PARTICLE MCMC FOR A TIME CHANGED LÉVY PROCESS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF APPLIED MATHEMATICS
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

AYHAN YÜKSEL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
FINANCIAL MATHEMATICS

OCTOBER 2015

Approval of the thesis:

PARTICLE MCMC FOR A TIME CHANGED LÉVY PROCESS

submitted by **AYHAN YÜKSEL** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Department of Financial Mathematics, Middle East Technical University** by,

Prof. Dr. Bülent Karasözen
Director, Graduate School of **Applied Mathematics** _____

Assoc. Prof. Dr. Ali Devin Sezer
Head of Department, **Financial Mathematics** _____

Assoc. Prof. Dr. Azize Hayfavi
Supervisor, **Financial Mathematics, METU** _____

Assoc. Prof. Dr. C. Coşkun Küçüközmen
Co-supervisor, **International Trade and Finance, İzmir University of Economics** _____

Examining Committee Members:

Prof. Dr. Gül Ergün
Statistics, Hacettepe University _____

Assoc. Prof. Dr. Azize Hayfavi
Financial Mathematics, METU _____

Assoc. Prof. Dr. Ömür Uğur
Financial Mathematics, METU _____

Assoc. Prof. Dr. Yeliz Yolcu Okur
Financial Mathematics, METU _____

Assoc. Prof. Dr. Tolga Omay
Economics, Çankaya University _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: AYHAN YÜKSEL

Signature :

ABSTRACT

PARTICLE MCMC FOR A TIME CHANGED LÉVY PROCESS

Yüksel, Ayhan

Ph.D., Department of Financial Mathematics

Supervisor : Assoc. Prof. Dr. Azize Hayfavi

Co-Supervisor : Assoc. Prof. Dr. C. Coşkun Küçüközmen

October 2015, 82 pages

For almost any type of financial modelling exercise, the most fundamental problem is finding suitable stochastic processes that capture the observed behaviour of asset prices well. Stochastic volatility models, and their extensions with jumps, are class of flexible models that can capture such empirical dynamics quite well. However this richer modelling environment comes at the expense of estimation challenges. Estimation of these flexible models involves some additional challenges that do not exist for simpler ones.

In this thesis, motivated by models with stochastic volatility and jumps, simulation based Bayesian inference methods are analyzed. First we discuss different Markov Chain Monte Carlo (MCMC) approaches in detail, develop various algorithms and implement them for a basic stochastic volatility model. Next we turn our attention to on-line inference and analyze particle filtering methods. We begin with a simple particle filter and then discuss methods to improve the basic filter. We also develop Monte Carlo algorithms to implement particle filters for our stochastic volatility model.

More advanced financial models typically include many latent random variables and complicated likelihood functions where standard MCMC methods may fail to efficiently estimate them. As a more effective alternative, we discussed the Particle MCMC methods recently proposed by C. Andrieu, A. Doucet, and R. Holenstein (Particle Markov Chain Monte Carlo. Journal of the Royal Statistical Society: Series B 72 (3), 2010, pp 269-342). Particle MCMC methods combine two strands of simulation

based Bayesian inference, namely, particle filtering and MCMC, and offer a powerful tool for estimating complex financial models. The theoretical foundations for particle MCMC as well as various samplers proposed in the literature are analyzed in the thesis.

In the final part of the thesis, we develop MCMC and particle MCMC methods for a stock price model with a time changed Lévy process. We assume that the stock price follows a Heston-type stochastic volatility plus variance-gamma jumps in returns. Variance-Gamma process is an infinite activity finite variation Lévy process obtained by subordinating an arithmetic Brownian motion with a Gamma process. The model is quite flexible in its nature and can capture most of the observed characteristics of stock prices.

Our main contribution to existing academic literature is the efficient particle MCMC algorithms that are developed for the Lévy based model. We compare MCMC and particle MCMC algorithms in an empirical implementation using S&P500 Index with 15 years of data. The results indicate that the particle MCMC algorithm is a more efficient alternative to standard MCMC and typically gives smaller standard errors and lower autocorrelations.

Keywords : bayesian estimation, markov chain monte carlo, particle filtering, sequential monte carlo, stochastic volatility, jump processes, Lévy processes

ÖZ

ZAMAN DEĞİŞTİRİLMİŞ BİR LÉVY SÜRECİ İÇİN PARÇACIK MARKOV ZİNCİRİ MONTE CARLO YAKLAŞIMI

Yüksel, Ayhan

Doktora, Finansal Matematik Bölümü

Tez Yöneticisi : Doç. Dr. Azize Hayfavi

Ortak Tez Yöneticisi : Doç. Dr. C. Coşkun Küçüközmen

Eylül 2015, 82 sayfa

Finansal modelleme çalışmalarında karşılaşılan en temel problemlerden biri varlık fiyatlarının gözlenen özeliklerine uygun rassal süreçlerin elde edilmesidir. Stokastik oynaklık modelleri ve bunların sıçrama süreçleri ile genişletilmiş versiyonları, bu tür ampirik dinamikleri yakalayabilecek esnek bir model sınıfını oluşturmaktadır. Ancak bu zengin modelleme imkanı, modellerin tahmin edilmesine ilişkin çeşitli zorlukları da beraberinde getirmektedir. Bu esnek modellerin tahmin edilmesi, daha basit modeller için mevcut olmayan bazı ek zorluklar içermektedir.

Bu tezde, stokastik oynaklık ve sıçrama süreçlerinin esnek yapısından hareketle, söz konusu modellerin tahmin edilmesinde kullanılmak üzere simülasyon bazlı Bayesci tahmin yöntemleri incelenmektedir. Bu çerçevede, öncelikle Markov Zinciri Monte Carlo (MZMC) yaklaşımları detaylı olarak incelenmiş ve basit bir stokastik oynaklık modeli için farklı MZMC algoritmaları oluşturulmuş ve uygulanmıştır. Daha sonra anlık tahmin yaklaşımları ele alınmış ve bu kapsamda parçacık filtresi yöntemleri incelenmiştir. Basit bir parçacık filtresi ile başlanılmış ve sonrasında filtreleme yönteminin geliştirilmesine ilişkin yaklaşımlar tartışılmıştır. Ayrıca incelenen stokastik oynaklık modeli için çeşitli parçacık filtrelerini uygulamak amacıyla Monte Carlo algoritmaları geliştirilmiştir.

İncelenen stokastik oynaklık modelinin aksine, daha gelişmiş finansal modeller genellikle gözlenemeyen birçok rastgele değişken ve karmaşık olabilirlik fonksiyonları içerebilmekte

ve dolayısıyla standart MZMC yöntemleri bu tür modelleri etkin bir şekilde tahmin etmekte yetersiz kalabilmektedir. Bu durumda daha etkin bir alternatif olarak kullanılmak üzere parçacık MZMC yaklaşımları (Particle Markov Chain Monte Carlo. Journal of the Royal Statistical Society: Seri B 72 (3), 2010, sayfa 269-342) yakın zamanda literatürde önerilmiştir. Parçacık MZMC yaklaşımı, iki temel simülasyon bazlı Bayesci tahmin yöntemi olan MZMC ve parçacık filtresi yöntemlerini birleştirmekte ve karmaşık finansal modelleri tahmin etmek için güçlü bir yaklaşım sunmaktadır. Tezde parçacık MZMC yaklaşımına ilişkin teorik çerçeve ile literatürde önerilen çeşitli parçacık MZMC algoritmaları incelenmiştir.

Tezin son bölümünde, zaman değiştirilmiş Lévy süreçlerine dayalı bir hisse fiyat modeli için MZMC ve parçacık MZMC algoritmaları geliştirilmiştir. Modelde hisse fiyatlarının Heston türü bir stokastik oynaklık süreci izlediği ve ayrıca getirilerin varyans–gama türü sıçrama süreçleri içerdiği varsayılmıştır. Varyans–gama süreci aritmetik Brownian sürecinin gama süreci ile zaman değiştirilmesi yoluyla elde edilmekte olup, sonsuz aktivite ve sonlu değişim içermektedir. Model bu yapısı ile oldukça esnektir ve hisse senedi fiyatlarının gözlenen özelliklerinin çoğunu yakalayabilme özelliğine sahiptir.

Tezde bu esnek model için MZMC ve parçacık MZMC algoritmaları geliştirilmiştir. Tezin akademik literatüre ana katkısı Lévy tabanlı bu model için geliştirilen etkin parçacık MZMC algoritmalarıdır. Model için geliştirilen MZMC ve parçacık MZMC algoritmaları, 15 yıllık veri ile S&P500 endeksi üzerinde yapılan ampirik uygulamalarda karşılaştırılmıştır. Sonuçlar parçacık MZMC yaklaşımının, standart MZMC yaklaşımına göre daha etkin bir tahmin yöntemi olduğunu ve parçacık MZMC yaklaşımı ile elde edilen standart hata ve otokorelasyonların daha düşük olduğunu göstermektedir.

Anahtar Kelimeler : bayesci tahmin yöntemleri, markov zinciri monte carlo yaklaşımı, parçacık filtresi, ardışık monte carlo yöntemi, stokastik oynaklık, sıçrama süreçleri, Lévy süreçleri

To my wife Dilek Yüksel whose constant love has always been a source of inspiration and encouragement for me.

And to my little princess Aynur Lina Yüksel, the most precious gift I have ever received.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor Assoc. Prof. Dr. Azize Hayfavi for her valuable guidance and important suggestions during all stages of my masters and doctoral studies at METU.

I am deeply grateful to my co–advisor and all–time–mentor Assoc. Prof. Dr. C. Coşkun Küçüközmen whose precious support and encouragement always helped me throughout my entire career.

I would like to thank all members and administrative staff of the Institute of Applied Mathematics for their guidance and support.

I also gratefully commemorate Prof. Dr. Hayri Körezlioğlu for his guidance and encouraging me for doctoral studies.

Finally, I am thankful by heart to my wife Dilek Yüksel for her ever undying love, enormous support and great patience. This thesis would not have been possible without her unconditional and endless support and encouragement.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ACF	Autocorrelation Function
APF	Auxiliary Particle Filter
AR	Autoregressive
ARCH	Autoregressive Conditional Heteroskedasticity
ARS	Adaptive Rejection Sampling
ARSV	Autoregressive Stochastic Volatility
ARMS	Adaptive Rejection Metropolis Sampling
GARCH	Generalized Autoregressive Conditional Heteroskedasticity
GBM	Geometric Brownian Motion
JD	Jump Diffusion
LM	Lagrange Multiplier
MC	Monte Carlo
MCMC	Markov Chain Monte Carlo
MH	Metropolis Hastings
OU	Ornstein–Uhlenbeck
PF	Particle Filtering
PG	Particle Gibbs
PIMH	Particle Independent Metropolis Hastings
PMCMC	Particle Markov Chain Monte Carlo
PMMH	Particle Marginal Metropolis Hastings
SDE	Stochastic Differential Equation
SD	Standard Deviation
SE	Standard Error
SIS	Sequential Importance Sampling
SISR	Sequential Importance Sampling and Resampling
SMC	Sequential Monte Carlo
SV	Stochastic Volatility
VG	Variance Gamma

CHAPTER 1

INTRODUCTION

For most quantitative finance problems, including asset allocation, risk estimation and derivative pricing, the first step would be assuming some stochastic processes for asset prices and other relevant variables. Once we determine the relevant stochastic processes, the results of the subsequent steps critically depend on our initial assumption. For instance pricing a derivative contract with a Gaussian model, while the underlying assets have highly skewed and fat–tailed distribution, will give very poor results. Therefore finding suitable stochastic processes that fits observed asset prices well is the most fundamental problem in financial modelling.

The Gaussian model is typically the starting point for many quantitative models. However it typically cannot capture the many properties of observed asset prices and thus we need more flexible processes for modelling.

1.1 Gaussian Assumption

Gaussian model is the most widely used assumption in financial modelling. Bachelier [3] is the first one who introduce a stock price model driven by Brownian motion. Samuelson [66] extended this by assuming geometric Brownian motion (GBM) for stock prices. Using this assumption Black and Scholes [8] and Merton [53] derived the famous Black-Scholes option pricing formula.

Although its simplicity, Gaussian assumption puts restrictions on the statistical properties of asset prices. The Black–Scholes model assumes that the stock price follows a geometric Brownian motion, i.e.

$$dS_t/S_t = \mu dt + \sigma dW_t \tag{1.1}$$

where S_t is the stock price at time t, μ and σ are constants, and W_t is a standard Brownian motion. Using Itô's lemma, the stochastic differential equation (SDE) has the analytic solution $S_t = S_0 \exp\left(\left(\mu - \frac{1}{2}\sigma^2\right) t + \sigma W_t\right)$, for an arbitrary initial value S_0. And this implies that, over a time interval of length Δt, log returns can be expressed as $\log\left(S_{t+\Delta t}/S_t\right) = \left(\mu - \frac{1}{2}\sigma^2\right)\Delta t + \sigma\left(W_{t+\Delta t} - W_t\right)$, and has a Gaussian distribution, i.e. $\log\left(S_{t+\Delta t}/S_t\right) \sim N\left(\left(\mu - \frac{1}{2}\sigma^2\right)\Delta t, \sigma^2\Delta t\right)$. The model implies that returns are independent and identically distributed, return distribution is symmetric and mesokurtic,

price paths are almost surely continuous and volatility (as captured by σ) is constant.

When we observe the actual asset prices, we see some common statistical properties, called 'stylized facts'. These properties are common across many instruments, markets and time periods. When compared with these stylized facts, Gaussian assumption seem to be very restrictive. Some of these stylized facts are as follows[1]:

- Linear autocorrelations of asset returns are often insignificant, except for very small intraday time scales. This suggests that returns have independent innovations.
- The unconditional distribution of returns have slight skewness (i.e. gains and losses are asymmetric) and fat tails (tails have a power–law decay).
- The prices exhibit spikes. This suggests that price process has jump components
- The shape of the return distributions changes for different time scales and for large time intervals become more like a normal distribution. This suggests a central limit theorem effect.
- Conditional volatility is not constant over time. Conditional volatility exhibits mean reversion and clustering.
- The residual returns corrected for volatility clustering still have fat tails. However, the tails are less heavy than in the unconditional distribution.
- There is a leverage effect, i.e. volatility is negatively correlated with returns.
- Autocorrelation of absolute returns have a long–range dependence. This suggests that volatility has a long memory.
- The implied volatilities derived from market prices have volatility smiles and/or skews. This suggests that the Black–Scholes model cannot capture observed characteristics.

These stylized properties suggest that the Gaussian assumption is overly simplistic and one should use more flexible models that can incorporate mean–reverting and clustering non–constant volatility, random jumps, leverage effects and can generate asymmetric and heavy–tailed return distributions. There are two classes of models that aim to capture these additional dynamics. These are stochastic volatility models and jump models. The following sections outline some popular stochastic volatility and jump models proposed in the literature.

1.2 Stochastic Volatility Models

Stochastic volatility models extends the simplistic constant volatility assumption of Black–Scholes model. There are two types of non–constant volatility models proposed in the literature.

[1] For a detailed discussion of stylized facts see Cont [12] and Sewel [69].

The first type is the class of *Generalized Autoregressive Conditional Heteroskedasticity (GARCH)* models. GARCH models are parsimonious models that can capture volatility clustering and generate skewed and heavy–tailed return distributions. The volatility is assumed to be non–constant, but conditionally deterministic. For most GARCH models there are no analytical solutions for the temporal aggregation properties and thus it is typically difficult to derive GARCH–based option pricing models. For a recent survey on GARCH models see, for example, [1].

The second type is the class of stochastic volatility (SV) models. SV models are typically built in continuous time and assume that the volatility itself follows certain stochastic differential equations. The general setup[2] for these models are as follows:

$$\begin{aligned} dS_t/S_t &= \mu dt + \sigma_t dW_t && (1.2)\\ \sigma_t &= f(Y_t) && (1.3)\\ dY_t &= \alpha(Y_t, t)\, dt + \gamma(Y_t, t)\, dB_t && (1.4) \end{aligned}$$

where σ_t is the value of (stochastic) volatility at time t, $f(\cdot)$ is a positive function, Y_t is the stochastic process underlying the volatility, $\alpha(\cdot,\cdot)$ and $\gamma(\cdot,\cdot)$ are functions of time and Y_t satisfying certain conditions so that the SDE admits a unique solution [3], W_t and B_t are two standard Brownian motions with correlation $d\langle W, B\rangle_t = \rho dt$.

Typically $f(\cdot)$ is chosen in such a way that volatility is positive, the $\alpha(\cdot,\cdot)$ and $\gamma(\cdot,\cdot)$ are chosen in such a way that ensures volatility clustering and mean–reversion, and $\rho < 0$ captures the leverage effects. With these choices, we can generate asymmetric and heavy–tailed unconditional return distributions. Because of Brownian motion assumptions, these models generate conditionally Gaussian return distributions. SV models can capture volatility smiles or smirks. However, like Black–Scholes model, SV models assume the continuity of price paths (i.e. no spikes). Some popular SV model specifications are given in Table 1.1

Table 1.1: Some Popular SV Model Specifications

Author	$\alpha(t, Y_t)$	$\gamma(t, Y_t)$	$f(Y_t)$	ρ
Hull and White [38]	αY_t	γY_t	$\sqrt{Y_t}$	$\rho = 0$
Scott [68]	$\alpha(m - Y_t)$	γ	$Y_t, \exp(Y_t)$	$\rho = 0$
Wiggins [76]	$\alpha(t, Y_t)$	γY_t	Y_t	$\rho \neq 0$
Stein and Stein [71]	$\alpha(m - Y_t)$	γ	$\lvert Y_t \rvert$	$\rho = 0$
Heston [36]	$\alpha(m - Y_t)$	$\gamma\sqrt{Y_t}$	$\sqrt{Y_t}$	$\rho \neq 0$
Ball and Roma [4]	$\alpha(m - Y_t)$	$\gamma\sqrt{Y_t}$	$\sqrt{Y_t}$	$\rho = 0$
Nelson [56]	$\alpha(m - Y_t)$	γY_t	$\sqrt{Y_t}$	$\rho = 0$
Heston [37]	$\alpha Y_t(m - Y_t)$	$\gamma Y_t^{3/2}$	$\sqrt{Y_t}$	$\rho \neq 0$
Hagan, Kumar, Lesniewski and Woodward[32]	0	γY_t	Y_t	$\rho \neq 0$

[2] Throughout this thesis, we are always interested in models for observed asset prices. Models for derivative pricing is out of our research scope. Therefore all stochastic processes presented here are assumed to be under real–world probability measure P.

[3] For such conditions see Theorem 3.5.3, p.49 of [48]

1.3 Jump Models

Although SV models extends the Black–Scholes model in certain ways, they still cannot capture some properties of observed asset prices. For instance, SV models assume continuous price paths, conditionally Gaussian returns and a smooth volatility process. To better replace these unrealistic assumptions, different jump models are proposed in the literature.

The most widely used jump process is Poisson process. Extending our previous notation, a Poisson jump model where jumps exist both in price and volatility has the following form:

$$\begin{aligned} dS_t/S_t &= \mu dt + \sigma_t dW_t + Z_{1t} dN_{1t} && (1.5)\\ \sigma_t &= f(Y_t) && (1.6)\\ dY_t &= \alpha(Y_t, t)\, dt + \gamma(Y_t, t)\, dB_t + Z_{2t} dN_{2t} && (1.7) \end{aligned}$$

where N_{1t} and N_{2t} represent the Poisson processes, Z_{1t} and Z_{2t} represent the jump sizes for price and volatility processes, respectively.

Some popular Poisson–based jump model specifications are given in Table 1.2. SDE specifications for the price process and for the volatility process are shown in the upper and lower parts of the table, respectively.

Table 1.2: Some Popular Poisson–based Jump Model Specifications

Article	Diffusive Part	Jump Frequency	Jump Size	Jump Intensity
		Price Process		
Merton [54]	GBM	Poisson	Normal	Constant
Kou [47]	GBM	Poisson	Double Exponential	Constant
Bates [6]	GBM	Poisson	Normal	Constant
Bates [7]	GBM	Poisson	Normal	Stochastic
Duffie, Pan and Singleton [19]*	GBM	Poisson	Normal	Constant
		Volatility Process		
Merton [54]	-	-	-	-
Kou [47]	-	-	-	-
Bates [6]	Heston	-	-	-
Bates [7]	Heston	-	-	-
Duffie, Pan and Singleton [19]*	Heston	Poisson	Exponential	Constant

*:There are two versions of this model: a) independent jumps in returns and volatility, b) jumps are governed by the same Poisson process and the jump sizes are correlated.

Poisson process can only have finite number of (possibly large) jumps within a finite time period. This restricts the flexibility of model. However the class of Lévy processes, which also includes the Poisson process as a special case, has more flexibility in designing jumps. For example, some Lévy processes have infinite activity, i.e can have infinite number of jumps within a finite time period. These models can capture

both small and large jumps simultaneously. Infinite activity Lévy processes include the inverse Gaussian model [5], the generalized hyperbolic class[20], the variance–gamma model [52], generalized variance–gamma model (also known as CGMY model)[9] and finite moment log–stable model[10]. Furthermore, by applying a stochastic time change, it is also possible to add a stochastic volatility component to a Lévy process. For instance we will discuss a stock price model with Heston type stochastic volatility and variance–gamma jumps in detail in Chapter 5. For a detailed discussion of Lévy–based models in finance, see [13] and [67].

1.4 Estimation Challenges and Bayesian Approach

In model building, adding new features to an existing model may seem to be an easy task at first sight. For instance, in previous sections, we discussed different models (with increasing complexity) that can better capture the observed properties of asset prices. Although these models give us a rich environment to work with, these models typically introduce many latent variables and yield to complicated likelihood functions. Therefore, the main challenge will be the estimation of these complex models. Most of the time, the complexity of these models prevent us to use least–squares or likelihood based estimation methods and necessitates a more advanced estimation method.

With the help of recent developments in Monte Carlo methods, simulation based Bayesian approach become a powerful alternative for estimating these complex models with latent variables. In a Bayesian setting, we first derive the posterior distribution for the set of any unknown parameter and latent variable. Then with the help of Monte Carlo techniques, we try to approximate this density using generated samples. Simulation based Bayesian approach encompasses two main estimation methods, namely Markov Chain Monte Carlo (MCMC) and particle filtering (a.k.a. Sequential Monte Carlo). MCMC algorithms allow us to sample from the posterior distribution based on constructing a Markov chain that has the posterior distribution as its equilibrium distribution. MCMC algorithms are widely used in estimating models with unknown parameters as well as latent variables. Particle filtering, on the other hand, assumes that the model parameters are known and approximates the conditional posterior of latent variables using sequential implementation of a certain type of importance sampling. Particle filtering is especially useful for filtering problems on non–linear non–Gaussian systems, where more standard algorithms such as Kalman filter fail to handle.

Although the MCMC algorithms give us a flexible method for estimation, the performance of the MCMC methods critically depend on finding good proposal distributions used in designing the Markov chains. If the proposal distributions explore the state space poorly and/or if highly correlated variables are updated independently, then the resulting MCMC estimates will be poor. And finding good high–dimensional proposals for the latent states, which typically have high dimensions, is not an easy task. Very recently, Particle MCMC (PMCMC) methods are proposed by [2] that aim to use particle filtering methods to build efficient high–dimensional proposals to be used within an MCMC setting. PMCMC approach combines two strands of simulation based Bayesian inference and offers a powerful approach for estimating complex

models.

In the literature there exist various papers that use MCMC algorithms for models with stochastic volatility and jumps. For instance [40] and [45] develop MCMC algorithms for an autoregressive stochastic volatility model. [41] provides algorithms for stochastic volatility models with leverage and fat tails. Furthermore [49] uses MCMC algorithms for different stock price models with stochastic volatility plus Poisson, variance–gamma and log–stable based jumps.

On the other hand, although PMCMC approach offers a compelling alternative to traditional MCMC and attracted a huge interest from the academic society, it is relatively new and the literature on PMCMC for stochastic volatility and jump models is not voluminous. In their original paper, [2] develops PMCMC algorithms for a stochastic volatility model where the volatility process is driven a Lévy process. [39] uses PMCMC algorithm for Hull–White type stochastic volatility model. [21] and [33] develop PMCMC algorithms for different autoregressive stochastic volatility models. Therefore we aim to contribute to the existing literature by developing PMCMC algorithms for a complex model with stochastic volatility and jumps.

1.5 Structure of the Thesis

In this thesis, after discussing MCMC and particle filtering methods, we will focus on PMCMC approach and design efficient algorithms for a complex stock price model. The model includes stochastic volatility and jumps that are governed by a time changed Lévy process, and thus is quite flexible in its nature. The efficient PMCMC algorithms designed for this model constitute the main contribution of the thesis to the existing academic literature. The outline of the thesis is as follows: In chapter 2 we will introduce the Bayesian framework and MCMC approach for inference, and implement various MCMC algorithms for a simple stochastic volatility model. Chapter 3 is devoted to particle filtering where we compare different filtering algorithms and implement them for the basic model. We will introduce PMCMC approach in Chapter 4 where we discuss the theoretical foundations of PMCMC and different PMCMC samplers. In Chapter 5, we will propose a flexible model with stochastic volatility and Lévy jumps that can capture observed characteristics of stock prices. We will develop MCMC and PMCMC algorithms for the model and compare them in an empirical study on S&P500 index. Chapter 5 concludes the thesis.

CHAPTER 2

MARKOV CHAIN MONTE CARLO METHODS

In this chapter, Bayesian approach and MCMC methods will be introduced as an inference tool. A simple stochastic volatility model is used to illustrate ideas and algorithms. First we will introduce the SV model and discuss challenges in estimating the model. Then general framework for Bayesian approach will be introduced. Finally, MCMC methods with different algorithms will be presented.

2.1 A Simple Stochastic Volatility Example

To illustrate the main ideas and algorithms, we will use the autoregressive stochastic volatility (ARSV) model. In this model, log-volatility follows an Ornstein-Uhlenbeck process (i.e. a continuous-time AR(1)):

$$\begin{aligned} dS_t &= \mu S_t dt + e^{h_t/2} S_t dW_t && (2.1) \\ dh_t &= \tilde{\alpha}\left(\tilde{m} - h_t\right) dt + \gamma dB_t && (2.2) \end{aligned}$$

where the Brownian motions, W_t and B_t, are independent. First order Euler approximation over a unit time period gives the following discrete–time equivalent:

$$\begin{aligned} Y_t &= \log(S_t/S_{t-1}) = \mu + e^{h_t/2}\epsilon_t && (2.3) \\ h_t &= \alpha + \beta h_{t-1} + \gamma\eta_t && (2.4) \end{aligned}$$

where Y_t is the continuously compounded log–return, h_t is the log–volatility, ϵ_t and η_t are i.i.d. $N(0,1)$, $\alpha = \tilde{\alpha}\tilde{m}$, $\beta = 1-\tilde{\alpha}$ with $|\beta| < 1$ to make the process stationary. The discrete–time version of this model is first used by [72] and its statistical properties are analyzed by [73]. The model ensures the autoregressive mean reverting volatility with no leverage effect.

This model is also an example of a non–linear Gaussian state–space model. For this, we have the observation equation given as (2.3) and state evolution equation given as (2.4). As we will discuss in later chapters, in the most general case, asset price models can be stated as non–linear non–Gaussian state–space models.

2.2 Bayesian Approach to Inference

Now consider we want to estimate ARSV model. We observe log–returns, $Y = \{Y_t\}_{t=1}^T$, and we want to infer latent log–volatility, $h = \{h_t\}_{t=1}^T$ and the values of parameters, $\Theta = (\mu, \alpha, \beta, \gamma^2)$. To establish a consistent notation, let $X = \{X_t\}_{t=1}^T$ be the collection all latent variables (i.e. states) in the model. In our simple model, the only latent variable is the log–volatility, thus $X_t = h_t$.

Bayesian approach assumes that anything we observe is fixed in value and for any other unobserved variable, the uncertainty caused by not observing them is encoded explicitly in a probability distribution[1]. For instance, in our problem, log–returns, Y, is observed, thus it is fixed. But for the value of parameters and the latent variables we make probabilistic descriptions. Before observing some data, these descriptions are called *prior distributions*, and after observing some data these descriptions are called *posterior distributions*. For instance, in our case, the posterior distribution of parameters and latent variables given observed log–returns, $p(X, \Theta|Y)$, captures our probabilistic description about X and Θ after we observed the data. Using the Bayes rule, we can express the posterior as:

$$\begin{aligned} p(X, \Theta|Y) &\propto p(Y|X, \Theta)p(X, \Theta) \\ &\propto p(Y|X, \Theta)p(X|\Theta)p(\Theta) \end{aligned} \tag{2.5}$$

The first term on the right hand side of (2.5) is the full-information likelihood, the second and third terms jointly express our *prior* beliefs and non–sample information about the parameters and the latent variables.

ARSV model implies that:

$$Y_t|h_t, \Theta \sim N(\mu, e^{h_t}) \tag{2.6}$$

$$h_t|h_{t-1}, \Theta \sim N(\alpha + \beta h_{t-1}, \gamma^2) \tag{2.7}$$

Additionally, to initialize the log–volatility we assume a Normal prior for the initial[2] log–volatility, i.e. $h_0 \sim N(m_0, C_0)$.

Thus the full–information likelihood, the first component of the posterior, is[3]:

$$L(X, \Theta|Y) \propto p(Y|X, \Theta) = \prod_{t=1}^{T} N(Y_t|\mu, e^{h_t}) \tag{2.8}$$

Note that marginal likelihood for parameters is given as follows and is much complicated than the full-information likelihood:

$$L(\Theta|Y) \propto p(Y|\Theta) = \int p(Y, h|\Theta)dh$$

[1] For a more general discussion on Bayesian approach please refer to [24], [64], [11] and [30].

[2] Log–volatility process can also be initialized by directly assuming a prior on h_1.

[3] For ease of notation we will use $N(x|\mu, \sigma^2)$ to represent the density for the random variable $X \sim N(\mu, \sigma^2)$.

$$
\begin{aligned}
&= \int p(Y|h,\Theta)p(h|\Theta)dh \\
&= \int \left[\prod_{t=1}^{T} p(Y_t|\Theta,h_t)\right] p(h_0) \left[\prod_{t=1}^{T} p(h_t|h_{t-1},\Theta)\right] dh \\
&= \int \left[\prod_{t=1}^{T} N(Y_t|\mu,e^{h_t})\right] \\
&\qquad N(h_0|m_0,C_0)\left[\prod_{t=1}^{T} N(h_t|\alpha+\beta h_{t-1},\gamma^2)\right] dh \qquad (2.9)
\end{aligned}
$$

Note that since the first part involves e^{h_t} terms, this marginal likelihood does not admit a closed–form solution. This complicates the implementation of a classical maximum likelihood approach.

The second component of the posterior distribution is the prior distribution for parameters and latent variables. Any non–sample information can be incorporated through prior. This may include prior beliefs about a parameter obtained from previous studies. *Uninformative* or *diffuse* prior distributions can be used if there isn't any non–sample information. Priors can also be used to impose certain statistical features like imposing stationarity, or ruling out near unit-root behavior. For analytical tractability purposes, it is common to use *conjugate* prior distributions.

In our example, we already assumed a prior for the initial log–volatility, h_0. Additionally, we also assume a prior for the parameters, $p(\Theta) = p(\mu,\alpha,\beta,\gamma^2)$. Assume now, as an example, we use the following conjugate priors:

$$
\begin{aligned}
p(\gamma^2) &= IG(\gamma^2|a_0,A_0) \\
p(\alpha,\beta|\gamma^2) &= N_2(b_0,\gamma^2 B_0)\mathbb{1}_{[-1<\beta<1]} \\
p(\mu) &= N(\mu|d_0,D_0)
\end{aligned}
$$

where IG represents the density function for inverse gamma[4] distribution, N_2 represents the density function for the bivariate Normal distribution and a_0, A_0, b_0, B_0, d_0 and D_0 are *hyperparameters* to be specified. A priori, we assume that, α and β jointly have a bivariate Normal distribution with covariance matrix that depends on γ^2. Our choice of Normal–Gamma priors for α,β and γ^2 is a common choice in Bayesian linear models. Having a joint prior for α and β is motivated from the fact that the form of marginal likelihood for parameters yield a high posterior correlation between these two parameters (see p. 92 of [29] for a discussion). We also truncate the prior for β to make the log–volatility process stationary.

By combining state equation with priors for Θ and h_0, we can obtain joint prior for

[4] Gamma distribution has the density $Ga(x|\alpha,\beta) = \frac{1}{\beta^\alpha\Gamma(\alpha)}x^{\alpha-1}\exp(-x/\beta), 0 < x,\alpha,\beta$ and inverse gamma distribution has the density $IG(x|\alpha,\beta) = \frac{\beta^\alpha}{\Gamma(\alpha)}x^{-\alpha-1}\exp(-\beta/x), 0 < x,\alpha,\beta$. If $X \sim IG(\alpha,\beta)$ then $Y = 1/X \sim Ga(\alpha,1/\beta)$.

latent variables and parameters:

$$\begin{aligned} p(X,\Theta) &= p(\Theta)p(h_0)\prod_{t=1}^{T} p(h_t|h_{t-1},\Theta) \\ &= p(\mu)p(\alpha,\beta|\gamma^2)p(\gamma^2)p(h_0)\prod_{t=1}^{T} p(h_t|h_{t-1},\Theta) \\ &= N(\mu|d_0,D_0)N_2(\alpha,\beta|b_0,\gamma^2 B_0)\mathbb{1}_{[-1<\beta<1]}IG(\gamma^2|a_0,A_0) \\ &\quad N(h_0|m_0,C_0)\prod_{t=1}^{T} N(h_t|\alpha+\beta h_{t-1},\gamma^2) \end{aligned}$$

Finally, combining this prior with the full information likelihood gives us the posterior:

$$\begin{aligned} p(X,\Theta|Y) &\propto p(Y|X,\Theta)p(X,\Theta) \\ &\propto \left[\prod_{t=1}^{T} N(Y_t|\mu,e^{h_t})\right] N(\mu|d_0,D_0)N_2(\alpha,\beta|b_0,\gamma^2 B_0)\mathbb{1}_{[-1<\beta<1]}IG(\gamma^2|a_0,A_0) \\ &\quad N(h_0|m_0,C_0)\prod_{t=1}^{T} N(h_t|\alpha+\beta h_{t-1},\gamma^2) \end{aligned} \tag{2.10}$$

For latent variables, depending on the conditioning information used, we may focus on different posterior distributions, given as follows:

$$\begin{aligned} \text{Smoothing} &: p(X_t|Y^T)\ t=1,2,\ldots,T \\ \text{Filtering} &: p(X_t|Y^t)\ t=1,2,\ldots,T \\ \text{Forecasting} &: p(X_{t+1}|Y^t)\ t=1,2,\ldots,T \end{aligned}$$

where Y^t denotes the observed variables up to time t, i.e. $Y^t = Y_1, Y_2, \ldots, Y_{t-1}, Y_t$. The smoothing problem is a static problem, solved once using all the data, but the filtering and forecasting problems are inherently sequential. We will deal with smoothing and filtering problems in more detail in the next chapters.

2.3 Markov Chain Monte Carlo Methods

2.3.1 General Setup

The main object for the Bayesian inference is the posterior obtained in (2.10). Optimal estimators (in the mean–squared sense) can be obtained as the mean of marginal posteriors that can be derived from this joint posterior. However, the main problem is that, this posterior is quite complex, very high dimensional ($T+4$ in our case), and does not correspond to a standard distribution. Therefore we need to use sampling methods for approximating this posterior. However because of its complexity, direct sampling

methods (e.g. inverse transform, auxiliary variables, rejection sampling) cannot be used in this case and importance sampling is very inefficient.

MCMC methods[5] are powerful sampling methods to generate samples from complex high–dimensional distributions. In an MCMC method, we begin with a target density, $\pi(\cdot)$, that we want to sample from. In our case, the target density is the joint posterior, i.e. $\pi(X, \Theta) = p(X, \Theta|Y)$. Then we design a Markov chain[6] on the space of (X, Θ), $\left\{X^{(g)}, \Theta^{(g)}\right\}_{g=0}^{G}$, where the limiting distribution of the chain is our target distribution. These samples do not form an i.i.d. sequence from the target distribution, however the marginal distribution of these samples *converges* to the target distribution, as $G \to \infty$.

Probabilistic structure of any Markov chain can be fully specified by specifying the distribution of the initial states (for this we specify the value of the initial states, i.e. use a degenerate distribution), $p(X^{(0)}, \Theta^{(0)})$, and for subsequent steps (i.e. $g = 1, 2, \ldots$) specifying a *transition distribution*, $p(X^{(g+1)}, \Theta^{(g+1)}|X^{(g)}, \Theta^{(g)})$ which determines the evolution of the chain.

The critical point here is that we should select this transition distribution in a way that the Markov chain will have our target distribution as its invariant (i.e. limiting) distribution. Metropolis–Hastings algorithm [55, 35] is a general method to achieve this. In this algorithm, we determine a proposal density, $q(X^{(g+1)}, \Theta^{(g+1)}|X^{(g)}, \Theta^{(g)})$, which is different from the transition probability. We should select proposal density such that we can easily sample from it. Then by sampling from this proposal density and using a certain type of accept/reject algorithm, a sequence of samples can be generated from the Markov chain which at the end have the right transition probability $p(X^{(g+1)}, \Theta^{(g+1)}|X^{(g)}, \Theta^{(g)})$ and thus have right invariant distribution. A full example[7] for a generic Metropolis–Hastings algorithm is given in Algorithm 2.1.

Metropolis–Hastings acceptance probability given in (2.11) is calculated using the ratios of target distribution (i.e. posterior) and the proposal distribution, evaluated at two different points, $(X^{(can)}, \Theta^{(can)})$ and $(X^{(g-1)}, \Theta^{(g-1)})$. Thus although we do not need to sample directly from the target distribution, we should be able to evaluate it at certain points. Additionally since the posterior is included in the formula as a ratio, we can still use this MCMC algorithm even if we don't know the integrating constant for the posterior.

The specific form of Metropolis–Hastings acceptance probability ensures that our target distribution is always the invariant distribution for our Markov chain. To see this we first need the following definition and theorem:

Definition 2.1. A Markov chain with transition distribution $p(z, z')$ satisfies the detailed balance condition if there exist a function f satisfying $f(z)p(z, z') = f(z')p(z', z)$.

Theorem 2.1. *If a Markov chain satisfies the detailed balance condition with respect*

[5] For a detailed treatment of MCMC methods we refer the reader to [65], [23] and [29]. Additionally, for MCMC implementations in financial modeling, see [42].

[6] A Markov chain is a stochastic process where the probability of the next state depends only on the current state.

[7] For more detailed MCMC algorithms and empirical implementations, see [77].

Algorithm 2.1 A Generic Metropolis–Hastings Algorithm

1: Initialize $(X^{(0)}, \Theta^{(0)})$
2: **for** $g = 1$ to G **do**
3: Sample $(X^{(can)}, \Theta^{(can)}) \sim q(X^{(can)}, \Theta^{(can)} | X^{(g-1)}, \Theta^{(g-1)})$
4: Calculate acceptance probability as:

$$\alpha\left((X^{(g-1)}, \Theta^{(g-1)}), (X^{(can)}, \Theta^{(can)})\right) = min\left\{1, \frac{\pi(X^{(can)}, \Theta^{(can)})}{\pi(X^{(g-1)}, \Theta^{(g-1)})} \frac{q(X^{(g-1)}, \Theta^{(g-1)} | X^{(can)}, \Theta^{(can)})}{q(X^{(can)}, \Theta^{(can)} | X^{(g-1)}, \Theta^{(g-1)})}\right\} \quad (2.11)$$

5: Sample $u \sim U(0, 1)$
6: **if** $u < \alpha\left((X^{(g-1)}, \Theta^{(g-1)}), (X^{(can)}, \Theta^{(can)})\right)$ **then**
7: $(X^{(g)}, \Theta^{(g)}) = (X^{(can)}, \Theta^{(can)})$
8: **else**
9: $(X^{(g)}, \Theta^{(g)}) = (X^{(g-1)}, \Theta^{(g-1)})$
10: **end if**
11: **end for**

to f where f is a density function, then f is the invariant distribution of the chain.

Proof. See [65, p. 230]. □

Now we can prove our claim:

Theorem 2.2. *Markov chain generated by Metropolis–Hastings algorithm satisfies the detailed balance condition with respect to joint posterior, $p(X, \Theta|Y)$, and thus this joint posterior is the invariant distribution of the chain.*

Proof. Let $z = (X^{(g)}, \Theta^{(g)})$ and $z' = (X^{(g+1)}, \Theta^{(g+1)})$ denote the state of the chain at g^{th} and $g + 1^{th}$ step. First note that the transition distribution of the chain can be obtained as:

$$p(z'|z) = q(z'|z)\alpha(z, z') + \mathbb{1}_{[z=z']} \int q(\tilde{z}|z)[1 - \alpha(z, \tilde{z})]d\tilde{z}$$

where the first component represents the case that a different state is proposed and accepted while the second component represents the case that the new proposed state is equal to the current state or a different state is proposed but rejected.

Note that detailed balance condition is always satisfied if chain does not move, i.e. if $z = z'$ then $f(z)p(z, z') = f(z')p(z', z)$. Thus we only need to show that detailed balance condition is satisfied (with respect to joint posterior $\pi(X, \Theta) = p(X, \Theta|Y)$) when a different state is proposed and accepted.

$$\begin{aligned} \pi(z)p(z'|z) &= \pi(z)q(z'|z)\alpha(z, z') \\ &= \pi(z)q(z'|z)min\left\{1, \frac{\pi(z')q(z|z')}{\pi(z)q(z'|z)}\right\} \end{aligned}$$

$$
\begin{aligned}
&= min\left\{\pi(z)q(z'|z), \pi(z')q(z|z')\right\} \\
&= \pi(z')q(z|z')min\left\{1, \frac{\pi(z)q(z'|z)}{\pi(z')q(z|z')}\right\} \\
&= \pi(z')q(z|z')\alpha(z', z) \\
&= \pi(z')p(z|z')
\end{aligned}
$$

□

2.3.2 Gibbs Sampler

The *Metropolis–Hastings algorithm* is a generic framework that encompasses different samplers. One special sampler is the *Gibbs sampler*, which uses the results of Hammersley–Clifford Theorem [34].

Theorem 2.3. *The joint distribution associated with the conditional densities* $p_1(\cdot|\cdot)$ *and* $p_2(\cdot|\cdot)$ *has the density:*

$$p(x_1, x_2) = \frac{p_2(x_2|x_1)}{\int \frac{p_2(x_2|x_1)}{p_1(x_1|x_2)} dx_2}$$

if $\int \frac{p_2(y|x_1)}{p_1(x_1|y)} dy$ *exists.*

Proof. We can decompose any joint distribution using conditional and marginal distributions:

$$p(x_1, x_2) = p_1(x_1|x_2)p_2(x_2) = p_2(x_2|x_1)p_1(x_1)$$

Thus:

$$
\begin{aligned}
\frac{p_2(x_2|x_1)}{p_1(x_1|x_2)} &= \frac{p_2(x_2)}{p_1(x_1)} \\
\int \frac{p_2(x_2|x_1)}{p_1(x_1|x_2)} dx_2 &= \int \frac{p_2(x_2)}{p_1(x_1)} dx_2 = \frac{1}{p_1(x_1)} \\
p(x_1, x_2) &= p_2(x_2|x_1)p_1(x_1) = \frac{p_2(x_2|x_1)}{\int \frac{p_2(x_2|x_1)}{p_1(x_1|x_2)} dx_2}
\end{aligned}
$$

□

The Hammersley–Clifford theorem allows us to characterize a joint distribution by using its conditional distributions. Using this idea, we can characterize our target distribution, which is the joint posterior $p(X, \Theta|Y)$, by using the conditional distributions given as $p(X|\Theta, Y)$ and $p(\Theta|X, Y)$. In Gibbs sampling, we generate samples from the joint distribution by sequentially sampling from the full conditional distributions.

This method allows us to reduce the dimension of the sampling problem. Sometimes it is also possible to apply Hammersley–Clifford theorem more than once to further get lower dimensional sampling.

In Gibbs sampling, after obtaining lower dimensional conditional distributions, we can use any sampling method to sample from these distributions. If direct sampling methods are available for the distributions, then the algorithm is called a Gibbs algorithm. On the other hand, if direct sampling methods are not available for the full conditionals, then we should use other Metropolis–Hastings samplers to sample from full conditionals. In this case the algorithm becomes a hybrid one, sometimes called as Metropolis–Hastings within Gibbs algorithm.

In Gibbs sampling, the acceptance probability is always one. This means that the chain always move, i.e. proposed values are always accepted. To show this, assume that we are using $p(X|\Theta, Y)$ and $p(\Theta|X, Y)$ for sampling from the joint posterior. Then, given that the current value of Markov chain is $(X^{(g)}, \Theta^{(g)})$, Gibbs sampling first proposes $(X^{(can)}, \Theta^{(g)})$ where $X^{(can)} \sim p(X^{(can)}|\Theta^{(g)}, Y) = \frac{\pi(X^{(can)}, \Theta^{(g)})}{p(\Theta^{(g)})}$. Thus the acceptance probability is:

$$\alpha\left((X^{(g)}, \Theta^{(g)}), (X^{(can)}, \Theta^{(g)})\right) = min\left\{1, \frac{\pi(X^{(can)}, \Theta^{(g)})}{\pi(X^{(g)}, \Theta^{(g)})} \frac{\pi(X^{(g)}, \Theta^{(g)})/p(\Theta^{(g)})}{\pi(X^{(can)}, \Theta^{(g)})/p(\Theta^{(g)})}\right\} = 1 \quad (2.12)$$

For ARSV model a deterministic scan[8] Gibbs algorithm is given in Algorithm 2.2. The algorithm updates one full conditional at each step. One exception is for α and β. Since the model yields high posterior correlations of these parameters, updating these individually is an inefficient method [29, p.90]. Therefore these variables are jointly updated[9].

The full conditionals for ARSV model can be obtained using the posterior given in (2.10). Any full conditional distribution is always proportional to the joint posterior. Therefore, by treating any irrelevant multiplicative term in joint posterior as a constant, we can obtain the full conditionals. The full conditionals are given as follows (details of the derivations are not shown):

$$\begin{aligned} p(\mu|X, Y) &\propto \left[\prod_{t=1}^{T} N(Y_t|\mu, e^{h_t})\right] N(\mu|d_0, D_0) \\ &\propto N(\mu|d_1, D_1) \\ &\quad d_1 = \frac{d_0/D_0 + \sum_{t=1}^{T} Y_t e^{-h_t}}{1/D_0 + \sum_{t=1}^{T} e^{-h_t}}\ ,\ D_1 = \frac{1}{1/D_0 + \sum_{t=1}^{T} e^{-h_t}} \\ p(\alpha, \beta, \gamma^2|X) &\propto N_2(\alpha, \beta|b_0, \gamma^2 B_0)\mathbb{1}_{[-1<\beta<1]} IG(\gamma^2|a_0, A_0) \end{aligned} \quad (2.13)$$

[8] The algorithm updates each variable in turn. It is also possible to design a random scan algorithm where the order of updating is chosen randomly.

[9] For instance jointly updating highly correlated variables (e.g. h_t) may improve the MCMC algorithm[29, p. 12]. [23] suggest that it is preferable to jointly update as much variable as possible, if we have a sampling method for this. Following sections include a discussion on this blocking approach.

Algorithm 2.2 Gibbs Algorithm

1: Initialize $(X^{(0)}, \Theta^{(0)})$
2: **for** $g = 1$ to G **do**
3: Sample $h_1^{(g)} \sim p(h_1^{(g)}|h_0^{(g-1)}, h_2^{(g-1)}, h_3^{(g-1)}, h_4^{(g-1)}, \ldots, h_T^{(g-1)}, \Theta^{(g-1)}, Y)$
4: Sample $h_2^{(g)} \sim p(h_2^{(g)}|h_0^{(g-1)}, h_1^{(g)}, h_3^{(g-1)}, h_4^{(g-1)}, \ldots, h_T^{(g-1)}, \Theta^{(g-1)}, Y)$
5: Sample $h_3^{(g)} \sim p(h_3^{(g)}|h_0^{(g-1)}, h_1^{(g)}, h_2^{(g)}, h_4^{(g-1)}, \ldots, h_T^{(g-1)}, \Theta^{(g-1)}, Y)$
6: $\vdots$
7: Sample $h_T^{(g)} \sim p(h_T^{(g)}|h_0^{(g-1)}, h_1^{(g)}, h_2^{(g)}, h_3^{(g)}, \ldots, h_{T-1}^{(g)}, \Theta^{(g-1)}, Y)$
8: Sample $h_0^{(g)} \sim p(h_0^{(g)}|h_1^{(g)}, h_2^{(g)}, h_3^{(g)}, \ldots, h_T^{(g)}, \Theta^{(g-1)}, Y)$
9: Sample $\mu^{(g)} \sim p(\mu^{(g)}|\alpha^{(g-1)}, \beta^{(g-1)}, \gamma^{2(g-1)}, X^{(g)}, Y)$
10: Sample $\gamma^{2(g)} \sim p(\gamma^{(g)}|\mu^{(g)}, \alpha^{(g-1)}, \beta^{(g-1)}, X^{(g)}, Y)$
11: Sample $(\alpha^{(g)}, \beta^{(g)}) \sim p((\alpha^{(g)}, \beta^{(g)})|\mu^{(g)}, \gamma^{2(g)}, X^{(g)}, Y)$
12: **end for**

$$
\begin{aligned}
& \prod_{t=1}^{T} N(h_t|\alpha + \beta h_{t-1}, \gamma^2) \\
\propto \; & N_2(\alpha, \beta|b_1, \gamma^2 B_1)\mathbb{1}_{[-1<\beta<1]} IG(\gamma^2|a_1, A_1) \\
& \Rightarrow p(\gamma^2|X) \propto IG(\gamma^2|a_1, A_1) \qquad (2.14) \\
& \Rightarrow p(\alpha, \beta|\gamma^2, X) \propto N_2(\alpha, \beta|b_1, \gamma^2 B_1)\mathbb{1}_{[-1<\beta<1]} \qquad (2.15) \\
& \tilde{X} = \begin{bmatrix} 1 & h_0 \\ \vdots & \vdots \\ 1 & h_{T-1} \end{bmatrix}, \; \tilde{Y} = \begin{bmatrix} h_1 \\ \vdots \\ h_T \end{bmatrix} \\
& B_1 = \left[B_0^{-1} + \tilde{X}'\tilde{X}\right]^{-1}, \; b_1 = B_1\left[B_0^{-1} b_0 + \tilde{X}'\tilde{Y}\right] \\
& a_1 = a_0 + T/2, \; A_1 = A_0 + 0.5\left[\tilde{Y}'\tilde{Y} + b_0' B_0^{-1} b_0 - b_1' B_1^{-1} b_1\right] \\
p(h_0|h_1, \Theta) \; \propto \; & N(h_0|m_0, C_0) N(h_1|\alpha + \beta h_0, \gamma^2) \\
\propto \; & N(h_0|m_0', C_0') \qquad (2.16) \\
& m_0' = \frac{m_0/C_0 + \beta(h_1 - \alpha)/\gamma^2}{1/C_0 + \beta^2/\gamma^2}, \; C_0' = \frac{1}{1/C_0 + \beta^2/\gamma^2}
\end{aligned}
$$

Since we used conjugate priors for the initial latent volatility and all parameters, we obtained standard distributions for the full conditionals. However, this is not true for other volatility states. The full conditional for the latent volatility states is given as:

$$
\begin{aligned}
p(h_t|h_{t-1}, h_{t+1}, \Theta, Y_t) \; \propto \; & N(Y_t|\mu, e^{h_t}) N(h_t|\alpha + \beta h_{t-1}, \gamma^2) N(h_{t+1}|\alpha + \beta h_t, \gamma^2) \\
\propto \; & N(Y_t|\mu, e^{h_t}) N(h_t|m_t, C_t) \qquad (2.17) \\
& m_t = \frac{\alpha(1-\beta) + \beta(h_{t-1} + h_{t+1})}{1 + \beta^2}, \; C_t = \frac{\gamma^2}{1 + \beta^2}
\end{aligned}
$$

and for $t = T$:

$$
p(h_T|h_{T-1}, \Theta, Y_T) \; \propto \; N(Y_T|\mu, e^{h_T}) N(h_T|\alpha + \beta h_{T-1}, \gamma^2)
$$

$$\begin{aligned} \propto \quad & N(Y_T|\mu, e^{h_T})N(h_T|m_T, C_T) \\ & m_T = \alpha + \beta h_{T-1}\ ,\ C_T = \gamma^2 \end{aligned} \tag{2.18}$$

Because of e^{h_t} in the variance term, full conditionals do not correspond to standard distributions. For non–standard distributions, common sampling algorithms include ratio–of–uniforms method, accept–reject method and slice sampling (see p.78, [29]).

In simple rejection method, we need an *envelope function* $E(\cdot)$ of the target distribution such that $E(h_t) \geq \pi(h_t), \forall h_t$. Then we obtain a density proportional to $E(\cdot)$, sample from it and accept the sample with probability $\pi(h_t)/E(h_t)$. This procedure is repeated until a draw is accepted. At the end, accepted values form an independent sample from the target density π. In this method, it is important to have the E to be close to π in order to increase acceptance probability and hence computational efficiency.

Therefore the critical issue with this approach is to find a tight envelope function. Fortunately, efficient envelope construction methods are available for log–concave densties[10]. Using tangents evaluated at certain points, [28] proposed an efficient method of finding an envelope $\log E(\cdot)$ to logarithm of a target density, $\log \pi(\cdot)$. The full conditional, $\pi(h_t)$ we obtained in (2.17) is also a log–concave density. The algorithm begins with selecting a number of points and tangents to $\log \pi$ are constructed at these points. Because of concavity, these tangent lines will always form an envelope to the log target density. With our full conditional with selected parameters, this idea is illustrated in Figure 2.1. In this case, the envelope E is piecewise exponential which can be easily integrated and sampled from. [28] also proposed an adaptive version called *adaptive rejection sampling (ARS)*. When a candidate value h^{can} is sampled from the envelope, in order to complete the rejection step, $\pi(h^{can})$ should also be evaluated. At this stage, the selected point $(h^{can}, \pi(h^{can}))$ can be used to update the envelope. This adaptive approach improves the efficiency by obtaining a tighter envelope at each step. The details of the algorithm can be found in [28] and section 2.4.2 of [65].

Using ARS, now we can fully specify our Gibbs algorithm for ARSV model. The steps are given in Algorithm 2.3.

Example 2.1. We simulate 500 data points for latent volatility and logarithmic returns using parameter values $\mu = 0$, $\alpha = -0.4$, $\beta = 0.95$ and $\gamma^2 = 0.04$. Results are shown in Figure 2.2. To infer the values of parameters and the latent log–volatility, we implement Algorithm 2.3 with 10000 iterations. We use extremely diffusive priors with hyperparameters $a_0 = 5, A_0 = 0.1, b_0 = (0,0), B_0 = I_2, m_0 = 0, C_0 = 100$, where I_2 is the two dimensional identity matrix. The chain is initialized at $\alpha = -0.5$, $\beta = 0.5$, $\gamma^2 = 0.01$ and $h_t^{(0)}, t = 0, 1, \ldots, T$ equal to logarithm of sample variance. In obtaining estimates, we ignore the first 5000 iterations[11].

Estimated posterior quantities are given in Table 2.1. Note that, in a simulation experiment, due to sampling variability, the maximum likelihood estimates (MLE) of the parameters can happen to be different than the true values. Therefore it is better to compare the MCMC estimates with the MLE rather than the true values. MLE are

[10] A density $f(x)$ is log–concave if $\frac{d^2 \log f(x)}{dx^2} \leq 0$.

[11] This initial period, in which the samples are ignored, is called burn–in period.

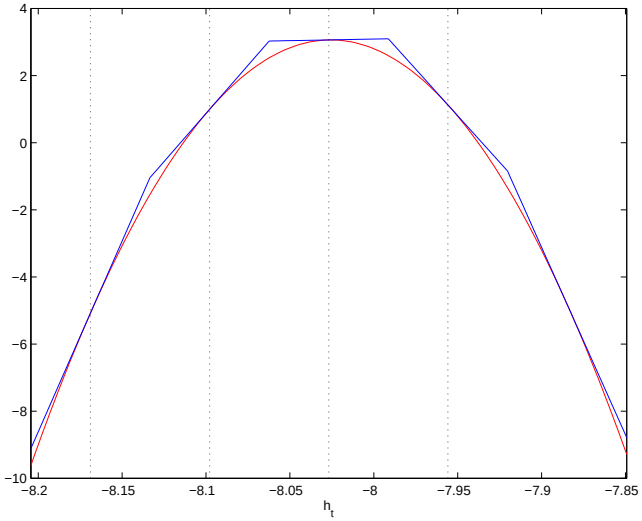

Figure 2.1: Envelope Function for Full Conditional. Red line represents the log of full conditional, $\pi(h_t)$. Blue line represents log of envelope function $E(h_t)$. The dotted lines represent the points at which tangents are drawn.

also given in the first line of Table 2.1. Estimation results are quite encouraging since posterior means are quite close to the MLE values and MLE values are included in the 90% credible intervals.

Table 2.1: Parameter Estimates from Gibbs Sampler

	α	β	γ^2
True Values (MLE)	-0.4070	0.9495	0.0405
Mean	-0.4768	0.9412	0.0356
Mode	-0.4607	0.9432	0.0337
Standard Deviation	0.1857	0.0230	0.0127
5th Percentile	-0.8101	0.8995	0.0189
95th Percentile	-0.2075	0.9743	0.0604

More detailed results are given in Figure 2.3. The top line graphs are the *trace plots* for the parameters which show the values visited by the Markov chain. The chains (especially for α and β) quickly converges to the true MLE values and oscillate around these values thereafter. The second line shows the autocorrelations for parameter chains. Normally smaller autocorrelations in MCMC chains improves the estimation results. The third line shows the posterior distributions obtained from MCMC samples (after ignoring burn–in period). The red vertical lines are the MLE values. These graphs also show that we obtained reasonable estimates for the parameters. The last line of graphs show the results for latent volatility, h_t. The first one is the trace plot for h_{250}, as an example. The second one shows the true and estimated (also with 90% credible interval) standard deviations, i.e. $\sqrt{e^{h_t}}$. The final graph shows the autocorrelations of h_t for all $t = 1, \ldots, T$. Volatilities (especially adjacent ones) are highly correlated. And since we are updating these highly correlated variables once at a time, the resulting chains

Algorithm 2.3 Gibbs Algorithm with ARS for Log–Volatility

Initialize $(X^{(0)}, \Theta^{(0)})$
for $g = 1$ to G **do**
  for $t = 1$ to T **do**
    Select some grid points from the support of $p(h_t)$.
    Construct the envelope $\log E(h_t)$ from the tangent lines to $\log p(h_t)$ evaluated at initial grid points.
    repeat
      Sample $h_t^{(can)} \propto E(h_t)$
      Accept it, $h_t^{(g)} = h_t^{(can)}$, with probability $min\left\{1, p(h_t^{(can)})/E(h_t^{(can)})\right\}$
      Update the envelope using the new point $(h_t^{(can)}, E(h_t^{(can)}))$
    until A draw is accepted
  end for
  Sample $h_0^{(g)} \sim N(m_0', C_0')$
  Sample $\mu^{(g)} \sim N(d_1, D_1)$
  Sample $\gamma^{2(g)} \sim IG(a_1, A_1)$
  Sample $(\alpha^{(g)}, \beta^{(g)}) \sim N_2(b_1, \gamma^{2(g)} B_1)$
end for

have high autocorrelations. Though, the decay in autocorrelations is quick.

2.3.3 Random Walk (within Gibbs) Algorithm

Gibbs sampler is only a very special type of Metropolis–Hastings algorithm. Metropolis–Hastings is a generic algorithm and (given some minor conditions) it is possible to use any proposal distribution in the algorithm. Two popular Metropolis–Hastings samplers are *random walk sampler* and *independence sampler*. In random walk sampler, the sampled values depends only on the current chain value while in independence sampler, new samples are generated independent of the current chain value.

In previous section, we used a componentwise approach where decompose the joint posterior into smaller components and update the components in a deterministic or random way. This approach can also be applied using other Metropolis–Hastings samplers. In such a setting, we design each update in such a way that all have the same joint posterior as their invariant distribution. Thus the combination of these updates also forms a Markov chain with same invariant distribution[12]. In such a componentwise approach, if we combine Gibbs sampler with other Metropolis–Hastings samplers, then such an algorithm is sometimes called a Metropolis–Hastings within Gibbs, Gibbs within Metropolis–Hastings or simply a hybrid algorithm (p.211, [23]).

For our ARSV model, full conditional distribution for the log–volatility (2.17), is not a standard distribution. For updating the volatility states, we will use random walk and independence samplers in this and next sections. Updates for parameters will be same as we did in Gibbs sampler (Algorithm 2.3).

[12] Componentwise updating with a deterministic (random) scan will generate a transition probability which is a convolution (mixture) of component transition probabilities. In both cases the resulting transition probability have the correct invariant distribution. For more details see section 6.4.1 of [23] and Section 10.3.2 of [65].

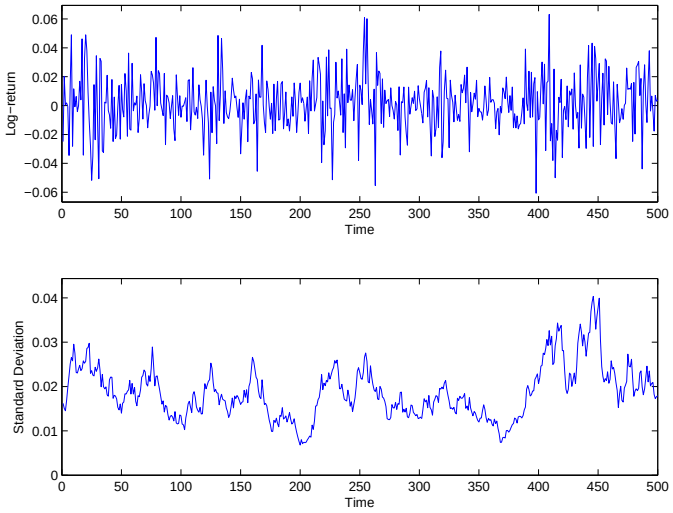

Figure 2.2: Simulated Log–Returns and Standard Deviations

Random walk sampler assumes a random walk model to derive the proposal distribution. For instance, for sampling h_t in our model, we propose $h_t^{(can)} = h_t^{(g-1)} + \zeta$ where ζ has some distribution f. By this chain explores *locally* the neighborhood of the current value, $h_t^{(g-1)}$. Using our previous notation, the proposal distribution becomes $q(h_t^{(can)}|h_t^{(g-1)}) = f(h_t^{(can)} - h_t^{(g-1)})$.

Typically, the distribution f is chosen such that it is symmetric around zero, i.e. $f(x) = f(-x)$ [55]. Most common choices are uniform, normal and student t distributions. When f is symmetric around zero, acceptance probability becomes:

$$\begin{aligned} \alpha(h_t^{(g-1)}, h_t^{(can)}) &= min\left\{1, \frac{p(h_t^{(can)})f(h_t^{(g-1)} - h_t^{(can)})}{p(h_t^{(g-1)})f(h_t^{(can)} - h_t^{(g-1)})}\right\} \\ &= min\left\{1, \frac{p(h_t^{(can)})}{p(h_t^{(g-1)})}\right\} \end{aligned}$$

We can use a random walk sampler for sampling volatility in our ARSV model by replacing the lines 3 to 10 in Algorithm 2.3 with the following Algorithm 2.4. We used a Normal proposal with variance ϑ^2.

In designing a random walk sampler, the dispersion (e.g. variance) of the proposal $f(\cdot)$ has a critical role. Large variance values gives a better exploration of state space but may yield very small acceptance rates. On the other hand, small values may give high acceptance rates but the chain can only move points close to the current values. Therefore there is a trade–off between exploration of parameter space and average acceptance rate for random walk sampler. This means that trying to maximize acceptance probability is not an optimal solution[65, p. 295] and a balance between these should be aimed. Regarding the optimal acceptance rates, a value around 0.25 is suggested in the literature for some specific cases and widely employed as a rule of thumb. With

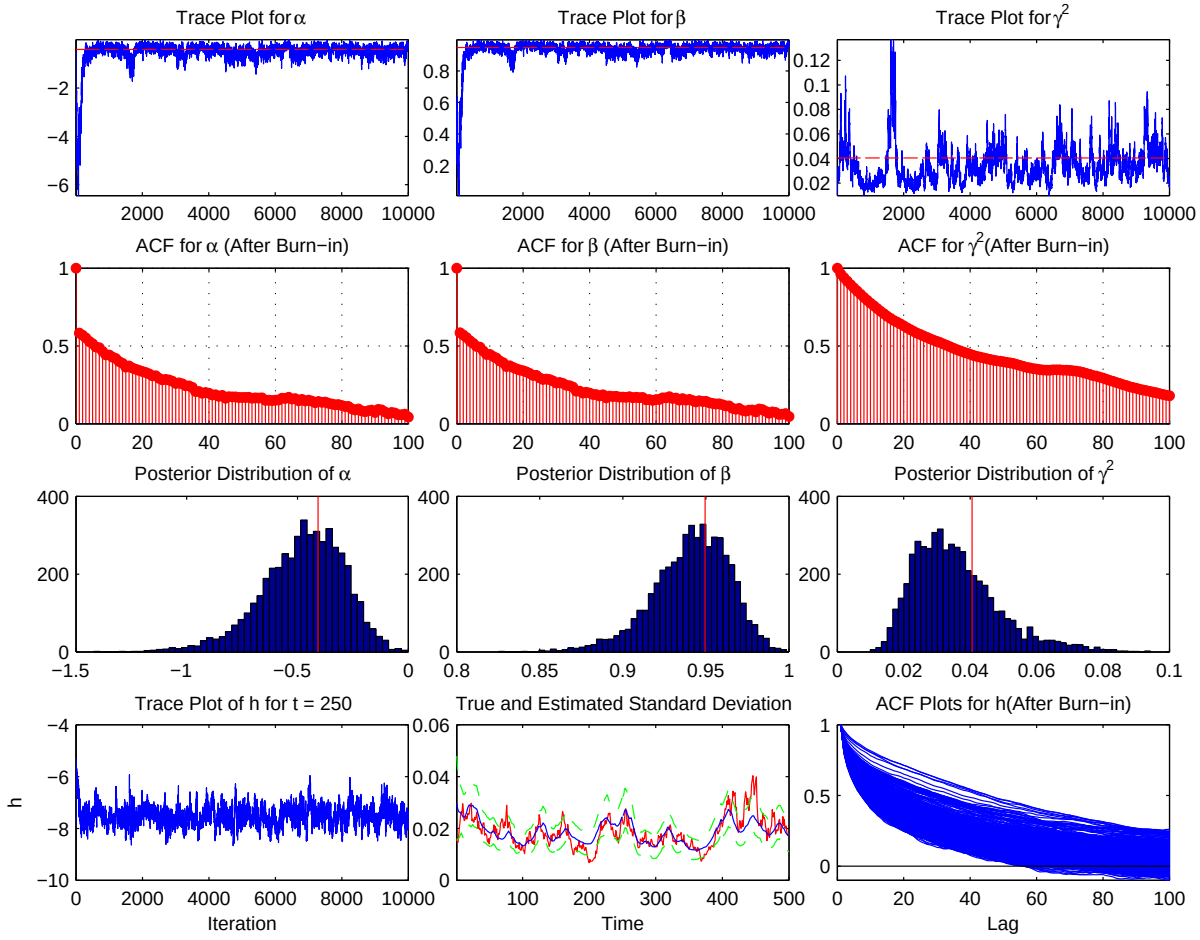

Figure 2.3: Estimation Results for Gibbs Sampler

this rule of thumb, Algorithm 2.4 can be fine tuned by changing the variance ϑ^2 to achieve an optimal average acceptance rate. Indeed, our tuning exercise also yields an optimal acceptance value around 0.25 as seen in Example 2.2 below.

Example 2.2. We performed 20 estimation studies with 20 different variance values and chain length of $G = 1000$. For each estimation step, we calculate average acceptance probability, root mean squared error (RMSE) as an overall measure of fit and average autocorrelation for h_t up to lag 100 over all $t = 1, 2, \ldots, T$. Our RMSE is defined as $RMSE = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(\sqrt{e^{\hat{h}_t}} - \sqrt{e^{h_t}})^2}$, where $\hat{h}_t$ is our Bayes estimate. We used $\vartheta^2 = 10^{-i}, i = -10, 9, \ldots, 8, 9$ as the variance. The results are shown in Figure 2.4. As seen from the graphs both average autocorrelations (an indicator for how good the mixing is) and RMSE (main indicator for the goodness of fit) are minimized when we have an average acceptance probability around 0.2. In our study, the optimum value for variance is found as $\vartheta^2 = 1$ with average acceptance probability of 25.44%, $RMSE = 0.0045$ and average autocorrelation of 0.2626.

Example 2.3. We implement Algorithm 2.4 with 30000 iterations. We use the same

Algorithm 2.4 MCMC Algorithm with Random Walk Sampler for Volatility (Replaces lines 3 to 10 in Algorithm 2.3)

1: **for** $t = 1$ to T **do**
2: Sample $h_t^{(can)} \sim N(h_t^{(g-1)}, \vartheta^2)$
3: Accept $h_t^{(g)} = h_t^{(can)}$ with probability:

$$\alpha(h_t^{(g-1)}, h_t^{(can)}) = min\left\{1, \frac{N(Y_t|\mu, e^{h_t^{(can)}})N(h_t^{(can)}|m_t, C_t)}{N(Y_t|\mu, e^{h_t^{(g-1)}})N(h_t^{(g-1)}|m_t, C_t)}\right\}$$

Otherwise set $h_t^{(g)} = h_t^{(g-1)}$
4: **end for**

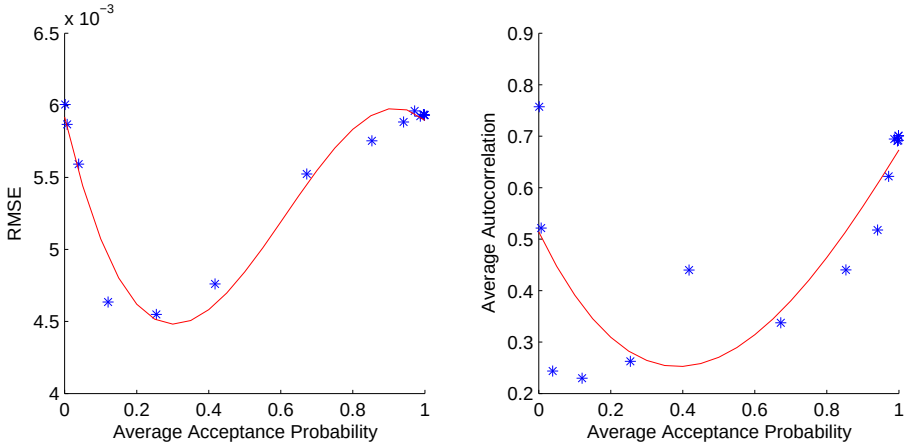

Figure 2.4: Fine Tuning for Random–Walk Sampler. Points show average autocorrelations and RMSE versus average acceptance probability. Lines represent fitted third degree polynomials.

values (log–returns, hyperparameters, etc.) used in Example 2.1. Burn–in period is equal to the half of the chain length. Estimated posterior quantities related to parameters are given in Table 2.2. Although credible intervals include the true values, when compared to Gibbs sampler (Table 2.1), the results are worse (i.e. higher credible intervals with point estimates more distant from true values) in random–walk sampler.

Reasons for this can be seen from the more detailed graphs given in Figure 2.5. As seen from the graphs, there is significant autocorrelation problem in this algorithm. The autocorrelations for the log–volatility (shown in the third graph in bottom line) are quite high and decaying very slowly. In random walk sampler, proposed value for $h_t^{(g)}$ depends on its own current value $h_t^{(g-1)}$ which is a source of autocorrelation. Additionally in random walk sampler there is significant probability that the chain does not move. Because of these two differences, our hybrid Gibbs–random walk algorithm produces higher correlations for volatility than the pure Gibbs algorithm (see Figure 2.3 for a comparison). And this increased autocorrelations in volatility

Table 2.2: Parameter Estimates from Hybrid Gibbs–Random Walk Algorithm

	α	β	γ^2
True Values (MLE)	-0.4070	0.9495	0.0405
Mean	-0.5766	0.9290	0.0452
Mode	-0.5316	0.9346	0.0416
Standard Deviation	0.2459	0.0305	0.0174
5th Percentile	-1.0644	0.8686	0.0250
95th Percentile	-0.2588	0.9681	0.0787

chains are transmitted to the parameter chains as well (shown in second line). These also support our previous comment that parameter uncertainty has relatively smaller effects on volatility estimation, but uncertainty in volatility may significantly worsen parameter estimates.

2.3.4 Independence (within Gibbs) Algorithm

Independence sampler [74] is a Metropolis–Hastings algorithm in which proposal distribution does not depend on the current state of the chain. For instance if we are using independence sampler for h_t, then the proposal will have the form $q(h_t^{(g)}|h_t^{(g-1)}) = f(h_t^{(g)})$ for some density function f. Then the Metropolis–Hastings acceptance probability reduces to:

$$\begin{aligned}
\alpha(h_t^{(g-1)}, h_t^{(g)}) &= min\left\{1, \frac{\pi(h_t^{(g)})q(h_t^{(g-1)}|h_t^{(g)})}{\pi(h_t^{(g-1)})q(h_t^{(g)}|h_t^{(g-1)})}\right\} \\
&= min\left\{1, \frac{\pi(h_t^{(g)})f(h_t^{(g-1)})}{\pi(h_t^{(g-1)})f(h_t^{(g)})}\right\} \\
&= min\left\{1, \frac{w(h_t^{(g)})}{w(h_t^{(g-1)})}\right\} \qquad (2.19)
\end{aligned}$$

where $\pi(h_t) = p(h_t|h_{t-1}, h_{t+1}, \Theta, Y_t)$ is the target distribution given in (2.17) and $w(h_t) = \pi(h_t)/f(h_t)$ is the ratio of target and proposal densities.

In independence sampler, proposal distribution f should be a good approximation to the target distribution and additionally proposal should have fatter tails than the target distribution. These conditions ensure that the ratio $w(h_t)$ is bounded and fairly stable [74]. In independence sampler, the convergence rate is positively related with the expected acceptance probability. Therefore, typically, once the functional form of the proposal is determined, free parameters of it are tuned to achieve the highest average acceptance rate.

Now we will use a hybrid Gibbs–independence algorithm for our model. First remem-

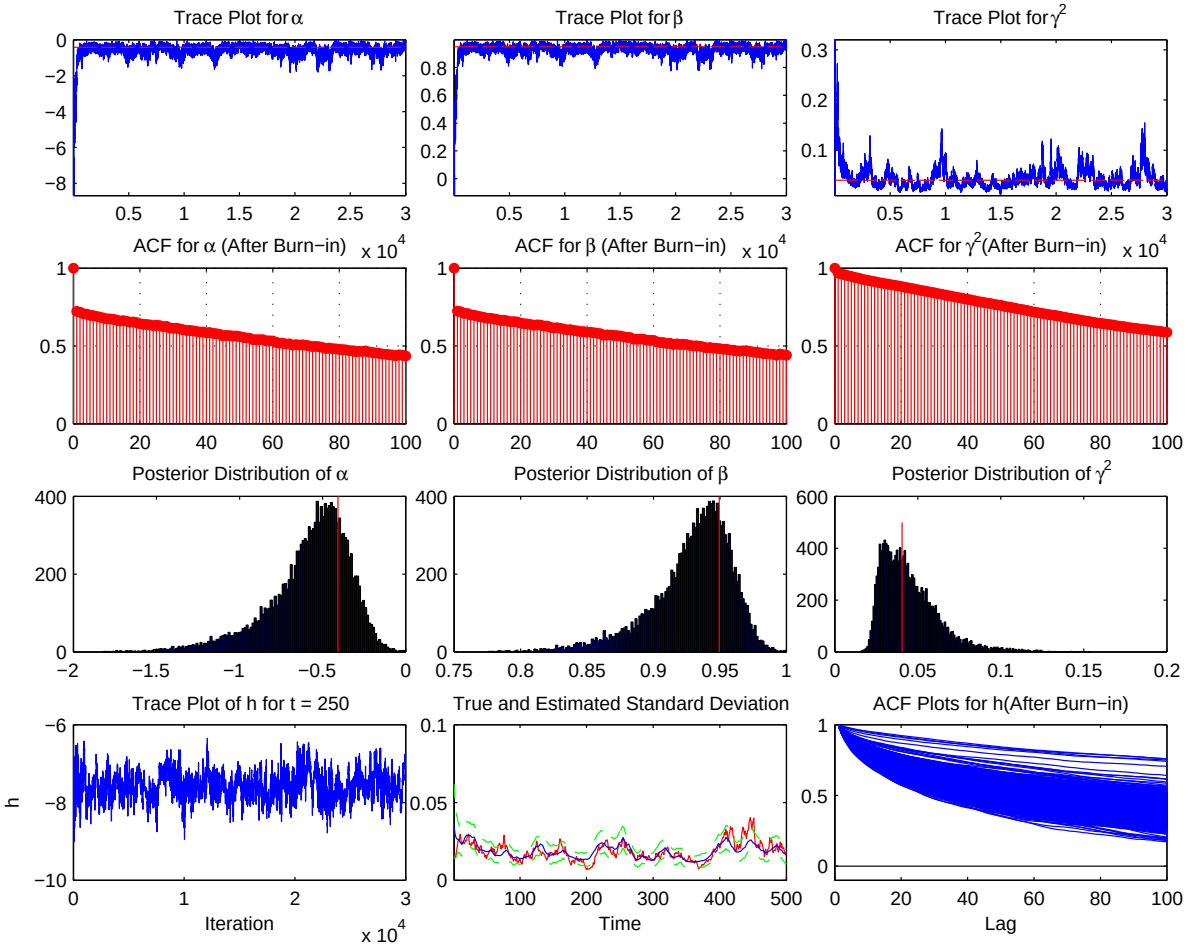

Figure 2.5: Estimation Results for Hybrid Gibbs–Random Walk Algorithm

ber that the full conditional for the log–volatility has the form:

$$p(h_t|h_{t-1}, h_{t+1}, \Theta, Y_t) \propto N(Y_t|\mu, e^{h_t})N(h_t|m_t, C_t)$$

The second component is Gaussian in h_t. However the first component is non–standard in h_t. The logarithm of the first component is:

$$\log N(Y_t|\mu, e^{h_t}) = \text{constant} - \frac{1}{2}h_t - \frac{(Y_t - \mu)^2}{2}\exp(-h_t)$$

[45] explored the fact that $\exp(-h_t)$ is a convex function and can be bounded by a function linear in h_t. Using the Taylor expansion of $\exp(-h_t)$ around m_t, i.e. $\exp(-h_t) \approx \exp(-m_t) - (h_t - m_t)\exp(-m_t)$, we can obtain an upper bound for this component:

$$\begin{aligned}\log N(Y_t|\mu, e^{h_t}) &\leq \text{constant} - \frac{1}{2}h_t - \frac{(Y_t - \mu)^2}{2}[\exp(-m_t) - (h_t - m_t)\exp(-m_t)] \\ &= \text{constant*} + \log g(h_t)\end{aligned}$$

where $g(h_t) = \exp\left\{-\frac{1}{2}h_t\left[1-(Y_t-\mu)^2\exp(-m_t)\right]\right\}$. Now we can use the following density as our proposal density:

$$\begin{aligned} q(h_t) &\propto g(h_t)N(h_t|m_t,C_t) \\ &\propto N(h_t|m_t',C_t) \end{aligned} \tag{2.20}$$

where $m_t' = m_t + 0.5C_t\left[(Y_t-\mu)^2\exp(-m_t)-1\right]$. This proposal density closely approximates the target density and because of convexity the ratio of (unnormalized) target to proposal is bounded, i.e. $p(h_t)/q(h_t) \leq c$ for some c. We can use this independence sampler for sampling volatility in our ARSV model by replacing the lines 3 to 10 in Algorithm 2.3 with the following Algorithm 2.5.

Algorithm 2.5 MCMC Algorithm with Independence Sampler for Volatility (Replaces lines 3 to 10 in Algorithm 2.3)

1: **for** $t = 1$ to T **do**
2: Sample $h_t^{(can)} \sim N(m_t', C_t)$
3: Accept $h_t^{(g)} = h_t^{(can)}$ with probability:

$$\alpha(h_t^{(g-1)}, h_t^{(can)}) = min\left\{1, \frac{N(Y_t|\mu,\exp(h_t^{(can)}))N(h_t^{(can)}|m_t,C_t)N(h_t^{(g-1)}|m_t',C_t)}{N(Y_t|\mu,\exp(h_t^{(g-1)}))N(h_t^{(g-1)}|m_t,C_t)N(h_t^{(can)}|m_t',C_t)}\right\}$$

Otherwise set $h_t^{(g)} = h_t^{(g-1)}$
4: **end for**

Example 2.4. We implement Algorithm 2.5 with 30000 iterations. We use the same values (log–returns, hyperparameters, etc.) used in Example 2.1 and burn–in period is equal to the half of the chain length. Estimated posterior quantities related to parameters are given in Table 2.3 and Figure 2.6 includes detailed graphs. The results are similar to random walk sampler, i.e. when compared to a pure Gibbs sampler these hybrid algorithms both have poorer estimates. The autocorrelation problem for the log–volatility also exists with the independence sampler. Average acceptance rate for this algorithm is %99.62. This is quite a high rate and suggests that the chain converges to the invariant distribution very fast.

Table 2.3: Parameter Estimates from Hybrid Gibbs–Independence Algorithm

	α	β	γ^2
True Values (MLE)	-0.4070	0.9495	0.0405
Mean	-0.5415	0.9333	0.0436
Mode	-0.5049	0.9377	0.0391
Standard Deviation	0.2358	0.0290	0.0204
5th Percentile	-0.9741	0.8799	0.0201
95th Percentile	-0.2277	0.9721	0.0835

In Algorithm 2.5, we follow a model–specific approach in which given the specific form of target distribution we design an efficient proposal. Although it is possible to

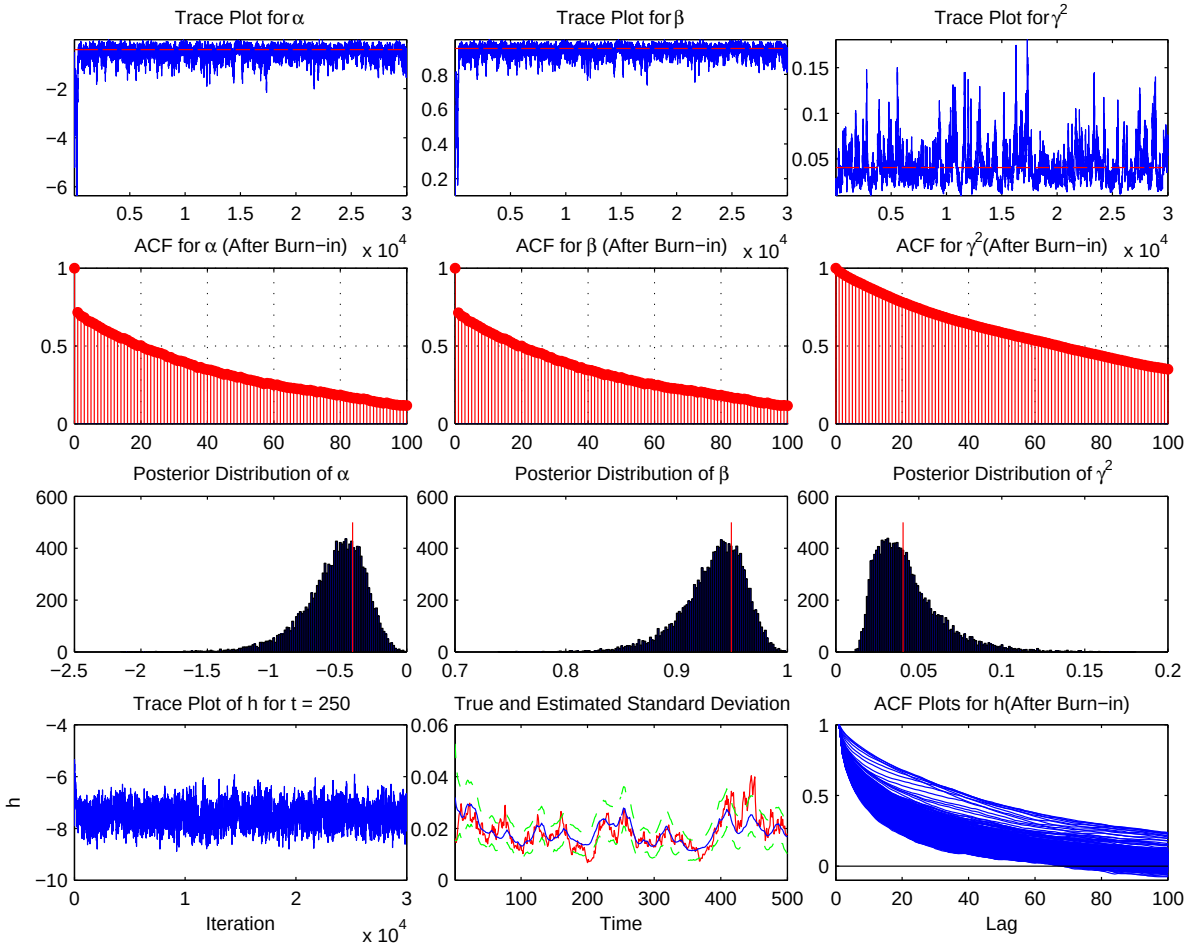

Figure 2.6: Estimation Results for Hybrid Gibbs–Independence Algorithm

find model–specific efficient proposals, there are also certain generic algorithms for finding such proposals. Two popular methods for this are *Adaptive Rejection Metropolis Sampling (ARMS)* (see [27]) and *Griddy Gibbs Method* (see [63]).

2.3.5 Block Sampling

As discussed in previous sections, blocking highly correlated variables and updating them jointly may significantly improve the MCMC algorithms [29, p. 12]. In previous algorithms we sequentially update each log–volatility h_t one at a time. Since the joint distribution $p(X|Y,\Theta) = p(h_0,\ldots,h_T|Y,\Theta)$ is a non–standard one, it is impossible to directly sample from it. For this, [45] utilized an importance sampling idea using a Gaussian approximation. They first transform the observation equation given in (2.3) (assuming $\mu = 0$) into:

$$\log Y_t^2 = h_t + \log \epsilon_t^2 \tag{2.21}$$

where ϵ_t^2 has a χ_1^2 distribution and $E[\log \epsilon_t^2] \approx -1.2704$ and $V[\log \epsilon_t^2] = \pi/2 \approx 4.935$. They approximate the disturbance term, $\log \epsilon_t^2$, with a seven component Gaussian mixture. The approximating model is given by:

$$\log Y_t^2 = h_t + z_t \tag{2.22}$$

$$f(z_t) = \sum_{i=1}^{7} q_i N(z_t | m_i - 1.2704, v_i^2) \tag{2.23}$$

where q_i are the component probabilities, $m_i - 1.2704$ are the component means and v_i^2 are the component variances. They find the optimum values for the constants $\{q_i, m_i, v_i\}$ that yields the best approximation to the exact density. Note that the mixture density can also be written in terms of a component indicator variable s_t such that:

$$z_t | (s_t = i) \sim N(m_i - 1.2704, v_i^2) \text{ with } P[s_t = i] = q_i \tag{2.24}$$

With this approximation, new observation equation (2.22) and the original state equation (2.4) now define a conditionally linear Gaussian system. For such systems posterior densities can be analytically calculated using Kalman filter recursions and all latent states can be jointly sampled using *Forward Filtering Backward Sampling (FFBS)* [22] algorithm[13].

This mixture proposal gives a very accurate approximation to the true density. And it is possible to correct the remaining small error by using an importance weighting. For this, first note that the *target* density is given by:

$$p(h^T | Y, \Theta) = N(h_0 | m_0, C_0) \prod_{t=1}^{T} N(h_t | \alpha + \beta h_{t-1}, \gamma^2) N(y_t | 0, e^{h_t})$$

while we are sampling from the proposal density given by:

$$q(h^T | Y, \Theta) = N(h_0 | m_0, C_0) \prod_{t=1}^{T} N(h_t | \alpha + \beta h_{t-1}, \gamma^2) \left[\sum_{i=1}^{7} q_i N(\log y_t^2 | h_t + m_i - 1.2704, v_i^2) \right]$$

Therefore the importance weights become:

$$\frac{p(h^T | Y, \Theta)}{q(h^T | Y, \Theta)} = \prod_{t=1}^{T} \frac{N(y_t | 0, e^{h_t})}{\sum_{i=1}^{7} q_i N(\log y_t^2 | h_t + m_i - 1.2704, v_i^2)}$$

Example 2.5. We implement the block sampling algorithm using the FFBS with 30000 iterations. We use the same values (log–returns, hyperparameters, etc.) used in Example 2.1 and a burn–in period that is equal to the half of the chain length. Estimation results are given in Figure 2.7.

Although the point estimates for parameters and latent volatilities are similar with previous algorithms, block sampling approach yield very low autocorrelations for the

[13] More information about Kalman filter, FFBS and other issues related to linear Gaussian systems can be found in [75].

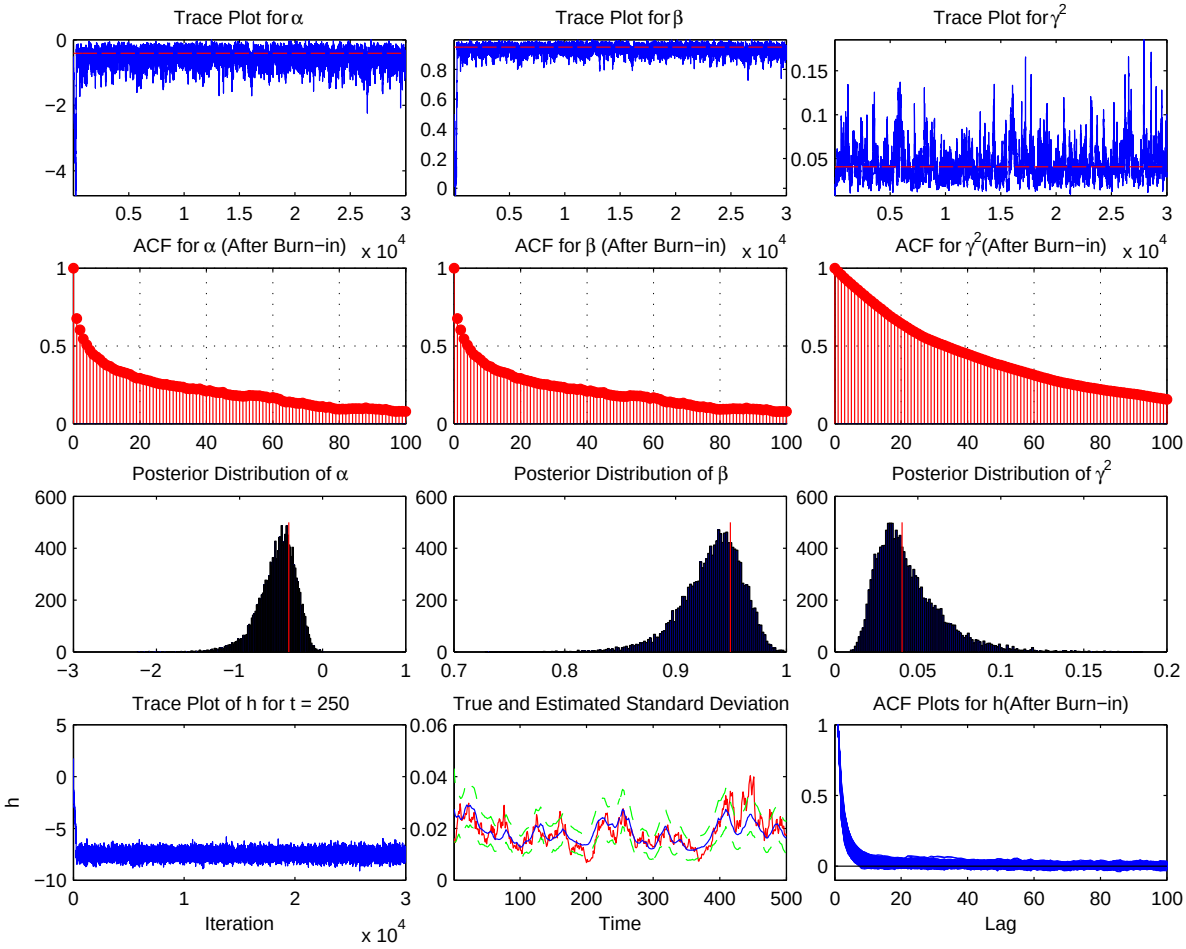

Figure 2.7: Estimation Results for Block Sampling Algorithm

chains. The autocorrelations in volatility chains quickly converge to zero within first ten lag and autocorrelations in parameter chains are smaller than what we obtained in previous algorithms. This lower autocorrelation implies that we can obtain a low estimator variance from the block sampling MCMC algorithm.

CHAPTER 3

PARTICLE FILTERING

In previous chapter, we introduced MCMC methods for inferring both latent states and parameters for a state–space model. MCMC algorithms approximate the joint posterior $p(X, \Theta|Y)$ with Monte Carlo samples. Now we will introduce a distinction between *off-line* (static) analysis and *on–line* (dynamic) analysis. In off–line analysis, all the data until a certain time T is observed and we estimate the parameters, Θ, and all latent variables until time T, $X_1, \ldots, X_T$. Thus the main objective is to estimate the final time joint posterior $p(x_{1:T}, \Theta|y_{1:T})$. In this setting, obtaining the marginal posterior distributions for the latent variables, i.e. $p(x_k|\Theta, y_{1:T}), k = 1, \ldots, T$, is called the *smoothing* problem. This is a static problem that will be solved once.

On the other hand, there is also a dynamic version of this problem called *filtering*. In filtering problem, as we sequentially observe data, $y_1, \ldots, y_{t-1}, y_t, \ldots$, we want to infer sequence of posterior distributions for latent variables. In this case, the target distributions $\{p(x_{1:t}|\Theta, y_{1:t})\}_{t=1}^{T}$ are called *filtering* distributions.

In this chapter, for performing filtering estimates, we introduce the *particle filters (a.k.a. sequential Monte Carlo methods)*[1]. Throughout the chapter, we assume that the parameters of the model, Θ, are known. We begin the chapter by introducing a new notation and setting up the general framework. Afterwards we introduce sequential importance sampling and then sequential importance sampling with resampling as the simplest particle filtering algorithm. Then we consider the most popular extension to the basic filter, namely the auxiliary particle filter.

We will now use the following new notation: Let $\mu(x_1)$ denote the distribution of the initial state, $f(x_t|x_{t-1}, \Theta)$ denote the state evolution density and $g(y_t|x_t, \Theta)$ denote the observation density. In our setting $f(\cdot)$ and $g(\cdot)$ are time–homogeneous but in more general cases they can be time–dependent. Parameters of the model are assumed to be known. Thus we will drop Θ from the notation. Now, given a fixed time t, the following distributions can be defined:

$$\text{Joint Distribution:} \qquad p(x_{1:t}, y_{1:t}) = \mu(x_1) \prod_{k=2}^{t} f(x_k|x_{k-1}) \prod_{k=1}^{t} g(y_k|x_k) \tag{3.1}$$

[1] There is a vast literature on particle filtering methods. For a textbook treatment see [15] and see [18] for a complete recent survey.

$$\text{Prior:} \quad p(x_{1:t}) = \mu(x_1)\prod_{k=2}^{t} f(x_k|x_{k-1}) \tag{3.2}$$

$$\text{Likelihood:} \quad p(y_{1:t}|x_{1:t}) = \prod_{k=1}^{t} g(y_k|x_k) \tag{3.3}$$

$$\text{Marginal Likelihood:} \quad p(y_{1:t}) = \int p(x_{1:t}, y_{1:t})dx_{1:t} \tag{3.4}$$

$$\text{Posterior:} \quad p(x_{1:t}|y_{1:t}) = \frac{p(x_{1:t}, y_{1:t})}{p(y_{1:t})} = \frac{p(x_{1:t})p(y_{1:t}|x_{1:t})}{\int p(x_{1:t}, y_{1:t})dx_{1:t}} \tag{3.5}$$

$$\text{One Step Forecast:} \quad p(y_{t+1}|y_{1:t}) = \int p(x_{1:t}|y_{1:t})f(x_{t+1}|x_t)g(y_{t+1}|x_{t+1})dx_{1:t+1} \tag{3.6}$$

Our analysis will be sequential for each time $t = 1, 2, \ldots, T$. Therefore we need updating rules (from $t-1$ to t) for these densities. Using simple probability rules (especially Bayes rule) we can define updating equations as follows:

- Joint distribution of observations and states:

  $$p(x_{1:t}, y_{1:t}) = p(x_{1:t-1}, y_{1:t-1})f(x_t|x_{t-1})g(y_t|x_t)$$

- Likelihood:

  $$\begin{aligned} p(y_{1:t}|x_{1:t}) &= p(y_{1:t-1}|x_{1:t-1})\frac{f(x_t|x_{t-1})g(y_t|x_t)}{p(x_t|x_{1:t-1})} \\ &\propto p(y_{1:t-1}|x_{1:t-1})f(x_t|x_{t-1})g(y_t|x_t) \end{aligned}$$

- Marginal Likelihood:

  $$p(y_{1:t}) = p(y_{1:t-1})\int p(x_{1:t-1}|y_{1:t-1})f(x_t|x_{t-1})g(y_t|x_t)dx_{1:t}$$

- Filtering Recursions:

  $$\begin{aligned} &\text{Posterior at } t-1: && p(x_{1:t-1}|y_{1:t-1}) && \text{(given)} \\ &\Rightarrow\text{Prior at } t: && p(x_{1:t}|y_{1:t-1}) && = p(x_{1:t-1}|y_{1:t-1})f(x_t|x_{t-1}) \\ &\Rightarrow\text{Forecast at } t: && p(y_t|y_{1:t-1}) && = \int p(x_{1:t-1}|y_{1:t-1})f(x_t|x_{t-1})g(y_t|x_t)dx_{1:t} \\ & && && = \int p(x_{1:t}|y_{1:t-1})g(y_t|x_t)dx_{1:t} \\ &\Rightarrow\text{Posterior at } t: && p(x_{1:t}|y_{1:t}) && = \frac{p(x_{1:t}|y_{1:t-1})g(y_t|x_t)}{p(y_t|y_{1:t-1})} \\ & && && \propto \underbrace{p(x_{1:t}|y_{1:t-1})}_{Prior}\underbrace{g(y_t|x_t)}_{Likelihood} \end{aligned}$$

$$= p(x_{1:t-1}|y_{1:t-1})\frac{f(x_t|x_{t-1})g(y_t|x_t)}{p(y_t|y_{1:t-1})}$$
$$\propto p(x_{1:t-1}|y_{1:t-1})f(x_t|x_{t-1})g(y_t|x_t)$$

In filtering problem, we want to (sequentially for each t) sample from the filtering density $p(x_{1:t}|y_{1:t})$ given as:

$$p(x_{1:t}|y_{1:t}) = \frac{p(x_{1:t}, y_{1:t})}{p(y_{1:t})}$$
$$\propto p(x_{1:t}, y_{1:t}) = \mu(x_1)\prod_{k=2}^{t} f(x_k|x_{k-1})\prod_{k=1}^{t} g(y_k|x_k)$$

which is typically complex, high–dimensional and does not correspond to a standard distribution. Furthermore target density has an increasing dimension with time t. Therefore direct sampling from this density is typically impossible. For these type of problems, *sequential importance sampling* (see Chapter 14 in [65]) is proposed in the literature.

3.1 Sequential Importance Sampling (SIS)

Assume that, we want to estimate the following integral:

$$I = E_h[\varphi(Z)] = \int \varphi(z)h(z)dz$$

for a function $\varphi(\cdot)$ and a density $h(\cdot)$. If we can sample from $h(\cdot)$, then we can approximate the integral using Monte Carlo samples. However, if it is not possible to sample from the target density $h(\cdot)$, then we can use an *importance sampling* approach. The method is based on the following identity:

$$I = \int \varphi(z)h(z)dz = \int \varphi(z)\frac{h(z)}{h'(z)}h'(z)dz = E_{h'}[\varphi(Z)\frac{h(Z)}{h'(Z)}]$$

where $h'(\cdot)$ is the proposal density with $supp(h') \supset supp(h \cdot \varphi)$. Then the estimator is given by $\hat{I} = \frac{1}{N}\sum_{i=1}^{N} \varphi(z_i)\frac{h(z_i)}{h'(z_i)}$ where $z_i \sim h'$. Here $w(Z) = \frac{h(Z)}{h'(Z)}$ is called the importance weight function.

Now assume a sequential filtering framework. At each time t, let $p_t(x_{1:t}) = p(x_{1:t}|y_{1:t})$ denote our target density, $\gamma_t(x_{1:t}) = p(x_{1:t}, y_{1:t})$ denote its unnormalized version and $q_t(x_{1:t})$ denote our proposal density for the importance sampling given as:

$$q_t(x_{1:t}) = q_{t-1}(x_{1:t-1})q_t(x_t|x_{1:t-1}) \tag{3.7}$$
$$= q_1(x_1)\prod_{k=2}^{t} q_k(x_k|x_{1:k-1}) \tag{3.8}$$

where each $q_t(x_t|x_{1:t-1})$ is such that we can easily sample from it.

Given this structure, the importance weights at any time t, $w_t(x_{1:t})$, is given by:

$$\begin{aligned} w_t(x_{1:t}) &= \frac{\gamma_t(x_{1:t})}{q_t(x_{1:t})} \\ &= \frac{\gamma_{t-1}(x_{1:t-1})}{q_{t-1}(x_{1:t-1})} \frac{\gamma_t(x_{1:t})}{\gamma_{t-1}(x_{1:t-1}) q_t(x_t|x_{1:t-1})} \end{aligned}$$

which can be written as:

$$\begin{aligned} w_t(x_{1:t}) &= w_{t-1}(x_{1:t-1}) \alpha_t(x_{1:t}) \\ &= w_1(x_1) \prod_{k=2}^{t} \alpha_k(x_{1:k}) \end{aligned}$$

where the incremental importance weight function $\alpha_t(x_{1:t})$ is given by:

$$\alpha_t(x_{1:t}) = \frac{\gamma_t(x_{1:t})}{\gamma_{t-1}(x_{1:t-1}) q_t(x_t|x_{1:t-1})} \tag{3.9}$$

Steps of SIS algorithm is given in Algorithm 3.1.

Algorithm 3.1 SIS Algorithm

1: **for** $t = 1$, for each $i = 1, \ldots, N$ **do**
2: Sample $X_1^i \sim q_1(x_1)$
3: Compute unnormalized weights $w_1(X_1^i)$
4: Normalize the weights, i.e. $W_1^i = \frac{w_1(X_1^i)}{\sum_{j=1}^N w_1(X_1^j)}$
5: **end for**
6: **for** $t \geq 2$, for each $i = 1, \ldots, N$ **do**
7: Sample $X_t^i \sim q_t(x_t|X_{1:t-1}^i)$
8: Compute unnormalized weights $w_t(X_{1:t}^i) = w_{t-1}(X_{1:t-1}^i)\alpha_t(X_{1:t}^i)$
9: Normalize the weights, i.e. $W_t^i = \frac{w_t(X_{1:t}^i)}{\sum_{j=1}^N w_t(X_{1:t}^j)}$
10: **end for**

For each time t, SIS algorithm generates N weighted samples for the full path of the states up to t, $\{X_{1:t}^i\}_{i=1}^N$, with weights $\{W_t^i\}_{i=1}^N$. Using these, we can approximate the target density, $p_t(x_{1:t})$ with:

$$\hat{p}_t(x_{1:t}) = \sum_{i=1}^{N} W_t^i \delta_{X_{1:t}^i}(x_{1:t})$$

and estimate its normalizing constant $Z_t := p_t(x_{1:t})/\gamma_t(x_{1:t})$ with:

$$\hat{Z}_t = \frac{1}{N} \sum_{i=1}^{N} w_t(X_{1:t}^i)$$

And for expectation of any test function φ_t given by:

$$I_t(\varphi_t) := E_{p_t}[\varphi_t(x_{1:t})] = \int \varphi_t(x_{1:t}) p_t(x_{1:t}) dx_{1:t}$$

we have the following estimate:

$$\hat{I}_t(\varphi_t) = \int \varphi_t(x_{1:t}) \hat{p}_t(x_{1:t}) dx_{1:t} = \sum_{i=1}^{N} W_t^i \varphi_t(X_{1:t}^i)$$

SIS algorithm is a generic one that can be used in sequential problems. However as the dimension of the target distribution increases, SIS algorithm gives increasingly inaccurate estimation results. Indeed variance of importance weights and thus the estimator variance increase (typically exponentially) with t[2].

3.2 Sequential Importance Sampling and Resampling (SISR)

As we discuss in previous section, SIS algorithm has weight degeneracy problem. To avoid this problem, we need to an additional *resampling* step. In this SISR setting, at each t, after generating the particles $\{X_{1:t}^i\}_{i=1}^N$ with corresponding weights $\{W_t^i\}_{i=1}^N$, we resample the particles according to these weights with replacement. This step ensures that low weight particles are eliminated and high weight particles survive and are multiplied.

After the resampling step, the new particles $\left\{\overline{X}_{1:t}^i\right\}_{i=1}^N$ have equal weights $\overline{W}_t^i = 1/N$ for all $i = 1, \ldots, N$. However resampling step introduces a trade–off: with resampling, we improve the estimates for the future time marginals, but at the same time increase the immediate variance.

Algorithm 3.2 shows the steps of SISR algorithm. In SISR, since we reset the system at each resampling step, the unnormalized weights are always equal to the incremental importance weights, $\alpha_t(X_{1:t}^i)$. This prevents weight accumulation over time and therefore avoid weight degeneracy. For the resampling step, different resampling algorithms are proposed in the literature. These include *multinomial*, *systematic*, *residual* and *stratified* resampling[3].

Deciding on the frequency of resampling step is also another important issue. Since resampling step some additional noise to the system, it is not optimal to resample at each iteration. For this problem, [51] proposed using a measure called *Effective Sample Size (ESS)*. ESS is based on the ratio of estimator variances for an importance sampling as opposed to i.i.d. sampling. The ratio is (approximately) equal to:

$$\frac{1}{1 + Var\left[w(X)\right]}$$

[2] See [18] for details of such problem.

[3] For details of resampling schemes, see [18],[14], [50] and [46].

Algorithm 3.2 SISR Algorithm

1: **for** $t = 1$, for each $i = 1, \ldots, N$ **do**
2: Sample $X_1^i \sim q_1(x_1)$
3: Compute unnormalized weights $w_1(X_1^i)$
4: Normalize the weights, i.e. $W_1^i \propto w_1(X_1^i)$
5: Resample $\left\{X_1^i, W_1^i\right\}$ to obtain N equally-weighted particles $\left\{\overline{X}_1^i, \frac{1}{N}\right\}$
6: **end for**
7: **for** $t \geq 2$, for each $i = 1, \ldots, N$ **do**
8: Sample $X_t^i \sim q_t(x_t|\overline{X}_{1:t-1}^i)$ and set $X_{1:t}^i \leftarrow (\overline{X}_{1:t-1}^i, X_t^i)$
9: Compute unnormalized weights $w_t(X_{1:t}^i) = \alpha_t(X_{1:t}^i)$
10: Normalize the weights, i.e. $W_t^i \propto w_t(X_{1:t}^i)$
11: Resample $\left\{X_{1:t}^i, W_t^i\right\}$ to obtain N equally-weighted particles $\left\{\overline{X}_{1:t}^i, \frac{1}{N}\right\}$
12: **end for**

where $w(X)$ is the importance weights. Thus, using sampled particles and their corresponding weights, ESS can be estimated as follows:

$$ESS = \frac{1}{\sum_{i=1}^{N}(W_t^i)^2} \tag{3.10}$$

Typically, resampling is done if ESS is smaller than a threshold, which is generally taken as $N/2$.

Example 3.1. For ARSV model, we simulate 100 data points with parameters $\mu = 0$, $\alpha = -0.4$, $\beta = 0.9$ and $\gamma = 1$. The prior for states, $p(x_{1:t})$ as given in (3.2), is used as the proposal density. With this choice, incremental importance weights becomes proportional to the likelihood, $\alpha_t(x_{1:t}) \propto g(y_t|x_t)$. We used $N = 5000$ particles. For resampling, we used multinomial and systematic resampling schemes and we resample only if $ESS < N/2$. The results obtained with systematic resampling are shown in Figure 3.1. We obtain very similar results for multinomial resampling as well. Contrary to the SIS algorithm, there is no weight degeneracy problem. The improvement over SIS algorithm as captured by ESS is shown in the last graph.

3.3 Optimal Importance Distribution and Adapted Filtering

In previous examples, we used the prior distribution for the states, $p(x_{1:t})$ as our importance distribution, $q_t(x_{1:t}) = p(x_{1:t})$, i.e. $q_1(x_1) = \mu(x_1)$ and $q_t(x_t|x_{1:t-1}) = f(x_t|x_{t-1})$ for $t > 1$. However we can also choose any other distribution (that meets certain criteria). Choosing an importance distribution is the most important step in a SISR framework and optimization criterion for this is choosing a proposal density such that, given $x_{1:t-1}$ and $y_{1:t}$, it minimizes the variance of the importance weights (see [17]):.

Proposition 3.1. *For SISR algorithm, the optimal importance proposal (3.8) that minimizes the variance of the importance weights at time t, given $x_{1:t-1}$ and $y_{1:t}$ is given by $q_t(x_t|x_{1:t-1}) = p(x_t|x_{t-1}, y_t)$.*

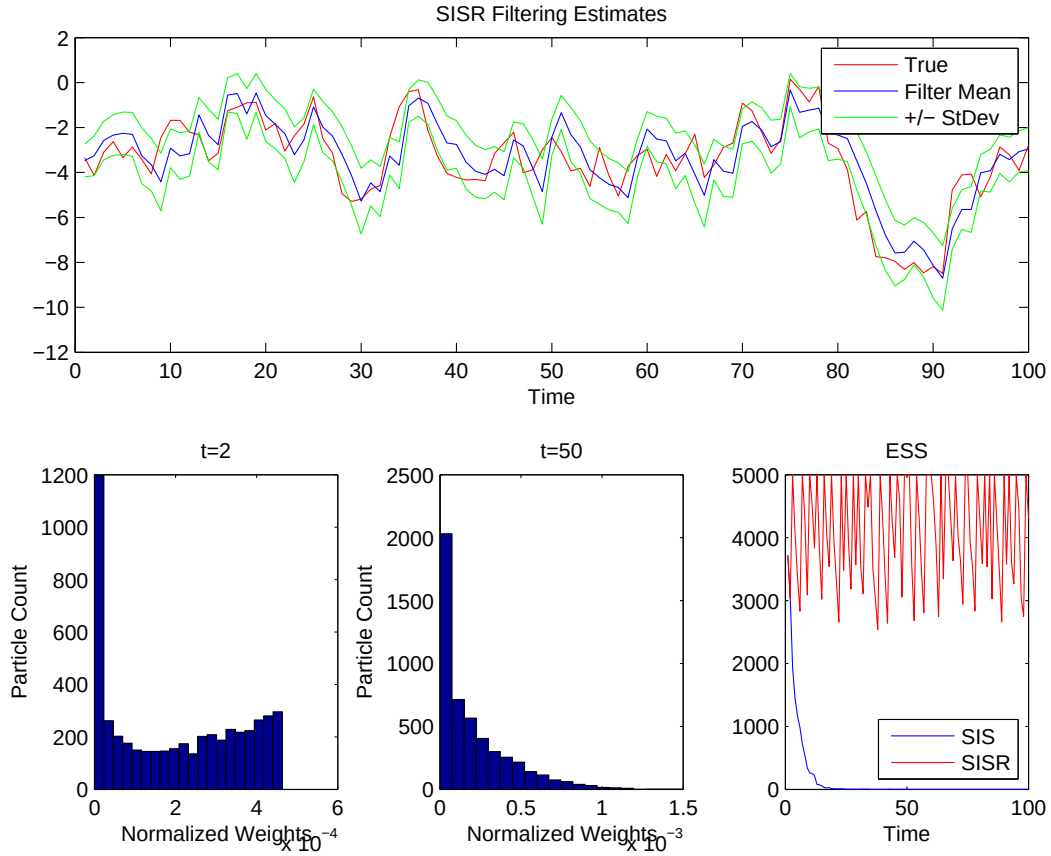

Figure 3.1: SISR Filtering Results. First graph shows true and estimated log–volatilities. Next two graphs show the histograms for normalized weights for different time periods. The last graph shows the ESS for SIS and SISR.

Proof. In SISR algorithm, given $x_{1:t-1}$ and $y_{1:t}$, new particles are sampled from $q_t(x_t|x_{1:t-1})$ and weighted proportional to the incremental importance weights $\alpha_t(x_{1:t})$. Therefore we want to minimize the conditional variance of these incremental weights (given $x_{1:t-1}$ and $y_{1:t}$) calculated with respect to q_t.

$$Var_{q_t}\left[\alpha_t(x_{1:t})\right] = Var_{q_t}\left[\frac{\gamma_t(x_{1:t})}{\gamma_{t-1}(x_{1:t-1})q_t(x_t|x_{1:t-1})}\right]$$
$$= E_{q_t}\left[\frac{\gamma_t^2(x_{1:t})}{\gamma_{t-1}^2(x_{1:t-1})q_t^2(x_t|x_{1:t-1})}\right] - \left\{E_{q_t}\left[\frac{\gamma_t(x_{1:t})}{\gamma_{t-1}(x_{1:t-1})q_t(x_t|x_{1:t-1})}\right]\right\}^2$$

Second term does not depend on q_t. Therefore we only need to minimize the first term. From Jensen's inequality:

$$E_{q_t}\left[\frac{\gamma_t^2(x_{1:t})}{\gamma_{t-1}^2(x_{1:t-1})q_t^2(x_t|x_{1:t-1})}\right] \geq \left\{E_{q_t}\left[\frac{\gamma_t(x_{1:t})}{\gamma_{t-1}(x_{1:t-1})q_t(x_t|x_{1:t-1})}\right]\right\}^2 = \left\{\int \frac{\gamma_t(x_{1:t})}{\gamma_{t-1}(x_{1:t-1})}dx_t\right\}^2$$

which provides a lower bound independent of q_t. This lower bound is attained by

choosing:

$$\begin{aligned} q_t(x_t|x_{1:t-1}) &= \frac{\gamma_t(x_{1:t})}{\gamma_{t-1}(x_{1:t-1})} = \frac{\gamma_t(x_{1:t})}{\int \gamma_t(x_{1:t})dx_t} \\ &= \frac{p(x_{1:t}, y_{1:t})}{\int p(x_{1:t}, y_{1:t})dx_t} = \frac{p(x_{1:t}, y_{1:t})}{p(x_{1:t-1}, y_{1:t})} = p(x_t|x_{1:t-1}, y_{1:t}) \\ &= p(x_t|x_{t-1}, y_t) = \frac{g(y_t|x_t)f(x_t|x_{t-1})}{p(y_t|x_{t-1})} \propto g(y_t|x_t)f(x_t|x_{t-1}) \end{aligned} \tag{3.11}$$

where the last line uses the Markov assumption for the states and conditional independence of observations. If we can use this optimal proposal, then the conditional variance of the incremental importance weights will be zero, which means equal incremental weights for all particles. □

Except for only limited number of cases, it is generally not possible to sample from this optimal proposal, $p(x_t|x_{t-1}, y_t)$. However, we should always aim to approximate it[4] by choosing a proposal as close to the optimal one as possible. Furthermore, if we use the optimal proposal, then incremental weights become proportional to the *predictive likelihood*, i.e. $\alpha_t(x_{1:t}) = p(y_t|x_{t-1}) = \int f(x_t|x_{t-1})g(y_t|x_t)dx_t$. In some cases, evaluating this density analytically can also be not possible.

When compared to our initial proposal distribution, optimal proposal has some important distinctions. First, contrary to our original proposal, optimal proposal uses new observation y_t when generating new particles. This allows us to generate new particles around regions of high probability mass. Such a filter is called *adapted* filter (see [60]). Second, the new weights $w_t(x_{1:t}) = \alpha_t(x_{1:t})$ do not depend on sampled states x_t. This allows us to perform a resample–then–sample filtering and can improve the filter's approximation since it provides a greater number of distinct particles for the approximation. This also allows obtaining i.i.d. samples. [60] call particle filters that generates i.i.d. samples from the target and therefore does not need resampling after sampling new states, as *fully adapted* filters.

The optimal proposal for ARSV model is:

$$p(x_t|x_{t-1}, y_t) \propto f(x_t|x_{t-1})g(y_t|x_t) = N(x_t|\alpha + \beta x_{t-1}, \gamma^2)N(y_t|0, e^{x_t}) \tag{3.12}$$

This density is not a standard one and thus it is not possible to sample from it using standard methods. Since it is log–concave, we can implement a rejection algorithm for sampling from it. However, incremental importance weights should also be calculated. They are given as:

$$\begin{aligned} \alpha_t(X_{t-1}^i) = p(y_t|X_{t-1}^i) &= \int f(x_t|X_{t-1}^i)g(y_t|x_t)dx_t \\ &= \int N(x_t|\alpha + \beta X_{t-1}^i, \gamma^2)N(y_t|0, e^{x_t})dx_t \end{aligned} \tag{3.13}$$

which cannot be evaluated analytically. Therefore, for ARSV model, we cannot implement a fully adaptive filter.

[4] For example, using Extended or Unscented Kalman Filters or using local approximation techniques, as suggested in the literature ([16, 18]).

3.4 Auxiliary Particle Filtering

As we discussed previously, typically, we cannot sample from the optimal proposal distribution and/or calculate the importance weights. Despite this, it is generally beneficial to use current observation y_t in both resampling and propagation steps. Pitt and Shephard utilizes this idea in their *Auxiliary particle filter (APF)* [60]. Contrary to the original bootstrap filter which is a sample–resample filter, APF is an example of resample–sample filter. In APF, we use $\hat{p}(y_t|x_{t-1})$, which is an approximation to the true predictive likelihood $p(y_t|x_{t-1}) = \int f(x_t|x_{t-1})g(y_t|x_t)dx_t$, for reweighting old particles and propagate new ones by using $q(x_t|x_{t-1})$, which is an approximation to the optimal proposal, $p(x_t|y_t, x_{t-1}) \propto f(x_t|x_{t-1})g(y_t|x_t)$. APF algorithm[5] is given in Algorithm 3.3 [6]

Algorithm 3.3 APF Algorithm

for $t = 1$, for each $i = 1, \ldots, N$ **do**
  Sample $X_1^i \sim q_1(x_1)$
  Weight the particles $W_1^i \propto \frac{g(y_1|X_1^i)\mu(X_1^i)}{q_1(X_1^i)}$
end for
for $t \geq 2$, for each $i = 1, \ldots, N$ **do**
  Reweight old particles $\tilde{W}_{t-1}^i \propto W_{t-1}^i \times \hat{p}(y_t|X_{t-1}^i)$
  Resample $\left\{X_{1:t-1}^i, \tilde{W}_{t-1}^i\right\}$ to obtain N equally-weighted particles $\left\{\tilde{X}_{1:t-1}^i, \frac{1}{N}\right\}$
  Sample $X_t^i \sim q_t(x_t|\tilde{X}_{t-1}^i)$ and set $X_{1:t}^i \leftarrow (\tilde{X}_{1:t-1}^i, X_t^i)$
  Weight new particles $W_t^i \propto \frac{g(y_t|X_t^i)f(X_t^i|X_{t-1}^i)}{\hat{p}(y_t|X_{t-1}^i)q_t(X_t^i|X_{t-1}^i)}$
end for

Johansen and Doucet [43] showed that the APF algorithm is just a special case of SISR algorithm with a specific target distribution. With this representation, in order to obtain an efficient APF algorithm, we should use good approximations $q(x_t|x_{t-1})$ and $\hat{p}(y_t|x_{t-1})$ to optimal proposal and predictive likelihood, respectively. Additionally, in order to ensure a finite estimator variance, we should select $\hat{p}(y_t|x_{t-1})q_t(x_t|x_{t-1})$ with thicker tails than $g(y_t|x_t)f(x_t|x_{t-1})$ (with respect to $x_{t-1:t}$), e.g. selecting $\hat{p}(y_t|x_{t-1})$ with thicker tails than $p(y_t|x_{t-1})$ and selecting $q_t(x_t|x_{t-1})$ with thicker tails than $p(x_t|y_t, x_{t-1})$.

As we discussed before, for ARSV model, we cannot implement a fully adapted filter using optimal proposal predictive likelihood. In this section we will implement APF algorithms for ARSV model. Five different algorithms will be compared in a simulation study. For all the algorithms we use, we choose an approximation $\hat{g}(y_t|x_t, x_{t-1})$ to the true likelihood $g(y_t|x_t)$ for defining the following densities:

$$\hat{p}(y_t|x_{t-1}) = \int \hat{g}(y_t|x_t, x_{t-1})f(x_t|x_{t-1})dx_t \tag{3.14}$$

$$q_t(x_t|x_{t-1}) \propto \hat{g}(y_t|x_t, x_{t-1})f(x_t|x_{t-1}) \tag{3.15}$$

[5] For more detailed particle filtering algorithms and empirical implementations, see [77].

[6] The original algorithm in [60] uses a mixture of distributions argument and samples an auxiliary variable which corresponds to the index of particles. However it is represented as a resample–then–sample filter here. This version of the algorithm generally outperforms the original representation [61, 18].

and calculating the following second stage weights:

$$W_t^i \propto \frac{g(y_t|x_t)f(x_t|x_{t-1})}{\hat{g}(y_t|x_t,x_{t-1})f(x_t|x_{t-1})} = \frac{g(y_t|x_t)}{\hat{g}(y_t|x_t,x_{t-1})} \tag{3.16}$$

Approximation 1)

In the original APF article [60], likelihood evaluated at a certain value μ_t, i.e. $\hat{g}(y_t|x_t,x_{t-1}) = g(y_t|\mu_t)$ is used as an approximation. μ_t can be chosen as the mean, mode, etc for the density $f(x_t|x_{t-1})$, and thus μ_t is a function of x_{t-1} and not x_t. Common choice in the literature is the mean.

The prior mean for ARSV model is $\mu_t = \alpha + \beta x_{t-1}$ and thus:

$$\hat{p}(y_t|x_{t-1}) = \int g(y_t|\mu_t)f(x_t|x_{t-1})dx_t = g(y_t|\mu_t) = N(y_t|0,\exp(\mu_t))$$
$$q_t(x_t|x_{t-1}) \propto g(y_t|\mu_t)f(x_t|x_{t-1}) \propto f(x_t|x_{t-1}) = N(x_t|\mu_t,\gamma^2)$$

where the second stage weights are given as:

$$W_t^i \propto \exp\left\{-\frac{x_t-\mu_t}{2} - \frac{y_t^2}{2}\left[e^{-x_t} - e^{-\mu_t}\right]\right\}$$

In this filter, we reweight particles using the likelihood evaluated at prior mean, and also second stage weights depend on (in a non–linear fashion) the distance to μ_t, e.g. if $x_t = \mu_t$ then $W_t^i \propto 1$. This filter has an adapted reweighting step, but the propagation is blind. Furthermore, the importance weights are not bounded, which means that for some parameter values and sampled particles the algorithm may yield infinite weights.

Approximation 2)

ARSV model has a likelihood which is log–concave in x_t. Using this, we can use a first–order Taylor expansion of log–likelihood around prior mean, as suggested by [60]:

$$\log\hat{g}(y_t|x_t,\mu_t) = \log g(y_t|\mu_t) + (x_t-\mu_t)\frac{\partial \log g(y_t|\mu_t)}{\partial x_t}$$
$$\hat{g}(y_t|x_t,\mu_t) \propto \exp\left\{-\frac{x_t}{2} - \frac{y_t^2}{2}\exp(-\mu_t)\left[1-(x_t-\mu_t)\right]\right\}$$

Log–concavity[7] ensures that we have (for all x_t) $\hat{g}(y_t|x_t,\mu_t) \geq g(y_t|x_t)$. This filter yields:

$$\hat{p}(y_t|x_{t-1}) \propto \exp\left\{\frac{\mu_t^{*2}-\mu_t^2}{2\gamma^2} - \frac{y_t^2}{2}\exp(-\mu_t)(1+\mu_t)\right\}$$
$$\text{where } \mu_t^* = \mu_t - \frac{\gamma^2}{2}\left[1-y_t^2\exp(-\mu_t)\right]$$

[7] For a concave function $f(x)$, for any x_0, we have $f(x) \leq f(x_0) + (x-x_0)f'(x_0)$, where the right hand side of the inequality is the first order Taylor series expansion of $f(x)$ around x_0.

$$q_t(x_t|x_{t-1}) \propto N(x_t|\mu_t^*, \gamma^2)$$
$$W_t^i \propto \exp\left\{-\frac{y_t^2}{2}\left[e^{-x_t} - e^{-\mu_t}(1 - x_t + \mu_t)\right]\right\}$$

Because of log–concavity, estimator variance is bounded in this filter. Additionally both the reweighting and propagation steps are adapted.

Approximation 3)

For ARSV model, it is also possible to use a second–order Taylor expansion of log–likelihood around the prior mean. This choice gives a second order polynomial approximation and yields a Gaussian approximation (in x_t) for the likelihood:

$$\log \hat{g}(y_t|x_t, \mu_t) = \log g(y_t|\mu_t) + (x_t - \mu_t)\frac{\partial \log g(y_t|\mu_t)}{\partial x_t} + \frac{1}{2}(x_t - \mu_t)^2\frac{\partial^2 \log g(y_t|\mu_t)}{\partial x_t^2}$$
$$\hat{g}(y_t|x_t, \mu_t) \propto \exp\left\{-\frac{x_t}{2} - \frac{y_t^2}{2}\exp(-\mu_t)\left[1 - (x_t - \mu_t) + \frac{1}{2}(x_t - \mu_t^2)\right]\right\}$$

And thus:

$$A = \frac{y_t^2}{2}\exp(-\mu_t), B = A + 1/\gamma^2$$
$$C = -0.5 + A(1 + \mu_t) + \frac{\mu_t}{\gamma^2}, D = A(1 + \mu_t) + \frac{A\mu_t^2}{2} + \frac{\mu_t^2}{2\gamma^2}$$
$$\hat{p}(y_t|x_{t-1}) \propto \frac{1}{\sqrt{B}}\exp\left\{\frac{C^2}{2B} - D\right\}$$
$$q_t(x_t|x_{t-1}) \propto N\left(x_t|\frac{C}{B}, \frac{1}{B}\right)$$
$$W_t^i \propto \exp\left\{\frac{y_t^2}{2}\exp(-\mu_t)\left[\frac{1}{2}(x_t - \mu_t)^2 - (x_t - \mu_t)\right]\right\}$$

This filter also has adapted steps both for weighting and propagation, but estimator variance is unbounded.

Approximation 4)

We can also use a second–order Taylor expansion of log–likelihood around MLE, as suggested by [70]. The approximation is:

$$\log \hat{g}(y_t|x_t, \mu_t) = \log g(y_t|\log y_t^2) + (x_t - \log y_t^2)\frac{\partial \log g(y_t|\log y_t^2)}{\partial x_t} + \frac{1}{2}(x_t - \log y_t^2)^2\frac{\partial^2 \log g(y_t|\log y_t^2)}{\partial x_t^2}$$
$$\hat{g}(y_t|x_t, \mu_t) \propto \frac{1}{y_t}N\left(x_t|\log y_t^2, 2\right)$$

i.e. which is a Gaussian approximation with MLE as the and a variance of 2.

Note that this approximation is proportional to (as a density for y_t) $N(\log y_t^2 | x_t - 1, 2)$. Thus, by using *Extended Kalman Filter* method, we use the following linear Gaussian system to build our proposal:

$$\begin{aligned} \log y_t^2 &= x_t - 1 + v_t, v_t \sim N(0, 2) \\ x_t &= \alpha + \beta x_{t-1} + w_t, w_t \sim N(0, \gamma^2) \end{aligned}$$

which yields:

$$\begin{aligned} A = \frac{1}{2} + \frac{1}{\gamma^2}, B = \frac{\log y_t^2}{2} + \frac{\mu_t}{\gamma^2}, \tilde{\mu}_t = \frac{B}{A} \\ \hat{p}(y_t | x_{t-1}) \propto \exp\left\{ \frac{\tilde{\mu}_t^2}{4} + \frac{\tilde{\mu}_t^2 - \mu_t^2}{2\gamma^2} \right\} \\ q_t(x_t | x_{t-1}) \propto N\left(x_t | \tilde{\mu}_t, \frac{1}{A} \right) \\ W_t^i \propto \exp\left\{ -\frac{x_t}{2} - \frac{y_t^2}{2} \exp(-x_t) + \frac{(x_t - \log y_t^2)^2}{4} \right\} \end{aligned}$$

This filter also has adapted steps both for weighting and propagation, but estimator variance is unbounded.

Approximation 5)

Approximation 4 has a fixed variance of 2 and thus there is no guarantee it dominates the likelihood for all possible values of y_t. This problem can be solved by using a more diffuse Gaussian distribution. The variance of the distribution can be set to a larger number to increase the possibility of bounded estimator variance. In an offline analysis it is also possible to calculate a value for σ^2 large enough to ensure that the density dominates the likelihood for all observed values of y_t in the sample.

This filter yields:

$$\begin{aligned} A = \frac{1}{\sigma^2} + \frac{1}{\gamma^2}, B = \frac{\log y_t^2}{\sigma^2} + \frac{\mu_t}{\gamma^2}, \tilde{\mu}_t = \frac{B}{A} \\ \hat{p}(y_t | x_{t-1}) \propto \exp\left\{ \frac{\tilde{\mu}_t^2}{2\sigma^2} + \frac{\tilde{\mu}_t^2 - \mu_t^2}{2\gamma^2} \right\} \\ q_t(x_t | x_{t-1}) \propto N\left(x_t | \tilde{\mu}_t, \frac{1}{A} \right) \\ W_t^i \propto \exp\left\{ -\frac{x_t}{2} - \frac{y_t^2}{2} \exp(-x_t) + \frac{(x_t - \log y_t^2)^2}{2\sigma^2} \right\} \end{aligned}$$

Example 3.2. We implemented the aforementioned five different APF algorithms. Since γ^2 (i.e. signal–to–noise ratio) critically affects the efficiency of filters, we use two different values, $\gamma = 0.3$ and $\gamma = 1$. We use $\alpha = -0.4$, $\beta = 0.9$ and $\sigma^2 = 20$ for

the last filter. With these parameter values, we implement 100 simulation and filtering with a time period of $T = 100$ and with $N = 5000$ particles. Systematic resampling is done if ESS is below $N/2$. As expected, these five filters require very similar computational time.

For each APF algorithm, we calculate the average ESS and gross root mean squared error defined as $RMSE = \sqrt{\frac{1}{100T}\sum_{m=1}^{100}\sum_{t=1}^{T}(\hat{X}_t^m - X_t^{True})^2}$, where $\hat{X}_t^m$ is the filtering estimate of X_t at m^{th} implementation. The results are shown in Table 3.1 and selected filtering examples are given in Figure 3.2.

Table 3.1: Average ESS and Relative RMSE for Different APF Algorithms

	$\gamma = 0.3$		$\gamma = 1$	
	Relative RMSE	Average ESS	Relative RMSE	Average ESS
Approximation 1	1.0154	3759	**1.0000**	3171
Approximation 2	1.0825	3831	1783.6	3839
Approximation 3	1.8332	3599	2.1106	2208
Approximation 4	1.0162	3470	1.0196	2573
Approximation 5	**1.0000**	3706	1.0666	3331

So, given the values of parameters and simulated log–returns, the estimation results suggest the following conclusions:

- In some cases we obtain better results with an unbounded weight function as compared to a bounded weight function. Such cases may occur if, a) proposal density decays fast enough that makes weight variance finite despite an unbounded weight function, b) weight function takes bounded values for the specific value of parameter and observed variables. However, there is always a possibility for a filter with unbounded weight function to fail at any moment for some other values of parameters and observations.
- For low signal–to–noise ratio (i.e. $\gamma = 0.3$), all except the third algorithm yield similar results. For high signal–to–noise ratio (i.e. $\gamma = 1$), first and fourth algorithms give better results and second algorithm yield extremely bad results.
- The second algorithm has a bounded weight function. However the variance of weights increases with γ. Therefore second algorithm performs the best for $\gamma = 0.3$, while we have a poor result for $\gamma = 1$.
- We used similar approximations in the last two algorithms and only the last algorithm has a bounded weight function. However the results favor the fourth algorithm over the fifth one.
- Average ESS and RMSE have not a monotonic relation. This confirms that ESS cannot be used as a measure for estimator variance if the filter is not providing a reasonable approximation.
- Finally, as expected, increasing the state variance γ^2 yields higher RMSE values.

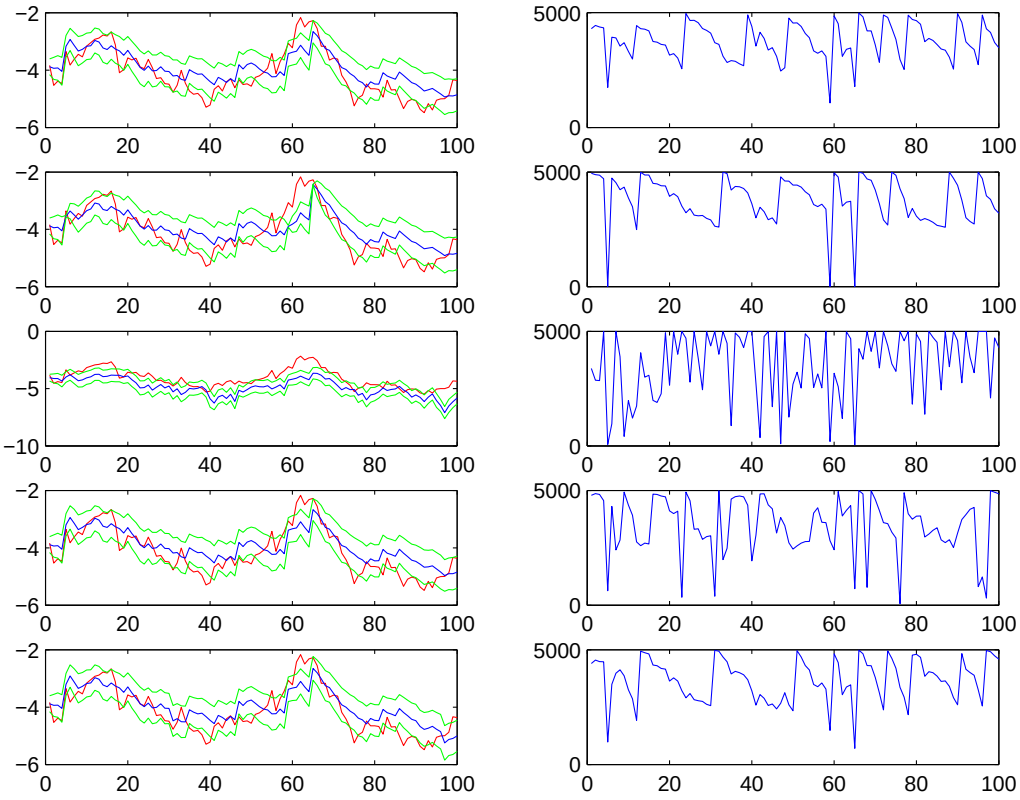

Figure 3.2: Results for Five Different Auxiliary Particle Filters. The graphs on the left show, the true values of latent variable (red), filtering estimate (blue) and +/- 1 standard deviation band (green). The graphs on the right show the ESS for each APF algorithm.

CHAPTER 4

PARTICLE MCMC

In previous chapters, we discussed particle filtering methods to infer latent states, under the assumption that model parameters are known. However in real life, we need to estimate both the parameters and the latent states. At first sight we may think of extending the state with the unknown parameters $Z_t = (X_t, \Theta)$ and then using particle filtering to estimate the sequence of posterior distributions $\{p(Z_{1:t}|y_{1:t})\}_{t=1}^{T}$ by using an initial density $p(\theta_1)\mu_{\theta_1}(x_1)$ and transition density $f_{\theta_t}(x_t|x_{t-1})\delta_{\theta_{t-1}}(\theta_t)$, i.e. $\theta_t = \theta_{t-1}$. However this means that the parameter space would only be explored at the initialization of the algorithm. As a result of the successive resampling steps, after a certain time t, the approximation $\hat{p}(\theta|y_{1:t})$ will only contain a single unique value for Θ. Although some methods (e.g. resample–move algorithm) are proposed in the literature to partially solve this inevitable degeneracy problem, this is a difficult problem and it cannot be considered to have been solved in full generality [44, 18]. For a survey of SMC methods for parameter estimation, see [44].

On the other hand, we also discussed MCMC methods that directly aim to estimate both the parameters and the latent states. However, the performance of the MCMC methods critically depend on finding good proposal distributions that approximate the posterior or, more commonly, some lower dimensional components of it. If the proposal distributions explore the state space poorly and/or if highly correlated variables are updated independently, then the resulting MCMC estimates will be poor. In the previous chapter we also saw that, although finding good proposals for full conditional distributions of parameters, $p(\Theta|X, Y)$, may be easier, finding good high–dimensional proposals for full conditionals of the latent states, $p(X|\Theta, Y)$, is not an easy task.

Very recently, *Particle MCMC* methods are proposed by [2] that aim to use SMC methods to build efficient high–dimensional proposal distributions to be used within an MCMC setting. To illustrate this idea, now assume an MCMC setting in which we sequentially sample from the full conditional of parameters and latent states. Typically sampling from $p(\Theta|X, Y)$ is easy. For $p(X|\Theta, Y)$, we can use a Metropolis–Hastings algorithm by proposing new states from a proposal distribution[1], $q(X^{(g)}|\Theta^{(g)}, Y)$ at iteration g, and then accepting it with the corresponding acceptance probability. At this stage, since the parameter values are fixed, we may think of using SMC to sample from $\hat{p}(X|\Theta, Y)$ which can be an efficient approximation of $p(X|\Theta, Y)$. Although SMC

[1] This density can also depend on $X^{(g-1)}$, as in random–walk samplers.

algorithms allow us to sample from this approximation, to calculate the acceptance probability, we also need to evaluate this density which is not available analytically.

Particle MCMC methods circumvent this problem by considering target distributions on an extended space which includes all the random variables that are produced by the SMC algorithm. Using this idea, [2] propose three novel algorithms called particle independent Metropolis–Hastings (PIMH) sampler, particle marginal Metropolis–Hastings sampler (PMMH) and particle Gibbs (PG) sampler. We will summarize the particle independent Metropolis–Hastings and particle Gibbs samplers here.

4.1 Particle Independent Metropolis–Hastings Sampler

To illustrate the ideas we need to work on an extended space just like we did in block sampling section. But now we will use a slightly different notation as our main focus in this section is the ancestral lineages as opposed to how many times a particle is resampled. We will represent a generic resample–sample filtering algorithm with the following steps:

- At time $t = 1$
  - Sample $X_1^i \sim q_1(\cdot)$
  - Weight $w_1(X_1^i)$ and $W_1^i \propto w_1(X_1^i)$
- For time $t \geq 2$
  - Resampling step is equivalent to selecting an index $A_t^i \in \{1, \ldots, N\}$ for each particle according to its weight W_{t-1}^i. For this, different schemes (e.g. multinomial, systematic) can be used. This step can be represented as sampling indices from a generic discrete distribution:
    $$(A_t^1, \ldots, A_t^N) \sim r(\cdot|W_{t-1}^1, \ldots, W_{t-1}^N)$$
    or with a more compact notation $\vec{A}_t \sim r(\cdot|\vec{W}_{t-1})$.
  - Propagate $X_t^i \sim q_t(\cdot|X_{1:t-1}^{A_{t-1}^i})$ and set $X_{1:t}^i \leftarrow (X_{1:t-1}^{A_{t-1}^i}, X_t^i)$, i.e. we are only extending the paths for the particles that survive the last resampling step.
  - Weight $w_t(X_{1:t}^i)$ and $W_t^i \propto w_t(X_{1:t}^i)$

At the end of the SMC algorithm, we can obtain the usual approximation for the target distribution as:

$$\hat{\pi}(x_{1:T}) = \sum_{i=1}^{N} W_T^i \delta_{X_{1:T}^i}(x_{1:T}) \tag{4.1}$$

and for the integrating constant as:

$$\hat{Z}_T = \prod_{t=1}^{T} \frac{1}{N} \sum_{i=1}^{N} w_t(X_{1:t}^i) \tag{4.2}$$

This representation clearly shows that the joint distribution of all particles $(\vec{X}_1, \ldots, \vec{X}_T)$, where $\vec{X}_t = (X_t^i, \ldots, X_t^N)$, and all indices $(\vec{A}_1, \ldots, \vec{A}_{T-1})$ is given by:

$$\psi(\vec{x}_1, \ldots, \vec{x}_T, \vec{a}_1, \ldots, \vec{a}_{T-1}) = \left[\prod_{i=1}^{N} q_1(x_1^i)\right] \prod_{t=2}^{T} \left\{ r(\vec{a}_{t-1}|\vec{w}_{t-1}) \prod_{i=1}^{N} q_t(x_t^i|x_{1:t-1}^{a_{t-1}^i}) \right\} \tag{4.3}$$

where $\vec{w}_{t-1}$ are the realized values of *normalized* importance weights.

Now we need to introduce one more notation to represent the ancestral lineages. First note that each X_t^i is generated from its ancestors $X_{1:t-1}^{A_{t-1}^i}$, not from $X_{1:t-1}^i$. Therefore we need to keep track of ancestral lineages. After performing the SMC algorithm until time T, we will have N final time particles $\{X_T^i\}_{i=1}^N$. For each final time particle $X_T^k, k \in \{1, \ldots, N\}$, we have an associated path that includes all ancestors of X_T^k. To represent this path, let B_t^k denote the index of the ancestor of X_T^k at time t. This yields $X_{1:T}^k = (X_1^{B_1^k}, X_2^{B_2^k}, \ldots, X_T^{B_T^k})$ where we have $B_T^k = k$ by definition. $B_{1:T}^k = (B_1^k, \ldots, B_T^k)$ represents the ancestral lineage of the particle X_T^k.

SMC methods allows us to obtain samples from a distribution $q_\Theta(x_{1:T}|y_{1:T})$ which is an approximation to the exact full conditional $p_\Theta(x_{1:T}|y_{1:T})$. In order to use the SMC within an independence Metropolis–Hastings algorithm, we need to do two things: First we need to sample from the proposal $q_\Theta(x_{1:T}|y_{1:T})$ and then we need to analytically evaluate this density to calculate the Metropolis–Hastings acceptance probability.

Sampling From the Proposal Distribution:

After implementing the SMC algorithm, using the realized values of the particles and the weights we obtain the empirical measure:

$$\hat{\pi}(x_{1:T}) = \sum_{i=1}^{N} W_T^i \delta_{X_{1:T}^i}(x_{1:T})$$

which is just a realization of a random measure approximating the exact full conditional. Sampling a path $X_{1:T}^*$ from this approximation is straightforward:

- Sample an index $K \sim F(\cdot|W_T^1, \ldots, W_T^N)$ where:

$$F(k|W_T^1, \ldots, W_T^N) \propto W_T^k \tag{4.4}$$

- Given $K = k$, select $X_{1:T}^* = X_{1:T}^k = (X_1^{B_1^k}, X_2^{B_2^k}, \ldots, X_T^{B_T^k})$.

Evaluating the Proposal Distribution:

To calculate the acceptance probability, we need the law of $X_{1:T}^*$. *Conditional* on the output of an SMC (i.e. for a specific realization $(\vec{x}_1, \ldots, \vec{x}_T, \vec{a}_1, \ldots, \vec{a}_{T-1})$), the law of this path is given by $\hat{\pi}(x_{1:T})$ generated by the SMC algorithm. However, we need the

unconditional law of this path, i.e. law over all possible values of $(\vec{X}_1, \ldots, \vec{X}_T, \vec{A}_1, \ldots, \vec{A}_{T-1})$. This law is given by:

$$q_\Theta(x_{1:T}|y_{1:T}) = E_\psi[\hat{\pi}(x_{1:T})]$$

where the expectation is taken with respect to the joint distribution $\psi(\cdot)$ given in (4.3), since this is the data generating process for the SMC algorithm. Unfortunately this distribution is not available from the output of the SMC algorithm.

To circumvent this problem, [2] suggest to work on the extended space which is the space of $(K, \vec{X}_1, \ldots, \vec{X}_T, \vec{A}_1, \ldots, \vec{A}_{T-1})$. Combining (4.3) and (4.4), we can obtain the proposal density on this extended space as:

$$\tilde{q}(k, \vec{x}_1, \ldots, \vec{x}_T, \vec{a}_1, \ldots, \vec{a}_{T-1}) = w_T^k \psi(\vec{x}_1, \ldots, \vec{x}_T, \vec{a}_1, \ldots, \vec{a}_{T-1}) \tag{4.5}$$

The extended target suggested by [2] is given by:

$$\tilde{\pi}(k, \vec{x}_1, \ldots, \vec{x}_T, \vec{a}_1, \ldots, \vec{a}_{T-1}) = \frac{\pi(x_{1:T}^k)}{N^T} \frac{\psi(\vec{x}_1, \ldots, \vec{x}_T, \vec{a}_1, \ldots, \vec{a}_{T-1})}{q_1(x_1^{b_1^k}) \prod_{t=2}^T r(b_{t-1}^k|\vec{w}_{t-1}) q_t(x_t^{b_t^k}|x_{t-1}^{b_{t-1}^k})} \tag{4.6}$$

where $x_{1:T}^k$ is the realized value for the sampled path $X_{1:T}^*$ and the factor $1/N^T$ corresponds to the uniform distribution on the set $\{1, \ldots, N\}^T$ for the variables $(K, A_1^{B_2^K}, \ldots, A_{T-1}^{B_T^K})$. This extended distribution has two important properties. First it admits our target distribution of interest as its marginal, i.e. if we generate samples $(K, \vec{X}_1, \ldots, \vec{X}_T, \vec{A}_1, \ldots, \vec{A}_{T-1}) \sim \tilde{\pi}(\cdot)$, then $X_{1:T}^K$ has marginally the distribution $\pi(\cdot)$.

Second, working with these extended target and proposal distributions allows us to avoid the need for direct evaluation of $q_\Theta(x_{1:T}|y_{1:T})$. This is obtained by the following important result (see proof of theorem 2 in [2]):

$$\frac{\tilde{\pi}(k, \vec{x}_1, \ldots, \vec{x}_T, \vec{a}_1, \ldots, \vec{a}_{T-1})}{\tilde{q}(k, \vec{x}_1, \ldots, \vec{x}_T, \vec{a}_1, \ldots, \vec{a}_{T-1})} = \frac{\hat{Z}_T}{Z_T} \tag{4.7}$$

where $\hat{Z}_T$ is the estimate of the integrating constant and Z_T is the true value of it. $\hat{Z}_T$ can easily be calculated from the output of the SMC algorithm as given in (4.2).

Implementing Particle Independent Metropolis–Hastings Sampler :

Now, by combining the above results, we can fully define particle independent Metropolis–Hastings sampler to sample from $p_\Theta(x_{1:T}|y_{1:T})$. For this, at each iteration of the MCMC algorithm, we run an SMC which gives the realized values for

$$M := (K, \vec{X}_1, \ldots, \vec{X}_T, \vec{A}_1, \ldots, \vec{A}_{T-1})$$

the sampled path $X_{1:T}^K$ and estimate for the normalizing constant $\hat{Z}_T$ as its output. The algorithm iterates as follows: Assume that we are at iteration g and from the previous iteration, the values $M(g-1)$, the sampled path $X_{1:T}^K(g-1)$ and $\hat{Z}_T(g-1)$ are available:

- Sample $M^* \sim \tilde{q}(\cdot)$ and given M^* obtain $X_{1:T}^{K*}$ and calculate $\hat{Z}_T^*$. This means that we run an SMC algorithm, obtain the empirical measure $\hat{\pi}(x_{1:T})$, sample from this measure $X_{1:T}^* \sim \hat{\pi}(x_{1:T})$, and calculate $\hat{Z}_T^*$ using the output of SMC algorithm.
- Calculate the Metropolis–Hastings acceptance probability as:

$$p = min\left\{1, \frac{\tilde{\pi}(M^*)\tilde{q}(M(g-1))}{\tilde{\pi}(M(g-1))\tilde{q}(M^*)}\right\} = min\left\{1, \frac{\tilde{\pi}(M^*)/\tilde{q}(M^*)}{\tilde{\pi}(M(g-1))/\tilde{q}(M(g-1))}\right\}$$
$$= min\left\{1, \frac{\hat{Z}_T^*/Z}{\hat{Z}_T(g-1)/Z}\right\} = min\left\{1, \frac{\hat{Z}_T^*}{\hat{Z}_T(g-1)}\right\} \tag{4.8}$$

- Set:

$$(X_{1:T}^K(g), \hat{Z}_T(g)) = \begin{cases} (X_{1:T}^{K*}, \hat{Z}_T^*) & \text{with probability } p \\ (X_{1:T}^K(g-1), \hat{Z}_T(g-1)) & \text{with probability } 1-p \end{cases} \tag{4.9}$$

[2] showed that, under certain conditions, this algorithm creates a Markov chain whose invariant distribution is the extended target $\tilde{\pi}(\cdot)$. Therefore, as the number of iterations goes to infinity, $G \to \infty$, the marginal distribution of the sampled paths $X_{1:T}^K(G)$ converges to the correct conditional posterior distribution $p_\Theta(x_{1:T}|y_{1:T})$.

Above result shows that, by using an SMC algorithm, it is possible to efficiently sample from the full conditional distribution of latent states given the known values of parameters. This is a powerful method that allows us to sample the latent states as a block even in non–trivial problems such as non–linear non–Gaussian state–space models. Additionally it reduces the problem of finding efficient proposals that directly approximate $p_\Theta(x_{1:T}|y_{1:T})$ to designing efficient SMC algorithms that targets $p_\Theta(x_{1:T}|y_{1:T})$, which only requires designing low dimensional proposal distributions that will be used sequentially within SMC algorithm. Now we can fully define a particle independent Metropolis–Hastings sampler to sample from $p(X|\Theta, Y)$, as given in Algorithm 4.1. [2] showed that this algorithm admits $p(\Theta, X|Y)$ as invariant density.

4.2 Particle Gibbs Sampler

In a standard MCMC algorithm with Gibbs sampler, we sequentially sample from the conditional posteriors $p(\Theta|X, Y)$ and $p(X|\Theta, Y)$. In most cases, sampling from $p(\Theta|X, Y)$ is relatively straightforward if full conditionals turn out to be standard distributions. However sampling from $p(X|\Theta, Y)$ as a block is much more challenging. As a naive method, if we replace sampling from $p(X|\Theta, Y)$ with sampling from the empirical measure of an SMC output, then the resulting Markov chain does not admit the relevant posterior as the invariant distribution.

Using the extended space framework that we discuss in previous section, [2] proposed Particle Gibbs sampler in which we sample from $p(X|\Theta, Y)$ as a block using a specific

Algorithm 4.1 Particle Independent Metropolis–Hastings Sampler

1: Initialization:

- Given Θ, run an SMC targeting $p_\Theta(x_{1:T}|y_{1:T})$
- Sample from the empirical measure $X_{1:t}^{(0)} \sim \hat{p}_\Theta(x_{1:T}|y_{1:T})$
- Calculate $\hat{Z}_t^{(0)}$ from the SMC output

2: **for** $g = 1$ to G **do**
3: Run an SMC targeting $p_\Theta(x_{1:T}|y_{1:T})$
4: Sample from the empirical measure $X_{1:t}^{(can)} \sim \hat{p}_\Theta(x_{1:T}|y_{1:T})$
5: Calculate $\hat{Z}_t^{(can)}$ from the SMC output
6: Calculate the acceptance probability as $p = min\left\{1, \hat{Z}_t^{(can)}/\hat{Z}_t^{(g-1)}\right\}$
7: Set $(X_{1:t}^{(g)}, \hat{Z}_t^{(g)}) = (X_{1:t}^{(can)}, \hat{Z}_t^{(can)})$ with probability p, otherwise set $(X_{1:t}^{(g)}, \hat{Z}_t^{(g)}) = (X_{1:t}^{(g-1)}, \hat{Z}_t^{(g-1)})$.
8: **end for**

type of SMC algorithm. They introduced an SMC algorithm with *conditional SMC update*, where a specified path $X_{1:T}$ with ancestral lineage $B_{1:T}$ is ensured to survive all the resampling steps, whereas the remaining $N-1$ particles are generated by usual SMC iterations. A generic particle Gibbs algorithm is given in Algorithm 4.2.

Algorithm 4.2 Particle Gibbs Sampler

1: Initialization:

- Initialize $\Theta^{(0)}$
- Run an SMC targeting $p_{\Theta^{(0)}}(x_{1:T}|y_{1:T})$
- Sample from the empirical measure $X_{1:T}^{(0)} \sim \hat{p}_{\Theta^{(0)}}(x_{1:T}|y_{1:T})$, thus implicitly select $B_{1:T}^{(0)}$

2: **for** $g = 1$ to G **do**

3: Sample $\Theta^{(g)} \sim p(\Theta|x_{1:T}^{(g-1)}, y_{1:T})$

4: Run a conditional SMC targeting $p_{\Theta^{(g)}}(x_{1:T}|y_{1:T})$ conditional on $X_{1:T}^{(g-1)}$ using the following steps:

- For $t = 1$:
  - For $k \neq B_1^{(g-1)}$ sample $X_1^k \sim q_{\Theta^{(g)}}(x_1|y_1)$
  - For $i = 1, 2, \ldots, N$ compute weights $w_1(x_1^i)$
  - For $i = 1, 2, \ldots, N$ normalize weights $W_1^i \propto w_1(x_1^i)$
- For $t = 2, \ldots, T$:
  - For $k \neq B_t^{(g-1)}$ sample $A_{t-1}^k \sim F(\cdot|\overrightarrow{W}_{t-1})$
  - For $k \neq B_t^{(g-1)}$ sample $X_t^k \sim q(x_t|y_t, x_{t-1}^{A_{t-1}^k})$
  - For $i = 1, 2, \ldots, N$ compute weights $w_t(x_{1:t}^i)$
  - For $i = 1, 2, \ldots, N$ normalize weights $W_t^i \propto w_t(x_{1:t}^i)$

5: Sample $X_{1:T}^{(g)} \sim \hat{p}_{\Theta^{(g)}}(x_{1:T}|y_{1:T})$, thus implicitly select $B_{1:T}^{(g)}$

6: **end for**

CHAPTER 5

PMCMC FOR A TIME CHANGED LÉVY MODEL

MCMC approach offers a flexible estimation method for models that include stochastic volatility and jumps. Accordingly, following the early paper of Jacquier, et. al. [40], an abundance of MCMC algorithms are proposed in the literature for such complex models. For instance [40] and [45] develop MCMC algorithms for an autoregressive stochastic volatility model. [41] provides algorithms for stochastic volatility models with leverage and fat tails. Furthermore [49] uses MCMC algorithms for different stock price models with stochastic volatility plus Poisson, variance–gamma and log–stable based jumps.

On the other hand, although PMCMC approach offers a compelling alternative to traditional MCMC and attracted a huge interest from the academic society, it is relatively new and the literature on PMCMC for stochastic volatility and jump models is not voluminous. In their original paper, [2] develops PMCMC algorithms for a stochastic volatility model where the volatility process is driven a Lévy process. [39] uses PMCMC algorithm for Hull–White type stochastic volatility model. [21] and [33] develop PMCMC algorithms for different autoregressive stochastic volatility models. Therefore we aim to contribute to the existing literature by developing PMCMC algorithms for a complex model with stochastic volatility and jumps in this chapter.

5.1 A Time Changed Lévy Model

In this chapter we will develop MCMC and PMCMC algorithms for a stock price model that includes stochastic volatility and jumps driven by a time changed Lévy process. Lévy processes are continuous time stochastic processes with stationary and independent increments. Brownian motion and compound Poisson are two popular examples of Lévy processes. However the family of Lévy processes is a rich one that encompasses many other processes that offers more flexible modelling. Lévy processes are quite flexible such that they allow discontinuous sample paths, non-Gaussian increments and more flexible jump structures that may have infinite activity.

Lévy processes can have finite or infinite activity. In the latter case, we have an infinite number of jumps within any finite time interval. In this case the sample path of the process may have finite or infinite variation. In the infinite variation case, the sum of

absolute increments is infinite over a finite time interval.

We can apply a stochastic time change to a Lévy process by allowing the time governed by an increasing stochastic process. This allows us to obtain a time inhomogeneous process which matches the observed characteristics of asset prices. An example of time changed Lévy process is a variance Gamma process, proposed by Madan, et. al. [52]. It is obtained by subordinating an arithmetic Brownian motion with drift by an independent Gamma process. That is:

$$VG(t) = \varphi\Gamma_t + \psi W_{\Gamma_t}$$

where φ and ψ are the drift and volatility of the arithmetic Brownian motion, and Γ_t is an independent Gamma process with unit mean rate and variance rate λ. With this time change, we turn the original diffusion process into a jump process. The process is an infinite activity, but finite variation model. Thus the process incorporates finitely many large jumps as well as infinitely many small jumps. Empirically, the variance Gamma process captures stock market dynamics better than finite activity jump models (e.g. Poisson models).

In general, estimation of Lévy processes can be quite challenging since for some Lévy processes we do not know the probability distribution in closed form and higher order moments do not exist. This rules out likelihood or moment based estimation methods. On the other hand, simulation based Bayesian estimation methods fits this problem well. We will use MCMC and PMCMC methods for estimating a time changed Lévy process from discretely observed data.

A second component that we use in our model is a Heston [36] type stochastic volatility process. Here, the variance ν is modelled by a square root process given by:

$$d\nu_t = \kappa(\theta - \nu_t)dt + \gamma\sqrt{\nu_t}dW_t$$

where κ is the mean reversion rate, θ is the long run average variance γ is the volatility of volatility and W_t is a Brownian motion. In this model volatility is stochastic, auto regressive and mean reverting. This feature allows volatility clustering as well as kurtosis in stock returns. It is also possible to add leverage effect to the model by using correlated Brownian motions for the stock price and the volatility process. The transition density is a non-central Gamma distribution.

By combining stochastic volatility and jump parts, we assume the following model for the stock price:

$$\begin{aligned} d\log(S_t) &= \mu dt + \sqrt{\nu_t}dW_t^1 + dVG_t && (5.1)\\ d\nu_t &= \kappa(\theta - \nu_t)dt + \gamma\sqrt{\nu_t}dW_t^2 && (5.2)\\ dVG_t &= \varphi d\Gamma_t + \psi dZ_{\Gamma_t} && (5.3) \end{aligned}$$

where W^1 and W^2 are two correlated Brownian motions with $cor(dW_t^1, dW_t^2) = \rho$. Z is also another Brownian motion, independent from others, and Γ_t is a Gamma process with $d\Gamma_t \sim Ga(1/\lambda, \lambda)$. With this structure, the model incorporates both mean reverting stochastic volatility and infinite activity jumps in returns. This allows

us to capture any volatility clustering, fat tailed returns, leverage effect and jumps that may be in the actual price process that is modelled.

First order Euler approximation over a unit time period gives the following discrete time equivalent for log returns as defined by $Y_t = log(S_t) - log(S_{t-1})$:

$$\begin{aligned} Y_t &= \mu + \sqrt{\nu_{t-1}}\epsilon_t + J_t & (5.4)\\ \nu_t &= \nu_{t-1} + \kappa(\theta - \nu_{t-1}) + \gamma\sqrt{\nu_{t-1}}\varepsilon_t & (5.5)\\ J_t &= \varphi G_t + \psi\sqrt{G_t}\eta_t & (5.6) \end{aligned}$$

where $\epsilon_t \sim N(0,1)$, $\varepsilon_t \sim N(0,1)$, $cor(\epsilon_t, \varepsilon_t) = \rho$, $\eta_t \sim N(0,1)$ and $G_t \sim Ga(1/\lambda, \lambda)$. For the jump part we used the property of Brownian motion that, conditional on $d\Gamma_t$, $dZ_{\Gamma_t} \sim N(0, d\Gamma_t)$. The model defines a state–space form where the first equation is the observation equation and the other two equations define the state evolution. The latent states, that are latent volatility, jumps and background Gamma variate, are non–Gaussian. Conditional on latent states, returns are conditionally Gaussian. Therefore the system is a non–linear non–Gaussian system.

A simulation of the model is given in Figure 5.1 and Figure 5.2.

5.2 MCMC

In this section we develop an MCMC algorithm for the above mentioned model. For this we need to derive the posterior distribution of parameters and latent states. The model includes eight parameters $\Theta = (\mu, \kappa, \theta, \gamma, \rho, \varphi, \psi, \lambda)$ and three latent variables $X = (\nu, J, G)$.

The posterior is given by:

$$\begin{aligned} p(\Theta, \nu, J, G|Y) &\propto p(Y, \Theta, \nu, J, G) & (5.7)\\ &= p(\Theta)p(\nu_0)p(G|\Theta)p(J|G,\Theta)p(Y,\nu|J,\Theta)\\ &= p(\Theta)p(\nu_0)\prod_{t=1}^{T} Ga(G_t|1/\lambda,\lambda)N(J_t|\varphi G_t, \psi^2 G_t)\\ &\quad N_2(Y_t, \nu_t|(\mu + J_t, \nu_{t-1} + \kappa(\theta - \nu_{t-1})), (1, \gamma\rho; \gamma\rho, \gamma^2)) \end{aligned}$$

where $N(.|.)$,$N_2(.|.)$ and $Ga(.|.)$ represent the density functions for normal, bivariate normal and Gamma distributions.

In setting priors for parameters and the initial variance, we select conjugate prior distributions if possible, and use parameters that lead to uninformative/difuse prior distributions. As proposed by [49], we transform (ρ, γ) to $(\phi = \rho\gamma, w = \gamma^2(1-\rho^2))$ to make posteriors tractable. The priors are given as:

$$\begin{aligned} p(\mu) &\propto N(a_0 = 0, A_0 = 1)\\ p(\kappa) &\propto N(b_0 = 0, B_0 = 1)1_{(\kappa>0)}\\ p(\theta) &\propto N(c_0 = 0, C_0 = 1)1_{(\theta>0)} \end{aligned}$$

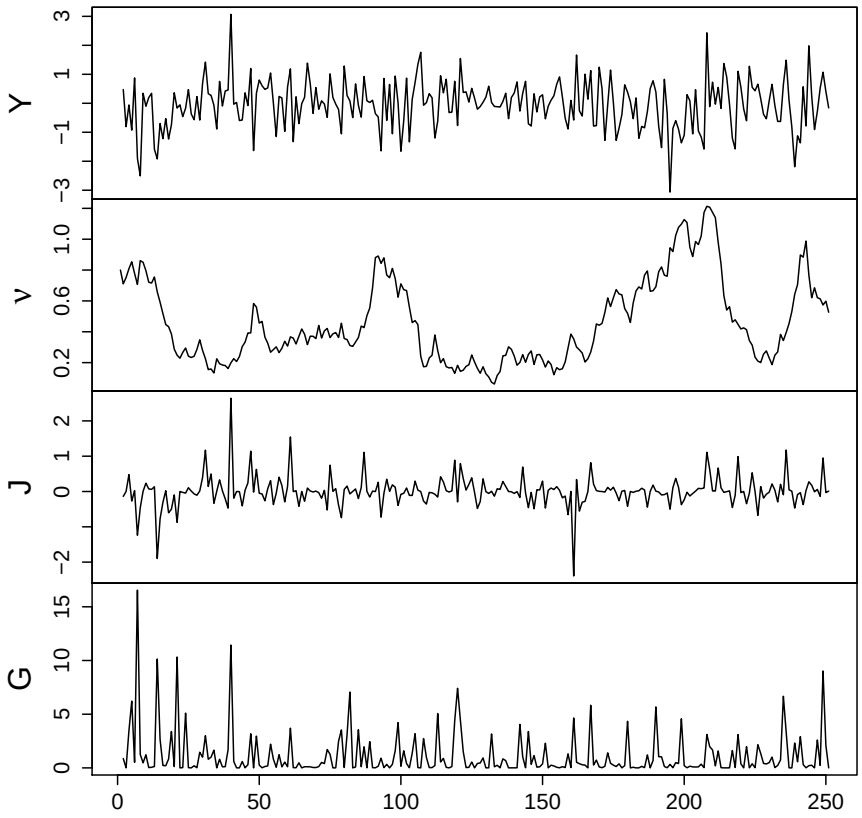

Figure 5.1: Simulated values of Y_t, ν_t,J_t and G_t from the model, using parameters $(\mu = 0.05, \kappa = 0.015, \theta = 0.8, \gamma = 0.1, \rho = -0.4, \varphi = -0.01, \psi = 0.4, \lambda = 3)$

$$\begin{aligned} p(w) &\propto IG(d_0 = 2, D_0 = 200) \\ p(\phi) &\propto N(e_0 = 0, wE_0 = w/2) \\ p(\varphi) &\propto N(p_0 = 0, P_0 = 1) \\ p(\psi^2) &\propto IG(q_0 = 2.5, Q_0 = 5) \\ p(\lambda) &\propto IG(r_0 = 10, R_0 = 1/10) \\ p(\nu_0) &\propto 1_{(\nu_0 > 0)} \end{aligned}$$

With this priors, the full conditional posteriors become as follows[1]:

$$p(\mu|.) \propto N(\frac{a_1}{A_1}, \frac{1}{A_1})$$

[1] Variables with an index 0 represents the corresponding parameters for the prior distribution. Detailed derivations are skipped.

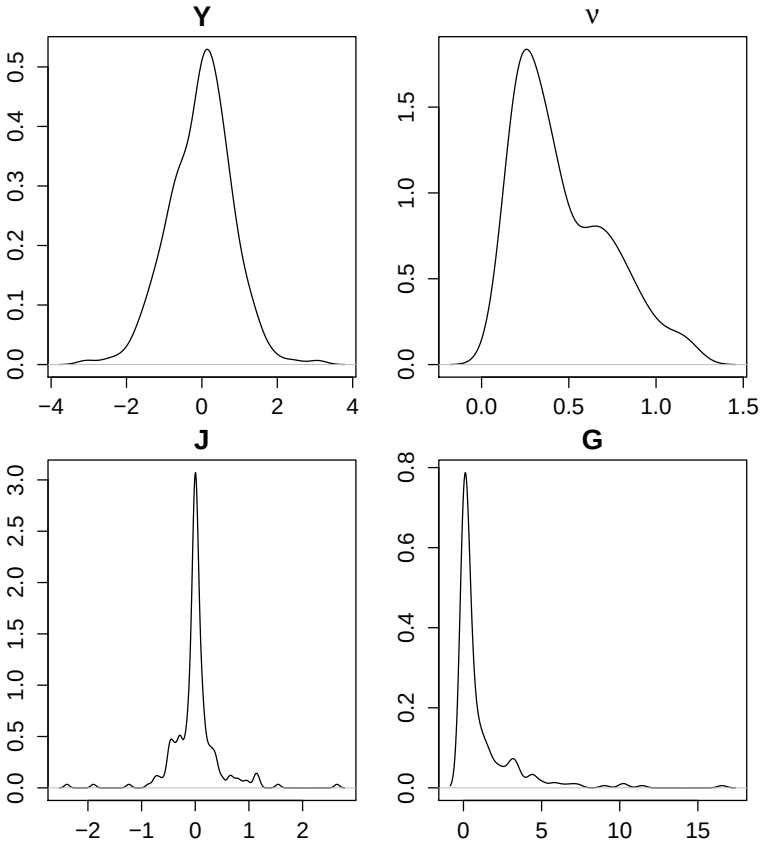

Figure 5.2: Density plots of Y_t, ν_t,J_t and G_t from the model, using parameters ($\mu = 0.05, \kappa = 0.015, \theta = 0.8, \gamma = 0.1, \rho = -0.4, \varphi = -0.01, \psi = 0.4, \lambda = 3$)

$$
\begin{aligned}
a_1 &= \frac{a_0}{A_0} + \frac{1}{1-\rho^2}\sum_{t=1}^{T}\frac{1}{\nu_{t-1}}(Y_t - J_t - \rho\frac{A_{2,t}}{\gamma}) \\
A_1 &= \frac{1}{A_0} + \sum_{t=1}^{T}\frac{1}{\nu_{t-1}(1-\rho^2)} \\
A_{2,t} &= \nu_t - \nu_{t-1} - \kappa(\theta - \nu_{t-1}) \\
p(\kappa|.) &\propto N(\frac{b_1}{B_1}, \frac{1}{B_1})1_{(\kappa>0)} \\
b_1 &= \frac{b_0}{B_0} + \frac{1}{(1-\rho^2)\gamma}\sum_{t=1}^{T}\frac{(\theta - \nu_{t-1})}{\nu_{t-1}}[\frac{\nu_t - \nu_{t-1}}{\gamma} - \rho(Y_t - \mu - J_t)] \\
B_1 &= \frac{1}{B_0} + \frac{1}{\gamma^2(1-\rho^2)}\sum_{t=1}^{T}\frac{(\theta - \nu_{t-1})^2}{\nu_{t-1}} \\
p(\theta|.) &\propto N(\frac{c_1}{C_1}, \frac{1}{C_1})1_{(\theta>0)}
\end{aligned}
$$

$$
\begin{aligned}
c_1 &= \frac{c_0}{C_0} + \frac{\kappa}{\gamma(1-\rho^2)} \sum_{t=1}^{T} \frac{1}{\nu_{t-1}} [\frac{\kappa\nu_{t-1} + \nu_t - \nu_{t-1}}{\gamma} - \rho(Y_t - \mu - J_t)] \\
C_1 &= \frac{1}{C_0} + \frac{\kappa^2}{\gamma^2(1-\rho^2)} \sum_{t=1}^{T} \frac{1}{\nu_{t-1}} \\
p(w|.) &\propto IG(d_1, D_1) \\
d_1 &= d_0 + T/2 \\
D_1 &= D_0 + \frac{e_0^2}{2E_0} + \sum_{t=1}^{T} \frac{(\nu_t - \nu_{t-1} - \kappa(\theta - \nu_{t-1}))^2}{2\nu_{t-1}} - \frac{e_1^2}{2E_1} \\
p(\phi|.) &\propto N(\frac{e_1}{E_1}, \frac{w}{E_1}) \\
e_1 &= \frac{e_0}{E_0} + \sum_{t=1}^{T} \frac{(Y_t - \mu - J_t)(\nu_t - \nu_{t-1} - \kappa(\theta - \nu_{t-1}))}{\nu_{t-1}} \\
E_1 &= \frac{1}{E_0} + \sum_{t=1}^{T} \frac{(Y_t - \mu - J_t)^2}{\nu_{t-1}} \\
p(\nu_0|.) &\propto 1_{(\nu_0>0)} \frac{1}{\nu_0} \exp(-\frac{f_1}{\nu_0}) \exp(-F_1\nu_0) \\
f_1 &= \frac{(Y_1 - \mu - J_1)^2}{2(1-\rho^2)} - \frac{\rho(Y_1 - \mu - J_1)(\nu_1 - \kappa\theta)}{\gamma(1-\rho^2)} + \frac{(\nu_1 - \kappa\theta)^2}{2\gamma^2(1-\rho^2)} \\
F_1 &= \frac{(\kappa-1)^2}{2\gamma^2(1-\rho^2)} \\
p(\nu_t|.) &\propto 1_{(\nu_t>0)} \frac{1}{\nu_t} \exp\{\frac{M_1}{\nu_t} + M_2\nu_t - M_3\nu_t^2\} \\
M_1 &= \frac{1}{\gamma(1-\rho^2)} \{\rho(Y_{t+1} - \mu - J_{t+1})(\nu_{t+1} - \kappa\theta) - \frac{\gamma}{2}(Y_{t+1} - \mu - J_{t+1})^2 - \frac{(\nu_{t+1} - \kappa\theta)^2}{2\gamma}\} \\
M_2 &= \frac{1}{\gamma(1-\rho^2)} \{\frac{\rho(Y_t - \mu - J_t)}{\nu_{t-1}} - \frac{(\kappa-1)^2}{2\gamma} + \frac{\kappa\theta + (1-\kappa)\nu_{t-1}}{\gamma\nu_{t-1}}\} \\
M_3 &= \frac{1}{2\gamma^2\nu_{t-1}(1-\rho^2)} \\
p(\nu_T|.) &\propto N(h_1, H_1) 1_{(\nu_T>0)} \\
h_1 &= \nu_{T-1} + \kappa(\theta - \nu_{T-1}) + \rho\gamma(Y_T - \mu - J_T) \\
H_1 &= (1-\rho^2)\gamma^2\nu_{T-1} \\
p(\varphi|.) &\propto N(\frac{p_1}{P_1}, \frac{1}{P_1}) \\
p_1 &= \frac{p_0}{P_0} + \frac{1}{\psi^2} \sum_{t=1}^{T} J_t \\
P_1 &= \frac{1}{P_0} + \frac{1}{\psi^2} \sum_{t=1}^{T} G_t
\end{aligned}
$$

$$
\begin{aligned}
p(\psi^2|.) &\propto IG(q_1, Q_1) \\
q_1 &= q_0 + \frac{T}{2} \\
Q_1 &= Q_0 + \sum_{t=1}^{T} \frac{(J_t - \varphi G_t)^2}{2G_t} \\
p(\lambda|.) &\propto [\frac{1}{\lambda^{1/\lambda}\Gamma(1/\lambda)}]^T \lambda^{-r_0-1} [\prod_{t=1}^{T} G_t]^{1/\lambda} \exp\{-\frac{1}{\lambda}(R_0 + \sum_{t=1}^{T} G_t)\} \\
p(J_t|.) &\propto N(\frac{r_1}{R_1}, \frac{1}{R_1}) \\
r_1 &= \frac{\varphi}{\psi^2} + \frac{1}{(1-\rho^2)\nu_{t-1}}(Y_t - \mu - \frac{\rho(\nu_t - \nu_{t-1} - \kappa(\theta - \nu_{t-1}))}{\gamma}) \\
R_1 &= \frac{1}{\psi^2 G_t} + \frac{1}{(1-\rho^2)\nu_{t-1}} \\
p(G_t|.) &\propto G_t^{1/\lambda - 3/2} \exp\{-G_t(\frac{1}{\lambda} + \frac{\varphi^2}{2\psi^2}) - \frac{1}{G_t}\frac{J_t^2}{2\psi^2}\}
\end{aligned}
$$

All posteriors except the ones for ν_0, ν_t, λ and G_t turn out to be standard distributions, which are easy to sample from. However, in order to implement a Gibbs algorithm, we also need to effectively sample from these non–standard distributions as well. For this we will use *Adaptive Rejection Metropolis Sampling* (see [27]).

ARMS is an extension to ARS method (see Algorithm 2.3). The main idea of ARMS is to use the envelope function generated by adaptive rejection sampling method (see Figure 2.1) even if the target density is not log–concave. Here the idea is illustrated using the notation of Algorithm 2.3. In this case the envelope generated by tangent lines may not provide an upper bound for the target density, i.e. $G(h_t) \geq \pi(h_t)$ does not always hold. Therefore G should be called as a pseudo–envelope. If we implement the usual accept–reject sampling using this pseudo–envelope, then the resulting samples will have the density $f(h_t) \propto \min(G(h_t), \pi(h_t))$, not $\pi(h_t)$ as we had in log–concave case. However to correct this, ARS step can be incorporated within a Metropolis–Hastings algorithm using $f(h_t)$ as the proposal density. The implementation[2] of ARMS for our ARSV model of Chapter 3 is given in Algorithm 5.1 (replaces lines 3 to 10 in Algorithm 2.3).

Using the posterior densities we previously obtain and with the help of ARMS method for sampling from non–standard densities, our full MCMC algorithm is defined as in Algorithm 5.2.

[2] We implement ARMS using the R package HI [57].

Algorithm 5.1 MCMC Algorithm with ARMS for Volatility

1: **for** $t = 1$ to T **do**
2: Construct the pseudo–envelope $G(h_t)$ using ARS methods
3: **repeat**
4: Sample $h_t^{(can)} \propto G(h_t)$
5: Accept it with probability $min\left\{1, p(h_t^{(can)})/G(h_t^{(can)})\right\}$
6: Update the pseudo–envelope using the new point $(h_t^{(can)}, G(h_t^{(can)}))$
7: **until** A draw is accepted
8: Accept $h_t^{(g)} = h_t^{(can)}$ with probability

$$\alpha(h_t^{(g-1)}, h_t^{(can)}) = min\left\{1, \frac{p(h_t^{(can)}) min\left\{p(h_t^{(g-1)}), G(h_t^{(g-1)})\right\}}{p(h_t^{(g-1)}) min\left\{p(h_t^{(can)}), G(h_t^{(can)})\right\}}\right\}$$

Otherwise set $h_t^{(g)} = h_t^{(g-1)}$
9: **end for**

5.3 Particle Filtering

For implementing PMCMC, we need a particle filtering algorithm for the model. First note that for our model, the optimal proposal that minimizes the variance of the importance weights (as defined in Proposition 3.1) is given as:

$$\begin{aligned}
q(X_t|X_{t-1}, Y_t) &= p(X_t|X_{t-1}, Y_t) \\
&= Ga(G_t|\frac{1}{\lambda}, \lambda) N(J_t|\frac{\tilde{m_j}}{\tilde{s_j^2}}, \frac{1}{\tilde{s_j^2}}) N(\nu_t|\tilde{m}_\nu, \tilde{s_\nu^2}) \\
\tilde{m_j} &= \frac{\varphi}{\psi^2} + \frac{Y_t - \mu}{\nu_{t-1}} \\
\tilde{s_j^2} &= \frac{1}{\psi^2 G_t} + \frac{1}{\nu_{t-1}} \\
\tilde{m}_\nu &= \nu_{t-1} + \kappa(\theta - \nu_{t-1}) + \rho\gamma(Y_t - \mu - J_t) \\
\tilde{s_\nu^2} &= (1 - \rho^2)\gamma\nu_{t-1}
\end{aligned}$$

The optimal proposal includes standard distributions and thus can be sampled. However the incremental importance weights for optimal proposal, which is equal to the predictive likelihood, is given as:

$$\begin{aligned}
\alpha_t &= p(Y_t|X_{t-1}) = \int p(Y_t, X_t|X_{t-1}) dX_t \\
&= \int \underbrace{p(J_t)}_{\text{Variance Gamma}} \underbrace{p(Y_t|J_t, \nu_{t-1})}_{\text{Normal}} dJ_t
\end{aligned}$$

This integral cannot be evaluated analytically. Thus we will use an approximation to the predictive likelihood in the second stage weights of the particle filter. The approximation is obtained by evaluating likelihood at prior mean of conditioning variables.

Algorithm 5.2 MCMC Algorithm For The Model

Initialize $(X^{(0)}, \Theta^{(0)})$
for $g = 1$ to G **do**
  Sample $\mu^{(g)} \sim N(\frac{a_1}{A_1}, \frac{1}{A_1})$
  Sample $\kappa^{(g)} \sim N(\frac{b_1}{B_1}, \frac{1}{B_1})1_{(\kappa>0)}$
  Sample $\theta^{(g)} \sim N(\frac{c_1}{C_1}, \frac{1}{C_1})1_{(\theta>0)}$
  Sample $w^{(g)} \sim IG(d_1, D_1)$
  Sample $\phi^{(g)} \sim N(\frac{e_1}{E_1}, \frac{w}{E_1})$
  Sample $\varphi^{(g)} \sim N(\frac{p_1}{P_1}, \frac{1}{P_1})$
  Sample $\psi^{2(g)} \sim IG(q_1, Q_1)$
  Sample $\lambda^{(g)} \sim [\frac{1}{\lambda^{1/\lambda}\Gamma(1/\lambda)}]^T \lambda^{-r_0-1}[\prod_{t=1}^T G_t]^{1/\lambda} \exp\{-\frac{1}{\lambda}(R_0 + \sum_{t=1}^T G_t)\}$ using ARMS
  Sample $\nu_0^{(g)} \sim 1_{(\nu_0>0)} \frac{1}{\nu_0} \exp(-\frac{f_1}{\nu_0}) \exp(-F_1\nu_0)$ using ARMS
  for $t = 1$ to $T-1$ **do**
    Sample $\nu_t^{(g)} \sim 1_{(\nu_t>0)} \frac{1}{\nu_t} \exp\{\frac{M_1}{\nu_t} + M_2\nu_t - M_3\nu_t^2\}$ using ARMS
    Sample $J_t^{(g)} \sim N(\frac{r_1}{R_1}, \frac{1}{R_1})$
    Sample $G_t^{(g)} \sim G_t^{1/\lambda-3/2} \exp\{-G_t(\frac{1}{\lambda} + \frac{\varphi^2}{2\psi^2}) - \frac{1}{G_t}\frac{J_t^2}{2\psi^2}\}$ using ARMS
  end for
  for $t = T$ **do**
    Sample $\nu_T^{(g)} \sim N(h_1, H_1)1_{(\nu_T>0)}$
    Sample $J_T^{(g)} \sim N(\frac{r_1}{R_1}, \frac{1}{R_1})$
    Sample $G_T^{(g)} \sim G_T^{1/\lambda-3/2} \exp\{-G_T(\frac{1}{\lambda} + \frac{\varphi^2}{2\psi^2}) - \frac{1}{G_T}\frac{J_t^2}{2\psi^2}\}$ using ARMS
  end for
end for

First note that the prior means of the latent variables are given as:

$$\begin{aligned} E[\nu_t|\nu_{t-1}] &= \nu_{t-1} + \kappa(\theta - \nu_{t-1}) \\ E[J_t] &= E[E[J_t|G_t]] = E[\varphi G_t] = \varphi E[G_t] = \varphi \end{aligned}$$

Therefore likelihood evaluated at these values is given as:

$$\begin{aligned} \hat{p}(Y_t|X_{t-1}) &= N(Y_t|\mu + E[J_t] + \frac{\rho}{\gamma}(E[\nu_t|\nu_{t-1}] - \nu_{t-1} + \kappa(\theta - \nu_{t-1})), (1-\rho^2)\nu_{t-1}) \\ &= N(Y_t|\mu + \varphi, (1-\rho^2)\nu_{t-1}) \end{aligned}$$

In an auxiliary particle filtering setting, these choices leads to second stage weights given as:

$$\begin{aligned} W_t &= \frac{g(Y_t|X_t, X_{t-1})f(X_t|X_{t-1})}{\hat{p}(Y_t|X_{t-1})q(X_t|X_{t-1})} \\ &= \frac{N(Y_t|m_Y, s_Y^2)N(\nu_t|m_\nu, s_\nu^2)Ga(G_t|\frac{1}{\lambda}, \lambda)N(J_t|\frac{m_j}{s_j^2}, \frac{1}{s_j^2})}{N(Y_t|\tilde{m_Y}, \tilde{s_Y^2})N(\nu_t|\tilde{m_\nu}, \tilde{s_\nu^2})Ga(G_t|\frac{1}{\lambda}, \lambda)N(J_t|\frac{\tilde{m_j}}{\tilde{s_j^2}}, \frac{1}{\tilde{s_j^2}})} \\ &= \frac{N(Y_t|m_Y, s_Y^2)N(\nu_t|m_\nu, s_\nu^2)N(J_t|\frac{m_j}{s_j^2}, \frac{1}{s_j^2})}{N(Y_t|\tilde{m_Y}, \tilde{s_Y^2})N(\nu_t|\tilde{m_\nu}, \tilde{s_\nu^2})N(J_t|\frac{\tilde{m_j}}{\tilde{s_j^2}}, \frac{1}{\tilde{s_j^2}})} \end{aligned}$$

$$
\begin{aligned}
m_Y &= \mu + J_t + \frac{\rho}{\gamma}(\nu_t - \nu_{t-1} - \kappa(\theta - \nu_{t-1})) \\
s_Y^2 &= (1-\rho^2)\nu_{t-1} \\
\tilde{m_Y} &= \mu + \varphi \\
\tilde{s_Y^2} &= (1-\rho^2)\nu_{t-1} \\
m_j &= \frac{\varphi}{\psi^2} \\
s_j^2 &= \frac{1}{\psi^2 G_t} \\
m_\nu &= \nu_{t-1} + \kappa(\theta - \nu_{t-1}) \\
s_\nu^2 &= \gamma^2 \nu_{t-1}
\end{aligned}
$$

The full APF algorithm with these choices is given in Algorithm 5.3.

Algorithm 5.3 APF Algorithm For The Model

for $t = 1$, for each $i = 1, \ldots, N$ **do**
  Sample $X_1^i \sim q_1(x_1)$ as follows:
  - Sample $G_1^i \sim Ga(\frac{1}{\lambda}, \lambda)$
  - Sample $J_1^i \sim N(\frac{\tilde{m_j}}{\tilde{s_j^2}}, \frac{1}{\tilde{s_j^2}})$
  - Sample $\nu_1^i \sim N(\tilde{m_\nu}, \tilde{s_\nu^2})$

  Weight the particles $W_1^i \propto \frac{N(Y_1|m_Y, s_Y^2) N(\nu_1|m_\nu, s_\nu^2) N(J_1|\frac{m_j}{s_j^2}, \frac{1}{s_j^2})}{N(\nu_1|\tilde{m_\nu}, \tilde{s_\nu^2}) N(J_1|\frac{\tilde{m_j}}{\tilde{s_j^2}}, \frac{1}{\tilde{s_j^2}})}$
end for
for $t \geq 2$, for each $i = 1, \ldots, N$ **do**
  Reweight old particles $\tilde{W}_{t-1}^i \propto W_{t-1}^i \times N(Y_t|\tilde{m_Y}, \tilde{s_Y^2})$
  Resample $\left\{X_{1:t-1}^i, \tilde{W}_{t-1}^i\right\}$ to obtain N equally-weighted particles $\left\{\tilde{X}_{1:t-1}^i, \frac{1}{N}\right\}$
  Sample $X_t^i \sim q_t(x_t|\tilde{X}_{t-1}^i)$ as follows:
  - Sample $G_t^i \sim Ga(\frac{1}{\lambda}, \lambda)$
  - Sample $J_t^i \sim N(\frac{\tilde{m_j}}{\tilde{s_j^2}}, \frac{1}{\tilde{s_j^2}})$
  - Sample $\nu_t^i \sim N(\tilde{m_\nu}, \tilde{s_\nu^2})$

  Set $X_{1:t}^i \leftarrow (\tilde{X}_{1:t-1}^i, X_t^i)$
  Weight new particles $W_t^i \propto \frac{N(Y_t|m_Y, s_Y^2) N(\nu_t|m_\nu, s_\nu^2) N(J_t|\frac{m_j}{s_j^2}, \frac{1}{s_j^2})}{N(Y_t|\tilde{m_Y}, \tilde{s_Y^2}) N(\nu_t|\tilde{m_\nu}, \tilde{s_\nu^2}) N(J_t|\frac{\tilde{m_j}}{\tilde{s_j^2}}, \frac{1}{\tilde{s_j^2}})}$
end for

5.4 PMCMC

Now we can fully define the particle Gibbs algorithm for our model. For this we can use the findings of previous sections: the conditional posterior for parameters,

$p(\Theta|X_{1:T}, Y_{1:T})$, is obtained in Section 5.2, the auxiliary particle filter algorithm is defined in Section 5.3 and the steps for conditional SMC update is defined in Algorithm 4.2. Full implementation of particle Gibbs algorithm for our model is given in Algorithm 5.4.

Algorithm 5.4 Particle Gibbs Sampler for the Model

1: Initialization:

- Initialize $\Theta^{(0)}$
- Run an SMC targeting $p_{\Theta^{(0)}}(x_{1:T}|y_{1:T})$ using Algorithm 5.3
- Sample from the empirical measure $X_{1:T}^{(0)} \sim \hat{p}_{\Theta^{(0)}}(x_{1:T}|y_{1:T})$, thus implicitly select $B_{1:T}^{(0)}$

2: **for** $g = 1$ to G **do**

3: Sample $\Theta^{(g)} \sim p(\Theta|x_{1:T}^{(g-1)}, y_{1:T})$ using the following steps:

- Sample $\mu^{(g)} \sim N(\frac{a_1}{A_1}, \frac{1}{A_1})$
- Sample $\kappa^{(g)} \sim N(\frac{b_1}{B_1}, \frac{1}{B_1})1_{(\kappa>0)}$
- Sample $\theta^{(g)} \sim N(\frac{c_1}{C_1}, \frac{1}{C_1})1_{(\theta>0)}$
- Sample $w^{(g)} \sim IG(d_1, D_1)$
- Sample $\phi^{(g)} \sim N(\frac{e_1}{E_1}, \frac{w}{E_1})$
- Sample $\varphi^{(g)} \sim N(\frac{p_1}{P_1}, \frac{1}{P_1})$
- Sample $\psi^{2(g)} \sim IG(q_1, Q_1)$
- Sample $\lambda^{(g)} \sim [\frac{1}{\lambda^{1/\lambda}\Gamma(1/\lambda)}]^T \lambda^{-r_0-1} [\prod_{t=1}^T G_t]^{1/\lambda} \exp\{-\frac{1}{\lambda}(R_0 + \sum_{t=1}^T G_t)\}$ using ARMS
- Sample $\nu_0^{(g)} \sim 1_{(\nu_0>0)} \frac{1}{\nu_0} \exp(-\frac{f_1}{\nu_0}) \exp(-F_1\nu_0)$ using ARMS

4: Run a conditional SMC targeting $p_{\Theta^{(g)}}(x_{1:T}|y_{1:T})$ using Algorithm 5.3, but conditional on $X_{1:T}^{(g-1)}$ as defined in Algorithm 4.2

5: **end for**

5.5 Empirical Implementation

In this section we will implement the model introduced in previous sections using S&P500 Index data. First we will discuss the descriptive statistics of the data. Then we will estimate the model using MCMC and PMCMC methods defined previously and compare the estimation results.

We use S&P500 Index log-returns in our estimations. The S&P500 is a free-float capitalization-weighted index of the prices of 500 large-cap common stocks actively traded in the United States. The stocks included in the S&P 500 are those of large publicly held companies that trade on either of the two largest US stock market exchanges; the NYSE Euronext and the NASDAQ OMX. We use 15 years of data (from 2000 to 2015; totally 3914 observations) to estimate the model.

Figure 5.3 show the empirical density, probability plot and autocorrelations for the

returns and squared returns. Returns exhibit significant kurtosis with a slight skewness. Thus empirical distribution is highly non-Normal. At first sight this may suggest modeling the data as an i.i.d. sequence from a non-Normal distribution. However last two graphs show that the return series have linear and non-linear dependence through time. For returns, there are significant autocorrelations at the first two lags. More importantly, squared returns have a very slowly decaying autocorrelation structure which can be taken as an informal evidence of non-constant volatility.

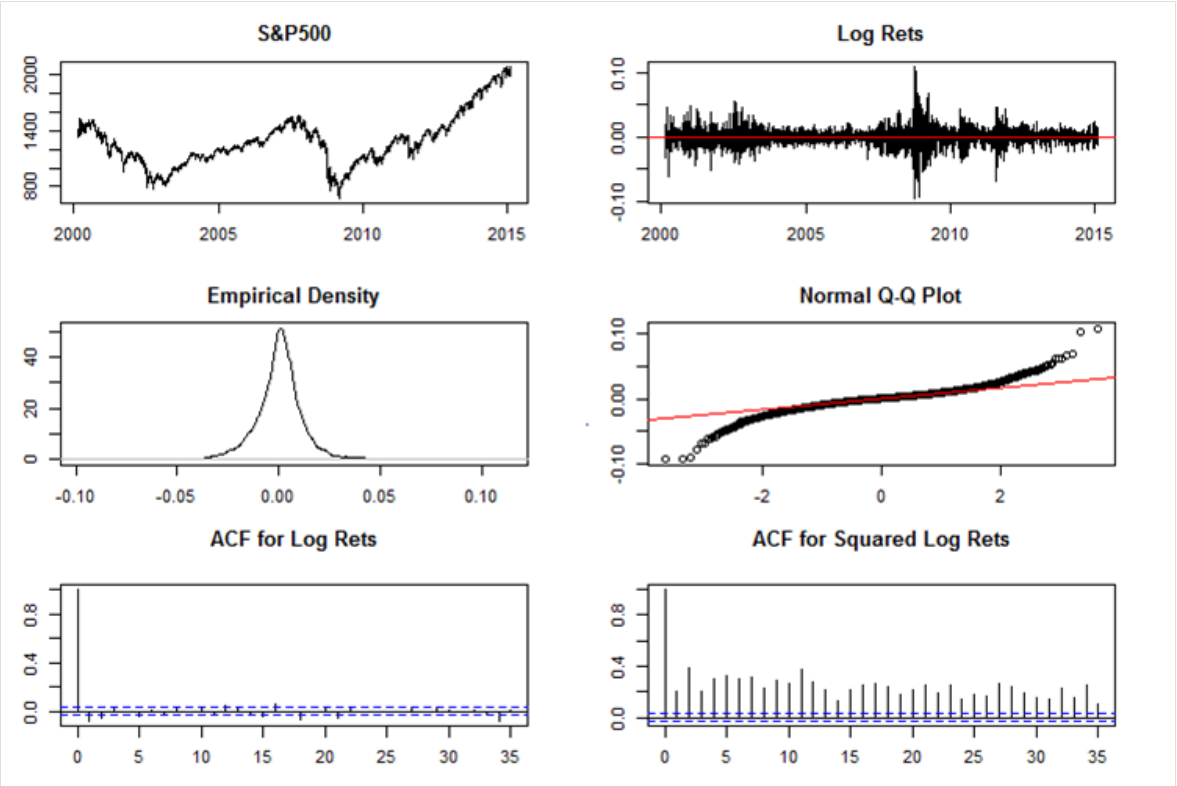

Figure 5.3: Descriptive Plots For S&P500 Index: Index level, log returns, empirical density for log returns, normal Q-Q plots for log returns, autocorrelations for log returns and squared log returns.

We first performed a unit root test for the S&P500 Index and its log returns. The augmented Dickey–Fuller test[3] results, as presented in Table 5.1 show that the index has a unit root while the log returns do not. Since the S&P500 Index have significant linear autocorrelations at first two lags, we apply an autoregressive filter to raw returns in order to remove linear dependence in conditional mean. For this, we first fit a second order autoregressive model given as:

$$Y_t = \pi_0 + \pi_1 Y_{t-1} + \pi_2 Y_{t-2} + \zeta_t \tag{5.8}$$

and then use the residuals ζ_t in our estimation. This linear filtering step is commonly used in time series modeling (e.g. [40]). We also test whether the residuals series has any ARCH effect or not using Lagrange Multiplier (LM) test[4]. The test statistics and

[3] Unit root tests are performed using R package urca [59].
[4] LM tests are performed using R package FinTS [31].

corresponding p–values with different lags up to 12 are given in Table 5.2. The results indicate that the residual return series has significant ARCH effect which should be modeled using a non–constant volatility model.

Table 5.1: Unit Root Testing for S&P500

	Test Statistics	Critical Levels		
		0.01	0.05	0.10
S&P500 Index	0.55	-2.58	-1.95	-1.62
Log Returns	-67.96	-2.58	-1.95	-1.62

Table 5.2: ARCH LM Test for Residual Returns

Lag	Test Statistics	p–value
1	146.9755	$<$ 2.2e–16
2	690.4289	$<$ 2.2e–16
3	722.7191	$<$ 2.2e–16
4	806.9449	$<$ 2.2e–16
5	958.5190	$<$ 2.2e–16
6	1020.0392	$<$ 2.2e–16
7	1056.6099	$<$ 2.2e–16
8	1057.9222	$<$ 2.2e–16
9	1068.2387	$<$ 2.2e–16
10	1079.3840	$<$ 2.2e–16
11	1144.6060	$<$ 2.2e–16
12	1156.1017	$<$ 2.2e–16

We fit our time changed Lévy model using MCMC and PMCMC methods. In both methods, we run MCMC chains for $G = 100.000$ iterations and ignore the first half, i.e. a burn-in period of $M = 50.000$. We use relatively diffusive priors as defined in previous sections to initialize the chains. The ARMS algorithm is implemented using the codes given in [58]. For the particle filtering algorithm, we use 5.000 particles and use systematic resampling if $ESS < N/2$. The estimation results are presented in Table 5.3 below and in figures 5.6 to 5.10 in the appendix.

In both methods, the parameter chains quickly converge to regions around certain values. This can be seen from figures in the appendix which show chain values after burn-in period, along with histogram of chain values. These graphs also show that autocorrelations in parameter chains has a fast decay. Exception to this is the parameter γ, for which we have a slower decay in the autocorrelations.

For the latent variables autocorrelations for the log-volatility have also slower decays. This is somewhat expected since for the volatility process, ν_{t-1} appears both in the mean and variance of ν_t and the volatility variables are highly correlated posteriori. Furthermore autocorrelations for the Gamma variate also have a slow decay.

When we compare the MCMC and PMCMC estimation results, we see that the estimates are not so distinct from each other. However the standard deviations and

Table 5.3: Parameter Estimates from MCMC and PMCMC

	MCMC			PMCMC		
	Mean	SD	MC SE	Mean	SD	MC SE
μ	0.000	0.012	0.009	0.000	0.014	0.008
κ	0.014	0.006	0.011	0.008	0.011	0.006
θ	0.763	0.212	0.253	0.642	0.150	0.138
ρ	-0.378	0.005	0.157	-0.423	0.022	0.129
γ	0.127	0.011	0.045	0.131	0.020	0.013
φ	0.009	0.005	0.012	0.007	0.004	0.017
ψ^2	0.392	0.023	0.156	0.425	0.014	0.235
λ	3.785	0.064	1.245	3.287	0.068	1.768

Table includes posterior mean and standard deviation, as well as Monte Carlo standard errors for each parameter.

Monte Carlo errors[5] are generally narrower in PMCMC. The main difference appears in the autocorrelations for the parameter γ and the latent variables, especially for the volatility process ν_t. The decay rate for the autocorrelations are faster in PMCMC when compared with MCMC. This finding is in accordance with the different updating structures used in MCMC and PMCMC algorithms. In MCMC algorithms we use a sequential approach to update each volatility one-at-a-time. On the other hand, PMCMC algorithm uses particle filtering to jointly update the latent variables and thus uses a block sampling approach. As discussed in previous chapters, blocking highly correlated components and updating them jointly may significantly improve the performance of MCMC algorithms (see p. 12 of [29]). [23] suggest that we should block as much as possible whenever we have a sampling method to jointly update the components.

From theory of Markov chains, we expect our MCMC and PMCMC chains to eventually converge to the stationary distribution, which is also our target distribution. However, there is no guarantee that we can achieve this convergence for a finite number of iterations. As discussed previously, quick convergence of chains to regions around certain values and decaying autocorrelation plots may constitute visual evidences for the convergence to a stationary distribution. Besides we also apply Geweke [25] convergence diagnostic to statistically test the convergence to target distribution. In Geweke convergence diagnostic, for each variable, the MCMC chain is divided into two parts containing the first 10% and the last 50% of the iterates. If the whole chain is stationary, the means of the values early and late in the sequence should be similar. Geweke's approach involves calculation of the sample mean and asymptotic variance in each window, the latter being determined by spectral density estimation. His convergence diagnostic Z is the difference between these two means divided by the asymptotic standard error of their difference. The values of Z statistic which fall in the extreme tails of a standard normal distribution suggest that the chain was not fully converged early on. The Geweke statistics[6] calculated for each parameter chain (after burn–in period)

[5] Monte Carlo standard errors are calculated using R package mcmc [26].

[6] Tests are performed using R package coda [62].

are given in Table 5.4. In principle, convergence diagnostics cannot guarantee that the chains have converged. Nonetheless these visual and statistical tests show no sign of non–convergence.

Table 5.4: Z statistics from Geweke convergence test

	MCMC	PMCMC
μ	0.23	0.50
κ	0.37	0.19
θ	0.29	-0.85
ρ	-0.79	-0.30
γ	1.32	0.94
φ	0.13	-0.36
ψ^2	-0.75	-0.38
λ	0.89	0.23

One more difference between the two methods is the computational time required to simulate a Markov chain with the same length. In our estimation study, we used the same chain lengths for MCMC and PMCMC. However, in PMCMC we run a particle filter at each iteration, whereas in standard MCMC the sampling scheme is much faster to sample from. As a result of this PMCMC method required 4.3 times more computation time than the standard MCMC.

To compare the fitted model with the actual data, we also performed a simulation study. The actual data we used spans a 15–year period. We generate 1000 different samples, each having a return series spanning 15 years, using our model with parameter values that are obtained using PMCMC. Figure 5.4 includes simulated and actual prices and returns. The left figure includes 1000 simulated paths of stock price (on a logarithmic scale) along with the actual price series used in estimation. Furthermore Table 5.5 includes the various moments calculated using the actual and simulated data. The realized path of stock prices are within the bounds of simulated paths and the distribution of returns are close to each other. However the actual returns exhibit higher levels of skewness and kurtosis. These findings show that the model is a reasonable approximation to the actual data used. On the other hand there also exists some further room to improve goodness of fit using a different model to capture the excess skewness and kurtosis observed in the actual data.

Table 5.5: Moments for Actual vs Simulated Returns

	Actual Returns	Simulated Returns		
		5%	Mean	95%
Mean	0.00	-2.00	0.01	3.47
Standard Deviation	1.26	0.90	1.03	1.20
Skewness	-0.32	-0.21	0.03	0.25
Kurtosis	10.64	5.13	6.15	7.40
First Order Autocorrelation	0.00	-0.03	0.00	0.03

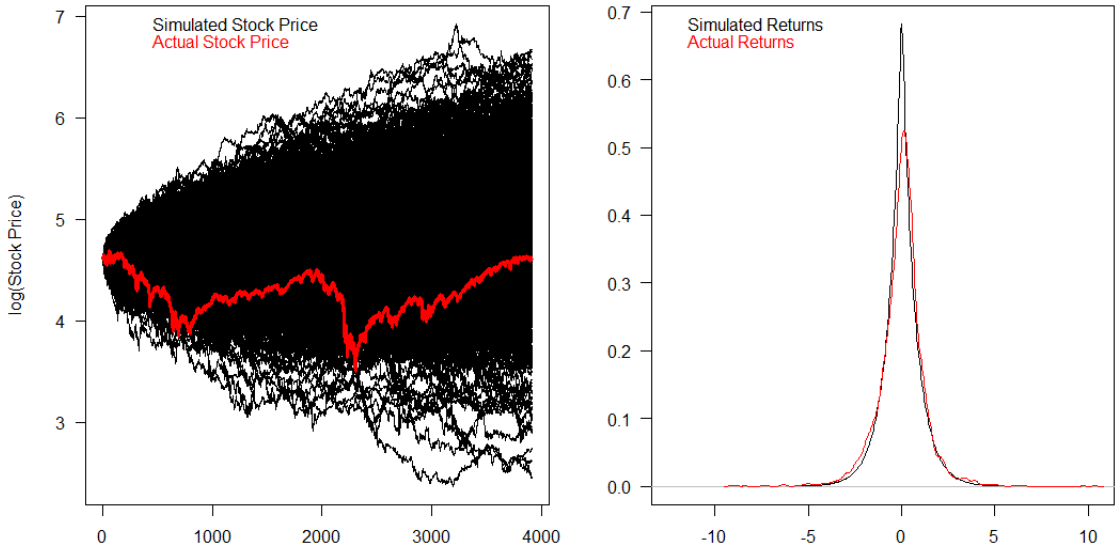

Figure 5.4: The left figure includes simulated paths of stock price from the model, as well as the actual path used in estimation. Both are shown in logarithmic scale. The right figure shows the empirical density of simulated and actual returns.

The estimation results show that PMCMC is a flexible estimation method to infer the parameters and latent variables of a complex model such as our Lévy based model. Once we are able to estimate the model, we can use the findings in different applications. One example is forecasting. Assume that we have data for time $t = 1, 2, \ldots, T$ and are interested in the next day return, Y_{T+1}. Then we can estimate the model using the available data up until time T. Since the density of Y_{T+1} is not available in closed form, we can use simulations to make inferences on it. Such an exercise is shown in Figure 5.5. The figure shows the last 20 values for S&P500 index that are used in estimation step. After estimating the model, we simulate 1000 samples for Y_{T+1} using the model with estimated parameters and latent variables, and then incorporate these returns in the second order autoregressive model given in 5.8 to simulate S&P500 index for time period $T + 1$. The simulated values for S&P500 index are shown as blue dots and their empirical density is plotted in red in the figure. The latest observation of S&P500 index that is used in estimation has a value of 2079.65. The forecasted values have a mean of 2080.99 with a standard deviation of 22.52. The 90% confidence interval for the point forecast is $(2045.47, 2119.45)$. The daily value–at–risk for a 99% confidence level is estimated as -2.46%.

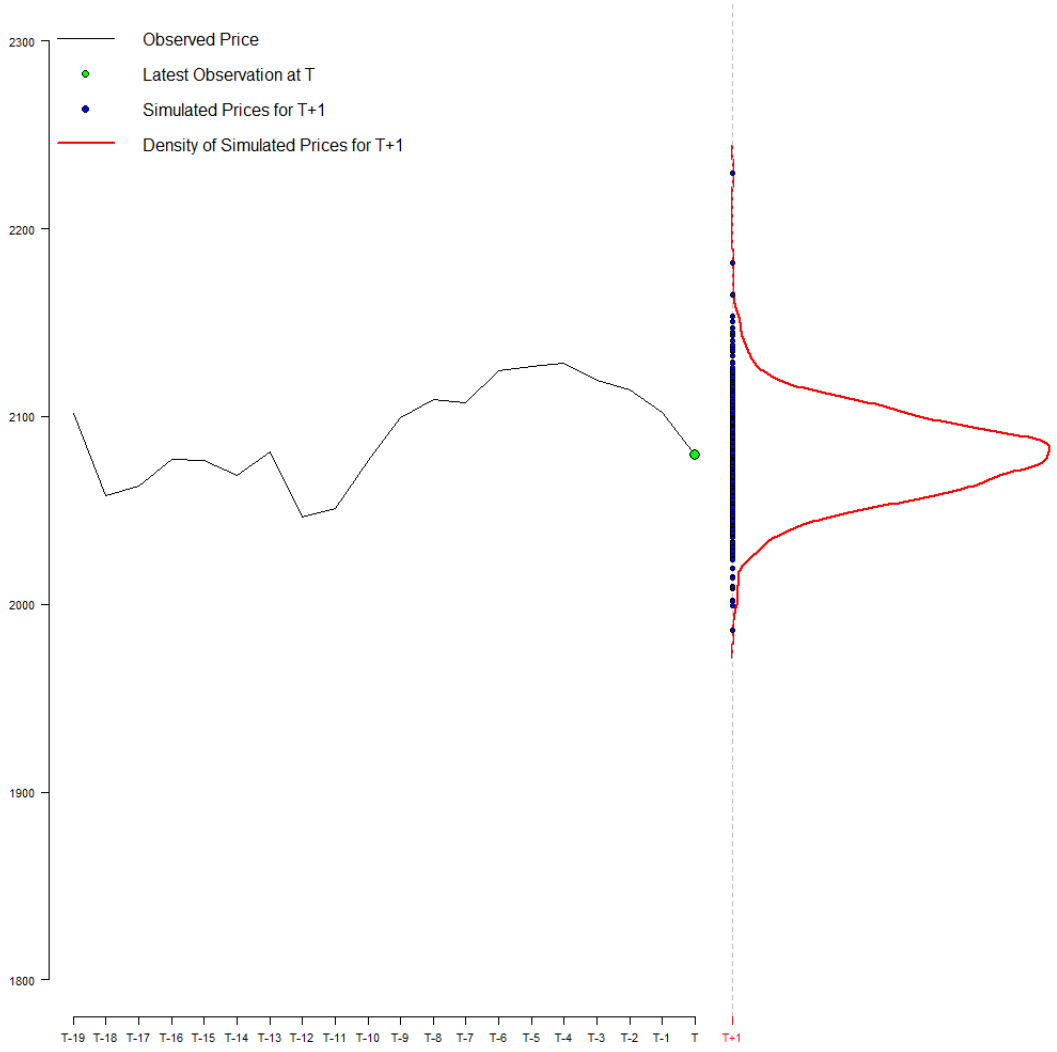

Figure 5.5: The figure includes latest 20 prices for S&P500 index, one day ahead forecasts using the Lévy model and their corresponding density.

5.6 Appendix

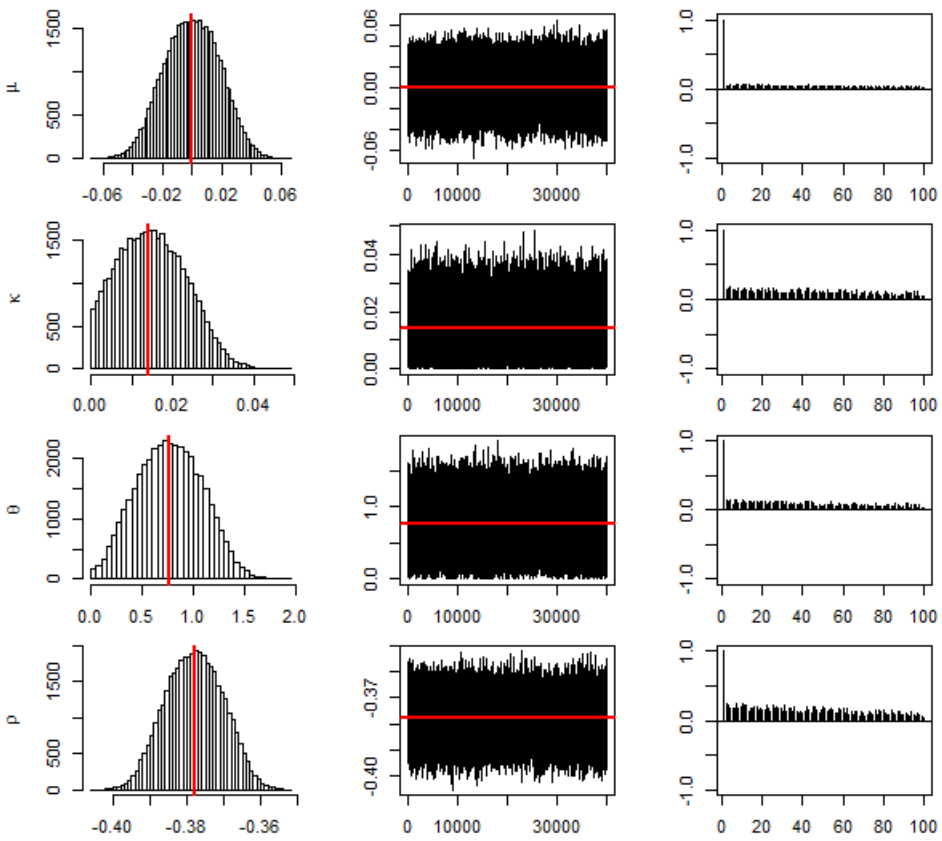

Figure 5.6: MCMC output for the model.

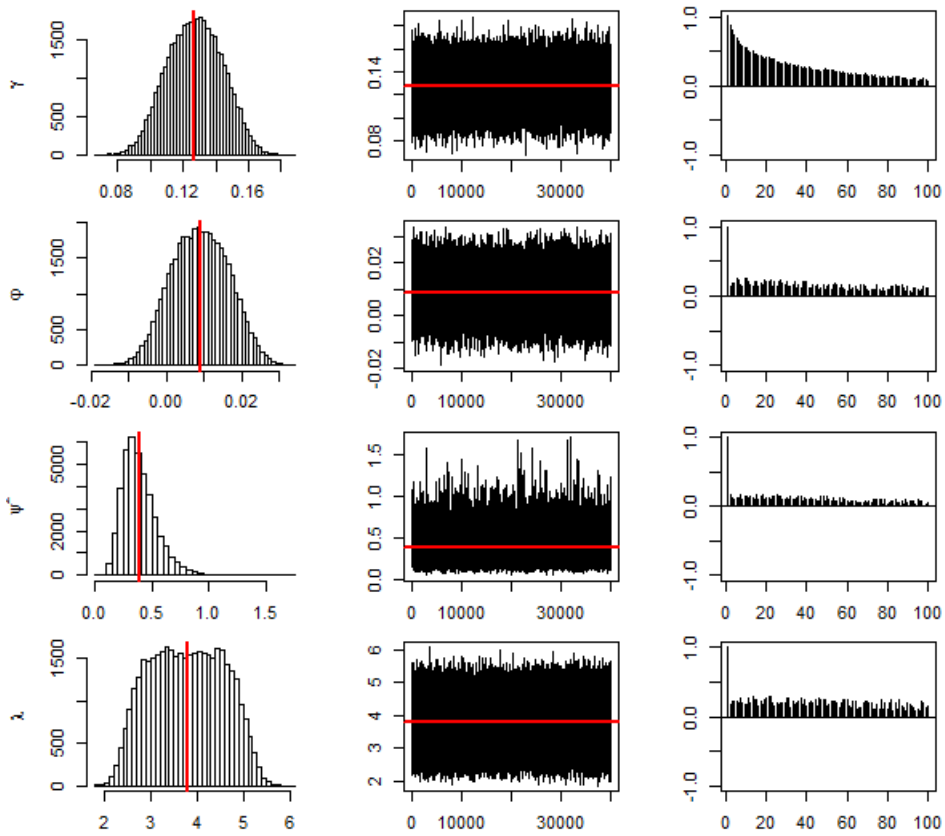

Figure 5.7: MCMC output for the model.

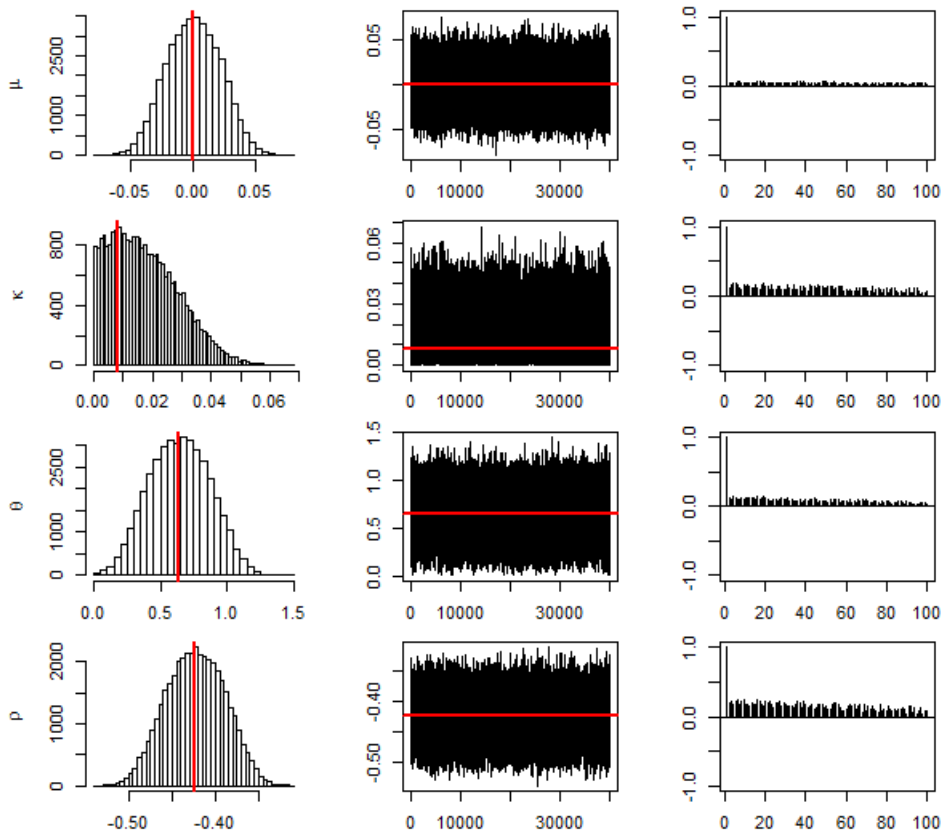

Figure 5.8: PMCMC output for the model.

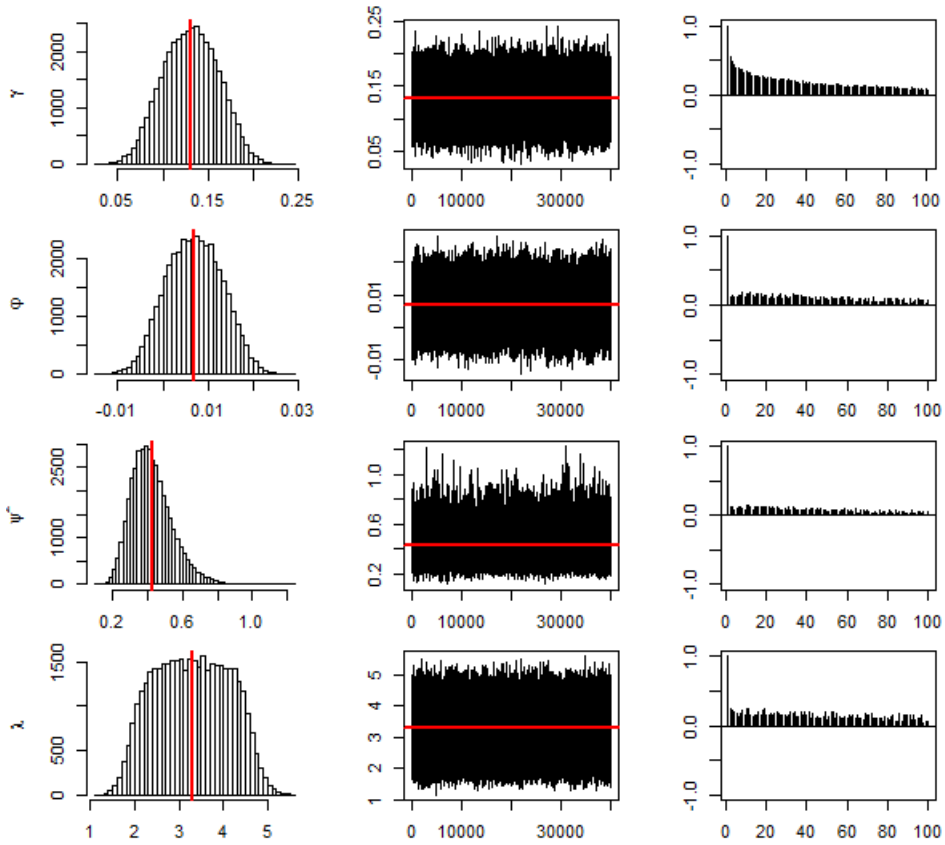

Figure 5.9: PMCMC output for the model.

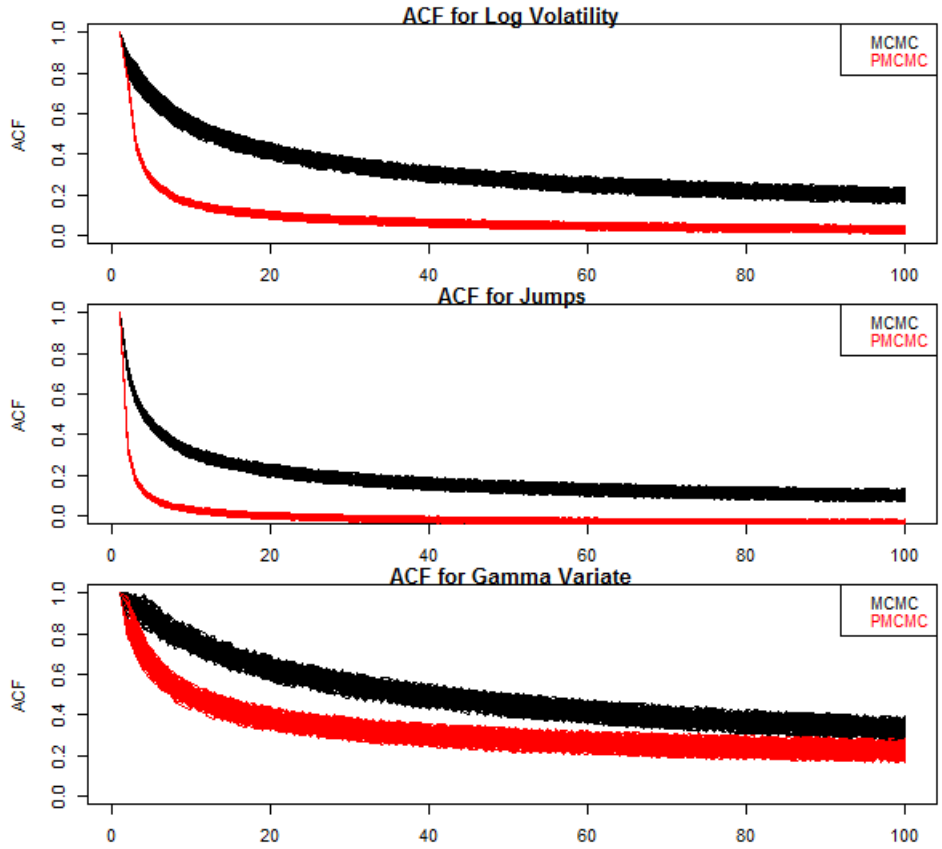

Figure 5.10: ACF plots for latent variables from MCMC and PMCMC methods.

CHAPTER 6

CONCLUSION

Finding suitable stochastic process is the starting point and the most fundamental step of any financial modelling exercise. The inefficiency of Gaussian models in capturing observed behavior of stock prices yields to the emergence of a vast literature on more advanced models that include stochastic volatility and jumps. This ever increasing complexity of models rules out the use of traditional estimation methods for inferring model parameters and necessitates more advanced estimation techniques. Thus the main objective of this research was to develop efficient algorithms for estimating a stock price model that is flexible enough to capture the different characteristics of observed data.

Therefore we resort to simulation based Bayesian estimation techniques. We begin with MCMC approach, discuss the theoretical underpinnings of this approach and develop various algorithms and implement them for a basic stochastic volatility model. MCMC approaches offers a flexible method for estimating unknown parameters and latent variables of a model by approximating the posterior distribution with Monte Carlo samples. However, our analysis confirm that the efficiency of an MCMC algorithm critically depends on the selection of good proposal distributions for latent variables and jointly updating highly correlated variables may significantly improve the MCMC algorithm.

Then we turn our attention to on–line estimation and discuss particle filtering methods in which the model parameters are assumed to be known and the latent states are dynamically inferred as we sequentially observe new data. Using auxiliary particle filtering, we test various algorithms for the stochastic volatility model and compare them in a simulation study.

Next we discuss the recently proposed particle MCMC methods that uses particle filters to build efficient high–dimensional proposal distributions to be used within an MCMC setting. With this enhancement, particle MCMC methods circumvent the main challenge of traditional MCMC methods and offer a powerful estimation technique for many complex models.

We develop MCMC and particle MCMC algorithms for a stock price model with a time changed Lévy component. We assume that the stock price process includes Heston–type stochastic volatility plus variance–gamma jumps in returns. Variance–Gamma

process is an infinite activity finite variation Lévy process obtained by subordinating an arithmetic Brownian motion with a Gamma process. The model is quite flexible in its nature and can capture most of the observed characteristics of stock prices. We developed MCMC and particle MCMC algorithms for the model and compare them in an empirical study using S&P500 Index with 15 years of data. The results indicate that the particle MCMC algorithm is a more efficient alternative to standard MCMC and typically gives smaller standard errors and lower autocorrelations.

The literature on Bayesian estimation of financial models includes various research papers that implement MCMC algorithms for models with stochastic volatility and jumps. However particle MCMC approaches are relatively new and thus research on these methods is not voluminous. We contributed to the existing literature by developing efficient particle MCMC algorithms for the first time for a complex model with stochastic volatility and Lévy based jumps. We see our new contribution as an example for the application of particle MCMC methods in different real life problems and expect to see this trend to continue in the future.

Simulation based inference is not an exact science and there always exists room for further improvement. For instance, although we obtain better estimation results in our particle MCMC algorithm than the standard MCMC, theoretically, it is still possible to obtain a more efficient algorithm that may yield much faster decay in autocorrelations and thus smaller standard errors. Therefore this research may serve as a starting step in searching for efficient estimation methods for complex financial models and as a stimulus for further research in this area.

REFERENCES

[1] T. G. Andersen, R. A. Davis, and J. Kreiβ, *Handbook of Financial Time Series*, Springer, 2009.

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

[3] L. Bachelier, Théorie de la spéculation, Annales Scientifiques de l'École Normale Supérieure, 3(17), pp. 21–86, 1900.

[4] C. A. Ball and A. Roma, Stochastic volatility option pricing, The Journal of Financial and Quantitative Analysis, 29(4), pp. 589–607, 1994.

[5] O. E. Barndorff-Nielsen, Processes of normal inverse Gaussian type, Finance and Stochastics, 2(1), pp. 41–68, 1998.

[6] D. Bates, Jumps and stochastic volatility: Exchange rate processes implicit in deutsche mark options, Review of Financial Studies, 9(1), pp. 69–107, 1996.

[7] D. Bates, Post–87 crash fears in s&p futures options, Journal of Econometrics, 94(1-2), pp. 181–238, 2000.

[8] F. Black and M. Scholes, The pricing of options and corporate liabilities, Journal of Political Economy, 81(3), pp. 637–659, 1973.

[9] P. Carr, H. Geman, D. Madan, and M. Yor, The fine structure of asset returns: an empirical investigation, Journal of Business, 75(2), pp. 305–332, 2002.

[10] P. Carr and L. Wu, Finite moment log stable process and option pricing, Journal of Finance, 58(2), pp. 753–777, 2003.

[11] P. Congdon, *Bayesian Statistical Modelling*, John Wiley & Sons, Ltd., 2001.

[12] R. Cont, Empirical properties of asset returns: stylized facts and statistical issues, Quantitative Finance, 1, pp. 223–236, 2001.

[13] R. Cont and P. Tankov, *Financial Modelling with Jump Processes*, Chapman & Hall/CRC, 2004.

[14] C.-O. Douc, R. and E. Moulines, Comparison of resampling schemes for particle filtering., in *4th International Symposium on Image and Signal Processing and Analysis (ISPA)*, 2005.

[15] A. Doucet, N. De Freitas, and N. Gordon, *Sequential Monte Carlo methods in practice*, 2001.

[16] A. Doucet, S. Godsill, and C. Andrieu, On sequential Monte Carlo sampling methods for bayesian filtering, Statistics and Computing, 10, pp. 197–208, 2000.

[17] A. Doucet, N. J. Gordon, and V. Krishnamurthy, Particle filters for state estimation of jump Markov linear systems, IEEE Transactions on Signal Processing, 49(3), pp. 613–624, 2001.

[18] A. Doucet and A. M. Johansen, *The Oxford Handbook of Nonlinear Filtering*, chapter A tutorial on particle filtering and smoothing: Fiteen years later, Oxford University Press, 2010.

[19] D. Duffie, J. Pan, and K. Singleton, Transform analysis and asset pricing for affine jump–diffusions, Econometrica, 68(6), pp. 1343–1376, 2000.

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

[21] T. Flury and N. Shephard, Bayesian inference based only on simulated likelihood: Particle filter analysis of dynamic economic models, Econometric Theory, 27, pp. 933–956, 2011.

[22] S. Fruhwirth-Schnatter, Data augmentation and dynamic linear models, Journal of Time Series Analysis, 15, p. 183–202, 1994.

[23] D. Gamerman and H. F. Lopes, *Markov Chain Monte Carlo: Stochastic Simulation for Bayesian Inference*, Chapman & Hall/CRC, second edition, 2006.

[24] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian Data Analysis*, Chapman & Hall, 2003.

[25] J. Geweke, *Bayesian Statistics 4*, chapter Evaluating the Accuracy of Sampling Based Approaches to the Calculation of Posterior Moments, pp. 169–193, Oxford University Press, 1992.

[26] C. J. Geyer and L. T. Johnson, *MCMC: Markov Chain Monte Carlo*, 2015.

[27] W. Gilks, N. Best, and K. Tan, Adaptive rejection Metropolis sampling within Gibbs sampling, Applied Statistics, 44, pp. 455–472, 1995.

[28] W. Gilks and P. Wild, Adaptive rejection sampling for Gibbs sampling, Applied Statistics, 41, pp. 337–348, 1992.

[29] W. R. Gilks, S. Richardson, and D. J. Spiegelhalter, *Markov Chain Monte Carlo in Practice*, Chapman & Hall, 1996.

[30] J. Gill, *Bayesian Methods: A Social and Behavioral Sciences Approach*, Chapman & Hall/CRC, 2008.

[31] S. Graves, *FinTS: Companion to Tsay (2005) Analysis of Financial Time Series*, 2014.

[32] P. S. Hagan, D. Kumar, A. S. Lesniewski, and D. E. Woodward, Managing smile risk, Wilmott Magazine, Sep, pp. 84–108, September 2002.

[33] J. Hallgren, Calibration of stochastic volatility models using particle Markov chain Monte Carlo methods, in *Bachelier Finance Society 7th World Congress*, 2012.

[34] J. M. Hammersley and M. S. Clifford, Markov fields on finite graphs and lattices, 1970, unpublished.

[35] W. K. Hastings, Monte carlo sampling methods using markov chains and their application, Biometrika, 57, pp. 97–109, 1970.

[36] S. Heston, A closed–form solution for options with stochastic volatility with applications to bond and currency options, Review of Financial Studies, 6, pp. 327–343, 1993.

[37] S. Heston, A simple new formula for options with stochastic volatility, 1997, working Paper.

[38] J. Hull and A. White, The pricing of options on assets with stochastic volatility, Journal of Finance, 42(2), pp. 281–300, 1987.

[39] F. L. J. Dahlin and T. Schon, editors, *Second Order Particle MCMC for Bayesian Parameter Inference*, Proceedings of the 19th IFAC World Congress, 2014.

[40] E. Jacquier, N. Polson, and P. Rossi, Bayesian analysis of stochastic volatility models, Journal of Business and Economic Statistics, 12(4), p. 371–417, 1994.

[41] E. Jacquier, N. Polson, and P. Rossi, Bayesian analysis of stochastic volatility models with fat-tails and correlated errors, Journal of Econometrics, 122(1), p. 185–212, 2004.

[42] M. Johannes and N. Polson, MCMC methods for continuous-time financial econometrics, 2003, working Paper.

[43] A. Johansen and A. Doucet, A note on auxiliary particle filters, Statistics and Probability Letters, 78(12), pp. 1498–1504, 2008.

[44] N. Kantas, A. Doucet, S. Singh, and J. Maciejowski, An overview of sequential monte carlo methods for parameter estimation in general state-space models, in *15th IFAC Symposium on System Identification, SYSID 2009, 6-8 July 2009*, 2009.

[45] S. Kim, N. Shephard, and S. Chib, Stochastic volatility: Likelihood inference and comparison with ARCH models, Review of Economic Studies, 65, pp. 361 – 393, 1998.

[46] G. Kitagawa, Monte-carlo filter and smoother for non-Gaussian nonlinear state space models, Journal of Computational and Graphical Statistics, 1, pp. 1–25, 1996.

[47] S. Kou, A jump diffusion model for option pricing, Management Science, 48(8), pp. 1086–1101, 2002.

[48] D. Lamberton and B. Lapeyre, *Introduction to stochastic calculus applied to finance*, Chapman & Hall, 1996.

[49] H. Li, M. T. Wells, and C. L. Yu, A bayesian analysis of return dynamics with Lévy jumps, The Review of Financial Studies, 21(5), pp. 2345–2378, 2008.

[50] J. Liu and R. Chen, Sequential Monte-Carlo methods for dynamic systems, Journal of the American Statistical Association, 93, p. 1032–1044, 1998.

[51] J. S. Liu, *Monte Carlo Strategies in Scientific Computing*, Springer-Verlag, 2001.

[52] D. Madan, P. Carr, and E. C. Chang, The variance gamma process and option pricing, European Finance Review, 2, pp. 79–105, 1998.

[53] R. Merton, The theory of rational option pricing, Bell Journal of Economics and Management Science, 4(1), pp. 141–183, 1973.

[54] R. Merton, Option pricing when the underlying stock returns are discontinuous, Journal of Financial Economics, 3(1-2), pp. 125–144, 1976.

[55] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, Equations of state calculations by fast computing machines, Journal of Chemical Physics, 21, pp. 1087–1092, 1953.

[56] D. B. Nelson, ARCH models as diffusion approximations, Journal of Econometrics, 45(1-2), pp. 7–38, 1990.

[57] G. Petris and L. T. original C code for ARMS by Wally Gilks., *HI: Simulation from distributions supported by nested hyperplanes*, 2013.

[58] G. Petris and L. Tardella, *HI: Simulation from distributions supported by nested hyperplanes*, 2013, r package version 0.4.

[59] B. Pfaff, *Analysis of Integrated and Cointegrated Time Series with R*, Springer, 2008.

[60] M. Pitt and N. Shephard, Filtering via simulation: Auxiliary particle filter, Journal of the American Statistical Association, 94(446), pp. 590–599, 1999.

[61] M. K. Pitt and N. Shephard, *Sequential Monte Carlo Methods in Practice*, chapter Auxiliary variable based particle Filters, pp. 271–293, Springer-Verlag, 2001.

[62] M. Plummer, N. Best, K. Cowles, and K. Vines, CODA: Convergence diagnosis and output analysis for MCMC, R News, 2006.

[63] C. Ritter and M. A. Tanner, Facilitating the Gibbs sampler: The Gibbs stopper and the griddy-Gibbs sampler, Journal of the American Statistical Association, 87(419), pp. 861–868, 1992.

[64] C. Robert, *The Bayesian Choice: From Decision-Theoretic Foundations to Computational Implementation*, Springer, 2001.

[65] C. Robert and G. Casella, *Monte Carlo Statistical Methods*, Springer, 2004.

[66] P. Samuelson, Rational theory of warrant pricing, Industrial Management Review, 6, pp. 13–31, 1965.

[67] W. Schoutens, *Lévy Processes in Finance*, John Wiley & Sons, Ltd., 2003.

[68] L. O. Scott, Option pricing when the variance changes randomly: Theory, estimation, and an application, The Journal of Financial and Quantitative Analysis, 22(4), pp. 419–438, 1987.

[69] M. Sewell, Characterization of financial time series, `http://finance.martinsewell.com/stylized-facts/characterization.pdf`, 2008.

[70] J. Smith and A. A. Santos, Second-order filter distribution approximations for financial time series with extreme outliers, Journal of Business & Economic Statistics, 24, pp. 329–337, 2006.

[71] E. M. Stein and J. C. Stein, Stock price distributions with stochastic volatility: An analytic approach, The Review of Financial Studies, 4(4), pp. 727–752, 1991.

[72] S. J. Taylor, *Time Series Analysis: Theory and Practice 1*, chapter Financial Returns Modelled by the Product of Two Stochastic Processes: A Study of Daily Sugar Prices, 1961–79, North-Holland Publishing Company, 1982.

[73] S. J. Taylor, *Modelling Financial Time Series*, John Wiley, 1986.

[74] L. Tierney, Markov chains for exploring posterior distributions (with discussion), The Annals of Statistics, 22(4), pp. 1701–1762, 1994.

[75] M. West and J. Harrison, *Bayesian Forecasting and Dynamic Models*, Springer, 1997.

[76] J. B. Wiggins, Option values under stochastic volatility: Theory and empirical estimates, Journal of Financial Economics, 19(2), pp. 351–372, 1987.

[77] A. Yuksel, *Markov Chain Monte Carlo and Particle Filtering Methods for Stochastic Volatility Models*, Master's thesis, Department of Statistics, Warick University, 2010.

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Yüksel, Ayhan
Nationality: Turkish
Date and Place of Birth: 1978, Ankara

EDUCATION

Degree	Institution	Year
Ph.D.	Financial Mathematics, METU	2015
M.S.	Statistics, Warwick University	2010
M.S.	Financial Mathematics, METU	2007
B.S.	Business Administration, Bilkent University	2000

PROFESSIONAL CERTIFICATES

Certificate		Year
FRM	Financial Risk Manager	2007
PRM	Professional Risk Managers	2008
CFA	Chartered Financial Analyst	2010
CMAL	Capital Market Activities Licenses	2011

PROFESSIONAL EXPERIENCE

Year	Place	Title
2015 - To Date	Finans Invest	Senior Vice President
2010 - 2015	Finans Asset Management	Vice President
2000 - 2010	BRSA	Banking Expert
2000	Vakıfbank	Assistant Internal Auditor

RESEARCH AND PUBLICATIONS

- Bayesian Inference for Stochastic Volatility Models: An Empirical Implementation, Financial Engineering Conference, Oct 2011, Izmir University of Economics.
- Markov Chain Monte Carlo and Particle Filtering Methods for Stochastic Volatility Models, MSc Thesis in Department of Statistics, Warwick University, Sep 2010.
- Credit Risk Modelling with Stochastic Volatility, Jumps and Stochastic Interest Rates, MSc Thesis in Department of Financial Mathematics, METU, Dec 2007. Published as a book by Lambert Academic Publishing AG in Aug 2010.
- A Macro-econometric Credit Risk Model for Stress Testing Credit Portfolio, paper presented at 13th Annual Conference of Multinational Finance Society, University of Edinburgh, UK, in June 2006.
- Credit Risk Modelling in Banking, Thesis, BRSA, Dec 2005.
- Potential Impacts of Basel-II on SME Loans, Working Paper, BRSA, Aug 2005.
- Basel-II Quantitative Impact Study Assessment Report, Research Report, BRSA, Dec 2004.