Thesis for master of philosophy in Financial Mathematics

Stochastic Time-Changed Lévy Processes with their Implementation

Odwa Sihlobo

February 2014

University of Cape Town
Faculty of Commerce
Department of Actuarial Science
Supervisor: Dr. Sure Mataramvura

I would like to thank my mom for her love and support.

Contents

Abstract

We focus on the implementation details for Lévy processes and their extension to stochastic volatility models for pricing European vanilla options and exotic options. We calibrated five models to European options on the S&P500 and used the calibrated models to price a cliquet option using Monte Carlo simulation. We provide the algorithms required to value the options when using Lévy processes. We found that these models were able to closely reproduce the market option prices for many strikes and maturities. We also found that the models we studied produced different prices for the cliquet option even though all the models produced the same prices for vanilla options. This highlighted a feature of model uncertainty when valuing a cliquet option. Further research is required to develop tools to understand and manage this model uncertainty. We make a recommendation on how to proceed with this research by studying the cliquet option's sensitivity to the model parameters.

1 Introduction

An important objective of developing strategies for pricing and hedging derivatives, as well as for asset allocation, value at risk (VaR) and managing risk, is to find asset return models that parsimoniously and accurately reflect the behaviour of the observed underlying asset. A quantitative analyst may proceed as follows:

- Specify a model. Several models will be needed if model uncertainty is to be assessed.
- Calibrate the model parameters asset returns or market prices of liquid options.
- Compute required values using the calibrated model. These could be VaR estimates, derivative prices, market risk factor sensitivities (e.g. Greeks) and other financial quantities.

In this work we consider Lévy process models as a refinement on the Brownian motion used in the Black-Scholes-Merton (BSM) world. Unlike Brownian motion, Lévy processes can capture some stylised features of market return innovations such as asset price jumps, heavy-tailed distribution, skewed returns. These characteristics are accentuated when we observe high-frequency data. We note that return volatilities are stochastic. Also, the returns are correlated with their volatilities (often negatively for equities), the so-called leverage effect described by [Bla76]. The return volatilities are also known to exhibit clustering behaviour. We need to take these characteristics into account in our modelling. We explore the use of time-changed Lévy processes to improve on the Black-Scholes model to allow us to parsimoniously reflect all the stylised features described above. We find that the time-changed Lévy processes accurately capture the volatility skew which is not reflected by the Black-Scholes option pricing model.

Lévy processes have been extensively studied in the literature, see [Ber96], [Sat99] and [App04]. Examples of Lévy models that we shall consider for modelling the stock innovations include the variance gamma (VG) model of [MCC98], the so-called CGMY model of [CGMY02], the normal inverse Gaussian (NIG) model of [BN97], the generalised hyperbolic (GH) model of [EKP98] and the Meixner model introduced in [ST01] . We model the stochastic time-changes as mean-reverting Ornstein-Uhlenbeck (OU) processes driven by Lévy processes. In particular, OU processes generated by the inverse Gaussian and gamma processes are used as well as the CIR process.

Once a model has been specified we need to estimate its parameters. We may estimate the statistical parameters from observed asset returns by method of moments

or maximum likelihood estimation. To estimate the risk neutral parameters we calibrate to market prices of European calls and puts. We price these using Lévy characteristic functions and the Fourier-cosine series expansion of [FO08]. Some challenges of parameter estimation include unknown probability densities for MLE, unknown theoretical moments for MOM estimation, objective functions with multiple local minima and insufficient implementation detail of the specified models. Our work contributes to overcoming these obstacles.

After we have calibrated our model parameters we can calculate values such as VaR, derivative prices and so forth. We investigate pricing a cliquet option by Monte Carlo (MC) simulation under different models which are calibrated to the same market data. We present the details of simulating from the exact distribution of all the models we consider. We implement a new method discussed by [BK11] that allows us to simulate a process by a fractional fast Fourier transform (FrFFT) from its characteristic function. This is extremely helpful when no simulation procedure is known for the model in question. This exercise demonstrates that valuing highly leveraged derivatives involves model uncertainty. We give some advice how to proceed when with such products.

There are many traps in model usage and implementation. These involve models with too many parameters which can be unstable over time, poor numerical methods in implementing the models and objective functions with many local minima. The original works introducing these models don't deal with these problems. We attempt to resolve these issues to equip practitioners and researcher with robust tools for working with Lévy processes. We use pseudo code in our work to make the implementations portable across different programming languages. The MATLAB implementation is available from the author through email request.

This work progresses as follows. The next section discusses the Black-Scholes option pricing model and its shortcomings. In §3 we explore the properties of Lévy processes and introduce the Lévy models that will used in our work as well as the processes used for the random clock. In §4 and §5 we apply the Lévy processes to express the uncertainty in the economy and we price vanilla European options via the characteric function using the COS method. §6 discusses the algorithms that we need to simulate the Lévy processes. §7 contains the results from applying the Lévy processes to price an exotic option. §8 contains our conclusions and recommendations.

2 The Black Scholes Merton market model

Here we introduce the Wiener stochastic process which is used to model returns in the Black Scholes Merton (BSM) framework. We briefly discuss some of its properties. We then explore some stylised characteristics of empirical returns and contrast them with the properties of a Wiener process and find that the Wiener process is an inadequate model of financial returns. We shall not consider other crucial assumptions of the BSM framework such as continuous delta hedging and transaction costs.

2.1 The Black-Scholes market model

A standard Brownian motion (SBM) $\mathbf{W} = \{W_t : t \geq 0\}$ is an $\mathbb{R}$-valued stochastic process on a probability space $(\Omega, F, \mathbb{P})$ and satisfies the conditions:

- If $0 \leq s < t$, then $W_t - W_s \sim N(0, t-s)$.
- $\mathbf{W}$ has independent increments; for $0 \leq s \leq t, W_t - W_s$ is independent of W_u for $u \leq s$.
- The paths $t \to W_t$ are almost surely continuous.

The sample paths of a SBM are of unbounded variation. That is, the process changes infinitely many times within any finite time interval. However, empirical asset return dynamics do not possess this property partly due to tick size bounds and bid-ask spreads getting in the way.
SBM also possesses the scaling property

$$W_t \stackrel{d}{=} \sqrt{c}\widetilde{W}_{\frac{t}{c}} \tag{2.1}$$

where $c > 0$, $\stackrel{d}{=}$ denotes equality in distribution and $W_t, \widetilde{W}_t$ are two different SBMs. This means that if we model asset returns using SBM then the return distribution is invariant to the time scale we observe the returns on. For instance the distribution of the annualised daily, weekly or monthly returns will all have the same distribution. This property holds for any stochastic process with independent and identically distributed increments.

Suppose that our market has a risk-less bank account with price process $\mathbf{B} = \{B_t : t \geq 0\}$ and a risky asset $\mathbf{S} = \{S_t : t \geq 0\}$ that pays a continuous dividend yield $q \geq 0$. We model the risky asset as a geometric Brownian motion (GBM) so that we have

$$B_t = e^{rt} \qquad S_t = S_0 \exp\Big((\mu - q - \frac{1}{2}\sigma^2)t + \sigma W_t\Big) \tag{2.2}$$

where $\mathbf{W} = \{W_t : t \geq 0\}$ is a SBM under the real-world measure $\mathbb{P}$. Under this measure the log-returns are normally distributed

$$\log\left(\frac{S_t}{S_0}\right) \sim N\Big((\mu - q - \frac{1}{2}\sigma^2)t, \sigma^2 t\Big) \tag{2.3}$$

Since the Black-Scholes market is complete, there exists a unique equivalent martingale measure $\mathbb{Q}$. We call this measure the risk-neutral measure. Under the risk-neutral measure $\mathbb{Q}$ the stock price process is given by

$$S_t = S_0 \exp\Big((r - q - \frac{1}{2}\sigma^2)t + \sigma\widetilde{W}_t\Big), \quad S_0 > 0 \tag{2.4}$$

where $\widetilde{W}_t$ is a SBM under the measure $\mathbb{Q}$.

2.2 Imperfections of the Black-Scholes model

Empirical evidence shows that some of the assumptions of the Black-Scholes model do not hold. In this chapter we shall focus on the following empirical features of financial data:

- Normality of log returns.
- Independent and identically distributed log returns.
- Continuity of log return trajectories. This is not possible due to tick size bound and bid-ask spreads. There are other causes for return path discontinuities.
- Stochastic volatility and volatility clustering.

Refer to [Con01] for an extended list of the stylised facts and statistical issues of asset returns.
We will also focus on the smile effect observed in the market prices of options.

2.2.1 Return moments

SBM does not posses all the properties required to capture observed asset return characteristics. For instance, the distribution of empirical annualised returns are not invariant under time scaling. In other words, observed returns are not independent

and identically distributed (i.i.d.). Table 2.1 contains the first four moments of annualised daily, weekly and monthly returns on the S&P500 index obtained from Yahoo! Finance. The data is for the period 3 Jan 2000 to 29 Jan 2010 and was accessed on 8 October 2013.

Table 2.1: Annualised sample moments of S&P500 index historical log returns.

	Daily	Weekly	Monthly
mean	-0.0019	-0.0041	-0.0075
variance	0.0501	0.0402	0.0268
skewness	-0.1028	-0.8366	-0.7648
kurtosis	10.6347	9.9402	4.2928

The differing moments over different measurement periods indicate that empirical returns are not i.i.d.. Also note that the returns have a non-zeros skewness and the kurtosis is not 3 as suggested by a Gaussian distribution. The returns stray further from the Gaussian distribution when we observe high-frequency data. Thus Brownian motion cannot capture some of the characteristics of asset returns observed in the market. Later we use Lévy processes to model asset returns as they can capture asymmetry and excess kurtosis.

2.2.2 Does my tail look fat in this?

From a probabilistic perspective, maximum likelihood estimation (MLE) allows us to choose those model parameter values that maximise the likelihood of the data occurring. The MLE estimates for the normal distribution are given by the sample mean and (unbiased) variance. We use a qqplot in Figure 2.1 for a visual assessment of how well the normal distribution fits daily returns. The fit is very poor especially at the tails, showing us that the Gaussian distribution has mild randomness versus empirical returns. The consequence is that we end up underestimating the chance of extreme market events.

Next we plot the empirical density and the normal density (using the MLE parameters) for daily returns. We applied the Gaussian kernel to approximate the empirical density.

The empirical density has a higher peak. This suggests that the market experiences small market moves more often than implied by a normal distribution. The number of decimal places to which the prices are quoted may accentuate this peak. The empirical distribution also has fatter tails. We plot the log of the PDFs to get a closer look at the tails. The log of the normal density tapers off at a quadratic rate, way faster than the almost linear decay for empirical returns. This shows that the normal distribution has mild randomness when compared to actual asset returns. We shall emphasise this important point with an example. The market tends to

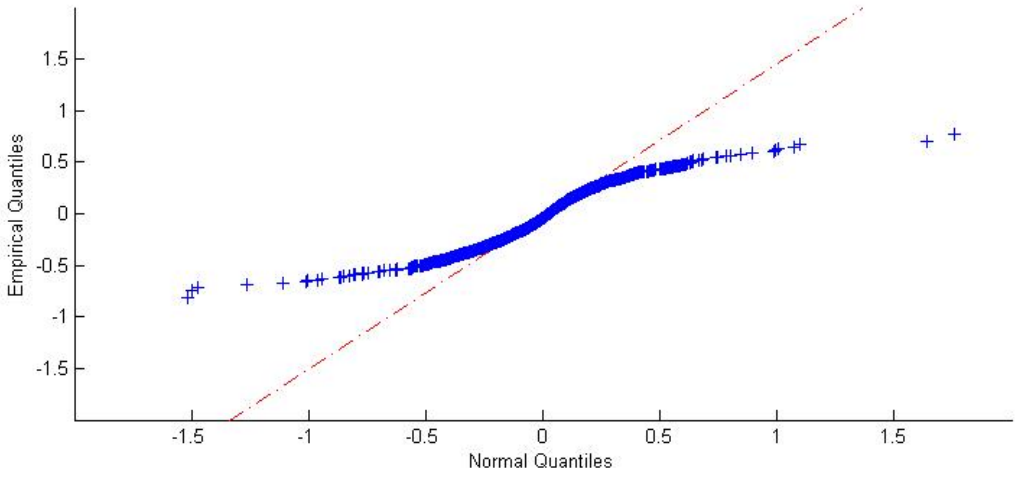

Figure 2.1: qqplot for normal distribution versus empirical distribution of log returns.

price far out-the-money (OTM) options above their BSM model prices. If you treat BSM as exact you would think that there were free profits to be made by shorting a large number of the "overpriced" near-term far OTM options at a fraction of the market price. These options will usually expire OTM. On occasion, the short parties will be caught. The fat tails imply that the short seller will be caught sooner than he thought and each extreme event will take him closer to ruin than he anticipated. A more approximately correct model must be able to capture market features such as skew and high kurtosis. The normal distribution cannot achieve this.

2.2.3 Stochastic volatility

Volatility is stochastic rather than constant as stipulated in the Black-Scholes model. We demonstrate this by calculating historical volatilities for our S&P500 time series. The historical volatility σ_i for each day is calculated as an exponentially weighted moving average (EWMA). The stock price series is $S_0, S_1, \ldots, S_t$:

$$\begin{aligned} x_i &= \log \frac{S_i}{S_{i-1}} \\ \sigma_0^2 &= 250 \sum_{i=1}^{25} x_i/25 \\ \sigma_i^2 &= \lambda \sigma_{i-1}^2 + 250(1-\lambda) x_i^2 \text{ for } 1 \leq i \leq t. \end{aligned} \tag{2.5}$$

We used $\lambda = 0.9$. A plot of these volatilities reveals that volatility varies randomly (Figure 2.3).

The highest peaks correspond to the 2000 dot-com bubble, 9/11 attacks and the 2008 market crash. Also note that the volatility mean reverts and remains within a range most of the time. When we examine the daily absolute returns in Figure 2.4 we note that days with high volatility events are clustered in time. This is termed volatility clustering and indicates that days with large stock movements are likely to be followed by large moves. This is a further indication that returns may not be i.i.d..
The observations made in this section are the constraints that a stochastic process should satisfy in order to capture the observed return dynamics.

2.2.4 Pricing with a smile

For each of the call options with strike K and maturity T in our data set we can back-out the implied volatility (IV) $\sigma(T, K)$. This volatility when used in the BS option pricing formula matches observed market prices. The collection of implied volatilities (IVs) for every strike and maturity combination forms the IV surface. The surface varies by strike (the smile or skew effect) and by maturity. The BS model dictates that the same volatility parameter must be used to price options of all strikes and maturities. This is another shortcoming of the BS model. Market participants however make allowances for these defects in their prices. We seek a model that can capture the observed skew and term structure of option prices.

In summary, an approximately correct model must be able to satisfy the constraints imposed by the empirical returns. The Weiner process does not satisfy these constraints.

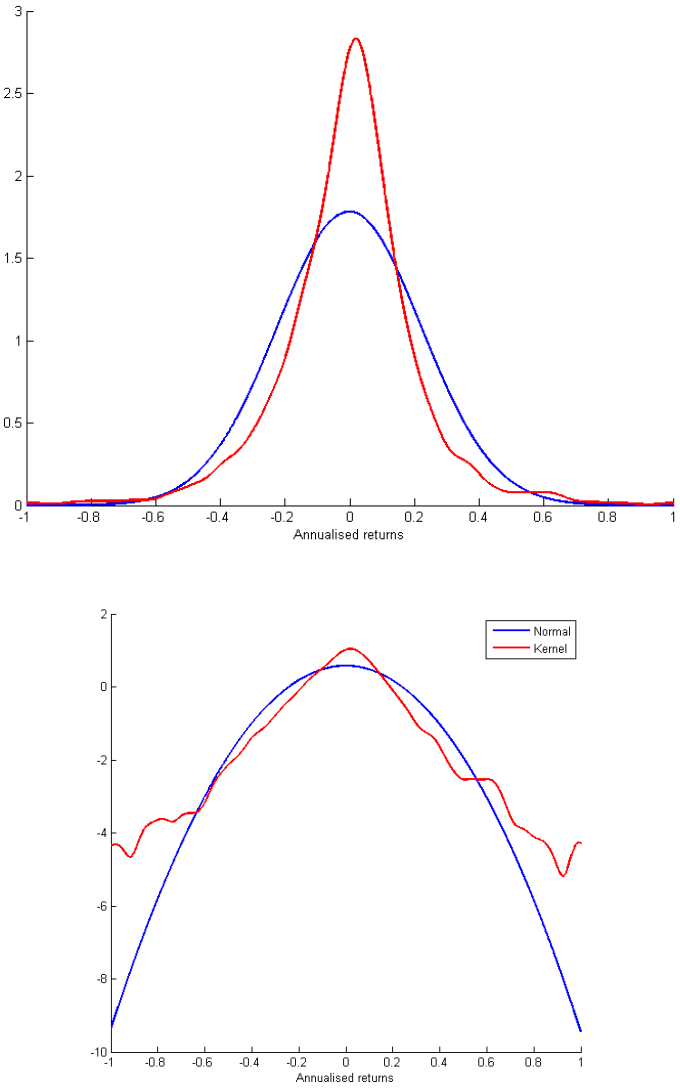

Figure 2.2: PDF and Logged PDF plot of empirical and Normal distribution.

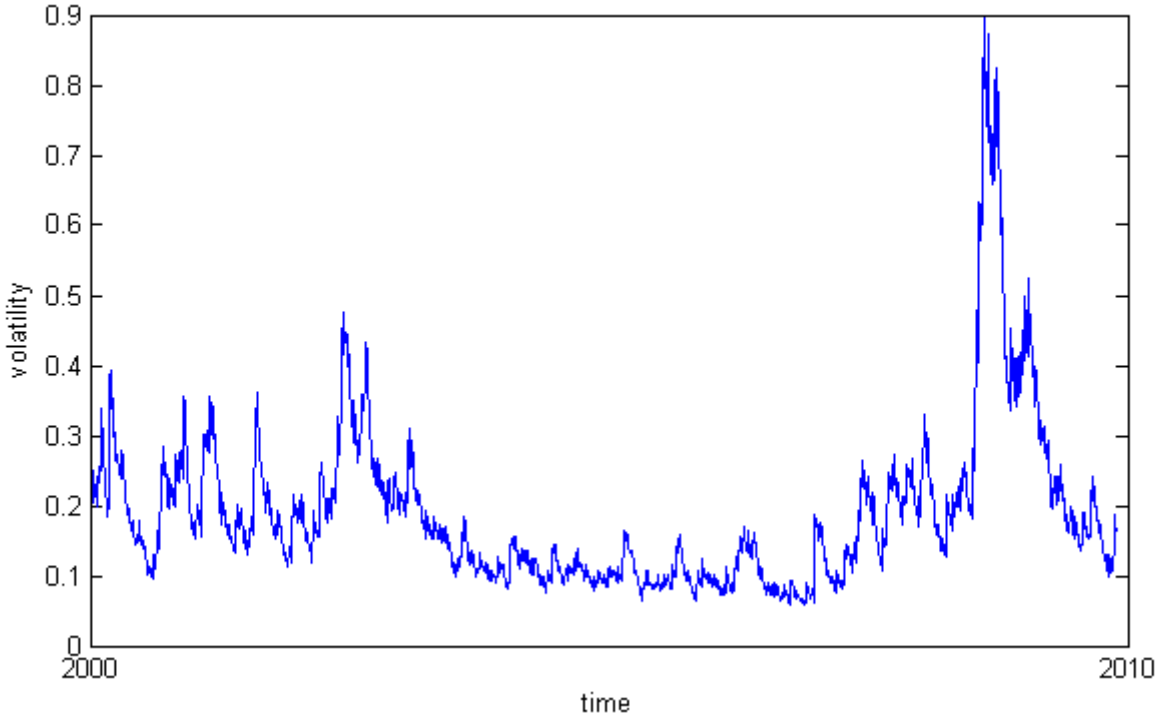

Figure 2.3: Plot of EWMA historical annualised volatility.

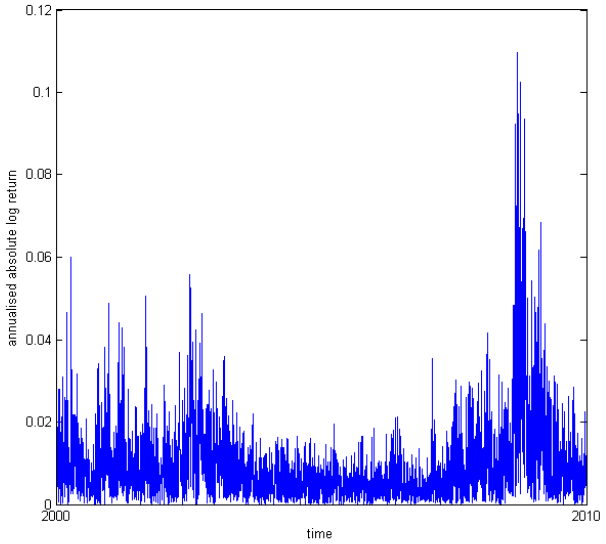

Figure 2.4: Plot S&P500 absolute daily log returns.

3 Lévy processes

We introduce Lévy processes and study the characteristic function of a general Lévy process. The characteristic function is the central tool to most financial applications of Lévy processes. We then examine the theoretical properties of general Lévy processes with the aid of the characteristic function. We contrast these properties with real financial phenomena thus giving us insight about the suitability of Lévy processes for modelling. We then look at examples of Lévy processes.

3.1 Definition

A Lévy process $\mathbf{L} = \{L_t : t \geq 0\}$ is a $\mathbb{R}$-valued stochastic process on a probability space $(\Omega, F, \mathbb{P})$ which satisfies these conditions:

- The paths of $\mathbf{L}$ are (almost surely) right-continuous with left limits (*càdlàg*).
- $\mathbf{L}$ has independent increments; for $0 \leq s \leq t, L_t - L_s$ is independent of L_u for $u \leq s$.
- $\mathbf{L}$ has stationary increments; for $0 \leq s \leq t, L_t - L_s$ is equal in distribution to L_{t-s}.

The first condition ensures that the paths of $\mathbf{L}$ do not explode (because the left limits must exist). It is implicit in the third condition that $\Pr(L_0 = 0) = 1$ (set $s = t$).

A deterministic linear drift $\{\mu t : t \geq 0, \mu \in \mathbb{R}\}$ satisfies all these conditions and is thus a Lévy process. We also know that SBM, Poisson and the compound Poisson processes satisfy these conditions. Further, the sum of a drift, SBM and an independent compound Poisson process is also a Lévy process.

3.2 Characteristic functions

Characteristic functions are going to play a crucial part when working with Lévy processes. Now we shall derive the characteristic function of a Lévy jump-diffusion. This derivation serves as an indication of how the Lévy-Khintchine formula arises for

characteristics functions of general Lévy processes. Let **L** be a Lévy jump-diffusion which has a drift, a Brownian motion and a mean adjusted compound Poisson:

$$L_t = -\mu t + \sigma W_t + \left(\sum_{i=1}^{N_t} J_i - t\lambda\kappa \right) \tag{3.1}$$

where $\mu \in \mathbb{R}, \sigma \geq 0$, $\mathbf{W} = \{W_t : t \geq 0\}$ is a standard Brownian motion, $\mathbf{N} = \{N_t : t \geq 0\}$ is a Poisson process with jump arrival rate $\lambda > 0$ and $\mathbf{J} = \{J_i : i \geq 1\}$ is an i.i.d. sequence of random variables with distribution function F and $\mathbb{E}[J] = \kappa < \infty$. The function F describes the distribution of the size of the jumps whose arrival is dictated by the Poisson process. All sources of randomness are independent. Note that the mean of the compound Poisson process is

$$\begin{aligned} \mathbb{E}\left[\sum_{i=1}^{N_t} J_i\right] &= \mathbb{E}\left[\mathbb{E}\left[\sum_{i=1}^{n} J_i | N_t = n\right]\right] \\ &= \mathbb{E}\left\{\mathbb{E}\left[nJ |\, N_t = n\right]\right\} \\ &= \mathbb{E}\left\{N_t\kappa\right\} \\ &= (\lambda t)\kappa. \end{aligned}$$

The characteristic function of $\mathbf{L}$, $\phi(u)$, is derived in Equation 3.2.

$$\begin{aligned}
\phi(u) &= \mathbb{E}\left\{e^{iuL_t}\right\} \\
&= \mathbb{E}\left[\exp\left[iu\left(-\mu t + \sigma W_t + \sum_{i=1}^{N_t} J_i - t\lambda\kappa\right)\right]\right] \\
&= e^{-iu\mu t}\mathbb{E}\left[e^{iu\sigma W_t}\exp\left[iu\left(\sum_{i=1}^{N_t} J_i - t\lambda\kappa\right)\right]\right] \\
&= e^{-iu\mu t}\mathbb{E}\left[e^{iu\sigma W_t}\right]\mathbb{E}\left[\exp\left(iu\sum_{i=1}^{N_t} J_i - iut\lambda\kappa\right)\right] \text{ ... independence} \\
&= e^{-iu\mu t}e^{-\frac{1}{2}\sigma^2u^2t}\mathbb{E}\left[\exp\left(iu\sum_{i=1}^{N_t} J_i - iut\lambda\kappa\right)\right] \text{ since } W_t \sim N(0,t) \\
&= e^{-iu\mu t}e^{-\frac{1}{2}\sigma^2u^2t}\mathbb{E}\left[\mathbb{E}\left[\exp\left(iu\sum_{i=1}^{n} J_i - iut\lambda\kappa\right)\right]|N_t = n\right] \\
&= e^{-iu\mu t}e^{-\frac{1}{2}\sigma^2u^2t}\mathbb{E}\left[\mathbb{E}\left[e^{iuJ}\right]^{N_t}\right]e^{-iut\lambda\kappa} \\
&= e^{-iu\mu t}e^{-\frac{1}{2}\sigma^2u^2t}e^{\lambda t\left(\mathbb{E}\left[e^{iuJ}-1\right]\right)}e^{-iut\lambda\kappa} \\
&= e^{-iu\mu t}e^{-\frac{1}{2}\sigma^2u^2t}e^{\lambda t\left(\mathbb{E}\left[e^{iuJ}-1\right]-iu\kappa\right)} \\
&= e^{-iu\mu t-\frac{1}{2}\sigma^2u^2t}\exp\left[\lambda t\left(\mathbb{E}\left[e^{iuJ}-1\right]-iu\mathbb{E}\left[J\right]\right)\right] \\
&= e^{-iu\mu t-\frac{1}{2}\sigma^2u^2t}\exp\left[\lambda t\left(\mathbb{E}\left[e^{iuJ}-1-iuJ\right]\right)\right] \\
&= e^{-iu\mu t-\frac{1}{2}\sigma^2u^2t}\exp\left[\lambda t\left(\int_{\mathbb{R}}(e^{iux}-1-iux)F(dx)\right)\right] \\
\phi(u) &= \exp\left[-t\left(iu\mu + \frac{1}{2}\sigma^2u^2 + \int_{\mathbb{R}}(1-e^{iux}+iux)\lambda F(dx)\right)\right] \qquad (3.2)
\end{aligned}$$

This is a special case of the Lévy-Khintchine theorem which shall be stated further down.

An important property of Lévy processes is infinite divisibility. We say that the law of a random variable L is infinitely divisible if for all $n \in \mathbb{N}$ there exists a random variable

$$L^{(1/n)} \qquad (3.3)$$

such that

$$\phi_L(u) = (\phi_{L^{(\frac{1}{n})}}(u))^n. \qquad (3.4)$$

Examples of infinitely divisible distributions include the normal distribution, the exponential, Poisson distribution, compound Poisson, gamma distribution, the Cauchy distribution, the negative binomial, the geometric distribution, and many other distributions. Counter-examples include the binomial and uniform distributions.

The Lévy process $\mathbf{L}$ is fully characterised by its characteristic function ϕ. The characteristic function is given by the Lévy-Khintchine theorem as

$$\phi(u) = \mathbb{E}\left[\exp(iuL_t)\right] = e^{-t\Psi(u)},\ t \geq 0, \tag{3.5}$$

where the characteristic exponent $\Psi(u)$ is given by

$$\Psi(u) = i\mu u + \frac{1}{2}\sigma^2 u^2 + \int_{\mathbb{R}\setminus\{0\}} (1 - e^{iux} + iux\mathbf{1}_{|x|<1})\Pi(dx) \tag{3.6}$$

for every $u \in \mathbb{R}$. See [CT04] for a derivation. This formula will play a central role when we price options. The triplet (μ, σ, Π) fully specifies the Lévy process L_t. The first component $\mu \in \mathbb{R}$ is the constant drift of the Lévy process. In option pricing it is determined by no-arbitrage relations and thus depends on the other two components. The second component $\sigma \geq 0$ is the constant diffusion coefficient. The third component is the Lévy measure (or jump measure) Π defined on $\mathbb{R}\setminus\{0\}$ and satisfying $\int_{\mathbb{R}\setminus\{0\}} \inf\{1, x^2\}\,\Pi(dx) < \infty$. This condition implies that the tails of Π are finite. The Lévy measure $\Pi(dx)$ is often of the form $\pi(x)dx$ and we refer to π as the Lévy density (or jump density). The Lévy measure Π dictates the arrival rates of jumps of every possible size x.

The Lévy-Khintchine theorem indicates that a general Lévy process consists of three orthogonal parts: a linear deterministic term, a Brownian motion and a pure jump process. So a general Lévy process is of the form

$$-\mu t + \sigma B_t + J_t,\ t \geq 0 \tag{3.7}$$

where $-\mu t$ is the linear deterministic term (contributing $i\mu u$ in $\Psi(u)$), $\mathbf{W} = \{W_t : t \geq 0\}$ is a standard Brownian motion (contributing the term $\sigma^2 u^2/2$ in $\Psi(u)$) and $\mathbf{J} = \{J_t : t \geq 0\}$ is a pure jump process (governed by Π) independent of $\mathbf{W}$. From this representation it is clear that Brownian motion (with and without drift) is the only Lévy process with continuous sample paths.

For the purposes of pricing options we will have to use the extended definition of the characteristic function from being defined on the real line to being defined on the complex plane. In particular, the support of the characteristic function is extended from $u \in \mathbb{R}$ to $u \in D \subseteq \mathbb{C}$, where the characteristic exponent is well-defined.

3.3 Properties

Here we note some of the fine and coarse path properties of Lévy processes. We quote results from the literature that examine in detail some of the fine and coarse path properties of Lévy processes. [KL05] look at these path features under the classifications of creeping, the ability to hit fixed points, drifting and oscillation. These path properties can be used to assess the appropriateness of using a certain Lévy process to price a given instrument.

3.3.1 Path activity

Here we characterise the number of jumps that the sample paths of a pure jump Lévy process takes in any finite time interval. We say that the Lévy process $\mathbf{L}$ exhibits finite activity if, and only if, the integral of the Lévy measure Π on the real line is finite. More formally, a pure jump Lévy process has finite activity if:

$$\int_{\mathbb{R}\setminus\{0\}} \Pi(dx) = \lambda < \infty, \tag{3.8}$$

where λ is the mean arrival rate of jumps. A pure jump process with finite activity has a finite number of jumps in any finite time interval. A pure jump Lévy process exhibits infinite activity if the following integral is infinite:

$$\int_{\mathbb{R}\setminus\{0\}} \Pi(dx) = \lambda = \infty. \tag{3.9}$$

An infinite activity jump process can generate an infinite number of jumps in any finite time interval. In the next section we explore the first variation of pure jump Lévy processes.

3.3.2 Path variation

A Lévy process has paths of bounded variation if and only if

$$\int_{\mathbb{R}} \inf\{1, |x|\}\, \Pi(dx) < \infty. \tag{3.10}$$

Otherwise, the Lévy process is said to be of unbounded variation (a property shared by Brownian motion). A Lévy process with bounded variation is more representative of observed share price trajectories. Any Lévy jump process with finite activity has bounded (finite) variation since there are a finite number of jumps, each of a finite size. So the total absolute distance traversed by the process is finite. On the other hand, infinite activity Lévy jump paths can exhibit bounded or unbounded variation.

For a process of unbounded variation the sum of the small jumps ($0 < \epsilon < 1$) does not converge. The sum of the small jumps compensated by their mean converges. This behaviour leads to the requirement of the truncation term $x\mathbf{1}_{|x|<1}$ in Equation 3.6.

A Lévy process has paths of bounded quadratic variation if and only if

$$\int_{\mathbb{R}} \inf\left\{1, x^2\right\} \Pi(dx) < \infty, \tag{3.11}$$

a necessary condition for the Lévy process to be a semi-martingale. The requirement in Equation 3.11 implies that the tails of Π are finite. Lévy processes have bounded

quadratic variation because Π is a measure satisfying $\int_{\mathbb{R}} \inf\{1, x^2\}\,\Pi(dx) < \infty$ and Brownian motion is also of finite quadratic variation.

The path properties in the following sections have direct links to some features of exotic options. Most of what follows is a reprise of the theory contained in the work of [KL05].

3.3.3 Hitting points

A Lévy process $\mathbf{L}$ can hit a fixed point $x \in \mathbb{R}$ if

$$\Pr\Big(L_t = x \text{ for at least one } t > 0\Big) > 0. \tag{3.12}$$

Let

$$C = \{x \in \mathbb{R} : \Pr\left(L_t = x \text{ for at least one } t > 0\right) > 0\} \tag{3.13}$$

be the set of points that a Lévy process can hit. A Lévy process can hit points if $C \neq \emptyset$. A theorem due to [Kes69] and [Bre71] gives us the following characterisation of hitting points.

Suppose that $\mathbf{L}$ is not a compound Poisson process. Then $\mathbf{L}$ can hit points if and only if

$$\int_{\mathbb{R}} \Re\left(\frac{1}{1+\Psi(u)}\right) du < \infty \tag{3.14}$$

where $\Re(x)$ gives the real component of x. Moreover,

1. If $\mathbf{L}$ has a Brownian component ($\sigma > 0$), then $\mathbf{L}$ can hit points and $C = \mathbb{R}$.
2. If $\sigma = 0$, but $\mathbf{L}$ is of unbounded variation and $\mathbf{L}$ can hit points, then $C = \mathbb{R}$.
3. If $\mathbf{L}$ is of bounded variation, then $\mathbf{L}$ can hit points, if and only if, $\mu \neq \int_{(-1,1)} x\Pi(dx)$. The latter equation is finite because we assume $\mathbf{L}$ is of bounded variation. Here, $C = \mathbb{R}$. If $\mathbf{L}$ or $-\mathbf{L}$ is a subordinator then $C = (0, \infty)$ or $C = (-\infty, 0)$, respectively.

Later in subsection 3.4.2 we shall consider the case of a compound Poisson process. The condition in Equation 3.14 may be tested by evaluating the integral numerically for example by using the `quadgk` function in MATLAB.

A callable put option makes use of the ability of a Lévy process to hit points. A callable put is an exotic option with the same structure as an American option except that the writer has the option to cancel the contract at any time before its expiry. If the writer exercises their option to cancel the contract the holder is paid the claim of an American put plus a constant penalty levied on the writer. This type of option is studied in detail by [KK05] and [KK07]. These authors find that under geometric Brownian motion the optimal exercise strategy for the writer is to cancel the put option when the underlying equals (hits) the strike price, provided that this occurs early enough in the contract.

3.3.4 Does your Lévy process creep?

Let

$$\tau_x^+ = \inf\{t > 0 : L_t > x\} \tag{3.15}$$

be the first passage time for each $x \geq 0$. We adopt the conventions $\inf\{\emptyset\} = \infty$ and if $\tau_x^+ = \infty$, then $L_{\tau_x^+} = \infty$. We say that the Lévy process $\mathbf{L}$ creeps upwards if for all $x \geq 0$

$$\Pr\left(L_{\tau_x^+} = x\right) > 0. \tag{3.16}$$

$\mathbf{L}$ creeps downwards if $-\mathbf{L}$ creeps upwards. This property means that a path of a Lévy process continuously passes through a fixed level x instead of jumping over it with a positive probability. We deduce that a Lévy process which creeps can hit points.

The collective work of [Mil73], [Rog84] and [Vig02] fully characterises upward creeping. The Lévy process $\mathbf{L}$ creeps upwards, if and only if, one of the following three situations occur:

1. $\mathbf{L}$ has bounded variation and $\mu > \int_{(-1,1)} x\Pi(dx)$.
2. $\mathbf{L}$ has a Brownian component, $(\sigma > 0)$.
3. $\mathbf{L}$ has unbounded variation, no Brownian component and

$$\int_0^1 \frac{x\Pi\left([x,\infty)\right)}{\int_{-x}^0 \int_{-1}^y \Pi\left((-\infty,u]\right) dudy} dx < \infty. \tag{3.17}$$

If an option involves first passage then the creeping property becomes relevant. So creeping is an important feature for barrier options and American put option. For example, the optimal exercise strategy for a perpetual American put option is given by the first passage below a fixed value of the underlying Lévy process.

3.4 Examples of Lévy Processes in Finance

In this section we describe the Lévy processes that we shall apply in the rest of the dissertation. This section has its roots in [Sch03]. We shall study the Lévy processes without a drift (or location) parameter, say $m \in \mathbb{R}$. The location parameter will play an important role in no-arbitrage option pricing. The subsequent inclusion of the location parameter does not influence the infinite divisibility property of the distribution. Let the drift-less Lévy process be $\mathbf{L} = \{X_t : t \geq 0\}$. When we equip $\mathbf{L}$ with a drift parameter m we get the new shifted Lévy process

$\mathbf{Y} = \{Y_t := L_t + mt : t \geq 0, m \in \mathbb{R}\}$. The process $\mathbf{Y}$ has characteristic function given by

$$\phi_Y(u) = \phi_L(u) \exp(ium). \tag{3.18}$$

The introduction of the location parameter only impacts the first component of the Lévy triplet. The Lévy triplet changes from $(\mu, \sigma, \Pi(dx))$ to $(\mu + m, \sigma, \Pi(dx))$. The Lévy triplets of the examples given below are included in chapter 10. The path properties discussed in section 3.3 for the given examples are also contained in chapter 10.

3.4.1 Poisson process

An $\mathbb{N}$-valued stochastic process $\mathbf{L} = \{L_t : t \geq 0\}$ is called a Poisson process with rate $\lambda > 0$ if $\mathbf{L}$ satisfies

- The paths of $\mathbf{L}$ are (almost surely) right-continuous with left limits (càdlàg).
- $\mathbf{L}$ has independent increments; for $0 \leq s \leq t, L_t - L_s$ is independent of L_u for $u \leq s$.
- $\mathbf{L}$ has stationary increments; for $0 \leq s \leq t, L_t - L_s$ is equal in distribution to L_{t-s}.
- $\Pr(L_t = k) = \frac{(\lambda t)^k}{k!} e^{-\lambda t}, k \geq 0, t \geq 0$ (Poisson distribution).

The Poisson process has jumps of size 1 spaced by independent exponential random variables with parameter $\lambda > 0$. The Lévy triplet is given by $(0,0,\lambda \mathbf{1}_{\{x=1\}})$ where x is the jump size. The characteristic function of the Poisson process L_t is

$$\phi(u) = \exp(\lambda t(\exp(iu) - 1)) \tag{3.19}$$

Note that $\phi(u; \lambda) = \phi(u; \lambda/n)^n$, thus the Poisson distribution is infinitely divisible.

3.4.2 Compound Poisson processes

Suppose that $Z_i, i \geq 1$ is a sequence of independent and identically distributed random variables with the common distribution F supported on $\mathbb{R}$ but with no atom at 0. Define

$$L_t = \sum_{i=1}^{N_t} Z_i, t \geq 0 \tag{3.20}$$

where $\mathbf{N} = \{N_t : t \geq 0\}$ is a Poisson process (independent of $Z_i, i \geq 1$) with rate $\lambda > 0$. The characteristic function of this compound Poisson process is

$$\phi(u) = \exp\left(-t \int_{\mathbb{R}\setminus\{0\}} \left(1 - e^{iux}\right) \Pi(dx)\right) \tag{3.21}$$

where $\Pi(dx) = \lambda F(dx)$. Note that the compound Poisson process is infinitely divisible.

3.4.3 Gamma process

An $\mathbb{R}^+$-valued ($\mathbb{R}^+ = [0, \infty)$) stochastic process $\mathbf{L} = \{L_t : t \geq 0\}$ is called a Gamma process with parameters $a > 0$ and $b > 0$ if $\mathbf{L}$ satisfies the usual conditions and $L_t \sim \Gamma(at, b)$ (Gamma distribution).

The density of L_t is given by

$$f(x) = \frac{b^a}{\Gamma(a)} x^{a-1} e^{-bx}, x > 0. \tag{3.22}$$

The characteristic function is

$$\phi(u) = (1 - iu/b)^{-at}. \tag{3.23}$$

From this we can see that $\phi(u; a, b) = \phi(u; a/n, b)^n$, thus the gamma distribution is infinitely divisible. We shall use the gamma process to construct another Lévy process called the variance gamma. We also use the gamma process to model stochastic volatility.

3.4.4 Inverse Gaussian process

Let $T^{(a,b)}$ be the first time $\mathbf{Y} = \{Y_t := W_t + bt\}$ (a Brownian motion with drift) reaches the fixed level $a > 0$. That is

$$L^{(a,b)} = \inf\{t > 0 : Y_t = a\} \tag{3.24}$$

This first passage time has support $(0, \infty)$ and is distributed according to an inverse Gaussian distribution. The PDF of IG(a, b) is given by

$$f(x) = \frac{a}{\sqrt{2\pi x^3}} \exp(ab) \exp\left(-\frac{1}{2}\left(a^2 x^{-1} + b^2 x\right)\right), \ x > 0 \tag{3.25}$$

where $a > 0$ and $b > 0$. If $\mathbf{L} \sim$IG(a, b) then $c\mathbf{L} \sim$IG$(\sqrt{c}a, b/\sqrt{c})$ where $c > 0$. The IG(a, b) law has characteristic function given by

$$\phi(u) = \exp\left(-a\left(\sqrt{-2iu + b^2} - b\right)\right). \tag{3.26}$$

From this we can see that $\phi(u; a, b) = \phi(u; a/n, b)^n$, thus the IG distribution is infinitely divisible. The IG process $\mathbf{L} = \{L_t : t \geq 0\}$ with parameters $a, b > 0$ is defined as the process which is càdlàg, has independent increments and stationary increments with $L_t \sim \text{IG}(at, b)$. We use the IG process in the construction of the normal inverse Gaussian (NIG) as well as for modelling stochastic volatility.

3.4.5 Variance gamma process

The work of [MCC98] gives a detailed treatment of the VG process and its application in finance. The variance gamma law $\mathrm{VG}(\sigma, \nu, \theta)$ is define on the whole real line and has probability density function given by

$$f(x) = \frac{2\exp\left(\frac{\theta x}{\sigma^2}\right)}{\nu^{\frac{1}{\nu}}\sqrt{2\pi}\sigma\Gamma\left(\frac{1}{\nu}\right)}\left(\frac{x^2}{\frac{2\sigma^2}{\nu}+\theta^2}\right)^{\frac{1}{2\nu}-\frac{1}{4}}\mathbf{K}_{\frac{1}{2\nu}-\frac{1}{2}}\left(\frac{1}{\sigma^2}\sqrt{x^2\left(\frac{2\sigma^2}{\nu}+\theta^2\right)}\right) \tag{3.27}$$

where $\theta \in \mathbb{R}, \sigma > 0, \nu > 0$. $\mathbf{K}_\nu(z)$ is the modified Bessel function of the third kind,

$$\mathbf{K}_\nu(z) = \frac{1}{2}\int_0^\infty y^{\nu-1}\exp\left(-\frac{1}{2}z\left(y+y^{-1}\right)\right)dy. \tag{3.28}$$

The book by [PFTV92] gives code for evaluating Bessel functions for integer or fractional indices ν. The characteristic function is given by

$$\phi(u) = \left(\frac{1}{1-i\theta\nu u+\frac{1}{2}\sigma^2\nu u^2}\right)^{\frac{1}{\nu}}. \tag{3.29}$$

Note that $\phi(u;\sigma,\nu,\theta) = \phi(u;\sigma\sqrt{n},\nu/n,n\theta)^n$, thus the VG distribution is infinitely divisible. The first four theoretical moments of the VG distribution are given in Table 3.1.

Table 3.1: Theoretical moments of the variance gamma process.

	$\mathrm{VG}(\sigma,\nu,\theta)$
mean	θ
variance	$\sigma^2+\nu\theta^2$
skewness	$\theta\nu\left(3\sigma^2+2\nu\theta^2\right)/\left(\sigma^2+\nu\theta^2\right)^{3/2}$
kurtosis	$3\left(1+2\nu-\nu\sigma^4\left(\sigma^2+\nu\theta^2\right)^{-2}\right)$

When $\theta = 0$ the distribution is symmetric. When $\theta < 0$ there is negative skewness. The parameter ν mainly controls the kurtosis. Under a different parametrisation $\mathrm{VG}(C,G,M)$ where

$$\begin{aligned} C &= \frac{1}{\nu} > 0, \\ G &= \left(\sqrt{\frac{1}{4}\theta^2\nu^2+\frac{1}{2}\sigma^2\nu}-\frac{1}{2}\theta\nu\right)^{-1} > 0 \\ M &= \left(\sqrt{\frac{1}{4}\theta^2\nu^2+\frac{1}{2}\sigma^2\nu}+\frac{1}{2}\theta\nu\right)^{-1} > 0 \end{aligned} \tag{3.30}$$

the characteristic function of the law of $\mathrm{VG}(C, G, M)$ is given by

$$\phi(u) = \left(\frac{GM}{GM + (M - G)\, iu + u^2}\right)^C. \tag{3.31}$$

Note that $\phi\left(u; C, G, M\right) = \phi\left(u; C/n, G, M\right)^n$. We define the VG process $\mathbf{L} = \{L_t : t \geq 0\}$ as the process which is càdlàg, has independent and stationary increments such that $L_t \sim \mathrm{VG}\left(tC, G, M\right)$. The theoretical moments under this parametrisation are in chapter 10. The MATLAB code for the characteristic function is in section 11.1 and the code for the moments is in section 11.3.

3.4.6 CGMY process

The CGMY process was first studied by [CGMY02]. The $\mathrm{CGMY}(C, G, M, Y)$ law has characteristic function

$$\phi\left(u\right) = \exp\left(C\Gamma\left(-Y\right)\left[\left(M - iu\right)^Y - M^Y + \left(G + iu\right)^Y - G^Y\right]\right). \tag{3.32}$$

where $C, G, M > 0$ and $Y < 2$. For $Y = 0$ and $Y = 1$ the characteristic function takes different forms. CGMY is infinitely divisible since $\phi(u; C, G, M, Y) = \phi(u; C/n, G, M, Y)^n$. Thus we can define the CGMY Lévy process $\mathbf{L} = \{L_t : t \geq 0\}$ and $L_t \sim CGMY\left(tC, G, M, Y\right)$. We require $Y < 2$ for the Lévy measure to be valid and to ensure finite quadratic variation. The difference in G and M generates asymmetry in the tails of the distribution. The moments of the CGMY process are given in chapter 10. The MATLAB code for the characteristic function is in section 11.1 and the code for the moments is in section 11.3.

The path properties of the CGMY process are determined by the power coefficient Y (see chapter 10). This parameter controls the arrival frequency of small jumps and hence the jump type. From the Lévy measure of the CGMY law one can deduce that the $\mathrm{VG}(C, G, M)$ process can be obtained from the CGMY process by letting Y tend to zero.

3.4.7 Normal inverse Gaussian process

Refer to [BN97] for a full discussion of the NIG process and its application in finance. This process was used as early as 1995 in a research report by Barndorff-Nielsen. The probability density function of the normal inverse Gaussian distribution $\mathrm{NIG}(\alpha, \beta, \delta)$ lives on the entire real line and is given by

$$f(x) = \frac{\alpha\delta}{\pi} \exp\left(\delta\sqrt{\alpha^2 - \beta^2} + \beta x\right) \frac{\mathbf{K}_1\left(\alpha\sqrt{\delta^2 + x^2}\right)}{\sqrt{\delta^2 + x^2}} \tag{3.33}$$

where $\alpha, \delta > 0$ and $\beta < |\alpha|$ and $\mathbf{K}_\lambda\left(z\right)$ is the modified Bessel function of the third kind. Later we shall use this pdf to estimate the NIG parameters by maximum

likelihood. MLE requires numerous evaluations of the pdf which can be computationally expensive. To slightly reduce the computational burden one can precompute or simplify certain expressions. For example, for the NIG pdf one should pre-compute $\sqrt{\delta^2+x^2} = \sqrt{\delta\times\delta + x\times x}$ since it is used twice. Another simplification is $\alpha^2-\beta^2 = (\alpha-\beta)(\alpha+\beta)$. One should constantly look out for these opportunities and be wary of expressions that may result in numerical overflow or underflow.

The characteristic function of the NIG(α,β,δ) is

$$\phi(u) = \exp\left(-\delta\left(\sqrt{\alpha^2-(\beta+iu)^2}-\sqrt{\alpha^2-\beta^2}\right)\right). \tag{3.34}$$

Note that $\phi(u;\alpha,\beta,\delta) = \phi(u;\alpha,\beta,\delta/n)^n$. Since the NIG law is infinitely divisible we can define the NIG process $\mathbf{X} = \{X_t : t \geq 0\}$ with $X_t \sim$NIG$(\alpha,\beta,\delta t)$. Note that the expression for the NIG characteristic function can be manipulated to reduce the number of operations.

The moments of the NIG process are given in chapter 10. The MATLAB code for the characteristic function is in section 11.1 and the code for the moments is in section 11.3. If $\beta = 0$, the distribution is symmetric and has a mean of zero. In fact all the odd moments are zero when $\beta = 0$.

3.4.8 Generalised hyperbolic process

The GH process applications in finance are studied in [Pra99]. The generalised hyperbolic distribution GH$(\alpha,\beta,\delta,\nu)$ has probability density function supported on the real line given by

$$\begin{aligned} f(x) &= a(\alpha,\beta,\delta,\nu)\left(\delta^2+x^2\right)^{\left(\nu-\frac{1}{2}\right)/2}\mathbf{K}_{\nu-\frac{1}{2}}\left(\delta\sqrt{\delta^2+x^2}\right)\exp(\beta x), \\ a(\alpha,\beta,\delta,\nu) &= \frac{\left(\alpha^2-\beta^2\right)^{\nu/2}}{\sqrt{2\pi}\alpha^{\nu-\frac{1}{2}}\delta^\nu\mathbf{K}_\nu\left(\delta\sqrt{\alpha^2-\beta^2}\right)} \end{aligned} \tag{3.35}$$

where

$$\begin{aligned} &\delta \geq 0,\ |\beta| < \alpha \text{ if } \nu > 0, \\ &\delta > 0,\ |\beta| < \alpha \text{ if } \nu = 0, \\ &\delta > 0,\ |\beta| \leq \alpha \text{ if } \nu < 0. \end{aligned} \tag{3.36}$$

The function $\mathbf{K}_\lambda(z)$ is the modified Bessel function of the third kind. One should precompute $\sqrt{\delta\times\delta + x\times x}$ and $\sqrt{(\alpha-\beta)(\alpha+\beta)}$. The characteristic function of the GH$(\alpha,\beta,\delta,\nu)$ is

$$\phi(u) = \left(\frac{\alpha^2-\beta^2}{\alpha^2-(\beta+iu)^2}\right)^{\nu/2}\frac{\mathbf{K}_\nu\left(\delta\sqrt{\alpha^2-(\beta+iu)^2}\right)}{\mathbf{K}_\nu\left(\delta\sqrt{\alpha^2-\beta^2}\right)}. \tag{3.37}$$

This distribution turns out to be infinitely divisible. The proof can be found in [Hal79]. Thus we can define a GH Lévy process X_t with characteristic function $\phi(u)^t$. Note that the characteristic function has expressions that can be pre-computed and factorised, e.g. $\alpha^2 - (\beta + iu)^2 = (\alpha - \beta - iu)(\alpha + \beta + iu)$. The Lévy measure is given by

$$\Pi(dx) = \frac{\exp(\beta x)}{|x|}\left(\int_0^\infty \frac{\exp\left(-|x|\sqrt{2y+\alpha^2}\right)}{\pi^2 y\left[\mathbf{J}^2_{|\nu|}(\delta\sqrt{2y}) + \mathbf{N}^2_{|\nu|}(\delta\sqrt{2y})\right]}dy + \mathbf{1}_{(\nu\geq 0)}\nu\exp(-\alpha|x|)\right) \tag{3.38}$$

where the $\mathbf{J}_\nu(z)$ and $\mathbf{N}_\nu(z)$ are Bessel functions of the first and second kind respectively.

$$\mathbf{J}_\nu(z) = \left(\frac{z}{2}\right)^\nu \sum_{k=0}^{\infty} \frac{(-z^2/4)^k}{k!\Gamma(\nu+k+1)} \tag{3.39}$$

$$\mathbf{N}_\nu(z) = \frac{\mathbf{J}_\nu(z)\cos(\nu\pi) - \mathbf{J}_{-\nu}(z)}{\sin(\nu\pi)} \tag{3.40}$$

The first two moments of the GH process are given in chapter 10. The MATLAB code for the characteristic function is in section 11.1. Higher moments are not known analytically so they must be determined numerically. This issue is addressed later in subsection 4.2.1. The MATLAB code for the moments approximation is in section 11.3.

For $\nu = -1/2$ we obtain NIG(α, β, δ). The NIG process shares the path characteristics of the GH process.

The VG$(\sigma, \nu^{VG}, \theta)$ process can be obtained from the GH process by taking $\nu = \sigma^2/\nu^{VG}, \alpha = \sqrt{(2/\nu^{VG}) + (\theta^2/\sigma^3)}, \beta = \theta/\sigma^2$ and $\delta \to 0$. However, the path properties of the VG processes cannot be deduced from those of the GH process since the VG process was obtained by a limiting procedure.

3.4.9 Meixner process

See [ST01] for a thorough treatment of the Meixner process applied in finance. The Meixner(α, β, δ) distribution has a probability density function given by

$$f(x) = \frac{(2\cos(\beta/2))^{2\delta}}{2\alpha\pi\Gamma(2\delta)}\exp\left(\frac{\beta x}{\alpha}\right)\left|\Gamma\left(\delta + \frac{ix}{\alpha}\right)\right|^2 \tag{3.41}$$

where $\alpha, \delta > 0$ and $|\beta| < \pi$. Note that the pdf requires an evaluation of the gamma function for a complex argument. Some computing software such as MATLAB don't have such a gamma function. You have to code the algorithm yourself (see

[PFTV92]) or find a reliable function on the web. The Meixner(α, β, δ) characteristic function is

$$\phi(u) = \left(\frac{\cos(\beta/2)}{2\cos\left((\alpha u - i\beta)/2\right)} \right)^{2\delta} . \tag{3.42}$$

Since $\phi(u; \alpha, \beta, \delta) = \phi(u; \alpha, \beta, \delta/n)^n$ the Meixner distribution is infinitely divisible. Thus we can define a Meixner Lévy process $\mathbf{L} = \{L_t : t \geq 0\}$ where $L_t \sim$ Meixner$(\alpha, \beta, \delta t)$.

All moments of the Meixner distribution exist. The first four moments of the Meixner process and path properties are given in chapter 10. The MATLAB code for the characteristic function is in section 11.1 and the code for the moments is in section 11.3.

4 The Lévy market model

In chapter 2 we modelled stock returns using SBM. In this chapter we apply the Lévy processes introduced in chapter 3 to better model the behaviour of stock returns. We shall see that the Lévy models are able to capture some of the stylised features of financial markets such as skewness, kurtosis and jumps which are observed in historical price data. We shall also apply these models to European option valuation and try to capture the smile effect observed in option market prices.

4.1 The Lévy market model

In chapter 2, log-returns were modelled using SBM. In chapter 2 we saw that we need a more flexible process to model the log-returns. In this section we replace the SBM in the Black-Scholes-Merton model with a Lévy model $\mathbf{L} = \{L_t : t \geq 0\}$. We model the stock price process as

$$S_t = S_0 \exp(mt + L_t - t \log(\mathbb{E}[e^{L_1}])) = S_0 \exp(mt + L_t - t \log(\phi(-i))) \quad (4.1)$$

where $m \in \mathbb{R}$ is the additional drift, L is the Lévy process governing the returns and $\phi(u)$ is the characteristic function of L_1. We use the Lévy processes of section 3.4 for the process $\mathbf{X}$.

Under the pricing measure with respect to the accumulator numeraire suppose that

$$S_t = S_0 \exp\left((r - q - \omega)t + L_t\right) \quad (4.2)$$

where r is the constant short rate, q is the continuous rate at which the stock pays dividends and $e^{\omega} = \mathbb{E}\left[\exp\left(L_1\right)\right] = \phi(-i)$ compensates for the drift in $\mathbf{X}$ to ensure that $\left(S_t e^{-(r-q)t}\right)_{t\geq 0}$ is a martingale. This assumption is the approach adopted by [EK95], [MCC98] and [BNS01a]. This is termed the mean-correcting martingale measure approach.

Table 4.1 contains the mean compensators for the Lévy processes we have studied in section 3.4.

4.2 Statistical estimation

Now we turn to fitting the Lévy distributions to the empirical distribution of stock returns. The Lévy stock price processes have the construction given in Equation 4.1.

Table 4.1: The ω parameter for the mean-correcting equivalent martingale measure.

Model	$\omega = \log \phi(-i)$
VG	$-C \log\left((M-1)(G+1)/(MG)\right)$
CGMY	$C\Gamma(-Y)\left((M-1)^Y - M^Y + (G+1)^Y - G^Y\right)$
NIG	$-\delta\left(\sqrt{\alpha^2-(\beta+1)^2} - \sqrt{\alpha^2-\beta^2}\right)$
GH	$\log\left(\left(\frac{\alpha^2-\beta^2}{\alpha^2-(\beta+1)^2}\right)^{\nu/2} \frac{\mathbf{K}_n u\left(\delta\sqrt{\alpha^2-(\beta+1)^2}\right)}{\mathbf{K}_n u\left(\delta\sqrt{\alpha^2-\beta^2}\right)}\right)$
Meixner	$2\delta\left(\log(\cos(\beta/2)) - \log(\cos([\alpha+\beta]/2))\right)$

Suppose that we have N log returns $x_t = \log(S_t/S_{t-1})$ where $t = 1, \ldots, N$. These returns could be over a time interval of five minutes, one day, one week and other time frequencies. Let $f(x_t;\theta)$ be the statistical probability density function (PDF) for these returns where θ is the set of unknown parameters that must be estimated. We will estimate θ using the method of moments (MOM) and the maximum likelihood estimation (MLE) approach.
Once we have the statistical estimate of our Lévy model we could use it to perform some analysis such as VaR and asset allocation calculations. Some VaR techniques involve Monte Carlo simulation, so we could use these statistical parameters to sample from the Lévy distributions using methods that will be discussed in section 6.2.

4.2.1 Method of moments

We estimate the parameter set θ by equating the theoretical moments of the Lévy distribution such as mean, variance, skewness and kurtosis to their sample counterparts. We then solve the system of equations to obtain the MOM estimate $\hat{\theta}_{MOM}$. The first four central sample moments are given by

$$\begin{aligned} \bar{x} &= \frac{1}{N}\sum_{j=1}^{N} x_j, \quad \bar{\sigma}^2 = \frac{1}{N-1}\sum_{j=1}^{N}(x_j-\bar{x})^2 \\ s &= \frac{1}{N}\sum_{j=1}^{N}\left(\frac{x_j-\bar{x}}{\bar{\sigma}}\right)^3, \quad k = \left[\frac{1}{N}\sum_{j=1}^{N}\left(\frac{x_j-\bar{x}}{\bar{\sigma}}\right)^4\right]. \end{aligned}$$

The formulas of the first four population moments for some of the Lévy models are given in chapter 10. We don't have the formulas for skewness and kurtosis in the GH distribution case. However, we can calculate the moments numerically from the characteristic function. The n^{th}-order non-central moment is

$$\mathbb{E}\left[X^n\right] = \left[\frac{d^n\phi(-iu)}{du^n}\right]_{u=0}. \tag{4.3}$$

It may be difficult to find these derivatives of the characteristic function analytically. One could use the symbolic mathematics capabilities of some of the scientific computing packages such as Mathematica. The formulas will be tediously long and complicated.
We propose the use of a Taylor series expansion of the characteristic function in a neighbourhood of zero to approximate the derivatives of the characteristic functions evaluated at zero (Equation 4.3).
Let u be a real number. Set $f(u) = \phi(-iu)$, thus f is a real-valued function. We use a fourth order Taylor expansion of f about the point u

$$f(u+h) - f(u) = \sum_{n=1}^{4} \frac{f^{(n)}(u)}{n!} h^n + R \tag{4.4}$$

where we assume that these derivatives exist and R is the residual term

$$R = \frac{h^5}{5!} f^{(5)}(u+bh), 0 < b < 1. \tag{4.5}$$

This expansion is always valid when the characteristic function is analytic in the considered region. We set up the following system of equations

$$Y_j \approx \beta_1 X_j + \beta_2 X_j^2 + \beta_3 X_j^3 + \beta_4 X_j^4 + \beta_5 X_j^5, j = 1, \ldots, 5 \tag{4.6}$$

where $Y_j = f(u+h_j) - f(u), X_j = h_j = j/M$ for $j = 1, \ldots, 5$ and $\beta_i := \frac{f^{(i)}(u)}{i!}, i = 1, \ldots, 4$ and $\beta_5 := \frac{f^{(5)}(u+bh)}{5!}$. We can estimate the βs by choosing an appropriate value for M and then solving this system of equations by Gauss reduction. To arrive at the estimates for the theoretical central moments:

- set $u = 0$, $f^{(0)}(0) := f(0)$.
- $f^{(n)}(0) = \beta_n \times n!$.
- $\mu_n = \sum_{j=0}^{n} \binom{n}{j} (-1)^{n-j} \mathbb{E}[X^j] \mathbb{E}[X]^{n-j} = \sum_{j=0}^{n} \binom{n}{j} (-1)^{n-j} f^{(j)}(0) [f^{(1)}(0)]^{n-j}$.

where μ_n is the n^{th} centred moment. This simple algorithm can be easily extended for estimating higher order derivatives. The difficulty is choosing M big enough so that we get accurate estimates of the derivatives in Equation 4.4 and small enough so that the higher order derivatives are not zero in the neighbourhood. While not mathematically rigorous we choose M such that the fourth term is smaller in magnitude than some small ϵ times the magnitude of the sum of the first three terms of Equation 4.6. We used ϵ =10e-5. This approach is useful when the moments of the Lévy distribution are not known explicitly as in the GH case and stochastic time-changed Lévy processes. Below is the table of parameters for some Lévy distributions that match the annualised daily returns of the S&P500 index given in Table 2.1. The parameter values in Table 4.2 are those that match the first four moments of annualised S&P500 returns. The sample code to produce the MOM parameter estimates is given in Listing 4.1.

Table 4.2: MOM parameter estimates based on S&P500 daily returns.

Model	m	Parameters			
		C	G	M	
VG	0.0234	0.6447	4.9802	5.1782	
		α	β	δ	
NIG	0.0234	3.5950	-0.0991	0.1797	
		C	G	M	Y
CGMY	0.0234	105.9336	10.5550	10.7528	-2.6664
		α	β	δ	
Meixner	0.0234	0.6818	-0.0675	0.2151	
		α	β	δ	ν
GH	0.0234	73.1451	-34.0787	2.7563	-10.1428

```
%Script to run routine to estimate statistical parameters by
    method of
%moments (MOM).

%Levy model characteristic function and initial guess of
   model parameters.
chatfun=@(u)charMeixner(u,pars,data);
x0=[1 1 1];

%Function containing the model parameter constraints.
constraints=@constraintMeixner;

%Function of theoretical moments.
moments=@momentsMeixner;
% moments=@(chatfun)momentsApprox(chatfun,0,4,0.001,10e-6,10
   e-5); %Function approximate moments using the
   characterastic function.

%Sample moments of the S&P500 index anualised daily returns.
%sampleMoments=[mean, variance, skew, kurtosis];
sampleMoments=[-0.0019 0.0500 -0.1029 10.6606];

%Run the parameter estimation.
[pars, estimateError]=sampleMOM(chatfun,x0,constraints,
   moments,sampleMoments);
```

Listing 4.1: Sample code to run calibration for the Meixner process to the first four moments.

4.2.2 Maximum likelihood estimators

Speaking from a probabilistic perspective, maximum likelihood estimation (MLE) allows us to choose those parameter values that maximise the likelihood of the data occurring. We want to find the parameter set θ that maximises the log-likelihood function given by

$$L(\theta) = \sum_{t=1}^{N} \log f(x_t; \theta). \tag{4.7}$$

The PDF's of some of the Lévy models are available in closed form and given in section 3.4. Even when the explicit form of the PDF is not known we may evaluate it through the characteristic function. The characteristic functions of our Lévy models have an analytic form and are given in section 3.4. We may evaluate the PDF by the Fourier transform

$$f(x_t; \theta) = \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{-iux_t} \phi_{\log(S_t/S_{t-1})}(u) du. \tag{4.8}$$

We use the fast Fourier transform (FFT) method to approximate the integral in Equation 4.8.
We will maximise the objective function L in Equation 4.7 numerically. Numerical optimisation will require that we evaluate the objective function L a lot of times. The calculation time of the objective function depends on the number of return observations N and the computational cost of evaluating the PDF. For each valuation of the objective function we have to evaluate the PDF for every data point $x_t, t = 1. \ldots, N$. This can be computationally expensive if we have a lot of data. Another concern is when we have to approximate the PDF using the FFT. The objective function evaluation will be even slower. Thus we need a way to address these issues.
To work around having too many return observations we shall maximise the log-likelihood of the binned data. We use the data to construct a histogram with M bins which are much fewer than the number of return observations N. Each of the M bins is centred at $a_i, i = 1, \ldots, M$. The PDF will be evaluated at the central points a_i. Also, the histogram will place a proportion p_i of the data in the bin centred around a_i. The binned log-likelihood is therefore given by

$$L(\theta) = \sum_{i=1}^{M} p_i \log f(a_i; \theta) \tag{4.9}$$

where $M << N$. We have thus reduced the effort of calculating the objective function. Note that the we can speed up the computations by employing parallel computing.

Another consideration is the optimisation algorithm used. Many objective functions in finance such as Equation 4.7 have multiple local minima. To help deal with this

issue one should combine local and global optimisation techniques. For all the optimisation results in this work we used a combination of the downhill simplex method (local) and particle swarm (global) optimisers. We used a hybrid particle swarm optimiser (combination of downhill simplex method (local) and particle swarm (global) optimisers) based on the work of [Leo04] coded in MATLAB. The global strategy optimises the location of a set of solutions in the parameter space. The local optimiser is used to improve the location of the solution sets produced in the global search. We could also use a gradient based local optimiser which converges faster than the downhill simplex method. We preferred the slower local optimiser since it doesn't terminate when the global routine produces parameter guesses where the Lévy model is not defined. One could use adjoint automatic differentiation to efficiently calculate the gradient of the objective function. Automatic differentiation is a method in computer science to evaluate the derivatives of a computer function with respect to its arguments. The adjoint automatic differentiation is more efficient than the finite difference derivative estimates used by most optimisation tools. More details on this technique are presented in [Gri92].

The MLE parameters for the demeaned S&P500 time series are in Table 4.3. Any parameter estimates should be reported with their error estimates. The errors may be estimated using cross-validation or other methods. We omit error estimates in this work. The maximum log-likelihood values attained are in Table 4.4. The VG

Table 4.3: Maximum likelihood parameter estimates.

Model	Parameters			
	θ	σ	ν	
VG	-0.0222	0.2177	1.1997	
	α	β	δ	
NIG	3.1774	-0.8105	0.1487	
	C	G	M	Y
CGMY	0.1009	2.6928	3.0735	0.7685
	α	β	δ	
Meixner	0.6875	-0.5759	0.1966	
	α	β	δ	ν
GH	3.6237	-0.8270	0.1310	-0.2938

process seems to capture the empirical distribution more closely. In Figure 4.1 are

Table 4.4: Log-likelihood evaluated at optimal parameter set.

	VG	NIG	CGMY	Meixner	GH
$L(\theta)$	484.1835	508.6712	484.4350	507.8936	508.3590

qqplots of two processes versus the observed return process. The plot is very close to being linear which indicates that these Lévy processes can be used in models of empirical returns. In particular these processes can capture tail events.

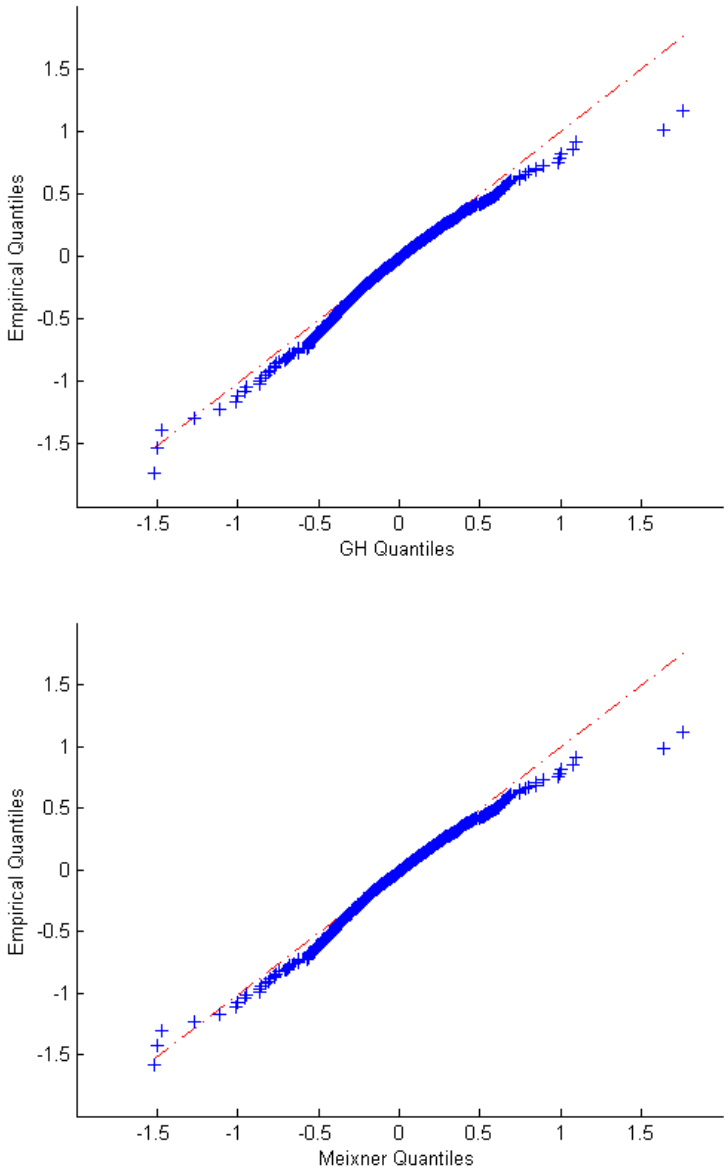

Figure 4.1: qqplot for GH (top) and Meixner (bottom) process.

4.3 Pricing European options

We want to price a European option whose risk-neutral price is given by

$$\begin{aligned} V(S_0, t_0) &= e^{-r(T-t_0)}\mathbb{E}^{\mathbb{Q}}[V(S_T, T)|S_0] \\ &= e^{-r(T-t_0)} \int_{\mathbb{R}} V(y, T) f(y|S_0) dy \end{aligned} \tag{4.10}$$

where $\mathbb{E}^{\mathbb{Q}}[\cdot]$ is the (conditional) expectation operator under the risk-neutral measure $\mathbb{Q}$ and $f(S_T|S_0)$ is the conditional PDF of the stock price.
There are various numerical integration methods discussed in the literature to approximate this integral. Some of these include the FFT implementation of [CM99], the fractional FFT of [Cho04] and the Fourier-cosine series expansion (COS method) of [FO08]. We shall use the COS method to price European options. For full details concerning this method the reader is referred to these authors. We focus on their key results that we shall be using to price our options.

4.3.1 The COS method

The point of departure for the COS method is Equation 4.10:

- Truncate the range of integration in the risk-neutral formula Equation 4.10 to $[a, b] \subset \mathbb{R}$.
- Replace the PDF $f(y|S_0)$ by its cosine series expansion on the truncated range.
- The coefficients in the above series expansion require knowledge of the PDF. The PDF may not always be known whereas the characteristic function may be available in closed-form. The authors thus replace the series coefficients in the latter point by the characteristic function approximation. See [FO08] for full details.
  Each of these three approximations introduce an error to the valuation formula in Equation 4.10. The truncated range of integration can be any bounded interval that spans the support of the pdf and still maintains accuracy. The authors also derived bounds for this error.

[FO08] propose that we set the integration range $[a, b]$ as

$$[a, b] = [c_1 - L\sqrt{c_2}, c_1 + L\sqrt{c_2}] \text{ with } L = 10 \tag{4.11}$$

where c_n denotes the n-th central moment of $\log(S_T)$. So the range covers ± 10 standard deviations from the mean. In section 3.4 we listed the first four moments of most of the Lévy distributions used in this work. For Lévy models where the moments are difficult to derive we use the approximation method in subsection 4.2.1.

For short maturities the pdf's of many Lévy processes are highly leptokurtic. This leptokurtic behaviour is captured by the kurtosis, c_4. For maturities in the range $T = 0.1$ and $T = 10$ the authors propose using the range given by

$$[a, b] = \left[c_1 - L\sqrt{c_2 + \sqrt{c_4}}, c_1 + L\sqrt{c_2 + \sqrt{c_4}}\right] \text{ with } L = 10. \tag{4.12}$$

Larger values of L give a wider a range of integration. However, larger values of L require a larger number of integration points N to achieve a high level of accuracy. For extremely short maturities like $T = 0.001$ a more accurate truncation rule is given by

$$[a, b] = \left[c_1 - L\sqrt{c_2 + \sqrt{c_4 + \sqrt{c_6}}}, c_1 + L\sqrt{c_2 + \sqrt{c_4 + \sqrt{c_6}}}\right] \tag{4.13}$$

The sixth moment c_6 is difficult to derive in closed form. One can approximate it using the methods of subsection 4.2.1. We can use the COS method to value European call and put options. However, the COS method becomes inaccurate when used to price call options when we use a wide range $[a, b]$. The authors attribute this to the fact that the call pay-off grows exponentially with the log of the stock price. This introduces large cancellation errors for wide intervals of integration. On the other hand, the pay-off of a put option is bounded by the strike price K and thus does not suffer from the cancellation error observed with call options. We value put options by the COS method and use put-call parity to find the price of call options. The MATLAB code to price a put option by the COS method is in section 11.4. The code in Listing 4.2 illustrates how to use the supplied code to price a European put and call option.

```
%Example script to price a call option using the COS method.

%Set up market inputs.
rfr=0.05;     q=0.02;   K=100;   S0=100;    T=1;
data=[rfr,T,q,K,S0];

%Levy model characteristic function and model parameters.
pars=[0.3977  -1.4940  0.3462];
chatfun=@(u)charMeixner(u,pars,data);

%Stochastic Volatility Levy model characteristic function
    and model parameters.
%chatfun1=@charMeixner;
%pars1=[0.1231  -0.5875  3.3588];
%chatfun2=@charIntCIR;
%pars2=[0.5705  1.5863  1.9592  1];
%chatfun=@(u)TimeChangedchar(u,chatfun1,pars1,chatfun2,pars2
    ,data);
```

```
%Price a put option and then a call option.
gridCOS=8*1024;          %number of grid points for COS
    pricing.
pricePut=COS(-220,200,data,gridCOS,chatfun,K);
priceCall=putcall(rfr,T,q,pricePut,K,S0);        %Obtaining a
     call price using the put price using put-call parity.
```

Listing 4.2: Sample code to price European vanilla options using the COS method.

4.4 Risk-neutral estimation

Here we search for the parameter set of our Lévy models that will produce model prices that match the market prices of European options. This procedure is referred to as calibration.

4.4.1 Data selection

Market prices of options may be quoted using closing, bid, ask and mid (mean of bid and ask) prices. The market quotes used in the calibration process are an important consideration. For liquid options today's quoted bids and asks will match and result in trades. Thus the closing price will not be far from the bid and ask prices. Illiquid options include deep out-the-money options and long term options. For infrequently traded options the bids and asks on a particular day may not match and no trade takes place. Thus, the closing price for that day may actually reflect a trade that occurred some days ago. This will result in the closing price being far from the bid and ask prices. This stale data does not express the true value of the option. Thus the bid and ask prices may be better to use than the closing prices. In the sequel we shall use the mid prices (mean of bid and ask) in our calibrations.
Note that these option prices are likely incorporate market factors that are not accurately captured by our models such as tax and dividend impacts.

4.4.2 Options data

We calibrate to European call options on the S&P500 index. We use the data reported in [Sch03]. The data set contains 75 mid prices of European call options on the S&P500 index on 18 April 2002. On this date the spot price closed at $S_0 = 1124.47$. As in [Sch03] we take the risk-free rate $r = 0.019$ and the continuous dividend rate $q = 0.012$. The continuous dividend rate can be estimated by setting the futures price equal to the theoretical forward price. The option prices are in chapter 9. The market prices are displayed in Figure 4.2.

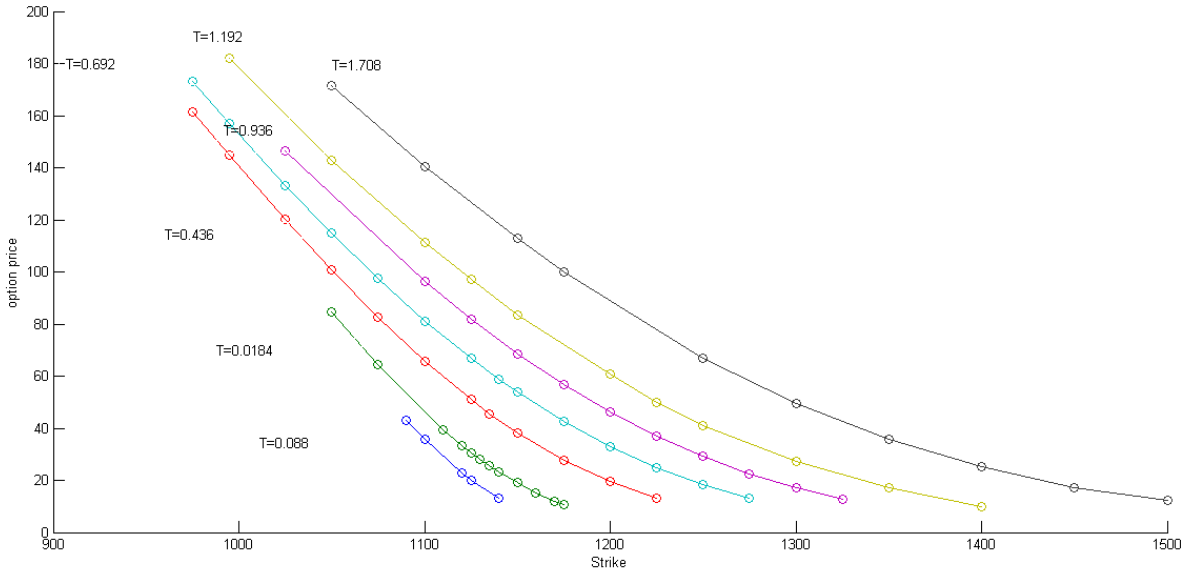

Figure 4.2: Plot of call option prices (in US dollars) on S&P500 on 18 April 2002.

4.4.3 Calibration strategy

The set of parameters obtained by calibrating to these market prices represent the current view of the market on the asset. So we are making the implicit assumption that all necessary information is reflected in today's market prices. We can use these estimated parameters to price OTC exotic options, whose prices are not directly observed in the market, and for spotting mis-pricing in portfolios of European options. We estimate a single set of parameters for each of our Lévy models across all strikes and maturities. This global set of parameters can be used to value path-dependent derivatives. If instead we search for a set of parameters to match prices at a single maturity then these can only be used to price option pay-offs occurring on that specific maturity.
We estimate the model parameters by minimising the square-root of the mean-square error (RMSE) between the observed market prices and the model prices. The RMSE is given by

$$\text{RMSE} = \sqrt{\sum_{\text{options}} \frac{(\text{Market price} - \text{Model price})^2}{\text{number of options}}}. \tag{4.14}$$

The model prices are calculated by the COS method described in subsection 4.3.1 using the characteristic function of Equation 4.2. The range of integration suggested in subsection 4.3.1 is estimated using moment estimation method presented in subsection 4.2.1. To compare the goodness of fit of the different models we use

several measures of fit. We look at the RMSE, average relative percentage error (ARPE), the average absolute error as a percentage of the mean price (APE) and the average absolute error (AAE):

$$\begin{aligned}
\text{ARPE} &= \frac{1}{\text{number of options}} \sum_{\text{options}} \frac{|\text{Market price} - \text{Model price}|}{\text{Market price}} \\
\text{APE} &= \frac{1}{\text{mean option price} \sum_{\text{options}} \frac{|\text{Market price}-\text{Model price}|}{\text{number of options}}} \\
\text{AAE} &= \sum_{\text{options}} \frac{|\text{Market price} - \text{Model price}|}{\text{number of options}}.
\end{aligned}$$

4.4.4 Results

The estimated risk-neutral parameters are in Table 4.5. The sample code to run the calibrations is given in Listing 4.3.

```
%Run the option calibrator.
data=[0.019, 32/365, 0.012, 1100, 1124.47];
rfr=data(1);     q=data(3);  K=data(4);  S0=data(5);

gridCOS=8*1024;       %Number of grid points for cosine
    expansion pricing method.

%char function (constant)
% chatfun=@charMeixner;

%char function (SV)
chatfun1=@CGMYchar;
chatfun2=@charIntCIR;
chatfun=@(u,x1,x2,data)TimeChangedchar(u,chatfun1,x1,
    chatfun2,x2,data);

%Initial model parameter guesses
x1=[7.5072 19.1520 40.0786 0.0392];
x2=[0.7168 0.6922 0.8756];
x0=[x2 x1];

strikes=[1090 1100 1120 1125 1140 1050 1075 1110 1120 1125
    1130 1135 1140 1150 1160 1170 1175 975 995 1025 1050 1075
     1100 1125 1135 1150 1175 1200 1225 975 995 1025 1050
    1075 1100 1125 1140 1150 1175 1200 1225 1250 1275 1025
    1100 1125 1150 1175 1200 1225 1250 1275 1300 1325 995
```

```
    1050 1100 1125 1150 1200 1225 1250 1300 1350 1400 1050
    1100 1150 1175 1250 1300 1350 1400 1450 1500];
market=[43.1 35.6 22.9 20.2 13.3 84.5 64.3 39.5 33.5 30.7
    28.0 25.6 23.2 19.1 15.3 12.1 10.9 161.6 144.8 120.1
    100.7 82.5 65.5 51.0 45.5 38.1 27.7 19.6 13.2 173.3 157.0
     133.1 114.8 97.6 81.2 66.9 58.9 53.9 42.5 33.0 24.9 18.3
     13.2 146.5 96.2 81.7 68.3 56.6 46.1 36.9 29.3 22.5 17.2
    12.8 182.1 143.0 111.3 97.0 83.3 60.9 49.8 41.2 27.1 17.1
     10.1 171.4 140.4 112.8 99.8 66.9 49.5 35.7 25.2 17.0
    12.2];

[x,fval,gfx,output]=calibrateOptions(data,gridCOS,chatfun,
    @constraintCGMY,x0);
```

Listing 4.3: Code used to produce calibrations to option prices.

Table 4.5: Risk neutral estimates.

Model	Parameters			
	C	G	M	
VG	1.4058	6.0046	14.6753	
	α	β	δ	
NIG	6.2332	-3.9606	0.1637	
	C	G	M	Y
CGMY	0.0219	0.0000	7.2094	1.3210
	α	β	δ	
Meixner	0.3910	-1.4908	0.3550	
	α	β	δ	ν
GH	3.2961	-3.2961	0.2149	-1.5436

The fit statistics are in Table 4.6. Note that the CGMY and GH processes have better fits than the other models under all measures. This is to be expected as these models have more parameters (degrees of freedom).

Table 4.6: Risk neutral fit summaries

Model	AAE	APE	ARPE	RSME
VG	2.8332	0.0459	0.0745	3.5555
NIG	2.4272	0.0393	0.0614	3.0984
Meixner	2.4932	0.0404	0.0635	3.1762
CGMY	1.9929	0.0323	0.0471	2.6885
GH	2.1775	0.0353	0.0555	2.8522

Figure 4.3 depicts how accurately the CGMY model (crosses) re-priced the market (circles).

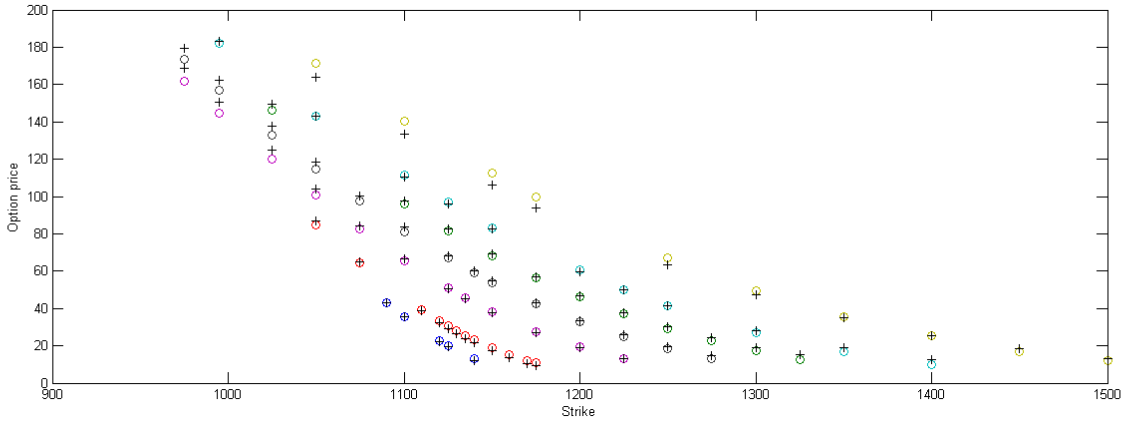

Figure 4.3: Calibration fit plot for CGMY process.

4.4.5 Further considerations for calibration

We would like to fit our models to out-the-money (OTM) prices at each strike. We would define the RMSE on call options when $K > S_0$ and on put options when $K \leq S_0$. This calibration procedure has been adopted as an industry standard for reasons that are described in [CW03]:

- Near-the-money (NTM) options are more expensive than far OTM options of the same maturity. Thus our RMSE optimisation criterion puts more weight on the NTM options than on the far OTM options.
- ITM options have a positive intrinsic value that is not affected by the model but can dominate the total value of the option. OTM options only have time value which is determined entirely by the model.
- If we spot differences between the market prices on the OTM options versus the ITM counterparts, the prices on the OTM options are generally preferred. This discrepancy arises when the put-call parity relation is violated and the two options yield different implied volatilities. We prefer the OTM options because they are more liquid. The greater liquidity may be due to OTM options being a cheaper alternative to hedge against or speculate on increases in future share price volatility.

Another consideration is how we define the pricing errors. The pricing error can be defined using absolute errors (Market price - Model price) or as relative errors (Model price/Market price - 1).

- If we use absolute error the optimisation criterion will tend to assign more weight to options with higher values such as NTM options and options with a larger time to maturity and less weight to short term as well as OTM options.
- The use of relative errors gives equal weight across all options. This may

be undesirable as it will give too much weight to illiquid options when these should logically contribute less.

5 Levy market model with stochastic volatility

In subsection 2.2.3 we noted that the return volatilities were stochastic and displayed clustering. In this chapter we extend the Lévy models employed in chapter 4 to incorporate stochastic volatility with clustering. This addition allows us to refine our approximation to the empirical distribution of asset returns.

5.1 Modelling the stochastic time change

We incorporate stochastic volatility in our models by making time stochastic. In this section we introduce the processes that will be used to model the stochastic clock. We will sub-ordinate our Lévy processes by these stochastic clocks.
In order to be a sub-ordinator, the stochastic time must be an increasing process and thus the activity rate process must be positive (a process with no Brownian component, non-negative drift and positive jumps). In subsection 2.2.3 we noted that the return volatilities display volatility clustering. We use a mean-reverting activity rate to capture this effect of bursts of volatility. We will use the Lévy-driven Ornstein-Uhlenbeck (OU) processes studied by [BNS01b], [BNS01a] and [BNS03] to model the activity rate. The activity rate is modelled as an OU process y_t which is defined as the solution of the stochastic differential equation (SDE) given by

$$dy_t = -\lambda y_t dt + dz_{\lambda t}, \quad y_0 > 0, \tag{5.1}$$

where z is a subordinator and $\lambda > 0$. The multiplication of time in $dz_{\lambda t}$ is to ensure that the marginal distribution of y_t is unchanged whatever the value of λ is. [BNS01b] show that

$$\begin{aligned} y_t &= y_0 e^{-\lambda t} + \int_0^t e^{-\lambda(t-s)} dz_{\lambda s} \\ &= y_0 e^{-\lambda t} + e^{-\lambda t} \int_0^{\lambda t} e^{-\lambda s} dz_s. \end{aligned}$$

Thus y_t is bounded below by $y_0 \exp(-\lambda t)$. We also note that y_t moves up exclusively by the jumps in z_t and then tails off exponentially.

We model the stochastic clock as the integral of the activity rate

$$Y_t = \int_0^t y_s ds. \tag{5.2}$$

For the activity rate y we use the Gamma-OU, IG-OU and CIR processes.

5.1.1 Gamma-OU process

The Γ-OU(a, b, λ) process y_t is given by the solution of the SDE Equation 5.1 where the Lévy driving process $z = \{z_t : t \geq 0\}$ is a compound Poisson process

$$z_t = \sum_{i=1}^{N_t} x_i, \tag{5.3}$$

where $\mathbf{N} = \{N_t : t \geq 0\}$ is a Poisson process with intensity a and $\{x_i : i = 1, 2, \ldots\}$ are i.i.d. random variables with common law $\Gamma(1, b)$. y_t has the marginal law $\Gamma(ab, b)$. The characteristic function of the stochastic clock Y_t (given y_0) is

$$\phi(u) = \exp\left[\frac{iuy_0}{\lambda}(1 - e^{-\lambda t}) + \frac{a\lambda}{iu - b\lambda}\left(b \log\left(\frac{b}{b - iu\lambda^{-1}(1 - e^{-t\lambda})}\right) - iut\right)\right] \tag{5.4}$$

The MATLAB code for the characteristic function is in section 11.1. Note that due to machine roundoff error $\exp(x) - 1$ can be zero for small x. To compute $\exp(x) - 1$ accurately for small values of x we used the `expm1` function in MATLAB.

5.1.2 IG-OU process

The IG-OU(a, b, λ) process y_t is given by the solution to Equation 5.1 where the driving process is defined as the sum of two independent Lévy processes $z = z^{(1)} + z^{(2)}$, where $z^{(1)}$ is given by

$$z^{(1)} = \frac{1}{b^2} \sum_{i=1}^{N_t} \nu_i^2 \tag{5.5}$$

where $\mathbf{N} = \{N_t : t \geq 0\}$ is a Poisson process with intensity $ab/2$ and $\{\nu_i : i = 1, 2, \ldots\}$ are i.i.d. standard normal random variates independent of the Poisson process. The process $z^{(2)}$ is an IG$(a/2, b)$ process.
The characteristic function of the integrated IG-OU process Y_t (given y_0) is

$$\phi(u) = \exp\left(\frac{iuy_0}{\lambda}(1 - e^{-t\lambda}) + \frac{2aiu}{b\lambda}A(u, t)\right), \tag{5.6}$$

where

$$\begin{aligned} A(u,t) &= \frac{1-\sqrt{1+\kappa(1-e^{-t\lambda})}}{\kappa} \\ &\quad + \frac{1}{\sqrt{1+\kappa}}\left(\tanh^{-1}\left(\frac{\sqrt{1+\kappa(1-e^{-t\lambda})}}{\sqrt{1+\kappa}}\right) - \tanh^{-1}\left(\frac{1}{\sqrt{1+\kappa}}\right)\right) \end{aligned}$$

and $\kappa = -2iu/(\lambda b^2)$. The MATLAB code for the characteristic function is in section 11.1.

5.1.3 The CIR process

The CIR process y_t is given by the solution to the SDE

$$dy_t = \kappa(\eta - y_t)dt + \lambda\sqrt{y_t}dW_t, \quad y_0 > 0, \tag{5.7}$$

where $\mathbf{W} = \{W_t : t \geq 0\}$ is a SBM. The characteristic function of the integrated CIR process Y_t (given y_0) is

$$\phi(u) = \frac{\exp(\kappa^2\eta t/\lambda^2)\exp\left(2iuy_0/(\kappa+\gamma\coth(\gamma t/2))\right)}{\left(\cosh(\gamma t/2)+\kappa\sinh(\gamma t/2)\right)^{2\kappa\eta/\lambda^2}} \tag{5.8}$$

where

$$\gamma = \sqrt{\kappa^2 - 2\lambda^2 iu}. \tag{5.9}$$

The MATLAB code for the characteristic function is in section 11.1.

5.2 The Lévy stochastic volatility market model

Now we can incorporate stochastic volatility (SV) in our returns by letting time evolve randomly. We model the risk-neutral stock price as

$$S_t = S_0\exp([r-q-\omega]t + L_{Y_t})$$

where $Y = \{Y_t : t \geq 0\}$ is the business time process, $L = \{L_t : t \geq 0\}$ is a Lévy process and ω the mean-correcting term such that $e^{\omega} = \mathbb{E}[\exp(L_{Y(1)})]$. The characteristic function of $L_{Y(t)}$ is

$$\begin{aligned} \phi(u) &= \mathbb{E}[\exp(iuL_{Y_t})] \\ &= \mathbb{E}\{\mathbb{E}[\exp(iuL_s)|Y(t)=s]\} \end{aligned} \tag{5.10}$$

where the inner expectation is on $L_{Y(t)}$ conditional on $Y_t = s$ and the outer is on Y_t. If the business time Y_t is independent of the Lévy process L_t (i.e. there is no leverage effect) then the above equation simplifies to

$$\phi(u) = \mathbb{E}[e^{-Y_t \Psi_L(u)}] = \phi_{Y(t)}(i \log \phi_{L(1)}(u)) \tag{5.11}$$

using Equation 3.5. [CW04] consider the general case that allows for leverage.

Note that $\phi(u)$ tends to zero as u tends to positive or negative infinity. Taking the log of $\phi_{L(1)}(u)$ for extreme values of u will lead to overflow. Implementation of Equation 5.11 should take this into account. The mean-correcting term is

$$\omega = \log \phi_{Y(1)}(i \log \phi_{L(1)}(-i)). \tag{5.12}$$

The characteristic function of the log returns $s_t := \log(S_t/S_0)$ is

$$\phi_{s_t}(u) = \exp(iu[r - q - \omega]t)\phi_{Y(t)}\left(i \log \phi_{L(1)}(u)\right). \tag{5.13}$$

The code to implement Equation 5.11 is given in section 11.1 as the function TimeChangedchar. We then use this explicit characteristic function to price European vanilla options using the COS function.

5.2.1 Risk neutral estimation

We calibrated the SV models to the option prices in subsection 4.4.2. In the calibrations we set the initial rate of time change to one, $y_0 = 1$. We estimate a global parameter set (across all strikes and maturities) by minimising the RMSE between market and model prices. The risk-neutral parameters for the SV models are in Table 5.1. Figure 5.1 depicts how closely the VG-CIR process fits the market prices. Similar fits are obtained for the Meixner, VG, CGMY and GH SV models. We conclude that our SV models are capable of closely fitting the market skew across a number of maturities. Table 5.2 shows the fit summaries for the SV models. Note that the CGMY and GH produce better fits than the Lévy processes with three parameters. A practical indication for a good fit would be to obtain model prices that are within the market bid-offer range. These calibrations can be used to price European vanilla options that were not in the calibration set, price exotic options, assess model uncertainty or to identify mis-priced portfolios of options.

5.2.2 Call option parameter sensitivities

Here we consider how sensitive the price of a call option is to minor movements in the individual model parameters. We carried out the sensitivity analysis on all 75 call options used in this work under the CGMY-CIR SV model parameters in Table 5.2. Here we show these results for a single option from that set with maturity T=0.088

Table 5.1: Stochastic volatility risk neutral parameter estimates.

Model	Parameters							
	κ	η	λ	y_0	C	G	M	
VG-CIR	0.3781	1.8082	1.6473	1	5.1468	15.6171	25.0345	
	a	b	λ	y_0	C	G	M	
VG-Gamma	0.5798	0.5222	0.7932	1	6.6192	17.0097	32.8467	
	a	b	λ	y_0	C	G	M	
VG-IG	0.7127	0.6975	0.8748	1	8.7246	19.5757	40.8761	
	κ	η	λ	y_0	α	β	δ	
NIG-CIR	0.3538	1.7825	1.5531	1	13.4274	-4.8415	0.3295	
	a	b	λ	y_0	α	β	δ	
NIG-Gamma	0.5850	0.5812	0.7223	1	19.4291	-9.3665	0.3893	
	a	b	λ	y_0	α	β	δ	
NIG-IG	0.7577	0.6925	0.9396	1	20.0553	-9.1138	0.4057	
	κ	η	λ	y_0	α	β	δ	
Meix-CIR	0.5627	1.5901	1.9479	1	0.1214	-0.5840	3.4418	
	a	b	λ	y_0	α	β	δ	
Meix-Gamma	0.5806	0.5454	0.7560	1	0.1340	-1.1442	2.3071	
	a	b	λ	y_0	α	β	δ	
Meix-IG	0.7362	0.6964	0.9173	1	0.1185	-1.1595	2.8165	
	κ	η	λ	y_0	C	G	M	Y
CGMY-CIR	0.3877	1.4562	1.3855	1	0.0073	0.0959	11.2092	1.6782
	a	b	λ	y_0	C	G	M	Y
CGMY-Gamma	0.6209	0.5897	0.9853	1	0.1228	6.7676	22.1925	1.0798
	a	b	λ	y_0	C	G	M	Y
CGMY-IG	0.7113	0.6982	0.8741	1	7.5073	19.1522	40.0785	0.0423
	κ	η	λ	y_0	α	β	δ	ν
GH-CIR	0.3389	1.7426	1.4748	1	5.4600	-4.9883	0.4586	-4.9167
	a	b	λ	y_0	α	β	δ	ν
GH-Gamma	0.5668	0.5349	0.8685	1	13.6955	-7.8429	0.5147	-4.7014
	a	b	λ	y_0	α	β	δ	ν
GH-IG	0.7379	0.6642	1.2070	1	20.2070	-7.9568	0.4801	-1.2925

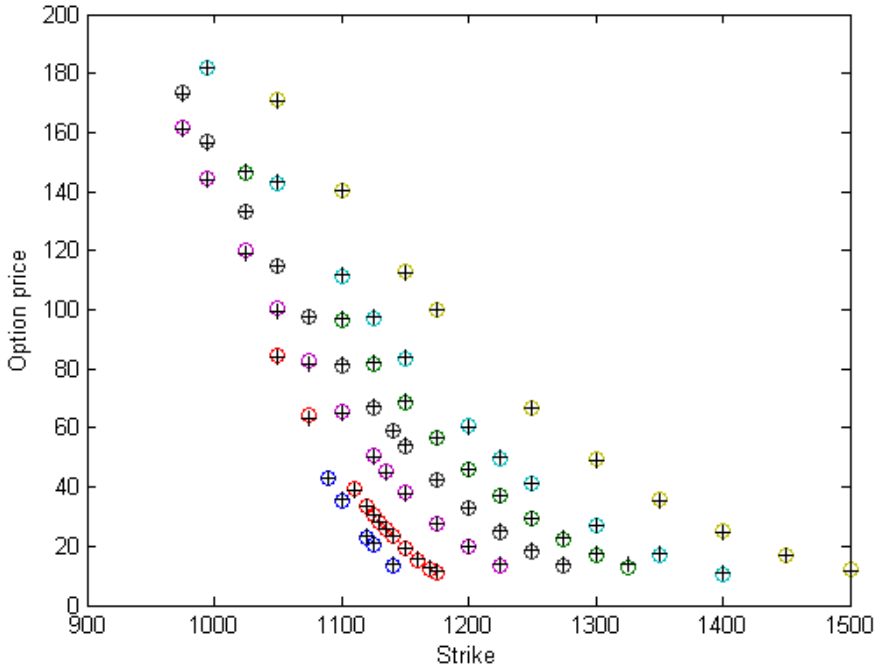

Figure 5.1: VG-CIR calibration fit to S&P500 options (circles are market prices, pluses are model prices).

Table 5.2: Risk neutral fit summaries for stochastic volatility models.

Model	AAE	APE	ARPE	RSME
VG-CIR	0.3860	0.0063	0.0123	0.4740
NIG-CIR	0.3798	0.0062	0.0121	0.4739
Meixner-CIR	0.3840	0.0062	0.0122	0.4746
CGMY-CIR	0.3296	0.0053	0.0106	0.4308
GH-CIR	0.3713	0.0060	0.0119	0.4727
VG-IG	0.3273	0.0053	0.0188	0.4263
NIG-IG	0.3730	0.0060	0.0126	0.4522
Meixner-IG	0.3484	0.0056	0.0122	0.4340
CGMY-IG	0.4023	0.0065	0.0156	0.5158
GH-IG	0.4098	0.0066	0.0135	0.4941
VG-Gamma	0.3101	0.0058	0.0119	0.7932
NIG-Gamma	0.3026	0.0049	0.0114	0.4149
Meixner-Gamma	0.3062	0.0050	0.0116	0.4112
CGMY-Gamma	0.3100	0.0050	0.0113	0.4320
GH-Gamma	0.3319	0.0054	0.0123	0.4570

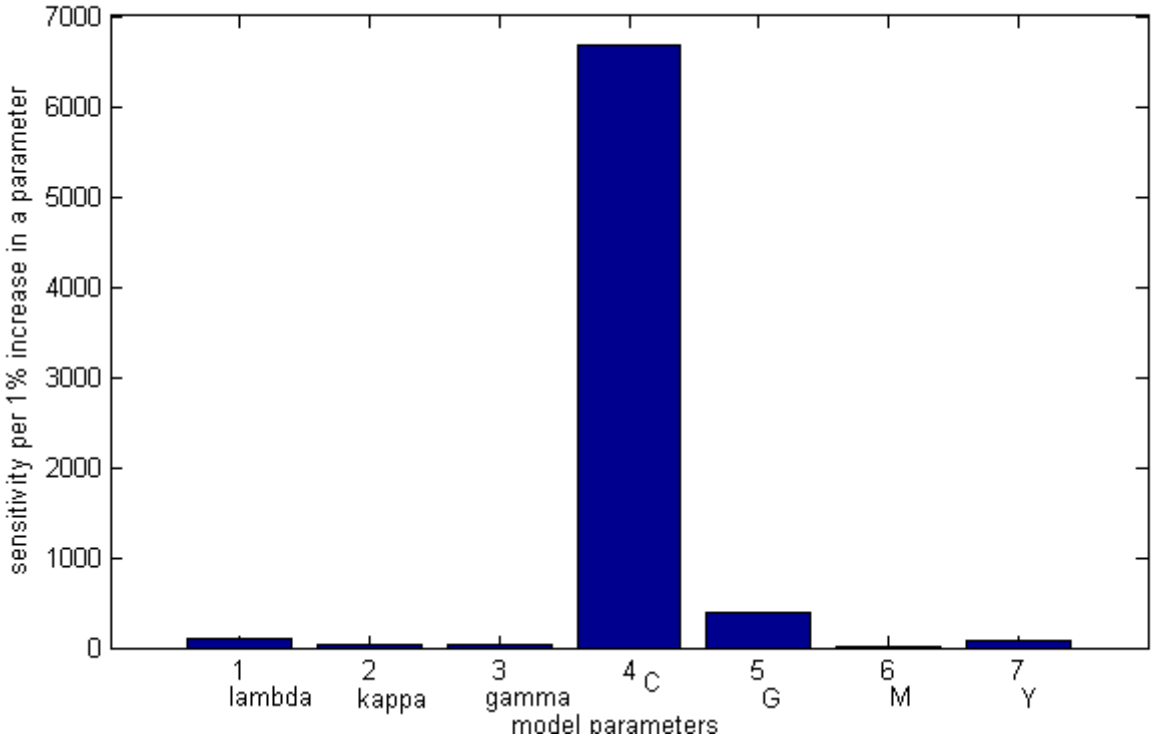

Figure 5.2: Call option price sensitivity for a 1 percent change in each of the model parameters of the CGMY-CIR calibration in subsection 1.2.1.

and strike K=1125. The parameter sensitivities are calculated as the change in price for a 1 percent move in a prameter divided by 1 percent change in the parameter. The parameter sensitivities are depicted in Figure 7.4.

The call option is mostly sensitive to changes in the parameter C. We found that all the options in this work are mostly sensitive to the parameter C under the CGMY-CIR model. This parameter sensitivity analysis is useful when we are pricing and hedging European vanilla options that are not traded in the market. It tells us that if we recalibrate the CGMY-CIR model and the parameter C changes slightly then our model prices and hedge ratios will change by a lot. This could result in mispricing and unstable hedge ratios. This phenomenon is termed model risk. To manage this model risk we could fix the parameter C so that it doesn't change when we recalibrate. The remaining parameters should still offer enough flexibility to enable us to fit market option prices closely.

6 Simulation techniques

Here we look at techniques for sampling from the Lévy and SV processes considered in this work. We consider sampling methods for the uniform, Poisson, normal, gamma, inverse Gaussian, VG, NIG, Meixner, CGMY, CIR, GH and other distributions. Some of these Lévy processes can be simulated as stochastic time-changed processes. For example instead of sampling from the Brownian motion W_t indexed by deterministic time t we may sample from the time-changed Brownian motion W_{X_t} where the time X_t is a stochastic process called a subordinator. The details of simulating a time-changed stochastic process are described in section 7.1.

6.1 Uniform random numbers

The issue of random numbers can be very delicate. Our objective is to get a fast random number generator that produces numbers that are as random as possible. However, good random number generators can be slow and poor random number generators are normally fast. In this section we look at a few pseudo random number generators and point out some important considerations when using them.

6.1.1 Pseudo random number generators

Pseudo random number generators produce sequences of random numbers with a finite length. So, eventually the numbers in the sequence will get repeated. This becomes a problem when we are dealing with a large-scale simulation problem that consumes a large amount of random numbers. The period of a generator is the smallest integer $\rho > 0$ such that the sequence of random numbers repeats after every ρ numbers.
Some generators can produce numbers that are correlated. The correlations may not be visible but if we group the numbers in certain ways then patterns may appear. There are statistical tests that can be performed to detect any known pathologies of random number generators (see [McC06]). The generator of choice must have passed all the tests before it can be used for scientific applications.
The generators need a seed block (initial numbers) to get started. This initialisation determines the sequence of random numbers that will be produced. Therefore, it is important to seed the generator carefully. For example, there are some generators that will produce correlated random numbers if seeded poorly. Also, if the period

of the generator is short, repeated seeding might result in overlapping streams of random numbers.

6.1.2 Commonly-used pseudo random number generators

Merssene Twister: The Mersenne Twister produces high quality random numbers and it is very fast. The implementation `mt19937()` has an astronomical period of $\rho = 2^{19937} - 1 \approx 10^{6001}$. This implementation is part of languages such as R, Python and Matlab. This is the generator of choice for the Monte Carlo simulations done in this writing.
WELL generators: These generators provide better equidistribution and bit mixing with the same period and speed as the Mersenne Twister.
ran2: [PFTV92] describe a number of pseudo RNGs. The generators `ran0()` and `ran1()` must be avoided for critical scientific applications. There are some serial correlations present in random numbers of `ran0()` and it has a period of $\rho = 2^{31} - 2 \approx 2.1 \times 10^9$. The generator `ran1()` passes the statistical tests that the routine `ran0()` fails. The routine `ran1()` still suffers from having a short period that can be exhausted by modern computers. `ran2()` passes all known statistical tests and has a period of $\approx 2.3 \times 10^{18}$. This generator is slow.
We mention in passing that the RNG generator in Excel should be avoided for any serious scientific applications. Refer to [McC08] for detailed reasons against using this RNG.

6.1.3 Random numbers and parallel computing

When simulating on parallel computing clusters, a vast amount of the random numbers can be consumed quickly. So some generators can lose their efficiency or their quality when paralleled. [Mer09] gives a thorough treatment of these problems and their solutions.

Care must be taken when choosing seeds on multi-core nodes for applications that are parallel. For example, when we have independent simulations on each processor that do not communicate. A traditional choice for the seed is to use the CPU time. When multiple instances of a RNG are initiated on a single node with multiple processor cores all the instances will have the same seeds because all the jobs on the different cores will use the same CPU time. One can combine the CPU time with a unique number for each processing core to avoid seed collision among job submissions.

6.1.4 Final recommendations

1. Run your simulations with two pseudo random number generators from different families for small testing instances.

2. Use tried and tested generators instead of relying on your own.
3. Know the limits of your generator. Know the period, the known problems for certain applications and any correlations at any time during the sequence.
4. Be careful with parallel simulations.

6.2 Simulation of Lévy processes

We often do not have closed form expressions under the Lévy process framework to perform calculations like VaR or exotic option prices. In this section we present simulation techniques for Lévy processes that will allow us to perform such calculations. Some of the processes may have several simulation algorithms. One should try to avoid using the slow methods. Also, avoid coding the mathematics of the method naively as it may result in slow simulations. We mostly consider inverse transform methods so that we can perform stratified sampling for option pricing.

6.2.1 Simulation of a Poisson process

We use the inverse transform method to sample from the Poisson distribution with mean $\lambda > 0$. This involves finding the smallest integer n such that the distribution $F(n) = \Pr(N \leq n) \leq U$ where U is Unif[0,1] and $F(n) = \Pr(N = 0) + \Pr(N = 1) + \ldots + \Pr(N = n)$. We shall use the relation $\Pr(N = k+1) = \lambda \Pr(N = k)/(k+1)$. The pseudo code is

Table 6.1: Pseudo code for simulating a Poisson process by inverse transform.

```
Set p = exp(−λ).
Set F = p.
Generate U ∼Unif[0, 1].
while U > F
    N+ = 1
    p = pλ/N
    F+ = p
return N
```

We want to simulate a Poisson process with mean $\lambda > 0$ up to a time point $T > 0$. First generate $N \sim$Poisson(λ). Next, simulate N independent random uniform numbers $\{u_i\}_{i=1,\ldots,N}$. Let $u_{(1)} < u_{(2)} < \cdots < u_{(N)}$ be the order-statistics of this sequence. Then the jump points of the Poisson process are given by the points $Tu_{(1)}, \ldots, Tu_{(N)}$. The Poisson process is zero for times $t < Tu_{(1)}$. At $t = Tu_{(1)}$ the process jumps to 1 and stays there until $t = Tu_{(2)}$, where it then jumps to 2, and so forth.

6.2.2 Simulation of standard Brownian motion

There are a number of ways of generating standard normal random variables. We use the inverse transform algorithm of [Ack04] downloadable in various programming languages from `http://home.online.no/pjacklam/notes/invnorm/index.html`.
Let $\mathbf{W} = \{W_t : t \geq 0\}$ be a standard Brownian motion. We discretise time by taking small time steps of size Δt. We simulate the path of the Brownian motion at time points $\{i\Delta t : i = 0, 1, \ldots\}$. We have

$$W_0 = 0, \quad W_{i\Delta t} = W_{(i-1)\Delta t} + \sqrt{\Delta t}\nu_i, \quad i \geq 1 \tag{6.1}$$

where $\{\nu_i\}_{i=1,2,\ldots}$ are independent standard normal random numbers.
Next we consider the Brownian bridge simulation scheme. Let $x = W_{t_j} - W_{t_i}$, $y = W_{t_k} - W_{t_j}$ be random increments over the intervals $[t_i, t_j]$ and $[t_j, t_k]$ respectively. Define $z = x + y$, the increment over $[t_i, t_k]$. Then $x \sim N(0, t_j - t_i)$, $y \sim N(0, t_k - t_j)$ and $z \sim N(0, t_k - t_i)$. Given a sample value z of Z we are interested in sampling X from the correct conditional distribution. So, given $z = W_{t_k} - W_{t_i}$, $x = W_{t_j} - W_{t_i} \sim N(z\frac{t_j - t_i}{t_k - t_i}, \frac{(t_j - t_i)(t_k - t_j)}{t_k - t_i})$, so that

$$W_{t_j} = \frac{t_k - t_j}{t_k - t_i} W_{t_i} + \frac{t_j - t_i}{t_k - t_i} W_{t_k} + \sqrt{\frac{(t_j - t_i)(t_k - t_j)}{t_k - t_i}} \nu_{t_j}, \tag{6.2}$$

where $\nu_{t_j} \sim N(0, 1)$.

6.2.3 Simulation of gamma process

We need an algorithm for generating $\Gamma(a, b)$ random variates and there are several choices available. We use the inverse transform method of [DMJ87] which is downloadable from `www.netlib.org/toms/654`. We sample from $\Gamma(a, 1)$ and use the property that if $X \sim \Gamma(a, 1)$ then bX has the gamma distribution $\Gamma(a, b)$. When applying this algorithm one must be mindful of how the gamma distribution is parametrised in their programming language.
Let $\mathbf{G} = \{G_t : t \geq 0\}$ be a Gamma process. The increments of the process $\mathbf{G}$ have a gamma distribution, $G_t - G_s \sim \Gamma(a(t - s), b)$ where $0 \leq s \leq t$. We discretise time by taking time steps of size Δt. We simulate the path of the gamma process at time points $\{i\Delta t : i = 0, 1, \ldots\}$. We have

$$G_0 = 0, \quad G_{i\Delta t} = G_{(i-1)\Delta t} + g_i, \quad i \geq 1 \tag{6.3}$$

where $\{g_i\}_{i=1,2,\ldots}$ is a sequence of independent $\Gamma(a\Delta t, b)$ random variates.
As in the Brownian motion case we can construct a gamma bridge simulation scheme. As in the previous section, let $x = G_{t_j} - G_{t_i}$, $y = G_{t_k} - G_{t_j}$ be random increments over the intervals $[t_i, t_j]$ and $[t_j, t_k]$ respectively. Define $z = x + y$, which is the increment over $[t_i, t_k]$. Then $x \sim \Gamma((t_j - t_i)/\nu, \nu)$, $y \sim \Gamma((t_k - t_j)/\nu, \nu)$ and $z \sim \Gamma((t_k - t_i)/\nu, \nu)$.

Given a sample z of Z we are interested in sampling X from the correct conditional distribution. [RW02] show that given g_{t_k} and g_{t_i} then $\frac{g_{t_j}-g_{t_i}}{g_{t_k}-g_{t_i}} \sim Beta(\frac{t_j-t_i}{\nu}, \frac{t_k-t_j}{\nu})$ (Beta distribution). So

$$g_{t_j} = g_{t_i} + \beta_{t_j}(g_{t_k} - g_{t_i}) \tag{6.4}$$

for $\beta_{t_j} \sim Beta\left(\frac{t_j-t_i}{\nu}, \frac{t_k-t_j}{\nu}\right)$.

6.2.4 Simulation of Gamma-OU process

We simulate the Gamma-OU process through its driving process z which is given by a compound Poisson process (see subsection 5.1.1). We simulate the Gamma-OU(a, b) process $y = \{y_t : t \geq 0\}$ at the time points $t_i = i\Delta t, i = 0, 1, 2, \ldots$ as follows

Simulate the Poisson process $\mathbf{N} = \{N_t : t \geq 0\}$ with intensity parameter $a\lambda$ at time points $t_i = i\Delta t, i = 0, 1, 2, \ldots$ using the methods described in subsection 6.2.1.
Simulate the independent exponential random numbers $x_i \sim \text{Exp}(b)$, $i = 1, 2, \ldots$:
$x_i = -\log(v_i)/b$, where $u_i \sim \text{Unif}[0, 1]$, $i = 1, 2, \ldots$ is a sequence of i.i.d. uniform variates.
Simulate the independent uniform random numbers $u_i \sim \text{Unif}[0, 1]$, $i = 1, 2, \ldots$.
Simulate the path of the Gamma-OU(a,b) process at the time points $t_i = i\Delta t, i = 1, 2, \ldots$ as

$$y_{i\Delta t} = e^{-\Delta t\lambda} y_{(i-1)\Delta t} + \sum_{j=N_{(i-1)\Delta t}+1}^{N_{i\Delta t}} x_j, \quad y_0 > 0.$$

6.2.5 Simulation of CIR process

We could simulate the CIR process by first forming an Euler discretisation of the SDE of the CIR process Equation 5.7 at time points $t_i = i\Delta t, i = 0, 1, 2, \ldots$ as

$$y_{i\Delta t} = y_{(i-1)\Delta t} + \kappa(\eta - y_{(i-1)\Delta t})\Delta t + \sigma\sqrt{y_{(i-1)\Delta t}}z_i, \; y_0 > 0 \tag{6.5}$$

with $z_1, z_2, \ldots$ independent standard normal random variables. The Euler discretisation may produce negative values of $y_{i\Delta t}$ and also introduce a discretisation error. We avoid both these issues by sampling from the exact transition law of the process using the method discussed in [Gla03]. [CIJR85] show that the distribution of the CIR process y_t, given y_u for some $u < t$, up to a scale factor is a non-central chi-square distribution. Let $\chi^2_d(\lambda)$ denote the non-central chi-square random variable with d degrees of freedom (d.o.f.) and non-centrality parameter λ. The usual chi-square random variable with d d.o.f. is denoted as $\chi^2_d = \chi^2_d(0)$. The transition law of y_t can be expressed as

$$y_t \overset{d}{=} \frac{\sigma^2(1 - e^{-\kappa(t-u)})}{4\eta}\chi^2_d\left(\frac{4\kappa e^{-\kappa(t-u)}}{\sigma^2(1 - e^{-\kappa(t-u)})}y_u\right), \quad u < t \tag{6.6}$$

where

$$d = \frac{4\eta\kappa}{\sigma^2} \text{ and } \lambda = \frac{4\kappa e^{-\kappa(t-u)}}{\sigma(1-e^{-\kappa(t-u)})} y_u. \tag{6.7}$$

Thus, to sample from y_t exactly we need a way to sample from the non-central chi-square distribution. [JKB94] prove that

$$\chi^2_d(\lambda) = \chi^2_1(\lambda) + \chi^2_{d-1} \stackrel{d}{=} (Z + \sqrt{\lambda})^2 + \chi^2_{d-1} \tag{6.8}$$

for any $d > 1$. So, to sample $\chi^2_d(\lambda)$ for $d > 1$ we must generate a standard Normal random variate Z and an independent χ^2_d.
[Gla03] shows that a non-central chi-square distribution with $d > 0$ d.o.f. can be represented as the usual chi-square random variable with a random d.o.f. parameter. In particular, if N is a Poisson random variate with intensity parameter $\frac{1}{2}\lambda$, then

$$\chi^2_{d+2N} \stackrel{d}{=} \chi^2_d(\lambda). \tag{6.9}$$

Thus to sample $\chi^2_d(\lambda)$ for $d > 0$ we first generate a Poisson random variate N and then, conditional on N, we sample a chi-square random variable χ^2_{d+2N}. Note that the chi-square random variable with d d.o.f. χ^2_d is equal in distribution to the gamma distribution $\Gamma(d/2, 1/2)$. Thus the chi-square random numbers can be sampled from the gamma distribution using the methods given in subsection 6.2.3.
To simulate y_t we use the method in Equation 6.8 for $d > 1$ and the Poisson representation in Equation 6.9 when $d \leq 1$. We simulate the CIR(κ, η, σ) process $y = \{y_t : t \geq 0\}$ at the time points $t_i = i\Delta t, i = 0, 1, 2, \ldots$ as follows

Set $d = 4\eta\kappa/\sigma^2$
Case $d > 1$
  Set $c = \sigma^2(1 - e^{-\kappa(t_{i+1}-t_i)})/(4\kappa)$.
  Set $\lambda = y_{t_i} e^{-\kappa(t_{i+1}-t_i)}/c$.
  Generate the standard normal random variate Z.
  Generate $X \sim \chi^2_{d-1}$ independent of Z.
  Return $y_{t_{i+1}} = c[(Z + \sqrt{\lambda})^2 + X]$.
Case $d \leq 1$
  Set $c = \sigma^2(1 - e^{-\kappa(t_{i+1}-t_i)})/(4\kappa)$.
  Set $\lambda = y_{t_i} e^{-\kappa(t_{i+1}-t_i)}/c$.
  Generate $N \sim \text{Poisson}(\lambda/2)$.
  Generate $X \sim \chi^2_{d+2N}$.
  Return $y_{t_{i+1}} = cX$.

6.2.6 Simulation of a VG process

The VG process $\mathbf{X} = \{X_t : t \geq 0\}$ with parameters (C, G, M) can be decomposed as the difference of two independent Gamma processes

$$X_t \stackrel{d}{=} G^{(1)}_t - G^{(2)}_t \tag{6.10}$$

where $G^{(1)} = \left\{G_t^{(1)} : t \geq 0\right\}$ is a gamma process with parameters $a = C$ and $b = M$ and $G^{(2)} = \left\{G_t^{(2)} : t \geq 0\right\}$ is another gamma process with parameters $a = C$ and $b = G$ independent of $G^{(1)}$. Thus we can simulate the VG process using the techniques described above for a gamma process. The gamma distribution $\Gamma(a, b)$ used here has PDF given in Equation 3.22.

The VG process can also be represented as a time-changed (subordinated) Brownian motion. This form is best expressed using the parametrisation (σ, ν, θ) instead of (C, G, M). We have

$$X_t \stackrel{d}{=} \theta G_t + \sigma W_{G_t} \tag{6.11}$$

where W_t is a standard Brownian motion and G_t is a gamma process $G_t \sim \Gamma(t/\nu, \nu)$. To sample from the VG process we have to generate a gamma variate and a normal random variable using the techniques discussed above. We can implement a bridged VG simulation by combining the Brownian bridge of subsection 6.2.2 with the gamma bridge techniques of subsection 6.2.3 through subordination. We describe the subordination implementation details in a later chapter.

6.2.7 Simulation of an IG process

We use the algorithm of [MSH76] to generate the IG variates. This generator uses two uniform random numbers to generate a single IG(a, b) number.

Generate $q \sim \chi_1^2$.
Set $x = (a/b) + q/(2b^2) - \sqrt{4abq + q^2}/(2b^2)$.
Generate $u \sim$Unif[0,1].
If $u \leq a/(a + xb)$,then return x as the IG(a, b) random number, else return $a^2/(b^2 x)$.

Let $\mathbf{I} = \{I_t : t \geq 0\}$ be an IG process. The increments of the process $\mathbf{I}$ have an IG distribution, $I_t - I_s \sim IG(a(t - s), b)$ where $0 \leq s \leq t$. We discretise time by taking time steps of size Δt. We simulate the path of the IG process at time points $\{n\Delta t : n = 0, 1, \ldots\}$. We have

$$I_0 = 0, \quad I_{n\Delta t} = I_{(n-1)\Delta t} + i_n, \quad n \geq 1 \tag{6.12}$$

where $\{i_n\}_{n=1,2,\ldots}$ is a sequence of independent $IG(a\Delta t, b)$ random variates.
[RW03] have developed a bridge method for the IG process. Let X, Y and $Z := X + Y$ be increments of an IG process over the time intervals $[t_i, t_j]$, $[t_j, t_k]$ and $[t_i, t_k]$ respectively for $0 \leq t_i \leq t_j \leq t_k$. Given i_{t_k} and i_{t_i} we want to sample i_{t_j} with the correct conditional distribution. We sample from the conditional distribution of $X|Y$ as follows:

Generate $q \sim \chi^2_1$.
Set $\lambda = \frac{(a(t_j - t_i))^2}{z}$ and $\mu = \frac{t_k - t_j}{t_j - t_i}$ and compute roots s_1 and s_2,
$s_1 = \mu + \frac{\mu^2 q}{2\lambda} - \frac{\mu}{2\lambda}\sqrt{4\mu\lambda q + (\mu q)^2}, s_2 = \frac{\mu^2}{s_1}$.
Set $s = s_1$ with probability $p = \frac{\mu(1+s_1)}{(1+\mu)(\mu+s_1)}$, else set $s = s_2$. In other words let u be a draw from the uniform distribution Unif[0,1]. If $p \geq u$ choose s_1, else choose s_2.
Set $i_{t_j} = i_{t_i} + \frac{i_{t_k} - i_{t_i}}{1+s}$.

At each bridge time two uniform variates are required. One for sampling from the χ^2_1 distribution and the second for deciding which root to use, s_1 or s_2.

6.2.8 Simulation of IG-OU processes

We simulate the IG-OU process using an exact algorithm. The IG-OU process can be written in the form

$$y_t = y_0 e^{-\lambda t} + e^{-\lambda t} \int_0^{\lambda t} e^s dz_s. \tag{6.13}$$

Thus we need a way to sample from the process

$$y'_t = \int_0^t e^{-\lambda(t-s)} dz_{\lambda s}. \tag{6.14}$$

We use a result of [ZZ08] which states that for fixed $t > 0$ the random variate y_t is equal in distribution to the sum of an IG random variable

$$z_t^{(1)} \sim \text{IG}(a[1 - \exp(-t\lambda/2)], b)$$

and a compound Poisson random variable

$$z_t^{(2)} = \sum_{i=1}^{N_t} w_i^t, \tag{6.15}$$

where $\mathbf{N} = \{N_t : t \geq 0\}$ is a Poisson process with intensity parameter $ab(1 - \exp(-t\lambda/2))$ and $\{w_i^t : i = 1, 2, \ldots\}$ is an i.i.d. sequence of random variables with common pdf

$$f^t(w) = \begin{cases} \frac{e^{-b^2 w/2} - e^{-b^2 w e^{t\lambda}/2}}{b w^{3/2}\sqrt{2\pi}(e^{t\lambda/2} - 1)} & \text{for } w > 0 \\ 0 & \text{otherwise} \end{cases} \tag{6.16}$$

The challenge is to draw random variates from the distribution of the above PDF. This is done by the acceptance-rejection method introduced by [VN51]. Note that acceptance-rejection methods do not allow use to exploit the benefits of stratified sampling. We recommend using the technique described in subsection 6.2.11 instead.

The PDF $f^t(w)$ is bounded (up to a scaling factor) by the PDF of the gamma distribution $\Gamma(1/2, b^2/2)$. That is

$$f^t(w) \leq \frac{1}{2}\left(1 + e^{\frac{1}{2}t\lambda}\right) f^{\Gamma}(w) = \frac{1}{2}\left(1 + e^{\frac{1}{2}t\lambda}\right) \frac{(b^2/2)^(1/2)}{\Gamma(1/2)w^{-1/2}e^{-b^2w/2}}$$

So we can employ the acceptance-rejection method on the $\Gamma(1/2, b^2/2)$ distribution.

---

do
    Generate $g \sim \Gamma(1/2, b^2/2)$.
    Draw $u \sim$Unif$[0, 1]$.
until $u \leq \frac{f^t(g)}{\frac{1}{2}\left(1+e^{\frac{1}{2}t\lambda}\right)f^{\Gamma}(g)}$
Set $w = g$ as a draw from $f^t(w)$.

Since the process y'_t has stationary increments we have

$$y_{t+s} \stackrel{d}{=} e^{-s\lambda}y_t + z_s^{(1)} + \sum_{i=1}^{N_s} w_i. \tag{6.17}$$

Below is a routine to simulate the IG-OU process using the representation in Equation 6.17.

Generate $\nu \sim$IG$(a(1 - \exp(-\Delta t\lambda/2)), b)$.
Generate $p \sim$Poisson$(ab(1 - \exp(-\Delta t\lambda/2)))$.
Generate the i.i.d. sequence $w_1, w_2, \ldots, w_p$ from the pdf $f^{\Delta t}(w)$.
Set $y_{t+\Delta t} = e^{-\Delta t\lambda}y_t + \nu + \sum_{i=1}^{p} w_i$.

6.2.9 Simulation of NIG process

The NIG(α, β, δ) process can also be represented as a time-changed Brownian motion. We have

$$X_t \stackrel{d}{=} \beta\delta^2 I_t + \delta W_{I_t} \tag{6.18}$$

where W_t is a standard Brownian motion and I_t is an IG process $I_t \sim IG(t, \delta\sqrt{\alpha^2 - \beta^2})$. To simulate the NIG process we sample from the inverse Gaussian and standard normal distribution. We can implement a bridged NIG simulation by combining the Brownian bridge of subsection 6.2.2 with the IG bridge techniques of subsection 6.2.7 through subordination.

6.2.10 Simulation of integrated activity rate

We need to simulate the integrated process $Y_t = \int_0^t y_s ds$ at time points $0 = t_0 < \cdots < t_M = T$ where y_s is either the CIR, IG-OU or Gamma-OU process. We could use the estimation

$$Y_{t_j} \approx \sum_{i=0}^{j} y_{t_i}\Delta t_i, j = 0, \ldots, M \tag{6.19}$$

for small enough $\Delta t_i = T/M$. To get an exact simulation of the stochastic clock we use characteristic function of Y_t and the inverse transform method described in subsection 6.2.11. Note that the increments of Y_t are not i.i.d..

6.2.11 Inverse transform coupled with Fourier transform

Some Lévy processes do not have the convenient simulation algorithms as those outlined thus far in this chapter, for example the CGMY, Meixner and GH processes. Alternatively we can sample through the inverse of the cumulative density function (CDF) of the random variable. For instance, suppose that the random variable X has CDF F_X. Then $F_X^{-1}(u)$, where $u \sim$Unif(0,1), is a draw from the distribution of X. However, the CDF may not be known. In this section we use the characteristic function and the Fourier transform to get the CDF as described in [Hug98]. This technique exploits the closed form of the characteristic function to retrieve the CDF of the process increments via Fourier inversion. This gives us a fast and exact simulation scheme. This Fourier method was used by [BK11] for MC option pricing using the CGMY process. We further apply it to simulate SV process. The ability to sample from the exact distribution gives us unbiased price estimates. [AR01] used the compound Poisson approximation coupled with a Brownian motion to approximate a general Lévy process. [MY08] represent the CGMY and Meixner processes as subordinated Brownian. Both of these methods rely on truncation procedures in order to sample from the processes. The truncation procedure refers to ignoring the small jump sizes that are below a pre-specified threshold ϵ and replacing these by their expected value or a Brownian motion. This truncation procedure introduces a bias in the price estimates which are difficult to quantify. Next we outline the set-up for this exact simulation.
Let X be a random variable with CDF F. Define

$$F_r(x) = F(x) - \frac{1}{2}F(x-D) - \frac{1}{2}F(x+D)$$

as in [Hug98] where $D > 0$ is a constant. Note that given x and large D $F(x)$ approaches $F_r(x) + \frac{1}{2}$. The function F_r is introduced since it satisfies the absolute integrability condition which is sufficient for its Fourier transform to exist. F_r has Fourier transform

$$\begin{aligned} \phi_r(u) &= \int_{\mathbb{R}} e^{iux} F_r(x) dx \\ &= \begin{cases} \frac{\cos(uD)-1}{iu}\phi(u) & u \neq 0 \\ 0 & u = 0, \end{cases} \end{aligned} \tag{6.20}$$

where ϕ is the characteristic function of X. Both $F_r(x)$ and $\phi(u)$ satisfy the absolutely integrable condition, thus F_r can be expressed as the inverse Fourier integral

$$F_r(x) = \frac{1}{2\pi} \int_{\mathbb{R}} e^{-iux} \phi_r(u) du. \tag{6.21}$$

So, for $x \in [-\frac{D}{2}, \frac{D}{2}]$ we approximate the CDF of X as $F(x) \approx F_r(x) + \frac{1}{2}$. [Hug98] derives the following error bound for this approximation

$$\left| F(x) - F_r(x) - \frac{1}{2} \right| \leq \frac{2^\alpha A}{D^\alpha}, \tag{6.22}$$

for $x \in [-\frac{D}{2}, \frac{D}{2}]$ provided there exist constants $\alpha > 1$ and $A > 0$ that satisfy $F(-x) \leq A|x|^{-\alpha}$ and $1 - F(x) \leq A|x|^{-\alpha}$ for $x > 0$. The author comments that these bounds hold for $A = \sigma$ and $\alpha = 2$ for any distribution within zero mean and finite variance. All of our models have finite variance. We can also adjust the processes to have zero mean so that the bound is applicable and add the mean back afterwards. We evaluate the Fourier integral in Equation 6.22 by truncating the integration range from the whole real line to a finite interval $[-\frac{L}{2}, \frac{L}{2}]$ for $L > 0$ so that

$$F_r(x) \approx \frac{1}{2\pi} \int_{-L/2}^{L/2} e^{-iux} \phi_r(u) du. \tag{6.23}$$

[Hug98] also shows that the error introduced by the above truncation is bounded

$$\left| F_r(x) - \frac{1}{2\pi} \int_{-L/2}^{L/2} e^{-iux} \phi_r(u) du \right| \leq \frac{2^{\beta+1} B}{\pi L^\beta \beta} \tag{6.24}$$

provided there exist constants $B > 0$ and $\beta > 0$ such that $|\phi(u)| \leq B|u|^{i\beta}$ for $u \in \mathbb{R}$. [Hug98] derives the parameters A and B as

$$A = \frac{1}{i^\alpha} \frac{d^\alpha \phi(0)}{du^\alpha}, \quad B = \max_u \left| u^\beta \phi(u) \right| \tag{6.25}$$

for $\alpha = 2n, n \in \mathbb{N}$, and $\beta \geq 1$. Define the error bounds in Equation 6.22 and Equation 6.24 respectively as

$$\epsilon_r = \frac{2^\alpha A}{D^\alpha}, \quad \epsilon_c = \frac{2^{\beta+1} B}{\pi L^\beta \beta}. \tag{6.26}$$

By varying the parameters α and β we can obtain a set of parameters A, α, B and β that achieves the required accuracy for the given interval sizes L and D.
Further, we discretise the truncated integral and approximate it as a sum. So we have

$$\begin{aligned} F_r(x_l) &\approx \frac{1}{2\pi} \sum_{j=0}^{N-1} e^{-iu_j x_l} \phi_r(u_j) \delta u, \\ &= \frac{1}{2\pi} e^{-iu_0(x_l - x_0)} \sum_{j=0}^{N-1} e^{-ijl\delta u \delta x} e^{-iu_j x_0} \phi(u_j) \delta u \end{aligned} \tag{6.27}$$

where $\mathbf{x} = \left\{(l - \frac{N}{2})\delta x\right\}_{l=0}^{N-1}$ and $\mathbf{u} = \left\{(j - \frac{N}{2})\right\}_{j=0}^{N-1}$ are N-point uniform grids and $\delta x = \frac{D}{N}$ and $\delta u = \frac{L}{N}$. If $\delta u \delta x = 2\pi/N$ in Equation 6.27 we get the usual discrete Fourier transform that can be evaluated fast and accurately using the FFT. This restriction requires that $LD = 2\pi N$. We would like to lift this restriction on the interval lengths so that $LD = 2\pi\alpha$ where α is any positive real number, which allows us to choose the interval lengths L and D independently. We can evaluate this unconstrained sum using the fractional Fourier transform (FrFT) developed by [BS91]. Next we describe how to implement the FrFFT.

6.2.11.1 The fractional Fourier transform

This method is useful when we want to approximate the Fourier tranfom by summation

$$\int_0^\infty e^{-ixu} f(u)du \approx \sum_{j=0}^{N-1} e^{-i2\pi jl\alpha} w_j \delta u. \tag{6.28}$$

The vector $\mathbf{w} = (w_j)_{j=0}^{N-1}$ corresponds to N evaluations of the function f at the input points $\mathbf{u} = (u_j)_{j=1}^{N-1}$. The output of the FrFT is an N-vector $\mathbf{W} = (W_l)_{l=0}^{N-1}$ where each W_l corresponds to an integral of the form Equation 6.28 evaluated at a predetermined point $x = x_l$. So we have that

$$\mathbf{W} = \mathrm{FrFT}(\mathbf{w}, \alpha) \tag{6.29}$$

where $\delta x = x_{l+1} - x_l$ and $\delta u = u_{j+1} - u_j$ are fixed spacings and $\alpha = \delta x \delta u/(2\pi)$. The implementation of the FrFFT is described by [BS91] and uses three $2N$-point FFT procedures. Define the $2N$-vectors

$$\begin{aligned} y_j &= w_j e^{-i\pi j^2 \alpha} \text{ for } 0 \le j \le N-1 \\ y_j &= 0 \text{ for } N \le j \le 2N-1 \\ z_j &= e^{ij^2\pi\alpha} \text{ for } 0 \le j \le N-1 \\ z_j &= e^{i\pi\alpha(2N-j)^2} \text{ for } N \le j \le 2N-1 \end{aligned} \tag{6.30}$$

The FrFT is given by

$$\mathbf{W} = \mathrm{FrFT}(\mathbf{w}, \alpha) = \left(e^{-i\pi\alpha j^2}\right)_{j=0}^{N-1} \odot \mathrm{FFT}^{-1}\left(\mathrm{FFT}(\mathbf{y}) \odot \mathrm{FFT}(\mathbf{z})\right) \tag{6.31}$$

where the symbol $\odot$ represents element-wise vector multiplication. Note that $\mathrm{FFT}^{-1}\left(\mathrm{FF}(\mathbf{y}) \odot \mathrm{FFT}(\right.$ returns a $2N$-vector but $\left(e^{-i\pi\alpha j^2}\right)_{j=0}^{N-1}$ is an N-vector. The FrFFT should return the element-wise multiplication of these two vectors by disregarding the last N entries of $\mathrm{FFT}^{-1}\left(\mathrm{FFT}(\mathbf{y}) \odot \mathrm{FFT}(\mathbf{z})\right)$. The code for implementing the FrFFT is given in

section 11.7 using the function name FrFFT.
Thus, from Equation 6.27 we have that

$$F_r(\mathbf{x}) \approx \Re(\frac{1}{2\pi} \odot e^{-iu_0(\mathbf{x}-\mathbf{x}_0)} \odot \text{FrFFT}(\phi_r(\mathbf{u}) \odot e^{-i\mathbf{u}x_0}, \frac{\delta u \delta x}{2\pi})) \tag{6.32}$$

We adopt the convention that a vector added or multiplied with a scalar means that the scalar operation is applied element-wise to the vector. We then calculate the CDF as

$$F(\mathbf{x}) \approx F_r(\mathbf{x}) + \frac{1}{2}. \tag{6.33}$$

So we know the value of $y_i := F(x_i)$ at points $x_0 < \ldots < x_{N-1}$ defined as $\mathbf{x} = \left\{(i - \frac{N}{2})\delta x\right\}_{i=0}^{N-1}$. Note that we also know $x_i = F^{-1}(y_i)$ at points $y_0 < \ldots < y_{N-1}$. We want to estimate $x = F^{-1}(y)$ for arbitrary $y \sim$Unif(0,1). We achieve this by interpolation. So, given arbitrary y we estimate x by interpolating the table $(y_i; x_i)$.

6.2.11.2 Pseudo code

Here we outline the pseudo code for drawing from a distribution given its characteristic function ϕ. Let X be a random variable with a Levy distribution with the nth central moment c_n.

1. The demeaned random variable $Y = X - c_1$ has characteristic function $\phi_Y(u) = \exp(-iuc_1)\phi_X(u)$.
2. Define the truncated domain of Y as $[-D/2, D/2] = \left[-L\sqrt{c_2 + \sqrt{c_4}}, L\sqrt{c_2 + \sqrt{c_4}}\right]$ as in subsection 4.3.1 for the COS method. We used $L = 200$ in our results.
3. Set a value for L in Equation 6.23. Values around $L = 10000$ sufficed in our work. The larger L is the more integration points are required.
4. Set the number of integration points $N = 2^p$ for some $p \in \mathbb{N}$. Small values of N may lead to poor approximations to the integral in Equation 6.23, especially when L is large. Large values of N (e.g. $p > 21$) can lead to RAM shortage issues depending on your machine. Care must be taken when implementing the FrFFT for large N.
5. Define $\mathbf{x} = \left\{(l - \frac{N}{2})\delta x\right\}_{l=0}^{N-1}$ and $\mathbf{u} = \left\{(j - \frac{N}{2})\delta u\right\}_{j=0}^{N-1}$ where $\delta x = \frac{D}{N}$ and $\delta u = \frac{L}{N}$.
6. Set $\alpha = \delta u \delta x$.
7. Let $w_j = \phi_Y(u_j)$.
8. Set the vectors y and z as in Equation 6.30.

9. Evaluate the CDF on N grid points $x \in [-D/2, D/2]$ as $F(x) = F_r(x) + 1/2$. $F_r(x)$ may be evaluated by a combination of Equation 6.32 and Equation 6.31.
10. Generate an $N \times M$-matrix of draws from Unif(0,1) where N may be the number of paths and M the number of time steps.
11. Generate an $N \times M$-matrix of realisations (y_{ij}) of the demeaned random variable Y by interpolation. We used the MATLAB implementation of the piecewise cubic Hermite interpolating polynomial in our simulations.
12. Set the realisations from X as $x_{iy} = y_{ij} + c_1$.

To assess the quality of these draws we may compare the sample moments to the theoretical moments. If the moments are not known in closed form we estimate them using the method introduced in subsection 4.2.1.
Now that we have adequate simulation algorithms we proceed to valuing a path dependent option.

7 A basis for Monte Carlo valuation

In this chapter we shall apply Monte Carlo (MC) techniques to price derivatives. In short, the MC method will use simulation techniques described in a previous chapter to generate sample paths for the underlying asset price process. We evaluate the pay-off function V_i for each of the $i = 1, 2, \ldots, N$ paths. The MC estimate of the derivative price is given by

$$\widehat{V} = e^{-rT}\left(\frac{1}{N}\sum_{i=1}^{N} V_i\right). \tag{7.1}$$

The standard error of the estimate is of the form $\sigma/\sqrt{N}$. We may estimate the parameter σ by the sample standard deviation

$$s = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\left(e^{-rT}V_i - \widehat{V}\right)^2}. \tag{7.2}$$

The standard error decreases with the square root of the number of sample paths. So, adding a single decimal point of precision will require 100 times as many sample paths. This scaling tells us that achieving convergence to within a desired error bound will require a lot of simulations and may thus be very slow.

In this chapter we shall implement techniques that deliver the required speed-ups to the plain MC method. These are stratified sampling and bridge methods. Other techniques such as importance sampling and control variates can be employed. We shall not consider them here.

7.1 Path generation for Lévy processes

As in chapter 4, we model the stock price under the risk-neutral measure as

$$S_t = S_0 \exp\left((r - q - w)t + L_t\right) \tag{7.3}$$

where L_t is a Lévy process, r is the constant short rate, q is the constant dividend yield and w compensates for the drift in L, to make sure that $\left(S_t e^{-(r-q)t}\right)_{t\geq 0}$ is a martingale.

We want to price an option with pay-off $V_T := V_T(\omega)$ at time of maturity T. V_T may depend on the sample path ω. The value c_t at time $t < T$ of the option under the martingale measure associated with the bank account numeraire is

$$c_t = \mathbb{E}_t \left[V_T e^{-r(T-t)} \right]. \tag{7.4}$$

The MC estimate $\widehat{c}_t$ to the option price c_t is given by

$$\widehat{c}_t = e^{-r(T-t)} \frac{1}{N} \sum_{i=1}^{N} V_T. \tag{7.5}$$

We simulate the motion of the stock price path at times $0 = t_0 < t_1 < \ldots < t_M = T$ as follows

$$S_0 > 0, \quad S_{t_{i+1}} = S_{t_i} \exp\Big((r - q - w)(t_{i+1} - t_i) + \Delta L_{t_{i+1}} \Big) \tag{7.6}$$

for $i = 0, \ldots, N-1$ where $\Delta L_{t_{i+1}} = L_{t_{i+1}} - L_{t_i}, L_0 = 0$. For example, in the NIG case $\Delta L_{t_{i+1}} \sim$NIG$(\alpha, \beta, \Delta t_{i+1}\delta)$ and for the GH process we would sample ΔL_{t_i} through the characteristic function $\phi_{X_1}(u)^{\Delta t_i}$.

Stochastic time-changed Lévy process We also need a simulation scheme for time-changed Lévy processes. We want to model the uncertainty in the economy by time-changing the Lévy process L_t by a stochastic clock Y_t. How can we simulate the paths of the time-changes process $L_t = L_{Y_t}$? Again, discretise the period $[0, T]$ as $0 = t_0 < t_1 < \ldots < t_M = T$ and set $\Delta t_i = t_{i+1} - t_i$. First simulate a path $\{Y_{t_i}\}_{i=0,\ldots,M}$ for Y_t.

1. Set $Y_0 = 0$.
2. Iteratively, sample from the distribution of the random increment $\Delta Y_{t_i} = Y_{t_{i+1}} - Y_{t_i}$ over the interval Δt_i using methods described in a previous chapter.

Now that we have $\{Y_{t_i}\}_{i=0,\ldots,M}$, we generate a path for $\mathbf{L}$ at these times.

1. Set $L_0 = 0$.
2. Iteratively, sample from the the conditional distribution (conditional on the given times Y_{t_i}) of the random increment $\Delta L_{Y(t_i)} = L_{Y(t_{i+1})} - L_{Y(t_i)}$ over the interval $\Delta Y(t_i) = Y(t_{i+1}) - Y(t_i)$. For example, if L_t is the Meixner process with $L_t \sim$Meixner$(\alpha, \beta, \delta t)$ then $\Delta L_{Y(t_i)} \sim$Meixner$(\alpha, \beta, \delta \Delta Y_{t_i})$.
3. $L_{Y(t_{i+1})} = L_{Y(t_i)} + \Delta L_{Y(t_i)}$.
4. Set $L_{t_i} = L_{Y(t_i)}$.

7.2 MC with variance reduction

Suppose we want to price an option whose payoff is dependent on the stock price at times $0 = t_0 < t_1 < \ldots < t_M = T$. The plain MC estimate $\widehat{c}_t$ will converge to the true option value c_t as N tend to infinity. However, convergence to within the desired error bound may be slow. We can improve on the plain MC method by using stratified draws from the laws of the Lévy processes that drive stock returns.

7.2.1 Stratified sampling

[RW02] point out that stratified sampling ensures that sample points form a less clustered draw from the sample space. This give us sample paths that are drawn more evenly under the risk-neutral measure.

We can ensure minimal clustering when sampling from the unit hypercube using a low discrepancy sequence such as the Sobol sequence. Low discrepancy sequences are necessary when we have to sample from a unit hypercube with a high dimension. We use the Sobol sequence based on [JK03] which samples uniformly for a hypercube of dimension up to 1111.
To form a stratified sample for the random variable L which has distribution function F_L we have to use the inverse transform method. If F_L^{-1} is computable (analytically or numerically) and $\{u^i\}_{i=1,\ldots,N}$ is a stratified sample then the set $\left\{\left(F^L\right)^{-1}(u^i)\right\}_{i=1,\ldots,N}$ is a stratified sample from F^L. We shall use this inverse transform method to sample evenly from the Lévy distributions used in our methods.

7.2.2 Bridge sampling

Here we present the bridge sampling process as explained by [RW02]. Suppose we have a Lévy process $\mathbf{L} = \{L_t : t \geq 0\}$ and L_t follows the distribution (conditional on L_0) F_t at time t. Suppose further that we have a sample $L_{i,M}, i = 1, \ldots, N$, of L_{t_M} (possibly stratified). Given L_0 (zero in our case), we want to derive the entire path $0 = L_0 = L_{i,0}, L_{i,1}, \ldots, L_{i,M}$ with the correct conditional distributions. So, we need to be able to sample $L_{i,j}$ at time $0 < t_j < t_M$, conditional on the values of $L_{i,0}$ and $L_{i,M}$. If the marginal densities of the process L_t can be found, then the conditional (bridge) density can be constructed and perhaps some sampling method can be applied. [RW02] and [RW03] found the bridge densities for the gamma and inverse Gaussian processes. These can be combined with the Brownian bridge to create bridged versions of the VG and NIG processes as mentioned in a previous chapter. Sampling from the bridged process allows us to improve the sampling over the path as a whole. For a subordinated Lévy process $L_t = L_{Y(t)}$ we use the bridge as follows.

1. Given $Y_{i,0} = 0$, construct a stratified sample $\{Y_{i,M}\}_{i=1,\ldots,N}$.

2. Using the bridge distribution, construct $Y_{i,M/2}$ conditional on $Y_{i,0}$ and $Y_{i,M}$.
3. By binary chops, continue to generate a path $Y_{i,j}$ for other times $t_j, j = 1, \ldots, M-1$.
4. Construct a stratified sample of L at the times given by the path for Y. This gives us a stratified bridge for $L_{Y(t_j)},\ t_j, j = 0, \ldots, M$.

Having a stratified bridge method gives us more accurate MC estimates for path dependent option prices than plain MC prices.

7.3 Simulation results

Here we demonstrate in more detail how to sample from a SV Lévy process. We simulate the processes at $M = 256$ evenly distributed points on the interval [0,1]. We then price a cliquet option under the SV processes.

7.3.1 Simulating a VG-CIR SV process

Denote by L_t^* the VG-CIR returns. This process is a VG process L_t time-changed by the integrated CIR process Y_t. The VG L_t can be represented as a subordinated Brownian motion as described in subsection 6.2.6. This form is best expressed using the parametrisation (σ, ν, θ) instead of (C, G, M). We use the VG-CIR parameters in Table 5.1

1. Simulate the CIR process $y_{t_i}, i = 0, \ldots, M$ as previously described where $y_{t_0} = y_0 = 1$.
2. The stochastic clock at time t_i is $Y_{t_i} = \sum_{j=0}^{i} y_{t_i}(t_i - t_{i-1})$ where $Y_{t_0} = 0$. Alternatively, simulate Y_t directly from it's characteristic function as described in subsection 6.2.10.
3. Convert the VG(C, G, M) parameters to VG(σ, ν, θ):
$$\begin{aligned} \nu &= 1/C \\ \sigma &= \sqrt{2C/GM} \\ \theta &= \begin{cases} C\sqrt{(1/G + 1/M)^2 - 4/GM} & \text{if } G < M \\ -C\sqrt{(1/G + 1/M)^2 - 4/GM} & \text{if } G > M \end{cases} \end{aligned}$$
4. Generate $\Delta G_{Y(t_i)} = G_{Y(t_i)} - G_{Y(t_{i-1})} \sim \Gamma(\Delta Y_{t_i}/\nu, \nu)$ and set $G_{Y(t_i)} = G_{Y(t_{i-1})} + \Delta G_{Y(t_i)}$ where $G_0 = 0$.
5. Generate $L(t_i)$ as $\theta G_{Y(t_i)} + \sqrt{G_{Y(t_i)}}\sigma Z$ where Z is an independent standard normal variate. One may also wish to sample the VG distribution more closely by using a Brownian bridge at the time points $G_{Y(t_i)}$.

Each of the 3 simulated processes is depicted in the following graphs. The code to produce the graphs in Figure 7.1 is in Listing 7.1.

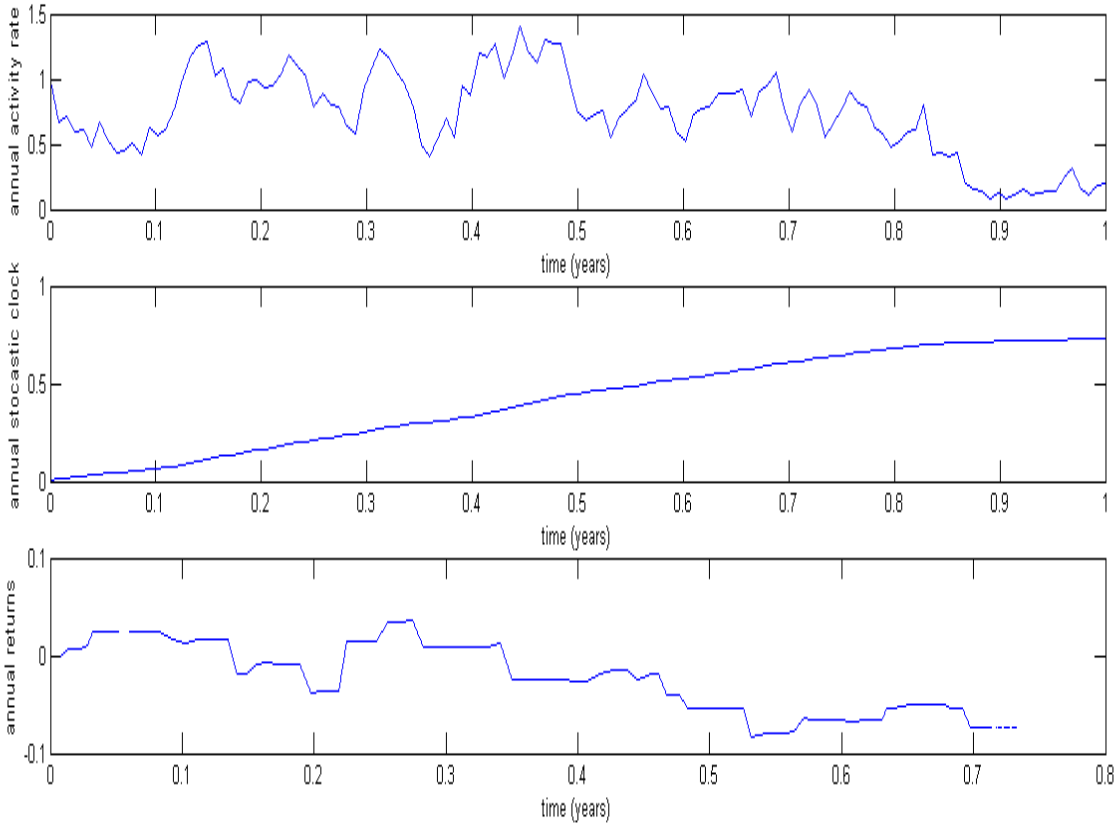

Figure 7.1: Simulated path of a VG-CIR stochastic volatility Levy process. This uses subordination.

Listing 7.1: Script to produce a path of a VG-CIR stochastic volatility Levy process in MATLAB.

```
%Example script to simulated a subordinated Levy process.

%Stochastic processes.
proc1=@procVGBridge;
proc2=@procCIRTrans;

%Model parameters.
x1=[12.5938 26.6875 36.2745];
x2=[0.6074 1.5809 2.0329 1.0000];

T=1;
```

```
N=256;

[subordinator, stochClock, timeChangedVG]=procVGSV(proc1,
    pars1,T,N,proc2,pars2);

%Plot output
subplot(3,1,1); plot(subordinator)
subplot(3,1,2); plot(stochClock)
subplot(3,1,3); plot(timeChangedVG)
```

7.3.2 Simulating a GH-OUIG SV model

Denote by L_t^* the GH-OUIG returns. This process is a GH process L_t time-changed by the integrated IG process Y_t. The GH L_t will be simulated via the FrFFT described in subsection 6.2.11. We use the GH-OUIG parameters in Table 5.1

1. Simulate the OU-IG process $y_{t_i}, i = 0, \ldots, M$.
2. The stochastic clock at time t_i is $Y_{t_i} = \sum_{j=0}^{i} y_{t_i}(t_i - t_{i-1})$ where $Y_{t_0} = 0$. Alternatively, simulate Y_t directly from it's characteristic function as described in subsection 6.2.10.
3. Generate ΔL_t via inverse transformation. The CDF is evaluated numerically from the characteristic function of the time-$Y(t_i)$ GH process $\phi_{GH}(u)^{\Delta Y(t_i)}$. Set $L(t_i) = L(t_{i-1}) + \Delta L(t_i)$.

Alternatively, sample directly from the GH-OUIG distribution.

1. Generate ΔL_t via the inverse transformation. The CDF is evaluated numerically from the characteristic function of the GH-OUIG process $\phi_{GH-OUIG}(u)^{\Delta t_i}$. The SV characteristic function is calculated using Equation 5.13.

The Meixner and CGMY processes, as well as their SV counterparts, may be simulated directly using only the characteristic function. In fact, all the SV processes in this work can be simulated in this way. The code to produce the graphs in Figure 7.2 is in Listing 7.2.

Listing 7.2: Script to produce a path of a GH-OUIG stochastic volatility Levy process in MATLAB.

```
%Example script to simulated a subordinated Levy process.

%Stochastic processes.
chatfun1=@charGH;
chatfun2=@charIntOUIG;

%Model parameters.
x1=[20.2070    -7.9568    0.4801    -1.2925];
```

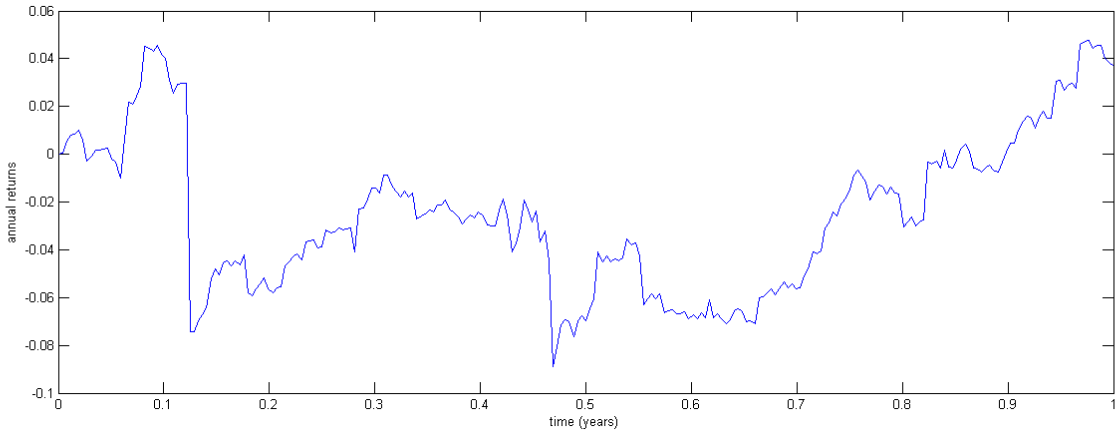

Figure 7.2: Simulated path of a GH-OUIG stochastic volatility Levy process. This was generated using the FrFFT algorithm.

```
x2=[0.7379      0.6642      1.2070      1.0000];

timeChangedGH=FrFFTDrawsDeep(@(u,mydata)TimeChangedchar(u,
    chatfun1,x1,chatfun2,x2,mydata),1,256,3000,2^21,1);

%Plot output
plot(timeChangedGH)
```

7.4 Exotic options

Our SV models were able to fit a realistic option price surface. Now we would like to apply these models to price exotics via MC simulations. We price a cliquet. This exercise brings to light some concerns about model uncertainty.

7.4.1 Cliquet options

The pay-off for a cliquet is

$$\min(cap_{global}, \max(floor_{global}, \sum_{i=1}^{N} \min(cap_{local}, \max(floor_{local}, \frac{S_{t_i} - S_{t_{i-1}}}{S_{t_{i-1}}})))). \quad (7.7)$$

Cliquets allow the buyer to get downside protection, whilst getting some exposure to upside. The upside is capped so that the contract is not too expensive. The exposure to returns can be capped/floored locally and globally. [Wil02] points out that the cliquet's value depends on how the model treats volatility. In particular, the

price will depend on the forward volatility smile of the SV model considered. [Que02] points out that using models that fit the smile of vanilla options does not guarantee reliable prices for options that are sensitive to the forward smile. We priced the cliquet using MC simulations over $N = 2^{20} - 1 = 1048575$ paths. The option matures in three years and returns are monitored every quarter. To assess whether our simulation scheme was biased we repriced all the options used in the calibration. The simulated and actual prices were only 0.6 percent off at worst. To check the convergence of the exotic option estimates we used a convergence diagram. We didn't use standard error estimates because we use Sobol quasi-random uniform variates. Quasi-random numbers are correlated by construction. A convergence diagram is a plot of the price estimates at different simulation sample sizes. To reduce the computational effort one can store the current price estimate after every 20000 simulation paths. A convergence diagram for the VG-CIR SV model is depicted in Figure 7.3 for every 20000 paths. Similar diagrams can be generated for the other SV models. We conclude that a price estimate is adequate when subsequent prices remain within the market bid-offer spread for larger simulation sizes. In fact, we could have limited the simulation paths to 700000 and thus reduce the computational effort whilst remaining in the same bid-offer corridor.

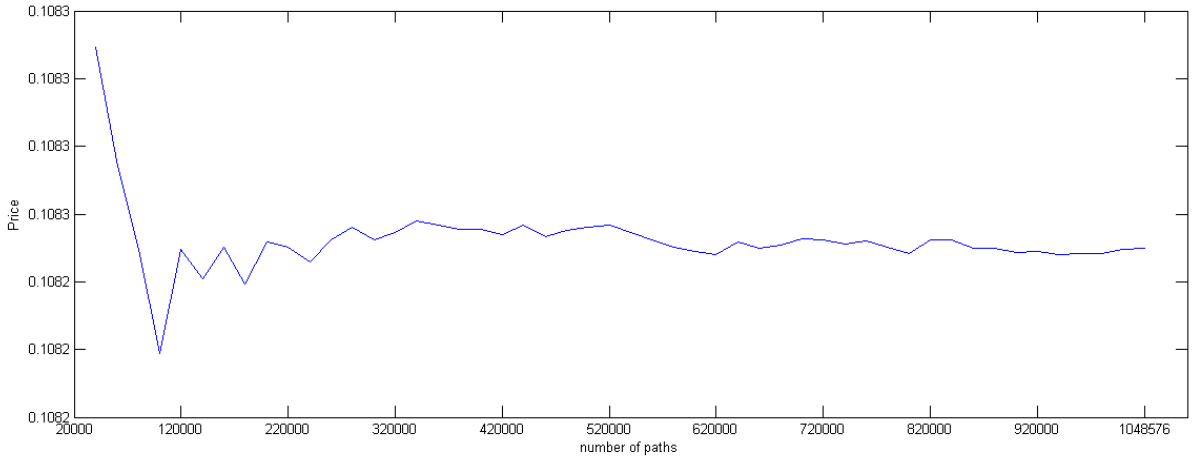

Figure 7.3: Convergence diagram for cliquet option under the VG-CIR SV model.

For reference, we summarise the cliquet prices in Table 7.1. Price variations of up to 4 percent are noted. The prices seem to be dominated by the volatility process applied to each Lévy model. The prices increase from CIR to Gamma and IG clocks. This suggests that the IG clock introduces the most volatile path dynamics. In lll we provide the code to price a cliquet option.

Listing 7.3: Script to price the cliquet option studied in this section.

```
%Script to run MC prices using FrFFT.
```

```
%data=[rfr T q K S0].
data=[0.019 3 0.012 1124.47  1124.47];

%Stochastic process parameters.
pars1=[0.1107 5.3962 15.2593 1.0423];
pars2=[0.3445 1.7580 1.5043 1];

%Stochastic process characteristic function.
chatfn1=@CGMYchar;
chatfn2=@charIntCIR;

%Number of paths (N) and number of cliquet resets until
    maturity(M).
M=12;
N=2^20-1;

payout=@payoffCliquet;

discrete=@nextSliceEuler2;

%Generate returns for a time-changed, mean-corrected Levy
    process.
procGHOUIG=[zeros(N,1) FrFFTDrawsDeep(@(u,mydata)
    TimeChangedchar(u,chatfn1,pars1,chatfn2,pars2,mydata),N,M
    ,3000,2^21,1,data)];

%Price a cliquet option.
prices=priceMCFrFFT(discrete,payout,data,w,procGHOUIG);
```

Clock	VG	NIG	Meixner	GH	CGMY
CIR	0.1116	0.1083	0.1083	0.1083	0.1081
Gamma	0.1128	0.1082	0.1087	0.1087	0.1087
IG	0.1095	0.1082	0.1100	0.1087	0.1103

Table 7.1: Cliquet prices: $cap_{local} = 0.05; floor_{local} = -0.03; floor_{global} = 0;$

Model uncertainty We refer to model uncertainty in the sense that very different prices can be obtained using different models to price the same instrument even when they are based on exactly the same:

- Market data on vanilla options.
- Good numerical schemes for calibration and Monte Carlo simulation.

Due to the leveraging in the instrument one can get very different results. The differences may also be due to pricing the product with inappropriate stochastic models, referring to the path properties discussed in section 3.3. The message with model uncertainty is:

- If the price differences are too high then you should think if you and your counterparty are able to understand all the risks in the product.
- Use a model as simple as possible relative to the instrument you are trying to price.
- In essence, be very careful about trading such instruments.

7.4.2 Further considerations

Here we consider how sensitive the price of the cliquet in subsection 7.4.1 is to minor movements in the individual model parameters of the CGMY-CIR SV model parameters in Table 5.2. The parameter sensitivities are calculated as the change in price for a 1 percent move in a prameter divided by a 1 percent change in the parameter. The parameter sensitivities are depicted in Figure 7.4.

The cliquet is mostly sensitive to changes in the parameter C under the CGMY-CIR model. This tells us that if we recalibrate the CGMY-CIR model and the parameter C changes slightly then our model prices and hedge ratios will change by a lot. This could result in mispricing and unstable hedge ratios. This analysis can assist with measuring and reducing model risk. To manage this model risk we could fix the parameter C so that it doesn't change when we recalibrate. The remaining parameters should still offer enough flexibility to enable us to fit the market prices of the calibration instruments closely.

If the model is going to be re-calibrated to new market data, e.g. daily, then one should be careful of unstable results due to naive re-calibration implementation. We could regularise the RMSE by including a penalty term to help keep parameter estimates and hedges stable. A Tikhonov regularisation method for calibrating Lévy models is discussed and implemented by [KMAE08].

Another approach would be to assess the impact of individually varying each model parameter on the skew. Varying a parameter may affect the level, slope, curvature or other property of the skew. One can then identify parameters that have a similar impact on the skew and fix those that are deemed redundant. Fixing some parameters may result in poorer fits to market data. However, spurious accuracy must be balanced against the effects of unstable parameters and hedges.

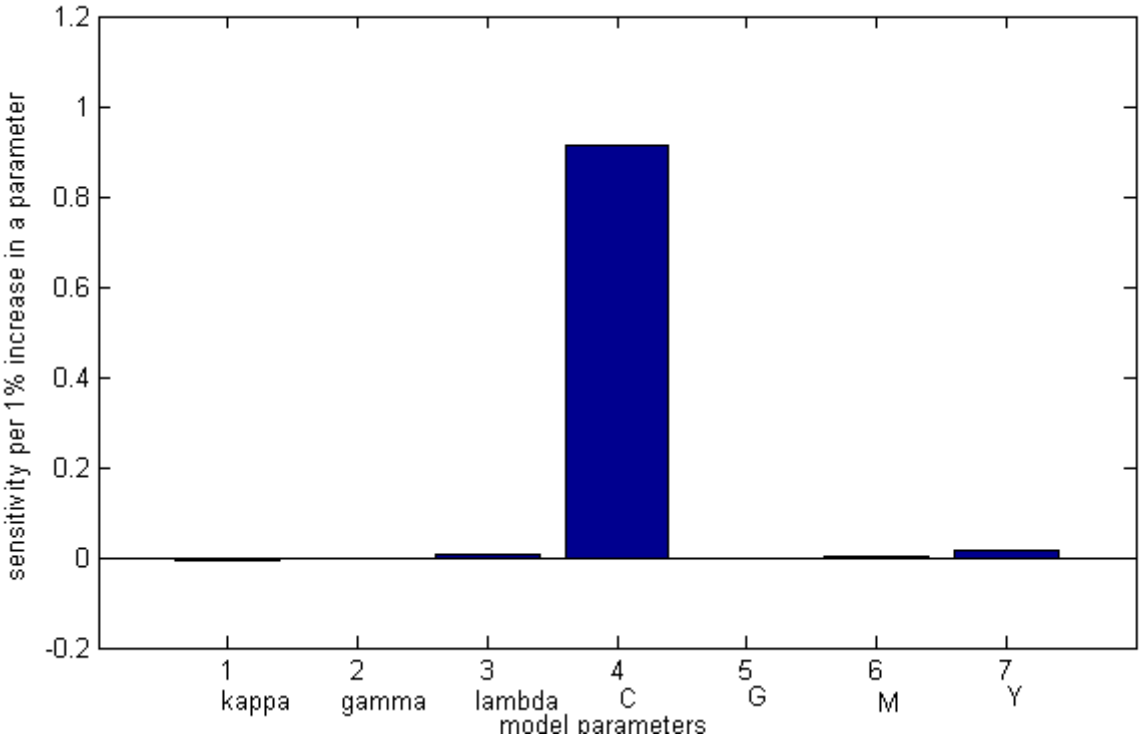

Figure 7.4: Call option price sensitivity for a 1 percent change in each of the model parameters of the CGMY-CIR calibration in subsection 1.2.1.

8 Conclusion

We explored tractable non-Gaussian models that better capture the characteristics of empirical returns. By extending the Lévy models to include stochastic volatility we were able to fit the volatility skew using a single set of parameters at all maturities. We discussed the implementation details for estimating the statistical and risk-neutral parameters for these models. We also pointed out some difficulties involved in this estimation process. We also presented simulation algorithms from the exact distribution of the models. These algorithms allowed us to estimate exotic option prices by Monte Carlo simulation. When pricing a cliquet we noticed the presence of model uncertainty where we get different prices for the same derivative under different models calibrated to the same market data. This work will be useful to practitioners and researchers who want to implement these models. Much of the implementation details are omitted in the original publications introducing the models. We aimed to fill these gaps in the literature. Matlab code has been provided.

Further work could use the tools we developed to investigate the practical application of these models. Key issues to address here are:

- Model re-calibration and their implication for exotic option pricing and hedging. This could involve keeping some model parameters constant, for example parameters that have little impact on the skew or forward skew.
- Dealing with model uncertainty. Suggest ways of choosing appropriate models for pricing and hedging of highly leveraged derivatives and examining any residual risks.

9 Appendix A

This table of option prices is taken from [Sch03].

Strike	May 2002 $T = 0.088$	June 2002 $T = 0.0184$	Sep 2002 $T = 0.436$	Dec 2002 $T = 0.692$	March 2003 $T = 0.936$	June 2003 $T = 1.192$	Dec 2003 $T = 1.708$
975			161.60	173.30			
995			144.80	157.00		182.10	
1025			120.10	133.10	146.50		
1050	84.50	100.70	114.80		143.00	171.40	
1075		64.30	82.50	97.60			
1090	43.10						
1100	35.60		65.50	81.20	96.20	111.30	140.40
1110		39.50					
1120	22.90	33.50					
1125	20.20	30.70	51.00	66.90	81.70	97.00	
1130		28.00					
1135		25.60	45.50				
1140	13.30	23.20		58.90			
1150		19.10	38.10	53.90	68.30	83.30	112.80
1160		15.30					
1170		12.10					
1175		10.90	27.70	42.50	56.60		99.80
1200			19.60	33.00	46.10	60.90	
1225			13.20	24.90	36.90	49.80	
1250				18.30	29.30	41.20	66.90
1275				13.20	22.50		
1300					17.20	27.10	49.50
1325					12.80		
1350						17.10	35.70
1400					10.10	25.20	
1450							17.00
1500							12.20

Table 9.1: Table of 75 call option mid prices on the S&P 500 at the close of business on 18 April 2002. The index closed at $S0 = 1124.47$. We used $r = 0.019$ and $q = 0.012$ per annum.

10 Appendix B

Model	Mean	Variance
Poisson(λ)	λ	λ
IG(a, b)	a/b	a/b^3
$\Gamma(a, b)$	a/b	a/b^2
VG(C,G,M)	$C\left(G-M\right)/MG$	$C\left(G^2+M^2\right)/\left(MG\right)^2$
CGMY(C, G, M, Y)	$C\left(M^{Y-1}-G^{Y-1}\right)\Gamma\left(1-Y\right)$	$C\left(M^{Y-2}-G^{Y-2}\right)\Gamma\left(2-Y\right)$
NIG(α, β, δ)	$\delta\beta/\sqrt{\alpha^2-\beta^2}$	$\delta\alpha^2\left(\alpha^2-\beta^2\right)^{-3/2}$
Meixner(α, β, δ)	$\alpha\delta\tan\left(\beta/2\right)$	$\frac{1}{2}\alpha^2\delta\left(\cos^{-2}\left(\beta/2\right)\right)$
GH($\alpha, \beta, \delta, \nu$)	$\beta\delta\left(\alpha^2-\beta^2\right)^{-1}\mathbf{K}_{\nu+1}(\zeta)\mathbf{K}_{\nu}^{-1}(\zeta)$	$\delta^2\left(\frac{\mathbf{K}_{\nu+1}(\zeta)}{\zeta\mathbf{K}_{\nu+1}(\zeta)}+\frac{\beta^2}{\alpha^2-\beta^2}\left(\frac{\mathbf{K}_{\nu+2}(\zeta)}{\mathbf{K}_{\nu}(\zeta)}-\frac{\mathbf{K}_{\nu+1}^2(\zeta)}{\mathbf{K}_{\nu}^2(\zeta)}\right)\right)$
Model	Skew	Kurtosis
Poisson(λ)	$1/\sqrt{\lambda}$	$3+1/\lambda$
IG(a, b)	$3/\sqrt{ab}$	$3\left(1+5\left(ab\right)^{-1}\right)$
$\Gamma(a, b)$	$2a^{-1/2}$	$3\left(1+2a^{-1}\right)$
VG(C,G,M)	$2C^{-1/2}\left(G^3-M^3\right)/\left(G^2+M^2\right)^{3/2}$	$3\left(1+2C^{-1}\left(G^4+M^4\right)/\left(M^2+G^2\right)^2\right)$
CGMY(C, G, M, Y)	$\frac{C\left(M^{Y-3}-G^{Y-3}\right)\Gamma(3-Y)}{\left(C(M^{Y-2}-G^{Y-2})\Gamma(2-Y)\right)^{3/2}}$	$3+\frac{C\left(M^{Y-4}-G^{Y-4}\right)\Gamma(4-Y)}{\left(C(M^{Y-2}-G^{Y-2})\Gamma(2-Y)\right)^2}$
NIG(α, β, δ)	$3\beta\alpha^{-1}\delta^{-1/2}\left(\alpha^2-\beta^2\right)^{-1/4}$	$3\left(1+\frac{\alpha^2+4\beta^2}{\delta\alpha^2\sqrt{\alpha^2-\beta^2}}\right)$
Meixner(α, β, δ)	$\sin\left(\beta/2\right)\sqrt{2/\delta}$	$3+\left(2-\cos\left(\beta\right)\right)/\delta$
GH($\alpha, \beta, \delta, \nu$)	-	-

Table 10.1: Table of population moments. $\zeta=\delta\sqrt{\alpha^2-\beta^2}$. The third and fourth moments of the GH distribution must be obtained numerically.

Model	Jump activity	Path variation
Poisson	Finite	Bounded
Comp. Poisson	Finite	Bounded
VG	Infinite	Bounded
CGMY	Finite $\Leftrightarrow Y < 0$	Unbounded $\Leftrightarrow Y \in [1, 2)$
NIG	Infinite	Unbounded
Meixner	Infinite	Unbounded
GH	Infinite	Unbounded
Model	**Hitting points**	**Creeping**
Poisson	$C = \mathbb{N}$	No creeping
Comp. Poisson	$C = \emptyset$	No creeping
VG	$C = \emptyset \Leftrightarrow \mu = 0$, otherwise $C = \mathbb{R}$.	Upward (downward) creeping $\Leftrightarrow \mu > 0 (\mu < 0)$
CGMY	$C = \emptyset \Leftrightarrow Y = 1$ or $Y \in (0, 1)$ and $\mu = 0$, otherwise $C = \mathbb{R}$.	Upward creeping $\Leftrightarrow Y \in (0, 1)$ and $\mu > 0$, Downward creeping $\Leftrightarrow Y \in (0, 1)$ and $\mu < 0$.
NIG	$C = \emptyset$	No creeping
Meixner	$C = \emptyset$	No creeping
GH	$C = \emptyset$	No creeping

Table 10.2: Path properties of the processes we used in the modelling.

Model	γ
Poisson	0
Comp. Poisson	$-\int_{\mathbb{R}\setminus\{0\}} x\Pi(dx)$
Gamma	$a(e^{-b}-1)/b$
IG	$\frac{a}{b}(2N(b)-1)$
CGMY	$C\left(\int_0^1 \exp(-Mx)\, x^{-Y}dx - \int_{-1}^0 \exp(Gx)\, \lvert x\rvert^{-Y}dx\right)$
NIG	$\frac{2\alpha\delta}{\pi}\int \sinh(\beta x)\, \mathbf{K}_1(\alpha x)\, dx$
Meixner	$\alpha\delta\tan(\beta/2) - 2\delta\int_1^\infty \frac{\sinh(\beta x/\alpha)}{\sinh(\pi x/\alpha)}dx$
GH	-

Model	$\Pi(dx)$
Poisson	$\lambda 1_{x=1}$
Comp. Poisson	$\lambda F(dx)$
Gamma	$ae^{-bx}x^{-1}1_{(x>0)}dx$
IG	$\frac{a}{\sqrt{2\pi x^3}}\exp(-b^2x/2)1_{(x>0)}dx$ VG
CGMY	$\left(C\exp(Gx)\,(-x)^{-1-Y} + C\exp(-Mx)\,x^{-1-Y}\right)dx$
NIG	$\frac{\alpha\delta}{\pi}\frac{\exp(\beta x)\mathbf{K}_1(\alpha\lvert x\rvert)}{\lvert x\rvert}dx$
Meixner	$\delta\frac{\exp(\beta x/\alpha)}{x\sinh(\pi x/\alpha)}dx$
GH	$\frac{\exp(\beta x)}{\lvert x\rvert}\left(\int_0^\infty \frac{\exp\left(-\lvert x\rvert\sqrt{2y+\alpha^2}\right)}{\pi^2 y\left[\mathbf{J}^2_{\lvert\nu\rvert}\left(\delta\sqrt{2y}\right)+\mathbf{N}^2_{\lvert\nu\rvert}\left(\delta\sqrt{2y}\right)\right]}dy + \mathbf{1}_{(\nu\geq 0)}\nu\exp(-\alpha\lvert x\rvert)\right)$

Table 10.3: The Lévy triplets of the processes used in this work ($\Sigma = 0$ for all processes in the table). The function $F(\cdot)$ denotes the cumulative density function. The first component of the Lévy triplet for the GH process is very complicated and thus omitted (see [Pra99] for further details).

11 MATLAB Code

11.1 Characteristic functions

```
function char=charCGMY(u,pars,data)
        %Characteristic function of CGMY process.
        %pars:  vector of model parameters.
        %data: vector of market inputs such as interest rate
            , dividend yield and the options time to maturity
            .
    C=pars(1); G=pars(2); M=pars(3); Y=pars(4);
    if nargin==3
                rfr=data(1);      %Risk-free interest rate.
        t=data(2);                %Time to maturity.
        q=data(3);                %Dividend yield.
                CgamY=C*gamma(-Y);
                MY=M^Y;
                GY=G^Y;
                w=-CgamY*((M-1)^Y-MY+(G+1)^Y-GY);
                char=exp(1i*u*(rfr-q+w)*t).*exp(t*CgamY*((M
                    -1i*u).^Y-MY+(G+1i*u).^Y-GY));
    elseif nargin==2
        char=exp(C*gamma(-Y)*((M-1i*u).^Y-M^Y+(G+1i*u).^Y-G^
            Y));
    end
end
```

```
function char=charGH(u,pars,data)
        %Characteristic function of generalised hyperbolic (
            GH) distribution.
        %pars:  vector of model parameters.
        %data: vector of market inputs such as interest rate
            , dividend yield and the options time to maturity
            .
    a=pars(1); b=pars(2); c=pars(3); nu=pars(4);
    a2_b2=(a-b)*(a+b);
    zeta=c*sqrt(a2_b2);
```

```
    if nargin==3
        rfr=data(1);      %Risk-free interest rate.
        t=data(2);                %Time to maturity.
        q=data(3);                %Dividend yield.
        w=-log(((a-b)*(a+b)/((a-b-1)*(a+b+1)))^(nu/2)*
            besselk(nu,c*sqrt((a-b-1)*(a+b+1)))/besselk(nu,c*
            sqrt((a-b)*(a+b))));
        char=exp(1i*u*(rfr-q+w)*t).*(((a-b)*(a+b)./((a-(b+1i
            *u)).*(a+b+1i*u))).^(nu/2).*besselk(nu,c*sqrt(a*a
            -(b+1i*u).*(b+1i*u)))/besselk(nu,c*sqrt((a-b)*(a+
            b)))).^t;
    elseif nargin==2
        char=(a2_b2./((a-b-1i*u).*(a+b+1i*u))).^(nu/2).*
            besselk(nu,c*sqrt((a-b-1i*u).*(a+b+1i*u)))/
            besselk(nu,zeta);
    end
end
```

```
function char=charNIG(u,pars,data)
        %Characteristic function of normal inverse gaussian
            (NIG).
        %pars:  vector of model parameters.
        %data: vector of market inputs such as interest rate
            , dividend yield and the options time to maturity
            .
    a=pars(1); b=pars(2); c=pars(3);
    if nargin==3
        rfr=data(1);      %Risk-free interest rate.
        t=data(2);                %Time to maturity.
        q=data(3);                %Dividend yield.
        w=c*(sqrt(a*a-(b+1)^2)-sqrt(a*a-b*b));
        char=exp(1i*u*(rfr-q+w)*t).*exp(-c*t*(sqrt(a*a-(b+1i
            *u).*(b+1i*u))-sqrt(a*a-b*b)));
    elseif nargin==2
        char=exp(-c*(sqrt(a*a-(b+1i*u).*(b+1i*u))-sqrt(a*a-b
            *b)));
    end
end
```

```
function char=charMeixner(u,pars,data)
        %Characteristic function of a meixner distribution
        %pars:  vector of model parameters.
        %data: vector of market inputs such as interest rate
            , dividend yield and the options time to maturity
```

```
            .
    a=pars(1); b=pars(2); c=pars(3);
    if nargin==3
                rfr=data(1);      %Risk-free interest rate.
        t=data(2);                %Time to maturity.
        q=data(3);                %Dividend yield.
                w=-2*c*(log(cos(b/2)/cos((a+b)/2)));
                char=exp(1i*u*(rfr-q+w)*t).*(cos(b/2)./cosh
                    ((a*u-1i*b)/2)).^(2*c*t);
    elseif nargin==2
        char=(cos(b/2)./cosh((a*u-1i*b)/2)).^(2*c);
    end
end
```

```
function char=charVG(u,pars,data)
        %Characteristic function of VG(C,G,M).
        %pars:  vector of model parameters.
        %data: vector of market inputs such as interest rate
            , dividend yield and the options time to maturity
            .
    C=pars(1); G=pars(2); M=pars(3);
    if nargin==3
        rfr=data(1);      %Risk-free interest rate.
        t=data(2);                %Time to maturity.
        q=data(3);                %Dividend yield.
        w=rfr-q+C*log((M-1)*(G+1)/(M*G));
        char=exp(1i*u*(w*t)).*(G*M./(G*M+(M-G)*1i*u+u.*u))
            .^(t*C);
    elseif nargin==2
        char=(G*M./(G*M+(M-G)*1i*u+u.*u)).^C;
    end
end
```

```
function char=charIntCIR(u,pars,t)
    %Characteristic function of integrated CIR process.
    %Used to model stochastic volatility.
        %pars:  vector of model parameters.
        %t:              time to maturity.
    kappa=pars(1); gamma=pars(2); lambda=pars(3); y0=pars(4)
        ;

    zeta=sqrt(kappa*kappa-2i*lambda*lambda*u);
    zeta_t_2=zeta*t/2;
```

```
    char=exp((kappa/lambda)^2*gamma*t+2i*y0*u./(kappa+zeta.*
        coth(zeta_t_2)))./(cosh(zeta_t_2)+kappa*sinh(zeta_t_2
        )./zeta).^(2*kappa*gamma/(lambda*lambda));
end
```

```
function char=charIntOUGamma(u,pars,t)
    %Characteristic function of integrated Ornstein-
        Uhlenbeck process.
    %Used to model stochastic volatility.
        %pars:  vector of model parameters.
        %t:             time to maturity.
    a=pars(1); b=pars(2); lambda=pars(3); y0=pars(4);
    exp_lambda=1i*u*(1-exp(-lambda*t))/lambda;

    char=exp(y0*exp_lambda+lambda*a*(b*log(b./(b-exp_lambda)
        )-1i*u*t)./(1i*u-lambda*b));
end
```

```
function char=charIntOUIG(u,pars,t)
    %Characteristic function of integrated Ornstein-
        Uhlenbeck inverse Gaussian (OUIG) process.
    %Used to model stochastic volatility.
        %pars:  vector of model parameters.
        %t:             time to maturity.
    a=pars(1); b=pars(2); lambda=pars(3); y0=pars(4);
    exp_lambda=(1-exp(-lambda*t));

    kap=-2i*u/(lambda*b*b);
    kap_sqrt=sqrt(1+kap);
    A=(1-sqrt(1+kap.*exp_lambda))./kap+(atanh(sqrt(1+kap.*
        exp_lambda)./kap_sqrt)-atanh(1.0./kap_sqrt))./
        kap_sqrt;
    A(u==0)=-0.5*(1-exp(-lambda*t));
    char=exp(1i*u.*y0*exp_lambda/lambda+a*2i*u.*A/(lambda*b)
        );
end
```

```
function char=TimeChangedchar(u,chatfun1,pars1,chatfun2,
    pars2,data)
        %Characteristic function of a time-changed
            stochastic volatility Levy model.
        %chatfun1:      Characteristic function of a Levy
            model to be time-changed.
        %pars1:         model parameters of chatfun1.
```

```
        %chatfun2:        Characteristic function of
            stochastic clock.
        %pars2:           model parameters of chatfun2.
        %data: vector of market inputs such as interest rate
            , dividend yield and the options time to maturity
            .
    if nargin==6
        rfr=data(1);     %Risk-free interest rate.
        t=data(2);                %Time to maturity.
        q=data(3);                %Dividend yield.
        temp=chatfun1(u,pars1);
        char=exp(1i*u*(rfr-q)*t).*chatfun2(-1i*log(temp),
            pars2,t)./chatfun2(-1i*log(chatfun1(-1i,pars1)),
            pars2,t).^(1i*u);
        char(temp==0)=0;
    end
    if nargin==5
        char=chatfun2(-1i*log(chatfun1(u,pars1)),pars2,1);
    end
end
```

11.2 Parameter constraints

```
function [c,ceq]=constrainthPSOCGMY(x0)
    %Constraints for CGMY
    %x0:          vector of model parameters.
        if length(x0)>4
                tempx0=x0(4:end);        %For stochastic
                    volatility model.
        else
                tempx0=x0;        %For usual Levy model.
        end
    C=tempx0(1);     G=tempx0(2);     M=tempx0(3);     Y=tempx0
        (4);

    c=[-C,-G,-M,Y-2];
    ceq=[];                          %Equality constraints.
end
```

```
function [c,ceq]=constrainthPSOGH(x0)
    %Constraints for GH
    %x0:          vector of model parameters.
```

```
        if length(x0)>4
                tempx0=x0(4:end);       %For stochastic
                    volatility model.
        else
                tempx0=x0;              %For usual Levy model.
        end
    a=tempx0(1); b=tempx0(2); c=tempx0(3); nu=tempx0(4);

    c=[-a,-c,-a-b+0.0003,-a+b+0.0003];
    ceq=[];                             %Equality constraints.
end
```

```
function [c,ceq]=constrainthPSOMeixner(x0)
    %Meixner model parameter constraints.
    %x0:          vector of model parameters.
        if length(x0)>4
                tempx0=x0(4:end);       %For stochastic
                    volatility model.
        else
                tempx0=x0;              %For usual Levy model.
        end
    a=tempx0(1);      b=tempx0(2);      c=tempx0(3);
    c=[-a,-c,-pi-b,b-pi];                   %Inequality
       constraints in the form f(x)<=0.
    ceq=[];                             %Equality constraints.
end
```

```
function [c,ceq]=constrainthPSONIG(x0)
    %NIG model parameter constraints.
        %x0:     vector of model parameters.
        if length(x0)>4
                tempx0=x0(4:end);       %For stochastic
                    volatility model.
        else
                tempx0=x0;              %For usual Levy model.
        end
    a=tempx0(1);      b=tempx0(2);      c=tempx0(3);

    c=[-a,-c,-a-b,b-a];             %Inequality constraints in
       the form f(x)<=0.
    ceq=[];                             %Equality constraints.
end
```

```
function [c,ceq]=constraintVG(x0)
    %VG model parameter constraints.
        %x0:     vector of model parameters.
        if length(x0)>4
                tempx0=x0(4:end);          %For stochastic
                    volatility model.
        else
                tempx0=x0;          %For usual Levy model.
        end
    C=tempx0(1); G=tempx0(2); M=tempx0(3);
    c=[-C,-G,-M];          %Inequality constraints in the form
        f(x)<=0.
    ceq=[];                          %Equality constraints.
end
```

11.3 Moments

```
function moments=momentsVG2(pars)
% Theoretical moments of variance gamma [VG(C,G,M)]
    distribution.
    C=pars(1); G=pars(2); M=pars(3);

    mg=M*G;
    g2=G*G;
    m2=M*M;

    mean=C*(G-M)/(mg);
    var=C*(g2+m2)/(mg*mg);
    skew=2*C^(-0.5)*(g2*G-m2*M)/(g2+m2)^1.5;
    kurt=3*(1+2*(g2*g2+m2*m2)/((g2+m2)*(g2+m2))/C);

        %Mean-correcting term.
        w=-C*log((M-1)*(G+1)/mg);

    moments=[mean-w, var, skew, kurt];
end
```

```
function moments=momentsMeixner(pars)
% Theoretical moments of Meixner distribution.
    a=pars(1);   b=pars(2);   c=pars(3);

    mean=a*c*tan(b/2);
```

```
    var=0.5*a*a*c/cos(b/2)/cos(b/2);
    skew=sin(0.5*b)*sqrt(2/c);
    kurt=3+(2-cos(b))/c;

        %Mean-correcting term.
        w=2*c*(log(cos(b/2)/cos((a+b)/2)))

    moments=[mean-w, var, skew, kurt];
end
```

```
function moments=momentsNIG(pars)
% Theoretical moments of NIG distribution.
    a=pars(1);  b=pars(2);  c=pars(3);

    mean=c*b/sqrt(a*a-b*b);
    var=a*a*c*(a*a-b*b)^(-3/2);
    kurt=3*(1+(a*a+4*b*b)/(c*a*a*sqrt(a*a-b*b)));
    skew=3*b*((a+b)*(a-b))^-0.25/a/sqrt(c);

    moments=[mean, var, skew, kurt];
end
```

```
function moments=momentsCGMY(pars)
% Theoretical moments of CGMY distribution.
    C=pars(1); G=pars(2); M=pars(3); Y=pars(4);

    mean=C*gamma(1-Y)*(M^(Y-1)-G^(Y-1));
    var=C*gamma(2-Y)*(M^(Y-2)+G^(Y-2));
    skew=C*gamma(3-Y)*(M^(Y-3)-G^(Y-3))/var^1.5;
    kurt=3+C*gamma(4-Y)*(M^(Y-4)+G^(Y-4))/var/var;

        %Mean-correcting term.
        w=C*gamma(-Y)*((M-1)^Y-M^Y+(G+1)^Y-G^Y);

    moments=[mean-w, var, skew, kurt];
end
```

```
function moments=momentsApprox(f,x0,n,h,minErr,maxErr)
%function to approximate the first derivative of f.
%Uses Taylor expansion. Increases or decreases h to achieve
   desired
%error.
%f:              function to differentiate.
%x0:     the point at which we are taking the derivative.
```

```
%n:       order of the highest desired derivative.
%h:       spacing for x values. Hint:start with a large h (e.g
    . 0.1)
%minErr:          the smallest allowed error.
%maxErr:          the largest allowed error.
     maxIterations=8;
     counterInterations=0;
     [derivatives,error1]=differentiate(f,x0,n,h);
     while (counterInterations<=maxIterations)
          if (error1<minErr)
               h=h*2;
          elseif (error1>maxErr)
               h=h*0.51;
          else
               return
          end
          counterInterations=counterInterations+1;
          [derivatives,error1]=differentiate(f,x0,n,h);
     end

     mu=derivatives(2);
     k=zeros(n,1);
     k(1,1)=mu;
     for i=2:n
          for j=0:i
               k(i,1)=k(i,1)+nchoosek(i,j)*(-1)^(i-j)*
                   derivatives(j+1)*mu^(i-j);
          end
     end

     moments=[k(1) k(2) k(3)*k(2)^-1.5 k(4)/(k(2)*k(2))];
end
```

```
function [yy, fval]=sampleMOM(chatfun,x0,constraints,moments
    ,sampleMoments)
%Function to calibrate models to sample moments

     options=optimset('Algorithm','interior-point','
         MaxFunEvals',4000,'TolX',1e-12);
     if nargin==4
          f=@(x)sqrt((moments(x)./sampleMoments-1)*(moments(x)
              ./sampleMoments-1)');
     else
```

```
        f=@(x)(momentsApprox2(@(u)chatfun(-1i*u,x,[0 1 0]))
            ./sampleMoments-1)*(momentsApprox2(@(u)chatfun(-1
            i*u,x,[0 1 0]))./sampleMoments-1)';
    end
    [yy, fval]=fminunc(@(x)Fun(x,f,constraints),x0,options);
end
```

11.4 Cosine option pricer

```
function func=COSFast(a,b,data,pars,chatfun,K)
        %Put option price via COS method
        %[a,b]:    the truncated range of integration.
        %data:     a vector of current market parameters.
        %pars:     a vector of model parameters.
        %chatfun: the characteristic function of the log
            stock price [ln(ST)].
        %K:                A grid of strike prices at which
            the option will be evaluated.
    N=pars;

    rfr=data(1);
    t=data(2);
    S0=data(5);
    func=K.*exp(-rfr*t).*real((chatfun((1./(b-a))*((1:N)-1)
        *pi).*(((2./(b-a))*ones(1,N)).*(-chi(a,zeros(size(K
        ,1),1),((1:N)-1),a,b)+mypsi(a,zeros(size(K,1),1),((1:
        N)-1),a,b))).*exp(1i*((log(S0./K)-a)*ones(1,N))
        .*((1./(b-a))*((1:N)-1) *pi)))*[0.5; ones(N-2,1);
        0.5]);
end
```

```
function func=chi(c,d,k,a,b)
        %COS method weights

    temp=(1./(b-a))*k *pi;
    temp2=((d-a)./(b-a))*k *pi;
    temp3=((c-a)./(b-a))*k *pi;
    part1=1.0./(1.0+temp.*temp);
    myDim=length(k);
%       func=part1.*(cos(temp2).*exp(d*ones(1,length(k)))-cos(
    temp3).*exp(c*ones(1,length(k)))+temp.*sin(temp2).*exp(d*
```

```
    ones(1,length(k)))-temp.*sin(temp3).*exp(c*ones(1,length(
    k))));
      func=part1.*(cos(temp2).*exp(repmat(d,1,myDim))-cos(
          temp3).*exp(repmat(c,1,myDim))+temp.*sin(temp2).*exp(
          repmat(d,1,myDim))-temp.*sin(temp3).*exp(repmat(c,1,
          myDim)));

end
```

```
function func=mypsi(c,d,k,a,b)
        %COS method weights

    j=[1,1:(length(k)-1)];
    temp=(1./(b-a))*j *pi;
    temp2=((d-a)./(b-a))*j *pi;
    temp3=((c-a)./(b-a))*j *pi;
    func=(sin(temp2)-sin(temp3))./temp;
    func(:,1)=d-c;
end
```

11.5 Risk-neutral calibration

```
function [x,fval,gfx,output]=calibrateOptions(data,pars,
    chatfun,constraints,x0,strikes,market)
%Function to calibrate Levy models to market data
    options=optimset('MaxFunEvals',3000,'TolX',1e-12,'
        MaxIter',2000,'Algorithm','interior-point');
    f=@(x)objfun(data,pars,chatfun,x,strikes,market);
    [x,fval,gfx,output]=fminunc(@(x)Fun(x,f,constraints),x0,
        options);
end
```

```
function ansa = objfun(data,pars,chatfun,pars1,strikes,
    market)
%Function to calculate the sum of squared errors.
    rfr=data(1);  q=data(3);  S0=data(5);
    T=[0.088,0.1840,0.4360,0.6920,0.9360,1.1920,1.7080];
    model=zeros(75,1);

    numStrikes=cumsum([1 5 12 12 14 11 11 10]);      %To keep
        track of the number of strikes for each maturity in
        T
```

```
    strikes1=numStrikes(1:end-1);
    strikes2=numStrikes(2:end);
    for i=1:7
        data=[rfr T(i) q 1 S0];

        if length(pars1)>=6
            chatfun2=@(u)chatfun(u,pars1(4:end),[pars1(1:3)
                1],data);
        else
            chatfun2=@(u)chatfun(u,pars1,data);
        end

        K=strikes(strikes1(i):(strikes2(i)-1));
        if length(pars1)>=6 %For stochastic volatility model
            price=putcall(rfr,T(i),q,COSFast(-220*K./K,200*K
                ./K,data,pars,chatfun2,K),K,S0);
        else     %For usual Levy process
            price=putcall(rfr,T(i),q,COSFast(-220*K./K,200*K
                ./K,data,pars,chatfun2,K),K,S0);
        end
        model(strikes1(i):(strikes2(i)-1),1)=price;
    end

    ansa=(market-model)'*(market-model);

end
```

11.6 Exact simulations

```
function CIRt=procCIRTrans(pars,T,N,M,skip)
    kappa=pars(1);
    gam=pars(2);
    lambda=pars(3);
    y0=pars(4);
    d=4*kappa*gam/(lambda*lambda);
    dt=T/M;
%       skip=skip+1;
%       q1 = qrandstream('sobol',M,'skip',skip*(N+skip)+5);
    skip=skip+1;
    q2=sobolset(M,'Skip',skip*(N+skip)+5,'Leap',2);
    q2 = scramble(q2,'MatousekAffineOwen');%qrandstream('
        sobol',M,'skip',skip*(N+skip)+5);
```

```
    CIRt=zeros(N,M+1);
    CIRt(:,1)=y0;

    if d>1
        chi=chi2inv(net(q2,N),d-1);
        clear q2 skip gam y0
        v=zscore(randn(N,M));
        c=lambda*lambda*(1-exp(-kappa*dt))/(4*kappa);
        for i=2:M+1
            CIRt(:,i)=c*((v(:,i-1)+sqrt(CIRt(:,i-1)*exp(-
                kappa*dt)/c)).^2+chi(:,i-1));
        end
    else
        c=lambda*lambda*(1-exp(-kappa*dt))/(4*kappa);
        ran2=net(q2,N);
        clear q2 skip gam y0
        for i=2:M+1
            poissN=random('poiss',CIRt(:,i-1)*exp(-kappa*dt)
                /(lambda*lambda*(1-exp(-kappa*dt))/(4*kappa))
                /2);
            chiSqrd=chi2inv(ran2(:,i-1),d+2*poissN);
            CIRt(:,i)=c*chiSqrd;
        end
    end
end
```

```
function VGt=procVGBridge(pars,T,N,MM,skip,Times)
    C=pars(1);
    G=pars(2);
    M=pars(3);

    nu=1/C;
    sigma=sqrt(2*C/(G*M));
    theta=-sqrt((1/G+1/M)^2-4/(G*M))/nu;

    if nargin==5
        dt=T/MM;
        Gammat=GammaBridge(ones(N,1)*(dt:dt:T),nu,skip);
    end
    if nargin==6
        Gammat=GammaBridge(Times(:,2:end),nu,skip);
    end
```

```
    VGt=theta*Gammat+sigma*BrownianBridge2(Gammat(:,1:MM),1,
        skip+10);
end
```

11.7 Fractional fast Fourier transform simulations

```
function func=FrFFTSimulations(chatfun,N,M,L,grid,skip,data)
%Draw from a distribution using its characteristic function
    and FrFFT.

    skip=skip+1;
    q=sobolset(M,'Skip',skip*(N+skip)+5);
    unif=net(q,N);

    func=zeros(N,M);
    T=data(3);
    for i=1:M
        dt=T*i/M;
        moments_temp=momentsApprox(@(u)chatfun(-1i*u,[0 dt
            0]),0,4,0.95,1e-6,1e-5);
        chatfun_centred=@(u)exp(-1i*moments_temp(1)*u).*
            chatfun(u,[0 dt 0]);
        moments=momentsApprox(@(u)chatfun_centred(-1i*u)
            ,0,4,0.95,10e-6,10e-5);
        D=2*(moments(1)+200*sqrt(moments(2)+sqrt(moments(4))
            ));

        xy=MCFrFFTFast(@(u)chatfun(u,[0 dt 0]),D,L,grid);
            %grid=2^24 for meix and cgmy

        Fx=max(xy(2,:),1e-16);

        tiny=min(unif(:,i));
        big=max(unif(:,i));

        start=find(Fx<=tiny,1,'last');
        stop=find(Fx>=big,1,'first');

        func(:,i)=pchip(Fx(start:(stop+1)),xy(1,start:(stop
            +1)),unif(:,i))+moments_temp(1);
    end
end
```

```
function func=MCFrFFTFast(chatfun,D,L,N)
%Function to retrieve the cumulative distribution function (
    CDF) given the
%characteristic function.
    du=L/N;
    dx=D/N;

    %Grid points.
    u1=-0.5*N*du;
    x1=-0.5*N*dx;

    %Augmented characteristic function of CDF.
    moments=momentsApprox(@(u)chatfun(-1i*u),0,4,0.9,10e
        -6,10e-5);
    chatfun_centred=@(u)exp(-1i*moments(1)*u).*chatfun(u);
    charfun=@(u)charfunr(u,chatfun_centred,D);

    alpha=du*dx;

    Fx=real(exp(-1i*u1*((((1:N)-1)-N/2)*dx-x1)).*FrFFT(exp
        (-1i*(((1:N)-1)-N/2)*du*x1).*charfun((((1:N)-1)-N/2)*
        du)*du,alpha/(2*pi)))/(2*pi)+0.5;
    func=[(((1:N)-1)-N/2)*dx;Fx];
end
```

```
function func=FrFFT(f,alpha)
%Fractional FFT
    N=length(f);

    edges=[0.5  1];

    temp=ifft(fft([f./exp(1i*pi*alpha*((1:N)-1).*((1:N)-1)),
        zeros(1,N)].*[edges ones(1,2*N-4) edges(2:-1:1)]).*
        fft([exp(1i*pi*alpha*((1:N)-1).*((1:N)-1)),exp(1i*pi*
        alpha*(N-((1:N)-1)).*(N-((1:N)-1)))].*[edges ones
        (1,2*N-4) edges(2:-1:1)]).*[edges ones(1,2*N-4) edges
        (2:-1:1)]);

    func=temp(1:N)./exp(1i*pi*alpha*((1:N)-1).*((1:N)-1));
end
```

```
function price=priceMCFrFFT(discrete,payout,data,w,X)
```

```
%Function to calculate an option price by Monte Carlo
    simulation.
%Inputs:
%       rng=function handle of a Random Number Generator (e.
    g. VG process).
%       discrete=discretisation scheme.
%       payout=function handle of payout function (e.g.
    myMax for vanilla options).
%       call_put=indicator. =1 for call; =-1 for put.
%       data= market data for the options contract. K, S0,
    rfr, T.
%       M=points per path.
%       N=number of MC simulations.

     %Option specific market data
     rfr=data(1);
     T=data(2);
     q=data(3);
     S0=data(5);

     riskNeutralDrift=(rfr-q+w);

     S=discrete(riskNeutralDrift,X,S0,T);
     pay=payout(S);

     price=mean(pay)*exp(-rfr*T);

end
```

11.8 Utility functions

```
function [derivatives,erorr1]=differentiate(f,x0,n,h)
     %An nth order Taylor polynomial approximation. n>=2.
     %n: order of the highest desired derivative.
     %h: spacing for x values. Hint:start with a large h (e.g
         . 0.1)
     x=x0+h*(1:n);     %Try h=2^-5. Aim for 0.005bps to 0.05
         bps. Then can look for further refinement
     b=f(x')-f(x0);
     A=x;
     for i=2:n
          A=[x;repmat(x,i-1,1).*A];
```

```
    end

    derivatives=(A'\b).*factorial((1:n)');
    ii=n-1;
    jj=n-1;

    erorr1=10000*abs(A(ii,1:jj)*derivatives(1:jj)-A(ii,1:jj
        +1)*derivatives(1:jj+1))./abs(A(ii,1:jj)*derivatives
        (1:jj));

    derivatives=[1;derivatives];

end
```

```
function cliq=payoffCliquet(S)
% Payoff function for a cliquet option.
    capGlob=inf;
    capLoc=0.05;
    floorGlob=0;
    floorLoc=-0.03;
    cliq=min(capGlob,max(floorGlob,sum(min(capLoc,max(
        floorLoc,(S(:,2:end)./S(:,1:end-1)-1)'))))));
end
```

```
function func=nextSliceEuler2(r_dt,X,S,T)
%Evolve share price using Euler discretisation scheme.
[Npaths,Nsteps]=size(X);
% VGt=r_dt+X;
% [mean(VGt(:,end)) var(VGt(:,end)) skewness(VGt(:,end))
    kurtosis(VGt(:,end))]
    func=S.*exp((T*ones(Npaths,1)*(1:Nsteps)/Nsteps)*r_dt+X)
        ;
%       hist(func,20)
end
```

```
% d-dimensional objective function
function z=Fun(u,objFunc,constraint)

% Objective
z=objFunc(u);

% Apply nonlinear constraints by penalty method
% Z=f+sum_k=1^N lam_k g_k^2 *H(g_k)
z=z+getnonlinear(u,constraint);
```

```
function Z=getnonlinear(u,constraint)
Z=0;
% Penalty constant
lam=10^15; %lameq=10^15;
[g,ceq]=constraint(u);

% Inequality constraints
for k=1:length(g),
    Z=Z+ lam*g(k)^2*getH(g(k));
end

% Equality constraints (when geq=[], length->0)
% for k=1:length(geq),
%    Z=Z+lameq*geq(k)^2*geteqH(geq(k));
% end


% Test if inequalities hold
function H=getH(g)
if g<=0,
    H=0;
else
    H=1;
end

% Test if equalities hold
% function H=geteqH(g)
% if g==0,
%     H=0;
% else
%     H=1;
% end
```

```
function parity=putcall(rfr,t,q,price,K,S)
%function to determine the European call option price by put
   -call parity

    parity=price-K*exp(-rfr*t)+S*exp(-q*t);
end
```

```
function summary=mystats(mkt,model)
%Function to produce APE, AAE, ARPE & RMSE goodness of fit
    statistics
```

```
%APE=average absolute error as % of mean price
%AAE=average absolute error
%ARPE=average relative percentage error
%RMSE=root mean-square error
%mkt=market prices, model=model prices
    abs_diff=abs(mkt-model);
    AAE=mean(abs_diff);
    APE=AAE/mean(mkt);
    ARPE=sum(abs_diff./mkt)/length(mkt);
    RSME=sqrt(mean(abs_diff.^2));
    summary=[APE*100,AAE,RSME,ARPE*100];
end
```

Bibliography

[Ack04] Peter John Acklam. An algorithm for computing the inverse normal cumulative distribution function, 2004.

[App04] David Applebaum. *Lévy processes and stochastic calculus*, volume 93 of *Cambridge Studies in Advanced Mathematics*. Cambridge university press, 2004.

[AR01] Søren Asmussen and Jan Rosiński. Approximations of small jumps of lévy processes with a view towards simulation. *Journal of Applied Probability*, 38(2): 482–493, 2001.

[Ber96] Jean Bertoin. Lévy processes, 1996.

[BK11] Laura Ballotta and Ioannis Kyriakou. Monte carlo option pricing coupled with fourier transform for the cgmy process. *Cass Business School Research*, 2011.

[Bla76] Fisher Black. Studies of stock market volatility changes, uproceedings of the 1976 meeting of the american statistical association. In *Business and Economic Statistics Section*, volume 177, page 181, 1976.

[BN97] Ole E Barndorff-Nielsen. Processes of normal inverse gaussian type. *Finance and stochastics*, 2(1): 41–68, 1997.

[BNS01a] Ole E Barndorff-Nielsen and Neil Shephard. *Modelling by Lévy processess for financial econometrics*. Springer, 2001.

[BNS01b] Ole E Barndorff-Nielsen and Neil Shephard. Non-gaussian ornstein–uhlenbeck-based models and some of their uses in financial economics. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2): 167–241, 2001.

[BNS03] Ole E Barndorff-Nielsen and Neil Shephard. Integrated ou processes and non-gaussian ou-based stochastic volatility models. *Scandinavian Journal of Statistics*, 30(2): 277–295, 2003.

[Bre71] Jean Bretagnolle. Résultats de kesten sur les processus à accroissements indépendants. In *Séminaire de Probabilités V Université de Strasbourg*, pages 21–36. Springer, 1971.

[BS91] David H Bailey and Paul N Swarztrauber. The fractional fourier transform and applications. *SIAM review*, 33(3): 389–404, 1991.

[CGMY02] Peter Carr, Hélyette Geman, Dilip B Madan, and Marc Yor. The fine structure of asset returns: An empirical investigation*. *The Journal of Business*, 75(2): 305–333, 2002.

[Cho04] Kyriakos Chourdakis. Option pricing using the fractional fft. *Journal of Computational Finance*, 8(2): 1–18, 2004.

[CIJR85] John C Cox, Jonathan E Ingersoll Jr, and Stephen A Ross. A theory of the term structure of interest rates. *Econometrica: Journal of the Econometric Society*, pages 385–407, 1985.

[CM99] Peter Carr and Dilip Madan. Option valuation using the fast fourier transform. *Journal of Computational Finance*, 2(4): 61–73, 1999.

[Con01] Rama Cont. Empirical properties of asset returns: stylized facts and statistical issues. *Quantitative Finance*, 1: 223–236, 2001.

[CT04] Rama Cont and Peter Tankov. *Chapman & Hall/CRC Financial Mathematics Series*. Boca Raton: Chapman and Hall/CRC, 2004.

[CW03] Peter Carr and Liuren Wu. The finite moment log stable process and option pricing. *The Journal of Finance*, LVIII, 2003.

[CW04] Peter Carr and Liuren Wu. Time-changed lévy processes and option pricing. *Journal of Financial Economics*, 71(1): 113–141, 2004.

[DMJ87] Armido R DiDonato and Alfred H Morris Jr. Algorithm 654: Fortran subroutines for computing the incomplete gamma function ratios and their inverse. *ACM Transactions on Mathematical Software (TOMS)*, 13(3): 318–319, 1987.

[EK95] Ernst Eberlein and Ulrich Keller. Hyperbolic distributions in finance. *Bernoulli*, 1(3): 281–299, 1995.

[EKP98] Ernst Eberlein, Ulrich Keller, and Karsten Prause. New insights into smile, mispricing, and value at risk: The hyperbolic model*. *The Journal of Business*, 71(3): 371–405, 1998.

[FO08] Fang Fang and Cornelis W Oosterlee. A novel pricing method for european options based on fourier-cosine series expansions. *SIAM Journal on Scientific Computing*, 31(2): 826–848, 2008.

[Gla03] Paul Glasserman. *Monte Carlo methods in financial engineering*, volume 53. Springer, 2003.

[Gri92] Andreas Griewank. Achieving logarithmic growth of temporal and spatial complexity in reverse automatic differentiation. *Optimization Methods and software*, 1(1): 35–54, 1992.

[Hal79] Christian Halgreen. Self-decomposability of the generalized inverse gaussian and hyperbolic distributions. *Probability Theory and Related Fields*, 47(1): 13–17, 1979.

[Hug98] Paul Hughett. Error bounds for numerical inversion of a probability characteristic function. *SIAM journal on numerical analysis*, 35(4): 1368–1392, 1998.

[JK03] Stephen Joe and Frances Y. Kuo. Remark on algorithm 659: implementing sobol's quasirandom sequence generator. *ACM Transactions on Mathematica Software (TOMS)*, 29(1): 49–57, 2003.

[JKB94] Norman Lloyd Johnson, Samuel Kotz, and Narayanaswamy Balakrishnan. Continuous univariate distributions. 1, 1994.

[Kes69] Harry Kesten. A convolution equation and hitting probabilities of single points for processes with stationary independent increments. *Bulletin of the American Mathematical Society*, 75(3): 573–578, 1969.

[KK05] Jan E Kallsen and Christoph Kühn. Convertible bonds: Financial derivatives of game type. In W. Schoutens A. Kyprianou and P. Wilmott, editors, *Exotic Option Pricing and Advanced Lévy Models*. Chichester: Wiley, West Sussex, England, 2005.

[KK07] Christoph Kühn and Andreas E Kyprianou. Callable puts as composite exotic options. *Mathematical Finance*, 17(4): 487–502, 2007.

[KL05] Andreas E Kyprianou and Ronnie Loeffen. Lévy processes in finance distinguished by their coarse and fine path properties. In W. Schoutens A. Kyprianou and P. Wilmott, editors, *Exotic Option Pricing and Advanced Lévy Models*. Chichester: Wiley, West Sussex, England, 2005.

[KMAE08] S Kindermann, P Mayer, H Albrecher, and HW Engl. Identification of the local speed function in a lévy model for option pricing. *Journal of Integral Equations and Applications*, 20(2): 161–200, 2008.

[Leo04] Alexandros Leontitsis. Hybrid particle swarm optimization in matlab, December 2004.

[MCC98] Dilip B Madan, Peter P Carr, and Eric C Chang. The variance gamma process and option pricing. *European Finance Review*, 2(1): 79–105, 1998.

[McC06] B.D. McCullough. A review of testu01. *Journal of Applied Econometrics*, 21(5): 677 – 682, 2006.

[McC08] B.D. McCullough. Microsoft excel's "not the wichmann - hill" random number generators. *Computational Statistics and Data Analysis*, 52: 4587 – 4593, 2008.

[Mer09] Stephan Mertens. Random number generators: A survival guide for large scale simulations. *arXiv preprint arXiv:0905.4238*, 2009.

[Mil73] PW Miller. Exit properties of stochastic processes with stationary independent increments. *Transactions of the American Mathematical Society*, 178: 459–479, 1973.

[MSH76] John R Michael, William R Schucany, and Roy W Haas. Generating random variates using transformations with multiple roots. *The American Statistician*, 30(2): 88–90, 1976.

[MY08] Dilip B Madan and Marc Yor. Representing the cgmy and meixner lévy processes as time changed brownian motions. *Journal of Computational Finance*, 12(1): 27, 2008.

[PFTV92] William H Press, Brian P Flannery, Saul A Teukolsky, and William T Vetterling. *Numerical Recipes in FORTRAN 77: The Art of Scientific Computing*, volume 1. Cambridge university press, 1992.

[Pra99] K Prause. The generalized hyperbolic model: Estimation, financial derivatives, and risk measures, 1999.

[Que02] Richard Quessette. New products, new risks. *Risk*, 15(3): 97–100, 2002.

[Rog84] LCG Rogers. A new identity for real lévy processes. In *Annals of the Institute of Henri Poincaré*, volume 20, pages 21–34. Elsevier, 1984.

[RW02] Claudia Ribeiro and Nick Webber. Valuing path dependent options in the variance-gamma model by monte carlo with a gamma bridge. 2002.

[RW03] Claudia Ribeiro and Nick Webber. A monte carlo method for the normal inverse gaussian option valuation model using an inverse gaussian bridge. *Preprint, City University*, 2003.

[Sat99] Ken-iti Sato. *Lévy processes and infinitely divisible distributions*, volume 68 of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, Cambridge, UK, 1999.

[Sch03] W. Schoutens, editor. *Lévy Processes in Finance*. Wiley Series in Probability and Stattistics. Chichester: Wiley, West Sussex, England, 2003.

[ST01] Wim Schoutens and JL Teugels. Meixner processes in finance. 2001.

[Vig02] Vincent Vigon. Votre lévy rampe-t-il? (does your lévy process creep?). *Journal of the London Mathematical Society*, 65(1): 243–256, 2002.

[VN51] John Von Neumann. Various techniques used in connection with random digits. *Applied Math Series*, 12(36-38), 1951.

[Wil02] Paul Wilmott. Cliquet options and volatility models. *Wilmott Magazine*, December: 78–83, 2002.

[ZZ08] Shibin Zhang and Xinsheng Zhang. Exact simulation of ig-ou processes. *Methodology and computing in applied probability*, 10(3): 337–355, 2008.