Option Pricing For Fractal Activity Time
Geometric Brownian Motion (FATGBM)

Priya Dev

December 2011

A thesis submitted for the degree of Doctor of Philosophy
of the Australian National University
In memory of Professor Chris Heyde (20 April 1939 - 6 March 2008)
Declaration

The work in this thesis is my own except where otherwise stated.

Priya Dev
Acknowledgements

I would like to thank my first mentors, Professor Ross Maller and the late Professor Chris Heyde, for leading me down the academic road of discovery. Without the both of you I never would have stumbled upon this opportunity and all the opportunities that will stem from this point forward. I am also most grateful for meeting Dr Allan Sly, my very good friend and mathematical mentor. Allan, your insight and wisdom has guided me through the thickest of fogs. I would also like to express my deepest gratitude to my mentor and friend, Professor Michael Martin, whose tireless support and encouragement has seen me through to the end of this long and challenging path. Lastly, all my love and appreciation to my parents, Bhu and Ratinder Dev, whose unconditional love and support have lead me to keep striving for great achievements.
Abstract

This thesis examines option pricing for a Long Range Dependent (LRD) stochastic process with student marginal distributions called Fractal Activity Time Geometric Brownian Motion (FATGBM), introduced in Heyde (1999). We address four separate problems involving the pricing of options under FATGBM and other LRD stochastic processes.

Following an introduction into the mechanics of derivative pricing, the thesis begins by addressing the problem of derivative pricing under FATGBM. We first develop the properties of FATGBM and show that the market is arbitrage-free but incomplete under this model. We then prove that there is no replicating strategy for this model except under special circumstances. We show that those special circumstances lead to the hedging of a Timer Option where interest rates are zero and we conclude by discussing the issue of completing the market by calibrating FATGBM to liquid risky assets such as European Options, as discussed in Carr et al. (2001).

We then describe how to price path dependent options under FATGBM. We first propose a non-recombining tree that is used to then construct a recombining tree to price path dependent options. Further, we prove that our discrete time
model converges to the continuous time one, resulting in a discrete approximation scheme for path dependent options. We then prove that the discrete approximation scheme results in an upper bound for the price of an American put.

The next chapter addresses the problem of sampling from the distribution of FATGBM conditional on price history. Given that FATGBM is a LRD process, it is imperative to be able to simulate future price paths given a price path history. We propose a Markov Chain Monte Carlo (MCMC) approach to develop two algorithms for two different LRD processes, one FATGBM and one similar to FATGBM that we call FATGBM 2. We prove that the algorithms result in a Markov chain with a stationary distribution identical to the conditional distribution from which we wish to sample. We then discuss the implementation of both algorithms and compare the mixing times and features of the resultant conditional distribution.

The final chapter combines the themes and results of the preceding chapters by using the MCMC algorithm in conjunction with the recombining tree developed in Chapter 2. The result is an analysis of the effect of long range dependence on option prices, the most compelling finding being that LRD has more of an impact on the option price than the impact of heavy tails alone, a phenomenon that has thus far been overlooked by the literature on option pricing. We conclude with an analysis of the implied volatility surface arising from FATGBM and discuss the implications of our research in the context of the existing literature.
Contents

Declaration iii

Acknowledgements v

Abstract vii

Chapter 1. Financial Mathematics 1
1. Prelude 1
2. Financial Markets 2
3. Introduction 5
4. Arbitrage and Models 7
5. Mathematics of Derivative Pricing – The Black Art 10
6. Long Memory Processes and Option Pricing 23

Chapter 2. Fractal Activity Time Geometric Brownian Motion – FATGBM 25
1. Introduction 25
2. Long Range Dependence and Self-Similarity 25
3. Fractal Activity Time Geometric Brownian Motion 30
4. No Arbitrage and FATGBM 39
5. Hedging and FATGBM 44
6. Conclusion 50
<table>
<thead>
<tr>
<th>Chapter 3.</th>
<th>Discrete Approximation Schemes for FATGBM</th>
<th>51</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Introduction</td>
<td>51</td>
</tr>
<tr>
<td>2.</td>
<td>FATGBM</td>
<td>53</td>
</tr>
<tr>
<td>3.</td>
<td>The Discrete Approximation Scheme</td>
<td>53</td>
</tr>
<tr>
<td>4.</td>
<td>The European Option</td>
<td>59</td>
</tr>
<tr>
<td>5.</td>
<td>The American Put</td>
<td>61</td>
</tr>
<tr>
<td>6.</td>
<td>Summary</td>
<td>62</td>
</tr>
<tr>
<td>7.</td>
<td>Discrete Approximation Scheme 2</td>
<td>63</td>
</tr>
<tr>
<td>8.</td>
<td>The American Option</td>
<td>67</td>
</tr>
<tr>
<td>9.</td>
<td>Conclusion</td>
<td>70</td>
</tr>
<tr>
<td>10.</td>
<td>Appendix</td>
<td>71</td>
</tr>
</tbody>
</table>

Chapter 4. A Markov Chain Monte Carlo Approach for Simulating
FATGBM and Related Processes | 81
1. Introduction | 81
2. Method | 82
3. Metropolis-Hastings Algorithm | 87
4. MCMC for FATGBM | 90
5. Implementation of Algorithm | 100
6. Diagnostics | 101
7. Conclusion | 113

Chapter 5. Long Range Dependence and Option Pricing | 119
1. Introduction | 119
# CONTENTS

2. The Complete Pricing Algorithm 121

3. Impact of Conditioning 125

4. Volatility Smiles 142

5. Conclusion 149

Chapter 6. Conclusion 151

Appendix A. Appendix A: C Code 155

Appendix B. Appendix B: Supplementary R Code 177

Appendix. Bibliography 179
CHAPTER 1

Financial Mathematics

1. Prelude

This is a thesis in mathematical finance that uses the application of probability theory, statistics and numerical methods to construct and implement models. The contents of this thesis has been directed towards individuals with expertise in probability and statistics and a rudimentary knowledge of financial markets. It attempts to be entirely self contained in that it includes a section on the introduction to the theory of mathematical finance. Such theory is important in any investigation of derivative pricing models.

“It's not a question of enough, pal. It's a zero sum game, somebody wins, somebody loses. Money itself isn't lost or made, it's simply transferred from one perception to another.” Gordon Gekko, Wall Street, 1987 film.
2. Financial Markets

Before we discuss what is contained in this thesis, let us provide a brief overview of financial markets and why financial mathematics is a useful tool to industry.

A Financial Market is the result of buyers and sellers coming together to trade in financial products for the purpose of lending, borrowing and limiting their downside risk. This activity is often conducted via institutions that facilitate the exchange, like stock exchanges, but trading can occur over the counter (OTC) as well. There are many types of financial markets such as equity markets, money markets, commodity markets and derivatives markets.

A derivative is a financial instrument the value of which depends on the value of an underlying, such as a stock, commodity, foreign exchange, or even another derivative. Derivatives include Forwards, Futures and Options contracts.

The prices of the financial products traded at these exchanges are driven by supply and demand, which is determined by the overall, “market” sentiment of the buyers and sellers. This sentiment is derived from many factors including company information, political and economic factors at a particular time. Such unpredictable variables result in future market sentiment and hence market prices being highly chaotic and “random”. Some financial products such as derivative products may be subject to arbitrage opportunities as well. These are trading opportunities which provide the investor with the possibility for positive profit,
with no downside risk. This has a substantial effect on the determination of market prices and forms the basis for how derivative products are priced.

Each financial market consists of participants called “Traders” whose buy-sell strategies or “trades” have been constructed with various goals in mind. Traders of derivatives products are generally classified as either Hedgers, Speculators, or Arbitrageurs. A Hedger will trade for the purpose of limiting losses on their existing portfolios, a Speculator bets on the future direction of the market and an Arbitrageur’s sole purpose is to make something from investing nothing, with absolutely no risk. Other types of traders may not fit into either category as they work on devising trading strategies which make them the highest possible profit without betting on the direction of the market.

Depending on the type of Trader, a market model may be used. The type of model will depend on the type of Trader. All of these models will be based on a set of assumptions. In building a market model for the purpose of pricing derivatives, the following is often assumed. Even though it is possible to develop theories around these assumptions, it is the subject of ongoing research.

- Traders in the market are assumed to be rational and sophisticated; in other words, the market is assumed to be competitive
- No arbitrage
- No risk of default on borrowings
• No market frictions, meaning no transaction costs, taxes or restrictions on short sales.

For the purpose of this Thesis, we are concerned with market models which can be used to price derivative products. Although these types of models can be used to construct trading strategies for the purposes of hedging, speculation and arbitrage, they are not used to make short term predictions, but rather to develop strategies to fit in with a trader’s goals. Generally speaking, most short term predictions do not require a model at all, as a trader will react to global economic and political information that will cause shocks to the market, such as new interest rate figures. They will also react to company information that they have obtained either publicly or illegally. Hence the most accurate predictions are not really “predictions” at all. It is therefore important to point out that mathematical models are used to develop trading strategies and also to quantify a trader’s exposure to risk based on ongoing market trends. During the time of writing this thesis, there has been an example of a trader’s failing trading strategy reported in the media. It is alleged that a French so-called rogue trader of Société Générale claimed to have identified that the bank could make money off futures markets via a doubling strategy often referred to as the martingale strategy, the very strategy attributed to Nick Leeson and the collapse of Barings Bank. It seems as though the bank or the trader had not taken the possibility of ruin seriously enough, and nor were they prepared for the possibility of infinite loss before ultimately making a profit, as can be an outcome of doubling strategies. For such a strategy, if ruin is a factor, one must know when to quit and be
made to do so. The trading strategies that arise from derivative pricing models are termed “admissible”, in that strategies such as the doubling strategy which allow for the possibility of infinite ruin are not permitted.

Many mathematical models in finance use diffusion processes since the price path of a stock over time looks almost absolutely continuous. The first diffusion model for pricing stock options dates back to 1900 and is attributed to Louis Bachelier who used Brownian Motion to model equity and options markets. His work predates Einstein’s work on Brownian Motion in particle physics and was said to be years ahead of its time in the sense that it set the pathway for models like the Black-Scholes model which was developed decades later, in the 1970’s.

3. Introduction

The area of financial mathematics uniquely combines two very distinct disciplines: mathematics and finance. Their union results in the machinery required to create the complex theory underpinning the techniques of derivative pricing. By utilising this theory, academics and practitioners have been able to assess the feasibility of their models for derivative pricing and are sometimes even able to find closed-form solutions to contingent claims. In particular, the legendary paper of Black and Scholes (Black and Scholes, 1973), gives explicit formulae for option prices along with replication strategies used for hedging and arbitrage purposes.

The Black-Scholes model serves as the de facto standard among practitioners
for asset pricing applications. Despite its widespread use, however, there is a substantial amount of evidence that suggests the model fails to capture important empirical properties of asset returns data. While some debate still continues as to whether a more sophisticated model can overcome the shortcomings of the Black-Scholes model entirely – in particular, to eliminate the so-called *smile-effect* (Ederington and Guan, 2002) – abundant research has shown that these models can, in fact, reduce the volatility smile, allowing them to be more accurate tools for pricing and hedging.

So why does the Black-Scholes model remain so popular? The underlying reason for this is primarily due to the great costs associated with implementing the more sophisticated yet also more complicated models. This thesis will address these costs along with the shortcomings of the Black-Scholes model and the critical need for more elaborate yet minimally descriptive pricing models.

The purpose of the thesis is to introduce an option pricing algorithm for a new model named Fractal Activity Time Geometric Brownian Motion, which was constructed by Professor Chris Heyde (Heyde, 1999) to specifically address key features of asset returns data in a minimally descriptive manner. Some of these features are the Fractal Activity Time and the FAT tails which are important empirical properties of asset returns data. The name FATGBM was hence coined as a result of these FAT features. From now on, we shall refer to Fractal Activity Time Geometric Brownian Motion in its abbreviated form, FATGBM. In what
follows includes an introduction to the mathematics behind no arbitrage derivative pricing. This will demonstrate to the reader the mathematical tools required to assess the feasibility of the models discussed in this thesis for the purpose of option pricing. Chapter Two will introduce the Fractal Activity Time Geometric Brownian Motion model and show that it is in fact arbitrage-free but does not allow for a replicating portfolio. Chapter Three will construct the option pricing algorithm which can be used to price European options and Exotic options of a path-dependent nature. Chapter Four will introduce a novel approach to incorporating long memory into the simulation of future price paths, while Chapter Five will implement this theory to investigate the effect of Long Range Dependence on option prices. This chapter makes a significant contribution to the finance literature by providing great motivation for the continued research into long memory processes with an application to finance.

4. Arbitrage and Models

Prior to the paper of Black and Scholes (Black and Scholes, 1973), arbitrage-free prices had only been discovered for Forwards and Futures contracts through the strict relationship that must hold between the Forward price and the underlying asset. If the relationship fails to hold, arbitrage can easily be carried out through a simple buy-hold strategy. Since a similar strict relationship does not hold for Options, a similar buy-hold strategy does not apply. Prior to Black and Scholes, Options were valued by taking an expectation of the payoff using a
suitable probability distribution function for the underlying asset at the expiration time, relying on the strong law of large numbers to realise a long run profit determined by its expected value (Jarrow, 2010). Black and Scholes showed that, under their model, it is possible to construct trading strategies which realise an arbitrage profit if the Option price did not equal the Black-Scholes price. Of course, this would only be possible if the model were to be a true reflection of the underlying asset price. Nevertheless, the Black and Scholes paper paved the way for finding arbitrage-free prices under a specified model where this had not been done before. In doing so, it also showed how an Option may be hedged with the theoretical cost of creating the hedge being equal to the Black-Scholes price of the Option. With a market full of participants applying similar no-arbitrage and hedging methods, the market price for Options began to shift from its long run average to arbitrage-free prices which limited short run losses via the use of the hedging strategies (Jarrow, 2010). This goes to show that we should not accept the market price of a derivative as the optimal price and we should be less concerned with finding a model which fits market prices and more concerned with finding strategies which realise profits in the short run as well as the long run.

Although the Black and Scholes paper is now a celebrated classic, it had in fact undergone several rejections from top-tier journals prior to its publication (Gans and Shepherd, 1994). Over time, however, the ideas were slowly accepted by the economics and finance communities. This acceptance gave rise to the construction of the discrete time analogue of the Black-Scholes model in 1979, namely
the Cox-Ross-Rubenstein binomial tree (Cox, Ross and Rubinstein, 1979), and later resulted in the birth of no-arbitrage pricing via the use of martingales in the papers by Harrison and Kreps (1979) and Harrison and Pliska (1981).

The construction of FATGBM and its option pricing algorithm rely heavily on the theories that govern no-arbitrage pricing, so it is of utmost importance to describe the basis of this theory. Central to the methods of option pricing is the concept of arbitrage (*making something out of nothing*). A mis-priced financial contract is one which permits arbitrage; in other words, an investor is able to exploit a risk-free profit using that contract, at the expense of the issuing agent of that contract. The existence of arbitrageurs in the market leads to the almost complete annihilation of arbitrage opportunities, as prices adjust in reaction to the arbitrageurs’ trading strategies, either driving prices higher or lower until they revert to an equilibrium position. An option pricing model is thus viewed as practical if it results in a market model that is arbitrage-free (that is, it does not allow for mis-priced options). A market model is a model of the securities readily traded in the market, such as stocks, interest rates, currencies or indexes.

Since underlying securities are random functions of time, it is natural to model them using stochastic processes. While discrete time stochastic processes are often studied (for example, the Cox Ross Rubenstein binomial tree model), continuous time processes are usually employed, as securities are traded constantly.
while exchange markets are open. There is, however, great importance in studying discrete time processes, as their limits can be used in derivative pricing as an approximation to their continuous time analogues. Often, closed form solutions for certain options fail to exist, as in the case of an American put under the Black-Scholes model. In these cases, either numerical, lattice, or Partial Differential Equation methods are employed as an approximation to the theoretical value of the option. In the case of the Black-Scholes model, the Cox-Ross-Rubenstein binomial tree is often used, specifically for early exercise options like the American put where an explicit formula does not exist. This approximation is possible because the underlying process of the Black-Scholes model is the limit of the underlying process of the Cox-Ross-Rubenstein model; specifically, Geometric Brownian Motion is the limiting process of an exponential random walk. In particular, it has been shown (Amin and Khanna, 1994) that the American put price under the Cox-Ross-Rubenstein model does, in fact, converge in the number of time steps to the theoretical American put price under the Black-Scholes model. Reasonable accuracy is achieved after roughly 100 time steps.

5. Mathematics of Derivative Pricing – The Black Art

"Derivatives are financial weapons of mass destruction, carrying dangers that, while now latent, are potentially lethal" – Warren Buffett

The Black-Scholes model is also attractive for other reasons. In addition to it being easy to implement, it is also compatible with the no-arbitrage pricing theory
which dates back to the work of D.M. Kreps (1981). To explain this further, we will focus on continuous time market models exclusively, although an analogous argument can be applied to a discrete time setting.

In reality, trading within financial markets comes with costs such as brokers’ fees, margin requirements and taxes. There are also restrictions on holding negative amounts of stock or short selling. The theoretical continuous time market model is constrained to a theoretical framework of a frictionless security market where there are no costs or restraints. Such a model assumes that market participants are able to trade continuously, without cost, over a fixed time horizon, \( T \). This is a good starting point for understanding a more realistic financial market where there are frictions. The theoretical market model also allows the random structure of securities to be modeled by a complete, filtered probability space. The notation used to express this is \((\Omega, \mathcal{F}, \mathcal{F}_t, \mathbb{P})\), where \( \Omega \) represents the non-empty set of all possible outcomes, \( \mathcal{F} \), the \( \sigma \)-algebra of subsets of \( \Omega \), \( \mathcal{F}_t \), the filtration of the \( \sigma \)-algebra indexed by time, and \( \mathbb{P} \) the associated probability measure. Specifically, a complete filtered probability space satisfies the following:

i) \( \mathcal{F}_0 \) contains all the \( \mathbb{P} \)-null sets of \( \mathcal{F} \),

ii) \( \mathcal{F}_t = \bigcap_{u > t} \mathcal{F}_u = \mathcal{F}_{t+} \); that is, \( \mathcal{F}_t \) is right-continuous.

In the notation of Bingham and Kiesel (2004), we will suppose the market model
contains $d+1$ traded assets whose random prices are modeled by stochastic processes given by $S_0, ..., S_d$. Here, $S_0$ represents a risk-free asset such as the money market account.

As mentioned in the preceding section, an efficient market should not allow a market participant to construct trades which result in a risk-free profit. Because in reality it is assumed that financial markets are efficient, it is necessary to have a market model which is also efficient; in other words, it is necessary to find market models which do not allow trading strategies that permit arbitrage. Since our market model allows market participants to trade continuously, we can express a trading strategy in terms of a stochastic process $\varphi_i(t)$ that is $\mathcal{F}_t^\omega$-measurable. This means trading strategies follow a predictable process. This measurability condition allows a trader to determine the amount of asset $i$ that is to be held at time $t$, based on information prior to time $t$. Since there are no restrictions in this theoretical market, short selling is allowed and the process may assume negative values. The process must also satisfy the following conditions to ensure that the stochastic integral $\int_0^T \varphi(u)dS(u)$ exists and the trading strategy is admissible:

i) $\int_0^T E(\varphi_0(t))dt < \infty$

ii) $\sum_{i=0}^d \int_0^T E(\varphi_i^2(t))dt < \infty$

A trading strategy is defined to be an $\mathbb{R}^{d+1}$-valued predictable, locally-bounded
5. MATHEMATICS OF DERIVATIVE PRICING – THE BLACK ART

process on $[0, T]$, satisfying conditions i) and ii) above and is given by,

\begin{equation}
\varphi(t) = (\varphi_0(t), \ldots, \varphi_d(t)), \ t \in [0, T].
\end{equation}

When working out arbitrage strategies or optimal trading strategies, a trader will be interested in holding a combination of the tradeable assets. We will refer to this combination of assets as a portfolio. In our market model, a portfolio will be a combination of the $d + 1$ tradeable assets. At any time point, its value can be defined as follows:

**DEFINITION 1.1.** The value of the portfolio $\pi$ at time $t$ is given by the scalar product

\begin{equation}
V_\varphi(t) := \varphi(t) \cdot S(t) = \sum_{i=0}^{d} \varphi_i(t)S_i(t), \ t \in [0, T].
\end{equation}

The gains from holding the portfolio can be expressed in terms of the gains process, $G_\varphi(t)$, given by

\begin{equation}
G_\varphi(t) = \int_0^t \varphi(u)dS(u) = \sum_{i=0}^{d} \int_0^t \varphi_i(u)dS_i(u).
\end{equation}

The gains process is important because it is of interest to analyse portfolios that are comprised of trading strategies that only involve an initial set up cost, $V_\varphi(0)$. This set up cost is the value of the portfolio at time 0. A portfolio of trading strategies which does not result in any cash inflow or outflow following set up is called a self-financing portfolio.
DEFINITION 1.2. A trading strategy $\varphi$ is called self-financing if the wealth process $V_\varphi(t)$ satisfies

$$V_\varphi(t) = V_\varphi(0) + G_\varphi(t).$$

In economics, we are often concerned about expressing the price of goods in relative terms so we do not have to express it in monetary units; for example, one may wish to express the market price of soup in terms of the market price of pasta in order to see how the price of soup grows relative to the price of pasta over time. Financial markets are no different, since we are analysing the market price of assets over time, it seems natural to look at relative prices as well. It turns out that in some circumstances the relative price process becomes a martingale under a change of measure. Such a measure would not necessarily exist if the asset was not expressed relative to another. This can be explained intuitively by the fact that an asset may experience infinite growth in absolute monetary terms due to inflation but not in relative terms. Mathematically, this standardisation allows certain requirements to be satisfied in order to apply a change of measure.

To explain this idea in detail, we must first consider the discounted stock price process $\tilde{S}_t = \frac{S_i(t)}{S_j(0)}$, $i \neq j$. The quantity $S_j$ is called a numéraire and is any of the $d$ tradeable assets that are strictly positive. In addition to considering the relative value of a stock price process with respect to a numéraire, we are also interested in the relative value of self-financing portfolios with respect to a numéraire. We define the discounted value of a portfolio to be

$$\tilde{V}_\varphi(t) := \frac{V_\varphi(t)}{X_t}.$$
The Numéraire Invariance Theorem (Geman, El Karoui and Rochet, 1995) states that a self-financing trading strategy, $\varphi$, remains self-financing after a change in numéraire. This result means that the discounted value process given in (5.5), can be expressed as,

\[
V^t = G^t = V^t / \mathbb{S}_t^\varphi,
\]

Our objective is to find market models which do not permit arbitrage so that we are able to find arbitrage-free prices for derivative products. We know that arbitrage occurs if we are able to set up a portfolio at no cost with the chance of strictly positive cash flow at some time in the future and with no chance of negative cash flow at any point in time. More formally, arbitrage is defined as follows:

**Definition 1.3.** A self-financing trading strategy $\varphi$ is called an arbitrage opportunity if the wealth process $V^\varphi$ satisfies the following set of conditions:

\[
V^\varphi(0) = 0, \quad \mathbb{P}(V^\varphi(T) > 0) = 1, \quad \text{and} \quad \mathbb{P}(V^\varphi(T) > 0) > 0.
\]

Central to no-arbitrage pricing theory is the concept of an equivalent martingale measure. This is a measure $\mathbb{Q}$ that makes the discounted stock price process, $\mathbb{S}_t$, a $\mathbb{Q}$-local martingale. If we can transform a discounted stock price process into a martingale via a change of measure, the value of a discounted self-financing trading strategy (5.6) will also be a martingale under the measure $\mathbb{Q}$. The value of the self-financing trading strategy at some time $0 \leq t \leq T$ can then be found by taking an expectation of the terminal payoff at $T$ under the measure $\mathbb{Q}$. If a
self-financing trading strategy has the same terminal payoff as a derivative product, the value of that derivative at some time $0 \leq t \leq T$ must have the same value as the self-financing trading strategy at that time, otherwise there would be an arbitrage opportunity in the market model. So, a self-financing trading strategy that has the same terminal payoff as a derivative is said to replicate the derivative. The portfolio of stocks that comprise the self-financing trading strategy is called a replicating portfolio.

The next theorem shows that if there exists a probability measure $Q$ defined on $(\Omega, \mathcal{F})$ that is equivalent to $\mathbb{P}$ and which results in the discounted stock price process $\tilde{S}$ being a $Q$-local martingale, then the market model contains no arbitrage opportunities. This result can be proved easily by considering the set of all self-financing trading strategies (as defined above) for which $V^\varphi(t) > Q$, $\forall \ t \in [0,T]$, and then noting that $V^\varphi(t)$ is a non-negative $Q$-local martingale, and hence, a supermartingale.

**Theorem 1.1.** Assume there exists at least one equivalent martingale measure for the discounted stock price process. Assume also that all self-financing trading strategies, $\varphi$, satisfy $\tilde{V}^\varphi(t) \geq 0$, $\forall \ t \in [0,T]$. Then the market model does not permit arbitrage.

**Proof.** $\tilde{V}^\varphi(t)$ is a non-negative martingale, hence a supermartingale. This means that

$$E_Q(\tilde{V}^\varphi(t) \mid \mathcal{F}_u) \leq \tilde{V}^\varphi(u), \ \forall \ u \leq t \leq T.$$
By definition, arbitrage can only exist if $\tilde{V}_\varphi(0) = 0$. Therefore,

\begin{equation}
E_Q(\tilde{V}_\varphi(t)) \leq 0, \quad \forall \ 0 \leq t \leq T.
\end{equation}

But, $\tilde{V}_\varphi(t) \geq 0, \forall \ t \in [0, T]$, so $E_Q(\tilde{V}_\varphi(t)) = 0$.

An arbitrage opportunity must also satisfy $P(\tilde{V}_\varphi(T) > 0) > 0$.

In this case, $Q(\tilde{V}_\varphi(T) > 0) = P(\tilde{V}_\varphi(T) > 0) = 0$. So, the market model does not contain any arbitrage opportunities.

We saw above that the existence of an equivalent martingale measure implies that the market model is free of arbitrage, but it is not possible to show that the reverse holds in general. It turns out that the reverse does hold for a wide range of processes, including some with infinite jumps, if a process satisfies the “No Free Lunch with Vanishing Risk” (NFLVR) condition. We will not prove that the NFLVR condition implies the existence of an equivalent martingale measure, as the mechanical details of the proof use heavy theory from functional analysis and so are not particularly relevant to the content of this thesis, but it is of interest to explain what is meant by this condition so that we can state a Fundamental Theorem of Asset Pricing which may be applied to a wide class of stochastic processes. In the next chapter, we will show that an equivalent martingale measure exists for the FATGBM process, which ensures that the model is free of arbitrage. This means we do not rely on the generalisations of the Fundamental Theorem of Asset Pricing, but it is still the basis of derivative pricing theory. The reader
In order to state the general version of the Fundamental Theorem of Asset Pricing, we will need to explain the No Free Lunch with Vanishing Risk condition, which is a stronger condition than the No Free Lunch condition of the Kreps-Yan theorem. Both conditions mathematically link the meaning of “essentially no-arbitrage” or “No Free Lunch” to a defined set of trading strategies for stock price processes which possess an equivalent martingale measure. The Kreps-Yan theorem uses simple admissible trading strategies defined by simple integrands (see Delbaen and Schachermayer, 1994) to describe the No Free Lunch condition, whereas the general version of the Fundamental Theorem of Asset Pricing moves to use general trading strategies defined by admissible integrands (again, see Delbaen and Schachermayer, 1994) to define the NFLVR condition. By generalising the type of trading strategies through admissible integrands and not simple integrands means that models like the Black-Scholes model which use Brownian Motion can be included in the Fundamental Theorem of Asset Pricing, as the replicating trading strategies for the Black-Scholes model are not defined through simple integrands. These generalisation of the trading strategies still come with some restrictions, one being that doubling or martingale strategies are not allowed, as admissible trading strategies must be uniformly bounded from below.

Let $S$ denote a locally-bounded $\mathbb{R}^d$-valued semi-martingale, and let $H$ denote an $\mathbb{R}^d$-valued predictable process that is $S$ integrable with $H_0 = 0$, $H \cdot S \geq -a$, ...
and assume that $\lim_{t \to \infty} (H \cdot S)_t$ exists a.s. The process $H$ is said to be admissible if there exists at least one $a > 0$ for which the above holds. In other words, $H$ is admissible if it is uniformly bounded from below. Define also the following sets:

$$\mathcal{K} = \{\text{admissible processes, } H\}$$

$$\mathcal{C}_0 = \mathcal{K} - L^\infty_+ = \{f - k \mid f \in K, f \in L^\infty, k \geq 0\}$$

$$\mathcal{C} = \mathcal{C}_0 \cap L^\infty.$$ 

A process $S$ satisfies the no-arbitrage property if:

$$\mathcal{K} \cap L^0_+ = \{0\},$$

or, equivalently:

$$\mathcal{C} \cap L^0_+ = \{0\}.$$ 

A process $S$ satisfies No Free Lunch with Vanishing Risk (NFLVR) if:

$$\tilde{\mathcal{C}} \cap L^\infty_+ = \{0\}.$$ 

This condition means that if the NFLVR property is satisfied, then there is no
positive element $f \in \tilde{C}$ for which the negative parts of a sequence of final payoffs relating to the trading strategy $(f_n^-)_{n \geq 1}$ in $C$ tend to 0 uniformly, and such that $f_n$ tends to $f$ with $P(f > 0) > 0$. We can now state the Fundamental Theorem of Asset Pricing.

**Theorem 1.2. (Fundamental Theorem of Asset Pricing).** For a financial market model with a (locally) bounded price process, there exists an equivalent (local) martingale measure if and only if the NFLVR condition holds.

This theorem can also be extended to unbounded price processes (see Delbaen and Schachermayer, 1994).

By applying the Fundamental Theorem of Asset Pricing, a practitioner can specify an arbitrage-free model which will ensure any derivative price derived using the model is free of arbitrage with respect to that model. We have already seen that if a contingent claim can be replicated by a self-financing trading strategy, then the price of that claim is the price of that trading strategy. So what we would really like to know now is: how and when can we be assured of being able to find a self-financing trading strategy that replicates the payoff of a contingent claim.

A replicating portfolio is an admissible self-financing trading strategy that replicates the payoff of a contingent claim. A contingent claim $X$ is attainable if it can be replicated by an admissible trading strategy. It follows that if $X$ is attainable,
then the price process of the contingent claim, $\pi_X(t)$, must have the same value as the replicating portfolio. If the discounted gains process,

$$\sum_{i=1}^{d} \int_{0}^{t} \varphi_i(t)d\tilde{S}_i(t),$$

is a $Q$-martingale, then the trading strategy is said to be $Q$-admissible and,

$$\frac{\pi_X(t)}{S_0(t)} = \mathbb{E}_Q \left[ \frac{X}{S_0(T)} \big| \mathcal{F}_t \right], \text{ where } S_0 \text{ is the numéraire.}$$

Further, if $Q$ is the unique equivalent martingale measure for the market model, then every contingent claim $X$ satisfying

$$\frac{X}{S_0(T)} \in L^1(\mathcal{F}, Q)$$

is attainable (see Harrison and Pliska, 1981), and the market model is said to be complete. Hence, uniqueness guarantees that replicating portfolios exist for every contingent claim. This means that the existence of a unique equivalent martingale measure implies that any discounted contingent claim admits an integral representation under $Q$ which is a martingale and of the form,

$$\frac{\pi_X(t)}{S_0(t)} = \tilde{V}_\varphi(0) + \int_{0}^{t} \varphi(t)d\tilde{S}(t).$$

The question now arises as to how to price contingent claims in incomplete market models, where there is no unique equivalent martingale measure and there is no way of proving that all contingent claims are attainable. One possibility
would be to find the set of $\varphi(t)$ which admits a martingale for the integral representation given by (5.13) under any of the equivalent martingale measures $P^*$. It is likely though that it may not result in a very large set of contingent claims that can be priced and therefore may not be very practical. In Chapter 2, we will explore the strategies used for pricing in incomplete markets, but we should always remember that in reality no asset price model actually reflects the true evolution of that asset and therefore there is no correct model, no correct price and no optimal trading strategy outside the constructs of the market model. In some cases, there are observable market prices and arbitrage-free bounds within which a derivative must lie. An arbitrage-free model ensures that prices obtained using that model lie within arbitrage-free bounds, but it does not ensure that prices obtained are consistent with the market, hence the volatility smile arising from the Black-Scholes model (see Chapter 5 for more details). In order to make distinctions between the various models used by practitioners and their feasibility in the finance world, it is important to refine what is meant by a desirable model. Desirability depends on what the practitioner’s goals are. Ideally a model would be able to price commonly-traded contingent claims, such as European, American and Exotic Call and Put Options, as well as being able to produce accurate, implementable trading strategies for the purpose of hedging and replication. Accurate hedging and replication strategies are what allow a practitioner to make money from options available on the market as well as to structure derivative products knowing they can cover their losses. Accuracy is very difficult to achieve, and trading strategies are not easily derived or implementable, so many practitioners
structure a derivative product without being able to cover their position optimally. A desirable model for them is one which can be calibrated to heavily traded derivative products, which can then be used to construct trading strategies to cover their position on their illiquid claim. Even though these types of models may result in some optimal strategy which covers the position of the party who constructed the product, it appears that, in many cases, practitioners are more concerned with deriving a price that the market is likely to find acceptable, rather than hedging their position. It should also be noted that models which do permit arbitrage can also be useful in producing optimal trading strategies as long as the trading strategies derived from the model which do admit arbitrage are not able to be implemented in practice. It may be that these models result in profitable trading strategies which fulfill the goals of some practitioners such as hedge funds. Where FATGBM sits within the context of trading strategies is still an area of ongoing research.

6. Long Memory Processes and Option Pricing

Long memory has been found in economic time series since the sixties when Granger (1966) considered it to be the “typical spectral shape of an economic variable” and Mandelbrot commented that most economic time series exhibit the “Joseph effect” (Mandelbrot and Wallis, 1968), a term which stems from Hurst’s discovery arising from the Nile River data (see Chapter 2). Long memory for stock market volatility has been studied by Bollerslev and Mikkelsen (1999); multifractality in asset returns has been considered by Calvet and Fisher (2002);
while option pricing for stochastic volatility models exhibiting long memory is discussed in Comte and Renault (1998). Fractional Brownian Motion is not a semi-martingale, and therefore this type of model does not guarantee an arbitrage-free market model (Rogers, 1997), but the market may be made free of arbitrage by considering a class of admissible strategies (Bender et al., 2007). Pricing and hedging strategies are therefore still useful to consider (see, for example, Shiryaev, 1998; Biagini and Oksendal, 2003; and Djehiche and Eddahbi, 2001). Long memory processes which are martingales such as FATGBM are not common and have not been studied in great detail for option pricing. This thesis extends the option pricing literature to consider pricing methods for martingales with long memory. Taylor (2001) states that “the long memory assumption is found to have relatively minor impact upon smile effects”. Our novel approach introduced in Chapter 4 which incorporates price path history in the simulation of future price paths, leads to the discovery that long memory can have a significant impact on the smile effect.
CHAPTER 2

Fractal Activity Time Geometric Brownian Motion – FATGBM

1. Introduction

In this chapter, we begin with a discussion on Long Range Dependence (LRD) and its relevance to finance applications. We then introduce Fractal Activity Time Geometric Brownian Motion (FATGBM), the LRD model first developed in Heyde and Leonenko (2005). Further, we discuss the construction and features of this process and use the theory developed in the previous chapter to assess whether FATGBM is a theoretically viable derivative pricing model. In particular, we prove that the FATGBM market model is arbitrage-free but not complete. In an attempt to develop a martingale representation theorem for FATGBM, we show that there is a restricted class of derivatives whose payoff can be replicated under strict conditions. We then conclude with a discussion on future research in the development of a martingale representation theorem for FATGBM which would enable completion of the market using liquid call options, as introduced by Carr et al. (2001) for the Variance Gamma Process.

2. Long Range Dependence and Self-Similarity

Harold E. Hurst was an English hydrologist who made a phenomenal discovery while analysing the Nile River discharge time series for the purpose of designing
the Aswan High Dam (Hurst, 1951). Hurst discovered a clustering of weather patterns that resulted in time periods of persistent wet years followed by time periods of persistent drought years, known as the “Hurst Phenomenon”. These Nile River patterns led him to a statistical discovery, namely that the statistic $\log(\frac{R}{S})$, where $\frac{R}{S}$ is the rescaled range statistic – the range of a series of values divided by the standard deviation – behaves like a constant plus $H \log k$, for large $k$, instead of $\frac{1}{2} \log k$, as is the case for Markov processes (Beran, 1994). In the preceding, the value $H$ is referred to as the Hurst exponent, and it is an index of long-range dependence. It was not until the sixties that B.B. Mandlebrot interpreted this power law as a symptom of underlying scaling and proposed that the Nile River discharge be modeled by a Fractional Brownian Motion (Mandelbrot and Van Ness, 1968). Until this time, these kinds of hydrological observations were being modeled by Markov and ARMA processes which required an increasing number of parameters with increasing sample size to compensate for slowly decaying autocorrelation functions (Beran, 1994). Long memory or long range dependence is a name given to processes which exhibit a correlation function that decays so slowly that they are not summable:

\[
(2.1) \quad \sum_{k=-\infty}^{\infty} \rho(k) = \infty.
\]

Of course, in the case of an AR(1) process where $\rho = 1$, otherwise known as a random walk, the autocorrelation function decays slowly, but the process is also non-stationary. For values of $\rho$ close to 1 and $-1$, the AR(1) process is stationary but exhibits properties that may appear non-stationary, such as a
fluctuating mean. We can look to a wider class of processes to construct a sta-
tionary series with non-summable correlations, resulting in a process which looks
non-stationary. These processes can be used to model a range of phenomena such
as the Nile River discharges or even asset price data which is known to exhibit
long memory (Willinger et al., 1999). In probability, the notion of long memory
is considered in association with stationary processes only (Samordnitsky, 2005).
Stationary processes which are long range dependent are defined as follows:

**Definition 2.1.** Let \( X_t \) be a stationary process for which the following holds:
there exists a real number \( \alpha \in (0, 1) \) and a constant \( c_\rho > 0 \) such that

\[
\lim_{k \to \infty} \frac{\rho(k)}{c_\rho k^{-\alpha}} = 1.
\]

Then \( X_t \) is called a stationary process with long memory or long range dependence
or strong dependence, or a stationary process with slowly decaying or long-range
correlations.

The Hurst parameter \( H \) is then set to equal \( 1 - \frac{\alpha}{2} \) so that long memory processes
arise for \( \frac{1}{2} < H < 1 \) (Beran, 1994).

Stationary processes which exhibit long range dependence fall under the class
of self-similar (ss) processes. Self-similarity means that an object is composed
of sub-units and sub-sub-units on multiple levels that (statistically) resemble the
structure of the whole object (Feder, 1988).
DEFINITION 2.2. An $\mathbb{R}^d$-valued stochastic process $\{X(t), t \geq 0\}$ is said to be “self-similar” if for any $a > 0$, there exists a value $b > 0$ such that

\begin{equation}
\{X(at)\} \stackrel{d}{=} \{bX(t)\}.
\end{equation}

In this context, a self-similar process is a mono-fractal process as well. The link between self-similar processes and the Hurst parameter $H$ is given by the following theorem.

THEOREM 2.1. (Lamperti, 1962) If $\{X(t), t \geq 0\}$ is nontrivial, stochastically continuous at $t = 0$ and self-similar, then there exists a unique $H \geq 0$ such that $b$ in Definition 2.2 can be expressed as $b = a^H$.

The following theorem can be used to show that Brownian Motion is $\frac{1}{2}$-ss.

THEOREM 2.2. (Taqqu, 1981) Let $\{X(t)\}$ be real-valued $H$-self-similar with stationary increments and suppose that $E[X(1)^2] < \infty$. Then,

\begin{equation}
E[X(t)X(s)] = \frac{1}{2} \{t^{2H} + s^{2H} - |t - s|^{2H}\} E[X(1)^2].
\end{equation}

This result can also be used to show that the autocorrelation function cannot be summed as in (2.1). Let $\{X(t), t \geq 0\}$ be an $H$-ss process with stationary increments, for $0 < H < 1$, and nondegenerate for each $t > 0$ with $E[X(1)^2] < \infty$, and define

\[ \xi(n) = X(n + 1) - X(n), \quad n = 0, 1, 2, \ldots, \]

\[ r(n) = E[\xi(0)\xi(n)], \quad n = 0, 1, 2, \ldots, \]
Then, by Theorem 2.2 (and assuming that $X(0) = 0$ almost surely), for $n \geq 1,$

\[
    r(n) = E[\xi(0)\xi(n)] = E[X(1)X(n+1)] - E[X(1)X(n)]
\]

\[
    = \frac{1}{2} (n+1)^{2H} - 2n^{2H} + (n-1)^{2H} E[X(1)^2],
\]

and by Taylor’s Theorem,

\[
    r(n) \sim H(2H - 1)n^{2H-2}E[X(1)^2], \quad \text{as } n \to \infty, \quad \text{if } H \neq \frac{1}{2},
\]

\[
    r(n) = 0, \quad \text{for all } n, \quad \text{if } H = \frac{1}{2},
\]

which implies the following:

- if $0 < H < \frac{1}{2}, \sum_{n=0}^{\infty} |r(n)| < \infty,$
- if $H = \frac{1}{2},$ then $\{\xi\}$ is uncorrelated,
- if $\frac{1}{2} < H < 1, \sum_{n=0}^{\infty} |r(n)| = \infty.$

This result tells us that if we have an $H - ss$ process with stationary increments, and $H \in (\frac{1}{2}, 1)$ with covariance function given by (2.4), then the process is Long Range Dependent. Using this theory, one is able to construct a long range dependent process which is also a martingale. This observation is very important in the context of finding a LRD derivative pricing model. Consider a LRD martingale which is $H - ss$, then Theorem 2.3 states that such a martingale can also be represented as a time change of a Brownian Motion. That means, it is natural to construct a LRD martingale as a Brownian Motion that is subordinated by a time change endowed with $H - ss$ properties. This is the basic construction of the Fractal Activity Time Geometric Brownian Motion process.
2. Fractal Activity Time Geometric Brownian Motion – FATGBM

**Definition 2.3.** Let \( A = (A_t)_{t \geq 0} \) be an adapted, right continuous increasing process, which need not always be finite valued. The **change of time** (also known as a time change) associated with \( A \) is the process

\[
\tau_t = \inf \{ s > 0 : A_s > t \}.
\]

**Theorem 2.3.** (Dambis/Dubins-Schwarz, 1965; see, for example, Theorem 42 of Protter (2005), p. 88.) Let \( M = (M_t)_{t \geq 0} \) be a continuous local martingale with \( M_0 = 0 \) and such that \( \lim_{t \to \infty} [M, M]_t = \infty \) a.s. Let

\[
T_s = \inf \{ t > 0 : [M, M]_t > s \}.
\]

Define \( \mathcal{G}_s = \mathcal{F}_{T_s} \) and \( B_s = M_{T_s} \). Then \( (B_s, \mathcal{G}_s)_{s \geq 0} \) is a standard Brownian motion. Moreover \( ([M, M]_t)_{t \geq 0} \) are stopping times for \( (\mathcal{G}_s)_{s \geq 0} \) and

\[
M_t = B_{[M, M]_t} \text{ a.s., } 0 \leq t < \infty.
\]

That is, \( M \) can be represented as a time change of a Brownian motion.

3. Fractal Activity Time Geometric Brownian Motion

In the Introduction we discussed that the Geometric Brownian Motion (GBM) price process is the most commonly used model in industry, despite its well known deficiencies, a key reason being its parsimony with regard to complexity. The GBM model is described as,

\[
S_t = S_0 e^{\mu t + \sigma W(t)},
\]
where $\mu$ and $\sigma$ are constants and $S_t$ is the stock price at time $t$. Letting

$$X_t = \log\left(\frac{S_t}{S_{t-1}}\right)$$

be the increments of the log price process, we see that

$$X_t = \mu + \sigma(W(t) - W(t - 1)) \stackrel{d}{=} \mu + \sigma W(1) \stackrel{d}{=} N(\mu, \sigma),$$

with stationary and independent increments. In contrast, most log asset returns series display leptokurtic distributions (heavy tails with high peaks) and evidence of strong dependence. The distributions of the logarithmic returns have been well studied and can often be fitted well by Student $t$-distributions (Hurst et al., 1997; Hurst and Platen, 1997; Heyde, 1999; Heyde and Liu, 2001; Heyde and Gay, 2002; and Bingham and Kiesel, 2002). In addition to the distribution functions themselves, sample autocorrelation functions reveal strong dependence in logarithmic returns. Financial time series reveal quickly decaying autocorrelation functions for log returns but slowly decaying autocorrelations for squared and absolute log returns, resulting in nonnegligible values for large lags (Heyde and Yang, 1997; Heyde, 1999; Heyde and Liu, 2001; Anh et al., 2002). Prior to Heyde (1999), self-similar stationary increment processes with long memory and Student $t$-distributed marginals had not been studied in great detail and with respect to applications in finance. Heyde (1999) introduced the Fractal Activity Time Geometric Brownian Motion (FATGBM) process as a minimally descriptive risky asset price model which overcomes the deficiencies of the GBM model,
while still retaining parametric parsimony. FATGBM is defined to be,

\[(3.4) \quad S_t = S_0 e^{\mu t + \sigma W(T_t)},\]

where,

\[(3.5) \quad T_t = \sum_{i=1}^{t} \frac{\nu - 2}{\sum_{j=1}^{\nu} \eta_j i},\]

and \(\eta_j(i) = \eta_{ij}, j = 1, ..., \nu\), are \(\nu\) independent copies of a multivariate Gaussian random vector with zero mean and covariance function \((1 + \tau^2)^{\frac{\alpha}{2}}\), for \(\alpha = 1 - H\), and \(\frac{1}{2} < H < 1\).

From the definition of \(T_t\), the process is not continuous as it is only defined for integer values of \(t\). But we can define a continuous \(T_t\) process by letting \(T_t\), for non-integer values of \(t\), be the line joining the two random points \((T_{[t]}, T_{[t]+1})\), where the slope of the line at \([t]\) is \(\mathcal{F}_{[t]}\)-measurable. The process \(T_t\) is then a strictly positive increasing function on all intervals in \((0, \infty)\).

The process \(T_t\) is also independent of \(W\), which means by Definition 2.3 and equation (2.5), \(\{W(T_t), \mathcal{F}_s, s > 0\}\) is a time change Brownian motion and a martingale, which results in a market model that is consistent with the efficient market hypothesis of no arbitrage (still yet to be proved formally). The process \(T_t\) is also known as Activity Time since it is not equal to clock time and can intuitively be thought of as an unobserved quantity representing trading time or trading volume.
We will now show that FATGBM has the following properties:

1. The Activity Time process has stationary but dependent increments
2. The Activity Time increments result in marginals of the Gamma type
3. The logarithmic FATGBM increments are equal in distribution to a symmetric scaled $t$-distribution
4. The logarithmic FATGBM increments are uncorrelated but their absolute and squared increments have slowly decaying autocorrelations
5. FATGBM is an asymptotically $H - ss$ process with stationary increments.

These properties are proved below:

1. From equation (3.5), the Activity Time increments are given by,

$$\sum_{j=1}^{\nu} \eta_j^2(i).$$

Since $\{\eta\}$ is a stationary process with dependent increments, by definition, it follows that,

$$T_t - T_{t-u} \overset{d}{=} T_u, \forall u \geq 1,$$

and $T_t - T_{t-u}$ is not independent of $T_{t-u}$.

2. It follows from equation (3.5) that the differenced Activity Time process consists of the sum of $\nu$ independent $\chi^2$ random variables, each with a single degree of freedom. By construction of the Noncentral Chi-square distribution, the sum of $\nu$ independent Chi-square random variables with
mean 0 results in a central Chi-square random variable with \( \nu \) degrees of freedom. Let \( X_i \sim N(\mu, \sigma), i = 1, \ldots, k \), be \( k \) independent Normal random variables. Then

\[
\sum_{i=1}^{k} \left( \frac{X_i}{\sigma_i} \right)^2
\]

is distributed according to a Noncentral Chi-squared distribution with \( k \) degrees of freedom and noncentrality parameter \( \lambda = \sum_{i=1}^{k} (\frac{\mu_i}{\sigma_i})^2 \). When \( \lambda = 0 \), the distribution is simply a central Chi-square distribution.

Let \( Y \sim \chi^2_{\nu} \). Since Chi-squared random variables appear in the denominator of equation (3.6), we are interested in the distribution of \( \frac{1}{Y} \). This distribution is called the Inverse Chi-squared distribution, which is also an Inverse Gamma distribution. From the definition of the Inverse Gamma distribution,

\[
(3.8) \quad \frac{1}{Y} \sim R\Gamma \left( \frac{\nu}{2}, \frac{1}{2} \right),
\]

which means that,

\[
(3.9) \quad \frac{\nu - 2}{Y} \sim R\Gamma \left( \frac{\nu}{2}, \frac{\nu - 2}{2} \right).
\]

Resulting in,

\[
(3.10) \quad E \left[ \frac{\nu - 2}{Y} \right] = \frac{\frac{\nu - 2}{2}}{\frac{\nu}{2} - 1} = 1.
\]

This shows that the differenced Activity Time process is centered around 1, so for every clock time increment of 1 unit, Activity Time is expected to
increase by 1 unit also, but since Activity Time is distributed according to an Inverse Gamma distribution, it will take on values in \((0, \infty)\) such that its mean is 1.

(3) Let

\begin{equation}
X(t) = \log \left( \frac{S_t}{S_{t-1}} \right),
\end{equation}

and note that,

\begin{equation}
W(T_t) - W(T_{t-1}) \overset{d}{=} W(T_t - T_{t-1}) \overset{d}{=} \sqrt{T_1} W(1).
\end{equation}

Then,

\begin{equation}
X(t) \overset{d}{=} \mu + \sigma \sqrt{T_1} W(1).
\end{equation}

Thus, the log FATGBM increments are the ratio of a Normal random variable, \(Z \sim N(0, 1)\), and the square root of an independent Chi-squared random variable, \(Y \sim \chi^2(\nu)\). More specifically,

\begin{equation}
\frac{(\nu - 2)Z}{\sqrt{Y}}
\end{equation}

has a symmetric scaled \(t\)-distribution centered around 0, and the log returns process \(X_t\) has the distribution \(T(\nu, \nu^{\frac{1}{2}}, \mu)\) (see Heyde (2005) for further details). This allows the heaviness of the tails to be controlled by the parameter \(\nu\), so for large \(\nu\), the distribution of the log returns is approximately Normal.

(4) Using \(X(t)\) defined in (3.11),

\[
\text{cov}(X_{t+k}, X_t) = \sigma^2 E\{[W(T_{t+k}) - W(T_{t+k-1})][W(T_t) - W(T_{t-1})]\}
\]
\[ = \sigma^2 E\{E\{[W(T_{t+k}) - W(T_{t+k-1})][W(T_t) - W(T_{t-1})]\}|\{T\}] = 0 \]

since the covariance of Brownian motion is given by \(\min(s, t)\).

While \(\{X\}\) is uncorrelated, it is not independent. We show that there is long memory in \(\{X\}\) through the dependence structure of the \(\{X^2\}\) process. Let,

(3.15) \[ \tau_t = T_t - T_{t-1}, \]

then

\[ \text{cov}([X_{t+k} - \mu]^2, [X_t - \mu]^2) = \sigma^4 E\{[W(\tau_{t+k})]^2[W(\tau_t)]^2\} \]
\[ = \sigma^4 E[\{W(1)\}]^4 \text{cov}(\tau_{t+k}, \tau_t). \]

Since \(\text{cov}(\tau_{t+k}, \tau_t)\) is defined to be a slowly decaying covariance function given by the properties of (3.5), the long range dependent structure in \(\{T\}\) is also evident in \(\{X\}\).

It follows that,

(3.16) \[ \text{cov}([X_{t+k} - \mu], [X_t - \mu]) = \sigma^2 (E[|W(1)|]\text{cov}(\tau_{t+k}^{1/2}, \tau_t^{1/2})), \]

where \(E[|W(1)|] = \frac{2}{\pi}\).

(5) Heyde (2005) shows that \(T_{[nt]} - [nt]\) is asymptotically self-similar. That is, to a good degree of first approximation, the process \(T_{ct} - ct, c > 0\), can be approximated in the sense of finite-dimensional distributions by the
process $c^H(T_t - t)$. This enables $T_t$ to be approximated by the following formula,

\begin{equation}
T_t \overset{d}{=} t + t^H(T_1 - 1).
\end{equation}

Heyde (2005) also shows that because \{T\} is a strictly increasing process, exact self-similarity is not possible because if (3.17) held exactly for all $t > 0$ and $c > 0$, then for any $0 < \delta < 1$,

\begin{align*}
T_{t+\delta} - T_t - \delta &\overset{d}{=} T_\delta - \delta \\
&\overset{d}{=} \delta^H(T_1 - 1),
\end{align*}

and

\begin{align*}
P(T_{t+\delta} - T_t < 0) &= P(\delta + \delta^H(T_1 - 1) < 0) \\
&= P(\delta^H T_1 < \delta^H - \delta) \\
&= P(T_1 < 1 - \delta^{1-H}) > 0.
\end{align*}

Property 5 sets FATGBM apart from stochastic volatility models which specify a stochastic differential equation for the volatility process \{\sigma(t)\}. In this case, the stochastic volatility component is represented by the Activity Time process \{T_t\}, but the hypothesised stochastic differential equation for $\sigma^2T_t = \int_0^t T_u du$ has been circumvented by using the asymptotic self-similarity of the process. It should be noted that this thesis does not rely on the asymptotic self-similarity property to price options in the way that Heyde (1999) proposes.
Even though the Activity Time process is not directly observed, Heyde (1999) proposes the use of Itô’s Lemma to empirically construct \( \{T_t\} \) in a discretised approximation. This will allow us to estimate \( \sigma \) as we will discuss briefly in chapter 4. We will now apply Itô’s Lemma to define this discretisation.

**Theorem 2.4. (Itô’s Lemma)** Let \( X \) be a continuous semi-martingale taking values in an open subset \( U \subseteq \mathbb{R} \). Then for any twice differentiable function \( f : U \to \mathbb{R} \), \( f(X) \) is a semi-martingale and,

\[
\begin{align*}
    df(X) &= f'(X)dX + \frac{1}{2}f''(X)d[X,X],
\end{align*}
\]

where \( d[X,X] \) is the bracket process or quadratic variation of \( X \).

**Corollary 2.5.** Let \( S \) be a FATGBM given by (3.4) and let \( f = \log(S) \), then

\[
\begin{align*}
    d\log S &= \frac{dS}{S} - \frac{1}{2} \sigma^2 d(T_t).
\end{align*}
\]

**Proof.** Follows from Itô’s Lemma, noting that \( d[S,S] = S^2 \sigma^2 d(T_t) \). \( \square \)

This allows for the empirical construction of \( \{T_t\} \) via a discrete approximation of \( d(T_t) \approx T_{t+\delta} - T_t \), for small \( \delta \).

In this section, we have outlined the properties of FATGBM. This has enabled us to understand the construction of the process which provides us with the tools required to proceed to the next section where we will investigate its uses as an option pricing model.
In Chapter 1, we introduced the theory that underlies derivative pricing. In particular, we showed that in order to define a market model that is consistent with the efficient market hypothesis, it is necessary for that model to admit an equivalent martingale measure for the discounted stock price process. The existence of such a measure ensures a market model free of arbitrage (see Chapter 1). This section introduces the market model proposed in Heyde (2005) for the purpose of option pricing under FATGBM. More specifically, we show that this model is consistent with the no-arbitrage pricing framework, however the market model raises some issues regarding hedging due to an incomplete market setting.

The risk-neutral price process proposed in Heyde (2005) is given by,

\[ S_t = S_0 e^{rt - \frac{1}{2}\sigma^2 T_t + \sigma W(T_t)}. \]

Heyde (2005) also proposes taking a conditional expectation of a modified Black-Scholes option pricing formula. This gives rise to the following European option pricing formula for a call option at time \( S \), resulting in a FATGBM call option price \( C_s(T_s) \):

\[ C_s(T_s) = S_0 \phi \left[ \frac{\log(S_0k^{-1}) + rs + \frac{1}{2}\sigma^2 T_s}{\sigma \sqrt{T_s}} \right] e^{-rs} \phi \left[ \frac{\log(S_0k^{-1}) + rs - \frac{1}{2}\sigma^2 T_s}{\sigma \sqrt{T_s}} \right], \]

and \( C_s(T_s) \) can then be calculated by conditioning on the activity time \( T_s \),

\[ E \{ E[C_s(T_s)|T_s]\}. \]
Heyde (2001) proposes that (4.2) introduces a range of appropriate prices for which the expectation (4.3) results in a price under the minimal equivalent martingale measure (corresponding to a zero risk premium).

Recall that \((S_t)_{0 \leq t \leq T}\) is a continuous martingale defined on the probability space \((\Omega, F, F_t, \mathbb{P})\). We will formally show that (4.1) is the price process under a probability measure \(Q \sim \mathbb{P}\), where \(\sim\) denotes “equivalent”. We will also show that this is not a minimal equivalent martingale measure and therefore that we cannot guarantee that the risk-neutral price process leads to trading strategies that result in a zero risk premium. We leave the investigation of trading strategies resulting in a zero risk premium for future research.

We now wish to show that (4.1) is the price process (3.4) under a measure \(Q \sim \mathbb{P}\), following Theorem 1.1 of Chapter 1. In order to do this, recall that two measures are equivalent to one another if they share the same null sets. Define an absolutely continuous equivalent measure, \(Q \sim \mathbb{P}\), by a random variable \(Z \in L^1(d\mathbb{P})\) such that

\[
\frac{dQ}{d\mathbb{P}} = Z, \tag{4.4}
\]

with \(\frac{dQ}{d\mathbb{P}} = (\frac{dQ}{d\mathbb{P}})^{-1}\) and \(E_\mathbb{P}\{Z\} = 1\), where \(E_\mathbb{P}\) denotes the expectation under the measure \(\mathbb{P}\). We can then set

\[
Z_t = E_\mathbb{P}\left\{\frac{dQ}{d\mathbb{P}} \mid F_t\right\}. \tag{4.5}
\]

The next theorem will help us find such a random variable, \(Z\).
Theorem 2.6. See, for example, Theorem 37, Protter (2005), p.84.

Doleans-Dade exponential. Let $X$ be a continuous semi-martingale, $X_0 = 0$. Then there exists a (unique) semi-martingale $Z$ that satisfies the equation $Z_t = 1 + \int_0^t Z_s dX_s$, given by,

$$Z_t = \mathcal{E}(X)_t = e^{\{X_t - \frac{1}{2}[X,X]_t\}}.$$

We now state the Girsanov-Meyer Theorem.

Theorem 2.7. Girsanov-Meyer Theorem. Let $\mathbb{P}$ and $\mathbb{Q}$ be equivalent. Let $X$ be a classical semi-martingale under $\mathbb{P}$ with decomposition $X = M + A$. Then $X$ is also a classical semi-martingale under $\mathbb{Q}$ and has a decomposition $X = L + C$, where

$$L_t = M_t - \int_0^t \frac{1}{Z_s} d[Z,M]_s,$$

is a $\mathbb{Q}$-local martingale, and $C = X - L$ is a $\mathbb{Q}$ Finite Variation (FV) process.

We can now use this theorem to find an equivalent martingale measure $\mathbb{Q}$ for a time changed Brownian motion.

Theorem 2.8. Let $W(T_t)$ be a time changed Brownian Motion on $(\Omega, \mathcal{F}, \mathcal{F}_t, \mathbb{P})$, such that $\|\frac{dT_s}{ds}\|_{L^\infty[0,t]} < \infty$ a.s. and let $\zeta \in L$ be bounded. Let

$$X_t = \int_0^t \zeta_s dT_s + W(T_t),$$

and define $\mathbb{Q}$ by $\frac{d\mathbb{Q}}{d\mathbb{P}} = e^{-\int_0^T \zeta_s dW(T_t) - \frac{1}{2} \int_0^T \zeta^2 d\tau}$, for some $T > 0$. Then under $\mathbb{Q}$, $X$ is a time changed Brownian Motion with no drift.
PROOF. Let \( \frac{dQ}{dP} = Z_T = e^{-\int_0^T \zeta_s dW(T_s) - \frac{1}{2} \int_0^T \zeta_s^2 dT_s} \). Then,

\[
E[|Z_T|] = e^{\frac{1}{2} \int_0^T \zeta_s^2 dT_s + \frac{1}{2} \int_0^T \zeta_s^2 dT_s} = 1,
\]

and \( E_{\mathbb{P}} \left\{ \frac{dQ}{d\mathbb{P}} \big| \mathcal{F}_t \right\} = Z_t \). Then \( Z_t \) is a uniformly integrable martingale, satisfying the equation

\[
Z_t = 1 - \int_0^t Z_s \zeta_s dW(T_s).
\]

By the Girsanov-Meyer Theorem, we have that,

\[
L_t = W(T_t) - \int_0^t 1 \frac{1}{Z_s} d[Z, W(T)]_s
\]

is a local martingale under \( Q \). By Theorem 29, Chapter 2 of Protter (2005),

\[
d[Z, W(T)]_s = -Z_s \zeta_s d[W(T_s), W(T_s)] = -Z_s \zeta_s d[T_s],
\]

and

\[
L_t = W(T_t) - \int_0^t -\frac{1}{Z_s} Z_s \zeta_s d[T_s]
\]

\[
= W(T_t) + \int_0^t \zeta_s dT_s
\]

\[
= X_t,
\]

thus \( X \) is a \( Q \)-local martingale. Since \( \int_0^t \zeta_s ds \) is a continuous Finite Variation (FV) process, we have that \( X \) is a continuous local martingale with \( [X, X] = [W(T), W(T)] = T_t \), and so by Theorem 2.3 (Dambis/Dubins-Schwarz), \( X \) must be a time changed Brownian Motion with time change \( T_t \).

We now apply this change of measure to show that under \( Q \), the process given by (3.4) is just the risk-neutral price process given by (4.1).
COROLLARY 2.9. Let $S_t$ be a FATGBM given by (3.4) and define $\mathbb{Q}$ by $\frac{d\mathbb{Q}}{d\mathbb{P}} = e^{-\int_0^T \zeta dW(T_s) - \frac{1}{2} \int_0^T \zeta^2 ds}$, for some $T > 0$, where

$$\zeta_s = \frac{1}{\sigma} \left( \frac{(\mu - r) + \frac{1}{2} \sigma^2 (T_s')}{\frac{dT_s}{ds}} \right).$$

Then under $\mathbb{Q}$, the price process given by (3.4) is the price process given by (4.1).

PROOF. By Theorem 2.8, $L_t$ is a martingale under $\mathbb{Q}$, where,

$$L_t = \sigma W(T_t) + \int_0^t \sigma \zeta_s dT_s.$$ 

Since $T_t$ is a continuous FV process whose derivative is locally bounded in $(0, t)$ for all $t$, we have,

$$\int_0^t \zeta_s dT_s = \int_0^t \zeta_s \frac{dT_s}{ds} ds = \frac{1}{\sigma} \int_0^t \left[ (\mu - r) + \frac{1}{2} \sigma^2 \frac{dT_s}{ds} \right] ds$$

$$= \frac{1}{\sigma} \left[ (\mu - r)t + \frac{1}{2} \sigma^2 T_t \right].$$

Therefore,

$$L_t = \sigma W(T_t) + (\mu - r)t + \frac{1}{2} \sigma^2 T_t,$$

and, by the Girsanov-Meyer Theorem,

$$X_t = \mu t + \sigma W(T_t) = L_t + C_t,$$

where $C_t$ is the FV process given by,

$$C_t = rt - \frac{1}{2} \sigma^2 T_t.$$

Thus, $X_t$ is a time changed Brownian Motion with drift $rt - \frac{1}{2} \sigma^2 T_t$ under $\mathbb{Q}$.
Thus far we have shown that (3.4) is (4.1) under an equivalent measure $Q$, however this measure $Q$ is not unique since $[W(T_t), W(T_t)] = T_t$ is a FV process and not a constant. In the next section, we will address some of the issues arising from this incompleteness.

\[ \square \]

5. Hedging and FATGBM

In Chapter 1 we stated that all contingent claims are not attainable (that is, they can neither be replicated nor hedged) in an incomplete market setting (Harrison and Pliska, 1981). This brings us to the issues surrounding option pricing in an incomplete market. In particular, this section highlights the mathematical challenges that need to be faced in order to achieve an optimal hedging rule under the FATGBM model. We saw in Chapter 1 that the central discussion point surrounding any dynamic trading strategy, which replicates the value of any contingent claim, is a martingale representation. In particular, any claim $H$ under the Geometric Brownian Motion model, admits an Itô representation of the form,

\[ H = H_0 + \int_0^T \xi_s^H dX_s. \tag{5.1} \]

This representation provides us with a dynamic trading strategy $\xi$, which costs us exactly $H_0$ to set up with no additional amount required or gained over $[0, T]$. This trading strategy replicates the value of the derivative $H$ with no risk. This tells us that any discounted claim $C_T$ being replicated by the strategy $H$ will
always equal $H$, and thus $C_t = C_T$. However, in the case of general square-integrable martingales, the Kunita-Watanabe decomposition exists instead:

\[
H = H_0 + \int_0^T \xi_s^H \, dX_s + L_T^H,
\]

with $H_0 \in L^2(\Omega, \mathcal{F}_0, \mathbb{P})$, where

\[
L^H = (L_t^H)_{0 \leq t \leq T}
\]

is a square-integrable martingale orthogonal to $X$. Such a decomposition allows us to find a mean-self-financing strategy in the sense that,

\[
E[C_T - C_t | \mathcal{F}_t] = 0, \quad 0 \leq t \leq T,
\]

and

\[
E[(C_T - C_t)^2 | \mathcal{F}_t]
\]

is minimised over all $t \in [0, T]$.

This brings us to the concept of a minimal equivalent martingale measure, which was introduced in Föllmer and Schweizer (1991) to obtain hedging strategies which are optimal for risk-minimisation. The question now arises as to whether the measure $\mathbb{Q}$ obtained in Corollary 2.9 is a minimal martingale measure resulting in the existence of an optimal hedging strategy. Schweizer (1990) describes the idea of the minimal equivalent martingale measure $\mathbb{P} \sim \mathbb{P}$ as being the choice of measure $\mathbb{P}$ which preserves the structure of $\mathbb{P}$ as far as possible under the constraint that a process $X$ becomes a martingale under $\mathbb{P}$. In some sense, the
minimal martingale measure is the measure which minimises some distance between \( \hat{P} \) and \( P \). Applying this idea, it seems reasonable that the FATGBM risk-neutral measure \( Q \) is in fact the minimal equivalent martingale measure leading to the existence of a mean-variance optimal hedging strategy. However, we shall now define the minimal equivalent martingale measure and state a theorem from Schweitzer (1990) which highlights that under FATGBM we cannot ensure its existence. Hence, further research beyond this thesis is required into this area to investigate whether \( Q \) really is an optimal equivalent martingale measure in some sense.

**Definition 2.4.** A martingale measure \( \hat{P} \sim P \) will be called minimal if

(5.6) \[ \hat{P} = P, \text{ on } F_0, \]

and if any square-integrable \( P \)-martingale which is orthogonal to \( M \) under \( P \) remains a martingale under \( \hat{P} \):

(5.7) \[ L \in \mathcal{M}^2 \text{ and } [L, M] = 0 \Rightarrow L \text{ is a martingale under } \hat{P}. \]

To ensure its existence, the following theorem holds.

**Theorem 2.10. (Schweitzer, 1990)** \( \hat{P} \) exists if and only if

(5.8) \[ \hat{G}_t = e^{-\int_0^t \xi_s dM_s - \frac{1}{2} \int_0^t \xi_s^2 d|x,x|_s}, 0 \leq t \leq T, \]

is a square-integrable martingale under \( P \); in that case, \( \hat{P} \) is given by \( \frac{d\hat{P}}{dP} = \hat{G}_T. \)
In the case of $Q$ being defined by $\frac{dQ}{d\mathbb{P}}$ in Corollary 2.9, which results in the risk-neutral price process for FATGBM, it is not the case that $\frac{dQ}{d\mathbb{P}}$ is square-integrable. This follows by taking a conditional expectation of $(\frac{dQ}{d\mathbb{P}})^2$:

\[
\left(\frac{dQ}{d\mathbb{P}}\right)^2 = E\left[E\left[e^{2(-\int_0^T \zeta_s dW(T_s) - \frac{1}{2} \int_0^T \zeta_s^2 dT_s)} | \{T\}\right] \right]
\]

\[
= E\left[E\left[e^{2(\zeta W(T_T) - \frac{1}{2} \zeta^2 T_T)} | \{T\}\right] \right]
\]

\[
= E\left[E\left[e^{2(\zeta \sqrt{T_T} \phi - \frac{1}{2} \zeta^2 T_T)} | \{T\}\right] \right]
\]

\[
= E\left[E\left[e^{\frac{3}{2} \zeta^2 T_T - \zeta^2 T_T} | \{T\}\right] \right]
\]

\[
= E\left[E[e^{\zeta^2 T_T} | \{T\}\right] \right] = \infty,
\]

since $\{T\}$ is a strictly increasing process with a heavy tail marginal distribution.

The concluding result so far is that we do not know whether $Q$ is a minimal martingale measure and since this investigation is beyond the scope of this thesis, we shall now focus our attention on whether we can specify a martingale representation for FATGBM, which may allow us to replicate a certain type of contingent claim. The following theorem shows that FATGBM does not admit a representation of the form (5.1), but in showing this, we can identify a type of derivative that we can hedge under FATGBM.
THEOREM 2.11. Let $W(t)$ be a FATGBM as described above and let $M = (M(t))_{t > 0}$ be a martingale with respect to the filtration, $\mathcal{F}_t$, generated by FAT-GBM. Then $M(t)$ does not admit an integral representation of the form

\begin{equation}
M(0) + \int_0^t H(s)dW(T_s), t \geq 0.
\end{equation}

PROOF. Let $\mathcal{G}_T$ be the $\sigma$-algebra generated by the activity time process $T_t$ at some time $T > 0$, and let $\mathcal{G} = \mathcal{F}_s \vee \mathcal{G}_T$. Define $T^{-1}(y)$ to be the clock time, $s$, such that $T_s = y$, $0 < y \leq T_T$. Then, there exists an $\mathcal{F}_y$-measurable function, $H(T^{-1}(y))$, such that

\begin{equation}
M(T) = M(0) + \int_0^{T_T} H(T^{-1}(y))dW(y),
\end{equation}

and there exists a unique $H(s)$ which is $\mathcal{G}$ measurable such that

\begin{equation}
M(T) = M(0) + \int_0^T H(s)dW(T_s), T \geq 0.
\end{equation}

But $H(s)$ is not $\mathcal{F}_s$-measurable. 

Equation (5.10) reveals that we are able to hedge claims that are contingent on a certain level of volatility being reached. Since this representation holds for the discounted price process, we will have to place an additional constraint on interest rates being zero. This will allow us to specify the trading strategy $H(T^{-1}(y))$ via the Clarke-Ocone Formula which is used to specify trading strategies under the Black-Scholes model. For example, the classic delta hedge for a European Call option (with $r = 0$) which is exercisable after a certain amount of volatility has
5. HEDGING AND FATGBM

been reached, is given by,

\[
\Delta = \frac{\log(S_0 k^{-1}) + \frac{1}{2} \sigma^2 T_s}{\sigma \sqrt{T_s}}.
\]

One common example of such a claim was introduced in Bick (1995), then later Société Générale Corporate and Investment Banking coined the name Timer Option (Vovk, 2011; Sawyer, 2007).

We will close this section with a short discussion on how we intend to proceed with respect to replication strategies under FATGBM. In addition to investigating whether \( Q \) is the minimal equivalent martingale measure, we also wish to proceed with an investigation of whether the market can be completed under the FATGBM model by introducing liquid European Call Options with a range of strike prices and maturity times, into the market model, as an additional risky asset. The purpose of this strategy will be to complete the market by matching the number of risky assets available with sources of risk. Such an approach has been taken by Carr et al. (2001) where they specify a martingale representation for a pure jump Lévy process which allows them to construct a dynamic trading strategy in a risky asset and European options of all strikes, on that asset. They then apply these results to the Variance Gamma model, which is also a subordinated Brownian Motion, but with independent increments. We believe that a similar approach may follow through for FATGBM as it is not evident thus far that a Markovian structure is necessary to formulate such a martingale representation. It is also the case that FATGBM would be a Lévy Process if it
had independent increments and it is the case that the Activity Time structure could be defined as a point process.

6. Conclusion

This section provided an introduction to long memory processes and the need for them in finance. We introduced the Fractal Activity Time Geometric Brownian Motion (FATGBM) model and highlighted its key features which impact how we solve the problem of pricing derivatives under FATGBM. We introduced the risk-neutral price process for FATGBM and proved that this process is the FATGBM process under an equivalent measure \( \mathbb{Q} \). We then showed that this measure is not necessarily the minimal equivalent martingale measure and we discussed some of the issues surrounding pricing in an incomplete market setting. We showed that we can hedge a certain type of contingent claim when interest rates are zero, under FATGBM. Such a contingent claim is known as a Timer option. We finally closed with a brief discussion on how we intend to proceed with regards to hedging under FATGBM.
CHAPTER 3

Discrete Approximation Schemes for FATGBM

1. Introduction

In this chapter, we present a discrete approximation scheme to FATGBM, a subordinated Brownian Motion model with Long Range Dependence (LRD), introduced in Chapter 2. More generally, we introduce a generalised binomial model for pricing options in a market where stock prices follow a subordinator model similar to that of FATGBM. Option pricing for related models, such as stochastic volatility models, are performed in many ways, depending on the complexity of the underlying asset price process. Sometimes stochastic volatility models have a closed form solution, for example for plain vanilla options (Heston, 1993), but often they require numerical techniques to approximate their theoretical prices. Finite difference methods have been used to solve partial integro-differential equations (PIDEs) for contingent claims arising from a stochastic volatility framework (Duffy, 2006), Fourier inversion techniques present a fast solution to approximating the derivative price (Carr and Madan, 1999), and Monte Carlo based approximations have been used to numerically integrate the governing stochastic differential equation (Glasserman, 2004). Another approach is the tree-based approach such as that described in Maller et al. (2005). Most of these methods are used mainly for pricing options in a market where stock price returns do not display long range dependence. Here we outline a discrete approximation scheme
which results in a possible approximation of the theoretical derivative price under FATGBM (subject to the restrictions outlined in Chapter 2). The scheme also allows for the calculation of many path-dependent options, such as barrier options. In addition to constructing this discrete approximation scheme, we derive an upper bound for the American Put. It should be noted that this is a possible method for pricing many path-dependent options where the underlying process is a subordinated Brownian Motion, and is not restricted to the Markovian class of processes.

The chapter begins by reintroducing the FATGBM model and some of its specialised features. Following that, we develop the discrete approximation scheme which is based on the binomial approximation to the Black-Scholes model. We prove convergence of the discrete time process to the continuous one, namely FATGBM, by first specifying a deterministic model for the subordinator and then extending it to a strictly increasing stochastic process. This setup allows the calculation of the option price using conditional expectations. We then present the method of Heyde and Leonenko (2005) for pricing a European option in the no-arbitrage framework. Further, we show that the tree-based approach can be used to derive the upper bound for the American option. Lastly, we summarise the chapter, leaving all remaining proofs to be found in the Appendix.
2. FATGBM

The Fractal Activity Time Geometric Brownian Motion (FATGBM) price process introduced in Heyde (1999) and Heyde and Liu (2001) is defined to be,

\[ S_t = S_0 e^{rt + \sigma W(Y_t)} , \]

where \( W(t) \) is a standard Brownian Motion and \( Y_t \) is a stochastic process independent of \( W() \). In particular, the \( Y_t - t \) process is an \( H \) self-similar fractal. Because \( H \) is chosen to lie between 0.5 and 1, FATGBM incorporates long range dependence, a feature that is observed in stock price returns data. Moreover, \( Y_t \) is empirically constructed so that \( W(Y_t) \) models the stylised features of asset returns data such as heavy tails. Generally speaking, \( Y_t \) is said to represent "activity time" or trading volume, which differs from normal clock time. More importantly, \( W(Y_t) \) is a martingale and therefore is consistent with the no-arbitrage pricing method.

3. The Discrete Approximation Scheme

3.1. Definitions and Setup. We have a Brownian Motion \( W(t) \) and an increasing process \( Y_t \), in \( C[0, T] \), both defined on the probability space \( \{ \Omega, \mathcal{F}, \mathbb{P} \} \).

First we introduce the risk-neutral price process for FATGBM. This is given by

\[ S_t^* = S_0 e^{rt - \frac{1}{2} \sigma^2 Y_t + \sigma W(Y_t)} , \]
where \( r > 0 \) is the risk-free rate and \( \sigma > 0 \) is a parameter.

Let

\[
X_t = rt - \frac{1}{2} \sigma^2 Y_t + \sigma W(Y_t).
\]

We would like to construct a discrete approximation \( X^n \), say, to \( X \) using an \( n \)-step binomial tree. To achieve this, we mimic the usual binomial approximation to the Black-Scholes model. The central idea is to replace the real time increments, \( \frac{T}{n} \), by the increments of the process \( Y_t \) over that time interval.

First consider an increasing deterministic function \( y_t \in C[0,T] \), with \( y_0 = 0 \). Let,

\[
\Delta y_{k,n} = \frac{y_{k+1}}{n} - \frac{y_k}{n}, \quad 1 \leq k \leq n.
\]

Further, assume

\[
\lim_{n \to \infty} \Delta y_{k,n} = 0 \quad \text{for each} \quad k = 1, 2, 3, \ldots
\]
and

\begin{equation}
\sigma \sqrt{\Delta y_{k,n}} \geq \frac{rT}{n}.
\end{equation}

Now define \( k = 1, 2, \ldots, n \), to be an independent sequence of random variables, independent of \( W \) and \( Y \), uniformly distributed on \([0,1]\). Then let

\begin{equation}
U_{k,y}^n := \begin{cases} 
\sigma \sqrt{\Delta y_{k,n}}, & \text{if } B_k^n < p_{k,y}^n \\
-\sigma \sqrt{\Delta y_{k,n}}, & \text{if } B_k^n > p_{k,y}^n
\end{cases},
\end{equation}

where

\[ p_{k,y}^n = \frac{e^{ct} - e^{-\sigma \sqrt{\Delta y_{k,n}}}}{e^{\sigma \sqrt{\Delta y_{k,n}}} - e^{-\sigma \sqrt{\Delta y_{k,n}}}}. \]

Note that, because of (3.3), \( p_{k,y}^n \in (0,1) \).

Let

\begin{equation}
X_t^n(y_t) = \sum_{k=1}^{[\frac{n}{T}]} U_{k,y}^n,
\end{equation}
and define

$$X_t(y_t) = rt - \frac{1}{2} \sigma^2 y_t + \sigma W(y_t).$$  

Returning to the original subordinator process $Y_t$, an increasing stochastic process in $C[0, T]$ with $Y_0 = 0$, define $\Delta Y_{k,n}$ as

$$\Delta Y_{k,n} = Y_{\frac{kT}{n}} - Y_{\frac{(k-1)T}{n}}, \quad 1 \leq k \leq n.$$  

Assume the following properties hold:

$$\lim_{n \to \infty} \Delta Y_{k,n} \xrightarrow{P} 0 \quad \text{for each} \quad k = 1, 2, 3, ...$$

$$\lim_{n \to \infty} P\left( \sigma \sqrt{\Delta Y_{k,n}} - \frac{rT}{n} \geq a \right) = 1 \quad \text{for some} \quad a > 0.$$

Finally we define

$$X_t^n = X_t^n(Y_t).$$
3.2. **Weak Convergence.** We would like to show the weak convergence of \( X^n \to X \) as \( n \to \infty \) in the space \( D \) of functions on \([0,T]\), endowed with the Skorohod topology (see Billingsley, 1968). This will be denoted by \( X^n \overset{\mathcal{L}}{\to} X \).

**Theorem 3.1.** For the triangular array scheme described by (3.12), we have

\[
X^n \overset{\mathcal{L}}{\to} X.
\]

**Remark 3.1.** We prove Theorem 3.1 by conditioning on the subordinator, the entire \( Y_T \) process. The proof can be shown using the Lindeberg-Feller theorem for row-wise triangular arrays and demonstrating that the \( X^n \) form a tight sequence. However we can avoid these calculations by applying Theorem 2.29 from Jacod and Shiryaev (1987, pp. 425-426). We use this theorem to prove the following proposition.

**Proposition 3.1.** Consider a \( d \)-dimensional semimartingale triangular array scheme with \( X^n_t = X^n_{t|y_t} \) as defined in (3.7), where \( y_t \) satisfies (3.4) and (3.5).

If the following conditions hold

\[
(3.13) \quad \sup_{s \leq t} \left[ \sum_{k=1}^{\lfloor ns/T \rfloor} \mathbb{E}[U^n_{k,y}] - \left( rs - \frac{1}{2} \sigma^2 y_s \right) \right] \to 0 \quad \text{for all } t \geq 0
\]

\[
(3.14) \quad \sum_{k=1}^{\lfloor n t/T \rfloor} \{ \mathbb{E}[\{U^n_{k,y}\}^2] - (\mathbb{E}[U^n_{k,y}])^2 \} \to \sigma^2 y_t \quad \text{for all } t \in [0, T]
\]

\[
(3.15) \quad \sum_{k=1}^{\lfloor n t/T \rfloor} \left| \mathbb{E}[1_{\{U^n_{k,y} \geq \frac{1}{a}\}}] \right| \to 0 \quad \text{for all } t \in [0, T], \quad a > 0.
\]
Then \( X^n(y_t) \xrightarrow{c} X(y_t) \)

**Proof.** See Appendix for the proof of (3.13) and (3.14).

Proof of (3.15). There exists an \( N \) such that for every \( n > N \),

\[
\max_{\{1 \leq k \leq \left[\frac{n}{t}\right]\}} \sigma \sqrt{\Delta y_{k,n}} < a \quad \text{(see Appendix)}.
\]

Therefore, for \( n > N \),

\[
E[1_{\{|u^n_{ik,t}| \geq \frac{a}{\sigma}\}}] = 0 \quad \forall \quad 1 \leq k \leq n,
\]

and (3.15) follows.

The proof is completed by application of Theorem 2.29 of Jacod and Shiryaev (1987, pp. 425-426). To do this, note that the limit process \( X(y_t) \) has independent increments and characteristics \( B = rt - C \frac{1}{2} \sigma^2 yt, C = \sigma^2 yt, \nu = 0 \), in the notation of Jacod and Shiryaev.

To complete the proof of Theorem 3.1, we must show that (3.7) and (3.8) converge when \( y_t \) is a process.

Consider the process \( Y_t \) defined above with increments as in (3.9) and satisfying properties (3.10) and (3.11).
The process \( Y_t \) is defined on \( \Omega \) and generates a \( \sigma \)-field \( \mathcal{F}_Y \). If \( f \) is a bounded function in \( D[0, T] \), continuous on the Skorohod topology then, by the Dominated Convergence Theorem and Proposition 3.1,

\[
\int f(X^n)dP = \int E[f(X^n) \mid \mathcal{F}_Y]dP \\
\rightarrow \int E[f(X) \mid \mathcal{F}_Y]dP \\
= \int f(X)dP
\]

and \( X^n \overset{D}{\to} X \).

\[\square\]

4. The European Option

Recall that under FATGBM

\[ S_t = S_0e^{\mu t + \sigma W(Y_t)}, \]

where the choice of \( Y_t \) can be specified so that the stock price returns process has a \( t \)-distribution with \( \nu \) degrees of freedom (Heyde and Leonenko, 2005). Under an equivalent measure \( \mathbb{Q} \), the price process \( S_t^* \) is given by

\[ S_t^* = S_0e^{rt - \frac{1}{2} \sigma^2 Y_t + \sigma W(Y_t)}. \]
This, of course, complies with the no arbitrage pricing requirement as the discounted price process under $Q$ is a martingale. Note that $Q$ is not unique, however. Heyde and Leonenko (2005) price the European call option as follows:

\begin{align}
\text{call price} &= E_Q[e^{-rt}(S_t^* - K)^+] \\
&= E_Q[E_Q[e^{-rt}(S_t^* - K)^+ \mid \{Y_t\}] \\
&= E_Q[BS(\text{call})], \quad \text{where BS is the Black-Scholes price of the option}
\end{align}

In this case, the expectation of the Black-Scholes price is the same under the real world measure as it is under $Q$. For the purposes of calculating the European option price, $Y_t$ is taken to be

\begin{equation}
Y_t \overset{d}{=} t + t^H(Y_1 - 1),
\end{equation}

where $Y_1$ has an inverse Gamma distribution and the Hurst index takes on a value between 0.7 and 0.9.

Using the methods of the previous section we can set up a discrete binomial scheme such that the discretely calculated option price converges to the continuous time version.
5. The American Put

Unlike the European case, path-dependent options such as the American Put require calculation of an optimal stopping time. This section proposes a technique for evaluating these options.

If we knew the entire activity time path history of $Y_t$ we could set up the triangular array scheme as in Section 3 and calculate the option price. In practice we do not know the entire path of $Y_t$, however if we specify a process as follows,

\begin{equation}
Y_t = f(Y, t),
\end{equation}

then the path of $Y_t$ only depends on a single random variable $Y$ and $Y_t$ is $\mathcal{F}_s$-measurable for $s \leq t$. Using a subordinator process which has the same distribution as that in (4.3) and having specified $Y_t$ as in (5.1), we can condition on the random variable $Y$ and calculate the value of a path-dependent option using the set up described in Section 3. Now we would like to briefly turn to the question of pricing the American Put under this tree construction. We will state the following proposition, which addresses this question, and then summarise our findings thus far, before returning to a proof of Proposition 3.2. The reason for the suspense is because we would like to pause to discuss some of the shortcomings of the model thus far.

**Proposition 3.2.** For $Y_t$ given by (5.1) and $X_t, X^n_t$ given by (3.2), (3.13), respectively, define
3. DISCRETE APPROXIMATION SCHEMES FOR FATGBM

\[ P^n_t = \text{ess sup}_{\tau \in [0,T]} E_Q[e^{-r(\tau-t)}g(X^n_\tau) \mid \mathcal{F}^n_t], \quad \text{for} \quad 0 \leq t \leq T \]

and

\[ P_t = \text{ess sup}_{\tau \in [0,T]} E_Q[e^{-r(\tau-t)}g(X_\tau) \mid \mathcal{F}_t], \quad \text{for} \quad 0 \leq t \leq T, \]

where \( \tau \) are stopping times on \([0,T]\).

Then, under appropriate conditions,

\[ P^n_t \to P_t. \]

6. Summary

In a market where the stock price returns process is modelled by a type of subordinated Brownian Motion, options can be priced by constructing a tree similar to that used to approximate the Black-Scholes model. A major and practically important difference between the Cox Ross Rubenstein (CRR) Binomial Tree model used to approximate the Black-Scholes price and the model presented thus far is that the tree method used to approximate FATGBM is non-recombining. This feature of the method presents some serious computational issues. We next propose an alternative, recombining tree which overcomes all of these. By following the process above, we prove that the recombining tree still converges to FATGBM. We then close this chapter with a proof of Proposition 3.2.
7. Discrete Approximation Scheme 2

7.1. Definitions and Setup. We have a Brownian Motion $W(t)$ and an increasing process $Y_t$, in $C[0,T]$, both defined on the probability space $\{\Omega, \mathcal{F}^W \vee \mathcal{F}^Y, \mathbb{P}\}$.

Recall (3.1) and once again let

\begin{equation}
X_t = rt - \frac{1}{2}\sigma^2 Y_t + \sigma W(Y_t).
\end{equation}

We would like to construct a discrete approximation $X^n$, say, to $X$ using a random-step recombining binomial tree. To do this, we mimic the Cox Ross Rubinstein Binomial Tree but this time the idea is to replace the real time increments of $\frac{1}{n}$ by equidistant activity time increments each equal to $\frac{1}{n}$. Note that these increments correspond to different real time increments.

Define $U^n_k$ to be independent random variables such that

\begin{equation}
P \left[ U^n_k = \frac{\sigma}{\sqrt{n}} - \ln \left( \frac{e^{\frac{\sigma^2}{2}} + e^{-\frac{\sigma^2}{2}}}{2} \right) \right] = P \left[ U^n_k = -\frac{\sigma}{\sqrt{n}} - \ln \left( \frac{e^{\frac{\sigma^2}{2}} + e^{-\frac{\sigma^2}{2}}}{2} \right) \right] = \frac{1}{2}.
\end{equation}

Let
64 3. DISCRETE APPROXIMATION SCHEMES FOR FATGBM

\[(7.3) \quad X_t^n = X_t^n(Y_t) = rt + \sum_{k=1}^{[nY_t]} U_k^n. \]

Now, similar to the argument for the first discrete approximation scheme, consider a strictly increasing deterministic function \(y_t \in C[0,T]\), with \(y_0 = 0\) and define

\[(7.4) \quad X_t^n(y_t) = rt + \sum_{k=1}^{[nY_t]} U_k^n. \]

**7.2. Weak Convergence.** We would like to show the weak convergence of \(X^n \to X\) as \(n \to \infty\) in the space \(D\) of functions on \([0,T]\), endowed with the Skorohod topology; that is, that \(X^n \overset{\mathcal{D}}{\to} X\).

**Theorem 3.2.** For the triangular array scheme described by (7.4), we have

\[X^n \overset{\mathcal{D}}{\to} X. \]

**Remark 3.2.** We prove Theorem 3.2 by once again conditioning on the subordinator and applying Theorem 2.4 from Jacod and Shiryaev (1987, pp. 419-420).

**Proposition 3.3.** Consider a d-dimensional semimartingale triangular array scheme with \(X_t^n = X_t^n(y_t)\) as defined in (7.4).
If the following conditions hold

\[(7.5)\quad r s + \sum_{k=1}^{[ny_t]} \mathbb{E}[U_k^n] \xrightarrow{P} r s - \frac{1}{2}\sigma^2 y_t \quad \text{for all } t \in [0, T] \]

\[(7.6)\quad \sum_{k=1}^{[ny_t]} \text{Var}[(U_k^n)] \xrightarrow{P} \sigma^2 y_t \quad \text{for all } t \in [0, T] \]

\[(7.7)\quad \sum_{k=1}^{[ny_t]} \mathbb{E}_1[(U_k^n)^2] \xrightarrow{P} 0 \quad \text{for all } t \in [0, T], \quad a > 0. \]

Then, \( X^n(y_t) \xrightarrow{\mathcal{L}} X(y_t) \)

**Proof.** See Appendix for proof of (7.5), (7.6) and (7.7).

The proof is completed by application of Theorem 2.4 of Jacod and Shiryaev (1987, pp. 419-420). Specifically, as before, note that the limit process \( X(y_t) \) has independent increments and characteristics \( B = rt - \frac{1}{2}\sigma^2 y_t, \quad \tilde{C} = \sigma^2 y_t, \quad \nu = 0, \) in the notation of Jacod and Shiryaev.

To complete the proof of Theorem 3.2, we must show that (7.4) converges when \( y_t \) is a process. This fact follows from the Dominated Convergence Theorem and Proposition 3.3.
We can construct the triangular array scheme by summing over a fixed number \( i \), say, of the Binomial random variables, \( U^n_k \). Let,

\[
t_i = \inf \{ nY \geq i \}.
\]

Then,

\[
X^n_{t_i} = r t_i + \sum_{k=1}^{i} U^n_k = \sum_{k=1}^{i} U^n_k + r(t_k - t_{k-1}).
\]

To calculate the option price as a conditional expectation, we can simulate a realisation of the \( Y_T \) process, construct the tree for the discounted stock price process, \( S_t e^{-rt} \), using the realised value \( y_t \), and calculate a price.

Figure 1 is a visual representation of the time increments associated with each step in the Cox Ross Rubensteins Binomial Tree, while Figure 2 is a visual representation of the time increments for the Binomial Tree scheme introduced in Section 3. These diagrams depict the differences between the two tree schemes. The first shows that the clock times associated with each activity time increment are all equal, whereas the second shows that equidistant clock times do not necessarily equate to equidistant activity time increments. Figure 3 is a visual representation of the final, recombining tree. Here, you may observe that the clock times are no longer equidistant, but the activity time increments are. This feature allows the tree to recombine.
When Activity Time equals Clock Time for the discrete approximation scheme 1:

Black-Scholes Model

\[ P(U^n_k = u) = p, \quad P(U^n_k = -u) = 1 - p, \]

where \( u = \sigma \sqrt{\Delta t} \),

\[ p = \frac{e^u - e^{-\sigma \sqrt{\Delta t}}}{e^{\sigma \sqrt{\Delta t}} - e^{-\sigma \sqrt{\Delta t}}} \]

**Figure 1.** Time Increments for Cox Ross Rubinstein Binomial Tree

8. The American Option

Now that we have achieved a computationally tractable approximation scheme, we may return to Proposition 3.2.

Using the same conditioning arguments as above, we can specify an entire activity time path for \( Y_T \) and set up a tree. Working backwards through the tree, we find a solution to the optimal stopping problem for that specified path. We now recall Proposition 3.2:
For $Y_t$ defined above and $X_t, X^n_t$ given by (7.1), (7.3), respectively, define

$$P^n_t = \sup_{\tau \in \mathcal{T}_{t,T}} E_Q[ e^{-r(\tau-t)} g(X^n_{\tau}) | \mathcal{F}^W_t \cup \mathcal{F}^Y_T ], \text{ for } 0 \leq t \leq T$$

and

$$P_t = \sup_{\tau \in \mathcal{T}_{t,T}} E_Q[ e^{-r(\tau-t)} g(X_{\tau}) | \mathcal{F}^W_t \cup \mathcal{F}^Y_T ], \text{ for } 0 \leq t \leq T,$$

where $\mathcal{T}_{t,T}$ are the set of all stopping times on $[t,T]$.

Then, under appropriate conditions,

$$P^n_t \to P_t.$$
Now we will show that if Proposition 3.2 holds, we can find an upper bound for the American option by constructing the tree for each realisation \( y_t \) of the activity time process.

**Proposition 3.4.** Under the non-unique measure \( Q \), \( P_t \) is greater than or equal to the American option price.

**Proof.** Define \( \tau^*_y := \inf\{t \geq 0 : Z(y_t) = X(y_t)\} \), where \( Z(y_t) \) is the Snell envelope of \( X(y_t) \).

The stopping time \( \tau^*_y \) solves the optimal stopping problem for \( X(y_t) \) and is

\[
P(U^n_k - u - c_a) = \frac{1}{2}, \quad P(U^n_k - u - c_a) = \frac{1}{2},
\]

where \( u = \frac{\sigma}{\sqrt{n}} \), \( c_a = \ln \left( \frac{e^{\frac{\sigma^2}{2}} + e^{-\frac{\sigma^2}{2}}}{2} \right) \).
adapted to the filtration $\mathcal{F}_t^W \lor \mathcal{F}_T^Y$. Hence,

$$E[e^{-r(\tau^*_T - t)}g(X_{\tau^*_T}) \mid \mathcal{F}_t^W \lor \mathcal{F}_T^Y] = \sup_{\tau \in \mathcal{T}_{t,T}} E_{Q}[e^{-r(\tau - t)}g(X_{\tau}) \mid \mathcal{F}_t^W \lor \mathcal{F}_T^Y].$$

The American option price is given by,

$$\sup_{\tau \in \mathcal{T}_{t,T}^Y} E_{Q}[e^{-r(\tau - t)}g(X_{\tau}) \mid \mathcal{F}_t^W \lor \mathcal{F}_T^Y],$$

where $\mathcal{T}_{t,T}^Y$ is the set of all stopping times adapted to the filtration $\mathcal{F}_t^W \lor \mathcal{F}_T^Y$.

Let $\tau^*$ be the stopping time that solves the optimal stopping problem for $X(Y_t)$.

Since $\mathcal{F}_t^W \lor \mathcal{F}_T^Y \subset \mathcal{F}_t^W \lor \mathcal{F}_T^Y$,

$$\sup_{\tau \in \mathcal{T}_{t,T}} E_{Q}[e^{-r(\tau - t)}g(X_{\tau}) \mid \mathcal{F}_t^W \lor \mathcal{F}_T^Y] \geq \sup_{\tau \in \mathcal{T}_{t,T}^Y} E_{Q}[e^{-r(\tau - t)}g(X_{\tau}) \mid \mathcal{F}_t^W \lor \mathcal{F}_T^Y]$$

and

$$E[e^{-r\tau^*_T}g(X_{\tau^*_T}) \mid \mathcal{F}_t^W \lor \mathcal{F}_T^Y] \geq E[e^{-r\tau^*_T}g(X_{\tau^*_T}) \mid \mathcal{F}_t^W \lor \mathcal{F}_T^Y].$$

\[ \square \]

9. Conclusion

In this chapter, we have constructed two discrete approximation schemes for general subordinated Brownian Motion processes, one which results in a non-recombining random step tree and one which results in a recombining random step tree. Using each of these trees, we are able to derive an upper bound for the
American Put price, however the latter tree gives rise to a computationally sound approximation. In particular, the tree can be used for subordinated Brownian Motion processes which exhibit long range dependence, hence we are able to calculate an upper bound and use it to price American options for FATGBM. Even more importantly, we are able to use these trees to price many path-dependent options, such as Barrier options, which do not require the solution to an optimal stopping problem.

10. Appendix

We prove here (3.14) and (3.15).

**Lemma 3.1.**

\[
\lim_{n \to \infty} E[X_t^n(y_t)] = rt - \frac{1}{2} \sigma^2 y_t.
\]

**Proof.** Since the \( U_k^n \) take only two values, we can calculate

\[
E[X_t^n(y_t)] = \sum_{k=1}^{\left\lceil \frac{y_t}{\Delta} \right\rceil} E[U_k^n]
\]

\[
= (\sigma \sqrt{\Delta y_{k,n}} p_{k,y}^n) - (\sigma \sqrt{\Delta y_{k,n}})(1 - p_{k,y}^n)
\]

(10.1)\[
= (\sigma \sqrt{\Delta y_{k,n}})(2p_{k,y}^n - 1).
\]

Using a Taylor series expansion, we have
Collecting the small order terms into \( A_{n,k} \) and \( B_{n,k} \), we obtain

\[
\begin{align*}
\varphi^n_{k,y} &= \frac{\left[1 + \frac{rT}{n} + \frac{1}{2!} \left(\frac{rT}{n}\right)^2 + \frac{1}{3!} \left(\frac{rT}{n}\right)^3 + \ldots\right] - \left[1 - \sigma \sqrt{\Delta y_{k,n}} + \frac{1}{2!} \left(\sigma \sqrt{\Delta y_{k,n}}\right)^2 + \ldots\right]}{\left[1 + \sigma \sqrt{\Delta y_{k,n}} + \frac{1}{2!} \left(\sigma \sqrt{\Delta y_{k,n}}\right)^2 + \ldots\right] - \left[1 - \sigma \sqrt{\Delta y_{k,n}} + \frac{1}{2!} \left(\sigma \sqrt{\Delta y_{k,n}}\right)^2 + \ldots\right]}.
\end{align*}
\]

Substituting into (10.1) yields

\[
2\varphi^n_{k,y} - 1 = \frac{\frac{rT}{n} - \frac{1}{2} \sigma^2 \Delta y_{k,n} + A_{n,k} - \frac{B_{n,k}}{2}}{\sigma \sqrt{\Delta y_{k,n}} + \frac{B_{n,k}}{2}}.
\]

Substituting into (10.1) yields

\[
E[U^n_{k}] = \frac{\frac{rT}{n} - \frac{1}{2} \sigma^2 \Delta y_{k,n} + A_{n,k} - \frac{B_{n,k}}{2}}{1 + \frac{B_{n,k}}{2 \sigma \sqrt{\Delta y_{k,n}}}}.
\]

(10.4)

\[
= \frac{\frac{rT}{n} - \frac{1}{2} \sigma^2 \Delta y_{k,n} + A_{n,k} - \frac{B_{n,k}}{2}}{1 + D_{n,k}},
\]

where \( D_{n,k} = \frac{B_{n,k}}{2 \sigma \sqrt{\Delta y_{k,n}}} \).

We need to show that the small order terms converge uniformly to 0 as \( n \to \infty \).
First,

\[
A_{n,k} = \left[ \frac{1}{2} \left( \frac{r^T}{n} \right)^2 + \frac{1}{3!} \left( \frac{r^T}{n} \right)^3 + \ldots \right] - \left[ -\frac{1}{3!} (\sigma \sqrt{\Delta y_{k,n}})^3 + \frac{1}{4!} (\sigma \sqrt{\Delta y_{k,n}})^4 - \ldots \right] \\
= e^{-\frac{r^T}{n}} - 1 - \frac{r^T}{n} - \left( e^{-\sigma \sqrt{\Delta y_{k,n}}} - 1 + \sigma \sqrt{\Delta y_{k,n}} - \frac{1}{2} \sigma^2 \Delta y_{k,n} \right),
\]

and

\[
\frac{B_{n,k}}{2} = \frac{1}{2} \left[ e^{\sigma \sqrt{\Delta y_{k,n}}} - e^{-\sigma \sqrt{\Delta y_{k,n}}} - 2\sigma \sqrt{\Delta y_{k,n}} \right] \\
= \frac{1}{2} \left[ \left( e^{\sigma \sqrt{\Delta y_{k,n}}} - 1 - \sigma \sqrt{\Delta y_{k,n}} - \frac{1}{2} \sigma^2 \Delta y_{k,n} \right) - \\
\left( e^{-\sigma \sqrt{\Delta y_{k,n}}} - 1 + \sigma \sqrt{\Delta y_{k,n}} - \frac{1}{2} \sigma^2 \Delta y_{k,n} \right) \right].
\]

Since

\[
\left| A_{n,k} - \frac{B_{n,k}}{2} \right| \leq \left| A_{n,k} \right| + \left| \frac{B_{n,k}}{2} \right| \\
\leq \left( \frac{r^T}{n} \right)^2 e^{\left( \frac{r^T}{n} \right)} + 2(\sigma \sqrt{\Delta y_{k,n}})^2 e^{\sigma \sqrt{\Delta y_{k,n}}},
\]

we can sum over all \( k \) to obtain

\[
\sum_{k=1}^{\left[ \frac{n}{T} \right]} \left| A_{n,k} - \frac{B_{n,k}}{2} \right| \leq \frac{Tr^2}{n} e^{\frac{r^T}{n}} + 2\max_{\left( 1 \leq k \leq \left[ \frac{n^2}{T} \right] \right)} \left( \sigma^3 \sqrt{\Delta y_{k,n}} \right) \sum_{k=1}^{\left[ \frac{n^2}{T} \right]} \Delta y_{k,n} \\
= \frac{Tr^2}{n} e^{\frac{r^T}{n}} + 2\max_{\left( 1 \leq k \leq \left[ \frac{n^2}{T} \right] \right)} \left( \sigma^3 \sqrt{\Delta y_{k,n}} \right) Y_t.
\]
Applying condition (3.4) we have,

$$\lim_{n \to \infty} \sum_{k=1}^{\left[ \frac{nT}{T} \right]} \left| A_{n,k} - B_{n,k} \right| = 0.$$  

To show that the mean converges, let

$$N_{n,k} = \frac{rT}{n} - \frac{1}{2} \sigma^2 \Delta y_{k,n} + A_{n,k} - \frac{B_{n,k}}{2}.$$  

We need to show the following converges to 0 as $n \to \infty$:

$$\sum_{k=1}^{\left[ \frac{nT}{T} \right]} \left| \frac{N_{n,k}}{1 + D_{n,k}} - N_{n,k} \right| = \sum_{k=1}^{\left[ \frac{nT}{T} \right]} \left| N_{n,k} \left( \frac{D_{n,k}}{1 + D_{n,k}} \right) \right|$$

$$\leq \max_{1 \leq k \leq \left[ \frac{nT}{T} \right]} \left( \frac{D_{n,k}}{1 + D_{n,k}} \right) \sum_{k=1}^{\left[ \frac{nT}{T} \right]} |N_{n,k}|.$$  

Because

$$\sum_{k=1}^{\left[ \frac{nT}{T} \right]} |N_{n,k}| \leq rt + \frac{1}{2} \sigma^2 Y_t + \sum_{k=1}^{n} \left| A_{n,k} - \frac{B_{n,k}}{2} \right| < \infty,$$

we just need to show that $D_{n,k} \to 0$. Now,

$$D_{n,k} = \frac{B_{n,k}}{2\sigma \sqrt{\Delta y_{k,n}}} \leq (\sigma \sqrt{\Delta y_{k,n}})^2 e^{\sigma \sqrt{\Delta y_{k,n}}}.$$
Thus by applying condition (3.4), we have

\[ \lim_{n \to \infty} \max_{1 \leq k \leq n} D_{n,k} \to 0, \]

and so

\[ \lim_{n \to \infty} \sum_{k=1}^{\left\lceil \frac{n}{2} \right\rceil} \frac{N_{n,k}}{1 + D_{n,k}} - N_{n,k} = 0. \]

This implies

\[ \lim_{n \to \infty} \sum_{k=1}^{\left\lfloor \frac{n}{2} \right\rfloor} \frac{N_{n,k}}{1 + D_{n,k}} = \lim_{n \to \infty} \sum_{k=1}^{\left\lceil \frac{n}{2} \right\rceil} N_{n,k}. \]

Therefore,

\[ \lim_{n \to \infty} E[X^n_t] = rt - \frac{1}{2} \sigma^2 Y_t, \]

and (3.14) is satisfied. \(\square\)

Next we show the variance converges.
Lemma 3.2.

\[ \lim_{n \to \infty} \text{Var}[X^n_t(y_t)] = \sigma^2 y_t. \]

Proof.

\[ \text{Var}[X^n_t(y_t)] = \sum_{k=1}^{[\frac{n}{T}]} \text{Var}[U^n_k] \]

\[ = \sum_{k=1}^{[\frac{n}{T}]} \sigma^2 \Delta y_{k,n} [1 - (2p_{k,y}^n - 1)^2]. \]

Thus we need only to show that \((2p_{k,y}^n - 1) \to 0\), for all \(k\). Alternatively, we can show that \(p_{k,y}^n \to \frac{1}{2}\) uniformly for all \(k\). We have

\[ p_{k,y}^n = \frac{e^{rT_n} - e^{-\sigma \sqrt{\Delta y_{k,n}}}}{e^{\sigma \sqrt{\Delta y_{k,n}}} - e^{-\sigma \sqrt{\Delta y_{k,n}}}} \]

\[ = \frac{e^{rT_n + \sigma \sqrt{\Delta y_{k,n}}} - 1}{e^{2\sigma \sqrt{\Delta y_{k,n}}} - 1}. \]

Let

\[ f(n) = e^{rT_n + \sigma \sqrt{\Delta y_{k,n}}} - 1 \]

and

\[ g(n) = e^{2\sigma \sqrt{\Delta y_{k,n}}} - 1. \]
Then, by applying L'Hopital's rule and condition (3.4),

\[
\lim_{n \to \infty} p_{k,y}^n = \lim_{n \to \infty} \frac{f'(n)}{g'(n)} = \frac{1}{2} + \frac{\frac{rT}{n} e^{rT} + \sqrt{\Delta y_{k,n}}}{2\sigma \sqrt{\Delta y_{k,n}} e^{2\sigma \sqrt{\Delta y_{k,n}}}} \to \frac{1}{2}.
\]

As a result, we have

\[
\left| \sum_{k=1}^{[ny]} \text{Var}[U_k^n] - \sum_{k=1}^{[ny]} \sigma^2 \Delta y_{k,n} \right| \leq \max_{1 \leq k \leq [ny]} (2p_{k,y}^n - 1)^2 \sum_{k=1}^{[ny]} \sigma^2 \Delta y_{k,n} \to 0.
\]

Therefore,

\[
\lim_{n \to \infty} \text{Var}[X^n_t(y_t)] = \sigma^2 y_t.
\]

Next, we prove (7.5), (7.6) and (7.7).

(7.5). We wish to show,

\[
\sum_{k=1}^{[ny]} E[U_{k}^n] \xrightarrow{p} -\frac{1}{2} \sigma^2 y_s \quad \text{for all } t \in [0, T].
\]

First consider

\[
E[U_{k}^n] = -\ln \left( \frac{e^{\frac{r}{\sqrt{n}}} + e^{-\frac{r}{\sqrt{n}}}}{2} \right).
\]
A Taylor series expansion yields
\[
\frac{e^{\frac{\sigma}{\sqrt{n}}} + e^{-\frac{\sigma}{\sqrt{n}}}}{2} = 1 + \frac{1}{2!} \left( \frac{\sigma}{\sqrt{n}} \right)^2 + \frac{1}{4!} \left( \frac{\sigma}{\sqrt{n}} \right)^4 + \ldots
\]

Now applying a Taylor expansion to the logarithmic terms yields
\[
\ln(\frac{e^{\frac{\sigma}{\sqrt{n}}} + e^{-\frac{\sigma}{\sqrt{n}}}}{2}) = \left[ \frac{1}{2!} \left( \frac{\sigma}{\sqrt{n}} \right)^2 + \frac{1}{4!} \left( \frac{\sigma}{\sqrt{n}} \right)^4 + \ldots \right] + \frac{1}{2!} \left[ \frac{1}{2!} \left( \frac{\sigma}{\sqrt{n}} \right)^2 + \frac{1}{4!} \left( \frac{\sigma}{\sqrt{n}} \right)^4 + \ldots \right]^2 + \ldots
\]

Summing the terms we have,
\[
\sum_{k=1}^{[ny]} \ln(\frac{e^{\frac{\sigma}{\sqrt{n}}} + e^{-\frac{\sigma}{\sqrt{n}}}}{2}) = \frac{1}{2} \sigma^2 y_t + O\left( \frac{1}{n} \right).
\]

Therefore,
\[
\sum_{k=1}^{[ny]} \mathbb{E}[U^n_k] \to -\frac{1}{2} \sigma^2 y_s,
\]
and (7.5) is satisfied.

(7.6). Now we show that \( \sum_{k=1}^{[ny]} \text{Var}[(U^n_k)] \to P \sigma^2 y_t \) for all \( t \in [0, T] \).

Taking expectations over squared terms we get,
\[
\mathbb{E}[U^n_k]^2 = \frac{\sigma^2}{n} + \left[ \ln\left( \frac{e^{\frac{\sigma}{\sqrt{n}}} + e^{-\frac{\sigma}{\sqrt{n}}}}{2} \right) \right]^2.
\]
Squaring the mean, we have

\[(E[U_k^n])^2 = \left[ \ln \left( \frac{e^{\frac{\sigma}{\sqrt{n}}} + e^{-\frac{\sigma}{\sqrt{n}}}}{2} \right) \right]^2.\]

So,

\[Var[U_k^n] = \frac{\sigma^2}{n},\]

and, thus,

\[\sum_{k=1}^{[nyt]} Var[U_k^n] = \sigma^2 y_t.\]

(7.7) We will now prove

\[\sum_{k=1}^{[nyt]} [E[1_{\{|U_k^n| \geq \frac{a}{n}\}}]] \overset{P}{\to} 0 \quad \text{for all } t \in [0,T], \quad a > 0.\]

There exists an \(N\) such that for every \(n > N\),

\[|U_k^n| = \frac{\sigma}{\sqrt{n}} - \ln \left( \frac{e^{\frac{\sigma}{\sqrt{n}}} + e^{-\frac{\sigma}{\sqrt{n}}}}{2} \right) < a, \quad \text{this follows from above.}\]

Therefore for \(n > N\),

\[E[1_{\{|U_k^n| \geq \frac{a}{n}\}}] = 0 \quad \forall \quad 1 \leq k \leq n,\]

and (7.7) is satisfied.
We close the Appendix with some additional definitions and theory to further elucidate the proofs in this chapter:

**Definition 3.1 (Conditional Independence).** For any \( \sigma \)-fields \( \mathcal{F}, \mathcal{H} \subseteq \mathcal{G} \), \( \mathcal{F} \) and \( \mathcal{H} \) are conditionally independent given \( \mathcal{G} \), if

\[
P(F \cap H \mid \mathcal{G}) = P(F \mid \mathcal{G})P(H \mid \mathcal{G}) \quad \text{a.s.} \quad F \in \mathcal{F}, H \in \mathcal{H}.
\]

We can show that the \( U_{k,y}^n \) conditional on \( \mathcal{F}_y^T \) form an independent sequence.

\[
\int_Y P\left(U_{k,y}^n \cap X_{\frac{n-1}{n}} \mid \mathcal{F}_y^T\right) d\mathbb{P}_y = \int_Y P\left(U_{k,y}^n \mid \mathcal{F}_y^T\right) P\left(X_{\frac{n-1}{n}} \mid \mathcal{F}_y^T\right) d\mathbb{P}_y
\]

It follows that

\[
P\left(U_{k,y}^n \mid \mathcal{F}_y^T \vee \mathcal{G}^{n\left(\frac{k-1}{n}\right)}\right) = P\left(U_{k,y}^n \mid \mathcal{F}_y^T\right).
\]

Next, we consider the convergence of \( \mathbb{P}^n \) to \( \mathbb{P} \). Let \( \mathbb{P}^n \) and \( \mathbb{P} \) be the measures induced on \( D[0,T] \) by \( X^n \) and \( X \), respectively. If \( \mathbb{P}^{n}_{yt}, \mathbb{P}_{yt} \) are the measures \( \mathbb{P}^n, \mathbb{P} \) conditioned on a further measure \( \nu_y \), where \( \nu_y \) is a measure in \( C[0,T] \) and \( \mathbb{P}^{n}_{yt} \rightarrow \mathbb{P}_{yt} \), then \( \mathbb{P}^n \rightarrow \mathbb{P} \). This follows from the Dominated Convergence Theorem.

We have, for any smooth bounded function \( f \) on \( D[0,T] \) and \( x \in D[0,T] \),

\[
\lim_{n \to \infty} \int_{D[0,T]} f(x) d\mathbb{P}^n(x) = \lim_{n \to \infty} \int_{C[0,T]} \int_{D[0,T]} f(x) d\mathbb{P}^n_{yt}(x) d\nu_y(y_t)
\]

\[
= \int_{C[0,T]} \int_{D[0,T]} f(x) d\mathbb{P}_{yt}(x) d\nu_y(y_t)
\]

\[
= \int_{D[0,T]} f(x) d\mathbb{P}_{yt}(x).
\]

Since \( \mathbb{P}^n \rightarrow \mathbb{P}, X^n \xrightarrow{\mathcal{L}} X \).
CHAPTER 4

A Markov Chain Monte Carlo Approach for Simulating
FATGBM and Related Processes

1. Introduction

In Chapter 3, we developed a binomial tree method to price path-dependent
options under FATGBM, which involved the simulation of future activity time
paths, Tt, to construct the tree. This chapter extends the work of Chapter 3 by
incorporating a new and novel method for simulating activity time paths. The
method uses a Markov Chain Monte Carlo (MCMC) algorithm to incorporate
asset price path history in the simulation of future activity time paths which can
then be used to construct the binomial tree. We also show how this method
can be used to simulate future activity time paths for another process similar
to FATGBM, and we compare the Markov Chain Monte Carlo mixing times for
both processes. This method can be used for any long range dependent process
to observe the effect of long range dependence on option prices.

Ideally, a long range dependent model should use historical price paths to simulate
future price paths, but the question is, how to do this? To stipulate the problem
precisely, we wish to estimate a conditional expectation, $E_Q[f(S_T)|A]$, under a
risk-neutral measure $Q$, by conditioning on some subset $A$ of price history, where
A = \{S_{-a}, \ldots, S_0\} and a > 0 is a positive integer. An additional question is: how large does a have to be, or how far back in time do we need to go, to accurately estimate the conditional expectation of interest? We initially focus on addressing the first problem: that is, how to estimate \( E_Q[f(S_T)|A] \) under FATGBM or any long range dependent model.

2. Method

Recall that the price process \( S_t \) is modelled under FATGBM as,

\[
S_t = S_0 e^{\mu t + \sigma W(T_t)},
\]

where \( W \) is independent of \( T_t \) and,

\[
T_t = \sum_{i=0}^{t} \left( \frac{\nu - 2}{\sum_{j=1}^{\nu} \eta_j^2(i)} \right).
\]

Because we are unable to directly observe the activity time process \( T_t \), we have no way of being able to use historical activity times to simulate future activity time paths. Note that we are unable to directly observe activity time because the \( \eta_j(i) \) represent independent copies of a theoretical (non-observable) zero mean Gaussian process with specified covariance function, at time \( i \). We can, however, observe the mean-corrected price increments \( Z_i = \log\left(\frac{S_i}{S_{i-1}}\right) - \mu = \sigma(W(T_i) - W(T_{i-1})) \), given the parameters \( \mu \) and \( \sigma \), which we can estimate directly from price data. One issue to note, however, is that the parameter \( \sigma \) is not completely trivial to estimate but this problem will be discussed briefly later in the chapter. Since the price process under the risk-neutral measure \( Q \) requires us to know the activity time path \( \{T_t\} \), we would like to be able to simulate future activity time paths,
given some subset $A$ of the past. The problem lies in the construction of the $\{T_t\}$ process. If we cannot observe the $\eta_j(i)$, and hence $T_t$, how can we then imply out activity time history given past price data? What we would really like to do is sample from the joint distribution of all the past $\eta$'s, denoted by $\{\eta\}$, conditional on all the past prices $A$; that is, we would like to sample from this distribution denoted here by $p(\{\eta\}|A)$. The ability to sample from this distribution will enable us to simulate future dependence given the past, hence enabling us to construct future activity time dependent on past price information. The problem is, this conditional distribution is very difficult to specify and hence sampling directly from this conditional distribution is not typically possible. Here, we propose an acceptance-rejection sampling algorithm that will allow us to sample from the required conditional distribution. Specification of the density $p(\{\eta\}|A)$ requires calculation of a normalising constant, and such a calculation is intractable. However, the specification of the ratio of two values of the density $p(\{\eta\}|A)$ for different values of $\eta$ is easier since the ratio does not explicitly involve the normalising constants that make specification of the original density difficult. Further, if we are able to consecutively sample from the distribution $p(\{\eta\}|A)$ such that each sample depends only on the previous sample then we will have constructed a Markov Chain. Then, if we are able to sample from some known proposal distribution $q(\{\eta\}|A)$ such that each proposal depends only on the previous sample, and we accept or reject our proposed value according to some probabilistic rule, then we still have a Markov Chain, but if our chain converges to the stationary distribution characterised by $p(\{\eta\}|A)$, then we will be able to sample from
our required conditional distribution. Hence, by constructing such a Monte Carlo Markov Chain (MCMC), we are able to sample from this complicated conditional distribution as long as we can show that our chain converges to our desired distribution. This algorithm will enable us to simulate a $\nu \times n$ dimensional matrix, $\{\eta\}$, conditional on $n$ observed mean-corrected log prices. This results in the matrix depicted in Figure 1. We can then choose a pair $(j, i)$ ($j \in \{1, \ldots, \nu\}$ and $i \in \{1, \ldots, n\}$) uniformly at random and update $\eta_{ji}$ to $\eta_{ji}^*$ according to some proposal distribution. We will then be able to accept/reject this new value according to some probabilistic rule. This will give rise to the updated matrix depicted in Figure 2. If we do this a large number of times, the matrix will then contain a set of updated values, $\{\eta^*\}$, conditional on the observed prices $\{Z\}$. If we choose our proposal distribution appropriately, the updated matrix will be a sample from the law of the stationary distribution or target density of this Markov chain. In
2. METHOD

Figure 2

Observed Price Increments $Z_1 \quad Z_2 \quad \ldots \quad Z_i \quad \ldots \quad Z_n$

1

$$\begin{pmatrix}
\eta_{11} & \eta_{12} & \ldots & \eta_{1i} & \ldots & \eta_{1n} \\
\eta_{21} & \eta_{22} & \ldots & \eta_{2i} & \ldots & \eta_{2n} \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
\eta_{ji} & \eta_{j2} & \ldots & \eta_{ji} & \ldots & \eta_{jn} \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
\eta_{iv} & \eta_{iv} & \ldots & \eta_{iv} & \ldots & \eta_{vn}
\end{pmatrix}$$

Activity Time Increments $T_1 \quad T_2 \quad \ldots \quad T_i^* \quad \ldots \quad T_n$

order to see why this works, we need to invoke some MCMC theory. In particular, we will consider the Metropolis-Hastings Algorithm and prove that this algorithm results in convergence to a stationary distribution. We will then adapt this algorithm to our specific problem and prove that our algorithm converges to the desired conditional distribution discussed above. There are many variants of the Metropolis-Hastings algorithm and indeed it will be necessary to vary this algorithm for our purposes, but the central principle is the same.

**Definition 4.1.** A Markov Chain Monte Carlo (MCMC) method for the simulation of a distribution $f$ is any method producing an ergodic Markov chain $(X^{(t)})$ whose stationary distribution is $f$. 
The Metropolis-Hastings algorithm starts with a target density $f$, the distribution from which you wish to sample. It then requires specification of a known conditional density $q(y|x)$ which is used to generate a proposal value $y$ given the current state of the chain $x$. The idea is that one can start at any initial state $x_0$ and as long as the chain is irreducible, positive recurrent and aperiodic, then it will be ergodic; in other words, if the algorithm runs for long enough (the burn-in period), then the distribution of the chain will converge to its stationary distribution and the empirical average of the sequence of dependent samples will be a good approximation to the theoretical expectation of an independent and identically distributed sequence under the stationary distribution. More precisely,

$$\frac{1}{M} \sum_{i=1}^{M} g(X^{(t)}) \xrightarrow{a.s.} E_f[g(X)], \text{ as } M \to \infty.$$  

(2.3)

It is important to bear in mind at this point is that our chain $X^{(t)}$ will not be one-dimensional but will rather be $\nu \times n$ dimensional, where only a single element of the matrix depicted in Figure 1, $\eta_{ji}$, will be replaced at any time point in the chain. This increased dimension will, of course, have implications on the burn in time necessary for the chain to stabilise and ultimately converge and on the dependence between $X^k$ and $X^t$ where $k$ is close to $t$ (see Figures 1 and 2), but in the limit ergodicity will still apply, and all of these considerations suggest that for our purposes $M$, the number of simulated sample points in the chain, may need to be very large.
We now outline the Metropolis-Hastings algorithm associated with the target density $f$ and proposal density $q(y|x)$.

Let $\mathcal{X}$ be a finite set and $f(x) > 0$ a probability distribution on $\mathcal{X}$. The Metropolis-Hastings algorithm is a procedure for drawing samples from $\mathcal{X}$ (Metropolis, Rosenbluth, Rosenbluth, Teller and Teller, 1953).

The algorithm requires specification of an irreducible and aperiodic Markov chain $Y_t$ with distribution $q(y|x)$ on $\mathcal{X}$. The chain is then modified to construct another irreducible, aperiodic Markov chain $X^{(t)}$ with distribution $p(x,y)$ on $\mathcal{X}$. Under these conditions, the Metropolis-Hastings algorithm produces a chain, $X^{(t)}$, whose distribution, $p(x,y)$, converges to the stationary distribution $f$.

**Algorithm 4.1. GENERAL METROPOLIS-HASTINGS ALGORITHM**

*Given $x^{(t)}$,*

1. Generate $Y_t \sim q(y|x^t)$.

2. Take

$$X^{(t+1)} = \begin{cases} 
Y_t, & \text{with probability } \rho(X^{(t)}, Y_t) \\
X^{(t)}, & \text{with probability } 1 - \rho(X^{(t)}, Y_t)
\end{cases}$$

where,

$$\rho(x, y) = \min \left\{ \frac{f(y) q(x|y)}{f(x) q(y|x)}, 1 \right\}.$$
We now state the following definitions and theorems (without proof) in order to establish that the distribution of the chain $X^{(t)}$ converges to $f$.

**Definition 4.2.** A vector $f = (f_1, \ldots, f_k)^T$ is said to be a stationary distribution for the Markov chain $\{X_0, X_1, X_2, \ldots\}$ if:

1. $f \geq 0 \ \forall \ i \in \{1, \ldots, k\}$ and $\sum_{i=1}^{k} f_i = 1$
2. $f^T P = f^T$.

**Definition 4.3.** Let $\mu = (\mu_1, \ldots, \mu_k)^T$ and $\nu = (\nu_1, \ldots, \nu_k)^T$ be two probability distributions on the state space $S = \{s_1, \ldots, s_k\}$. The total variation distance between $\mu$ and $\nu$ is defined as

$$d_{TV}(\mu, \nu) = \frac{1}{2} \sum_{i=1}^{k} |\mu_i - \nu_i|.$$  

A sequence of probability distributions $f^i$ converges in total variation to a distribution $f$ if

$$\lim_{i \to \infty} d_{TV}(f^i, f) = 0.$$  

As shorthand, we write $f^i \xrightarrow{TV} f$.

**Theorem 4.1.** (*Markov chain convergence theorem*) Consider an irreducible and aperiodic Markov chain $\{X_0, X_1, X_2, \ldots\}$. If we denote the chain’s distribution after the $n$th transition by $\mu^n$, then we have for any initial distribution
\( \mu^0 \) and a stationary distribution \( f \):

\[ (3.5) \quad \mu^n \xrightarrow{TV} f. \]

This result means that if we run the Markov chain for a very long time, its distribution will be very close to the stationary distribution \( f \).

**Definition 4.4.** A probability distribution \( f \) on the state space \( S = \{s_1, ..., s_k\} \) is **reversible** for the Markov chain \( \{X_0, X_1, X_2, ...\} \) with transition matrix \( P \) if for all \( i, j \in 1, ... k \) we have

\[ (3.6) \quad f_i P_{i,j} = f_j P_{j,i}. \]

**Proposition 4.1.** If the probability distribution \( f \) is reversible for a Markov chain, then it is also a stationary distribution for the chain.

**Proof.**

\[
 f_j = f_j \sum_{i=1}^{k} P_{j,i} = \sum_{i=1}^{k} f_j P_{j,i} = \sum_{i=1}^{k} f_i P_{i,j} \\
\Rightarrow f^T = f^T P. \quad \square
\]

**Theorem 4.2.** The chain \( X^{(t)} \) of Algorithm 4.1 is an irreducible, aperiodic Markov chain on \( \mathcal{X} \) with

\[ (3.7) \quad f(x)p(x, y) = f(y)p(y, x) \quad \forall x, y. \]

In particular, for all \( x, y \)

\[ (3.8) \quad \lim_{n \to \infty} p^n(x, y) = f(y). \]
PROOF. To prove (3.7), we just need to show that the probability distribution $f$ is reversible for the chain $X^{(t)}$.

\[ f(x)p(x, y) = f(x)q(y|x)p(x, y) \]
\[ = f(x)q(y|x)\min \left\{ \frac{f(y)q(x|y)}{f(x)q(y|x)}, 1 \right\} \]
\[ = \min \{f(x)q(y|x), f(y)q(x|y)\} \]
\[ = f(y)q(x|y)\min \left\{ \frac{f(x)q(y|x)}{f(y)q(x|y)}, 1 \right\} \]
\[ = f(y)q(x|y)\rho(y, x) = f(y)p(y, x). \]

Equation (3.8) then follows from Theorem 4.1 and Proposition 4.1, since the chain is aperiodic and irreducible by assumption. \qed

**Lemma 4.1.** Assume $f$ is bounded and positive on every compact set of the support $\mathcal{E}$. If there exist positive numbers $\epsilon$ and $\delta$ such that

\[(3.9) \quad q(y|x) > \epsilon \text{ if } |x - y| < \delta, \]

then the Metropolis-Hastings Markov chain $X^{(t)}$ is irreducible and aperiodic.

4. MCMC for FATGBM

We now consider our specific application, noting the definitions and results above. Here we will define new notation and incorporate an additional step which will be specific to our algorithm. The additional step is required because we wish
to update an entire $\nu \times n$ matrix with proposed values by choosing a pair $(j, i)$ one step at a time. Note that our desired stationary distribution $f(x)$ is given by $f(\{\eta\} \mid \{Z\})$, and our known proposal distribution, $q(y|x)$, is denoted by $f(\eta_{ij} \mid \{\eta\}_{\neq ij})$, while $\rho(x, y)$, the probability of accepting $\{\eta^*\}(\rightarrow y)$, given $\{\eta\}(\rightarrow x)$, is given by $\rho(\eta, \eta^*)$.

Consider the process $W(T_i)$, defined in Chapter 2, whose increments have the following distribution:

\[(4.1) \quad W(T_i) - W(T_{i-1}) \overset{d}{=} W_1 \sqrt{T_i}.\]

Let $Z_i = W_i \sqrt{T_i}$ be the $i$th observed increment of the process $\{Z\}$. As defined in Chapter 2,

\[(4.2) \quad T_i = \frac{\nu - 2}{\sum_{j=1}^{\nu} \eta_j^2(i)},\]

where $\eta_j(i) = \eta_{ij}$, $j = 1, ..., \nu$, are $\nu$ independent copies of a multivariate Gaussian random vector with zero mean and covariance function $(1 + \tau^2)^{-\frac{3}{2}}$, and where $\alpha = 1 - H$, and $\frac{1}{2} < H < 1$.

Let $\{\eta\} = \{\eta_{ij}, \forall i = 1, ..., n, \ j = 1, ..., \nu\}$, and $\{Z\} = Z_i, \forall i = 1, ..., n$. We wish to use the Metropolis-Hastings algorithm given by Algorithm 4.2, below, to sample from $f(\{\eta\} \mid \{Z\})$. Define $\{\eta\}_{\neq ij} = \{\eta_{im} : \forall \{(l, m) : (l, m) \neq (i, j)\}\}$.

**Algorithm 4.2.**

*Given an initial state $\{\eta\}^1$ and a sequence of increments $\{Z\}$,*

1. Choose an $\eta_{ij}$ uniformly at random.
2. Generate \( \eta_{ij}^* \sim f(\eta_{ij} \mid \{\eta\} \neq i,j) \)

Let \( \{\eta^*\} = \{\eta_{ij}^*, \{\eta\} \neq i,j\} \)

3. Solve for \( W_i^* = \frac{Z_i}{\sqrt{T_i^*}}, \) where \( T_i^* = \frac{e^{-2}}{\eta_{ij}^2 + \sum_{m \neq j} \eta_{im}^2} \)

4. Generate

\[
\{\eta\}^{t+1} = \begin{cases} 
\{\eta^*\}, & \text{with probability } \rho(\{\eta\}^t, \{\eta^*\}) \\
\{\eta\}^t, & \text{with probability } 1 - \rho(\{\eta\}^t, \{\eta^*\})
\end{cases}
\]

where

\[
\rho(\eta, \eta^*) = \min \left\{ \frac{f(W_i^*)}{f(W_i)} \left( \frac{\eta_{ij}^* + \sum_{m \neq j} \eta_{im}^2}{\eta_{ij}^2 + \sum_{m \neq j} \eta_{im}^2} \right), 1 \right\}
\]

5. Repeat for \( t = 2, \ldots \) until satisfactory convergence.

**Corollary 4.3.** Algorithm 4.2 produces a Markov Chain (\( \{\eta^{(t)}\} \)) with stationary distribution \( f(\{\eta\} \mid \{Z\}) \).

**Proof.** Since this is a Metropolis-Hastings algorithm with \( q(y \mid x) \) satisfying Lemma 4.1, all that needs to be shown is that

\[
\frac{f(\{\eta^*\} \mid \{Z\}) f(\eta_{ij} \mid \{\eta\} \neq i,j)}{f(\{\eta\} \mid \{Z\}) f(\eta_{ij}^* \mid \{\eta\} \neq i,j)} = \frac{f(W_i^*)}{f(W_i)} \left( \frac{\eta_{ij}^2 + \sum_{m \neq j} \eta_{im}^2}{\eta_{ij}^2 + \sum_{m \neq j} \eta_{im}^2} \right).
\]

Let \( \eta_{im} = \{\eta_{m} : m \neq j\} \) and \( \{\eta\}_{ij} = \{\eta_{ij} : l \neq i\} \). Then,

\[
\frac{f(\{\eta^*\} \mid \{Z\})}{f(\{\eta\} \mid \{Z\})} = \frac{f(\eta_{ij}^* \mid \{\eta\} \neq i,j, Z_i)}{f(\eta_{ij} \mid \{\eta\} \neq i,j, Z_i)} = \frac{f(\{\eta\} \neq i,j, Z_i \mid \eta_{ij}) \times f(\eta_{ij})}{f(\{\eta\} \neq i,j, Z_i \mid \eta_{ij}) \times f(\eta_{ij})} = 1.
\]
Therefore,

\[
\frac{f(\{\eta^*\} \mid \{Z\}) \times f(\eta_{ij} \mid \{\eta\} \neq \{\eta_{ij}\})}{f(\{\eta\} \mid \{Z\}) \times f(\eta_{ij}^* \mid \{\eta\} \neq \{\eta_{ij}\})} = \frac{f(\{\eta\} \mid \{Z\}, \eta_{ij})}{f(\{\eta\} \mid \{Z\}, \eta_{ij})} \times \frac{f(\eta_{ij})}{f(\eta_{ij}^*)} \times \left| \frac{J(Z, \eta_{ij}, \{\eta\} \mid \{Z\})}{J(Z, \eta_{ij}, \{\eta\} \mid \{Z\})} \times \frac{f(\eta_{ij})}{f(\eta_{ij}^*)} \right|
\]

Recall,

\[
Z_i = W_i \sqrt{\frac{\nu - 2}{\sum_{j=1}^{\nu} \eta_{ij}^2}}.
\]

Thus,

\[
J(Z, \eta_{ij}, \{\eta\} \mid \{Z\}) =
\begin{vmatrix}
\frac{\partial W}{\partial Z} & \frac{\partial W}{\partial \eta_{ij}} & \frac{\partial W}{\partial (\eta) \mid \{Z\}} \\
\frac{\partial \eta_{ij}}{\partial Z} & \frac{\partial \eta_{ij}}{\partial \eta_{ij}} & \frac{\partial \eta_{ij}}{\partial (\eta) \mid \{Z\}} \\
\frac{\partial (\eta) \mid \{Z\}}{\partial Z} & \frac{\partial (\eta) \mid \{Z\}}{\partial \eta_{ij}} & \frac{\partial (\eta) \mid \{Z\}}{\partial (\eta) \mid \{Z\}}
\end{vmatrix}
\]

\[
= \sqrt{\frac{\eta_{ij}^2 + \sum_{m \neq \{Z\}} \eta_{im}^2}{\nu - 2}} \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix}
\]

\[
= \begin{vmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{vmatrix}
\]

\[
= \sqrt{\frac{\eta_{ij}^2 + \sum_{m \neq \{Z\}} \eta_{im}^2}{\nu - 2}}.
\]
Therefore,

$$\frac{f(W_i^*)}{f(W_i)} \times \frac{|J(Z, \eta_j, \{\eta\}_\text{im})|}{|J(Z, \eta_j, \{\eta\}_\text{im})|} = \frac{f(W_i^*)}{f(W_i)} \sqrt{\frac{\eta_{ij}^2 + \sum_{m \neq j} \eta_{im}^2}{\eta_{ij}^2 + \sum_{m \neq j} \eta_{im}^2}}.$$

One of the restrictions of FATGBM is that it is only defined for integer values of \( \nu \).

We now consider the process introduced by Sly (2006) which extends FATGBM so that it is defined for non-integer values of \( \nu \). We will call this new process FATGBM 2. We wish to come up with a new algorithm for FATGBM 2 using a Metropolis-Hastings method analogous to that described above. This algorithm will allow us to sample from FATGBM 2 by conditioning on the price history while also enabling us to compare mixing times for the new MCMC algorithm with mixing times for the MCMC algorithm outlined above, for various parameter values of \( \nu \) and \( H \). Sly (2006) defines the process as follows. Let \( R \Gamma(\beta, \alpha) \) denote the inverse gamma distribution with density function

$$\frac{\alpha^\beta}{\Gamma(\beta)} x^{-\beta-1} e^{-\frac{\alpha}{x}}, x > 0,$$

which has moment index \( \beta \). Let \( F_\nu \) be the distribution function of the random variable \( [(\nu - 1)/\nu]R \Gamma(\nu/2, \nu) \) and let \( \Phi \) denote the standard normal distribution function. To extend this setting to general non-integer values of \( \nu \), take

$$f_\nu(x) = F_\nu^{-1}[\Phi(x)].$$
Let $X(t)$ be a stationary Gaussian process with zero mean and covariance function $p_X(t) = (1 + t^2)^{-\frac{1-H}{2}}$ for some $\frac{1}{2} < H < 1$. Then set

$$T_n = \sum_{s=1}^{n} f_\nu(X(s)).$$

The activity time increment $T_i$ thus has the distribution of $f_\nu(X(i))$, which is $[(\nu^2 - 1)/(\nu^2)]R\Gamma(\nu, \nu/2)$. Sly (2006) then showed that the process $T_t - t$ is asymptotically self-similar for various values of $\nu$. More specifically, when $\nu > 4$,

$$T_{ct} - ct \overset{d}{\rightarrow} KB_H(t)$$

as $c \rightarrow \infty$, where $K = E(f_\nu(X(s))X(s))$ and $B_H(t)$ is Fractional Brownian Motion with Hurst parameter $H$. This result shows us that FATGBM 2 retains the fractal properties of FATGBM, asymptotically, and is long range dependent with parameter $H$, for $\frac{1}{2} < H < 1$. Another benefit of FATGBM 2 is that it no longer requires us to simulate a $\nu \times n$ matrix but instead requires simulation of a vector of length $n$. This reduction in the dimension of the problem should speed up the mixing time of the MCMC algorithm, which will allow for speedier computation.

Once again, we are interested in sampling from a complicated distribution which involves conditioning on the observed increments of the process $W\sqrt{T_i}$. First, denote the observed increments, $Z_i = W_i\sqrt{T_i}$. We now wish to sample from the distribution $f(\{X\}|\{Z\})$, where $\{X\} = \{X(i), \forall i = 1, \ldots, n\}$. A Metropolis-Hastings algorithm for sampling from this distribution is given by
4. MARKOV CHAIN MONTE CARLO FOR FATGBM

Algorithm 4.3.

Given an initial vector \( \{X\}^1 \) and a sequence of increments \( \{Z\} \),

1. Choose an \( X(i) \) uniformly at random

2. Generate \( X^*(i) \sim f(X(i) \mid \{X\} \neq i) \)

Let \( \{X^*\} = \{X^*(i), \{X\} \neq i\} \)

3. Solve for \( W^*_i = \frac{Z_i}{\sqrt{T^*_i}} \), where \( T^*_i = f_{\nu}(X(i)) \)

4. Generate

\[
\{X\}^{t+1} = \begin{cases} 
\{X^*\}, & \text{with probability } \rho(\{X\}^t, \{X^*\}) \\
\{X\}^t, & \text{with probability } 1 - \rho(\{X\}^t, \{X^*\})
\end{cases}
\]

where

\[
\rho(X, X^*) = \min \left\{ \frac{f(W^*_i)}{f(W_i)}, \sqrt[\frac{f_{\nu}(X(i))}{f_{\nu}(X^*(i))}}, 1 \right\}
\]

5. Repeat for \( t = 2, \ldots \) until satisfactory convergence.

The algorithm can be visually depicted as a \( 2 \times n \) matrix, the first row being the \( n \) historical prices and the second row being the initial vector \( \{X\}^1 \) which may be chosen arbitrarily. The initial matrix is depicted in Figure 3. The matrix can then be updated by choosing an \( X(i) \) uniformly at random and then updating to \( X^*(i) \) according to step 2 of Algorithm 4.3. The updated matrix is shown in Figure 4. Note that the vector object updated in each step of the algorithm described by Figures 3 and 4 is of lower dimension and size than the matrix that arises for the algorithm for the FATGBM case because FATGBM 2 is much simpler in
construction than FATGBM in that it does not require $\nu$ independent Gaussian random vectors of length $n$ but rather relies on only one random vector of length $n$ with a slightly different covariance function. The resulting distribution is still the same as FATGBM; that is, at any time point, it will have a Student-t distribution with $\nu$ degrees of freedom. We now prove that Algorithm 4.3 results in a Markov Chain with the stationary distribution from which we wish to sample.

**Corollary 4.4.** Algorithm 4.3 produces a Markov Chain ($\{X^{(i)}\}$) with stationary distribution $f(\{X\} | \{Z\})$. 
PROOF. Since this is a Metropolis-Hastings algorithm with \( q(y|x) \) satisfying Lemma 4.1, all that needs to be shown is that,

\[
\frac{f(\{X^*\} \mid \{Z\}) \ f(X(i) \mid \{X\} \neq i)}{f(\{X\} \mid \{Z\}) \ f(X^*(i) \mid \{X\} \neq i)} = \frac{f(W_i^*)}{f(W_i)} \sqrt{\frac{f_\nu(X(i))}{f_\nu(X^*(i))}}.
\]

Note that

\[
\frac{f(\{X^*\} \mid \{Z\})}{f(\{X\} \mid \{Z\})} = \frac{f(X^*(i) \mid \{X\} \neq i, Z_i)}{f(X(i) \mid \{X\} \neq i, Z_i)} = \frac{f(\{X\} \neq i, Z_i \mid X^*(i))}{f(\{X\} \neq i, Z_i \mid X(i))} \times \frac{f(X^*(i))}{f(X(i))} = \frac{f(Z_i \mid X^*(i)) \times f(\{X\} \neq i \mid X^*(i))}{f(Z_i \mid X(i)) \times f(\{X\} \neq i \mid X(i))} \times \frac{f(X^*(i))}{f(X(i))}
\]

Therefore,

\[
\frac{f(\{X^*\} \mid \{Z\}) \times f(X(i) \mid \{X\} \neq i)}{f(\{X\} \mid \{Z\}) \times f(X^*(i) \mid \{X\} \neq i)} = \frac{f(Z_i \mid X^*(i))}{f(Z_i \mid X(i))} = \frac{f(W_i^*)}{f(W_i)} \times \frac{|J(Z, X^*)|}{|J(Z, X)|}
\]

Recall,

\[Z_i = W_i \sqrt{f_\nu(X(i))}.
\]

Hence,

\[
J(Z, X) = \begin{vmatrix}
\frac{\partial W}{\partial Z} & \frac{\partial W}{\partial X} \\
\frac{\partial X}{\partial Z} & \frac{\partial X}{\partial X}
\end{vmatrix}
\]

\[
= \begin{vmatrix}
\frac{1}{\sqrt{f_\nu(X(i))}} & 0 \\
\frac{\partial X}{\partial Z} & 1
\end{vmatrix}
\]
Therefore,

\[
\frac{f(W_t^*)}{f(W_t)} \times \frac{|J(Z,X^*)|}{|J(Z,X)|} = \frac{f(W_t^*)}{f(W_t)} \sqrt{\frac{f_{\nu}(X(i))}{f_{\nu}(X^*(i))}}.
\]

To obtain a convenient computational formula for \( f_{\nu}(x) = F_{\nu}^{-1}[\Phi(x)] \), first denote \( Y = f_{\nu}(x) \) and note that \( Y = \frac{\nu - 2}{\nu} G \), where \( G \sim \Gamma\left(\frac{\nu}{2}, \frac{2}{\nu y}\right) \).

Then,

\[
P(Y \leq y) = P\left(\frac{1}{G} \leq \frac{\nu y}{\nu - 2}\right) = P\left(G \geq \frac{\nu - 2}{\nu y}\right)
\]

so,

\[
1 - P\left(G \leq \frac{\nu - 2}{\nu y}\right) = \Phi(X).
\]

Therefore,

\[
P(G \leq \frac{\nu - 2}{\nu y}) = 1 - \Phi(X),
\]

and,

\[
\frac{\nu - 2}{\nu y} = \Gamma^{-1}(1 - \Phi(X)).
\]

Therefore,

\[
y = \frac{\nu - 2}{\nu} \frac{1}{\Gamma^{-1}(1 - \Phi(X))}.
\]
5. Implementation of Algorithm

Before we discuss how we perform diagnostic tests of the algorithms in practice, we return to the problem of estimating the parameters in the pricing model, \( \mu \) and \( \sigma \), as an important requirement of the algorithm is the specification of these parameters. If we chose instead to sample from \( f(\{\eta\}|S_0, ... S_{-a}) \), that is, to sample from the joint distribution of \( \eta \) conditional on price history as opposed to sampling from the joint distribution of \( \eta \) conditional on the mean-corrected price history, \( \{Z\} \), we would still encounter the same necessity of having to specify the parameters \( \mu \) and \( \sigma \). This circumstance arises because \( \log\left( \frac{S_t}{S_{t-1}} \right) \overset{d}{=} \mu + \sigma W(T_i) \), and hence the probability of acceptance in the algorithm will be affected by this new distribution which still requires specification of \( \mu \) and \( \sigma \). The estimation of \( \mu \) is not commonly required in most option pricing models, since risk-neutral pricing requires a change of measure which generally results in a price process where \( \mu \) is replaced by \( r \), the risk free rate. Nevertheless, estimation of \( \mu \) in long range dependent processes is a statistical exercise that has been investigated in the literature; see, for example, Hu, Nualart, Xiao and Zhang (2011). The requirement that \( \mu \) be estimated is not related to stability over time but rather to mean correction in the sense that the objective is to specify the de-trended price series so that our observations have zero drift, and unit variance parameter. While it may be more straightforward to simply remove the drift component of our series, it is less obvious how to estimate the parameter \( \sigma \), since it may seem from our initial description of FATGBM that volatility could be entirely captured by the Activity Time process \( \{T\} \), resulting in us being able to use \( \sigma = 1 \) with
no loss of generality. However, a result arising from the properties of FATGBM can be used to show that $\sigma$ may not always be set equal to 1. Further, however, the result suggests a way that $\sigma$ may be estimated appropriately. Consider the Activity Time increments

$$X_i = T_i - T_{(i-1)} = \frac{\nu - 2}{\eta_1^2 + \ldots + \eta_\nu^2} I_i,$$

where $T_i \sim R \Gamma(\frac{\nu}{2}, \frac{\nu-2}{2})$, so that $E[T_1] = 1$. Then, by the Birkhoff Ergodic theorem, which states that

$$\frac{1}{n} \sum_{i=1}^{n} X_i \rightarrow Y,$$

where $Y$ is a random variable satisfying $E[Y] = E[\bar{X}_n] = E[X_1]$, and where $X$ is a strictly stationary process, we have

$$\lim_{t \to \infty} \frac{1}{t} T_t = 1, a.s.$$ (5.1)

Then, applying Itô’s formula to the price process (2.1), we can empirically construct Activity Time:

$$d(\log S_t) - \frac{dS_t}{S_t} = \frac{1}{2} \sigma^2 dT_t,$$ (5.2)

and estimate $\sigma$ by solving for the value of $c$ for which $\frac{T_t}{t} = c$ for large $t$.

6. Diagnostics

We now discuss some diagnostic procedures to determine the efficacy of our MCMC algorithms in practice. First, we describe some practical choices that we made when we implemented the MCMC algorithms described above in the context of investigating the chain for a simulated set of observed increments. We
first need to simulate the $n$ observed values $\{Z\}$. We may do this by noting that $W(T_i) \overset{d}{=} W(1)\sqrt{T_i}$, where $W(1) \overset{d}{=} N(0, 1)$. This observation allows us to generate an independent normal random variate for each $i$ and multiply it by $\sqrt{T_i}$, which itself is constructed by generating $\nu$ independent realisations of random Gaussian vectors of length $n$ with the specified covariance function. To specify the initial starting matrix of $\{\eta\}$ and $\{X\}$, we begin by setting each $T_i = 1$ so that in the initial state $Z_i = W_i$. Given this simple relationship, we can then solve readily for each $\eta(i)$ and $X(i)$. Note that this choice of initial state is done more as a convenience than as a requirement of the algorithm, since arbitrary starting values of $\{\eta\}$ and $\{X\}$ could be used, as the target stationary distribution does not depend on the chain’s starting values. However, by specifying them in this way, the mean of our initial matrix of $\eta$’s is not unusual, in the sense that $E[T_i] = 1$. The implementation of both of the algorithms described above was carried out first in the R statistical programming language, but later it became necessary to rewrite the implementation in C using Microsoft Visual Studio. This choice was precipitated by the slowness of the original implementation in R in the context of the large number of MCMC iterations that were needed to provide stable convergence of the chain. Moving the implementation to C resulted in dramatic improvement in the speed of the algorithm, and it is a common practice to implement MCMC algorithms in programs such as C and Java rather than in R because of the dramatic resultant speed boost that typically accrues. Within the C implementation, the GNU Scientific Library was used to provide certain statistical and other functions, and the Armadillo C++ Linear Algebra library
was used to optimise matrix calculations. Once the algorithm was implemented, we monitored the stability of the simulated chain to gauge the mixing time of the chain and therefore the convergence of the algorithm. We noticed that the extent of the long range dependence assumed had the biggest impact on mixing time, with greater long range dependence resulting in considerably slower convergence of the chain. We also tested the algorithms for both processes using the covariance function for Fractional Brownian Motion (FBM) as well as the covariance functions typically used for FATGBM and FATGBM 2. The covariance function for FBM is a result of the following theorem.

**Theorem 4.5. (Taqqu, 1981)** Let \( \{B(t)\} \) be real-valued \( H \)-self-similar with stationary increments and suppose that \( E[B(1)^2] < \infty \). Then

\[
E[B(s)B(t)] = \frac{1}{2}\left(t^{2H} + s^{2H} - |t - s|^{2H}\right)E[B(1)^2].
\]

The covariance function for the increments of Brownian Motion can then be readily derived.

**Corollary 4.6.** Let \( B(t) \) be an \( H \)-self-similar Fractional Brownian Motion and define \( X(t) = B(t) - B(t - 1) \) to be the increments of the Fractional Brownian Motion process. Then

\[
E[X(t)X(u)] = \frac{1}{2}\left[ (s + 1)^{2H} + (s - 1)^{2H} - 2|s|^{2H} \right],
\]

where \( s = t - u \).
PROOF.

\[ E[X(t)X(u)] = E[(B(t) - B(t - 1))(B(u) - B(u - 1))] \]

\[ = E[B(t)B(u)] - E[B(t)B(u - 1)] \]

\[ - E[B(t - 1)B(u)] + E[B(t - 1)B(u - 1)] \]

\[ = \frac{1}{2} [t^{2H} + u^{2H} - |t - u|^{2H} - |t^{2H} \]

\[ + (u - 1)^{2H} - |t - u + 1|^{2H}] \]

\[ - [(t - 1)^{2H} + (u)^{2H} - |t - u - 1|^{2H}] \]

\[ + [(t - 1)^{2H} + (u - 1)^{2H} - |t - u|^{2H}] \]

\[ = \frac{1}{2} [(t - u + 1)^{2H} - 2|t - u|^{2H} + |t - u - 1|^{2H}] \]

Setting \( u = t - s \),

\[ E[X(t)X(u)] = \frac{1}{2} [(s + 1)^{2H} + (s - 1)^{2H} - 2|s|^{2H}]. \]

The rationale for using FBM increments for the Activity Time process as opposed to the previously specified Gaussian random vectors underlying FATGBM and FATGBM 2 is that the use of FBM increments results in a FATGBM-type process that exhibits weaker dependence than FATGBM and FATGBM 2. In other words, using FBM increments results in a FATGBM-type process that is not \( H \)-self-similar but rather \( cH \)-self-similar where \( 0 < c < 1 \). This result can be proved using limit theorems that are beyond the scope of this thesis (see Taqqu, 1975).
We now present a series of graphs which depict a trace of the mean of the \( \{\eta\} \) and \( \{X\} \) matrices for \( n = 100 \) observations and five million MCMC iterations. This trace plot provides evidence that the MCMC chain is well mixed within one million steps in the chain. We also provide a trace of the means of these matrices for various values of the parameters \( H \) and \( \nu \) for one million MCMC iterations. We observe that the MCMC for FATGBM appears well mixed after far fewer steps in the chain when using the FBM covariance function rather than the typical covariance function used for FATGBM and FATGBM 2. We also observe the effect of increasing \( H \) on the mixing times of the mean of \( \{\eta\} \) and \( \{X\} \) for both FATGBM and FATGBM 2, using their respective covariance functions as well as the covariance function for FBM. In particular, we note that the means of \( \{\eta\} \) and \( \{X\} \) are close to zero, as is the case for the unconditional means, and the volatility of the means of \( \{\eta\} \) and \( \{X\} \) do not exceed 1, as the volatilities of the means are bounded below 1. The reason for these preliminary diagnostic checks is because the primary objective for using MCMC is to provide us with a solution to the problem of being unable to specify the unknown conditional distribution, \( \{\eta\} \) and \( \{X\} \), given the observed values \( \{Z\} \). This means that we have no obvious way of assessing whether the observed realisations actually arise from our complicated conditional distribution – that is, we do not know when the chain has actually converged except by considering the stability of features of the chain as it progresses. By employing a series of diagnostic tests, we will be confident that our chain has converged sufficiently for us to assume that the
sample values we obtain do indeed arise from the stationary distribution of interest. Firstly, after running the chain for five million iterations, we were satisfied that the chain had mixed well even before the one millionth step in the chain – see Figures 5 to 10. Secondly, the traces of the means of the \{η\} and \{X\} matrices reveal values centred around 0, which is the theoretical mean for the unconditional \{η\} and \{X\}'s. Thirdly, Normal quantile-quantile (Q-Q) plots for the MCMC realisations of \{η\} and \{X\} using the covariance function of FBM with $H = \frac{1}{2}$, revealed that the corresponding chain’s stationary distribution is standard Normal, as required – see Figures 14 and 15. Of course, as we have proven that the chain must converge to the desired stationary distribution, we know that the computed chain will eventually converge, but, equally, such convergence could, in principle, take a very long time. A further diagnostic test for the practical convergence of the chain would be to store the mean of \{η\} used in the initial construction of \{Z\}, the observed increments on which the required conditional distribution is based, and then check to see if it lies within about 95 of the 95% confidence intervals constructed from 100 independent MCMC simulations. However, since this process would take months of computational time even using the current fast implementation of the algorithm in C, we see no appreciable benefit of this method of diagnosis in any practical sense, especially given the positive signs arising from the diagnostics already discussed. Therefore, based on the evidence gathered, some of which is presented here (the remainder omitted for brevity), we are confident that the algorithm has achieved practical convergence, and produces samples from the desired conditional distribution.
Figures 5 to 8 are MCMC traces of the means of the \( \{ \eta \} \) matrix. The centre line is the sample average for the true underlying \( \{ \eta \} \) matrix, used to generate the increments, \( \{ Z \} \), which are conditioned upon, and the dotted lines, above and below the centre line are the upper and lower 95% CI for the 5 million MCMC sample averages, respectively. This plot demonstrates that the CI captures the actual mean and also the theoretical unconditional mean, which is zero. In fact, the conditional mean could also be zero. These graphs also show that the chain is well mixed well before the one millionth MCMC step. The mixing times appear slower for the more dependent chains, revealing that as \( H \) increases, so does the required mixing time. It can also be observed that the mean of the \( \{ \eta \} \) matrix takes many MCMC steps to change for higher \( H \) values. This is reasonable as the dependence effectively prevents dramatic changes from one step to another, since the distribution of each \( \eta_{ij} \) relies heavily on the row values of the other \( \{ \eta \} \).

Figure 9 is a mean trace plot for FATGBM with an underlying FBM covariance function. This plot shows that the chain mixes considerably faster because the FBM covariance function results in a less dependent FATGBM process.

Figure 10 is a plot of the MCMC trace for the mean of the \( \{ X \} \) vector arising from FATGBM 2. The centre line in this graph is the sample average of the five million means. It also reveals a conditional mean of approximately zero. This graph suggests faster mixing for FATGBM 2 than for FATGBM. This is because FATGBM 2 only has a vector of length \( n = 100 \) to update as opposed to a matrix of size \( \nu \times n = 4 \times 100 = 400 \). The plots also reveal the effect that increased dependence has on increasing the variability of the mean, as would be expected,
since the theoretical variance increases as dependence increases, but is bounded below 1. In addition to analysing the effect of LRD on mixing times, we also wanted to observe if mixing times would be affected by increasing the FAT tail parameter \( \nu \). Figure 11 shows that mixing time is not affected by increasing this parameter, but volatility in the mean is. This observation is consistent with what we would expect, since increasing \( \nu \) only changes the likelihood of extremal values regardless of the other values, hence not prohibiting the mean from moving about from one MCMC iteration to the next.

A further question we wish to assess is what the effect of conditioning on the
distribution of \( \{\eta\} \) and \( \{X\} \) has been. Specifically, we would like to answer the question: “is the conditional distribution significantly different from the unconditional distribution?” If the answer is yes, then we would expect the conditional option price to differ from the unconditional option price, motivating the purpose for the next chapter.

This question leads naturally to the question of how can we determine whether the two high-dimensional distributions, conditional and unconditional, differ? Since we know that the unconditional distribution of \( \{\eta\} \) and \( \{X\} \) is multivariate Normal, \( N(0, \Sigma) \), the question then becomes one of assessing whether the conditional
The distributions \( f(\{\eta\}|\{Z\}) \) and \( f(\{X\}|\{Z\}) \) are also multivariate Normal with covariance matrix \( \Sigma \). Of course, in a simple univariate setting, visual tools such as quantile-quantile plots would provide a simple way to check whether the distributions agree or not, but in the current high-dimensional setting, the problem becomes considerably more difficult. One approach is to construct a low-dimensional measure of the difference between the two distributions, and then to use a low-dimensional graphic such as a quantile-quantile plot to investigate the nature of the difference between the two distributions. In order to do this, we need a measure of “distance” between the \( d \)-dimensional theoretical unconditional distribution and the samples arising from the \( d \)-dimensional conditional distribution. The simplest approach would be to calculate the Euclidean distance.
between a sample of the conditional $d$-dimensional vector $x$ and the mean of the theoretical $d$-dimensional vector $y$:

$$d(x, y) = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2},$$

but the problem with this method is that it ignores the correlation structure inherent in the $x$ values and assumes that the $x$ values are independent. By not accounting for the correlation structure, the Euclidean distance measure places higher weight or emphasis on highly correlated values. Hence, we consider the Mahalanobis distance measure (Mahalanobis, 1936), which more appropriately accounts for the correlation structure intrinsic to the data. Consider a multivariate vector $x = (x_1, x_2, ..., x_d)^T$ from a joint distribution with mean
\( \mu = (\mu_1, \mu_2, \ldots, \mu_d)^T \) and covariance matrix \( \Sigma \). The Mahalanobis distance is defined to be

\[
D_M(x) = \sqrt{(x - \mu)^T \Sigma^{-1} (x - \mu)}.
\]  

Now consider a \( d \)-dimensional random vector \( X \sim N(0, \Sigma) \). Then the squared Mahalanobis distance has a Chi-square distribution with \( d \) degrees of freedom, \( D_M(X) \sim \chi^2(d) \). We will use this result to construct Quantile-Quantile plots for both \( \{\eta\} \) and \( \{X\} \) conditional on the vector, \( \{Z\} \), generated using our MCMC algorithms for various values of \( H \). Each Quantile-Quantile plot for the FATGBM case describes 1000 independent realisations, each of length \( d = 100 \) and each based on 100,000 MCMC steps, while each Q-Q plot for the FATGBM 2 case describes 500 independent realisations, each of length \( d = 100 \) and each based on 50,000 MCMC steps. The Quantile-Quantile plot of squared Mahalanobis distances is used to measure the dissimilarity of the realisations arising from the conditional distribution to a Multivariate Normal with mean zero and covariance matrix defined for either FATGBM or FATGBM 2, respectively. Quantile-quantile plots of the squared Mahalanobis distances versus the quantiles of a \( \chi^2_{100} \) distribution for FATGBM with \( H = 0.9, 0.8 \) and 0.6 are given in Figure 12 (left column of plots), while corresponding plots for FATGBM 2 with the same \( H \) values are presented in Figure 13 (left column). In each of Figures 12 and 13, the right column of plots displays analogous squared Mahalanobis distance Q-Q plots for sets of 100 \( \eta \)'s generated unconditionally. The left columns of Q-Q plots in each of Figures 12 and 13 reveal evidence of significant curvature, which suggests
that the conditional distribution is far from Multivariate Normal with the specified covariance function. By way of contrast, the Q-Q plots in the right columns of those figures reveal no evidence of departure from multivariate Normality. The plots for the FATGBM 2 case are qualitatively similar to those for the FATGBM case. It is noticeable that as $H$ decreases, the extent of the curvature — that is, the extent of the departure from normality — also decreases, but curvature is visible in all of the Q-Q plots for which conditional distributions are considered. Figures 14 and 15 show that both FATGBM and FATGBM 2 MCMC algorithms produce independent Normal random variables when $H = 0.5$, which is further evidence that the MCMC algorithm is producing the correct desired distribution in each case.

The final series of plots depict the autocorrelation functions from the one millionth MCMC samples. The Autocorrelation Function (ACF) appear to be consistent with the assumed dependence structure — as $H$ increases, the ACF dampens down to zero at a slower rate.

7. Conclusion

In this chapter, we implemented a new and novel approach to incorporate price path history so that we are able to simulate future price paths in a long range dependent asset price model. We constructed a Markov Chain Monte Carlo algorithm to sample from the complicated conditional distribution of interest, namely, the distribution of unobserved Activity Times, conditional on an observed price path history. This method allowed us to observe the characteristics of this
conditional distribution and conclude as to whether conditioning on price path history significantly alters the distribution of the underlying Activity Times. Our conclusion was that conditioning on price path history does have a significant effect on the Activity Time samples, which will in turn have an effect on the simulation of future Activity Time paths. This finding motivates our progression onto the final chapter, where we shall describe and discuss the effect of these conditional paths on the pricing of options.
**Figure 12**

FATGBM - Mahalanobis QQ plots to assess $MN(0, \Sigma)$ for conditional (left column) and unconditional (right column) $\eta$. Left column $\eta$'s each based on 100,000 MCMC iterations.
FATGBM 2 – Mahalanobis QQ plots to assess $\text{MN}(0, \Sigma)$ for conditional (left column) and unconditional (right column) $\eta$'s. Left column $\eta$'s each based on 50,000 MCMC iterations.
7. CONCLUSION

Figure 14
FATGBM, $H=0.5$, 100,000 MCMC steps, Normal Q-Q plot

---

Figure 15
FATGBM 2, $H=0.5$, 50,000 MCMC steps, Normal Q-Q Plot

---

Figure 16
1 millionth MCMC sample of $f(\eta|Z)$, Cov 1, Eta length=100
ACF $1$, $H=0.6$

---

Figure 17
1 millionth MCMC sample of $f(\eta|Z)$, Cov 1, Eta length=100
ACF $1$, $H=0.9$

---

Figure 18
1 millionth MCMC sample of $f(\eta|Z)$, Cov 1, Eta length=100
ACF $1$, $H=0.9$

---

Figure 19
1 millionth MCMC sample of $f(\eta|Z)$, Cov 1, Eta length=100
ACF $1$, $H=0.9$
Figure 20

1 millionth MCMC sample of $f(X|Z)$, Cov 1, $X$ length=100

ACF $1, H = 0.6$

ACF $1, H = 0.8$

ACF $1, H = 0.9$

ACF $1, H = 0.95$
CHAPTER 5

Long Range Dependence and Option Pricing

1. Introduction

In this chapter, we combine the results from the preceding chapters to implement an option pricing algorithm which is capable of pricing path dependent options under the two long memory processes we have considered in this thesis, FATGBM and FATGBM 2, with the ability to condition on price path history. Our goal in this chapter is to implement the entire option pricing algorithm to study the effect of long range dependence on option prices. We would like to answer the question of whether – and by how much – conditioning on the price path history affects the option price for long memory processes?

There is an extensive list of stochastic models used to price options in the current mathematical finance literature. These models include Jump Diffusion models, Stochastic Volatility models and Long Memory Processes such as FATGBM or Fractional Brownian Motion. Each type of model has been justified in the context of modeling particular kinds of observed market behavior. For example, Jump Diffusion models are able to reproduce the volatility smile, or skew, as a result of the jumps, which produce a distribution with heavy tails (see Cont and Tankov, 2004). This type of model is consistent with the theory that the market factors
in a fear of crashes (termed “crashaphobia”). Note that we will return to a dis-
cussion of volatility smiles later in the chapter. Stochastic Volatility models are
also able to reproduce the volatility smile and skew observed in option price data.
These types of models are consistent with the leverage effect, that is, the observed
negative correlation between asset prices and volatility (for instance, when asset
prices are low, volatility tends to be high) (see Cont and Tankov, 2004). Long
memory processes, of which Fractional Brownian Motion is a common example,
are able to model the dependence structure of asset returns data, but option
pricing for these models is very challenging (see Rostek, 2009) and it is not yet
clear that their inherent complexity is offset by the addition of significant value
to practitioners in the derivative pricing world, especially given that models with
independent increments seem perfectly capable of delivering key features desired
in a model by industry practitioners, such as an arbitrage-free and complete mar-
ket model that allows them to efficiently calibrate to volatility surfaces.

The huge literature on each of these types of models details closed form solutions,
procedures and approximation techniques to pricing a vast array of derivatives
under these types of models. This chapter will make a significant contribution to
the existing literature by quantifying the effect of long memory on the price of
options. In particular, we will compare this effect to the resultant effect of heavy
tails on option prices. We will then conclude by comparing the implied volatil-
ity surfaces produced by FATGBM and FATGBM 2, conditional on price path
history, with implied volatility surfaces arising from unconditional price paths,
thereby addressing the question “does long memory impact implied volatility surfaces, and, further, are we able to calibrate our models to implied volatility surfaces better if we build in long memory?”

2. The Complete Pricing Algorithm

In order to address the questions posed in the introduction, we return to the findings from Chapter 4: if we condition on price path history, the distribution of future price paths is significantly altered. That is, if we incorporate past memory into the future simulation of our long memory process, then the resultant conditional distribution is different to the unconditional distribution. This result provides us with the motivation to continue pursuing the question of whether long memory matters.

In Chapter 2 we noted that the risk-neutral price process under FATGBM is,

\[
S_t^* = S_0 e^{rt - \frac{1}{2} \sigma^2 T_t + \sigma W(T_t)}.
\]

If we introduce a risk-free asset which represents the money market account, defined as \( B_t = e^{rt} \), then \( \frac{S_t^*}{B_t} \) is a martingale under the risk-neutral price process, and the market model is arbitrage free. In some special circumstances, in particular if the derivative pays off when some level of volatility has been reached, we may be able to replicate the value of the derivative, but in most situations this will not be possible. Nevertheless, we will proceed by valuing options under FATGBM by computing the expectation of a contingent claim, discounted by the risk-free asset, under the risk-neutral density. In other words, we shall set the derivative
price, $V_0$, under FATGBM to equal the following expectation,

$$V_0 = E \left[ \frac{f(S_T^*)}{B_T} \right].$$

Note, however, that this risk-neutral expectation involves the specification of future Activity Time paths under the real-world probability measure. Since long memory is embedded into the Activity Time process, it makes sense to use this memory to specify future Activity Time paths and price contingent claims incorporating this feature. Intuitively, long memory processes are designed to replicate the volatility clustering we observe in real world data. This means that by conditioning on historical volatility, we are able to price contingent claims by placing higher weight on the more likely Activity Time paths, given the past. A result from the previous chapter shows us that the conditional distribution $f(\{\eta\}|A)$ differs from the unconditional distribution $f(\{\eta\})$. This result tells us that we can compare the unconditional derivative price, $V_0$, to the conditional derivative price denoted $V_0^*$, defined below, to observe if conditioning can have an effect on the price of options. We define the conditional derivative price $V_0^*$ to be

$$V_0^* = E \left[ \frac{f(S_T^*)}{B_T} \mid A \right],$$

where $A$ is some subset of price path history $\{S_0, \ldots S_a\}$, for some $a > 0$. We can then use the set $A$ to estimate the Hurst parameter $H$, as well as the parameters $\mu$ and $\sigma$ as discussed in the preceding chapter. We now outline the process required to calculate the expectations given by (2.2) and (2.3) for any derivative payoff, including payoffs which are dependent on the price path such as barrier options.
<table>
<thead>
<tr>
<th>Simulate</th>
<th>1</th>
<th>construct ( {\eta} ) or ( {X} ), a Gaussian random matrix or vector of length ( k ), using the Durbin-Levinson algorithm.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
<td>Use ( {\eta} ) or ( {X} ) to construct ( k ) Activity Time increments, ( T_1, \ldots, T_k ).</td>
</tr>
<tr>
<td>Tree</td>
<td>1</td>
<td>Sum Activity Time increments ( T = \sum_{i=1}^{k} T_i ).</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Divide ( T ) into ( \lfloor nT \rfloor ) steps of equal length ( \left( \frac{1}{n} \right) ).</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>Find the resultant clock times ( c_1, \ldots, c_{\lfloor nT \rfloor} ) such that ( c_i := \min{t : T_i &gt; \frac{i}{n}} ).</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>Build the Binomial Tree with ( \lfloor nT \rfloor ) steps, each of size ( \frac{1}{n} ) and compute the price of the asset at each step of the tree.</td>
</tr>
<tr>
<td>Price</td>
<td>1</td>
<td>Solve for the payoff at the end points of the tree and recursively work backwards through the tree to compute the price at the beginning node.</td>
</tr>
<tr>
<td>Repeat</td>
<td>1</td>
<td>Numerically approximate ( E\left[ \frac{f(S_T)}{B_T} \right] ) by repeating this procedure a large number of times and computing the average price.</td>
</tr>
</tbody>
</table>
Numerical approximation of $E\left[ \frac{f(S_T)}{B_T} \big| A \right]$

<table>
<thead>
<tr>
<th>Data</th>
<th>1</th>
<th>Use log($S_0$), ..., log($S_{-a}$) to estimate the parameters $\mu$ and $\sigma$ (see Chapter 4 for more details).</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
<td>Construct the vector of normalised historical prices ${Z} = {W(T_i)}$ for $i = 0, \ldots, a$.</td>
</tr>
<tr>
<td>MCMC</td>
<td>1</td>
<td>Use the vector ${Z}$ constructed in the previous step to run the MCMC algorithm. Stop the algorithm when the chain has mixed (100,000 iterations is usually enough, with the required number of steps impacted by the extent of the long range dependence).</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Obtain a sample from the conditional distribution $f({\eta}</td>
</tr>
<tr>
<td>Simulate</td>
<td>1</td>
<td>Construct ${\eta^<em>}$ or ${X^</em>}$ a Gaussian random matrix or vector of length $k$, conditioning on the ${\eta}$ matrix or ${X}$ vector generated in the previous step.</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Use ${\eta^<em>}$ or ${X^</em>}$ to construct $k$ conditional, future Activity Time increments $T_1, \ldots, T_k$.</td>
</tr>
<tr>
<td>Price</td>
<td>1</td>
<td>Build the Binomial Tree and Price (See table above for more detail).</td>
</tr>
<tr>
<td>Repeat</td>
<td>1</td>
<td>Numerically approximate $E\left[ \frac{f(S_T)}{B_T} \big</td>
</tr>
</tbody>
</table>
3. Impact of Conditioning

We now consider the impact of conditioning on a price path history on the pricing of options. Recall that a European call option is defined to have the payoff $(S_T - K)^+$ and a European put option is defined to have the payoff $(K - S_T)^+$, where both payoffs occur at time $T$, and $K$ is a known constant. American Options allow for early exercise, and while we are able to compute an upper bound for these options using our Binomial Tree, we prefer to focus on an analysis of conditioning on the European option price first. We will quantify the effect that conditioning has on option prices by computing the difference between the conditional and unconditional option price, $E[(S_T^A) | A] - E[(S_T^B) | B_T]$ for varying $T$ and for a particular price path history $A$. Sometimes this quantity will be negative and other times it will be positive, so we might prefer to consider the absolute difference, without loss of generality. The following perspective plots depict the price difference between the conditional and unconditional Call and Put option prices for the two processes FATGBM and FATGBM 2. Each plot prices the options for varying initial stock prices and strike prices, option maturities, $T$ and Hurst parameter $H$. Perspective plots are based on the following inputs into the model unless otherwise stated:
These parameters typically reflect a time increment of 1 day, so $T = 90$ would represent a 3 month option. Figures 1 and 3 depict the price differences for FATGBM for European Put and Call options and for varying Hurst parameter values, $H$, with analogous price differences for FATGBM 2 depicted in Figures 2 and 4. For the sake of brevity and to focus the discussion, we have chosen not to include every case we considered for each model because our investigations revealed that both models achieve qualitatively similar results, and any differences in the resultant perspective plots tended to be due to the effect of conditioning on different price path histories. For example, Figures 1 and 3 are produced from FATGBM increments which involved the simulation of a single FATGBM price path history, whereas Figures 2 and 4 are the result of FATGBM 2 increments which involved the simulation of a single FATGBM 2 price path history.

These perspective plots provide evidence to support the conclusion that conditioning on price path history has a significant effect on the option prices, even

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Price</td>
<td>$S_0 = 0, ..., 100$</td>
</tr>
<tr>
<td>Strike</td>
<td>$K = 0, ..., 100$</td>
</tr>
<tr>
<td>rate</td>
<td>0.0003</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>0.01</td>
</tr>
<tr>
<td>$\nu$</td>
<td>4</td>
</tr>
</tbody>
</table>
when there is weak dependence. We observe that price differences are greatest for long dated at-the-money options and that the longer the maturity date $T$, the greater the price difference. The reason for this phenomenon is because given a maturity time $T$, deep in-the-money and deep out-of-the-money options (that is, options which have high/low chance of being exercised due to the large difference between the strike and current asset price, $|S_0 - K|$) are very unlikely to end up in the opposite state at maturity, irrespective of the conditioning – the opposite state being, for example, out-of-the-money for a currently in-the-money option. However, at-the-money options can be greatly affected by price path history, since the dependence can place higher/lower weight on larger absolute movements in the stock price than would be the case without conditioning on the price path history, which in turn causes more positive/zero value states at maturity, regardless of the type of the option. Of course, the longer the maturity time, the more chance in-the-money options can end up out-of-the-money, and vice versa, if the impact of history is to cause future volatility to fluctuate more. In other words, we have observed that conditioning on price path history has a material effect on the future evolution of the volatility of stock price movements under each of FAT-GBM and FATGBM 2. Probabilistically speaking, we can explain the observed surfaces through the following conjecture.

**Conjecture 5.1.** Consider a given $S_0 \approx K$ and a price path history, $A$. If

$$P(T_i|A > \epsilon) < P(T_i > \epsilon)$$
for some \( \epsilon > 0 \) and any \( i \), then

\[
\]

Moreover, when \( |S_0 - K| > \xi \),

\[
E[(S_T - K^+)|A] \approx E[(S_T - K^+)]
\]

for some \( \xi > 0 \), depending on the price history, \( A \), the level of dependence, \( H \), and the maturity of the option, \( T \).

The same conjecture is made for a Put option and for the case where \( A \) causes larger movements in future volatility.

Figures 5 to 8 are a result of price differences from the FATGBM model with Hurst parameter equal to 0.8, while Figures 9 and 10 show analogous plots for the case when \( H = 0.9 \). These plots demonstrate clearly that the option price difference increases as the maturity time increases. The plots also show a greater absolute price difference from that displayed in Figures 1 to 4, where the Hurst parameter is equal to 0.6, reflecting weaker dependence. Despite this evidence, it is difficult to generalise our observation to claim that an increasing Hurst parameter results in greater absolute price differences.

Figure 11 is a plot of price differences arising from the FATGBM 2 model with Hurst parameter 0.9. This figure is consistent with the conjecture that increasing \( H \) increases the absolute price difference, since the greatest price difference in this case is very high, exceeding $1. However, Figures 9, 10 and 12 tell a very different
story. These figures show price differences arising from the FATGBM model with Hurst parameter also equal to 0.9, but these figures depict a smaller absolute price difference than those observed in the earlier sets of plots for smaller Hurst values. The other interesting thing about these plots is that the Figures 9 and 10 show a positive price difference whereas Figure 12 shows a negative price difference. All three plots are based on the same model with the same price history. It appears that the conditional volatility outlook over the short time horizon fluctuates more than the unconditional volatility, whereas the conditional volatility over the long time horizon fluctuates less than the unconditional volatility. These observations impact generalisations of the conjecture stated earlier, and indicate a need for further investigation that lies outside the scope of the current study, but which will form the basis for future research.
FIGURE 1
European Put, rate=0.0003, sigma=0.01, T=10, H=0.6

European Call, rate=0.0003, sigma=0.01, T=10, H=0.6

FIGURE 2
European Put, rate=0.0003, sigma=0.01, T=200, H=0.6, FATGBM2

European Call, rate=0.0003, sigma=0.01, T=200, H=0.6, FATGBM2
Figure 3
European Put, rate=0.0003, sigma=0.01, T=500, H=0.6

Figure 4
European Put, rate=0.0003, sigma=0.01, T=1000, H=0.6, FATGBM2

European Call, rate=0.0003, sigma=0.01, T=500, H=0.6

European Call, rate=0.0003, sigma=0.01, T=1000, H=0.6, FATGBM2
FIGURE 5
European Put, rate=0.0003, sigma=0.01, T=10, H=0.8
European Call, rate=0.0003, sigma=0.01, T=10, H=0.8

FIGURE 6
European Put, rate=0.0003, sigma=0.01, T=50, H=0.8
European Call, rate=0.0003, sigma=0.01, T=50, H=0.8
Figure 9

European Put, rate=0.0003, sigma=0.01, T=10, H=0.9

European Call, rate=0.0003, sigma=0.01, T=10, H=0.9

Figure 10

European Put, rate=0.0003, sigma=0.01, T=50, H=0.9

European Call, rate=0.0003, sigma=0.01, T=50, H=0.9
Figure 11
European Put, rate=0.0003, sigma=0.01, T=500, H=0.9, FATGBM2

European Call, rate=0.0003, sigma=0.01, T=500, H=0.9, FATGBM2

Figure 12
European Put, rate=0.0003, sigma=0.01, T=1000, H=0.9

European Call, rate=0.0003, sigma=0.01, T=1000, H=0.9
FIGURE 13
European Put, rate=0.0003, sigma=0.01, T=10, H=0.95
European Call, rate=0.0003, sigma=0.01, T=10, H=0.95

FIGURE 14
European Put, rate=0.0003, sigma=0.01, T=50, H=0.95
European Call, rate=0.0003, sigma=0.01, T=50, H=0.95
**Figure 15**
European Put, rate=0.0003, sigma=0.01, T=200, H=0.95
European Call, rate=0.0003, sigma=0.01, T=200, H=0.95

**Figure 16**
European Put, rate=0.0003, sigma=0.01, T=700, H=0.95
European Call, rate=0.0003, sigma=0.01, T=700, H=0.95
Figures 13 to 16 reveal a very interesting picture. These figures depict price differences for the FATGBM model for which $H = 0.95$. While at-the-money options result in a positive price difference, in-the-money and out-of-the-money options result in a negative price difference, with deep in-the-money and deep out-of-the-money exhibiting minimal to no price difference. Figures 13 and 14 have a similar interpretation to that arising from Figures 9 and 10, where the maturity time is short. However, Figure 16 ($T = 700$) is a perspective plot of what occurs before the maturity time increases to $T = 1000$. For this latter maturity time ($T = 1000$), we observe that the perspective plot inverts as in Figure 12. Patterns in these figures may again be explained by fluctuating rates in volatility. For time periods when the conditional volatility is low, the distribution may be more leptokurtic, resulting in higher conditional prices for at-the-money options but lower conditional prices for in/out-of-the-money options. For long maturity times, the conditional distribution of aggregated volatility may be similar to the unconditional distribution, though not as spread out, resulting in lower conditional prices for both at-the-money and in/out-of-the-money options, according to the conjecture stated earlier in the chapter.

So far we have observed perspective plots of price differences arising from conditioning on price path history. The price differences have at some points been as low as 5c or greater than $1 for options on assets worth $100. So, naturally, one might ask how significant these amounts are, particularly with regard to the price differences that arise from changing the heavy tail parameter, $\nu$. Figure 17 shows the price differences arising from two different sets of unconditional prices. The
purpose of this figure is to show that the price differences shown above, between conditional and unconditional cases, are not the result of chance or random error, which may arise from a numerical approximation of the expected payoff under the risk-neutral distribution, but rather are of a more significant size. The scale on this figure shows that there is no effective price difference between the two numerical approximations.

Next, we consider the effect of the heavy tail parameter, $\nu$. Figures 18 and 19 show the unconditional price differences calculated for the FATGBM 2 model between $\nu = 5$ and $\nu = 10$ at maturities $T = 90$ and $T = 500$, respectively. These figures show a maximum absolute price difference of around 0.10, which is much smaller than the absolute price difference observed in the equivalent conditional
5. LONG RANGE DEPENDENCE AND OPTION PRICING

Figure 18

Price difference between $n=5,10$ FATGBIV $2$, $T=90$, $H=0.75$

Figure 19

Price difference between $n=5,10$ FATGBM $2$, $T=500$, $H=0.75$
versus unconditional plot (Figure 20), which depicts a maximum absolute price difference of more than 0.40 for a maturity time of 3 months. This suggests quite clearly that conditioning has more of an effect on option prices than heavy tails. The other observation to make is that as maturity time, $T$, increases, the absolute price difference does not appear to increase for changing values of the heavy tail parameter $\nu$, whereas conditioning seems to have a clear effect on price differences at $T$ increases. This observation indicates that models which build in past behaviour are likely to have more of an impact on option prices than models which build in a higher possibility of extremal events, unconditional on the past. This motivating piece of evidence suggests that the volatility surface discussed in the next section will be somewhat altered as a result of conditioning on the past.
A comprehensive study of this effect of conditioning lies beyond the scope of this thesis, but is an interesting topic for ongoing research.

4. Volatility Smiles

We now turn to a brief discussion on volatility smiles. Recall that the risk-neutral price process that underlies the Black-Scholes option pricing formula is given by:

\[(4.1) \quad S_t = S_0 e^{r t - \frac{1}{2} \sigma^2 t + \sigma W(t)}.\]

A significant drawback of this model is that it assumes constant volatility for the log returns, \(\log\left(\frac{S_t}{S_{t-u}}\right)\), where \(0 < u < t\). Liquid European option prices can be used to imply out a volatility parameter under the Black-Scholes option price formula, for various strike prices and times to maturity. This computation can be achieved using an optimisation technique, such as a Newton-Raphson approach, to iteratively solve for the volatility parameter estimate, \(\hat{\sigma}\), for which the market option price satisfies the Black-Scholes European option price formula (for option maturity \(T\)). This latter formula is given by

\[(4.2) \quad c_{mp} = S_0 \phi\left(\frac{\log\left(\frac{S_0}{K}\right) + (r + \frac{1}{2} \hat{\sigma}^2) T}{\hat{\sigma} \sqrt{T}}\right) - K e^{-rT} \phi\left(\frac{\log\left(\frac{S_0}{K}\right) + (r - \frac{1}{2} \hat{\sigma}^2) T}{\hat{\sigma} \sqrt{T}}\right),\]

where \(\phi\) denotes the standard Normal density function.

The implied volatility surface constructed from liquid European option prices is simply a 3-dimensional plot of the implied volatility \(\hat{\sigma}\), for a range of stock prices, \(S_0\), strike prices, \(K\), and maturity times, \(T\). The surface often depicts a “smile” or skew which indicates that the options market does not value options
with the assumption of constant volatility. See Figure 21 for an example of such a surface produced by Jex et al. (1999).

This implication is also consistent with the volatility observed in real asset price data. Many models are calibrated to the liquid options market as a means for completing an incomplete market. This strategy matches the number of traded assets to the same number of sources of randomness (Bingham and Kiesel, 2004). The volatility smile is well known to be a result of the October 1987 market crash because post crash Black-Scholes implied volatilities for S&P 500 index options have consistently exhibited smile effects (Jackwerth and Rubenstein, 1996). The smile and skew are usually present for shorter maturity options than longer maturity ones. The smile represents a symmetry between the high implied volatilility
arising from low strike options (deep in-the-money calls and deep out-of-the-money puts) and the high implied volatility arising from high strike options (deep in-the-money puts and deep out-of-the-money calls), whereas the volatility skew represents an asymmetry, with high strike options having a lower implied volatility than low strike options. The explanation for this phenomenon is that the market places higher probability on events which may lead to a market crash. Since the Black-Scholes model results in a Log Normal distribution for asset returns, this means that any model resulting in a heavy-tailed distribution, such as a Student’s-t distribution, for log asset returns would have the effect of producing a volatility smile, whereas a model which allows for negative jumps would have the effect of producing a volatility skew. The Black-Scholes model is thus known for underpricing deep in-the-money and out-of-the-money call options and over-pricing at-the-money call options.

We have seen in the preceding section how the value of an option can change when historical prices or historical volatility is incorporated in its calculation. If the market suffers from “crashaphobia” then this could actually be the market’s way of accounting for historical volatility into the future. We now wish to investigate if FATGBM has the potential for modeling implied volatility surfaces observed in real world data by incorporating price path history. It would be of interest to investigate the implied volatility surfaces conditioned on real world data to see if it is consistent with the market. Such an extensive investigation lies beyond our current scope, but we conclude the thesis by briefly investigating FATGBM’s
potential for doing this by analysing volatility smiles resulting from theoretical prices generated from the FATGBM model. Implied volatility for FATGBM is also discussed in Heyde and Gay (2002). We will be extending their analysis by comparing the unconditional volatility surface to the conditional volatility surface generated from the FATGBM and FATGBM 2 models.

Figure 22 is a plot of implied volatility surfaces for unconditional FATGBM prices for varying values of the heavy-tail parameter, $\nu$, and the Hurst parameter, $H$. The inputs into the model are:

<table>
<thead>
<tr>
<th>Price $S_0 = 30$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strike $K = 24, ..., 36$</td>
</tr>
<tr>
<td>rate $0.05/12$</td>
</tr>
<tr>
<td>$\sigma$ $0.3/\sqrt{12}$</td>
</tr>
</tbody>
</table>

These parameters reflect a time increment of 1 month. Viewing Figure 22 by row, from left to right, the parameter $\nu$ increases, while by column, from top to bottom the parameter $H (= 1 - \alpha)$ increases. Figure 22 reveals that varying $H$ has little to no effect on the volatility surface, but that increasing $\nu$ has the clear effect of flattening the smile. This effect occurs because log returns under FATGBM have a $t$-distribution with $\nu$ degrees of freedom, thus for large $\nu$, the distribution is approximately Normal.
We now consider the implied volatility surface arising from conditioning on price path history for varying values of $H$. We will not present results of the conditional implied volatility surfaces for varying values of $\nu$ because as we have observed in the previous section, $H$ dominates changes in the option price, so conditioning will still cause a smile effect even for large values of $\nu$.

Figures 23 to 28 are implied volatility surfaces arising from the FATGBM and FATGBM 2 models, for various $H$. The parameter inputs are exactly the same as those used to create the implied volatility surfaces of Figure 22. FATGBM and FATGBM 2 produce similar implied volatility plots, any small differences arising from the differing price path histories between the two processes. Conditioning appears to change both the implied volatility and the surface. While a volatility smile still exists for short-dated options, it appears to flatten out after a longer period of time. Further, after a certain maturity time, the volatility looks constant across the range of strike prices, but not across maturity times. This observation suggests that by conditioning on price path history, we may be able to calibrate to volatility surfaces more effectively. A further investigation of conditioning on various real world price path histories to calibrate to volatility surfaces could reveal some interesting findings, such as whether the market factors in only recent information or long dated history as well, but a detailed investigation of the effect of conditioning on volatility smiles lies beyond the scope of this thesis and will form the basis for future research.
5. Conclusion

In this chapter we have investigated the effect of using price path history on option prices. We have observed that price path history dictates the evolution of future volatility under FATGBM and FATGBM 2, thus materially altering the option price. We have noticed that this effect is most pronounced for long dated at-the-money options, but can result in alternative price difference patterns for a variety of maturities when the volatility fluctuates over a time period. A major finding in this chapter is that Long Range Dependence affects option prices more than do heavy tails, a result which has the subsequent effect of changing the surface of implied volatilities. This change in surface is a very promising direction for further research into calibrating Long Range Dependent models to implied volatility surfaces by incorporating price path history.
CHAPTER 6

Conclusion

In this thesis we examined four separate problems related to the pricing of options for a long range dependent stochastic process called Fractal Activity Time Geometric Brownian Motion (FATGBM). The thesis has been written for an audience with a solid background in probability and statistics and a basic knowledge of mathematical finance. The content of the thesis comprises 6 distinct chapters (including this conclusion), the first of which stated and discussed the fundamental theories that underlie derivative pricing, namely the no-arbitrage approach to derivative pricing which is consistent with the efficient market hypothesis. In the introductory chapter, we also provided a review of the existing literature on option pricing for long range dependent processes. We discussed that much of this research is focussed on Fractional Brownian Motion, which is not a semi-martingale, and that little attention has been paid to minimally descriptive asset price models which exhibit long range dependence and are also martingales. In this way, this thesis, on option pricing for the process FATGBM, a process which is also a time changed Brownian Motion with dependent increments, builds upon and extends the existing literature by investigating how to price options in a
long range dependent semi-martingale market, and by exploring the effect of long range dependence on option prices, a topic that has received very little attention until now.

Following the introduction into the theory and mathematics that underlie derivative pricing, we introduced in Chapter 2 the concepts of long range dependence and self-similarity. We then proceeded to define the FATGBM model and explored the various features of the model which make that model both interesting and challenging for which to develop option pricing methods. In particular, we highlighted that FATGBM has stationary but dependent increments, Student-$t$ type marginals for the logarithmic process, and logarithmic increments that are uncorrelated. Our major new contribution to the existing literature in Chapter 2 is the proof that the risk-neutral FATGBM process is the FATGBM process under a non-unique measure change. This result means that although an equivalent martingale measure exists for the discounted price process resulting in an arbitrage-free market, this equivalent martingale measure is not necessarily the minimal equivalent martingale measure. Furthermore, we showed that while there is no standard martingale representation for FATGBM allowing for the replication of contingent claims, there is a representation which allows for the hedging of a Timer Call. We also highlighted future research which will stem from this chapter, where we wish to investigate the extension of the paper by Carr et al. (2001) which discusses replication strategies for the Variance Gamma process. It will also be of interest in the future to explore the question of whether the
equivalent martingale measure proposed in this chapter is in fact the minimal equivalent martingale measure.

Chapter 3 used the fact that we are able to price options using the risk-neutral FATGBM price process in an arbitrage-free manner to develop a unique option price algorithm for FATGBM. This algorithm uses a lattice-based construction to specify a discrete approximation scheme which converges to the theoretical FATGBM price. In this chapter, we detailed the lattice construction and proved that the discrete approximation scheme converges to the risk neutral FATGBM price process. Consequently, we showed that we are able to price path dependent options under FATGBM except in the case where optimal stopping is required, such as is the case with the American Put. In this important case, we proved that the discrete approximation scheme provides an upper bound for the price of an American Put under FATGBM.

In Chapter 4, we investigated the problem of simulating future price paths conditional on a price path history under FATGBM and related processes. We explained that under any long range dependent process model, it is important to utilise past observations in the specification of the distribution of future ones. This chapter introduced a novel approach to solving this problem. More specifically, we specified a Markov Chain Monte Carlo algorithm to sample from the conditional distribution of future price path increments dependent on a given price path history. We showed that our chain converges to the conditional future
price distribution and that the conditional future price distribution is materially different from the unconditional future price distribution. This finding was the impetus for an investigation of the effect of long memory on option prices.

Our final chapter prior to this conclusion focussed on this investigation of the effect of long range dependence on option pricing. The key to carrying out this study was the combination of our results from Chapters 3 and 4. The frameworks and algorithms proposed and developed in those earlier chapters provided us with the tools to first simulate future FATGBM price paths given a specified FATGBM price path history and then to compute option prices for the conditional future FATGBM price path as well as the unconditional future FATGBM price path using our binomial tree algorithm from Chapter 3. Our results showed that long memory can have a significant effect on the option price through consideration of the differences between the unconditional and conditional option prices. We found that conditioning on the past resulted in materially different option prices than those calculated unconditionally. We further showed that long memory can have a more significant effect on the option price than heavy tails do. We then showed that long memory has a significant effect on the volatility surface. This work provides direction for further investigation using real world data. We also showed that option prices are affected by the length of history used in the conditioning on past information for a single price path history. This result indicates that there may be an optimal historical length to consider when computing the conditional option price.
Appendix A: C Code

001 /* C Code for FATGBM simulations */
002
003 /* First load the standard math libraries and the GNU Scientific
004 Library (used for all random number generation in the code) and
005 the Armadillo library (used for all the linear algebra in the
006 code). GSL Reference: GNU Scientific Library Reference Manual -
009 Sanderson. Armadillo: An Open Source C++ Linear Algebra Library
010 for Fast Prototyping and Computationally Intensive Experiments.
012
013 #include <math.h>
014 #include <stdlib.h>
015 #include <gsl/gsl_rng.h>
016 #include <gsl/gsl_randist.h>
017 #include <gsl/gsl_cdf.h>
018 #include <time.h>
019 #include <stdio.h>
020 #include <gsl/gsl_matrix.h>
021 #include <gsl/gsl_blas.h>
022 #include "armadillo"
023
024 using namespace arma;
025 using namespace std;
026
027 void
028 simfatlrd(int *n, int *p, int *nu, double *alpha, char **covf,
029 double *alpha, char **covf,
029 double *out)
030 {
031 /* Implementation of Heyde's FATGBM process: is the length of
032 process to generate; p is the number of such simulations; nu
033 is the number of elements on which T is based; alpha = 1-H
034 and covf determines whether Heyde's covariance function or
035 Fractional Brownian Motion covariance is used */
036 const gsl_rng_type *T;
037 gsl_rng *r;
038 long i, j, k, l, jj;
039 long seed;
040 double *normal;
041 double *acov;
042 double *vee;
043 double *phil;
044 double *phi2;
045 double *output;
046 double *bigout;
047 normal = new double[*n + 2];
048 acov = new double[*n + 1];
049 vee = new double[*n + 1];
050 phil = new double[*n + 1];
051 phi2 = new double[*n + 1];
052 output = new double[*n + 1];
053 bigout = new double[*n + 1];
054 /* select random number generator */

APPENDIX A
T = gsl_rng_default;
* T = gsl_rng_default;
* r = gsl_rng_alloc(gsl_rng_mt19937);
* r = gsl_rng_alloc(gsl_rng_mt19937);
* seed = time(NULL);
* seed = time(NULL);
* gsl_rng_set(r, seed);
* gsl_rng_set(r, seed);
* //set seed
* //set seed
* */ All the vectors necessary are allocated dynamically. */
* */ All the vectors necessary are allocated dynamically. */
* */ * sigma2 = *sigma * (*sigma); In this code sigma is taken without
* */ * sigma2 = *sigma * (*sigma); In this code sigma is taken without
* * loss of generality to be 1
* * loss of generality to be 1
* */ */ autcovariance function for this particular H. */
* */ autcovariance function for this particular H. */
* if (**covf == 'A') {
* if (**covf == 'A') {
* */ A is the code to use for Heyde’s covariance function */
* */ A is the code to use for Heyde’s covariance function */
* for (i = 0; i <= *n; i++)
* for (i = 0; i <= *n; i++)
* acov[i] = (pow((double) (1 + i * i), -*alpha/2));
* acov[i] = (pow((double) (1 + i * i), -*alpha/2));
* }
* }
* if (**covf == 'B') {
* if (**covf == 'B') {
* */ B is the code to use for Fractional Brownian Motion
* */ B is the code to use for Fractional Brownian Motion
* */ */
* */ */
* for (i = 0; i <= *n; i++)
* for (i = 0; i <= *n; i++)
* acov[i] = (pow((double) (i + 1), 2 * (1 - *alpha)) - 2 *
* acov[i] = (pow((double) (i + 1), 2 * (1 - *alpha)) - 2 *
* pow((double) i, 2 * (1 - *alpha)) +
* pow((double) i, 2 * (1 - *alpha)) +
* pow((double) abs(i - 1),
* pow((double) abs(i - 1),
* 2 * (1 - *alpha)))/2;
* 2 * (1 - *alpha)))/2;
* }
* }
* for (l = 1; l <= *p; l++) {
* for (l = 1; l <= *p; l++) {
* */ Initialise the output vector to zero */
* */ Initialise the output vector to zero */
* for (jj = 1; jj <= (*n); jj++) {
* for (jj = 1; jj <= (*n); jj++) {
* bigout[jj] = 0;
* bigout[jj] = 0;
* }
* }
* for (k = 1; k <= *nu; k++) {
* for (k = 1; k <= *nu; k++) {
* */ Generating Normals */
* */ Generating Normals */
* for (i = 1; i <= *nu; i++) {
* for (i = 1; i <= *nu; i++) {
* normal[i] = gsl_ran_gaussian(r, 1);
* normal[i] = gsl_ran_gaussian(r, 1);
* }
* }
* */ Determining the Durbin-Levinson coefficients and
* */ Determining the Durbin-Levinson coefficients and
* the output vector, recursively
* the output vector, recursively
* */
* */
* phi1[1] = acov[1]/acov[0];
* phi1[1] = acov[1]/acov[0];
* phi2[1] = phi1[1];
* phi2[1] = phi1[1];
* vee[0] = acov[0];
* vee[0] = acov[0];
* output[1] = sqrt(vee[0]) * normal[1];
* output[1] = sqrt(vee[0]) * normal[1];
* for (i = 2; i <= *nu; i++) {
* for (i = 2; i <= *nu; i++) {
* phi1[i] = acov[i];
* phi1[i] = acov[i];
* for (j = 1; j <= i - 1; j++)
* for (j = 1; j <= i - 1; j++)
* phi1[i] = phi2[j] * acov[i - j];
* phi1[i] = phi2[j] * acov[i - j];
* phi1[i] = phi2[i]/vee[i - 1];
* phi1[i] = phi2[i]/vee[i - 1];
* vee[i] = vee[i - 1] * (1 - phi1[i] * phi1[i]);
* vee[i] = vee[i - 1] * (1 - phi1[i] * phi1[i]);
* output[i] = sqrt(vee[i - 1]) * normal[i];
* output[i] = sqrt(vee[i - 1]) * normal[i];
* for (j = 1; j <= i - 1; j++) {
* for (j = 1; j <= i - 1; j++) {
* phi1[j] = phi2[j] - phi1[i] * phi2[i - j];
* phi1[j] = phi2[j] - phi1[i] * phi2[i - j];
* output[i] += phi2[i] * output[i - j];
* output[i] += phi2[i] * output[i - j];
* }
* }
* for (j = 1; j <= i; j++)
* for (j = 1; j <= i; j++)
* phi2[j] = phi1[j];
* phi2[j] = phi1[j];
A. APPENDIX A: C CODE

```c
for (i = 1; i <= *n; i++) {
    /* Now form the sum of squares of etas */
    bigout[i] += pow(output[i], 2);
}

for (i = 1; i <= *n; i++) {
    /* Calculate T */
    bigout[i] = ((*nu) - 2)/bigout[i];
}

for (i = 1; i <= *n; i++) {
    /* Output the values in a vector to be partitioned into a matrix in R */
    out[(l - 1) * (*n) + i] = bigout[i];
}

delete[] acov;
delete[] vee;
delete[] phi1;
delete[] phi2;
delete[] output;
delete[] bigout;

void simfatlrd2(int *n, int *p, int *nu, double *alpha, char **covf,
    double *out, double *etas)
{
    /* Implementation of Heyde's FATGBM process: is the length of process to generate; p is the number of such simulations; nu is the number of elements on which T is based; alpha = 1-H and covf determines whether Heyde's covariance function or Fractional Brownian Motion covariance is used */
    const gsl_rng_type *T;
    gsl_rng *r;
    long i, j, k, l, jj;
    long seed;
    double *normal;
    double *acov;
    double *vee;
    double *phi1;
    double *phi2;
    double *output;
    double *bigout;

    normal = new double[*n + 2];
    acov = new double[*n + 1];
    vee = new double[*n + 1];
    phi1 = new double[*n + 1];
    phi2 = new double[*n + 1];
    output = new double[*n + 1];
    bigout = new double[*n + 1];

    /* select random number generator */
    T = gsl_rng_default;
    r = gsl_rng_alloc(gsl_rng_mt19937);
    seed = time(NULL);
    gsl_rng_set(r, seed);
    //set seed
    /* All the vectors necessary are allocated dynamically. */
```

A. APPENDIX A: C CODE 157
/*
 * sigma2 = *sigma * (*sigma); In this code sigma is taken without
 * loss of generality to be 1
 */

/* autocovariance function for this particular H. */
if (**covf == 'A') {
    /* A is the code to use for Heyde’s covariance function */
    for (i = 0; i <= *n; i++)
        acov[i] = (pow((double) (1 + i * i), -*alpha/2));
}
if (**covf == 'B') {
    /* B is the code to use for Fractional Brownian Motion */
    for (i = 0; i <= *n; i++)
        acov[i] = (pow((double) (i + 1), 2 * (1 - *alpha)) - 2 *
                   pow((double) i, 2 * (1 - *alpha)) +
                   pow((double) abs(i - 1),
                   2 * (1 - *alpha)))/2;

    for (l = 1; l <= *p; l++) {
        /* Initialise the output vector to zero */
        for (jj = 1; jj <= (*n); jj++)
            bigout[jj] = 0;

        for (k = 1; k <= *nu; k++) {
            /* Generating Normals */
            for (i = 1; i <= *n; i++)
                normal[i] = gsl_ran_gaussian(r, 1); /*

            Determining the Durbin-Levinson coefficients and
            the output vector, recursively
            */
            phil[l] = acov[l]/acov[0];
            phi2[l] = phil[l];
            vee[0] = acov[0];
            vee[l] = vee[0] * (1 - phil[l] * phi2[l]);
            output[l] = sqrt(vee[0]) * normal[l];

            for (i = 2; i <= *n; i++) {
                phil[i] = acov[i];
                for (j = 1; j <= i - 1; j++)
                    phi1[i] -= phi2[j] * acov[i - j];
                phi1[i] = phil[i]/vee[i - 1];
                vee[i] = vee[i - 1] * (1 - phi1[i] * phi1[i]);
                output[i] = sqrt(vee[i - 1]) * normal[i];
                for (j = 1; j <= i - 1; j++)
                    phi1[j] = phi2[j] - phi1[i] * phi2[i - j];
                output[i] += phi2[j] * output[i - j];
            }

            for (j = 1; j <= i; j++)
                phi2[j] = phi1[j];
        }

        for (i = 1; i <= *n; i++)
            /* Now form the sum of squares of etas */
            bigout[i] += pow(output[i], 2);
    etas[(l-l)*(*nu)*(*n)+(k-l)*(*n)+(i-l)]=output[i];
A. APPENDIX A: C CODE

```c
234     }
235 }
236 /* Calculate T */
237 for (i = 1; i <= *n; i++) {
238     bigout[i] = ((*nu) - 2)/bigout[i];
239 }
240 for (i = 1; i <= *n; i++) {
241     /* Output the values in a vector to be partitioned
242     * into a matrix in R
243     */
244     out[(l - 1) * (*n) + i] = bigout[i];
245 }
246 }
247 delete[] acov;
248 delete[] vee;
249 delete[] phil;
250 delete[] phi2;
251 delete[] output;
252 delete[] bigout;
253 }
254 }
255 void
256 simfatlrdaKint *n, int *p, int *nu, double *alpha, char **covf,
257 double *out)
258 
259 {
260 /* Implementation of Sly's FATGBM process. n is the
261 length of process to generate; p is the number of such
262 simulations; nu is the number of elements on which T is
263 based; alpha = 1-H and covf determines whether Sly's
264 covariance function or Fractional Brownian Motion covariance
265 is used */
266 const gsl_rng_type *T;
267 gsl_rng *r;
268 long i, j, l, jj;
269 long seed;
270 double *normal;
271 double *acov;
272 double *vee;
273 double *phi;
274 double *phi2;
275 double *output;
276 double *bigout;
277 normal = new double[*n + 2];
278 acov = new double[*n + 1];
279 vee = new double[*n + 1];
280 phi = new double[*n + 1];
281 phi2 = new double[*n + 1];
282 output = new double[*n + 1];
283 bigout = new double[*n + 1];
284 /* select random number generator */
285 T = gsl_rng_default;
286 r = gsl_rng_alloc(gsl_rng_mt19937);
287 seed = time(NULL);
288 gsl_rng_set(r, seed);
289 /* set seed */
290 /* All the vectors necessary are allocated dynamically. */
291 /* sigma2 = *sigma * (*sigma); Again, wlog sigma is 1 */
292 /* autocovariance function for this particular H. */
293 if (**covf == 'A') {

/* 
A is the code to use for Sly's covariance function where 
alpha = 1-H */

for (i = 0; i <= *n; i++)
    acov[i] = (pow((double) (1 + i * i), -*alpha));

if (**covf == 'B') {
    /*
    * B is the code to use for Fractional Brownian Motion 
    * covariance function
    */
    for (i = 0; i <= *n; i++)
        acov[i] = (pow((double) (i + 1), 2 * (1 - *alpha)) - 
                   2 * pow((double) i, 2 * (1 - *alpha)) + 
                   pow((double) abs(i - 1), 
                   2 * (1 - *alpha)))/2;
}

for (l = 1; l <= *p; 1++) {
    for (jj = 1; jj <= (*n); jj++) {
        bigoutCjj[ ] = 0;
    }
}

for (i = 1; i <= *ii; i++) {
    normal[i] = gsl_ran_gaussian(r, 1);
}

Determining the Durbin-Levinson coefficients and the
output vector, recursively */

phil[l] = acov[l]/acov[0];

phi2[l] = phil[l];

vee[0] = acov[0];

vee[1] = vee[0] * (1 - phil[l] * phil[l]);

output[1] = sqrt(vee[0]) * normal[1];

for (i = 2; i <= *n; i++) {
    phil[i] = acov[i];
    for (j = 1; j <= i - 1; j++)
        phil[i] -= phi2[j] * acov[i - j];
    phil[i] = phil[i]/vee[i - 1];
    vee[i] = vee[i - 1] * (1 - phi1[i] * phi1[i]);
    output[i] = sqrt(vee[i - 1]) * normal[i];
    for (j = 1; j <= i - 1; j++) {
        phi1[j] = phi2[j] = phil[j] * phi2[i - j];
        output[i] += phi2[j] * output[i - j];
    }
    for (j = 1; j <= i; j++)
        phi2[j] = phi1[j];
}

for (i = 1; i <= *nu; i++) {
    /* To develop Sly's process, need to solve
    F(y)=Phi(x) where F is the cdf of a scaled inverse
    gamma rv */
    double pn = gsl_cdf_ugaussian_Q(output[i]);
    bigout[i] = ((*nu) - 2)/(gsl_cdf_gamma_Pinv(pn,(double)
                     (*nu)/2, (double) 2/(*nu)) * (*nu));
}

for (i = 1; i <= *n; i++) {
    /* Output the values in a
    * vector to be partitioned
    * into a matrix in R */
```c
void fgbmbtCint *nyt, int *n, double *price, double *strike,
     double *rate, double *sigma, double *ct, char **exercise,
     char **corp, double *output) {
/* This function generates a price using a binomial tree. The
value nyt is the number of steps in the tree \((Y_t = nyt * 1/n)\),
exercise is 'E' for a European option and 'A' for an American
option, and corp is 'P' for a put option and 'C' for a call
option, and ct is a vector of clock times corresponding to the
activity times generated in R. In general, sigma = 1. Other
inputs are self-explanatory. */
long i, j;
double u;
double d;
double z;
double *dct;
double *df;
dct = new double[*nyt + 1];
df = new double [*nyt + 1];
for (i = 0; i <= (*nyt - 1); i++) {
    /* clock time increments */
    dct[i] = ct[i + 1] - ct[i];
    df[i] = exp(-(*rate) * dct[i]);
}
/* 'up' and 'down' probabilities */
u = exp((*sigma)/sqrt((double) *n)) -
    log((exp((*sigma)/sqrt((double) *n)) +
    exp(-(*sigma)/sqrt((double) *n)))/2));
d = exp(-(*sigma)/sqrt((double) *n) -
    log((exp((*sigma)/sqrt((double) *n)) +
    exp(-(*sigma)/sqrt((double) *n)))/2));
z = 1.0; /* z=1 for a call option, -1 for a put option */
if (**corp == 'P') {
    z = -1.0;
}
output[0] = 0;
for (i = 0; i <= *nyt; i++) { /* Values at terminal tree nodes */
    output[i] = z * ((*price) * (pow(u, i)) * (pow(d, (*nyt - i))) *
    exp((*rate) * (ct[*nyt])) - (*strike));
    output[i] = (abs(output[i]) + output[i])/2;
}
if (**exercise == 'E') { /* European option */
    for (j = (*nyt - 1); j >= 0; j--) {
        for (i = 0; i <= j; i++) {
            output[i] = (output[i + 1] + output[i]) * df[j]/2;
        }
    }
} else { /* American option */
```

The code provided generates a price using a binomial tree. It takes into account various inputs such as the number of steps in the tree, the exercise type (European or American), the option type (put or call), and clock times corresponding to the activity times generated in R. It also calculates the 'up' and 'down' probabilities and uses them to compute the values at terminal tree nodes.
for (j = (*nyt - 1); j >= 0; j--) {
    for (i = 0; i <= j; i++) {
        output[i] = (z * ((*price) * (pow(u, i)) * 
        (pow(d, (j - i))) * exp((*rate) * (ct[j])) - 
        (*strike)) > (output[i + 1] + output[i]) * df[j]/2) ? 
        (z * ((*price) * (pow(u, i)) * (pow(d, (j-i))* 
        exp((*rate) * (ct[j])) - (*strike))) : 
        (output[i + 1] + output[i]) * df[j]/2;
    }
}
delete[] dct;
delete[] df;
}

void mh(double *x, int *nsim, int *n, int *nu, double *alpha, char **covf, double *wmeans, double *etameans, double *ws, double *etas) {
    /* Metropolis-Hastings algorithm for FATGBM using Heyde’s 
    covariance function (A) or Fractional Brownian Motion 
    covariance function (B). Here, x is the increments on which 
    to condition, nsim is the number of MH iterations, n is the 
    length of x, alpha = 1-H. In the algorithm, values of the w 
    vector and eta matrix are updated and the respective means 
    are collected at each iteration along with the final values 
    of w and eta. */
    const gsl_rng_type *T;
    gsl_rng *r;
    long i, j, k;
    long seed;
    int ii, jj;
    mat rho = randu < mat > (*n, *n);
    rho.ones();
    mat eta = randu < mat > (*nu, *n);
    eta.ones();
    vec w(*n);
    /* select random number generator */
    T = gsl_rng_default;
    r = gsl_rng_alloc(gsl_rng_mt19937);
    seed = time(NULL);
    // set seed
    for (i = 0; i < *n; i++) { /* Set starting value of w=x */
        w(i) = x[i];
    }
    /* Now define the underlying covariance function for the etas. 
    A = Heyde, B = Fractional BM */
    if (**covf == 'A') {
        for (i = 0; i < *n - 1; i++) {
            for (j = i + 1; j < *n; j++) {
                rho(i, j) = (pow((double) (i + (j - i) * (j - i)), 
                -*alpha/2));
                rho(j, i) = rho(i, j);
            }
        }
    } else {
        for (i = 0; i < *n - 1; i++) {
            for (j = i + 1; j < *n; j++) {
                rho(i, j) = (pow((double) (i + (j - i) * (j - i)), 
                -*alpha/2));
                rho(j, i) = rho(i, j);
            }
        }
    }
}
for (i = 0; i < *n - 1; i++) {
    for (j = i + 1; j < *n; j++) {
        rho(i, j) = (pow((double) (j - i + 1), 2 *
            (1 - *alpha)) - 2 * pow((double) (j - i),
            2 * (1 - *alpha)) + pow((double) abs(j - i - 1),
            2 * (1 - *alpha)))/2;
        rho(j, i) = rho(i, j);
    }
}

vec lastw = w;

for (k = 0; k < *nsim; k++) {
    mat xla = eta;
    mat B = rho;
    mat Ca = rho;

    ii = gsl_rng_uniform_int(r, *n);
    jj = gsl_rng_uniform_int(r, *nu);

    mat etasqO = square(eta);
    rowvec sumetasqO = sum(etasqO, 0);

    mat etasq = square(eta);
    rowvec sumetasq = sum(etasq, 0);

    double scale = (pow((double) (*(nu - 2)/sumetasq(ii)), 0.5));

    jacratio = (pow((double) sumetasq(ii), 0.5)) / (pow((double) sumetasq0(ii), 0.5));
    w(ii) = x[ii]/scale;
    double prop = exp(0.5 * (lastw(ii) - w(ii)) *
        (lastw(ii) + w(ii))) * jacratio;
    accrej = (!(prop < 1) ? 1 : prop);
    if (gen < accrej) {
        lastw(ii) = w(ii);
        lasteta(jj, ii) = eta(jj, ii);
    }
}
A. APPENDIX A: C CODE

} /* Collect means */
wmeans[k] = mean(lastw);
etameans[k] = mean(mean(lasteta));
for (i = 0; i < *n; i++) { /* Store final w and eta values */
    ws[i] = lastw(i);
    for (j = 0; j < *nu; j++) {
        etas[j * (*n) + i] = lasteta(j, i);
    }
}

void mhf(double *x, int *nsim, int *msim, int *n, int *q, int *nu,
         double *alpha, char **covf, double *futurez)
{
    /* This function takes in a set of n increments (x) and
     produces a set of msim future paths (each of length q), each
     based on generating a set of etas (given x) using nsim
     Metropolis-Hastings steps. Nu and alpha are as for previous
     code, and the code covf='A' features FATGBM using Heyde's
     covariance function and 'B' using Fractional Brownian Motion
     covariance. This algorithm nests the MH loop within the
     future path loop (i.e. uses a different eta generated by MH
     for each future path) and is hence an nsim*msim complexity
     algorithm. */
    const gsl_rng_type *T;
    gsl_rng *r;
    long i, j, k, ll, lll, p;
    long seed;
    int ii, jj;
    mat rho2 = randu < mat > (*n + *q, *n + *q);
    rho2.ones();
    mat eta = randu < mat > (*nu, *n);
    eta.ones();
    vec w(*n);
    vec lastw = w;
    mat lasteta = eta;
    mat etasq0 = square(eta);
    rowvec sumetasq0 = sum(etasq0, 0);
    mat etasq = square(eta);
    rowvec sumetasq = sum(etasq, 0);
    mat newetasq = square(eta);
    rowvec newsumetasq = sum(newetasq, 0);
    mat xl = randu < mat > (*n - 1, 1);
    mat C = randu < mat > (*n - 1, 1);
    mat tC = trans(C);
    vec norms = randu < vec > (*q);
    vec norms2 = randu < vec > (*q);
    vec incrs = randu < vec > (*q);
    vec mus = randu < vec > (*q);
    vec cond = randu < vec > (*n);
    vec newout = randu < vec > (*q);
    mat neweta = randu < mat > (*nu, *q);
    mat newz = randu < mat > (*msim, *q);
    neweta.ones();
    newz.ones();
    /* select random number generator */
    T = gsl_rng_default;
```c
r = gsl_rng_alloc(gsl_rng_mt19937);
seed = time(NULL);
gsl_rng_set(r, seed);

//set seed
if (**covf == 'A') {
    /* Heyde's covariance function -
    here a matrix of size (n+q)*(n+q). This will be divided
    into an n*n submatrix (past) and a q*q matrix (future)
    as the diagonal blocks. */
    for (i = 0; i < *n + *q - 1; i++) {
        for (j = i + 1; j < *n + *q; j++) {
            rho2(i, j) = (pow((double) (1 + (j - i) * (j - i)), -*alpha/2));
            rho2(j, i) = rho2(i, j);
        }
    }
}

if (**covf == 'B') {
    for (i = 0; i < *n + *q - 1; i++) {
        for (j = i + 1; j < *n + *q; j++) {
            rho2(i, j) = (pow((double) (j - i + 1), 2 *
            (1 - *alpha)) - 2 * pow((double) (j - i), 2 *
            (1 - *alpha)) + pow((double) abs(j - i - 1),
            2 * (1 - *alpha)))/2;
            rho2(j, i) = rho2(i, j);
        }
    }
}

/* Past covariance function */
mat rho = rho2.submat(0, 0, *n - 1, *n - 1);
mat xla = eta;
mat B = rho;
mat Ca = rho;
mat Ba = rho;
mat A1 = rho2.submat(*n, *n, *n + *q - 1, *n + *q -
        1);
mat CC = rho2.submat(*n, 0, *n + *q - 1, *n - 1);
mat news = A1 - CC * inv(B) * trans(CC);
/* Cholesky decomposition used to generate MVN later */
mat cholnews = trans(chol(news));
for (p = 0; p < *msim; p++) {
    for (i = 0; i < *n; i++) { /* Initialise w=x */
        w(i) = x[i];
    }
}
lastw = w;
eta.fill((pow((double) (*nu - 2)/(*nu), 0.5)));
lasteta = eta;
for (k = 0; k < *nsim; k++) {
    /* Metropolis-Hastings loop */
    xla = eta;
    B = rho;
    Ca = rho;
    ii = gsl_rng_uniform_int(r, *n);
    jj = gsl_rng_uniform_int(r, *nu);
etasq0 = square(eta);
sumetasq0 = sum(etasq0, 0);
    xla.shed_col(ii);
    x1 = trans(xla.row(jj));
    B.shed_row(ii);
```
```c
B.shed_col(ii);
Ca.shed_row(ij);
C = Ca.col(ii);
tC = trans(C);
double newmu = as_scalar(tC * inv(B) * x);
double newsig = as_scalar(rho(ii, ii) - tC * inv(B) * C);
et(a JJ, ii) = newmu + gsl_ran_gaussian(r, 1) *
    (pow((double) newsig, 0.5));
etasq = square(eta);
sumetasq = sum(etasq, 0);
double scale = (pow((double)((nu-2)/sumetasq(ii)), 0.5));
double jacratio = (pow((double) sumetasq(ii), 0.5)) /
    (pow((double) sumetasqO(ii), 0.5));
w(ii) = x[ii]/scale;
double prop = exp(0.5 * (lastw(ii) - w(ii)) *
    (lastw(ii) + w(ii))) * jacratio;
double accrej = (!((prop < 1) ? 1 : prop));
double gen = gsl_rng_uniform(r);
if (gen < accrej) {
lastw(ii) = w(ii);
lasteta(jj, ii) = eta(jj, ii);
}
et = lasteta;
w = lastw;
}

/* Now generate future values given eta produced by MH */
for (11 = 0; 11 < *nu; 11++) {
    cond = trans(eta.row(ll));
    mus = CC * inv(Ba) * cond;
    for (i = 0; i < *q; i++) {
        norms(i) = gsl_ran_gaussian(r, 1);
    }
    newout = mus + cholnews * norms;
    neweta.row(ll) = trans(newout);
}
newetasq = square(neweta);
newsumetasq = (*nu - 2)/sum(newetasq, 0);
newz.row(p) = newsumetasq;
for (111 = 0; 111 < *q; 111++) { /* Store future values */
    futurez[p * (*q) + 111] = newz(p, 111);
}
```

A. APPENDIX A: C CODE

This function takes in a set of n increments (x) and produces a set of msim future paths (each of length q), each set based on one of netas initial sets of etas (given x) using nsim Metropolis-Hastings steps. The difference between this function and the preceding function is that multiple future paths are generated from each set of generated etas, rather than generating a new eta for each future path generation. Nu and alpha are as for previous code, and the code covf='A' features FATGBM using Heyde's covariance function and 'B' using Fractional Brownian Motion covariance. This algorithm places the MH loop BEFORE the future path loop.
(i.e. uses the same eta generated by MH for each future path) and is hence an nsim+msim complexity algorithm. Comments are as for the preceding algorithm - the only difference is that the future path loop is sequential to the MH loop rather than the MH loop being nested within the future path loop. */

const gsl_rng_type *T;
gsl_rng *r;
long i, j, k, ll, lll, p, lps;
long seed;
int ii, jj;
mat rho2 = randu < mat > (*n + *q, *n + *q);
mat rho2.ones();
mat eta = randu < mat > (*nu, *n);
eta.ones();
vec w(*n);
vec lastw = w;
mat lasteta = eta;
mat etasq0 = square(eta);
rowvec sumetasq0 = sum(etasq0, 0);
mat etasq = square(eta);
rowvec sumetasq = sum(etasq, 0);
mat newetasq = square(eta);
rowvec newsumetasq = sum(etasq, 0);
mat x1 = randu < mat > (*n - 1, 1);
mat C = randu < mat > (*n - 1, 1);
mat tC = trans(C);
vec norms = randu < vec > (*q);
vec norms2 = randu < vec > (*q);
vec incrs = randu < vec > (*q);
vec mus = randu < vec > (*q);
vec cond = randu < vec > (*n);
vec newout = randu < vec > (*q);
mat neweta = randu < mat > (*nu, *q);
mat newz = randu < mat > (*msim, *q);
neweta.ones();
newz.ones();
/* select random number generator */
T = gsl_rng_default;
r = gsl_rng_alloc(gsl_rng_mt19937);
seed = time(NULL);
gsl_rng_set(r, seed);
//set seed
if (**covf == 'A') {
for (i = 0; i < *n + *q - 1; i++) {
 for (j = i + 1; j < *n + *q; j++) {
 rho2(i, j) = (pow((double) (1 + (j-i)*(j-i)), -*alpha/2));
 rho2(j, i) = rho2(i, j);
 }
}
}
if (**covf == 'B') {
for (i = 0; i < *n + *q - 1; i++) {
 for (j = i + 1; j < *n + *q; j++) {
 rho2(i, j) = (pow((double) (j - i - 1),
 2 * (1 - *alpha)) - 2 * pow((double) (j - i),
 2 * (1 - *alpha)) + pow((double) abs(j - i - 1),
 2 * (1 - *alpha)))/2;
 rho2(j, i) = rho2(i, j);
 }
}
mat rho = rho2.submat(0, 0, *n - 1, *n - 1);
mat xla = eta;
mat B = rho;
mat Ca = rho;
mat Ba = rho;
mat A1 = rho2.submat(*n, *n, *n + *q - 1, *n + *q - 1);
mat CC = rho2.submat(*n, 0, *n + *q - 1, *n - 1);
mat news = A1 - CC * inv(B) * trans(CC);
mat cholnews = trans(chol(news));
for(lps=0; lps<*netas; lps++) {
    xla = eta;
    B = rho;
    Ca = rho;
    Ba = rho;
    newz.ones();
    neweta.ones();
    for (i = 0; i < *n; i++) {
        w(i) = x[i];
    }
    lastw = w;
    eta.fill((pow((double) (*nu - 2)/(*nu), 0.5)));
    lasteta = eta;
    for (k = 0; k < *nsim; k++) {
        xla = eta;
        B = rho;
        Ca = rho;
        ii = gsl_rng_uniform_int(r, *n);
        jj = gsl_rng_uniform_int(r, *nu);
        etasq0 = square(eta);
        sumetasq0 = sum(etasq0, 0);
        xla.shed_col(ii);
        xl = trans(xla.row(jj));
        B.shed_row(ii);
        B.shed_col(ii);
        Ca.shed_row(ii);
        Ca.shed_col(ii);
        C = Ca.col(ii);
        tC = trans(C);
        double newmu = as_scalar(tC * inv(B) * xl);
        double newsig = as_scalar(rho(ii, ii) - tC * inv(B) * C);
        eta(jj, ii) = newmu + gsl_ran_gaussian(r, 1) *
            (pow((double) newsig, 0.5));
        etasq = square(eta);
        sumetasq = sum(etasq, 0);
        double scale = (pow((double) ((*nu - 2)/sumetasq(ii)), 0.5));
        double jacratio = (pow((double) sumetasq(ii), 0.5)) /
            (pow((double) sumetasq0(ii), 0.5));
        w(ii) = x[ii]/scale;
        double prop = exp(0.5 * (lastw(ii) - w(ii)) *
            (lastw(ii) + w(ii))) * jacratio;
        double accrej = (!((prop < 1) ? 1 : prop));
        double gen = gsl_rng_uniform(r);
        if (gen < accrej) {
            lastw(ii) = w(ii);
            lasteta(jj, ii) = eta(jj, ii);
        }
    }
    eta = lasteta;
    w = lastw;
for (p = 0; p < *msim; p++) {
    /* Future path loop follows a single MH loop rather than
generating a new eta using MH for each future path */
    for (ll = 0; ll < *nu; ll++) {
        cond = trans(eta.row(ll));
        mus = CC * inv(Ba) * cond;
        for (i = 0; i < *q; i++) {
            norms(i) = gsl_ran_gaussian(r, 1);
        }
        newout = mus + cholnews * norms;
        neweta.row(ll) = trans(newout);
    }
    newetasq = square(neweta);
    newsumetasq = (*nu - 2)/sum(newetasq, 0);
    newz.row(p) = newsumetasq;
    for (111 = 0; 111 < *q; 111++) {
        futurez[(lps*(*msim)*(*q))+(p * (*q)) + 111] = newz(p, 111);
    }
}

void mh2(double *x, int *nsim, int *n, int *nu, double *alpha,
         char **covf, double *wmeans, double *etameans, double *ws,
         double *etas)
{
    /* Metropolis-Hastings algorithm for Sly’s version of FATGBM
    Motion covariance function (B). Here, x is the increments on
    which to condition, nsim is the number of MH iterations, n is
    the length of x, alpha = 1-H. In the algorithm, values of the
    w vector and eta vector are updated and the respective means
    are collected at each iteration along with the final values
    of w and eta. */
    const gsl_rng_type *T;
    gsl_rng *r;
    long i, j, k;
    long seed;
    int ii;
    mat rho = randu < mat > (*n, *n);
    rho. ones 0;
    vec eta(*n);
    eta.fill(l.O);
    vec w(*n);
    /select random number generator */
    T = gsl_rng_default;
    r = gsl_rng_alloc(gsl_rng_mt19937);
    seed = time(NULL);
    gsl_rng_set(r, seed);
    if (**covf == 'A') { /* Sly’s covariance function */
        for (i = 0; i < *n - 1; i++) {
            for (j = i + 1; j < *n; j++) {
                rho(i, j) = (pow((double) (1 + (j-i)*(j-i)), -*alpha));
            }
        }
    }
if (**covf == 'B') {
    for (i = 0; i < *n - 1; i++) {
        /* Fractional Brownian Motion covariance function */
        for (j = i + 1; j < *n; j++) {
            rho(i, j) = (pow((double) (j - i + 1),
                2 * (1 - *alpha)) - 2 * pow((double) (j - i),
                2 * (1 - *alpha)) + pow((double) abs(j - i - 1),
                2 * (1 - *alpha)))/2;
            rho(j, i) = rho(i, j);
        }
    }
    /* Now, generate initial value of T based on Sly's
       formulation: i.e. F'(-1)(Phi(x)) where F is the cdf of
       ((nu-2)/nu *RG(nu/2,nu/2) */
    double pn = gsl_cdf_ugaussian_Q(1.0);
    double ww = ((*(nu)-2)/(gsl_cdf_gamma_Pinv(pn,(double)(*nu)/2,
                (double) 2/(*nu)) * (*nu));
    for (i = 0; i < *n; i++) { /* Initialise w */
        w(i) = x[i]/pow((double) ww, 0.5);
    }
    vec lastw = w;
    vec lasteta = eta;
    for (k = 0; k < *nsim; k++) { /* Metropolis-Hastings steps */
        mat x1 = eta;
        mat B = rho;
        mat Ca = rho;
        ii = gsl_rng_uniform_int(r, *n); /* Choose an eta at random */
        double chosen = eta(ii);
        double pn0 = gsl_cdf_ugaussian_Q(chosen);
        double a0 = ((*(nu)-2)/(gsl_cdf_gamma_Pinv(pn0,
                (double) 2/(*nu)) * (*nu));
        x1.shed_row(ii);
        B.shed_row(ii);
        B.shed_col(ii);
        Ca.shed_row(ii);
        mat C = Ca.col(ii);
        mat tC = trans(C);
        double newmu = as_scalar(tC * inv(B) * x1);
        double newsig = as_scalar(rho(ii, ii) - tC * inv(B) * C);
        /* Generate new candidate eta */
        eta(ii) = newmu + gsl_ran_gaussian(r, 1) *
            (pow((double) newsig, 0.5));
        double pn1 = gsl_cdf_ugaussian_Q((double) eta(ii));
        double a1 = ((*(nu)-2)/(gsl_cdf_gamma_Pinv(pn1,
                (double) 2/(*nu)) * (*nu));
        w(ii) = x[ii]/pow((double) a1, 0.5);
        /* Ratio of Jacobians - see proof */
        double jacratio = pow((double) (a0/a1), 0.5);
        double prop = exp(0.5 * (lastw(ii) - w(ii)) *
            (lastw(ii) + w(ii))) * jacratio;
        /* Accept-reject based on computed transition probability */
        double accrej = (1/(prop < 1) ? 1 : prop);
        double gen = gsl_rng_uniform(r);
        if (gen < accrej) {
            lastw(ii) = w(ii);
            lasteta(ii) = eta(ii);
        }
    }
}
A. APPENDIX A: C CODE

944 /* Collect means */
945 wmeans[k] = mean(lastw);
946 etameans[k] = mean(lasteta);
947 }
948 for (i = 0; i < *n; i++) { /* Store final w and eta */
949 ws[i] = lastw(i);
950 etas[i] = lasteta(i);
951 }
952 
953 void
954 mh2f(double *x, int *nsim, int *msim, int *n, int *q, int *nu,
955 double *alpha, char **covf, double *futurez)
956 {
957 /* This function takes in a set of n increments (x) and
produces a set of msim future paths (each of length q), each
based on generating a set of etas (given x) using nsim
Metropolis-Hastings steps. Nu and alpha are as for previous
code, and the code covf='A' features Sly's FATGBM
construction using Sly's covariance function and 'B' using
Fractional Brownian Motion covariance. This algorithm nests
the MH loop within the future path loop (i.e. uses a
different eta generated by MH for each future path) and is
hence an nsim*msim complexity algorithm. */
958 const gsl_rng_type *T;
959 gsl_rng *r;
960 long i, j, k, ll, ll1, p;
961 long seed;
962 int ii;
963 mat rho2 = randu < mat > (*n + *q, *n + *q);
964 rho2.ones();
965 vec eta = randu < vec > (*n);
966 eta.fill(l.0);
967 vec w(*n);
968 vec lastw = w;
969 vec lasteta = eta;
970 mat C = randu < mat > (*n - 1, 1);
971 mat tC = trans(C);
972 vec norms = randu < vec > (*q);
973 vec norms2 = randu < vec > (*q);
974 vec mus = randu < vec > (*q);
975 vec newout = randu < vec > (*q);
976 vec neweta = randu < vec > (*q);
977 mat newz = randu < mat > (*msim, *q);
978 neweta.ones();
979 newz.ones();
980 /* select random number generator */
981 T = gsl_rng_default;
982 r = gsl_rng_alloc(gsl_rng_mt19937);
983 seed = time(NULL);
984 gsl_rng_set(r, seed);
985 
986 if (**covf == 'A') {
987 /* Sly's covariance function - here a matrix of size
(n+q)*(n+q). This will be divided into an n*n submatrix
(past) and a q*q matrix (future) as the diagonal blocks. */
988 for (i = 0; i < *n + *q - 1; i++) {
989 for (j = i + 1; j < *n + *q; j++) {
990 rho2(i, j) = (pow((double) (1 + (j-i)*(j-i)), -*alpha));
991 rho2(j, i) = rho2(i, j);}
```c
if (**covf == 'B') {
    for (i = 0; i < *n + *q - 1; i++) {
        for (j = i + 1; j < *n + *q; j++) {
            rho2(i, j) = (pow((double) (j - i + 1),
                2 * (1 - *alpha)) - 2 * pow((double) (j - i),
                2 * (1 - *alpha)) + pow((double) abs(j - i - 1),
                2 * (1 - *alpha)))/2;
            rho2(j, i) = rho2(i, j);
        }
    }
    mat rho = rho2.submat(0, 0, *n - 1, *n - 1);
    vec xl = eta;
    mat B = rho;
    mat Ca = rho;
    mat Ba = rho;
    mat A1 = rho2.submat(*n, *n, *n + *q - 1, *n + *q - 1);
    mat CC = rho2.submat(*n, 0, *n + *q - 1, *n - 1);
    mat news = A1 - CC * inv(B) * trans(CC);
    /* Cholesky decomposition used to generate MVN later */
    mat cholnews = trans(chol(news));
    for (p = 0; p < *msim; p++) {
        double pn = gslcdf_ugaussian_Q(1.0);
        double ww = ((*nu)-2)/(gsl_cdf_gamma_Pinv(pn, (double)(*nu)/2, (double) 2/(*nu)) * (*nu));
        for (i = 0; i < *n; i++) {
            w(i) = x[i]/pow((double) ww, 0.5);
        }
    }
    lastw = w;
    lasteta = eta;
    for (k = 0; k < *nsim; k++) {
        /* Metropolis-Hastings loop */
        x1 = eta;
        B = rho;
        Ca = rho;
        ii = gsl_rng_uniform_int(r, *n);
        double chosen = eta(ii);
        double pn0 = gslcdf_ugaussian_Q(chosen);
        double a0 = ((*nu)-2)/(gslcdfgamma_Pinv(pn0, (double)(*nu))/2, (double) 2/(*nu)) * (*nu));
        x1.shed_row(ii);
        B.shed_row(ii);
        B.shed_col(ii);
        Ca.shed_row(ii);
        C = Ca.col(ii);
        tC = trans(C);
        double newmu = as_scalar(tC * inv(B) * x1);
        double newsig = as_scalar(rho(ii, ii) - tC * inv(B) * C);
        eta(ii) = newmu + gslran_gaussian(r, 1) *
            (pow((double) newsig, 0.5));
        double pnl = gslcdf_ugaussian_q((double) eta(ii));
        double al = ((*nu)-2)/(gslcdfgamma_Pinv(pnl, (double)(*nu))/2, (double) 2/(*nu)) * (*nu));
        w(ii) = x[iil]/pow((double) a1, 0.5);
    }
    /* Past covariance function */
    mat rho = rho2.submat(0, 0, *n - 1, *n - 1);
    vec x1 = eta;
    mat B = rho;
    mat Ca = rho;
    mat Ba = rho;
    mat A1 = rho2.submat(*n, *n, *n + *q - 1, *n + *q - 1);
    mat CC = rho2.submat(*n, 0, *n + *q - 1, *n - 1);
    mat news = A1 - CC * inv(B) * trans(CC);
    /* Cholesky decomposition used to generate MVN later */
    mat cholnews = trans(chol(news));
    for (p = 0; p < *msim; p++) {
        double pn = gslcdf_ugaussian_Q(1.0);
        double ww = ((*nu)-2)/(gsl_cdf_gamma_Pinv(pn, (double)(*nu)/2, (double) 2/(*nu)) * (*nu));
        for (i = 0; i < *n; i++) {
            w(i) = x[i]/pow((double) ww, 0.5);
        }
    }
    lastw = w;
    lasteta = eta;
    for (k = 0; k < *nsim; k++) {
        /* Metropolis-Hastings loop */
        x1 = eta;
        B = rho;
        Ca = rho;
        ii = gsl_rng_uniform_int(r, *n);
        double chosen = eta(ii);
        double pn0 = gslcdf_ugaussian_Q(chosen);
        double a0 = ((*nu)-2)/(gslcdfgamma_Pinv(pn0, (double)(*nu))/2, (double) 2/(*nu)) * (*nu));
        x1.shed_row(ii);
        B.shed_row(ii);
        B.shed_col(ii);
        Ca.shed_row(ii);
        C = Ca.col(ii);
        tC = trans(C);
        double newmu = as_scalar(tC * inv(B) * x1);
        double newsig = as_scalar(rho(ii, ii) - tC * inv(B) * C);
        eta(ii) = newmu + gslran_gaussian(r, 1) *
            (pow((double) newsig, 0.5));
        double pnl = gslcdf_ugaussian_q((double) eta(ii));
        double al = ((*nu)-2)/(gslcdfgamma_Pinv(pnl, (double)(*nu))/2, (double) 2/(*nu)) * (*nu));
        w(ii) = x[iil]/pow((double) a1, 0.5);
    }
}```
double jacratio = pow((double) (aO/al), 0.5);
double prop = exp(0.5 * (lastw(ii) - w(ii)) *
  (lastw(ii) + w(ii))) * jacratio;
double accrej = (! (prop < 1) ? 1 : prop);
double gen = gsl_rng_uniform(r);
if (gen < accrej) {
lastw(ii) = w(ii);
lasteta(ii) = eta(ii);
}
eta = lasteta;
w = lastw;

for (ll = 0; ll < *nu; ll++) {
  /* Now generate future values given the eta produced by MH */
  mus = CC * inv(Ba) * eta;
  for (i = 0; i < *q; i++) {
    norms(i) = gsl_ran_gaussian(r, 1);
  }
  neweta = mus + cholnews * norms;
}

for (i = 0; i < *q; i++) {
  double pnl = gsl_cdf_ugaussian_q((double) neweta(i));
  newz(p,i) = (*nu)-2)/(gsl_cdf^gamma_Pinv(pnl,(double)(*nu)/2,
                                  (double)2/(*nu)) * (*nu));
}

for (111 =0; 111 < *q; 111++) { A
  futurezLp * (*q) + 111] = newz(p, 111);
}

void mh2fb(double *x, int *nsim, int *msim, int *n, int *q, int *nu,
            int *netas, double *alpha, char **covf, double *futurez)
{
  /* This function takes in a set of n increments (x) and
     produces a set of msim future paths each length q, each set
     based on one of netas initial sets of etas (given x) using
     nsim Metropolis-Hastings steps. The difference between this
     function and the preceding function is that multiple future
     paths are generated from each set of generated etas,
     rather than generating a new eta for each future path
     generation. Nu and alpha are as for previous code, and the
     code covf='A' features Sly's FATGBM using Sly's covariance
     function and 'B' using Fractional Brownian Motion covariance.
     This algorithm places the MH loop BEFORE the future path loop
     (i.e. uses the same eta generated by MH for each future path)
     and is hence an nsim+msim complexity algorithm. Comments are
     as for the preceding algorithm - the only difference is that
     the future path loop is sequential to the MH loop rather than
     the MH loop being nested within the future path loop. */
  const gsl_rng_type *T;
  gsl_rng *r;
  long i, j, k, ll, l11, p, lps;
  long seed;
  int ii;
  mat rho2 = randu < mat > (*n + *q, *n + *q);
  rho2.ones();
  vec eta = randu < vec > (*n);
etaf.fill(1.0);
vec w(*n);
vec lastw = w;
vec lasteta = eta;
mat C = randu<mat>(*n - 1, 1);
mat tC = trans(C);
vec norms = randu<vec>(*q);
vec norms2 = randu<vec>(*q);
vec mus = randu<vec>(*q);
vec newout = randu<vec>(*q);
vec neweta = randu<vec>(*q);
mat newz = randu<mat>(*msim, *q);
neweta.ones();
newz.ones();

/* select random number generator */
T = gsl_rng_default;
r = gsl_rng_alloc(gsl_rng_mt9937);
seed = time(NULL);
gsl_rng_set(r, seed);
//set seed
if (**covf == 'A') {
for (i = 0; i < *n + *q - 1; i++) {
for (j = i + 1; j < *n + *q; j++) {
  rho2(i, j) = (pow((double) (j - i), -2 * alpha)) / 2;
  rho2(j, i) = rho2(i, j);
}
}
if (**covf == 'B') {
for (i = 0; i < *n + *q - 1; i++) {
for (j = i + 1; j < *n + *q; j++) {
  rho2(i, j) = (pow((double) (j - i), 2 * (1 - alpha)) - 2 * pow((double) (j - i), 2 * (1 - alpha)) + pow((double) abs(j - i - 1), 2 * (1 - alpha)) / 2;
  rho2(j, i) = rho2(i, j);
}
}
mat rho = rho2.submat(0, 0, *n - 1, *n - 1);
vec x1 = eta;
vec B = rho;
mat Ca = rho;
mat Ba = rho;
mat A1 = rho2.submat(*n, *n, *n + *q - 1, *n + *q - 1);
mat CC = rho2.submat(*n, 0, *n + *q - 1, *n - 1);
mat news = A1 - CC * inv(B) * trans(CC);
mat cholnews = trans(chol(news));
for (lps = 0; lps < *netas; lps++) {
x1 = eta;
B = rho;
Ca = rho;
Ba = rho;
newz.ones();
neweta.ones();
double pn = gsl_cdf_ugaussian_Q(1.0);
double WW = ((*nu) - 2) / gsl_cdf_gamma_Pinv(pn, (double)(*nu) / 2, (double) 2 / (*nu)) * (*nu);
for (i = 0; i < *n; i++) {
    w(i) = x[i]/pow((double) ww, 0.5);
}

lastw = w;
lasteta = eta;
for (k = 0; k < *nsim; k++) {
    xl = eta;
    B = rho;
    Ca = rho;
    ii = gsl_rng_uniform_int(r, *n);
    double chosen = eta(ii);
    double pn0 = gsl_cdf_ugaussian_Q(chosen);
    double a0 = ((*nu)-2)/(gsl_cdf_gamma_Pinv(pn0, (double)(*nu)) * (*nu));
    xl.shed_row(ii);
    B.shed_row(ii);
    B.shed_col(ii);
    Ca.shed_row(ii);
    C = Ca.col(ii);
    tC = trans(C);
    double newmu = as_scalar(tC * inv(B) * xl);
    double newsig = as_scalar(rho(ii, ii) - tC * inv(B) * C);
    eta(ii) = newmu + gsl_ran_gaussian(r, 1) * 
          (pow((double) newsig, 0.5));
    double pn1 = gsl_cdf_ugaussian_Q((double) eta(ii));
    double a1 = ((*nu)-2)/(gsl_cdf_gamma_Pinv(pn1, (double)(*nu)) * (*nu));
    double jacratio = pow((double) (a0/a1), 0.5);
    double prop = exp(0.5 * (lastw(ii) - w(ii)) * 
                   (lastw(ii) + w(ii))) * jacratio;
    double accrej = !(prop < 1) ? 1 : prop;
    double gen = gsl_rng_uniform(r);
    if (gen < accrej) {
        lastw(ii) = w(ii);
        lasteta(ii) = eta(ii);
    }
    eta = lasteta;
    w = lastw;
}
for (p = 0; p < *msim; p++) {
    /* Future path loop follows a single MH loop rather than 
     * generating a new eta using MH for each future path generated. 
     */
    for (ll = 0; ll < *nu; ll++) {
        mus = CC * inv(Ba) * eta;
        for (i = 0; i < *q; i++) {
            norms(i) = gsl_ran_gaussian(r, 1);
        }
        neweta = mus + cholnews * norms;
    }
    for (i = 0; i < *q; i++) {
        double pn1 = gsl_cdf_ugaussian_Q((double) neweta(i));
        newz(p,i)=((*nu)-2)/(gsl_cdf_gamma_Pinv(pn1,(double)(*nu))/2, 
                   (double)(*nu)) * (*nu));
    }
    for (lll = 0; lll < *q; lll++) {
        futurez[(lls*(msim)*(q))+(p*(q)) + lll] = newz(p, lll)];
    }
APPENDIX B

Appendix B: Supplementary R Code

gat=function(n,incs) {
# This function takes a set of activity time increments (incs) and a
# "step size" (equal to 1/n) and returns the resultant clock times # n
defines the steps (1/n,2/n,...,nY_T/n) and incs is a set of
# increments (FATGBM increments)
w <- cumsum(incs)
# Create the cumulative sum of increments - i.e. the y_t series
goal <- floor(n*w[length(incs)])
# The (discrete) number of steps required
raw <- findInterval((1:goal)/n,c(0,w),rightmost.closed=T)
# This step places the items 1/n,2/n,3/n,... into the intervals
# defined by the y_t series. This tells you where the first hits are
ipl <- apply(cbind(raw,(1:goal)/n),l,iplt,c(0,w))
# This interpolates between the values in the intervals just discovered
# in the preceding step, using a linear interpolation. I wrote a
# simple function to do the interpolation called iplt.
# Return a matrix whose first column is the 1/n,2/n,etc. and whose
# second column is the t_i's
# To return a 2 column matrix with the grid of activity times and
# interpolated
# real times, use cbind((1:goal)/n,ipl)
ipl}
# Functions for finding the implied volatility
#Newton Raphson Method for implied volatility (Black Scholes Call Option)
fn <- function(sigma,c,price,strike,rate,t){
A <- log(price/strike)/sqrt(t)+rate*sqrt(t)
d1 <- A/sigma + sigma*sqrt(t)/2
d2 <- A/sigma - sigma*sqrt(t)/2
d1d <- -A/(sigma^2)+sqrt(t)/2
d2d <- -A/(sigma^2)-sqrt(t)/2
f <- price*pnorm(d1,0,1)-strike*exp(-rate*t)*pnorm(d2,0,1)-c
fprime <- price*dnorm(d1,0,1)*dd1-strike*exp(-rate*t)*dnorm(d2,0,1)*dd2
c(f,fprime)
}
findimpvol <- function(fv,p=price,s=strike,r=rate,m=m){
r <- r+12
t <- (1:m)/12
c <- fv
oldsigma <- 0.3
sigma2=0
for(i in 1:length(t)){
diff <- 0.01
tol <- 0.001
while(abs(diff)>tol){
evals <- fn(oldsigma,c[i],p,s,r,t[i])
sigma2[i] <- oldsigma - evals[1]/evals[2]
diff <- sigma2[i]-oldsigma
oldsigma <- sigma2[i]}
sigma2}
}
Bibliography


179
[53] Protter, P 2005, Stochastic Integration and Differential Equations, Springer-Verlag
[56] Rostek, S 2009, Option Pricing in Fractional Brownian Markets, Springer-Verlag