt, dy)$ as in (4.13), we get

$$\alpha(\cdot, t) = -b(\cdot, t)E(e^Y - Y - 1). \quad (4.14)$$

where expectation is taken with respect to the jump size distribution $k(y)$, and Y is the random jump size.

We present the stochastic speed Levy model as a natural extension to the local speed Levy model above. The only difference is that the speed b_t is no longer deterministic. It can be a general stochastic process. This introduces an additional factor to the system, in much the same way as a stochastic volatility model extends its local volatility counterpart. The stochastic speed Levy model is:

$$F_t = F_0 \exp(X_t) \quad (4.15)$$

$$dX_t = \alpha_t dt + \int y(\mu - \nu)(dt, dy) \quad (4.16)$$

$$\nu(dt, dy) = b_t k(y) dy dt. \quad (4.17)$$

$$db_t = \beta(X_{t-}, b_t, t) dt + \sigma(b_t, t) dW_t. \quad (4.18)$$

where α_t is the drift process completely determined by the speed process b_t ,

$$\alpha_t = -b_t E(e^Y - Y - 1). \quad (4.19)$$

The equation (4.18) gives the stochastic evolution of the speed function, parallel to the dynamics of a stochastic volatility. For simplicity, we assume a single diffusive factor for b_t . $\beta(X_{t-}, b_t, t)$ and $\sigma(b_t, t)$ are the drift and volatility functions, respectively. A typical choice, which will be used later, is:

$$\beta(X_{t-}, b_t, t) = \kappa(\rho(X_{t-}, t) - b_t). \quad (4.20)$$

$$\sigma(b_t, t) = \sigma \sqrt{b_t}. \quad (4.21)$$

The introduction of stochastic speed greatly complicates matters. However, based on the idea of Markovian projection, we can reduce it to a local speed model, which can be calibrated equivalently. In [48], it has been shown that if two underlyings with its own diffusive SDEs have equal expected values of their

SDE coefficients, then the two SDEs will have identical one-dimensional marginal distributions, although they have entirely different dynamics and joint distributions. European option prices depend only on the terminal distribution; therefore, the two models will produce identical vanilla option prices, hence the validity of calibration. In other words, we can construct a “reduced” local speed Levy model as a projection of the stochastic speed Levy model, and perform calibration with the reduced model. This is summarized in the following main result:

Theorem 4.1 *Given a stochastic speed Levy model (4.17), there exists a Markovian projected local speed Levy model (4.13). The local speed function is given by conditional expectation of the speed process:*

$$b(t, x) = E(b_t | X_t = x). \tag{4.22}$$

Proof of Theorem: See the Appendix

4.5 Calibration Procedures

Given a stochastic speed Levy model, the previous extension of Markovian projection provides a clear two-step procedure of calibration. At step 1, we calibrate the projected local speed Levy model to the vanilla option prices. Next, we apply our Gyongy result to convert the known local speed model into its original stochastic speed model. The overall effect is a calibration to all vanilla options, while retaining a more realistic (than local speed) dynamics that might be better for more complicated exotic product pricing.

4.5.1 Calibration to local speed

In [5], there are extensive discussions on the recovery of local speed function based on option prices. The idea parallels that of the original Dupire formula for local volatility models. We reinstate the main results here. Some derivations are included in the Appendix for completeness.

Given a local speed Levy model (4.13), the function $b(\cdot, t)$ is actually the log-speed corresponding to log forward prices. We rewrite equation (4.13) as:

$$\nu(dt, dy) = b(X_{t-}, t)k(y)dydt = a(F_{t-}, t)k(y)dydt, \quad (4.23)$$

where $a(\cdot, t) = b(\log(\frac{\cdot}{F_0}), t)$ is the equivalent speed function expressed using the forward price itself. Recall here as a special case, if the speed function is constant λ , then $\nu(dt, dy) = \lambda k(y)dydt$ represents a compound Poisson process with arrival rate λ and jump size distribution $k(y)$. We choose our vanilla options to be written on the forward prices, namely, the option price $C(K, T) = E(F_T - K)_+$. Knowledge of all $C(K, T)$ for the continuum of all strike K for a given maturity T is equivalent to the knowledge of the marginal distribution of F_t at time T . Hence, a Dupire-like formula will recover the local speed function from option prices.

We use the Meyer-Tanaka formula for a convex function f :

$$\begin{aligned} f(X_t) &= f(X_0) + \int_0^t f'(X_s)dX_s + \frac{1}{2} \int_0^t f''(X_{s-})d[X, X]_s^m \\ &+ \sum_{0 < s \leq t} (f(X_s) - f(X_{s-}) - \Delta X_s f'(X_{s-})). \end{aligned}$$

Our $f(\cdot) = (\cdot - K)_+$, we have $f'(\cdot) = 1_{(\cdot > K)}$ and $f''(\cdot) = \delta_{(\cdot = K)}$. We have:

$$\begin{aligned} (F_T - K)_+ &= (F_0 - K)_+ + \int_0^T 1_{(F_{t-} > K)}dF_t + \sum_{t \leq T} 1_{(F_{t-} > K)}(K - F_t)^+ \\ &+ \sum_{t \leq T} 1_{(F_{t-} < K)}(F_t - K)^+. \end{aligned}$$

Taking expectation on both sides, notice $E(dF_t) = 0$ by martingale of F_t to get:

$$C(K, T) = C(K, 0) + E \sum_{t \leq T} 1_{(F_{t-} > K)} (K - F_t)^+ + E \sum_{t \leq T} 1_{(F_{t-} < K)} (F_t - K)^+. \quad (4.24)$$

For $E \sum_{t \leq T} 1_{(F_{t-} > K)} (K - F_t)^+$, the expression within \sum is nonzero iff there is a jump from $F_{t-} = x'$ to $F_t = x'e^y$ with probability $k(y)dy$, and on average there are a number of $a(x', t)dt$ jumps during $(t, t + dt)$. Combine things together, we get:

$$\begin{aligned} & E \sum_{t \leq T} 1_{(F_{t-} > K)} (K - F_t)^+ \\ &= \int_{t=0}^T \int_{x'=K}^{\infty} p(t, x') dx' \int_{y=-\infty}^{\infty} (K - x'e^y)_+ k(y) dy a(x', t) dt \\ &= \int_{t=0}^T \int_{x'=K}^{\infty} p(t, x') dx' \int_{y=-\infty}^{\log(K) - \log(x')} (K - x'e^y) k(y) dy a(x', t) dt. \end{aligned}$$

Similarly:

$$\begin{aligned} & E \sum_{t \leq T} 1_{(F_{t-} < K)} (F_t - K)^+ \\ &= \int_{t=0}^T \int_{x'=0}^K p(t, x') dx' \int_{y=\log(K) - \log(x')}^{\infty} (x'e^y - K) k(y) dy a(x', t) dt. \end{aligned}$$

define the double exponential tail function

$$\psi_e(z) = \begin{cases} \int_{-\infty}^z (e^z - e^y) k(y) dy, & z < 0 \\ \int_z^{\infty} (e^y - e^z) k(y) dy, & z > 0 \end{cases}$$

and taking $\frac{\partial}{\partial T}$ to get:

$$\frac{\partial}{\partial T} C(K, T) = \int_0^{\infty} x' p(x', T) a(x', T) \psi_e\left(\log\left(\frac{K}{x'}\right)\right) dx'. \quad (4.25)$$

The option price can be expressed directly using the transition density at maturity

$$C(K, T) = \int_K^{\infty} p(T, x') (x' - K) dx'$$

. Differentiating twice to get

$$\frac{\partial^2}{\partial K^2} C(x, T) = p(T, x).$$

The Dupire-like equation for local speed model is:

$$\frac{\partial}{\partial T} C(K, T) = \int_0^\infty x' \frac{\partial^2}{\partial K^2} C(x', T) a(x', T) \psi_e(\log(\frac{K}{x'})) dx'. \quad (4.26)$$

To get a more compact expression, redefine the log variables $\bar{x}' = \log(x')$, $\bar{K} = \log(K)$, $\bar{C}(\bar{K}, t) = C(K, t)$, $\bar{a}(\bar{x}', t) = e^{2\bar{x}'} a(e^{\bar{x}'}, t) \frac{\partial^2 C}{\partial K^2}(e^{\bar{x}'}, t)$. Using the new variables and functions, we substitute into (4.26) to get:

$$\frac{\partial}{\partial T} \bar{C}(\bar{K}, T) = \int_{-\infty}^\infty e^{2\bar{x}'} \frac{\partial^2 \bar{C}}{\partial K^2}(e^{\bar{x}'}, T) a(e^{\bar{x}'}, T) \psi_e(\bar{K} - \bar{x}') d\bar{x}'.$$

further simplified to:

$$\frac{\partial}{\partial T} \bar{C}(\bar{K}, T) = \int_{-\infty}^\infty \bar{a}(\bar{x}', T) \psi_e(\bar{K} - \bar{x}') d\bar{x}'. \quad (4.27)$$

The equation (4.27) is a convolution in the \bar{x}' variable: $\frac{\partial \bar{C}}{\partial T} = \bar{a} * \psi_e$. By Fourier transform, we define the transformed functions:

$$\begin{aligned} \widehat{\bar{C}}(u, t) &\triangleq \int_{-\infty}^\infty \bar{C}(\bar{K}, t) e^{iu\bar{K}} d\bar{K}, \quad \widehat{\bar{a}}(u, t) \triangleq \int_{-\infty}^\infty \bar{a}(\bar{K}, t) e^{iu\bar{K}} d\bar{K}. \\ \widehat{\psi}_e(u) &\triangleq \int_{-\infty}^\infty \psi_e(\bar{K}) e^{iu\bar{K}} d\bar{K}. \end{aligned}$$

In the transform domain, equation (4.27) becomes simple multiplication:

$$\frac{\partial}{\partial T} \widehat{\bar{C}}(u, T) = \widehat{\bar{a}}(u, T) \widehat{\psi}_e(u). \quad (4.28)$$

Given all vanilla option prices $C(K, T)$, together with the jump size related double exponential tail function $\psi_e(\cdot)$, we use (4.28) to obtain the local speed function.

4.5.2 From local speed to stochastic speed

Once the local speed function $b(x, t)$ is fully constructed market data, the next step is to find the stochastic speed dynamics consistent with the extended Gyongy theorem. The functional form of the stochastic speed b_t , namely $\beta(X_{t-}, b_t, t)$ and $\sigma(b_t, t)$ needs to be specified. We use the parametrization in (4.20) and (4.21). The forward PIDE for the transition pdf $p(t, x, b)$ is (substitution into equation (A.8))

$$\begin{aligned} \frac{\partial}{\partial t} p(t, x, b) &= -\frac{\partial}{\partial b} (p(t, x, b) \kappa(\rho(x, t) - b)) + \frac{1}{2} \frac{\partial^2}{\partial b^2} (\sigma^2 b p(t, x, b)) \\ &\quad + b \int_y k(y) p(t, x - y, b) dy - b p(t, x, b) \\ &\quad + E(e^Y - 1) b \frac{\partial}{\partial x} p(t, x, b). \end{aligned} \quad (4.29)$$

The goal of calibration at this stage is to find the function $\rho(x, t)$ satisfying (4.22). Notice also that there are more degrees of freedom here than for the local speed model, with free parameters κ and σ governing the dynamics of the speed b_t . Here we do not calibrate to these two parameters, but rather keep them available for future calibration to more exotic products. To solve for the unknown $\rho(x, t)$, the trick is to remove the partial derivatives $\frac{\partial}{\partial b}$ in (4.29). Define the second moment function $H(t, x)$

$$H(t, x) \triangleq \int_b b^2 p(t, x, b) db. \quad (4.30)$$

Apply $\int_b b \{ \dots \} db$ on (4.29). The second term on the right hand side vanishes. We get:

$$\begin{aligned} \frac{\partial}{\partial t} G(t, x) &= \kappa \rho(t, x) p(t, x) - \kappa G(t, x) + \int_y k(y) H(t, x - y) dy \\ &\quad - H(t, x) + E(e^Y - 1) \frac{\partial}{\partial x} H(t, x). \end{aligned} \quad (4.31)$$

According to the theorem result (A.12), $G(t, x) = p(t, x)b(t, x)$. Differentiating with respect to t in (A.12) to have:

$$\frac{\partial}{\partial t}G(t, x) = b(t, x)\frac{\partial}{\partial t}p(t, x) + p(t, x)\frac{\partial}{\partial t}b(t, x). \quad (4.32)$$

Replace the first term $\frac{\partial}{\partial t}p(t, x)$ with (A.10) and have:

$$\begin{aligned} \frac{\partial}{\partial t}G(t, x) &= b(t, x)\left\{\int_y k(y)G(t, x-y)dy - G(t, x)\right. \\ &\quad \left.+ E(e^Y - 1)\frac{\partial}{\partial x}G(t, x)\right\} + p(t, x)\frac{\partial}{\partial t}b(t, x). \end{aligned} \quad (4.33)$$

Equating (4.31) and (4.33) and eliminating $\frac{\partial}{\partial t}G(t, x)$, we finally solve out $\rho(x, t)$ as:

$$\begin{aligned} \rho(x, t) &= b(t, x) + \frac{1}{\kappa}\frac{\partial}{\partial t}b(t, x) + \frac{b(t, x)}{\kappa p(t, x)}\left(\int_y k(y)G(t, x-y)dy - G(t, x)\right. \\ &\quad \left.+ E(e^Y - 1)\frac{\partial}{\partial x}G(t, x)\right) - \frac{1}{\kappa p(t, x)} \\ &\quad \cdot \left(\int_y k(y)H(t, x-y)dy - H(t, x) + E(e^Y - 1)\frac{\partial}{\partial x}H(t, x)\right). \end{aligned} \quad (4.34)$$

(4.34) can be substituted into (A.8) to get an integro-differential equation of $p(t, x, b)$, which can be solved numerically in theory. Here a more realistic approach is to solve iteratively using (A.8), the definitions of $G(t, x)$ and $H(t, x)$, and (4.34). For implementation, we repeat the following steps, starting with known values of all the density elements $p(idt, x, b), \forall x, b$ at current time i .

1. From (A.11) and (4.30), integrate to get $G(idt, x)$ and $H(idt, x)$.
2. From (4.34), obtain $\rho(x, idt)$.
3. From (4.29), propagate to the next time $i + 1$; and solve out the next time step density elements $p((i + 1)dt, x, b), \forall x, b$.

Then the iteration moves to the next time step. To initialize this, the initial density elements $p(0dt, x, b), \forall x, b$ must be all known.

APPENDIX A

PROOF OF THEOREM 4.1

The proof is based on the Markovian property of Levy process. We proceed by establishing the forward PIDEs corresponding to the marginal transition density functions. In order to have a projected local model, the two processes should yield identical PIDEs. The derivation here is very similar for the two models. We start with the stochastic speed Levy model (4.17). From the dynamics, clearly the underlying X_t is not one-dimensional Markovian, but (X_t, b_t) is two-dimensional Markovian. We define $p(t, x, b; t', x', b')$ as the transition density function from $(X_t = x, b_t = b)$ to $(X_{t'} = x', b_{t'} = b')$. We are interested in the forward PIDE of this Markovian process, so we fix the initial time at 0, and suppressing initial variables, write the forward pdf as $p(t, x, b) = p(0, x_0, b_0; t, x, b)$. Notice here that b_t has no jumps, so $b_{t-} = b$ implies $b_t = b$. Moreover, for any fixed time t , $X_t = X_{t-} + \Delta X_t$. With probability 1 there is no jump at exactly this time point, namely $P(\Delta X_t = 0) = 1$. So we don't distinguish the pdfs of X_t and X_{t-} . From now on, we work with $p(t, x, b)$.

Take a smooth function $g(x, b)$, expand the conditional expectation:

$$E(g(X_t, b_t)|X_0 = x_0, b_0 = b_0) = \int_x \int_b p(t, x, b)g(x, b)dxdb. \quad (\text{A.1})$$

For a time increment ϵ , similarly we have

$$E(g(X_{t+\epsilon}, b_{t+\epsilon})|X_0 = x_0, b_0 = b_0) = \int_x \int_b p(t + \epsilon, x, b)g(x, b)dxdb. \quad (\text{A.2})$$

Subtracting (A.1) from (A.2) to get:

$$E(dg(X_t, b_t)|X_0 = x_0, b_0 = b_0) = \int_x \int_b (p(t + \epsilon, x, b)g(x, b) - p(t, x, b)g(x, b))dxdb. \quad (\text{A.3})$$

where the increment of g here is

$$dg(X_t, b_t) = g(X_{t+\epsilon}, b_{t+\epsilon}) - g(X_t, b_t). \quad (\text{A.4})$$

sending $\epsilon \rightarrow 0^+$, the right hand side (*RHS*) of (A.3) becomes:

$$RHS = \epsilon \int_x \int_b \frac{\partial}{\partial t} p(t, x, b) g(x, b) dx db. \quad (\text{A.5})$$

For the left hand side (*LHS*) of (A.3), we need to expand the differential term $dg(X_t, b_t)$. Recall the Ito's formula for semimartingales, and also b_t is diffusive, while X_t has Levy jumps, we get:

$$\begin{aligned} dg(X_t, b_t) &= g(X_{t+\epsilon}, b_{t+\epsilon}) - g(X_t, b_t) \\ &= \frac{\partial g}{\partial b}(X_{t-}, b_t) db_t + \frac{1}{2} \frac{\partial^2 g}{\partial b^2}(X_{t-}, b_t) (db_t)^2 + \frac{\partial^2 g}{\partial x \partial b}(X_{t-}, b_t) d[X, b]_t^c \\ &+ \frac{\partial g}{\partial x}(X_{t-}, b_t) dX_t + \frac{1}{2} \frac{\partial^2 g}{\partial x^2}(X_{t-}, b_t) d[X, X]_t^c + g(X_{t-} + \Delta X_t, b_t) \\ &- g(X_{t-}, b_t) - \Delta X_t \frac{\partial}{\partial x} g(X_{t-}, b_t). \end{aligned}$$

Since X_t has only jump parts, the two cross quadratic terms $d[X, X]_t^c$ and $d[X, b]_t^c$ are 0. Plugging into above the dynamics of X_t and b_t as in (4.16) and (4.18), we have

$$\begin{aligned} dg(X_t, b_t) &= \frac{\partial g}{\partial b}(X_{t-}, b_t) \beta(X_{t-}, b_t, t) dt + \frac{\partial g}{\partial b}(X_{t-}, b_t) \sigma(b_t, t) dW_t \\ &+ \frac{1}{2} \frac{\partial^2 g}{\partial b^2}(X_{t-}, b_t) \sigma^2(b_t, t) dt + \frac{\partial g}{\partial x}(X_{t-}, b_t) \alpha_t dt \\ &+ \frac{\partial g}{\partial x}(X_{t-}, b_t) \int_y y \mu(dt, dy) - \frac{\partial g}{\partial x}(X_{t-}, b_t) \int_y y \nu(dt, dy) \\ &+ \int_y (g(X_{t-} + \Delta X_t, b_t) - g(X_{t-}, b_t) - y \frac{\partial}{\partial x} g(X_{t-}, b_t)) \mu(dt, dy). \end{aligned}$$

Here we have used the fact as $\epsilon \rightarrow 0^+$, ϵ is effectively dt in the above expression.

When $dg(X_t, b_t)$ is applied to the expectation (A.3), the martingale increments will vanish, that is, $E(\text{martingale increments} | X_0 = x_0, b_0 = b_0) = 0$. So we ignore

the two terms $\frac{\partial g}{\partial b}(X_{t-}, b_t)\sigma(b_t, t)dW_t$ and $\int_y(g(X_{t-} + \Delta X_t, b_t) - g(X_{t-}, b_t))(\mu - \nu)(dt, dy)$. Also, we substitute the compensated Levy measure and the drift process by their respective values:

$$\begin{aligned}\nu(dt, dy) &= b_t k(y) dy dt \\ \alpha_t &= -b_t E(e^Y - Y - 1)\end{aligned}$$

We have (ignoring martingale components):

$$\begin{aligned}dg(X_t, b_t) &= \epsilon \left(\frac{\partial g}{\partial b}(X_{t-}, b_t) \beta(X_{t-}, b_t, t) + \frac{1}{2} \frac{\partial^2 g}{\partial b^2}(X_{t-}, b_t) \sigma^2(b_t, t) \right. \\ &\quad \left. - E(e^Y - 1) b_t \frac{\partial g}{\partial x}(X_{t-}, b_t) \right. \\ &\quad \left. + \int_y (g(X_{t-} + y, b_t) - g(X_{t-}, b_t)) b_t k(y) dy \right).\end{aligned}$$

Upon taking $E(\cdot | X_0 = x_0, b_0 = b_0)$, we finally have the *LHS* of (A.3):

$$\begin{aligned}LHS &= E(dg(X_t, b_t) | X_0 = x_0, b_0 = b_0) \\ &= \int_x \int_b p(t, x, b) E(dg(X_t, b_t) | X_{t-} = x, b_{t-} = b) dx db \\ &= \int_x \int_b p(t, x, b) \left\{ \frac{\partial g}{\partial b}(x, b) \beta(x, b, t) + \frac{1}{2} \frac{\partial^2 g}{\partial b^2}(x, b) \sigma^2(b, t) \right. \\ &\quad \left. - E(e^Y - 1) b \frac{\partial g}{\partial x}(x, b) + \int_y (g(x + y, b) - g(x, b)) b k(y) dy \right\} dx db.\end{aligned}$$

The last step is integration by parts for each term for the *LHS* and rearrange in terms of $\int_x \int_b g(x, b) \{ \dots \} dx db$. All boundary values during integration vanish. For example, the first two terms on *LHS* becomes:

$$\int_x \int_b p(t, x, b) \frac{\partial g}{\partial b}(x, b) \beta(x, b, t) dx db = \int_x \int_b g(x, b) \left\{ - \frac{\partial}{\partial b} (p(t, x, b) \beta(t, x, b)) \right\} dx db.$$

and (after integration by parts twice)

$$\int_x \int_b p(t, x, b) \frac{1}{2} \frac{\partial^2 g}{\partial b^2}(x, b) \sigma^2(b, t) dx db = \int_x \int_b g(x, b) \left\{ \frac{1}{2} \frac{\partial^2}{\partial b^2} (\sigma^2(b, t) p(t, x, b)) \right\} dx db. \tag{A.6}$$

Finally, by equating *LHS* with *RHS*, this holds true for all such functions g . So the integrands must equal. This is actually one method to derive the forward PDE for the special case of diffusion-only dynamics. We now remove $\int_x \int_b g(x, b) \{ \dots \} dx db$ on both sides, equate the integrands and get the forward PIDE for the transition density function $p(t, x, b)$:

$$\begin{aligned} \frac{\partial}{\partial t} p(t, x, b) &= -\frac{\partial}{\partial b} (p(t, x, b) \beta(t, x, b)) + \frac{1}{2} \frac{\partial^2}{\partial b^2} (\sigma^2(b, t) p(t, x, b)) \\ &\quad + b \int_y k(y) p(t, x - y, b) dy - b p(t, x, b) \end{aligned} \quad (\text{A.7})$$

$$+ E(e^Y - 1) b \frac{\partial}{\partial x} p(t, x, b). \quad (\text{A.8})$$

Clearly, (A.8) is a PIDE containing an integral term $\int_y k(y) p(t, x - y, b) dy$ that resembles a convolution.

In an identical manner, we can work out the forward PIDE for the corresponding local speed Levy model. Some notations are slightly different. We have a local speed function $b(X_t, t)$. In the local setting, X_t is one-dimensional Markovian, so we focus on the forward density function $p(t, x) = p(0, x_0; t, x)$, when the underlying process moves from x_0 at time 0 to x at time t . The derivation involves expanding the conditional expectation $E(g(X_t) | X_0 = x_0) = \int_x p(t, x) g(x) dx$, taking a small time increment, applying Ito's formula to $dg(X_t)$ and rearranging terms. The calculation is almost identical, so the details are omitted here. We have the forward PIDE for the transition density function $p(t, x)$:

$$\frac{\partial}{\partial t} p(t, x) = \int_y k(y) p(t, x - y) b(t, x - y) dy - p(t, x) b(t, x) + E(e^Y - 1) \frac{\partial}{\partial x} (p(t, x) b(t, x)). \quad (\text{A.9})$$

When the local model becomes a Markovian projection of the stochastic model, the two models will have identical marginal distribution for the underlying X_t . In other words, the two PIDEs (A.8) and (A.9) will produce the same one-dimensional

density $p(t, x)$. For the stochastic speed model, such a density is a simple integration of the joint density $p(t, x, b)$ over b : $p(t, x) = \int_b p(t, x, b)db$. So we integrate out b over equation (A.8) to get:

$$\frac{\partial}{\partial t}p(t, x) = \int_y k(y)G(t, x - y)dy - G(t, x) + E(e^Y - 1)\frac{\partial}{\partial x}G(t, x). \quad (\text{A.10})$$

where for notational simplicity, denote

$$G(t, x) \triangleq \int_b bp(t, x, b)db. \quad (\text{A.11})$$

Comparing (A.9) and (A.10), and with identical $p(t, x)$ we immediately obtain the simple result:

$$G(t, x) = p(t, x)b(t, x). \quad (\text{A.12})$$

This is the extension of Gyongy theorem to the general Levy jump case. An equivalent way to rewrite this yields:

$$b(t, x) = \frac{G(t, x)}{p(t, x)} = \frac{\int_b bp(t, x, b)db}{\int_b p(t, x, b)db} = E(b_t|X_t = x). \quad (\text{A.13})$$

Proof is complete.

BIBLIOGRAPHY

- [1] Attias (2000), “A Variational Bayesian framework for graphical models”, *Advances in Neural Processing Systems 2000*
- [2] Attias (1999), “Inferring parameters and structure of latent variable models by variational Bayes”, *15th conference on Uncertainty in Artificial Intelligence, 1999*
- [3] Antonov,A. Misirpashaev,T (2006), “ Markovian Projection onto a Displaced Diffusion: Generic Formulas with Applications.”,
- [4] Beal, M. Ghahramani,Z, 2003 “The Variational Bayesian EM Algorithm for Incomplete Data: with Application to Scoring Graphical Model Structures”,
- [5] Carr, P., H.Geman, D.Madan, M.Yor (2004), “From Local Volatility to Local Levy Models” ,,
- [6] Corduneanu,A. and Bishop, C (2001), “Variational Bayesian Model Selection for Mixture Distributions”, *Artificial Intelligence and Statistics 2001*
- [7] Chappell,M. Groves,A. Whitcher,B. Woolrich,M, 2009, “Variational Bayesian Inference for a Nonlinear Forward Model”,
- [8] Crisan,D. Moral,P and Lyons T (1999), “Non-linear filtering using branching and interacting particle systems”, *markov processes Related Fields, vol 5 no. 3, pp 293-319, 1999*
- [9] Crisan, D. “Particle filters– A theoretical perspective”, *Sequential Monte Carlo Methods in Practice, Berlin: Springer Verlag 2001*
- [10] Carvalho,C.M, M.Johannes, H.Lopes and N.Polson (2008) “Particle Learning and Smoothing”, *Working Paper: Duke University,*
- [11] Dempster, A.P, Laird, N.M, Rubin, D.B 1977, Maximum likelihood from incomplete data via the EM algorithm (with discussion) , *J.R.Stat. Soc.Ser. B 39(1), 1-38(1977)*
- [12] Dupire, B.(1994), “Pricing with a smile,”, *Risk 7, 18-20,*
- [13] Dupire, B. (1997), “A Unified Theory of Volatility,”

- [14] Eraker, B., M. Johannes and N. Polson (2003) "The impact of jumps in equity index volatility and returns", *Journal of Finance*, 58: 1269-1300,
- [15] Green (1993), "Reversible jump Markov chain Monte Carlo computation and Bayesian model determination", *Biometrika*, vol 82, pp 711-732, 1995
- [16] Ghahramani, Z and Beal, M (2001), "Graphical models and variational methods", *Advanced Mean Field Methods, The MIT Press, 2001*
- [17] Giesecke, K. Goldberg, L (2005), "A Top-down approach to multi-name credit"
- [18] Golightly, A (2007) "Bayesian Filtering for Jump-Diffusions with Applications to Stochastic Volatility",
- [19] Gordon, N., D. Salmond and F. Smith (1993) "A novel approach to nonlinear and non-Gaussian Bayesian state estimation", *IEE Proceedings. Part F: Radar and Sonar Navigation* 140(2): 107-113,
- [20] Golightly, A. and D. Wilkinson (2006) "Bayesian sequential inference for nonlinear multivariate diffusions", *Stat Comput*(2006)16: 323-338,
- [21] Gyongy, I. (1986), "Mimicking the one-dimensional distributions of processes having an Ito differential," , *Prob. Th. Rel. Fields*, 71: 501-516,
- [22] Hinton, G.E and Camp, D. van (1993), "Keeping neural networks simple by minimizing the description length of the weights", *Proceedings of the 6th Annual Workshop on Computer Learning Theory*, pp.5-13, *ACM Press, New York NY. 1993*,
- [23] Heston (1993), "A closed-form solution for options with stochastic volatility," ,
- [24] Hunter, D. Lange, K (2003) "A Tutorial on MM Algorithms" ,
- [25] He, Z. and J. Maheu (2008) "Real Time Detection of Structural Breaks in GARCH Models" ,
- [26] Isard, M. and Blake, A (1998) "CONDENSATION: conditional density propagation for visual tracking", *Int. J. Comput. Vis*, vol 29 no 1, pp 5-28, 1998

- [27] Johannes,M.and N.Polson (2007)“Particle Filtering and Parameter Learning”,*Working Paper, University of Chicago GSB*,
- [28] Johannes,M., N.Polson and J.Stroud (2009)“Optimal Filtering of Jump Diffusions: Extracting Latent States from Asset Prices”,*Review of Financial Studies, Jan 2009*,
- [29] Jasra, A., D.Stephens, A.Doucet, T.Tsagaris (2008), “Inference for Levy Driven Stochastic Volatility Models via Adaptive Sequential Monte Carlo” ,,
- [30] Kitagawa, G. “Monte Carlo filter and smoother for non-Gaussian non-linear state space models”,*J. Comput.Graph.Statist, vol 5 no 1 pp 1–25, 1996*
- [31] Kitagawa, G. “Self-organizing state space models”,*J. Amer.Statist. Assoc, vol 93, pp 1203-1215, 1998*
- [32] Lange,K. Hunter,D. Yang,I (1999), “Optimization Transfer Using Surrogate Objective Functions” ,
- [33] Li,J. (2009)“Sequential Bayesian Analysis of Time-Changed Infinite Activity Derivatives Pricing Models”,*Working Paper: Bocconi University, Italy*,
- [34] Li,L (2010), “Statistical Methods and Analysis for Genome-Wide Association Studies”, Ph.D. Thesis, Cornell university, 2010
- [35] Lopatin, A., T. Misirpashaev (2007), “Two-Dimensional Markovian Model for Dynamics of Aggregate Credit Loss,” ,
- [36] Lopes,H. and N.Polson (2009)“Extracting S&P500 and NASDAQ volatility: The Credit Crisis of 2007-2008” ,
- [37] Little, R.J.A, Rubin, D.B, Statistical Analysis with Missing Data, 2nd edition,*Wiley-Interscience, New York*
- [38] Liu, J. and M.West(2001), “Combined parameter and state estimation in simulation-based filtering.”,A.Doucet, N.de Freitas, and N.Gordon, Eds *Sequential Monte Carlo Methods in Practice, Springer, New York, NY*,
- [39] Li, H., M.Wells and C.Yu(2008), “A Bayesian Analysis of Return Dynamics with Stochastic Volatility and Levy Jumps” ,

- [40] MacKay, D.J.C(1995), “Free energy minimization algorithms for decoding and cryptanalysis”, *Electronics Letters*, vol.31, no.6, pp.446-447,
- [41] MacKay, D.J.C(1995), “Developments in probabilistic modelling with neural networks – ensemble learning”, *Neural Networks: Artificial Intelligence and Industrial Application. Proceedings of the 3rd Annual Symposium on Neural Networks, Nijmegen, Netherlands, 14-15 September 1995, (Berlin) pp.191-198, Springer 1995*,
- [42] Merwe, R., A.Doucet, N.Freitas, E.Wan (2000), “The Unscented Particle Filter” ,,
- [43] McLachaln, G.J, Krishnan,T 2008, McLachaln, G.J, Krishnan,T (2008) “The EM Algorithm and Extensions”, 2nd edition, *Wiley-Interscience, New York*,
- [44] McGrory, C.A, Titterington, D.M, 2008, “Variational Bayesian Analysis for Hidden Markov Models”
- [45] Neal,R.M and Hinton,G.E (1998), “A New View of the EM Algorithm that justifies Incremental, Sparse, and other variants”, *NATO Science Series, Dordrecht: Kluwer Academic Publishers 1998*
- [46] Opper,M and Saad, D., *Advanced Mean Field Methods: Theory and Practice. Cambridge, Massachusetts: The MIT Press 2001*
- [47] Parisi,G., *Statistical Field Theory, Reading Massachusetts: Addison Wesley, 1988*
- [48] Piterbarg, V.(2006), “Markovian Projection methods for Volatility Calibration” ,,
- [49] Pitt, M. and N.Shephard(1999), “Filtering via simulation: auxiliary particle filters.”, *Journal of the American Statistical Association 94(446), 590-599*,
- [50] Qi, Y. and Jaakkola, TS (2007), “Parameter expanded variational Bayesian methods”, *Advances in Neural Information Processing Systems, 2007*
- [51] Raggi, D., S.Bordignon (2006), “Sequential Monte Carlo Methods for Stochastic Volatility Models with Jumps”,

- [52] Ren, Y., D.Madan, M.Qian (2007), “Calibrating and pricing with embedded local volatility models,” , *Risk September 2007: 138-143*,
- [53] Saul, L.K, Jaakkola, T.S and Jordan, M.I (1996), “Mean field theory for sigmoid belief networks.” , *Journal of Artificial Intelligence Research, vol.4, pp.61-76, 1996*,
- [54] Sarkka, S. Nummenmaa, A, 2008, “Recursive Noise Adaptive Kalman Filtering by Variational Bayesian Approximation” ,
- [55] Smidl, V. Quinn, A. (2006), “The Variational Bayes Method in Signal Processing,” , *Springer, Signals and Communication Technology Series, 2006*
- [56] Smidl, V, Quinn, A (2007), “The Restricted Variational Bayes Approximation in Bayesian Filtering,”
- [57] Storvik, G.(2002), “Particle Filters in State Space Models with the Presence of Unknown Static Parameters.” ,*IEEE Transactions Signal Processing, 50, 281-289*,
- [58] Teh,Y.W, Newman,D. Welling,M (2006), “A Collapsed Variational Bayesian Inference Algorithm for Latent Dirichlet Allocation” ,
- [59] Zhe, C. (2006), “Bayesian Filtering: From Kalman Filters to Particle Filters, and Beyond” ,
- [60] Zhang,Z. Kwok,J. Yeung,D-Y (2004), “Surrogate maximization/Minimization Algorithms for AdaBoost and the Logistic Regression Model” ,